Denotation of $\lambda$-Terms

$\llbracket e \rrbracket_\rho \in D$

$\lambda$-term $e \in \Lambda$ and environment $\rho \in D^\gamma$

defined by recursion on the structure of $e$:

- $\llbracket x \rrbracket_\rho = \rho(x)$
- $\llbracket \lambda x. e \rrbracket_\rho = \text{fun}(d \in D \mapsto \llbracket e \rrbracket_\rho([x \mapsto d]))$
- $\llbracket ee' \rrbracket_\rho = \text{app}(\llbracket e \rrbracket_\rho, \llbracket e' \rrbracket_\rho)$

updated environment, maps $x$ to $d$ and otherwise acts like $\rho$
Properties of \([\cdot - \cdot]\):

Support

\((\forall x \in f\{e\}, \rho(x) = \rho'(x)) \Rightarrow [e] \rho = [e] \rho'\)
Properties of $\llbracket - \rrbracket$

Support

\[(\forall x \in \text{fv}(e). \ \rho(x) = \rho'(x)) \supset \llbracket e \rrbracket \rho = \llbracket e \rrbracket \rho',\]

(set of free variables of $e$

(proved by induction on structure of $e$)

So for closed expressions ( $\text{fv}(e) = \emptyset$)

$\llbracket e \rrbracket \rho$ is independent of which $\rho$ we use—just write $\llbracket e \rrbracket$ for $\llbracket e \rrbracket \rho$ in this case.
Properties of $\llbracket - \rrbracket$:

**Support**

$(\forall x \in f(v(e)). \rho(x) = \rho'(x)) \Rightarrow \llbracket e \rrbracket \rho = \llbracket e \rrbracket \rho'$

**Compositionality**

$\llbracket e[e'/x] \rrbracket \rho = \llbracket e \rrbracket (\rho[x] \rightarrow \llbracket e' \rrbracket \rho)$

(proved by induction on the structure of $e$, using the support property in case $e = \lambda x.e_1$)
Properties of $\llbracket \cdot \rrbracket$:

**Support**

$(\forall x \in \text{fv}(e). \rho(x) = \rho'(x)) \Rightarrow \llbracket e \rrbracket \rho = \llbracket e \rrbracket \rho'$

**Compositionality**

$\llbracket e[e'/x] \rrbracket \rho = \llbracket e \rrbracket (\rho[x] \rightarrow \llbracket e' \rrbracket \rho)$

**Soundness**

$e \Rightarrow c \Rightarrow \llbracket e \rrbracket = \llbracket c \rrbracket$

proved by induction on the derivation of $e \Rightarrow c$
Eg. induction step for \( e_1 \Rightarrow \lambda x. e \quad e[e_2/x] \Rightarrow c \) \\
\( e_1 e_2 \Rightarrow c \)

If \([e_1]_\rho = [\lambda x. e]_\rho \) \& \([e[e_2/x]]_\rho = [c]_\rho \)
then
\([e_1 e_2]_\rho = \text{app}([e_1]_\rho, [e_2]_\rho)\)
\[= \text{app}([\lambda x. e]_\rho, [e_2]_\rho)\]
\[= \text{app}(\text{fun}(d \mapsto [e]_\rho[x \mapsto d]), [e_2]_\rho)\]
\[= [e]_\rho[x \mapsto [e_2]_\rho]\]
\[= [e[e_2/x]]_\rho\]
\[= [c]_\rho \quad \text{by substitution prop.} \]
N.B. converse of Soundness need not hold

E.g. \[ [\lambda y. (\lambda x.x)y] = [\lambda y. y] \] (See above), but
\[ \lambda y. (\lambda x.x)y \not\equiv \lambda y. y. \]
N.B. converse of Soundness need not hold

However, we can hope for

\[
\llbracket e \rrbracket \neq \bot \subset e \downarrow
\]

and hence \( \llbracket e \rrbracket \neq \bot \equiv e \downarrow \)

since \( e \downarrow \supset \exists c. e \Rightarrow c \)

\( \supset \exists c. \llbracket e \rrbracket = \llbracket c \rrbracket \neq \bot \)

\( \Rightarrow \) denotation of \( c \) is fun(\ldots) \neq \bot
N.B. converse of Soundness need not hold

However, we can hope for

\[ \llbracket e \rrbracket \neq \perp \Rightarrow e \Downarrow \]

This property implies

Computational Adequacy

\[ \llbracket e_1 \rrbracket \subseteq \llbracket e_2 \rrbracket \Rightarrow e_1 \leq_{dx} e_2 \]
N.B. converse of Soundness need not hold

However, we can hope for
\[ [e] \neq \perp \Rightarrow e \Downarrow \]

This property implies

**Computational Adequacy**

\[ [e_1] \models [e_2] \Rightarrow e_1 \leq_{c} e_2 \]

because if \( [e_1] \models [e_2] \), then
\[
e[e_1/z] \Downarrow \Rightarrow [e[e_1/x]] \neq \perp \Rightarrow [e[e_2/x]] = [e][x \rightarrow [e_2]] = \underset{\Downarrow}{[e][x \rightarrow [e_1]]} \\
\Rightarrow e[e_2/z] \Downarrow
\]
N.B. converse of Soundness need not hold

However, we can hope for

\[ \mathcal{L} e \neq \bot \]

This property implies

Computational Adequacy

\[ \mathcal{L} e_1 \supseteq \mathcal{L} e_2 \Rightarrow e_1 \leq e_2 \]

This holds when \( i : (D \rightarrow D)_\bot \equiv D \) is the "minimal invariant" for \( ((-\rightarrow (-))_\bot \)."
Locally continuous functors
Categories of domains

\( \text{Dom} \) = category whose objects are domains (\( \omega \)-chain complete cpos with least elements) and whose morphisms are continuous functions.

\( \text{Dom}_\perp \) = category whose objects are domains and whose morphisms are strict continuous functions.

As usual (in category theory) \( \text{Dom}^{\text{op}} \) is the opposite of \( \text{Dom}_\perp \)—same objects and morphisms given by:

\[ \text{Dom}^{\text{op}}_\perp(D, E) = \text{Dom}_\perp(E, D) \]
Dom\(_\bot\), Dom\(_\bot\)\(^\text{op}\), & Dom\(_\bot\)\(^\text{op}\) \times Dom are examples of a **cpo-enriched category**

- an ordinary category \(\mathcal{C}\), plus
- cpo structure on each hom \(\mathcal{C}(A,B)\) such that composition

\[
\mathcal{C}(A,B) \times \mathcal{C}(B,C) \to \mathcal{C}(A,C)
\]

\((f, g) \mapsto g \circ f\)

is a **continuous** function

---

Functors \(F : \mathcal{C} \to \mathcal{C}'\) are cpo-enriched, or **locally continuous**, if each function

\[
\mathcal{C}(A,B) \to \mathcal{C}'((FA,FB) \mapsto F(f))
\]

is continuous.
All the constructions on domains determine locally continuous functors:

\[
(-)_\perp: \text{Dom}_\perp \to \text{Dom}_\perp
\]

\[f \in D \to E \mapsto f_\perp \in D_\perp \to E_\perp\]

\[f_\perp(x) \triangleq \begin{cases} f(d) & \text{if } x = d \in D \\ \perp & \text{if } x = \perp \end{cases}\]
All the constructions on domains determine locally continuous functors:

\[\Box (-) \times (-) : \text{Dom}_\bot \times \text{Dom}_\bot \to \text{Dom}_\bot\]

\[f_i \in D_i \to E_i, \quad f_2 \in D_2 \to E_2 \quad \mapsto \quad f_1 \times f_2 \in D_1 \times D_2 \to E_1 \times E_2\]

\[(f_1 \times f_2)(d_1, d_2) = (f_1(d_1), f_2(d_2))\]
All the constructions on domains determine locally continuous functors:

\( (-) \otimes (-) : \text{Dom}_\perp \times \text{Dom}_\perp \to \text{Dom}_\perp \)

\[
\begin{align*}
f_1 \in D_1 &\to E_1 \\
f_2 \in D_2 &\to E_2 \\
\mapsto f_1 \otimes f_2 \in D_1 \otimes D_2 &\to E_1 \otimes E_2
\end{align*}
\]

\[
\begin{cases}
(f_1 \otimes f_2)(\perp) = \perp \\
(f_1 \otimes f_2)(d_1, d_2) = (f_1(d_1), f_2(d_2)) &\text{if } f_1(d_1) \neq \perp \& f_2(d_2) \neq \perp \\
\perp &\text{otherwise}
\end{cases}
\]
All the constructions on domains determine locally continuous functors:

\[ (-) \oplus (-) : \text{Dom}_\perp \times \text{Dom}_\perp \to \text{Dom}_\perp \]

\[ f_1 \in D_1 \to E_1, \quad f_2 \in D_2 \to E_2 \quad \mapsto \quad f_1 \oplus f_2 \in D_1 \oplus D_2 \to E_1 \oplus E_2 \]

\[
(f_1 \oplus f_2)(x) \triangleq \begin{cases} 
 f_1(d_1) & \text{if } x = d_1 \in D_1 \downarrow \land f_1(d_1) \neq \bot \\
 f_2 \downarrow d_2 & \text{if } x = d_2 \in D_2 \downarrow \land f_2(d_2) \neq \bot \\
 \bot & \text{otherwise}
\end{cases}
\]
All the constructions on domains determine locally continuous functors:

\[ (-) \to (-) : \text{Dom}_\perp \times \text{Dom}_\perp \to \text{Dom}_\perp \]

\[
\begin{align*}
&f_i \in E_i \to D_i \\
&f_2 \in D_2 \to E_2 \quad \mapsto \quad f_i \to f_2 \in (D_i \to D_2) \to (E_1 \to E_2)
\end{align*}
\]

\[
( f_i \to f_2 )(f) \triangleq f_2 \circ f \circ f_1
\]

(Note that \( f_i \to f_2 \) is a strict function, because...
All the constructions on domains determine locally continuous functors:

\[ \mathit{(-) \to (-)} : \mathbf{Dom}_\bot^\mathcal{O} \times \mathbf{Dom}_\bot \to \mathbf{Dom}_\bot \]

\[ f_1 \in E_1 \to D_1, \quad f_2 \in D_2 \to E_2 \quad \mapsto \quad f_1 \circ f_2 \in (D_1 \to D_2) \to (E_1 \to E_2) \]

\[(f_1 \circ f_2)(f) \triangleq f_2 \circ f \circ f_1 \]
An occurrence of $X$ in $\Phi(X)$ is **negative** if one passes through an **odd** number of left-hand branches of $\rightarrow$ or $\leftarrow$ constructions between the occurrence and the *root* of the parse tree; the occurrence is **positive** otherwise.

E.g.

$$(X \rightarrow X)_\perp$$

$$(X \rightarrow \Xi_\perp) \rightarrow \Xi_\perp$$
Given a domain construction $\mathfrak{D}(x)$, by separating the positive and negative occurrences of $x$, we get a locally continuous functor

$$F: \text{Dom}^\text{op} \times \text{Dom}_\perp \to \text{Dom}_\perp$$

such that $\mathfrak{D}(D) = F(D, D)$ for all $D \in \text{Dom}_\perp$.

E.g. from $\mathfrak{D}(x) = (X \to X)_\perp$ we get

$$F(-, +) = ((-) \to (+))_\perp$$

from $\mathfrak{D}(x) = (X \to \mathbb{Z}_\perp) \to \mathbb{Z}_\perp$

$$F(-, +) = ((+) \to \mathbb{Z}_\perp) \to \mathbb{Z}_\perp$$
Given a domain construction $\Phi(x)$, by separating the & -ve occurrences of $x$, we get a locally continuous functor

$$F: \text{Dom}_D^0 \times \text{Dom}_D \rightarrow \text{Dom}_D$$

such that $\Phi(D) = F(D, D)$ for all $D \in \text{Dom}_D$

solutions $D \cong \Phi(D)$ ↔ "invariants" $D \cong F(D, D)$