INFORMATION AND CONTROL 71, 95–130 (1986)

An Ideal Model for Recursive Polymorphic Types*

DAVID MACQUEEN

AT & T Bell Laboratories, Murray Hill, New Jersey 07974

GORDON PLOTKIN

Department of Computer Science, University of Edinburgh, Edinburgh EH9 3J2, United Kingdom

AND

RAVI SETHI

AT & T Bell Laboratories, Murray Hill, New Jersey 07974

1. INTRODUCTION

When constants are added to the pure lambda calculus, run-time errors can occur if the constants are used improperly, for example, if an attempt is made to apply a natural number as if it were a function or if the first argument of a conditional is not a truth value. We consider "types" as somehow being or generating constraints on expressions. A consistent type discipline ensures that any expression satisfying the constraints will not produce a "run-time error."

1.1. *Expressions and Their Semantics*

In the following syntax for the language in this paper, e ranges over expressions, c over a suitable set of constants, and x over variables:

$$e ::= c \mid x \mid \lambda x.e \mid e(e).$$

A routine denotational semantics for this language appears in Section 5. The semantic domain of values allows constants to represent truth values,

* A condensed version of this paper was presented at the Eleventh Annual ACM Symposium on Principles of Programming Languages, January 1984, Salt Lake City, Utah. The semantic model in this paper supplants that of [17].

0019-9958/86 $3.00

natural numbers, functions, products, and sums (**wrong** formalizes run-time errors):

$$\mathbf{V} \cong \mathbf{T} + \mathbf{N} + (\mathbf{V} \to \mathbf{V}) + (\mathbf{V} \times \mathbf{V}) + (\mathbf{V} + \mathbf{V}) + \{\mathbf{wrong}\}_\perp.$$

The basic theory of domains and the solution of such domain equations is summarized in Section 2.

1.2. *Types*

During any evaluation of the expression $f(x)$, if the value of x satisfies type constraint σ, then it suffices that f satisfies the constraint of being a function that sends all values satisfying σ to values satisfying some constraint τ. These constraints on the values of f and x will be written as

$$x : \sigma$$

$$f : \sigma \to \tau.$$

The inference that the value of $f(x)$ satisfies τ can be written as the rule

$$\frac{f : \sigma \to \tau \quad x : \sigma}{f(x) : \tau}.$$

Inferences like $f(x) : \tau$ will be made using a formal system of axioms and rules in which constraints like σ, τ, and $s \to \tau$ are called *type expressions*, or simply *types*. The following productions in the syntax of types correspond to summands of the value domain (σ ranges over types):

$$\sigma ::= \mathbf{bool} \mid \mathbf{int} \mid \sigma \to \sigma \mid \sigma \times \sigma \mid \sigma + \sigma.$$

Motivation for the following additional productions is given below (t ranges over type variables):

$$\sigma ::= t \mid \forall t.\sigma \mid \exists t.\sigma \mid \mu t.\sigma \mid \sigma \cap \sigma \mid \sigma \cup \sigma.$$

Types bound by $\forall$ generalize the implicit form of polymorphism used in the programming languages ML [13] and Hope [4]. A typical polymorphic function is the identity function $I = \lambda x.x$ that maps truth values to values, natural numbers to natural numbers, and so on, for any type. Its type is represented by $\forall t.t \to t$, of which **bool** → **bool** and **int** → **int** are instances. Existential quantification can be used to represent abstract data types, in the sense of information hiding, as pointed out by Mitchell and Plotkin [20]. (See also Cartwright [5] and Reynolds [25].) Intersection of types has been studied by Coppo *et al.* [8], Pottinger [23], and Sallé [26].

Self-applications like $x(x)$ motivate types bound using μ. Reasoning as for $f(x)$ above,

$$x : s$$

$$x : s \to \tau.$$

If we equate the constraints on x, we get $s = s \to \tau$. The notation $\mu s.s \to \tau$ denotes a solution of this equation. Morris [21, pp. 122–124] observes that such recursive or circular types allow types to be inferred for combinators like Y.

Any pure lambda expression has type $\mu t.t \to t$, that is, $t = t \to t$, since the expression can be used either as a function or as an argument. It follows that constants like 3 are needed to construct expressions like $3(x)$ that do not have types.

The discussion of $x(x)$ extends to the larger expression $\lambda x.xx$.[1] Since the type of x is $s = s \to \tau$, xx has type τ, $\lambda x.xx$ has type $s = s \to \tau$, and the type of $(\lambda x.xx)(\lambda x.xx)$ is τ. No assumptions were made about the type τ, so $(\lambda x.xx)(\lambda x.xx)$ has type τ for any type expression τ. The meaning of $(\lambda x.xx)(\lambda x.xx)$ is $\perp$ (in the model we consider below), so $\perp$ has every type. The sets of values used to model types must therefore always be nonempty. Moreover, nonterminating expressions can have well defined types.

1.3. *Semantics of Types*

The semantics of types in Section 5 is a formalization of the naive view of types as sets of values; a value a has a type σ if a is a member of the set of values modeling σ. Types are modeled by sets of values sharing a common structure, where "structure" represents notions like being a function, or being a pair. The structural distinctions expressed by types are preserved when we go

1. "downward" to approximations and
2. "upward" to least upper bounds of consistent sets of values.

These properties are noted in Milner [19]; they form the basis for the definition of types as "ideals" in Section 3 (see also Shamir and Wadge [28, 31]). A precursor to the semantic model in this paper is described in MacQueen and Sethi [17]. Other models (e.g., McCracken [18])—developed for the explicit form of polymorphism expressed in terms of type parameters following Reynolds [24] and Girard [12]—do not lend themselves to such an intuitive interpretation of types as sets of values.

It is nontrivial to model types as "sets of values" and find a set of values

[1] As usual, function application is indicated by juxtaposition and associates to the left; xx is equivalent to $x(x)$; both $f(x)y$ and fxy are equivalent to $(f(x))(y)$.

s satisfying the equality $s = s \to \tau$. The main technical innovation of this paper is the use of a metric structure on types to establish the existence and uniqueness of solutions of this and similar equations, using the Banach fixed-point theorem. Arnold and Nivat [1] and de Bakker and Zucker [11] have given semantics for nondeterminacy and concurrency using topological completion to create complete metric spaces. It seems fair to say that their concern was to develop alternatives to methods using complete partial orders; here there are, as yet, no alternatives to metric techniques. See Coppo [6], however.

1.4. *Informal Types for Self Application*

The constraint $f: \sigma \to \tau$ requires that f map all values satisfying σ to values satisfying τ. If σ' is a weaker constraint than σ, then, informally, σ' denotes a larger set than σ. Weakening σ to σ' has the opposite effect on the type $\sigma \to \tau$ of functions from σ to τ, because $f: \sigma' \to \tau$ requires f to map the larger set σ' to values satisfying τ; $\sigma' \to \tau$ is thus a stronger constraint on f than $\sigma \to \tau$. This inverse effect is limited to the first argument σ in $\sigma \to \tau$ because the set of functions satisfying $\sigma \to \tau$ increases or decreases as the set of values satisfying the second argument τ increases or decreases, respectively.

Intuition about the role of $\to$ can be provided by considering a particular sequence of sets associated with the equality $s = s \to \tau$. The semantic counterpart of the operator $\to$ on types is the operator $\boxrightarrow$ on sets of values modeling types. Informally, $I \boxrightarrow J$ is the set of all functions that map elements of I to elements of J, where $I, J \subseteq \mathbf{V}$. So if $I \supseteq I'$ and $J \subseteq J'$, then $I \boxrightarrow J \subseteq I' \boxrightarrow J'$.

In keeping with the view of types as sets, let J be the set of values denoted by the type τ. Starting with the set $\mathbf{V}$ of all values, we estimate the set I modeling s by writing the sequence:

$$I_0 = \mathbf{V}$$

$$I_1 = I_0 \boxrightarrow J = \mathbf{V} \boxrightarrow J$$

$$I_2 = I_1 \boxrightarrow J = (\mathbf{V} \boxrightarrow J) \boxrightarrow J$$

$$I_3 = I_2 \boxrightarrow J = ((\mathbf{V} \boxrightarrow J) \boxrightarrow J) \boxrightarrow J.$$

Since I_0 is the entire set $\mathbf{V}$ of values, I_1 consists of functions that map all values in $\mathbf{V}$ to elements of J—a fairly restrictive condition. By definition, I_1 must be a subset of $I_0 = \mathbf{V}$, and $I_1 \subseteq I_0$ implies $I_1 \boxrightarrow J \supseteq I_0 \boxrightarrow J$. Therefore, I_2 is a larger set than I_1. The inclusions we get are (see Fig. 1):

$$I_0 \supseteq I_2 \supseteq I_4 \supseteq \cdots$$

$$I_1 \subseteq I_3 \subseteq I_5 \subseteq \cdots.$$

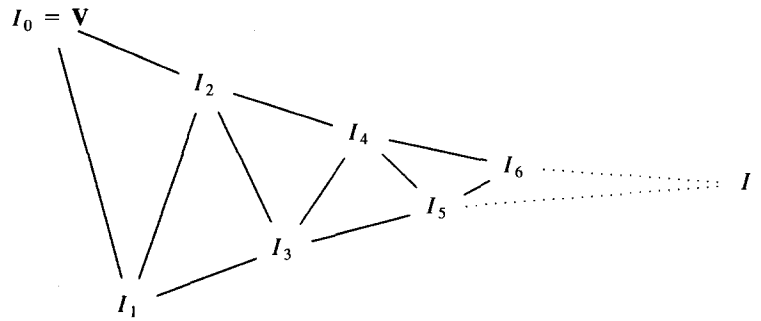

FIG. 1. The sequence $I_0, I_1, \ldots$ converges to I, yielding a solution to the equation $s = s \to \tau$.

Fortunately, it can be shown that the limits, $\bigcap_{n \geq 0} I_{2n}$ and $\bigcup_{n \geq 0} I_{2n+1}$, of the even and odd sequences are the same, and there is a unique set I corresponding to a solution of the equality $s = s \to \tau$. However, the techniques used to show this result are not directly based on the convergence of such nested sequences. Instead, convergence is established using a metric on sets modeling types.

The progression in this paper is as follows:

1. We begin with the semantic domain of values **V** in Section 2.

2. In Section 3, types are modeled by the so-called weak ideals of **V**, used in [17].

3. Recursive type equations are solved by considering the convergence of particular sequences of types. Note that the sequence converging to I in Fig. 1 is neither monotonically increasing nor decreasing. Convergence of sequences cannot therefore be proved using monotonicity or continuity properties. Instead, we set up a "complete" metric space of weak ideals, defined in Section 3 via a rank function on the finite elements of **V**. The solution of domain equations is reviewed in Section 2 because the definition of the rank function is based on the structure of the domain **V**.

4. The Banach fixed-point theorem [2] establishes the existence of unique fixed points for "contractive" functions on complete metric spaces. We show in Section 4 how the various constructions of interest on ideals become contractive functions, so this theorem can be applied to the metric space of types.

The above results enable us in Section 5 to give a semantics to type expressions permitting recursion and universal and existential quantification. This semantics is used to show the soundness of a suitable set of type-inference rules in Section 6. Finally, in the Appendix we consider a variant, the *strong* ideals that also appeared in [17]. Analogous results hold for the strong ideals, except that certain of the type-inference rules are

no longer sound, and it is not clear if there are alternative sound rules of interest.

2. Domains

Expressions will be interpreted using a space of values **V** satisfying the isomorphism:

$$\mathbf{V} \cong \mathbf{T} + \mathbf{N} + (\mathbf{V} \to \mathbf{V}) + (\mathbf{V} \times \mathbf{V}) + (\mathbf{V} + \mathbf{V}) + \{\mathbf{wrong}\}_\perp. \tag{2.1}$$

In words, **V** is (isomorphic to) the sum of the truth values **T**, the natural numbers **N**, continuous functions from **V** to **V**, the product of **V** with itself, the sum of **V** with itself, and a value **wrong** standing for (dynamic) type-errors. This section contains a quick overview of the definitions and results needed to solve equations like (2.1). These mathematical ideas are due to Scott [27]; details may be found in many places, such as the notes by Plotkin [22].

2.1. *Complete Partial Orders*

Solutions of equations like (2.1) can be found in a particular class of partially ordered sets called complete partial orders. A *complete partial order* (cpo for short) is a pair $(D, \sqsubseteq)$ consisting of a set D and a partial order $\sqsubseteq$ on D, such that

(i) there is a least element $\perp$ in $(D, \sqsubseteq)$, and

(ii) each increasing sequence $x_0 \sqsubseteq \cdots \sqsubseteq x_n \sqsubseteq \cdots$ has a least upper bound (lub) $\bigsqcup_{n \geq 0} x_n$.

The set D can be identified by writing $\sqsubseteq_D$ and $\perp_D$ instead of $\sqsubseteq$ and $\perp$. Any set X yields the *flat* cpo $X_\perp = X \cup \{\perp\}$, ordered by putting $x \sqsubseteq y$ iff $x = \perp$ or $x = y$. In particular, the right side of (2.1) contains

$\mathbf{T} = T_\perp = \{true, false\}_\perp$	cpo of truth values
$\mathbf{N} = N_\perp$	the cpo of natural numbers
$\mathbf{W} = \{\mathbf{wrong}\}_\perp$	the type error cpo.

The functions of interest between cpos are the *continuous* ones, preserving lubs of increasing sequences; that is,

$$f\left(\bigsqcup_{n \geq 0} x_n\right) = \bigsqcup_{n \geq 0} f(x_n).$$

Functions preserving $\perp$ are termed *strict*. Every continuous function $f: D \to D$ has a *least* fixed point, namely $\bigsqcup_{n \geq 0} f^n(\perp_D)$.

Any function $f: X \to Y$ on sets can be considered as a strict and continuous function $f: X_\perp \to Y_\perp$. We shall make particular use of the functions

$$\begin{aligned} +1: N &\to N && \text{add one} \\ -1: N &\to N && \text{subtract one (with } 0-1=0) \\ Z: N &\to T && \text{test for zero.} \end{aligned}$$

The identity function $\mathrm{id}_D: D \to D$ is continuous, and continuous functions are closed under composition, as are the strict continuous ones; thus we have a category of cpos and continuous functions and a subcategory of the strict continuous functions.

2.2. *Constructions on CPOs*

The right side of the isomorphism (2.1) uses three constructions on cpos:

1. the function space $(D \to E)$ of all continuous functions from a cpo D to another E with the pointwise ordering,

2. the Cartesian product $(D \times E)$ of two cpos D and E with the coordinate-wise ordering, and

3. the coalesced sum $(D + E)$ consisting of the disjoint union of two cpos D and E, with their least elements identified, and with the evident inherited orderings.

Details of these constructions are given in Table 1. Elements of cpos can be denoted using the typed λ-calculus [3, Appendix A].

A few useful continuous functions are also shown in Table 1. The details of the functions *isr*, *inr*, and *outr* associated with the coalesced-sum construction are similar to those of *isl*, *inl*, and *outl*, respectively. Analogous functions exist for the n-ary disjoint sum $(0 \leqslant i < n)$:

$$\begin{aligned} \mathrm{is}_i &: D_0 + \cdots + D_{n-1} \to \mathbf{T} \\ \mathrm{in}_i &: D_i \to (D_0 + \cdots + D_{n-1}) \\ \mathrm{out}_i &: (D_0 + \cdots + D_{n-1}) \to D_i. \end{aligned}$$

It will be important later that the injection functions in_i are strict and indeed both preserve and reflect any existing lubs.[2]

[2] A function f *preserves* lubs if whenever the lub $x \sqcup y$ exists, the lub $f(x) \sqcup f(y)$ also exists and $f(x) \sqcup f(y) = f(x \sqcup y)$. A function f *reflects* lubs if whenever the lub $f(x) \sqcup f(y)$ exists, the lub $x \sqcup y$ also exists and $f(x) \sqcup f(y) = f(x \sqcup y)$. The definitions of preserve and reflect apply equally to other notions such as glbs.

TABLE 1

Constructions on Complete Partial Orders

Construction	*Function Space*, $D \to E$
Ordering	$f \sqsubseteq_{D \to E} g$ iff $\forall x \in D, f(x) \sqsubseteq_E g(x)$
Functions	$Y: (D \to D) \to D$ least fixed point operator, where $Yf = \bigsqcup_{n \geq 0} f^n(\perp)$
Notation	If e denotes an element of E continuous in the free variable x (if it appears at all) where x is a variable ranging over D, then the *abstraction* $(\lambda x \in D.e)$ denotes the function $f: D \to E$ where $f(x) = e$. The function $(\lambda x \in D.e)$ is continuous in any variable that e is. If e_1 and e_2 are expressions denoting elements of $D \to E$ and D, respectively (given values for their free variables), then the *application* $e_1(e_2)$ denotes the element of E obtained by applying the denotation of e_1 to that of e_2. The element $e_1(e_2)$ is continuous in any variable that e_1 and e_2 are.
Construction	*Cartesian Product*, $D \times E$
Ordering	$\langle u, v \rangle \sqsubseteq_{D \times E} \langle x, y \rangle$ iff $u \sqsubseteq_D x$ and $v \sqsubseteq_E y$
Functions	pair: $D \to (E \to (D \times E))$, where pair $(d)(e) = \langle d, e \rangle$ $\pi_1: D \times E \to D$ a projection function $\pi_2: D \times E \to E$ a projection function
Notation	It will be convenient to write pair $(e_1)(e_2)$ as $\langle e_1, e_2 \rangle$, for functions $f: D \times E \to F$ to write $f(e_1, e_2)$ instead of $f(\langle e_1, e_2 \rangle)$, and similarly for the general n-ary case.
Construction	*Coalesced Sum*, $D + E = \{\langle 0, d \rangle \mid d \in D, \quad d \neq \perp\} \cup \{\langle 1, e \rangle \mid e \in E, e \neq \perp\} \cup \{\perp\}$
Ordering	$x \sqsubseteq_{D+E} y$ iff $x = \begin{cases} \perp, \text{ or} \\ \langle 0, x' \rangle,\ y = \langle 0, y' \rangle, \text{ and } x' \sqsubseteq_D y', \text{ or} \\ \langle 1, x' \rangle,\ y = \langle 1, y' \rangle, \text{ and } x' \sqsubseteq_E y' \end{cases}$
Functions	isl, isr: $(D + E) \to \mathbf{T}$, where $\text{isl}(x) = \begin{cases} \text{true} & (x = \langle 0, d \rangle \text{ for some } d \text{ in} D) \\ \text{false} & (x = \langle 1, e \rangle \text{ for some } e \text{ in } E) \\ \perp & (\text{otherwise}) \end{cases}$ inl: $D \to (D + E)$ and inr: $E \to (D + E)$, where $\text{inl}(d) = \begin{cases} \langle 0, d \rangle & (d \neq \perp) \\ \perp & (d = \perp) \end{cases}$ outl: $(D + E) \to D$ and outr: $(D + E) \to E$, where $\text{outl}(x) = \begin{cases} d & (x = \langle 0, d \rangle \text{ for some } d \text{ in } D) \\ \perp & (\text{otherwise}) \end{cases}$

Finally, the continuous *conditional* function cond: $\mathbf{T} \to (D \to (D \to D))$ is defined by

$$\text{cond } txy = \begin{cases} x & (t = \text{true}) \\ y & (t = \text{false}) \\ \bot & (t = \bot) \end{cases}$$

We write **if** e_1 **then** e_2 **else** e_3 instead of cond $e_1 e_2 e_3$.

Both the function space and Cartesian product constructions can be taken as functors on the category of continuous functions by the following definitions of the exponentiation and product of functions (note that f is a member of different cpos in the two cases):

for $f: D' \to D$ and $g: E \to E'$ we have $(f \to g): (D \to E) \to (D' \to E')$, where

$$(f \to g) = \lambda h \in (D \to E).\, g \circ h \circ f$$

for $f: D \to D'$ and $g: E \to E'$ we have $(f \times g): (D \times E) \to (D' \times E')$, where

$$(f \times g) = \lambda z \in D \times E.\, \langle f(\pi_1(z)),\, g(\pi_2(z)) \rangle.$$

Exponentiation, product, and sum can be taken to be functors on the subcategory of strict functions, with the same definition for the exponential and product. Sums of functions are defined by

for $f: D \to D'$ and $g: E \to E'$ we have $(f + g): (D + E) \to (D' + E')$, where

$$(f + g) = \lambda z \in D + E.\ \textbf{if } \text{isl}(z) \textbf{ then } \text{inl}(f(\text{outl}(z))) \textbf{ else } \text{inr}(g(\text{outr}(z))).$$

In fact, the first two functors show why the category of continuous functions is Cartesian closed and the last one is the categorical sum in the subcategory of strict functions.

2.3. *Construction of CPOs*

The value space **V** can be constructed using embeddings; a continuous map $\phi: D \to E$ is an *embedding* if there exists $\psi: E \to D$ such that $\psi \circ \phi = \text{id}_D$ and $\phi \circ \psi \sqsubseteq \text{id}_E$. The map ψ is called a *projection* and being determined uniquely by ϕ is written ϕ^R. *Projection pairs* $\langle \phi, \psi \rangle$ are a special case of adjoint pairs of maps between partial orders [22], which can be characterized by:

1. for all x in D and y in E, $x \sqsubseteq \psi y$ if and only if $\phi x \sqsubseteq y$, and
2. ϕ preserves all existing lubs and ψ preserves all existing glbs.

As an example, the functions inl, inr, in_i are embeddings with the corresponding projections being outl, outr, and out_i. The identity is an embedding and a projection; embeddings and projections are preserved by composition. For two embeddings

$$D \xrightarrow{\phi_0} E \xrightarrow{\phi_1} F \qquad \text{we have} \quad (\phi_1 \circ \phi_0)^R = \phi_0^R \circ \phi_1^R.$$

Finally, if $\phi_0 : D_0 \to E_0$ and $\phi_1 : D_1 \to E_1$ are embeddings, we get

Embedding	Corresponding Projection
$(\phi_0^R \to \phi_1)$	$(\phi_0 \to \phi_1^R)$
$(\phi_0 \times \phi_1)$	$(\phi_0^R \times \phi_1^R)$
$(\phi_0 + \phi_1)$	$(\phi_0^R + \phi_1^R)$

The cpo $\mathbf{V}$ is constructed via a limiting process. First define cpos $\mathbf{V}_n$ starting from $\mathbf{V}_0 = \varnothing_\perp$,

$$\mathbf{V}_{n+1} = \mathbf{T} + \mathbf{N} + (\mathbf{V}_n \to \mathbf{V}_n) + (\mathbf{V}_n \times \mathbf{V}_n) + (\mathbf{V}_n + \mathbf{V}_n) + \mathbf{W}.$$

Next connect them up by embeddings $\phi_n : \mathbf{V}_n \to \mathbf{V}_{n+1}$ starting from $\phi_0(x) = \perp$ and then putting:

$$\phi_{n+1} = \text{id}_\mathbf{T} + \text{id}_\mathbf{N} + (\phi_n^R \to \phi_n) + (\phi_n \times \phi_n) + (\phi_n + \phi_n) + \text{id}_\mathbf{W}.$$

Then $\mathbf{V}$ is the colimit of the chain $\langle \mathbf{V}_n, \phi_n \rangle$ in the category of embeddings. That is, there is a cone $\mu : \langle \mathbf{V}_n, \phi_n \rangle \to \mathbf{V}$ such that $\langle \mu_n \circ \mu_n^R \rangle_{n \geqslant 0}$ is an increasing sequence with lub $\text{id}_\mathbf{V}$. (To say that $\mu = \langle \mu_n \rangle_{n \geqslant 0}$ is a cone means that for all n, $\mu_n : \mathbf{V}_n \to \mathbf{V}$ and that $\mu_n = \mu_{n+1} \circ \phi_n$.)

The desired isomorphism θ is constructed via a cone

$$\nu_n : \langle \mathbf{V}_{n+1}, \phi_{n+1} \rangle \to (\mathbf{T} + \mathbf{N} + (\mathbf{V} \to \mathbf{V}) + (\mathbf{V} \times \mathbf{V}) + (\mathbf{V} + \mathbf{V}) + \mathbf{W})$$

of embeddings where

$$\nu_n = \text{id}_\mathbf{T} + \text{id}_\mathbf{N} + (\mu_n^R \to \mu_n) + (\mu_n \times \mu_n) + (\mu_n + \mu_n) + \text{id}_\mathbf{W}$$

by putting $\theta = \bigsqcup \nu_n \circ \mu_{n+1}^R$ and its inverse is $\bigsqcup \mu_{n+1} \circ \nu_n^R$. Note that the cones μ and ν are related by the equations

$$\nu_n = \theta \circ \mu_{n+1}$$
$$\mu_n = \theta^{-1} \circ \nu_n.$$

It will be very convenient, notationally, in what follows to regard both θ and θ^{-1} as actual identities and to view the evident injections of $\mathbf{T}$, $\mathbf{N}$,

$(\mathbf{V} \to \mathbf{V})$, $(\mathbf{V} \times \mathbf{V})$, $(\mathbf{V} + \mathbf{V})$, and $\mathbf{W}$ as actual inclusions. Below implicit use will be made of the fact already mentioned that these injections preserve and reflect the order and all existing lubs. See Smyth and Plotkin [29] for further details on the construction of $\mathbf{V}$.

2.4. *Domains*

It will be necessary to know much more of the structure of $\mathbf{V}$ than just that it is a cpo. First it is *consistently complete* meaning that any consistent subset of $\mathbf{V}$ has a least upper bound; here $X \subseteq \mathbf{V}$ is consistent if it has an upper bound in $\mathbf{V}$, that is, there is a $y \in \mathbf{V}$ such that $x \sqsubseteq y$ for all x in X. All flat cpos are consistently complete and consistent completeness is preserved by all the constructions considered above and by the kinds of limits used to construct $\mathbf{V}$.

Next, $\mathbf{V}$ has a very pleasant kind of basis of finite elements. An element of a cpo is *ω-finite* if and only if whenever it is less than the lub of an increasing sequence it is less than some element of the sequence. A cpo is *ω-algebraic* if and only if it has countably many ω-finite elements and given any element x, the set of ω-finite elements less than x is directed and has x as its least upper bound (whenever a and b are ω-finite and $a \sqcup b$ exists, it is ω-finite). The ω-finite elements in any subset X of a cpo are denoted by X°. All ω-algebraic cpos have lubs of arbitrary directed sets (sometimes cpos are taken to the partial orders with such lubs and $\perp$); the ω-finite elements are even *finite*, meaning that when one is below the lub of a directed set it is below some element of that set. Finally, a cpo is a *domain* if and only if it is consistently complete and ω-algebraic.

All countable flat cpos are domains (all their elements being finite) and all the constructions send domains to domains. For the product and sum of domains D and E we have the simple formulae: $(D \times E)^\circ = D^\circ \times E^\circ$ and $(D + E)^\circ = \mathrm{inl}(D^\circ) \cup \mathrm{inr}(E^\circ)$; for the function space $(D \to E)$ we first define the continuous *step function* $(a \Rightarrow b)$ for any a, b in D°, E°, respectively, by

$$(a \Rightarrow b)(x) = \begin{cases} b & (x \sqsupseteq a) \\ \perp & (\text{otherwise}) \end{cases}$$

and then the finite functions are the finite lubs of step functions. (The lub of $a_1 \Rightarrow b_1, \ldots, a_n \Rightarrow b_n$ exists if and only if whenever $\{a_{i_1}, \ldots, a_{i_k}\}$ is a subset of the a_i's with an upper bound, then $\{b_{i_1}, \ldots, b_{i_k}\}$ has an upper bound too.) Finally, the kind of limit construction used for $\mathbf{V}$ produces domains from chains of domains and the relevant formula here is $\mathbf{V}^\circ = \bigcup \mu_n(\mathbf{V}_n^\circ)$ which is easily obtained from the fact that embeddings preserve the ω-finite elements and the formula $\mathrm{id}_{\mathbf{V}} = \bigsqcup_{n \geqslant 0} \mu_n \circ \mu_n^R$ given above.

3. Metric Space of Ideals

Type expressions will be interpreted using certain subsets of **V**, called ideals, as mentioned in Section 1.3. The definition of ideals makes precise the notion of a type as a collection of structurally similar values.

3.1. *Ideals*

For technical reasons the definition is given in two stages: a subset I of some partial order P is an *order ideal* if and only if

1. $I \neq \varnothing$
2. $\forall y \in I. \forall x \in D. x \sqsubseteq y$ implies $x \in I$.

We write $\mathscr{I}_0(P)$ for the order ideals of a partial order P.

A subset I of a domain D is an *ideal* if and only if it is an order ideal satisfying the additional constraint:

3. $\forall$ increasing sequences $\langle x_n \rangle . (\forall n. x_n \in I)$ implies $\bigsqcup x_n \in I$.

That is, ideals are the nonempty left-closed sets closed under lubs of increasing sequences. Nonemptiness is needed because $\perp$ has every type (see Sect. 1.2 and 4).

If D is a domain then an ideal I is *strong* if and only if

4. $\forall x, y \in I. x \sqcup y$ exists implies $x \sqcup y \in I$.

The collection of all ideals is denoted by $\mathscr{I}(D)$ and of all strong ideals by $\mathscr{I}^+(D)$. In this section we develop only the mathematics of ideals; the strong ideals require more insight into the structure of **V** and are considered in the Appendix.

Ideals are determined by their finite elements. Regarding D° as a partial order (inherited from D) and ordering the ideals by subset we find

Proposition 1. *The correspondence* $I \mapsto I^\circ$ *is an isomorphism of* $\langle \mathscr{I}(D), \subseteq \rangle$ *and* $\langle \mathscr{I}_0(D^\circ), \subseteq \rangle$ *with inverse* $J \mapsto \{\bigsqcup a_n \mid \langle a_n \rangle$ *an increasing sequence in* $J\}$.

Ideals in D might better be called closed ideals and in fact they are exactly the nonempty closed sets in the Scott topology of D [27]; they form a complete lattice with the glbs being the set-theoretic ones and the lubs being given by the formula:

$$\left(\bigsqcup_\lambda I_\lambda\right)^\circ = \bigcup_\lambda I_\lambda^\circ$$

(finite lubs are given by the set-theoretic union).

3.2. *Distance between Ideals*

As discussed above, the idea now is to solve recursive type equations by structuring the ideals as a complete metric space. The distance between two ideals will be measured via a notion of the smallest rank of a finite element in one but not the other.

DEFINITION. A *rank* function in D is any function $r: D^\circ \to N$. If I and J are ideals then a *witness* for I and J is any element of $I^\circ \oplus J^\circ$ (the idea being that a difference between I and J is witnessed).[3] The *closeness* $c(I, J)$ of I and J is the least possible rank of a witness for I and J, and if none exists it is ∞.

Let us consider some fixed rank function r in what follows. The following proposition gives the elementary properties of the closeness function.

PROPOSITION 2. (i) $c(I, J) = \infty$ *iff* $I = J$.

(ii) $c(I, J) = c(J, I)$.

(iii) $c(I, K) \geqslant \min(c(I, J), c(J, K))$.

Proof. For part (i), $c(I, J) = \infty$ iff $I^\circ \oplus J^\circ = \varnothing$ iff $I^\circ = J^\circ$ iff $I = J$ (by the previous proposition). Part (ii) is clear from the definition of the closeness function.

Part (iii) is immediate if $c(I, K) = \infty$. Otherwise, let b be a witness of minimum rank for I and K. Now b must be a witness either for I and J or for J and K, since otherwise $b \in I$ iff $b \in K$. In the first case we see that $c(I, K) = r(b) \geqslant c(I, J)$ while in the second case $c(I, K) \geqslant c(J, K)$. ∎

Given such a closeness function, one can, as is well known [16, 30], define a metric.[4] Here we take $d(I, J) = 2^{-c(I,J)}$ where, by convention, $2^{-\infty} = 0$. This is even an *ultrametric* [16, Ex. 6, p. 70] meaning that

$$d(I, K) \leqslant \max(d(I, J), d(J, K))$$

holds (by part (iii) of the previous proposition), which is stronger than the triangle inequality.

Now $\langle I_i \rangle_{i \geqslant 0}$ is called a *Cauchy sequence* if given any $\varepsilon > 0$ there exists an n such that for all $i, j \geqslant n$, $d(I_i, I_j) < \varepsilon$. A metric space is *complete* if every Cauchy sequence converges.

[3] Given ideals I and J, $I \oplus J$ is their symmetric difference $(I - J) \cup (J - I)$.

[4] A *metric* d is a mapping from a set to the nonnegative reals satisfying three properties: (1) $d(I, J) = 0$ iff $I = J$; (2) $d(I, J) = d(J, I)$; and (3) the triangle inequality $d(I, K) \leqslant d(I, J) + d(J, K)$.

THEOREM 3. *The metric space* $\langle \mathscr{I}(D), d\rangle$ *is complete. Indeed if* $\langle I_i\rangle_{i\geqslant 0}$ *is a Cauchy sequence then its limit is* I *where* $I^\circ = \{b \in D^\circ \mid b \text{ is in almost all } I_i\}$.

Proof. Clearly I° as defined is an ideal in D° and so I is determined. In terms of closeness, we must show

(1) $\forall m \geqslant 0.\exists n \geqslant 0.\forall i \geqslant n.c(I, I_i) > m$, and what we know is

(2) $\forall m \geqslant 0.\exists n \geqslant 0.\forall i, j \geqslant n.c(I_i, I_j) > m$.

To demonstrate (1) let $m \geqslant 0$ be given and choose n as guaranteed by (2). Take any $i \geqslant n$. Let b be a witness of minimum rank for I and I_i (if there is none, it is trivially true that $c(I, I_i) > m$). If $b \in I$ then b is in almost every I_j and hence is in I_j for some $j \geqslant n$; if $b \notin I$ then b is not in I_j for infinitely many j and hence not in I_j for some $j \geqslant n$. In either case b is a witness for I_i and I_j for some $j \geqslant n$ and we can calculate that

$$c(I, I_i) = r(b) \geqslant c(I_i, I_j) > m$$

by our choice of n, thereby concluding the proof. ∎

At first sight this theorem is a little surprising given the arbitrary nature of the rank function. But note, for example, that if the rank function is constant then the only Cauchy sequences are those that are eventually constant. The topology determined by this metric turns out to be compact iff every $r^{-1}(n)$ is finite.

3.3. *Rank of an Element*

In order to apply this result to $\mathbf{V}$ we need to construct a rank function and consider its properties. Here we just take the rank of a finite element to be the first place it appears in the chain; that is, $r(c)$ is the least $n \geqslant 0$ with $c \in \mu_n(\mathbf{V}_n^\circ)$ (which is well defined by the above remarks on $\mathbf{V}^\circ$). The following properties of this rank function will be used in Section 4 to show that the standard type constructions are "contractive."

PROPOSITION 4. (i) $r(c) = 0$ *iff* $c = \perp_{\mathbf{V}}$.

(ii) *Suppose* a *and* b *are finite elements whose lub exists. Then* $r(a \sqcup b) \leqslant \max(r(a), r(b))$.

(iii) *Any element* c, *other then* $\perp$, *of* $\mathbf{N}$, $\mathbf{T}$, *or* $\mathbf{W}$ *has rank* 1.

(iv) *Any finite element* c *of* $(\mathbf{V} \times \mathbf{V})$, *other than* $\perp$, *is equal to* $\langle a, b\rangle$ *with* a *and* b *finite,* $r(a) < r(c)$ *and* $r(b) < r(c)$.

(v) *Any finite element* c *of* $(\mathbf{V} + \mathbf{V})$, *other than* $\perp$, *is equal either to* $\mathrm{inl}(a)$ *with* a *finite and* $r(a) < r(c)$ *or to* $\mathrm{inr}(b)$ *with* b *finite and* $r(b) < r(c)$.

(vi) *Any finite element* c *of* $(\mathbf{V} \to \mathbf{V})$, *other than* $\perp$, *is equal to* $(a_1 \Rightarrow b_1) \sqcup \cdots \sqcup (a_n \Rightarrow b_n)$ *with the* a_i *and* b_i *finite and* $r(a_i) < r(c)$ *and* $r(b_i) < r(c)$ *for all* i.

Proof. (i) Immediate from the definition of r.

(ii) Suppose $r(a)=n$ and $r(b)=m$ with, say, $m \geqslant n$. Then $a=\mu_n(a')$ and $b=\mu_m(b')$ for some $a' \in \mathbf{V}_n$ and $b' \in \mathbf{V}_m$. So $a=\mu_m(\phi_{nm}a')$, where ϕ_{nm} is the composition $\phi_{m-1} \circ \cdots \circ \phi_n : \mathbf{V}_n \to \mathbf{V}_m$. Therefore

$$\mu_m((\phi_{nm}a') \sqcup b') = \mu_m(\phi_{nm}a') \sqcup \mu_m(b') = a \sqcup b$$

since embeddings reflect lubs. Therefore, $r(a \sqcup b) \leqslant m$.

(iii) Suppose, for example, that c is true $\in \mathbf{T}$, or more precisely that $\theta(c)=\mathrm{in}_0(\mathrm{true})$, where θ is the isomorphism defined in Section 2.3. Then

$$c=\theta^{-1}(\mathrm{in}_0(\mathrm{true}))=\theta^{-1}(v_0(\mathrm{in}_0(\mathrm{true})))=\mu_1(\mathrm{in}_0(\mathrm{true}))$$

since $\theta \circ \mu_{n+1}=v_n$. Thus $r(c) \leqslant 1$ and equality follows from part (i).

(iv) To say that c is in $\mathbf{V} \times \mathbf{V}$ means that $\theta(c)=\langle a, b \rangle \in \mathbf{V} \times \mathbf{V}$ (ignoring the injection into the sum). Since c is finite, so are a and b. As $c \neq \perp$ we can assume $r(c)=n+1$ for some $n \geqslant 0$, and hence $c=\mu_{n+1}(d)$ for some $d \in \mathbf{V}_{n+1}^{\circ}$. Then

$$\langle a, b \rangle = \theta(c) = \theta(\mu_{n+1}(d)) = v_n(d)$$

and so $d=\langle a', b' \rangle$ for some $a', b' \in \mathbf{V}_n^{\circ}$, with $a=\mu_n(a')$ and $b=\mu_n(b')$. Therefore $r(a), r(b) \leqslant n$.

(v) Similar to (iv).

(vi) Assume c is in $\mathbf{V} \to \mathbf{V}$, meaning $\theta(c) \in \mathbf{V} \to \mathbf{V}$ (once again ignoring injections where convenient). Since $c \neq \perp$, we have $r(c)=n+1$ and $c=\mu_{n+1}(f)$ for some $n \geqslant 0$ and $f \in \mathbf{V}_{n+1}^{\circ}$. Then we have $\theta(c)=\theta(\mu_{n+1}(f))=v_n(f) \in \mathbf{V} \to \mathbf{V}$ and hence $f \in \mathbf{V}_n \to \mathbf{V}_n$. Thus by the remarks above about the representation of finite functions, $f=(a_1' \Rightarrow b_1') \sqcup \cdots \sqcup (a_m' \Rightarrow b_m')$. Now for any $x \in \mathbf{V}$

$$\begin{aligned}
\theta(c)(x) &= v_n(f)(x) = (\mu_n^R \to \mu_n)(f)(x) \\
&= \mu_n(f(\mu_n^R(x))) \\
&= \mu_n\left(\bigsqcup \{b_i' \mid \mu_n^R(x) \sqsupseteq a_i'\}\right) \\
&= \mu_n\left(\bigsqcup \{b_i' \mid x \sqsupseteq \mu_n(a_i')\}\right) \\
&= \bigsqcup \{\mu_n(b_i') \mid x \sqsupseteq \mu_n(a_i')\} \\
&= ((\mu_n(a_1') \Rightarrow \mu_n(b_1')) \sqcup \cdots \sqcup (\mu_n(a_m') \Rightarrow \mu_n(b_m')))(x).
\end{aligned}$$

Therefore $\theta(c) = (a_1 \Rightarrow b_1) \sqcup \cdots \sqcup (a_m \Rightarrow b_m)$, where $a_i = \mu_n(a_i')$ and $b_i = \mu_n(b_i')$ and so $r(a_i), r(b_i) \leqslant n < r(c)$. ∎

3.4. *Unique Fixed Points*

In order to find ideals satisfying such equations as $t = t \to \text{int}$ we use the Banach fixed-point theorem [2, 30, p. 130], which guarantees the existence of unique fixed points for a certain kind of functions on complete metric spaces. A (*uniformly*) *contractive* map $f: X \to Y$ on metric spaces is one such that there is a real number $0 \leqslant r < 1$ such that for all $x, x' \in X$, we have

$$d_Y(f(x), f(x')) \leqslant r d_X(x, x'),$$

and it is *nonexpansive* if this holds but with $r \leqslant 1$ (we will usually omit the identifying subscripts on the metrics when these are clear from the context). The generalization to n-variable functions requires

$$d(f(x_1, \ldots, x_n), f(x_1', \ldots, x_n')) \leqslant r \max\{d(x_i, x_i') \mid 1 \leqslant i \leqslant n\}.$$

The Banach fixed-point theorem states that if X is a nonempty complete metric space and $f: X \to X$ is contractive then it has a unique fixed point, namely $\lim_{n \geqslant 0} f^n(x_0)$, where x_0 is any point in X. The next proposition characterizes these concepts in the case of $\mathscr{I}(D)$ in terms of the closeness function.

PROPOSITION 5. *A function $f: \mathscr{I}(D)^n \to \mathscr{I}(D)$ is nonexpansive iff for all ideals $I_1, \ldots, I_n$ and $J_1, \ldots, J_n$ we have*

$$c(f(I_1, \ldots, I_n), f(J_1, \ldots, J_n)) \geqslant \min_i c(I_i, J_i);$$

and it is contractive iff for all ideals $I_1, \ldots, I_n$ and $J_1, \ldots, J_n$, with some $I_i \neq J_i$ we have

$$c(f(I_1, \ldots, I_n), f(J_1, \ldots, J_n)) > \min_i c(I_i, J_i).$$

Proof. We confine ourselves to proving the second and harder of these two assertions. Take I_i and J_i as required, supposing f to be contractive. Then we calculate that for some $0 \leqslant r < 1$,

$$\begin{aligned} c(f(I_1, \ldots, I_n), f(J_1, \ldots, J_n)) &= -\log_2 d(f(I_1, \ldots, I_n), f(J_1, \ldots, J_n)) \\ &\geqslant -\log_2 (r \max_i d(I_i, J_i)) \\ &= (-\log_2 r) + \min_i c(I_i, J_i) \\ &> \min_i c(I_i, J_i) \end{aligned}$$

(where by convention, $-\log_2 0 = \infty$).

Conversely suppose that when $I_i \neq J_i$ for some i we have

$$c(f(I_1,..., I_n), f(J_1,..., J_n)) > \min_i c(I_i, J_i).$$

Then, as both sides are positive integers:

$$c(f(I_1,..., I_n), f(J_1,..., J_n)) \geqslant (\min_i c(I_i, J_i)) + 1.$$

Therefore,

$$\begin{aligned} d(f(I_1,..., I_n), f(J_1,..., J_n)) &= 2^{-c(f(I_1,...,I_n),f(J_1,...,J_n))} \\ &\leqslant 2^{-(\min_i c(I_i,J_i)-1} \\ &= 1/2 \max_i 2^{-c(I_i,J_i)} \\ &= 1/2 \max_i d(I_i, J_i) \end{aligned}$$

showing f to be contractive as required. ∎

A closer look at the above proof will convince the reader that in the present case we get the same class of contractive functions if in the definition we allow r to depend on the arguments of f, this is not true for arbitrary metric spaces.

4. Contractive Maps on Ideals

In order to apply the Banach fixed-point theorem to determine ideals modeling recursive types, we need to consider the contractiveness of maps on ideals. Some care is needed, since union and intersection have the weaker property of being nonexpansive.

4.1. *Auxiliary Maps*

The projection functions $\pi_i: X_1 \times \cdots \times X_n \to X_i$ are not contractive but are nonexpansive. The composition of a map $f: X_1 \times \cdots \times X_n \to Y$ with $g_i: V_1 \times \cdots \times V_m \to X_i$ is nonexpansive if f and all the g_i are; if f is contractive and all the g_i are nonexpansive then the composition is contractive and this holds also if all the g_i are contractive and f is nonexpansive. Finally, we note that when Y is an ultrametric space then a map $f: X_1 \times \cdots \times X_n \to Y$ as above is contractive (nonexpansive) iff it is contractive (nonexpansive) in each argument taken separately.

Proposition 6. *Intersection and union are not contractive but are nonexpansive, considered as binary functions over ideals.*

Proof. Since $d(I \cap I,\ J \cap J) = d(I \cup I,\ J \cup J) = d(I, J) \geqslant \max(d(I, J), d(I, J))$, neither union nor intersection are contractive. To show that intersection is nonexpansive it is sufficient, by Proposition 5, to establish that for any ideals I, J, I', J',

$$c(I \cap J, I' \cap J') \geqslant \min(c(I, I'), c(J, J')).$$

If $I \cap J = I' \cap J'$ this is immediate, so assume they are not equal and that b is a witness of minimum rank for $I \cap J$ and $I' \cap J'$, say $b \in (I \cap J) - (I' \cap J')$. If $b \notin I'$ then it is a witness for I and I' so $r(b) \geqslant c(I, I')$; otherwise $b \notin J'$, implying that it is a witness for J and J' and $r(b) \geqslant c(J, J')$. Hence in either case we find that

$$c(I \cap J, I' \cap J') = r(b) \geqslant \min(c(I, I'), c(J, J')).$$

The equally easy proof that union is nonexpansive is left to the reader. ∎

4.2. *Type Constructors*

We define three binary functions on ideals corresponding to the three basic type constructions. The *sum* $I \boxplus J$, *product* $I \boxtimes J$, and *exponentiation* (or *function ideal*) $I \boxrightarrow J$ of two ideals I and J are defined by

$$I \boxplus J = \mathrm{inl}(I) \cup \mathrm{inr}(J)$$

$$I \boxtimes J = I \times J$$

$$I \boxrightarrow J = \{f \in \mathbf{V} \to \mathbf{V} \mid f(I) \subseteq J\}$$

where the right-hand sides of these definitions are subsets of $\mathbf{V} + \mathbf{V}$, $\mathbf{V} \times \mathbf{V}$, and $\mathbf{V} \to \mathbf{V}$, respectively, that are viewed as subsets of $\mathbf{V}$ via the isomorphism θ. It is straightforward to show that, when viewed as a subset of $\mathbf{V}$, each of these sets is an ideal. The idea behind the definition of the function ideal appears in many papers (see [19, 17], for example). Hindley [15] calls it the *simple semantics* and attributes it to Reynolds [24] and Scott.

The next theorem is central to the results of this paper.

THEOREM 7. *All three functions, sum, product, and exponentiation, are contractive.*

Proof. The basic idea is that two distinct compound ideals (e.g., $I \boxtimes J$ and $I' \boxtimes J'$) have a compound witness of least rank (e.g., $\langle i, j \rangle \in I \boxtimes J - I' \boxtimes J'$) whose components are, by Proposition 4, simpler (i.e., lower rank) witnesses to the differences between the component ideals (e.g., $i \in I - I'$ or $j \in J - J'$). This implies that the compound elements are closer together than their components.

Sum. Assume $I \boxplus J \neq I' \boxplus J'$ and let c be a witness of minimum rank for $I \boxplus J$ and $I' \boxplus J'$, say $c \in (I \boxplus J) - (I' \boxplus J')$. Now $c \neq \bot$ so by Proposition 4 we either have $c = \mathrm{inl}(a)$ for some finite $a \in I$ with $r(c) > r(a)$ or $c = \mathrm{inr}(b)$ for some finite $b \in J$ with $r(c) > r(b)$. In the first case, since $c \notin I' \boxplus J'$ we have $a \notin I'$ and so a is a witness for I and I' and we have $c(I \boxplus J,\ I' \boxplus J') = r(c) > r(a) \geqslant c(I, I') \geqslant \min(c(I, I'),\ c(J, J'))$, as required by Proposition 5. The second case is similar.

Product. Let c be a witness of minimum rank for $I \boxtimes J$ and $I' \boxtimes J'$, with, say, $c \in I \boxtimes J$. Since $c \neq \bot$, Proposition 4 states that $c = \langle a, b \rangle$, where a and b are finite elements of I and J respectively and $r(c) > \max(r(a), r(b))$. As $c \notin I' \boxtimes J'$ we must have either $a \notin I'$ or $b \notin J'$. In the first case a is a witness for I and I' of rank less than $r(c)$ and in the second case b is a witness for J and J' of rank less than $r(c)$, and the proof concludes as before.

Exponentiation. Let c be a witness of minimum rank for $I \boxrightarrow J$ and $I' \boxrightarrow J'$, being, say, only in the former ideal. Then $c \neq \bot$ and so by Proposition 4, $c = (a_1 \Rightarrow b_1) \sqcup \cdots \sqcup (a_n \Rightarrow b_n)$, where $n > 0$ and a_i, b_i are finite elements of $\mathbf{V}$ with $r(c) > \max(r(a_i), r(b_i))$. Since $c \notin I' \boxrightarrow J'$ there must be an $x \in I'$ such that $c(x) \notin J'$. Let $a = \bigsqcup \{a_i \mid a_i \sqsubseteq x\}$ and $b = \bigsqcup \{b_i \mid a_i \sqsubseteq x\} = c(x)$. Then $a \in I'$ as $a \sqsubseteq x \in I'$ and $b \notin J'$ and, by Proposition 4, $r(a) \leqslant \max\{r(a_i) \mid a_i \sqsubseteq x\} < r(c)$ and similarly $r(b) < r(c)$.

Now there are two cases. If $a \notin I$ then it is a witness for I and I' of rank less than $r(c)$. Otherwise, $a \in I$ and so $b = c(a) \in J$ since $c \in I \boxrightarrow J$, and thus b is a witness for J and J' of rank less than $r(c)$. In either case, we have $c(I \boxrightarrow J,\ I' \boxrightarrow J') = r(c) > \min(c(I, I'), c(J, J'))$. ∎

Note that this theorem would fail if we kept the same definition of exponentiation but allowed arbitrary sets. Since $\varnothing \boxrightarrow \varnothing = \mathbf{V} \to \mathbf{V}$ and $(\mathbf{V} \to \mathbf{V}) \boxrightarrow \varnothing = \varnothing$, exponentiation would not be contractive in its first argument.

4.3. *Quantification*

Suppose $f: \mathscr{I}(D)^{n+1} \to \mathscr{I}(D)$ is a function of $n+1$ arbuments. Then we can produce a function of n arguments by "quantifying" over its first argument. The *universal quantification* of f relative to a given collection of ideals $\mathscr{K} \subseteq \mathscr{I}(D)$ is defined by

$$(\forall_{\mathscr{K}} f)(J_1, \ldots, J_n) = \bigcap_{I \in \mathscr{K}} f(I, J_1, \ldots, J_n)$$

and the *existential quantification* by

$$(\exists_{\mathscr{K}} f)(J_1, \ldots, J_n) = \bigsqcup_{I \in \mathscr{K}} f(I, J_1, \ldots, J_n).$$

It is here that fixing on a particular collection of sets as the types—the ideals—makes a difference to the *definition* of our operations, since it affects the range of variation of the ideal I in the above definitions.

THEOREM 8. *If* $f: \mathscr{I}(D)^{n+1} \to \mathscr{I}(D)$ *is contractive (nonexpansive) in its last n arguments, so are its universal and existential quantifications.*

Proof. For universal quantification, assume f is contractive in its last n arguments and let $g = \forall_{\mathscr{K}} f$. Let b be a witness of minimum rank for $g(J_1,..., J_n)$ and $g(J_1',..., J_n')$, with, say, $b \in g(\mathbf{J}) - g(\mathbf{J}')$. Since $b \notin g(\mathbf{J}')$ there exists an ideal $I \in \mathscr{K}$ such that $b \notin f(I, \mathbf{J}')$, while $b \in f(I, \mathbf{J})$ since $b \in \bigcap_{K \in \mathscr{K}} f(K, \mathbf{J})$. Hence b is a witness for $f(I, \mathbf{J})$ and $f(I, \mathbf{J}')$ and we have

$$\begin{aligned} c(g(\mathbf{J}), g(\mathbf{J}')) &= r(b) \\ &\geqslant c(f(I, \mathbf{J}), f(I, \mathbf{J}')) \\ &> \min_i c(J_i, J_i') \end{aligned}$$

by the assumption that f is contactive in its last n arguments.

The proof that $\forall_{\mathscr{K}} f$ is nonexpansive when f is in its last n arguments is very similar, differing only in the last step. The proofs for existential quantification are also similar, given the fact that $(\bigsqcup_\lambda I_\lambda)^\circ = \bigcup_\lambda I_\lambda^\circ$, and are left to the reader. ∎

4.4. *Fixed Points*

Our last construction makes sense in a general metric space setting. Let $f: X \times Y_1 \times \cdots \times Y_n \to X$ be a function of non-empty complete metric spaces that is contractive in its first argument. Define the "parameterized fixed-point" function $\mu f: Y_1 \times \cdots \times Y_n \to X$ by taking $(\mu f)(y_1,..., y_n)$ to be the unique element x of X such that $x = f(x, y_1,..., y_n)$ as guaranteed by the Banach fixed-point theorem.

THEOREM 9. *If* f *is contractive (nonexpansive) so is* μf.

Proof. Suppose f is contractive with coefficient $0 < r < 1$. It is easy to show that f is contractive in its first argument for each fixed set of values for its last n arguments, so the function $g = \mu f$ is well defined. We define a sequence of functions $g_m: Y_1 \times \cdots \times Y_n \to X$ converging to g as follows: take some fixed x_0 and let

$$\begin{aligned} g_0(\mathbf{y}) &= x_0 \\ g_{m+1}(\mathbf{y}) &= f(g_m(\mathbf{y}), \mathbf{y}). \end{aligned}$$

Then for all $\mathbf{y}$, $g(\mathbf{y}) = \lim_{m \geqslant 0} g_m(\mathbf{y})$.

Next we prove by induction on m that each g_m is contractive with coefficient r. For $m=0$ this is immediate, since g_0 is a constant function. For $m+1$ we calculate

$$
\begin{aligned}
&d(g_{m+1}(\mathbf{y}), g_{m+1}(\mathbf{y}')) \\
&= d(f(g_m(\mathbf{y}), \mathbf{y}), f(g_m(\mathbf{y}'), \mathbf{y}')) \\
&\leqslant r \max\{d(g_m(\mathbf{y}), g_m(\mathbf{y}')), d(y_1, y_1'), \ldots, d(y_n, y_n')\} \quad (f \text{ is contractive}) \\
&\leqslant r \max\{(r \max_i d(y_i, y_i')), d(y_1, y_1'), \ldots, d(y_n, y_n')\} \quad (\text{induction hypothesis}) \\
&= r \max_i d(y_i, y_i').
\end{aligned}
$$

Now to show that g itself is contractive with coefficient r we note that, by the triangle inequality, for every $m \geqslant 0$ we have

$$
\begin{aligned}
d(g(\mathbf{y}), g(\mathbf{y}')) &\leqslant d(g(\mathbf{y}), g_m(\mathbf{y})) + d(g_m(\mathbf{y}), g_m(\mathbf{y}')) + d(g_m(\mathbf{y}'), g(\mathbf{y}')) \\
&\leqslant r \max_i d(y_i, y_i') + d(g(\mathbf{y}), g_m(\mathbf{y})) + d(g_m(\mathbf{y}'), g(\mathbf{y}')).
\end{aligned}
$$

Since the latter two summands tend to 0 as m goes to infinity, this establishes that g is contractive as required.

The proof of nonexpansiveness is essentially the same, but with the coefficient r satisfying the weaker constraint $0 < r \leqslant 1$. ▮

The functions shown to be contractive/nonexpansive in the above theorems are the semantic counterparts of constructors appearing in the type expressions below. The results of this section will be applied to show that the semantics of type expressions is well defined.

5. Expressions and Their Semantics

Here we formally specify a language **Exp** of expressions (ranged over by e) and a language **TExp** of type expressions (ranged over by σ, τ). As may be expected from the above, our expression language is nothing but the untyped λ-calculus equipped with a suitable set of constants. It is given by the following abstract syntax grammar, where x ranges over the set of value variables, **Var**, and c ranges over the set of constants: true, false, cond, 0, $+1$, -1, Z, pair, π_1, π_2, inl, inr, outl, outr, isl, isr.

$$e ::= c \mid x \mid \lambda x.e \mid e(e).$$

The syntax of type expressions is also given by an abstract syntax grammar, where t ranges over the set of type variables, **TVar**.

$$\sigma ::= \mathbf{bool} \mid \mathbf{int} \mid t \mid \sigma \to \sigma \mid \sigma \times \sigma \mid \sigma + \sigma \mid \sigma \cap \sigma \mid \sigma \cup \sigma \mid \forall t.\sigma \mid \exists t.\sigma \mid \mu t.\sigma$$

In fact, we cannot allow all such expressions since we can only give meaning to $\mu t.\sigma$ when σ denotes a contractive function of t. So say σ is (*formally*) *contractive* in t iff one of the following conditions hold:

1. σ has one of the forms **bool**, **int**, t' (with $t' \neq t$), $\sigma_1 \to \sigma_2$, $\sigma_1 \times \sigma_2$, or $\sigma_1 + \sigma_2$.

2. σ has one of the forms $\sigma_1 \cap \sigma_2$ or $\sigma_1 \cup \sigma_2$ with both σ_1 and σ_2 contractive in t.

3. σ has one of the forms $\forall t'.\sigma_1$, $\exists t'.\sigma_1$, or $\mu t'.\sigma_1$ with either $t' = t$ or σ_1 contractive in t.

Now we take **TExp** to be the set of *well-formed* type expressions where σ is *well formed* iff one of the following conditions hold:

1. σ is **bool**, **int**, or t.

2. σ has one of the forms $\sigma_1 \to \sigma_2$, $\sigma_1 \times \sigma_2$, $\sigma_1 + \sigma_2$, $\sigma_1 \cap \sigma_2$, or $\sigma_1 \cup \sigma_2$ with both σ_1 and σ_2 well formed.

3. σ has one of the forms $\forall t.\sigma_1$ or $\exists t.\sigma_1$ with σ_1 well formed.

4. σ has the form $\mu t.\sigma_1$ with σ_1 well formed and contractive in t.

Note that if we omit the $\cap$ and $\cup$ operators, as we normally shall in a practical type system because of the type checking difficulties associated with those operations, then the only type expression that is not contractive in t is t itself, so the only type expressions that are not well formed are those containing $\mu t.t$.

We provide semantic interpretations of these languages by defining the two semantic functions

$$\mathscr{E}: \mathbf{Exp} \to \mathbf{Env} \to \mathbf{V}$$

$$\mathscr{T}: \mathbf{TExp} \to \mathbf{TEnv} \to \mathscr{I}(\mathbf{V})$$

where $\mathbf{Env} = \mathbf{Var} \to \mathbf{V}$ is the set of *environments* (ranged over by ρ) and $\mathbf{TEnv} = \mathbf{TVar} \to \mathscr{I}(\mathbf{V})$ is the set of *type environments* (ranged over by ν).

The definition of $\mathscr{E}$ is by structural induction:

$$\mathscr{E}[\![\text{true}]\!]\rho = \text{true}$$

$$\mathscr{E}[\![\text{cond}]\!]\rho = \lambda a \in \mathbf{V}.\mathbf{if}\ \text{is}_0(a)\ \mathbf{then}\ \text{cond}\ a\ \mathbf{else}\ \mathbf{wrong}$$

$$\mathscr{E}[\![x]\!]\rho = \rho[\![x]\!]$$

$$\mathscr{E}[\![\lambda x, e]\!]\rho = \lambda a \in \mathbf{V}.\mathscr{E}[\![e]\!]\ \rho[a/x]$$

$$\mathscr{E}[\![e'(e)]\!]\rho = \textbf{if}\ \mathrm{is}_2(\mathscr{E}[\![e']\!]\rho)\ \textbf{then}\ (\mathscr{E}[\![e']\!]\rho)(\mathscr{E}[\![e]\!]\rho)\ \textbf{else wrong}$$

where $\rho[a/x]$ is the environment that is identical to ρ except that it maps x to a. In the above we have made free use of the convention that $\mathbf{V}$ can be regarded as given by an actual equality and that the evident injections can be regarded as inclusions. We have also only treated two of the constants; the others are handled in a similar way.

$\mathscr{T}$ is defined by the following structural induction, in which $\mathscr{K} \subseteq \mathscr{I}(D)$ is the collection of ideals not containing **wrong**.

$$\mathscr{T}[\![\textbf{bool}]\!]v = \mathbf{T}$$

$$\mathscr{T}[\![\textbf{int}]\!]v = \mathbf{N}$$

$$\mathscr{T}[\![t]\!]v = v[\![t]\!]$$

$$\mathscr{T}[\![\sigma_1 \to \sigma_2]\!]v = \mathscr{T}[\![\sigma_1]\!]v \boxminus \mathscr{T}[\![\sigma_2]\!]v$$

$$\mathscr{T}[\![\sigma_1 \times \sigma_2]\!]v = \mathscr{T}[\![\sigma_1]\!]v \boxtimes \mathscr{T}[\![\sigma_2]\!]v$$

$$\mathscr{T}[\![\sigma_1 + \sigma_2]\!]v = \mathscr{T}[\![\sigma_1]\!]v \boxplus \mathscr{T}[\![\sigma_2]\!]v$$

$$\mathscr{T}[\![\sigma_1 \cap \sigma_2]\!]v = \mathscr{T}[\![\sigma_1]\!]v \cap \mathscr{T}[\![\sigma_2]\!]v$$

$$\mathscr{T}[\![\sigma_1 \cup \sigma_2]\!]v = \mathscr{T}[\![\sigma_1]\!]v \cup \mathscr{T}[\![\sigma_2]\!]v$$

$$\mathscr{T}[\![\forall t.\sigma]\!]v = \forall_{\mathscr{K}}(\lambda I \in \mathscr{I}(D).\mathscr{T}[\![\sigma]\!](v[I/t]))$$

$$\mathscr{T}[\![\exists t.\sigma]\!]v = \exists_{\mathscr{K}}(\lambda I \in \mathscr{I}(D).\mathscr{T}[\![\sigma]\!](v[I/t]))$$

$$\mathscr{T}[\![\mu t.\sigma]\!]v = \mu(\lambda I \in \mathscr{I}(D).\mathscr{T}[\![\sigma]\!](v[I/t])).$$

THEOREM 10. *The semantic function $\mathscr{T}$ is well defined.*

Proof. We prove by structural induction on σ that: (1) for all v, $\mathscr{T}[\![\sigma]\!]v$ is well defined; (2) for any t, $\lambda I \in \mathscr{I}(D).\mathscr{T}[\![\sigma]\!](v[I/t])$ is nonexpansive, and is contractive if σ is contractive in t. The results of the last section make such a proof quite straightforward. It is straightforward to prove that provided $v(t)$ does not contain **wrong** for any t then neither does $\mathscr{T}[\![\sigma]\!]v$. ∎

6. RULES FOR TYPE INFERENCE

We give rules to infer type assertions of the form $e : \sigma$, relative to an assignment $\mathscr{A}$ of type expressions to a finite set of variables. Thus the rules concern *sequents* of the form $\mathscr{A} \vdash e : \sigma$, where the type expression σ is assumed to be well formed.

The rules are keyed to the syntax of type expressions, with most of them being introduction and elimination rules for the various type constructs:

Constants. Types are assigned to the constants as follows:

true: **bool**	false: **bool**	cond: $\forall t.\mathbf{bool} \to (t \to (t \to t))$
0: **int**	z: **int** $\to$ **bool**	
$+1$: **int** $\to$ **int**	-1: **int** $\to$ **int**	
pair: $\forall s.\forall t.s \to t \to (s \times t)$	π_1: $\forall s.\forall t.(s \times t) \to s$	π_2: $\forall s.\forall t.(s \times t) \to t$
inl: $\forall s.\forall t.s \to (s + t)$	outl: $\forall s.\forall t.(s + t) \to s$	isl: $\forall s.\forall t.(s + t) \to$ **bool**
inr: $\forall s.\forall t.t \to (s + t)$	outr: $\forall s.\forall t.(s + t) \to t$	isr: $\forall s.\forall t.(s + t) \to$ **bool**.

The rule for constants is then

$$\mathscr{A} \vdash c : \sigma \qquad \text{(if } c : \sigma \text{ appears above).}$$

Variables. $\mathscr{A} \vdash x : \sigma$ (if $\mathscr{A}$ assigns σ to x).

Exponentiation. Below, $\mathscr{A}, x : \sigma$ is the assignment identical to $\mathscr{A}$, except that it assigns σ to x,

$$\textit{Introduction} \qquad \frac{\mathscr{A}, x : \sigma \vdash e : \tau}{\mathscr{A} \vdash \lambda x.e : \sigma \to \tau},$$

$$\textit{Elimination} \qquad \frac{\mathscr{A} \vdash e : \sigma \to \tau \quad \mathscr{A} \vdash e' : \sigma}{\mathscr{A} \vdash e(e') : \tau}.$$

Intersection.

$$\textit{Introduction} \qquad \frac{\mathscr{A} \vdash e : \sigma \quad \mathscr{A} \vdash e : \tau}{\mathscr{A} \vdash e : \sigma \cap \tau},$$

$$\textit{Elimination} \qquad \frac{\mathscr{A} \vdash e : \sigma \cap \tau}{\mathscr{A} \vdash e : \sigma} \quad \frac{\mathscr{A} \vdash e : \sigma \cap \tau}{\mathscr{A} \vdash e : \tau}.$$

Union. Below, $e'[e/x]$ is the expression obtained from e' by substituting e for x, renaming bound variables as necessary,

$$\textit{Introduction} \qquad \frac{\mathscr{A} \vdash e : \sigma}{\mathscr{A} \vdash e : \sigma \cup \tau} \quad \frac{\mathscr{A} \vdash e : \tau}{\mathscr{A} \vdash e : \sigma \cup \tau},$$

$$\textit{Elimination} \qquad \frac{\mathscr{A} \vdash e : \sigma \cup \sigma' \quad \mathscr{A}, x : \sigma \vdash e' : \tau \quad \mathscr{A}, x : \sigma' \vdash e' : \tau}{\mathscr{A} \vdash e'[e/x] : \tau}.$$

Universal Quantification. Below, "t is not free in $\mathscr{A}$" means that t is not free in any type assigned by $\mathscr{A}$, and $\sigma[\tau/t]$ is the result of substituting τ for t in σ, renaming bound variables as necessary (which can easily be shown to preserve well-formedness),

Introduction $\dfrac{\mathscr{A} \vdash e : \sigma}{\mathscr{A} \vdash e : \forall t.\sigma}$ (t is not free in $\mathscr{A}$),

Elimination $\dfrac{\mathscr{A} \vdash e : \forall t.\sigma}{\mathscr{A} \vdash e : \sigma[\tau/t]}$.

Existential Quantification.

Introduction $\dfrac{\mathscr{A} \vdash e : \sigma[\tau/t]}{\mathscr{A} \vdash e : \exists t.\sigma}$,

Elimination $\dfrac{\mathscr{A} \vdash e : \exists t.\sigma \quad \mathscr{A}, x : \sigma \vdash e' : \tau}{\mathscr{A} \vdash e'[e/x] : \tau}$ (t is not free in $\mathscr{A}$ or τ).

Recursion.

Introduction $\dfrac{\mathscr{A} \vdash e : \sigma[\mu t.\sigma/t]}{\mathscr{A} \vdash e : \mu t.\sigma}$,

Elimination $\dfrac{\mathscr{A} \vdash e : \mu t.\sigma}{\mathscr{A} \vdash e : \sigma[\mu t.\sigma/t]}$.

We say that $\mathscr{A} \vdash e : \sigma$ holds iff there is a proof of it using the above rules, and read: "e is *well typed*, with type σ, relative to $\mathscr{A}$." Note that if $\mathscr{A} \vdash e : \sigma$ holds then $\mathscr{A}$ assigns types to all free variables of e and, further, for any $\mathscr{A}$, $\mathscr{A} \vdash e : \sigma$ holds iff $\mathscr{A}' \vdash e : \sigma$ holds, where $\mathscr{A}'$ is the *restriction* of $\mathscr{A}$ to the free variables of e. That the rules do not all come in introduction/elimination pairs (Gentzen style) is due to our use of constants to treat types where possible (Hilbert style). Intersection, universal quantification, and perhaps (see below) recursion, union, and existential quantification do not seem best treated in this way; they seem descriptive rather than computational.

Table 2 shows the use of the rules to show that $\vdash Y : \forall t.(t \to t) \to t$ indeed holds (we omit $\mathscr{A}$ when it is empty). Recall that

$$Y = \lambda f.(\lambda x.f(xx))(\lambda x.f(xx)).$$

The exponentiation rules are those of Curry's system of functionality [9] treated in the fashion of Milner [19]. They bear a clear relation to the usual types λ-calculus: $\vdash e : \sigma$ holds under these rules iff there is a typed term e' of type σ that becomes e when the types are removed.

The rules for intersection are found in the work of Coppo *et al.* [7, 8] and Pottinger [23]. Coppo *et al.* [8] also consider a universal type $\top$ that we discuss below, taking $\top =_{\text{def}} \exists t.t$. From Coppo and Dezani–Ciancaglini [7] and Pottinger [23], a term is typable in this intersection type discipline iff it is strongly normalizable, an undecidable property. The rules for

TABLE 2

Proof of $\vdash Y: \forall t.\ (t \to t) \to t$

Type Assignment $\vdash$	Expression : Type	Ruel
$x : \mu s.(s \to t), f: t \to t \vdash$	$x : \mu s.(s \to t)$	Variable
$x : \mu s.(s \to t), f: t \to t \vdash$	$x : (\mu s.(s \to t) \to t)$	Recursion elimination
$x : \mu s.(s \to t), f: t \to t \vdash$	$xx : t$	Exponentiation elimination
$x : \mu s.(s \to t), f: t \to t \vdash$	$f : t \to t$	Variable
$x : \mu s.(s \to t), f: t \to t \vdash$	$f(xx) : t$	Exponentiation elimination
$f: t \to t \vdash$	$\lambda x.f(xx) : (\mu s.(s \to t) \to t)$	Exponentiation introduction
$f: t \to t \vdash$	$\lambda x.f(xx) : \mu s.(s \to t)$	Recursion introduction
$f: t \to t \vdash$	$(\lambda x.f(xx))(\lambda x.f(xx)) : t$	Exponentiation elimination
$\vdash$	$\lambda f.(\lambda x.f(xx))(\lambda x.f(xx)) : (t \to t) \to t$	Exponentiation introduction
$\vdash$	$\lambda d.(\lambda x.f(xx))(\lambda x.f(xx)) : \forall t.\ (t \to t) \to t$	Universal quantification introduction

union seem to be novel; the introduction rule is obvious, and the elemination rule is intended as a finite analog of the elimination rule for existential quantification, discussed below. As remarked above, type-checking difficulties seem to make intersection and union awkward in practice; moreover it is not clear if there are any potential benefits from their use.

The rules for universal quantification are related to Reynolds' second-order typed λ-calculus in much the same way that the rules for exponentiation were related to the ordinary typed λ-calculus (see [25]). With the addition of the rules for existential quantification this relation can be extended to Girard's broader second-order calculus [12]; however, the lack of a linguistic correlate for the elimination construct for existential types in the typed λ-calculus forces us to use substitution instead. Perhaps it would be better to have a construct, **with** e **for** x **in** e', meaning the same as $e'[e/x]$, and the corresponding typing rule:

$$\frac{\mathscr{A} \vdash e : \exists t.\sigma \quad \mathscr{A}, x : \sigma \vdash e' : \tau}{\mathscr{A} \vdash (\textbf{with } e \textbf{ for } x \textbf{ in } e') : \tau} \qquad (t \text{ not free in } \mathscr{A} \text{ or } \tau).$$

A similar construct might lead to the right elimination rule for union types.

As particular cases of quantification we can consider "universal" and "empty" (better, "top" and "bottom") types, $\top =_{\text{def}} \exists t.t$ and $\bot =_{\text{def}} \forall t.t$ with the following derived rules for $\top$:

Introduction $$\frac{\mathscr{A} \vdash e : \sigma}{\mathscr{A} \vdash e : \top},$$

Elimination $$\frac{\mathscr{A} \vdash e : \top \quad \mathscr{A}, x : t \vdash e' : \sigma}{\mathscr{A} \vdash e'[e/x] : \sigma} \qquad (t \text{ not free in } \mathscr{A} \text{ or } \sigma);$$

and for $\bot$:

$$\textit{Introduction} \quad \frac{\mathscr{A} \vdash e : t}{\mathscr{A} \vdash e : \bot} \quad (t \text{ not free in } \mathscr{A}),$$

$$\textit{Elimination} \quad \frac{\mathscr{A} \vdash e : \bot}{\mathscr{A} \vdash e : \sigma}.$$

Note the difference from the rules of Coppo *et al.* for $\top$; these arise because $\top$ is not in fact universal, since it does not contain **wrong**.

Finally, the rules for recursion are motivated by our solving type equations, not just isomorphisms. It is not clear how useful the rules are in this form. They are hardly very restrictive, since, as noted in Section 1.2, every term of the untyped λ-calculus receives the type $\mu t.t \to t$. On the other hand, they are not sufficient to establish compatibility between certain equivalent recursive type expressions (i.e., type expressions having the same infinite unfoldings). In practice, the use of recursive types is often mediated by certain explicit constructs, such as pointers in Pascal or constructors in Standard ML, and these simplify the problem of type checking with recursive types.

Turning to the soundness of the system, we define $\models_{\rho,v} \mathscr{A}$ as meaning that $\mathscr{E}[\![x]\!]\rho \in \mathscr{T}[\![\sigma]\!]v$ whenever $\mathscr{A}$ assigns σ to x; then we define $\mathscr{A} \models_{\rho,v} e : \sigma$ as meaning that if $\models_{\rho,v} \mathscr{A}$ then $\mathscr{E}[\![e]\!]\rho \in \mathscr{T}[\![\sigma]\!]v$; finally, $\mathscr{A} \models e : \sigma$ means that for all ρ in **Env** and v in **TEnv**, $\mathscr{A} \models_{\rho,v} e : \sigma$.

THEOREM 11 (Soundness). *If* $\mathscr{A} \vdash e : \sigma$ *then* $\mathscr{A} \models e : \sigma$.

Proof. The proof proceeds by induction on the size of the formal proof tree of $\mathscr{A} \vdash e : \sigma$. We content ourselves with a few illustrative cases.

For the conditional it is enough to show $\mathscr{E}[\![\text{cond}]\!]\rho$ $(=f$, say) is in $\mathbf{T} \boxminus (I \boxminus (I \boxminus I))$ for any ideal I. But this is equivalent to showing that $fabc \in I$ whenever $a \in \mathbf{T}$, $b \in I$, $c \in I$. Since the only possibilities for a are *true*, *false*, and $\bot$, we have that $fabc$ is either b, c, or $\bot$, and hence it is in I.

For exponentiation the proof goes as in [19, 14], for intersection, the proof goes as in [8], and we leave the cases of union and universal quantification to the reader.

For existential quantification, first consider the introduction rule. Suppose, by the induction hypothesis, that $\mathscr{A} \models e : \sigma[\tau/t]$ and we wish to show, for given ρ, v, that $\mathscr{A} \models e : \exists t.\sigma$. Assuming that $\models_{\rho,v} \mathscr{A}$, we have

$$\mathscr{E}[\![e]\!]\rho \in \mathscr{T}[\![\sigma[\tau/t]]\!]v = \mathscr{T}[\![\sigma]\!](v[\mathscr{T}[\![\tau]\!]v/t]) \subset \mathscr{T}[\![\exists t.\sigma]\!]v$$

where the equality follows by an easily proved substitution lemma.

As for the elimination rule, by the induction hypothesis we can suppose

that $\mathscr{A} \models e : \exists t, \sigma$ and $\mathscr{A}, x : \sigma \models e' : \tau$ (with t not **free** in $\mathscr{A}$ or τ) and we wish to show for given ρ, v satisfying $\models_{\rho,v} \mathscr{A}$ that $\mathscr{A} \models_{\rho,v} e'[e/x] : \tau$. We have $\mathscr{E}[\![e]\!]\rho \in \mathscr{T}[\![\exists t.\sigma]\!]v$ by the definition of $\models$. Let a be finite with $a \sqsubseteq \mathscr{E}[\![e]\!]\rho$. Then since $a \in \mathscr{T}[\![\exists t.\sigma]\!]v$, Proposition 1 yields an ideal $I \in \mathscr{K}$ such that $a \in \mathscr{T}[\![\sigma]\!]v'$, where $v' = v[I/t]$. Now letting $\rho' = \rho[a/x]$ we have $\models_{\rho',v'} \mathscr{A}, x : \sigma$, as t is not **free** in $\mathscr{A}$. Hence $\mathscr{E}[\![e']\!]\rho' \in \mathscr{T}[\![\tau]\!]v' = \mathscr{T}[\![\tau]\!]v$, as t is not **free** in τ. Thus

$$\begin{aligned}\mathscr{E}[\![e'[e/x]]\!]\rho &= \mathscr{E}[\![e']\!]\ \rho[\mathscr{E}[\![e]\!]\rho/x] \\ &= \bigsqcup \{\mathscr{E}[\![e']\!]\ \rho[a/x] \mid a \text{ finite and } a \sqsubseteq \mathscr{E}[\![e]\!]\rho\}\end{aligned}$$

by an easy substitution lemma and the continuity of $\mathscr{E}$, respectively. The latter expression is in $\mathscr{T}[\![\tau]\!]v$ by the closure of ideals under limits.

Finally, the soundness of the recursion rules is straightforward. ∎

7. Conclusion

This paper justifies the extension of the type system of [17] to include recursive types. However, in contrast to the type system of Milner [19, 10], it is difficult to decide in general whether a given expression has a given type. It can be shown that this is a Π_1-complete question, even when restricted to terms of the pure λ-calculus and the type **int** → **int**. It follows that no recursively enumerable axiomatic type system can be complete for the true-type assertions.

On a practical level, this paper justifies the extension of unification-based type checking algorithms for the type systems of [14, 19, 10] to allow circular unification. Similar algorithms can be applied to check the Algol family of languages, even though the types of procedure parameters are not specified. Note that dialects of Pascal that require full declaration of the types of procedure parameters do not allow self-application to be expressed, since they do not support recursive functional types.

APPENDIX: Strong Ideals

Here we obtain results for strong ideals completely analogous to those obtained for the weak ideals. In the case of weak ideals, we could use finite elements to treat lubs as set-theoretic unions, using the formula $(\bigsqcup I_\lambda)^\circ = \bigcup I_\lambda^\circ$. This equality does not hold for the strong ideals, which can, however, be treated by using more special kinds of elements than the finite ones.

A.1. *Primal Domains*

DEFINITION. Let D be a cpo. An element p of D is *prime* (resp. an *ω-prime*) iff whenever X is a finite subset (resp. any countable subset) of D, whose lub exists, such that $p \sqsubseteq \bigsqcup X$, then $p \sqsubseteq x$ for some $x \in X$.

Clearly, every ω-prime is an ω-finite prime; the converse holds in any consistently complete cpo. Note that $\perp$ is not a prime according to this definition. Primes play a very important role in lattice theory. The reader should beware that our primes are often termed the *co-primes* by other authors as their primes are the duals of ours.

DEFINITION. A consistently complete cpo is *ω-primal* iff it has countably many ω-primes and every element is the lub of the ω-primes beneath it. The ω-primal elements of any $X \subseteq D$ are denoted by $X^\bullet$.

Every ω-primal consistently complete cpo is ω-algebraic and the finite elements are the finite lubs of ω-primes. Conversely, every domain (ω-algebraic consistently-complete cpo) in which the finite elements are the finite lubs of ω-primes is ω-primal. To show that some particular domain D is ω-primal it is enough, therefore, to display a set of ω-primes and show every finite element is a finite lub of some of these ω-primes; that set is then the set of all ω-primes. The ω-prime elements of an ω-primal domain are even *completely prime* in the sense that whenever less than the lub of an arbitrary subset they are less than some element of the subset.

In an ω-primal domain D, a *decomposition* of an element x is a set of primes $X \subseteq D^\bullet$, such that $x = \bigsqcup X$. Every finite element $a \in D^\circ$ has a unique decomposition into a finite, "independent," set of primes, called the *components* of a. The notion of independence is defined as follows:

DEFINITION. Let D be consistently complete cpo. Say $x \in D$ *depends* on a consistent $X \subseteq D$ if $x \sqsubseteq \bigsqcup X$. A set $X \subseteq D$ is *independent* if no x in X depends on $X \setminus \{x\}$.

FACT 12. *Let D be an ω-primal domain. Then for every element b of D° there is a unique independent set X of primes such that $b = \bigsqcup X$ and further X is a finite set of ω-primes.*

Now we turn to our constructions with the aim of showing **V** to be ω-primal. First, every flat cpo is ω-primal as all its elements, except $\perp$, are primes.

Let D and E be ω-primal domains. Their product $D \times E$ is ω-primal as we have

$$(D \times E)^\bullet = (D^\bullet \times \{\perp_E\}) \cup (\{\perp_D\} \times E^\bullet)$$

since any finite element $\langle a, b\rangle$ can be decomposed into

$$\{\langle p_1, \perp\rangle, \ldots, \langle p_m, \perp\rangle, \langle \perp, q_1\rangle, \ldots, \langle \perp, q_n\rangle\}$$

if a can be decomposed into $\{p_1, \ldots, p_m\}$ and b into $\{q_1, \ldots, q_n\}$. Further, this decomposition is independent if those of a and b are. Their sum $D + E$ is ω-primal with

$$(D + E)^\bullet = \mathrm{inl}(D^\bullet) + \mathrm{inr}(E^\bullet)$$

as every finite element is either $\mathrm{inl}(a)$ with $a \in D^\circ$, in which case it can be decomposed into $\{\mathrm{inl}(p_i)\}$, where $\{p_i\}$ is a decomposition of a, or else it is $\mathrm{inr}(b)$, which has decomposition $\{\mathrm{inr}(q_j)\}$, where $\{q_j\}$ is a decomposition of $b \in E^\circ$. As in the case of products, independence, and uniqueness of decompositions are inherited. Finally,

FACT 13. *If D and E are ω-primal domains, then the function space, $(D \to E)$, is ω-primal and*

$$(D \to E)^\bullet = \{a \Rightarrow q \mid a \in D^\circ, q \in E^\bullet\}.$$

If $f = \bigsqcup\{a_i \Rightarrow b_i\}$ is a finite function and if $\{q_{ij}\}$ is a decomposition of b_i into primes, then $\{a_i \Rightarrow q_{ij}\}$ is a decomposition of f.

In some other parts of semantics, a strict product construction:

$$D \otimes E \underset{\text{def}}{=} \{\langle d, e\rangle \in D \times E \mid s = \perp_D \text{ iff } e = \perp_E\}$$

with partial order inherited from $D \times E$, has been considered [22]. Unfortunately, this is not ω-primal even if D and E are. For example, $(\mathbf{W} \times \mathbf{W}) \otimes (\mathbf{W} \times \mathbf{W})$ is not ω-primal. So, if we had added an extra summand $\mathbf{V} \otimes \mathbf{V}$ to the definition of $\mathbf{V}$ the approach via primes of this Appendix would not work; it is an interesting open question how one should then proceed.

Turning to $\mathbf{V}$, we begin by relating embeddings to primes.

DEFINITION. Let $f: D \to E$ be a continuous function where D and E are consistently-complete cpos. Then f is *additive* iff whenever $x \sqcup y$ exists in D then $f(x \sqcup y) = f(x) \sqcup f(y)$.

Clearly, the identity is additive and the additive functions are closed under composition. Also all embeddings are additive.

LEMMA 14. *Let $\phi: D \to E$ be an embedding, where D and E are consistently-complete cpos. Then if ϕ^R is additive, ϕ preserves primes (and so also preserves ω-primes).*

It is easy to show that if f and g are additive, so are $(f+g)$, $(f\times g)$, and $(f\to g)$. It follows by an easy induction on n that all the ϕ_n^R are additive (and of course the ϕ_n are additive, too).

LEMMA 15. *The* $\mu_n^R\colon \mathbf{V}\to\mathbf{V}_n$ *are additive.*

Note that it follows that the v_n are also additive and of course both θ and θ^{-1} are additive as are all isomorphisms. Because of all this regarding θ and θ^{-1} as identities does not create ambiguities as to which elements are prime and neither does viewing the injection of $\mathbf{T}$, $\mathbf{N}$, $(\mathbf{V}\to\mathbf{V})$, and so on, in $\mathbf{V}$ as inclusions.

FACT 16. *The domain* $\mathbf{V}$ *is* ω*-primal and indeed we have*

$$\mathbf{V}^\bullet = \bigcup_n \mu_n(\mathbf{V}_n^\bullet).$$

Further, if $\{p_i\}$ *is any decomposition of* a *in* $\mathbf{V}_n^\circ$ *into primes, then* $\{\mu_n(p_i)\}$ *is one of* $\mu_n(a)$ *into primes; it is independent if that of* a *is.*

A.2. *Strong Ideals*

Let D be an ω-primal domain. Recall from Section 3.1 that $\mathscr{I}_0(P)$ and $\mathscr{I}^+(P)$ denote the collection of all order ideals and of all strong ideals, respectively, of any given partical order P. Then regarding $D^\bullet$ as a partial order (inherited from D) and structuring ideals by subset we have

PROPOSITION 17. *The correspondence* $I\mapsto I^\bullet$ *is an isomorphism of* $\langle \mathscr{I}^+(D), \subseteq\rangle$ *and* $\langle \mathscr{I}_0(D^\bullet), \subseteq\rangle$ *with inverse* $J\mapsto \{x\in D \mid \forall p\in D^\bullet. p\sqsubseteq x$ *implies* $p\in J\}$. *Further* $\mathscr{I}^+(D)$ *is a complete lattice with meets given by intersection and joins given by* $(\bigsqcup_\lambda I_\lambda)^\bullet = \bigcup_\lambda I_\lambda{}^\bullet$ *and finite joins by the formula* $I\sqcup I' = \{x\sqcup y \mid x\in I,\ y\in I',$ *and* $x\sqcup y$ *exists*$\}$.

Proof. Let $f\colon \mathscr{I}^+(D)\to\mathscr{I}_0(D^\bullet)$ and $g\colon \mathscr{I}_0(D^\bullet)\to\mathscr{I}^+(D)$ name the correspondences. Clearly $f(I)$ is an order ideal in $\mathscr{I}_0(D^\bullet)$ by condition (2) in the definition of strong ideals. To see that $g(J)$ is a strong ideal note first that $\bot\in g(J)$, since $p\sqsubseteq\bot$ never holds for a prime p. Next if $y\in g(J)$ and $x\sqsubseteq y$ then for any $p\in D^\bullet$ if $p\sqsubseteq x$ then $p\sqsubseteq y$ and so $p\in J$, and therefore we have $x\in g(J)$. Next suppose $\{x_n\}$ is an increasing sequence of elements of $g(J)$. Then if $p\sqsubseteq \bigsqcup x_n$ is an ω-prime, $p\sqsubseteq x_n$ for some x_n and so $p\in J$; therefore $\bigsqcup x_n$ is in $g(J)$. Finally, a similar argument shows $x\sqcup y$ is in $g(J)$ if x and y are and $x\sqcup y$ exists. Thus $g(J)$ is indeed a strong ideal.

Now note that

$$f(g(J)) = \{q\in D^\bullet \mid \forall p\in D^\bullet. p\sqsubseteq q \text{ implies } p\in J\} = J$$

since J is an ideal of $D^\bullet$. Also $x \in g(f(I))$ iff $\forall p \in D^\bullet$, $p \sqsubseteq x$ implies $p \in I^\bullet$. So clearly, if $x \in I$ then $x \in g(f(I))$ by condition (2) on strong ideals. Conversely, take $x \in g(f(I))$ and let $p_0, p_1, p_2, \ldots$ enumerate the ω-primes less than x (there may be none, or finitely many of them); they are all in $I^\bullet$. Then each $x_n =_{\text{def}} \bigsqcup_{i \leqslant n} p_i$ is in I by conditions (1) and (4) and so $x = \bigsqcup x_n$ is in I by condition (3). Therefore $g(f(I)) = I$ as required and we see that f is indeed an isomorphism with inverse g. Both f and g are clearly monotonic, as well.

Clearly $\bigsqcap_\lambda I_\lambda = \bigcap I_\lambda$ as the intersection of strong ideals is a strong ideal. Hence the assertion for meets holds and $\mathscr{I}^+(D)$ is a complete lattice. For the first formula for joins, we calculate that

$$\left(\bigsqcup_\lambda I_\lambda\right)^\bullet = \bigsqcup I_\lambda^\bullet \qquad \text{(since } f \text{ is an isomorphism)}$$

$$= \bigcup I_\lambda^\bullet \qquad \text{(since joins are set-theoretic unions for ideals of partial orders).}$$

Since $(I \sqcup I')^\bullet = I^\bullet \cup (I')^\bullet$, to prove the formula for finite joints it suffices to show that if $J = \{x \sqcup y \mid x \in I,\ y \in I',\ x \sqcup y \text{ exists}\}$ then $J = g(I^\bullet \cup (I')^\bullet)$. Take $x \in I$, $y \in I'$ such that $x \sqcup y$ exists. If $p \sqsubseteq (x \sqcup y)$ is an ω-prime then $p \sqsubseteq x$ or $p \sqsubseteq y$ and so $p \in I^\bullet$ or $p \in (I')^\bullet$, and hence $(x \sqcup y) \in g(I^\bullet \cup (I')^\bullet)$, thus showing that $J \subset g(I^\bullet \cup (I')^\bullet)$. Suppose $z \in g(I^\bullet \cup (I')^\bullet)$ and let $x = \bigsqcup\{p \in I^\bullet \mid p \sqsubseteq z\}$ and $y = \bigsqcup\{p \in (I')^\bullet \mid p \sqsubseteq z\}$. Then clearly $z = x \sqcup y$ and $x, y \in I$, so $z \in J$, concluding the proof. ∎

Assume now that we have a rank function $r: D^\circ \to N$. A *prime* witness for two strong ideals I and J is simply a witness that is prime, that is, an element of $I^\bullet \oplus J^\bullet$. We define the closeness function $c^+(I, J)$ to be the least possible rank of a prime witness for I and J, and if none exists, it is ∞. Analogous to Proposition 2 we have for strong ideals:

PROPOSITION 18. (i) $c^+(I, J) = \infty$ *iff* $I = I$.

(ii) $c^+(I, J) = c^+(J, I)$.

(iii) $c^+(I, K) \geqslant \min(c^+(I, J), c^+(J, K))$.

Proof. The proof is just like that of Proposition 2, but using the previous proposition instead of Proposition 1. ∎

As before, we can now define an ultrametric, $d^+(I, J) =_{\text{def}} 2^{-c^+(I,J)}$ and we get, analogously to Theorem 3 (and with the analogous proof):

THEOREM 19. *The metric space* $\langle \mathscr{I}^+(D), d^+ \rangle$ *is complete. Indeed if* $\langle I_i \rangle_{i \geqslant 0}$ *is a Cauchy sequence then its limit is* I *where* $I^\bullet = \{b \in D^\bullet \mid b \text{ is in almost all } I_i\}$. ∎

Rather than repeating proofs, we could instead have defined $c^+(I, J)$ as in Section 3.1, but using the rank function $r^+(a) =_{\text{def}} \max\{r(p_i)\}$, where $\{p_i\}$ is the prime decomposition of a.

Now turning to **V**, Proposition 4 can be read

PROPOSITION 20. (i) *Any ω-prime element c of $(\mathbf{V} \times \mathbf{V})$ is equal to $\langle a, b \rangle$, with a and b ω-prime and $r(a) < r(c)$ and $r(b) < r(c)$.*

(ii) *Any ω-prime element c of $(\mathbf{V} + \mathbf{V})$ is equal to $\mathrm{inl}(a)$, with a ω-prime and $r(a) < r(c)$ or to $\mathrm{inr}(b)$, with b ω-prime and $r(b) < r(c)$.*

(iii) *Any ω-prime element c of $(\mathbf{V} \to \mathbf{V})$ is equal to $(a \Rightarrow b)$, with a finite, b ω-prime, $r(a) < r(c)$, and $r(b) < r(c)$.*

Proof. We prove just (iii) as an example. As c is ω-prime, it is not $\perp$ and so by Proposition 4(vi) we have $c = \bigsqcup_{i=1}^{n} a_i \Rightarrow b_i$ with a_i, b_i finite and $r(a_i) < r(c)$ and $r(b_i) < r(c)$. But as c is ω-prime, we have $n = 1$ and b_1 is ω-prime, employing Fact 13. ∎

A.3. *Contractive Functions*

The analog for strong ideals of Proposition 5 holds with the corresponding proof.

PROPOSITION 21. *Meet and join (of strong ideals) are nonexpansive but not, in general, contractive.*

Proof. This is as for Proposition 6, except when showing $I \sqcup J$ nonexpansive. So, suppose $I \neq I'$ or $J \neq J'$ and let p be a prime witness of minimum rank for $I \sqcup J$ and $I' \sqcup J'$, being, say, in the former, but not the latter. As $(I \sqcup J)^\bullet = I^\bullet \cup J^\bullet$ and similarly for I' and J', we have that $p \in I - I'$ (or $J - J'$) and the proof concludes as usual. ∎

The sum, product, and exponentiation constructions are defined as before, it being easily seen that they send strong ideals to strong ideals.

THEOREM 22. *The sum, product, and exponentiation constructions are all contractive.*

Proof. We just consider the hardest case, exponentiation. Take strong ideals I, I', J, J', and let p be a prime witness of minimum rank for $I \boxrightarrow J$ and $I' \boxrightarrow J'$, being, say, only in the first of these.

By Proposition 20 we have $p = (a \Rightarrow q)$ with a finite, q ω-prime, $r(a) < r(p)$, and $r(q) < r(p)$. Since $p \notin I' \boxrightarrow J'$, there is an x in I' with $p(x) \notin J'$. This implies that $p(x) = q \notin J'$ and hence $a \sqsubseteq x$ and so $a \in I'$.

Now if $q \in J$ we are done, since then $q \in J \oplus J'$ implying $c^+(J, J') \leqslant r(q) < r(p) = c^+(I \boxrightarrow J, I' \boxrightarrow J')$. If $q \notin I$. Let $\{s_i\}$ be the unique indepen-

dent decomposition of a. Then for some i, $s_i \notin I$ since I is a strong ideal. Also, $s_i \in I'$ since $s_i \sqsubseteq a \in I'$, so s_i is a witness for I and I'. Furthermore, $r(s_i) \leqslant r(a)$ by Fact 16 and the uniqueness of independent prime decompositions. Hence $c^+(I, I') \leqslant r(s_i) < r(p)$ and we are done. ∎

Turning to quantification, we consider a collection $\mathscr{K} \subseteq \mathscr{I}^+(D)$ of strong ideals and define universal and existential quantification as in the case of weak ideals. Theorem 8 also holds for the strong ideals, the proof is the same for universal quantification, but for existential quantification we use ω-primes as in the proof of Proposition 17 for finite lubs.

Finally, the general remarks on fixed points apply to strong ideals just as well as they did to the weak ones.

Turning to type expressions and their semantics we may define $\mathscr{T}^+ : \mathbf{TExp} \to \mathbf{TEnv}^+ \to \mathscr{I}^+(\mathbf{V})$ in complete analogy to $\mathscr{T}$ in Section 5, where of course, $\mathbf{TEnv}^+ =_{\text{def}} \mathbf{TVar} \to \mathscr{I}^+(\mathbf{V})$ is the set of *strong*-type environments. Along the way one notes that as a function of the denotations of any of its free variables, the denotation of a type expression is nonexpansive, and it is contractive if the expression is contractive in the variable.

Turning to type inference, we define the truth of sequents $\mathscr{A} \models^+ e : \sigma$ as we did in the weak ideal case in Section 6. Unfortunately, the corresponding soundness theorem does not hold. Let e be the expression $\pi_2(x)(\pi_1(x))$. Then, we have that $x : \sigma \times (\sigma \to \mathbf{bool}) \vdash e : \mathbf{bool}$ holds for any σ. So, first, taking σ to be **int** and **bool** and applying the elimination rule, for union types, followed by the introduction rule for functional types, we get $\vdash \lambda x.e : \tau \to \mathbf{bool}$, where $\tau =_{\text{def}} (\mathbf{int} \times (\mathbf{int} \to \mathbf{bool}) \cup \mathbf{bool} \times (\mathbf{bool} \to \mathbf{bool}))$. And, second, taking σ to be t, we get, this time with the elimination rule for existential types, that $\vdash \lambda x.e : \tau' \to \mathbf{bool}$, where $\tau' =_{\text{def}} (\exists t.t \times (t \to \mathbf{bool}))$. However, neither $\models^+ \lambda x.e : \tau \to \mathbf{bool}$, nor $\models^+ \lambda x.e : \tau' \to \mathbf{bool}$ holds (omitting the empty environment) and it is enough to prove the first of these assertions. Note that $a =_{\text{def}} \langle 0, \perp_{\mathbf{V}} \rangle$ is in $\mathbf{N} \boxtimes (\mathbf{N} \boxplus \mathbf{T})$ and that $b =_{\text{def}} \langle \perp_{\mathbf{V}}, \lambda x \in \mathbf{V}.\mathrm{cond}_{\mathbf{V}}\, x\, \perp_{\mathbf{V}}\, \perp_{\mathbf{V}} \rangle$ is in $\mathbf{T} \boxtimes (\mathbf{T} \boxplus \mathbf{T})$ (where $\mathrm{cond}_{\mathbf{V}}$ is the denotation of cond). But under the strong ideal interpretation $a \sqcup b$ is in the denotion of τ, and if x denotes $a \sqcup b$, then e denotes **wrong** and so $\models^+ \lambda x.e : \tau \to \mathbf{bool}$ fails.

As may be supposed, the problem lies with the elimination rules for union and existential types. They are not sound, in that under the strong-ideal interpretation, their conclusions can be false although the premises are true. On the other hand, all the other rules are sound, and it remains to be seen whether adequate alternative elimination rules can be found.

RECEIVED October 3, 1985; ACCEPTED April 2, 1986

REFERENCES

1. ARNOLD, A., AND NIVAT, M. (1980), Metric interpretations of infinite trees and semantics of non deterministic recursive programs, *Theoret. Comput. Sci.* **11**, 181–205.
2. BANACH, S. (1922), Sur les opérations dans les ensembles abstraits et leurs applications aux équations intégrales, *Fund. Math.* **3**, 7–33.
3. BARENDREGT, H. P. (1981), "The Lambda Calculus: Its Syntax and Semantics," North-Holland, Amsterdam.
4. BURSTALL, R. M., MACQUEEN, D. B., AND SANNELLA, D. T. (1980), Hope: An experimental applicative language, *in* "Lisp Conference," August, pp. 136–143, Assos. Comput. Mach., New York.
5. CARTWRIGHT, R. (1985), Types as intervals, *in* "Twelfth Ann. ACM Sympos. Principles of Programm. Lang." pp. 22–36.
6. COPPO, M. (1985), A completeness theorem for recursively defined types, *in* "12th Colloq., Nafplion, Lecture Notes in Computer Science Vol. **194**, pp. 120–129, Springer-Verlag, New York/Berlin.
7. COPPO, M., AND DEZANI-CIANCAGLINI, M. (1980), An extension of the basic functionality theory for the λ-calculus, *Notre Dame J. Formal Logic* **21**, No. 4, 685–693.
8. COPPO, M., DEZANI-CIANCAGLINI, AND VENNERI, B. (1981), Functional characters of solvable terms, *Z. Math. Logik Grundlag. Math.* **26**, 45–58.
9. CURRY, H. B., AND FEYS, R. (1958), "Combinatory Logic," Vol. 1, North-Holland, Amsterdam.
10. DAMAS, L. AND MILNER, R. (1982), Principal type-schemes for Functional programs, *in* "Ninth Annu11 ACM Symposium on Principles of Programming Languages," pp. 207–212.
11. DE BAKKER, J. W., AND ZUCKER, J. I. (1982), Denotational semantics of concurrency, *in* "Fourteenth Annu. ACM Sympos. Theory of Computing" pp. 153–158.
12. GIRARD (1972), "Interpretation fonctionelle et élimination des coupres de l'arithmétique d'ordre supérieur," Thèse d'État, Université Paris VII.
13. GORDON, M. J., MILNER, A. J., AND WADSWORTH, C. P. (1979), "Edinburgh LCF," Lecture Notes in Computer Science Vol. **78**, Springer-Verlag, New York/Berlin.
14. HINDLEY, R. (1969), The principal type-scheme of an object in combinatory logic, *Trans. Amer. Math. Soc.* **146**, 29–60.
15. HINDLEY, R. (1983), The completeness theorem for typing λ-terms, *Theoret. Comput. Sci.* **22**, 1–17.
16. KAPLANSKY, I. (1972), "Set Theory and Metric Spaces," Allyn & Bacon, Boston, Mass.
17. MACQUEEN, D. B. AND SETHI, R. (1982), A higher order polymorphic type system for applicative languages, *in* "Symposium on Lisp and Functional Programming, Pittsburgh, Pa., pp. 243–252.
18. MCCRACKEN, N. J. (1979), "An investigation of a programming language with a polymorphic type structure," Ph.D. thesis, Computer and Information Science, Syracuse Univ.
19. MILNER, R. (1978), A theory of type polymorphism in programming, *J. Comput. System. Sci.* **17**, No. 3, 348–375.
20. MITCHELL, J. C., AND PLOTKIN, G. D. (1985), Abstract types have existential types, *in* "Twelfth Annu. ACM Sympos. Principles of Programming Languages," pp. 37–51.
21. MORRIS, J. H., JR.(1968), "Lambda-calculus Models of Programming Languages," Ph.D. thesis, Sloan School of Management, MIT.
22. PLOTKIN, G. (1978), Advanced domains, Summer School, Pisa.
23. POTTINGER, G. (1980), A type assignment for the strongly normalizable λ-terms, *in* J. P. Seldin and J. R. Hindley, "To H. B. Curry: Essays on Combinatory Logic, Lambda

Calculus and Formalism" (J. P. Seldin and J. R. Hindley, Eds.), pp. 561–577, Academic Press, London.
24. REYNOLDS, J. C. (1974), Towards a theory of type structure, Lecture Notes in Computer Science Vol. **19**, pp. 408–425, Springer-Verlag, Berlin/New York.
25. REYNOLDS, J. C. (1985), Three approaches to type structure, *in* "Mathematical Foundations of Software Development," Lecture Notes in Computer Science Vol. **185**, pp. 97–138, Springer-Verlag, Berlin.
26. SALLE, P. (1978), Une extension de la théorie des types, Lecture Notes in Computer Science Vol. **62**, pp. 398–410, Springer-Verlag, Berlin/New York.
27. SCOTT, D. S. (1972), Continuous lattices, Lecture Notes in Mathematics Vol. **274**, pp. 97–136, Springer-Verlag, New York/Berlin.
28. SHAMIR, A. AND WADGE, W. W. (1977), Data types as objects, "4th Colloq. Automata, Languages and Programming," Turku, Lecture Notes in Computer Science Vol. **52**, pp. 465–479, Springer-Verlag, Berlin/New York.
29. SMYTH, M. B. AND PLOTKIN, G. D. (1982), The category-theoretic solution of recursive domain equations, *SIAM J. Comput.* **11**, No. 4, 761–783.
30. SUTHERLAND, W. A. (1975), "Introduction to Metric and Topological Spaces," Oxford Univ. Press, London.
31. WADGE, W. W. (1978), Personal communication to R. Milner, March.