Denotational Semantics

12 lectures for Part II CST 2011/12

Andrew Pitts

Course web page:

http://www.cl.cam.ac.uk/teaching/1112/DenotSem/

copies of slides

Styles of formal semantics

Operational. ← IB Sem of PLs

Meanings for program phrases defined in terms of the steps of computation they can take during program execution.

Axiomatic. ← II Hoare Logic

Meanings for program phrases defined indirectly via the axioms and rules of some logic of program properties.

Denotational. ← This course!

Concerned with giving mathematical models of programming languages. Meanings for program phrases defined abstractly as elements of some suitable mathematical structure.

Why do we care?

- Rigour.

  . . . specification of programming languages

  . . . justification of program transformations

- Insight.

  . . . generalisations of notions computability

  . . . higher-order functions

  . . . data structures

- Feedback into language design.

  . . . continuations

  . . . monads

- Reasoning principles.

  . . . Scott induction

  . . . Logical relations

  . . . Co-induction

Basic idea of denotational semantics

$$\text{Syntax} \xrightarrow{[\![-]\!]} \text{Semantics}$$

$$P \mapsto [\![P]\!]$$

Basic idea of denotational semantics

Syntax	$\xrightarrow{[\![-]\!]}$	Semantics
Recursive program	$\mapsto$	Partial recursive function
Boolean circuit	$\mapsto$	Boolean function
P	$\mapsto$	$[\![P]\!]$

Characteristic features of a denotational semantics

- Each phrase (= part of a program), P, is given a denotation, $[\![P]\!]$ — a mathematical object representing the contribution of P to the meaning of *any* complete program in which it occurs.

- The denotation of a phrase is determined just by the denotations of its subphrases (one says that the semantics is compositional).

Basic idea of denotational semantics

$$
\begin{array}{rcl}
\text{Syntax} & \xrightarrow{[\![-]\!]} & \text{Semantics} \\
\text{Recursive program} & \mapsto & \text{Partial recursive function} \\
\text{Boolean circuit} & \mapsto & \text{Boolean function} \\
P & \mapsto & [\![P]\!]
\end{array}
$$

Concerns:

- Abstract models (*i.e.* implementation/machine independent).

  $\leadsto$ 1st 1/3rd of course "domain theory"

- Compositionality.

  $\leadsto$ 2nd 1/3rd of course "PCF"

- Relationship to computation (*e.g.* operational semantics).

  $\leadsto$ last 1/3rd of course

Basic example of denotational semantics (I)

IMP^{-} syntax

Arithmetic expressions

$$A \in \mathbf{Aexp} \quad ::= \quad \underline{n} \;\mid\; L \;\mid\; A + A \;\mid\; \ldots$$

where n ranges over *integers* and

L over a specified set of *locations* $\mathbb{L}$

Boolean expressions

$$\begin{aligned} B \in \mathbf{Bexp} \quad ::= & \quad \mathbf{true} \mid \mathbf{false} \mid A = A \mid \ldots \\ \mid & \quad \neg B \mid \ldots \end{aligned}$$

Commands

$$\begin{aligned} C \in \mathbf{Comm} \quad ::= & \quad \mathbf{skip} \;\mid\; L := A \;\mid\; C ; C \\ \mid & \quad \mathbf{if}\, B \,\mathbf{then}\, C \,\mathbf{else}\, C \end{aligned}$$

Basic example of denotational semantics (II)

Semantic functions

$$\mathcal{A} : \quad \mathbf{Aexp} \rightarrow (State \rightarrow \mathbb{Z})$$

set of all (total) functions from set State to set $\mathbb{Z}$

where

$$\mathbb{Z} \quad = \quad \{\ldots, -1, 0, 1, \ldots\}$$

$$State \quad = \quad (\mathbb{L} \rightarrow \mathbb{Z})$$

Basic example of denotational semantics (II)

Semantic functions

$$\mathcal{A}: \quad \mathbf{Aexp} \to (State \to \mathbb{Z})$$
$$\mathcal{B}: \quad \mathbf{Bexp} \to (State \to \mathbb{B})$$

where

$$\begin{array}{rcl} \mathbb{Z} & = & \{\ldots, -1, 0, 1, \ldots\} \\ \mathbb{B} & = & \{\, true, false \,\} \\ State & = & (\mathbb{L} \to \mathbb{Z}) \end{array}$$

Basic example of denotational semantics (II)

Semantic functions

$$\mathcal{A}: \quad \mathbf{Aexp} \to (State \to \mathbb{Z})$$

$$\mathcal{B}: \quad \mathbf{Bexp} \to (State \to \mathbb{B})$$

$$\mathcal{C}: \quad \mathbf{Comm} \to (State \rightharpoonup State)$$

set of all partial functions from set State to set State

where

$$\begin{aligned} \mathbb{Z} &= \{\ldots, -1, 0, 1, \ldots\} \\ \mathbb{B} &= \{\, true, false \,\} \\ State &= (\mathbb{L} \to \mathbb{Z}) \end{aligned}$$

Partial functions

ordered pairs $\{(x,y) \mid x \in X \wedge y \in Y\}$

i.e. for all $x \in X$ there is at most one $y \in Y$ with $(x,y) \in f$

Definition. A partial function from a set X to a set Y is specified by any subset $f \subseteq X \times Y$ satisfying

$$(x,y) \in f \wedge (x,y') \in f \rightarrow y = y'$$

for all $x \in X$ and $y, y' \in Y$.

Basic example of denotational semantics (III)

Semantic function $\mathcal{A}$

$$\mathcal{A}[\![\underline{n}]\!] = \lambda s \in State.\, n$$

$$\mathcal{A}[\![L]\!] = \lambda s \in State.\, s(L)$$

$$\mathcal{A}[\![A_1 + A_2]\!] = \lambda s \in State.\, \mathcal{A}[\![A_1]\!](s) + \mathcal{A}[\![A_2]\!](s)$$

Basic example of denotational semantics (IV)

Semantic function $\mathcal{B}$

$$\mathcal{B}[\![\mathbf{true}]\!] = \lambda s \in State.\, true$$

$$\mathcal{B}[\![\mathbf{false}]\!] = \lambda s \in State.\, false$$

$$\mathcal{B}[\![A_1 = A_2]\!] = \lambda s \in State.\, eq\big(\mathcal{A}[\![A_1]\!](s), \mathcal{A}[\![A_2]\!](s)\big)$$

where $eq(a, a') = \begin{cases} true & \text{if } a = a' \\ false & \text{if } a \neq a' \end{cases}$

Basic example of denotational semantics (V)

Semantic function $\mathcal{C}$

$$[\![\mathbf{skip}]\!] \quad = \quad \lambda s \in State.\, s$$

NB: From now on the names of semantic functions are omitted!

A simple example of compositionality

Given partial functions $[\![C]\!], [\![C']\!] : \mathit{State} \rightharpoonup \mathit{State}$ and a function $[\![B]\!] : \mathit{State} \to \{\mathit{true}, \mathit{false}\}$, we can define

$$[\![\mathbf{if}\ B\ \mathbf{then}\ C\ \mathbf{else}\ C']\!] = \lambda s \in \mathit{State}.\ \mathit{if}\big([\![B]\!](s), [\![C]\!](s), [\![C']\!](s)\big)$$

where

$$\mathit{if}(b, x, x') = \begin{cases} x & \text{if } b = \mathit{true} \\ x' & \text{if } b = \mathit{false} \end{cases}$$

Basic example of denotational semantics (VI)

Semantic function $\mathcal{C}$

$$[\![L := A]\!] =$$

$$\lambda s \in State.\, \lambda \ell \in \mathbb{L}.\, if\left(\ell = L, [\![A]\!](s), s(\ell)\right)$$

Denotational semantics of sequential composition

Denotation of sequential composition $C; C'$ of two commands

$$[\![C; C']\!] = [\![C']\!] \circ [\![C]\!] = \lambda s \in State.\, [\![C']\!]\big([\![C]\!](s)\big)$$

given by composition of the partial functions from states to states $[\![C]\!], [\![C']\!] : State \rightharpoonup State$ which are the denotations of the commands.

Denotational semantics of sequential composition

Denotation of sequential composition $C; C'$ of two commands

$$[\![C; C']\!] = [\![C']\!] \circ [\![C]\!] = \lambda s \in State.\, [\![C']\!]\big([\![C]\!](s)\big)$$

given by composition of the partial functions from states to states $[\![C]\!], [\![C']\!] : State \rightharpoonup State$ which are the denotations of the commands.

$[\![C']\!]([\![C]\!](s))$ is undefined if

- either $[\![C]\!](s)$ is undefined
- or $[\![C]\!](s) = s'$, say, and $[\![C']\!](s')$ is undefined.

Denotational semantics of sequential composition

Denotation of sequential composition $C; C'$ of two commands

$$[\![C; C']\!] = [\![C']\!] \circ [\![C]\!] = \lambda s \in State.\, [\![C']\!]\big([\![C]\!](s)\big)$$

given by composition of the partial functions from states to states $[\![C]\!], [\![C']\!] : State \rightharpoonup State$ which are the denotations of the commands.

Cf. operational semantics of sequential composition:

$$\frac{C, s \Downarrow s' \quad C', s' \Downarrow s''}{C; C', s \Downarrow s''}\ .$$

$[\![\textbf{while } B \textbf{ do } C]\!]$

Extend the language IMP^- to a language IMP by extending the grammar of commands:

$$C \in Comm ::= \ldots \mid \mathrm{while}\ B\ \mathrm{do}\ C$$

$[\![\textbf{while } B \textbf{ do } C]\!]$

Operational semantics of while-loops

$\langle \text{while } B \text{ do } C, s\rangle \rightarrow$

$\quad \langle \text{if } B \text{ then } C ; (\text{while } B \text{ do } C) \text{ else skip}, s\rangle$

Suggests looking for a denotation $[\![\text{while } B \text{ do } C]\!]$ satisfying

$$[\![\text{while } B \text{ do } C]\!] = [\![\text{if } B \text{ then } C ; (\text{while } B \text{ do } C) \text{ else skip}]\!]$$

Fixed point property of $[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!]$

$$[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!] = f_{[\![B]\!],[\![C]\!]}([\![\mathbf{while}\ B\ \mathbf{do}\ C]\!])$$

where, for each $b : State \to \{true, false\}$ and $c : State \rightharpoonup State$, we define

as
$$f_{b,c} : (State \rightharpoonup State) \to (State \rightharpoonup State)$$

$$f_{b,c} = \lambda w \in (State \rightharpoonup State).\, \lambda s \in State.\\ if\,\big(b(s), w(c(s)), s\big).$$

Fixed point property of $[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!]$

$$[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!] = f_{[\![B]\!],[\![C]\!]}([\![\mathbf{while}\ B\ \mathbf{do}\ C]\!])$$

where, for each $b : \mathit{State} \to \{\mathit{true}, \mathit{false}\}$ and $c : \mathit{State} \rightharpoonup \mathit{State}$, we define

as
$$f_{b,c} : (\mathit{State} \rightharpoonup \mathit{State}) \to (\mathit{State} \rightharpoonup \mathit{State})$$

$$f_{b,c} = \lambda w \in (\mathit{State} \rightharpoonup \mathit{State}).\, \lambda s \in \mathit{State}.\; \mathit{if}\,\big(b(s), w(c(s)), s\big).$$

- Why does $w = f_{[\![B]\!],[\![C]\!]}(w)$ have a solution?
- What if it has several solutions—which one do we take to be $[\![\mathbf{while}\ B\ \mathbf{do}\ C]\!]$?

$[\![\mathbf{while}\ X > 0\ \mathbf{do}\ (Y := X * Y\ ;\ X := X - 1)]\!]$

Let

$$\mathit{State} \stackrel{\text{def}}{=} (\mathbb{L} \to \mathbb{Z})$$ integer assignments to locations

$$D \stackrel{\text{def}}{=} (\mathit{State} \rightharpoonup \mathit{State})$$ partial functions on states

For $[\![\mathbf{while}\ X > 0\ \mathbf{do}\ Y := X * Y\ ;\ X := X - 1]\!] \in D$ we seek a minimal solution to $w = f(w)$, where $f : D \to D$ is defined by:

$$f(w)\big([X \mapsto x, Y \mapsto y]\big)$$

$$= \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w\big([X \mapsto x - 1, Y \mapsto x * y]\big) & \text{if } x > 0. \end{cases}$$

$$D \stackrel{\text{def}}{=} (State \rightharpoonup State)$$

- **Partial order $\sqsubseteq$ on D:**

$w \sqsubseteq w'$ iff for all $s \in State$, if w is defined at s then so is w' and moreover $w(s) = w'(s)$.

iff the graph of w is included in the graph of w'.

- **Least element $\perp \in D$ w.r.t. $\sqsubseteq$:**

$\perp$ = totally undefined partial function

= partial function with empty graph

(satisfies $\perp \sqsubseteq w$, for all $w \in D$).

$f : D \to D$ is given by

$$f(w)[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w[X \mapsto x-1, Y \mapsto x*y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \bot$, $w_1 = f(\bot)$, $w_2 = f(f(\bot))$, etc.

$$w_0[X \mapsto x, Y \mapsto y] = \text{undefined}$$

$f : D \to D$ is given by

$$f(w)[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w[X \mapsto x-1, Y \mapsto x * y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \perp$, $w_1 = f(\perp)$, $w_2 = f(f(\perp))$, etc.

$$w_1[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ \text{undefined} & \text{if } x \geq 1 \end{cases}$$

$f : D \to D$ is given by

$$f(w)[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w[X \mapsto x-1, Y \mapsto x*y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \bot$, $w_1 = f(\bot)$, $w_2 = f(f(\bot))$, etc.

$$w_2[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ [X \mapsto 0, Y \mapsto y] & \text{if } x = 1 \\ \text{undefined} & \text{if } x \geq 2 \end{cases}$$

$f : D \to D$ is given by

$$f(w)[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w[X \mapsto x-1, Y \mapsto x * y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \bot$, $w_1 = f(\bot)$, $w_2 = f(f(\bot))$, etc.

$$w_3[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ [X \mapsto 0, Y \mapsto y] & \text{if } x = 1 \\ [X \mapsto 0, Y \mapsto 2y] & \text{if } x = 2 \\ \text{undefined} & \text{if } x \geq 3 \end{cases}$$

$f : D \to D$ is given by

$$f(w)[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w[X \mapsto x-1, Y \mapsto x*y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \bot$, $w_1 = f(\bot)$, $w_2 = f(f(\bot))$, etc.

$$w_4[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ [X \mapsto 0, Y \mapsto y] & \text{if } x = 1 \\ [X \mapsto 0, Y \mapsto 2y] & \text{if } x = 2 \\ [X \mapsto 0, Y \mapsto 6y] & \text{if } x = 3 \\ \text{undefined} & \text{if } x \geq 4 \end{cases}$$

$f : D \to D$ is given by

$$f(w)\,[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w\,[X \mapsto x-1, Y \mapsto x * y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \bot$, $w_1 = f(\bot)$, $w_2 = f(f(\bot))$, etc.

Union $w_\infty = w_0 \cup w_1 \cup w_2 \cup \cdots$ is the function

$$w_\infty[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ [X \mapsto 0, Y \mapsto !x * y] & \text{if } x > 0 \end{cases}$$

$f : D \to D$ is given by

$$f(w)\,[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w\,[X \mapsto x-1, Y \mapsto x*y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \bot$, $w_1 = f(\bot)$, $w_2 = f(f(\bot))$, etc.

Union $w_\infty = w_0 \cup w_1 \cup w_2 \cup \cdots$ is the function

$$w_\infty[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ [X \mapsto 0, Y \mapsto !x * y] & \text{if } x > 0 \end{cases}$$

It satisfies $w_\infty = f(w_\infty)$ — fixed point we seek for definition of $[\![\text{while } X > 0 \text{ do } (Y := Y * X\, ;\, X := X - 1)]\!]$

$f : D \to D$ is given by

$$f(w)[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ w[X \mapsto x-1, Y \mapsto x*y] & \text{if } x > 0 \end{cases}$$

Want to find $w \in D$ s.t. $w = f(w)$

Define $w_0 = \bot$, $w_1 = f(\bot)$, $w_2 = f(f(\bot))$, etc.

Union $w_\infty = w_0 \cup w_1 \cup w_2 \cup \cdots$ is the function

$$w_\infty[X \mapsto x, Y \mapsto y] = \begin{cases} [X \mapsto x, Y \mapsto y] & \text{if } x \leq 0 \\ [X \mapsto 0, Y \mapsto !x * y] & \text{if } x > 0 \end{cases}$$

It satisfies $w_\infty = f(w_\infty)$ and

$(\forall w)\ w = f(w) \Rightarrow w_\infty \subseteq w$ — w_∞ is a <u>least</u> fixed point for f