DOMAINS

GORDON PLOTKIN

Department of Computer Science

University of Edinburgh

1983

CONTENTS

NOTES TO THE TEX EDITION AND ACKNOWLEDGEMENTS

This is a TEX*-ed edition of the course notes "Domains" by Professor Gordon Plotkin and the copyright of this edition belongs to him+.

This edition was prepared by Yugo Kashiwagi† and Hidetaka Kondoh‡ using Prof. Donald Knuth's plain TEX with some macros (especially for the picture environment) from Dr. Leslie Lamport's LATEX macro package after some modifications for use in plain TEX and Prof. Michael Barr's macro package for category theoretic diagrams. Misprints in the original notes were corrected without notices, and some notations were changed uniformly due to typographical difficulties imposed by TEX.

The editors of this edition wish to express their deepest thanks to Dr. Tatsuya Hagino for lending them his copy of the original Prof. Plotkin's course notes so that they could photocopy it, without which the present edition could never be produced, and his memos in his copy were very helpful in correcting misprints. They also thank to Prof. Michael Barr for e-mailing one of them (HK) his diagram macro packages, which enable them to typeset many diagrams found in the present notes.

The present edition was typeset as carefully as possible. The editors, however, are afraid that there may be some misprints not appeared in the original course notes.

Therefore comments and information on misprints and/or typographical problems/advice are welcome. They are preferably notified by e-mail to one of the editors (HK‡).

June, 1992 HK

After the first version of this TEX edition was released, the editors have received more than 170 direct requests to this edition, and they have been very much encouraged by these so many responses. The editors wish to deeply thank to the readers for their encouragements. When the editors prepared the first version, regrettably however, they had no spell-checker available at hand, hence there remained so many misspellings in the first version. They sincerely apologise the readers as well as Prof. Plotkin for the low quality of the first version.

In the present second version many misspellings as well as several misprints in mathematical formulae are corrected. In this new version, the editors have changed a few typographical conventions in order to increase the typographical consistency, have changed the original spellings in the first version according to the direction of the original author, Prof. Plotkin, have recovered an original notation which seemed to be difficult for them to typeset in TEX due to their illiteracy about TEX when they prepared the first version of this TEX edition, and have made use of an AMS-font to get better output for a family of symbols appeared in Chapter 8.

First of all, the editors wish to express their deepest thanks to Prof. Plotkin for his warm encouragements and his generous permission to the distribution of this TEX edition. The editors also wish to acknowledge (in alphabetical order) Yutaka Kikuchi at Tokyo Institute of Technology, Ralf Treinen at Univ. des Saarlandes, and David Wald at Massachusetts Institute of Technology for supporting ftp services of the TEX edition, which substantially decreased the editors' efforts required for the distribution, and without their supports, the number of people accessible to this TEX edition would be significantly less than as it is, and David Wald also have repositioned the dvi file of the TEX edition in A4 size so that it is appropriate for US letter-size papers.

Last but not least, the editors wish to sincerely thank Professor John C. Mitchell, who informed them that there remained many misspellings in the first version and strongly advised them to use some spell checker, and also have kindly spent his important time to pass the TeX source files to a spell-checker and told them the results with the original author's comments and directions concerning misspellings and several original (British) spellings in the notes, and Marcelo Fiore at Univ. of Edinburgh and Daisuke Suzuki at Univ. of Tokyo for notifying them of many misprints.

November, 1992 HK

* TEX is a trademark of the American Mathematical Society.

† `kasiwagi@hcrlgw.hitachi.co.jp`

‡ `kondoh@harl.hitachi.co.jp`

1. Complete Partial Orders and Continuous Functions

It is possible to carry out a good deal of denotational semantics without invoking any mathematical apparatus other than sets, simple constructions on them and ordinary functions over them. We illustrate this with an extended example and then see what problems arise to force us out of the comfortable world of sets. Please note that we are assuming a standard approach to denotational semantics in which we ascribe meanings to syntactical objects, following the syntactic structure of these objects. The meanings are elements of some abstract domain, which is often a set. In computer science, however, it seems that sets are not enough. First, here is a simple imperative language (or, rather, a family of such languages) called *Imp*.

Syntax. There are three sets of syntactic items.

1. **BExp** — a given set of *boolean expressions*, ranged over by the metavariable, b.
2. **Act** — a given set of primitive *actions*, ranged over by the metavariable, a.
3. **Stat** — a set of *statements*, ranged over by the metavariable, s and given in terms of **BExp** and **Act** by the grammar:

$$s ::= a \mid \mathbf{dummy} \mid (s; s) \mid (\mathbf{if}\ b\ \mathbf{then}\ s\ \mathbf{else}\ s).$$

Semantic Domains. There are three semantic domains used to provide denotations for syntactic items.

1. $\mathbf{T}$ — the set, $\{tt, ff\}$, of truth-values, ranged over by t.
2. $\mathbf{S}$ — a given set of *states*, ranged over by σ.
3. $\mathbf{C} = (\mathbf{S} \to \mathbf{S})$ — the set of *commands*, ranged over by θ.

Denotations. There are three denotational functions, one for each syntactic class.

1. $\mathcal{B}\colon \mathbf{BExp} \to (\mathbf{S} \to \mathbf{T})$ — a given denotational function.
2. $\mathcal{A}\colon \mathbf{Act} \to \mathbf{C}$ — a given denotational function.
3. $\mathcal{C}\colon \mathbf{Stat} \to \mathbf{S}$ — defined by structural induction on statements by:

(i) $\mathcal{C}[\![a]\!](\sigma) = \mathcal{A}[\![a]\!](\sigma)$;

(ii) $\mathcal{C}[\![\mathbf{dummy}]\!](\sigma) = \sigma$;

(iii) $\mathcal{C}[\![s_1; s_2]\!](\sigma) = \mathcal{C}[\![s_2]\!](\mathcal{C}[\![s_1]\!](\sigma))$;

(iv) $\mathcal{C}[\![(\mathbf{if}\ b\ \mathbf{then}\ s_1\ \mathbf{else}\ s_2)]\!](\sigma) = \mathit{Cond}(\mathcal{B}[\![b]\!](\sigma), \mathcal{C}[\![s_1]\!](\sigma), \mathcal{C}[\![s_2]\!](\sigma))$

where the function $\mathit{Cond}\colon \mathbf{T} \times \mathbf{S} \times \mathbf{S} \to \mathbf{S}$ is given by:

$$\mathit{Cond}(t, \sigma, \sigma') = \begin{cases} \sigma & (\text{if } t = tt), \\ \sigma' & (\text{if } t = ff). \end{cases}$$

The reader will note how these definitions follow his computational intuitions. It is not our intention to follow up these connections in detail here, although the connection between denotational and operational semantics is of course an important topic. However, we shall rely heavily on our computational intuitions when formulating semantics. It could even be argued that we should concentrate first on an analysis of computability and then describe (or even design) programming languages accordingly. This is advice we will not take: existing languages are used to develop our ideas on computability in the first instance and so we begin with them.

The next example is a simple expressional language (or rather family of such languages) called *Dec*.

Syntax.

1. **Id** is a given set of *identifiers* ranged over by x.
2. $\mathbf{Fun}_n$ is a given set of *function symbols* ranged over by f_n $(n = 1, 2, \ldots)$.
3. $\mathbf{Op}_n$ is a given set of *operations* ranged over by o_n $(n = 1, 2, \ldots)$.
4. **BExp** is a given set of *boolean expressions* ranged over by b.
5. **Exp** is a set of *expressions* ranged over by e and given by:

$$e ::= x \mid o_0() \mid o_1(e) \mid o_2(e, e) \mid \cdots \mid (\mathbf{if}\ b\ \mathbf{then}\ e\ \mathbf{else}\ e) \mid (\mathbf{let}\ x\ \mathbf{be}\ e\ \mathbf{in}\ e) \mid (\mathbf{let}\ f_1(x)\ \mathbf{be}\ e\ \mathbf{in}\ e) \mid \cdots.$$

Semantic Domains.

1. $\mathbf{V}$ is a given set of *values* ranged over by v.
2. $\mathbf{IEnv} = \mathbf{V}^{\mathbf{Id}}$ is the set of *identifier environments* ranged over by ρ.
3. $\mathbf{FEnv} = \prod_{n \geq 1} (\mathbf{V}^n \to \mathbf{V})^{\mathbf{Fun}_n}$ is the set of *function environments*, ranged over by φ.

Denotations.

1. $\mathcal{I}_n: \mathbf{Op}_n \rightarrow (\mathbf{V}^n \rightarrow \mathbf{V})$ is given.
2. $\mathcal{B}: \mathbf{BExp} \rightarrow (\mathbf{IEnv} \times \mathbf{FEnv} \rightarrow \mathbf{T})$ is given.
3. $\mathcal{E}: \mathbf{Exp} \rightarrow (\mathbf{IEnv} \times \mathbf{FEnv} \rightarrow \mathbf{V})$ is given by structural induction:

(i) $\mathcal{E}[\![x]\!](\rho, \varphi) = \rho[\![x]\!]$;

$(\mathrm{ii})_{\mathrm{n}}$ $\mathcal{E}[\![o_n(e_1, \ldots, e_n)]\!](\rho, \varphi) = \mathcal{I}_n[\![o_n]\!](\mathcal{E}[\![e_1]\!](\rho, \varphi), \ldots, \mathcal{E}[\![e_n]\!](\rho, \varphi))$;

(iii) $\mathcal{E}[\![(\mathbf{if}\ b\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2)]\!](\rho, \varphi) = \mathit{Cond}(\mathcal{B}[\![b]\!](\rho, \varphi), \mathcal{E}[\![e_1]\!](\rho, \varphi), \mathcal{E}[\![e_2]\!](\rho, \varphi))$;

(iv) $\mathcal{E}[\![(\mathbf{let}\ x\ \mathbf{be}\ e_1\ \mathbf{in}\ e_2)]\!](\rho, \varphi) = \mathcal{E}[\![e_2]\!](\rho[\mathcal{E}[\![e_1]\!](\rho, \varphi)/x], \varphi)$
where $\cdot[\cdot/\cdot]: \mathbf{IEnv} \times \mathbf{V} \times \mathbf{Id} \rightarrow \mathbf{IEnv}$ is given by:

$$\rho[v/x][\![x']\!] = \begin{cases} v & (\text{if } x = x'), \\ \rho[\![x']\!] & (\text{if } x \neq x'); \end{cases}$$

$(\mathrm{v})_{\mathrm{n}}$ $\mathcal{E}[\![\mathbf{let}\ f_n(x_1, \ldots, x_n)\ \mathbf{be}\ e_1\ \mathbf{in}\ e_2]\!](\rho, \varphi) =$

$$\mathcal{E}[\![e_2]\!](\rho, \varphi[\lambda v_1 \in \mathbf{V}, \ldots, v_n \in \mathbf{V}.\mathcal{E}[\![e_1]\!](\rho[v_1/x_1]\cdots[v_n/x_n], \varphi)/f_n])$$

where $\cdot[\cdot/\cdot]: \mathbf{FEnv} \times (\mathbf{V}^n \rightarrow \mathbf{V}) \times \mathbf{Fun}_n \rightarrow \mathbf{FEnv}$ is given by:

$$(\varphi[g/f_n])_{n'}[\![f'_{n'}]\!] = \begin{cases} g & (\text{if } n' = n \text{ and } f'_{n'} = f_n), \\ \varphi_{n'}[\![f'_{n'}]\!] & (\text{otherwise}). \end{cases}$$

Both languages can be put together by giving **Act** by the little grammar: $a ::= (x := e)$, then giving $\mathbf{S}$ as $\mathbf{IEnv} \times \mathbf{FEnv}$ and giving $\mathcal{A}$ by:

$$\mathcal{A}[\![x := e]\!](\rho, \varphi) = (\rho[\mathcal{E}[\![e]\!](\rho, \varphi)/x], \varphi).$$

Notice the various methods we have used. First we pick out various sets either as being somehow given (parameters) or given by some syntactic means (and we are not worrying about details of syntax) or given by constructions such as Cartesian product, or powers, or sets of functions. Next we then pick out some denotational functions over these sets either as given or else as defined by ordinary mathematical expression (if you allow the typed λ-calculus). When we use more complicated objects than sets, we will have to consider these methods anew.

To see that sets are not enough and that instead we need "partial" or "incomplete" entities as well as the usual "total" ones, suppose we added a recursive declaration facility to *Dec* as for example:

$$e ::= (\mathbf{letrec}\ f(x)\ \mathbf{be}\ e\ \mathbf{in}\ e).$$

So for example if $\mathbf{V} = \mathbf{N}$, the set of integers, and **Op**, includes the successor function, $(\cdot + 1)$ then the intention of the expression $\mathbf{letrec}\ f(x)\ \mathbf{be}\ f(x) + 1\ \mathbf{in}\ f(0)$ is to declare f to be a function $g: \mathbf{N} \rightarrow \mathbf{N}$ such that $g(n) = g(n) + 1$ for all integers n and then to give as result $g(0)$. (We feel free to use infix and other notations rather than the strict function applications considered above.) However there is no such function as g and computational intuition suggests that the expression does not terminate and indeed after such a declaration any expression of the form $f(e)$ will not terminate. So we expect that the value of the expression is undefined and we will have to use partial functions on the integers.

Again iteration will, like recursion, introduce non-termination. Suppose we extend *Imp* by adding the following clause to the definition of **Stat**:

$$s ::= (\mathbf{while}\ b\ \mathbf{do}\ n).$$

Then we expect the meaning of $\mathbf{while}\ b\ \mathbf{do}\ s$ to be the same as that of

$$\mathbf{if}\ b\ \mathbf{then}\ s; (\mathbf{while}\ b\ \mathbf{do}\ s)\ \mathbf{else}\ \mathbf{dummy}.$$

So if this meaning is θ and that of b is $p: \mathbf{S} \rightarrow \mathbf{T}$ and that of s is $\theta': \mathbf{S} \rightarrow \mathbf{S}$ we expect that:

$$\theta(\sigma) = \mathit{Cond}(p(\sigma), \theta(\theta'(\sigma)), \sigma) \tag{1}$$

and if, for example b is **true**, the constantly true Boolean expression, and s is **dummy** then we have:

$$\theta(\sigma) = \theta(\sigma)$$

and θ can be any function at all; but again computational intuition suggests it ought to be always undefined.

As is well-known, Scott introduced an element, $\perp$, into the value domain, $\mathbf{V}$, rather than worrying about partial functions and undefined results: to coin a phrase, he objectifies (some of) the metalanguage. This is particularly appropriate as it is possible for a subexpression to be undefined but for the whole expression to have a value, as in:

$$e = (\textbf{letrec } f(x) \textbf{ be } f(x)+1 \textbf{ in } (\textbf{let } g(x) \textbf{ be } 0 \textbf{ in } g(f(0))))$$

which under Algol's copy rule (call-by-name) has value 0 although $f(0)$ would have been undefined. So we want to consider partiality in the domain as well as the codomain. So in the integer example we will use $\mathbf{N}_\perp$ ($\stackrel{\text{def}}{=} \mathbf{N} \cup \{\perp\}$) and define $\perp + 1 = \perp$ in accord with computational intuition and then the equation $g(n) = g(n) + 1$ has the (unique) solution $g(n) = \perp$. But having introduced $\perp$ we cannot allow arbitrary functions $w\colon \mathbf{N}_\perp \to \mathbf{N}_\perp$ as then we would just get the same problems over again. Once more, computational intuition comes to the rescue and we expect that if $w(\perp) \neq \perp$ then $w(\perp) = w(n)$ for all integers, n. This can be put much better: since $\perp$ is undefined it is less defined than anything else and we introduce a *partial order*, $\sqsubseteq$, into $\mathbf{N}_\perp$ by:

$$x \sqsubseteq y \stackrel{\text{def}}{=} (x = \perp) \vee (x = y).$$

Then the condition on w is just that it is *monotonic* i.e. that if $x \sqsubseteq y$ then $w(x) \sqsubseteq w(y)$. Here is a picture of $\mathbf{N}_\perp$:

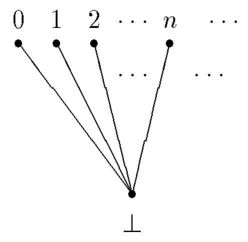


Figure I: The integer domain, $\mathbf{N}_\perp$.

Similarly we would introduce $\mathbf{S}_\perp$ and then expect that $\theta(\sigma) = \perp$, always; we also introduce $\mathbf{T}_\perp$ which looks like:

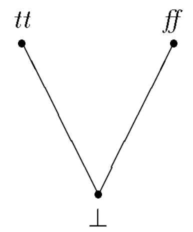


Figure II: The truth-value domain, $\mathbf{T}_\perp$.

These simple partial orders are called *flat* or *discrete* being characterised by the fact that if $x \sqsubseteq y \sqsubseteq z$ then $x = y$ or $y = z$. More complicated ones arise if we consider functions of several arguments such as $\mathit{Cond}\colon \mathbf{T}_\perp \times \mathbf{S}_\perp \times \mathbf{S}_\perp \to \mathbf{S}_\perp$. Again computational experience suggests that Cond is to be defined by: $\mathit{Cond}(\perp, \sigma, \sigma') = \perp$, $\mathit{Cond}(\mathit{tt}, \sigma, \sigma') = \sigma$ and $\mathit{Cond}(\mathit{ff}, \sigma, \sigma') = \sigma'$. So we expect that Cond — and all other computable functions of several arguments — will be monotonic in each of their arguments. We can rephrase this by defining the partial order on the Cartesian product to be the coordinatewise one:

$$\langle t, \sigma_0, \sigma_1 \rangle \sqsubseteq \langle t', \sigma_0', \sigma_1' \rangle \stackrel{\text{def}}{=} (t \sqsubseteq t') \wedge (\sigma_0 \sqsubseteq \sigma_0') \wedge (\sigma_1 \sqsubseteq \sigma_1')$$

and then noting that monotonicity in each argument is exactly monotonicity on the newly defined partial order. Here is an "aerial" picture of $\mathbf{T}_\perp \times \mathbf{T}_\perp$ which shows that the orders are no longer flat:

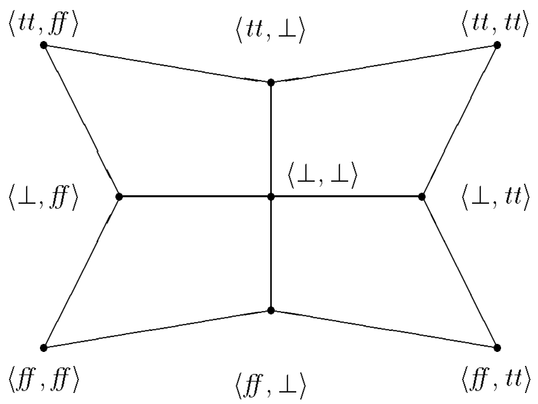


Figure III.

Thus we are led to the general idea of considering our domains to be partial orders rather than sets: the relation $x \sqsubseteq y$ means that the abstract object, x, viewed as the result of a computation (or more generally when involved in one) contains less information than y does. This general idea makes it reasonably clear that $\sqsubseteq$ is a partial order: reflexivity and transitivity are obvious and the point of antisymmetry is that if two elements contain exactly the same amount of information then they have the same uses and cannot be told apart — so we identify them. (It does not make much difference to the theory if we drop antisymmetry.) What is more there is always a least element, $\perp$, which corresponds to no information at all. So we have the following thesis (axiom) for all domains where we will take our denotations to lie:

Thesis 1. *All domains are partial orders with a least element.*

Our experience with functions now suggests a thesis about the computable functions over domains:

Thesis 2. *All computable functions are monotonic.*

This is the reasonable requirement that the amount of information in the output of a function grows as we increase the amount of information in the input.

Of course the above discussion is not a knockdown argument and many other structures than partial orders might fit the bill just as well. Let us look for further examples. Returning to equation (1) we see that it can be rewritten as:

$$\theta = F(\theta)$$

where the second-order functional, $F: (\mathbf{S}_\perp \rightarrow_{\mathrm{M}} \mathbf{S}_\perp) \rightarrow (\mathbf{S}_\perp \rightarrow_{\mathrm{M}} \mathbf{S}_\perp)$ is defined by:

$$F(\theta)(\sigma) = \mathit{Cond}(p(\sigma), \theta(\theta'(\sigma)), \sigma).$$

Here $\mathbf{S}_\perp \rightarrow_{\mathrm{M}} \mathbf{S}_\perp$ is the set of all monotonic functions from $\mathbf{S}_\perp$ to $\mathbf{S}_\perp$. Similarly recursive function declarations can be read as needing the fixed-point of a second-order functional, of a slightly different type. Now arbitrary functionals do not always have fixed-points and we return again to our computational intuitions. If $F(\theta)(\sigma)$ is defined ($\sigma \in \mathbf{S}$) then in the corresponding computation finitely many (one!) values of θ will be needed (namely the value at $\theta'(\sigma)$). So if we replace θ by another function $\overline{\theta}$ which always returns a result with more information, for any given argument, $F(\overline{\theta})(\sigma)$ will be defined too, and have the same value. This observation is just that F is monotonic, if we define the natural partial order on $\mathbf{S} \rightarrow_{\mathrm{M}} \mathbf{S}$ by:

$$\theta \sqsubseteq \overline{\theta} \equiv \forall \sigma \in \mathbf{S}_\perp . \theta(\sigma) \sqsubseteq \overline{\theta}(\sigma).$$

So we have another example, $\mathbf{S}_\perp \rightarrow_{\mathrm{M}} \mathbf{S}_\perp$ of a partial order and another example, $F: (\mathbf{S}_\perp \rightarrow_{\mathrm{M}} \mathbf{S}_\perp) \rightarrow_{\mathrm{M}} (\mathbf{S}_\perp \rightarrow_{\mathrm{M}} \mathbf{S}_\perp)$ of a monotonic function. However we have now hit a second obstacle: not all monotonic functions have fixed-points. The reader is invited to follow out the corresponding considerations on recursive function declarations where he will encounter exactly the same difficulty.

Now while $\mathbf{S}_\bot$ is a simple partial order, $(\mathbf{S}_\bot \to_{\mathrm{M}} \mathbf{S}_\bot)$ is not, as it contains nontrivial countably infinite increasing chains if $\mathbf{S}$ is infinite. All these chains have least upper bounds (lubs) defined pointwise by: $(\bigsqcup_n \theta_n)(\sigma) = \bigsqcup_n \theta_n(\sigma)$. Furthermore F preserves the lubs of such increasing chains. Let $\langle \theta_n \rangle_{n \in \omega}$ be an increasing chain in $(\mathbf{S}_\bot \to_{\mathrm{M}} \mathbf{S}_\bot)$. Then $F(\bigsqcup_n \theta_n)(\sigma) = \sigma'$ where σ and σ' are in $\mathbf{S}$. Now the value of $F(\bigsqcup_n \theta_n)(\sigma)$ depends, as remarked above, on finitely many values of $(\bigsqcup_n \theta_n)$ which therefore must also be given by one of the θ_n and then $F(\theta_n)(\sigma) = \sigma'$ for that θ_n showing $(\bigsqcup_n F(\theta_n))(\sigma) = \sigma'$. Thus we have proved the reverse inclusion (the other cases are trivial), albeit using a rather informal argument.

These facts make it easy to show that F has a fixed-point. For the sequence $\langle F^n(\bot) \rangle_{n \in \omega}$ is increasing as is easily shown by induction using the monotonicity of F. Let its lub be fix_F. Then:

$$F(\mathit{fix}_F) = F(\bigsqcup_n F^n(\bot)) = \bigsqcup_n F^{n+1}(\bot) = \mathit{fix}_F.$$

As is well-known, fix_F is the *least* fixed-point of F for if θ is any other one one easily shows by induction that $F^n(\bot) \sqsubseteq \theta$. Intuitively this minimality corresponds to the computational situation where we only expect θ to be defined when that can be proved — forced to be true by virtue of the computation rules. Indeed let us consider the iterates, $F^n(\bot)$. Writing them using the definition of F we find:

$$\begin{aligned}
&F^0(\bot)(\sigma) = \bot, \\
&F^1(\bot)(\sigma) = \mathit{Cond}(p(\sigma), \bot, \sigma), \\
&F^2(\bot)(\sigma) = \mathit{Cond}(p(\sigma), \mathit{Cond}(p(\theta'(\sigma)), \bot, \theta'(\sigma)), \sigma), \\
&F^3(\bot)(\sigma) = \mathit{Cond}(p(\sigma), \mathit{Cond}(p(\theta'(\sigma)), \mathit{Cond}(p(\theta'(\theta'(\sigma))), \bot, \theta'(\theta'(\sigma))), \theta'(\sigma)), \sigma).
\end{aligned}$$

So $F^n(\bot)$ corresponds to going around the loop up to n times at most and so it does seem that fix_F is the right fixed-point to choose. Essentially the same considerations apply to recursive declaration of functions.

All this suggests that we ought to consider domains which are partial orders with a least element which have lubs of increasing ω-chains; these partial orders are called *complete partial orders* (cpos):

Thesis 3. *Every domain is a cpo.*

What is more we ought to consider only these monotonic functions between cpos which preserve the lubs of increasing ω-chains; these functions are called *continuous* functions:

Thesis 4. *Every computable function is continuous.*

It turns out that we can do a great deal just with these two simple notions of cpos and of the continuous functions between them. We study them both together as a *category*:

Definition 1. The category, $\mathcal{CPO}$ has as objects the cpos and as morphisms the continuous functions between them. Composition of morphisms is ordinary function composition.

One easily checks that the identity function, $id_{\mathbf{D}}: \mathbf{D} \to \mathbf{D}$ is continuous and that the composition $\mathbf{D} \xrightarrow{f} \mathbf{E} \xrightarrow{g} \mathbf{F}$ of two continuous functions is continuous, showing that $\mathcal{CPO}$ is indeed a category.

Note that all flat partial orders are cpos (as is any partial order, such as $\mathbf{T}_\bot \times \mathbf{T}_\bot$, with a least element which only has eventually constant increasing ω-chains); further all monotonic functions over such partial orders are continuous.

The above considerations on fixed-point are quite general. Let $\mathbf{D} \xrightarrow{f} \mathbf{D}$ be a continuous function. A *prefixed-point* of f is an element d of $\mathbf{D}$ such that $f(d) \sqsubseteq d$.

Theorem 1 The Fixed-point Theorem. *Let* $\mathbf{D}$ *be a cpo and let* $f: \mathbf{D} \to \mathbf{D}$ *be a continuous function. Then* $\mathit{fix}_f \stackrel{\text{def}}{=} \bigsqcup_n f^n(\bot_{\mathbf{D}})$ *is the least prefixed-point of* f.

Proof. This is a straightforward generalisation of the particular case considered in the text and is left to the reader. ∎

The applications of this theorem are so widespread and important in the present approach to the denotational semantics of programming languages, that it is sometimes called the fixed-point approach. Let us now consider cpos and continuous functions at the general level of information content, before giving further examples. In a cpo $\mathbf{D}$, the lub, $\bigsqcup_{\mathbf{D}} d_n$ of an increasing chain, $\langle d_n \rangle_{n \in \omega}$ of elements of $\mathbf{D}$ is just all the information in the d_n. To make it feasible that such lubs exist we consider a certain idea of finite information which is implicit in some of the above discussion, and which will be studied more thoroughly in the chapter on algebraicity.

We suppose that some elements in $\mathbf{D}$ are finite, that is, contain only a finite amount of information, while the others are infinite. For example in this sense every element of $\mathbf{N}_\perp$ (or any other flat cpo) is finite while the finite elements of, say, $\mathbf{N}_\perp \rightarrow_{\mathrm{M}} \mathbf{T}_\perp$ are those monotonic functions which have value $\perp$ almost everywhere. Now consider the following axioms on finiteness for a partial order, $\mathbf{D}$, with least element $\perp$.

1. Let d be finite and suppose $\langle x_n \rangle$ is an increasing ω-chain whose lub, $\bigsqcup_{\mathbf{D}} x_n$, exists. Then if $d \sqsubseteq \bigsqcup_{\mathbf{D}} x_n$ it follows that for some n, $d \sqsubseteq x_n$.
2. Each infinite element, x, is the lub of an increasing ω-chain of finite elements.
3. Every increasing ω-chain of finite elements has a lub in $\mathbf{D}$.

We believe Axioms 1 and 2 are evident, Axiom 3 says that all "potential" infinities are allowed. It is a consequence of these axioms that $\mathbf{D}$ is a cpo. For let $\langle x_n \rangle$ be an increasing ω-chain in $\mathbf{D}$ and suppose that, by Axiom 2, for every n, $\langle e_{nm} \rangle_{m \in \omega}$ is an increasing chain of finite elements of $\mathbf{D}$ with lub x_n. By Axiom 1 for each e_{nm} and $n' \geq n$, as $e_{nm} \sqsubseteq x_n \sqsubseteq x_{n'} = \bigsqcup_{m'} e_{n'm'}$, there is an $e_{n'm'}$ so that $e_{nm} \sqsubseteq e_{n'm'}$. It follows that we can find an increasing chain $\langle e_n \rangle$ of finite elements such that each e_n is some e_{nm}, e_0 is e_{00}, and for all $n' < n$, $m' < n$, $e_{n'm'} \sqsubseteq e_n$. Then by Axiom 3, $\bigsqcup_{\mathbf{D}} e_n$ exists and is easily seen to be $\bigsqcup_{\mathbf{D}} x_n$.

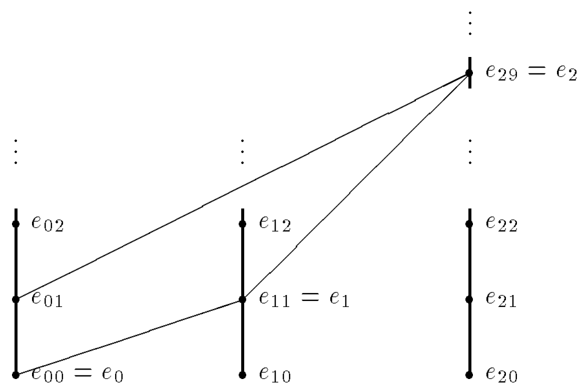


Figure IV: Construction of the chain $\langle e_n \rangle$.

Similarly we can give a general version of the above argument for the continuity of the second-order functional, F. Suppose $\mathbf{D}$ and $\mathbf{E}$ obey the above axioms on finiteness and $f: \mathbf{D} \rightarrow \mathbf{E}$ is monotonic. Then f is continuous iff wherever $e \sqsubseteq f(x)$ and e is finite there is a finite d such that $d \sqsubseteq x$ and $e \sqsubseteq f(d)$. What this condition says is that any finite amount of information in the output only depends on a finite amount of information in the input. Suppose the condition holds and $\langle d_n \rangle$ is an increasing chain of elements in $\mathbf{D}$. Then by monotonicity $f(\bigsqcup_n d_n) \sqsupseteq \bigsqcup_n f(d_n)$. Now if $e \sqsubseteq f(\bigsqcup_n d_n)$ where e is finite, it follows by the condition that there is a finite d in $\mathbf{D}$ such that $d \sqsubseteq \bigsqcup_n d_n$ and $e \sqsubseteq f(d)$. But then by Axiom 1, $d \sqsubseteq d_n$, for some n and so $e \sqsubseteq \bigsqcup_n f(d_n)$. Now by Axiom 2 $f(\bigsqcup_n d_n) = \bigsqcup_n e_n$, where $\langle e_n \rangle$ is an increasing ω-chain of finite elements of $\mathbf{E}$. So by the above reasoning $e_n \sqsubseteq \bigsqcup_n f(d_n)$ for each n and so $f(\bigsqcup_n d_n) \sqsubseteq \bigsqcup_n f(d_n)$ which completes the proof that f is continuous. Conversely suppose f is continuous and $e \sqsubseteq f(x)$ where $\langle d_n \rangle$ is an increasing ω-chain of finite elements of $\mathbf{D}$ and so $e \sqsubseteq f(x) = f(\bigsqcup_n d_n) = \bigsqcup_n f(d_n)$ (by continuity) and so, by Axiom 1, $e \sqsubseteq f(d_n)$ for some n, which establishes the condition. (Note that Axiom 3 was not used in this argument.)

We would like to suggest another thesis on continuous functions which refines Thesis 2. There are many monotonic but noncomputable functions in, for example, $\mathbf{N}_\perp \rightarrow_{\mathrm{M}} \mathbf{N}_\perp$, but they are all computable by infinite programs, or more generally by machines with infinitely many internal states. Of course such computation is not effective or even mechanical in the usual sense of a finite machine. Nonetheless it is physically feasible. And indeed at higher types continuity is a real restriction. Again with the above characterisation of continuous functions we could imagine an infinite machine which stored (codes for) all the pairs (d, e) of finite elements such that $e \sqsubseteq f(d)$ used this store to give out the information in each e whenever it had received all the information in the corresponding d. Of course, this depends on the input and output domains being themselves physically feasible (and we would dearly like to be able to put down the right axioms for that notion), but let us state a thesis for such domains:

Thesis 5. *A monotonic function is continuous iff it is physically feasible.*

This implies Thesis 4 as surely every computable function is physically feasible. Since the thesis is an equivalence it may be easier to give more convincing arguments for its truth than Thesis 4, in some cases at least.

One kind of information is to do with (occurrences of) events: namely the information that those events occurred. For example in the case of $\mathbf{N}_\perp$, $\perp$ might mean that no event occurred and an integer n, might mean that the event occurred of the integer n being output (or, in another circumstance being input). For an example with more structure, suppose we have a Turing machine (TM) with an input tape and an output tape. The tape squares can be blank, or, after printing, contain a 0 or a 1. The events that can occur are the printing of a 0 or a 1 on the (next) output tape square by the TM or the inputting (by some unspecified agency) of a 0 or a 1 on the (next) input tape square. Thus the output domain is just the collection of all possible sets of events that could occur, ordered by the subset ordering; equivalently this is the set of all finite or infinite binary sequences with the subsequence ordering. For example 0111 means that 0 was printed on the first tape square, and one on the second, third and fourth tape squares, and thereafter there was no more output. The same (isomorphic) domain is associated with the input. This cpo is called **Tapes**, not unnaturally, and the reader will see that the function $f\colon \mathbf{Tapes} \to \mathbf{Tapes}$ computed by the TM must be continuous.

By taking a different view of what constitutes an output event, we can obtain a different cpo for the same physical situation. Let us agree that outputting a 0 followed by n 1's followed by a 0 constitutes an outputting of the integer n (and so the same event may occur several times). Then the appropriate cpo is $\mathbf{P}\boldsymbol{\omega}$, the set of all subsets of ω $(= \mathbf{N})$ with the subset ordering. It must be admitted that what can legitimately be called an event is not a very clear notion, although there must be some rules. For example it does not seem correct to allow as an event "the printing of infinitely many 1's". Again if we regard the input tape also as specifying an element of $\mathbf{P}\boldsymbol{\omega}$, the TM may not compute a function from $\mathbf{P}\boldsymbol{\omega}$ to $\mathbf{P}\boldsymbol{\omega}$ at all — we should insist either that the input has a special form or that the TM has a certain kind of behaviour.

Just as more than one domain may be appropriate when describing a physical situation, so the same domain may be used to describe different situations. For example suppose we have infinitely many lights and a computational mechanism which is switching them on (and never off). Then "the switching on of the nth light" would be a bona fide event and the appropriate cpo would then be $\mathbf{P}\boldsymbol{\omega}$ again.

In all those last examples we have not considered any problem of giving a denotational semantics; however the above domains would clearly arise if we wanted to consider the semantics of languages for outputting or inputting on tapes or for switching on infinite arrays of lights.

In terms of events a finite amount of information should just be a finite set of events. Then if $\bigsqcup_n x_n = \bigcup_n x_n$ (as seems reasonable) the above axioms seem reasonable enough and we also believe that it is reasonably self-evident that any physically feasible function must be monotonic and obey the condition proved equivalent to continuity as an output event could hardly depend on infinitely many input events as a machine should only be able to do a finite amount of computation before causing a given output event. However further analysis of events is required to show that every continuous function is feasible. But in particular cases, such as the above interpretation of **Tapes**, this is not difficult.

In the above we have argued for the use of the category, $\mathcal{CPO}$ of cpos and continuous functions, rather than the category of sets and (ordinary) functions. We have come across a few examples of cpos. Each set, $\mathbf{X}$, gives us a flat cpo, $\mathbf{X}_\perp = \mathbf{X} \cup \{\perp\}$ ordered by: $x \sqsubseteq y$ iff $x = \perp$ or $x = y$. Particular examples are given in Figures I, II above and we should also mention $\mathbf{U} \stackrel{\text{def}}{=} \emptyset_\perp$ and $\mathbf{O} \stackrel{\text{def}}{=} \{\top\}_\perp$. We have considered such continuous functions as $(+1)\colon \mathbf{N}_\perp \to \mathbf{N}_\perp$ and should also mention $(-1)\colon \mathbf{N}_\perp \to \mathbf{N}_\perp$ and $Z\colon \mathbf{N}_\perp \to \mathbf{T}_\perp$ where:

$$x - 1 = \begin{cases} n & (\exists n \in \mathbf{N}. x = n+1), \\ \perp & (\text{otherwise}); \end{cases}$$

$$Z(x) = \begin{cases} \mathit{tt} & (\text{if } x = 0), \\ \mathit{ff} & (\text{if } x \neq 0, \perp), \\ \perp & (\text{otherwise}). \end{cases}$$

We have also seen nonflat cpos such as **Tapes** and $\mathbf{P}\boldsymbol{\omega}$. Some basic continuous functions for these cpos are $(0 \cdot -), (1 \cdot -)$, $\mathit{tl}\colon \mathbf{Tapes} \to \mathbf{Tapes}$, $\mathit{hd}\colon \mathbf{Tapes} \to \mathbf{T}_\perp$, $(+1), (-1)\colon \mathbf{P}\boldsymbol{\omega} \to \mathbf{P}\boldsymbol{\omega}$ and $Z\colon \mathbf{P}\boldsymbol{\omega} \to \mathbf{T}_\perp$ (we don't mind introducing a small amount of ambiguity for names of functions), where $(0 \cdot -)$ and $(1 \cdot -)$ append 0 (respectively 1) to a tape and tl and hd are characterised by the equations, $\mathit{tl}(0 \cdot x) = \mathit{tl}(1 \cdot x) = x$, $\mathit{tl}(\perp) = \perp$, $\mathit{hd}(0 \cdot x) = \mathit{ff}$, $\mathit{hd}(1 \cdot x) = \mathit{tt}$, $\mathit{hd}(\perp) = \perp$ and the functions over $\mathbf{P}\boldsymbol{\omega}$ are given by:

$$x + 1 = \{n + 1 \mid n \in x\};$$
$$x - 1 = \{n \mid n + 1 \in x\};$$
$$Z(x) = \begin{cases} \mathit{tt} & (0 \in x), \\ \perp & (\text{otherwise}). \end{cases}$$

We will consider other cpos and functions of several arguments when we discuss the construction in $\mathcal{CPO}$.

Let us conclude with some discussion of some choices we have not made. First we have not used complete lattices, that is we have not assumed that every subset has a lub (although, of course, the lattices are included). For the presence of the top element, $\top$ $(= \bigsqcup_{\mathbf{D}} \mathbf{D})$ causes problems, leading to extra decisions when extending functions and cases when making proofs, in an unsystematic way. For example suppose we make **T** into a lattice by adding a top element, giving:

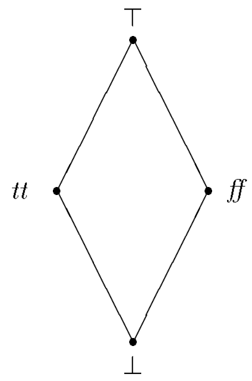


Figure V: The truth-value lattice.

Now, even apart from the fact that $\top$ has no computational interpretation, when using $\mathbf{T}_\perp$ in the cases discussed above, it is not clear how to extend the definition of $Cond: \mathbf{T}_\perp \times \mathbf{N}_\perp \times \mathbf{N}_\perp \to \mathbf{N}_\perp$ to a function on the lattices. By monotonicity we must make $\mathbf{N}_\perp$ into a lattice too, say by adding a top element, and then we must choose between a minimum and a maximum extension:

$$Cond(\top, x, y) = x \sqcup y$$

or

$$Cond(\top, x, y) = \top.$$

Either choice means that one of the following program equivalence will fail in general, although they are correct from the computational point of view:

$$(\textbf{if } b \textbf{ then } (\textbf{if } b \textbf{ then } e_0 \textbf{ else } e_1) \textbf{ else } (\textbf{if } b \textbf{ then } e_2 \textbf{ else } e_3)) \equiv (\textbf{if } b \textbf{ then } e_0 \textbf{ else } e_3),$$
$$(\textbf{if } b_0 \textbf{ then } (\textbf{if } b_1 \textbf{ then } e_0 \textbf{ else } e_1) \textbf{ else } (\textbf{if } b_1 \textbf{ then } e_2 \textbf{ else } e_3)) \equiv$$
$$(\textbf{if } b_1 \textbf{ then } (\textbf{if } b_0 \textbf{ then } e_0 \textbf{ else } e_2) \textbf{ else } (\textbf{if } b_0 \textbf{ then } e_1 \textbf{ else } e_3)).$$

One often sees another definition of cpo in which it is assumed that every directed set has a lub, where a set X is *directed* iff it is nonempty and any two elements have an upper bound in X. Let us call such partial orders *dcpos* to distinguish them from ours. We decided to make the choice of assuming only lubs of increasing ω-chains because lubs of general directed sets are not so easily justified from a computational point of view and because it is a weaker assumption. But we shall see that all the cpos we are interested in to turn out to be dcpos, so this choice is of little importance in the end.

As to functions, we have not assumed that our functions are strict (preserve the least element). For then all least fixed-points would be $\perp$, which is certainly wrong. But on the other hand from the categorical point of view the strict continuous functions are important as they preserve all the cpo structure. So we will often consider the subcategory $\mathcal{CPO}_\perp$ of all strict continuous functions and write $f: \mathbf{D} \to_\perp \mathbf{E}$ to mean f is strict and continuous. Other axioms and kinds of functions will be considered later.

Exercises

1. Add a syntax for **BExp** to the specification of *Dec*, and extend the semantics accordingly.

2. In the semantics given above for nonrecursive function declarations, the free variables (other than the parameters) in the body of the function declaration receive their values from the "definition-time" environment. Can you change the semantics so that this value is taken from the "call-time" environment instead?

3. Suppose we add a class **Proc** of (parameterless) procedure variables to the syntax of *Imp* and add procedure declaration and calling facilities to **Stat** by including the extra clauses:

$$s ::= (\mathbf{proc}\ p\ \mathbf{begin}\ s\ \mathbf{end}) \mid (\mathbf{call}\ p).$$

What difficulties do you discover when trying to do the denotational semantics?

4. Add some facilities to *Imp* to allow reading a boolean value from an input channel and outputting one to an output channel. What happens with the denotational semantics? (Repeated input requests and outputs are allowed.)

5. Let $\mathbf{S}$ be a set, $p\colon \mathbf{S} \to \mathbf{T}$ be a predicate, $\theta'\colon \mathbf{S} \to \mathbf{S}$ be a command as above. What commands, θ, satisfy the fixed-point equation, $\theta(\sigma) = \mathit{Cond}(p(\sigma), \theta \circ \theta'(\sigma), \sigma)$.

6. Write Turing machines to compute the functions $(0 \cdot -)$, $(1 \cdot -)$, $tl\colon \mathbf{Tapes} \to \mathbf{Tapes}$, $hd\colon \mathbf{Tapes} \to \mathbf{T}_\perp$, $(+1), (-1)\colon \mathbf{P}\boldsymbol{\omega} \to \mathbf{P}\boldsymbol{\omega}$ and $Z\colon \mathbf{P}\boldsymbol{\omega} \to \mathbf{T}_\perp$.

7. Let $\mathbf{D}$ be a cpo and let $f\colon \mathbf{D} \to \mathbf{D}$ be continuous. A *prefixed-point* (*fixed-point*, *postfixed-point*) of f is an element, d of $\mathbf{D}$ such that $fd \sqsubseteq d$ ($fd = d$, $fd \sqsupseteq d$). Show that the set of prefixed-points (respectively fixed-points, postfixed-points) of f form a cpo under the partial order inherited from $\mathbf{D}$.

8. Show that every monotonic function, $f\colon \mathbf{D} \to \mathbf{D}$ where $\mathbf{D}$ is a dcpo has a least fixed-point. Give an example of a monotonic function $f\colon \mathbf{D} \to \mathbf{D}$, where $\mathbf{D}$ is a cpo, which does not have any fixed-point.

9. The cpo, **Pred**, is the set of partial function $f\colon \mathbf{N} \to_{\mathrm{P}} \mathbf{T}$ ordered by:

$$f \sqsubseteq g \quad \text{iff} \quad \forall n \in \mathbf{N}. f(n) \text{ defined } \Rightarrow (g(n) \text{ defined and } f(n) = g(n));$$

the partial order, $(\mathbf{T}_\perp)^\omega$ is the Cartesian Product of denumerably many copies of $\mathbf{T}_\perp$; the partial order $\mathbf{N}_\perp \to_\perp \mathbf{T}_\perp$ is the subset of strict functions of $\mathbf{N}_\perp \to_{\mathrm{M}} \mathbf{T}_\perp$ with the inherited ordering. Show that all these cpos are isomorphic to the cpo,

$$\{\langle x, y\rangle \mid x \subseteq \mathbf{N},\ y \subseteq \mathbf{N},\ x \cap y = \emptyset\}$$

under the ordering: $\langle x, y\rangle \sqsubseteq \langle u, v\rangle$ iff $x \subseteq u$ and $y \subseteq v$.

10. Why is the function $\mathit{Tot}\colon \mathbb{O} \to \mathbf{T}$ not intuitively computable, where: $\mathit{Tot}(\top) = tt$, $\mathit{Tot}(\perp) = \mathit{ff}$? Find some other, similar examples.

11. Find an example of a monotonic but intuitively non-computable function from **Pred** (see Exercise 9) to $\mathbf{T}_\perp$. Show it is not continuous. Find some other, similar examples.

12. The function $\partial_{\mathbf{D}}\colon \mathbf{D} \to \mathbb{O}$ where $\mathbf{D}$ is a cpo defined by:

$$\partial_{\mathbf{D}}(d) = \begin{cases} \top & (d \neq \perp), \\ \perp & (d = \perp). \end{cases}$$

Show $\partial_{\mathbf{D}}$ is continuous. Give an argument to show that $\partial_{\mathbf{Tapes}}$ is intuitively computable for at least one interpretation of **Tapes**; do the same for $\partial_{\mathbf{Pred}}$. A function $f\colon \mathbf{D} \to \mathbf{E}$ where $\mathbf{D}$, $\mathbf{E}$ are partial orders is *multiplicative* iff whenever the glb, $x \sqcap y$ of two elements x and y of $\mathbf{D}$ exists then $f(x \sqcap y) = f(x) \sqcap f(y)$; it is *dual-continuous* iff whenever $\langle d_n\rangle_{n\in\omega}$ is a decreasing sequence in $\mathbf{D}$ with glb $\bigsqcap_{\mathbf{D}} x_n$ then $f(\bigsqcap_{\mathbf{D}} x_n) = \bigsqcap_{\mathbf{E}} f(x_n)$. Is $\partial_{\mathbf{D}}$ always multiplicative? Is it always dual-continuous?

13. A function is *additive* iff it preserves binary lubs (cf. Exercise 12). Give an example of a continuous, intuitively computable function between cpos which is not additive. Why is the functional $F\colon (\mathbf{S}_\perp \to \mathbf{S}_\perp) \to (\mathbf{S}_\perp \to \mathbf{S}_\perp)$ involved in the above discussion on while-loops additive?

14. A function $\overline{\mu}\colon \mathbf{Pred} \to \mathbf{N}$ is a *minimalisation operator* iff $\overline{\mu}(f) = n \neq \perp$ implies $f(n) = tt$ and $\forall m < n. f(m) \neq tt$. Show there is a greatest monotonic minimalisation operator and that is continuous. Is the case $\overline{\mu}\colon (\mathbf{N}_\perp \to_{\mathrm{M}} \mathbf{T}_\perp) \to \mathbf{N}_\perp$ much different?

15. A function $\overline{\exists}\colon (\mathbf{N}_\perp \to_{\mathrm{M}} \mathbf{T}_\perp) \to \mathbf{T}_\perp$ is an *existential quantifier* iff

$$\overline{\exists} f = tt \Rightarrow \text{ there is an element, } d \text{ of } \mathbf{N} \text{ s.t. } f(d) = tt,$$
$$\overline{\exists} f = \mathit{ff} \Rightarrow \text{ for all elements, } d \text{ of } \mathbf{N} \text{ s.t. } d \neq \perp, f(d) = \mathit{ff}.$$

What are the maximal monotonic and continuous existential quantifiers, respectively? How is the continuous one intuitively computable? What happens if you replace $(\mathbf{N}_\perp \to_{\mathrm{M}} \mathbf{T}_\perp)$ by **Pred**?

16. Draw the following partial orders: $\mathbb{O} \rightarrow_{\mathrm{M}} \mathbb{O}$, $\mathbb{O} \rightarrow_{\mathrm{M}} \mathbf{T}_\perp$, $\mathbf{T}_\perp \rightarrow_{\mathrm{M}} \mathbb{O}$, $\mathbf{T}_\perp \rightarrow_{\mathrm{M}} \mathbf{T}_\perp$, $(\mathbb{O} \rightarrow_{\mathrm{M}} \mathbb{O}) \rightarrow_{\mathrm{M}} (\mathbb{O} \rightarrow_{\mathrm{M}} \mathbb{O})$, $\mathbb{O} \times \mathbb{O}$, $(\mathbb{O} \times \mathbb{O}) \rightarrow_{\mathrm{M}} \mathbb{O}$, **Tapes**.

17. The definition of **Tapes** of tapes uses $\{0,1\}$ as a parameter. Define X^∞ similarly for any set X so that $\mathbf{Tapes} = \{0,1\}^\infty$ and $\boldsymbol{\omega} \stackrel{\text{def}}{=} \{0\}^\infty$ looks like:

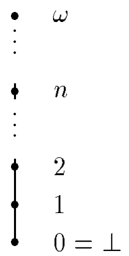


Figure VI: The cpo $\boldsymbol{\omega}$.

18. Define the partial order $\mathbf{R}$ (closed intervals on the real line) as the set $\{[\underline{x}, \overline{x}] \mid -\infty \leq \underline{x} \leq \overline{x} \leq +\infty\}$ where the intervals $[\underline{x}, \overline{x}] \stackrel{\text{def}}{=} \{y \in \mathbf{Reals} \mid \underline{x} \leq y \leq \overline{x}\}$ are ordered by reverse inclusion, i.e. $[\underline{x}, \overline{x}] \sqsubseteq [\underline{x}', \overline{x}']$ iff $\underline{x} \leq \underline{x}' \leq \overline{x}' \leq \overline{x}$. Show that $\mathbf{R}$ is a cpo. How would you use $\mathbf{R}$ when computing functions over reals?

19. Cantor's Theorem shows there is no nontrivial set X which is isomorphic to (in bijection with) its own function space, X^X. Show there is no nontrivial partial order, $\mathbf{D}$, such that $\mathbf{D} \cong \mathbf{D} \rightarrow_{\mathrm{M}} \mathbf{D}$.

This gives yet another argument for restricting attention to the continuous functions, as we will see that the answer for the corresponding question on cpos and continuous functions is positive. [*Hint* Consider $\mathbf{D} \rightarrow \mathbb{O}$ and the cardinalities of sets of incomparable elements.]

20. Show that every *denumerable* directed subset of a cpo has a least upper bound. Let $\mathbf{D}$, $\mathbf{E}$ be cpos. Show that $f\colon \mathbf{D} \rightarrow \mathbf{E}$ is continuous iff f preserves lubs of denumerable directed sets (i.e. iff for every denumerable directed subset, X, of $\mathbf{D}$, $f(X)$ is directed and $f(\bigsqcup_{\mathbf{D}} X) = \bigsqcup_{\mathbf{E}} f(X)$). Let $\mathbf{D}$ be a cpo and let $\langle d_{m,n}\rangle_{m,n \in \omega}$ be a double increasing sequence (i.e. $m \leq m'$, $n \leq n'$ implies $d_{m,n} \sqsubseteq d_{m',n'}$). Show that $\bigsqcup_m (\bigsqcup_n d_{m,n}) = \bigsqcup_{m,n} d_{m,n} = \bigsqcup_n d_{n,n}$.

21. Let $\mathbf{D}$ be a cpo. Show that a T_0-topology can be defined in $\mathbf{D}$ by taking a subset, O, to be open iff:

(1) Whenever x, y are elements of $\mathbf{D}$ such that $x \sqsubseteq y$ and $x \in O$ then $y \in O$.
(2) Whenever $\langle x_n\rangle_{n \in \omega}$ is an increasing sequence of elements of $\mathbf{D}$ s.t. $(\bigsqcup_n x_n) \in O$, then for some n, $x_n \in O$.

Show that the order can be recovered from the topology by: $x \sqsubseteq y$ iff $\forall$ open sets $O.(x \in O \Rightarrow y \in O)$. Show too that if $\mathbf{D}$, $\mathbf{E}$ are cpos then $f\colon \mathbf{D} \rightarrow \mathbf{E}$ is continuous iff it is topologically continuous with respect to the above topology. Show that this is the only assignment of topologies to cpos which makes topological continuity coincide with order continuity. [*Hint* Consider $\mathbb{O}$, $\boldsymbol{\omega}$ (Exercise 17) and then an arbitrary cpo.]

22. Let $\mathbf{D}$, $\mathbf{E}$ be dcpos. A function $f\colon \mathbf{D} \rightarrow \mathbf{E}$ is *(directed) continuous* iff for every directed subset, X, of $\mathbf{D}$, $f(X)$ is directed and $f(\bigsqcup_{\mathbf{D}} X) = \bigsqcup_{\mathbf{E}} f(X)$. Give a T_0-topology for dcpos so that this definition of continuity coincides with topological continuity. Give a topology for partial orders so that topological continuity coincides with monotonicity.

2. Products and Function Spaces

Now that we have an idea of what our domains are, we need some language (metalanguage) to denote elements of and continuous functions between domains. This will be a more or less normal mathematical language of terms, including a typed λ-calculus, and looks much like the language used to describe the set-theoretic semantics of *Imp* and *Dec* in Chapter 1. We will treat this language and its semantics in a semi-formal fashion, leaving the reader, if he cares, to provide a formal definition in set theory.

In fact we expand the language as we go along, beginning with:

E1. There are constants, $a, \ldots$ denoting elements of cpos. Their types and what they denote are given when they are introduced. For example, $\perp_{\mathbf{D}}$ denotes the least element of $\mathbf{D}$ (and we just write $\perp$ when $\mathbf{D}$ is understood from the context). We also use *tt*, *ff* for elements of $\mathbf{T}_\perp$, numerals for elements of $\mathbf{N}_\perp$, $\top_{\mathbf{D}}$ for the top element of $\mathbf{D}$, if it exists, all as in Chapter 1.

E2. There are variables, $x, \ldots$ denoting elements of cpos. Their types and what they denote are determined by the context of their use.

E3. There are constants, $id_{\mathbf{D}}$, $(+1)$, (-1), Z, $\ldots$ denoting morphisms, that are continuous functions between cpos. Their types and what they denote are given when they are introduced. Again we have introduced a few of these in Chapter 1. Again we tolerate a certain amount of ambiguity — as long as this can be disambiguated by the context; for example for each constant a, denoting an element of a domain $\mathbf{E}$, we have the constant K_a denoting the function $f\colon \mathbf{D} \to \mathbf{E}$ where $f(d) = a$, and $\mathbf{D}$ is to be understood from the context.

E4. If α, β are expressions denoting morphisms of types $\mathbf{A} \to \mathbf{B}$, $\mathbf{B} \to \mathbf{C}$ respectively then $(\beta \circ \alpha)$ is an expression of type $\mathbf{A} \to \mathbf{C}$ which denotes the composition of β and α. Because function composition is associative we can always omit brackets.

E5. If α is an expression denoting a morphism $f\colon \mathbf{A} \to \mathbf{B}$ and e is an expression denoting an element, a of $\mathbf{A}$, then $\alpha(e)$ denotes $f(a)$. Actually we won't stick to this rigid applicative syntax, but will also infix and postfix operators and so on.

We will write $e\colon \mathbf{D}$ to mean that the expression e has type $\mathbf{D}$ and also $\alpha\colon \mathbf{D} \to \mathbf{E}$ to mean that the expression α has type $\mathbf{D} \to \mathbf{E}$. Note that there is a correspondence between expressions denoting elements and expressions denoting morphisms. If e is an expression of type $\mathbf{E}$ with one free variable, x, of type $\mathbf{D}$ then we can define a function, f, from $\mathbf{D}$ to $\mathbf{E}$ by: $f(x) = e$. This is continuous as can be shown by a little inductive proof to the effect that each such function can be denoted by an expression: a variable x of type $\mathbf{D}$ gives the function $id_{\mathbf{D}}$; the expression $\alpha(e)$ gives the function $\alpha \circ \beta$ where e gives β.

What we need next are functions of several variables and they go hand in hand with products in $\mathcal{CPO}$. Let $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ be cpos. Their *Cartesian product*, $\prod_{i<n} \mathbf{D}_i$ is the set $\{\langle x_0, \ldots, x_{n-1}\rangle \mid \forall i < n. x_i \in \mathbf{D}_i\}$ of n-tuples ordered coordinatewise by:

$$\langle x_0, \ldots, x_{n-1}\rangle \sqsubseteq \langle x'_0, \ldots, x'_{n-1}\rangle \equiv \forall i < n. x_i \sqsubseteq_{\mathbf{D}_i} x'_i.$$

(We have already seen an example of this construction in Chapter 1, namely $\mathbf{T}_\perp \times \mathbf{S}_\perp \times \mathbf{S}_\perp$.) Sometime we write $\mathbf{D}_0 \times \cdots \times \mathbf{D}_n$ for $\prod_{i<n} \mathbf{D}_i$ or $\mathbf{D}^n$ when the $\mathbf{D}_i$ are all the same. Note that the Cartesian product is a cpo with the least element, $\langle \perp_{\mathbf{D}_0}, \ldots, \perp_{\mathbf{D}_{n-1}}\rangle$ and with lubs of increasing sequence $\langle x^{(m)}\rangle_{m\in\omega}$ determined coordinatewise as: $\bigsqcup_m x^{(m)} = \langle \bigsqcup_{m\ \mathbf{D}_0} x_0^{(m)}, \ldots \bigsqcup_{m\ \mathbf{D}_{n-1}} x_{n-1}^{(m)}\rangle$, (and in fact all lubs are determined coordinatewise).

This product is also the normal categorical product. We have the natural projection maps,

$$\pi_i\colon (\prod_{i<n} \mathbf{D}_i) \to \mathbf{D}_i \qquad (i < n)$$

where $\pi_i(x) = x_i$ — and these are continuous as can be seen from the remark on lubs. Sometimes $\pi_i(x)$ is written as $(x \downarrow i)$ or even $(x)_i$. The projection maps have the universal property that if $f_i\colon \mathbf{F} \to \mathbf{D}_i$ $(i < n)$ are n continuous maps then there is a unique (mediating) morphism, $h\colon \mathbf{F} \to (\prod_{i<n} \mathbf{D}_i)$ which makes all the following diagrams commute:

$$\begin{tikzcd} \mathbf{F} \arrow[r, "h"] \arrow[dr, "f_i"'] & \prod_{i<n} \mathbf{D}_i \arrow[d, "\pi_i"] \\ & \mathbf{D}_i \end{tikzcd} \qquad (i < n)$$

To see this, suppose h is such a morphism. Then for any element, x, of F:

$$\begin{aligned} h(x) &= \langle h(x)_0, \ldots, h(x)_{n-1} \rangle \\ &= \langle \pi_0 \circ h(x), \ldots, \pi_{n-1} \circ h(x) \rangle \\ &= \langle f_0(x), \ldots, f_{n-1}(x) \rangle, \end{aligned}$$

showing uniqueness. Conversely we can use the formula to define h, proving continuity by:

$$\begin{aligned} h(\bigsqcup_m x_m) &= \langle f_0(\bigsqcup_m x_m), \ldots, f_{n-1}(\bigsqcup_m x_m) \rangle \\ &= \langle \bigsqcup_m f_0(x_m), \ldots, \bigsqcup_m f_{n-1}(x_m) \rangle && \text{the } f_i \text{ are continuous} \\ &= \bigsqcup_m \langle f_0(x_m), \ldots, f_{n-1}(x_m) \rangle && \text{lubs are determined coordinatewise} \\ &= \bigsqcup_m h(x_m) && \text{by definition of } h. \end{aligned}$$

We write h as $\langle f_0, \ldots, f_{n-1} \rangle$ and the double use of pointed brackets should not cause any confusion. Note the particular case, $n = 0$, when $\prod_{i<0} \mathbf{D}_i$ is just $\mathbf{U}$ and we see that for any cpo, $\mathbf{F}$, there is a unique morphism $\mathbf{F} \rightarrow \mathbf{U}$, showing that $\mathbf{U}$ is the *final* object in $\mathcal{CPO}$. Note, too, that as usual the universal property determines $\prod_{i<n} \mathbf{D}_i$ up to isomorphism.

We can now expand our stock of expressions:

E6. If $e_0, \ldots, e_{n-1}$ are expressions of types $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ respectively, then $\langle e_0, \ldots, e_{n-1} \rangle$ is an expression of type $\mathbf{D}_0 \times \cdots \times \mathbf{D}_{n-1}$ and denotes the n-tuple of the values denoted by the $e_0, \ldots, e_{n-1}$, taken in that order. When $\alpha: (\mathbf{D}_0 \times \cdots \times \mathbf{D}_{n-1}) \rightarrow \mathbf{E}$ we will often write $\alpha(e_0, \ldots, e_{n-1})$ instead of the more pedantic $\alpha(\langle e_0, \ldots, e_{n-1} \rangle)$; this is just the usual notation for functions of several arguments.

E7. If $\alpha_0, \ldots, \alpha_{n-1}$ are expressions of types $(\mathbf{F} \rightarrow \mathbf{D}_0), \ldots, (\mathbf{F} \rightarrow \mathbf{D}_{n-1})$ respectively then $\langle \alpha_0, \ldots, \alpha_{n-1} \rangle$ is an expression of type $\mathbf{F} \rightarrow (\prod_{i<n} \mathbf{D}_i)$, and it denotes the evident mediating morphism, as described above.

One reason for interest in the categorical properties of the Cartesian product is that they serve as a guide for the right axioms, which we can give in terms of either the morphisms or the elements. We call the first kind of axiomatisation *external* and the second *internal*. Unfortunately, we will not have time to pursue questions of proof much further than informally stating a few axioms.

External Axiomatisation.

1. $\forall f_0: \mathbf{F} \rightarrow \mathbf{D}_0. \ldots \forall f_{n-1}: \mathbf{F} \rightarrow \mathbf{D}_{n-1}. \forall i < n. \pi_i \circ \langle f_0, \ldots, f_{n-1} \rangle = f_i$.
2. $\forall h: \mathbf{F} \rightarrow (\prod_{i<n} \mathbf{D}_i). h = \langle \pi_0 \circ h, \ldots, \pi_{n-1} \circ h \rangle$.

These axioms are clearly true. Let us see that they imply the requisite universal property. Let $f_i: \mathbf{F} \rightarrow \mathbf{D}_i$ $(i < n)$ be n morphisms then let the mediating morphism be $h \stackrel{\text{def}}{=} \langle f_0, \ldots, f_{n-1} \rangle$. By Axiom 1 all the required diagrams commute. Now let h' be any morphism which makes all the diagrams commute. Then $h = \langle f_0, \ldots, f_{n-1} \rangle = \langle \pi_0 \circ h', \ldots, \pi_{n-1} \circ h' \rangle$ (h' makes all the diagrams commute) $= h'$ (by Axiom 2).

Internal Axiomatisation.

1. $\forall x_0: \mathbf{D}_0. \ldots \forall x_{n-1}: \mathbf{D}_{n-1}. \pi_i(x_0, \ldots, x_{n-1}) = x_i$.
2. $\forall x: \prod_{i<n} \mathbf{D}_i. x = \langle \pi_0(x), \ldots, \pi_{n-1}(x) \rangle$.

Again the axioms are true and they imply the external ones (using the extensionality of functions, the definition of composition and the definition of the tupling operator on functions). Note that the external axiomatisation refers to "external" cpos, such as $\mathbf{F}$, whereas the internal one refers only to the $\mathbf{D}_i$ (and their product).

Returning to expressions we note an extended version of the correspondence between expressions for elements and expressions for morphisms. For any expression $e{:}\,\mathbf{D}$ suppose $x_0{:}\,\mathbf{D}_0, \ldots, x_{n-1}{:}\,\mathbf{D}_{n-1}$ is a list (without repetition) of variables of the indicated types which includes all the free variables of e (i.e. *all* the variables). Then we can define a function $f{:}\prod_{i<n}\mathbf{D}_i \to \mathbf{D}$ by $f(x_0, \ldots, x_{n-1}) = e$. This is a continuous function as it can be given by an expression $\alpha[e; x_0, \ldots, x_{n-1}]$ which is defined by structural induction on expressions e, thus:

(i) $$\alpha[a; x_0, \ldots, x_{n-1}] = K_a,$$

(ii) $$\alpha[x_i; x_0, \ldots, x_{n-1}] = \pi_i,$$

(iii) $$\alpha[\beta(e); x_0, \ldots, x_{n-1}] = \beta \circ (\alpha[e; x_0, \ldots, x_{n-1}]),$$

(iv) $$\alpha[\langle e_0, \ldots, e_{m-1}\rangle; x_0, \ldots, x_{n-1}] = \langle \alpha[e_0; x_0, \ldots, x_{n-1}], \ldots, \alpha[e_{m-1}; x_0, \ldots, x_{n-1}]\rangle.$$

It follows if we fix some arguments of a continuous function $f{:}\prod_{i<n}\mathbf{D}_i \to \mathbf{D}$ we still obtain a continuous function. For suppose we fix the ith argument to be the element d_i in $\mathbf{D}$ then the function $g{:}\,\mathbf{D}_0 \times \cdots \times \mathbf{D}_{i-1} \times \mathbf{D}_{i+1} \times \cdots \times \mathbf{D}_{n-1} \to \mathbf{D}$ where $g(x_0, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{n-1}) = f(x_0, \ldots, x_{i-1}, d_i, x_{i+1}, \ldots, x_{n-1})$ is continuous as the expression on the right of the equation is one of the expression we have considered above. Now it is clear that the same holds if we fix any subset of the arguments of f. As it happens the converse is true: if a function is continuous in each of its arguments it is continuous (where a function is continuous in its ith argument means that the function obtained by fixing all the other arguments is continuous). We illustrate this for binary functions. Certainly monotonicity in each argument implies monotonicity; for continuity suppose $f{:}\,\mathbf{D} \times \mathbf{E} \to \mathbf{F}$ and $\langle d_n, e_n\rangle_{n\in\omega}$ is an increasing chain of elements in $\mathbf{D} \times \mathbf{E}$. Then:

$$\begin{aligned}
f(\bigsqcup_n \langle d_n, e_n\rangle) &= f(\langle \bigsqcup_n d_n, \bigsqcup_n e_n\rangle) && \text{lubs are determined coordinatewise}\\
&= \bigsqcup_n f(d_n, \bigsqcup_n e_n) && f \text{ is continuous in its first argument}\\
&= \bigsqcup_n \bigsqcup_m f(d_n, e_m) && f \text{ is continuous in its second argument}\\
&= \bigsqcup_n f(d_n, e_n) && \text{by Exercise 1.20, as } f \text{ is monotonic.}
\end{aligned}$$

Example. Any function $f{:}\,X_0 \times \cdots \times X_{n-1} \to Y$ on sets extends to a monotonic and so continuous function, $f_\perp{:}\,(X_0)_\perp \times \cdots \times (X_{n-1})_\perp \to Y_\perp$ where

$$f_\perp(d_0, \ldots, d_{n-1}) = \begin{cases} \perp & (\text{if some } d_i \text{ is } \perp),\\ f(d_0, \ldots, d_{n-1}) & (\text{otherwise}).\end{cases}$$

This is the minimal extension and is the one we will normally consider. For example the resulting truth-table for $or_\perp{:}\,\mathbf{T}_\perp \times \mathbf{T}_\perp \to \mathbf{T}_\perp$ is:

$or_\perp$	tt	ff	$\perp$
tt	tt	tt	$\perp$
ff	tt	ff	$\perp$
$\perp$	$\perp$	$\perp$	$\perp$

A program for $or_\perp$ is: **if** x **then** (**if** y **then true else true**) **else** (**if** y **then true else false**). It is the "call-by-value" or. However there are three other monotonic extensions which we might call or_{L}, or_{R}, or_{P} with truth tables:

or_{L}	tt	ff	$\perp$
tt	tt	tt	tt
ff	tt	ff	$\perp$
$\perp$	$\perp$	$\perp$	$\perp$

or_{R}	tt	ff	$\perp$
tt	tt	tt	$\perp$
ff	tt	ff	$\perp$
$\perp$	tt	$\perp$	$\perp$

or_{P}	tt	ff	$\perp$
tt	tt	tt	tt
ff	tt	ff	$\perp$
$\perp$	tt	$\perp$	$\perp$

One occasionally sees programming constructs corresponding to or_{L}. One can write a program for it as: **if** x **then true else** (**if** y **then true else false**) and a similar one for or_{R}. However one can't write a sequential program for or_{P} — parallel or. This distinction raises the question where one can axiomatise the *sequential* continuous functions, as the others play no role in most programming languages.

The conditional function $cond\colon \mathbf{T}_\perp \times \mathbf{N}_\perp \times \mathbf{N}_\perp \to \mathbf{N}_\perp$ defined in Chapter 1 is also not the minimal extension of the set-theoretic condition, and this is in accord with the computational intuition that, according to the value of the first argument, either the second or the third argument is not evaluated. There is also a larger extension, $parcond\colon \mathbf{T}_\perp \times \mathbf{N}_\perp \times \mathbf{N}_\perp \to \mathbf{N}_\perp$ given by:

$$parcond(t,x,y) = \begin{cases} x & (t = tt \text{ or } x = y), \\ y & (t = f\!f \text{ or } x = y), \\ \perp & (\text{otherwise}). \end{cases}$$

Note that $parcond(\perp, x, x) = x$ showing its parallel nature. More generally we define $cond_{\mathbf{D}}\colon \mathbf{T}_\perp \times \mathbf{D} \times \mathbf{D} \to \mathbf{D}$ by:

$$cond_{\mathbf{D}}(t,x,y) = \begin{cases} x & (t = tt), \\ y & (t = f\!f), \\ \perp & (t = \perp). \end{cases}$$

We will often write $(e \to e' \mid e'')$ instead of $cond_{\mathbf{D}}(e, e', e'')$. When $\mathbf{D}$ has a continuous glb operation, $\sqcap$ (as is the case with flat domains) we define:

$$parcond_{\mathbf{D}}(t,x,y) = \begin{cases} x & (t = tt), \\ y & (t = f\!f), \\ x \sqcap y & (t = \perp) \end{cases}$$

and it is easily verified that $cond_{\mathbf{D}}$ and $parcond_{\mathbf{D}}$ are continuous. Note that or_{P} can be defined in terms of $parcond_{\mathbf{T}_\perp}$ by:

$$or_{\mathrm{P}}(x,y) = parcond_{\mathbf{T}_\perp}(x, tt, cond_{\mathbf{T}_\perp}(y, tt, f\!f)).$$

With the above exceptions, and unless otherwise stated, we will always use the minimal extensions hereafter and will drop the suffix, $\perp$ — similarly we will drop the suffices on $\mathbf{T}_\perp$, $\mathbf{N}_\perp$, ... when it is clear we intend a cpo rather than a set. So for example when e, e' are expressions of type $\mathbf{X}_\perp$, we write the expression $=_\perp (e, e')$ of type $\mathbf{T}$ in the less pedantic form $(e = e')$. Note this is different from the *assertion* $e = e'$ which means that e and e' denote the same value: so for example the assertion $\perp = \perp$ is true but the expression $(\perp = \perp)$ denotes $\perp$, as does $(e = \perp)$ or $(\perp = e)$.

Finally we consider the example of finite and infinite binary trees possibly with integers on their tips, ordered by the sub-tree ordering.

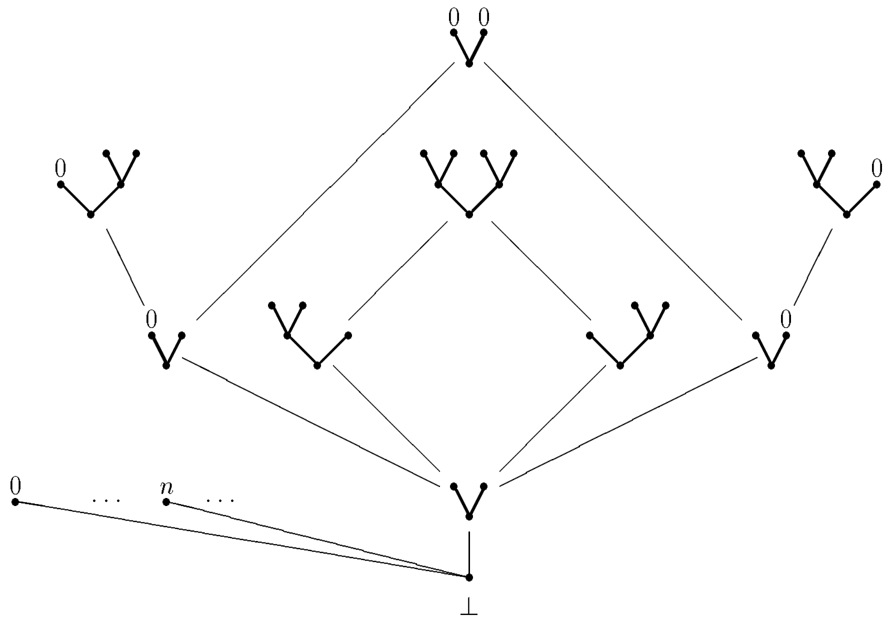


Figure VII: The cpo Btrees.

Some of this domain, **Btrees**, is drawn in the diagram. It is only one of a great variety of domains of trees. We will give a construction of **Btrees** later, but intuitively it can be thought of as generated by the following rules:

1. Any element of $\mathbf{N}_\perp$ is a Btree: if t_0, t_1 are elements so is $\overset{t_0\ t_1}{\curlyvee}$.

2. For any element, t, we have $\perp \sqsubseteq t$; if $t_0 \sqsubseteq t_0'$ and $t_1 \sqsubseteq t_1'$ then $\overset{t_0\ t_1}{\curlyvee} \sqsubseteq \overset{t_0'\ t_1'}{\curlyvee}$, for the finite elements, and then adding the infinite ones as limits of ω-chains with the rule: $\bigsqcup_n t_n \sqsubseteq \bigsqcup_n t_n' \equiv \forall n.\exists m.t_n \sqsubseteq t_m'$ to decide the approximation order between two chains.

The appropriate functions are $tip\colon \mathbf{N} \to \mathbf{Btrees}$, $mktree\colon \mathbf{Btrees} \times \mathbf{Btrees} \to \mathbf{Btrees}$, $val\colon \mathbf{Btrees} \to \mathbf{N}$, $ltree, rtree\colon \mathbf{Btrees} \to \mathbf{Btrees}$ and $istip\colon \mathbf{Btrees} \to \mathbf{T}$ where:

$$tip(n) = \begin{cases} \perp & (n = \perp), \\ \underset{\bullet}{n} & (n \neq \perp). \end{cases}$$

$$maketree(t_0, t_1) = \overset{t_0\ t_1}{\curlyvee}.$$

$$val(t) = \begin{cases} n & (t = \underset{\bullet}{n}), \\ \perp & (\text{otherwise}). \end{cases}$$

$$ltree(t) = \begin{cases} t_0 & (t = \overset{t_0\ t_1}{\curlyvee}), \\ \perp & (\text{otherwise}). \end{cases}$$

$$rtree(t) = \begin{cases} t_1 & (t = \overset{t_0\ t_1}{\curlyvee}), \\ \perp & (\text{otherwise}). \end{cases}$$

$$istip(t) = \begin{cases} tt & (t = \underset{\bullet}{n}), \\ ff & (t = \overset{t_0\ t_1}{\curlyvee}), \\ \perp & (t = \bullet). \end{cases}$$

We conclude the discussion of the Cartesian product construction by noting that there is a product operation on the morphisms too. Suppose $f\colon \mathbf{D}_i \to \mathbf{E}_i$ $(i < n)$ are continuous functions. Then $\prod_{i<n} f_i\colon \prod_{i<n} \mathbf{D}_i \to \prod_{i<n} \mathbf{E}_i$ is defined by:

$$\prod_{i<n} f_i = \langle f_0 \circ \pi_0, \ldots, f_{n-1} \circ \pi_{n-1} \rangle$$

so that $(\prod_{i<n} f_i)(x_0, \ldots, x_{n-1}) = \langle f(x_0), \ldots, f(x_{n-1}) \rangle$. Sometimes it is written as $f_0 \times \cdots \times f_{n-1}$ instead. The two product operations (on objects and morphisms) together make Cartesian product into a covariant *functor*, $\prod\colon \mathcal{CPO}^n \to \mathcal{CPO}$, which just means that the following equations are true:

(i) $$\prod_{i<n} id_{\mathbf{D}_i} = id_{\prod_{i<n} \mathbf{D}_i},$$

(ii) $$\prod_{i<n} (g_i \circ f_i) = (\prod_{i<n} g_i) \circ (\prod_{i<n} f_i).$$

It will prove very useful when we come to solving recursive domain equations (try Exercise 1.3) to consider domain constructions, such as Cartesian product, as functors.

In applications of the fixed-point theorem we will usually want to use domains of functions, as illustrated in Chapter 1 and so we now consider the function-space construction (exponentiation). Let $\mathbf{D}$ and $\mathbf{E}$ be cpos. Their function space, $\mathbf{D} \to \mathbf{E}$ in the set of all continuous functions from $\mathbf{D}$ to $\mathbf{E}$ with the pointwise ordering:

$$f \sqsubseteq g \quad \text{iff} \quad \forall x \in \mathbf{D}. f(x) \sqsubseteq_{\mathbf{E}} g(x).$$

Note that $(\mathbf{D} \to \mathbf{E})$ is a cpo with least element $K_{\perp_{\mathbf{E}}}$ and with lubs of increasing sequence determined pointwise by: $(\bigsqcup_n f_n)(x) = \bigsqcup_n (f_n(x))$.

We can characterise exponentiation by a universal property relating it to Cartesian product. Note that there is a natural application (evaluation) function, $apply\colon ((\mathbf{D} \to \mathbf{E}) \times \mathbf{D}) \to \mathbf{E}$ defined by:

$$apply(f, x) = f(x).$$

This is continuous as it is continuous in each argument separately. We have already seen in our discussion of expressions that $f(x)$ is continuous in x, when f is a fixed function (the value of some constant). The formula for lubs of increasing chains of continuous functions shows that the expression $f(x)$ is continuous in its first argument too.

The universal property is that if $f\colon \mathbf{F} \times \mathbf{D} \to \mathbf{E}$ is any continuous function (so that $\mathbf{F}$ is a "trial" function-space) there is a unique mediating morphism $g\colon \mathbf{F} \to (\mathbf{D} \to \mathbf{E})$ so that the following diagram commutes:

$$\begin{tikzcd}
(\mathbf{D} \to \mathbf{E}) \times \mathbf{D} \arrow[d, "apply"'] & \mathbf{F} \times \mathbf{D} \arrow[l, "g \times id_{\mathbf{D}}"'] \arrow[dl, "f"] \\
\mathbf{E} &
\end{tikzcd}$$

To see this suppose g is such a morphism. Then for any elements v of $\mathbf{F}$ and d of $\mathbf{D}$ we have:

$$\begin{aligned}
g(v)(d) &= apply(g(v), d) \\
&= apply(g \times id_{\mathbf{D}}(v, d)) \\
&= apply \circ (g \times id_{\mathbf{D}})(v, d) \\
&= f(v, d)
\end{aligned}$$

showing that $g(v)$ is uniquely determined as the function $z_v\colon (\mathbf{D} \to \mathbf{E})$ such that $z_v(d) = f(v, d)$, which is continuous with a fixed f and v, by the above remarks on expressions. To see that g itself is continuous, let $\langle v_n \rangle_{n \in \omega}$ be an increasing sequence of elements of $\mathbf{F}$ and calculate:

$$\begin{aligned}
g(\bigsqcup_n v_n)(d) &= f(\bigsqcup_n v_n, d) \\
&= \bigsqcup_n f(v_n, d) && f \text{ is continuous and so continuous in its first argument} \\
&= \bigsqcup_n (g(v_n)(d)) \\
&= (\bigsqcup_n g(v_n))(d) && \text{limits being determined pointwise}
\end{aligned}$$

which shows that $g(\bigsqcup_n v_n) = \bigsqcup_n g(v_n)$ by extensionality. In category-theoretic jargon, we have shown that $\mathcal{CPO}$ is Cartesian closed.

We write g as $Curry(f)$ and expand our stock of expressions for continuous functions:

E8. If α is an expression of type $(\mathbf{F} \times \mathbf{D}) \to \mathbf{E}$ then $Curry(\alpha)$ is one of type $\mathbf{F} \to (\mathbf{D} \to \mathbf{E})$, with denotation as specified above.

External Axiomatisation.

1. $\forall f\colon (\mathbf{F} \times \mathbf{D}) \to \mathbf{E}. f = apply \circ (Curry(f) \times id_{\mathbf{D}})$.
2. $\forall g\colon \mathbf{F} \to (\mathbf{D} \to \mathbf{E}). g = Curry(apply \circ (g \times id_{\mathbf{D}}))$.

These axioms are easily seen to be true. Axiom 1 implies the existence part of the universal property: Axiom 2 implies uniqueness as if $g\colon \mathbf{F} \to (\mathbf{D} \to \mathbf{E})$ makes the above diagram commute, for a given $f\colon (\mathbf{F} \times \mathbf{D}) \to \mathbf{E}$, then $g = \mathit{Curry}(\mathit{apply} \circ (g \times id_{\mathbf{D}}))$ (by Axiom 2) $= \mathit{Curry}(f)$. For the internal axiomatisation we have to expand our language further by introducing the *typed* λ*-calculus*:

E9. If e is an expression of type $\mathbf{E}$ and x is a variable of type $\mathbf{D}$ then $(\lambda x{:}\,\mathbf{D}.e)$ is an expression of type $(\mathbf{D} \to \mathbf{E})$.

Note that $\lambda x{:}\,\mathbf{D}.[\,]$ is a binding operator and so x is not a free variable in $\lambda x{:}\,\mathbf{D}.e$. We define the denotation of $(\lambda x{:}\,\mathbf{D}.e)$ whenever $e{:}\,\mathbf{E}$ satisfies the following conditions:

A) the denotation of e is defined for all values of its free variables.

B) if $x_0{:}\,\mathbf{D}_0, \ldots, x_{m-1}{:}\,\mathbf{D}_{m-1}$ is a list of variables of the indicated types which includes all the free variables of e then the function $f{:}\prod_{i<m} \mathbf{D}_i \to \mathbf{E}$ associated with e by: $f(x_0, \ldots, x_{m-1}) = e$ can be defined by an expression, $\alpha[e; x_0, \ldots, x_{m-1}]$ built up using E3, E4, E7, E8.

Then if the free variables of $(\lambda x{:}\,\mathbf{D}.e)$ are $x_0, \ldots, x_{n-1}$, the value of $(\lambda x{:}\,\mathbf{D}.e)$ at $x_0, \ldots, x_{n-1}$ is the function $f{:}\,\mathbf{D} \to \mathbf{E}$ given by $f(x) = e$. This is a continuous function since it is obtained from either $\alpha[e; x_0, \ldots, x_{n-1}]$ or $\alpha[e; x_0, \ldots, x_{n-1}]$ by fixing the values of $x_0, \ldots, x_{n-1}$. So we see that $(\lambda x{:}\,\mathbf{D}.e)$ satisfies condition A. We now check it also satisfies condition B. Let $x_0{:}\,\mathbf{D}_0, \ldots, x_{m-1}{:}\,\mathbf{D}_{m-1}$ be a list of variables including all the free variables of $(\lambda x{:}\,\mathbf{D}.e)$. Now let $\mathbf{D}_m \stackrel{\text{def}}{=} \mathbf{D}$ and let $\theta{:}\,(\prod_{i<m} \mathbf{D}_i) \times \mathbf{D} \to \prod_{i\leq m} \mathbf{D}_i$ be the isomorphism $\langle \pi_0 \circ \pi_0, \ldots, \pi_{m-1} \circ \pi_0, \pi_1\rangle$ (which has inverse $\langle\langle \pi_0, \ldots, \pi_{m-1}\rangle, \pi_m\rangle$). Then we can take $\alpha[\lambda x{:}\,\mathbf{D}.e; x_0, \ldots, x_{m-1}]$ to be $\mathit{Curry}(\alpha[e; x'_0, \ldots, x'_{m-1}, x] \circ \theta)$ where $x'_0{:}\,\mathbf{D}_0, \ldots, x'_{m-1}{:}\,\mathbf{D}_{m-1}, x{:}\,\mathbf{D}_m$ are a list of distinct variables such that x'_i is x_i unless x_i is x).

It follows from these remarks and (i) to (iv) above that any expression e for an element built up by E1–E9 obeys conditions A and B. On the other hand we can now see that the expression for morphisms can be viewed as abbreviations for expressions for elements, now that the function-space construction allows us to think of morphisms as elements. Suppose we just allow an expression-forming rules E1, E2, E5′ (which is E5 restricted to the case where α is a constant), E6, and E9. Then if $\alpha{:}\,\mathbf{B} \to \mathbf{C}$ and $\beta{:}\,\mathbf{A} \to \mathbf{B}$ are abbreviations for e' and e respectively, $\alpha \circ \beta$ is one for $\lambda x{:}\,\mathbf{A}.\mathit{apply}(e', \mathit{apply}(e, x))$; if $\alpha{:}\,\mathbf{A} \to \mathbf{B}$ abbreviates e' then $\alpha(e)$ abbreviates $\mathit{apply}(e', e)$; if $\alpha_0{:}\,(\mathbf{F} \to \mathbf{D}_0), \ldots, \alpha_{n-1}{:}\,(\mathbf{F} \to \mathbf{D}_{n-1})$ abbreviates $e_0, \ldots, e_{n-1}$ respectively $\langle \alpha_0, \ldots, \alpha_{n-1}\rangle$ abbreviates $\lambda v{:}\,\mathbf{F}.\langle \mathit{apply}(e_0, v), \ldots, \mathit{apply}(e_{n-1}, v)\rangle$; if $\alpha{:}\,\mathbf{F} \times \mathbf{D} \to \mathbf{E}$ abbreviates e then $\mathit{Curry}(\alpha)$ abbreviates $\lambda v{:}\,\mathbf{F}.\lambda d{:}\,\mathbf{D}.\mathit{apply}(e, \langle v, d\rangle)$.

The rules E1, E2, E5′, E6, E9 give us a *typed* λ*-calculus* and it is convenient to introduce a few more abbreviations. For expressions $e'{:}\prod_{i<n} \mathbf{A}_i \to \mathbf{B}$, $e_0{:}\,\mathbf{A}_0, \ldots, e_{n-1}{:}\,\mathbf{A}_{n-1}$ $e'(e_0, \ldots, e_{n-1})$ abbreviates $\mathit{apply}(e', \langle e_0, \ldots, e_{n-1}\rangle)$; when $\mathbf{D}$ can be understood from the context $(\lambda x.e)$ abbreviates $(\lambda x{:}\,\mathbf{D}.e)$; the expression $(\lambda x_0{:}\,\mathbf{D}_0, x_1{:}\,\mathbf{D}_1, \ldots, x_{n-1}{:}\,\mathbf{D}_{n-1}.e)$ abbreviates

$$(\lambda x{:}\prod_{i<n} \mathbf{D}_i.(\lambda x_0{:}\,\mathbf{D}_0(\ldots(\lambda x_{n-1}{:}\,\mathbf{D}_{n-1}.e)\ldots))(\pi_0(x))\ldots(\pi_{n-1}(x))).$$

Now the internal axiomatisation of the function space construction just consists of the familiar rules for the typed λ-calculus.

Internal Axiomatisation.

1. *β-reduction*: $(\lambda x.\alpha[e; x])e' = \alpha(e')$.
2. *η-reduction*: $(\lambda x.e(x)) = e$ (x not a free variable of e).

The above axioms are asserted for all expressions e and all values of their free variables. They are evidently true and we check that they imply the external axioms (together with our other axioms and a few elementary ones and principles of logic). Suppose $f{:}\,(\mathbf{F} \times \mathbf{D}) \to \mathbf{E}$ and calculate for each w in $\mathbf{F} \times \mathbf{D}$.

$$\begin{aligned}
(\mathit{apply} \circ (\mathit{Curry}(f) \times id_{\mathbf{D}}))(w) &= \mathit{apply}((\mathit{Curry}(f) \times id_{\mathbf{D}})(w)) \\
&= \mathit{apply}(\mathit{Curry}(f)(\pi_0(w)), \pi_1(w)) \\
&= (\lambda v{:}\,\mathbf{F}.\lambda d{:}\,\mathbf{D}.\mathit{apply}(f, \langle v, d\rangle))(\pi_0(w))(\pi_1(w)) \\
&= (\lambda d{:}\,\mathbf{D}.f(\pi_0(w), d))(\pi_1(w)) && \beta\text{-reduction} \\
&= f(\pi_0(w), \pi_1(w)) && \beta\text{-reduction} \\
&= f(w).
\end{aligned}$$

Then Axiom 1 follows by extensionality. For Axiom 2, suppose $g\colon \mathbf{F} \to (\mathbf{D} \to \mathbf{E})$ and calculate for each v in $\mathbf{F}$:

$$
\begin{aligned}
Curry(apply \circ (g \times id_{\mathbf{D}}))(v) &= (\lambda v\colon \mathbf{F}.\lambda d\colon \mathbf{D}.apply(apply \circ (g \times id_{\mathbf{D}}), \langle v, d\rangle))(v) \\
&= \lambda d\colon \mathbf{D}.apply((g \times id_{\mathbf{D}})(v, d)) && \beta\text{-reduction} \\
&= \lambda d\colon \mathbf{D}.g(v)(d) \\
&= g(v) && \eta\text{-reduction.}
\end{aligned}
$$

and then Axiom 2 follows by extensionality. Note by the way that all the operations such as composition, $\circ$, the two tupling operators, $\langle \cdot, \ldots, \cdot \rangle$, and Currying ($Curry$) are all continuous as we have $\circ = \lambda f.\lambda g.g \circ f$ etc; also $\times$ acts continuously on morphisms.

As a functor exponentiation, $\to$, is defined on morphisms by defining $f \to f'\colon (\mathbf{A} \to \mathbf{A}') \to (\mathbf{B} \to \mathbf{B}')$ where $f\colon \mathbf{B} \to \mathbf{A}$, $f'\colon \mathbf{A}' \to \mathbf{B}'$ by:

$$f \to f' = \lambda x\colon \mathbf{A} \to \mathbf{A}'.f' \circ x \circ f.$$

Note that $f\colon \mathbf{B} \to \mathbf{A}$ rather than $f\colon \mathbf{A} \to \mathbf{B}$ which indicates that $\to\colon \mathcal{CPO}^2 \to \mathcal{CPO}$ is a bifunctor which is contravariant in its first argument and covariant in its second argument, which is to say that the following equations are true, where $f\colon \mathbf{B} \to \mathbf{A}$, $g\colon \mathbf{C} \to \mathbf{B}$, $f'\colon \mathbf{A}' \to \mathbf{B}'$, $g'\colon \mathbf{B}' \to \mathbf{C}'$:

(i) $id_{\mathbf{D}} \to id_{\mathbf{E}} = id_{\mathbf{D} \to \mathbf{E}}$,
(ii) $(f \circ g) \to (g' \circ f') = (g \to g') \circ (f \to f')$.

Again, exponentiation acts continuously on morphisms.

Now that we have the basic constructions on cpos and a language for expressing elements and morphisms we return to fixed-points and then see how to give a denotational semantics for *Imp* and *Dec* when extended by adding iteration and recursive declaration facilities. The least fixed-point, fix_f, of a continuous function $f\colon \mathbf{D} \to \mathbf{D}$ is itself a continuous function of f, $\mathit{fix}\colon (\mathbf{D} \to \mathbf{D}) \to \mathbf{D}$ as we have:

$$\mathit{fix}_f = \bigsqcup_n f^n(\bot) = \bigsqcup_n (\lambda f.f^n(\bot))(f) = (\bigsqcup_n \lambda f.f^n(\bot))(f)$$

showing that $\mathit{fix} = \bigsqcup_n \lambda f.f^n(\bot)$, a continuous function. Often fix is called, Y, after the paradoxical combinator of combinatory logic; we will often use the abbreviation $(\mu x\colon \mathbf{D}.e)$ (or just $\mu x.e$ when $\mathbf{D}$ is understood from the context) to stand for $\mathit{fix}(\lambda x.e)$ — clearly $\mu x.e$ is the least element, x such that $x = e$. Sometimes we just define elements, x, recursively by writing $x = e$, meaning $x = \mu x.e$, or even $f(x_0, \ldots, x_{n-1}) = e$ for functions meaning $f = \mu f.(\lambda x_0, \ldots, x_{n-1}.e)$.

Examples

1. In the case of the statement **while** b **do** s we will be given a meaning $p_\bot\colon \mathbf{S}_\bot \to \mathbf{T}_\bot$ for b (in the case that evaluation of b always terminates) and a meaning $\theta'\colon \mathbf{S}_\bot \to \mathbf{S}_\bot$ for s and take the meaning of the statement to be $\theta = \mathit{fix}(F)$ where $F\colon (\mathbf{S}_\bot \to \mathbf{S}_\bot) \to (\mathbf{S}_\bot \to \mathbf{S}_\bot)$ is $F = \lambda\theta\colon (\mathbf{S}_\bot \to \mathbf{S}_\bot).\lambda\sigma\colon \mathbf{S}_\bot.cond(p_\bot(\sigma), \theta(\theta'(\sigma)), \sigma)$.

2. In the case of the recursive declaration **letrec** $f(x)$ **be** e **in** e' we shall take the meaning of f to be $\mu g.G(g)$ where $G(g)(v)$ is the value of e in the current environment, altered so that f and x have values g and v respectively.

3. Of course we can just define elements without using any particular programming language. So addition can be defined by taking over the usual primitive recursive definition as:

   $$+ = \mu f\colon (\mathbf{N}_\bot \times \mathbf{N}_\bot \to \mathbf{N}_\bot).\lambda m\colon \mathbf{N}_\bot.\lambda n\colon \mathbf{N}_\bot.(n = 0 \to m \mid f(m, n-1) + 1).$$

   We can also define functions by minimalisation. Suppose we have $p_\bot\colon \mathbf{N}_\bot^2 \to \mathbf{T}_\bot$ and wish to define $f\colon \mathbf{N}_\bot \to \mathbf{N}_\bot$ as $f(m) =$ the smallest integer n such that $p_\bot(m, n) = tt$. Then

   $$f = \lambda n.(\mu h.\lambda m.(p(n, m) \to m \mid h(m+1)))(0).$$

   It should now be clear that in this way we can define (the minimal extension of) any partial recursive function or predicate over the integers. We can also define many functionals at higher types. For instance suppose we want to define $sum\colon ((\mathbf{N}_\bot \to \mathbf{N}_\bot) \times \mathbf{N}_\bot) \to \mathbf{N}_\bot$ so that $sum(f, n) = f(0) + \cdots + f(n-1)$, then sum is defined recursively by: $sum(f, n) = (n = 0 \to 0 \mid f(n-1) + sum(f, n-1))$, or, in other words: $sum = \mu sum.\lambda f.\lambda n.(n = 0 \to 0 \mid f(n-1) + sum(f, n-1))$. Later on we will see that all the functions and functionals we can define are computable, in a sense to be described.

4. In the case of domains such as **Tapes** we can define some infinite elements by recursion: for example $010101\ldots$ is just $\mu x.0.1.x$. One way to obtain nonperiodic sequence is to take $p{:}\,\mathbf{N} \to \{0,1\}$ and define $g{:}\,\mathbf{N} \to \mathbf{Tapes}$ recursively by:

$$g(n) = (p(n) = 0 \to 0.g(n+1) \mid 1.g(n+1)).$$

Then $g(0)$ is $p(0)p(1)p(2)p(3)\ldots$.

Similarly in the case of $\mathbf{P}\boldsymbol{\omega}$ take $p{:}\,\mathbf{N}_\perp \to \mathbf{T}_\perp$ and define $g{:}\,\mathbf{N}_\perp \to \mathbf{P}\boldsymbol{\omega}$ by:

$$g(n) = \{g(n+1)+1\} \cup (p(n) \to \{0\} \mid \perp)$$

where $\cup{:}\,\mathbf{P}\boldsymbol{\omega}^2 \to \mathbf{P}\boldsymbol{\omega}$ is the (continuous) union function and $\{0\}$ is the singleton zero set. Then $g(0) = \{n \in \omega \mid p(n) = tt\}$.

5. Simultaneous fixed-points are easily reduced to single ones. Suppose $f_i{:}\,(\prod_{i<n} \mathbf{D}_i) \to \mathbf{D}_i$ $(i < n)$ are continuous functions. The elements $d_0, \ldots, d_{n-1}$ of $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ respectively are simultaneous fixed-points of $f_0, \ldots, f_{n-1}$ iff

$$\begin{aligned} d_0 &= f_0(d_0, \ldots, d_{n-1}),\\ &\;\vdots\\ d_{n-1} &= f_{n-1}(d_0, \ldots, d_{n-1}). \end{aligned}$$

Then $d_0, \ldots, d_{n-1}$ are simultaneous fixed-points of $f_0, \ldots, f_{n-1}$ iff $\langle d_0, \ldots, d_{n-1}\rangle$ is a fixed-point of $\langle f_0, \ldots, f_{n-1}\rangle$ and we see that the least simultaneous fixed-points are $d_i = (\mathit{fix}(\langle f_0, \ldots, f_{n-1}\rangle))_i$.

For instance we can define functions $g{:}\,\mathbf{N}^2 \to \mathbf{N}$ recursively by:

$$\begin{aligned} g(x,y) &= (x = 0 \to 7 \mid g(x-1, h(y))),\\ h(x) &= (x = 0 \to 6 \mid g(x-1, x+1)) \end{aligned}$$

meaning that g, h should be the least simultaneous fixed points of the functions F_0, F_1 where $F_0 = \lambda g{:}\,(\mathbf{N}^2 \to \mathbf{N}), h{:}\,\mathbf{N} \to \mathbf{N}.\lambda x{:}\,\mathbf{N}.\lambda y{:}\,\mathbf{N}.(x = 0 \to 7 \mid g(x-1, h(y)))$ and $F_1 = \lambda g, h.\lambda x.(x = 0 \to 6 \mid g(x-1, x+1))$.

The idea of making definitions with the aid of the least *fixed-point* operator, *fix*, leads to a major proof rule, due to Scott, called *fixed-point induction* (sometimes *computational induction*).

Definition 2. Let $\mathbf{D}$ be a cpo. A property $\mathbf{P}$ (a subset of $\mathbf{D}$) is *ω-inductive* (= *inclusive*) iff whenever $\langle x_n\rangle_{n\in\omega}$ is an increasing sequence of elements in $\mathbf{P}$, its lub, $\bigsqcup_n x_n$ is also in $\mathbf{P}$.

Fixed-point Induction. Let $\mathbf{P} \subseteq \mathbf{D}$ be ω-inductive and let $f{:}\,\mathbf{D} \to \mathbf{D}$ be continuous. Then we have: $[\mathbf{P}(\perp) \wedge (\forall x{:}\,\mathbf{D}.\mathbf{P}(x) \Rightarrow \mathbf{P}(fx))] \Rightarrow \mathbf{P}(\mathit{fix}(f))$.

The validity of this principle is clear: note that it can be rephrased as every subcpo closed under a continuous function contains fix_f. One idea behind fixed-point induction is to avoid integer reasoning to show $\forall n \in \omega.\mathbf{P}(f^n(\perp))$ when proving $\mathbf{P}(\mathit{fix}_f)$.

Examples. For $f{:}\,\mathbf{D} \to \mathbf{D}$ prove $\mathit{fix}_f \sqsubseteq f(\mathit{fix}_f)$ by using $\mathbf{P}(x) \stackrel{\text{def}}{=} x \sqsubseteq fx$. The opposite inclusion needs a separate axiom in any formal system.

It is important to be able to manufacture inclusive predicates.

1. **Basic Relations.** Let $\mathbf{D}$ be a cpo. Then $\emptyset$, $\mathbf{D}$ are inclusive predicates on $\mathbf{D}$; $x = y$ and $x \sqsubseteq y$ are inclusive relations on $\mathbf{D} \times \mathbf{D}$.

2. **Inverse Image.** Let $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ be cpos and let $\mathbf{R} \subseteq \prod_{i<n} \mathbf{D}_i$ be inclusive. Then if $f_i{:}\,\prod_{j<m} \mathbf{E}_j \to \mathbf{D}_i$ are i continuous functions $(i < n)$ then

$$\mathbf{Q}(y_0, \ldots, y_{m-1}) \stackrel{\text{def}}{=} \mathbf{R}(f_0(y_0, \ldots, y_{m-1}), \ldots, f_{n-1}(y_0, \ldots, y_{m-1}))$$

is inclusive too. In particular taking $\mathbf{D}_0 = \mathbf{D}_1$ and $\mathbf{R}(x_0, x_1) \equiv x_0 \sqsubseteq x_1$ we find that inclusions of the form $f_0(y_0, \ldots, y_{m-1}) \sqsubseteq f_1(y_0, \ldots, y_{m-1})$ are inclusive.

3. **Logical.** Let $\mathbf{D}$ be a cpo and if $\mathbf{P}, \mathbf{Q} \subseteq \mathbf{D}$ are inclusive so are $\mathbf{P} \cap \mathbf{Q}$ and $\mathbf{P} \cup \mathbf{Q}$: If $\mathbf{E}$ is another cpo and $\mathbf{R} \subseteq \mathbf{D} \times \mathbf{E}$ is inclusive with respect to its first argument then $\forall y{:}\,\mathbf{E}.\mathbf{R}(x, y)$ is inclusive.

4. **Products.** Let $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ be cpos and let $\mathbf{P}_0 \subseteq \mathbf{D}_0, \ldots, \mathbf{P}_{n-1} \subseteq \mathbf{D}_{n-1}$ be inclusive. Then $\prod_{i<n} \mathbf{P}_i \subseteq \prod_{i<n} \mathbf{D}_i$ is inclusive where $\prod_{i<n} \mathbf{P}_i = \{x \in \prod_{i<n} \mathbf{D}_i \mid \forall i < n. \mathbf{P}_i((x)_i)\}$.
5. **Exponentiation.** Let $\mathbf{D}, \mathbf{E}$ be cpos and let $\mathbf{P} \subseteq \mathbf{D}$, $\mathbf{Q} \subseteq \mathbf{E}$ be inclusive. Then $(\mathbf{P} \to \mathbf{Q}) \subseteq (\mathbf{D} \to \mathbf{E})$ is inclusive where: $(\mathbf{P} \to \mathbf{Q}) = \{f: \mathbf{D} \to \mathbf{E} \mid \forall d: \mathbf{D}. \mathbf{P}(d) \Rightarrow \mathbf{Q}(fd)\}$.

Unfortunately, as remarked above, we do not have time to pursue issues connected with proof much beyond stating the relevant axioms. Next we see that it is now possible to give the semantics of *Dec* and *Imp* with the added iterative and recursive declaration features.

For *Imp* we have the syntax of Chapter 1, including the iteration statement. The semantic domains are $\mathbf{T}_\perp$, $\mathbf{S}_\perp$, and now $\mathbf{C} = (\mathbf{S}_\perp \to \mathbf{S}_\perp)$ the function space cpo. The denotational functions are $\mathcal{B}: \mathbf{BExp} \to (\mathbf{S}_\perp \to \mathbf{T}_\perp)$ (where the range of $\mathbf{BExp}$ is the cpo $(\mathbf{S}_\perp \to \mathbf{T}_\perp)$), $\mathcal{A}: \mathbf{Act} \to \mathbf{C}$ and $\mathcal{C}: \mathbf{Stat} \to \mathbf{C}$ which is defined by structural induction on statements by:

(i) $\mathcal{C}[\![a]\!] = \mathcal{A}[\![a]\!]$;

(ii) $\mathcal{C}[\![\mathbf{dummy}]\!] = \lambda\sigma: \mathbf{S}_\perp.\sigma$;

(iii) $\mathcal{C}[\![(s_1; s_2)]\!] = \mathcal{C}[\![s_2]\!] \circ \mathcal{C}[\![s_1]\!]$;

(iv) $\mathcal{C}[\![\mathbf{if}\ b\ \mathbf{then}\ s_1\ \mathbf{else}\ s_2]\!] = \lambda\sigma: \mathbf{S}_\perp.(\mathcal{B}[\![b]\!](\sigma) \to \mathcal{C}[\![s_1]\!](\sigma) \mid \mathcal{C}[\![s_2]\!](\sigma))$;

(v) $\mathcal{C}[\![\mathbf{while}\ b\ \mathbf{do}\ s]\!] = \mu\theta: \mathbf{C}.\lambda\sigma: \mathbf{S}_\perp.(\mathcal{B}[\![b]\!](\sigma) \to \theta(\mathcal{C}[\![s]\!](\sigma)) \mid \sigma)$.

We have abstracted the free variables on the right-hand side of the definitions in Chapter 1 to make it a little clearer that the meaning of a statement is a function of the meanings of its components. Note that our discussion on the typed λ-calculus we are using as a metalanguage show that, by structural induction, each statement receives a meaning expressible in this typed λ-calculus providing each primitive expression of the form $\mathcal{A}[\![a]\!]$ or $\mathcal{B}[\![b]\!]$ is taken as a constant. This is, perhaps, a little awkward; ideally we might like to have a single expression in the metalanguage for the function $\mathcal{C}$, and that would require us to take the syntax as a cpo too. It is most convenient to leave this idea for the time being and return to it when considering recursively specified domain equations.

Equations (i)–(iv) are just taken over from the equations we had when using ordinary sets to give the semantics. Equation (v) should be exactly what the reader expects given the discussion of the iteration statement in Chapter 1 and Example 1 above. Notice that the cpo $\mathbf{C}$ is too large in the sense that commands should never be nonstrict. This point will be taken up when we consider the strict function space construction in Chapter 3.

Turning to *Dec*, we have the syntax of Chapter 1, including the recursive declaration facility extended to functions of several arguments. We assume that there are fixed limits (m and n) on the arity of the function and operation symbols, as we have only considered finite products so far. For the semantic domains we take $\mathbf{V}$ to be a fixed cpo of values, $\mathbf{IEnv}$ to be $(\mathbf{Id}_\perp \to \mathbf{V})$. $\mathbf{FEnv}$ to be $(\mathbf{Fun}_j)_\perp \to (\mathbf{V}^j \to \mathbf{V})$. Again we may question why nonstrict environments are allowed. The denotational functions are $\mathcal{I}_i: \mathbf{Op}_i \to (\mathbf{V}^i \to \mathbf{V})$ $(i < n)$, $\mathcal{B}: \mathbf{BExp} \to ((\mathbf{IEnv} \times \mathbf{FEnv}) \to \mathbf{T})$ and $\mathcal{E}: \mathbf{Exp} \to ((\mathbf{IEnv} \times \mathbf{FEnv}) \to \mathbf{V})$ which is defined by structural induction:

(i) $\mathcal{E}[\![x]\!] = \lambda\rho: \mathbf{IEnv}, \varphi: \mathbf{FEnv}.\rho[\![x]\!]$;

(ii) $\mathcal{E}[\![o_i(e_0, \ldots, e_{i-1})]\!] = \lambda\rho: \mathbf{IEnv}, \varphi: \mathbf{FEnv}.\mathcal{I}_i[\![o_i]\!](\mathcal{E}[\![e_0]\!](\rho, \varphi), \ldots, \mathcal{E}[\![e_{i-1}]\!](\rho, \varphi))$ $\quad (i < n)$;

(iii) $\mathcal{E}[\![\mathbf{if}\ b\ \mathbf{then}\ e_0\ \mathbf{else}\ e_1]\!] = \lambda\rho: \mathbf{IEnv}, \varphi: \mathbf{FEnv}.(\mathcal{B}[\![b]\!](\rho, \varphi) \to \mathcal{E}[\![e_0]\!](\rho, \varphi) \mid \mathcal{E}[\![e_1]\!](\rho, \varphi))$;

(iv) $\mathcal{E}[\![\mathbf{let}\ x\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] = \lambda\rho: \mathbf{IEnv}, \varphi: \mathbf{FEnv}.\mathcal{E}[\![e_1]\!](\rho[\mathcal{E}[\![e_0]\!](\rho, \varphi)/x], \varphi)$;

(v)$_j$ $\mathcal{E}[\![\mathbf{let}\ f_j(x_0, \ldots, x_{j-1})\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] =$

$$\lambda\rho: \mathbf{IEnv}, \varphi: \mathbf{FEnv}.\mathcal{E}[\![e_1]\!](\rho, \varphi[\lambda w: \mathbf{V}^j.\mathcal{E}[\![e_0]\!](\rho[(w)_0/x_0] \ldots [(w)_{j-1}/x_{j-1}], \varphi)/f_j]);$$

(vi)$_j$ $\mathcal{E}[\![\mathbf{letrec}\ f_j(x_0, \ldots, x_{j-1})\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] =$

$$\lambda\rho: \mathbf{IEnv}, \varphi: \mathbf{FEnv}.\mathcal{E}[\![e_1]\!](\rho, \varphi[\mu f: \mathbf{V}^j \to \mathbf{V}.\lambda w: \mathbf{V}^j.\mathcal{E}[\![e_0]\!](\rho[(w)_0/x_0] \ldots [(w)_{j-1}/x_{j-1}], \varphi[f/f_j])/f_j]).$$

Here $\cdot[\cdot/\cdot]: \mathbf{IEnv} \times \mathbf{V} \times \mathbf{Id}_\perp \to \mathbf{IEnv}$ is defined by:

$$\rho[v/x] = \lambda y: \mathbf{Id}_\perp.((y = x) \to v \mid \rho[\![y]\!]),$$

and $\cdot[\cdot/\cdot]\colon \mathbf{FEnv} \times (\mathbf{V}^j \to \mathbf{V}) \times (\mathbf{Fun}_j)_\perp \to \mathbf{FEnv}$ is defined by:

$$\varphi[g/f] = \langle \varphi_0, \ldots, \varphi_{j-1}, \lambda f_j\colon (\mathbf{Fun}_j)_\perp.(f_j = f \to g \mid (\varphi)_j(f_j)), \varphi_{j+1}, \ldots, \varphi_{m-1}\rangle.$$

Again equations (i)–(v) are taken over from Chapter 1 and equation (vi) should be as expected from previous discussions. Again there is a slight awkwardness in treating syntax in terms of sets. Further this semantics corresponds to call-by-name (see Exercise 33, below) and the reader may wonder how call-by-value can be catered for in the present framework. Finally we note that the semantic equation for assignments in Chapter 1 carries over to the present situation without any alteration.

Exercises

1. Let $\mathbf{D}$ be a cpo. Use the external axiomatisation of Cartesian product to show that $\langle f_0, \ldots, f_{n-1}\rangle \circ g = \langle f_0 \circ g, \ldots, f_{n-1} \circ g\rangle$ and then prove the functor laws for $\prod$. Define $\Delta_{\mathbf{D}}\colon \mathbf{D} \to \mathbf{D} \times \cdots \times \mathbf{D}$ (the diagonal map) by: $\Delta_{\mathbf{D}} = \langle \mathbf{id_D}, \ldots, \mathbf{id_D}\rangle$ and show that: $\langle f_0, \ldots, f_{n-1}\rangle = (f_0 \times \cdots \times f_{n-1}) \circ \Delta_{\mathbf{D}}$.

2. Prove the isomorphisms: $\mathbf{U} \times \mathbf{D} \cong \mathbf{D} \times \mathbf{U} \cong \mathbf{D}$; $\mathbf{D} \times \mathbf{E} \cong \mathbf{E} \times \mathbf{D}$; $\mathbf{D} \times (\mathbf{E} \times \mathbf{F}) \cong (\mathbf{D} \times \mathbf{E}) \times \mathbf{F}$. Formulate and prove general versions of these commutative and associative laws for arbitrary finite products.

3. For any map $\delta\colon [m] \to [n]$ (where $[m] = \{j \in \mathbf{N} \mid j < m\}$ for any m in $\mathbf{N}$) and cpos $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ and $\mathbf{E}_0, \ldots, \mathbf{E}_{m-1}$ where $\mathbf{E}_j = \mathbf{D}_{\delta(j)}$ define the natural *substitution* map $\mathcal{S}_\delta\colon \prod_{i<n} \mathbf{D}_i \to \prod_{j<m} \mathbf{E}_j$. Show that $\mathcal{S}_\delta \circ \mathcal{S}_{\delta'} = \mathcal{S}_{\delta' \circ \delta}$. Show that for any expression, e, of the metalanguage whose free variables are included in the list $x_0\colon \mathbf{D}_0, \ldots, x_{n-1}\colon \mathbf{D}_{n-1}$ and also in the list $y_0\colon \mathbf{E}_0, \ldots, y_{m-1}\colon \mathbf{E}_{m-1}$ (given via a map $\delta\colon [m] \to [n]$ as above) we have: $\alpha[e; x_0, \ldots, x_{n-1}] = \alpha[e; y_0, \ldots, y_{m-1}] \circ \mathcal{S}_\delta$.

4. Investigate products $\prod_{i \in S} \mathbf{D}_i$ over any set S. Is it the case that a function of infinitely many arguments is continuous iff it is continuous in each of its arguments separately? [*Hint* Consider the case $S = \omega$ $(= \mathbf{N})$ and $\mathbf{D}_i = \mathbb{O}$.] When the $\mathbf{D}_i$ are all the same cpo, $\mathbf{D}$, we obtain a power, $\mathbf{D}^S$. Show that $\mathbf{T}^\omega \cong \mathbf{T} \times \mathbf{T}^\omega$. Why are $\mathbf{T}^\omega$ and $\mathbf{Tapes}$ not isomorphic?

5. Let $\mathbf{D}$ and $\mathbf{E}_0, \ldots, \mathbf{E}_{n-1}$ be cpos. Show that for any continuous function $f\colon \mathbf{D} \to \prod_{i<n} \mathbf{E}_i$ there are unique maps $f_i\colon \mathbf{D} \to \mathbf{E}_i$ $(i < n)$ such that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{D} \arrow[d, "\Delta"'] \arrow[dr, "f"] & \\
\mathbf{D}^n \arrow[r, "\prod_{i<n} f_i"'] & \prod_{i<n} \mathbf{E}
\end{tikzcd}$$

6. Show that $\prod_{i<n} \mathbf{D}_i$ is also the product in $\mathcal{CPO}_\perp$. [*Hint* Check everything is strict.]

7. Check the isomorphisms: $\langle \cdot, \ldots, \cdot\rangle\colon \prod_{i<n}(\mathbf{D} \to \mathbf{E}_i) \cong \mathbf{D} \to \prod_{i<n} \mathbf{E}_i$; $(\mathbf{D} \to \mathbf{U}) \cong \mathbf{U}$; $(\mathbf{U} \to \mathbf{D}) \cong \mathbf{D}$; $\mathbf{D} \to \mathbf{E}^S \cong (\mathbf{D} \to \mathbf{E})^S$.

8. Let $\mathbf{C}$, $\mathbf{D}$, $\mathbf{E}$ be cpos. Define $\mathit{pair}\colon \mathbf{C} \to (\mathbf{D} \to (\mathbf{C} \times \mathbf{D}))$ by $\mathit{pair} = \mathit{Curry}(\mathit{id}_{\mathbf{C}\times\mathbf{D}}) = \lambda x.\lambda y.\langle x, y\rangle$. Show that for any continuous $f\colon \mathbf{C} \to (\mathbf{D} \to \mathbf{E})$ there is a unique $g\colon \mathbf{C} \times \mathbf{D} \to \mathbf{E}$ (namely $\mathit{Curry}^{-1}(f)$) such that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{C} \arrow[d, "\mathit{pair}"'] \arrow[dr, "f"] & \\
\mathbf{D} \to (\mathbf{C} \times \mathbf{D}) \arrow[r, "(\mathit{id}_{\mathbf{D}} \to g)"'] & (\mathbf{D} \to \mathbf{E})
\end{tikzcd}$$

Show $\mathit{Curry} = \lambda f.(\mathit{id} \times f) \circ \mathit{pair}$ and $\mathit{Curry}^{-1} = \lambda g.\mathit{apply} \circ \langle g \circ \pi_0, \pi_1\rangle$.

9. Show that exponentiation preserves strictness giving a bifunctor $\rightarrow: \mathcal{CPO}_{\perp}^{2} \rightarrow \mathcal{CPO}_{\perp}$ which in contravariant in its first argument and covariant in its second argument.

10. Show that every continuous function, $p: \mathbf{T}^n \rightarrow \mathbf{T}$, can be defined from *tt*, *ff* and *parcond* as the only constants using the typed λ-calculus (you will only need first-order terms — those with no λ's). A collection of type symbols so are $(\sigma \times \tau)$ and $(\sigma \rightarrow \tau)$; the cpos $\mathbf{T}_\sigma$ are defined by: $\mathbf{T}_0 = \mathbf{T}$, $\mathbf{T}_{\sigma\times\tau} = \mathbf{T}_\sigma \times \mathbf{T}_\tau$, $\mathbf{T}_{\sigma\rightarrow\tau} = \mathbf{T}_\sigma \rightarrow \mathbf{T}_\tau$. Show that every element of every $\mathbf{T}_\sigma$ can be defined from *tt*, *ff* and *parcond* using the typed λ-calculus. [*Warning* This last part is a good deal harder.]

11. Let $\mathbf{X}_0, \ldots, \mathbf{X}_{n-1}, \mathbf{Y}$ be sets. Show any function $f: \mathbf{X}_0 \times \cdots \mathbf{X}_{n-1} \rightarrow \mathbf{Y}$ has a *maximal* extension to a continuous function $g: \prod_{i<n} (\mathbf{X}_i)_\perp \rightarrow \mathbf{Y}_\perp$. Is $parcond_{\mathbf{D}}$ always such a maximal extension?

12. Both Milner and Vuillemin have given definitions of sequentiality. These depend on viewing a function $f: \prod_{i<n} \mathbf{D}_i \rightarrow \mathbf{E}$ as being of n arguments (viewed as being more or less arguments may change its character according to these definitions!). Such a function is *M-sequential* if either it is constant or there is an integer i (with $0 \leq i < n$) such that f is strict in its i-th argument ($(x)_i = \perp$ implies $f(x) = \perp$) and the function obtained by fixing its i-th argument (to, say, a_i)

$$(\lambda x_0: \mathbf{D}_0, \ldots, x_{i-1}: \mathbf{D}_{i-1}, x_{i+1}: \mathbf{D}_{i+1}, \ldots, x_{n-1}: \mathbf{D}_{n-1}. f(x_0, \ldots, x_{i-1}, a_i, x_{i+1}, \ldots, x_{n-1}))$$

is M-sequential. Also such a function is *V-sequential* iff either it is a constant or there is an integer i (with $0 \leq i < n$) such that $y \sqsupseteq x$ and $(y)_i = (x)_i$ implies $f(y) = f(x)$. Which of the functions of several arguments we have considered are sequential in one or other of these senses? Show that if $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}, \mathbf{E}$ are all flat cpos then the two definitions are equivalent. Show that all sequential functions $p: \mathbf{T}^n \rightarrow \mathbf{T}$ are λ-definable using as constants *tt*, *ff* and *cond*.

13. Define the minimalisation operator $\overline{\mu}: (\mathbf{N} \rightarrow \mathbf{T}) \rightarrow \mathbf{T}$ in the typed λ-calculus, using the primitive sequential functions on $\mathbf{N}$ and $\mathbf{T}$. What about the function $\partial: (\mathbf{N} \rightarrow \mathbf{T}) \rightarrow \mathbf{T}$? What is the greatest existential quantifier $\overline{\exists}: (\mathbf{N} \rightarrow \mathbf{T}) \rightarrow \mathbf{T}$ you can define using the primitive functions and *parcond*? (See Exercise 1.14, 1.15).

14. We can define elements of **Tapes** by simultaneous fixed-points using functions defined by first-order terms with $(0 \cdot -)$, $(1 \cdot -)$ as consists such as:

$$x = 0 \cdot 0 \cdot 1 \cdot y,$$

$$y = 1 \cdot 0 \cdot 0 \cdot x.$$

In this way only ultimately periodic functions can be defined. What happens in the case of **Btrees** if we allow as constants numerals (for the corresponding numbers), *tip* and *mktree*?

15 (Bekič). Let $f: \mathbf{D} \times \mathbf{E} \rightarrow \mathbf{D}$, $g: \mathbf{D} \times \mathbf{E} \rightarrow \mathbf{E}$ be continuous functions. We can solve the simultaneous fixed-point equations:

$$x = f(x, y),$$

$$y = g(x, y).$$

by a method of elimination. First define a parameterised solution to the first equation by: $h(y) = \mu x. f(x, y)$, then substitute in the second to obtain the equation $y = g(h(y), y) = k(y)$, say. Then show that $\overline{x}$, $\overline{y}$ are the least simultaneous fixed-points of the above equations where $\overline{y} = \mu y. k(y)$ and $\overline{x} = h(\overline{y})$. What does this amount to when $f(x, y)$ is independent of y? What if, further, g does not depend on x?

16. Theorem 1 in Chapter 1 shows the validity of the following principle of *recursion induction* (McCarthy, Park). Let $f: \mathbf{D} \rightarrow \mathbf{D}$ be continuous. Then:

$$\forall x: \mathbf{D}. (f(x) \sqsubseteq x \Rightarrow Yf \sqsubseteq x).$$

Establish this principle using fixed-point induction.

17. Give examples of non-inclusive predicates on $\mathbf{P}\boldsymbol{\omega}$, **Fun**. Show that if $\mathbf{P} \subseteq \mathbf{D}$ is inclusive $\mathbf{D} - \mathbf{P}$ need not be; if $\mathbf{R} \subseteq \mathbf{D} \times \mathbf{E}$ is inclusive $\forall y. \mathbf{R}(x, y)$ need not be; if $\mathbf{R} \subseteq \mathbf{D} \times \mathbf{E}$, $\mathbf{S} \subseteq \mathbf{E} \times \mathbf{F}$ are inclusive, $\mathbf{S} \circ \mathbf{R}$ need not be.

18. Show that $\mathbf{R} \subseteq \prod_{i<n} \mathbf{D}_i$ is inclusive iff it is inclusive in each argument taken separately.

19. Show, using fixed-point induction, the following principle of simultaneous fixed-point induction. Let $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ be cpos, $\mathbf{R} \subseteq \prod_{i<n} \mathbf{D}_i$ be inclusive and $f_i: \prod_{i<n} \mathbf{D}_i \to \mathbf{D}_i$ $(i < n)$ be continuous functions. Then:

$$[\mathbf{R}(\perp, \ldots, \perp) \wedge (\forall x_0: \mathbf{D}_0, \ldots, x_{n-1}: \mathbf{D}_{n-1}.\mathbf{R}(x_0, \ldots, x_{n-1}) \Rightarrow \mathbf{R}(f_0(x_0, \ldots, x_{n-1}), \ldots, f_{n-1}(x_0, \ldots, x_{n-1})))] \\ \Rightarrow \mathbf{R}(Y(\langle f_0, \ldots, f_{n-1}\rangle)).$$

20. Let $f: \mathbf{D} \to \mathbf{D}$, $g: \mathbf{D} \to \mathbf{D}$ be continuous functions. Show $f(Y(g \circ f)) = Y(f \circ g)$. Show that if $f\perp = g\perp$ and f and g commute ($f \circ g = g \circ f$) then $Yf = Y(f \circ g) = Yg$. This and the next few exercises should be tried using fixed-point induction or recursion induction, as appropriate. (Try Exercise 15 this way too).

21. Let $f: \mathbf{D} \to \mathbf{D}$, $g: \mathbf{D} \to \mathbf{D}$ be continuous functions. Show that if $f \circ f \circ g = g \circ f$ and $f\perp = g\perp$ then $Yf = Yg$. [*Warning* This is hard by fixed-point induction. The only known proof needs an inclusive predicate defined by the conjunction of three inequalities (order relations of the form $t \sqsubseteq u$).]

22. Define $h: \mathbf{N} \to \mathbf{N}$ recursively by: $h(x) = h(x) + 1$. Show $h = \perp$. For $a: \mathbf{D}$, $h: \mathbf{D} \to \mathbf{D}$, and $p: \mathbf{D} \to \mathbf{T}$ define $g: \mathbf{D}^2 \to \mathbf{D}$ recursively by: $g(x, y) = (p(x) \to g(x, h(y)) \mid a)$. Show $g = \perp$.

23. For $p: \mathbf{D} \to \mathbf{T}$, $h: \mathbf{D} \to \mathbf{D}$, $k: \mathbf{D} \to \mathbf{D}$ define $f, g: \mathbf{D}^2 \to \mathbf{D}$ recursively by:

$$f(x, y) = (p(x) \to y \mid h(f(k(x), y))),$$
$$g(x, y) = (p(x) \to y \mid g(k(x), h(y))).$$

Show $h(f(x, y)) = f(x, hy)$ and then show $f = g$.

24. Suppose $a: \mathbf{D}$, $p: \mathbf{D} \to \mathbf{T}$, $d: \mathbf{D} \to \mathbf{D}$, $b: \mathbf{D} \times \mathbf{D} \to \mathbf{D}$ are continuous functions which satisfy the following axioms:

Associativity	$\forall x, y, z: \mathbf{D}.b(x, b(y, z)) = b(b(x, y), z).$
Commutativity	$\forall x, y: \mathbf{D}.b(x, y) = b(y, x).$
Left Identity	$\forall x: \mathbf{D}.b(a, x) = x.$
Bistrictness	$\forall x: \mathbf{D}.b(\perp, x) = b(x, \perp) = \perp.$

Define $f: \mathbf{D} \to \mathbf{D}$, $g: \mathbf{D}^2 \to \mathbf{D}$ recursively by:

$$f(x) = (p(x) \to a \mid b(x, f(d(x)))),$$
$$g(x, y) = (p(x) \to y \mid g(d(x), b(x, y))).$$

Show $g(x, y) = b(f(x), y)$ and deduce $f(x) = g(x, a)$. Interpret this result when $\mathbf{D} = \mathbf{N}$, $a = 1$, $p = Z$, $d = (\cdot - 1)$ and $b = \times$. Why does fixed-point induction on f not help to show that $\partial \circ f(x) = \partial(x)$ for all x in $\mathbf{D}$ (termination)? What does help?

25. Let $\mathbf{D}$ be a cpo. Show that $Y_{\mathbf{D}} = \mu F: ((\mathbf{D} \to \mathbf{D}) \to \mathbf{D}).\lambda f: (\mathbf{D} \to \mathbf{D}).f(Ff)$.

26. If $\mathbf{D}$ and $\mathbf{E}$ are partial orders, we say a function $f: \mathbf{D} \to \mathbf{E}$ is *continuous* iff it is monotonic and preserves lubs of increasing ω-chains (that is, if $\langle x_n \rangle_{n \in \omega}$ is an increasing ω-chain in $\mathbf{D}$ such that $\bigsqcup_{\mathbf{D}} x_n$ exists then $f(\bigsqcup_{\mathbf{D}} x_n) = \bigsqcup_{\mathbf{E}} f(x_n)$). Is it the case that if $\mathbf{D}$ is any partial order with a least element, but which is not a cpo then there is a continuous function $f: \mathbf{D} \to \mathbf{D}$ with no fixed-point? This seems to be an open question — a solution would help show the necessity of considering cpos.

27. A *fixed-point operator of type* $\mathbf{D}$ is a function $F_{\mathbf{D}}: (\mathbf{D} \to \mathbf{D}) \to \mathbf{D}$ (not necessarily continuous) such that for all continuous $f: \mathbf{D} \to \mathbf{D}$ we have $f(Ff) = Ff$; it is *invariant* iff for every automorphism, $\theta: \mathbf{D} \to \mathbf{D}$, we always have: $Ff = \theta^{-1}(F(\theta \circ f \circ \theta^{-1}))$. What are the (continuous, invariant) fixed-point operators on $\mathbf{T}^{\omega}$? Show $Y_{\mathbf{D}}$ is always invariant.

28. A strict *monic* is a monic in $\mathcal{CPO}_\perp$, i.e. a strict continuous function, $m: \mathbf{D} \to_\perp \mathbf{E}$ such that for any $f, g: \mathbf{C} \to_\perp \mathbf{D}$, $m \circ f = m \circ g$ implies $f = g$. A strict *order* monic is a strict continuous function $m: \mathbf{D} \to_\perp \mathbf{E}$ such that for any $f, g: \mathbf{C} \to_\perp \mathbf{D}$, $m \circ f \sqsubseteq m \circ g$ implies $f \sqsubseteq g$. Show that $m: \mathbf{D} \to_\perp \mathbf{E}$ is a strict monic iff for all x, y in $\mathbf{D}$, $m(x) = m(y)$ implies $x = y$; show it is a strict order monic iff for all x, y in $\mathbf{D}$, $m(x) \sqsubseteq m(y)$ implies $x \sqsubseteq y$. Let $\mathbf{P} \subseteq \mathbf{D}$ be an inclusive predicate such that $\mathbf{P}(\perp)$. Show that $\mathbf{P}$ is a cpo under the order inherited from $\mathbf{D}$ and that the natural inclusion map $\iota: \mathbf{P} \to_\perp \mathbf{D}$ is a strict order monic. Show that $m: \mathbf{C} \to_\perp \mathbf{D}$ is a strict order monic iff it factors as $\mathbf{C} \xrightarrow{\theta}_\perp \mathbf{P} \xrightarrow{\iota}_\perp \mathbf{D}$ (i.e. $m = \iota \circ \theta$) where $\mathbf{P} \subseteq \mathbf{D}$ is an inclusive predicate on $\mathbf{D}$ such that $\mathbf{P}(\perp)$ and θ is an isomorphism. We conclude that the strict order monics are the categorical equivalent of inclusive predicates containing $\perp$. What about the inclusive predicates?

29. Show that fixed-point induction can be reformulated in the following way: For every $f\colon \mathbf{D} \to \mathbf{D}$, if $m\colon \mathbf{C} \to \mathbf{D}$ is a strict order monic and $g\colon \mathbf{C} \to \mathbf{C}$ is a (necessarily unique) continuous function such that the following diagram commutes:

$$\begin{tikzcd} \mathbf{C} \arrow[r, "g"] \arrow[d, "m"'] & \mathbf{C} \arrow[d, "m"] \\ \mathbf{D} \arrow[r, "f"'] & \mathbf{D} \end{tikzcd}$$

then $m(Yg) = Yf$.

30. A collection, $F_{\mathbf{D}}$, of fixed-point operators (one for each cpo $\mathbf{D}$) is *uniform* iff wherever $f\colon \mathbf{D} \to \mathbf{D}$, $g\colon \mathbf{E} \to \mathbf{E}$ and $h\colon \mathbf{D} \to_{\perp} \mathbf{E}$ are such that the following diagram commutes:

$$\begin{tikzcd} \mathbf{D} \arrow[r, "f"] \arrow[d, "h"'] & \mathbf{D} \arrow[d, "h"] \\ \mathbf{E} \arrow[r, "g"'] & \mathbf{E} \end{tikzcd}$$

then $h(F_{\mathbf{D}}f) = F_{\mathbf{D}}g$. Show that the collection, $Y_{\mathbf{D}}$, is uniform and that it is the only uniform collection. [*Hint* Use Exercise 29.] This is further justification of our choice of least fixed-points. Show that invariance of each of a collection of fixed-point operators (one for each cpo) can be rephrased as uniformly w.r.t. automorphisms. It is an open question whether Y is the only collection of fixed-point operators invariant over the embeddings (see Chapter 4).

31. Show that $\mathcal{C}[\![\mathbf{while}\ b\ \mathbf{do}\ s]\!] = \mathcal{C}[\![\mathbf{if}\ b\ \mathbf{then}\ (s;\ \mathbf{while}\ b\ \mathbf{do}\ s)\ \mathbf{else\ dummy}]\!]$. Define the semantics of a construct $\mathbf{until}\ b\ \mathbf{do}\ s$ and show the equivalence, $\mathcal{C}[\![\mathbf{while}\ b\ \mathbf{do}\ s]\!] = \mathcal{C}[\![\mathbf{until}\ \neg b\ \mathbf{do}\ s]\!]$ (with the evident semantics for $\neg b$).

32. Define a syntactic substitution, $[e_0/x]e_1$ of expressions for identifiers in expressions and prove that $\mathcal{E}[\![\mathbf{let}\ x\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] = \mathcal{E}[\![[e_0/x]e_1]\!]$. Define a syntactic substitution $[e_0/f_n(x_0, \ldots, x_{n-1})]e_1$ of expressions for function symbols with indicated identifiers as parameters in expressions so that for example, $[e_0/f_1(x_0)](f_1(e_1)) = [[e_0/f_1(x_0)]e_1/x_0]e_0$. Show that

$$\mathcal{E}[\![\mathbf{let}\ f_n(x_0, \ldots, x_{n-1})\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] = \mathcal{E}[\![[e_0/f_n(x_0, \ldots, x_{n-1})]e_1]\!]$$

and also that

$$\mathcal{E}[\![\mathbf{letrec}\ f_n(x_0, \ldots, x_{n-1})\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] = \mathcal{E}[\![\mathbf{letrec}\ f_n(x_0, \ldots, x_{n-1})\ \mathbf{be}\ e_0\ \mathbf{in}\ [e_0/f_n(x_0, \ldots, x_{n-1})]e_1]\!].$$

These last two equations show why our semantics is "call-by-name" — that Algol copy rule is valid. [*Warning* Make sure that free identifiers or function symbols are not captured by binding mechanisms when defining your syntactic substitution.]

33. Extend *Imp* to allow assignments and recursive assignments to functions thus:

$$s ::= f_n(x_0, \ldots, x_{n-1}) := e \mid f_n(x_0, \ldots, x_{n-1}) \Rightarrow e.$$

This is only possible because **FEnv** is part of the state. Given an alternative semantics of *Imp* without this extension in which **FEnv** is part of the environment so that $\mathbf{S} = \mathbf{IEnv}$ and $\mathcal{C}$ has type $\mathcal{C}\colon \mathbf{Stat} \to (\mathbf{FEnv} \to \mathbf{C})$ (where $\mathbf{C} = \mathbf{S} \to \mathbf{S}$).

34. Write a few example programs and prove that they denote what you think they ought to.

35. Consider the combination of *Imp* and *Dec* given in Chapter 1. Can you define a syntactic substitution of expressions for identifiers in statements so that $\mathcal{C}[\![x := e;\ s]\!]$ and $\mathcal{C}[\![[e/x]s]\!]$ bear a suitably close relationship?

3. Other Constructions

In the last chapter we were able to give the semantics of recursively defined functions which take their arguments by call-by-name; this was accomplished with the aid of the product and function-space constructions. Now we will consider call-by-value as a prop for introducing the strict versions of these constructions. This will entail consideration of $\mathcal{CPO}_\perp$ rather than $\mathcal{CPO}$.

Under the call-by-value regime, if the evaluation of an expression, e, in **Exp** does not terminate then neither does that of $f_1(e)$. So we expect that f_1 should denote a *strict* function. To ensure this we will define the cpo, $(\mathbf{V} \to_\perp \mathbf{V})$ of the strict functions and when f_1 is recursively defined we will take its denotation as the least fixed-point of a suitable functional, $F\colon (\mathbf{V} \to_\perp \mathbf{V}) \to (\mathbf{V} \to_\perp \mathbf{V})$. In the case of functions of n arguments, if the evaluation of any one of the expressions $e_0, \ldots, e_{n-1}$ does not terminate then, under the call-by-value regime, neither does the evaluation of $f_n(e_0, \ldots, e_{n-1})$ terminate. Thus the denotation of f_n should, perhaps, be an *n-strict* function, $f_n\colon \mathbf{V}^n \to_\perp \mathbf{V}$ (that is, if for some i, with $i < n$, $(x)_i = \perp$ then $f_n(x) = \perp$). For recursively defined functions this would lend us to consider cpos of n-strict functions of n arguments. In the case of arbitrary continuous functions of n-arguments we consider products such as $\mathbf{V}^n$, and unary functions $\mathbf{V}^n \to \mathbf{V}$ rather than create special cpos of functions of n-arguments. In just the same way we will consider strict products and unary strict functions.

Let $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ be n cpos. Their *strict product* (= *smash* product), $\bigotimes_{i<n} \mathbf{D}_i$ is the subset, $\{x \in \prod_{i<n} \mathbf{D}_i \mid (\forall i < n.(x)_i \neq \perp) \vee (x = \perp)\}$, of $\prod_{i<n} \mathbf{D}_i$ with the coordinatewise ordering inherited from $\prod_{i<n} \mathbf{D}_i$. Sometimes we write $\mathbf{D_0} \otimes \cdots \otimes \mathbf{D}_{n-1}$ for $\bigotimes_{i<n} \mathbf{D}_i$ or $\mathbf{D}^{ⓝ}$ when the $\mathbf{D}_i$ are all the same cpo, $\mathbf{D}$. The smash product is a cpo; indeed it has the same least element and the same lubs of increasing sequences as the Cartesian product. Thus it is even a subcpo of the Cartesian product and so the natural inclusion map $\iota\colon \bigotimes_{i<n} \mathbf{D}_i \to_\perp \prod_{i<n} \mathbf{D}_i$ is continuous. There is another natural continuous map in the opposite direction, namely $\otimes\colon \prod_{i<n} \mathbf{D}_i \to_\perp \bigotimes_{i<n} \mathbf{D}_i$ where:

$$\otimes(x) = \begin{cases} x & (\forall i < n.(x)_i \neq \perp), \\ \perp & (\text{otherwise}). \end{cases}$$

Note that $\otimes$ is n-strict. When we look at the universal property for the smash product we shall see that $\otimes$ is the universal n-strict map.

Let $\mathbf{D}$ and $\mathbf{E}$ be cpos. Their *strict* function space, $\mathbf{D} \to_\perp \mathbf{E}$, is the set $\{f\colon \mathbf{D} \to \mathbf{E} \mid f(\perp) = \perp\}$ of strict continuous functions from $\mathbf{D}$ to $\mathbf{E}$, with the pointwise order inherited from $\mathbf{D} \to \mathbf{E}$. Again we find that $(\mathbf{D} \to_\perp \mathbf{E})$ is a subcpo of $(\mathbf{D} \to \mathbf{E})$, meaning that it has the same least element and lubs of increasing sequences. Again the natural inclusion map $\iota\colon (\mathbf{D} \to_\perp \mathbf{E}) \to (\mathbf{D} \to \mathbf{E})$ is continuous. We now introduce our last piece of metalanguage:

E10. If e is an expression of type $\mathbf{E}$ and x is a variable of type $\mathbf{D}$ then $(\lambda_\perp x\colon \mathbf{D}.e)$ is an expression of type $(\mathbf{D} \to_\perp \mathbf{E})$ which is to have the same denotation as $(\lambda x\colon \mathbf{D}.(\partial x \to e \mid \perp))$ (but note the types are different). When $\mathbf{D}$ is clear from the context $(\lambda_\perp x\colon \mathbf{D}.e)$ may be abbreviated to $(\lambda_\perp x.e)$.

For example this strict kind of abstraction allows us to define a map $\mathit{strict}\colon (\mathbf{D} \to \mathbf{E}) \to (\mathbf{D} \to_\perp \mathbf{E})$ by:

$$\mathit{strict} = \lambda f\colon \mathbf{D} \to \mathbf{E}.\lambda_\perp d\colon \mathbf{D}.f(d).$$

Whenever $\mathbf{A}$ is a subcpo of $\mathbf{B}$ we will include an anonymous constant, ι, for the inclusion map $\iota\colon \mathbf{A} \to \mathbf{B}$ in our metalanguage. Often we will write e instead of $\iota(e)$ when no confusion thereby results. For example when $f\colon \mathbf{D} \to_\perp \mathbf{E}$ and $x\colon \mathbf{D}$, we might write $f(x)$ instead of $\iota(f)(x)$. Again we introduce the abbreviation $(\lambda_\perp x_0\colon \mathbf{D}_0, \ldots, x_{n-1}\colon \mathbf{D}_{n-1}.e)$ to stand for:

$$\lambda_\perp w\colon \bigotimes_{i<n} \mathbf{D}_i.((\lambda_\perp x_0\colon \mathbf{D}_0.(\ldots(\lambda_\perp x_{n-1}\colon \mathbf{D}_{n-1}.e)\ldots))((w)_0)\ldots((w)_{n-1}));$$

when the $\mathbf{D}_i$ are clear from the context we might just write $(\lambda_\perp x_0, \ldots, x_{n-1}.e)$ instead. Thus, for example, we can define the composition map, $\circ$, of strict functions, of type $((\mathbf{E} \to_\perp \mathbf{F}) \otimes (\mathbf{D} \to_\perp \mathbf{E})) \to_\perp (\mathbf{D} \to_\perp \mathbf{F})$ by $(\lambda_\perp g, f.\lambda_\perp d.g(f(d)))$. Finally, we will often write $e_0 \otimes \cdots \otimes e_{n-1}$ as an abbreviation for $\otimes(e_0, \ldots, e_{n-1})$.

Turning to categorical considerations we first observe that $\otimes: \prod_{i<n} \mathbf{D}_i \rightarrow_\perp \bigotimes_{i<n} \mathbf{D}_i$ is universal in the sense that if $f: \prod_{i<n} \mathbf{D}_i \rightarrow_\perp \mathbf{E}$ is any n-strict continuous map then there is a unique strict continuous $g: \bigotimes_{i<n} \mathbf{D}_i \rightarrow_\perp \mathbf{E}$ such that the following diagram commutes:

$$\begin{tikzcd} \prod_{i<n} \mathbf{D}_i \arrow[d, "\otimes"'] \arrow[dr, "f"] & \\ \bigotimes_{i<n} \mathbf{D}_i \arrow[r, "g"'] & \mathbf{E} \end{tikzcd}$$

For if this diagram commutes for a given g we have $g = f \circ \iota$ as: $g(w) = g(\otimes(\iota(w)))$ (as $\otimes \circ \iota = id$) $= f(\iota(w))$. This proves uniqueness. For existence let g be the strict function $f \circ \iota$. Then if $w \in \prod_{i<n} \mathbf{D}_i$ then either $(w)_i = \perp$ for some i, with $i < n$, or not. In the first case $f(w) = \perp$ and $g \circ \otimes(w) = g(\perp) = \perp$; in the second case $g \circ \otimes(w) = (f \circ \iota)(w) = f(w)$, proving existence. This universal property shows how n-strict functions $f: \prod_{i<n} \mathbf{D}_i \rightarrow_\perp \mathbf{E}$ are to be replaced by strict functions, $g: \bigotimes_{i<n} \mathbf{D}_i \rightarrow_\perp \mathbf{E}$ (see Exercise 1).

One now easily sees that the following external axioms characterise this universal property:

External Axiomatisation.

1. $\forall f: \prod_{i<n} \mathbf{D}_i \rightarrow_\perp \mathbf{E}.(f\ n\text{-strict}) \Rightarrow f = (f \circ \iota) \circ \otimes$.
2. $\forall g: \bigotimes_{i<n} \mathbf{D}_i \rightarrow_\perp \mathbf{E}.g = (g \circ \otimes) \circ \iota$.

Internal Axiomatisation.

1. $\forall x_0: \mathbf{D}_0, \ldots, \forall x_{n-1}: \mathbf{D}_{n-1}.\pi_i(x_0 \otimes \cdots \otimes x_{n-1}) = (((\partial x_0) \text{ and } \ldots \text{ and } (\partial x_{n-1})) \rightarrow x_i \mid \perp)$.
2. $\forall w: \bigotimes_{i<n} \mathbf{D}_i.w = \pi_0(w) \otimes \cdots \otimes \pi_{n-1}(w)$.

These axioms are clearly true. To see they imply the external ones we begin by showing that they imply $\otimes$ is n-strict. It is given that $\otimes$ is strict (which is also a consequence of 2). Suppose $x_0: \mathbf{D}_0, \ldots, x_{n-1}: \mathbf{D}_{n-1}$ and some x_i is $\perp$. Then by part 1, $\pi_i(x_0 \otimes \cdots \otimes x_{n-1}) = \perp$ for all i ($i < n$) and so by part 2, $x_0 \otimes \cdots \otimes x_{n-1} = \perp$. For part 1, suppose $x_0: \mathbf{D}_0, \ldots, x_{n-1}: \mathbf{D}_{n-1}$. If some x_i is $\perp$ then: $(f \circ \iota) \circ \otimes(x_0, \ldots, x_{n-1}) = f \circ \iota(\perp) = \perp = f(x_0, \ldots, x_{n-1})$ as f is n-strict. Otherwise:

$$\begin{aligned} (f \circ \iota) \circ \otimes(x_0, \ldots, x_{n-1}) &= f(\pi_0(x_0 \otimes \cdots \otimes x_{n-1}), \ldots, \pi_{n-1}(x_0 \otimes \cdots \otimes x_{n-1})) \\ &= f(x_0, \ldots, x_{n-1}) \qquad \text{by Axiom 1.} \end{aligned}$$

Finally we show that $\otimes \circ \iota = id$, to prove the second external axiom. We have:

$$\otimes \circ \iota(w) = \otimes(\pi_0(w), \ldots, \pi_{n-1}(w)) = w \qquad \text{by Axiom 2.}$$

Note that if $\mathbf{D}$ and $\mathbf{E}$ are flat cpos so is $\mathbf{D} \otimes \mathbf{E}$. For example $\mathbf{T} \otimes \mathbf{T}$ is drawn in Figure VIII:

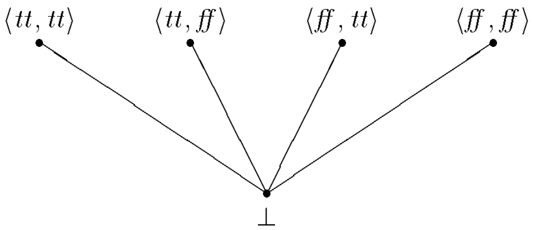


Figure VIII: The cpo $\mathbf{T} \otimes \mathbf{T}$.

This idea will help us to consider syntax as a cpo, in a systematic way, as mentioned above. Note too, that it follows that when $f: \mathbf{X}_0 \times \cdots \times \mathbf{X}_{n-1} \rightarrow \mathbf{Y}$ is a function on sets, then f has a *unique* extension to a continuous function, $\hat{f}: (\mathbf{X}_0)_\perp \otimes \cdots \otimes (\mathbf{X}_{n-1})_\perp \rightarrow_\perp \mathbf{Y}_\perp$ and the extension, $f_\perp$, considered in Chapter 2 is just $\iota(\hat{f}) \circ \otimes$. Conversely, every n-strict function on flat cpos is the extension of an ordinary function on the corresponding sets. So we see that the strict constructions allow us to import functions of several arguments over sets more accurately than the ordinary product and function space constructions. But, of course, we still need many continuous functions other than the n-strict ones; it is absolutely necessary to have $cond: \mathbf{T} \times \mathbf{D} \times \mathbf{D} \rightarrow \mathbf{D}$ for instance.

We conclude the discussion of the smash product construction by showing that it can be considered as a covariant functor, $\bigotimes : \mathcal{CPO}_\perp^n \to \mathcal{CPO}_\perp$ by defining its action on morphisms $f_i\colon \mathbf{D}_i \to_\perp \mathbf{E}_i$ $(i < n)$ by:

$$\bigotimes_{i<n} f_i = \otimes \circ (\prod_{i<n} f_i) \circ \iota$$

where we are considering $\prod$ as a functor, $\prod\colon \mathcal{CPO}_\perp^n \to \mathcal{CPO}_\perp$ (see Exercise 2.6). In this case the functor laws are:

(i) $$\bigotimes_{i<n} id_{\mathbf{D}_i} = id_{\bigotimes \mathbf{D}_i},$$

(ii) $$\bigotimes_{i<n}(g_i \circ f_i) = (\bigotimes_{i<n} g_i) \circ (\bigotimes_{i<n} f_i).$$

The universal property for the strict function space, $\mathbf{D} \to_\perp \mathbf{E}$ is specified in exactly the same way as that for the function space, $\mathbf{D} \to \mathbf{E}$ but using the smash product in place of the Cartesian one. Define $\mathit{apply}\colon ((\mathbf{D} \to_\perp \mathbf{E}) \otimes \mathbf{D}) \to_\perp \mathbf{E}$ by:

$$\mathit{apply} = \lambda_\perp f\colon \mathbf{D} \to_\perp \mathbf{E}.x\colon \mathbf{D}.f(x).$$

Then if $f\colon \mathbf{F} \otimes \mathbf{D} \to_\perp \mathbf{E}$ is any continuous function there is a unique mediating $g\colon \mathbf{F} \to_\perp (\mathbf{D} \to_\perp \mathbf{E})$ such that the following diagram commutes:

$$\begin{tikzcd}
(\mathbf{D} \to_\perp \mathbf{E}) \otimes \mathbf{D} \arrow[d, "\mathit{apply}"'] & \mathbf{F} \otimes \mathbf{D} \arrow[l, "g \otimes id_{\mathbf{D}}"'] \arrow[dl, "f"] \\
\mathbf{E} &
\end{tikzcd}$$

In fact g must be $\mathit{Curry}(f)$ where $\mathit{Curry}\colon ((\mathbf{F} \otimes \mathbf{D}) \to_\perp \mathbf{E}) \to_\perp (\mathbf{F} \to_\perp (\mathbf{D} \to_\perp \mathbf{E}))$ is defined by:

$$\mathit{Curry} = \lambda_\perp f\colon (\mathbf{F} \otimes \mathbf{D}) \to_\perp \mathbf{E}.\lambda_\perp v\colon \mathbf{F}.\lambda_\perp d\colon \mathbf{D}.f(v \otimes d).$$

External Axiomatisation.

1. $\forall f\colon (\mathbf{F} \otimes \mathbf{D}) \to_\perp \mathbf{E}.f = \mathit{apply} \circ (\mathit{Curry}(f) \otimes id_{\mathbf{D}})$.
2. $\forall g\colon \mathbf{F} \to_\perp (\mathbf{D} \to_\perp \mathbf{E}).g = \mathit{Curry}(\mathit{apply} \circ (g \otimes id_{\mathbf{D}}))$.

Internal Axiomatisation.

1. $\beta_\perp$*-reduction*: $(\lambda_\perp x.e)(x) = (\partial x \to e \mid \perp)$.
2. $\eta_\perp$*-reduction*: $\lambda_\perp x.f(x) = f$ $\qquad (f\colon \mathbf{D} \to_\perp \mathbf{E}, x\colon \mathbf{D})$.

Note that 1 and 2 are largely axioms on ι (assuming ordinary β-reduction and η-reduction). Thus 1 may be rephrased as: $\iota(\lambda_\perp x.e) = \lambda x.(\partial x \to e \mid \perp)$ and 2 may be rephrased as: $\iota(f)(\perp) = \perp$.

As a functor, $\to_\perp$, is defined just like $\to$. For $f\colon \mathbf{B} \to_\perp \mathbf{A}$, $f'\colon \mathbf{A}' \to_\perp \mathbf{B}'$ we define $(f \to_\perp f')\colon (\mathbf{A} \to_\perp \mathbf{A}') \to_\perp (\mathbf{B} \to_\perp \mathbf{B}')$ by:

$$(f \to_\perp f') = \lambda_\perp x\colon \mathbf{A} \to_\perp \mathbf{A}'.f' \circ x \circ f.$$

We obtain a functor, $\to_\perp\colon \mathcal{CPO}_\perp^2 \to \mathcal{CPO}_\perp$ which is contravariant in its first argument and covariant in its second argument so that the functor laws are:

(i) $$id_{\mathbf{D}} \to_\perp id_{\mathbf{E}} = id_{(\mathbf{D} \to_\perp \mathbf{E})},$$

(ii) $$(f \circ g) \to_\perp (g' \circ f') = (g \to_\perp g') \circ (f \to_\perp f').$$

for f, f', g, g' of the appropriate types. In category-theoretic jargon what we have essentially shown about $\otimes$, $\to_\perp$ and their relation to each other is that $\mathcal{CPO}_\perp$ equipped with $\otimes$ and $\to_\perp$ is a *closed* category.

It is now quite straightforward to change the semantics of *Dec* to correspond to call-by-value. We keep the same syntax as in Chapter 2. For the semantic domains we again take $\mathbf{V}$ to be a fixed cpo of values; we can now model $\mathbf{IEnv}$ more accurately as $(\mathbf{Id}_\perp \rightarrow_\perp \mathbf{V}_\perp)$ and $\mathbf{FEnv}$ should now be $\prod_{j<m}(\mathbf{Fun}_j)_\perp \rightarrow_\perp (\mathbf{V}^{\textcircled{j}} \rightarrow_\perp \mathbf{V})$. We assume the basic operation also take their arguments by value and so we have $\mathcal{I}_i\colon \mathbf{Op}_i \rightarrow (\mathbf{V}^{\textcircled{i}} \rightarrow_\perp \mathbf{V})$ $(i < n)$. Next $\mathcal{B}\colon \mathbf{BExp} \rightarrow ((\mathbf{IEnv} \times \mathbf{FEnv}) \rightarrow \mathbf{T})$ and finally $\mathcal{E}\colon \mathbf{Exp} \rightarrow ((\mathbf{IEnv} \times \mathbf{FEnv}) \rightarrow \mathbf{V})$ is defined by structural induction:

(i) (as in Chapter 2);

(ii)$_\mathrm{i}$ $\mathcal{E}[\![o_i(e_0, \ldots, e_{n-1})]\!] = \lambda\rho\colon \mathbf{IEnv}, \varphi\colon \mathbf{FEnv}.\mathcal{I}[\![o_i]\!](\mathcal{E}[\![e_0]\!](\rho,\varphi) \otimes \cdots \otimes \mathcal{E}[\![e_{i-1}]\!](\rho,\varphi))$ $\qquad (i < n)$;

(iii) (as in Chapter 2);

(iv) (as in Chapter 2);

(v)$_\mathrm{j}$ $\mathcal{E}[\![\mathbf{let}\ f_j(x_0, \ldots, x_{j-1})\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] =$

$$\lambda\rho\colon \mathbf{IEnv}, \varphi\colon \mathbf{FEnv}.\mathcal{E}[\![e_1]\!](\rho, \varphi[(\lambda_\perp w\colon \mathbf{V}^{\textcircled{j}}.\mathcal{E}[\![e_0]\!](\rho[(w)_0/x_0]\ldots[(w)_{j-1}/x_{j-1}],\varphi))/f_j]) \qquad (j < m);$$

(vi)$_\mathrm{j}$ $\mathcal{E}[\![\mathbf{letrec}\ f_j(x_0, \ldots, x_{j-1})\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1]\!] =$

$$\lambda\rho\colon \mathbf{IEnv}, \varphi\colon \mathbf{FEnv}.\mathcal{E}[\![e_1]\!](\rho, \varphi[\mu f\colon \mathbf{V}^{\textcircled{j}} \rightarrow_\perp \mathbf{V}.(\lambda_\perp w\colon \mathbf{V}^{\textcircled{j}}.\mathcal{E}[\![e_0]\!](\rho[(w)_0/x_0]\ldots[(w)_{j-1}/x_{j-1}], \varphi[f/f_j]))/f_j]) \qquad (j < m).$$

Here $\cdot[\cdot/\cdot]\colon \mathbf{IEnv} \times \mathbf{V} \times \mathbf{Id}_\perp \rightarrow \mathbf{IEnv}$ is defined by:

$$\rho[v/x] = \lambda_\perp y\colon \mathbf{Id}_\perp.((v = x) \rightarrow v \mid \rho[\![y]\!])$$

and the definition of $\cdot[\cdot/\cdot]\colon (\mathbf{FEnv} \times (\mathbf{V}^{\textcircled{j}} \rightarrow_\perp \mathbf{V}) \times (\mathbf{Fun}_j)_\perp) \rightarrow \mathbf{FEnv}$ is left to the reader.

In order to introduce our next construction suppose we want to specify $\mathbf{V}$ so that expressions can have either integers or truth-values as their values and then we can identify $\mathbf{BExp}$ with $\mathbf{Exp}$. Then we expect that an expression evaluates to an integer, or to a truth-values or else evaluation does not terminate. So we expect $\mathbf{V}$ to be as in Figure IX:

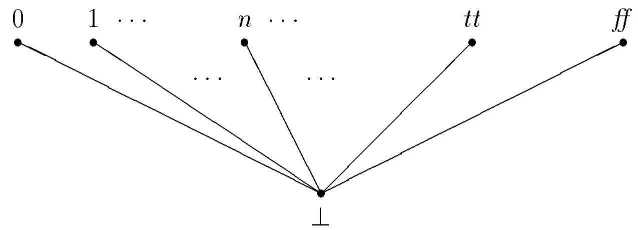


Figure IX: The cpo $\mathbf{N} + \mathbf{T}$.

This cpo is called the (coalesced) sums of $\mathbf{N}$ and $\mathbf{T}$ and is useful whenever values of identifiers can be in any of several different cpos — that is they are, in a sense, of ambiguous type. We now turn to the general construction.

Let $\mathbf{D}_0$, ..., $\mathbf{D}_{n-1}$ be cpos. Their *coalesced sum* (= *amalgamated* sum), $\sum_{i<n} \mathbf{D}_i$ is the set:

$$\left(\bigcup_{i<n}\{\langle i, d\rangle \mid d \in \mathbf{D}_i \wedge d \neq \perp\}\right) \cup \{\perp\}$$

with the order

$$x \sqsubseteq y \quad \text{iff} \quad (x = \perp) \text{ or } (\exists i < n.\exists d, d' \in \mathbf{D}_i.d \sqsubseteq d' \wedge x = \langle i, d\rangle \wedge y = \langle i, d'\rangle).$$

Thus the order is inherited from the $\mathbf{D}_i$ with the additional requirement that the $\perp_{\mathbf{D}_i}$ are identified. A picture for the case $n = 2$ should help:

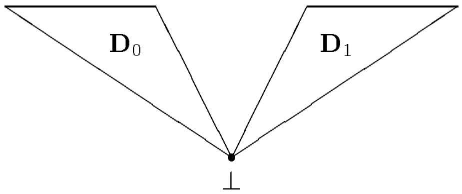


Figure X: The cpo $\mathbf{D}_0 + \mathbf{D}_1$.

Sometimes we write $\mathbf{D}_0 + \cdots + \mathbf{D}_{n-1}$ for $\sum_{i<n} \mathbf{D}_i$. Note that the sum is a cpo as all non-trivial increasing chains essentially lie within one or other of the $\mathbf{D}_i$. This is also the reason why the natural injection functions, $in_i: \mathbf{D}_i \to_\perp \sum_{i<n} \mathbf{D}_i$ $(i < n)$ are continuous where:

$$in_i(d) = \begin{cases} \perp & (d = \perp), \\ \langle i, d \rangle & (\text{otherwise}). \end{cases}$$

These injections (sometimes written $\amalg_i$) show how the sum is a compound type of cpo which contains all its summands; this leads to the following universal property. Let $f_i: \sum_{i<n} \mathbf{D}_i \to_\perp \mathbf{E}$ such that for all i with $i < n$ the following diagram commutes.

$$\begin{tikzcd} \mathbf{D}_i \arrow[d, "in_i"'] \arrow[dr, "f_i"] & \\ \sum_{i<n} \mathbf{D}_i \arrow[r, "h"'] & \mathbf{E} \end{tikzcd}$$

What this says is that $\sum_{i<n} \mathbf{D}_i$ is the categorical sum in $\mathcal{CPO}_\perp$; it is dual to the product. Clearly h is unique as we must have $h(\perp) = \perp$ and $h(\langle i, d \rangle) = h(in_i(d)) = f_i(d)$, when $d \neq \perp$, and this determines h as a strict continuous function which makes the diagram commute. We write h as $[f_0, \ldots, f_{n-1}]$ and one easily checks that $[\cdot, \ldots, \cdot]: (\prod_{i<n} (\mathbf{D}_i \to_\perp \mathbf{E})) \to ((\sum_{i<n} \mathbf{D}_i) \to_\perp \mathbf{E})$ is continuous. Note the case $n = 0$ when the sum $\sum_{i<0} \mathbf{D}_i$ is just $\mathbf{U}$ and we see that for any cpo $\mathbf{E}$ there is a unique morphism $[\,]: \mathbf{U} \to \mathbf{E}$ (namely $\perp$) showing that $\mathbf{U}$ is the *initial* object in $\mathcal{CPO}_\perp$.

External Axiomatisation.

1. $\forall f_0: \mathbf{D}_0 \to_\perp \mathbf{E}. \ldots \forall f_{n-1}: \mathbf{D}_{n-1} \to_\perp \mathbf{E}. \forall i < n. [f_0, \ldots, f_{n-1}] \circ in_i = f_i$ $\qquad (0 \le i < n)$.
2. $\forall h: \sum_{i<n} \mathbf{D}_i \to_\perp \mathbf{E}. h = [h \circ in_0, \ldots, h \circ in_{n-1}]$.

To give the internal axiomatisation it is convenient to introduce a few more functions. First projection functions $out_i: \sum_{i<n} \mathbf{D}_i \to_\perp \mathbf{D}_i$ $(i < n)$ are defined by:

$$out_i = [\perp, \ldots, \perp, \lambda_\perp x: \mathbf{D}_i. x, \perp, \ldots, \perp]$$

and next discriminator functions, $is_i: \sum_{i<n} \mathbf{D}_i \to_\perp \mathbf{T}$ are defined by:

$$is_i = [\lambda_\perp x_0. ff, \ldots, \lambda_\perp x_{i-1}. ff, \lambda_\perp x_i. tt, \lambda_\perp x_{i+1}. ff, \ldots, \lambda_\perp x_{n-1}. ff].$$

Sometimes out_i is written as $(\cdot \mid \mathbf{D}_i)$ when the $\mathbf{D}_i$ are all different.

Internal Axiomatisation.

1_i. $\forall x: \mathbf{D}_i. x = out_i(in_i(x))$ $\qquad (0 \le i < n)$.

2. $\forall x: \sum_{i<n} \mathbf{D}_i. x = (is_0(x) \to in_0(out_0(x)) \mid (is_1(x) \to in_1(out_1(x))$
$\qquad \mid \ldots \mid (is_{n-1} \to in_{n-1}(out_{n-1}(x)) \mid \perp) \ldots))$.

To obtain the external axioms from the internal ones we must define $[\cdot, \ldots, \cdot]$ in terms of the functions out_i, is_i by:

$$[\cdot, \ldots, \cdot] = \lambda w{:}\, (\prod_{i<n}(\mathbf{D}_i \rightarrow_\perp \mathbf{E})).\lambda_\perp x. \sum_{i<n} \mathbf{D}_i.(is_0(x) \rightarrow (w \downarrow 0)(out_0(x)) \mid (is_1(x) \rightarrow (w \downarrow 1)(out_1(x)) \mid \ldots$$
$$\mid (is_{n-1}(x) \rightarrow (w \downarrow n-1)(out_{n-1}(x)) \mid \perp) \ldots)).$$

Finally to consider $\sum$ as a covariant functor $\sum{:}\, \mathcal{CPO}_\perp^n \rightarrow \mathcal{CPO}_\perp$ we define, for $f_i{:}\, \mathbf{D}_i \rightarrow_\perp \mathbf{E}_i$ $(i < n)$:

$$\sum_{i<n} f_i = [in_0 \circ f_0, \ldots, in_{n-1} \circ f_{n-1}]$$

where $in_i{:}\, \mathbf{E}_i \rightarrow_\perp \sum \mathbf{E}_i$ and note the appropriate functor laws:

(i) $$\sum_{i<n} in_{\mathbf{D}_i} = id_{\sum \mathbf{D}_i},$$

(ii) $$\sum_{i<n}(g_i \circ f_i) = (\sum_{i<n} g_i) \circ (\sum_{i<n} f_i).$$

Returning to our example where $\mathbf{V} = \mathbf{N} + \mathbf{T}$ we can now see how to specify *Dec* in a bit more detail. We could take $\mathbf{Op}_0 = \{\mathbf{0}, \mathbf{true}, \mathbf{false}\}$, $\mathbf{Op}_1 = \{\mathbf{succ}, \mathbf{pred}, \mathbf{zero}\}$ and $\mathbf{Op}_n = \emptyset$ for $n > 1$. Then **BExp** could be given by the trivial grammar: **BExp** ::= **Exp**. The functions $\mathcal{I}_n$ $(n < 2)$ are given by:

$$\begin{aligned}
\mathcal{I}[\![\mathbf{0}]\!] &= inl(0);\\
\mathcal{I}[\![\mathbf{true}]\!] &= inr(tt);\\
\mathcal{I}[\![\mathbf{false}]\!] &= inr(f\!f);\\
\mathcal{I}[\![\mathbf{succ}]\!] &= inl \circ (+1) \circ outl;\\
\mathcal{I}[\![\mathbf{pred}]\!] &= inl \circ (-1) \circ outl;\\
\mathcal{I}[\![\mathbf{zero}]\!] &= inr \circ (z) \circ outl.
\end{aligned}$$

Note that when there are just two summands we often use (as here) *inl*, *inr*, *outl*, *outr*, *isl*, *isr* instead of respectively in_0, in_1, out_0, out_1, is_0, is_1. Finally $\mathcal{B}{:}\, \mathbf{BExp} \rightarrow ((\mathbf{IEnv} \times \mathbf{FEnv}) \rightarrow \mathbf{T})$ is specified by:

(i) $$\mathcal{B}[\![e]\!] = \lambda\rho{:}\, \mathbf{IEnv}, \varphi{:}\, \mathbf{FEnv}.outr(\mathcal{E}[\![e]\!](\rho, \varphi)).$$

The other details are filled in as usual with whatever parameter-calling discipline is desired. Note that $\mathcal{B}$ and $\mathcal{E}$ are mutually recursive, and are still well-defined by structural induction on (the parse trees of) expressions.

The final construction considered here — lifting — does not seem to be of great significance by itself but, rather, is often useful in combination with other constructions in order to achieve an accurate model of computation. Categorically, lifting allows continuous functions to be turned into strict continuous functions.

Let $\mathbf{D}$ be a cpo. Its *lifting* $\mathbf{D}_\perp$ is the set $\{\langle o, d\rangle \mid d \in \mathbf{D}\} \cup \{\perp\}$ with the ordering:

$$x \sqsubseteq y \quad \text{iff} \quad (x = \perp) \text{ or } (\exists d, d' \in \mathbf{D}. d \sqsubseteq d' \wedge x = \langle o, d\rangle \wedge y = \langle o, d'\rangle).$$

Clearly this makes $\mathbf{D}_\perp$ a cpo; roughly speaking one notes that all non-trivial chains in $\mathbf{D}_\perp$ are (eventually) in $\mathbf{D}$ and the lub is then inherited from $\mathbf{D}$. This observation also shows why the function $up{:}\, \mathbf{D} \rightarrow \mathbf{D}_\perp$ is continuous where:

$$up(d) = \langle o, d\rangle \quad (d \in \mathbf{D}).$$

The cpo $\mathbf{D}_\perp$ can be pictured thus:

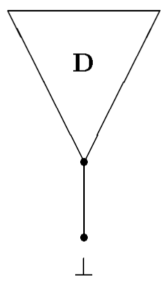


Figure XI: The cpo $\mathbf{D}_\perp$.

The universal property for $\mathbf{D}_\perp$ says that $up: \mathbf{D} \to \mathbf{D}_\perp$ is the basic example of a continuous function from $\mathbf{D}$: all others can be obtained from it via a factorisation through a strict function. For if $f: \mathbf{D} \to \mathbf{E}$ is any continuous function then there is a unique strict continuous function $h: \mathbf{D}_\perp \to_\perp \mathbf{E}$ such that the following diagram commutes:

$$\begin{tikzcd} \mathbf{D} \arrow[d, "up"'] \arrow[dr, "f"] & \\ \mathbf{D}_\perp \arrow[r, "h"'] & \mathbf{E} \end{tikzcd}$$

For we must have $h(\perp) = \perp$ and, for each member, d, of $\mathbf{D}$, $h(\langle o, d\rangle) = h(up(d)) = f(d)$. This proves uniqueness and existence too should now be clear. We write $\mathit{lift}(f)$ for h and indeed $\mathit{lift}: (\mathbf{D} \to \mathbf{E}) \to (\mathbf{D}_\perp \to \mathbf{E})$ is a continuous function.

External Axiomatisation.

1. $\forall f: \mathbf{D} \to \mathbf{E}. f = \mathit{lift}(f) \circ up$.
2. $\forall h: \mathbf{D}_\perp \to_\perp \mathbf{E}. h = \mathit{lift}(h \circ up)$.

For the internal axiomatisation it is convenient to define as auxiliary functions, $down: \mathbf{D}_\perp \to_\perp \mathbf{D}$ and the "termination predicate" $\partial: \mathbf{D}_\perp \to_\perp \mathbf{T}$ by:

$$down = \mathit{lift}(id_{\mathbf{D}}),$$
$$\partial = \mathit{lift}(\lambda x: \mathbf{D}. tt).$$

Internal Axiomatisation.

1. $\forall x: \mathbf{D}. x = down(up(x))$.
2. $\forall y: \mathbf{D}_\perp. y = (\partial y \to up(down(y)) \mid \perp)$.

To establish the external axiomatisation using the internal ones one notes that:

$$\mathit{lift} = \lambda f: \mathbf{D} \to \mathbf{E}. \lambda_\perp d: \mathbf{D}_\perp. (\partial d \to f(down(d)) \mid \perp).$$

To view $(\cdot)_\perp$ as a covariant functor: $(\cdot)_\perp: \mathcal{CPO} \to \mathcal{CPO}_\perp$ we define for $f: \mathbf{D} \to \mathbf{E}$:

$$f_\perp = \mathit{lift}(up \circ f).$$

The functor laws are:

(i) $$(id_{\mathbf{D}})_\perp = (id_{\mathbf{D}_\perp})$$

(ii) $$(g \circ f)_\perp = g_\perp \circ f_\perp$$

A typical use of lifting occurs when we do not want to amalgamate the bottom elements of the summands of $\sum_{i<n} \mathbf{D}_i$ in the case where they play different computational roles. Instead we might use the *separated* sum $\sum_{i<n} (\mathbf{D}_i)_\perp$; on occasion we might prefer *semi-separated* sums such as $\mathbf{D}_\perp + \mathbf{E}$.

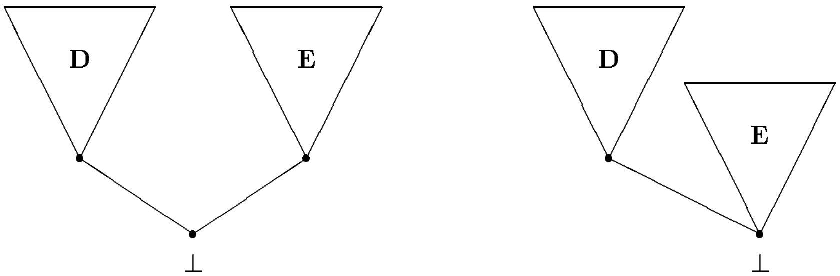


Figure XII: The cpos $\mathbf{D}_\perp + \mathbf{E}_\perp$ and $\mathbf{D}_\perp + \mathbf{E}$.

An example can be found in Exercise 21.

Exercises

1. Let $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}, \mathbf{E}$ be cpos. Define the cpo, $\mathit{Strict}_n(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}; \mathbf{E})$ of n-strict continuous functions and establish the isomorphism:

$$\mathit{Strict}_n(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}; \mathbf{E}) \cong (\bigotimes_{i<n} \mathbf{D}_i) \rightarrow_\bot \mathbf{E}.$$

Prove the functor laws for $\bigotimes$. What difficulties arise if you consider infinite smash products?

2 (Lehmann-Smyth). A morphism $f\colon \mathbf{D} \rightarrow_\bot \mathbf{E}$ is *very* strict if $f(x) = \bot$ iff $x = \bot$. Show that $\bigotimes$ is the categorical product in the subcategory of the very strict morphisms.

3. Show that $\bigotimes$ obeys commutative and associative laws. Show $\mathbf{U} \otimes \mathbf{D} \cong \mathbf{U}$ and $\mathbb{O} \otimes \mathbf{D} \cong \mathbf{D}$.

4. Let $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}, \mathbf{E}$ be cpos. A continuous function, $f\colon \prod_{i<n} \mathbf{D}_i \rightarrow \mathbf{E}$ is (m, n) strict iff whenever any m of $x_0, \ldots, x_{n-1}$ are $\bot$ then $f(x_0, \ldots, x_{n-1})$ is $\bot$ $(x_0\colon \mathbf{D}_0, \ldots, x_{n-1}\colon \mathbf{D}_{n-1})$. Define a cpo $\mathit{Strict}_{m,n}(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}; \mathbf{E})$ and a product $\mathit{Prod}_{m,n}(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1})$ and prove:

$$\mathit{Strict}_{m,n}(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}; \mathbf{E}) \cong \mathit{Prod}_{m,n}(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}) \rightarrow_\bot \mathbf{E}.$$

5. Fill in the details on the connection between the external and internal axiomatisations of $\rightarrow_\bot$. Show that $f \rightarrow_\bot g = \mathit{Curry}(g \circ \mathit{apply} \circ (\mathit{id} \otimes f))$ and use this to prove the functor laws for $\rightarrow_\bot$. Why can't $\rightarrow_\bot$ be extended to a bifunctor over $\mathcal{CPO}$?

6. Establish the following isomorphisms: $\mathit{Curry}\colon (\mathbf{D} \otimes \mathbf{E}) \rightarrow_\bot \mathbf{F} \cong \mathbf{D} \rightarrow_\bot (\mathbf{E} \rightarrow_\bot \mathbf{F})$; $\mathbf{X}_\bot \rightarrow_\bot \mathbf{D} \cong \mathbf{D}^\mathbf{X}$ ($\mathbf{X}$ is a set); $\mathbf{U} \rightarrow_\bot \mathbf{D} \cong \mathbf{D} \rightarrow_\bot \mathbf{U} \cong \mathbf{U}$; $\mathbb{O} \rightarrow_\bot \mathbf{D} \cong \mathbf{D}$; $\mathbf{D} \rightarrow_\bot (\prod_{i<n} \mathbf{E}_i) \cong \prod_{i<n} (\mathbf{D} \rightarrow_\bot \mathbf{E}_i)$.

7. Formulate and prove the analogue of Exercise 2.9, but with $\rightarrow_\bot$ in place of $\rightarrow$.

8. Make the necessary changes in the denotational semantics of *Imp* when we use the more accurate $\mathbf{S}_\bot \rightarrow_\bot \mathbf{S}_\bot$ in place of $\mathbf{S}_\bot \rightarrow \mathbf{S}_\bot$.

9. Show that, with the semantics of call-by-value the following equivalence holds:

$$\mathcal{E}[\![\mathbf{let}\ f_j(x_0, \ldots, x_{j-1})\ \mathbf{be}\ e\ \mathbf{in}\ f_j(a_0(), \ldots, a_{j-1}())]\!] = \mathcal{E}[\![[a_0()/x_0] \ldots [a_{j-1}()/x_{j-1}]e]\!]$$

where the a_l are constants (elements of $\mathbf{Op}_0$) such that $\mathcal{I}_0[\![a_l()]\!] \neq \bot$. Show that the more general equivalence in Exercise 2.32 fails. What is the appropriate equivalence for calls of recursively defined functions?

10. Fill in the details on the connection between the external and internal axiomatisations of $\sum$. Show that $h \circ [f_0, \ldots, f_{n-1}] = [h \circ f_0, \ldots, h \circ f_{n-1}]$ and use this result to establish the functor laws for $\sum$. Formulate and prove the analogue of Exercise 2.5 for $\sum$ (the analogue of Δ is written as ∇ and is called the codiagonal function).

11. Show that $\mathcal{CPO}$ does not have any initial object or a binary categorical sum. What goes wrong if you attempt to extend $\sum$ to a functor over $\mathcal{CPO}$?

12. Show that $\sum$ obeys suitable commutative and associative laws. Establish the isomorphisms: $\mathbf{U} + \mathbf{D} \cong \mathbf{D}$; $\mathbf{T} \cong \mathbb{O} + \mathbb{O}$; $\mathbf{N} \cong \mathbb{O} + \mathbf{N}$; $\mathbf{D} \otimes (\mathbf{E} + \mathbf{F}) \cong (\mathbf{D} \otimes \mathbf{E}) + (\mathbf{D} \otimes \mathbf{F})$; $\mathbf{T} \otimes \mathbf{D} \cong \mathbf{D}_\bot + \mathbf{D}_\bot$. Why do the analogous isomorphisms for $\times$ fail?

13. Show that the flat cpo, $\mathbf{X}_\bot$, associated with any set $\mathbf{X}$ can be characterised by the following universal property of the natural injection function, $\iota\colon \mathbf{X} \rightarrow \mathbf{X}_\bot$: if $f\colon \mathbf{X} \rightarrow \mathbf{D}$ is any function from $\mathbf{X}$ to the carrier of a cpo $\mathbf{D}$ then there is a unique $h\colon \mathbf{X}_\bot \rightarrow_\bot \mathbf{D}$ such that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{X} \arrow[d, "\iota"'] \arrow[dr, "f"] & \\
\mathbf{X}_\bot \arrow[r, "h"'] & \mathbf{D}
\end{tikzcd}$$

In the category of sets, the categorical sum of two sets $\mathbf{X}$ and $\mathbf{Y}$ is their disjoint union, $\mathbf{X} + \mathbf{Y} \stackrel{\text{def}}{=} \{\langle 0, x\rangle \mid x \in \mathbf{X}\} \cup \{\langle 1, y\rangle \mid y \in \mathbf{Y}\}$ and the categorical product is the ordinary Cartesian product. Show that for any sets $\mathbf{X}$ and $\mathbf{Y}$: $(\mathbf{X} + \mathbf{Y})_\bot \cong \mathbf{X}_\bot + \mathbf{Y}_\bot$ and $(\mathbf{X} \times \mathbf{Y})_\bot \cong \mathbf{X}_\bot \otimes \mathbf{Y}_\bot$.

14. Investigate infinite sums, $\sum_{i \in S} \mathbf{D}_i$.

15. Let $\mathbf{D}$ be a cpo. Let $\mathbf{D}^* \stackrel{\text{def}}{=} \sum_{i \in \omega} \mathbf{D}^{\textcircled{\scriptsize i}}$, which is just the infinite sum, $\mathbb{O} + \mathbf{D} + (\mathbf{D} \otimes \mathbf{D}) + \cdots$. What is $(\mathbf{N})^*$? Define the natural functions, $nil: \mathbb{O} \rightarrow \mathbf{D}^*$, $cons: \mathbf{D} \otimes \mathbf{D}^* \rightarrow \mathbf{D}^*$. Show that they are universal in the sense that if $\mathbf{E}$ is a cpo and we have $nil': \mathbb{O} \rightarrow_\perp \mathbf{E}$, $cons': \mathbf{D} \otimes \mathbf{E} \rightarrow_\perp \mathbf{E}$ then there is a unique $h: \mathbf{D}^* \rightarrow_\perp \mathbf{E}$ such that the following two diagrams commute:

$$\begin{tikzcd}
\mathbb{O} \arrow[d, "nil"'] \arrow[dr, "nil'"] & \\
\mathbf{D}^* \arrow[r, "h"'] & \mathbf{E}
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\mathbf{D} \otimes \mathbf{D}^* \arrow[r, "id_D \otimes h"] \arrow[d, "cons"'] & \mathbf{D} \otimes \mathbf{E} \arrow[d, "cons'"] \\
\mathbf{D}^* \arrow[r, "h"'] & \mathbf{E}^*
\end{tikzcd}$$

Give internal and external axiomatisations of $\mathbf{D}^*$ relating them to each other and the above universal property. [*Hint* For the internal axiomatisation you may find it helpful to introduce functions, $isnil: \mathbf{D}^* \rightarrow_\perp \mathbf{T}$, $head: \mathbf{D}^* \rightarrow_\perp \mathbf{D}$, and $tail: \mathbf{D}^* \rightarrow_\perp \mathbf{D}^*$.] Show how $(\cdot)^*$ can be considered as a covariant functor, $(\cdot)^*: \mathcal{CPO}_\perp \rightarrow \mathcal{CPO}_\perp$. Establish the isomorphism, $\mathbf{D}^* \cong \mathbb{O} + \mathbf{D} \otimes \mathbf{D}^*$.

16. Define $\lambda: \mathbf{D}^*$, $\iota: \mathbf{D} \rightarrow_\perp \mathbf{D}^*$, $\cdot: \mathbf{D}^* \otimes \mathbf{D}^* \rightarrow_\perp \mathbf{D}^*$, by:

$$\begin{aligned}
\lambda &= nil(\top), \\
\iota &= \lambda_\perp d: \mathbf{D}.cons(d \otimes \lambda), \\
\cdot &= \lambda_\perp x: \mathbf{D}^*, y: \mathbf{D}^*.(isnil(x) \rightarrow y \mid cons(head(x), tail(x) \cdot y)).
\end{aligned}$$

Note the infix notation used in the recursive definition of $\cdot$; in general $e \cdot e'$ is to abbreviate $\cdot(e \otimes e')$. Show that $(\mathbf{D}^*, \cdot, \lambda)$ is a monoid in that the following equations hold:

(1) *Identity* $\quad \forall x: \mathbf{D}^*. \lambda \cdot x = x \cdot \lambda = x$.
(2) *Associativity* $\quad \forall x, y, z: \mathbf{D}^*. x \cdot (y \cdot z) = (x \cdot y) \cdot z$.

Show that $(\mathbf{D}, \mathbf{D}^*, \lambda, \iota, \cdot)$ is the free monoid over $\mathbf{D}$ in the sense that if we have a structure $(\mathbf{D}, \mathbf{E}, \lambda', \iota', \cdot')$ where $\mathbf{E}$ is a cpo, $\lambda': \mathbf{E}$, $\iota': \mathbf{D} \rightarrow_\perp \mathbf{E}$, $\cdot': \mathbf{E} \otimes \mathbf{E} \rightarrow_\perp \mathbf{E}$ and $(\mathbf{E}, \lambda', \cdot')$ is a monoid in the above sense then there is a unique $h: \mathbf{D}^* \rightarrow_\perp \mathbf{E}$ such that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{D} \arrow[d, "\iota"'] \arrow[dr, "\iota'"] & \\
\mathbf{D}^* \arrow[r, "h"'] & \mathbf{E}
\end{tikzcd}$$

and such that h is a monoid homomorphism in that $h(\lambda) = \lambda'$ and for all x, y in $\mathbf{D}^*$, $h(x \cdot y) = h(x) \cdot' h(y)$.

17. Show that the external axioms for the lifting construction follow from the internal ones and establish the functor laws. Show that lifting can be taken as a covariant functor $(\cdot)_\perp: \mathcal{CPO}_\perp \rightarrow \mathcal{CPO}_\perp$.

18. Establish the isomorphism $\boldsymbol{\omega} \cong \boldsymbol{\omega}_\perp$ and, more generally, $\mathbf{X}^\infty \cong (\mathbf{X}_\perp) \otimes \mathbf{X}^\infty$ (see Exercise 1.17). Show that $\mathbf{Tapes} \cong \mathbf{Tapes}_\perp + \mathbf{Tapes}_\perp$, $\mathbf{D}_\perp \otimes \mathbf{E}_\perp \cong (\mathbf{D} \times \mathbf{E})_\perp$, $\mathbf{Btrees} \cong \mathbf{N} + (\mathbf{Btrees}_\perp \otimes \mathbf{Btrees}_\perp)$ and find a cpo $\mathbf{D}$ such that $\mathbf{D} \cong \mathbf{N} \otimes \mathbf{D}_\perp \otimes \mathbf{D}_\perp$.

19. Establish the following universal property of the maps $in_i \circ up: \mathbf{D}_i \rightarrow \sum(\mathbf{D}_i)_\perp$: if $f_i: \mathbf{D}_i \rightarrow \mathbf{E}$ $(i < n)$ are continuous maps then there is a unique $h: \sum(\mathbf{D}_i)_\perp \rightarrow_\perp \mathbf{E}$ such that, for all i, with $i < n$, the following diagram commutes:

$$\begin{tikzcd}
\mathbf{D}_i \arrow[d, "in_i \circ up"'] \arrow[dr, "f_i"] & \\
\sum(\mathbf{D}_i)_\perp \arrow[r, "h"'] & \mathbf{E}
\end{tikzcd}$$

and show, too, that $\prod_{i<n}(\mathbf{D}_i \rightarrow \mathbf{E}) \cong (\sum_{i<n}(\mathbf{D}_i)_\perp) \rightarrow_\perp \mathbf{E}$. Consider the semi-separated sum $\mathbf{D}_\perp + \mathbf{E}$ in the analogous way.

20. Investigate $\sum_{n\in\omega}(\mathbf{D}^n)_\perp$ $(= (\mathbf{D}_\perp)^*)$ in an analogous way to $\mathbf{D}^*$ (see Exercises 15 and 16). What happens in the other two cases, namely $\sum \mathbf{D}^n$ and $\sum(\mathbf{D}^{ⓝ})_\perp$?

21. Suppose we want to specify *Dec* with $\mathbf{Op}_0 = \{\mathbf{0}, \mathbf{true}, \mathbf{false}\}$ and $\mathbf{Op}_1 = \{\mathbf{succ}, \mathbf{pred}, \mathbf{zero}, \mathbf{isnum}\}$ where the elements of $\mathbf{Op}_0$ and the arithmetical elements of $\mathbf{Op}_1$ will have much the same meanings as in the text. However we want to model an evaluation mechanism which first tries to calculate the type (number or truth-value) of an expression and, if that terminates, it then tries to calculate its value. So, if the type of e is number (truth-value) that of $pred(e)$ is number (evaluation does not terminate). If the type of e is number (truth-value) that of $isnum(e)$ is truth-value (truth-value) and the value is tt ($f\!f$). Give a denotational semantics which uses $\mathbf{V} = \mathbf{N}_\perp + \mathbf{T}_\perp$, the separated sum.

4. The Category of CPOs Considered as a CPO Itself

The problem of finding recursively specified domains (and its solution) can be considered to be one of the main achievements of the present theory. In order to solve it, we will have to develop certain technical tools and that is the concern of the present chapter.

We have already seen several examples of cpos which satisfy recursive domain equations. For example $\mathbf{N}$, the cpo of the integers, satisfies: $\mathbf{N} \cong \mathbb{O} + \mathbf{N}$. Equations for $\mathbf{T}^\omega$, $\mathbf{D}^*$, $\mathbf{X}^\infty$, **Btrees** can be found in Exercises 2.4, 3.15, 3.18. The desire to solve such equations in general arises when we consider certain difficulties in denotational semantics. They often concern explicit or implicit forms of self-application where one has to make sense of expressions of the kind "$x(x)$". Then if $\mathbf{V}$ is a cpo of possible values for x, it seems natural to require that $\mathbf{V}$ somehow contains its own function space, $\mathbf{V} \to \mathbf{V}$. This is well-known to be impossible when $\mathbf{V}$ is a set and $(\mathbf{V} \to \mathbf{V})$ is the partial order of all monotonic functions over $\mathbf{V}$ (see Exercise 1.19). However it is possible when $\mathbf{V}$ is a cpo and we restrict attention to the continuous functions over $\mathbf{V}$.

As a concrete example consider the extension of *Imp* (see Exercise 1.3) where we add a syntactic class, **Proc**, of parameterless procedure variables (ranged over by p) and use them to add procedure declaration and calling clauses to the definition of **Stat**:

$$s ::= (\mathbf{proc}\ p\ \mathbf{begin}\ s\ \mathbf{end}) \mid (\mathbf{call}\ p).$$

The idea is that the declaration **proc** p **begin** s **end** should have the effect of assigning to p the command s, which is run when p is called. Thus we will want a cpo, approximately:

$$\mathbf{C} = (\mathbf{S}_\perp \times \mathbf{PEnv}) \to (\mathbf{S}_\perp \times \mathbf{PEnv})$$

(a more accurate model would be $(\mathbf{S}_\perp \otimes \mathbf{PEnv}_\perp) \to_\perp (\mathbf{S}_\perp \otimes \mathbf{PEnv})$). But this is not a proper definition of **PEnv** and **C** as the definition is circular. What we will do is relax the equalities to isomorphisms and try to find cpos **C** and **PEnv** which satisfy the simultaneous recursive domain equations:

(1) $$\mathbf{C} \cong (\mathbf{S}_\perp \times \mathbf{PEnv}) \to (\mathbf{S}_\perp \times \mathbf{PEnv}),$$

(2) $$\mathbf{PEnv} \cong (\mathbf{Proc}_\perp \to_\perp \mathbf{C}).$$

Equivalently we could just try to find a cpo $\mathbf{C}$ satisfying the equation:

(3) $$\mathbf{C} \cong (\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \to_\perp \mathbf{C})) \to (\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \to_\perp \mathbf{C}))$$

and then specify $\mathbf{PEnv} \stackrel{\text{def}}{=} (\mathbf{Proc}_\perp \to_\perp \mathbf{C})$.

The implicit self-application lies in the meaning of the statement **call** p. This will be a command, γ, which takes the current store σ (state plus procedure environment and so an element of $\mathbf{S}_\perp \times \mathbf{PEnv}$) and applies the component of it corresponding to p, namely $(\sigma \downarrow 1)[\![p]\!]$ to σ itself. Thus we have the implicit (and slightly indirect) self-application:

$$(\sigma \downarrow 1)[\![p]\!](\sigma).$$

Later on we shall consider the untyped λ-calculus where the self-application is completely explicit.

Our idea for solving such recursive domain equations is to consider them as a kind of fixed-point equation. Instead of looking for an element of a cpo, which satisfies a fixed-point condition we look for a cpo (an object in $\mathcal{CPO}_\perp$) which satisfies the condition. We will be guided very explicitly by our earlier considerations on finding least fixed-points of continuous functions in cpos. There is a well-known analogy between quasi-orders (partial orders less the antisymmetry condition) and categories. Every quasiorder $(\mathbf{D}, \sqsubseteq)$ is a category if we take the elements of $\mathbf{D}$ to be the objects and stipulate that there is at most one morphism from an element, d, to another, e, which we write as $(d \to e)$; it exists iff $d \sqsubseteq e$. Thus we can consider order relation as analogous to morphisms, the reflexivity relation $d \sqsubseteq d$, is analogous to the identity, $id_{\mathbf{D}}: d \to d$ and the law of transitivity: $(c \sqsubseteq d \wedge d \sqsubseteq e \Rightarrow c \sqsubseteq e)$ is analogous to composition: $((f: c \to d \wedge g: d \to e) \Rightarrow g \circ f: c \to e)$. One can perhaps see a morphism $f: c \to d$ as showing *why* $c \sqsubseteq d$. It should be stressed that we are only offering an analogy as the reason for the particular mathematical development we are following. Some hints at a justification can be found in Chapter 4. More can be found in the detailed analysis of particular examples of semantics.

The analogue of a monotonic function, f, (where $d \sqsubseteq e$ implies $f(d) \sqsubseteq f(e)$) is a functor F (where $f: d \rightarrow e$ implies $Ff: Fd \rightarrow Fe$ and F preserves the identity and composition). Note that in forming a category-theoretic analogue one must add extra uniformity conditions. So we might expect to consider the category $\mathcal{CPO}_\perp$ and functors $F: \mathcal{CPO}_\perp \rightarrow \mathcal{CPO}_\perp$. However this does not work for equations such as (3) since one cannot compose covariant and contravariant functors in such a cavalier way. The natural attempt at the definition of F would be to put, for each cpo $\mathbf{D}$:

$$F(\mathbf{D}) \stackrel{\text{def}}{=} (\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \rightarrow_\perp \mathbf{D})) \rightarrow (\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \rightarrow_\perp \mathbf{D})) \tag{4}$$

and for each $f: \mathbf{D} \rightarrow_\perp \mathbf{E}$ to define $F(f): F(\mathbf{D}) \rightarrow_\perp F(\mathbf{E})$ by:

$$F(f) = (id_{\mathbf{S}_\perp} \times (id_{\mathbf{Proc}_\perp} \rightarrow_\perp f)) \rightarrow (id_{\mathbf{S}_\perp} \times (id_{\mathbf{Proc}_\perp} \rightarrow_\perp f)). \tag{5}$$

However this is not a good definition; the expression on the right has the wrong type. Its type is $(\mathbf{Y} \rightarrow \mathbf{X}) \rightarrow_\perp (\mathbf{X} \rightarrow \mathbf{Y})$ where $\mathbf{X}$ is $[\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \rightarrow_\perp \mathbf{D})]$ and $\mathbf{Y}$ is $[\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \rightarrow_\perp \mathbf{E})]$ but what is wanted is an expression of type $(\mathbf{X} \rightarrow \mathbf{X}) \rightarrow_\perp (\mathbf{Y} \rightarrow \mathbf{Y})$. This would be obtained if the first occurrence of the variable f on the right hand side were replaced by an expression of type $\mathbf{E} \rightarrow_\perp \mathbf{D}$, which should, presumably, involve f.

Luckily we can trace this to a somewhat questionable assumption in the analogy: there is no reason why *any* $f: \mathbf{D} \rightarrow \mathbf{E}$ should show why $\mathbf{D} \sqsubseteq \mathbf{E}$. Indeed we have not discussed how one cpo could possibly be considered as an approximation to another. Now in modelling a given computational situation we might be faced with a choice between $\mathbf{D}$ and $\mathbf{E}$. It may be that we can use both but in such a way that on the one hand the elements of $\mathbf{D}$ give only a degraded version of the information given by those in $\mathbf{E}$, and on the other hand for each element in $\mathbf{D}$ there is an element in $\mathbf{E}$ which contains exactly the same information. Thus $\mathbf{E}$ can be used to paint a much finer picture than $\mathbf{D}$. Of course, this does not mean $\mathbf{E}$ is better than $\mathbf{D}$: it all depends on the circumstances which is the more appropriate. We try to capture these correspondences between elements of $\mathbf{D}$ and $\mathbf{E}$ as a pair of continuous functions, $i: \mathbf{D} \rightarrow \mathbf{E}$, $j: \mathbf{E} \rightarrow \mathbf{D}$. The first function shows how for any element d, of $\mathbf{D}$ there is an element $i(d)$ in $\mathbf{E}$ with the same information as d; the second one shows how for any element e in $\mathbf{E}$ there is an element $j(e)$ in $\mathbf{D}$ which contains less information than e and, since $\mathbf{D}$ is to be a degraded version of $\mathbf{E}$, $j(e)$ is the best approximation to e in $\mathbf{D}$. Thus we expect the following two equations to hold:

$$\forall d: \mathbf{D}. j \circ i(d) = d,$$
$$\forall e: \mathbf{E}. i \circ j(e) \sqsubseteq e.$$

For since $i(d)$ contains the same information as d and $j(i(d))$ contains less than $i(d)$ we have $j(i(d)) \sqsubseteq d$. But since d is an approximation to $i(d)$ and $j(i(d))$ is the best one we have $j(i(d)) \sqsupseteq d$. This "demonstrates" the first equation. For the second we note that $i(j(e))$ contains the same information as $j(e)$ which is an approximation to e. Conversely suppose we have i, j which satisfy the equations and rend the phrase "d approximates e" for an element d of $\mathbf{D}$ and an element e of $\mathbf{E}$ as "$i(d) \sqsubseteq e$". Then the second equation says that $j(e)$ approximates e and also if d approximates e then $d = j(i(d)) \sqsubseteq j(e)$, showing that $j(e)$ is the best approximation to e in $\mathbf{D}$. This seems to be a reasonable argument for regarding such pairs $\langle i, j \rangle$ as giving one kind of approximation relation between domains and we make a suitable definition.

Definition 1. Let $\mathbf{D}$ and $\mathbf{E}$ be cpos and let $i: \mathbf{D} \rightarrow \mathbf{E}$, $j: \mathbf{E} \rightarrow \mathbf{D}$ be continuous functions. Then $\langle i, j \rangle$ is a *projection pair* (from $\mathbf{D}$ to $\mathbf{E}$) and i is an *embedding* and j is a *projection* and $\mathbf{D}$ is a *projection* of $\mathbf{E}$ provided only that $j \circ i = id_{\mathbf{D}}$ and $i \circ j \sqsubseteq id_{\mathbf{E}}$.

Examples

1. For $n \in \omega$ put $[n] = \{m \in \omega \mid m < n\}$. Then the natural inclusion maps $\iota_{mn}: [m]_\perp \rightarrow [n]_\perp$ $(m \leq n)$ are embeddings. The projection $\pi_{mn}: [n]_\perp \rightarrow [m]_\perp$ $(m \leq n)$ corresponding to ι_{mn} is defined by:

$$\pi_{nm}(l) = \begin{cases} l & (l < m), \\ \perp & (l = \perp \text{ or } l \geq m). \end{cases}$$

2. The cpo **Tapes** is a projection of $\mathbf{T}^\omega$. With the aid of the evident functions $cons: \mathbf{T} \times \mathbf{T}^\omega \rightarrow \mathbf{T}^\omega$, $head: \mathbf{T}^\omega \rightarrow \mathbf{T}$, $tail: \mathbf{T}^\omega \rightarrow \mathbf{T}^\omega$ we can define the projection pair, $\mathbf{Tapes} \xrightarrow{i} \mathbf{T}^\omega \xrightarrow{j} \mathbf{Tapes}$ recursively by:

$$i(t) = cons(hd(t) = 0, i(tl(t))),$$
$$j(v) = (head(v) \rightarrow 0 \cdot j(tail(v)) \mid 1 \cdot j(tail(v))).$$

3. For any cpos $\mathbf{D}$, $\mathbf{E}$ we have the projection pair, $(\mathbf{D} \rightarrow_\perp \mathbf{E}) \xrightarrow{\iota} (\mathbf{D} \rightarrow \mathbf{E}) \xrightarrow{strict} (\mathbf{D} \rightarrow_\perp \mathbf{E})$.

The analysis of information in terms of events in Chapter 1 warns us that more subtle approximation relations between cpos can exist. For example it was shown there that, in a sense, $\mathbf{P}\boldsymbol{\omega}$ is a degraded version of **Tapes**. But the discussion there only indicated a map $j\colon \mathbf{Tapes} \to \mathbf{P}\boldsymbol{\omega}$ and none in the other direction: indeed it is not hard to show that $\mathbf{P}\boldsymbol{\omega}$ is not a projection of **Tapes**. However projections will prove perfectly satisfactory for the solution of recursive domain equations.

Lemma 1. *Let $\mathbf{D} \xrightarrow{i} \mathbf{E} \xrightarrow{j} \mathbf{D}$ be a projection pair. Then both i and j are strict. If $\mathbf{D} \xrightarrow{i'} \mathbf{E} \xrightarrow{j'} \mathbf{D}$ is another projection pair then $i \sqsubseteq i'$ iff $j \sqsupseteq j'$.*
Proof. First $i(\perp_{\mathbf{D}}) \sqsubseteq i \circ j(\perp_{\mathbf{E}}) \sqsubseteq \perp_{\mathbf{E}}$ showing i is strict. Second $j(\perp_{\mathbf{E}}) = j \circ i(\perp_{\mathbf{D}}) = \perp_{\mathbf{D}}$ showing j is strict.

If $i \sqsubseteq i'$ then $j \sqsupseteq j \circ i' \circ j' \sqsupseteq j \circ i \circ j' = j'$. Conversely, if $j \sqsupseteq j'$ then $i = i \circ j' \circ i' \sqsubseteq i \circ j \circ i' \sqsubseteq i'$. ∎

The second part of this lemma shows that every embedding i has a *unique* corresponding projection, called its *right adjoint*, and written as i^{R}; further, it shows that every projection, j, has a unique corresponding embedding, called its *left adjoint* and written as j^{L}. It follows that $i^{\mathrm{RL}} = i$ and $j^{\mathrm{LR}} = j$. Note that we are not claiming that i^{R} is a continuous function of i, nor that j^{L} is a continuous function of j, we are just introducing some notation. We write $i\colon \mathbf{D} \lhd \mathbf{E}$ to indicate that i is an embedding and $j\colon \mathbf{E} \rhd \mathbf{D}$ to indicate that j is a projection. With all this under our belts we now have a good idea what the category of cpos and approximations ought to be:

Definition 2. The category $\mathcal{CPO}^{\mathrm{E}}$ has as objects the cpos and as morphisms the embeddings and composition is the usual composition of functions.

To see that this is a good definition note that $id_{\mathbf{D}}\colon \mathbf{D} \lhd \mathbf{D}$ as $id_{\mathbf{D}}^{\mathrm{R}} = id_{\mathbf{D}}$ and that if $f\colon \mathbf{D} \lhd \mathbf{E}$ and $g\colon \mathbf{E} \lhd \mathbf{F}$ then $(g \circ f)\colon \mathbf{D} \lhd \mathbf{F}$ as $(g \circ f)^{\mathrm{R}} = f^{\mathrm{R}} \circ g^{\mathrm{R}}$. The first claim is obvious; for the second calculate $(f^{\mathrm{R}} \circ g^{\mathrm{R}}) \circ (g \circ f) = f^{\mathrm{R}} \circ f = id_{\mathbf{D}}$ and $(g \circ f) \circ (f^{\mathrm{R}} \circ g^{\mathrm{R}}) \sqsubseteq (g \circ g^{\mathrm{R}}) \sqsubseteq id_{\mathbf{E}}$. We could have worked with $\mathcal{CPO}^{\mathrm{P}}$, the category of projections, instead of $\mathcal{CPO}^{\mathrm{E}}$. However this would not make any real mathematical difference as the two categories are dual and so, since the embeddings "go in the same way" as the approximation relation we chose $\mathcal{CPO}^{\mathrm{E}}$. Another possibility would be to use the (isomorphic) category of projection pairs and here the choice of $\mathcal{CPO}^{\mathrm{E}}$ is on grounds of simplicity.

When an embedding is an inclusion map, $\iota\colon \mathbf{D} \subseteq \mathbf{E}$ the defining equations for a projection pair take on the simple form, $\forall d\colon \mathbf{D}.\, i(d) = d$ and $\forall e\colon \mathbf{E}.\, j(e) \sqsubseteq e$. Any embedding $i\colon \mathbf{D} \lhd \mathbf{E}$ is essentially of this form, for if we let the image, $i(\mathbf{D})$ be given the partial order inherited from $\mathbf{E}$ we find it in a subcpo of $\mathbf{E}$ as $\perp_{\mathbf{E}} = i(\perp_{\mathbf{D}})$ and if $\langle i(d_n)\rangle_{n\in\omega}$ is an increasing ω-chain so is $\langle j \circ i(d_n)\rangle_{n\in\omega} = \langle d_n\rangle_{n\in\omega}$ and we see, by the continuity of i that $\bigsqcup_{\mathbf{E}} i(d_n) = i(\bigsqcup_{\mathbf{D}} d_n)$. It follows that the inclusion map $\iota\colon i(\mathbf{D}) \subseteq \mathbf{E}$ is continuous. Next one easily checks that $\theta\colon \mathbf{D} \to i(\mathbf{D})$, where $\theta(d) \stackrel{\mathrm{def}}{=} i(d)$. In an isomorphism with inverse $\theta^{-1} = j \upharpoonright (i(\mathbf{D}))$; it is clear that θ and θ^{-1} are monotonic and we have $\theta^{-1} \circ \theta(d) = j \circ i(d) = d$ and $\theta \circ \theta^{-1}(i(d)) = i \circ j \circ i(d) = i(d)$. Now we have the factorisation of i:

$$\begin{tikzcd}
\mathbf{D} \arrow[r, "\theta"] \arrow[dr, "i"'] & i(\mathbf{D}) \arrow[d, "\iota"] \\
 & \mathbf{E}
\end{tikzcd}$$

and ι is an embedding as $\iota^{\mathrm{R}} = \theta \circ j$ for: $(\theta \circ j) \circ \iota = \theta \circ j \circ \iota \circ \theta \circ \theta^{-1} = \theta \circ j \circ i \circ \theta^{-1} = id$ and $\iota \circ (\theta \circ j) = i \circ j \sqsubseteq id$. Because of all this when $i\colon \mathbf{D} \lhd \mathbf{E}$ we often use an *identification convention* where such element d, of $\mathbf{D}$, is *identified* with its image $i(d)$. Then $\mathbf{D}$ is identified with $i(\mathbf{D})$ and i with the inclusion map, ι.

We now return to the pursuit of our analogy attempting to regard $\mathcal{CPO}^{\mathrm{E}}$ as a kind of cpo in which the approximation relation are replaced by embeddings, $i\colon \mathbf{D} \lhd \mathbf{E}$. Corresponding to the idea of a least element $\perp$ (for any x, $\perp \sqsubseteq x$) we have the idea of an initial element $\perp$ (for any $\mathbf{D}$, there is a *unique* $\perp \lhd \mathbf{D}$). The initial element must be $\mathbf{U}$, the one-point cpo, and the morphism $\mathbf{U} \lhd \mathbf{D}$ is just $K_{\perp_{\mathbf{D}}}$, the only strict map from $\mathbf{U}$ to $\mathbf{D}$. This is an embedding as $K_{\perp_{\mathbf{D}}}^{\mathrm{R}} = K_{\perp_{\mathbf{U}}}$.

Turning to functors we see next how all our functors over $\mathcal{CPO}_{\perp}$, whether covariant or contravariant, become covariant functor over $\mathcal{CPO}^{\mathrm{E}}$! The clue lies in the above remarks on the type of the function $f\colon \mathbf{D} \to_{\perp} \mathbf{E}$ when we tried to define a functor for equation (3). What was needed was an element of type $\mathbf{E} \to_{\perp} \mathbf{D}$ instead. This is now easy, given $i\colon \mathbf{D} \lhd \mathbf{E}$ we will use $j\colon \mathbf{E} \rhd \mathbf{D}$.

Example 1 Products. Starting with $\prod: \mathcal{CPO}_\perp^n \to \mathcal{CPO}_\perp$ (see Exercise 2.6) we define $\prod^{\mathrm{E}}: (\mathcal{CPO}^{\mathrm{E}})^n \to \mathcal{CPO}^{\mathrm{E}}$ by putting for cpos $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$:

$$\prod_{i<n}^{\mathrm{E}} \mathbf{D}_i = \prod_{i<n} \mathbf{D}_i$$

and for embeddings $f_i: \mathbf{D}_i \triangleleft \mathbf{E}_i$ $(i<n)$:

$$\prod_{i<n}^{\mathrm{E}} f_i = \prod_{i<n} f_i.$$

We must prove that the right hand side is also an embedding and so we show that $(\prod_{i<n} f_i)^{\mathrm{R}} = \prod_{i<n} f_i^{\mathrm{R}}$, by calculating:

$$\begin{aligned}
(\prod_{i<n} f_i^{\mathrm{R}}) \circ (\prod_{i<n} f_i) &= \prod_{i<n} (f_i^{\mathrm{R}} \circ f_i) && \text{functor laws for } \textstyle\prod \\
&= \prod_{i<n} id_{\mathbf{D}_i} && \text{property of embeddings} \\
&= id_{\prod \mathbf{D}_i} && \text{functor laws for } \textstyle\prod \\
&= id_{\prod^{\mathrm{E}} \mathbf{D}_i} && \text{definition of } \textstyle\prod^{\mathrm{E}},
\end{aligned}$$

and

$$\begin{aligned}
(\prod_{i<n} f_i) \circ (\prod_{i<n} f_i^{\mathrm{R}}) &= \prod_{i<n} (f_i \circ f_i^{\mathrm{R}}) \\
&\sqsubseteq \prod_{i<n} id_{\mathbf{E}_i} && \textstyle\prod \text{ acts monotonically on morphisms} \\
&= id_{\prod^{\mathrm{E}} \mathbf{E}_i}.
\end{aligned}$$

The functor laws for $\prod^{\mathrm{E}}$ follow immediately from those for $\prod$.

Example 2 Exponentiation. Starting with $\to: \mathcal{CPO}_\perp^2 \to \mathcal{CPO}_\perp$ (see Exercise 2.9) we define the *covariant* functor $\to^{\mathrm{E}}: (\mathcal{CPO}^{\mathrm{E}})^2 \to \mathcal{CPO}^{\mathrm{E}}$ by putting for cpos $\mathbf{D}$ and $\mathbf{E}$:

$$(\mathbf{D} \to^{\mathrm{E}} \mathbf{E}) = (\mathbf{D} \to \mathbf{E})$$

and for $f: \mathbf{D} \triangleleft \mathbf{D}'$, $g: \mathbf{E} \triangleleft \mathbf{E}'$:

$$(f \to^{\mathrm{E}} g) = f^{\mathrm{R}} \to g,$$

using the idea hinted at above. To show $f \to^{\mathrm{E}} g: (\mathbf{D} \to \mathbf{E}) \triangleleft (\mathbf{D}' \to \mathbf{E}')$ we prove that $(f \to^{\mathrm{E}} g)^{\mathrm{R}} = (f \to g^{\mathrm{R}})$ by calculating:

$$\begin{aligned}
(f \to g^{\mathrm{R}}) \circ (f^{\mathrm{R}} \to g) &= (f^{\mathrm{R}} \circ f) \to (g^{\mathrm{R}} \circ g) && \text{functor laws} \\
&= id_{\mathbf{D}} \to id_{\mathbf{E}} \\
&= id_{\mathbf{D} \to \mathbf{E}}
\end{aligned}$$

and

$$\begin{aligned}
(f^{\mathrm{R}} \to g) \circ (f \to g^{\mathrm{R}}) &= (f \circ f^{\mathrm{R}}) \to (g \circ g^{\mathrm{R}}) \\
&\sqsubseteq id_{\mathbf{D}'} \to id_{\mathbf{E}'} && \to \text{ acts monotonically on morphisms} \\
&= id_{\mathbf{D}' \to \mathbf{E}'}.
\end{aligned}$$

We should also check the functor laws:

$$(id_{\mathbf{D}} \to^{\mathrm{E}} id_{\mathbf{E}}) = id_{\mathbf{D}} \to id_{\mathbf{E}} = id_{(\mathbf{D} \to \mathbf{E})} = id_{(\mathbf{D} \to^{\mathrm{E}} \mathbf{E})}$$

and for $f: \mathbf{D} \triangleleft \mathbf{D}'$, $f': \mathbf{D}' \triangleleft \mathbf{D}''$, $g: \mathbf{E} \triangleleft \mathbf{E}'$, $g': \mathbf{E}' \triangleleft \mathbf{E}''$:

$$\begin{aligned}
(f' \to^{\mathrm{E}} g') \circ (f \to^{\mathrm{E}} g) &= (f'^{\mathrm{R}} \to g') \circ (f^{\mathrm{R}} \to g) \\
&= ((f^{\mathrm{R}} \circ f'^{\mathrm{R}}) \to (g' \circ g)) \\
&= (f' \circ f)^{\mathrm{R}} \to (g' \circ g) \\
&= (f' \circ f) \to^{\mathrm{E}} (g' \circ g).
\end{aligned}$$

In order to avoid tediously checking the details for all our other functors we formulate a general result.

Definition 3. Let $T{:}\,\mathcal{CPO}_\perp^{m+n} \rightarrow \mathcal{CPO}_\perp$ $(m, n \geq 0)$ be contravariant in its first m arguments and covariant in its following n arguments. Then T is *locally monotonic* iff whenever $f_i, f_i'{:}\,\mathbf{D}_i' \rightarrow \mathbf{D}_i$ $(i < m)$ and $g_j, g_j'{:}\,\mathbf{E}_j \rightarrow \mathbf{E}_j'$ $(j < n)$ are such that $f_i \sqsubseteq f_i'$ $(i < m)$ and $g_j \sqsubseteq g_j'$ $(j < n)$ then:

$$T(f_0, \ldots, f_{m-1}, g_0, \ldots, g_{n-1}) \sqsubseteq T(f_0', \ldots, f_{m-1}', g_0', \ldots, g_{n-1}').$$

Clearly all the functors $\prod$, $\rightarrow$, $\bigotimes$, $\rightarrow_\perp$, $\sum$, $(\cdot)_\perp$, $(\cdot)^*$ (see Exercise 3.15) are or can be regarded as locally monotonic functions. There are two other important examples. The covariant projection functor $P_i{:}\,\mathcal{CPO}_\perp^n \rightarrow \mathcal{CPO}_\perp$ $(n \geq 0, i < n)$ is defined by:

$$P_i(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}) = \mathbf{D}_i,$$
$$P_i(f_0, \ldots, f_{n-1}) = f_i.$$

Both the projection and constant functors are clearly locally monotonic.

Lemma 2. *Every locally monotonic functor $T{:}\,\mathcal{CPO}_\perp^{m+n} \rightarrow \mathcal{CPO}_\perp$, as in Definition 3 can be turned into a covariant functor $T^{\mathrm{E}}{:}\,(\mathcal{CPO}^{\mathrm{E}})^{m+n} \rightarrow \mathcal{CPO}^{\mathrm{E}}$ by stipulating for cpos $\mathbf{D}_0$, ..., $\mathbf{D}_{m-1}$, $\mathbf{E}_0$, ..., $\mathbf{E}_{n-1}$ that:*

$$T^{\mathrm{E}}(\mathbf{D}_0, \ldots, \mathbf{D}_{m-1}, \mathbf{E}_0, \ldots, \mathbf{E}_{n-1}) = T(\mathbf{D}_0, \ldots, \mathbf{D}_{m-1}, \mathbf{E}_0, \ldots, \mathbf{E}_{n-1})$$

and for embeddings, $f_i{:}\,\mathbf{D}_i \lhd \mathbf{D}_i'$ $(i < m)$, $g_i{:}\,\mathbf{E}_i \lhd \mathbf{E}_i'$ $(i < n)$ that

$$T^{\mathrm{E}}(f_0, \ldots, f_{m-1}, g_0, \ldots, g_{n-1}) = T(f_0^{\mathrm{R}}, \ldots, f_{m-1}^{\mathrm{R}}, g_0, \ldots, g_{n-1}).$$

Proof. The proof is left to the reader. ∎

It is now clear how to define the functor, $F{:}\,\mathcal{CPO}^{\mathrm{E}} \rightarrow \mathcal{CPO}^{\mathrm{E}}$ needed for equation (3). For a cpo $\mathbf{D}$ we write, as in equation (4):

$$F(\mathbf{D}) = (\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \rightarrow_\perp \mathbf{D})) \rightarrow (\mathbf{S}_\perp \times (\mathbf{Proc}_\perp \rightarrow_\perp \mathbf{D}))$$

but for an embedding $f{:}\,\mathbf{D} \lhd \mathbf{E}$ we write:

$$F(f) = (id_{\mathbf{S}_\perp} \times (id_{\mathbf{Proc}_\perp} \rightarrow_\perp f^{\mathrm{R}})) \rightarrow (id_{\mathbf{S}_\perp} \times (id_{\mathbf{Proc}_\perp} \rightarrow_\perp f)).$$

What this means is that F is a composition of covariant functors over $\mathcal{CPO}^{\mathrm{E}}$ as:

$$F = \rightarrow^{\mathrm{E}}(G, G), \quad \text{where}$$
$$G = \times^{\mathrm{E}}(K^{\mathrm{E}}_{\mathbf{S}_\perp}, \rightarrow_\perp{}^{\mathrm{E}}(K^{\mathrm{E}}_{\mathbf{Proc}_\perp}, Id^{\mathrm{E}}))$$

and we leave it to the reader to check that these two definitions amount to the same thing.

When we want to solve simultaneous equations, such as (1), (2) we have to find simultaneous fixed-points of covariant functors over $\mathcal{CPO}^{\mathrm{E}}$. For example in this case we define $F_0, F_1{:}\,(\mathcal{CPO}^{\mathrm{E}})^2 \rightarrow \mathcal{CPO}^{\mathrm{E}}$ by:

$$F_0 = \rightarrow^{\mathrm{E}}(\times^{\mathrm{E}}(K^{\mathrm{E}}_{\mathbf{S}_\perp}, P_0^{\mathrm{E}}), \times^{\mathrm{E}}(K^{\mathrm{E}}_{\mathbf{S}_\perp}, P_0^{\mathrm{E}}))$$
$$F_1 = \rightarrow_\perp{}^{\mathrm{E}}(K^{\mathrm{E}}_{\mathbf{Proc}_\perp}, P_1^{\mathrm{E}})$$

and then we have to find cpos $\mathbf{C}$, $\mathbf{PEnv}$ such that:

$$\mathbf{C} = F_0(\mathbf{C}, \mathbf{PEnv})$$
$$\mathbf{PEnv} = F_1(\mathbf{C}, \mathbf{PEnv}).$$

As may be expected, it will not prove much harder to solve such simultaneous equations than it is to solve ordinary ones.

Continuing with the analogy we now see what corresponds to lubs of increasing ω-chains, $\sqcup_{n\in\omega} x_n$. First of all the analogue of an increasing ω-chain, $x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \cdots$ is a (*direct*) *ω-chain* $\Delta = \langle \mathbf{D}_m, f_{mn}\rangle$ of cpos $\mathbf{D}_n$ and embeddings $f_{mn}\colon \mathbf{D}_m \lhd \mathbf{D}_n$ $(m \le n < \omega)$ such that all the following triangles commute where $l \le m \le n$:

$$\begin{tikzcd} \mathbf{D}_l \arrow[r, "f_{lm}"] \arrow[dr, "f_{ln}"'] & \mathbf{D}_m \arrow[d, "f_{mn}"] \\ & \mathbf{D}_n \end{tikzcd}$$

Clearly the embeddings $f_{m(m+1)}$ determine all the others and sometimes we write (or define) an ω-chain as $\langle \mathbf{D}_m, f_m\rangle$ where $f_m\colon \mathbf{D}_m \lhd \mathbf{D}_{m+1}$ is $f_{m(m+1)}$; we can picture ω-chain as:

$$\mathbf{D}_0 \overset{f_0}{\lhd} \mathbf{D}_1 \overset{f_1}{\lhd} \mathbf{D}_2 \overset{f_2}{\lhd} \cdots \overset{f_{n-1}}{\lhd} \mathbf{D}_n \overset{f_n}{\lhd} \cdots.$$

An example of an ω-chain in $\langle [m], \iota_{mn}\rangle$ which we might picture as:

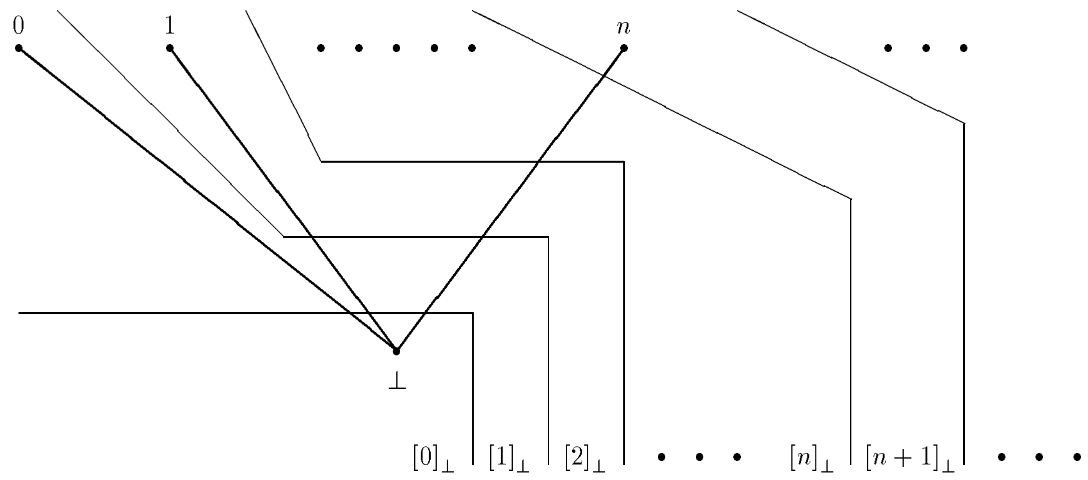


Figure XIII: A direct ω-chain.

Corresponding to an upper bound, x, of an increasing ω-chain, we clearly need for a chain $\Delta = \langle \mathbf{D}_m, f_{mn}\rangle$ not only a cpo $\mathbf{D}$ but also a whole collection $\rho = \langle \rho_m\rangle_{m\in\omega}$ of embeddings $\rho_m\colon \mathbf{D}_m \lhd \mathbf{D}$ which satisfy a suitable coherence condition. So we say that $\rho\colon \Delta \to \mathbf{D}$ is a *cone* from Δ to $\mathbf{D}$ if ρ is a sequence $\langle \rho_n\rangle_{n\in\omega}$ of embeddings $\rho_n\colon \mathbf{D}_n \lhd \mathbf{D}$ such that all the following triangles commute where $m \le n < \omega$:

$$\begin{tikzcd} \mathbf{D}_m \arrow[r, "\rho_m"] \arrow[d, "f_{mn}"'] & \mathbf{D} \\ \mathbf{D}_n \arrow[ur, "\rho_n"'] & \end{tikzcd}$$

Note that it is only necessary to check that they commute for the cases with $n = m+1$ as the other cases then follow using the commuting triangles for chains. For the chain $\Delta = \langle [m], \iota_{mn}\rangle$ we have the cone $\rho\colon \Delta \to \mathbf{N}$ where $\rho_m = \iota_{m\omega}$.

Finally corresponding to the idea of a *least* upper bound of an increasing chain we have the idea of an *initial* (also called *universal*) cone. If $\rho\colon \Delta \to \mathbf{D}$, $\rho'\colon \Delta \to \mathbf{D}'$ are cones a morphism, $\theta\colon \rho \lhd \rho'$ is defined to be an embedding $\theta\colon \mathbf{D} \lhd \mathbf{D}'$ such that all the following triangles commute:

$$\begin{tikzcd}
\mathbf{D}_m \arrow[r, "\rho_m"] \arrow[dr, "\rho'_m"'] & \mathbf{D} \arrow[d, "\theta"] \\
 & \mathbf{D}'
\end{tikzcd}$$

This, as may easily be verified, gives a category of cones from Δ and an initial cone is one from which there is exactly one morphism to any other cone. This specifies ρ up to an isomorphism.

In other words, $\rho\colon \Delta \to \mathbf{D}$ is initial (= universal) iff whenever $\rho'\colon \Delta \to \mathbf{D}'$ is a cone there is exactly one embedding $\theta\colon \mathbf{D} \lhd \mathbf{D}'$ (called the mediating morphism) which makes all the above triangles commute. Clearly this specifies $\mathbf{D}$ and also the ρ_n up to isomorphism, as we have already essentially remarked; when $\rho\colon \Delta \to \mathbf{D}$ is universal we sometimes write $\mathbf{D} = \lim_{\to} \Delta$ (the *direct* limit of Δ) and in the light of the above remarks on isomorphism this should cause little confusion.

For example, as the reader no doubt expected, the cone $\langle \iota_{m\omega} \rangle_{m \in \omega}\colon \langle [m], \iota_{mn} \rangle \to \mathbf{N}$ is universal as if $\rho\colon \langle [m], \iota_{mn} \rangle \to \mathbf{D}$ is a cone then we must have for the mediating morphism, θ, that $\theta(\perp) = \perp$ and for $m < \omega$, $\theta(m) = \theta(\iota_{m\omega}(m)) = \rho_m(m)$, which clearly determines θ and shows that it exists.

It is awkward to work with the category-theoretic definition directly and in order to show, following the analogy, that direct limits always exist we find an equivalent local order-theoretic condition. If $\rho\colon \Delta \to \mathbf{D}$ is a cone where $\Delta = \langle \mathbf{D}_m, f_m \rangle$ then the chain $\langle \rho_m \circ \rho_m^{\mathrm{R}}(x) \rangle_{m \in \omega}$ of the best approximations to an element of x at level m is increasing as we have:

$$\rho_m \circ \rho_m^{\mathrm{R}} = (\rho_{m+1} \circ f_m) \circ (\rho_{m+1} \circ f_m)^{\mathrm{R}} = \rho_{m+1} \circ f_m \circ f_m^{\mathrm{R}} \circ \rho_{m+1}^{\mathrm{R}} \sqsubseteq \rho_{m+1} \circ \rho_{m+1}^{\mathrm{R}}.$$

It is natural to expect that if $\mathbf{D}$ is to be the direct limit then x is the limit of the chain and indeed we have:

Lemma 3. *Let $\Delta = \langle \mathbf{D}_m, f_{mn} \rangle$ be a direct ω-chain of embeddings and let $\rho\colon \Delta \to \mathbf{D}$ be a cone of embeddings. Then if $(\bigsqcup_n \rho_n \circ \rho_n^{\mathrm{R}}) = id_{\mathbf{D}}$ it follows that ρ is universal.*

Proof. Suppose $\rho'\colon \Delta \to \mathbf{D}'$ is a cone of embeddings. Then $\langle \rho'_m \circ \rho_m^{\mathrm{R}} \rangle_{m \in \omega}$ is an increasing ω-chain as $\rho'_m \circ \rho_m^{\mathrm{R}} = \rho'_{m+1} \circ f_m \circ f_m^{\mathrm{R}} \circ \rho_{m+1}^{\mathrm{R}} \sqsubseteq \rho'_{m+1} \circ \rho_{m+1}^{\mathrm{R}}$ and so we may define $\theta\colon \mathbf{D} \to \mathbf{D}'$ by:

$$\theta = \bigsqcup_m \rho'_m \circ \rho_m^{\mathrm{R}}.$$

To prove θ is an embedding we note that $\langle \rho_m \circ \rho_m'^{\mathrm{R}} \rangle_{m \in \omega}$ must also be an increasing ω-sequence and prove that $\theta^{\mathrm{R}} = \bigsqcup_m \rho_m \circ \rho_m'^{\mathrm{R}}$ by calculating:

$$\begin{aligned}
(\bigsqcup_m \rho_m \circ \rho_m'^{\mathrm{R}}) \circ (\bigsqcup_m \rho'_m \circ \rho_m^{\mathrm{R}}) &= \bigsqcup_m \rho_m \circ \rho_m'^{\mathrm{R}} \circ \rho'_m \circ \rho_m'^{\mathrm{R}} && \text{composition is continuous} \\
&= \bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}} \\
&= id_{\mathbf{D}} && \text{by assumption}
\end{aligned}$$

and

$$\begin{aligned}
(\bigsqcup_m \rho'_m \circ \rho_m^{\mathrm{R}}) \circ (\bigsqcup_m \rho_m \circ \rho_m'^{\mathrm{R}}) &= \bigsqcup_m \rho'_m \circ \rho_m^{\mathrm{R}} \circ \rho_m \circ \rho_m'^{\mathrm{R}} \\
&= \bigsqcup_m \rho'_m \circ \rho_m'^{\mathrm{R}} \\
&\sqsubseteq id_{\mathbf{D}'}.
\end{aligned}$$

Next we must check θ is mediating:

$$\begin{aligned}\theta \circ \rho_m &= \bigsqcup_{n\geq m} \rho'_n \circ \rho_n^{\mathrm{R}} \circ \rho_m \\ &= \bigsqcup_{n\geq m} \rho'_n \circ \rho_n^{\mathrm{R}} \circ \rho_n \circ f_{mn} \\ &= \bigsqcup_{n\geq m} \rho'_n \circ f_{mn} \\ &= \rho'_m.\end{aligned}$$

Finally we check uniqueness. Let $\theta' : \mathbf{D} \lhd \mathbf{D}'$ be a mediating morphism. Then,

$$\theta' = \theta' \circ id_{\mathbf{D}} = \theta' \circ (\bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}}) = \bigsqcup_m (\theta' \circ \rho_m) \circ \rho_m^{\mathrm{R}} = \bigsqcup_m \rho'_m \circ \rho_m^{\mathrm{R}}. \blacksquare$$

It turns out that we can always construct the direct limit by using the criterion of Lemma 3.

Lemma 4. *Let $\Delta = \langle \mathbf{D}_m, f_{mn} \rangle$ be a direct ω-chain of embeddings. Then there is a cone $\rho: \Delta \to \mathbf{D}$ of embedding such that $\bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}} = id_{\mathbf{D}}$.*

Proof. Let $\mathbf{D}$ be the subset of $\prod_{m\in\omega} \mathbf{D}_m$ where, $\mathbf{D} = \{d \in \prod_{m\in\omega} \mathbf{D}_m \mid \forall m. f_m^{\mathrm{R}}(d_{m+1}) = d_m\}$, with the inherited coordinatewise order. Then $\mathbf{D}$ is a subcpo of the product with $\perp_{\mathbf{D}} = \langle \perp_{\mathbf{D}_m} \rangle_{m\in\omega}$ and $\bigsqcup_n d^{(n)} = \langle \bigsqcup_n d_m^{(n)} \rangle_{m\in\omega}$. Now define a cone $\rho: \Delta \to \mathbf{D}$ by: putting for any d in $\mathbf{D}_m$

$$\rho_m(d) = \langle f_{mn}(d) \rangle_{n\in\omega},$$

where if $m > n$, $f_{mn} \stackrel{\mathrm{def}}{=} f_{nm}^{\mathrm{R}}$. One easily checks that $\rho_m(d)$ is an element of $\mathbf{D}$ and that ρ_m is continuous; further ρ_m is an embedding as one can check that ρ_m^{R} is given by: $\rho_m^{\mathrm{R}}(d) = (d \downarrow m)$.

As $f_m^{\mathrm{R}} \circ \rho_{m+1}^{\mathrm{R}}(d) = f_m^{\mathrm{R}}(d \downarrow (m+1)) = (d \downarrow m) = \rho_m^{\mathrm{R}}(d)$ we have $\rho_m^{\mathrm{R}} = f_m^{\mathrm{R}} \circ \rho_{m+1}^{\mathrm{R}}$ and so $\rho_m = \rho_m^{\mathrm{RL}} = (f_m^{\mathrm{R}} \circ \rho_{m+1}^{\mathrm{R}})^{\mathrm{L}} = \rho_{m+1}^{\mathrm{RL}} \circ f_m^{\mathrm{RL}} = \rho_{m+1} \circ f_m$ showing that ρ is a cone.

Finally as $\rho_m \circ \rho_m^{\mathrm{R}} \sqsubseteq id_{\mathbf{D}}$ on the one hand and on the other for any d in $\mathbf{D}$, $(\rho_m \circ \rho_m^{\mathrm{R}}(d)) \downarrow m = d_m$ we have, $\bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}} = id_{\mathbf{D}}$, which concludes the proof. $\blacksquare$

Theorem 1. *For any direct ω-chain, Δ, of embeddings, $\lim_{\rightarrow} \Delta$ always exists. Indeed $\rho: \Delta \to \mathbf{D}$ is universal iff $\bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}} = id_{\mathbf{D}}$.*

Proof. By Lemma 4 a cone $\rho: \Delta \to \mathbf{D}$ satisfying the conditions of Lemma 3 exists and so $\lim_{\rightarrow} \Delta$ exists. Lemma 3 shows that if $\rho': \Delta \to \mathbf{D}'$ satisfies the condition that $\bigsqcup_m \rho'_m \circ \rho_m'^{\mathrm{R}} = id_{\mathbf{D}}$ then it is universal. Conversely if ρ' is universal then there is an isomorphism $\theta: \rho \to \rho'$ and so $\bigsqcup_m \rho'_m \circ \rho_m'^{\mathrm{R}} = \bigsqcup_m (\theta \circ \rho_m) \circ (\theta \circ \rho_m)^{\mathrm{R}} = \bigsqcup_m \theta \circ \rho_m \circ \rho_m^{\mathrm{R}} \circ \theta^{-1} = \theta \circ (\bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}}) \circ \theta^{-1} = \theta \circ \theta^{-1} = id_{\mathbf{D}}$. $\blacksquare$

Suppose $\rho: \Delta \to \mathbf{D}$ is a chain where $\Delta = \langle \mathbf{D}_m, f_{mn} \rangle$. Then, as remarked earlier, we can consider that all the ρ_m are inclusions $\rho_m: \mathbf{D}_m \subseteq \mathbf{D}$. Then each f_{mn} is also an inclusion as for d in $\mathbf{D}_m$ we have $f_{mn}(x) = \rho_n^{\mathrm{R}} \circ \rho_n \circ f_{mn}(x) = \rho_n^{\mathrm{R}} \circ \rho_m(x) = \rho_n^{\mathrm{R}}(x) = \rho_n^{\mathrm{R}} \circ \rho_n(x) = x$. So we have $\mathbf{D}_0 \subseteq \mathbf{D}_1 \subseteq \cdots$ and $\mathbf{D} \supseteq \bigcup_m \mathbf{D}_m$. Suppose now that ρ is universal; then each element x of $\mathbf{D}$ is the lub of its best approximants in $\mathbf{D}_n$ as: $x = \bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}}(x) = \bigsqcup_m \rho_m^{\mathrm{R}}(x)$, and so $x \sqsubseteq y$ iff $\forall m. \rho_m^{\mathrm{R}}(x) \sqsubseteq \rho_m^{\mathrm{R}}(y)$. Thus we can consider $\mathbf{D}$ as being $\bigcup_m \mathbf{D}_m$ with lubs added to turn it into a cpo.

For example in the case where the $\mathbf{D}_m$ are all flat all chains of best approximants will be trivial (eventually constant) and we will have $\mathbf{D} = \bigcup_m \mathbf{D}_m$, as is the case with the universal cone to $\mathbf{N}$ which we discussed above.

Continuing the analogy we now see what corresponds to continuous functions, which are those which preserve lubs of increasing ω-chains. Now monotonic functions preserves increasing ω-chains and upper bounds of them. Analogously, if $F: \mathcal{CPO}^{\mathrm{E}} \to \mathcal{CPO}^{\mathrm{E}}$ is a covariant functor and $\rho: \Delta \to \mathbf{D}$ is a cone where $\Delta = \langle \mathbf{D}_m, f_{mn} \rangle$ then $F(\rho): F(\Delta) \to F(\mathbf{D})$ is also a cone where $F(\Delta) \stackrel{\mathrm{def}}{=} \langle F(\mathbf{D}_m), F(f_{mn}) \rangle$ and $F(\rho) \stackrel{\mathrm{def}}{=} \langle F(\rho_m) \rangle_{m\in\omega}$. More generally for a covariant functor of several arguments, $F: (\mathcal{CPO}^{\mathrm{E}})^n \to \mathcal{CPO}^{\mathrm{E}}$ if $\rho^{(i)}: \Delta^{(i)} \to \mathbf{D}^{(i)}$ are cones where $\Delta^{(i)} = \langle \mathbf{D}_m^{(i)}, f_{mm'}^{(i)} \rangle$ (for $i < n$) then

$$F(\rho^{(0)}, \ldots, \rho^{(n-1)}): F(\Delta^{(0)}, \ldots, \Delta^{(n-1)}) \to F(\mathbf{D}^{(0)}, \ldots, \mathbf{D}^{(n-1)})$$

is a cone where

$$F(\Delta^{(0)}, \ldots, \Delta^{(n-1)}) \stackrel{\text{def}}{=} \langle F(\mathbf{D}_m^{(0)}, \ldots, \mathbf{D}_m^{(n-1)}), F(f_{mm'}^{(0)}, \ldots, f_{mm'}^{(n-1)}) \rangle$$

and

$$F(\rho^{(0)}, \ldots, \rho^{(n-1)}) \stackrel{\text{def}}{=} \langle F(\rho_m^{(0)}, \ldots, \rho_m^{(n-1)}) \rangle_{m \in \omega}.$$

Definition 4. A covariant functor, $F\colon (\mathcal{CPO}^{\mathrm{E}})^n \to \mathcal{CPO}^{\mathrm{E}}$ is *continuous* iff whenever $\rho^{(i)}\colon \Delta^{(i)} \to \mathbf{D}^{(i)}$ are universal cones (for $i < n$) so is $F(\rho^{(0)}, \ldots, \rho^{(n-1)})\colon F(\Delta^{(0)}, \ldots, \Delta^{(n-1)}) \to F(\mathbf{D}^{(0)}, \ldots, \mathbf{D}^{(n-1)})$. Note that this definition implies that $F(\lim_{\to} \Delta^{(0)}, \ldots, \lim_{\to} \Delta^{(n-1)}) \cong \lim_{\to} F(\Delta^{(0)}, \ldots, \Delta^{(n-1)})$; essentially it is obtained from the latter condition by adding uniformity conditions on the morphisms.

Example. We show that exponentiation is continuous. Let $\rho\colon \Delta \to \mathbf{D}$ and $\rho'\colon \Delta' \to \mathbf{D}'$ be universal cones of embeddings. Then $(\rho \to^{\mathrm{E}} \rho')\colon (\Delta \to^{\mathrm{E}} \Delta') \to (\mathbf{D} \to^{\mathrm{E}} \mathbf{D}')$ is universal as we calculate:

$$\begin{aligned}
\bigsqcup_m (\rho \to^{\mathrm{E}} \rho')_m \circ (\rho \to^{\mathrm{E}} \rho')_m^{\mathrm{R}} &= \bigsqcup_m (\rho_m \to^{\mathrm{E}} \rho'_m) \circ (\rho_m \to^{\mathrm{E}} \rho'_m)^{\mathrm{R}} \\
&= \bigsqcup_m (\rho_m^{\mathrm{R}} \to \rho'_m) \circ (\rho_m \to \rho_m'^{\mathrm{R}}) \\
&= \bigsqcup_m (\rho_m \circ \rho_m^{\mathrm{R}}) \to (\rho'_m \circ \rho_m'^{\mathrm{R}}) \\
&= (\bigsqcup_m \rho_m \circ \rho_m^{\mathrm{R}}) \to (\bigsqcup_m \rho'_m \circ \rho_m'^{\mathrm{R}}) \\
&\qquad\qquad \text{exponentiation acts continuously on morphisms} \\
&= id_{\mathbf{D}} \to id_{\mathbf{D}'} \qquad \text{by Theorem 1} \\
&= id_{\mathbf{D} \to^{\mathrm{E}} \mathbf{D}'}.
\end{aligned}$$

and by Theorem 1 that shows that $\rho \to \rho'$ is universal.

Rather than consider all our other constructions separately we formulate a general result.

Definition 5. Let $T\colon (\mathcal{CPO})_{\perp}^{m+n} \to \mathcal{CPO}_{\perp}$ be contravariant in its first m arguments and covariant in its following n arguments. Then T is *locally continuous* iff whenever $\langle f_l^{(i)} \rangle_{l \in \omega}$ is an increasing chain of continuous functions, $f_l^{(i)}\colon \mathbf{D}'_i \to_{\perp} \mathbf{D}_i$, for $i < m$ and also $\langle g_l^{(j)} \rangle_{l \in \omega}$ is an increasing chain of continuous functions $g_l^{(j)}\colon \mathbf{E}_j \to_{\perp} \mathbf{E}'_j$ for $j < n$ then

$$T(f_l^{(0)}, \ldots, f_l^{(m-1)}, g_l^{(0)}, \ldots, g_l^{(n-1)})\colon T(\mathbf{D}_0, \ldots, \mathbf{D}_{m-1}, \mathbf{E}_0, \ldots, \mathbf{E}_{n-1}) \to_{\perp} T(\mathbf{D}'_0, \ldots, \mathbf{D}'_{m-1}, \mathbf{E}'_0, \ldots, \mathbf{E}'_{n-1})$$

is an increasing chain of continuous functions with the lub, $T(\bigsqcup_l f_l^{(0)}, \ldots, \bigsqcup_l f_l^{(m-1)}, \bigsqcup_l g_l^{(0)}, \ldots, \bigsqcup_l g_l^{(n-1)})$.

Clearly any locally-continuous functor is locally-monotonic. All our functors, $\prod$, $\to$, $\bigotimes$, $\to_{\perp}$, $\sum$, $(\cdot)_{\perp}$, $(\cdot)^*$, P_i, $K_{\mathbf{D}}$ are locally-continuous.

Theorem 2. *Let $T\colon \mathcal{CPO}_{\perp}^{m+n} \to \mathcal{CPO}_{\perp}$ be a locally continuous functor as in Definition 5. Then T^{E} is continuous.*

Proof. The proof is similar to the case of exponentiation and is left to the reader. ∎

We should check that continuity is preserved under composition. Suppose $F_i\colon (\mathcal{CPO}^{\mathrm{E}})^m \to \mathcal{CPO}^{\mathrm{E}}$ $(i < n)$ are n continuous functors of degree m and $F\colon (\mathcal{CPO}^{\mathrm{E}})^n \to \mathcal{CPO}^{\mathrm{E}}$ is also continuous. Then so is the composition $F(F_0, \ldots, F_{n-1})\colon (\mathcal{CPO}^{\mathrm{E}})^m \to \mathcal{CPO}^{\mathrm{E}}$ of F with $F_0, \ldots, F_{n-1}$. For suppose $\rho^{(0)}, \ldots, \rho^{(m-1)}$ are universal cones. Then as the F_i are all continuous the cones $F_i(\rho^{(0)}, \ldots, \rho^{(m-1)})$ are also universal. But then as F is continuous the cone $F(F_0(\rho^{(0)}, \ldots, \rho^{(m-1)}), \ldots, F_{n-1}(\rho^{(0)}, \ldots, \rho^{(m-1)})) = F(F_0, \ldots, F_{n-1})(\rho^{(0)}, \ldots, \rho^{(m-1)})$ is also universal.

This concludes all the basic technical material needed to solve recursive domain equations. We recapitulate the analogy, as developed so far, in a table.

Order $x \sqsubseteq y$	Morphism $f\colon \mathbf{D} \lhd \mathbf{E}$
Quasi-order $x \sqsubseteq x$ $(x \sqsubseteq y \wedge y \sqsubseteq z) \Rightarrow x \sqsubseteq z$	**Category** $id_{\mathbf{D}}\colon \mathbf{D} \lhd \mathbf{D}$ $(f\colon \mathbf{D} \lhd \mathbf{E} \wedge g\colon \mathbf{E} \lhd \mathbf{F}) \Rightarrow (g \circ f)\colon \mathbf{D} \lhd \mathbf{F}$
Least element $\perp \sqsubseteq x$	**Initial element** $\exists! f\colon \mathbf{U} \lhd \mathbf{D}$
Monotonic function $x \sqsubseteq y \Rightarrow fx \sqsubseteq fy$ (and several arguments)	**Covariant functor** $f\colon \mathbf{D} \lhd \mathbf{E} \Rightarrow Ff\colon F\mathbf{D} \lhd F\mathbf{E}$ $F(id_{\mathbf{D}}) = id_{F\mathbf{D}}$ $F(g \circ f) = F(g) \circ F(f)$ (and several arguments)
Lub of ω-chain $x_n \sqsubseteq \bigsqcup_n x_n$ $(\forall n. x_n \sqsubseteq y) \Rightarrow \bigsqcup_n x_n \sqsubseteq y$	**Direct limit of ω-chain** $\rho\colon \Delta \to \lim_{\to} \Delta$ $(\rho'\colon \Delta \to \mathbf{E}) \Rightarrow (\exists! \theta\colon \rho \lhd \rho')$
Continuous function $f(\bigsqcup_n x_n) = \bigsqcup_n f(x_n)$ (and several arguments)	**Continuous functor** $(\rho \text{ universal}) \Rightarrow (F(\rho) \text{ universal})$ (and several arguments)

Exercises

1. Suppose we add a class **Proc** of (parameterless) procedure variables to the syntax of *Imp* and add procedure declaration of limited scope (rather than the unbounded scope considered in this chapter so far) and calling clauses to the syntax of *Stat*:

$$s ::= (\textbf{letproc}\ p\ \textbf{be}\ s\ \textbf{in}\ s) \mid (\textbf{call}\ p)$$

Show that it is possible to give a denotational semantics without needing any recursively specified cpos.

2. Show that $\mathbf{P\omega} \times \mathbf{P\omega} \lhd \mathbf{P\omega}$, $\mathbf{T}^\omega \times \mathbf{T}^\omega \lhd \mathbf{T}^\omega$, $\mathbf{Tapes} \lhd \mathbf{Btrees}$, but that $(\mathbf{P\omega} \to \mathbf{P\omega}) \not\lhd \mathbf{P\omega}$ (this is not easy), that $\mathbf{P\omega} \not\lhd \mathbf{Tapes}$, $\mathbf{Tapes} \times \mathbf{Tapes} \not\lhd \mathbf{Tapes}$, $\mathbf{T}^\omega \to \mathbf{T}^\omega \not\lhd \mathbf{T}^\omega$.

3. Show that if $\mathbf{D}$ and $\mathbf{E}$ are finite and $\mathbf{D} \lhd \mathbf{E} \lhd \mathbf{D}$ then $\mathbf{D} \cong \mathbf{E}$; show that $\mathbf{T}^\omega \lhd \mathbf{N}^\omega \lhd \mathbf{T}^\omega$ but $\mathbf{T}^\omega \not\cong \mathbf{N}^\omega$.

4. Show that $\pi_i\colon \prod_i \mathbf{D}_i \rhd \mathbf{D}_i$, $(\mathbf{D} \to \mathbf{E}) \rhd \mathbf{E}$, $\amalg_i\colon \mathbf{D}_i \lhd \sum_i \mathbf{D}_i$, $\mathbf{D} \otimes \mathbf{E} \lhd \mathbf{D} \times \mathbf{E}$, $\mathbf{D} \lhd \mathbf{D}_\perp$, $\mathbf{D} \lhd \mathbf{D}^*$. When do we have $\mathbb{O} \lhd \mathbf{D}$? Show that if $\mathbf{D}$ has a continuous glb operation, $\sqcap$, then $\sqcap\colon \mathbf{D} \times \mathbf{D} \rhd \mathbf{D}$.

5. For any continuous $p\colon \mathbf{D} \to \mathbf{D}$ let $\mathbf{Fix}_p$ be the cpo of the fixed-points of p, under the inherited order. Sometimes a $p\colon \mathbf{D} \to \mathbf{D}$ is called a projection if $p = p \circ p \sqsubseteq id_{\mathbf{D}}$. Show that if p is a projection in this sense then $\iota\colon \mathbf{Fix}_p \subseteq \mathbf{D}$ is an embedding with $\iota^{\mathrm{R}}(d) = p(d)$. Show that if $i\colon \mathbf{C} \to \mathbf{D}$ is an embedding then $i \circ j$ is a projection with $i(\mathbf{C}) = \mathbf{Fix}_{i \circ j}$.

6. The cpo $[0, 1]$ is the real interval with the usual ordering on the reals. Show that the Cantor middle third set is a projection of it and it is a projection of the Cantor middle third set. (The Cantor middle third set is the set of all elements of $[0, 1]$ whose ternary expansion contains no 2's.) What are the relations between $[0, 1]$ and $[-\infty, +\infty]$ and $[0, +\infty]$?

7. Let $\mathbf{Tapes}_n$ $(n \leq \omega)$ be the subcpo of $\mathbf{Tapes}$ consisting of all tapes of length $\leq n$. Show that the inclusions $\mathbf{Tapes}_m \subseteq \mathbf{Tapes}_n \subseteq \mathbf{Tapes}$ $(m \leq n)$ are embeddings and hence find a universal cone to $\mathbf{Tapes}$. Do the same sort of thing for $\mathbf{P\omega}$, $\mathbf{T}^\omega$, $\boldsymbol{\omega}$ and $\mathbf{Btrees}$.

8. A pair $\mathbf{D} \xrightarrow{i} \mathbf{E} \xrightarrow{j} \mathbf{D}$ of continuous maps is an adjoint pair iff $j \circ i \sqsupseteq id_{\mathbf{D}}$ and $i \circ j \sqsubseteq id_{\mathbf{E}}$ and we say that i is the *left* adjoint and j is the *right* adjoint. Show that $\langle i, j\rangle$ is an adjoint pair iff $(\forall x\colon \mathbf{D}, y\colon \mathbf{E}. x \sqsubseteq jy \text{ iff } ix \sqsubseteq y)$. Show that if $\mathbf{D} \xrightarrow{i'} \mathbf{E} \xrightarrow{j'} \mathbf{D}$ is another adjoint pair then $i \sqsubseteq i'$ iff $j \sqsupseteq j'$, proving that left and right adjoints are unique (and we have i^{R}, j^{L} as before). Let $\langle i, j\rangle$ be a fixed adjoint pair from now on. Show that i is an embedding iff i is 1-1 iff j is onto. If $j \circ i \sqsupseteq id_{\mathbf{D}}$ and $i \circ j = id_{\mathbf{E}}$ we say that i (and also $j \circ i$) is a *closure*, ($f\colon \mathbf{D} \to \mathbf{D}$ is a closure in general iff $f = f \circ f \sqsupseteq id_{\mathbf{D}}$). Show that j is a closure iff i is onto iff j is into. Show i is an isomorphism iff it is an embedding and a closure. Show that $i = i \circ j \circ i$ and $j = j \circ i \circ j$. Show that i is completely additive (preserves all lubs) and that j is completely multiplicative (preserves all glbs), generating strictness as $\perp = \bigsqcup_{\mathbf{D}} \emptyset = \bigsqcap_{\mathbf{D}} \mathbf{D}$. Show that i factorises as:

$$\begin{tikzcd} \mathbf{D} \arrow[r, "i'"] \arrow[dr, "i"'] & \mathbf{X} \arrow[d, "i''"] \\ & \mathbf{E} \end{tikzcd}$$

where i' is a closure and i'' is an embedding. What goes wrong with the general theory if we use left adjoints instead of embeddings? Can it be repaired? (This is an interesting open question which is probably not hard.)

9. Show that $\mathbf{P}\boldsymbol{\omega} \times \mathbf{P}\boldsymbol{\omega}$ and $\mathbf{P}\boldsymbol{\omega} \to \mathbf{P}\boldsymbol{\omega}$ are both closures of $\mathbf{P}\boldsymbol{\omega}$. Show that there are no non-trivial closures of **Tapes** or $\mathbf{T}^\omega$.

10. Find an increasing chain of isomorphisms, $\theta_n\colon \mathbf{D} \cong \mathbf{D}$ such that $\bigsqcup_n \theta_n$ is not an embedding or a projection; find a decreasing chain of isomorphisms $\theta_n\colon \mathbf{D} \cong \mathbf{D}$ whose glb, $\bigsqcap_n \theta_n$, exists but is not an embedding or a projection. Show that neither of $(\cdot)^{\mathrm{L}}$, $(\cdot)^{\mathrm{R}}$ are even monotonic.

11. Is $\prod^{\mathrm{E}} \mathbf{D}_i$ the categorical product in $\mathcal{CPO}^{\mathrm{E}}$? What about the other constructions.

12. Define the natural "forgetful functor" $V\colon \mathcal{CPO}^{\mathrm{E}} \to \mathcal{CPO}_\perp$ and show that if $T\colon \mathcal{CPO}_\perp \to \mathcal{CPO}_\perp$ is covariant and locally monotonic then T^{E} is the unique functor, $F\colon \mathcal{CPO}^{\mathrm{E}} \to \mathcal{CPO}^{\mathrm{E}}$ such that the following diagram of functors commutes:

$$\begin{tikzcd} \mathcal{CPO}^{\mathrm{E}} \arrow[r, "F"] \arrow[d, "V"'] & \mathcal{CPO}^{\mathrm{E}} \arrow[d, "V"] \\ \mathcal{CPO}_\perp \arrow[r, "T"'] & \mathcal{CPO}_\perp \end{tikzcd}$$

Extend this to covariant functors of several arguments. What can you do with general functors which may be mixed covariant and contravariant? [*Hint* Look at $\mathcal{CPO}^{\mathrm{D}}$ which has as morphisms pairs $\mathbf{D} \xrightarrow{f}_\perp \mathbf{E} \xrightarrow{g}_\perp \mathbf{D}$ of strict continuous functions.]

13. Show that if $\rho\colon \Delta \to \mathbf{D}$ is universal, where $\Delta = \langle \mathbf{D}_m, f_{mn} \rangle$, then so is $\rho^-\colon \Delta^- \to \mathbf{D}$ where $\Delta^- = \langle \mathbf{D}_{m+1}, f_{(m+1)(n+1)} \rangle$ and $\rho^- = \langle \rho_{m+1} \rangle_{m \in \omega}$. In other words dropping a few terms makes no difference to universality. What happens if we drop an infinite subset, but still leave an infinite subset?

14. Fill in the missing details in the proof of Lemma 3.

15. Suppose $\rho\colon \langle \mathbf{D}_m, f_{mn} \rangle \to \mathbf{D}$ is a universal cone of embeddings and the ρ_m and f_{mn} are all inclusions. Suppose too that all increasing ω-chains in any of the $\mathbf{D}_m$ are trivial. Let $\bigcup_m \mathbf{D}_m$ be the partial order where for x in $\mathbf{D}_m$, y in $\mathbf{D}_n$, $x \sqsubseteq y$ iff $x \sqsubseteq_{\mathbf{D}_l} y$ where $l = max(m, n)$. Show that the monotonic inclusion map, $\iota\colon \bigcup_m \mathbf{D}_m \to \mathbf{D}$ is universal in the sense that if $f\colon \bigcup_m \mathbf{D}_m \to \mathbf{E}$ is any other monotonic map then there is a unique continuous map $h\colon \mathbf{D} \to \mathbf{E}$ such that the following diagram commutes:

$$\begin{tikzcd} \bigcup_n \mathbf{D}_n \arrow[d, "\iota"'] \arrow[dr, "f"] & \\ \mathbf{D} \arrow[r, "h"'] & \mathbf{E} \end{tikzcd}$$

Show that ι need not be continuous in general. What goes wrong if you drop the assumption on the $\mathbf{D}_m$?

16. Try to axiomatise direct limits in $\mathcal{CPO}^{\mathrm{E}}$ in something of the same style we axiomatised the other constructions such as $\prod$ and $\rightarrow$.

17 Coincidence of Direct and Inverse Limits. A direct ω-chain in $\mathcal{CPO}_\perp$ is a structure $\Delta = \langle \mathbf{D}_m, f_{mn}\rangle$ where $f_{mn}: \mathbf{D}_m \rightarrow_\perp \mathbf{D}_n$ $(m \le n)$ and then a cone $\rho: \Delta \rightarrow \mathbf{D}$ in $\mathcal{CPO}_\perp$ is a system of $\rho_m: \mathbf{D}_m \rightarrow_\perp \mathbf{D}$ making the appropriate triangles commute. Define a universal cone in $\mathcal{CPO}_\perp$ and show that if Δ is a direct ω-chain in $\mathcal{CPO}^{\mathrm{E}}$ and $\rho: \Delta \rightarrow \mathbf{D}$ is a universal cone in $\mathcal{CPO}^{\mathrm{E}}$ then it is universal in $\mathcal{CPO}_\perp$ too. An *inverse* ω-chain in $\mathcal{CPO}_\perp$ is a structure $\Delta = \langle \mathbf{D}_m, f_{mn}\rangle$ where $f_{mn}: \mathbf{D}_n \rightarrow_\perp \mathbf{D}_m$ $(m \le n)$. Define cones $\rho: \mathbf{D} \rightarrow \Delta$ and universal cones $\rho: \mathbf{D} \rightarrow \Delta$ in $\mathcal{CPO}_\perp$ (and here we write $\mathbf{D} = \lim_{\leftarrow} \Delta$, the inverse limit). Show that if $\rho: \langle \mathbf{D}_m, f_{mn}\rangle \rightarrow \mathbf{D}$ is a universal cone in $\mathcal{CPO}^{\mathrm{E}}$ then $\langle \rho^{\mathrm{R}}_m\rangle_{m\in\omega}: \mathbf{D} \rightarrow \langle \mathbf{D}_m, f^{\mathrm{R}}_{mn}\rangle$ is a universal cone in $\mathcal{CPO}_\perp$.

18 Inverse Limits. Show that both $\mathcal{CPO}$ and $\mathcal{CPO}_\perp$ have inverse limits of inverse ω-chains.

19 Equalisers. If $f, g: \mathbf{D} \rightarrow \mathbf{E}$ are morphisms (in any category) an equaliser is a morphism $h: \mathbf{C} \rightarrow \mathbf{D}$ such that $f \circ h = g \circ h$ and (universal property) such that if $h': \mathbf{C}' \rightarrow \mathbf{D}$ is any morphism with that property then there is a unique $\theta: \mathbf{C}' \rightarrow \mathbf{C}$ such that $h' = h \circ \theta$. Show that $\mathcal{CPO}$, $\mathcal{CPO}_\perp$ but not $\mathcal{CPO}^{\mathrm{E}}$ always have equalisers. Show that equalisers in $\mathcal{CPO}_\perp$ are order-monics. Are order-monics always equalisers? Show that embeddings are order-monics too, but the converse fails in general.

20 Direct Limits. Show that $\mathcal{CPO}$ does not have direct limits of direct ω-chains, but that $\mathcal{CPO}_\perp$ does. [*Warning* This is rather hard.]

21 Co-Equalisers. If $f, g: \mathbf{D} \rightarrow \mathbf{E}$ are morphisms (in any category) a co-equaliser is a morphism $h: \mathbf{E} \rightarrow \mathbf{F}$ such that $h \circ f = h \circ g$ (universal property) such that if $h': \mathbf{E} \rightarrow \mathbf{F}'$ is any other morphism with that property then there is a unique $\theta: \mathbf{F} \rightarrow \mathbf{F}'$ such that $h' = \theta \circ h$. Which of $\mathcal{CPO}$, $\mathcal{CPO}_\perp$, $\mathcal{CPO}^{\mathrm{E}}$ always have co-equalisers? [*Warning* This is rather hard. The same difficulty as in Exercise 5 arises.]

22. We can generalise from sums of cpos over sets $\sum_{i\in S} \mathbf{D}_i$ to sums over cpos. Let $\mathbf{D}$ be a cpo, which can as remarked in this chapter be considered as a category. A functor $F: \mathbf{D} \rightarrow \mathcal{CPO}_\perp$ is continuous iff whenever $\langle d_n\rangle_{n\in\omega}$ is an increasing ω-chain in $\mathbf{D}$ with lub d then $\rho: \Delta \rightarrow F(d)$ is universal where $\Delta = \langle F(d_n), F(d_n \rightarrow d_{n+1})\rangle$ and $\rho_n = F(d_n \rightarrow d)$. [Everything works just as well if we use functors $F: \mathbf{D} \rightarrow \mathcal{CPO}^{\mathrm{E}}$ instead.]

Now if $\mathbf{D}$ is a cpo and $F: \mathbf{D} \rightarrow \mathcal{CPO}_\perp$ is an ω-continuous functor, then $\sum_{\mathbf{D}} F$ is defined as the set $\{\langle d, e\rangle \mid e \in F(d)\}$ ordered by:

$$\langle d, e\rangle \sqsubseteq \langle d', e'\rangle \quad \text{iff} \quad d \sqsubseteq d' \quad \text{and} \quad F(d \rightarrow d')(e) \sqsubseteq_{F(d')} e'.$$

Show that $\sum_{\mathbf{D}} F$ is a cpo. Show that by suitable choice of $\mathbf{D}$ and F, the sum $\sum_{\mathbf{D}} F$ can be any one of: $\mathbf{A}_\perp$, $\mathbf{A}_\perp + \mathbf{B}_\perp$, $\mathbf{A} \times \mathbf{B}$, $\sum_{i\in S} (\mathbf{A}_i)_\perp$.

23. We can generalise from products of cpos over sets to products over cpos too. Let $\mathbf{D}$ be a cpo and let $F: \mathbf{D} \rightarrow \mathcal{CPO}_\perp$ be a continuous functor as in Exercise 22. A function $f: \mathbf{D} \rightarrow \bigcup_{d\in\mathbf{D}} F(d)$ is *well-typed* iff $\forall d \in \mathbf{D}. f(d) \in F(d)$; further it is *monotonic* iff $\forall d, d' \in \mathbf{D}. d \sqsubseteq d' \rightarrow (F(d \rightarrow d')(f(d)) \sqsubseteq f(d'))$; further it is *continuous* iff whenever $\langle d_n\rangle_{n\in\omega}$ is an increasing ω-chain with lub d then $f(d) = \bigsqcup_n F(d_n \rightarrow d)(f(d_n))$.

We define $\prod_{\mathbf{D}} F$ to be the set $\{f: \mathbf{D} \rightarrow \bigcup_{d\in\mathbf{D}} F(d) \mid f \text{ is well-typed, monotonic and continuous}\}$ with the pointwise ordering:

$$f \sqsubseteq g \quad \text{iff} \quad \forall d \in \mathbf{D}. f(d) \sqsubseteq_{F(d)} g(d).$$

Show that $\prod_{\mathbf{D}} F$ is a cpo. Show that by a suitable choice of $\mathbf{D}$ and F the product $\prod_{\mathbf{D}} F$ can be any one of: $\mathbf{A} \times \mathbf{B}$, $\mathbf{A} \rightarrow \mathbf{B}$, $\prod_{i\in S} \mathbf{A}_i$.

24. Let $\mathbf{D}$ be a cpo and $F: \mathbf{D} \rightarrow \mathcal{CPO}_\perp$ be a continuous functor as used in Exercises 22 and 23. Using an informal notation which we will leave informal here, define the natural function, $in: \prod_{\mathbf{D}} \lambda d: \mathbf{D}. (F(d) \rightarrow_\perp \sum_{\mathbf{D}} F)$ by:

$$in(d) = \lambda e: F(d). \langle d, e\rangle.$$

Show in is well-defined. Show that if $\mathbf{E}$ is any cpo and $f: \prod_{\mathbf{D}} \lambda d: \mathbf{D}. (F(d) \rightarrow_\perp \mathbf{E})$ then there is a unique continuous $[f]: \sum_{\mathbf{D}} F \rightarrow_\perp \mathbf{E}$ such that for all d in $\mathbf{D}$ the following diagram commutes:

$$\begin{tikzcd}
F(d) \arrow[r, "in(d)"] \arrow[dr, "f(d)"'] & \sum_{\mathbf{D}} F \arrow[d, "{[f]}"] \\
& \mathbf{E}
\end{tikzcd}$$

Show that $[\cdot]\colon [\prod_{\mathbf{D}} \lambda d\colon \mathbf{D}.(F(d) \to_{\perp} \mathbf{E})] \cong (\sum_{\mathbf{D}} F \to_{\perp} \mathbf{E})$. Give internal and external axiomatisations for $\sum_{\mathbf{D}} F$ (perhaps without worrying too much about the details of the metalanguage). Investigate $\prod_{\mathbf{D}} F$ along similar lines.

5. Solving Recursive Domain Equations

With the aid of our grand analogy we can now solve recursive domain equations by following the idea of the fixed-point theorem. Let $F: \mathcal{CPO}^{\mathrm{E}} \to \mathcal{CPO}^{\mathrm{E}}$ be a continuous functor, which is analogous to a continuous function, $f: \mathbf{D} \to \mathbf{D}$. One fixed-point of f is the lub of the increasing ω-chain $\langle f^m(\perp)\rangle_{m\in\omega}$. As $\mathbf{U}$ is the analogous of $\perp$, we should find a direct ω-chain $\Delta = \langle F^m(\mathbf{U}), f_{mn}\rangle$ and as direct limits are the analogue of lubs of ω-chains we should then consider a universal cone, $\rho: \Delta \to (\lim_{\to} \Delta)$. It remains to specify the f_m $(= f_{m(m+1)})$. Now f_0 must be $\perp: \mathbf{U} \to F(\mathbf{U})$ and it seems natural to take $f_m = T^m(\perp)$. With this choice we can calculate:

$$F(\varinjlim \Delta) \cong \varinjlim\langle F(F^m(\mathbf{U})), F(F^m(\perp))\rangle \qquad F \text{ is continuous}$$

$$= \varinjlim \Delta \qquad \text{dropping a term does not affect direct limits — see Exercise 4.13.}$$

and we have indeed the equation $\mathbf{D} \cong F(\mathbf{D})$. Note that the f_m are determined by the requirement that the above proof works. A little analysis shows what the isomorphism actually is. As $\rho: \Delta \to \lim_{\to} \Delta$ is universal we have $F\rho: F\Delta \to F(\lim_{\to} \Delta)$ is universal as F is continuous and on the other hand $\rho^-: \Delta^- \to \lim_{\to} \Delta$ is universal too where $\Delta^- = \langle F^{m+1}(\mathbf{U}), F^{m+1}(\perp)\rangle$ and $\rho^- = \langle \rho_{m+1}\rangle_{m\in\omega}$ by the considerations of Exercise 4.13. But $F\Delta = \Delta^-$ and so there is an isomorphism $\theta: F\rho \cong \rho^-$, which by the considerations of Theorem 4 is $\bigsqcup_m \rho_{m+1} \circ (F\rho_m)^{\mathrm{R}}$, and θ^{-1} is $\bigsqcup_m (F\rho_m) \circ \rho_{m+1}^{\mathrm{R}}$. We usually write $\theta: F(\lim_{\to} \Delta) \cong \lim_{\to} \Delta$ as η_F.

It is also quite easy to solve simultaneous equations. Suppose for example we have two continuous functors, $F_i: (\mathcal{CPO}^{\mathrm{E}})^2 \to \mathcal{CPO}^{\mathrm{E}}$ $(i = 0,1)$. Define $\Gamma = \langle \mathbf{C}_n, f_n\rangle$, $\Delta = \langle \mathbf{D}_n, g_n\rangle$ by:

$$\mathbf{C}_0 = \mathbf{D}_0 = \mathbf{U};$$

$$\mathbf{C}_{n+1} = F_0(\mathbf{C}_n, \mathbf{D}_n); \quad \mathbf{D}_{n+1} = F_1(\mathbf{C}_n, \mathbf{D}_n);$$

$$f_0 = \perp; \quad g_0 = \perp;$$

$$f_{n+1} = F_0(f_n, g_n); \quad g_{n+1} = F_1(f_n, g_n).$$

Now we can calculate:

$$F_0(\varinjlim \Gamma, \varinjlim \Delta) \cong \varinjlim F_0(\Gamma, \Delta) = \varinjlim \Gamma^- = \varinjlim \Gamma$$

and similarly

$$F_1(\varinjlim \Gamma, \varinjlim \Delta) \cong \varinjlim \Delta.$$

All in all we have essentially proved that:

Theorem 1. *Let $F_i: (\mathcal{CPO}^{\mathrm{E}})^n \to \mathcal{CPO}$ $(i < n)$ be n continuous functors then there are cpos $\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}$ such that:*

$$\mathbf{D}_i \cong F_i(\mathbf{D}_0, \ldots, \mathbf{D}_{n-1}) \quad (i < n).$$

Proof. A slight generalisation of the above consideration and left to the reader. ∎

In view of our analogy it is now natural to ask if our solutions to recursive domain equations are initial (analogue of least) in some sense. However we postpone this discussion for a while as we can now solve many problems in denotational semantics by using the solution. We now present a series of typical examples of the use of recursive domain equations.

Example 1 The Integers. We have already seen that $\mathbf{N}$ satisfies the equation $\mathbf{N} \cong \mathbb{O} + \mathbf{N}$ and so $\mathbf{N} \cong T(\mathbf{N})$ where $T = +^{\mathrm{E}}(K^{\mathrm{E}}_{\mathbf{0}}, Id^{\mathrm{E}})$. Then we calculate $T(\mathbf{U})$, $T^2(\mathbf{U})$, …, $T^n(\mathbf{U})$, … are:

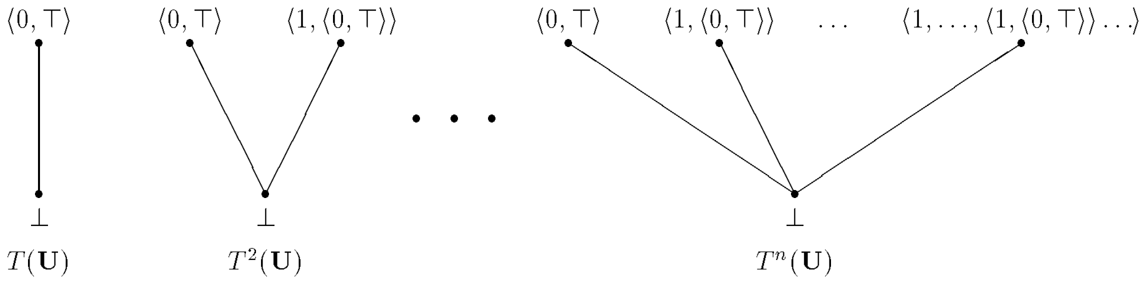


Figure XIV.

So $\mathbf{U} \subseteq T(\mathbf{U}) \subseteq T^2(\mathbf{U}) \subseteq \cdots \subseteq T^n(\mathbf{U}) \subseteq \cdots$. Further the embeddings $T^m(\perp)$ are the inclusions as $\perp$ is and T preserves inclusions. Since the $T^m(\mathbf{U})$ are all discrete it follows that $\mathbf{D} = \lim_{\to}\langle T^m(\mathbf{U}), T^m(\perp)\rangle$ is $\bigcup_m T^m(\mathbf{U})$ with the evident (flat) ordering (following the considerations of Chapter 4). Thus the solution $\mathbf{D}$ is isomorphic to $\mathbf{N}$.

Example 2 Procedures in ***Imp*****.** We consider the extension of *Imp* discussed in Chapter 4. By Theorem 1 there are solutions to the simultaneous recursive domain equations say:

$$\alpha\colon \mathbf{C} \cong (\mathbf{S} \times \mathbf{PEnv}) \to (\mathbf{S} \times \mathbf{PEnv}),$$
$$\beta\colon \mathbf{PEnv} \cong (\mathbf{Proc}_\perp \to_\perp \mathbf{C}).$$

The semantic functions $\mathcal{B}\colon \mathbf{BExp} \to (\mathbf{S} \to \mathbf{T})$, $\mathcal{A}\colon \mathbf{Act} \to (\mathbf{S} \to \mathbf{S})$ are assumed given, as before, and $\mathcal{C}\colon \mathbf{Stat} \to \mathbf{C}$ is defined by structural induction:

(i) $\mathcal{C}[\![a]\!] = \alpha^{-1}(\lambda\sigma\colon \mathbf{S}, \pi\colon \mathbf{PEnv}.\langle \mathcal{A}[\![a]\!](\sigma), \pi\rangle)$;

(ii) $\mathcal{C}[\![\mathbf{dummy}]\!] = \alpha^{-1}(\lambda\sigma\colon \mathbf{S}, \pi\colon \mathbf{PEnv}.\langle\sigma, \pi\rangle)$;

(iii) $\mathcal{C}[\![s_0; s_i]\!] = \alpha^{-1}(\alpha(\mathcal{C}[\![s_1]\!]) \circ \alpha(\mathcal{C}[\![s_0]\!]))$;

(iv) $\mathcal{C}[\![\mathbf{if}\ b\ \mathbf{then}\ e_0\ \mathbf{else}\ e_1]\!] = \alpha^{-1}(\lambda\sigma\colon \mathbf{S}, \pi\colon \mathbf{PEnv}.\mathcal{B}[\![b]\!](\sigma) \to \alpha(\mathcal{C}[\![e_0]\!])(\sigma, \pi) \mid \alpha(\mathcal{C}[\![e_1]\!])(\sigma, \pi))$;

(v) $\mathcal{C}[\![\mathbf{while}\ b\ \mathbf{do}\ s]\!] = \mu\gamma\colon \mathbf{C}.\alpha^{-1}(\lambda\sigma\colon \mathbf{S}, \pi\colon \mathbf{PEnv}.\mathcal{B}[\![b]\!](\sigma) \to \alpha(\gamma)(\alpha(\mathcal{C}[\![s]\!])(\sigma, \pi)) \mid \langle\sigma, \pi\rangle)$;

(vi) $\mathcal{C}[\![\mathbf{proc}\ p\ \mathbf{begin}\ s\ \mathbf{end}]\!] = \alpha^{-1}(\lambda\sigma\colon \mathbf{S}, \pi\colon \mathbf{PEnv}.\langle\sigma, \pi[\mathcal{C}[\![s]\!]/p]\rangle)$;

(vii) $\mathcal{C}[\![\mathbf{call}\ p]\!] = \alpha^{-1}(\lambda\sigma\colon \mathbf{S}, \pi\colon \mathbf{PEnv}.\alpha(\beta(\pi)[\![p]\!])(\sigma, \pi))$.

Here $\cdot[\cdot/\cdot]\colon \mathbf{PEnv} \times \mathbf{C} \times \mathbf{Proc} \to \mathbf{PEnv}$ is defined by:

$$\pi[\gamma/p] = \beta^{-1}(\lambda p'\colon \mathbf{Proc}.p' = p \to \gamma \mid \beta(\pi)[\![p']\!]).$$

It is annoying to continually write all the isomorphisms such as α, α^{-1}, β, β^{-1} above and we usually omit them.

Example 3 An Applied $\boldsymbol{\lambda}$-calculus.

Syntax.

1. **Id** — a set of *identifiers*, ranged over by x.
2. **Exp** — the set of *expressions* ranged over by e and given in terms of **Id** and **Exp** by the syntax:

$$e ::= x \mid \mathbf{0} \mid \mathbf{true} \mid \mathbf{false} \mid \mathbf{succ} \mid \mathbf{pred} \mid \mathbf{zero} \mid \delta \mid (\mathbf{if}\ e\ \mathbf{then}\ e\ \mathbf{else}\ e) \mid e(e) \mid \lambda x.e.$$

Semantic Domains. Expressions will have values in a domain of values, $\mathbf{V}$, which should contain $\mathbf{T}$ and $\mathbf{N}$ and since explicit self-application is allowed its function space $(\mathbf{V} \to \mathbf{V})$ is also included and we define $\mathbf{V}$ recursively by:

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

The use of the semi-separated sum is to differentiate the least function from $\perp$; the idea is that δ which is to discriminate between functions and basic values $(\mathbf{T} + \mathbf{N})$ should be, in some sense, sequential.

Denotations. The denotation function is $\mathcal{E}\colon \mathbf{Exp} \to (\mathbf{Env} \to \mathbf{V})$ where $\mathbf{Env} = \mathbf{Id} \to_\perp \mathbf{V}$. It is defined by structural induction on expressions:

(i) $\mathcal{E}[\![x]\!] = \lambda\rho\colon \mathbf{Env}.\rho[\![x]\!]$;

(ii) $\mathcal{E}[\![\mathbf{0}]\!] = \lambda\rho\colon \mathbf{Env}.in_1(0)$;

(iii) $\mathcal{E}[\![\mathbf{true}]\!] = \lambda\rho\colon \mathbf{Env}.in_0(tt)$;

(iv) $\mathcal{E}[\![\mathbf{false}]\!] = \lambda\rho\colon \mathbf{Env}.in_0(f\!f)$;

(v) $\mathcal{E}[\![\mathbf{succ}]\!] = \lambda\rho\colon \mathbf{Env}.in_2 \circ up(\lambda v\colon \mathbf{V}.in_1((v \mid \mathbf{N}) + 1))$;

(vi) $\mathcal{E}[\![\mathbf{pred}]\!] = \lambda\rho\colon \mathbf{Env}.in_2 \circ up(\lambda v\colon \mathbf{V}.in_1((v \mid \mathbf{N}) - 1))$;

(vii) $\mathcal{E}[\![\mathbf{zero}]\!] = \lambda\rho\colon \mathbf{Env}.in_2 \circ up(\lambda v\colon \mathbf{V}.in_0 \circ Z(v \mid \mathbf{N}))$;

(viii) $\mathcal{E}[\![\delta]\!] = \lambda\rho\colon \mathbf{Env}.in_2 \circ up(\lambda v\colon \mathbf{V}.in_0 \circ is_2(v))$;

(ix) $\mathcal{E}[\![\mathbf{if}\ e_0\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2]\!] = \lambda\rho\colon \mathbf{Env}.((\mathcal{E}[\![e_0]\!](\rho) \mid \mathbf{T}) \to \mathcal{E}[\![e_1]\!](\rho) \mid \mathcal{E}[\![e_2]\!](\rho))$;

(x) $\mathcal{E}[\![e_0(e_1)]\!] = \lambda\rho\colon \mathbf{Env}.down(out_2(\mathcal{E}[\![e_0]\!](\rho)))(\mathcal{E}[\![e_1]\!](\rho))$;

(xi) $\mathcal{E}[\![\lambda x.e]\!] = \lambda\rho\colon \mathbf{Env}.in_2(up(\lambda v\colon \mathbf{V}.\mathcal{E}[\![e]\!](\rho[v/x])))$;

where $\cdot[\cdot/\cdot]\colon \mathbf{Env} \times \mathbf{V} \times \mathbf{Id} \to \mathbf{Env}$ is defined much as usual.

Example 4 Input/Output.

Syntax. We extend *Imp* with a command for inputting from an input channel and outputting to an output channel (we assume one of each). So the syntactic sets are **BExp**, **Act**, as before and **Stat** is given by:

$$s ::= a \mid \mathbf{dummy} \mid \mathbf{read} \mid \mathbf{write} \mid (s; s) \mid (\mathbf{if}\ b\ \mathbf{then}\ s\ \mathbf{else}\ s) \mid (\mathbf{while}\ b\ \mathbf{do}\ s).$$

Semantic Domains. As well as a flat cpo of states, $\mathbf{S}$, there will be an input cpo $\mathbf{I}$ and one output cpo $\mathbf{O}$ (e.g. $\mathbf{I} = \mathbf{N}$ and $\mathbf{O} = \mathbf{T}$). There are also two basic functions: $read\colon \mathbf{I} \otimes \mathbf{S} \to_\perp \mathbf{S}$, $write\colon \mathbf{S} \to_\perp \mathbf{O}$ (for example read might put the input in a standard location and write might take the output from some standard location). Commands are now elements of a cpo $\mathbf{C} = \mathbf{S} \to_\perp \mathbf{R}$ where $\mathbf{R}$ is a domain of results (or resumptions). The cpo $\mathbf{R}$ contains $\mathbf{S}$, for final states, $\mathbf{I} \to_\perp \mathbf{R}$, for reading and $\mathbf{O} \otimes \mathbf{R}_\perp$ for outputting. So it is recursively defined by:

$$\mathbf{R} \cong \mathbf{S} + (\mathbf{I} \to_\perp \mathbf{R})_\perp + (\mathbf{O} \otimes \mathbf{R}_\perp).$$

Note that in this case there is not implicit or explicit self-application as can be seen from the fact that on the right hand side $\mathbf{R}$ never occurs in the contravariant position of an exponentiation functor.

Denotations. We have a given $\mathcal{B}\colon \mathbf{BExp} \to (\mathbf{S} \to \mathbf{T})$ and a given $\mathcal{A}\colon \mathbf{Act} \to (\mathbf{S} \to \mathbf{S})$ and $\mathcal{C}\colon \mathbf{Stat} \to \mathbf{C}$ is defined by structural induction on statements. We only give a few of the equations here.

(iii) $\mathcal{C}[\![\mathbf{read}]\!] = \lambda_\perp \sigma\colon \mathbf{S}.in_1(up(\lambda_\perp i\colon \mathbf{I}.in_0(read(i \otimes \sigma))))$;

(iv) $\mathcal{C}[\![\mathbf{write}]\!] = \lambda_\perp \sigma\colon \mathbf{S}.in_2(write(\sigma) \otimes up(in_0(\sigma)))$;

(v) $\mathcal{C}[\![s_0; s_1]\!] = \lambda_\perp \sigma\colon \mathbf{S}.\mathcal{C}[\![s_0]\!](\sigma) * \mathcal{C}[\![s_1]\!]$.

Here $*\colon \mathbf{R} \times \mathbf{C} \to \mathbf{R}$ is defined recursively by:

$$\begin{aligned} r * \gamma = {} & (is_0(r) \to \gamma(out_0(r)) \\ & \mid (is_1(r) \to in_1(up(\lambda_\perp i\colon \mathbf{I}.(down(out_1(r)))(i) * \gamma)) \\ & \mid (is_2(r) \to in_2((out_2(r) \downarrow 0) \otimes up(down(out_2(r) \downarrow 1) * \gamma)) \\ & \mid \perp))). \end{aligned}$$

Example 5 Dynamic Binding. This is a version of *Dec* in which, as well as the usual static binding mechanism in which, in an expression of the form ($\mathbf{let}\ x\ \mathbf{be}\ e_0\ \mathbf{in}\ e_1$), the variable x is assigned the value of e_0 calculated in the "definition-time" environment, we also include a more "dynamic" version in which the calculation of e_0 is postponed until the value of x is required for the evaluation of e_1 and then the "use-time" environment is used.

Syntax.

1. **Id** — a class of identifiers ranged over by x.
2. **Exp** — a class of expressions ranged over by e and given by:

$$e ::= x \mid \mathbf{0} \mid \mathbf{1} \mid (e + e) \mid (\mathbf{let}\ x\ \mathbf{bestat}\ e\ \mathbf{in}\ e) \mid (\mathbf{let}\ x\ \mathbf{bedyn}\ e\ \mathbf{in}\ e).$$

For example $\mathbf{let}\ x\ \mathbf{bestat}\ \mathbf{2}\ \mathbf{in}\ (\mathbf{let}\ y\ \mathbf{bedyn}\ x + \mathbf{1}\ \mathbf{in}\ (\mathbf{let}\ x\ \mathbf{bedyn}\ \mathbf{10}\ \mathbf{in}\ x + y))$ where $\mathbf{2}$ abbreviates $\mathbf{1} + \mathbf{1}$ and so on, will have value 21 whereas if static assignment had been used throughout the expression would have had value 13.

Semantic Domains. The cpo of environments, **Env**, rather than being $\mathbf{Id} \to_\perp \mathbf{N}$ must give an identifier a function of **Env** itself in order that the value of the identifier may depend on the call-time environment. So we define **Env** recursively by:

$$\mathbf{Env} \cong \mathbf{Id}_\perp \to_\perp (\mathbf{Env} \to \mathbf{N}).$$

Denotations. The denotation function is $\mathcal{E}\colon \mathbf{Exp} \to (\mathbf{Env} \to \mathbf{N})$ and it is defined by structural induction on expressions; we only give a few of the equations.

(i) $\mathcal{E}[\![x]\!] = \lambda\rho\colon \mathbf{Env}.\rho[\![x]\!](\rho)$;

(v) $\mathcal{E}[\![\mathbf{let}\ x\ \mathbf{bestat}\ e_0\ \mathbf{in}\ e_1]\!] = \lambda\rho\colon \mathbf{Env}.\mathcal{E}[\![e_1]\!](\rho[\lambda\rho'\colon \mathbf{Env}.\mathcal{E}[\![e_0]\!](\rho)/x])$;

(vi) $\mathcal{E}[\![\mathbf{let}\ x\ \mathbf{bedyn}\ e_0\ \mathbf{in}\ e_1]\!] = \lambda\rho\colon \mathbf{Env}.\mathcal{E}[\![e_1]\!](\rho[\lambda\rho'\colon \mathbf{Env}.\mathcal{E}[\![e_0]\!](\rho')/x])$.

Here $\cdot[\cdot/\cdot]\colon \mathbf{Env} \times (\mathbf{Env} \to \mathbf{N}) \times \mathbf{Id} \to \mathbf{Id}$ is defined as usual.

Example 6 Syntax Considered as a CPO. As a matter of fact we could have considered syntax as a cpo much earlier. For example, in the case of *Imp*, we would consider the flat cpo $\mathbf{Stat}_\perp$. However we would have had to define various functions over $\mathbf{Stat}_\perp$ and it turns out that if we define $\mathbf{Stat}_\perp$ by a recursive domain equation we get them for free, so to speak. Further as the recursive domain equation follows the grammar closely, we have a reasonably systematic method. (The caveat is inserted as it is not so clear how to accommodate non-context-free grammars, although we will always expect to have some sort of phrase structure.)

Syntax. There are three *cpos* of syntactic items.

1. **BExp** — a given flat cpo of *boolean expressions* ranged over by b.
2. **Act** — a given flat cpo of *primitive actions* ranged over by a.
3. **Stat** — a cpo of statements specified by the recursive domain equation:

$$\mathbf{Stat} \cong \mathbf{Act} + \mathbb{O} + \mathbf{Stat} \otimes \mathbf{Stat} + \mathbf{BExp} \otimes \mathbf{Stat} \otimes \mathbf{Stat}.$$

Note how the definition follows the syntactical specification in Chapter 1. Each alternative is modelled by a (strict) sum and the various components of a clause are put together by the strict product. This gives a flat cpo, which is a kind of abstract syntax in that there is no commitment as to how particular statements are to be written.

It is particularly tedious when using such syntactical domains to continually make type conversions using *in*, *out*, $\langle\cdot,\cdot\rangle$, $(\cdot\downarrow i)$, *up*, *down*, etc. We therefore define some helpful extra functions:

$$\begin{aligned}
isact &= is_0;\\
actof &= out_0;\\
mkact &= in_0;\\
isdum &= is_1;\\
dumof &= out_1;\\
mkdum &= in_1;\\
isseq &= is_2;\\
seqof_0 &= \pi_0 \circ out_2;\\
seqof_1 &= \pi_1 \circ out_2;\\
mkseq &= in_2 \circ \otimes;\\
iscond &= is_3;\\
condof_0 &= \pi_0 \circ out_3;\\
condof_1 &= \pi_1 \circ out_3;\\
condof_2 &= \pi_2 \circ out_3;\\
mkcond &= in_3 \circ \otimes.
\end{aligned}$$

Semantic Domains. There are the flat cpos, $\mathbf{T}$, of truth-values and $\mathbf{N}$, of integers and also the cpo $\mathbf{C} = (\mathbf{S} \to_\perp \mathbf{S})$ of commands.

Denotations. The functions $\mathcal{B}\colon \mathbf{BExp} \to_\perp (\mathbf{S} \to_\perp \mathbf{T})$ and $\mathcal{A}\colon \mathbf{Act} \to_\perp \mathbf{C}$ are assumed given and are now elements of cpos. The functions $\mathcal{C}\colon \mathbf{Stat} \to_\perp \mathbf{C}$ is defined by *recursion* as:

$$\begin{aligned}
\mathcal{C} = \lambda_\perp s\colon \mathbf{Stat}.(&isact(s) \to \mathcal{A}[\![actof(s)]\!] \mid\\
&(isdum(s) \to (\lambda_\perp \sigma\colon \mathbf{S}.\sigma) \mid\\
&(isseq(s) \to \mathcal{C}[\![seqof_1(s)]\!] \circ \mathcal{C}[\![seqof_0(s)]\!] \mid\\
&(iscond(s) \to \lambda_\perp \sigma\colon \mathbf{S}.(\mathcal{B}[\![condof_0(s)]\!](\sigma) \to \mathcal{C}[\![condof_1(s)]\!](\sigma) \mid \mathcal{C}[\![condof_2(s)]\!](\sigma)) \mid \perp)))).
\end{aligned}$$

Once we have the idea of specifying syntax in this way we can explore other kinds of equations for **Stat**. For example we might want to allow partially specified programs such as $(\perp; (a; \perp))$ (where $s; s'$ is $mkseq(s, s')$ and so on) other than $\perp$. This in turn will lead to infinite programs as the limit of increasing chains of the partial ones. There is a standard choice available: replace every occurrence of a recursively specified domain, $\mathbf{D}$, on the right hand side by $(\mathbf{D})_\perp$ to allow non-strict combination with $\perp$. So we would now specify **Stat** by:

$$\mathbf{Stat} \cong \mathbf{Act} + \mathbb{O} + \mathbf{Stat}_\perp \otimes \mathbf{Stat}_\perp + \mathbf{BExp} \otimes \mathbf{Stat}_\perp \otimes \mathbf{Stat}_\perp$$

which is the same as (see Exercise 3.18):

$$\mathbf{Stat} \cong \mathbf{Act} + \mathbb{O} + (\mathbf{Stat} \times \mathbf{Stat})_\perp + \mathbf{BExp} \otimes (\mathbf{Stat} \times \mathbf{Stat})_\perp.$$

It would then be necessary to make a few changes in the definitions of the auxiliary functions. For example we would put:

$$\begin{aligned} isseq &= is_2; \\ seqof_0 &= down \circ \pi_0 \circ out_2; \\ seqof_1 &= down \circ \pi_1 \circ out_2; \\ mkseq &= in_2 \circ \otimes \circ \langle up, up \rangle. \end{aligned}$$

But then with these changes made the denotational function would have exactly the same definition as before. Note that we now have some identifications such as: $\perp_{\mathbf{Stat}}; \perp_{\mathbf{Stat}} = \perp_{\mathbf{Stat}}$ and $(\mathbf{if}\ \perp_{\mathbf{BExp}}\ \mathbf{then}\ e_0\ \mathbf{else}\ e_1) = \perp_{\mathbf{Stat}}$, but not others such as $(\perp_{\mathbf{Stat}}; s) = \perp_{\mathbf{Stat}}$ which we might want. All this is a matter of the choice of domain equation.

We now see how our solution is initial and concentrate on a fixed functor $F: \mathcal{CPO}^{\mathrm{E}} \rightarrow \mathcal{CPO}^{\mathrm{E}}$ of one argument. Recall that the solution we constructed was obtained from a universal cone $\rho: \Delta \rightarrow \lim_{\rightarrow} \Delta$ where $\Delta = \langle F^m(\mathbf{U}), F^m(\perp) \rangle$. Let us choose a fixed ρ for F and call $\lim_{\rightarrow} \Delta$, $\mathbf{Fix}_F$. The isomorphism $\eta_F: F(\mathbf{Fix}_F) \rightarrow \mathbf{Fix}_F$ is $\eta_F = \bigsqcup_m \rho_{m+1} \circ F(\rho_m)^{\mathrm{R}}$ and its inverse is $\bigsqcup_m F(\rho_m) \circ \rho^{\mathrm{R}}_{m+1}$, as we saw above. Our solution of the domain equation, $\mathbf{D} \cong F(\mathbf{D})$, clearly consists of the pair $\langle \mathbf{Fix}_F, \eta_F \rangle$.

Definition 1. A *fixed-point* of F is a pair $\langle \mathbf{D}, \alpha \rangle$ where $\alpha: F\mathbf{D} \cong \mathbf{D}$. A *prefixed-point* of F is a pair $\langle \mathbf{D}, \alpha \rangle$ where $\alpha: F\mathbf{D} \lhd \mathbf{D}$.

Sometimes prefixed-points are called *F-algebras* for reasons to be seen below.

Definition 2. Let $\langle \mathbf{D}, \alpha \rangle$ and $\langle \mathbf{D}', \alpha' \rangle$ be prefixed-points of F. A morphism $h: \langle \mathbf{D}, \alpha \rangle \rightarrow \langle \mathbf{D}', \alpha' \rangle$ (sometimes called an *F-homomorphism*) is an embedding $h: \mathbf{D} \lhd \mathbf{D}'$ such that the following diagram commutes:

$$\begin{tikzcd} F\mathbf{D} \arrow[r, "\alpha"] \arrow[d, "Fh"'] & \mathbf{D} \arrow[d, "h"] \\ F\mathbf{D}' \arrow[r, "\alpha'"'] & \mathbf{D}' \end{tikzcd}$$

We already know that $\langle \mathbf{Fix}_F, \eta_F \rangle$ is a fixed-point of F; we now find the analogue of the fixed-point theorem:

Theorem 2 The Initial Fixed-point Theorem. *The fixed-point $\langle \mathbf{Fix}_F, \eta_F \rangle$ is the initial prefixed-point of F.*

Proof. Suppose $\alpha: F\mathbf{D} \lhd \mathbf{D}$ is a prefixed-point of F. Put $\mathbf{F}_m = F^m(\mathbf{U})$, $f_m = F^m(\perp)$, $\Delta = \langle \mathbf{F}_m, f_m \rangle$ and then as $\rho: \Delta \rightarrow \mathbf{Fix}_F$ is the universal cone, each triangle:

$$\begin{tikzcd} F(\mathbf{F}_m) \arrow[d, "F(\rho_m)"'] \arrow[dr, "\rho_{m+1}"] & \\ F(\mathbf{Fix}_F) \arrow[r, "\eta_F"'] & \mathbf{Fix}_F \end{tikzcd}$$

commutes.

For uniqueness, suppose $h\colon \mathbf{Fix}_F \lhd \mathbf{D}$ is a morphism of F-algebras so that this square commutes:

$$\begin{tikzcd}
F(\mathbf{Fix}_F) \arrow[r, "\eta_F"] \arrow[d, "Fh"'] & \mathbf{Fix}_F \arrow[d, "h"] \\
F(\mathbf{D}) \arrow[r, "\alpha"'] & \mathbf{D}
\end{tikzcd}$$

Now $h \circ \rho_0 = \perp$ and $h \circ \rho_{m+1} = h \circ \eta_F \circ F(\rho_m) = \alpha \circ F(h) \circ F(\rho_m) = \alpha \circ F(h \circ \rho_m)$. So each $h \circ \rho_m$ is determined and so too therefore is h as $h = h \circ id_{\mathbf{Fix}_F} = \bigsqcup_m (h \circ \rho_m) \circ \rho_m^{\mathrm{R}}$.

For existence define $\rho'_m\colon \mathbf{F}_m \lhd \mathbf{D}$ by: $\rho'_0 = \perp$, $\rho'_{m+1} = \alpha \circ F(\rho'_m)$. Then $\rho' \stackrel{\text{def}}{=} \langle \rho'_m \rangle_{m \in \omega}$ is a cone from Δ to $\mathbf{D}$, i.e. each of these triangles commutes:

$$\begin{tikzcd}
\mathbf{F}_m \arrow[r, "f_m"] \arrow[dr, "\rho'_m"'] & \mathbf{F}_{m+1} \arrow[d, "\rho'_{m+1}"] \\
 & \mathbf{D}
\end{tikzcd}$$

For $\rho'_1 \circ f_0 = \rho'_1 \circ \perp = \perp = \rho'_0$ and then, by induction, $\rho'_{m+2} \circ f_{m+1} = \alpha \circ F(\rho'_{m+1}) \circ F(f_m) = \alpha \circ F(\rho'_m) = \rho'_{m+1}$.

Hence there is a mediating $h\colon \mathbf{Fix}_F \lhd \mathbf{D}$ from ρ to ρ' so that all of these triangles commute:

$$\begin{tikzcd}
\mathbf{F}_m \arrow[r, "\rho_m"] \arrow[dr, "\rho'_m"'] & \mathbf{Fix}_F \arrow[d, "h"] \\
 & \mathbf{D}
\end{tikzcd}$$

But this makes the above homomorphism square commute. For both $h \circ \eta_F$ and $\alpha \circ Fh$ mediate between the universal cone $F\rho$ and $(\rho')^{-}\colon F\Delta \to \mathbf{D}$ as can be seen from the following two diagrams:

$$\begin{tikzcd}
F(\mathbf{F}_m) \arrow[r, "F\rho_m"] \arrow[d, "\rho'_{m+1}"'] \arrow[dr, "\rho_{m+1}"'] & F(\mathbf{Fix}_F) \arrow[d, "\eta_F"] \\
\mathbf{D} & \mathbf{Fix}_F \arrow[l, "h"]
\end{tikzcd}
\qquad
\begin{tikzcd}
F(\mathbf{F}_m) \arrow[r, "F\rho_m"] \arrow[d, "\rho'_{m+1}"'] \arrow[dr, "F\rho'_m"'] & F(\mathbf{Fix}_F) \arrow[d, "Fh"] \\
\mathbf{D} & F(\mathbf{D}) \arrow[l, "\alpha"]
\end{tikzcd}$$

This concludes the proof. ∎

What we would like next would be a proof rule, an axiomatisation, for recursively defined domains. However this is not available at present and constitutes an interesting open problem. There are techniques available for particular cones. Investigations of typeless λ-calculi have looked at the $F^m(\mathbf{U})$ in detail by labelling expressions and subexpressions by integers, for example $e^{(m)}$, and then investigating $\mathcal{E}[\![e^{(m)}]\!] \stackrel{\text{def}}{=} \rho_m^{\mathrm{R}}(\mathcal{E}[\![e]\!])$. Other investigations have used recursively defined inductive relations to establish the particular properties in question. The difficulty arises from contravariance as we are able to give a satisfactory axiomatisation for covariant functors $F\colon \mathcal{CPO}_\perp \to \mathcal{CPO}_\perp$ where $F \stackrel{\text{def}}{=} T^{\mathrm{E}}$. That is we have a satisfactory solution when domain equations do not contain any occurrences of the recursively specified cpos in contravariant positions (such as the first argument of an exponentiation). Our solution will be a general form of structural induction.

We begin by looking at T-algebra in $\mathcal{CPO}_\perp$, where the terminology will make good sense. We consider a fixed covariant locally continuous functor, $T{:}\,\mathcal{CPO}_\perp \to \mathcal{CPO}_\perp$.

Definition 3. A T-*algebra* is a pair $\langle \mathbf{D}, \alpha\rangle$ where $\alpha{:}\,T\mathbf{D} \to_\perp \mathbf{D}$ is a strict continuous function. A T-homomorphism, $h{:}\,\langle \mathbf{D}, \alpha\rangle \to \langle \mathbf{D}', \alpha'\rangle$ between T-algebras is a strict continuous function, $h{:}\,\mathbf{D} \to_\perp \mathbf{D}'$ such that the following diagram commutes:

$$\begin{tikzcd} T\mathbf{D} \arrow[r, "\alpha"] \arrow[d, "Th"'] & \mathbf{D} \arrow[d, "h"] \\ T\mathbf{D}' \arrow[r, "\alpha'"'] & \mathbf{D}' \end{tikzcd}$$

Example 1 The Integers. Consider the continuous algebras $A = \langle \mathbf{A}, 0, succ\rangle$ where $\mathbf{A}$ is a cpo, 0 is an element of $\mathbf{A}$, and $succ{:}\,\mathbf{A} \to_\perp \mathbf{A}$ is a strict continuous function. A morphism $h{:}\,\langle \mathbf{A}, 0, succ\rangle \to \langle \mathbf{A}', 0', succ'\rangle$ is a continuous function $h{:}\,\mathbf{A} \to_\perp \mathbf{A}'$ such that $h(0) = 0'$ and $h(succ(x)) = succ'(h(x))$, for all elements x of $\mathbf{A}$.

Now suppose for the purpose of this example that $T{:}\,\mathcal{CPO}_\perp \to \mathcal{CPO}_\perp$ is given by $T(\mathbf{D}) = \mathbb{O} + \mathbf{D}$ (more accurately, by $T = +(K_{\mathbb{O}}, Id)$). Any algebra $\langle \mathbf{A}, 0, succ\rangle$ induces the T-algebra $\alpha_{\mathbf{A}}{:}\,\mathbb{O} + \mathbf{A} \to \mathbf{A}$ where $\alpha_{\mathbf{A}} = [\lambda_\perp x.0, succ]$; conversely any T-algebra $\alpha{:}\,\mathbb{O}+\mathbf{D} \to \mathbf{D}$ induces the algebra $\mathbf{A}_\alpha = \langle \mathbf{D}, \alpha \circ inl(\top), \alpha \circ inr\rangle$. The correspondences are functorial as if $h{:}\,\mathbf{A} \to_\perp \mathbf{A}'$ is a homomorphism then $h{:}\,\langle \mathbf{A}, \alpha_{\mathbf{A}}\rangle \to \langle \mathbf{A}', \alpha_{\mathbf{A}'}\rangle$ is a T-homomorphism and conversely. Then we see that the functors are mutual inverses — that is the categories are isomorphic.

Note that we have already shown how the integers form a T-algebra $\alpha{:}\,\mathbb{O} + \mathbf{N} \to \mathbf{N}$ where $\alpha = [\lambda_\perp x.0, strict((+1))]$, and indeed α is an isomorphism. One easily proves that $\langle \mathbf{N}, \alpha\rangle$ is the initial T-algebra by induction. For suppose $\langle \mathbf{D}, \alpha'\rangle$ is another T-algebra and $h{:}\,\langle \mathbf{N}, \alpha\rangle \to_\perp \langle \mathbf{D}, \alpha'\rangle$ is a T-homomorphism. Then $h(\perp) = \perp$, $h(0) = h \circ \alpha(inl(\top)) = \alpha' \circ (id_{\mathbb{O}} + h)(inl(\top)) = \alpha'(inl(\top))$ and $h(n+1) = h \circ \alpha(inr(n)) = \alpha' \circ (id_{\mathbb{O}} + h)(inr(n)) = \alpha'(h(n))$. So it follows by ordinary induction that h is determined. One easily shows the converse: that if h is defined by the above equation (namely $h(0) = h \circ \alpha(inl(\top))$ and $h(n+1) = \alpha'(h(n))$) then h is a T-homomorphism. We saw above that $\mathbf{Fix}_{T^{\mathrm{E}}}$ was isomorphic to the integers. We now see that (among many other things), $\langle \mathbf{Fix}_{T^{\mathrm{E}}}, \eta_{T^{\mathrm{E}}}\rangle$ is the initial T-algebra and so we see that $\langle \mathbf{N}, \alpha\rangle$ and $\langle \mathbf{Fix}_{T^{\mathrm{E}}}, \eta_{T^{\mathrm{E}}}\rangle$ are isomorphic as *algebras*.

Theorem 3 The Initial Algebra Theorem. *The T^{E}-algebra, $\langle \mathbf{Fix}_{T^{\mathrm{E}}}, \eta_{T^{\mathrm{E}}}\rangle$ is the initial T-algebra.*
Proof. Define $\mathbf{F}_m$, f_m, Δ, ρ as in the proof of Theorem 2. Let $\langle \mathbf{D}, \alpha\rangle$ be a T-algebra. Uniqueness of a possible T-homomorphism, $h{:}\,\langle \mathbf{Fix}_{T^{\mathrm{E}}}, \eta_{T^{\mathrm{E}}}\rangle \to_\perp \langle \mathbf{D}, \alpha\rangle$ is proved just as in Theorem 2.

For existence define $\rho'_m{:}\,\mathbf{F}_m \to_\perp \mathbf{D}$ by $\rho'_0 = \perp$, $\rho'_{m+1} = \alpha \circ T(\rho'_m)$ then, as in the proof of Theorem 2, $\rho' \stackrel{\mathrm{def}}{=} \langle \rho'_m\rangle_{m\in\omega}$ is a cone, $\rho'{:}\,\Delta \to \mathbf{D}$. By Exercise 4.17, ρ is universal in $\mathcal{CPO}_\perp$ and so there is a unique mediating $h{:}\,\rho \to \rho'$. The proof then finishes as in Theorem 2 using the fact that $T\rho$ is universal in $\mathcal{CPO}_\perp$ as it is universal in $\mathcal{CPO}^{\mathrm{E}}$ (being $T^{\mathrm{E}}\rho$). ∎

One application of this theorem is when T is chosen for the specification of a recursively defined syntactical cpo. For instance, continuing Example 6 above we can consider:

$$T = +(K_{\mathbf{Act}}, K_{\mathbb{O}}, \otimes(Id, Id), \otimes(K_{\mathbf{BExp}}, Id, Id).$$

Then if one can take the semantic domain, here $\mathbf{C} = (\mathbf{S} \to_\perp \mathbf{S})$, and turn it into a T-algebra, $\langle \mathbf{C}, \alpha\rangle$ it follows that there is a unique homomorphism $h{:}\,\langle \mathbf{Stat}, \eta_{T^{\mathrm{E}}}\rangle \to \langle \mathbf{C}, \alpha\rangle$ which we can take as the semantics. This view of semantics is called *initial-algebra* semantics. To see that it is reasonable, we note that in this case following the ideas in Example 1 above, a T-algebra amounts to a structure $\langle \mathbf{C}, mkact', mkdum', mkseq', mkcond'\rangle$ where $mkact'{:}\,\mathbf{Act} \to_\perp \mathbf{C}$ is strict, $mkdum'{:}\,\mathbb{O} \to_\perp \mathbf{C}$ is strict, $mkseq'{:}\,\mathbf{C} \times \mathbf{C} \to_\perp \mathbf{C}$ is bistrict and $mkcond'{:}\,\mathbf{BExp} \times \mathbf{C} \times \mathbf{C} \to_\perp \mathbf{C}$ is tristrict and we have already seen that for $\mathbf{Stat}$ we obtain the algebra $\langle \mathbf{Stat}, mkact, mkdum, mkseq, mkcond\rangle$. Then a homomorphism $h{:}\,\langle \mathbf{Stat}, \ldots\rangle \to \langle \mathbf{C}, \ldots\rangle$ is a strict continuous function $h{:}\,\mathbf{Stat} \to_\perp \mathbf{C}$ such that:

$$\begin{aligned} h(mkact(a)) &= mkact'(a) && (a{:}\,\mathbf{Act}),\\ h(mkdum(\top)) &= mkdum'(\top), &&\\ h(mkseq(s_0, s_1)) &= mkseq'(h(s_0), h(s_1)) && (s_0, s_1{:}\,\mathbf{Stat}),\\ h(mkcond(b, s_0, s_1)) &= mkcond'(b, h(s_0), h(s_1)) && (b{:}\,\mathbf{BExp}, s_0, s_1{:}\,\mathbf{Stat}) \end{aligned}$$

and this is just an abstract form of the usual semantics if we define:

$$\begin{aligned}
mkact' &= \mathcal{A},\\
mkdum' &= \lambda_\perp x{:}\,\mathbb{O}.(\lambda_\perp \sigma{:}\,\mathbf{S}.\sigma),\\
mkseq' &= \lambda\gamma{:}\,\mathbf{C}, \gamma'{:}\,\mathbf{C}.\gamma' \circ \gamma,\\
mkcond' &= \lambda b{:}\,\mathbf{BExp}, \gamma{:}\,\mathbf{C}, \gamma'{:}\,\mathbf{C}.\lambda_\perp \sigma{:}\,\mathbf{S}.(\mathcal{B}[\![b]\!](\sigma) \to \gamma(\sigma) \mid \gamma'(\sigma))
\end{aligned}$$

and indeed with this definition it turns out that $h = \mathcal{C}$. Of course it is equally easy to give an initial algebra semantics treatment of the other kinds of syntactic cpos considered above.

From now on, although it is not really essential, we assume that T preserves inclusions (that is if $\iota{:}\,\mathbf{D} \subseteq \mathbf{E}$ then $T(\iota){:}\,T(\mathbf{D}) \subseteq T(\mathbf{E})$). This is the case for all our covariant functors and also for each of the functors: $\to(K_{\mathbf{D}}, Id)$, $\to_\perp(K_{\mathbf{D}}, Id)$ (i.e. those obtained by fixing the contravariant argument of the exponentiation functor); further this property is preserved under the composition of functors.

Theorem 4 The Structural Induction Theorem. *Let $\alpha{:}\,T\mathbf{D} \to_\perp \mathbf{D}$ be a T-algebra. Then the following assertions are equivalent.*

(1) $\langle \mathbf{D}, \alpha\rangle$ is the initial T-algebra.

(2) The morphism, α, is a strict order-monic and further for every strict order-monic $m{:}\,\mathbf{P} \to_\perp \mathbf{D}$ if $\mathbf{P}$ can be extended to a T-algebra $\langle \mathbf{P}, \beta\rangle$ such that $m{:}\,\langle \mathbf{P}, \beta\rangle \to \langle \mathbf{D}, \alpha\rangle$ is a T-homomorphism then m is an isomorphism.

(3) The morphism m is an order-monic and further for every ω-inductive $\mathbf{P} \subseteq \mathbf{D}$ the following principle of structural induction holds:

$$(\mathbf{P}(\perp) \wedge (\forall x{:}\,T(\mathbf{D}).T(\mathbf{P})(x) \Rightarrow \mathbf{P}(\alpha x))) \Rightarrow \forall x{:}\,\mathbf{D}.\mathbf{P}(x).$$

(4) The morphism α is an isomorphism and $id_{\mathbf{D}} = \mu h.\alpha \circ T(h) \circ \alpha^{-1}$.

Proof. (1) $\Rightarrow$ (2). From Theorem 1 we know that in the initial T-algebra $(\mathbf{Fix}_{T^{\mathrm{E}}}, \eta_{T^{\mathrm{E}}})$ $\eta_{T^{\mathrm{E}}}$ is an isomorphism; so α is as well. Now suppose $m{:}\,\mathbf{P} \to_\perp \mathbf{D}$ is a strict order-monic and also $m{:}\,\langle \mathbf{P}, \beta\rangle \to \langle \mathbf{D}, \alpha\rangle$ is a T-homomorphism. As $\langle \mathbf{D}, \alpha\rangle$ is initial there is a T-homomorphism $h{:}\,\langle \mathbf{D}, \alpha\rangle \to \langle \mathbf{P}, \beta\rangle$. Then $m \circ h{:}\,\langle \mathbf{D}, \alpha\rangle \to \langle \mathbf{D}, \alpha\rangle$ is a T-homomorphism which must be $id_{\mathbf{D}}$ as $\langle \mathbf{D}, \alpha\rangle$ is initial. Now we have $m \circ (h \circ m) = m \circ id$ which shows that $h \circ m = id$ as m is a monic. Thus m is an isomorphism with inverse h.

(2) $\Rightarrow$ (1). First we show α is an isomorphism. Considering the diagram:

$$\begin{tikzcd}
T(T\mathbf{D}) \arrow[r, "T\alpha"] \arrow[d, "T\alpha"'] & T\mathbf{D} \arrow[d, "\alpha"] \\
T\mathbf{D} \arrow[r, "\alpha"'] & \mathbf{D}
\end{tikzcd}$$

we can apply (2) with $\mathbf{P} = T\mathbf{D}$ and $m = \alpha$. So α is an isomorphism.

Next let $\langle \mathbf{P}, \beta\rangle$ be the initial T^{E}-algebra, viz. $\langle \mathbf{Fix}_{T^{\mathrm{E}}}, \eta_{T^{\mathrm{E}}}\rangle$. Then as we now know that $\langle \mathbf{D}, \alpha\rangle$ is a T^{E}-algebra, as α is an embedding being an isomorphism, there must be an embedding $m{:}\,\mathbf{P} \triangleleft \mathbf{D}$ which is a homomorphism. As every embedding is a strict order-monic, we can apply (2) to see that m is an isomorphism and hence an isomorphism of the T-algebra. So $\langle \mathbf{D}, \alpha\rangle$ is isomorphic to $\langle \mathbf{P}, \beta\rangle$ which is the initial T^{E}-algebra and so, by Theorem 3, is the initial T-algebra.

(2) $\Rightarrow$ (3). Let $\mathbf{P} \subseteq \mathbf{D}$ be ω-inductive and suppose that $\mathbf{P}(\perp)$ and $\forall x{:}\,T(\mathbf{D}).T(\mathbf{P})(x) \Rightarrow \mathbf{P}(\alpha x)$. Then $\iota{:}\,\mathbf{P} \subseteq \mathbf{D}$ is a strict order-monic. Define $\beta{:}\,T\mathbf{P} \to \mathbf{P}$:

$$\beta(x) = \alpha \circ T\iota(x).$$

This is well-defined by the assumption and since $T\iota$ is an inclusion as T preserves inclusions. Next β is strict as $\beta(\perp_{T\mathbf{P}}) = \alpha(T\iota(\perp_{T\mathbf{P}})) = \alpha(\perp_{T\mathbf{D}}) = \perp_{\mathbf{D}} = \perp_{\mathbf{P}}$; also β is continuous as $\beta(\bigsqcup_{T\mathbf{P}} x_n) = \alpha(T\iota(\bigsqcup_{T\mathbf{P}} x_n)) = \alpha(\bigsqcup_{T\mathbf{D}}(T\iota(x_n))) = \bigsqcup_{\mathbf{D}} \alpha \circ T\iota(x_n) = \bigsqcup_{\mathbf{P}} \beta(x_n)$. Next the following diagram commutes:

$$\begin{tikzcd} T\mathbf{P} \arrow[r, "\beta"] \arrow[d, "T\iota"'] & \mathbf{P} \arrow[d, "\iota"] \\ T\mathbf{D} \arrow[r, "\alpha"'] & \mathbf{D} \end{tikzcd}$$

For any x in $T(\mathbf{P})$, $\iota \circ \beta(x) = \iota \circ \alpha \circ T\iota(x) = \alpha \circ T\iota(x)$. So we can apply (2) to find that ι is an isomorphism. But the ι is onto and we have that $\mathbf{P} = \mathbf{D}$.

(3) $\Rightarrow$ (2). Suppose $m: \mathbf{P} \to_\perp \mathbf{D}$ is a strict order-monic and there is a $\beta: T\mathbf{P} \to \mathbf{P}$ such that $m: \langle \mathbf{P}, \beta\rangle \to \langle \mathbf{D}, \alpha\rangle$ is a T-homomorphism. Then m factorises as $\mathbf{P} \xrightarrow{\theta} \mathbf{Q} \xrightarrow{\iota} \mathbf{D}$ where θ is an isomorphism and ι is an inclusion and $\mathbf{Q}(\perp_\mathbf{D})$ holds and $\mathbf{Q}$ is ω-inductive and we have the following commuting diagram.

$$\begin{tikzcd} T\mathbf{P} \arrow[r, "\beta"] \arrow[d, "T\theta"'] & \mathbf{P} \arrow[d, "\theta"] \\ T\mathbf{Q} \arrow[r, "T\theta^{-1} \circ \beta \circ \theta"] \arrow[d, "T\iota"'] & \mathbf{Q} \arrow[d, "\iota"] \\ T\mathbf{D} \arrow[r, "\alpha"'] & \mathbf{D} \end{tikzcd}$$

Now suppose $x{:}T(\mathbf{D})$ and $T(\mathbf{Q})(x)$ then $\alpha x = \alpha \circ T\iota(x)$ ($T\iota$ is an inclusion) $= \iota \circ (T\theta^{-1} \circ \beta \circ \theta)(x) = T\theta^{-1} \circ \beta \circ \theta(x) \in \mathbf{Q}$. So we may apply (3) to see that $\mathbf{D} \subseteq \mathbf{Q}$ and so $\iota = id_\mathbf{D}$ and $m = \theta$ is an isomorphism.

(1) $\Rightarrow$ (4). We already know (1) implies α is an isomorphism. Let $h = \mu h.\alpha \circ Th \circ \alpha^{-1}$. Then $h = \alpha \circ Th \circ \alpha^{-1}$ and so $h \circ \alpha = \alpha \circ Th$ which shows that $h: \langle \mathbf{D}, \alpha\rangle \to \langle \mathbf{D}, \alpha\rangle$ is a T-isomorphism. But then we must have $h = id_\mathbf{D}$ as $\langle \mathbf{D}, \alpha\rangle$ is initial.

(4) $\Rightarrow$ (2). Suppose $m: \mathbf{P} \to_\perp \mathbf{D}$ is a strict order-monic and $m: \langle \mathbf{P}, \beta\rangle \to \langle \mathbf{D}, \alpha\rangle$ is a T-homomorphism. Let α^{-1} be the inverse of α. Let $g = \mu g.\beta \circ Tg \circ \alpha^{-1}$. Then $g = \beta \circ Tg \circ \alpha^{-1}$ and so $g \circ \alpha = \beta \circ Tg$ and we see that the following diagram commutes.

$$\begin{tikzcd} T\mathbf{P} \arrow[r, "\beta"] \arrow[d, "Tm"'] & \mathbf{P} \arrow[d, "m"] \\ T\mathbf{D} \arrow[r, "\alpha"] \arrow[d, "Tg"'] & \mathbf{D} \arrow[d, "g"] \\ T\mathbf{P} \arrow[r, "\beta"'] & \mathbf{P} \end{tikzcd}$$

We now show that $m \circ g = id$. It will then follow that m is an isomorphism with inverse g from the fact that m is a monic, as in the first part of the proof.

First we show that $m \circ g \sqsubseteq id$ by fixed point induction on the definition of g. Clearly $m \circ \perp = \perp \sqsubseteq id$, as m is strict. Now suppose, for the sake of the induction that $m \circ g \sqsubseteq id$. Then: $m \circ (\beta \circ Tg \circ \alpha^{-1}) = (\alpha \circ Tm) \circ (Tg \circ \alpha^{-1}) = \alpha \circ T(m \circ g) \circ \alpha^{-1} \sqsubseteq id$ (by the induction hypothesis).

Finally we use the hypothesis that $id = \mu h.\alpha \circ Th \circ \alpha^{-1}$ to prove that $id \sqsubseteq m \circ g$ by fixed-point induction on the definition of $\mu h.\alpha \circ Th \circ \alpha^{-1}$. Clearly $\perp \sqsubseteq m \circ g$. Suppose $h \sqsubseteq m \circ g$. Then: $\alpha \circ Th \circ \alpha^{-1} \sqsubseteq \alpha \circ T(m \circ g) \circ \alpha^{-1} = \alpha \circ (Tm \circ Tg) \circ \alpha^{-1} = (m \circ \beta) \circ Tg \circ \alpha^{-1} = m \circ g$ (as $g = \beta \circ Tg \circ \alpha^{-1}$). This concludes the proof. ∎

Note that the only place the assumption that T preserves inclusions was used was in the proof of the equivalence of (2) and (3). If this were to fail we could just use (2) in place of (3). For information on strict order-monics consult Exercise 2.28. Part (4) is sometimes used in axiomatisations, but it is invariably used to derive a form of structural induction and we only include it for the sake of completeness.

So as our internal axiomatisation we can now just take Theorem 3.3. In this form the axioms are a little unfamiliar and we now see how they work out with a few examples.

Example 1 The Integers. Here we have the algebra $\alpha: \mathbb{O} + \mathbf{D} \cong \mathbf{D}$ where $T = +(K_{\mathbb{O}}, Id)$, $\alpha = \eta_T$ and $\mathbf{D} = \mathbf{Fix}_T$. Now we can define $0 = \alpha(inl(\top))$, $succ = \alpha \circ inr$ and $iszero = isl \circ \alpha^{-1}$, $pred = outr \circ \alpha^{-1}$. Conversely α and α^{-1} can be defined in terms of them by: $\alpha = [\lambda_\perp x\colon \mathbb{O}.0, succ]$, $\alpha^{-1} = (\lambda_\perp n\colon \mathbf{D}.iszero(n) \rightarrow inl(\top) \mid inr(pred(n)))$ as is easily verified.

The following axioms are equivalent to $\alpha^{-1} \circ \alpha = id$, as is also easily verified (and $\alpha^{-1} \circ \alpha = id$ implies α is strict order-monic):

(1) $pred(0) = \perp$,
(2) $pred(succ(x)) = x$,
(3) $iszero(0) = tt$,
(4) $iszero(succ(x)) = (\partial x \rightarrow \mathit{ff} \mid \perp)$.

We obtain essentially the usual form of induction by rewriting the main clause of the structural induction principle:

$$\begin{aligned}(\forall x\colon T(\mathbf{D}).T(\mathbf{P})(x) \Rightarrow \mathbf{P}(\alpha x)) &\equiv \forall x\colon \mathbb{O} + \mathbf{D}.(\mathbb{O} + \mathbf{P})(x) \Rightarrow \mathbf{P}(\alpha x)\\ &\equiv [\forall x\colon \mathbb{O}.\mathbf{P}(\alpha \circ inl(x))] \wedge [\forall x\colon \mathbf{P}.\mathbf{P}(\alpha \circ inr(x))]\\ &\equiv \mathbf{P}(\perp) \wedge \mathbf{P}(0) \wedge \forall x\colon \mathbf{P}.\mathbf{P}(succ(x)).\end{aligned}$$

Thus we obtain the following principle for all ω-inductive $\mathbf{P} \subseteq \mathbf{D}$ (which is to say, all $\mathbf{P} \subseteq \mathbf{D}$):

(5) $[\mathbf{P}(\perp) \wedge \mathbf{P}(0) \wedge (\forall x\colon \mathbf{P}.\mathbf{P}(succ(x)))] \Rightarrow \forall x.\mathbf{P}(x)$.

This together with (1)–(4) can be considered as a form of the Peano axioms adapted from the category of sets to $\mathcal{CPO}_\perp$. Instead of proving that $(\mathbf{N}, 0, +1)$ was also the initial T-algebra we could just observe that it satisfies the above axioms (with (-1) and Z). Then the isomorphism with $(\mathbf{D}, \eta_T)$ follows from Theorem 4.

Example 2 Tapes. Here we consider $T(\mathbf{D}) = \mathbf{T} \otimes \mathbf{D}_\perp$ and the initial algebra $\alpha: \mathbf{T} \otimes \mathbf{D}_\perp \rightarrow_\perp \mathbf{D}$. The appropriate auxiliary functions are $cons = \alpha \circ \otimes(id_{\mathbf{T}} \times up)\colon \mathbf{T} \times \mathbf{D} \rightarrow \mathbf{D}$; $head = \pi_0 \circ \alpha^{-1}\colon \mathbf{D} \rightarrow_\perp \mathbf{T}$; $tail = down \circ \pi_1 \circ \alpha^{-1}\colon \mathbf{D} \rightarrow_\perp \mathbf{D}$. Then the equivalent axioms to $\alpha^{-1} \circ \alpha = id$ are:

(1) $cons(\perp, x) = x$,
(2) $head(cons(t, x)) = t$,
(3) $tail(cons(t, x)) = (\partial t \rightarrow x \mid \perp)$.

For structural induction we look at:

$$\begin{aligned}\forall x\colon \mathbf{T} \otimes \mathbf{D}_\perp.(\mathbf{T} \otimes \mathbf{P}_\perp)(x) \Rightarrow \mathbf{P}(\alpha x) &\equiv \forall t\colon \mathbf{T}.\forall x\colon \mathbf{P}_\perp.\mathbf{P}_\perp(x) \Rightarrow \mathbf{P}(\alpha(t \otimes x))\\ &\equiv [\forall t\colon \mathbf{T}.\forall x\colon \mathbf{P}.\mathbf{P}(x) \Rightarrow \mathbf{P}(\alpha(t \otimes up(x)))] \wedge \forall t\colon \mathbf{T}.\mathbf{P}(\alpha(t \otimes \perp))\\ &\equiv [\forall t\colon \mathbf{T}.\forall x\colon \mathbf{P}.\mathbf{P}(x) \Rightarrow \mathbf{P}(cons(t, x))] \wedge \mathbf{P}(\perp)\end{aligned}$$

and we have the following structural induction principle for ω-inductive subsets of $\mathbf{D}$:

$$[\mathbf{P}(\perp) \wedge (\forall t\colon \mathbf{T}.\forall x\colon \mathbf{P}.\mathbf{P}(x) \Rightarrow \mathbf{P}(cons(t, x)))] \Rightarrow \forall x.\mathbf{P}(x).$$

Again we can use this axiomatisation to show that $\langle \mathbf{D}, \alpha \rangle$ is isomorphic to an algebra with **Tapes**.

Example 3 Results as in Example 4. Here $T = +(K_{\mathbf{S}}, (\rightarrow_\perp(K_{\mathbf{I}}, Id))_\perp, \otimes(K_{\mathbf{O}}, (Id)_\perp))$ and the initial algebra is, say, $\alpha\colon \mathbf{S} + (\mathbf{I} \rightarrow_\perp \mathbf{R})_\perp + (\mathbf{O} \otimes \mathbf{R}_\perp) \rightarrow_\perp \mathbf{R}$. An auxiliary functions we define:

$$\begin{aligned}
\mathit{mkend} &= \alpha \circ \mathit{in}_0\colon \mathbf{S} \rightarrow_\perp \mathbf{R};\\
\mathit{isend} &= \mathit{is}_0 \circ \alpha^{-1}\colon \mathbf{R} \rightarrow_\perp \mathbf{T};\\
\mathit{endof} &= \mathit{out}_0 \circ \alpha^{-1}\colon \mathbf{R} \rightarrow_\perp \mathbf{S};\\
\mathit{mkinp} &= \alpha \circ \mathit{in}_1 \circ \mathit{up}\colon (\mathbf{I} \rightarrow_\perp \mathbf{R}) \rightarrow \mathbf{R};\\
\mathit{isinp} &= \mathit{is}_1 \circ \alpha^{-1}\colon \mathbf{R} \rightarrow_\perp \mathbf{T};\\
\mathit{inpof} &= \mathit{down} \circ \mathit{out}_1 \circ \alpha^{-1}\colon \mathbf{R} \rightarrow (\mathbf{I} \rightarrow \mathbf{R});\\
\mathit{mkout} &= \alpha \circ \otimes \circ (\mathit{id} \times \mathit{up})\colon \mathbf{O} \times \mathbf{R} \rightarrow \mathbf{R};\\
\mathit{outof} &= \pi_0 \circ \mathit{out}_2 \circ \alpha^{-1}\colon \mathbf{R} \rightarrow_\perp \mathbf{O};\\
\mathit{restof} &= \mathit{down} \circ \pi_1 \circ \mathit{out}_2 \circ \alpha^{-1}\colon \mathbf{R} \rightarrow_\perp \mathbf{R};
\end{aligned}$$

The equation for $\alpha^{-1} \circ \alpha = \mathit{id}$ are:

$$\begin{array}{lll}
(1) & \mathit{endof}(\mathit{mkend}(\sigma)) = \sigma & (\sigma\colon \mathbf{S});\\
(2) & \mathit{inpof}(\mathit{mkinp}(f)) = f & (f\colon \mathbf{I} \rightarrow_\perp \mathbf{R});\\
(3) & \mathit{mkout}(\perp, r) = \perp & (r\colon \mathbf{R});\\
(4) & \mathit{outof}(\mathit{mkout}(o, r)) = o & (o\colon \mathbf{O},\ r\colon \mathbf{R});\\
(5) & \mathit{restof}(\mathit{mkout}(o, r)) = (\partial o \rightarrow r \mid \perp) & (o\colon \mathbf{O}, r\colon \mathbf{R});\\
(6) & \mathit{isend}(\mathit{mkend}(\sigma)) = (\partial \sigma \rightarrow \mathit{tt} \mid \perp) & (\sigma\colon \mathbf{S});\\
(7) & \mathit{isend}(\mathit{mkinp}(f)) = \mathit{ff} & (f\colon \mathbf{I} \rightarrow_\perp \mathbf{R});\\
(8) & \mathit{isend}(\mathit{mkout}(o, r)) = (\partial o \rightarrow \mathit{ff} \mid \perp) & (o\colon \mathbf{O},\ r\colon \mathbf{R});\\
(9) & \mathit{isinp}(\mathit{mkend}(\sigma)) = (\partial \sigma \rightarrow \mathit{ff} \mid \perp) & (\sigma\colon \mathbf{S});\\
(10) & \mathit{isinp}(\mathit{mkinp}(f)) = \mathit{tt} & (f\colon \mathbf{I} \rightarrow_\perp \mathbf{R});\\
(11) & \mathit{isinp}(\mathit{mkout}(o, r)) = (\partial o \rightarrow \mathit{ff} \mid \perp) & (o\colon \mathbf{O},\ r\colon \mathbf{R}).
\end{array}$$

For structural induction we look at: $\forall x\colon \mathbf{S} + (\mathbf{I} \rightarrow_\perp \mathbf{P})_\perp + (\mathbf{O} \otimes \mathbf{P}_\perp).\mathbf{P}(\alpha x) \equiv A \wedge B \wedge C$ where:

$$\begin{aligned}
A &= \forall \sigma\colon \mathbf{S}.\mathbf{P}(\alpha \circ \mathit{in}_0(\sigma))\\
&\equiv \forall \sigma\colon \mathbf{S}.\mathbf{P}(\mathit{mkend}(\sigma))\\
B &= \forall x\colon (\mathbf{I} \rightarrow_\perp \mathbf{P})_\perp.\mathbf{P}(\alpha \circ \mathit{in}_1(x))\\
&\equiv \mathbf{P}(\perp) \wedge \forall f\colon \mathbf{I} \rightarrow_\perp \mathbf{P}.\mathbf{P}(\alpha \circ \mathit{in}_1 \circ \mathit{up}(f))\\
&\equiv \mathbf{P}(\perp) \wedge \forall f\colon \mathbf{I} \rightarrow_\perp \mathbf{R}.[(\forall i\colon \mathbf{I}.\mathbf{P}(fi)) \Rightarrow \mathbf{P}(\mathit{mkinp}(f))]\\
C &= \forall w\colon \mathbf{O} \otimes \mathbf{P}_\perp.\mathbf{P}(\alpha \circ \mathit{in}_2(w))\\
&\equiv \forall o\colon \mathbf{O}.\forall x\colon \mathbf{P}_\perp.\mathbf{P}(\alpha \circ \mathit{in}_2(o \otimes x))\\
&\equiv \mathbf{P}(\perp) \wedge \forall o\colon \mathbf{O}.\forall r\colon \mathbf{P}.\mathbf{P}(\alpha \circ \mathit{in}_2(o \otimes \mathit{up}(r)))\\
&\equiv \mathbf{P}(\perp) \wedge \forall o\colon \mathbf{O}.\forall r\colon \mathbf{R}.(\mathbf{P}(r) \Rightarrow \mathbf{P}(\mathit{mkout}(o, r)))
\end{aligned}$$

and this gives us the following induction principle for ω-inductive subsets, $\mathbf{P}$, of $\mathbf{R}$:

$$\begin{aligned}
[\mathbf{P}(\perp) &\wedge (\forall \sigma\colon \mathbf{S}.\mathbf{P}(\mathit{mkend}(\sigma)))\\
&\wedge (\forall f\colon \mathbf{I} \rightarrow_\perp \mathbf{R}.[(\forall i\colon \mathbf{I}.\mathbf{P}(fi)) \Rightarrow \mathbf{P}(\mathit{mkinp}(f))])\\
&\wedge (\forall o\colon \mathbf{O}.\forall r\colon \mathbf{R}.(\mathbf{P}(r) \Rightarrow \mathbf{P}(\mathit{mkout}(o, r))))] \Rightarrow \forall r\colon \mathbf{R}.\mathbf{P}(r).
\end{aligned}$$

6. Algebraic Domains

Algebraicity is an important idea which formalises some intuitive ideas in CW/2 and Chapter 1 pp. 5–8 of *finiteness* (finite amount of information) and objects as *limits* of their finite approximations. All of the previous constructions easily accommodate algebraicity except for exponentiation (which will need yet another axiom). Algebraicity allows definitions of *computability* to provide links with recursive function theory and allow results on *definability*. It allows easy consideration of constructions such as *powerdomains* and enables us to visualise domains as *completions* of structures of finite information.

Definition. Let $\mathbf{D}$ be a cpo. An element d is *finite* (= *isolated* = *compact*) iff for all increasing chains, $\langle x_n \rangle$ of elements of $\mathbf{D}$, $d \sqsubseteq \bigsqcup_n x_n$ implies that there is an n such that $d \sqsubseteq x_n$.
Idea: finite — finite amount of information.

Examples

0. $\perp$ is finite.

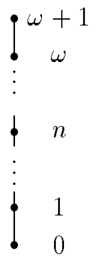
1. Every element in a discrete or a finite cpo is finite.

2. The finite elements in $\mathbf{V}_\omega$, $\mathbf{P}\boldsymbol{\omega}$, $\mathbf{T}^\omega$, **Tapes**, **Trees**, ... are the evident ones.

3. For cpos $\mathbf{D}$, $\mathbf{E}$ and finite elements d, e of $\mathbf{D}$, $\mathbf{E}$ respectively. The *step-functions* $(d \Rightarrow e)$ (or, sometimes, written $\bar{e}[d, e]$) defined by:
$$(d \Rightarrow e)(x) = \begin{cases} e & (x \sqsupseteq d), \\ \perp & \text{(otherwise)} \end{cases}$$
is continuous and finite.

4. If d, e are finite elements of $\mathbf{D}$, so is $d \sqcup e$ if it exists, but $d \sqcap e$ need not be when it exists.

5. *Finite* $\neq$ *Finite Height.* e.g. $\mathbf{V}_{\omega+1}$:



or $(K\top)$ $(= \perp \Rightarrow \top)$ in $\mathbf{N} \to \mathbb{O}$. But it is often the case as in $\mathbf{P}\boldsymbol{\omega}$, $\mathbf{T}^\omega$, **Tapes**, **Trees**, ...

Definition. A cpo is ω-*algebraic* (= *countably* algebraic) iff for every point x in $\mathbf{D}$ there is an increasing chain $\langle d_n \rangle$ of finite elements such that $x = \bigsqcup_n d_n$ and, furthermore, there are denumerably many finite elements altogether.
Idea: Points are "real" (= finite) or "ideal" (= limits).

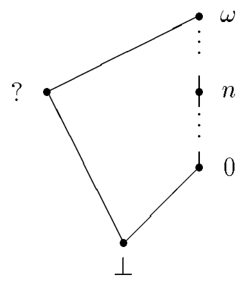
Examples

1. Discrete and finite cpos; $\mathbf{V}_\omega$, $\mathbf{P}\boldsymbol{\omega}$, $\mathbf{T}^\omega$, **Tapes**, **Trees**, ...

2. *Origin of Names.* Let $A = \langle A, + \rangle$ be an algebra with one binary operation $+ : A^2 \to A$ (say). The *subalgebras* are (essentially) the sets $X \subseteq A$ such that $X + X \subseteq X$ (i.e. closed under $+$). Under inclusion the subalgebras form a complete lattice which is ω-algebraic.

3. The interval $[0, 1]$ under $\leq$ is a cpo which is not algebraic, similarly $\mathbf{R}$ (CW/3 p. 9) is not ω-algebraic. Can you add more points to $[0, 1]$ (intervals to $\mathbf{R}$) to make it ω-algebraic? These kinds of cpos can be dealt with by an idea of ω-*continuity*.

4. The example here is not ω-algebraic:



Note. On the whole ω-continuous domains are not much used although there is hope [Nasser Saheb-Djahromi: Probabilistic Non-Determinism, CSR-7-77, Dept. of Computer Science, University of Edinburgh]. So we do not consider them in this part of the course.

Exercises

1 Alternative Definitions. One often sees the following alternative series of definitions. Let $\mathbf{D}$ be a partial order. A subset X of $\mathbf{D}$ is *directed* iff $X \neq \emptyset$ and if $d, e \in X$ then there is an x in X such that $d \sqsubseteq x$ and $e \sqsubseteq x$. Then $\mathbf{D}$ is a *dcpo* iff it has a least element, $\perp$, and every directed subset, X, has a lub, $\bigsqcup_{\mathbf{D}} X$. An element, d, of a dcpo $\mathbf{D}$ is *finite* iff for every directed set X, $d \sqsubseteq \bigsqcup X$ implies $d \sqsubseteq x$ for some member, x, of X. The dcpo $\mathbf{D}$ is *algebraic* iff for every element x, the set $B_x = \{d \in \mathbf{D} \mid d \text{ finite} \wedge d \sqsubseteq x\}$ is directed and has lub x; it is ω-*algebraic* if it is algebraic and has denumerably many finite elements. (If we want to talk about both sets of concepts at once we can write ω-cpo/dcpo ω-finite/dfinite etc.: only in the exercises will this be needed.)

Show that these two definitions of ω-algebraic coincide.

2 Topology. The *order-topology* on any cpo $\mathbf{D}$ has as open sets those sets O s.t.:

(1) $x \in O \wedge x \sqsubseteq y \Rightarrow y \in O$,
(2) $\bigsqcup_n x_n \in O \wedge \langle x_n \rangle \uparrow \Rightarrow \exists n. x_n \in O$.

Show that if $\mathbf{D}$ is ω-algebraic, the sets $O_d = \{x \in D \mid x \sqsupseteq d\}$ (d finite) form a denumerable subbasis of compact open sets for the order-topology.

3 Notions of Finiteness. Call the property of being ω-finite I_0. Define for a given ω-cpo $\mathbf{D}$:

$$I_1(d) \equiv \forall \text{ increasing } \langle d_n \rangle . d = \bigsqcup_n d_n \Rightarrow \exists n. d = d_n .$$

For some notions of finite height, first in any partial order, $\mathbf{P}$ is $a \sqsubseteq b$ are elements of $\mathbf{P}$ the interval $[a, b] = \{x \mid a \sqsubseteq x \sqsubseteq b\}$ and a *chain* in $\mathbf{P}$ is just a totally ordered subset of $\mathbf{P}$ (i.e. $X \subseteq \mathbf{P}$ is a chain iff $\forall x, y \in X. x \sqsubseteq y \vee y \sqsubseteq x$). So far in these notes only ω-chains were considered. Now define in $\mathbf{D}$:

$H_0(d) \equiv [\perp, d]$ is finite,
$H_1(d) \equiv$ there is an upper bound on the cardinality of any chain in $[\perp, d]$,
$H_2(d) \equiv$ every chain in $[\perp, d]$ is finite,
$H_3(d) \equiv$ there is no increasing ω-chain $\langle d_n \rangle$ with $d_n \sqsubseteq d$ for all n,
$H_4(d) \equiv$ there is no decreasing countably infinite chain $d = d_0 \sqsupset d_1 \sqsupset d_2 \ldots$.

Show that following implications and that these are all that hold for an arbitrary cpo.

$$I_0 \rightarrow I_1 \qquad\qquad H_0 \rightarrow H_1 \begin{matrix} \nearrow H_2 \\ \searrow H_3 \end{matrix}$$

What happens if $\mathbf{D}$ is given to be ω-algebraic?

Continuous Functions

We formalise CW/2 pp. 13, 14 and Chapter 1 p. 6. Idea (2) on p. 13 gives as "ε-δ" version of continuity. Note parts (1), (3) are *true* for ω-algebraic cpos.

Theorem 1. *Let* $\mathbf{D}$ *and* $\mathbf{E}$ *be* ω*-algebraic cpos. Then* $f\colon \mathbf{D} \to \mathbf{E}$ *is continuous iff for all* x *in* $\mathbf{D}$ *and finite* c *in* $\mathbf{E}$*:*

$$c \sqsubseteq f(x) \equiv \exists \textit{ finite } b \textit{ in } \mathbf{D}. b \sqsubseteq x \wedge c \sqsubseteq f(b).$$

Proof. $\Rightarrow$) Suppose f is continuous.

Suppose $c \sqsubseteq f(x)$ for x in $\mathbf{D}$, c finite in $\mathbf{E}$. Then $x = \bigsqcup_n b_n$ where $\langle b_n \rangle$ is an increasing ω-chain of finite elements in $\mathbf{E}$. So $c \sqsubseteq f(x) = \bigsqcup_n f(b_n)$ by continuity. So as c is finite, for some n, $c \sqsubseteq f(b_n)$.

Suppose $c \sqsubseteq f(b) \wedge b \sqsubseteq x$ for a finite b in $\mathbf{D}$. Then, by monotonicity $c \sqsubseteq f(b) \sqsubseteq f(x)$.

$\Leftarrow$) Suppose the equivalence holds. For monotonicity suppose $x \sqsubseteq y$ holds in $\mathbf{D}$, and $c \sqsubseteq f(x)$ for a finite c. Then for a finite b in $\mathbf{D}$, $b \sqsubseteq x \wedge c \sqsubseteq f(b)$. So $b \sqsubseteq y$ (as $x \sqsubseteq y$) $\wedge c \sqsubseteq f(b)$. So $c \sqsubseteq f(y)$. So $\forall$ finite c in $\mathbf{E}. c \sqsubseteq f(x) \Rightarrow c \sqsubseteq f(y)$. So $f(x) \sqsubseteq f(y)$.

For continuity suppose $\langle x_n \rangle$ is an increasing ω-chain in $\mathbf{D}$. By monotonicity, $f(\bigsqcup_n x_n) \sqsupseteq \bigsqcup_n f(x_n)$. Suppose $c \sqsubseteq f(\bigsqcup_n x_n)$ for a finite c. Then for a finite $b \sqsubseteq \bigsqcup_n x_n$, $c \sqsubseteq f(b)$. Then for some n, $b \sqsubseteq x_n$ and so $c \sqsubseteq f(b) \sqsubseteq f(x_n)$, by monotonicity, showing that for any finite c, $c \sqsubseteq f(\bigsqcup_n x_n) \Rightarrow c \sqsubseteq \bigsqcup_n f(x_n)$. Thus $f(\bigsqcup_n x_n) \sqsubseteq f(x_n)$. ∎

Exercises

1. Reformulate the ε-δ condition as the conjunction of:

(1) f is monotonic,
(2) $\forall x \in \mathbf{D}. \forall$ finite $c \in \mathbf{E}.(c \sqsubseteq f(x) \Rightarrow \exists$ finite $b \in \mathbf{D}. b \sqsubseteq x \wedge c \sqsubseteq f(b))$.

2. Use ε-δ to show that the identity is continuous, that the composition of continuous functions is continuous and that $\sqcap$ if it always exists is continuous. Is the latter true in a general cpo?

3. Show that for a continuous $f\colon \mathbf{D} \to \mathbf{E}$ between ω-algebraic cpos, $f\colon \mathbf{D} \to \mathbf{E}$ is continuous iff:

$$f = \bigsqcup \{b \Rightarrow c \mid b, c \text{ finite elements of } \mathbf{D}, \mathbf{E} \text{ respectively}\}.$$

Completion by Ideals

Every ω-algebraic cpo is determined by its basis (its set of finite elements). We investigate this determination.

Definition. A *countable pointed quasiorder* (ω*-pq*) is a structure $\langle \mathbf{P}, \sqsubseteq, \bot \rangle$ with $\mathbf{P}$ countable, $\sqsubseteq$ a quasiorder and $\bot$ the least element.

If $\mathbf{D}$ is ω-algebraic $B_{\mathbf{D}} \stackrel{\text{def}}{=} \langle \{b \in \mathbf{D} \mid b \text{ finite}\}, \sqsubseteq, \bot \rangle$ is a countable pointed po. (all inherited from $\mathbf{D}$); $B_{\mathbf{D}}$ is called the *basis* of $\mathbf{D}$. (It is convenient to let the above equation define $B_{\mathbf{D}}$ for *all* cpos $\mathbf{D}$.)
Ideas: Let $\mathbf{P}$ be a countable pointed quasiorder. A subset X of $\mathbf{P}$ is an ideal iff:

(1) *Pointed* $\quad \bot \in X$,
(2) *Left Closed* $\quad \forall y \in X. \forall x \in \mathbf{P}. x \sqsubseteq y \Rightarrow x \in X$.

An ideal is *directed* iff any two elements of the ideal have an upper bound in the ideal.

Exercise

1. Show that for any subset X of $\mathbf{P}$ there is a smallest ideal which contains it (called the *generated* ideal). Show that the directed ideals are just the ideals generated by (= least ideals containing) increasing ω-chains.

Completion

Let $\mathbf{P}$ be an ω-pq. Its *completion by ideals* is $\overline{\mathbf{P}} \stackrel{\text{def}}{=} \{X \subseteq \mathbf{P} \mid X \text{ is a directed ideal of } \mathbf{P}\}$ ordered by inclusion: $X \sqsubseteq Y$ iff $X \subseteq Y$. Let $[\cdot]\colon \mathbf{P} \to \overline{\mathbf{P}}$ be the monotonic map where $[x] \stackrel{\text{def}}{=} \{y \in \mathbf{P} \mid y \sqsubseteq x\}$.

Theorem 2.

(1) For any countable pointed quasiorder $\mathbf{P}$, $\overline{\mathbf{P}}$ *is* ω*-algebraic. Further for any other* ω*-algebraic cpo* $\mathbf{D}$ *and (strict) monotonic map* $f\colon \mathbf{P} \to \mathbf{D}$ *there is a unique continuous map* $f^\sharp\colon \overline{\mathbf{P}} \to \mathbf{D}$ *such that the following (strict) diagram commutes:*

$$\begin{tikzcd}
\mathbf{P} \arrow[d, "{[\cdot]}"'] \arrow[dr, "f"] & \\
\overline{\mathbf{P}} \arrow[r, "f^\sharp"'] & \mathbf{D}
\end{tikzcd}$$

(2) For any ω*-algebraic* $\mathbf{D}$, $\mathbf{D} \cong \overline{B_{\mathbf{D}}}$. *(Further for any countable pointed quasiorder,* $B_{\overline{\mathbf{P}}} = \{[x] \mid x \in \mathbf{P}\} = [\mathbf{P}]$, *i.e.* $\mathbf{P}/\equiv$ *where* $\equiv$ *is the equivalence associated with the quasiorder.)*

Proof. (1) The least element of $\overline{\mathbf{P}}$ is $\{\perp_{\mathbf{P}}\}$. $\bigsqcup$ is $\bigcup$ on increasing ω-chains, as is all obvious. For $b \in \mathbf{P}$, $[b]$ is finite as $[b] \sqsubseteq \bigcup X_n \Rightarrow b \in X_n$ for some X_n. For any directed ideal X there is a cofinal increasing chain x_n in X (i.e. $\forall x \in X. \exists n. x \sqsubseteq x_n$). Then $X = \bigcup [x_n]$ expresses the ideal as a lub of an increasing chain of finite elements of the form $[x]$. So every finite element has that form and $\overline{\mathbf{P}}$ is ω-algebraic.

Suppose the diagram commutes. Then:

$$\begin{aligned}
f^\sharp(X) &= f^\sharp(\bigsqcup_n [x_n]) && x_n \text{ a cofinal chain in } X\\
&= \bigsqcup_n f^\sharp([x_n])\\
&= \bigsqcup_n f(x_n).
\end{aligned}$$

So $f^\sharp$ is determined. Conversely use this equation to determine $f^\sharp$, as the definition is independent of the choice of cofinal set and check the triangle ($f^\sharp$ is strict iff f is).

(2) The isomorphism $\theta\colon \mathbf{D} \to \overline{B_{\mathbf{D}}}$ is: $\theta(x) = \{b \in B_{\mathbf{D}} \mid b \sqsubseteq x\}$. It has inverse $\theta^{-1}(X) = \bigsqcup_{\mathbf{D}} x_n$. The last assertion is part of the proof of (1). ∎

Examples

1. If, as with discrete cpos or finite ones, all points are finite then $\mathbf{D} \cong B_{\mathbf{D}}$ and $\overline{B_{\mathbf{D}}} \cong B_{\mathbf{D}}$. (So $\mathbf{D} \cong \overline{\mathbf{D}}$. Any other examples?)
2. $\mathbf{V} = \overline{\boldsymbol{\omega}}$ where $\boldsymbol{\omega} = \langle \{n \in \omega\}, \leq \rangle$. What is $\overline{\boldsymbol{\omega} \times \boldsymbol{\omega}}$? What is $\overline{\mathbf{V}_\omega}$?
3. $\mathbf{P}\boldsymbol{\omega} = \overline{\text{Finite Subsets of } \omega}$, $\mathbf{Tapes} = \overline{\text{Finite Tapes}}$ etc, etc.

Exercises

1 Cantor Space. What is $\overline{\mathbf{Rat}}$ where $\mathbf{Rat}$ is the rationals in 0, 1 with the usual order? It is not $[0, 1]$ as that is not ω-algebraic. Show $[0, 1] \lhd \overline{\mathbf{Rat}}$ showing projections need not preserve ω-algebraicity.

2 Infinite Words. Let Σ be a finite alphabet. Order $(\Sigma \cup \{\perp\})^*$ by

(1) $\perp \sqsubseteq x$,
(2) If $x \sqsubseteq x'$, $y \sqsubseteq y'$ then $xy \sqsubseteq x'y'$.

Let $\mathbf{Words} = \overline{(\Sigma \cup \{\perp\})^*}$. It is a domain of finite and infinite words. (Say $\Sigma = \{0, 1\}$.) What are the solutions to the equations $x = 0x$, $x = x0$, $x = 0x0$? Are they the same? Does $\mathbf{Words}$ have lubs/glbs of pairs of elements?

3. If $f\colon \mathbf{P} \to \mathbf{P}'$ is a monotonic function of countable pointed quasiorders then $\overline{f}\colon \overline{\mathbf{P}} \to \overline{\mathbf{P}'}$ is defined by:

$$\overline{f}(X) = \{y \in \mathbf{P}' \mid \exists x \in X. y \sqsubseteq f(x)\}.$$

Show this makes $\overline{f}$ well-defined and continuous and indeed that $\overline{f} = ([\cdot]' \circ f)^\sharp$, the unique continuous $g\colon \overline{\mathbf{P}} \to \overline{\mathbf{P}'}$ s.t. the following diagram commutes:

$$\begin{tikzcd}
\mathbf{P} \arrow[r, "f"] \arrow[d, "{[\cdot]}"'] & \mathbf{P'} \arrow[d, "{[\cdot]'}"] \\
\overline{\mathbf{P}} \arrow[r, "g"'] & \overline{\mathbf{P'}}
\end{tikzcd}$$

Conclude that the correspondence $f \mapsto \overline{f}$ is functorial i.e. that $\overline{g \circ f} = \overline{g} \circ \overline{f}$ and $\overline{id_{\mathbf{P}}} = id_{\overline{\mathbf{P}}}$. Show that the correspondence is an *isomorphism* of $\mathbf{P} \to \mathbf{P'}$, the monotonic functions from $\mathbf{P}$ to $\mathbf{P'}$, and $\overline{\mathbf{P}} \to_{\mathrm{fin}} \overline{\mathbf{P'}}$ the continuous functions from $\overline{\mathbf{P}}$ to $\overline{\mathbf{P'}}$ which preserves finiteness.

4. Show that the correspondence f to $f^\sharp$ of the theorem is an isomorphism as is shown by the following axioms (external):

$$\forall f : \mathbf{P} \to \mathbf{D}. f = f^\sharp \circ [\cdot],$$
$$\forall g : \overline{\mathbf{P}} \to \mathbf{D}. g = (g \circ [\cdot])^\sharp.$$

Note that in the case $\mathbf{P} = B_{\mathbf{C}}$ for $\mathbf{C}$ ω-algebraic we have $(\mathbf{C} \to \mathbf{D}) \cong (B_{\mathbf{C}} \to \mathbf{D})$ showing that the continuous functions are just given by monotonic functions on their base.

5. A clumsier way to define completions is to use cofinal chains. If $\langle a_n \rangle$, $\langle b_m \rangle$ are increasing chains in an ω-pq $\mathbf{P}$ then they are cofinal, written $\langle a_n \rangle \sim \langle b_m \rangle$ iff:

$$(\forall n. \exists m. a_n \sqsubseteq b_m) \wedge (\forall m. \exists n. b_m \sqsubseteq a_n).$$

Show that $\overline{\mathbf{P}} \cong$ the equivalence classes of cofinal chains in $\mathbf{P}$ ordered by:

$$[\langle a_n \rangle]_\sim \sqsubseteq [\langle b_m \rangle]_\sim \quad \text{iff} \quad \forall n. \exists m. a_n \sqsubseteq b_m.$$

6. The definition of $\overline{\mathbf{P}}$ does not use the countability of $\mathbf{P}$. Develop a version of Theorem 2, for arbitrary quasiorders, $\mathbf{P}$, and algebraic dcpos $\mathbf{D}$. Show that the map $\bigsqcup : \overline{\mathbf{D}} \to \mathbf{D}$ (where $\mathbf{D}$ is any algebraic dcpo) defined by:

$$\bigsqcup(X) = \bigsqcup_{\mathbf{D}} x$$

is continuous and universal in the sense that if $f : \overline{\mathbf{P}} \to \mathbf{D}$ is any other continuous map then there is a unique monotonic map $f^\flat : \mathbf{P} \to \mathbf{D}$ such that the following diagram commutes:

$$\begin{tikzcd}
\overline{\mathbf{D}} \arrow[d, "\bigsqcup"'] & \overline{\mathbf{P}} \arrow[l, "\overline{f^\flat}"'] \arrow[dl, "f"] \\
\mathbf{D} &
\end{tikzcd}$$

7. Show that there is a completion $\tilde{\mathbf{P}}$, which *retains existing ω-lubs*. More precisely, a function $f : \mathbf{P} \to \mathbf{Q}$ of quasiorders is continuous if for every increasing chain $\langle x_n \rangle$ in $\mathbf{P}$ s.t. $\bigsqcup_{\mathbf{P}} x_n$ exists, so does $\bigsqcup_{\mathbf{Q}} f(x_n)$ and $f(\bigsqcup_{\mathbf{P}} x_n) = \bigsqcup_{\mathbf{Q}} f(x_n)$.

We want a cpo $\tilde{\mathbf{P}}$ and a continuous map $[\cdot] : \mathbf{P} \to \tilde{\mathbf{P}}$ such that if $f : \mathbf{P} \to \mathbf{D}$ is any other continuous map with $\mathbf{D}$ a cpo then there is a unique continuous map $f^\sharp : \tilde{\mathbf{P}} \to \mathbf{D}$ s.t. the following diagram commutes:

$$\begin{tikzcd}
\mathbf{P} \arrow[d, "{[\cdot]}"'] \arrow[dr, "f"] & \\
\tilde{\mathbf{P}} \arrow[r, "f^\sharp"'] & \mathbf{D}
\end{tikzcd}$$

(*Warning* This is quite hard. If you solve it you should then be able to solve Exercises 20, 21 of Chapter 4.)

The following little lemma will be handy.

Lemma 1. *For any cpo* $\mathbf{D}$, *if* $B \subseteq B_{\mathbf{D}}$ *has the property that every element in* $\mathbf{D}$ *is a lub of an increasing* ω*-chain of elements in* B *then* $B_{\mathbf{D}} = B$ *and* $\mathbf{D}$ *is* ω*-algebraic if* B *is denumerable.*
Proof. Easy exercise. ∎

I: Products. The product $\mathbf{P} \times \mathbf{Q}$ of two countable pointed quasiorders is defined as the Cartesian product with the coordinatewise ordering. As such it is also a countable pointed quasiordering.

Fact. *For any* ω*-algebraic cpos* $\mathbf{D}$ *and* $\mathbf{E}$, $\mathbf{D} \times \mathbf{E}$ *is* ω*-algebraic and, indeed,* $B_{\mathbf{D}\times\mathbf{E}} = B_{\mathbf{D}} \times B_{\mathbf{E}}$.
Proof. If $\langle d, e\rangle \in B_{\mathbf{D}} \times B_{\mathbf{E}}$ and $\langle x_n, y_n\rangle$ is an increasing chain in $\mathbf{D} \times \mathbf{E}$ s.t. $\langle d, e\rangle \sqsubseteq \bigsqcup_n \langle x_n, y_n\rangle$ then $d \sqsubseteq \bigsqcup_n x_n$, $e \sqsubseteq \bigsqcup_n y_n$ and so for some n_1, n_2 $d \sqsubseteq x_{n_1}$, $e \sqsubseteq y_{n_2}$ and so for $n = \max(n_1, n_2)$, $\langle d, e\rangle \sqsubseteq \langle x_n, y_n\rangle$. Thus $B_{\mathbf{D}} \times B_{\mathbf{E}} \subseteq B_{\mathbf{D}\times\mathbf{E}}$.

If $\langle x, y\rangle \in \mathbf{D} \times \mathbf{E}$ then $x = \bigsqcup_n d_n$ for an increasing ω-chain of $\mathbf{D}$-finite elements and similarly $y = \bigsqcup_n e_n$. Then $\langle x, y\rangle = \bigsqcup_n \langle d_n, e_n\rangle$ expresses $\langle x, y\rangle$ as the lub of an increasing chain of elements of $B_{\mathbf{D}} \times B_{\mathbf{E}}$. So by Lemma 1, $B_{\mathbf{D}} \times B_{\mathbf{E}} = B_{\mathbf{D}\times\mathbf{E}}$ and $\mathbf{D} \times \mathbf{E}$ is ω-algebraic. ∎

Corollary. *For any two countable pointed quasiorders,* $\mathbf{P}$, $\mathbf{Q}$:

$$\overline{\mathbf{P} \times \mathbf{Q}} \cong \overline{\mathbf{P}} \times \overline{\mathbf{Q}}.$$

Proof.

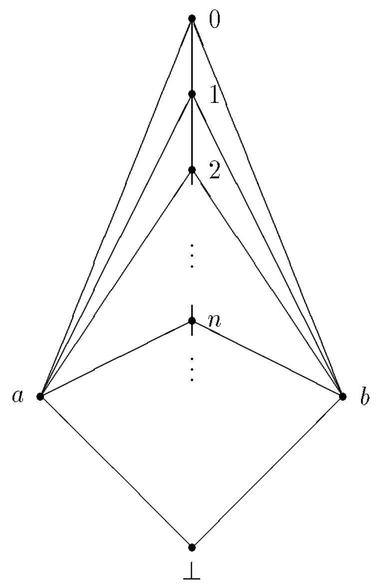
$$\begin{aligned}
B_{\overline{\mathbf{P}}\times\overline{\mathbf{Q}}} &= B_{\overline{\mathbf{P}}} \times B_{\overline{\mathbf{Q}}} && \text{by the Fact} \\
&\cong (\mathbf{P}/\equiv) \times (\mathbf{Q}/\equiv) && \text{Theorem 2} \\
&\cong (\mathbf{P} \times \mathbf{Q})/\equiv && \\
&\cong B_{\overline{\mathbf{P}\times\mathbf{Q}}} && \text{Theorem 2.} \blacksquare
\end{aligned}$$

Exercises
1. Show that $\overline{\mathbf{P}} \xleftarrow{\pi_0} \overline{\mathbf{P} \times \mathbf{Q}} \xrightarrow{\pi_1} \overline{\mathbf{Q}}$ form the universal pair of projection maps.

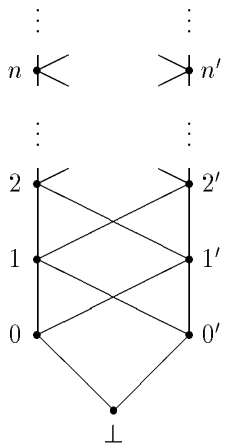
2. Show ω-algebraicity is preserved by finite and ω-products.

II: Function Space. The function space construction does not preserve ω-algebraicity.

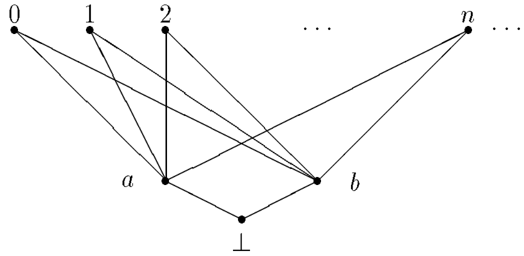
Counterexamples
1.



2.



3.



Open Problem. Is there a weaker condition than ω-algebraicity which will remove this blemish but keep all the other good points of ω-algebraicity? (One possibility is that the complete lattice of open sets (under inclusion) of the cpo is ω-algebraic.)

We will deal with function spaces by adding *completeness* axioms (lattice, etc.).

Exercise 1. Show $\overline{\mathbf{P} \overset{\text{monotonic}}{\rightarrow} \mathbf{Q}} \neq \overline{\mathbf{P}} \overset{\text{continuous}}{\rightarrow} \overline{\mathbf{Q}}$.

This means that there is no evident way to turn $B_{\mathbf{D}}$ into a functor. However from the ε-δ Theorem 1, any continuous $f\colon \mathbf{D} \rightarrow \mathbf{E}$ is given by $\{\langle a, b\rangle \in B_{\mathbf{D}} \times B_{\mathbf{E}} \mid b \sqsubseteq f(a)\}$ and we can use this to turn the countable pointed partial orders into a suitable category. Define a morphism $R\colon \mathbf{P} \rightarrow \mathbf{Q}$ of countable pointed partial orders as any $R \subset \mathbf{P} \times \mathbf{Q}$ such that:

(1) $\forall a \in \mathbf{P}.\exists b \in \mathbf{Q}.aRb$.
(2) $\forall a, a' \in \mathbf{P}.\forall b, b' \in \mathbf{Q}.a' \sqsupseteq aRb \sqsupseteq b' \Rightarrow a'Rb'$.
(3) $\forall a, a', a'' \in \mathbf{P}.\forall b', b'' \in \mathbf{Q}.(a \sqsupseteq a' \wedge a \sqsupseteq a'' \wedge a'Rb' \wedge a''Rb'' \Rightarrow \exists b \sqsupseteq b', b''.aRb)$.

Provide a definition of identity and composition to obtain a category ω-$\mathcal{PP}$ and define the evident functor $B\colon \omega\text{-}\mathcal{ALG} \rightarrow \omega\text{-}\mathcal{PP}$ where ω-$\mathcal{ALG}$ is the category of ω-algebraic cpos and continuous functions. Define, too, a functor $\overline{(\cdot)}\colon \omega\text{-}\mathcal{PP} \rightarrow \omega\text{-}\mathcal{ALG}$ and show that the two functors are a natural isomorphism of categories (see MacLane for a definition of this concept).

III: Disjoint Sum. For two given countable pointed quasiorders their disjoint sum is defined just as we did for cpos.

Fact. *For any ω-algebraic cpos $\mathbf{D}$ and $\mathbf{E}$, $\mathbf{D}+\mathbf{E}$ is ω-algebraic with $B_{\mathbf{D}+\mathbf{E}} = inl(B_{\mathbf{D}}) \cup inr(B_{\mathbf{E}}) = B_{\mathbf{D}}+B_{\mathbf{E}}$.*
Proof. Left to the reader (the last part is by definition). ∎

Corollary. *For any two countable pointed quasiorders $\mathbf{P}$ and $\mathbf{Q}$:*

$$\overline{\mathbf{P}+\mathbf{Q}} \cong \overline{\mathbf{P}} + \overline{\mathbf{Q}}.$$

IV: Smash Product. For two given countable pointed quasiorders their smash product is defined just as we did for cpos.

Fact. *For any two ω-algebraic cpos $\mathbf{D}$ and $\mathbf{E}$, $\mathbf{D}\otimes\mathbf{E}$ is ω-algebraic with $B_{\mathbf{D}\otimes\mathbf{E}} = B_{\mathbf{D}}\otimes B_{\mathbf{E}}$.*

Corollary. *For any two countable pointed quasiorders $\mathbf{P}$, $\mathbf{Q}$:*

$$\overline{\mathbf{P}\otimes\mathbf{Q}} \cong \overline{\mathbf{P}}\otimes\overline{\mathbf{Q}}.$$

V: Lifting. For one given countable pointed quasiorder $\mathbf{P}$, its lifting $(\mathbf{P}_\perp)$ is defined just as we did for cpos.

Fact. *For any one ω-algebraic cpo, $\mathbf{D}$, $\mathbf{D}_\perp$ is ω-algebraic with $B_{\mathbf{D}_\perp} = (B_{\mathbf{D}})_\perp$.*

Corollary. *For any one countable pointed quasiorder $\mathbf{P}$: $(\overline{\mathbf{P}})_\perp \cong (\overline{\mathbf{P}_\perp})$.*

Exercise 1. Show the following for two countable pointed quasiorders $\mathbf{P}$, $\mathbf{Q}$:

(1) $\overline{\mathbf{P}} \xrightarrow{\overline{inl}} \overline{\mathbf{P}+\mathbf{Q}} \xleftarrow{\overline{inr}} \overline{\mathbf{Q}}$ is a universal pair.

(2) $\overline{\mathbf{P}\times\mathbf{Q}} \xrightarrow{\overline{\otimes}} \overline{\mathbf{P}\otimes\mathbf{Q}}$ is a universal bistrict map.

(3) $\overline{\mathbf{P}} \xrightarrow{\overline{up}} \overline{\mathbf{P}_\perp}$ is a universal continuous map.

Here $\overline{inl}$, ..., $\overline{up}$ are defined by analogy with Chapter 3.

VI: Direct Limit of Embeddings.

Lemma 2. *If $f\colon\mathbf{D}\lhd\mathbf{E}$ then $f(B_{\mathbf{D}})\subseteq B_{\mathbf{E}}$.*

Proof. Take $d\in B_{\mathbf{D}}$ and let $\langle x_n\rangle$ be an increasing ω-chain in $\mathbf{E}$ such that $f(d)\sqsubseteq\bigsqcup_n x_n$, then $d = f^{\mathrm{R}}(f(d))\sqsubseteq\bigsqcup_n f^{\mathrm{R}}(x_n)$ and so for some n, $d\sqsubseteq f^{\mathrm{R}}(x_n)$. Then $f(d)\sqsubseteq f\circ f^{\mathrm{R}}(x_n)\sqsubseteq x_n$ showing $f(d)$ is finite. ∎

Exercise 1. Show $f^{\mathrm{R}}(B_{\mathbf{E}})\subseteq B_{\mathbf{D}}$ fails, in general.

Fact. *Let $\Delta\colon\mathbf{D}_0 \overset{f_0}{\lhd} \mathbf{D}_1 \overset{f_1}{\lhd} \mathbf{D}_2 \overset{f_2}{\lhd} \cdots$ be a direct chain in $\mathcal{CPO}^{\mathrm{E}}$ where the $\mathbf{D}_i$ are all ω-algebraic, with $B_{\mathbf{D}} = \bigcup_n \rho_n(B_{\mathbf{D}_n})$.*

Proof. By the lemma $\rho_n(B_{\mathbf{D}_n})\subseteq B_{\mathbf{D}}$. Take x in $\mathbf{D}$. For each $\rho_n^{\mathrm{R}}(x)$ let $\langle d_m^{(n)}\rangle_{m\in\omega}$ be an increasing chain of elements of $B_{\mathbf{D}}$ s.t. $\rho_n^{\mathrm{R}}(x) = \bigsqcup_m d_m^{(n)}$. Then $B' = \{\rho_n(d_m^{(n)}) \mid n,m\in\omega\}$ is a countable directed set of elements of $\bigcup_n\rho_n(B_{\mathbf{D}_n})$ with lub x as:

$$\begin{aligned}\bigsqcup B' &= \bigsqcup_{n\in\omega}\bigsqcup_{m\in\omega}\rho_n(d_m^{(n)})\\ &= \bigsqcup_{n\in\omega}\rho_n(\bigsqcup_{m\in\omega} d_m^{(n)})\\ &= \bigsqcup_{n\in\omega}\rho_n(\rho_n^{\mathrm{R}}(x))\\ &= x\end{aligned}$$

(the arguments for existence of the lub go “from x to $\bigsqcup B'$”).

Finally as B' is countable and directed it has a cofinal increasing ω-chain. Using Lemma 2 the proof concludes. ∎

Corollary. *If $F\colon\mathcal{CPO}^{\mathrm{E}}\to\mathcal{CPO}^{\mathrm{E}}$ is continuous and preserves ω-algebraicity then $\mathbf{Fix}_F$ is ω-algebraic (and similarly for multiple fixed points).*

Proof. $\mathbf{U}$ is ω-algebraic so the $F^n(\mathbf{U})$ are too as F preserves ω-algebraicity. Now apply VI to the definition of $\mathbf{Fix}_F$. ∎

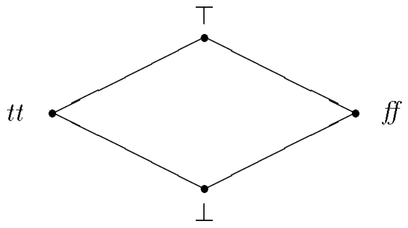
Thus all the domain equations we solve have ω-algebraic solutions (except, perhaps, for those involving exponentiation).

Note. If the functor F is defined using $+$, $\times$, $\otimes$, $(\cdot)_\perp$, projections and constant functors $K_{\mathbf{D}}$ where every point in $\mathbf{D}$ is finite then F preserves the property of being finite and we see that $B_{\mathbf{Fix}_F} = \bigcup_n\rho_n(F^n(\mathbf{U}))$. So e.g. the domain of trees is the limit of the domains of *finite* trees.

Exercise 2. Suppose $\mathbf{D}$ is ω-algebraic and $f\colon\mathbf{D}\to\mathbf{D}$ is a closure. Show that the cpo of fixed-points of $\mathbf{D}$ is also ω-algebraic and that: $B_{\mathbf{Fix}_f} = f(B_{\mathbf{D}})$. Find a condition on projections, f, on $\mathbf{D}$ so that $\mathbf{Fix}_f$ is ω-algebraic if $\mathbf{D}$ is. Find a necessary and sufficient condition for $\mathbf{Fix}_f$ to be ω-algebraic when f is a retraction.

Completeness

The most obvious axiom is: *every* subset has a lub. This gives us the complete lattices, $\mathbf{D}$, which have the top element $\top_{\mathbf{D}} \stackrel{\text{def}}{=} \bigsqcup \mathbf{D}$. This can be interpreted as "inconsistent" or "overloaded" and causes problems. For example the truth-values, $\mathbf{T}$, not being a lattice, would have to be made into one by adding a top element giving:



It is now unclear how the conditional function $(\cdot \rightarrow \cdot \mid \cdot)\colon \mathbf{T} \times \mathbf{D} \times \mathbf{D} \rightarrow \mathbf{D}$ is to be defined at the top element:

(1) $$(\top \rightarrow d \mid e) = \top,$$

(2) $$(\top \rightarrow d \mid e) = d \sqcup e.$$

Both result in the failure of natural identities which hold for the three-valued cpo of truth-values:

(1) $$(p \rightarrow (p \rightarrow a \mid b) \mid c) = (p \rightarrow a \mid c),$$

(2) $$(p \rightarrow (q \rightarrow a \mid b) \mid (q \rightarrow c \mid d)) = (q \rightarrow (p \rightarrow a \mid c) \mid (p \rightarrow b \mid d)),$$

respectively.

Instead we adopt a compromise axiom.

Definition. Let $\mathbf{D}$ be a cpo. A subset, X, of $\mathbf{D}$ is *consistent* iff it has an upper bound in $\mathbf{D}$, and then we write $\uparrow S$ (and otherwise $\# S$). (A subset X is *ω-consistent* if every finite subset of X has an upper bound in $\mathbf{D}$.) Then $\mathbf{D}$ is *consistently complete* iff every consistent subset has a lub. We want to consider the consistently complete ω-algebraic cpos and there are several equivalent axiomatisations.

Exercises

1. Consider the four statements about a cpo $\mathbf{D}$:

(1) Every ω-consistent subset has a lub.
(2) (*Consistent completeness*) Every consistent subset has a lub.
(3) Every consistent pair of elements has a lub.
(4) Every nonempty subset has a glb.

Show (2) $\Leftrightarrow$ (4), (1) $\Leftrightarrow$ (2) + $\mathbf{D}$ is a dcpo $\Leftrightarrow$ (3) + $\mathbf{D}$ is a dcpo. Conclude that if $\mathbf{D}$ is an ω-algebraic cpo then (1) $\Leftrightarrow$ (2) $\Leftrightarrow$ (3) $\Leftrightarrow$ (4).

2 Complete Lattices. Consider the following three statements about a cpo $\mathbf{D}$:

(1) Every subset has a lub.
(2) Every subset has a glb.
(3) Every pair of elements has a lub.

Show (1) $\Leftrightarrow$ (2) $\Leftrightarrow$ (3) + $\mathbf{D}$ is a dcpo. Conclude that if $\mathbf{D}$ is an ω-algebraic cpo then (1) $\Leftrightarrow$ (2) $\Leftrightarrow$ (3).

The evidence for consistent completeness is rather empirical: all our basic examples are consistently complete and all the constructions $+$, $\times$, $\otimes$, $(\cdot)_\perp$, $\rightarrow$, $\rightarrow_\perp$ preserves it.

Notice that the basis of a consistently complete ω-algebraic cpo is closed under lubs of consistent pairs. Conversely: if a countable pointed quasiorder, $\mathbf{P}$ has lubs of consistent pairs of elements, then $\overline{\mathbf{P}}$ is consistently complete.

Exercise 3. Prove this.

There is also an idea of a *subbasis*.

Definition. Let $\mathbf{D}$ be a cpo. A set $X \subseteq \mathbf{D}$ is a *subbasis* iff for any element x of $\mathbf{D}$, $x = \bigsqcup\{u \in X \mid u \sqsubseteq x\}$.

Lemma 3. *If a consistently complete cpo $\mathbf{D}$ has a denumerable subbasis of finite elements it is ω-algebraic.*
Proof. Apply Lemma 1 to the set, B, of lubs of consistent finite subsets of the subbasis. ∎

Function Spaces

First consistent-completeness is preserved. Let $\mathbf{E}$ be consistently complete and suppose $F \subseteq (\mathbf{D} \to \mathbf{E})$ has a lub, g. Then:

$$(\bigsqcup F)(x) = \bigsqcup_{f \in F} f(x).$$

For, the lub on the RHS exists as $F(x)$ has $g(x)$ as an upper bound. For continuity: $(\bigsqcup F)(\bigsqcup_n x_n) = \bigsqcup_{f\in F} f(\bigsqcup_n x_n) = \bigsqcup_{f \in F} \bigsqcup_n f(x_n)$ and also, $\bigsqcup_n (\bigsqcup F)(x_n) = \bigsqcup_n \bigsqcup_{f\in F} f(x_n)$ and we can interchange the order of the lubs.

Now if $\mathbf{D}$, $\mathbf{E}$ are ω-algebraic and consistently complete so is $\mathbf{D} \to \mathbf{E}$. We have seen it is consistently complete. We have seen that $\{d \Rightarrow e \mid d \in B_{\mathbf{D}} \wedge e \in B_{\mathbf{E}}\}$ is a subbasis of finite elements (Example 3, p. 59, Exercise 3, p. 61). So by Lemma 3, it is ω-algebraic too: its basis is the lubs of consistent finite subsets of elements of the form $d \Rightarrow e$.

Exercises

1. Let $d_1, \ldots, d_n$ be finite elements of $\mathbf{D}$ and $e_1, \ldots, e_n$ be finite elements of $\mathbf{E}$. Show

$$\uparrow\{d_i \Rightarrow e_i \mid 1 \le i \le n\} \quad \text{iff} \quad \forall I \subseteq \{1, \ldots, n\}.(\uparrow\{d_i \mid i \in I\} \Rightarrow \uparrow\{e_i \mid i \in I\}).$$

2. Show $(d \Rightarrow e) \sqsubseteq \bigsqcup\{d_i \Rightarrow e_i \mid i = 1, \ldots, n\}$ iff

$$\exists I \subseteq \{1, \ldots, n\}.\uparrow\{d_i \mid i \in I\} \wedge d \sqsupseteq \bigsqcup\{d_i \mid i \in I\} \wedge e \sqsubseteq \bigsqcup\{e_i \mid i \in I\}.$$

3 Product. If $X \subseteq \mathbf{D} \times \mathbf{E}$ is consistent where $\mathbf{D}$, $\mathbf{E}$ are consistently complete then

$$\bigsqcup_{\mathbf{D}\times\mathbf{E}} X = \langle \bigsqcup\nolimits_{\mathbf{D}}{}_{x\in X} x_0, \bigsqcup\nolimits_{\mathbf{E}}{}_{x \in X} x_1\rangle.$$

So product preserves consistent completeness.

Show $\uparrow X$ iff $\uparrow\pi_0(X) \wedge \uparrow\pi_1(X)$.

4 Sum. Show sum preserves consistent completeness. If $X \subseteq \mathbf{D} + \mathbf{E}$ where $\mathbf{D}$, $\mathbf{E}$ are consistently complete then $\uparrow X$ iff $(X \subseteq \mathit{inl}(\mathbf{D}) \wedge \uparrow\mathit{outl}(X)) \vee (X \subseteq \mathit{inr}(\mathbf{E}) \wedge \uparrow\mathit{outr}(X))$, and then $\bigsqcup X = \mathit{inl}(\bigsqcup_{\mathbf{D}} \mathit{outl}(X))$ if $X \subseteq \mathit{inl}(\mathbf{D})$ and $\bigsqcup X = \mathit{inr}(\bigsqcup_{\mathbf{E}} \mathit{outr}(X))$ if $X \subseteq \mathit{inr}(\mathbf{E})$.

5 Smash Product. Show smash product preserves consistent completeness: If $X \subseteq \mathbf{D} \otimes \mathbf{E}$ where $\mathbf{D}$ and $\mathbf{E}$ are consistently complete then $\uparrow X$ iff $\uparrow\pi_0(X) \wedge \uparrow\pi_1(X)$ and if $\uparrow X$ then $\bigsqcup X = (\bigsqcup_{\mathbf{D}} \pi_0(X)) \otimes (\bigsqcup_{\mathbf{E}} \pi_1(X))$.

6 Lifting. Show lifting preserves consistent completeness: If $X \subseteq \mathbf{D}$ where $\mathbf{D}$ is consistently complete then $\uparrow X$ iff $\uparrow\mathit{down}(X)$ and if $\uparrow X$ then $\bigsqcup_{\mathbf{D}_\perp} X = \perp$ if $X \subseteq \{\perp\}$ and $\bigsqcup_{\mathbf{D}_\perp} X = \mathit{up}(\bigsqcup_{\mathbf{D}} \mathit{down}(X))$ otherwise.

7 Strict Function Space. Show strict function space preserve consistent completeness and consistent completeness + ω-algebraicity. Treat this construction as we just did the function space. As subbasis we can take $\{d \Rightarrow e \mid d \in B_{\mathbf{D}} \wedge e \in B_{\mathbf{E}} \wedge d \neq \perp\}$ or, more precisely, $\{d \Rightarrow_\perp e \mid d \in B_{\mathbf{D}} \wedge e \in B_{\mathbf{E}}\}$ where:

$$(d \Rightarrow_\perp e)(x) = \begin{cases} e & (x \sqsupseteq d \wedge x \neq \perp), \\ \perp & (\text{otherwise}). \end{cases}$$

8 Direct Limits. Show that consistent completeness is preserved by direct limits of ω-chains in $\mathcal{CPO}^{\mathrm{E}}$. In particular if $\Delta = \langle \mathbf{D}_n, f_n\rangle$ is such a chain and $\rho \colon \Delta \to \mathbf{D}$ is a universal cone then for $X \subseteq \mathbf{D}$, $\uparrow X$ iff $\forall n. \uparrow_{\mathbf{D}_n} \rho_n^{\mathrm{R}}(X)$ and $\bigsqcup_{\mathbf{D}} X = \bigsqcup_n \rho_n(\bigsqcup_{\mathbf{D}_n} \rho_n^{\mathrm{R}}(X))$ where $\langle \rho_n(\bigsqcup_{\mathbf{D}_n} \rho_n^{\mathrm{R}}(X))\rangle_{n\in\omega}$ is an increasing ω-chain.

If $X \subseteq \bigcup_{n \le m} \rho_n(\mathbf{D}_n)$ we have a more constructive version:

$$\uparrow X \quad \text{iff} \quad \uparrow_{\mathbf{D}_m} \rho_m^{\mathrm{R}}(X) \text{ and } \bigsqcup_{\mathbf{D}} X = \rho_m(\bigsqcup_{\mathbf{D}_m} \rho_m^{\mathrm{R}}(X)).$$

9. Characterise $\bigsqcap$ (glb) of nonempty sets in all the above constructions.

10. Show that if $r \colon \mathbf{D} \to \mathbf{D}$ is a retraction where $\mathbf{D}$ is consistently complete then $\mathbf{Fix}_r$ is also consistently complete.

We easily conclude that if $F: \mathcal{CPO}^{\mathrm{E}} \to \mathcal{CPO}^{\mathrm{E}}$ is continuous and preserves the combination, consistently complete + ω-algebraic then $\mathbf{Fix}_F$ is also consistently complete and ω-algebraic and similarly for multiple fixed-points. So all the domain equations we solve, as above, give consistently-complete ω-algebraic cpos.

Exercises

1 Stable Functions. Let $\mathbf{D}$, $\mathbf{E}$ be consistently complete cpos and let $f: \mathbf{D} \to \mathbf{E}$ be a continuous function. Show the following two conditions are equivalent:

(1) **Stability.** $\forall x \in \mathbf{D}. \forall y \in \mathbf{E}. y \sqsubseteq f(x) \Rightarrow [\exists t \sqsubseteq x. y \sqsubseteq f(t) \wedge \forall u \sqsubseteq x. y \sqsubseteq f(u) \Rightarrow u \sqsupseteq t]$. Here the t whose existence is postulated is clearly unique and is written as $m(f, x, y)$ and has the properties that, if $y \sqsubseteq f(x)$:

(a) $y \sqsubseteq f(m(f, x, y))$,
(b) $y \sqsubseteq f(u) \wedge u \sqsubseteq x \Rightarrow u \sqsupseteq m(f, x, y)$.

Thus $m(f, x, y)$ is the part of x *required* for y.

(2) $\forall X \subseteq \mathbf{D}. (\uparrow X \wedge X \neq \emptyset) \Rightarrow f(\bigsqcap X) = \bigsqcap f(X)$.

Suppose further that $\mathbf{D}$, $\mathbf{E}$ are ω-algebraic and that $\mathbf{D}$ obeys:

Axiom I: $\quad \forall d \in B_{\mathbf{D}}. \{x \in \mathbf{D} \mid x \sqsubseteq d\}$ is finite.

Then show stability is also equivalent to:

(3) $\forall x, y \in \mathbf{D}. x \uparrow y \Rightarrow f(x \sqcap y) = f(x) \sqcap f(y)$.

Find examples of stable and nonstable functions. The *stable order* is defined by, for stable $f, g: \mathbf{D} \to \mathbf{E}$ ($\mathbf{D}$, $\mathbf{E}$ consistently complete):

$$f \leq g \equiv (f \sqsubseteq g) \wedge \forall x \in \mathbf{D}. \forall y \in \mathbf{E}. [y \sqsubseteq f(x) \Rightarrow m(f, x, y) = m(g, x, y)].$$

Show it can also be defined by: $f \leq g \equiv (f \sqsubseteq g) \wedge [\forall x, y \in \mathbf{D}. x \uparrow y \Rightarrow f(x) \sqcap g(y) = f(y) \sqcap g(x)]$. Show that the set of stable functions, ordered by $\leq$, is a cpo, given that $\sqcap$ is continuous.

2 The Category $\mathcal{SFP}$. There are weaker completeness conditions than consistent completeness which work. A *strongly algebraic* cpo $\mathbf{D}$ ($\mathcal{SFP}$ object) is one such that $\mathbf{D} \cong \varinjlim_{\mathcal{CPO}^{\mathrm{E}}} \langle \mathbf{D}_n, f_n \rangle$ where each $\mathbf{D}_n$ is finite. Show that the strongly algebraic cpos are closed under $+$, $\times$, $\otimes$, $\vec{\to}$, $\to_\perp$ and $\varinjlim_{\mathcal{CPO}^{\mathrm{E}}}$.

Conjecture. *If* $\mathbf{D}$ *and* $(\mathbf{D} \to \mathbf{D})$ *are* ω*-algebraic then* $\mathbf{D}$ *is strongly algebraic.*

If true this would imply that $\mathcal{SFP}$ is the largest subcategory of the ω-algebraic cpos closed under exponentiation. Let $\mathbf{D}$ be a cpo. A *minimal upper bound*, u of a subset X of $\mathbf{D}$ is an upper bound which dominates no other upper bound; i.e. $(\forall x \in X. v \sqsupseteq x) \wedge (u \sqsupseteq v) \Rightarrow u = v$. The set $U(X)$ is the set of minimal upper bounds of X; $U^*(X)$ is the smallest set, Y, such that $X \subseteq Y$ and if $X \subseteq Y$ then $U(X) \subseteq Y$; i.e. $U^*(X)$ is the least set closed under U^* and containing X. Show that the strongly algebraic cpos are just the ω-algebraic cpos $\mathbf{D}$ such that for every finite subset X of $B_{\mathbf{D}}$:

(1) If v is an upper bound of X then for some $u \in U(X)$, $v \sqsupseteq u$.
(2) The set $U^*(X)$ is finite.

3 Other Completeness Conditions. There are completeness conditions stronger than consistent completeness but weaker than being a complete lattice. Let $\mathbf{D}$ be a cpo. For $\kappa \leq \omega$, we say a subset X of $\mathbf{D}$ is κ*-consistent* iff for every $Y \subseteq \mathbf{D}$ s.t. $\|Y\| < \kappa$, Y has an upper bound in $\mathbf{D}$. We say $\mathbf{D}$ is κ*-consistently complete* iff every κ-consistent subset has a lub. Show $\mathbf{D}$ κ-consistently complete implies $\mathbf{D}$ λ-consistently complete for $\kappa \leq \lambda \leq \omega$. Show that if $2 \leq \kappa < \lambda$ then there are κ-consistently complete cpos which are not λ-consistently complete (and 0- = 1- = 2- = complete lattice). Show that the κ-consistently complete ω-algebraic cpos are closed under $+$, $\times$, $\otimes$, $\to$, $\to_\perp$, $\varinjlim_{\mathcal{CPO}^{\mathrm{E}}}$.

7. Computability

We give a theory of computability for consistently complete ω-algebraic cpos — which from now on we will just call *domains*. The idea is to postulate a suitable recursive structure on the finite elements and define the computable elements to be the limits of effective chains of finite elements. We will use some standard ideas and notation from classical recursive function theory. Thus we assume n-tupling functions $\langle m_0, \ldots, m_{n-1}\rangle$ and corresponding projections $\pi_i = (\cdot)_i$ $(i < n)$ and $\varphi_m = \{m\}$ is the mth pr. function in the standard enumeration, W_m is the mth re set, F_m is the mth finite set (and $\{\!|m_0, \ldots, m_{n-1}|\!\}$ is the code of $\{m_0, \ldots, m_{n-1}\}$).

Definition 1. Let $\mathbf{D}$ be a domain. An *enumeration* $\langle \mathbf{D}, \varepsilon\rangle$ of its basis is any surjection, $\varepsilon\colon \mathbf{N} \to B_{\mathbf{D}}$ (and we write ε_m for $\varepsilon(m)$).

Further it is *effectively given*, w.r.t. ε, (by an *index* $\langle c, b, u\rangle$) if

(1) The relation $\varepsilon_m \uparrow \varepsilon_n$ on m and n is recursive (and has index c).
(2) $\varepsilon_b = \bot$.
(3) The relation $\varepsilon_n = \varepsilon_m \sqcup \varepsilon_{m'}$ on m, m', n is recursive (and has index u).

Examples

1. *Truthvalues.* $\varepsilon_0, \varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots$ is $\bot$, *tt*, *ff*, *ff*, *ff*, $\ldots$ with $\varepsilon_m \sqsubseteq \varepsilon_n \equiv m = 0 \vee n = m \vee (m \geq 2 \wedge n \geq 2)$, $\varepsilon_m \uparrow \varepsilon_n \equiv (\varepsilon_m \sqsubseteq \varepsilon_n \vee \varepsilon_n \sqsubseteq \varepsilon_m)$, $\varepsilon_0 = \bot$, $\varepsilon_n = \varepsilon_m \sqcup \varepsilon_{m'} \equiv (\varepsilon_n = \varepsilon_{\max(m,m')}) \wedge \varepsilon_m \uparrow \varepsilon_{m'}$.
2. *Integers.* $\varepsilon_0, \varepsilon_1, \varepsilon_2, \ldots$ is $\bot, 0, 1, \ldots$.

We always use these "natural" enumerations except where otherwise stated (and similarly in other cases).

Note that we can always insist that $\varepsilon_0 = \bot$ for starting with ε we change to ε' where $\varepsilon'_0 = \bot$, $\varepsilon'_{n+1} = \varepsilon_n$ and perform the evident (recursive) change in the index. Note too that $\varepsilon_m \sqsubseteq \varepsilon_n$ is recursive in m and n as it is equivalent to $\varepsilon_n = \varepsilon_m \sqcup \varepsilon_n$.

Exercises

1. Show that any finite domain can be effectively given as can **Tapes**, $\mathbf{V}_\omega$, $\mathbf{P}\boldsymbol{\omega}$, $\mathbf{T}^\omega$, **Trees**, $\ldots$.

2. Let $A = \langle \omega, +\rangle$ be a recursive algebras (i.e. $+\colon \omega^2 \to \omega$ is recursive). Show that the lattice of subalgebras is *effectively given*. What about partial algebras and pr. functions?

3. Show that the partial order of consistent theories of propositional logic (over given letters P_0, P_1, $\ldots$) is an effectively given domain. What happens with theories in the predicate calculus? What happens with equational theories? (They are sets of equations $t = u$ closed under deduction; they are consistent if not every equation can be proved.)

There is a useful alternative definition in terms of subbases.

Lemma 1. *Let* $\mathbf{D}$ *be a domain. Then it is effectively given iff there is an enumeration* β_0, β_1, $\ldots$ *of a subbasis of* $\mathbf{D}$ *such that*

(1) $\uparrow\{\beta_{m_0}, \ldots \beta_{m_{n-1}}\}$ *is a recursive predicate of* $\{\!|m_0, \ldots, m_{n-1}|\!\}$,
(2) The relation $\beta_m \sqsubseteq \bigsqcup\{\beta_{m_0}, \ldots, \beta_{m_{n-1}}\}$ *is recursive in* m *and* $\{\!|m_0, \ldots, m_{n-1}|\!\}$.

Proof. $\Rightarrow$) Clear, taking $\beta = \varepsilon$.
$\Leftarrow$) Define

$$\varepsilon_{\{\!|m_0,\ldots,m_{n-1}|\!\}} = \begin{cases} \bigsqcup_{i<n} \beta_{m_i} & (\uparrow\{\beta_{m_0}, \ldots, \beta_{m_{n-1}}\}), \\ \bot & \text{(otherwise)}. \end{cases}$$

The rest of the proof is left to the reader. ∎

With the evident definition of an *index* of such a subbasis one notes from the above proof that there are recursive functions translating each type of index to the other (note that given the subbasis one finds a code for $\bot$ by searching for an m s.t. $\beta_m \sqsubseteq \bigsqcup\emptyset$).

Exercises

1. Show that one can replace condition (3) of the definition of an effectively given domain (making appropriate changes in the definition of index) by:

(3′) The relation $\varepsilon_m = \varepsilon_n$ is recursive.
(4′) There is a recursive function, f, such that $\varepsilon_{f(m,n)} = \varepsilon_m \sqcup \varepsilon_n$ wherever $\varepsilon_m \uparrow \varepsilon_n$.

2. Show that one can develop a theory of *weakly effectively given domains* with the conditions (1), (4′), and:

(3″) (a) The relation $\varepsilon_m = \varepsilon_n$ is re.
(b) The predicate $\varepsilon_m = \perp$ is recursive.

3. What can you get in the way of a theory of effectively given ω-algebraic cpos? Try as axioms

(3‴) The relation $\varepsilon_n \sqsubseteq \varepsilon_n$ is recursive and there could also be a weaker version à la (3″) (a), (b) with $=$ replaced by $\sqsubseteq$.

4. Show there are uncountably many domains which are not effectively given (and try to give at least one example!) Find domains which are weakly effectively given but not effectively given as ω-algebraic cpos.

5 $\mathcal{SFP}$**.** Develop a theory of effectively given $\mathcal{SFP}$ objects with the definition that $\mathbf{D}$ is eff. given w.r.t. ε with index $\langle a, b\rangle$:

(1) The relation $\varepsilon_m = \varepsilon_n$ is recursive in m and n (and has index a).
(2) There is a recursive function u such that for any m:

$$u(\varepsilon(F_m)) = \varepsilon(F_{u(m)})$$

(and u has index b).

Include a characterisation of eff. given $\mathcal{SFP}$ objects in terms of effective colimits of finite objects in $\mathcal{SFP}^{\mathrm{E}}$ (see the later development of effective colimits in $\mathcal{CPO}^{\mathrm{E}}$ for some hints).

6. Let $\langle \mathbf{P}, \sqsubseteq, \perp\rangle$ be a countable pointed quasi-order with lubs of pairs of consistent elements. It is *effectively given* iff there is an enumeration (= surjection) $\varepsilon\colon \mathbf{N} \to \mathbf{P}$ and an index satisfying (1), (2), (3) of Definition 1. Show $\mathbf{P}$ effectively given iff $\overline{\mathbf{P}}$ is (and so $\mathbf{D}$ is effectively given iff $B_{\mathbf{D}}$ is).

The following definitions will prove useful on occasion.

Definition 1. Consider an enumeration $\langle \mathbf{D}, \varepsilon\rangle$. The relations $\lesssim$, $\simeq$ on $\mathbf{N}$ are defined by:

$$m \lesssim n \equiv \varepsilon(m) \sqsubseteq \varepsilon(n),$$
$$m \simeq n \equiv \varepsilon(m) = \varepsilon(n).$$

A *morphism* of enumerations is a pair $\langle r, f\rangle\colon \langle \mathbf{D}, \varepsilon\rangle \to \langle \mathbf{D}', \varepsilon'\rangle$ where $r\colon \mathbf{N} \to \mathbf{N}$ is recursive, $f\colon \mathbf{D} \to \mathbf{D}'$ is continuous and the following diagram commutes:

$$\begin{tikzcd} \mathbf{N} \arrow[r, "r"] \arrow[d, "\varepsilon"'] & \mathbf{N} \arrow[d, "\varepsilon'"] \\ \mathbf{D} \arrow[r, "f"'] & \mathbf{D}' \end{tikzcd}$$

Exercises

1. Show the enumerations and their morphisms form a category. If $\langle r, f\rangle\colon \langle \mathbf{D}, \varepsilon\rangle \to \langle \mathbf{D}, \varepsilon'\rangle$ is a morphism then r clearly determines f. Show that any recursive $r\colon \mathbf{N} \to \mathbf{N}$ can be extended to a morphism iff it is monotonic w.r.t. $\lesssim$; show too that it is not enough that r preserves $\simeq$.

2. Show that if $\mathbf{D}$ is finite then there is always a morphism from $\langle \mathbf{D}, \varepsilon\rangle$ to $\langle \mathbf{D}', \varepsilon'\rangle$ (thus *reducing* ε to ε'); show this need not hold in general — e.g. when $\mathbf{D} = \mathbf{N}$.

3. Show that for any enumeration $\langle \mathbf{D}, \varepsilon\rangle$ we have $B_{\mathbf{D}} \cong \langle \mathbf{N}, \lesssim\rangle/\simeq$ (and deduce there are only countably many eff. given domains, to within isomorphism).

Computability

We use the idea that a computable element is just a limit of an effective chain of finite elements, imagining a machine generating more and more information about the element, but at any finite point in its computation, only having generated a finite amount. This is sometimes called the *enumeration* model of computation following the case where, for example, one enumerates all the elements of a set: at finite time one has generated a finite subset.

Definition. Let $\mathbf{D}$ be effectively given w.r.t. ε. An *effective chain* in $\mathbf{D}$ with index g is any increasing chain $\langle \varepsilon_{\{g\}_n} \rangle_{n \in \omega}$. An element d is computable if there is an effective chain with lub d. We write $C_{\langle \mathbf{D}, \varepsilon \rangle}$ for the set of computable elements of $\mathbf{D}$ (and $C_{\mathbf{D}}$ when ε is understood).

Exercises

1. Every finite element is computable.

2. The computable elements of **Tapes** are just the finite ones and the recursive infinite sequence (of 0's and 1's); the computable elements of $\mathbf{P}\boldsymbol{\omega}$ are the re. sets, of $\mathbf{T}^{\omega}$ are the pr. predicates. What about $\mathbf{V}_{\omega}$, **Trees**, ...

3. Show how to use the typed λ-calculus to define all computable elements of **Tapes**, $\mathbf{P}\boldsymbol{\omega}$, $\mathbf{T}^{\omega}$, ...
[*Hint* Using the idea of generation just define appropriate functions $f \colon \mathbf{N} \to \mathbf{Tapes}$ so that for a given tape, $t = t_0 t_1 t_2 \ldots$, we have:

$$f(n) = t_n t_{n+1} \ldots .$$

and use similar ideas in the other cases.]

4. Show that by varying ε one can vary $C_{\langle \mathbf{P}\boldsymbol{\omega}, \varepsilon \rangle}$. Kanda and Park have shown (When are two effectively given domains identical? Proc. 4th AI Conference.) that for some $\mathbf{D}$, ε, ε' we even have: $C_{\langle \mathbf{D}, \varepsilon \rangle} \not\cong C_{\langle \mathbf{D}, \varepsilon' \rangle}$. Can we take $\mathbf{D} = \mathbf{P}\boldsymbol{\omega}$?

Now there is a little technical difficulty if we want to *enumerate* all the computable elements as not all integers, g, are indexes as $\{g\}$ may not be recursive and $\langle \varepsilon_{\{g\}(n)} \rangle$ may not be increasing. To avoid dealing with partial surjections (*Exercise* try working with them) we note that for any g there is an effectively obtainable g' with $\mathit{range}(\{g'\}) = \mathit{range}(\{g\}) \cup \{\text{a code of } \bot\}$. Now define $f \colon \mathbf{N} \to \mathbf{N}$ by:

$$f(0) = \text{a code of } \bot,$$
$$f(n+1) = \begin{cases} \text{a code of } \varepsilon_{f(n)} \sqcup \varepsilon_{g'(n+1)} & (\text{if } \varepsilon_{f(n)} \uparrow \varepsilon_{g'(n+1)}), \\ f(n) & (\text{otherwise}). \end{cases}$$

This f is clearly recursive and an index for it is effectively obtainable (say via d) from m (via n); clearly for $d(m)$ is an index of a computable element and:

$$\bigsqcup_n \varepsilon_{\{d(g)\}(n)} = \bigsqcup_n \varepsilon_{\{g\}(n)}$$

if g is an index. We can now define our (total) enumeration:

Definition 1. Let $\langle \mathbf{D}, \varepsilon \rangle$ be an enumerated domain. Then the enumeration $\zeta \colon \mathbf{N} \to C_{\mathbf{D}}$ is defined by:

$$\zeta_g = \bigsqcup_n \varepsilon_{\{d(g)\}(n)}.$$

Exercises

1. Let $\mathbf{D}$ be effectively given w.r.t. ε. Show that the following conditions on an element d are all effectively equivalent.

(a) d is computable.
(b) $\{n \mid \varepsilon_n \sqsubseteq d\}$ is re.
(c) For some k, $\bigsqcup \varepsilon(W_k) = d$.
(d) For some k, $\varepsilon(W_k)$ is directed and has lub d.

2. Use 1(c) to obtain another total enumeration ζ^{S} of $C_{\mathbf{D}}$; this is the *single-valued* enumeration of Egli-Constable; ours, ζ, is the *directed* enumeration of Kanda-Park.

3. Show that as every finite element is computable there is a recursive function k such that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{N} \arrow[r, "k"] \arrow[d, "\varepsilon"'] & \mathbf{N} \arrow[d, "\zeta"] \\
B_{\mathbf{D}} \arrow[r, "\iota"'] & C_{\mathbf{D}}
\end{tikzcd}$$

This will prove useful later.

4. (1) Show that the relation $\varepsilon_m \sqsubseteq \zeta_n$, is re. in m and n. Is it ever recursive (ignoring here — and below — the case $\mathbf{D} = \bot$)?

(2) Show that the relation $\zeta_m \# \zeta_n$ is re. in m and n. Is it ever recursive?

(3) Show that there is a recursive function, $\sqcup$,

$$\zeta_{m \sqcup n} = \zeta_m \sqcup \zeta_n \qquad (\text{when } \zeta_m \uparrow \zeta_n).$$

(4) Show there is a recursive function, $\sqcap$, of two arguments such that:

$$\zeta_{m \sqcap n} = \zeta_m \sqcap \zeta_n.$$

(5) Are either of the relations $\zeta_m \sqsubseteq \zeta_n$, $\zeta_m = \zeta_n$ (or their negations) ever re?

5 Effective Limits. Show the lubs of increasing effective chains are computable and effectively obtainable. That is there is a recursive function, $\bigsqcup$, of one argument such that if c is the index of an *effective chain of computable elements* (i.e. if $\{c\}$ is recursive and $\langle \zeta_{\{c\}(n)} \rangle$ is increasing) then:

$$\zeta_{\bigsqcup c} = \bigsqcup_n \zeta_{\{c\}(n)}.$$

6 Computable Open Sets. An open set O of an effectively given $\mathbf{D}$, is *computable* iff given any effective chain of computable elements with index c with $\zeta_{\bigsqcup c}$ in O one can effectively obtain from c an n s.t. $\zeta_{\{c\}(n)} \in O$. Show that an open set O is computable iff

$$O = \bigcup_{n \in W_m} O_{\varepsilon_n}$$

for some m. Show the computable open sets are closed under intersection and unions of re. collections of open sets. Note all proofs and closure properties are effective.

Computable Functions

To obtain a definition of computable function we effectivise the ε-δ version of continuity: given a finite amount of information about the input one can enumerate all the finite pieces of information about the output.

Definition. Let $\mathbf{D}$ and $\mathbf{E}$ be domains effectively given with respect to enumerations ε, ε'. Then a continuous function $f: \mathbf{D} \to \mathbf{E}$ is *computable* iff

$$Graph(f) = \{\langle m, n \rangle \mid \varepsilon'_n \sqsubseteq f(\varepsilon_m)\}$$

is re. (and g is an index for f if $Graph(f) = W_g$). We delay consideration of enumerating the computable functions until the discussion of function spaces.

Examples

1. The computable $f: \mathbf{N} \to \mathbf{N}$ are the constant functions and the (functions corresponding to) the partial recursive functions.

2. The identity is computable as:

$$Graph(id) = \{\langle m, n \rangle \mid \varepsilon_n \sqsubseteq \varepsilon_m\}.$$

3. If $\mathbf{D} \xrightarrow{f} \mathbf{E} \xrightarrow{g} \mathbf{F}$ are computable w.r.t. ε, ε' and ε', ε'' then $g \circ f$ is too w.r.t. ε, ε'' as:

$$Graph(g \circ f) = Graph(f) \circ Graph(g)$$

(and an idea for $g \circ f$ is clearly effectively obtainable from those of g and f). Thus we have a *category* of enumerated effectively given domains and computable functions.

Exercises

1. What are the computable $f: \mathbf{N} \to \mathbf{T}$? What about $f: \mathbf{P\omega} \to \mathbf{P\omega}$? [*Hint* Read about enumeration operators in Rogers.] What about $f: \mathbf{T}^\omega \to \mathbf{T}^\omega$, $f: \mathbf{Tapes} \to \mathbf{Tapes}$? Are they all definable in the typed λ-calculus? [*Hint* Try using *parcond* if needed.]

2. Show that a continuous $f: \mathbf{D} \rightarrow \mathbf{E}$ can be extended to a morphism of enumerations $\langle \mathbf{D}, \varepsilon \rangle$, $\langle \mathbf{E}, \varepsilon' \rangle$ of effectively given domains iff f preserves finiteness and $Graph(f)$ is recursive.

Another natural idea for a definition of the computability of a function f is that given a method of generating an argument x one can effectively obtain a method of generating the result $f(x)$. But it turns out that this is a theorem:

Theorem 1. *Let $\langle \mathbf{D}, \varepsilon \rangle$, $\langle \mathbf{E}, \varepsilon' \rangle$ be enumerations of effectively given domain. Then a continuous function $f: \mathbf{D} \rightarrow \mathbf{E}$ is computable w.r.t. ε, ε' iff there is a recursive $r: \mathbf{N} \rightarrow \mathbf{N}$ such that the following diagram commutes:*

$$
\begin{tikzcd}
\mathbf{N} \arrow[r, "r"] \arrow[d, "\zeta"'] & \mathbf{N} \arrow[d, "\zeta'"] \\
\mathbf{D} \arrow[r, "f"'] & \mathbf{E}
\end{tikzcd}
$$

Proof. $\Rightarrow$) If f is computable then for any computable x with $\zeta(g) = x$:

$$
\begin{aligned}
X &= \{n \mid \varepsilon'(n) \sqsubseteq f(x)\} \\
&= \{n \mid \exists \langle m, n \rangle \in Graph(f).\varepsilon(m) \sqsubseteq \zeta(g)\}.
\end{aligned}
$$

and clearly X is re. (see Exercise 4, p. 73) and an index for it is effectively obtainable from one for $Graph(f)$. Then one effectively obtains an index from X for $f(x)$ (see Exercise 1, p. 72).

$\Leftarrow$) With k as in Exercise 3, p. 72, we have: $\langle m, n \rangle \in Graph(f)$ iff $\varepsilon'_m \sqsubseteq f(\varepsilon_n)$ iff $\varepsilon'_m \sqsubseteq f(\zeta_{k(n)})$ iff $\varepsilon'_m \sqsubseteq \zeta(r(k(n)))$ and by Exercise 4, p. 73, this is clearly a partial recursive relation. ∎

Of course the theorem says, among other things, that computable functions preserve computability. The theorem suggests a natural definition of morphisms $\langle r, f \rangle: \langle \mathbf{D}, \zeta \rangle \rightarrow \langle \mathbf{D}', \zeta' \rangle$ of enumerations of computable elements as pairs $r: \mathbf{N} \rightarrow \mathbf{N}$, $f: \mathbf{D} \rightarrow \mathbf{D}'$ of recursive functions and continuous functions which make the diagram of the theorem commute. The theorem says that if $\langle r, f \rangle$ is a morphism then f is computable and if f is computable there is a recursive r making $\langle r, f \rangle$ a morphism; note the equivalence is effective. We shall have occasion in the exercises to greatly strengthen this result.

Exercises

1. Show that a continuous $f: \mathbf{D} \rightarrow \mathbf{D}'$ of domains effectively given w.r.t. ε, ε' is computable iff $O' \mapsto f^{-1}(O')$ effectively sends computable open sets in $\mathbf{D}'$ to computable open sets in $\mathbf{D}$.

2 (Rice-Shapiro). Show that for any $X \subseteq C_{\mathbf{D}}$ we have $\zeta^{-1}(X)$ re. iff there is a (necessarily unique) computable open set O with $X = O \cap C_{\mathbf{D}}$. This means that any partial recursive predicate on codes (= texts) for computable elements which respects extensional equality ($m \simeq_\zeta n \cong \zeta(m) = \zeta(n)$) must be continuous on $\mathbf{D}$. [*Hint* Apply the idea of the proof of the classical Rice-Shapiro theorem.]

3 (Myhill-Shepherdson). Let $\mathbf{D}$, $\mathbf{D}'$ be eff. given w.r.t. ε, ε'. Show that if $r: \mathbf{N} \rightarrow \mathbf{N}$ is a recursive function which respects extensional equality (i.e. $m \simeq_\zeta n \Rightarrow r(m) \simeq_{\zeta'} r(n)$) then there is a (necessarily unique) computable function $f: \mathbf{D} \rightarrow \mathbf{D}'$ which makes $\langle r, f \rangle$ a morphism of enumerations of computable elements. [*Hint* By Theorem 1 the only problem is to prove continuity. This can be done by applying Exercise 2, or the Myhill-Shepherdson.] Again this shows how an intensional operator which respects extensional equivalence is (essentially) a computable function — providing more evidence for continuity.

Least Fixed-Points

Of course, we can also effectivise the least fixed-point construction. Let $f: \mathbf{D} \rightarrow \mathbf{D}$ be computable. Then with r as in Theorem 1 we have:

$$f^n(\perp) = \zeta_{r^n(k(b))}.$$

Hence by Exercise 5, p. 73 we obtain an index of Yf and this index is effectively obtainable from an index for f.

Exercises

1. Let $\mathbf{D}$ be eff. given w.r.t. ε. Show that for any partial recursive function f there is a total recursive g, such that for any m:

(1) If $f(n)\downarrow$ then $g(n)\simeq_\zeta f(n)$,

(2) If $f(n)\uparrow$ then $g(n)\simeq_\zeta k(b)$.

2 Ershov-Kleene Fixed-Point Theorem. Show that for any partial recursive r there is an n (effectively obtainable from any index for r) such that if $r(n)\downarrow$ then $r(n_r)\simeq_\zeta n_r$.

Open Question. In case $\langle r, f\rangle: \langle \mathbf{D},\zeta\rangle \to \langle \mathbf{D},\zeta\rangle$ is a morphism do we have $\zeta_n = Yf$? That is do intensional and extensional fixed-points coincide — the answer may depend on exactly how n_r is defined.

Remark. In "Gödel numbering of domain theoretic computable functions — a generalisation of Rogers Indexing Theory, Report 138, University of Leeds", Kanda considers general systems of indexings, τ, where if $\mathbf{D}$ is eff. given w.r.t. ε then $\tau_\varepsilon: \mathbf{N}\to C_{\mathbf{D}}$ is an enumeration of the computable elements of $\mathbf{D}$. He gave axioms for such a system to be *admissible* (our ζ's form an admissible system) and showed that if ζ, ζ' are admissible then for any $\langle \mathbf{D},\varepsilon\rangle$ there is a recursive isomorphism

$$\langle r, id\rangle: \tau_\varepsilon \to \tau'_\varepsilon$$

effectively obtainable from the index of $\mathbf{D}$.

We now turn to effectivising the concerns of Chapter 2, 3, 4, 5.

Products

Let $\mathbf{D}$, $\mathbf{E}$ be eff. given domains w.r.t. ε, ε' with indices $\langle c, b, u\rangle$ and $\langle c', b', u'\rangle$. Then we can define an enumeration $\varepsilon'' = \varepsilon\times\varepsilon'$ of $B_{\mathbf{D}\times\mathbf{E}} = B_{\mathbf{D}}\times B_{\mathbf{E}}$ by Chapter 6:

$$\varepsilon\times\varepsilon'(\langle m, n\rangle) = \langle \varepsilon_m, \varepsilon'_n\rangle.$$

This makes $\mathbf{D}\times\mathbf{E}$ effectively given as we can see from:

(1) $\varepsilon''_{\langle m,m'\rangle}\uparrow\varepsilon''_{\langle n,n'\rangle} \equiv (\varepsilon_m\uparrow\varepsilon_n)\wedge(\varepsilon'_{m'}\uparrow\varepsilon'_{n'})$.

(2) $\varepsilon''_{\langle b,b'\rangle} = \bot_{\mathbf{D}\times\mathbf{E}}$.

(3) $\varepsilon''_{\langle n,n'\rangle} = \varepsilon''_{\langle l,l'\rangle}\sqcup\varepsilon''_{\langle m,m'\rangle} \equiv (\varepsilon_n = \varepsilon_l\sqcup\varepsilon_m \wedge \varepsilon'_{n'} = \varepsilon'_{l'}\sqcup\varepsilon'_{m'})$.

Exercise

1. Show that $\zeta'': \mathbf{N}\to C_{\mathbf{D}\times\mathbf{E}}$ is recursively equivalent to $\langle\zeta,\zeta'\rangle: \mathbf{N}\to(C_{\mathbf{D}}\times C_{\mathbf{E}})$ $(= C_{\mathbf{D}\times\mathbf{E}})$ in that there are morphisms $\langle r, id\rangle: \zeta''\to\langle\zeta,\zeta'\rangle$ and $\langle\overline{r}, id\rangle: \langle\zeta,\zeta'\rangle\to\zeta''$.

Of course all this is effective in the indices of $\mathbf{D}$ and $\mathbf{E}$ and so there is a recursive x so that if i, j are indices of $\mathbf{D}$ and $\mathbf{E}$ then $i\times j$ is one of $\mathbf{D}\times\mathbf{E}$.
Note that the projections $\mathbf{D}\xleftarrow{\pi_0}\mathbf{D}\times\mathbf{E}\xrightarrow{\pi_1}\mathbf{E}$ are computable as:

$$Graph(\pi_0) = \{\langle m, n\rangle \mid \varepsilon_n\sqsubseteq\varepsilon_{\pi_0(m)}\},$$
$$Graph(\pi_1) = \{\langle m, n\rangle \mid \varepsilon_n\sqsubseteq\varepsilon_{\pi_1(m)}\}.$$

Let $\mathbf{C}$ also be an effectively given domain, w.r.t. ε'' and suppose $f: \mathbf{C}\to\mathbf{D}$, $g: \mathbf{C}\to\mathbf{E}$ are computable w.r.t. ε'', $\varepsilon\times\varepsilon'$ as:

$$Graph(\langle f, g\rangle) = \{\langle m, n\rangle \mid \langle m, n_0\rangle\in Graph(f)\wedge\langle m, n_1\rangle\in Graph(g)\}$$

and this is clearly an effective construction. As a consequence if $f: \mathbf{D}'\to\mathbf{D}$, $g: \mathbf{E}'\to\mathbf{E}$ are computable so is $f\times g: \mathbf{D}'\times\mathbf{E}'\to\mathbf{D}\times\mathbf{E}$ and the product construction is effective on the graphs.

Exercises

2. Carry out the same details for finite and effective denumerable products.

3. Show $\sqcap: \mathbf{D}\times\mathbf{D}\to\mathbf{D}$ is computable (see Exercise 4) for any effectively given $\mathbf{D}$ as is $B_\sqcup: \mathbf{D}^3\to\mathbf{D}$ where $B_\sqcup(x, y, z) = (x\sqcap z)\sqcup(y\sqcap z)$. If $\mathbf{D}$ is a complete lattice show $\top$ and $\sqcup$ are computable.

4. For any eff. given $\mathbf{D}$ w.r.t. ε show $\#_n: \mathbf{D}\to\mathbf{T}$ is computable where:

$$\#_n(x) = \begin{cases} tt & (x\,\#\,\varepsilon_n),\\ f\!f & (x\sqsupseteq\varepsilon_n),\\ \bot & (\text{otherwise}).\end{cases}$$

Show also that $cond$, $parcond: \mathbf{T}\times\mathbf{D}\times\mathbf{D}\to\mathbf{D}$, $=: \mathbf{D}\times\mathbf{D}\to\mathbf{T}$ ($\mathbf{D}$ discrete), $\partial: \mathbf{D}\to\mathbf{T}$ are all computable.

Function Spaces

Let $\mathbf{D}$, $\mathbf{E}$ be effectively given domains w.r.t. ε, ε' and indices i, j. Then by p. 68, Chapter 6 we can define an enumeration, β, of a subbasis of $\mathbf{D} \to \mathbf{E}$ by:

$$\beta_{\langle m,n\rangle} = (\varepsilon_m \Rightarrow \varepsilon'_n).$$

By Exercise 2, p. 68, Chapter 6 we see that the conditions of Lemma 1 hold and we obtain an enumeration $(\varepsilon \to \varepsilon')$ of $(\mathbf{D} \to \mathbf{D}')$ as an effectively given domain, where:

$$(\varepsilon \to \varepsilon')_{\{\!|m_0,\ldots,m_{n-1}|\!\}} = \begin{cases} \bigsqcup_{a\in\{m_0,\ldots,m_{n-1}\}} \varepsilon_{(a)_0} \Rightarrow \varepsilon'_{(a)_1} & (\uparrow\{\beta_{m_i} \mid i < n\}), \\ \perp & (\text{otherwise}). \end{cases}$$

This is all effective and so we have a recursive function $\to$ s.t. if i, j are indices of $\mathbf{D}$, $\mathbf{E}$ w.r.t. ε, ε' then $i \to j$ is one of $\mathbf{D} \to \mathbf{E}$ w.r.t. $\varepsilon \to \varepsilon'$.

Now we have an enumeration $\zeta\colon \mathbf{N} \to C_{(\mathbf{D}\to\mathbf{E})}$ of the computable elements of $(\mathbf{D} \to \mathbf{E})$ and see further that

$$\begin{array}{lll} f\colon \mathbf{D} \to \mathbf{E} \text{ is in } C_{\mathbf{D}\to\mathbf{E}} & \text{iff} & \{m \mid (\varepsilon \to \varepsilon')_m \sqsubseteq f\} \text{ is re.} \\ & \text{iff} & \{\{\!|m_0,\ldots,m_{n-1}|\!\} \mid \uparrow\{\beta_{m_0},\ldots,\beta_{m_{n-1}}\} \wedge \forall i < n.\beta_{m_i} \sqsubseteq f\} \text{ is re.} \\ & \text{iff} & \{\{\!|m_0,\ldots,m_{n-1}|\!\} \mid \uparrow\{\beta_{m_0},\ldots,\beta_{m_{n-1}}\} \wedge \forall i < n.m_i \in \mathit{Graph}(f)\} \text{ is re.} \\ & \text{iff} & \mathit{Graph}(f) \text{ is re.} \\ & \text{iff} & f \text{ is computable.} \end{array}$$

All this is effective and one recursively translates between indices m with $f = \zeta(m)$ and indices of f as a computable function. This gives us the enumeration of computable functions we shall use.

Exercises

1. Show there is a recursive function id so that if $\mathbf{D}$ has index i then $id(i)$ is an index of $id_{\mathbf{D}}$.

2. Let $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$ be eff. given w.r.t. ε, ε', ε'' with indices i, j, k. Show there is a recursive function, $\circ$, such that if $\zeta\colon \mathbf{N} \to C_{(\mathbf{D}\to\mathbf{E})}$, $\zeta'\colon \mathbf{N} \to C_{(\mathbf{E}\to\mathbf{F})}$, $\zeta''\colon \mathbf{N} \to C_{(\mathbf{D}\to\mathbf{F})}$ are the appropriate enumerations of $C_{(\mathbf{D}\to\mathbf{E})}$ etc. w.r.t. $(\varepsilon \to \varepsilon')$ etc. then for all m, n:

$$\zeta_m \circ \zeta_n = \zeta_{m \circ_{i,j,k} n}$$

(where $m \circ_{i,j,k} n$ means $\circ(m,i,j,k,n)$).

3. Let $\mathbf{D}$ be eff. given w.r.t. ε. Show that $Y_{\mathbf{D}}\colon (\mathbf{D} \to \mathbf{D}) \to \mathbf{D}$ is computable w.r.t. $(\varepsilon \to \varepsilon) \to \varepsilon$ and there is a recursive function Y s.t. if i is an index of $\mathbf{D}$ then $Y_{\mathbf{D}}$ has index $Y(i)$.

4. Show that there are recursive functions π_0, π_1, $\langle\cdot,\cdot\rangle$, $\times$ such that if $\mathbf{D}$, $\mathbf{E}$, $\mathbf{C}$, $\mathbf{D}'$, $\mathbf{E}'$ are eff. given domains w.r.t. ε_0, ε_1, ε, ε'_0, ε'_1 and indices i, j, k, l, m then:

$$\pi_0\colon \mathbf{D} \times \mathbf{E} \to \mathbf{D} = \zeta_{\pi_0(\langle i,j\rangle)},$$
$$\pi_1\colon \mathbf{D} \times \mathbf{E} \to \mathbf{E} = \zeta_{\pi_1(\langle i,j\rangle)}$$

and if $f\colon \mathbf{C} \to \mathbf{D}$, $g\colon \mathbf{C} \to \mathbf{E}$ have indices a, b then

$$\langle f, g\rangle = \zeta_{\langle a,b\rangle_{i,j,k}}$$

and if $f\colon \mathbf{D} \to \mathbf{D}'$, $g\colon \mathbf{E} \to \mathbf{E}'$ have indices a, b then

$$f \times g = \zeta_{a \times_{i,j,l,m} b}.$$

In general we will drop the indices of the spaces when they can be understood from the context. Continuing the study of function space we now see that $\mathit{eval}\colon (\mathbf{D} \to \mathbf{E}) \times \mathbf{D} \to \mathbf{E}$ is computable w.r.t. $(\varepsilon \to \varepsilon') \times \varepsilon$, ε'. One way is to note that

$$\begin{aligned} \mathit{eval} &= \bigsqcup\{\langle \varepsilon_m \Rightarrow \varepsilon'_n, \varepsilon_m\rangle \Rightarrow \varepsilon'_n\} \\ &= \bigsqcup\{\langle \beta_{\langle m,n\rangle}, \varepsilon_m\rangle \Rightarrow \varepsilon'_n\} \\ &= \bigsqcup\{\langle (\varepsilon \to \varepsilon')_{\{\!|\langle m,n\rangle|\!\}}, \varepsilon_m\rangle \Rightarrow \varepsilon'_n\} \\ &= \bigsqcup\{((\varepsilon \to \varepsilon') \times \varepsilon)_k \Rightarrow \varepsilon'_n \mid k = \langle \{\!|\langle m,n\rangle|\!\}, m\rangle\} \\ &= \bigsqcup\{(((\varepsilon \to \varepsilon') \times \varepsilon) \to \varepsilon')_k \mid k = \{\!|\langle\langle \{\!|\langle m,n\rangle|\!\}, m\rangle, n\rangle|\!\}\} \end{aligned}$$

and use Exercise 1, p. 72. Another way is to use the effective nature of Theorem 1. Of course this is all effective and we have a recursive *eval* with

$$eval = \zeta_{eval(i,j)}.$$

Now let F be eff. given w.r.t. ε'' and with index k. We check that if $f: (\mathbf{F} \times \mathbf{D}) \to \mathbf{E}$ is computable w.r.t. $\varepsilon'' \times \varepsilon$, ε' then $Curry(f): \mathbf{F} \to (\mathbf{D} \to \mathbf{E})$ is computable w.r.t. ε'', $(\varepsilon \Rightarrow \varepsilon')$. For $Curry(f) = \bigsqcup\{\varepsilon''_k \Rightarrow (\varepsilon_l \Rightarrow \varepsilon'_m) \mid \varepsilon'_m \sqsubseteq f(\varepsilon''_k, \varepsilon_l)\}$. This is all effective in $Graph(f)$ and we see that *Curry* itself is computable (e.g. use Theorem 1). Finally we see there is a recursive curry with

$$Curry = \zeta_{Curry(i,j,k)}.$$

Now we also see there is a recursive $\to$ s.t. if $\mathbf{D}'$, $\mathbf{D}$, $\mathbf{E}$, $\mathbf{E}'$ are eff. given w.r.t. ε'_0, ε_0, ε_1, ε'_1 and indices i, j, k, l then for any computable $f: \mathbf{D}' \to \mathbf{D}$, $g: \mathbf{E} \to \mathbf{E}'$ w.r.t. ε'_0, ε_0 and ε_1, ε'_1 and indices a, b we have:

$$f \to g = \zeta_{a \to_{i,j,k,l} b}.$$

Exercises

1 Combinatory Logic. Given $\mathbf{C}$, $\mathbf{D}$, $\mathbf{E}$, eff. given w.r.t. ε, ε', ε'' show $K: \mathbf{D} \to (\mathbf{E} \to \mathbf{D})$, $S: (\mathbf{C} \to \mathbf{D} \to \mathbf{E}) \to (\mathbf{C} \to \mathbf{D}) \to (\mathbf{C} \to \mathbf{E})$ by demonstrating the formulae:

$$\begin{aligned} K &= \bigsqcup\{\varepsilon'_m \Rightarrow (\varepsilon''_n \Rightarrow \varepsilon'_m)\}, \\ S &= \bigsqcup\{(\varepsilon_l \Rightarrow \varepsilon'_m \Rightarrow \varepsilon''_n) \Rightarrow (\varepsilon_l \Rightarrow \varepsilon'_m) \Rightarrow (\varepsilon_l \Rightarrow \varepsilon''_n)\}. \end{aligned}$$

What is the formula for $Y: (\mathbf{D} \to \mathbf{D}) \to \mathbf{D}$?

2 λ-calculus. Show that all typed λ-calculus expressions denote computable functions of their free variables.

3 Smash Product and Strict Function Space. Treat the smash product $\mathbf{D} \otimes \mathbf{E}$ of effectively given domains $\mathbf{D}$, $\mathbf{E}$ w.r.t. ε, ε' using the enumeration:

$$\varepsilon \otimes \varepsilon'_{\langle m,n \rangle} = \varepsilon_m \otimes \varepsilon'_n.$$

showing that $\mathbf{D} \otimes \mathbf{E}$ is eff. given w.r.t. $\varepsilon \otimes \varepsilon'$ and $\otimes: \mathbf{D} \times \mathbf{E} \to \mathbf{D} \otimes \mathbf{E}$, $\iota: \mathbf{D} \otimes \mathbf{E} \to \mathbf{D} \times \mathbf{E}$ being computable (and all indices are effectively obtainable from indices for $\mathbf{D}$ and $\mathbf{E}$).

Treat the strict function space $\mathbf{D} \to_\perp \mathbf{E}$ w.r.t. the enumeration

$$\beta_{\perp\langle m,n \rangle} = \varepsilon_m \Rightarrow \varepsilon'_n.$$

Show that by considering appropriate formulae that the following functions are all computable: $\otimes: [(\mathbf{D} \to_\perp \mathbf{E}) \otimes (\mathbf{D}' \to_\perp \mathbf{E}')] \to (\mathbf{D} \otimes \mathbf{D}' \to_\perp \mathbf{E} \otimes \mathbf{E}')$ and $eval: (\mathbf{D} \to_\perp \mathbf{E}) \otimes \mathbf{D} \to_\perp \mathbf{E}$ and $Curry: (\mathbf{F} \otimes \mathbf{D} \to_\perp \mathbf{E}) \to_\perp (\mathbf{F} \to_\perp \mathbf{D} \to_\perp \mathbf{E})$ and $\to_\perp: (\mathbf{D}' \to_\perp \mathbf{D}) \times (\mathbf{E} \to_\perp \mathbf{E}') \to (\mathbf{D} \to_\perp \mathbf{E}) \to_\perp (\mathbf{D}' \to_\perp \mathbf{E}')$ and that all indices are effectively obtainable from indices for $\mathbf{D}$, $\mathbf{E}$, $\mathbf{D}'$, $\mathbf{E}'$, $\mathbf{F}$.

4. Let $\mathbf{D}$, $\mathbf{E}$ be eff. given and show that each element of the projection pair $(\mathbf{D} \to_\perp \mathbf{E}) \to (\mathbf{D} \to \mathbf{E}) \to (\mathbf{D} \to_\perp \mathbf{E})$ given in Chapter 4 is computable. Conclude $f: \mathbf{D} \to_\perp \mathbf{E}$ is computable iff it is computable as an element of $\mathbf{D} \to \mathbf{E}$.

5. Show that if we turn $\times$, $\to$ into functors on $\mathcal{CPO}_\perp$ then they still act computably on morphisms.

6 Disjoint Sum. Let $\mathbf{D}$, $\mathbf{E}$ be eff. given domain w.r.t. ε, ε'. Show that $\mathbf{D} + \mathbf{E}$ is eff. given w.r.t. $\varepsilon + \varepsilon'$ where:

$$(\varepsilon + \varepsilon')_m = \begin{cases} inl(\varepsilon_n) & (\text{if } m = 2n), \\ inr(\varepsilon'_n) & (\text{if } m = 2n+1). \end{cases}$$

Show that *inl*, *inr*, *outl*, *outr*, *isl*, *isr*, $[\cdot,\cdot]$, $+$, ∇ are all computable by demonstrating such appropriate formulae as:

$$\nabla = \bigsqcup(\{inl(\varepsilon_m) \Rightarrow_\perp \varepsilon_m\} \cup \{inr(\varepsilon'_n) \Rightarrow_\perp \varepsilon'_n\})$$

and note everything is recursive in the indices of $\mathbf{D}$ and $\mathbf{E}$.

7 Lifting. Treat lifting.

8. Show that $x \in \mathbf{D} \otimes \mathbf{E}$ is computable iff $(x)_0$ and $(x)_1$ are; that $x \in \mathbf{D} + \mathbf{E}$ is computable iff it is $\perp$ or $inl(d)$ for a computable d in $\mathbf{D}$ or $inr(e)$ for a computable e in $\mathbf{E}$; that $x \in \mathbf{D}_\perp$ is computable iff it is $\perp$ or $up(d)$ for a computable d in $\mathbf{D}$. What about $\mathbf{D}^*$?

9. Treat iterated finite sums and smash products and effective denumerable sums and smash products.

10. Treat the $(\cdot)^*$ functor.

11. Let $\mathbf{D}$ be effectively given w.r.t. ε and let $r\colon \mathbf{D} \to \mathbf{D}$ be a closure operation with recursive graph. Show $\mathbf{Fix}_r$ is effectively given w.r.t. $\varepsilon^{\mathrm{R}}_m = r(\varepsilon_m)$. What happens if r is only computable?

12. Show that the computable $f\colon \mathbf{N}^{\textcircled{m}} \to_\perp \mathbf{N}$ correspond exactly to the pr. functions of m arguments.

13. Show the computable $f\colon \mathbf{A} \to \mathbf{B}$ where $\mathbf{A}$, $\mathbf{B}$ are $\mathbf{N}^k$, $(\mathbf{T}^\omega)^k$, $(\mathbf{Tapes})^k$, … etc. are definable in the typed λ-calculus using *parcond* (or, equally, parallel *or*). What difficulties do you find if you try to define the computable functions $f\colon \mathbf{N}^k \to \mathbf{N}$ which are Milner-Vuillemin sequential? Find a notion of effective sequentiality and show all of the effectively sequential $f\colon \mathbf{N}^k \to \mathbf{N}$ can be defined with parallel *or* (or equally *parcond*).

Easy Open Question. What basic stable functions (e.g. no *parcond*) are needed to define all computable stable $f\colon \mathbf{N}^k \to \mathbf{N}$? Perhaps you will need a notion of effective stability.

14. Define *type symbols*, σ, recursively by saying: o, ι are symbols and $(\sigma \times \tau)$ and $(\sigma \to \tau)$ are if σ, τ are. Let $\mathbf{D}_o = \mathbf{T}$, $\mathbf{D}_\iota = \mathbf{N}$ and $\mathbf{D}_{\sigma\times\tau} = \mathbf{D}_\sigma \times \mathbf{D}_\tau$ and $\mathbf{D}_{\sigma\to\tau} = \mathbf{D}_\sigma \to \mathbf{D}_\tau$. Show that every computable element of $\mathbf{D}_\sigma$ is definable by the typed λ-calculus using *parcond* and the (computable!) function $\exists\colon \mathbf{D}_{(\iota\to\iota)\to o}$ where:

$$\exists f = \begin{cases} tt & (\exists n. f(n) = tt), \\ ff & (f(\perp) = ff). \end{cases}$$

[*Warning* This is rather hard. See "Plotkin, LCF considered as a programming language, TCS, 5, pp. 223–255" for ideas.]

What we have done so far is show that all our constructions $\times$, $\to$, $\otimes$, $\to_\perp$, $+$, $(\cdot)_\perp$ are *computable.*

Definition. Let $T\colon \mathcal{CPO}^m_\perp \times \mathcal{CPO}^n_\perp \to \mathcal{CPO}_\perp$ be a locally continuous functor contravariant in its first m arguments and covariant in its last n arguments. It is *computable* if there is a function ε_T on enumerations and two recursive functions both called T, such that:

(1) If $\mathbf{D}_0, \ldots, \mathbf{D}_{m-1}$, $\mathbf{E}_0, \ldots, \mathbf{E}_{n-1}$ are eff. given w.r.t. $\varepsilon_0, \ldots, \varepsilon_{m-1}$, $\varepsilon'_0, \ldots, \varepsilon'_{n-1}$ and have indices $i_0, \ldots, i_{m-1}$, $j_0, \ldots, j_{n-1}$ then $T(\mathbf{D}_0, \ldots, \mathbf{D}_{m-1}, \mathbf{E}_0, \ldots, \mathbf{E}_{n-1})$ is eff. given w.r.t. $\varepsilon_T(\varepsilon_0, \ldots, \varepsilon_{m-1}, \varepsilon'_0, \ldots, \varepsilon'_{n-1})$ with index $T(i_0, \ldots, i_{m-1}, j_0, \ldots, j_{n-1})$.

(2) If $\mathbf{D}'_k$, $\mathbf{E}_l$, $\mathbf{D}_k$, $\mathbf{E}'_l$ are eff. given w.r.t. ε'_k, $\overline{\varepsilon}_l$, ε_k, $\overline{\varepsilon}'_l$ and indices i'_k, $\overline{\imath}_l$, i_k, $\overline{\imath}'_l$ then if $f_k\colon \mathbf{D}'_k \to \mathbf{D}_k$, $g_l\colon \mathbf{E}_l \to \mathbf{E}'_l$ are computable with indices a_k, b_k we have that $T(\ldots f_k \ldots, \ldots g_l \ldots)$ is computable w.r.t. $\varepsilon_T(\ldots \varepsilon_k \ldots, \ldots \overline{\varepsilon}_l \ldots)$, $\varepsilon_T(\ldots \varepsilon'_k \ldots, \ldots \overline{\varepsilon}'_l \ldots)$ with index $T_{\ldots i'_k \ldots, \ldots \overline{\imath}_l \ldots, \ldots i_k \ldots, \ldots \overline{\imath}'_l \ldots}(\ldots a_k \ldots, \ldots b_l \ldots)$.

Clearly too the projection functors and constant functors $K_{\mathbf{D}}$, with $\mathbf{D}$ eff. given, are computable in this sense.

Embeddings

Unfortunately it is not enough to take the computable embeddings to be the embeddings which are computable as their right adjoints need not be computable (see Exercise 1 below).

Definition. Let $\mathbf{D}$, $\mathbf{E}$ be eff. given w.r.t. ε, ε'. An embedding $i\colon \mathbf{D} \triangleleft \mathbf{E}$ is *computable* w.r.t. ε, ε' and with index $\langle a, b \rangle$ iff a is an index of i w.r.t. ε, ε' (as a computable function) and b is an index of i^{R} w.r.t. ε', ε.

Exercises

1 Category.

(1) For any eff. given $\mathbf{D}$ the embedding $id\colon \mathbf{D} \triangleleft \mathbf{D}$ is computable, an index being eff. obtainable from one for $\mathbf{D}$.

(2) For any eff. given $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$ w.r.t. ε, ε', ε'' if $f\colon \mathbf{D} \triangleleft \mathbf{E}$, $g\colon \mathbf{E} \triangleleft \mathbf{F}$ are computable embedding so is $g \circ f$; indeed there is a recursive function, $\circ$, such that if a is an index for f, w.r.t. ε, ε' and b is an index for g w.r.t. ε', ε'' then $b \circ a$ is one for $g \circ f$ w.r.t. ε, ε'' and further an index for $\circ$ is effectively obtainable from ones for $\mathbf{D}$, $\mathbf{E}$, $\mathbf{F}$.

2 Initial Object. For any eff. given $\langle \mathbf{D}, \varepsilon \rangle$ the embedding $\perp: \mathbf{U} \lhd \mathbf{D}$ is computable, an index being effectively obtainable from one of $\mathbf{D}$.

3. All the examples of embeddings in Chapter 4 are computable.

Theorem 2. *Let $i: \mathbf{D} \lhd \mathbf{E}$ be an embedding of eff. given domains $\mathbf{D}$, $\mathbf{E}$ w.r.t. ε, ε'. Then i is computable iff for some recursive r we have that $\langle r, i \rangle: \langle \mathbf{D}, \varepsilon \rangle \rightarrow \langle \mathbf{D}, \varepsilon' \rangle$ i.e. that the following diagram commutes:*

$$\begin{tikzcd}
\mathbf{N} \arrow[r, "r"] \arrow[d, "\varepsilon"'] & \mathbf{N} \arrow[d, "\varepsilon'"] \\
\mathbf{D} \arrow[r, "i"'] & \mathbf{D}'
\end{tikzcd}$$

Proof. $\Rightarrow$) We know that for any finite a in $\mathbf{D}$ we have $i(a)$ finite. Now:

$$\begin{aligned} i(a) = b &\equiv (b \sqsubseteq i(a)) \wedge (i(a) \sqsubseteq b) \\ &\equiv (b \sqsubseteq i(a)) \wedge (a \sqsubseteq i^{\mathrm{R}}(b)). \end{aligned}$$

So as i and i^{R} are computable, the relation $i(\varepsilon_m) = \varepsilon'_n$ is re. in m and n. But then it must be recursive as given m and n one searches for (and finds) a k with $i(\varepsilon_m) = \varepsilon'_k$ and checks if $\varepsilon_k = \varepsilon'_n$.

$\Leftarrow$) We have $\varepsilon'_n \sqsubseteq i(\varepsilon_m) \equiv \varepsilon'_n \sqsubseteq \varepsilon'_{r(m)}$ which is re. and also $\varepsilon_n \sqsubseteq i^{\mathrm{R}}(\varepsilon'_m)$ iff $i(\varepsilon_n) \sqsubseteq \varepsilon'_m \equiv \varepsilon'_{r(n)} \sqsubseteq \varepsilon'_m$ which is re. ∎

Note the proof of the equivalence is effective.

Exercise
1. Find an embedding $i: \mathbb{O}^\omega \rightarrow (\mathbb{O}_\perp)^\omega$ which is computable as a function but not as an embedding. [*Hint* Use the two embeddings of $\mathbb{O}$ in $\mathbb{O}_\perp$.] Is it the case the i^{R} computable implies i computable, in general? If not is the condition *Graph*(i^{R}) recursive strong enough?

Definition. Let $F: (\mathcal{CPO}^{\mathrm{E}})^n \rightarrow \mathcal{CPO}^{\mathrm{E}}$ be a covariant functor. It is *computable* if there is a function ε_F on enumerations and two recursive functions both called F such that

(1) If $\ldots \mathbf{D}_k \ldots$ are eff. given w.r.t. $\ldots \varepsilon_k \ldots$ and indices $\ldots i_k \ldots$ then $F(\ldots \mathbf{D}_k \ldots)$ is eff. given w.r.t. $\varepsilon_F(\ldots \varepsilon_k \ldots)$ and index $F(\ldots i_k \ldots)$.
(2) If $\ldots \mathbf{D}_k \ldots, \ldots \mathbf{E}_k \ldots$, are eff. given w.r.t. $\ldots \varepsilon_k \ldots, \ldots \varepsilon'_k \ldots$ and indices $\ldots i_k \ldots, \ldots j_k \ldots$ then if $\ldots f_k: \mathbf{D}_k \lhd \mathbf{E}_k \ldots$ are computable embeddings w.r.t. $\ldots \varepsilon_k \ldots, \ldots \varepsilon'_k \ldots$ with indices $\ldots a_k \ldots$ then $F(\ldots f_k \ldots)$ is computable w.r.t. $\varepsilon_F(\ldots \varepsilon_k \ldots)$, $\varepsilon_F(\ldots \varepsilon'_k \ldots)$ and has index $F_{\ldots i_k \ldots, \ldots j_k \ldots}(\ldots a_k \ldots)$.

Clearly if $T: \mathcal{CPO}_\perp^m \times \mathcal{CPO}_\perp^n \rightarrow \mathcal{CPO}_\perp$ is computable so is $T^{\mathrm{E}}: (\mathcal{CPO}^{\mathrm{E}})^{m+n} \rightarrow \mathcal{CPO}^{\mathrm{E}}$; further any composition of computable functors over $\mathcal{CPO}^{\mathrm{E}}$ is also computable. We leave the proof of this assertion to the reader (for simplicity he should just consider the case $m = n = 1$).

Direct Limits

Definition. Let $\Delta = \langle \mathbf{D}_m, \varepsilon_m, f_m \rangle$ be a chain of eff. given domains $\mathbf{D}_m$ w.r.t. ε_m, and with $f_m: \mathbf{D}_m \lhd \mathbf{D}_{m+1}$ being computable embeddings w.r.t. ε_m, ε_{m+1}. Then Δ is *effective*, with index δ, if $g = \{\delta\}$ is a recursive function such that

(1) $\pi_0 \circ g(m)$ is an index of $\mathbf{D}_m$ w.r.t. ε_m.
(2) $\pi_1 \circ g(m)$ is an index of f_m w.r.t. ε_m, ε_{m+1}.

Definition. Let $\Delta = \langle \mathbf{D}_m, \varepsilon_m, f_m \rangle$ be an effective chain. Then an effective cone $\rho: \Delta \rightarrow \langle \mathbf{D}, \varepsilon \rangle$ from Δ to an effectively given $\mathbf{D}$ w.r.t. ε is a cone $\rho: \langle \mathbf{D}_m, f_m \rangle \rightarrow \mathbf{D}$ with $\rho_m: \mathbf{D}_m \lhd \mathbf{D}$ computable w.r.t. ε_m, ε and such that:

(1) d is an index $\mathbf{D}$ w.r.t. ε.
(2) For any m, $\{\rho\}(m)$ is an index of $\mathbf{D}_m$ w.r.t. ε_m.

Definition. Let $\Delta = \langle \mathbf{D}_m, \varepsilon_m, f_m \rangle$ be an effective chain, and let $\rho\colon \Delta \to \langle \mathbf{D}, \varepsilon \rangle$ be an effective cone. Then ρ is *effectively universal* iff $\rho\colon \langle \mathbf{D}_m, f_m \rangle \to \mathbf{D}$ is universal and if $\rho'\colon \Delta \to \langle \mathbf{D}', \varepsilon \rangle$ is any effective cone there is a mediating computable $\theta\colon \mathbf{D} \triangleleft \mathbf{D}'$ w.r.t. ε, ε' and an index for θ is effectively obtainable from one for ρ' (and an index for ρ is $\langle a, b \rangle$ where a is an index for ρ as an effective cone and b is an index s.t. if m is an index for ρ' then $\{a\}(m)$ is one for θ w.r.t. ε, ε').

Theorem 3. *Let $\Delta = \langle \mathbf{D}_m, \varepsilon_m, f_m \rangle$ be an effective chain and let $\rho\colon \langle \mathbf{D}_m, f_m \rangle \to \mathbf{D}$ be universal (see Chapter 4). Define $\varepsilon\colon \mathbf{N} \to B_{\mathbf{D}}$ by:*

$$\varepsilon_{\langle m,n \rangle} = \rho_m(\varepsilon_m(n))$$

(see Chapter 5). Then $\rho\colon \Delta \to \langle \mathbf{D}, \varepsilon \rangle$ is effectively universal and an index for ρ is effectively obtainable from one for Δ.

Proof. First we check that $\mathbf{D}$ is effectively given w.r.t. ε by the formulae:

1.
$$\begin{aligned}
\varepsilon_{\langle k,l \rangle} \uparrow \varepsilon_{\langle m,n \rangle} &\equiv \rho_k(\varepsilon_k(l)) \uparrow \rho_m(\varepsilon_m(n)) \\
&\equiv \rho_r[f_{kr}(\varepsilon_k(l))] \uparrow \rho_r[f_{mr}(\varepsilon_m(n))] && \text{where } r = \max(k, m) \\
&\equiv \rho_r[\varepsilon_r(g_{kr}(l))] \uparrow \rho_r[\varepsilon_r(g_{mr}(n))] \\
&= \varepsilon_r(g_{kr}(l)) \uparrow \varepsilon_r(g_{mr}(n))
\end{aligned}$$

(where $g_m\colon \mathbf{N} \to \mathbf{N}$ is the recursive function for $f_m\colon \mathbf{D}_m \triangleleft \mathbf{D}_{m+1}$ guaranteed effectively obtainable from any index of f_m by Theorem 2, and g_{ln} is an evident product of g_m's).

2. If b is a code of $\perp_{\mathbf{D}_0}$ then $\langle 0, b \rangle$ is one of $\perp_{\mathbf{D}}$.

3.
$$\begin{aligned}
\varepsilon_{\langle i,j \rangle} = \varepsilon_{\langle k,l \rangle} \sqcup \varepsilon_{\langle m,n \rangle} \\
&\equiv \rho_i(\varepsilon_i(j)) \\
&= \rho_k(\varepsilon_k(l)) \sqcup \rho_m(\varepsilon_m(n)) \\
&\equiv \rho_r(\varepsilon_r(g_{ir}(j))) \\
&= \rho_r(\varepsilon_r(g_{kr}(l))) \sqcup \rho_r(\varepsilon_r(g_{mr}(n))) && \text{where } r = \max(i, k, m) \\
&\equiv \varepsilon_r(g_{ir}(j)) \\
&= \varepsilon_r(g_{kr}(l)) \sqcup \varepsilon_r(g_{mr}(n))
\end{aligned}$$

and so, clearly, an index for $\mathbf{D}$ w.r.t. ε is effectively obtainable from one for Δ.
To see that $\rho_m\colon \mathbf{D}_m \to \mathbf{D}$ is computable w.r.t. ε_m, ε just note that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{N} \arrow[r, "\langle m, - \rangle"] \arrow[d, "\varepsilon_m"'] & \mathbf{N} \arrow[d, "\varepsilon"] \\
\mathbf{D}_m \arrow[r, "\rho_m"'] & \mathbf{D}
\end{tikzcd}$$

and apply Theorem 2. We have now essentially seen that ρ is an effective cone and an index for it is effectively obtainable from one for Δ.

Now let $\rho'\colon \Delta \to \langle \mathbf{D}', \varepsilon' \rangle$ be any other effective cone. The mediating $\theta\colon \mathbf{D} \triangleleft \mathbf{D}'$ is given by the formula:

$$\theta = \bigsqcup \rho'_n \circ \rho_n^{\mathrm{R}}$$

and by Exercises 5, p. 73, 1, p. 78 and since ρ' and ρ are effective the function θ is computable. Equally the formula $\theta^{\mathrm{R}} = \bigsqcup \rho_n \circ {\rho'_n}^{\mathrm{R}}$ shows θ^{R} to be computable and we see that θ is a computable embedding with index effectively obtainable from ρ'. Thus ρ is effectively universal with an index effectively obtainable from any one for Δ. ∎

What this theorem says is that the inverse limit construction, $\lim_{\rightarrow}$ is effective when viewed as a function from effective chains to universal cones.

Theorem 4 Continuity of Functors. *Let $T\colon \mathcal{CPO}^{\mathrm{E}} \to \mathcal{CPO}^{\mathrm{E}}$ be a computable functor. Then it is effectively continuous in that if $\rho\colon \langle \mathbf{D}, \varepsilon_m, f_m \rangle \to \langle \mathbf{D}, \varepsilon \rangle$ is effectively universal so is $T(\rho)\colon \langle T(\mathbf{D}), \varepsilon_T(\varepsilon_m), T(f_m) \rangle \to \langle T(\mathbf{D}), \varepsilon_T(\varepsilon) \rangle$ and an index for $T(\rho)$ is effectively obtainable from one for ρ.*

Proof. This is an effective version of Theorem 2, Chapter 4 and is left to the reader. ∎

Algebras

The end of our long haul is now in sight! In what follows we let $T: \mathcal{CPO}^{\mathrm{E}} \to \mathcal{CPO}^{\mathrm{E}}$ be a computable functor.

Definition.

(1) A *computable* T-algebra with index $\langle a, b\rangle$ is $\langle \mathbf{D}, \varepsilon, \alpha\rangle$ where $\langle \mathbf{D}, \alpha\rangle$ is a T-algebra, $\mathbf{D}$ is eff. given w.r.t. ε with index a and α is computable w.r.t. $\varepsilon_T(\varepsilon)$, ε with index b.

(2) An *effectively-initial* T-algebra with index $\langle a, b\rangle$ is a computable T-algebra $\langle \mathbf{D}, \varepsilon, \alpha\rangle$ with index a and with $\langle \mathbf{D}, \alpha\rangle$ initial such that for any computable T-algebra $\langle \mathbf{E}, \varepsilon', \beta\rangle$ with index a' the unique T-homomorphism, $\theta: \langle \mathbf{D}, \beta'\rangle \to \langle \mathbf{E}, \beta\rangle$ is computable w.r.t. ε, ε' and has index $\{b\}(a')$.

Theorem 5 Effective Initial Solution of Recursive Domain Equations. *The initial algebra $\langle \mathbf{Fix}_T, \eta_T\rangle$ is effectively initial (for a suitable ε).*

Proof. As $\mathbf{U}$ is eff. given w.r.t. the evident ε^u and by Exercise 2, p. 79 and as T is computable, the chain $\Delta = \langle T^n(\mathbf{U}), \varepsilon_T(\varepsilon^u), T^n(\perp)\rangle$ is effective. Let $\rho: \langle T^n(\mathbf{U}), T^n(\perp)\rangle \to \mathbf{D}$ be universal and take ε as in Theorem 3. Then $\rho: \Delta \to \langle \mathbf{D}, \varepsilon\rangle$ is effectively universal. Then $T\rho: T\Delta \to \langle T\mathbf{D}, \varepsilon_T(\varepsilon)\rangle$ is also effectively universal, by Theorem 4. Further $\eta_T = \bigsqcup_n \rho_{n+1} \circ (T\rho_n)^{\mathrm{R}}$ and so η_T is computable w.r.t. $\varepsilon_T(\varepsilon)$, ε. Thus $\langle \mathbf{Fix}_T, \varepsilon, \eta_T\rangle$ is a computable T-algebra. Let $\langle \mathbf{D}, \varepsilon, \alpha'\rangle$ be any other one. Then following the proof of Theorem 1, Chapter 5 define $\rho'_m: T^n(\mathbf{U}) \lhd \mathbf{D}$ by $\rho'_0 = \perp$ and $\rho'_{m+1} = \alpha \circ T(\rho'_m)$. Then as in Theorem 1 we see ρ' is a cone from $\langle T^n(\mathbf{U}), T^n(\perp)\rangle$ to $\mathbf{D}$ and it is easily seen (by induction on n) to be an effective cone from Δ to $\langle \mathbf{D}, \varepsilon\rangle$. But $\rho: \Delta \to \langle \mathbf{D}, \varepsilon\rangle$ is effectively universal and so we can obtain an index for the mediating computable $\theta: \mathbf{Fix}_T \to \mathbf{D}$ effectively from one for ρ and hence from one for $\langle \mathbf{D}, \varepsilon\rangle$. But, as in the proof of Theorem 1, this θ is the required mediating T-homomorphism and we have seen that $\langle \mathbf{Fix}_T, \varepsilon, \eta_T\rangle$ is indeed effectively initial. ∎

In fact this proof is also effective and, with suitable definitions, one obtains an index for the effectively-initial T-algebra from one for T. So: taking effectively-initial fixed-points of computable functors is itself an effective operation.

The above theorem easily extends to multiple fixed-points and we see that all the domains and functions we have considered in the notes are computable and so whatever the programming language for any program, P, and semantic function, M, defined by our methods we have that $M(P)$ is computable. This gives the *possibility* of an operational semantics. For example if $\mathbf{S} = \mathbf{Id} \to_\perp \mathbf{N}$ and $M(P): \mathbf{S} \to_\perp \mathbf{S}$ we have that given an index for σ we get one for $M(P)(\sigma)(x)$ for any variable x and the required integer (if it exists) can be found by running a Turing Machine on this index.

However this is all only of theoretical interest. It would be more useful to patch up the various spotty results on definability mentioned in the exercises by showing that some (formally defined) metalanguage can define exactly the computable elements and that the language has a good operational semantics (as in, for example, the LCF paper) and then one can apply Mosses' ideas to write a *complete* compiler-compiler.

Exercises

1. Give an effective version of Theorem 3, p. 54, Chapter 5.

2. Is there anything more one can say along the lines of Theorem 4, p. 55, Chapter 5 by considering effectiveness?

3. It is boring to continually mention indices. Can you formulate a metatheoretical assertion to the effect that if a proof is effective (in a suitable intuitionistic logic) then one obtains, via a Kleene realizability, a suitable statement on indices?

8. Nondeterminism and Parallelism

We show how to extend our theory to deal with problems of nondeterminism and parallelism. The interest of the latter is clear; that of the former arises when we try to reduce parallelism to nondeterminism and the idea is to explain a parallel construct $c||c'$ in terms of the possible *atomic actions* of c and c' and the set of their *interleavings*. It is surprising how many difficulties arise and we do not regard the theory as settled.

Discrete Powerdomains

Suppose we enrich the usual simple imperative language with a nondeterministic choice construct, c **or** c'. The idea is that an execution of c **or** c' consists of a choice of c and an execution of c or a choice of c' and an execution of c'. (This choice may be made by an implementation or forced by some external circumstance.) We regard c **or** c' as correct, terminating (etc.). Now consider the following three programs (where **loop** $\stackrel{\mathrm{def}}{=}$ **while** *true* **do sleep**)

(1) $x := 1$,

(2) $x := 1$ **or loop**,

(3) **loop**.

Which should be considered equivalent? One view is that none are since (1) always terminates unlike (2) & (3) and (2) can terminate unlike (3). Another is that $(1) = (2) \neq (3)$ since (1) and (2) give the same set of results, if any. Another is that $(1) \neq (2) = (3)$ since (2) and (3) can both (unlike (1)) fail to terminate and so nothing can be guaranteed of either of them. All three views have their place: the first is "what actually happens", the second for partial correctness properties (Hoare), the third for total correctness (Dijkstra). We shall give semantics for all three.

The Relational View

To this end let $\mathbf{S}$ be the countable set of states and suppose we want a semantics for the second view (the simplest case). Then for a given initial state σ a program c should give us a *set* $X \subseteq \mathbf{S}$ of final states (since we do not wish to record nontermination). Thus we could expect

$$\mathcal{C}[\![c]\!]: \mathbf{S} \to \mathcal{P}(\mathbf{S})$$

(where, of course, $\mathcal{P}(\mathbf{S})$ is the set of all subsets of $\mathbf{S}$). That is we use a *nondeterministic function* for our semantics. And it will also be natural to order $\mathcal{P}(\mathbf{S})$ by subset and use the induced pointwise order on $\mathcal{C}[\![c]\!]$.

Exercise 1. Show that

$$\mathbf{S} \to \mathcal{P}(\mathbf{S}) \cong \mathcal{P}(\mathbf{S} \times \mathbf{S})$$

The set $\mathcal{P}(\mathbf{S} \times \mathbf{S})$ is the set of all *relations* over $\mathbf{S}$; try rephrasing what we do below in terms of these relations.

In order to compare nondeterministic functions we note a very important fact which will prove a reliable guide in more general situations.

Fact 1. *Let X and Y be countable sets. Then for any $f: X \to \mathcal{P}(Y)$ there is a unique strict continuous, additive $f^\dagger: \mathcal{P}(X) \to_{\mathrm{SA}} \mathcal{P}(Y)$ (called the extension of f) such that the following diagram commutes.*

$$\begin{tikzcd}
X \arrow[d, "\{\cdot\}"'] \arrow[dr, "f"] & \\
\mathcal{P}(X) \arrow[r, "f^\dagger"'] & \mathcal{P}(Y)
\end{tikzcd}$$

Further $(\cdot)^\dagger: (X \to \mathcal{P}(Y)) \cong \mathcal{P}(X) \to_{\mathrm{SA}} \mathcal{P}(Y)$ is an isomorphism of partial orders.

Proof. Suppose $g: \mathcal{P}(X) \to_{\mathrm{SA}} \mathcal{P}(Y)$ makes the diagram commute.
$\Leftarrow$) Then for any $A \subseteq X$ we have

$$\begin{aligned} g(A) &= g(\bigcup_{a \in A} \{a\}) \\ &= \bigcup_{a \in A} g(\{a\}) && g \text{ is strict, additive and continuous and } A \subseteq X \text{ is countable} \\ &= \bigcup_{a \in A} f(a) \end{aligned}$$

showing uniqueness. Conversely using this formula to define $f^\dagger$ we see that $f^\dagger$ is clearly continuous and additive and makes the diagram commute as:

$$f^\dagger(\{a\}) = \bigcup_{a' \in \{a\}} f(a') = f(a).$$

It follows at once that $\dagger$ is a bijection with inverse $\lambda g.g \circ \{\cdot\}$ and as both $\dagger$ and its inverse are evidently monotonic they are therefore isomorphisms (of partial orders). ∎

Exercise 2. Where was the countability of X and Y used in the above (if at all)? Can you extend the definitions of nondeterministic functions so that Fact 1 holds for arbitrary X?

Exercise 3. Noting that $X \to \mathcal{P}(Y) \cong X_\perp \to_\perp \mathcal{P}(Y)$ can you provide a variant of Fact 1 to apply to nondeterministic functions of type $X_\perp \to \mathcal{P}(Y)$?

Exercise 4. Show using Exercise 1 and Fact 1 that there is an isomorphism $\Lambda: (X \to \mathcal{P}(Y)) \cong \mathcal{P}(Y) \to_{\mathrm{SA}} \mathcal{P}(X)$ given by:

$$\Lambda(m)(B) = \{a \in X \mid m(a) \cap B \neq 0\}.$$

Definition 1. For any function $f: \mathcal{P}(Y) \to \mathcal{P}(X)$, its *dual* $\tilde{f}: \mathcal{P}(Y) \to \mathcal{P}(X)$ is defined by:

$$\tilde{f}(B) = X \backslash f(Y \backslash B).$$

Exercise 5. Show that $\tilde{\tilde{f}} = f$, that $f \sqsubseteq g$ iff $\tilde{f} \sqsupseteq \tilde{g}$, that f is monotonic, continuous, additive, ... iff $\tilde{f}$ is monotonic, dual-continuous, multiplicative, ... Conclude that $\tilde{(\cdot)}: [\mathcal{P}(Y) \to \mathcal{P}(X)] \to [\mathcal{P}(Y) \to \mathcal{P}(X)]$ is an isomorphism and cuts down to an isomorphism of continuous and dual continuous functions, etc., etc.

Exercise 6. For any $m: X \to \mathcal{P}(Y)$ and $B \subseteq Y$ we define the *weakest liberal precondition* $wlp(m, B)$ by

$$wlp(m, B) = \{a \in X \mid m(a) \subseteq B\}.$$

Show that $\lambda B.wlp(m, B)$ is the dual of $\Lambda(m)$.

Composition. Now we can compose two nondeterministic primitives $X \xrightarrow{f} \mathcal{P}(Y)$ and $Y \xrightarrow{g} \mathcal{P}(Z)$ by defining $f;g: X \to \mathcal{P}(Z)$ as

$$f;g = g^\dagger \circ f.$$

Exercise 7. Show that composition is associative with unit the singleton function. [*Hint* This is easy using Fact 1.]

Now consider the following little nondeterministic imperative language

$$c ::= a \mid \mathbf{skip} \mid c;c \mid \mathbf{if}\ b\ \mathbf{then}\ c\ \mathbf{else}\ c \mid \mathbf{while}\ b\ \mathbf{do}\ c \mid c\ \mathbf{or}\ c$$

where we are given semantics $\mathcal{A}: \mathbf{ACom} \to (\mathbf{S} \to \mathbf{S})$ and $\mathcal{B}: \mathbf{BExp} \to (\mathbf{S} \to \mathbf{T})$ for atomic commands and Boolean expressions. Here is a semantics

$$\mathcal{C}: \mathbf{Com} \to (\mathbf{S} \to \mathcal{P}(\mathbf{S}))$$

for the language:

$$\begin{aligned} &\mathcal{C}[\![a]\!] = \lambda\sigma \in \mathbf{S}.\{\mathcal{A}[\![a]\!]\}; \\ &\mathcal{C}[\![\mathbf{skip}]\!] = \{\cdot\}; \\ &\mathcal{C}[\![c_1; c_2]\!] = \mathcal{C}[\![c_1]\!]; \mathcal{C}[\![c_2]\!]; \\ &\mathcal{C}[\![\mathbf{if}\ b\ \mathbf{then}\ c_1\ \mathbf{else}\ c_2]\!] = \lambda\sigma \in \mathbf{S}.\mathbf{if}\ \mathcal{B}[\![b]\!](\sigma)\ \mathbf{then}\ \mathcal{C}[\![c_1]\!](\sigma)\ \mathbf{else}\ \mathcal{C}[\![c_2]\!](\sigma); \\ &\mathcal{C}[\![\mathbf{while}\ b\ \mathbf{do}\ c]\!] = \mu m \in \mathbf{S} \to \mathcal{P}(\mathbf{S}).\lambda\sigma \in \mathbf{S}.\mathbf{if}\ \mathcal{B}[\![b]\!](\sigma)\ \mathbf{then}\ \mathcal{C}[\![c]\!]; m\ \mathbf{else}\ \{\sigma\}; \\ &\mathcal{C}[\![c_1\ \mathbf{or}\ c_2]\!] = \lambda\sigma \in \mathbf{S}.\mathcal{C}[\![c_1]\!](\sigma) \cup \mathcal{C}[\![c_2]\!](\sigma). \end{aligned}$$

So all we need is singleton, extension (for composition) and binary union.

Exercise 8. Show that Cartesian Product, $\mathcal{P}(A) \times \mathcal{P}(B) \xrightarrow{\times} \mathcal{P}(A \times B)$ is strict continuous and additive in each argument. Show that for any $f: A \times B \rightarrow \mathcal{P}(C)$ there is a unique $f^{\dagger_2}: \mathcal{P}(A) \times \mathcal{P}(B) \rightarrow \mathcal{P}(C)$ which is strict, continuous and additive in each argument and which makes the following diagram commute

$$\begin{tikzcd} A \times B \arrow[d, "\{\cdot\} \times \{\cdot\}"'] \arrow[dr, "f"] & \\ \mathcal{P}(A) \times \mathcal{P}(B) \arrow[r, "f^{\dagger_2}"'] & \mathcal{P}(C) \end{tikzcd}$$

Show too that $\dagger_2$ is given by: $f^{\dagger_2} = f^{\dagger} \circ \times$ (and so $\times = \{\cdot\}^{\dagger_2}$). What is the natural function from $\mathcal{P}(A \times B)$ to $\mathcal{P}(A) \times \mathcal{P}(B)$? What relationship do you thereby find between $\mathcal{P}(A \times B)$ and $\mathcal{P}(A) \times \mathcal{P}(B)$? Extend all this to products of several powersets.

Exercise 9. Let Σ be a finite alphabet. Consider the following little applicative language for denoting subsets of Σ^*:

$$e ::= a \mid \lambda \mid e.e \mid e \ \mathbf{or} \ e \mid \mu x.e \mid x$$

where a ranges over Σ, λ is the empty word, $e.e'$ is language product (the extension considered in the previous exercise of semigroup multiplication in Σ^*) and the rest is obvious. Show that the closed expressions give exactly the context-free languages. How would you alter the notation to obtain exactly the regular languages?

Exercise 10. Nondeterministic phenomena arise even in deterministic languages when we perform *dataflow analysis*. This exercise gives a *very* rough picture. Consider the simple language

$$c ::= a \mid \mathbf{skip} \mid c; c \mid \mathbf{if}\ b\ \mathbf{then}\ c\ \mathbf{else}\ c \mid \mathbf{while}\ b\ \mathbf{do}\ c$$

where we are given $\mathcal{A}: \mathbf{ACom} \rightarrow (\mathbf{S} \rightarrow \mathbf{S})$ and $\mathcal{B}: \mathbf{BExp} \rightarrow (\mathbf{S} \rightarrow \mathbf{T})$ and define $\mathcal{C}: \mathbf{Com} \rightarrow (\mathbf{S} \rightarrow \mathbf{S}_\perp)$ is usual. Suppose we try to represent $\mathbf{S}$ by a set $\mathbf{R}$ via a surjective *representation* function $r: \mathbf{S} \rightarrow \mathbf{R}$ (*example* $\mathbf{S} = \mathbf{N}$ and $\mathbf{R} = \{$"odd", "even"$\}$). We regard the inverse of r as being of type $r^{-1}: \mathbf{R} \rightarrow \mathcal{P}(\mathbf{S})$. Now define abstractions $\mathcal{A}_{\mathrm{R}}: \mathbf{ACom} \rightarrow (\mathbf{R} \rightarrow \mathcal{P}(\mathbf{R}))$ and $\mathcal{B}_{\mathrm{R}}: \mathbf{BExp} \rightarrow (\mathbf{S} \rightarrow \mathcal{P}(\mathbf{T}))$ of the semantics of atomic commands and Boolean expressions by

$$\mathcal{A}_{\mathrm{R}}[\![a]\!] = r^{-1}; \{\cdot\} \circ r \circ \mathcal{A}[\![a]\!],$$
$$\mathcal{B}_{\mathrm{R}}[\![b]\!] = r^{-1}; \{\cdot\} \circ \mathcal{B}[\![b]\!].$$

The exercise is to define an abstraction $\mathcal{C}_{\mathrm{R}}: \mathbf{Com} \rightarrow (\mathbf{R} \rightarrow \mathcal{P}(\mathbf{R}))$ of the semantics of commands and prove that

$$\mathcal{C}_{\mathrm{R}}[\![c]\!] \supseteq r^{-1}; (\{\cdot\} \circ \mathcal{C}[\![c]\!]).$$

Reference. P. Cousot and R. Cousot. Abstract Interpretation: A Unified Lattice Model for Static Analysis of Program by Construction or Approximation or Fixpoints. 4th POPL.

The Egli-Milner Order

Now we turn to the first possibility where we want to record "what actually happens". It is natural here to consider subsets of $\mathbf{S}_\perp$ but we must answer two questions.

A. What subsets should we consider?
B. How should they be ordered?

This situation is similar to that faced before when we had to decide which functions between cpo's to consider and how to order them.

A. *What Subsets?* First since every program in our language must produce something (even if it is just $\perp$) we consider only nonempty subset. (Variants with an empty subset are considered in the exercises — they provide inessential, but useful alternatives.) More importantly consider the following attempt to write a program to output an arbitrary integer:

$$\begin{array}{l} x := 0;\ \mathit{alarm} := \text{“off”}; \\ \textbf{while}\ \mathit{alarm} = \text{“off”} \\ \textbf{do}\ x := x + 1\ \textbf{or}\ \mathit{alarm} := \text{“on”}. \end{array}$$

Clearly this can either produce an arbitrary integer or fail to terminate. This is not accidental as the following informal argument shows. Consider the tree of all possible execution paths. In the case of the above program it looks like this

$$\begin{tikzcd} \langle 0, \text{“off”}\rangle \arrow[r] \arrow[d] & \langle 1, \text{“off”}\rangle \arrow[r] \arrow[d] & \langle 2, \text{“off”}\rangle \arrow[r] \arrow[d] & \langle 3, \text{“off”}\rangle \arrow[r] \arrow[d] & \ldots \\ \langle 0, \text{“on”}\rangle & \langle 1, \text{“on”}\rangle & \langle 2, \text{“on”}\rangle & \langle 3, \text{“on”}\rangle & \ldots \end{tikzcd}$$

and here and in general it will be a *finitely-branching tree*, with the branching corresponding to the possible executions of nondeterministic constructs. Therefore if there are infinitely many different possible final states (e.g. all pairs $\langle n, \text{“on”}\rangle$) then there are infinitely many nodes in the tree and so by *König's Lemma* there must be an infinite branch in the tree, that is, nontermination is possible. This is a major phenomenon when considering *finite nondeterminism* where at most finitely many choices can made at any one point.

Summing up we take, for countable X, the *powerdomain* $\mathcal{P}(X_\perp)$ to consist of all nonempty subsets of $X_\perp$ which are either finite or contain $\perp$ and consider next:

B. *What Order?* With products, the *coordinatewise* ordering was natural where each coordinate is increased to increase the order. Now the *elementwise* ordering is natural where each element is increased perhaps to a set of greater elements. So we expect

$$A \sqsubseteq_{\mathrm{EM}} B$$

just when for each x in A there is a nonempty set Y_x such that

(1) $\forall y \in Y_x . x \sqsubseteq y$,
(2) $B = \bigcup_{x \in A} Y_x$.

Here is another way (due to Egli & Milner) of saying the same thing

$$A \sqsubseteq_{\mathrm{EM}} B \quad \text{iff} \quad \begin{array}{l} (\downarrow)\ \forall x \in A . \exists y \in B . x \sqsubseteq y, \\ \wedge\ (\uparrow)\ \forall y \in B . \exists x \in A . x \sqsubseteq y. \end{array}$$

Note that, putting $Y_x = \{y \in B \mid x \sqsubseteq y\}$ condition, $(\downarrow)$ says Y_x is nonempty and (1) above on Y_x is automatic and (2) follows from the new $(\uparrow)$. You can prove that the nonemptiness requirement, (1) and (2) imply $(\downarrow)$ and $(\uparrow)$. This definition makes sense for any preorder and always defines a preorder on subsets. In the case at hand where we want $A, B \subseteq X_\perp$ then

$$A \sqsubseteq_{\mathrm{EM}} B \quad \text{iff} \quad \begin{array}{ll} \text{either} & (1)\ \perp \notin A \wedge A = B, \\ \text{or} & (2)\ \perp \in A \wedge A \backslash \{\perp\} \subseteq B \backslash \{\perp\}. \end{array}$$

So $\mathcal{P}(X_\perp)$ equipped with $\sqsubseteq_{\mathrm{EM}}$ is even a cpo. (For it is clearly a partial order with least elements $\{\perp\}$. And if $A_0 \sqsubseteq A_1 \sqsubseteq \cdots$ is an increasing ω-chain then either $\perp \notin A_n$ for some A_n, when $\bigsqcup_i A_i = A_n$ or else $\perp \in A_n$ for all n when $\bigsqcup_i A_i = \bigcup_n A_n$.)

Union. The binary union function $\cup : \mathcal{P}(X_\perp)^2 \to \mathcal{P}(X_\perp)$ is continuous. (Easy exercise)

Definition 2. A function $f : \mathcal{P}(X_\perp) \to \mathcal{P}(Y_\perp)$ is *linear* iff $f(x \cup y) = f(x) \cup f(y)$.

Fact 2. *Let X and Y be countable sets. Then for any $f: X \to \mathcal{P}(Y_\perp)$ there is a unique strict continuous linear $f^\dagger: \mathcal{P}(X_\perp) \to \mathcal{P}(Y_\perp)$ such that the following diagram commutes*

$$\begin{tikzcd} X \arrow[d, "\{\cdot\}"'] \arrow[dr, "f"] & \\ \mathcal{P}(X_\perp) \arrow[r, "f^\dagger"'] & \mathcal{P}(Y_\perp) \end{tikzcd}$$

further $\dagger$ is an isomorphism of $X \to \mathcal{P}(Y_\perp)$ and the partial order of strict continuous linear function from $\mathcal{P}(X_\perp)$ to $\mathcal{P}(Y_\perp)$.
Proof. For uniqueness we note that for *finite* $A \subseteq Y_\perp$ we have

$$f^\dagger(A) = \bigcup_{a \in A} f(a)$$

as before and for *infinite* $A = \{\perp, a_1, a_2, \ldots\}$ we have

$$\begin{aligned} f^\dagger(A) &= f^\dagger(\bigsqcup_n \{\perp, a_1, \ldots, a_n\}) \\ &= \bigsqcup_n f^\dagger(\{\perp, a_1, \ldots, a_n\}) \\ &= \bigsqcup_n [f^\dagger(\{\perp\}) \cup \bigcup_{i<n} f^\dagger(\{a_i\})] \\ &= \bigsqcup_n [\{\perp\} \cup \bigcup_{i<n} f(a_i)] \\ &= \bigcup_{a \in A} f(a) \end{aligned}$$

and the rest of the proof is an exercise for the reader. ∎

Now the composition $f; g: X \to \mathcal{P}(Z_\perp)$ of $X \xrightarrow{f} \mathcal{P}(Y_\perp)$ and $Y \xrightarrow{g} \mathcal{P}(Z_\perp)$ is defined as before by: $f; g = g^\dagger \circ f$ and again it is associative with unit the singleton function. And, again as before we can define a semantics

$$\mathcal{C}: \mathbf{Com} \to (\mathbf{S} \to \mathcal{P}(\mathbf{S}_\perp))$$

which looks just like it did before.

Exercise 11. Draw $\mathcal{P}(\mathbf{U})$, $\mathcal{P}(\mathbb{O})$, $\mathcal{P}(\mathbf{T})$, $\mathcal{P}(\mathbf{N}_\perp)$.

Exercise 12. Show that $\mathcal{P}(\mathbf{N}_\perp)$ is ω-algebraic and its finite elements are just those of finite cardinality. Is it also consistently complete?

Exercise 13. Try to find an *analogue* of $\mathbf{S} \to \mathcal{P}(\mathbf{S}) \cong \mathcal{P}(\mathbf{S} \times \mathbf{S})$. [*Hint* Consider certain pairs $\langle T, R \rangle$ with $T \subseteq \mathbf{S}$, $R \subseteq \mathbf{S}^2$, the idea being that R is the relation and T is its "termination domain".]

Exercise 14. Investigate analogue of Exercises 2, 3, 8 above.

Exercise 15. Show that $\mathcal{P}(\mathbf{S}) \cong \mathcal{P}(\mathbf{S}_\perp)/\sqsubseteq_{\mathrm{R}}$ where

$$X \sqsubseteq_{\mathrm{R}} Y \quad \text{iff} \quad (\downarrow)\ \forall x \in X. \exists y \in Y. x \subseteq y.$$

(If $\langle \mathbf{P}, \sqsubseteq \rangle$ is a preorder and $\leq$ is a preorder on $\mathbf{P}$ including $\sqsubseteq$ then $\mathbf{P}/\leq$ is the partial order $\langle \{a\downarrow \mid a \in \mathbf{P}\}, \subseteq \rangle$ where $a\downarrow \stackrel{\text{def}}{=} \{b \in \mathbf{P} \mid b \leq a\}$. Note that natural monotonic map $\downarrow: \mathbf{P} \to \mathbf{P}/\leq$.) So there is a natural map $\mathcal{P}(\mathbf{S}_\perp) \to \mathcal{P}(\mathbf{S})$. What is it?

It is possible to consider applicative programming languages for various classes of functions: $f: \mathcal{P}(X_\perp^{(1)}) \times \cdots \times \mathcal{P}(X_\perp^{(n)}) \to \mathcal{P}(Y_\perp)$ which are continuous and perhaps n-strict and perhaps $\subseteq$-monotonic or linear in each argument. For information see M. Hennessy "Powerdomain and Nondeterministic Recursive Definitions" (to appear).

The Smyth Order

Finally we consider the third possibility that (1) $\neq$ (2) = (3) (due to M. Smyth). Here we want to *identify* all sets containing $\perp$ since nothing can be guaranteed of any of them. However we want to go even further; for example we want the program

(4) $x := 1$ **or** $x := 2$.

to have a *smaller* meaning than (1) above, the idea being that less can be guaranteed of it. So if in some context (1) fails to terminate so must (4), but the converse does not hold (consider the context $C[\,] = [\,]$; **if** $x = 2$ **then loop else skip**). Thus we want: $\{1, 2\} \sqsubseteq \{1\}$.

For a formal definition we identify all sets containing $\perp$ with a standard one which for convenience we take to be $\mathbf{N}_\perp$. The remaining sets are the finite nonempty ones and so we take for any countable set X:

$$\mathcal{P}_{\mathrm{S}}(X_\perp) = \langle \{A \subseteq X_\perp \mid A = X_\perp \text{ or } A \text{ is nonempty and finite, and } A \subseteq X\}, \supseteq \rangle.$$

Exercise 16. Show that $\mathcal{P}_{\mathrm{S}}(X_\perp)$ is an ω-algebraic, consistently complete cpo and draw a few examples.

Clearly $\cup: \mathcal{P}_{\mathrm{S}}(X_\perp)^2 \to \mathcal{P}_{\mathrm{S}}(X_\perp)$ is again well-defined and continuous. It is left to the reader as of **Exercise 17** to work out an analogue to Fact 2 thereby obtaining a third semantics for the simple imperative language.

Predicate Transformers

The Smyth order is strongly linked to Dijkstra's 'weakest precondition' predicate transformers.

Definition 3. Let X and Y be countable sets. A *predicate transformer* is any $p: \mathcal{P}(Y) \to \mathcal{P}(X)$ which is continuous, strict and multiplicative. (The last condition means that for any $A, B \subseteq Y$ we have $p(A \cap B) = p(A) \cap p(B)$.)

We develop the theory by means of some exercises.

Exercise 18. Suppose $m: X \to \mathcal{P}_{\mathrm{S}}(Y_\perp)$. Define the *weakest precondition* function by putting for any $B \subseteq Y$

$$wp(m, B) = \{a \in X \mid m(a) \subseteq B\}.$$

Show that $\Omega(m) \stackrel{\text{def}}{=} \lambda B.wp(m, B)$ is a predicate transformer. Show too that Ω is monotonic and that the following properties hold:

$$\begin{aligned}\Omega(\{\cdot\}_X) &= id_{\mathcal{P}(X)},\\ \Omega(m; m') &= \Omega(m) \circ \Omega(m'),\\ \Omega(m \cup m') &= \Omega(m) \cap \Omega(m')\end{aligned}$$

(where $m \cup m' \stackrel{\text{def}}{=} \lambda a \in X.m(a) \cup m'(a)$ and for any $p, q: \mathcal{P}(Y) \to \mathcal{P}(X)$, $p \cap q \stackrel{\text{def}}{=} \lambda B \subseteq Y.p(B) \cap q(B)$).

Exercise 19. Let $p: \mathcal{P}(Y) \to \mathcal{P}(X)$ be a predicate transformer. Show that p is a *stable* function in the sense of Chapter 6 Exercise 1 p. 69. That is show that if $a \in p(Y)$ then there is a finite set $M(p, a)$ such that

$$\forall B \subseteq Y.a \in p(B) \quad \text{iff} \quad M(p, a) \subseteq B.$$

Show too that p is *completely multiplicative* in that if $\langle B_\lambda \rangle_{\lambda \in \Lambda}$ is a collection of subsets of Y then

$$p(\bigcap_\lambda B_\lambda) = \bigcap_\lambda p(B_\lambda).$$

Exercise 20. Let $ST(X, Y)$ be $X \to \mathcal{P}_{\mathrm{S}}(Y)$ with the pointwise ordering and let $PT(X, Y)$ be the set of predicate transformers $p: \mathcal{P}(Y) \to \mathcal{P}(X)$ with the pointwise ordering ($p \sqsubseteq q$ iff $\forall B \subseteq Y.p(B) \subseteq q(B)$). Show that

$$\Omega: ST(X, Y) \to PT(X, Y)$$

is even an *isomorphism* of partial orders. [*Hint* Use the previous exercise to find a definition of Ω^{-1}.]

Exercise 21. Define a predicate transformer semantics

$$\mathcal{W}\colon \mathbf{Com} \to PT(\mathbf{S}, \mathbf{S})$$

(remember to prove that $\mathcal{W}[\![c]\!]$ is always a predicate transformer!). Let $\mathcal{C}_{\mathrm{S}}\colon \mathbf{Com} \to ST(\mathbf{S}, \mathbf{S})$ be the evident semantics based on the Smyth Powerdomain. Show that the following diagram commutes.

$$\begin{tikzcd}
\mathbf{Com} \arrow[r, "\mathcal{C}_{\mathrm{S}}"] \arrow[dr, "\mathcal{W}"'] & ST(\mathbf{S}, \mathbf{S}) \arrow[d, "\Omega"] \\
& PT(\mathbf{S}, \mathbf{S})
\end{tikzcd}$$

These exercises demonstrate complete equivalence between the predicate transformer approach and the nondeterministic position approach using the Smyth powerdomain.

Exercise 22. Let $DPT(X, Y)$ be the additive predicate transformers from Y to X (i.e. those ones p such that for all $A, B \subseteq Y$ we have $p(A \cup B) = p(A) \cup p(B)$.

Show that they correspond to the deterministic state transformation functions in the sense that Ω cuts down to an isomorphism

$$(X \to_{\mathrm{P}} Y) \cong DPT(X, Y).$$

Exercise 23. Find a relational view of predicate transformers and Smyth nondeterministic state transformation functions.

References on Predicate Transformers

E. W. Dijkstra. A Discipline of Programming. Prentice-Hall, 1976.

G. D. Plotkin. Dijkstra's Predicate Transformers and Smyth's Powerdomains. Abstract Software Specifications (ed. D. Bjørner), LNCS Vol. 86, 1980.

Note the reversal of direction in the above between state transformation functions and predicate transformers. What we have is a *duality* between the category $\mathcal{ST}$ of state transformation functions and the category $\mathcal{PT}$ of predicate transformers. (A *duality* of categories $\mathcal{K}$ and $\mathcal{L}$ is an isomorphism $\mathcal{K}^{\mathrm{op}} = \mathcal{L}$.)

Note that Exercises 4, 5, 6 give a duality theory for the relational case, there seems not to be such a theory for the Egli-Milner ordering.

General Powerdomains

We begin by generalising the relational powerdomain to an arbitrary ω-algebraic cpo $\mathbf{D}$. The easy mathematics involved will make that needed for the other two cases more comprehensible. More importantly we will quickly be in a position to demonstrate the techniques made available by powerdomain for the denotational semantics of parallelism. As in the case of a simple nondeterministic language the form of the semantics remains unchanged for the other semantics; only the mathematical properties of the resulting models will differ.

How to Order Subsets of D

Let $\precsim$ be the desired preorder (with associated equivalence $\simeq$); the intuition is that $X \precsim Y$ means that anything X can do Y can do better, and we look for various formalisations of this idea. We will restrict attention to *nonempty* X and Y. First we generalise a definition of Exercise 15.

Definition 4. For X, Y nonempty subsets of $\mathbf{D}$,

$$X \sqsubseteq_{\mathrm{R}} Y \quad \text{iff} \quad \forall x \in X. \exists y \in Y. x \sqsubseteq y$$

and let $=_{\mathrm{R}}$ be the associated equivalence.

Note that $X \subseteq Y$ implies $X \sqsubseteq_{\mathrm{R}} Y$. From the above discussion we would expect that

$$X \sqsubseteq_{\mathrm{R}} Y \quad \text{implies} \quad X \underset{\sim}{\sqsubset} Y.$$

Exercise 24. Show that if $f\colon \mathbf{D} \to \mathbf{E}$ is continuous and if for two nonempty subsets X and Y of $\mathbf{D}$ we have $X \sqsubseteq_{\mathrm{R}} Y$ then $f(X) \sqsubseteq_{\mathrm{R}} f(Y)$.

It may now seem natural that $\underset{\sim}{\sqsubset}$ should be preserved by continuous functions and since it should be $\sqsubseteq_{\mathrm{R}}$ on discrete cpos we define for all nonempty $X, Y \subseteq \mathbf{D}$

$$X \underset{\sim}{\sqsubset}_{\mathrm{R}} Y \quad \text{iff} \quad \forall f\colon \mathbf{D} \to \mathbb{O}. f(X) \sqsubseteq_{\mathrm{R}} f(Y)$$

(and let $\simeq_{\mathrm{R}}$ be the associated equivalence) and expect that

$$X \underset{\sim}{\sqsubset} Y \quad \text{implies} \quad X \underset{\sim}{\sqsubset}_{\mathrm{R}} Y.$$

Thinking of $f\colon \mathbf{D} \to \mathbb{O}$ as an *experiment* or *detectable property* of $\mathbf{D}$, one can understand $X \underset{\sim}{\sqsubset}_{\mathrm{R}} Y$ as meaning that any experiment which can succeed on X can also succeed on Y.

Exercise 25. Show that for any continuous $f\colon \mathbf{D} \to \mathcal{P}(Z)$ with Z countable we have: $X \underset{\sim}{\sqsubset}_{\mathrm{R}} Y$ implies $\bigcup f(X) \subseteq \bigcup f(Y)$.

Thus there is no loss in considering only $\mathbb{O}$.

Now $\underset{\sim}{\sqsubset}$ cannot be $\sqsubseteq_{\mathrm{R}}$ if union is to work as a continuous function; for let X be a set containing an increasing sequence $x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \cdots$ but not containing any upper bound of $x = \bigsqcup_n x_n$. Then $X \neq_{\mathrm{R}} X \cup \{x\}$ but $X \cup \{x\} = X \cup \{\bigsqcup_n x_n\} \simeq \bigsqcup_n (X \cup \{x_n\})$ (by assumption) $= X$. *We therefore take* $\underset{\sim}{\sqsubset}$ *equal to* $\underset{\sim}{\sqsubset}_{\mathrm{R}}$. (The reader can adapt the argument given below to show that $\underset{\sim}{\sqsubset}$ should be $\underset{\sim}{\sqsubset}_{\mathrm{R}}$ for finitely generable sets.) The next theorem gives several characterisations of $\underset{\sim}{\sqsubset}_{\mathrm{R}}$ to increase the reader's confidence in the naturality of the relation.

Theorem 1. *Consider the following conditions on nonempty subsets X and Y of $\mathbf{D}$.*

(1) $X \sqsubseteq_{\mathrm{R}} Y$,
(2) $X \underset{\sim}{\sqsubset}_{\mathrm{R}} Y$,
(3) $\forall x \in X. \forall b \in B_{\mathbf{D}}. (b \sqsubseteq x \Rightarrow \exists y \in Y. b \sqsubseteq y)$,
(4) $\forall A \subseteq \mathbf{D}. (A \text{ finite} \wedge A \sqsubseteq_{\mathrm{R}} X) \Rightarrow A \sqsubseteq_{\mathrm{R}} Y$.

Then (1) $\Rightarrow$ (2) $\Leftrightarrow$ (3) $\Leftrightarrow$ (4) and if $X \subseteq B_{\mathbf{D}}$ or Y is finite then (1) $\Leftrightarrow$ (2).

Proof. (1) $\Rightarrow$ (2). Already known.

(2) $\Rightarrow$ (3). Suppose $b \sqsubseteq x \in X$. Then $\top \in (b \Rightarrow \top)(X) \sqsubseteq_{\mathrm{R}} (b \Rightarrow \top)(Y)$ and so $\top \in (b \Rightarrow \top)(Y)$ and so $b \sqsubseteq$ some y in Y.

(3) $\Rightarrow$ (4). Trivial.

(4) $\Rightarrow$ (2). Left as an exercise.

(1) $\Leftrightarrow$ (2) when $X \subseteq B_{\mathbf{D}}$ or Y is finite. Immediate from (3). ∎

So for finite nonempty sets we have $X \sqsubseteq_{\mathrm{R}} Y \equiv X \underset{\sim}{\sqsubset}_{\mathrm{R}} Y$ and condition (3) show how close $\underset{\sim}{\sqsubset}_{\mathrm{R}}$ is to being $\sqsubseteq_{\mathrm{R}}$ in general and (4) says any finite information about X's possibilities is also true of Y.

There is no need in the relational case to consider any restriction on the sets allowed into the powerdomain; all are "computationally feasible".

Definition 5. For any nonempty $X \subseteq \mathbf{D}$ put $X^\circ \overset{\text{def}}{=} \{b \in B_{\mathbf{D}} \mid \exists x \in X. b \sqsubseteq x\}$. (Note that X° is an *ideal* in $B_{\mathbf{D}}$.)

Now Theorem 1(3) shows that $X \simeq_{\mathrm{R}} X^\circ$. Let $X^\circ = \{b_{(1)}, b_{(2)}, \ldots\}$. Let $P_{(n)}$ be a program whose behaviour is $b_{(n)}$, if possible. (For example when $\mathbf{D} = \mathbf{Tapes}$ $P_{(n)}$ could be **print** $b_{(n)}$.) The set of computations of

$$\mathbf{letrec}\ f(n) = P_{(n)}\ \mathbf{or}\ f(n+1)\ \mathbf{in}\ f(0)$$

is X° since the possible nontermination of $f(0)$, $\perp$ is already accounted for in X°.

So we can take the powerdomain to be $\mathcal{P}_{\mathrm{R}}(\mathbf{D}) \overset{\text{def}}{=} \langle \mathcal{P}(\mathbf{D}), \underset{\sim}{\sqsubset}_{\mathrm{R}} \rangle / \sqsubseteq_{\mathrm{R}}$ where $\mathcal{P}(\mathbf{D})$ is the set of all nonempty subsets of $\mathbf{D}$. But a clearer picture is easily obtained.

Definition 6. For any $X \subseteq \mathbf{D}$ put $Cl_{\mathrm{R}}(X) \stackrel{\mathrm{def}}{=} \{x \mid \forall b \sqsubseteq x. \exists y \in X. b \sqsubseteq y\}$ $(= \{x \mid B_x \subseteq X^\circ\})$.

Exercise 26. Show that Cl_{R} is a *topological closure* operator. That is

(1) $Cl_{\mathrm{R}}(X) \supseteq X$,
(2) $Cl_{\mathrm{R}}(Cl_{\mathrm{R}}(X)) = Cl_{\mathrm{R}}(X)$,
(3) $Cl_{\mathrm{R}}(X \cup Y) = Cl_{\mathrm{R}}(X) \cup Cl_{\mathrm{R}}(Y)$.

In fact $Cl_{\mathrm{R}}(X) = X$ iff X is closed in the Scott topology of $\mathbf{D}$.

Lemma 1. *For nonempty subsets X and Y of $\mathbf{D}$,*

(1) $X \simeq Cl_{\mathrm{R}}(X)$,
(2) $X \simeq Y$ iff $Cl_{\mathrm{R}}(X) = Cl_{\mathrm{R}}(Y)$,
(3) $X \sqsubseteq_{\mathrm{R}} Y$ iff $Cl_{\mathrm{R}}(X) \subseteq Cl_{\mathrm{R}}(Y)$.

Proof. An easy **Exercise 27.** ∎

Lemma 2. *The relational powerdomain $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is $\langle$Closed nonempty subsets of $\mathbf{D}, \subseteq\rangle$. Further $\mathcal{P}(\mathbf{D})$ is an ω-algebraic complete lattice: $\bot = (\bot_{\mathbf{D}})$ and $\bigsqcup \chi = Cl_{\mathrm{R}}(\bigsqcup_{X \in \chi} X)$ (and $X \sqcup Y = X \cup Y$) and the finite elements are those of the form $Cl_{\mathrm{R}}(A) = A^\circ$ where A is a finite nonempty subsets of $B_{\mathbf{D}}$.*

Proof. First the maps $\mathcal{P}(\mathbf{D}) \underset{\delta}{\overset{\theta}{\rightleftharpoons}} \langle$Closed nonempty subsets of $\mathbf{D}, \sqsubseteq\rangle$ defined by $\theta(X{\downarrow}) = Cl_{\mathrm{R}}(X)$, $\delta(X) = X{\downarrow}$ are well-defined using Theorem 1(2) and monotonic using 1(3). Therefore using 1(1) they are mutual inverses, showing they are isomorphisms. The rest is easy **Exercise 28** (use $X \simeq X^\circ$). ∎

Clearly $\cup$ is continuous (since it is $\sqcup$); also we have an "empty set": $\emptyset_{\mathrm{R}} \stackrel{\mathrm{def}}{=} \{\bot\}$ as $\forall X \in \mathcal{P}_{\mathrm{R}}(\mathbf{D}). \{\bot\} \subseteq X$; also we can define a singleton function $\{\cdot\}_{\mathrm{R}} : \mathbf{D} \to \mathcal{P}_{\mathrm{R}}(\mathbf{D})$ by

$$\{d\}_{\mathrm{R}} \stackrel{\mathrm{def}}{=} \{d\}{\downarrow} \qquad (\text{corresponding to } \{x \in \mathbf{D} \mid x \sqsubseteq d\} = Cl_{\mathrm{R}}(\{d\}))$$

and will often drop the suffix below.

Exercise 29. This exercise develops $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ in terms of the basis. Show that for any nonempty sets X, Y:

(1) $X \simeq_{\mathrm{R}} X^\circ$,
(2) $X \simeq_{\mathrm{R}} Y$ iff $X^\circ = Y^\circ$,
(3) $X \sqsubseteq_{\mathrm{R}} Y$ iff $X^\circ \subseteq Y^\circ$.

Now show that $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is (isomorphic to) $\langle$Nonempty ideals of $B_{\mathbf{D}}, \subseteq\rangle$ with $\bot = \{\bot_{\mathbf{D}}\}$, $\bigsqcup \chi = \bigcup \chi$ and finite elements A° with A a finite nonempty subset of $B_{\mathbf{D}}$.

Now we want to develop a suitable notion of function extension. In fact we can characterise $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ as the *free ω-complete cpo over* $\mathbf{D}$ (ω-complete cpo being one with lubs of all countable sets).

Theorem 2. *For any ω-complete cpo $\mathbf{L}$ and (strict) continuous $f : \mathbf{D} \to_{(\bot)} \mathbf{L}$ there is a unique (strict) additive and continuous $f^\dagger : \mathcal{P}_{\mathrm{R}}(\mathbf{D}) \to_{(\mathrm{S})\mathrm{A}} \mathbf{L}$ such that the following diagram commutes:*

$$\begin{tikzcd}
\mathbf{D} \arrow[d, "\{\cdot\}"'] \arrow[dr, "f"] & \\
\mathcal{P}_{\mathrm{R}}(\mathbf{D}) \arrow[r, "f^\dagger"'] & \mathbf{L}
\end{tikzcd}$$

Further $(\cdot)^\dagger : \mathbf{D} \to_{(\bot)} \mathbf{L} \cong \mathcal{P}_{\mathrm{R}}(\mathbf{D}) \to_{(\mathrm{S})\mathrm{A}} \mathbf{L}$ is an isomorphism of partial orders.

Proof. Let $f : \mathbf{D} \to \mathbf{L}$ be continuous and supposing $g : \mathcal{P}_{\mathrm{R}}(\mathbf{D}) \to \mathbf{L}$ is continuous, additive and makes the diagram commute. Then using the ideal characterisation of $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ we have

$$\begin{aligned}
g(X) &= g\Big(\bigcup_{b \in X} \{b\}\Big) \\
&= \bigsqcup_{b \in X} g(\{b\}) && \text{as } X \text{ is nonempty and countable and } g \text{ is continuous and additive} \\
&= \bigsqcup_{b \in X} f(b).
\end{aligned}$$

So g is unique and is strict if f is. Thus defined g is clearly additive and continuous. To see the diagram commutes, we calculate:

$$\begin{aligned} g(\{x\}) &= g(\{b \in B_{\mathbf{D}} \mid b \sqsubseteq x\}) \\ &= \bigsqcup_{b \sqsubseteq x} f(b) \\ &= f(x) && \text{as } f \text{ is continuous.} \end{aligned}$$

Clearly $(\cdot)^{\dagger}$ has inverse $\lambda g. g \circ \{\cdot\}$ and both are monotonic showing they are isomorphisms. ∎

For ω-algebraic $\mathbf{D}$, $\mathbf{E}$ and $\mathbf{F}$ we define the composition of non-deterministic functions as usual: if $f: \mathbf{D} \to \mathcal{P}_{\mathrm{R}}(\mathbf{E})$ and $g: \mathbf{E} \to \mathcal{P}_{\mathrm{R}}(\mathbf{F})$ then $f; g = g^{\dagger} \circ f$. Of course composition is associative continuous and the unit is the singleton function.

We can now note that $\mathcal{P}_{\mathrm{R}}$ is (easily considered as) a locally ω-continuous functor over ω-$\mathcal{ALG}$. For any $f: \mathbf{D} \to \mathbf{E}$ let $\mathcal{P}_{\mathrm{R}}(f): \mathcal{P}_{\mathrm{R}}(\mathbf{D}) \to \mathcal{P}_{\mathrm{R}}(\mathbf{E})$ be the unique additive and continuous function such that

$$\begin{tikzcd} \mathbf{D} \arrow[r, "f"] \arrow[d, "\{\cdot\}"'] & \mathbf{E} \arrow[d, "\{\cdot\}"] \\ \mathcal{P}_{\mathrm{R}}(\mathbf{D}) \arrow[r, "\mathcal{P}_{\mathrm{R}}(f)"'] & \mathcal{P}_{\mathrm{R}}(\mathbf{E}) \end{tikzcd}$$

commutes (by Theorem 2).

Exercise 30. Prove that $\mathcal{P}_{\mathrm{R}}$ is a locally continuous functor which preserves strictness. [Naturally this is all immediate from Theorem 2.]

Hence we can solve domain equations in ω-$\mathcal{ALG}$ provided we do not have any function spaces; if we use the consistently complete ω-algebraic cpos we can also allow ourselves function-spaces as well, for since $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is always a complete lattice it cuts down to a functor on the latter category. This concludes the mathematics we need. A few exercises follow.

Exercise 31. Show that $\mathbf{D}$ is a *closure* of $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ when $\mathbf{D}$ is itself an ω-complete cpo.

Exercise 32. Let $\mathbf{D}$ and $\mathbf{E}$ be ω-algebraic cpos. A *relation* between $\mathbf{D}$ and $\mathbf{E}$ is any $R \subseteq B_{\mathbf{D}} \times B_{\mathbf{E}}$ such that

(1) $\forall a \in B_{\mathbf{D}}. aR\bot$,
(2) $\forall a, a' \in B_{\mathbf{D}}. \forall b, b' \in B_{\mathbf{E}}. a \sqsupseteq a' R b' \sqsupseteq b \Rightarrow aRb$.

The collection of such relations is written as $Rel(\mathbf{D}, \mathbf{E})$. Show that $\mathbf{D} \to \mathcal{P}_{\mathrm{R}}(\mathbf{E}) \cong \langle Rel(\mathbf{D}, \mathbf{E}), \sqsubseteq \rangle$. How should the composition of relations be defined? What is the identity relation for this composition? Reformulate this exercise in terms of relations $R \subseteq \mathbf{D} \times \mathbf{E}$.

Exercise 33 Duality. Show that $(\mathbf{D} \to \mathcal{P}_{\mathrm{R}}(\mathbf{E})) \cong \mathcal{O}(\mathbf{E}) \to_{\mathrm{STA}} \mathcal{O}(\mathbf{D})$ the partial order of strict, top-preserving, additive, continuous functions from $\mathcal{O}(\mathbf{E})$ to $\mathcal{O}(\mathbf{D})$ (here $\mathcal{O}(\mathbf{D}), \mathcal{O}(\mathbf{E})$ are the ω-algebraic complete lattices of the open subsets of $\mathbf{D}$, resp. $\mathbf{E}$ ordered by subset (recall $\mathcal{O}(\mathbf{D}) \cong \mathbf{D} \to \mathbb{O}$ etc.). The isomorphism is Λ where

$$\Lambda(m)(V) = \{d \in \mathbf{D} \mid m(d) \cap V \neq \emptyset\}.$$

Show that the *strict* functions from $\mathbf{D}$ to $\mathcal{P}_{\mathrm{R}}(\mathbf{E})$ correspond under Λ to those g which *reflect* $\top$ (i.e. for all open V, $g(V) = \mathbf{D}$ implies $V = \mathbf{E}$).

Establish the following equalities (the first two showing that Λ is a duality of categories):

(1) $\Lambda(\{\cdot\}) = id$,
(2) $\Lambda(f; g) = \Lambda(f) \circ \Lambda(g)$,
(3) $\Lambda(f \cup g) = \Lambda(f) \cup \Lambda(g)$ (with the evident pointwise definitions).

For any $m: \mathbf{D} \to \mathcal{P}_{\mathrm{R}}(\mathbf{E})$ and Scott-closed $B \subseteq \mathbf{E}$ define the *weakest liberal precondition* $wlp(m, B)$ by:

$$wlp(m, B) = \{d \in \mathbf{D} \mid m(d) \subseteq B\}.$$

How does *wlp* relate to Λ? Conclude that $\mathbf{D}\to\mathcal{P}_{\mathrm{R}}(\mathbf{E})$ is isomorphic to the collection of strict $\perp$-reflecting, $\top$-preserving, multiplicative, dual continuous maps from $\mathcal{C}(\mathbf{E})$ to $\mathcal{C}(\mathbf{D})$ (where $\mathcal{C}(\mathbf{D})$, $\mathcal{C}(\mathbf{E})$ are the partial orders of the Scott-closed subsets of $\mathbf{D}$ resp. $\mathbf{E}$ partially ordered by subset). Specialise this to a correspondence between $\mathbf{D}\to_{\perp}\mathcal{P}_{\mathrm{R}}(\mathbf{E})$ and certain maps from $\mathcal{P}_{\mathrm{R}}(\mathbf{E})$ to $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$.

Exercise 34. Extend Exercise 8 to general relational powerdomains.

Exercise 35. Let $\mathbf{P}\subseteq\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ be ω-inductive. Show the following rule is valid:

$$\frac{\forall d\in\mathbf{D}.\mathbf{P}(\{d\}) \quad \forall X,Y\in\mathcal{P}_{\mathrm{R}}(\mathbf{D}).\mathbf{P}(X)\wedge\mathbf{P}(Y)\supset\mathbf{P}(X\cup Y)}{\forall X.\mathbf{P}(X)}.$$

Exercise 36. Let $\mathbf{P}\subseteq\mathbf{D}$ be ω-inductive. Show that an ω-inductive $\mathbf{Q}\subseteq\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ can be defined by: $\mathbf{Q}(X)$ iff $X\subseteq\mathbf{P}$. Show that $\mathbf{Q}(X)$ iff $\exists d.X=Cl_{\mathrm{R}}(\{d\})$ is ω-inductive. What about $\exists d,d'.X=Cl_{\mathrm{R}}(\{d,d'\})$?

Exercise 37. Show that if $\mathbf{D}$ is effectively given so is $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ and the index of $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is effectively obtainable from that of $\mathbf{D}$. Show that $\cup\colon\mathcal{P}_{\mathrm{R}}(\mathbf{D})^2\to\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is computable. Show that $\{\cdot\}\colon\mathbf{D}\to\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is computable. Show that if $f\colon\mathbf{D}\to\mathcal{P}_{\mathrm{R}}(\mathbf{E})$ is computable (where $\mathbf{E}$ is also effectively given) then $f^{\dagger}\colon\mathcal{P}_{\mathrm{R}}(\mathbf{D})\to\mathcal{P}_{\mathrm{R}}(\mathbf{E})$ is computable, an index for it being effectively obtainable from one for f. Conclude that $\mathcal{P}_{\mathrm{R}}$ is computable.

Denotational Semantics

We consider languages where there are several processes sharing a common store. Later in the chapter we shall have a look at languages where processes communicate with each other along logical channels (lines, through ports, by broadcasting, etc.) and each has its own local store. A basic language is supplied by the grammar:

$$c ::= a \mid \mathbf{skip} \mid c;c \mid \mathbf{if}\ b\ \mathbf{then}\ c\ \mathbf{else}\ c \mid \mathbf{while}\ b\ \mathbf{do}\ c \mid c\|c$$

where atomic commands and Boolean expressions are as usual and we shall feel free to give specific examples without formal definition. The important construction $c_1\|c_2$ conceived of as the concurrent execution of c_1 and c_2. So $c_1\|\cdots\|c_n$ corresponds to Dijkstra's **parbegin** $c_1;\cdots;c_n$ **parend**.

When there is no interference between concurrent processes (perhaps achieved by additional syntactic constraints) there is a legitimate semantics of type $\mathcal{C}\colon\mathbf{Com}\to(\mathbf{S}\to\mathcal{P}_{\mathrm{R}}(\mathbf{S}_{\perp}))$ with crucial clause:

$$\mathcal{C}[\![c_1\|c_2]\!]=(\mathcal{C}[\![c_1]\!];\mathcal{C}[\![c_2]\!])\cup(\mathcal{C}[\![c_2]\!];\mathcal{C}[\![c_1]\!]).$$

But this is not valid when there is interference. For example take $\mathbf{S}=\mathbf{N}$, $c_1=\mathbf{skip}$, $c_2=(x:=1)$, $c_3=(c:=x+1;\ x:=x-1)$. Then $\mathcal{C}[\![c_1\|c_2]\!](2)$ should be $\{1\}$ whereas (assuming programs interruptible between but not during assignments) $\mathcal{C}[\![c_3\|c_2]\!](2)$ should be $\{0,1\}$. But these are different, $\mathcal{C}[\![c_1]\!]$ and $\mathcal{C}[\![c_3]\!]$ should be the same and so there can be *no* semantics of the form

$$\mathcal{C}[\![c_1\|c_2]\!]=\varphi(\mathcal{C}[\![c_1]\!],\mathcal{C}[\![c_2]\!]).$$

To get over this problem we have to account for the phenomenon of interruption. Stating c in σ we can reach in interruption point with an intermediate state σ' and some more program to execute when control returns after the interruption. Of course we can also terminate with a final state and since c itself could be of the form $c_1\|c_2$ several of each of these possibilities could occur. Let r be the meaning of c. Then $r(\sigma)$ should be a set of things either states σ' of pairs $\langle\sigma',r'\rangle$ where σ' is the interruption state and r' is the meaning of the program remaining to execute. Thus we introduce $\mathbf{R}$ as a cpo of *resumption* given by the domain equations

$$\mathbf{R}\cong\mathbf{S}_{\perp}\to_{\perp}\mathcal{P}_{\mathrm{R}}(\mathbf{S}_{\perp}+(\mathbf{S}_{\perp}\otimes\mathbf{R}_{\perp})).$$

In fact this idea (and variations) is all that we need for the semantics of concurrency. It was introduced in a complicated form by Milner in

A. J. R. G. Milner. Processes: A Mathematical Model of Computing Agents, Logic Colloquium '73. North Holland.

and independently by Bekic in

H. Bekic. Towards a Mathematical Theory of Processes. Technical Report TR25.125, IBM Laboratory Vienna, 1971.

Powerdomains were not available in that paper (as a substitute for $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ one used $(\mathbf{O}\to\mathbf{D})$ where $\mathbf{O}$ is a suitable cpo of *oracles* e.g. **Tapes** or $\mathbf{N}^{\omega}$). In

G. D. Plotkin. A Powerdomain Construction, SIAM J. Comput. Vol. 5, No. 3, 1976

the present form was introduced and it was applied to a language for communicating processes in

G. Milne and A. J. R. G. Milner. Concurrent Processes and Their Syntax. JACM 1977.

Every resumption can be *flattened* to produce its corresponding state transformation. To define this we introduce a very handy notation which makes this and other definitions quite transparent. Let e_1 be an expression of type $\mathcal{P}_{\mathrm{R}}(\mathbf{S}_\perp + (\mathbf{S}_\perp \otimes \mathbf{R}_\perp))$ and let $e_2[\sigma']$ be an expression of type $\mathbf{D}$ (possibly with an occurrence of σ', a variable of type $\mathbf{S}_\perp$) and let $e_3[\sigma', r']$ be another expression of type $\mathbf{D}$ (possibly with free occurrences of variables σ', r' of types $\mathbf{S}_\perp$ and $\mathbf{R}$ respectively). Then

$$\begin{aligned}&\textbf{cases } e_1\\ &\quad \textbf{first } \sigma'.e_2[\sigma']\\ &\quad \textbf{second } \sigma', r'.e_3[\sigma', r']\end{aligned}$$

is an expression of type $\mathbf{D}$ which means the same as:

$$[\lambda_\perp \sigma'.e_2[\sigma'], \lambda_\perp \sigma', x.\mathit{lift}(\lambda r'.e_3[\sigma', r'])(x)]^\dagger(e_1).$$

Now we may recursively define a flattening combinator $|\cdot|: \mathbf{R} \to (\mathbf{S}_\perp \to \mathcal{P}(\mathbf{S}))$ by

$$\begin{aligned}|r|(\sigma) = &\textbf{cases } r(\sigma)\\ &\quad \textbf{first } \sigma'.\{\sigma'\}\\ &\quad \textbf{second } \sigma', r'.|r'|(\sigma').\end{aligned}$$

Exercise 38. Note that the cases construction is strict and linear in its first argument. Establish these two identities (where we have omitted some injection functions) where $\sigma' \in \mathbf{S}$, $r' \in \mathbf{R}$,

(1) $\textbf{cases } \{\mathit{inl}(\sigma')\} \textbf{ first } \sigma'.e_2 \textbf{ second } \sigma, r'.e_3 = e_2$,
(2) $\textbf{cases } \{\mathit{inr}(\langle \sigma', \mathit{up}(r')\rangle)\} \textbf{ first } \sigma'.e_2 \textbf{ second } \sigma', r'.e_3 = e_3$.

Now only one or two more combinators are needed.

Composition. We define $*: \mathbf{R}^2 \to \mathbf{R}$ by

$$\begin{aligned}r_1 * r_2(\sigma) = &\textbf{cases } r_1(\sigma)\\ &\quad \textbf{first } \sigma'.\{\mathit{inr}(\sigma' \otimes \mathit{up}(r_2))\}\\ &\quad \textbf{second } \sigma', r_1'.\{\mathit{inr}(\sigma' \otimes \mathit{up}(r_1' * r_2))\}.\end{aligned}$$

Parallelism. We define $\|: \mathbf{R}^2 \to \mathbf{R}$ by

$$\begin{aligned}r_1 \| r_2(\sigma) = &\textbf{cases } r_1(\sigma)\\ &\quad \textbf{first } \sigma'.\{\mathit{inr}(\sigma' \otimes \mathit{up}(r_2))\}\\ &\quad \textbf{second } \sigma', r_1'.\{\mathit{inr}(\sigma' \otimes \mathit{up}(r_1' \| r_2))\}\\ &\qquad \cup\\ &\textbf{cases } r_2(\sigma)\\ &\quad \textbf{first } \sigma'.\{\mathit{inr}(\sigma' \otimes \mathit{up}(r_1))\}\\ &\quad \textbf{second } \sigma', r_2'.\{\mathit{inr}(\sigma' \otimes \mathit{up}(r_1 \| r_2'))\}.\end{aligned}$$

Exercise 39. This parallel operator is one where "$r_1 \| r_2$ terminates when r_1 and r_2 do"; write one which terminates when one does.

Exercise 40. Establish whether $*$ is associative with identity $\lambda_\perp \sigma.\{\mathit{inl}(\sigma)\}$ and whether $\|$ is associative and commutative.

<u>Semantics</u>. $\mathcal{R}: \mathbf{Com} \to \mathbf{R}$ is defined by

$\mathcal{R}[\![a]\!] = \lambda_\perp \sigma.\{\mathit{inl}(\mathcal{A}[\![a]\!](\sigma))\}$;
$\mathcal{R}[\![\mathbf{skip}]\!] = \lambda_\perp \sigma.\{\mathit{inl}(\sigma)\}$;
$\mathcal{R}[\![c_1; c_2]\!] = \mathcal{R}[\![c_1]\!] * \mathcal{R}[\![c_2]\!]$;
$\mathcal{R}[\![\mathbf{if}\ b\ \mathbf{then}\ c_1\ \mathbf{else}\ c_2]\!] = \lambda_\perp \sigma.\mathbf{if}\ \mathcal{B}[\![b]\!](\sigma)\ \mathbf{then}\ \{\mathit{inr}(\sigma \otimes \mathit{up}(\mathcal{R}[\![c_1]\!]))\}\ \mathbf{else}\ \{\mathit{inr}(\sigma \otimes \mathit{up}(\mathcal{R}[\![c_2]\!]))\}$;
$\mathcal{R}[\![\mathbf{while}\ b\ \mathbf{do}\ c]\!] = \mu r.\lambda_\perp \sigma.\mathbf{if}\ \mathcal{B}[\![b]\!](\sigma)\ \mathbf{then}\ \{\mathit{inr}(\sigma \otimes \mathit{up}(\mathcal{R}[\![c]\!] * r))\}\ \mathbf{else}\ \{\mathit{inr}(\sigma \otimes \mathit{up}(\lambda_\perp \sigma.\{\mathit{inl}(\sigma)\}))\}$;
$\mathcal{R}[\![c_1 \| c_2]\!] = \mathcal{R}[\![c_1]\!] \| \mathcal{R}[\![c_2]\!]$.

Exercise 41. In the semantics of conditional and while-loops there is an interruption point between the evaluation of the Boolean expression and the body. Give a semantics where no such point exists (essentially the so-called *test and set* idea).

Exercise 42. Give a semantics of the construct **critical-section**(c) (alternative syntax: $[c]$) which is c but with *no* interruption points.

Exercise 43. Give a semantics of assignment statements where there is an interruption point between the evaluation of the right-hand-side of an assignment and the updating of the variable.

Suppose atomic commands are assignments $x := e$ where the class of expressions is, say, $e ::= 0 \mid 1 \mid e+e \mid e*e \mid \cdots$. Give a semantics where expressions can be interrupted at any point during their evaluation. You may find it useful to use *expression resumptions* defined by

$$\mathbf{E} \cong \mathbf{S}_\perp \to_\perp \mathcal{P}_{\mathrm{R}}((\mathbf{N}_\perp \otimes \mathbf{S}_\perp) + (\mathbf{S}_\perp \otimes \mathbf{E}_\perp)).$$

This exercise can be taken quite far by allowing recursive functions to be defined as in: **letrec** $f(x) = e$ **in** e or commands in expressions as in **begin** c **result** e.

Synchronisation Methods

We consider these via some exercises.

Exercise 44. The *when construct* is $b \Rightarrow c$ (alternative concrete syntax is: **when** b **do** c or **await** $b; c$). The idea is that $b \Rightarrow c$ cannot execute unless b is true. Give this a semantics using as resumptions:

$$\mathbf{R} \cong \mathbf{S}_\perp \to_\perp \mathcal{P}_{\mathrm{R}}(\mathbf{S}_\perp + \mathbf{S}_\perp \otimes \mathbf{R}_\perp)$$

the idea being that $r(\sigma) = \emptyset_{\mathrm{R}}$ corresponds to r's "not being able to execute" i.e. *deadlock*. Your semantics should be such that the first interruption point of $b \Rightarrow c$ is the first interruption point of c. (If it were instead just after b or just after c that would then corresponding to the semantics of $b \Rightarrow$ **skip**$; c$ and $b \Rightarrow [c]$ (equivalently $[b \Rightarrow c]$; give alternative semantics for these two properties and establish the correspondence).

Add the operator $c_1[]c_2$ which can execute just when c_1 **or** c_2 can.

Consider allowing interruption points during the execution of the conditional.

Exercise 45 Semaphores. Let **SId** be a finite set of semaphore identifiers ranged over by v; they will be integer semaphores. We introduce the commands $P(v)$ (alternate syntax; $Wait(v)$, $Grab(v)$) and $V(v)$ (alternate syntax: $Signal(v)$, $Release(v)$). In terms of the **when** construct — see previous exercise — $P(v)$ means $v > 0 \Rightarrow v := v - 1$ and $V(v)$ means $v := v + 1$ where the assignments are uninterruptible.

Give a resumption semantics, using resumptions:

$$\mathbf{R} \cong \mathbf{States} \to_\perp \mathcal{P}_{\mathrm{R}}(\mathbf{States} + \mathbf{States} \otimes \mathbf{R}_\perp)$$

with the same idea as in the previous exercise but where

$$\mathbf{States} = \mathbf{S}_\perp \times (\mathbf{Sid}_\perp \to_\perp \mathbf{N}_\perp \otimes \mathbf{T}).$$

The point of the truth-values is to show whether the semaphore is busy or not. As the separation of identifiers is becoming clumsy, you might want to consider allowing only one set of identifiers but introduce a suitable declaration mechanism for variables and semaphores.

Exercise 46 I/O Devices. Add the construct **print** e where e is an arithmetic expression. As a resumption model you could take:

$$\mathbf{R} \cong \mathbf{S}_\perp \to_\perp \mathcal{P}_{\mathrm{R}}(\mathbf{S}_\perp + \mathbf{N}_\perp \otimes \mathbf{R}_\perp + \mathbf{S}_\perp \otimes \mathbf{R}_\perp).$$

Define a suitable flattening function $|\cdot|: \mathbf{R} \to \mathbf{NS}$ where $\mathbf{NS}$ is the domain of finite sequences of integers, given by the domain equation:

$$\mathbf{NS} \cong \mathbf{N}_\perp + \mathbf{N}_\perp \otimes \mathbf{NS}_\perp.$$

This clearly continues the ideas of Chapter 5, Example 4 allowing infinitely proceeding output and showing the connection between the present resumptions and those considered there. Adapt the ideas of that example to allow a read command. Extend all this to more sophisticated I/O devices. You may find points of contact with the semantics of communicating processes.

Exercise 47 Parallel For-Loops. Invent such a concept and give its semantics.

Exercise 48 Critical Regions. The idea here is to have "structured semaphores". Each variable has attached to it a binary semaphore and the new syntactic construct is **with** x **do** c which means $P(v); c; V(v)$ where v is the binary semaphore, and so there are no explicit semaphore. Give this a semantics. Strictly speaking there should also be a static semantics so that:

(1) There is no use of variables except inside **with** statements. The point of this is to ensure that processes obey the semaphores discipline of not accessing resources until they are owned by the process.

(2) No two parallel processes have common variables with left-hand-side occurrences. This ensures resources are uniquely allocated to processes at any one time (though the allocation can change as **with** statements are entered and exited).

Give such a static semantics (see Mike Gordon's book for hints on how to do that).

Add **when** statements to your language and semantics. This allows the explanation of some well-known constructs. For example

with r **when** b **do** c	means	**with** r **do** $(b \Rightarrow c)$
region r **do**	means	**with** r **do**
begin ... **await** $b; c \ldots$ **end**		$\ldots (b \Rightarrow c) \ldots$

$$\textbf{with } r \textbf{ do } (b_0 \Rightarrow c_0 [] \cdots [] b_n \Rightarrow c_n)$$

Exercise 49. This is a fairly substantial exercise, perhaps more of a mini-project. We think of a *monitor* as a class treated as a resource and you should begin by ignoring the parallel features and just find a semantics of classes.

Syntax.
Ground Types. These are $gt \in \{int, bool\}$.

Types. These are $t \in \textbf{Types} = \{int, bool, \textbf{proc } int, \textbf{proc } bool\}$ for integer or boolean variables and call-by-value procedures with one argument being either integers or booleans.

Monitor Types. These are $mt \in \textbf{MTypes} = \textbf{Id} \rightarrow_{\text{fin}} \textbf{Types} \stackrel{\text{def}}{=} \sum_{X:\text{a finite subset of } \textbf{Id}} X \rightarrow \textbf{Types}$.

Declarations. These are $d \in \textbf{Dec}$ where

$$d ::= \textbf{var } x\colon gt \mid \textbf{proc } x\colon gt; c \mid d \textbf{ within } d \mid d; d \mid d \textbf{ init } c \mid \textbf{monitor } m\colon mt\ d$$

For d **within** d and $d; d$ see Mike Gordon's book pp. 80, 135. The initialisation construct d **init** c is elaborated by elaborating d and then executing c (for initialisation purposes). Private declarations and initialisation are useful within monitor declarations.

Commands.

$$c ::= (\text{usual stuff}) \mid c \| c \mid b \Rightarrow c \mid c \,[]\, c \mid d; c \mid p(e) \mid \textbf{with } m \textbf{ do } c.$$

In the static semantics monitors are treated as resources so that any call of a monitor procedure or reference to a variable must be inside an appropriate **with** statement and also no concurrent access to such monitor resources is permitted.

Communicating Processes

Consider the following where α, β, γ range over a finite set **Lab** of *communication channels*

$$c ::= a \mid \textbf{skip} \mid \alpha?x \mid \alpha!e \mid c; c \mid \textbf{if } b \textbf{ then } c \textbf{ else } c \mid \textbf{while } b \textbf{ do } c \mid c \| c \mid c \backslash \alpha \mid c[\alpha/\beta] \mid c \,[]\, c$$

(there should be some static semantics to ensure concurrent processes do not use common variables). Here $\alpha?x$ means: input a value off α and update x to it and $\alpha!e$ means evaluate e and send the result along α. Let us look for a semantics where we intend *synchronised communication* so that the only communication actions are an uninterruptible sending and then receiving; $c\backslash\alpha$ is c less the capability of communicating on α and $c[\alpha/\beta]$ is c but with the capabilities of communication over α transferred to β. See, for example

C. A. R. Hoare. Communicating Sequential Processes. CACM Vol. 21(8), pp. 666–677, 1978.

A. J. R. G. Milner. A Calculus of Communicating Systems, LNCS Vol. 92, 1980.

H. Ledgard. ADA: An Introduction. Springer-Verlag.

We take

$$\mathbf{R} \cong \mathbf{S}_\perp \to_\perp \mathcal{P}_{\mathrm{R}}(\mathbf{S}_\perp + \mathbf{Lab}_\perp \otimes (\mathbf{N}_\perp \to_\perp \mathbf{S}_\perp \otimes \mathbf{R}_\perp)_\perp + \mathbf{Lab}_\perp \otimes \mathbf{N}_\perp \otimes \mathbf{R}_\perp + \mathbf{S}_\perp \otimes \mathbf{R}_\perp).$$

The Parallel Combinator

$$\begin{array}{l}
r_1 \| r_2(\sigma) = \mathbf{cases}\ r_1(\sigma) \\
\qquad\qquad \mathbf{first}\ \sigma'.\{in_3(\sigma' \otimes up(r_2))\} \\
\qquad\qquad \mathbf{second}\ \alpha, f.\{in_1(\alpha \otimes up(\lambda_\perp n.f(n)_1 \otimes up(f(n)_2 \| r_2)))\} \\
\qquad\qquad \mathbf{third}\ \alpha, n, r_1'.\{in_2(\alpha \otimes n \otimes (up(r_1' \| r_2)))\} \\
\qquad\qquad \mathbf{fourth}\ \sigma', r_1'.\{in_3(\sigma' \otimes up(r_1' \| r_2))\} \\
\qquad\qquad\quad \cup \\
\qquad\quad \mathbf{cases}\ r_1(\sigma) \\
\qquad\qquad \mathbf{first}\ \sigma'.\emptyset \\
\qquad\qquad \mathbf{second}\ \alpha, f. \\
\qquad\qquad\quad \mathbf{cases}\ r_2(\sigma) \\
\qquad\qquad\qquad \mathbf{first}\ \sigma'.\emptyset \\
\qquad\qquad\qquad \mathbf{second}\ \beta, g.\emptyset \\
\qquad\qquad\qquad \mathbf{third}\ \beta, n', r_2'. \\
\qquad\qquad\qquad\quad \mathbf{if}\ \alpha = \beta \\
\qquad\qquad\qquad\qquad \mathbf{then}\ \{in_3(f(n)_1 \otimes up(f(n)_2 \| r_2'))\} \\
\qquad\qquad\qquad\qquad \mathbf{else}\ \emptyset \\
\qquad\qquad\qquad \mathbf{fourth}\ \sigma', r_2'.\emptyset \\
\qquad\qquad \mathbf{third}\ \alpha, n, r_1'. \\
\qquad\qquad\quad \mathbf{cases}.r_2(\sigma) \\
\qquad\qquad\qquad \mathbf{first}\ \sigma'.\emptyset \\
\qquad\qquad\qquad \mathbf{second}\ \beta, f. \\
\qquad\qquad\qquad\quad \mathbf{if}\ \alpha = \beta \\
\qquad\qquad\qquad\qquad \mathbf{then}\ \{in_3(f(n)_1 \otimes up(r_1' \| f(n)_2))\} \\
\qquad\qquad\qquad\qquad \mathbf{else}\ \emptyset \\
\qquad\qquad\qquad \mathbf{third}\ \beta, n', r_2'.\emptyset \\
\qquad\qquad\qquad \mathbf{forth}\ \sigma', r_2'.\emptyset \\
\qquad\qquad \mathbf{fourth}\ \sigma', r_1'.\emptyset \\
\qquad\qquad\quad \cup \\
\qquad\quad \mathbf{cases}\ r_2(\sigma) \\
\qquad\qquad \mathbf{first}\ \sigma'.\{in_3(\sigma' \otimes up(r_1))\} \\
\qquad\qquad \mathbf{second}\ \alpha, f.\{in_1(\alpha \otimes up(\lambda_\perp n.f(n)_1 \otimes up(r_1 \| f(n)_2)))\} \\
\qquad\qquad \mathbf{third}\ \alpha, n, r_2'.\{in_2(\alpha \otimes n \otimes up(r_1 \| r_2'))\} \\
\qquad\qquad \mathbf{fourth}\ \sigma', r_2'.\{in_3(\sigma' \otimes up(r_1 \| r_2'))\}.
\end{array}$$

The notation is now become barbaric!

Exercise 50. Rewrite the above definition, without using the empty set.

The Elision Combinator

$$\begin{array}{l}
r \backslash \alpha(\sigma) = \mathbf{cases}\ r(\sigma) \\
\qquad\qquad \mathbf{first}\ \sigma'.\{in_0(\sigma')\} \\
\qquad\qquad \mathbf{second}\ \beta, f.\ \mathbf{if}\ \alpha = \beta\ \mathbf{then}\ \emptyset\ \mathbf{else}\ \{in_1(\beta \otimes up(\lambda_\perp n.f(n)_1 \otimes up(f(n)_2 \backslash \alpha)))\} \\
\qquad\qquad \mathbf{third}\ \beta, n, r'.\mathbf{if}\ \alpha = \beta\ \mathbf{then}\ \emptyset\ \mathbf{else}\ \{in_2(\beta \otimes n \otimes up(r' \backslash \alpha))\} \\
\qquad\qquad \mathbf{fourth}\ \sigma', r'.\{in_3(\sigma' \otimes up(r' \backslash \alpha))\}.
\end{array}$$

It is here that the empty set is really needed.

The Substitution Combinator

$$
\begin{aligned}
r[\alpha/\beta](\sigma) = &\mathbf{cases}\ r(\sigma)\\
&\quad \mathbf{first}\ \sigma.\{in_0(\sigma')\}\\
&\quad \mathbf{second}\ \gamma, f.\\
&\quad\quad \mathbf{if}\ \gamma = \beta\\
&\quad\quad\quad \mathbf{then}\ \{in_1(\alpha \otimes up(\lambda_\perp n.f(n)_1 \otimes up(f(n)_2[\alpha/\beta])))\}\\
&\quad\quad\quad \mathbf{else}\ \{in_1(\gamma \otimes up(f))\}\\
&\quad \mathbf{third}\ \gamma, n, r'.\\
&\quad\quad \mathbf{if}\ \gamma = \beta\\
&\quad\quad\quad \mathbf{then}\ \{in_2(\alpha \otimes n \otimes up(r'[\alpha/\beta]))\}\\
&\quad\quad\quad \mathbf{else}\ \{in_2(\gamma \otimes n \otimes up(r'[\alpha/\beta]))\}\\
&\quad \mathbf{fourth}\ \sigma, r'.\{in_3(\sigma' \otimes up(r'[\alpha/\beta]))\}.
\end{aligned}
$$

All the semantics $\mathcal{R}\colon \mathbf{Com} \to \mathbf{R}$ is now obvious but for these two clauses (perhaps):

$$
\begin{aligned}
&\mathcal{R}[\![\alpha?x]\!](\sigma) = \{in_1(\alpha \otimes up(\lambda_\perp n.\sigma[x = n] \otimes up(\lambda_\perp \sigma.\{in_0(\sigma)\})))\};\\
&\mathcal{R}[\![\alpha!e]\!](\sigma) = \{in_2(\alpha \otimes \mathcal{E}[\![e]\!](\sigma) \otimes up(\lambda_\perp \sigma.\{in_0(\sigma)\}))\}.
\end{aligned}
$$

Exercise 51. The same language can be used for other communication disciplines. For example the acts of inputting and outputting can be completely independent. Give this idea a semantics using as resumptions

$$\mathbf{R} \cong \mathbf{States}_\perp \to_\perp \mathcal{P}_{\mathrm{R}}(\mathbf{States}_\perp + \mathbf{States}_\perp \otimes \mathbf{R}_\perp)$$

where $\mathbf{States} = \mathbf{S}_\perp \times (\mathbf{Lab}_\perp \to_\perp \mathbf{N}_\perp)$.

Adapt this idea to the case where the channels are actually buffers holding sequences of integers.

Exercise 52. The above language is a mixture of those employed by Hoare and Milner. Give a semantics to Hoare's CSP as in the CACM paper, but (at least at first) without his termination conventions. You may find it helpful to look for a denotational version of

G. D. Plotkin. An Operational Semantics for CSP. IFIP WG 2.2 Working Conference, Garmisch, 1982.

Exercise 53. Give a semantics to Milner's CCS which is an applicative version of the above concurrent language:

$$
\begin{aligned}
t ::= &\mathbf{nil} \mid t + t \mid \alpha?x.t \mid \alpha!e.t \mid \mathbf{if}\ b\ \mathbf{then}\ t\ \mathbf{else}\ t \mid \mathbf{let}\ x\ \mathbf{be}\ e\ \mathbf{in}\ t \mid \mathbf{let}\ \ldots \mid\\
&P \mid \mathbf{let}\ P\ \mathbf{be}\ t\ \mathbf{in}\ t \mid \mathbf{letrec}\ P\ \mathbf{be}\ t\ \mathbf{in}\ t \mid \ldots \mid t|t \mid t\backslash\alpha \mid t[\alpha/\beta].
\end{aligned}
$$

The meaning of something is its set of communication capabilities, e.g. **nil** means $\emptyset$, having none. Give a semantics using:

$$\mathbf{R} \cong \mathcal{P}_{\mathrm{R}}(\mathbf{Lab}_\perp \otimes \mathbf{N}_\perp \otimes \mathbf{R}_\perp + \mathbf{Lab}_\perp \otimes (\mathbf{N}_\perp \to_\perp \mathbf{R})_\perp).$$

Exercise 54. This is a mini-project. Formulate a version of Hewitt's actors language and find a resumption-based semantics.

As an exercise along these lines, take the language of Exercise 51 and think of α a "port in the ether". Then $\alpha!e$ means send the value of e to α through the ether and $\alpha?e$ means take a value from the port α. One way to do this is to use

$$\mathbf{States} = \mathbf{S}_\perp \times (\mathbf{Lab}_\perp \to_\perp \mathcal{M}(\mathbf{N})_\perp)$$

where $\mathcal{M}(\mathbf{N})$ is the set of finite *multisets* of integers.

General Powerdomains: The Plotkin Powerdomain

We now try to construct a powerdomain of an arbitrary ω-algebraic cpo $\mathbf{D}$ generalising the Egli-Milner approach in the discrete case. As a typical example we can take $\mathbf{D} = \mathbf{Tapes}$, the set of finite strings of 0's and 1's under the subsequence ordering. To see this in a computational setting we add a print instruction to our language:

$$c ::= \mathbf{print}\ t$$

(where t ranges over $\{0,1\}$) and desire a semantics such that $\mathcal{C}[\![c]\!](\sigma)$ is the set of tapes that might be output. Here are some example programs:

(5) (**print** 0; **loop**) **or** (**print** 000; **loop**),

(6) (**print** 0; **loop**) **or** (**print** 00; **loop**) **or** (**print** 000; **loop**),

(7) $alarm$:= "off";
(**while** $alarm$ = "off" **do** $x := x + 1$ **or** $alarm$:= "on");
(**while** $x > 0$ **do print** 0; $x := x - 1$);
(**while** $true$ **do print** 1),

(8) (**while** $true$ **do print** 0) **or** (7).

Clearly (from an operational point of view) we expect (5), (6), (7), (8) to output the following sets

$$\begin{aligned}
A_5 &= \{0, 000\} = \{0\} \cup \{000\},\\
A_6 &= \{0, 00, 000\} = \{0\} \cup \{00\} \cup \{000\},\\
A_7 &= \{\bot\} \cup \{0^n 1^\omega \mid n \geq 0\},\\
A_8 &= \{\bot\} \cup \{0^n 1^\omega \mid n \geq 0\} \cup \{0^\omega\}.
\end{aligned}$$

So we may expect that (5) $\neq$ (6) and (7) $\neq$ (8) in the anticipated semantics. *But this is not so*: we will have both (5) = (6) (and A_5 "=" A_6) and (7) = (8) (and A_7 "=" A_8) and we can show: *these identifications cannot be avoided in our present framework of cpos and continuous functions* (under a few reasonable assumptions). The latter identification is particularly upsetting: A_7 is a set of sequences which are guaranteed to contain a 1 if they contain a 0 but A_8 is not such a set.

Investigating these points will lead us to an algebraic characterisation of powerdomains, generalising Facts 1, 2 above. It will be very easy using the completion ideas of Chapter 6 to show that such powerdomains exist. Unfortunately this does not give us much insight since under this presentation the powerdomains are *not* given as collections of sets. In order to do that we have to proceed in analogy with development of the Egli-Milner order considering what subsets we expect and how to order them.

Now what a powerdomain $\mathbf{P}$ of an ω-algebraic cpo $\mathbf{D}$ should supply us with is a binary union function: $\cup: \mathbf{P}^2 \to \mathbf{P}$ and a singleton function $\{\cdot\}: \mathbf{D} \to \mathbf{P}$; what is more this data should be the universal such data in order to get an analogue of Fact 2 and Theorem 2 allowing us (for one thing!) to compose nondeterministic functions $f: \mathbf{D} \to \mathbf{P}$. Let us turn these ideas into definitions.

Definition 7. A *semilattice* is a structure $\langle \mathbf{P}, \cup \rangle$ where $\cup: \mathbf{P}^2 \to \mathbf{P}$ is a binary function satisfying the following three axioms:

Associativity	$\forall a, b, c \in \mathbf{P}. a \cup (b \cup c) = (a \cup b) \cup c.$
Commutativity	$\forall a, b \in \mathbf{P}. a \cup b = b \cup a.$
Absorption	$\forall a \in \mathbf{P}. a \cup a = a.$

A function $f: \mathbf{P} \to \mathbf{Q}$ between semilattice is *linear* iff for all a, b in $\mathbf{P}$ we have $f(a \cup b) = f(a) \cup f(b)$. A structure $\langle \mathbf{P}, \sqsubseteq, \cup \rangle$ is a *continuous semilattice* iff $\langle \mathbf{P}, \sqsubseteq \rangle$ is a cpo, $\langle \mathbf{P}, \cup \rangle$ is a semilattice and $\cup$ is continuous (w.r.t. the ordering $\sqsubseteq$).

Exercise 55. Let $\mathbf{P}$ be a semilattice and define a relation by $x \subseteq y$ iff $x \cup y = y$. Show $\subseteq$ is a partial order with lubs of pairs given by $\cup$; this gives 1-1 correspondence between semilattices and partial orders with lubs of pairs. Assuming $\mathbf{P}$ is a continuous semilattice show that $\subseteq$ is ω-inductive.

Exercise 56. Show that the ω-complete cpos are exactly the continuous semilattices such that for all x, y

$$x \cup y = x \sqcup y$$

and these are the continuous semilattices such that for all x, y

$$x \sqsubseteq x \cup y$$

and these are the same as the continuous semilattices such that for all x, y

$$x \subseteq y \Rightarrow x \sqsubseteq y$$

and these are the same as the continuous semilattices such that for all x, y equivalence: $x \subseteq y \Leftrightarrow x \sqsubseteq y$.

Also the additive, continuous $f\colon \mathbf{D} \to \mathbf{E}$ between ω-complete cpos are just linear, continuous $f\colon \mathbf{D} \to \mathbf{E}$ when $\mathbf{D}$ and $\mathbf{E}$ are viewed as these kinds of semilattice. One can then reinterpret Theorem 2 as saying that $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is just the free continuous semilattice over $\mathbf{D}$ where subset and approximation agree.

So a powerdomain of $\mathbf{D}$ should give us a map $\mathbf{D} \overset{\{\cdot\}}{\to} \mathbf{P}$ with $\mathbf{P}$ a continuous semilattice and we will look for the universal one. Before doing so let us see why the above identifications of sets can be expected to hold. Let $\mathbf{D} \overset{\{\cdot\}}{\to} \langle \mathbf{P}, \cup \rangle$ be as expected with $\mathbf{D} = \mathbf{Tapes}$. Then we expect that the elements in $\mathbf{P}$ corresponding to A_5 and A_6 are $\{0\} \cup \{000\}$ and $\{0\} \cup \{00\} \cup \{000\}$ respectively and calculate:

$$\begin{aligned}
\{0\} \cup \{000\} &= (\{0\} \cup \{0\}) \cup \{000\} \quad \text{(by absorption)} && \\
&\sqsubseteq (\{0\} \cup \{00\}) \cup \{000\} && \text{since } 0 \sqsubseteq 00 \text{ and } \{\cdot\} \text{ is monotonic} \\
&\sqsubseteq (\{0\} \cup \{000\}) \cup \{000\} && \text{same reason} \\
&= \{0\} \cup \{000\} && \text{by associativity and absorption.}
\end{aligned}$$

Thus $\{0\} \cup \{000\} = \{0\} \cup \{00\} \cup \{000\}$ in any reasonable powerdomain (this is what was meant by A_5 "=" A_6).

Turning to the second pair of sets, let A and B be the elements of $\mathbf{P}$ corresponding to A_7 and A_8. We expect that $B = A \cup \{0^\omega\}$ and $A = A \cup \{\bot\} \cup \{0^n 1^\omega\}$ (any n). It follows from the second of these that $A = A \cup \{0^n\}$ (any n) — by a similar argument to the above that $A_5 = A_6$. But then

$$\begin{aligned}
B &= A \cup \{0^\omega\} && \\
&= A \cup \bigsqcup_n \{0^n\} && \{\cdot\} \text{ is continuous} \\
&= \bigsqcup_n (A \cup \{0^n\}) && \text{as union is continuous} \\
&= A. &&
\end{aligned}$$

Thus it seems that these identifications cannot be avoided in the framework of cpos and continuous functions; we hope later to sweeten this bitter pill. In his thesis Lehman suggested replacing partial orders by categories; how best to do this remains an interesting open problem.

Now we give a quick construction of the powerdomain. For any $\mathbf{D}$ put

$$\mathcal{F}(\mathbf{D}) \stackrel{\text{def}}{=} \langle \{A \subseteq B_{\mathbf{D}} \mid A \text{ is finite and nonempty}\}, \sqsubseteq_{\mathrm{EM}} \rangle.$$

That is $\mathcal{F}(\mathbf{D})$ is the collection of finite nonempty sets of finite elements of $\mathbf{D}$ under the Egli-Milner ordering. As such it is a preorder (not, in general, a partial order) with a least element $\{\bot\}$. Following Chapter 6 we put

$$\mathcal{P}(\mathbf{D}) \stackrel{\text{def}}{=} \overline{\mathcal{F}(\mathbf{D})}$$

and define $\mathbf{D} \overset{\{\cdot\}_{\mathrm{P}}}{\to} \mathcal{P}(\mathbf{D})$ by

$$\{d\}_{\mathrm{P}} = \{\{b_1, \ldots, b_n\} \mid n > 0 \text{ and } b_i \in B_{\mathbf{D}} \wedge b_i \sqsubseteq d, \text{ for } i = 1, n\}$$

and define the union function by

$$I \cup_{\mathrm{P}} J = \{A \cup B \mid A \in I, B \in J\}$$

Exercise 57. Verify that $\{\cdot\}_{\mathrm{P}}$ and $\cup_{\mathrm{P}}$ are well-defined and continuous; check that $\{b\}_{\mathrm{P}} = [\{b\}]$ for b in $B_{\mathbf{D}}$ and $[A] \cup_{\mathrm{P}} [B] = [A \cup B]$ for A, B in $\mathcal{F}(\mathbf{D})$.

The simple definition of union makes it clear that $\langle \mathcal{P}(\mathbf{D}), \cup_{\mathrm{P}} \rangle$ is a semilattice; we shall drop the subscripts when clear from the context.

Theorem 3. *$\mathcal{P}(\mathbf{D})$ is an ω-algebraic cpo which with $\cup$ is a continuous semilattice. Further $\{\cdot\}: \mathbf{D} \to \mathcal{P}(\mathbf{D})$ is a (strict) continuous function which is universal in that if $f: \mathbf{D} \to \mathbf{P}$ is any (strict) continuous function to a continuous semilattice there is a unique linear (strict) continuous function $f^\dagger: \mathcal{P}(\mathbf{D}) \to \mathbf{P}$ such that the following diagram commutes*

$$\begin{tikzcd}
\mathbf{D} \arrow[d, "\{\cdot\}_{\mathrm{P}}"'] \arrow[dr, "f"] & \\
\mathcal{P}(\mathbf{D}) \arrow[r, "f^\dagger"'] & \mathbf{P}
\end{tikzcd}$$

Further $\dagger$ is an isomorphism between the continuous semilattice of the (strict) continuous functions from $\mathbf{D}$ to $\mathbf{P}$ and the continuous semilattice of the linear (strict) continuous function from $\mathcal{P}(\mathbf{D})$ to $\mathbf{P}$ (where the unions of the functions are defined pointwise).

Proof. By construction $\mathcal{P}(\mathbf{D}) = \overline{\mathcal{F}(\mathbf{D})}$ is ω-algebraic and we know it is a continuous semilattice. We know the singleton function is continuous; to see it is strict note that $\{\perp_{\mathbf{D}}\} = [\{\perp\}] = [\perp_{\mathcal{F}(\mathbf{D})}] = \perp_{\mathcal{P}(\mathbf{D})}$.

Next suppose $f: \mathbf{D} \to \mathbf{P}$ is a (strict) continuous linear function from $\mathbf{D}$ to a continuous semilattice $\mathbf{P}$. Since $f^\dagger$ is continuous it is enough to show it is determined on the finite elements of $\mathcal{P}(\mathbf{D})$ in order to demonstrate uniqueness. So let us calculate:

$$\begin{aligned}
f^\dagger([\{b_1, \ldots, b_n\}]) &= f^\dagger([\{b_1\}] \cup_{\mathrm{P}} \cdots \cup_{\mathrm{P}} [\{b_n\}]) \\
&= f^\dagger(\{b_1\}_{\mathrm{P}} \cup_{\mathrm{P}} \cdots \cup_{\mathrm{P}} \{b_n\}_{\mathrm{P}}) \\
&= f^\dagger(\{b_1\}_{\mathrm{P}}) \cup \cdots \cup f^\dagger(\{b_n\}_{\mathrm{P}}) && \text{as } f^\dagger \text{ is linear} \\
&= f(b_1) \cup \cdots \cup f(b_n) && \text{as the diagram commutes.}
\end{aligned}$$

Conversely define $g: \mathcal{F}(\mathbf{D}) \to \mathbf{P}$ by:

$$g(\{b_1, \ldots, b_n\}) = f(b_1) \cup \cdots \cup f(b_n).$$

This is monotonic, for if $A \sqsubseteq_{\mathrm{EM}} B$ then we can write $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$ with $a_i \sqsubseteq b_i$ $(i = 1, n)$ making it evident that $g(A) \sqsubseteq g(B)$. Now let $f^\dagger$ be the unique continuous extension of g to $\mathcal{P}(\mathbf{D})$. Since it is continuous we can check the above diagram commutes by looking only at the finite elements of $\mathbf{D}$:

$$f^\dagger(\{b\}_{\mathrm{P}}) = f^\dagger([\{b\}]) = g(\{b\}) = f(b).$$

(If f is strict this shows that $f^\dagger(\perp) = f^\dagger(\{\perp\}_{\mathrm{P}}) = f(\perp) = \perp$ and so $f^\dagger$ is too.) It only remains to check that $f^\dagger$ is linear and by continuity it is enough to check that for the finite elements of $\mathcal{P}(\mathbf{D})$ and we note:

$$\begin{aligned}
f^\dagger([A] \cup_{\mathrm{P}} [B]) &= f^\dagger([A \cup B]) \\
&= \bigcup_{a \in A} g(a) \cup \bigcup_{b \in B} g(b) \\
&= g(A) \cup g(B) \\
&= f^\dagger([A]) \cup f^\dagger([B]).
\end{aligned}$$

To complete the proof we note that $(\cdot)^\dagger$ is monotonic, as if $f \sqsubseteq f'$ then $g \sqsubseteq g'$ (using an evident notation and so $f^\dagger \sqsubseteq {f'}^\dagger$. It therefore follows as per Fact 1 that $(\cdot)^\dagger$ is an isomorphism of partial orders with inverse. $H = \lambda h. h \circ \{\cdot\}_{\mathrm{P}}$. Note that H is linear:

$$\begin{aligned}
H(h \cup h') &= (h \cup h') \circ \{\cdot\}_{\mathrm{P}} \\
&= \lambda d \in \mathbf{D}. h \cup h'(\{d\}_{\mathrm{P}}) \\
&= \lambda d. h(\{d\}_{\mathrm{P}}) \cup h'(\{d\}_{\mathrm{P}}) \\
&= H(h) \cup H(h').
\end{aligned}$$

It follows that $\dagger$ is linear:

$$\begin{aligned}(f \cup f')^\dagger &= (H(f^\dagger) \cup H({f'}^\dagger))^\dagger \\ &= H(f^\dagger \cup {f'}^\dagger)^\dagger \\ &= f^\dagger \cup {f'}^\dagger\end{aligned}$$

(and it is clear that strictness is also preserved). ∎

The import of this theorem is that powerdomains conceived of as free continuous algebras always exist. They are built by making a free monotonic algebra (here $\mathcal{F}(\mathbf{D})/\sqsubseteq_{\mathrm{EM}}$ which is $\mathcal{F}(\mathbf{D})$, modulo $\sqsubseteq_{\mathrm{EM}}$) and completing it to get the free continuous one. This is a very simple procedure and can be applied to any collection of equations and even inequations. Later on we shall take advantage of the idea to construct various other flavours of powerdomains. This by no means makes use of all the possibilities available and we do not really know what scope they have.

Exercise 58. Show that for $f: \mathbf{D} \to \mathcal{P}(\mathbf{E})$ the extension $f^\dagger: \mathcal{P}(\mathbf{D}) \to \mathcal{P}(\mathbf{E})$ is given by the formula:

$$f^\dagger(I) = \bigcup_{A \in I} \bigcup_{a \in A} f(a).$$

It is easy to see that $\mathcal{P}$ is a locally continuous covariant functor on ω-$\mathcal{ALG}$. It is already defined on objects and for morphisms $f: \mathbf{D} \to \mathbf{E}$ we define

$$\mathcal{P}(f) = (\{\cdot\}_{\mathbf{E}} \circ f)^\dagger$$

as with the relational powerdomain.

Exercise 59. Show this does define a locally continuous functor. Find a formula for $\mathcal{P}(f)$.

It follows that we can solve domain equations in ω-$\mathcal{ALG}$ using $+$, $\times$, $(\cdot)_\perp$, $\otimes$, $\mathcal{P}$ (just repeat the developments of Chapter 4 and 5 but in ω-$\mathcal{ALG}$ using the information given in Chapter 6). And with a little effort (using $\mathcal{SFP}$) in ω-$\mathcal{ALG}$ we can add in $\to$ and $\to_\perp$. For we know that $\mathcal{SFP}$ is closed under $+$, $\times$, $(\cdot)_\perp$, $\otimes$, $\to$ and $\to_\perp$. To see that it is closed under $\mathcal{P}$ we note that the above construction shows $\mathcal{P}(\mathbf{D})$ finite when $\mathbf{D}$ is. Therefore if $\mathbf{D}$ is an $\mathcal{SFP}$ object we can write $\mathbf{D} = \lim_\to \langle \mathbf{D}_n, f_n \rangle$ in $\mathcal{CPO}^{\mathrm{E}}$ and so in ω-$\mathcal{ALG}^{\mathrm{E}}$ with the $\mathbf{D}_n$ finite and so $\mathcal{P}(\mathbf{D}) = \lim_\to \langle \mathcal{P}(\mathbf{D}_n), \mathcal{P}(f_n) \rangle$ in ω-$\mathcal{ALG}^{\mathrm{E}}$ (and hence $\mathcal{CPO}^{\mathrm{E}}$) with the $\mathcal{P}(\mathbf{D}_n)$ finite, using the local continuity of $\mathcal{P}$, by analogy with Chapter 4. This shows $\mathcal{SFP}$ closed under $\mathcal{P}$ and we can solve all our domain equations in $\mathcal{SFP}$ (again repeating Chapter 4/5 in $\mathcal{SFP}$ and using the information on $\mathcal{SFP}$ given in Chapter 6).

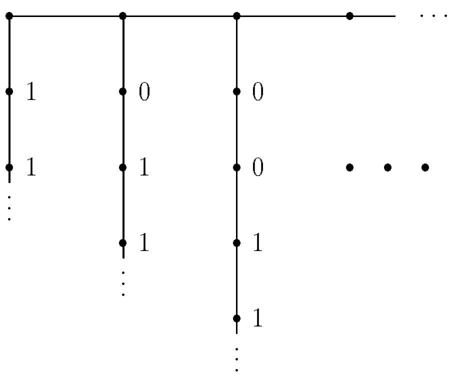
Exercise 60. The above discussion uses the fact that if all the $\mathbf{D}_n$ are ω-algebraic then $\mathbf{D} = \lim_\to \langle \mathbf{D}_n, f_n \rangle$ in $\mathcal{CPO}^{\mathrm{E}}$ iff $\mathbf{D} = \lim_\to \langle \mathbf{D}_n, f_n \rangle$ in ω-$\mathcal{ALG}^{\mathrm{E}}$. Prove this.

Exercise 61. Prove directly using the minimal upper bound characterisation of $\mathcal{SFP}$ objects given in Chapter 6 p. 69 that $\mathcal{P}(\mathbf{D})$ is in $\mathcal{SFP}$ object if $\mathbf{D}$ is.

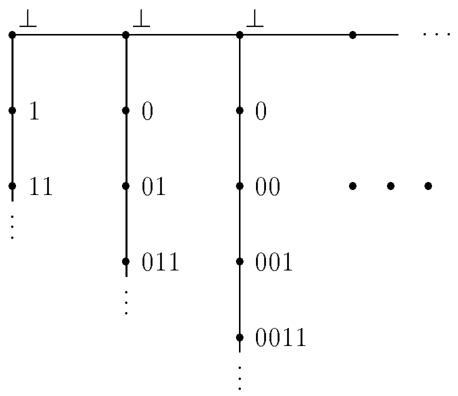
We now turn to the question of what $\mathcal{P}(\mathbf{D})$ looks like considered as a family of subsets of $\mathbf{D}$. As before we need to see what sets we want and how to order them. We will then have to check that the resulting structure is isomorphic to $\mathcal{P}(\mathbf{D})$ as defined above.

Finitely Generable Sets

If we write down the execution tree of program (7) above we arrive at the following



where we have only indicated the effects of the print instructions. Thus A_7 is obtained as the sequence of what is printed out along the branches. As a slight reformulation we might instead write what has been printed so far at a node



and then A_7 is obtained as the lubs along the branches. We have here to deal with *finitely branching trees*. A *generating finitarily branching tree* is a finitarily branching tree with nodes labelled by elements of $\mathbf{D}$ so that the closer a node is to the root the smaller is its label. Every complete branch *generates* the lub of the labels of its nodes and the tree generates the set of elements of $\mathbf{D}$ generated by its nodes. What we do is take (the finite elements) of **Tapes** as our canonical finitarily branching tree.

Definition 8. Let $\mathbf{D}$ be an ω-algebraic cpo. A subset X of $\mathbf{D}$ is *finitely generable* (*fg*) iff there is a continuous function $f\colon \mathbf{Tapes} \to \mathbf{D}$ such that $X = Bd(f)$ where the *boundary* of f is given by

$$Bd(f) = \{f(w) \mid |w| = \omega\}$$

(that is it is the set of values of f on infinite tapes). We write $\mathcal{F}_g(\mathbf{D})$ for the collection of fg subsets of $\mathbf{D}$.

Exercise 62. Show every finite nonempty set is finitely generable. Show that union of two fg sets is fg. Show that if $f\colon \mathbf{D} \to \mathbf{E}$ is continuous and $X \subseteq \mathbf{D}$ is fg then so is $f(X)$. Show that $\{0^\omega\} \cup \{0^n 1^\omega\}$ is fg.; show that $\{0^n 1^\omega\}$ is *not* fg (and neither is $\mathbf{N}$ as a subset of $\mathbf{N}_\perp$). How many fg subsets of $\mathbf{D}$ are there? Show that fg sets are *not* closed under intersection, complementation, infinite union or inverse functions (the last meaning that if $Y \subseteq \mathbf{E}$ is fg then $f^{-1}(Y)$ need not be with f continuous).

Exercise 63. Show that a set is fg iff it is generated by a finitely branching generating tree.

Lemma 3. *If $X \subseteq \mathbf{D}$ is finitely generable then it is $Bd(f)$ for some f such that $f(w)$ is finite when w is.*
Proof. Let $X = Bd(g)$. Let $b_0, b_1, \ldots$ enumerate $B_{\mathbf{D}}$. Define f by putting $f(\perp) = \perp$ and for $w \in B_{\mathbf{Tapes}}$, $t \in \{0, 1\}$

$$f(w \frown t) = \text{the first } b_n \text{ greater than } f(w) \text{ and } b_n \sqsubseteq g(w \frown t) \text{ and } b_j \sqsubseteq b_n \text{ (s.t. } \forall j \leq |w|.b_j \sqsubseteq g(w \frown t)).$$

Note that $f(w)$ is well-defined and less than $g(w)$ (for finite w). Then f is monotonic on finite tapes and so is uniquely extended to a continuous function on **Tapes**. Clearly, then, $f \sqsubseteq g$. The converse is left as **Exercise 64.** ∎

How to Order Our Sets

Let us denote by $\underset{\sim}{\sqsubset}$ the order we are looking for on subsets of any given ω-algebraic $\mathbf{D}$. We expect that this contains $\sqsubseteq_{\mathrm{EM}}$ but know from the above that since $A_8 \sqsubseteq_{\mathrm{EM}} A_7$ it cannot be identical with it.

Exercise 65. It would have been natural to take the collection of the fg sets of $\mathbf{D}$ with the natural singleton and union functions and preordered by $\sqsubseteq_{\mathrm{EM}}$ and then divide out by the induced equivalence to get a candidate powerdomain. But we know from the above investigation that something must go wrong even with $\mathbf{D} = \mathbf{Tapes}$: either we don't get a cpo or singleton or union is not continuous. Which of these happens?

Exercise 66. Show that if $f\colon \mathbf{D} \to \mathbf{E}$ is continuous and for two subsets X and Y of $\mathbf{D}$ we have $X \sqsubseteq_{\mathrm{EM}} Y$ then $f(X) \sqsubseteq_{\mathrm{EM}} f(Y)$.

Following the last exercise it seems natural that $\underset{\sim}{\sqsubset}$ should be preserved by continuous functions and since it should be $\sqsubseteq_{\mathrm{EM}}$ on discrete cpos we expect that if we define for all subsets of $\mathbf{D}$

$$X \underset{\sim}{\sqsubset}_{\mathrm{EM}} Y \quad \text{iff} \quad \forall f\colon \mathbf{D} \to \mathbb{O}. f(X) \sqsubseteq_{\mathrm{EM}} f(Y)$$

(and let $\simeq_{\mathrm{EM}}$ be the associated equivalence) then $X \underset{\sim}{\sqsubset} Y$ implies $X \underset{\sim}{\sqsubset}_{\mathrm{EM}} Y$. Thus we have: $\sqsubseteq_{\mathrm{EM}} \subseteq \underset{\sim}{\sqsubset} \subseteq \underset{\sim}{\sqsubset}_{\mathrm{EM}}$. One can think of $f^*\colon \mathbf{D} \to \mathbb{O}$ as an *experiment* on $\mathbf{D}$.

Exercise 67. Show that for any continuous $f\colon \mathbf{D} \to \mathcal{P}(X_\perp)$ with X countable we have: $X \underset{\sim}{\sqsubset}_{\mathrm{EM}} Y$ implies $\bigcup f(X) \sqsubseteq_{\mathrm{EM}} \bigcup f(Y)$.

We now study $\underset{\sim}{\sqsubset}_{\mathrm{EM}}$ with a view to arguing that it should be our choice of $\underset{\sim}{\sqsubset}$.

Theorem 4. *Let $\mathbf{D}$ be an ω-algebraic cpo. Let X and Y be subsets of $\mathbf{D}$ and consider the following conditions.*

(1) $X \sqsubseteq_{\mathrm{EM}} Y$,
(2) $X \underset{\sim}{\sqsubset}_{\mathrm{EM}} Y$,
(3) $\forall x \in X. \forall b \in B_{\mathbf{D}}. (b \sqsubseteq x \Rightarrow \exists y \in Y. b \sqsubseteq y) \wedge \forall y \in Y. \exists x \in X. x \sqsubseteq y$,
(4) $\forall A \subseteq B_{\mathbf{D}}. (A \textit{ finite} \wedge A \sqsubseteq_{\mathrm{EM}} X) \Rightarrow A \sqsubseteq_{\mathrm{EM}} Y$.

Then we here (1) $\Rightarrow$ (2) $\Leftrightarrow$ (3) $\Rightarrow$ (4) with (3) $\Leftrightarrow$ (4) if X is finitely generable and (1) $\Leftrightarrow$ (2) if Y is finite or if X is a subset of $B_{\mathbf{D}}$.

Proof. (1) $\Rightarrow$ (2). Already known.

(2) $\Rightarrow$ (3). That (2) implies the first conjunct is known from Theorem 1. For the converse take $y \in Y$. Define $f\colon \mathbf{D} \to \mathbb{O}$ by $f(d) = \top$ iff $d \not\sqsubseteq y$. Then $\perp \in f(Y)$ and so $\perp \in f(X)$; then for some x in X, $f(x) = \perp$, which is to say $x \sqsubseteq y$.

(3) $\Rightarrow$ (2). Left as an exercise.

(3) $\Rightarrow$ (4). Left as an exercise.

(4) $\Rightarrow$ (3) when X is fg. Take $b \sqsubseteq x$ with $b \in B_{\mathbf{D}}$ and $x \in X$. Then $\{\perp, b\} \sqsubseteq_{\mathrm{EM}} X$ and so $\{\perp, b\} \sqsubseteq_{\mathrm{EM}} Y$ and so $b \sqsubseteq y$ for some y in Y. Next take $y \in Y$ and $f\colon \mathbf{Tapes} \to \mathbf{D}$ generating X. Suppose for the sake of contradiction that for all $x \in X$ we have $x \not\sqsubseteq y$. Then for any infinite w in **Tapes**, $f(w) \not\sqsubseteq y$ and so there is a finite $a \sqsubseteq w$ with $f(a) \not\sqsubseteq y$. Now as **Tapes** is a finitarily branching tree it follows by König's Lemma that there is a finite cross section A of **Tapes** with $A \subseteq B_{\mathbf{Tapes}}$ and for all a in A, $f(a) \not\sqsubseteq y$. Then $f(A) \sqsubseteq_{\mathrm{EM}} X$ and so $f(A) \sqsubseteq_{\mathrm{EM}} Y$ but then there must be an $f(a)$ with $f(a) \sqsubseteq y$ giving us the required contradiction.

(2) $\Rightarrow$ (1) when X is a subset of $B_{\mathbf{D}}$. Immediate from (2) $\Rightarrow$ (4).

(2) $\Rightarrow$ (1) when Y is finite. Left as **Exercise 68** [*Hint* Use (3)]. ∎

So for finite sets we have $\sqsubseteq_{\mathrm{EM}} = \underset{\sim}{\sqsubset} = \underset{\sim}{\sqsubset}_{\mathrm{EM}}$ and indeed we know that for finite $\mathbf{D}$ we have $\mathcal{P}(\mathbf{D}) = \mathcal{F}(\mathbf{D})/\sqsubseteq_{\mathrm{EM}}$.

Condition (3) shows how close $\underset{\sim}{\sqsubset}_{\mathrm{EM}}$ is to being $\sqsubseteq_{\mathrm{EM}}$ and condition (4) shows that for fg sets $X \underset{\sim}{\sqsubset}_{\mathrm{EM}} Y$ iff any finite Egli-Milner information about X (= picture of a cross-section of the development of X) is also about Y. In fact it is natural to expect any fg set is the limit of its cross-sections and we now show this is true w.r.t. $\underset{\sim}{\sqsubset}_{\mathrm{EM}}$.

Definition 9. For $f\colon \mathbf{Tapes} \to \mathbf{D}$ define $Bd_n(f) = \{f(w) \mid |w| = n\}$.

Clearly for $m \leq n \leq \omega$ we have $Bd_m(f) \sqsubseteq_{\mathrm{EM}} Bd_n(f) \sqsubseteq_{\mathrm{EM}} Bd(f)$.

Lemma 4. *Let $f\colon \mathbf{Tapes} \to \mathbf{D}$ generate X. If A is a finite subset of $B_{\mathbf{D}}$ such that $A \sqsubseteq_{\mathrm{EM}} X$ then for some n, $A \sqsubseteq_{\mathrm{EM}} Bd_n(f)$. Further if $Bd_n(f) \underset{\sim}{\sqsubset}_{\mathrm{EM}} Y$ for all n then $X \underset{\sim}{\sqsubset}_{\mathrm{EM}} Y$.*

Proof. As $A \sqsubseteq_{\mathrm{EM}} X$ for every infinite w in **Tapes** there is a finite $v \sqsubseteq w$ with $a \sqsubseteq f(v)$ for some a in A. Hence by König's Lemma there is an n_0 such that $|v| \geq n_0$ implies $a \sqsubseteq f(v)$ for some a in A. Equally as $A \sqsubseteq_{\mathrm{EM}} X$ for every a in A there is an infinite w with $a \sqsubseteq f(w)$ and hence a finite $v_a \sqsubseteq w$ with $a \sqsubseteq f(v_a)$. Now put $n = \max(n_0, \max_{a \in A}(|v_a|))$ and then $A \sqsubseteq Bd_n(f)$. The rest now follows using from Theorem 4 that (4) $\Rightarrow$ (2) for fg X and (1) $\Leftrightarrow$ (2) for finite subsets of finite elements. ∎

Now assuming a similar property holds of $\lesssim$ we can show it must be $\lesssim_{\mathrm{EM}}$ at least as regards fg sets. Specifically, assume that:

$$\forall f\colon \mathbf{Tapes} \to \mathbf{D}.\forall Y \subseteq \mathbf{D}.(\forall n.Bd_n(f) \lesssim Y) \Rightarrow Bd(f) \lesssim Y.$$

Now suppose $X \lesssim_{\mathrm{EM}} Y$ with X fg. Then by Lemma 3 there is an $f\colon \mathbf{Tapes} \to \mathbf{D}$ with $X = Bd(f)$ and $Bd_n(f) \subseteq B_{\mathbf{D}}$ for every n. Then for any n we have $Bd_n(f) \lesssim_{\mathrm{EM}} Y$; so by Theorem 4 we have $Bd_n(f) \sqsubseteq_{\mathrm{EM}} Y$ (since $Bd_n(f)$ is a finite set of finite elements) and so $Bd_n(f) \lesssim Y$. Therefore by the assumption $X = Bd(f) \lesssim Y$. *Thus we have established, under reasonable assumptions, that* $\lesssim$ *is* $\lesssim_{\mathrm{EM}}$.

Our proposed powerdomain according to this analysis is therefore $\langle \mathcal{F}_g(\mathbf{D}), \lesssim_{\mathrm{EM}}\rangle/\lesssim_{\mathrm{EM}}$. Since we already have a proposal from Theorem 3 we must now prove that the two possibilities are isomorphic.

Lemma 5. *Let $A_0 \sqsubseteq_{\mathrm{EM}} A_1 \sqsubseteq_{\mathrm{EM}} \cdots$ be an increasing sequence of elements of $\mathcal{F}(\mathbf{D})$. Then there is an $f\colon \mathbf{Tapes} \to \mathbf{D}$ with every A_n being some $Bd_m(f)$ (with $m \geq n$). Further $Bd(f) = \{\bigsqcup a_n \mid \forall n.a_n \in A_n,\ a_0 \sqsubseteq a_1 \sqsubseteq \cdots\}$.*
Proof. Define a finitary tree $T = \{\langle a_0, \ldots, a_n\rangle \mid a_i \in A_i$ and $a_0 \sqsubseteq \cdots \sqsubseteq a_n\}$ and label $\langle a_0, \ldots, a_n\rangle$ by a_n for $n > 0$ and by $\perp$ for $n = 0$. Then A_n is the set of labels of the set of nodes of depth $n + 1$. Finally pad out T so that it is (isomorphic to) $B_{\mathbf{Tapes}}$ and choose the labelling so that every A_n is the set of labels of some $\mathbf{Tapes}_m$ and use the labels to construct f. The last part is an easy König's Lemma argument on $\mathbf{T}$. ∎

We also need a lemma about ω-algebraic cpos.

Lemma 6. *Let P be a partial order with a least element. Suppose $B \subseteq P$ is a countable subset of P such that:*

(1) Every element of P is the lub of an increasing sequence of elements of B.
(2) Every increasing sequence of elements of B has a lub in P.
(3) If $b \sqsubseteq \bigsqcup_P b_i$ where $b_0 \sqsubseteq b_1 \sqsubseteq \cdots$ and b and the b_i are all in B then $b \sqsubseteq$ some b_n.

Then P is ω-algebraic and B its set of finite elements.
Proof. To see P is a cpo let $d_0 \sqsubseteq d_1 \sqsubseteq \ldots$ be an increasing sequence in P. Let $X = \{b \in B \mid b \sqsubseteq \text{some } d_i\}$. Then $\perp$ is in X. Then one sees from (1) and (3) that X is directed. Hence as X is countable we can take a cofinal chain in X to show by (2) that $d = \bigsqcup X$ exists. By (1) we have that d is an upper bound of every d_n and the definition of X shows it must be the least one.

Next take b in B and let $d_0 \sqsubseteq d_1 \sqsubseteq \cdots$ be an increasing sequence and suppose $b \sqsubseteq \bigsqcup_n d_n$. Define X as above. We know $b \sqsubseteq \bigsqcup X$ and hence as X is directed and countable it follows from (3) that $b \sqsubseteq$ some element of $X \sqsubseteq$ some d_n. Hence b is finite and so $B_0 \subseteq B$.

Therefore by Chapter 6 Lemma 1, $\mathbf{D}$ is ω-algebraic. ∎

It follows from Lemma 6 that $\mathcal{F}_g(\mathbf{D})/\lesssim_{\mathrm{EM}}$ is ω-algebraic and its set of finite elements $\{A\downarrow \mid A \in \mathcal{F}(\mathbf{D})\}$ (where, of course, $A\downarrow = \{X \in \mathcal{F}_g(\mathbf{D}) \mid X \lesssim_{\mathrm{EM}} A\}$). For condition (1) of that lemma is immediate from Lemma 3 and Lemma 4; condition (2) follows from Lemma 4 and Lemma 5; condition (3) from Lemma 5 and Lemma 4. But now we note

$$\begin{aligned} \langle B_{\mathcal{F}_g(\mathbf{D})/\lesssim_{\mathrm{EM}}}, \subseteq\rangle &\cong \langle \mathcal{F}(\mathbf{D}), \lesssim_{\mathrm{EM}}\rangle/\lesssim_{\mathrm{EM}} && \text{since for } A,\ B \text{ in } \mathcal{F}(\mathbf{D}),\ A\downarrow \subseteq B\downarrow \text{ iff } A \lesssim_{\mathrm{EM}} B \\ &= \langle \mathcal{F}(\mathbf{D}), \sqsubseteq_{\mathrm{EM}}\rangle/\sqsubseteq_{\mathrm{EM}} && \text{by remark after Theorem 4.} \end{aligned}$$

Therefore $\mathcal{F}_g(\mathbf{D})/\lesssim_{\mathrm{EM}} = \overline{\mathcal{F}(\mathbf{D})} = \mathcal{P}(\mathbf{D})$ showing as promised that the two possibilities are indeed isomorphic.

Exercise 69. Show that we should define $\{\cdot\}_{\mathrm{P}}\colon \mathbf{D} \to \mathcal{F}_g(\mathbf{D})/\lesssim_{\mathrm{EM}}$ and $\cup_{\mathrm{P}}\colon (\mathcal{F}_g(\mathbf{D})/\lesssim_{\mathrm{EM}})^2 \to \mathcal{F}_g(\mathbf{D})/\lesssim_{\mathrm{EM}}$ by $\{d\}_{\mathrm{P}} = \{d\}\downarrow$ and $A\downarrow \cup_{\mathrm{P}} B\downarrow = (A \cup B)\downarrow$ respectively. No particularly revealing formula for function extension is known.

Thus we have now greatly increased our understanding of powerdomains: the sets are the finitely generable sets, modulo $\lesssim_{\mathrm{EM}}$ and the ordering is $\lesssim_{\mathrm{EM}}$; further the singleton and union functions have natural definitions and Lemma 5 shows us what lubs of increasing chains of finite elements look like.

Canonical Representatives and Scott-Compact Sets

We now explore the equivalence relation $\simeq_{\mathrm{EM}}$ in order to obtain canonical (actually maximal) elements of the equivalence classes. Following Theorem 3(3) there is a very natural closure operation. For any set X define X^* by:

$$X^* = \{y \mid (\exists x \in X.x \sqsubseteq y) \wedge (\forall b \in B_{\mathbf{D}}.b \sqsubseteq y \Rightarrow \exists x \in X.b \sqsubseteq x).$$

Proposition 1. *For any set $X \subseteq \mathbf{D}$ we have*

(1) $X \subseteq X^*$,
(2) $X^{**} = X^*$,
(3) $X \simeq_{\mathrm{EM}} Y \quad$ iff $\quad X^* = Y^*$

(and so (4) $X \simeq_{\mathrm{EM}} X^*$).
Proof. Exercise 70. ∎

Thus $(\cdot)^*$ is a closure operation and X^* is the maximal member of its equivalence class. What we would like to prove now is that X^* is fg; however it is *open* whether this is true in general but we can proceed further by considering the *Scott-compact* sets which, as we will see, generalise fg sets (Proposition 2) but are the fg sets, to within $\simeq_{\mathrm{EM}}$. (It should be noted that better results will be obtained later for *special kinds* of ω-algebraic cpos, the so-called 2/3 SFP ones.)

Exercise 71. This exercise looks at the convexity part of $\simeq_{\mathrm{EM}}$ as illustrated by $A_5 = A_6$.

Definition 10. Let P be a preorder. A subset X of P is *convex* iff $\forall x, z \in X. \forall y \in P. x \sqsubseteq y \sqsubseteq z \Rightarrow y \in X$. The least convex set containing a given $X \subseteq P$ is:

$$Con(X) = \{y \in P \mid \exists x, z \in X. x \sqsubseteq y \sqsubseteq z\}.$$

Show that $Con(X^*) = X^* \supseteq Con(X)$ but equality does not always hold. Show that $\mathcal{F}(\mathbf{D})/\precsim_{\mathrm{EM}} \cong \langle \{Con(A) \mid A \in \mathcal{F}(\mathbf{D})\}, \sqsubseteq_{\mathrm{EM}} \rangle$ and conclude that for *finite* $\mathbf{D}$, we have $\mathcal{P}(\mathbf{D}) \cong \langle \{Con(A) \mid A \subseteq \mathbf{D}, A \neq \emptyset\}, \sqsubseteq_{\mathrm{EM}} \rangle$. Note that *Con* is *not* a topological closure operator (that is we do not always have $Con(X \cup Y) = Con(X) \cup Con(Y)$) and conclude the same for $(\cdot)^*$.

Definition 11. A set $C \subseteq B_{\mathbf{D}}$ *covers* $X \subseteq \mathbf{D}$ iff $\forall x \in X. \exists c \in C. c \sqsubseteq x$. A set $X \subseteq \mathbf{D}$ is *Scott-compact* iff whenever C covers X so does a finite subset of C.

The idea is to use compactness to replace König's Lemma arguments.

Exercise 72. Show that X is Scott-compact iff it is compact in the Scott topology. Show that whenever C is finite and covers X then $C \cup X$ is Scott-compact. Conclude there are Scott-compact sets which are not fg; how many are there?

Proposition 2. *Every finitely generable set is Scott-compact.*
Proof. Let X be fg with generating function $f: \mathbf{Tapes} \to \mathbf{D}$. Let C cover X. Then for every infinite w in $\mathbf{Tapes}$ $f(w) \sqsupseteq$ some c in C. So $f(v) \sqsupseteq c$ for some finite $v \sqsubseteq w$. Applying König's Lemma we obtain a cross-section of such v's and the corresponding c's provide the needed finite subset of C. ∎

This proposition makes it plain why sets like $\mathbf{N} \subseteq \mathbf{N}_\perp$ are not fg.

Exercise 73. Show that if $X \simeq_{\mathrm{EM}} Y$ then X is Scott-compact iff Y is.

So we know that for a fg X, X^* is Scott-compact and so by Proposition 1 it is the maximal member of the equivalence class $\{Y \subseteq \mathbf{D} \mid Y \text{ is Scott-compact and } Y \simeq_{\mathrm{EM}} X\}$. This suggests that

$$\mathcal{P}(\mathbf{D}) \cong \langle (\cdot)^*\text{-closed nonempty Scott-compact subsets of } \mathbf{D}, \precsim_{\mathrm{EM}} \rangle$$

and to establish this it only remains to show that every Scott-compact set is $\simeq_{\mathrm{EM}}$-equivalent to a fg set, which we now do.

Lemma 7. *Let $C \subseteq \mathbf{D}$ be nonempty and Scott-compact. Then $I(C) \stackrel{\mathrm{def}}{=} \{A \in \mathcal{F}(\mathbf{D}) \mid A \sqsubseteq_{\mathrm{EM}} C\}$ is a directed ideal and has $\precsim_{\mathrm{EM}}$-lub C (that is if for some $X \subseteq \mathbf{D}$ we have $\forall A \in I(C). A \precsim_{\mathrm{EM}} X$ then $C \precsim X$).*
Proof. To see $I(C)$ is an ideal the only nontrivial point is to show that it is directed. Clearly $\{\perp\}$ is in $I(C)$ so it is nonempty. Suppose A and B are in $I(C)$. Then C is covered by A and B and so as C is compact it is covered by $\{x \in B_{\mathbf{D}} \mid \exists a \in A, b \in B, c \in C. a \sqsubseteq x \wedge b \sqsubseteq x \wedge x \sqsubseteq c\}$ and so by some finite subcollection, X, of the latter set. Thus we have an X in $\mathcal{F}(\mathbf{D})$ which is covered by both A and B and with $X \sqsubseteq_{\mathrm{EM}} C$.

For any a in A there is a c in C with $c \sqsupseteq a$ and for that c there is an x in X so that $c \sqsupseteq x$. But then there is an a' such that $a, x \sqsubseteq a' \sqsubseteq c$. Let A' be the finite collection of these a''s. Then $A \sqsubseteq_{\mathrm{EM}} X \cup A' \sqsubseteq_{\mathrm{EM}} C$ and B covers $X \cup A'$. Defining B' similarly we find that $A, B \sqsubseteq_{\mathrm{EM}} X \cup A' \cup B' \sqsubseteq_{\mathrm{EM}} C$, concluding the proof that $I(C)$ is directed.

Next suppose that $\forall A \in I(C).A \precsim_{\mathrm{EM}} X$. We will show that $C \precsim X$. Take any $f\colon \mathbf{D} \to \mathbb{O}$ to show that $f(C) \sqsubseteq_{\mathrm{EM}} f(X)$. If $\top \in f(C)$ then for some c in C and finite $a \sqsubseteq c$ we have $\top = f(a)$ and so considering $\{a, \bot\}$ in $I(C)$ we find that $\top \in f(X)$. finally suppose $\bot \notin f(C)$. Then $\{a \in B_{\mathbf{D}} \mid f(a) = \top \wedge \exists c \in C.a \sqsubseteq c\}$ covers C and so as C is compact it has a finite subcover, A, say. Then $A \sqsubseteq_{\mathrm{EM}} C$ and $\bot \notin f(A)$. So as $A \sqsubseteq_{\mathrm{EM}} X$ we have $\bot \notin f(X)$ either. ∎

(Compare this last argument to the König's Lemma argument in the last part of theorem).

Theorem 5. *Every Scott-compact set is $\simeq_{\mathrm{EM}}$-equivalent to some fg set.*
Proof. Immediate from Lemmas 7, 4 and Theorem 4. ∎

This establishes our desired characterisation of $\mathcal{P}(\mathbf{D})$ as the collection of $(\cdot)^*$-closed Scott-compact sets partially ordered by $\sqsubseteq_{\mathrm{EM}}$. So we have our first characterisation of $\mathcal{P}(\mathbf{D})$ as a collection of subsets; the open problem is still to see if all $(\cdot)^*$-closed Scott-compact sets are actually finitely generable.

Exercise 74. Show that in terms of the latest characterisation of $\mathcal{P}(\mathbf{D})$ we have for $x \in \mathbf{D}$, $X, Y \in \mathcal{P}(\mathbf{D})$, $f\colon \mathbf{D} \to \mathcal{P}(\mathbf{E})$, $g\colon \mathbf{D} \to \mathcal{P}(\mathbf{E})$:

$$\begin{aligned} \{x\}_{\mathrm{P}} &= \{x\}, \\ X \cup_{\mathrm{P}} Y &= (X \cup Y)^*, \\ f^\dagger(X) &= (\bigcup_{x \in X} f(X))^*, \\ \mathcal{P}(g)(X) &= (g(X))^*. \end{aligned}$$

The Lawson Topology

Our understanding of powerdomain is still incomplete in that we have not systematically considered the identification $A_7 = A_8$. What seems to be wanted is a notion of limit of a sequence so that, for example:

$$\lim_{n \to \infty} 0^n 1^\omega = 0^\omega.$$

Attending to this desire will have several pleasant consequences: $(\cdot)^*$-closure will be explained in terms of these limits and convex closure; the X^* will be fg; $\precsim_{\mathrm{EM}}$ will just be $\sqsubseteq_{\mathrm{EM}}$ on the X^*. (But all this will only work for certain ω-algebraic cpos, remember.)

So let us say when $\langle x_n \rangle$ has *limit* x (or *converges* to x) writing:

$$\begin{aligned} \lim_n x_n = x \quad &\text{iff} \quad \forall b \in B_{\mathbf{D}}.[b \sqsubseteq x \Leftrightarrow \forall_\infty n.b \sqsubseteq x_n] \wedge [b \not\sqsubseteq x \Leftrightarrow \forall_\infty n.b \not\sqsubseteq x_n] \\ &(\text{iff} \quad \forall b \in B_{\mathbf{D}}.[b \sqsubseteq x \Rightarrow \forall_\infty n.b \sqsubseteq x_n] \wedge [b \not\sqsubseteq x \Rightarrow \forall_\infty n.b \not\sqsubseteq x_n]) \\ &(\text{iff} \quad \forall b \in B_{\mathbf{D}}.[b \sqsubseteq x \Rightarrow \forall_\infty n.b \sqsubseteq x_n] \wedge [\exists_\infty n.b \sqsubseteq x_n \Rightarrow b \sqsubseteq x]). \end{aligned}$$

(Here the notations $\forall_\infty n.P(n)$, $\exists_\infty n.P(n)$ mean that P holds for all but finitely many n, respectively infinitely many n.)

The elementary properties of these limits form an exercise.

Exercise 75.

(1) Let $\langle x_n \rangle$ and $\langle y_n \rangle$ be sequences with respective limits x and y. Show that if $x_n \sqsubseteq y_n$ for all n then $x \sqsubseteq y$.

(2) Show that if $\langle x_n \rangle$ is an increasing sequence then it has limit, $\bigsqcup_n x_n$. Show that if $\langle x_n \rangle$ is a decreasing sequence then it has limit x iff $x = \bigsqcap_n x_n$.

(3) Show that if a sequence has limit x then any infinite subsequence has the same limit.

(4) Suppose $\lim_n x_{mn} = x_m$ for each m and $\lim_m x_m = x$. Then there are functions r, r' such that

$$\lim_m x_{r(m)r'(m)} = x.$$

[*Hint* Choose r and r' so that for any $1 \leq m$, $b \sqsubseteq x$ iff $b \sqsubseteq x_{r(m)r'(m)}$.]

Because of part (1) of this exercise, $\lim_n x_n$ is determined. Because of (2) and (4) if we put for any $X \subseteq \mathbf{D}$

$$\mathcal{C}l(X) \stackrel{\text{def}}{=} \{\lim_n x_n \mid x_n \text{ is a sequence of elements of } X \text{ with a limit}\}$$

then $\mathcal{C}l(X)$ is the least set of elements of $\mathbf{D}$ containing X and closed under limits. Because of (3) we have that $\mathcal{C}l(X \cup Y) = \mathcal{C}l(X) \cup \mathcal{C}l(Y)$ and so we have a *topological* closure operator. This topology is called the *Lawson* topology and the next few exercises discuss it (they will not be used in the main development).

Exercise 76.

(1) Show that a subbasis of the topology is provided by the sets $P_b = \{x \in \mathbf{D} \mid x \sqsupseteq b\}$ and $N_b = \mathbf{D} \backslash P_b$ $(b \in B_{\mathbf{D}})$.

(2) Show one can define a *metric* as follows. Let $b_0, b_1, \ldots$ be an enumeration of $B_{\mathbf{D}}$. Define

$$d(x,y) = \begin{cases} 0 & (x = y), \\ \dfrac{1}{1 + \mu n . b_n \in B_x \oplus B_y} & (x \neq y) \end{cases}$$

(where, of course, $X \oplus Y = (X \backslash Y) \cup (Y \backslash X)$).
Indeed show that this is even an *ultrametric* in the sense that the triangle inequality can be strengthened to

$$d(x,z) \leq \max(d(x,y), d(y,z)).$$

Show that $\lim_n x_n = x$ iff this is true in the metric. Show that $\langle x_n \rangle$ is a *Cauchy sequence* iff $\forall m.(\forall_\infty n. b_m \sqsubseteq x_n) \vee (\forall_\infty n. b_m \not\sqsubseteq x_n)$.

To proceed further we need to know that the Lawson topology is *compact*. What this means in terms of limits is that every sequence has a convergent subsequence. Strangely it turns out that an ω-algebraic cpo has compact Lawson topology iff it obeys 2 out of the 3 conditions needed to make it strongly algebraic!

Lemma 8. *Every sequence $\langle x_m \rangle$ contains a subsequence, $\langle y_m \rangle$, such that for all finite b either $(\forall_\infty m. b \sqsubseteq y_m)$ or else $\forall_\infty m. b \not\sqsubseteq y_m$.*
Proof. Let $b_0, b_1, \ldots$ enumerate the finite elements of $\mathbf{D}$. Put $B_n = \{b_i \mid 0 \leq i \leq n\}$. Say a partition $\langle P, Q \rangle$ of B_n is *good* if $\exists m \geq n. (\forall b \in P. b \sqsubseteq x_m) \wedge (\forall b \in Q. b \not\sqsubseteq x_m)$ and clearly every B_n has a good partition and so by König's Lemma there is an infinite sequence $\langle P_0, Q_0 \rangle, \langle P_1, Q_1 \rangle, \ldots$ of good partitions of $B_0, B_1, \ldots$ with $P_0 \subseteq P_1 \subseteq \cdots$ and $Q_0 \subseteq Q_1 \subseteq \cdots$. Now choose a subsequence $\langle y_m \rangle$ of $\langle x_n \rangle$ so that $(\forall b \in P_m . b \sqsubseteq y_m) \wedge (\forall b \in Q_m . b \not\sqsubseteq y_m)$. ∎

Recall that in Chapter 6 we defined $U(X)$ as the set of minimal upper bounds of X; we say $U(X)$ is *complete* if every upper bound of X is greater than or equal to some element of $U(X)$. Recall that if X is in $\mathcal{F}(\mathbf{D})$ then every element of $U(X)$ is finite.

Theorem 6 The 2/3 SFP Theorem. *The Lawson topology on $\mathbf{D}$ is compact iff every $U(\{a,b\})$ with a, b finite is both complete and finite.*
Proof. $\Leftarrow$) Suppose every such $U(\{a,b\})$ is both complete and finite. Let $\langle x_n \rangle$ be a sequence and choose an infinite subsequence $\langle y_m \rangle$ according to Lemma 7. Let $X = \{b \in B_{\mathbf{D}} \mid \forall_\infty m. b \sqsubseteq y_m\}$. We claim that X is directed. For if a, b are in X then since $U(\{a,b\})$ is complete we have that almost all y_m are greater than both a and b and hence some minimal upper bound of a and b. Hence as $U(\{a,b\})$ is finite it has an element c with $c \sqsubseteq y_m$ for infinitely many y_m. But then by the construction of $\langle y_m \rangle$ we have $c \sqsubseteq y_m$ for almost all y_m. Thus X is indeed directed and then one easily proves that $\lim_m y_m = \bigsqcup X$ (show this!).

$\Rightarrow$) We know from Exercise 75 that every decreasing sequence has a glb. Now take finite a, b and suppose x is an upper bound of a and b. Let C be $\{c \in B_{\mathbf{D}} \mid x \sqsupseteq c, c \sqsupseteq a, c \sqsupseteq b\}$. Clearly $C \neq \emptyset$. Let c_0, $c_1, \ldots$ be an enumeration of C and let $c_{r(n)}$ be defined by $c_{r(0)} = c_0$, $r(n+1) = \mu m > r(n). c_{r(n)} \sqsupseteq c_m$ if that exists and $r(n)$ otherwise. Then $c = \bigsqcap c_{r(n)}$ exists. By construction c is less than x, a lower bound of C and an upper bound of the finite elements a and b. It is also in $U(\{a,b\})$ as otherwise there is a finite c_m an upper bound of a, b and $\not\sqsupseteq c$ and m is some $r(n)$, so $c \sqsubseteq c_m$ a contradiction. So $U(\{a,b\})$ is complete.

Next suppose $U(\{a,b\})$ is infinite for the sake of contradiction. Let $c_0, c_1, \ldots$ be an enumeration without repetitions of an infinite subset of $U(\{a,b\})$. We can assume $\lim_n c_n = x$ exists by compactness. Now since a is less than all the c_n's we have $a \sqsubseteq x$ and similarly $b \sqsubseteq x$. Therefore some c in $U(\{a,b\})$ is less than or equal to x and so $c \sqsubseteq$ almost all the c_n and choosing $c_m \neq c$ we get $c \sqsubseteq c_m$ contradicting the assumption that c_m is a minimal upper bound of a and b and concluding the proof. ∎

Exercise 77.

(1) Show that the assumption of the 2/3 SFP Theorem are equivalent to saying that $U(A)$ is finite and complete for any finite subset, A, of $B_{\mathbf{D}}$; or equally that the intersection of Scott-compact sets is Scott-compact.

(2) Show that the metric of Exercise 76 is complete iff the Lawson topology is compact.

From now on we assume the Lawson topology is compact.

Lemma 9. *Let X be a subset of $\mathbf{D}$. Then $Con(Cl(X))$ is the least convex, Lawson closed set containing X.*

Proof. It is enough to show $Y = Con(Cl(X))$ Lawson-closed.

Let $\langle y_n \rangle$ be a convergent sequence in Y. Then there are $\langle x_n \rangle$, $\langle z_n \rangle$ in $Cl(X)$ with $x_n \sqsubseteq y_n \sqsubseteq z_n$. By compactness we can assume $\langle x_n \rangle$ and $\langle z_n \rangle$ to be convergent. Then by Exercise 75(1) $\lim_n x_n \sqsubseteq \lim_n y_n \sqsubseteq \lim_n z_n$ and so $\lim_n y_n$ is in Y showing that Y is Lawson-closed as required. ∎

Now we can see how well we can understand $(\cdot)^*$-closed and $\underset{\sim}{\sqsubseteq}_{\mathrm{EM}}$.

Lemma 10. *Let X be Scott-compact and let $\langle x_n \rangle$ be a convergent sequence in X. Then for some y in X, $y \sqsubseteq \lim_n x_n$.*

Proof. Suppose, for the sake of contradiction that no y in X is less than $x \stackrel{\text{def}}{=} \lim_n x_n$. Then for any such y we have a finite $a \sqsubseteq y$ with $a \not\sqsubseteq x$. By compactness, we can assume a finite collection $\{a_1, \ldots, a_n\}$ covering X with no $a_i \sqsubseteq x$. But now some a_i is less than infinitely many x_n and so less than x, being the required contradiction. ∎

Again the reader will appreciate how compactness replaces König's Lemma if she proves Lemma 10, but restricted to fg sets.

Theorem 7.

(1) For any $X \subseteq \mathbf{D}$, $X^* = \{y \mid \exists$ convergent sequence $x_n \in X, z \in X, z \sqsubseteq y \sqsubseteq \lim x_n\}$.

(2) For Scott-compact X, $X^* = Con(Cl(X))$, but not for arbitrary X.

(3) For Scott-compact Y, $X^* \underset{\sim}{\sqsubseteq}_{\mathrm{EM}} Y^*$ iff $X^* \sqsubseteq_{\mathrm{EM}} Y^*$, but not for arbitrary Y.

Proof. (1) Let $Y = \{y \mid \exists$ convergent sequence $\langle x_n \rangle \subseteq X, z \in X . z \sqsubseteq y \sqsubseteq \lim_n x_n\}$. Clearly $Y \subseteq X^*$. For the converse suppose $y \in X^*$. There $y \sqsupseteq$ some z in X and taking $b_0 \sqsubseteq b_1 \sqsubseteq \cdots$ in B_0 with lub y we obtain $\langle x_n \rangle$ in X with $b_n \sqsubseteq x_n$; by compactness we can assume the $\langle x_n \rangle$ converges and clearly then $y \sqsubseteq \lim_n x_n$.

(2) Part (1) shows that $X^* \subseteq Con(Cl(X))$; Lemma 10 and the easily established convexity of X^* shows the converse conclusion. Finally a counterexample for arbitrary X is given by taking $\mathbf{D} = \mathbf{Tapes}$, $X = \{0^n 1^\omega \mid n \in \omega\}$.

(3) Clearly $\sqsubseteq_{\mathrm{EM}}$ is included in $\underset{\sim}{\sqsubseteq}_{\mathrm{EM}}$. Conversely suppose $X^* \underset{\sim}{\sqsubseteq}_{\mathrm{EM}} Y^*$. If $y \in Y^*$ then by Theorem 4, some x in X^* is less than y. If $x \in X^*$, let $b_0 \sqsubseteq b_1 \sqsubseteq \ldots$ be an increasing sequence of finite elements with lub x. Then by Theorem 4 we find y_0, y_1, $\ldots$ in Y^* with $a_n \sqsubseteq y_n$. By compactness we can assume the $\langle y_n \rangle$ converges, say to y. By part (2) we have y in Y^* and clearly $x \sqsubseteq y$. For a counterexample for general Y take $\mathbf{D} = \mathbf{Tapes}$, $Y = \{0^n 1^\omega \mid n \in \omega\}$, $X = Y \cup \{0^\omega\}$. ∎

Proposition 2. *Every Lawson-closed set is Scott-compact.*

Proof. This follows from general topological principles. For since the Lawson topology is compact any Lawson-closed set is Lawson-compact and so compact also in the coarser Scott topology. Alternatively suppose C is Lawson-closed and is covered by $\{x_0, x_1, \ldots\} \subseteq B_{\mathbf{D}}$. If C is not Scott-compact then for any n there is a c_n in C such that $c_n \not\sqsupseteq x_0, \ldots, c_n \not\sqsupseteq x_n$. Taking c as the limit of a convergent subsequence of the c_n we find $c \sqsupseteq$ no x_n, the required contradiction. ∎

Now we know that the nonempty $(\cdot)^*$-closed Scott-compact sets are just the nonempty convex-closed, Lawson closed sets (use Theorem 7(2) and Lemma 9 for one half on the proof and Proposition 2 and Theorem 7(2) for the other half). But putting this and Theorem 7(3) together with the above characterisation of $\mathcal{P}(\mathbf{D})$ we get another one:

$$\mathcal{P}(\mathbf{D}) = \langle \text{nonempty, convex and Lawson-closed sets}, \sqsubseteq_{\mathrm{EM}} \rangle$$

and this is the clearest picture we can obtain of our sets.

As promised we now show that the X^* are finitely generable.

Lemma 11. *Every nonempty Lawson-closed set is fg (and so every $(\cdot)^*$-closed Scott-compact nonempty set is too).*

Proof. Let X be a nonempty Lawson-closed set. Let $b_0, \ldots, b_n, \ldots$ be an enumeration without repetitions of $B_{\mathbf{D}}$ (if $\mathbf{D}$ is finite the whole lemma is trivial). Consider the tree with nodes those nonempty sequences $\langle\langle P_0, Q_0, a_0\rangle, \ldots, \langle P_k, Q_k, a_k\rangle, \ldots, \langle P_m, Q_m, a_m\rangle\rangle$ where $\langle P_k, Q_k\rangle$ is a partition of $\{b_i \mid 0 \le i < k\}$, $a_k \in U(P_k)$ and $a_k \sqsubseteq x$ for some x in X and for all b in Q_k, $b \not\sqsubseteq x$ and furthermore $P_0 \subseteq \cdots \subseteq P_k \subseteq \cdots$ and $Q_0 \subseteq \cdots \subseteq Q_k \subseteq \cdots$ and $a_0 \sqsubseteq \cdots \sqsubseteq a_k \sqsubseteq \cdots$; label such a node by a_m. The nodes are to be ordered by the subsequence ordering (note that the root is $\langle\langle\emptyset, \emptyset, \bot\rangle\rangle$). Now prune off all nodes not dominated by infinitely many nodes. This gives a generating tree for X.

Exercise 78. Prove this! ∎

Exercise 79. Improve previous formulae for 2/3 SFP ω-algebraic $\mathbf{D}$.

(1) Show that for $X, Y \subseteq \mathbf{D}$ we have
$$X \cup_{\mathrm{P}} Y = \mathit{Con}(X \cup Y).$$
Can the formula: $\mathcal{P}(g)(X) = g(X)^*$ for $g\colon \mathbf{D} \to \mathbf{E}$ be improved similarly?

(2) Define the "big union" function $\bigcup_{\mathrm{P}}\colon \mathcal{P}(\mathcal{P}(\mathbf{D})) \to \mathcal{P}(\mathbf{D})$ to be $(id_{\mathcal{P}(\mathbf{D})})^\dagger$. Is it true that for any χ in $\mathcal{P}(\mathcal{P}(\mathbf{D}))$ the formula
$$\bigcup\nolimits_{\mathrm{P}} \chi = \mathit{Con}(\bigcup_{X \in \chi} X)$$
holds?

(3) Let $X_0 \sqsubseteq X_1 \sqsubseteq \cdots$ be an increasing chain in $\mathcal{P}(\mathbf{D})$. Show that
$$\bigsqcup X_n = \{\bigsqcup_n x_n \mid x_n \in X_n,\ x_0 \sqsubseteq x_1 \sqsubseteq \cdots\}^*.$$
Can this be improved to Con, even for X_n finite?

General Powerdomains: The Smyth Powerdomain

Now we consider the general form of the Smyth powerdomain; this will be dealt with rather briefly, leaving many points as exercises. The idea here is that the *more* elements a set contains, the *smaller* it should be in the order — just the dual of the relational case. In terms of semilattices we have the following natural observations:

Exercise 80. Let $\langle \mathbf{D}, \sqsubseteq, \cup\rangle$ be a continuous semilattice. Show that the following statements are equivalent:

(1) $\forall x, y. x \subseteq y \Rightarrow x \sqsupseteq y$,
(2) $\forall x, y. x \subseteq y \Leftrightarrow x \sqsupseteq y$,
(3) $\forall x, y. x \cup y \sqsubseteq x$,
(4) $\forall x, y. x \cup y = x \sqcap y$

and all imply that $x \cup \bot = \bot$ (but not conversely).
Thus we have to do with cpos with a continuous meet operation and the linear maps are the multiplication ones.

As an order on sets one naturally defines the appropriate half of the Egli-Milner order for any cpo $\mathbf{D}$.
$$X \sqsubseteq_{\mathrm{S}} Y \text{ iff } \forall y \in Y. \exists x \in X. x \sqsubseteq y$$
(and the associated equivalence is $=_{\mathrm{S}}$). (Note that X covers C just means that $X \sqsubseteq_{\mathrm{S}} C$.) The contextual variant is
$$X \precsim_{\mathrm{S}} Y \quad \text{iff} \quad \forall f\colon \mathbf{D} \to \mathbb{O}. f(X) \sqsubseteq_{\mathrm{S}} f(Y)$$
(and the associated equivalence is $\simeq_{\mathrm{S}}$). There is no controversy over choice of order since:

Exercise 81. Show that $\sqsubseteq_{\mathrm{S}}$ and $\precsim_{\mathrm{S}}$ are identical. Show that when $\mathbf{D}$ is ω-algebraic, $X \sqsubseteq_{\mathrm{S}} Y$ implies that:
$$\forall A \subseteq B_{\mathbf{D}}. (A \text{ a finite subset of } B_{\mathbf{D}} \wedge A \sqsubseteq_{\mathrm{S}} X) \Rightarrow A \sqsubseteq_{\mathrm{S}} Y$$
and the converse holds if X is finitely generable (or, more generally, Scott compact). [Plunder Theorem 4.]

For a quick construction of the Smyth powerdomain for an ω-algebraic $\mathbf{D}$, put:

$$\mathcal{F}_{\mathrm{S}}(\mathbf{D}) = \langle \{\text{finite nonempty subsets of } B_{\mathbf{D}}\}, \sqsubseteq_{\mathrm{S}} \rangle.$$

This is a preorder with least element $\{\bot\}$. Put

$$\begin{aligned} \mathcal{P}_{\mathrm{S}}(\mathbf{D}) &= \overline{\mathcal{F}_{\mathrm{S}}(\mathbf{D})}, \\ \{d\}_{\mathrm{S}} &= \{\{b_1, \ldots, b_n\} \mid \exists i. b_i \sqsubseteq d\}, \\ I \cup_{\mathrm{S}} J &= \{A \cup B \mid A \in I,\, B \in J\}. \end{aligned}$$

You should verify that $\{\cdot\}_{\mathrm{S}}$ and $\cup_{\mathrm{S}}$ are well-defined and continuous and that $\{b\}_{\mathrm{S}} = [\{b\}]$ for b in $B_{\mathbf{D}}$ and $[A] \cup_{\mathrm{S}} [B] = [A \cup B]$ for A, B in $\mathcal{F}_{\mathrm{S}}(\mathbf{D})$. Clearly too $\langle \mathcal{P}_{\mathrm{S}}(\mathbf{D}), \cup_{\mathrm{S}} \rangle$ is a semilattice and you should verify that $\cup_{\mathrm{S}}$ is the meet operation on the cpo $\mathcal{P}_{\mathrm{S}}(\mathbf{D})$, as $I \cup_{\mathrm{S}} J = I \cap J$.

Exercise 82. Obtain the following universal characterisation of $\mathcal{P}_{\mathrm{S}}(\mathbf{D})$ for ω-algebraic $\mathbf{D}$:

If $f\colon \mathbf{D} \to \mathbf{P}$ is any (strict) continuous function to a cpo $\mathbf{P}$ with a continuous meet then there is a unique (strict) multiplicative continuous function $f^{\dagger}\colon \mathcal{P}_{\mathrm{S}}(\mathbf{D}) \to \mathbf{P}$ such that the following diagram commute:

$$\begin{tikzcd} \mathbf{D} \arrow[d, "\{\cdot\}_{\mathrm{S}}"'] \arrow[dr, "f"] & \\ \mathcal{P}_{\mathrm{S}}(\mathbf{D}) \arrow[r, "f^{\dagger}"'] & \mathbf{P} \end{tikzcd}$$

Show further that $(\cdot)^{\dagger}$ is an isomorphism of the continuous semilattice of (strict) continuous $f\colon \mathbf{D} \to \mathbf{P}$ and the continuous semilattice of (strict) multiplicative continuous $g\colon \mathcal{P}_{\mathrm{S}}(\mathbf{D}) \to \mathbf{P}$. [Plunder Theorem 3.]

This exercise indicates that we can use relations $t \sqsubseteq u$ as well as equalities $t = u$ when constructing free continuous algebras, since the Smyth case depends on the relation $x \cup y \sqsubseteq x$ (note, as a joke, that this is inconsistent with the relational powerdomain requirement that $x \cup y \sqsupseteq x$ as then we would have $x = y$).

Anyway, now we can view $\mathcal{P}_{\mathrm{S}}$ as a locally continuous functor over ω-$\mathcal{ALG}$ putting:

$$\mathcal{P}_{\mathrm{S}}(f) = (\{\cdot\} \circ f)^{\dagger}.$$

The usual arguments then show that $\mathcal{P}_{\mathrm{S}}$ is a functor over $\mathcal{SFP}$ and so we can solve domain equations involving all the functors we know. In fact, however, as long as we don't want $\mathcal{P}_{\mathrm{P}}$ too we can work with consistent completeness as that is preserved by $\mathcal{P}_{\mathrm{S}}$; indeed as the next exercise demonstrates the situation is somewhat curious:

Exercise 83. Let $\mathbf{D}$ be ω-algebraic. Show that $\mathcal{P}_{\mathrm{S}}(\mathbf{D})$ is consistently complete iff $\mathbf{D}$ is 2/3 SFP, demonstrating the formula

$$[A] \cup [B] = [\bigcup_{a \in A} \bigcup_{b \in B} U(\{a, b\})]$$

which should be interpreted as implying that the left-hand-side exists when the set on the right is nonempty.

We now look for nicer description of $\mathcal{P}_{\mathrm{S}}(\mathbf{D})$ as we did for the Egli-Milner case. From now on we fix an ω-algebraic $\mathbf{D}$.

Lemma 12. *Let $f\colon \mathbf{Tapes} \to \mathbf{D}$ generate X. If A is a finite subset of $B_{\mathbf{D}}$ such that $A \sqsubseteq_{\mathrm{S}} X$ then for some n, $A \sqsubseteq_{\mathrm{S}} Bd_n(f)$. Further if $Bd_n(f) \sqsubseteq_{\mathrm{S}} Y$ for all n then $X \sqsubseteq_{\mathrm{S}} Y$ (any $Y \subseteq \mathbf{D}$).*
Proof. Exercise 84. [Plunder Lemma 4.] ∎

Lemma 13. *Let $A_0 \sqsubseteq_{\mathrm{S}} A_1 \sqsubseteq_{\mathrm{S}} \cdots$ be an increasing sequence in $\mathcal{P}_{\mathrm{S}}(\mathbf{D})$. Then there is an $f\colon \mathbf{Tapes} \to \mathbf{D}$ with every A_n being $\sqsubseteq_{\mathrm{S}}$ some $Bd_n(f)$.*
Proof. Exercise 85. [Plunder Lemma 5.] ∎

It follows much as before that

$$\mathcal{P}_{\mathrm{S}}(\mathbf{D}) = \langle \mathcal{F}_g(\mathbf{D}), \sqsubseteq_{\mathrm{S}} \rangle / \sqsubseteq_{\mathrm{S}}$$

and the finite elements have the form $A{\downarrow}$ for A in $\mathcal{P}_{\mathrm{S}}(\mathbf{D})$.

Passing on to canonical representatives for any $X \subseteq \mathbf{D}$ we define:

$$X^\dagger = \{y \mid \exists x \in X. y \sqsupseteq x\}$$

and note that:

(1) $X \subseteq X^\dagger$,
(2) $X^{\dagger\dagger} = X^\dagger$,
(3) $(X \cup Y)^\dagger = X^\dagger \cup Y^\dagger$,
(4) $X =_\mathrm{S} Y$ iff $X \simeq_\mathrm{S} Y$ iff $X^\dagger = Y^\dagger$,
(5) $X \sqsubseteq_\mathrm{S} Y$ iff $X \precsim_\mathrm{S} Y$ iff $X^\dagger \supseteq Y^\dagger$.

Thus $(\cdot)^\dagger$ is a *topological* closure operation (the topology is called the *lower* topology as the open sets are the downwards closed ones). Also $X^\dagger$ is the maximal member of the equivalence class of X and $\sqsubseteq_\mathrm{S}$ is just $\supseteq$ on these maximal members. It would now be nice to know that $X^\dagger$ is fg if X is; however this too is an open problem and we consider the Scott compact subsets again.

Lemma 14. *Let $C \subseteq \mathbf{D}$ be nonempty and Scott-compact. Then $I_\mathrm{S}(C) \stackrel{\text{def}}{=} \{A \in \mathcal{F}_\mathrm{S}(\mathbf{D}) \mid A \sqsubseteq_\mathrm{S} C\}$ is a directed ideal and has $\sqsubseteq_\mathrm{S}$-lub C.*

Proof. Exercise 86. [Plunder Lemma 7.] ∎

Together with Theorem 5 above this establishes that

$$\mathcal{P}_\mathrm{S}(\mathbf{D}) = \langle \text{nonempty upper-closed Scott compact sets}, \supseteq \rangle$$

which is not too bad a characterisation of the powerdomain. When $\mathbf{D}$ is 2/3 SFP we know, of course, that the nonempty upper-closed Scott-compact sets are exactly the nonempty upper-closed, Lawson-closed sets and they in turn are just the upper-closed finitely generable sets.

Exercise 87. Prove these last few assertions!

Exercise 88. Establish the following formulae for $x \in \mathbf{D}$, $X, Y \in \mathcal{P}_\mathrm{S}(\mathbf{D})$, $f: \mathbf{D} \to \mathcal{P}_\mathrm{S}(\mathbf{E})$ and $g: \mathbf{D} \to \mathbf{E}$;

$$\begin{aligned} \{x\}_\mathrm{S} &= \{y \in \mathbf{D} \mid y \sqsupseteq x\}, \\ X \cup_\mathrm{S} Y &= X \cup Y, \\ f^\dagger(X) &= (\bigcup_{x \in X} f(X)). \end{aligned}$$

Show that if $X_0 \supseteq X_1 \supseteq \cdots$ is an increasing chain in $\mathcal{P}_\mathrm{S}(\mathbf{D})$ then:

$$\bigsqcup\nolimits_\mathrm{S} X = \bigcap_n X_n.$$

[*Hint* Consider $Y = \bigsqcup_\mathrm{S} X_n$, which must exist as $\mathcal{P}_\mathrm{S}(\mathbf{D})$ is a cpo.]

Duality

It was realised by M. Smyth that the duality theory connecting Smyth powerdomain and Dijkstra's predicate transformers can be carried far beyond the simple case of discrete cpos to arbitrary ω-algebraic cpos (for example).

Definition 12. Let $\mathbf{D}$ and $\mathbf{E}$ be ω-algebraic. A *predicate transformer* from $\mathbf{E}$ to $\mathbf{D}$ is any strict continuous and multiplicative map

$$p: \mathcal{O}(\mathbf{E}) \to \mathcal{O}(\mathbf{D})$$

Exercise 89. Suppose $m: \mathbf{D} \to \mathcal{P}_\mathrm{S}(\mathbf{E})$. Define the weakest precondition function by putting for any open V of $\mathbf{E}$:

$$wp(m, V) = \{x \in \mathbf{D} \mid m(x) \subseteq V\}.$$

Show that $\Omega(m) \stackrel{\text{def}}{=} \lambda V. wp(m, V)$ is a predicate transformer. [*Hint* Use Scott-compactness.] Show too that Ω is monotonic and that the following properties hold:

$$\begin{aligned} \Omega(\{\cdot\}_\mathrm{S}) &= id, \\ \Omega(m; m') &= \Omega(m) \circ \Omega(m'), \\ \Omega(m \cup_\mathrm{S} m') &= \Omega(m) \cap \Omega(m') \end{aligned}$$

(where, of course, $m; m' = m'^\dagger \circ m$ and the other new operations are defined pointwise).

Exercise 90. Let $\mathcal{ST}(\mathbf{D},\mathbf{E})$ be the partial order $\mathbf{D} \to \mathcal{P}_S(\mathbf{E})$ of continuous functions from $\mathbf{D}$ to $\mathcal{P}_S(\mathbf{E})$. Let $\mathcal{PT}(\mathbf{E},\mathbf{D})$ be the partial order of predicate transformers with the pointwise ordering. Show that

$$\Omega\colon \mathcal{ST}(\mathbf{D},\mathbf{E}) = \mathcal{PT}(\mathbf{E},\mathbf{D})$$

is an isomorphism of partial orders with inverse given by

$$\Omega^{-1}(p)(x) = \bigcap\{V \in \mathcal{O}(\mathbf{E}) \mid x \in p(V)\}.$$

[*Hint* Show first that $\{A^\dagger \mid A \in \mathcal{F}_S(\mathbf{D}) \wedge x \in p(A^\dagger)\}$ is $\sqsubseteq_S$-directed and has lub $\Omega^{-1}(p)(x)$.]

Now show that Ω is an isomorphism of continuous semilattices and can be viewed as a duality of categories:

$$\Omega\colon \mathcal{ST}^{op} \cong \mathcal{PT}.$$

It is to be hoped that this duality will (together with that of Exercise 33 above) lay the foundations for a systematic investigation of the connection between denotational and axiomatic semantics. This however is a substantial programme of investigation which largely remains to be undertaken. This is further underlined by:

Exercise 91. Let $\mathcal{DPT}(\mathbf{E},\mathbf{D})$ be the additive transformers. Show that Ω cuts down to an isomorphism

$$\Omega\colon \omega\text{-}\mathcal{ALG} \cong \mathcal{DPT}$$

of categories.

Exercise 92. The above does not yet generalise the case of a flat cpo since $\mathcal{ST}(X,Y)$ corresponds to $X_\perp \to_\perp \mathcal{P}_S(Y_\perp)$ rather than $X_\perp \to \mathcal{P}_S(Y_\perp)$. So put $\mathcal{ST}_\perp(\mathbf{D},\mathbf{E}) = \mathbf{D} \to_\perp \mathcal{P}_S(\mathbf{E})$ and find an appropriate *strict* version of the above exercises.

One can object that the above treatment of the duality is incomplete since the objects of $\mathcal{PT}$ should really be the complete lattices $\mathcal{O}(\mathbf{D})$ rather than the $\mathbf{D}$ themselves, and we do not yet know what complete lattices we are talking about. We now fill in this gap.

Definition 13. Let P be a partial order. An element x of P is an ω-*prime* (prime) iff whenever $x \sqsubseteq \bigsqcup Y$ where Y is a countable (finite) nonempty set then $x \sqsubseteq$ some element of Y. (Note that $\perp$ is not a prime!) Then P is said to be ω-*prime algebraic* if every element of P is the lub of the ω-primes less than it and there are ω such ω-primes altogether.

Exercise 93. Show that not all ω-algebraic cpos are ω-prime algebraic and neither are all ω-prime algebraic cpos ω-algebraic.

But, for example, all ω-prime algebraic consistently complete cpos are ω-algebraic and then the ω-prime are exactly the finite primes and every finite element is a finite lub of ω-primes. Furthermore the ω-prime algebraic consistently complete cpos are *completely distributive* meaning that:

$$x \sqcap (\bigsqcup Y) = \bigsqcup_{y \in Y} (x \sqcap y)$$

in that when $\bigsqcup Y$ exists then so does the right-hand side and then both sides are equal. Finally any ω-algebraic consistently complete cpo in which every finite element is a lub of primes is ω-prime algebraic.

Exercise 94.

(1) Show that if $\mathbf{D}$ is ω-algebraic then $\mathcal{O}(\mathbf{D})$ is an ω-prime algebraic complete lattice with ω-prime top element.

(2) Let L be an ω-prime algebraic complete lattice with ω-prime top element. Define

$$Pr(L) = \langle \omega\text{-primes of } L, \sqsubseteq \rangle$$

and prove that:

(a) $\mathbf{D} \cong \overline{Pr(\mathcal{O}(\mathbf{D}))^{op}}$,
(b) $L \cong \mathcal{O}(\overline{Pr(L)^{op}})$

(where for any preorder $\langle P, \sqsubseteq \rangle$, P^{op} is $\langle P, \sqsupseteq \rangle$ and the ideal completion of Chapter 6 is intended).

(3) Reformulate Exercise 89 in terms of an *equivalence* between the categories $\mathcal{ST}^{op}$ and the category of ω-prime algebraic complete lattices with ω-prime top elements and with morphisms the strict, continuous, additive functions. (We also call this a duality result.)

Exercise 95. Show that the duality of Exercise 94 cuts down to one of the 2/3 $\mathcal{SFP}$ objects and the ω-prime algebraic, ω-arithmetic complete lattices with ω-prime top element. (An *ω-arithmetic* complete lattice is one which is ω-algebraic and in which the glb of any two finite elements is finite.)

Exercise 96. Show that the duality of Exercise 94 cuts down to one of the consistently complete ω-algebraic cpos and the ω-prime algebraic, complete lattices in which the meet of any two ω-primes is an ω-prime.

We should also mention that much of the above has been inspired by the voluminous

> A Compendium of Continuous Lattices by G. Gierz, J. D. Lawson, K. H. Hofmann, M. Mislove, K. Keimel, D. S. Scott.

which contains a great deal of further potentially computationally relevant material (Open Problem!!).

The Vietoris Topology

As long ago as 1921 Vietoris investigated nondeterministic functions in a topological setting. That this amounts to an alternative treatment of powerdomains was realised by M. Smyth and this includes predicate transformers both of the weakest precondition and (the dual of) the weakest liberal precondition variety. We now indicate some of the connections.

Definition 14.

(1) Let X and Y be sets and suppose $f: X \to \mathcal{P}(Y)$ is a multi-function from X to Y (i.e. from X to the collection of all subsets of Y). Then the *strong inverse* of f is $\Omega(f): \mathcal{P}(Y) \to \mathcal{P}(X)$ where:

$$\Omega(f)(B) = \{a \in X \mid f(a) \subseteq B\}$$

and the *weak inverse* of f is $\Lambda(f): \mathcal{P}(Y) \to \mathcal{P}(X)$ where:

$$\Lambda(f)(B) = \{a \in X \mid f(a) \cap B \neq \emptyset\}.$$

(2) Let X and Y be topological spaces in the above. Then f is *upper semi-continuous* if $\Omega(f)(V)$ is open when V is; it is *lower semi-continuous* if $\Lambda(f)(V)$ is open when V is; it is *continuous* if it is both upper and lower semi-continuous.

Definition 15. Let X be a topological space. We define three topologies on $\mathcal{P}(X)$.

(1) The *upper topology* has basis $\{U_O \mid O \text{ is open in } X\}$ where $U_O \overset{\text{def}}{=} \{A \subseteq X \mid A \subseteq O\}$. Note that $U_O \cap U_{O'} = U_{O \cap O'}$.

(2) The *lower topology* has subbasis $\{L_O \mid O \text{ is open in } X\}$ where $L_O \overset{\text{def}}{=} \{A \sqsubseteq X \mid O \cap A \neq \emptyset\}$.

(3) The *Vietoris topology* is the common refinement of the lower and upper topologies. It has a basis $\{V_{O_1,\ldots,O_n} \mid O_1, \ldots, O_n \text{ are open in } X\} \cup \{\{\emptyset\}\}$ where

$$V_{O_1,\ldots,O_n} = \{A \subseteq X \mid A \subseteq \bigcup_i O_i,\ \forall i. A \cap O_i \neq \emptyset\}$$

(where a *basis* for a topology is a collection of open sets such that every set is a union of basis sets; it need not be closed under intersections).

Exercise 97. Prove the above does define a basis for the Vietoris topology. Show the only topology contained in both the upper and lower topologies is the trivial one.

These topologies *classify* the various kinds of continuity possible for multi-functions in the following sense:

Exercise 98. Show that for a multi-function $f: X \to \mathcal{P}(X)$

(1) f is upper semi-continuous iff it is continuous (in the usual topological sense) relative to the upper topology.

(2) f is lower semi-continuous iff it is continuous relative to the lower topology.

(3) f is continuous iff it is continuous relative to the Vietoris topology.

The standard connection between topologies and preorders is given by:

Definition 16. Let X be a topological space. A preorder is defined on X by putting for all x, y in X:

$$x \sqsubseteq y \quad \text{iff} \quad \forall \text{ open } V.x \in V \Rightarrow y \in V.$$

The preorders defined on $\mathcal{P}(X)$ relative to the upper, lower, Vietoris topologies will be denoted by $\sqsubseteq_{\mathrm{U}}$, $\sqsubseteq_{\mathrm{L}}$, $\sqsubseteq_{\mathrm{V}}$.

Recall that in a cpo we recover the original partial order in the above way from the Scott topology. Now here is the connection between Vietoris' ideas and ours:

Exercise 99. Let $\mathbf{D}$ be a cpo. Let $\sqsubseteq_{\mathrm{S}}$, $\sqsubseteq_{\mathrm{R}}$, $\sqsubseteq_{\mathrm{EM}}$ be defined as above on the collection of $\mathcal{P}(\mathbf{D})$ of all subsets of $\mathbf{D}$ removing any restriction to nonempty sets or ω-algebraic cpos. Regarding $\mathbf{D}$ as also being a topological space via the Scott topology show that:

(1) $\sqsubseteq_{\mathrm{U}} = \sqsubseteq_{\mathrm{S}}$,
(2) $\sqsubseteq_{\mathrm{L}} = \sqsubseteq_{\mathrm{R}}$,
(3) $\sqsubseteq_{\mathrm{V}} = \sqsubseteq_{\mathrm{EM}}$.

Perhaps we should speak of the Egli-Milner-Vietoris ordering!

Just as we did with cpos, one normally works with the various topologies on $\mathcal{P}(X)$ by concentrating on particular kinds of sets e.g. closed or compact. For example metric spaces have been proposed by Nivat and his co-workers as a substitute for cpos e.g.

G. Comyn, M. Dauchet. Approximations of Infinitary Objects. ICALP '82

and de Bakker and Zucker have used Vietoris topologies on compact subsets of metric spaces as a substitute for powerdomain in

Denotational Semantics of Concurrency. SIGACT, 1982.

We now make some connection with our powerdomains. We already know that, for example, $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is the partial order on the Scott-closed subsets of $\mathbf{D}$ derived from the lower topology. So in the sense we can derive our powerdomain from Vietoris' ideas applied to ω-algebraic $\mathbf{D}$'s and a clever choice of $\mathbf{D}$'s. The converse also holds:

Exercise 100. Show that the Scott topology on $\mathcal{P}_{\mathrm{S}}(\mathbf{D})$, $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$, $\mathcal{P}_{\mathrm{P}}(\mathbf{D})$ are just the restrictions (to appropriate kinds of subsets) of the upper, lower and Vietoris topologies.

Exercises

Finally here are some exercises on miscellaneous topics.

Exercise 101 The Empty Set. This is a variation on powerdomains where one asks for an empty set, considering semilattices with a constant $\emptyset$ satisfying the law:

$$\emptyset \cup x = x$$

and considering linear maps preserving $\emptyset$. Clearly in semilattices satisfying the "relational inequality": $x \cup y \sqsupseteq x$, we must have $\emptyset = \bot$ and $\mathcal{P}_{\mathrm{R}}(\mathbf{D})$ is the powerdomain we want, as shown by Theorem 2 restricted to strict functions, noting that here the strict linear maps also preserve $\emptyset$.

(1) **The Plotkin Powerdomain.** The situation here is a little curious. First show that for any ω-algebraic $\mathbf{D} \overset{inro\{\cdot\}_{\mathrm{P}}}{\rightarrow} \{\emptyset\}_\bot + \mathcal{P}_{\mathrm{P}}(\mathbf{D})$ is a strict continuous function such that any strict continuous $f: \mathbf{D} \to \mathbf{A}$ (where $\mathbf{A}$ is a continuous semilattice with $\emptyset$) has a unique strict, linear, continuous extension $f^\dagger: \{\emptyset\}_\bot + \mathcal{P}_{\mathrm{P}}(\mathbf{D}) \to \mathbf{A}$. Second show that there is *no* continuous $\varepsilon: \mathbf{D} \to \mathbf{P}$ (a continuous semilattice with $\emptyset$) such that any other continuous $f: \mathbf{D} \to \mathbf{A}$ (a continuous semilattice with $\emptyset$) has a unique linear continuous extension $f^\dagger: \mathbf{P} \to \mathbf{A}$. [*Hint* Begin by proving that if it exists $\mathbf{P}$ must be $\{\emptyset\}_\bot + \{1\}_\bot$ when $\mathbf{D} = \{1\}_\bot$.] You should also see why the natural candidate $\{\emptyset\}_\bot + \mathcal{P}_{\mathrm{P}}(\mathbf{D})_\bot$ does not work.

(2) **The Smyth Powerdomain.** Here everything goes smoothly. In semilattices satisfying "Smyth inequality": $x \cup y \sqsubseteq x$ we clearly have $\emptyset$ as the top element (since $x = \emptyset \cup x \sqsubseteq \emptyset$). For any cpo $\mathbf{D}$ let $\mathbf{D}_\top$ be $\mathbf{D}$ with a top element added and let $\mathbf{D} \to \mathbf{D}_\top$ be the natural (anonymous) inclusion.

Show that for any ω-algebraic $\mathbf{D} \overset{\{\cdot\}_{\mathrm{S}}}{\rightarrow} \to \mathcal{P}_{\mathrm{S}}(\mathbf{D}) \to \mathcal{P}_{\mathrm{S}}(\mathbf{D})_\top$ is a (strict) continuous function such that any (strict) continuous $f: \mathbf{D} \to \mathbf{A}$ (where $\mathbf{A}$ is a cpo with a continuous meet and a top element) has a unique (strict) continuous, multiplicative, top-preserving extension $f^\dagger: \mathcal{P}_{\mathrm{S}}(\mathbf{D})_\top \to \mathbf{A}$.

Exercise 102 Bistrict Union. This is a variation where we ask for union to be *bistrict* so that

$$x \cup \bot = \bot \cup x = \bot.$$

Clearly this is inconsistent with the relational cases, as then $\bot = x \cup \bot = x$. Also it is implied in the Smyth case as $x \cup \bot = x \sqcap \bot = \bot$. So only the Egli-Milner case remains and variations with and without the empty set are possible.

(1) Show that for any ω-algebraic $\mathbf{D}$ there is a strict continuous $\varepsilon\colon \mathbf{D} \to \mathbf{P}$ (a continuous semilattice with bistrict union) such that any strict continuous $f\colon \mathbf{D} \to \mathbf{A}$ (a continuous semilattice with bistrict union) has a unique strict continuous linear extension $f^\dagger\colon P \to \mathbf{A}$. Note what $\mathbf{P}$ is when $\mathbf{D}$ is flat.

Show that all of this becomes false if the strictness qualifications are omitted.
[*Hint* Consider $\mathbf{D} =$ ¶ and prove that if it exists then $P =$ ¶.]

(2) Show that for any ω-algebraic $\mathbf{D}$ there is a strict continuous $\varepsilon\colon \mathbf{D} \to \mathbf{P}$ (a continuous semilattice with $\emptyset$ and bistrict union) such that any strict continuous $f\colon \mathbf{D} \to \mathbf{A}$ (a continuous lattice with $\emptyset$ and bistrict union) has a unique strict continuous linear extension $f^\dagger\colon P \to \mathbf{A}$. And show all of this becomes false if the strictness condition is removed.

Exercise 103 Multisets — an Open Problem. Investigate multiset constructions $\mathcal{M}(\mathbf{D})$ where the appropriate laws are:

Associativity	$x \cup (y \cup z) = (x \cup y) \cup z,$
Commutativity	$x \cup y = y \cup x$

and one is considering $\mathcal{M}(\mathbf{D})$ as a free abelian semigroup. In this setting the convexity phenomena in powerdomains do not arise and the theory may well be very pleasant. This should be an interesting and straightforward problem.

Exercise 104 Cartesian Products and Multilinear Functions. Show the Cartesian product gives a continuous bilinear (= linear in each argument) function $\times\colon \mathcal{P}_{\mathrm{P}}(\mathbf{D}) \times \mathcal{P}_{\mathrm{P}}(\mathbf{E}) \to \mathcal{P}_{\mathrm{P}}(\mathbf{D} \times \mathbf{E})$ and that satisfies the formula

$$\{d\}_{\mathrm{P}} \times \{e\}_{\mathrm{P}} = \{\langle d, e\rangle\}_{\mathrm{P}}.$$

[*Hint* Establish that the product of fg sets is fg and that the formula $X^* \times Y^* = (X \times Y)^*$ holds for arbitrary sets.] Now show that this classifies the bilinear continuous functions in that to any such $f\colon \mathcal{P}_{\mathrm{P}}(\mathbf{D}) \times \mathcal{P}_{\mathrm{P}}(\mathbf{E}) \to \mathbf{A}$ such that the following diagram commutes:

$$\begin{tikzcd}
\mathcal{P}_{\mathrm{P}}(\mathbf{D}) \times \mathcal{P}_{\mathrm{P}}(\mathbf{E}) \arrow[d, "\times"'] \arrow[dr, "f"] & \\
\mathcal{P}_{\mathrm{P}}(\mathbf{D} \times \mathbf{E}) \arrow[r, "g"'] & \mathbf{A}
\end{tikzcd}$$

Also conclude that any continuous $f\colon \mathbf{D} \times \mathbf{E} \to \mathbf{A}$ has a unique bilinear continuous extension $f^\dagger\colon \mathcal{P}_{\mathrm{P}}(\mathbf{D}) \times \mathcal{P}_{\mathrm{P}}(\mathbf{E}) \to \mathbf{A}$.

Now carry this exercise on the Smyth powerdomain and redo Exercise 34 in the same style.

Exercise 105 Collapsing Morphisms. Find formulae for the natural functions $\alpha_{\mathrm{S}}\colon \mathcal{P}_{\mathrm{P}}(\mathbf{D}) \to \mathcal{P}_{\mathrm{S}}(\mathbf{D})$ and $\alpha_{\mathrm{R}}\colon \mathcal{P}_{\mathrm{P}}(\mathbf{D}) \to \mathcal{P}_{\mathrm{R}}(\mathbf{D})$ (the extensions of $\{\cdot\}_{\mathrm{S}}$, $\{\cdot\}_{\mathrm{R}}$ to strict linear function on $\mathcal{P}_{\mathrm{P}}(\mathbf{D})$). Note that α_{S} and α_{R} are onto. Show that there are no such extensions $\beta_{\mathrm{S}}\colon \mathcal{P}_{\mathrm{S}}(\mathbf{D}) \to \mathcal{P}_{\mathrm{P}}(\mathbf{D})$, $\beta_{\mathrm{R}}\colon \mathcal{P}_{\mathrm{R}}(\mathbf{D}) \to \mathcal{P}_{\mathrm{P}}(\mathbf{D})$ of $\{\cdot\}_{\mathrm{P}}$.

Exercise 106 Computability. Show that $\mathcal{P}_{\mathrm{S}}$ is a computable functor over the effectively given domains of Chapter 7 and show along the way that all associated functions are computable including Cartesian product. The same can be done for $\mathcal{P}_{\mathrm{P}}$ but first one must set up a computability theory for $\mathcal{SFP}$. This has been done by A. Kanda.

¶ Editorial Note: These seem to be intentionally left blank by the author.

Exercise 107 Arbitrary CPOs. It is possible to find powerdomains of *arbitrary* cpos. For example let $\mathbf{D}$ be a cpo. Let P be the preorder given by:

$$P = \langle \text{finite nonempty subsets of } \mathbf{D}, \sqsubseteq_{\mathrm{EM}} \rangle.$$

Then we can take $\mathcal{P}(\mathbf{D}) = \tilde{P}$ the completion keeping existing lubs given in Exercise 7, p. 63, Chapter 6. Show that in this way one obtains the free continuous semilattice over $\mathbf{D}$. This would have provided another approach to powerdomains that would have been much more general but would have been even more abstract than the approach adopted.