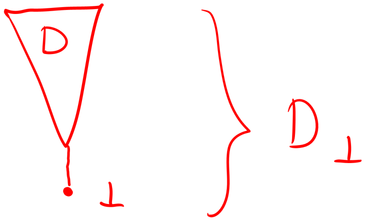
Constructions on domains

Lifting and unlifting

Lift of a cpo D is the domain

$$D_\perp \triangleq D \cup \{\perp\}$$

where $\perp$ is some element not in D and the partially order on $D_\perp$ is $\sqsubseteq_D \cup \{(\perp, x) \mid x \in D_\perp\}$.



Lifting and unlifting

Lift of a cpo D is the domain

$$D_\perp \triangleq D \cup \{\perp\}$$

where $\perp$ is some element not in D and the partially order on $D_\perp$ is $\sqsubseteq_D \cup \{(\perp, x) \mid x \in D_\perp\}$.

Unlift of a domain D is the cpo

$$D_\downarrow \triangleq \{d \in D \mid d \neq \perp\}$$

with partial order as for D.

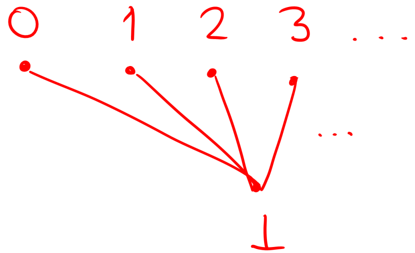
Discrete cpos and flat domains

The discrete cpo on a set S is given by the partial order

$$x \sqsubseteq_S x' \triangleq x = x' \qquad \text{(all } x, x' \in S\text{)}$$

Flat domains $S_\perp$ are the lifts of discrete cpos.

$\mathbb{N}_\perp$:



Products

The product of two cpos $(D_1, \sqsubseteq_1)$ and $(D_2, \sqsubseteq_2)$ has underlying set

$$D_1 \times D_2 = \{(d_1, d_2) \mid d_1 \in D_1 \ \& \ d_2 \in D_2\}$$

and partial order $\sqsubseteq$ defined by

$$(d_1, d_2) \sqsubseteq (d_1', d_2') \ \triangleq \ d_1 \sqsubseteq_1 d_1' \ \& \ d_2 \sqsubseteq_2 d_2'$$

Lubs of chains are calculated componentwise:

$$\bigsqcup_{n \geq 0} (d_{1,n}, d_{2,n}) = (\bigsqcup_{i \geq 0} d_{1,i}, \bigsqcup_{j \geq 0} d_{2,j})$$

Products

The product of two cpos $(D_1, \sqsubseteq_1)$ and $(D_2, \sqsubseteq_2)$ has underlying set

$$D_1 \times D_2 = \{(d_1, d_2) \mid d_1 \in D_1 \;\&\; d_2 \in D_2\}$$

and partial order $\sqsubseteq$ defined by

$$(d_1, d_2) \sqsubseteq (d_1', d_2') \;\triangleq\; d_1 \sqsubseteq_1 d_1' \;\&\; d_2 \sqsubseteq_2 d_2'$$

Lubs of chains are calculated componentwise:

$$\bigsqcup_{n \geq 0} (d_{1,n}, d_{2,n}) = \Big(\bigsqcup_{i \geq 0} d_{1,i}, \bigsqcup_{j \geq 0} d_{2,j}\Big)$$

If $(D_1, \sqsubseteq_1)$ and $(D_2, \sqsubseteq_2)$ are domains so is $(D_1 \times D_2, \sqsubseteq)$ and $\perp_{D_1 \times D_2} = (\perp_{D_1}, \perp_{D_2})$.

Smash product and coalesced sum

Smash product of domains D and E:

$$D \otimes E \triangleq (D_{\downarrow} \times E_{\downarrow})_{\perp}$$

strict continuous functions $D \otimes E \longrightarrow F$

are in bijection with

continuous functions $f : D \times E \to F$

that are <u>strict in each variable separately</u>

$f(\perp, e) = \perp \qquad f(d, \perp) = \perp$

(all $e \in E$, $d \in D$)

Smash product and coalesced sum

Smash product of domains D and E:

$$D \otimes E \triangleq (D_\downarrow \times E_\downarrow)_\perp$$

Coalesced sum of domains D and E:

$$D \oplus E \triangleq (D_\downarrow \uplus E_\downarrow)_\perp$$

(is the coproduct in the category of domains & strict cts fns)

(Disjoint union of two sets X and Y:

$$X \uplus Y \triangleq \{(0, x) \mid x \in X\} \cup \{(1, y) \mid y \in Y\}$$

is the coproduct of X and Y in the category of sets and functions.)

Function cpos and domains

Given cpos $(D, \sqsubseteq_D)$ and $(E, \sqsubseteq_E)$, the function cpo $(D \to E, \sqsubseteq)$ has underlying set

$$D \to E \ \triangleq\ \{f \mid f : D \to E \text{ is a } \textit{continuous} \text{ function}\}$$

and partial order: $f \sqsubseteq f' \ \triangleq\ \forall d \in D \,.\, f(d) \sqsubseteq_E f'(d)$.

Lubs of chains are calculated 'argumentwise' (using lubs in E):

$$(\bigsqcup_{n \geq 0} f_n)(d) = \bigsqcup_{n \geq 0} f_n(d)$$

Function cpos and domains

Given cpos $(D, \sqsubseteq_D)$ and $(E, \sqsubseteq_E)$, the function cpo $(D \to E, \sqsubseteq)$ has underlying set

$$D \to E \triangleq \{f \mid f : D \to E \text{ is a \textit{continuous} function}\}$$

and partial order: $f \sqsubseteq f' \triangleq \forall d \in D \,.\, f(d) \sqsubseteq_E f'(d)$.

Lubs of chains are calculated 'argumentwise' (using lubs in E):

$$(\bigsqcup_{n \geq 0} f_n)(d) = \bigsqcup_{n \geq 0} f_n(d)$$

If E is a domain, then so is $D \to E$: $\perp_{D \to E}$ is the constant function mapping each $d \in D$ to $\perp_E$.

Domain of strict functions

Given domains D and E we get a domain

$$D \multimap E \triangleq \{f \in (D \to E) \mid f(\perp_D) = \perp_E\}$$

with partial order, lubs of chains and least element as for $D \to E$.

Domain equations

$$X \cong \Phi(X)$$

where $\Phi(X)$ is a formal expression built up from the variable X and constants ranging over domains, using the domain constructions $(-)_\perp$, $(-) \times (-)$, $(-) \otimes (-)$, $(-) \oplus (-)$, $(-) \to (-)$ and $(-) \multimap (-)$.

Eg. $\Phi(X) \triangleq (X \to X)_\perp$

or $\Phi(X) \triangleq (\mathbb{Z}_\perp \multimap X) \to (\mathbb{Z}_\perp \otimes (\mathbb{Z}_\perp \multimap X))$

Domain equations

$$X \cong \Phi(X)$$

where $\Phi(X)$ is a formal expression built up from the variable X and constants ranging over domains, using the domain constructions $(-)_\perp$, $(-) \times (-)$, $(-) \otimes (-)$, $(-) \oplus (-)$, $(-) \to (-)$ and $(-) \multimap (-)$.

Aim to show that every domain equation has a solution

$$D \cong \Phi(D)$$

that is minimal is a sense to be explained.

Domain equations

$$X \cong \Phi(X)$$

where $\Phi(X)$ is a formal expression built up from the variable X and constants ranging over domains, using the domain constructions $(-)_\perp$, $(-) \times (-)$, $(-) \otimes (-)$, $(-) \oplus (-)$, $(-) \to (-)$ and $(-) \multimap (-)$.

Aim to show that every domain equation has a solution

$$D \cong \Phi(D)$$

the domain obtained by replacing X in $\Phi(X)$ by D

that is minimal is a sense to be explained.

isomorphism (in the category of domains & cts functions)

Example

Denotational semantics of call-by-name λ-calculus

$$\boxed{\begin{array}{llll}\lambda\text{-Terms} & e \in \Lambda & ::= & x \qquad (x \in \mathbb{V}) \\ & & \mid & \lambda x.e \\ & & \mid & e\,e\end{array}}$$

countably infinite set

Call-by-name evaluation relation $\boxed{e \Rightarrow c}$ between closed terms e, c is inductively generated by

$$\frac{}{\lambda x.e \Rightarrow \lambda x.e} \qquad\qquad \frac{e_1 \Rightarrow \lambda x.e \quad e[e_2/x] \Rightarrow c}{e_1 e_2 \Rightarrow c}$$

substitution

Suppose given $\begin{cases} \text{domain } D \\ \text{isomorphism } i : (D \to D)_\perp \cong D \end{cases}$

Using i, define continuous functions

$$\begin{aligned} fun : (D \to D) &\longrightarrow D \\ f &\longmapsto i(f) \end{aligned}$$

$$\begin{aligned} app : D \times D &\longrightarrow D \\ (d, d') &\longmapsto \begin{cases} i^{-1}(d)\, d' & \text{if } i^{-1}(d) \neq \perp \\ \perp & \text{if } i^{-1}(d) = \perp \end{cases} \end{aligned}$$

Note ~~that~~

$$app(fun(f), d) = i^{-1}(i(f))\, d = f(d)$$

Suppose given $\begin{cases} \text{domain } D \\ \text{isomorphism } i : (D \to D)_\perp \cong D \end{cases}$

Using i, define continuous functions

$$\begin{aligned} fun : (D \to D) &\longrightarrow D \\ f &\longmapsto i(f) \end{aligned}$$

$$\begin{aligned} app : D \times D &\longrightarrow D \\ (d, d') &\longmapsto \begin{cases} i^{-1}(d)\, d' & \text{if } i^{-1}(d) \neq \perp \\ \perp & \text{if } i^{-1}(d) = \perp \end{cases} \end{aligned}$$

Define a domain of environments :

$$Env \triangleq D^{\mathbb{V}} \quad \text{(countable product of } D)$$

Denotation of λ-Terms

$$[\![e]\!]\rho \in D$$

λ-term $e \in \Lambda$

environment $\rho \in D^V$

defined by recursion on the structure of e :

- $[\![x]\!]\rho = \rho(x)$
- $[\![\lambda x.e]\!]\rho = \mathrm{fun}(d \in D \mapsto [\![e]\!](\rho[x \mapsto d]))$
- $[\![e\,e']\!]\rho = \mathrm{app}([\![e]\!]\rho, [\![e']\!]\rho)$

Denotation of λ-Terms

$$[\![e]\!]\rho \in D$$

λ-term $e \in \Lambda$ environment $\rho \in D^V$

defined by recursion on the structure of e:

- $[\![x]\!]\rho = \rho(x)$
- $[\![\lambda x.e]\!]\rho = \mathrm{fun}(d \in D \mapsto [\![e]\!](\rho[x \mapsto d]))$
- $[\![e\,e']\!]\rho = \mathrm{app}([\![e]\!]\rho, [\![e']\!]\rho)$

updated environment, maps x to d and otherwise acts like ρ

E.g. $$[\![\lambda x.x]\!]\rho = \mathrm{fun}(d \mapsto [\![x]\!](\rho[x \mapsto d]))$$
$$= \mathrm{fun}(id_D)$$

and so

$$[\![\lambda y.(\lambda x.x)y]\!]\rho$$
$$= \mathrm{fun}(d \mapsto [\![(\lambda x.x)y]\!](\rho[y \mapsto d]))$$
$$= \mathrm{fun}(d \mapsto \mathrm{app}(\mathrm{fun}(id_D), d))$$
$$= \mathrm{fun}(id_D)$$
$$= [\![\lambda y.y]\!]\rho$$