Denotation semantics

William Steingartner

william.steingartner@tuke.sk

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

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 as follows:

- there is a semantic clause for each of the basis 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 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:

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

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

Denotation of statements

Denotation of assignment:

$$(1_{ds}) \qquad \mathscr{S}_{ds}[\![x{:=}e]\!]s = s[x \mapsto \mathscr{E}[\![e]\!]s]$$

Denotation of an empty statement:

$$(2_{ds}) \qquad \mathscr{S}_{ds}[\![\mathtt{skip}]\!] = \mathrm{id}_{\mathbf{State}}$$

Denotation of statements sequence:

$$(3_{ds}) \qquad \mathscr{S}_{ds}[\![S_1; S_2]\!] = \mathscr{S}_{ds}[\![S_2]\!] \circ \mathscr{S}_{ds}[\![S_1]\!]$$

defined for an initial state s as follows:

$$\mathscr{S}_{ds}[\![S_1; S_2]\!]s = (\mathscr{S}_{ds}[\![S_2]\!] \circ \mathscr{S}_{ds}[\![S_1]\!])s$$

$$= \begin{cases} s', & \text{if there exists } s'' \text{ such that } \mathscr{S}_{ds}[\![S_1]\!]s = s'' \\ & \text{and } \mathscr{S}_{ds}[\![S_2]\!]s'' = s'; \\ \\ \perp, & \text{if } \mathscr{S}_{ds}[\![S_1]\!]s = \perp \text{ or if there exists } s'' \text{ such that} \\ & \mathscr{S}_{ds}[\![S_1]\!]s = s'' \text{ but } \mathscr{S}_{ds}[\![S_2]\!]s'' = \perp. \end{cases}$$

Denotation of statements

We define an auxiliary function $cond$ with the following functionality:

$$\begin{aligned} cond : \ & (\mathbf{State} \to \mathbf{B}) \times (\mathbf{State} \rightharpoonup \mathbf{State}) \times (\mathbf{State} \rightharpoonup \mathbf{State}) \\ \to \ & (\mathbf{State} \rightharpoonup \mathbf{State}) \end{aligned}$$

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

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

Denotation of statements

Denotation of conditional statement

$$(4_{ds}) \qquad \mathscr{S}_{ds}[\![\texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2]\!] = cond(\mathscr{B}[\![b]\!], \mathscr{S}_{ds}[\![S_1]\!], \mathscr{S}_{ds}[\![S_2]\!])$$

for an initial state s

$$\mathscr{S}_{ds}[\![\texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2]\!]s = cond(\mathscr{B}[\![b]\!], \mathscr{S}_{ds}[\![S_1]\!], \mathscr{S}_{ds}[\![S_2]\!])s$$

$$= \begin{cases} s', & \text{if } \mathscr{B}[\![b]\!]s = \mathbf{tt} \text{ and } \mathscr{S}_{ds}[\![S_1]\!]s = s', \\ & \text{or if } \mathscr{B}[\![b]\!]s = \mathbf{ff} \text{ and } \mathscr{S}_{ds}[\![S_2]\!]s = s', \\ & \\ \perp, & \text{if } \mathscr{B}[\![b]\!]s = \mathbf{tt} \text{ and } \mathscr{S}_{ds}[\![S_1]\!]s = \perp, \\ & \text{or if } \mathscr{B}[\![b]\!]s = \mathbf{ff} \text{ and } \mathscr{S}_{ds}[\![S_2]\!]s = \perp. \end{cases}$$

Denotation of statements

We observe that the **effect of loop**

$$\texttt{while } b \texttt{ do } S$$

must be the same as that of

$$\texttt{if } b \texttt{ then } (S; \texttt{while } b \texttt{ do } S) \texttt{ else skip}$$

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

$$\mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!] = cond(\mathscr{B}[\![b]\!], \mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!] \circ \mathscr{S}_{ds}[\![S]\!], \mathrm{id}) \tag{1}$$

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

Denotation of statements

The equation (1) expresses that

$$\mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!]$$

must be fixed point of the functional F defined by

$$F g = cond(\mathscr{B}[\![b]\!], g \circ \mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!], \mathrm{id}),$$

that is $\mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!] = F\,(\mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!])$.
Thus we write:

$$\mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!] = \mathrm{fix}\ F,$$

The functionality of the auxiliary function fix is:

$$\mathrm{fix} : ((\mathbf{State} \rightharpoonup \mathbf{State}) \to (\mathbf{State} \rightharpoonup \mathbf{State})) \to (\mathbf{State} \rightharpoonup \mathbf{State})\,.$$

Fixed points

To prepare for a framework that guarantees the existence of the desired fixed point $\mathrm{fix}\ F$ 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 $d_1, d_2, d_3 \in D$:

$$\begin{array}{ll} d_1 \sqsubseteq d_1, & \text{(reflexivity)} \\ \text{if } d_1 \sqsubseteq d_2 \text{ and } d_2 \sqsubseteq d_1, \text{ then } d_1 = d_2, & \text{(antisymmetry)} \\ \text{if } d_1 \sqsubseteq d_2 \text{ and } d_2 \sqsubseteq d_3, \text{ then } d_1 \sqsubseteq d_3. & \text{(transitivity)} \end{array}$$

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

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 the **least upper bound** of Y, if and only if d' is an upper bound of Y implies

$$d \sqsubseteq d''.$$

$\square$

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

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

Subset $Y \subseteq D$ is called **chain** in D if it is consistent in the sense that if we take any two elements of Y then one will share its information with 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. $\square$

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

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

$\square$

Continuous functions

Definition 5: Let $(D, \sqsubseteq)$ and $(D', \sqsubseteq')$ be chain complete posets. We say that function $g : D \to D'$ 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 postes and let $g : D \to D'$ be a monotone function. We say that function g is **continuous** if

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

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

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

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

A **fixed point** of functional F is such a function $g_0 : D \to D'$ 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 about an existence of the **least fixed point** $\mathrm{fix}\ F$, where the clause

if g_0 is a fixed point of the functional F, i.e. $F g_0 = g_0$, then $\mathrm{fix}\ F \sqsubseteq g_0$ holds.

Fixed points

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

Theorem 2: Let $f : D \to D$ 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:

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

Denotation of the loop

A functional of partially defined recursive function

$$\mathscr{S}_{ds}[\![\texttt{while } b \texttt{ do } S]\!] : \ \mathbf{State} \rightharpoonup \mathbf{State}$$

is the following function

$$F : \ (\mathbf{State} \rightharpoonup \mathbf{State}) \to (\mathbf{State} \rightharpoonup \mathbf{State})$$

defined for $g : \mathbf{State} \rightharpoonup \mathbf{State}$

$$F g = cond(\mathscr{B}[\![b]\!], g \circ \mathscr{S}_{ds}[\![S]\!], \mathrm{id}).$$

Steps necessary for denotation of loop:

I. we must to define partial ordering $\sqsubseteq$ on the set of partially defined functions $\mathbf{State} \rightharpoonup \mathbf{State}$ and to prove that the set $(\mathbf{State} \rightharpoonup \mathbf{State}, \sqsubseteq)$ is a poset with the least element.
II. we must to prove that $(\mathbf{State} \rightharpoonup \mathbf{State}, \sqsubseteq)$ 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 $\mathrm{fix}\ F$ 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 $g, g' : \mathbf{State} \rightharpoonup \mathbf{State}$ be partially defined functions. We say that $g \sqsubseteq g'$ if

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

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

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

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

(An alternative characterization of ordering)

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

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

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

$$\perp\ s = \perp$$

Denotation of the loop

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

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

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

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

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

$$\begin{aligned} F_1\ g &= cond(\mathscr{B}[\![b]\!], g, \mathrm{id}) \quad \text{and} \\ F_2\ g &= g \circ \mathscr{S}_{ds}[\![S]\!] \end{aligned}$$

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

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

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

$$F\ g = cond(f_0, g, f_1)$$

is continuous.

Denotation of the loop

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

$$F\ g = g \circ g_0$$

for $g : \mathbf{State} \rightharpoonup \mathbf{State}$ is continuous.

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

$$(5_{ds}) \qquad \mathscr{S}_{ds}[\![\texttt{while}\ b\ \texttt{do}\ S]\!] = \text{fix}\ F,$$

where the functional F is defined as follows:

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

for $g : \mathbf{State} \rightharpoonup \mathbf{State}$.

Example 1

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

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

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

$$\mathscr{S}_{ds}[\![S]\!]s_0 = (\text{fix}\ F)s_0[y \mapsto \mathbf{1}],$$

whereby

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

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 $\mathbf{1}$ or $\mathbf{2}$, 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 $\mathbf{1}$ to $\mathbf{n}$. So

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

Then

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

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

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

where $\mathbf{6}$ is the resulting value, the factorial of $\mathbf{3}$.

□

Example 2

Example 2: We find a denotation of the statement

$$\texttt{while } \neg(\texttt{x}=0) \texttt{ do skip}$$

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

It holds from (5_{ds})

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

whereby

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

because $\mathscr{S}_{ds}[\![\texttt{skip}]\!]s = s$. That means, every partially defined function $g : \mathbf{State} \rightharpoonup \mathbf{State}$

$$g \ s = s$$

is a fixed point of functional F'.

Example 2

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

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

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

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

$$\ldots$$

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

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

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

defined

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

and the denotation is

$$\mathscr{S}_{ds}[\![\texttt{while}\ \neg(\texttt{x}{=}\ 0)\ \texttt{do skip}]\!]{=}\ \mathrm{fix}\ F'{=}\ g_0.$$

Semantic equivalence

Similarly we can define the denotation of the statement

```
while true do skip
```

as

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

$$\mathscr{S}_{ds}[\![S_1]\!] = \mathscr{S}_{ds}[\![S_2]\!]$$

holds.

$\square$