

Denotation semantics

William Steingartner

`william.steingartner@tuke.sk`

Department of Computers and Informatics
Faculty of Electrical Engineering and Informatics
Technical University of Košice, Slovakia

Semantics of programming languages

Spring 2022/2023

Theme 06 – Denotational semantics

Denotation semantics

- In the operational approach we were interested in **how** a program is executed.
- In the denotational approach we are merely interested in the **effect** of executing a program, i.e. an association between initial states and final states.
- The main idea is to define a **semantic function** for each **syntactic category**.
- Semantic function maps each **syntactic construct** to a **mathematical object** (often a function), that describes the effect of executing that construct.

Semantic functions are defined compositionally:

- there is a semantic clause for each of the basic elements of the syntactic category, and
- for each method of constructing a composite element (in the syntactic category) there is a semantic clause defined in terms of the semantic function applied to the immediate constituents of the composite element.

Semantic function

The effect of executing a statement S

$$S \in \mathbf{Statm}$$

is to change the state so we shall define the meaning of S to be a partial function on states:

$$\mathcal{S}_{ds} : \mathbf{Statm} \rightarrow (\mathbf{State} \rightarrow \mathbf{State}).$$

Denotations of particular statements we define with **denotation equations**. We define one equation for each alternative in the production rule of an abstract syntax.

Denotation of statements

Denotation of assignment:

$$(1_{ds}) \quad \mathcal{S}_{ds} \llbracket x := e \rrbracket s = s[x \mapsto \mathcal{E} \llbracket e \rrbracket s]$$

Denotation of an empty statement:

$$(2_{ds}) \quad \mathcal{S}_{ds} \llbracket \text{skip} \rrbracket = \text{id}_{\text{State}}$$

Denotation of statements sequence:

$$(3_{ds}) \quad \mathcal{S}_{ds} \llbracket S_1; S_2 \rrbracket = \mathcal{S}_{ds} \llbracket S_2 \rrbracket \circ \mathcal{S}_{ds} \llbracket S_1 \rrbracket$$

defined for an initial state s as follows:

$$\begin{aligned} \mathcal{S}_{ds} \llbracket S_1; S_2 \rrbracket s &= (\mathcal{S}_{ds} \llbracket S_2 \rrbracket \circ \mathcal{S}_{ds} \llbracket S_1 \rrbracket) s \\ &= \begin{cases} s', & \text{if there exists } s'' \text{ such that } \mathcal{S}_{ds} \llbracket S_1 \rrbracket s = s'' \\ & \text{and } \mathcal{S}_{ds} \llbracket S_2 \rrbracket s'' = s', \\ \perp, & \text{if } \mathcal{S}_{ds} \llbracket S_1 \rrbracket s = \perp \text{ or if there exists } s'' \text{ such that} \\ & \mathcal{S}_{ds} \llbracket S_1 \rrbracket s = s'' \text{ but } \mathcal{S}_{ds} \llbracket S_2 \rrbracket s'' = \perp. \end{cases} \end{aligned}$$

Denotation of statements

We define an auxiliary function *cond*:

$$\begin{aligned} \text{cond} : (\mathbf{State} \rightarrow \mathbf{B}) \times (\mathbf{State} \rightarrow \mathbf{State}) \times (\mathbf{State} \rightarrow \mathbf{State}) \\ \rightarrow (\mathbf{State} \rightarrow \mathbf{State}) \end{aligned}$$

defined for $\varphi : \mathbf{State} \rightarrow \mathbf{B}$ and $f_1, f_2 : \mathbf{State} \rightarrow \mathbf{State}$ as follows:

$$\text{cond}(\varphi, f_1, f_2)s = \begin{cases} f_1 s, & \text{if } \varphi s = \mathbf{tt}, \\ f_2 s, & \text{if } \varphi s = \mathbf{ff}. \end{cases}$$

Denotation of statements

Denotation of conditional statement

$$(4_{ds}) \quad \mathcal{S}_{ds}[\text{if } b \text{ then } S_1 \text{ else } S_2] = \text{cond}(\mathcal{B}[b], \mathcal{S}_{ds}[S_1], \mathcal{S}_{ds}[S_2])$$

for an initial state s

$$\begin{aligned} \mathcal{S}_{ds}[\text{if } b \text{ then } S_1 \text{ else } S_2]s &= \text{cond}(\mathcal{B}[b], \mathcal{S}_{ds}[S_1], \mathcal{S}_{ds}[S_2])s \\ &= \begin{cases} s', & \text{if } \mathcal{B}[b]s = \mathbf{tt} \text{ and } \mathcal{S}_{ds}[S_1]s = s', \\ & \text{or if } \mathcal{B}[b]s = \mathbf{ff} \text{ and } \mathcal{S}_{ds}[S_2]s = s', \\ \perp, & \text{if } \mathcal{B}[b]s = \mathbf{tt} \text{ and } \mathcal{S}_{ds}[S_1]s = \perp \\ & \text{or if } \mathcal{B}[b]s = \mathbf{ff} \text{ and } \mathcal{S}_{ds}[S_2]s = \perp. \end{cases} \end{aligned}$$

Denotation of statements

We observe that the **effect of loop**

`while b do S`

must equal that of

`if b then (S; while b do S) else skip`

Using the parts of \mathcal{S}_{ds} that have already been defined, this gives

$$\mathcal{S}_{ds}[\text{while } b \text{ do } S] = \text{cond}(\mathcal{B}[b], \mathcal{S}_{ds}[S] \circ \mathcal{S}_{ds}[\text{while } b \text{ do } S], id) \quad (1)$$

We note that we cannot use the equation above as the definition of the denotation of a loop. However, this equation is **recursive**.

Denotation of statements

The equation (1) expresses that

$$\mathcal{S}_{ds}[\text{while } b \text{ do } S]$$

must be a fixed point of the functional F defined by

$$Fg = \text{cond}(\mathcal{B}[b], g \circ \mathcal{S}_{ds}[\text{while } b \text{ do } S], \text{id}),$$

that is $\mathcal{S}_{ds}[\text{while } b \text{ do } S] = F(\mathcal{S}_{ds}[\text{while } b \text{ do } S])$.

Thus we write:

$$\mathcal{S}_{ds}[\text{while } b \text{ do } S] = \text{fix } F,$$

The functionality of the auxiliary function fix is:

$$\text{fix} : ((\text{State} \rightarrow \text{State}) \rightarrow (\text{State} \rightarrow \text{State})) \rightarrow (\text{State} \rightarrow \text{State})$$

Fixed points

To prepare for a framework that guarantees the existence of the desired fixed point $fix\ F$ we will develop a framework where

- 1 we impose requirements on the fixed points and show that there is at most one fixed point fulfilling these requirements, and
- 2 all functionals originating from statements in *Jane* do have a fixed point that satisfies these requirements.

For that, we introduce:

- 1 ordering on partial functions,
- 2 complete partially ordered sets,
- 3 graph of the function,
- 4 continuous function.

Partial ordering

Definition 1: Let D be a set and \sqsubseteq binary relation on this set, $\sqsubseteq \subseteq D \times D$, which is reflexive, antisymmetric and transitive, i.e. for any $d_1, d_2, d_3 \in D$:

$$d_1 \sqsubseteq d_1 \quad (\text{reflexivity})$$

$$\text{if } d_1 \sqsubseteq d_2 \text{ and } d_2 \sqsubseteq d_1, \text{ then } d_1 = d_2 \quad (\text{antisymmetry})$$

$$\text{if } d_1 \sqsubseteq d_2 \text{ and } d_2 \sqsubseteq d_3, \text{ then } d_1 \sqsubseteq d_3 \quad (\text{transitivity})$$

Such a relation \sqsubseteq we call **partial ordering** and a tuple (D, \sqsubseteq) we call **partially ordered set (poset)**. \square

The **least element** of poset D we denote ω . For all $d \in D$ it holds that $\omega \sqsubseteq d$. If D has the least element then it is **unique**.

Partial ordering

Definition 2: Let (D, \sqsubseteq) be a poset and let Y be a subset of D , $Y \subseteq D$. An **upper bound** of Y is an element $d \in D$ such that for any $d' \in Y$ holds the following:

$$d' \sqsubseteq d.$$

An upper bound d of $Y \subseteq D$ is a **least upper bound** of Y , if and only if d' is an upper bound of Y implies

$$d \sqsubseteq d'.$$

□

If Y has a least upper bound d then d is unique and we denote it $\sqcup Y$.

Partial ordering

Definition 3: Let (D, \sqsubseteq) be a poset. A relation \sqsubseteq we call **relation of linear ordering**, if for any elements $d_1, d_2 \in D$ holds

$$\text{either } d_1 \sqsubseteq d_2 \text{ or } d_2 \sqsubseteq d_1.$$

Subset $Y \subseteq D$ is called **chain** in D it is consistent in the sense that if we take any two elements of Y then one will share its information with the other; formally this is expressed by linear ordering on Y .

A partially ordered set (D, \sqsubseteq) is called a **chain complete** poset (abbreviated ccpo) whenever $\sqcup Y$ exists for all subsets Y of D . □

Definition 4: Let $g : D \rightarrow D'$ be a function. A **graph** g is a set

$$\text{graph}(g) = \{(d, d') \in D \times D' \mid g d = d'\}.$$
□

Continuous functions

Definition 5: Let (D, \sqsubseteq) and (D', \sqsubseteq') be chain complete posets. We say that function $g : D \rightarrow D'$ is **monotone** if and only if for all choices $d_1, d_2 \in D$ holds the following:

$$\text{if } d_1 \sqsubseteq d_2, \text{ then } g d_1 \sqsubseteq' g d_2.$$

□

The composition of two monotone functions is a monotone function.

Definition 6: Let (D, \sqsubseteq) and (D', \sqsubseteq') are chain complete posets and let $g : D \rightarrow D'$ be a monotone function. We say that function g is **continuous** if

$$\sqcup' \{g y \mid y \in Y\} = g(\sqcup Y) \quad (2)$$

holds for all non-empty chains $Y \subseteq D$.

□

If (2) holds for an empty chain, $Y = \emptyset$, that is $\omega = g \omega$ holds, then we shall say that g is **strict**.

Fixed points

Definition 7: Let (D, \sqsubseteq) and (D', \sqsubseteq') be chain closed posets.

A **functional** F

$$F : (D \rightarrow D') \rightarrow (D \rightarrow D')$$

is a function which assigns to monotone function $g : D \rightarrow D'$ the monotone function $F g : D \rightarrow D'$.

A **fixed point** of functional F is such a function $g_0 : D \rightarrow D'$ that

$$F g_0 = g_0$$

holds, i.e. by applying the functional on this function we get the same function as a result. □

A functional F :

- can have no fixed point, or
- can have one or more fixed points.

We are interested in an existence of the **least fixed point** $fix F$.

Fixed points

Theorem 1: Continuous functions on chain closed posets **always** have the least fixed points.

Theorem 2: Let $f : D \rightarrow D$ be a continuous function on the ccpo (D, \sqsubseteq) with the least element \perp . Then

$$\text{fix } f = \sqcup \{f^n \omega \mid n \geq 0\}$$

defines an element of D and this element is the least fixed point of f . A construction is given as follows:

$$\begin{aligned} f^0 &= \text{id}, \\ f^{n+1} &= f \circ f^n, \quad \text{for } n \geq 0. \end{aligned} \tag{3}$$

Denotation of the loop

A functional of partially defined recursive function

$$\mathcal{S}_{ds}[\text{while } b \text{ do } S]: \text{State} \rightarrow \text{State}$$

is the following function

$$F: (\text{State} \rightarrow \text{State}) \rightarrow (\text{State} \rightarrow \text{State})$$

defined for $g: \text{State} \rightarrow \text{State}$

$$Fg = \text{cond}(\mathcal{B}[b], g \circ \mathcal{S}_{ds}[S], \text{id}).$$

Steps necessary for denotation of loop:

- I. we must to define partial ordering \sqsubseteq on the set of partially defined functions $\text{State} \rightarrow \text{State}$ and to prove that the set $(\text{State} \rightarrow \text{State}, \sqsubseteq)$ is a poset with the least element.
- II. we must to prove that $(\text{State} \rightarrow \text{State}, \sqsubseteq)$ is a chain complete poset with the least upper bound.
- III. we must to prove that functional F is a continuous function.

Then we can say that the least fixed point $\text{fix } F$ of the functional F exists and it is the **denotation of loop statements**.

Denotation of the loop

I. We define partial ordering \sqsubseteq of partially defined functions as follows:

Let $g, g' : \mathbf{State} \rightarrow \mathbf{State}$ be partially defined functions. We say that $g \sqsubseteq g'$ if

$$\text{if } g \text{ s} = s' \text{ then } g' \text{ s} = s'$$

holds for any states $s, s' \in \mathbf{State}$.

Lemma 8: If $g, g' : \mathbf{State} \rightarrow \mathbf{State}$ are partially defined functions and \sqsubseteq is ccpo, then

$$g \sqsubseteq g' \quad \text{if and only if } \text{graph}(g) \subseteq \text{graph}(g').$$

(An alternative characterization of ordering)

Lemma 9: A set $(\mathbf{State} \rightarrow \mathbf{State}, \sqsubseteq)$ is a poset with the least element

$$\perp : \mathbf{State} \rightarrow \mathbf{State}.$$

A function \perp s (or $\perp (s)$) send any input $s, s \in \mathbf{State}$, into undefined value as resulting value:

$$\perp \text{ s} = \perp .$$

Denotation of the loop

II. We prove, that $(\mathbf{State} \rightarrow \mathbf{State}, \sqsubseteq)$ is ccpo with the least upper bound.

Lemma 10: A set $(\mathbf{State} \rightarrow \mathbf{State}, \sqsubseteq)$ is ccpo. The equation (property)

$$\mathit{graph}(\sqcup Y) = \bigcup \{\mathit{graph}(g) \mid g \in Y\}$$

holds for the least upper bound $\sqcup Y$ of its chain $Y \subseteq \mathbf{State} \rightarrow \mathbf{State}$.

Denotation of the loop

III. We must to show that F is continuous.

To do so we first observe that

$$F g = F_1(F_2 g)$$

where

$$F_1 g = \text{cond}(\mathcal{B}[b], g, \text{id}) \text{ and}$$

$$F_2 g = g \circ \mathcal{S}_{ds}[S]$$

for $f, g : \mathbf{State} \rightarrow \mathbf{State}$.

The continuity of F we obtain by showing that F_1 and F_2 are continuous.

Lemma 11: Let $f_0 : \mathbf{State} \rightarrow \mathbf{B}$ and $f_1 : \mathbf{State} \rightarrow \mathbf{State}$ are continuous. Then F defined as follows

$$F g = \text{cond}(f_0, g, f_1)$$

is continuous.

Denotation of the loop

Lemma 12: Let $g_0 : \text{State} \rightarrow \text{State}$ be partially defined function, then F defined as

$$F g = g \circ g_0$$

for $g : \text{State} \rightarrow \text{State}$ is continuous.

Denotation of the loop can be now defined by the following denotation equation

$$(5_{ds}) \quad \mathcal{S}_{ds}[\text{while } b \text{ do } S] = \text{fix } F,$$

where the functional F is defined as follows:

$$F g = \text{cond}(\mathcal{B}[b], g \circ \mathcal{S}_{ds}[S], \text{id}),$$

for $g : \text{State} \rightarrow \text{State}$.

Example 1

Example 1: We find a denotation of S in an initial state $s_0 = [x \mapsto \mathbf{3}]$, where

$$S = y := 1; \text{while } \neg(x = 1) \text{ do } (y := y * x; x := x - 1).$$

From (1_{ds}) and (5_{ds}) for an initial state s_0 we have

$$\mathcal{S}_{ds} \llbracket S \rrbracket s_0 = (fix\ F) s_0 [y \mapsto \mathbf{1}],$$

whereby

$$(F\ g)\ s = \begin{cases} g(\mathcal{S}_{ds} \llbracket y := y * x; x := x - 1 \rrbracket s), & \text{if } \mathcal{B} \llbracket \neg(x=1) \rrbracket s = \mathbf{tt}, \\ s, & \text{if } \mathcal{B} \llbracket \neg(x=1) \rrbracket s = \mathbf{ff}. \end{cases}$$

This can be written also as follows:

$$(F\ g)\ s = \begin{cases} g(s[y \mapsto (s\ y) * (s\ x)][x \mapsto (s\ x) - \mathbf{1}]), & \text{if } s\ x \neq \mathbf{1}, \\ s, & \text{if } s\ x = \mathbf{1}. \end{cases}$$

Example 1

We construct functions $F^n \perp$ according to Theorem 2, form. (3):

$$(F^0 \perp)s = \perp,$$

$$(F^1 \perp)s = \begin{cases} s, & \text{if } s x = \mathbf{1}, \\ (F^0 \perp) = \perp, & \text{if } s x \neq \mathbf{1}, \end{cases}$$

$$(F^2 \perp)s = \begin{cases} s, & \text{if } s x = \mathbf{1}, \\ (F^1 \perp)s = \begin{cases} s[y \mapsto (s y) * (s x)][x \mapsto (s x) - \mathbf{1}], & \text{if } s x = \mathbf{2}, \\ \perp, & \text{otherwise.} \end{cases} \end{cases}$$

This means that only if x has value **1** or **2**, then the function $F^2 \perp$ applied on the state s provides concrete value for the variable y .

$$(F^3 \perp)s = \begin{cases} s, & \text{if } s x = \mathbf{1}, \\ s[y \mapsto (s y) * (s x)][x \mapsto (s x) - \mathbf{1}], & \text{if } s x = \mathbf{2}, \\ s[y \mapsto (s y) * (s x) * ((s x) - \mathbf{1})][x \mapsto ((s x) - \mathbf{1}) - \mathbf{1}], & \text{if } s x = \mathbf{3}, \\ \perp, & \text{otherwise.} \end{cases}$$

Example 1

We generalize:

n^{th} function $F^n \perp$ provides values of y for values in variable x from 1 to n . So

$$(F^n \perp)_s = \begin{cases} \perp, & \text{if } s\ x < 1 \text{ or } s\ x > n, \\ s[y \mapsto (s\ y) * s\ x \dots * ((s\ x) - (n - 1))][x \mapsto ((s\ x) - (n - 1))], & \text{if } s\ x \leq n \text{ and } s\ x > 1 \\ s, & \text{if } s\ x = 1 \end{cases}$$

Then

$$(fix\ F)_s = \begin{cases} \perp, & \text{if } s\ x < 1, \\ s[y \mapsto (s\ y) * n \dots * 2][x \mapsto 1], & \text{if } s\ x = n, \\ & n > 1. \end{cases}$$

For an initial state $s_0\ x=3$ we have

$$(fix\ F)(s_0[y \mapsto 1]) = s_0[y \mapsto 1 * 3 * 2][x \mapsto 1] = [y \mapsto 6, x \mapsto 1],$$

where **6** is the resulting value, the factorial of **3**.

□

Example 2

Example 2: We find a denotation of the statement

`while $\neg(x=0)$ do skip`

for an initial state $s \in \mathbf{State}$.

It holds from (5_{ds})

$$\mathcal{S}_{ds}[\text{while } \neg(x=0) \text{ do skip}]s = (\text{fix } F')s$$

whereby

$$(F' g)s = \begin{cases} g s, & \text{if } s.x \neq 0, \\ s, & \text{if } s.x = 0, \end{cases}$$

because $\mathcal{S}_{ds}[\text{skip}]s = s$. That means, every partially defined function $g : \mathbf{State} \rightarrow \mathbf{State}$

$$g s = s$$

is a fixed point of functional F' .

Example 2

This is the situation when the functional has more fixed points. The least fixed point $\text{fix } F'$ is found after construction of functions

$$(F'^0 \perp)s = \perp$$

$$(F'^1 \perp)s = (F'(F'^0 \perp))s = \begin{cases} \perp, & \text{if } s \neq \mathbf{0}, \\ s, & \text{if } s = \mathbf{0}, \end{cases}$$

$$(F'^2 \perp)s = (F'(F'^1 \perp))s = \begin{cases} \perp, & \text{if } s \neq \mathbf{0}, \\ s, & \text{if } s = \mathbf{0}, \end{cases}$$

...

$$(F'^n \perp)s = \begin{cases} \perp, & \text{if } s \neq \mathbf{0}, \\ s, & \text{if } s = \mathbf{0}. \end{cases}$$

So the least fixed point of functional F' is a function

$$\text{fix } F' = g_0$$

defined

$$g_0 s = \begin{cases} \perp, & \text{if } s \neq \mathbf{0}, \\ s, & \text{if } s = \mathbf{0}, \end{cases}$$

and the denotation is

$$\mathcal{S}_{ds} \llbracket \text{while } \neg(x=0) \text{ do skip} \rrbracket = \text{fix } F' = g_0$$

Semantic equivalence

Similarly we can define the denotation of the statement

```
while true do skip.
```

Definition 13: We say that statements S_1 and S_2 are *semantically equivalent according to denotational semantics*, if they have the same denotation, that is

$$\mathcal{I}_{ds}[[S_1]] = \mathcal{I}_{ds}[[S_2]]$$

holds.

