Semantics of Programming Languages

Denotational Semantics

Prof. Peter Müller

Software Component Technology

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

3. Denotational Semantics

Requirements for Well-Definedness

- The denotational semantics is well-defined, if the equations for $\mathcal{S}_{DS}$ define a function
  - Trivial for all equations except for loops

$$\mathcal{S}_{DS}[\![\texttt{while}\ b\ \texttt{do}\ s\ \texttt{end}]\!] = FIX\ F$$
$$\text{where } F(g) = cond(\mathcal{B}[\![b]\!], g \circ \mathcal{S}_{DS}[\![s]\!], id)$$

- For loops, we have to show that F is continuous
  - $(\text{State} \hookrightarrow \text{State}, \sqsubseteq)$ is a ccpo (Lemma 3.4)

Continuity of F

- Observe that

$$F(g) = cond(\mathcal{B}[\![b]\!], g \circ \mathcal{S}_{DS}[\![s]\!], id) = F_1(F_2(g))$$

  where

  - $F_1(g) = cond(\mathcal{B}[\![b]\!], g, id)$
  - $F_2(g) = g \circ \mathcal{S}_{DS}[\![s]\!]$

- By Lemma 3.7, F is continuous if F_1 and F_2 are continuous
- We prove these properties in the following

Continuity of $cond$

Lemma 3.9:

Let g, g_0 : State $\hookrightarrow$ State, p : State $\longrightarrow$ Bool and define

$$F_1(g) = cond(p, g, g_0)$$

Then F_1 is continuous.

We prove:

1. F_1 is monotone
2. F_1 preserves least upper bounds

Proof: Part 1—Monotonicity

- We have to show that $g_1 \sqsubseteq g_2 \Rightarrow F_1(g_1) \sqsubseteq F_1(g_2)$ where $F_1(g) = cond(p, g, g_0)$

- Case 1: $p(\sigma) = tt$
  - We can assume $g_1(\sigma) = \sigma' \Rightarrow g_2(\sigma) = \sigma'$
  - We get $F_1(g_1)\sigma = g_1(\sigma)$ and $F_1(g_2)\sigma = g_2(\sigma)$
  - Therefore, $F_1(g_1)\sigma = \sigma' \Rightarrow F_1(g_2)\sigma = \sigma'$

- Case 2: $p(\sigma) = ff$
  - We get $F_1(g_1) = g_0$ and $F_1(g_2) = g_0$

Proof: Part 2—Continuity

- We have to show that $F_1(\sqcup Y) = \sqcup\{F_1(g)|g \in Y\}$ where $F_1(g) = cond(p, g, g_0)$ and Y is a non-empty chain in State $\hookrightarrow$ State

- From monotonicity of F_1 and Lemma 3.6, we get $\sqcup\{F_1(g)|g \in Y\} \sqsubseteq F_1(\sqcup Y)$
- By anti-symmetry of $\sqsubseteq$, it remains to prove $F_1(\sqcup Y) \sqsubseteq \sqcup\{F_1(g)|g \in Y\}$

Proof: Part 2—Continuity (cont'd)

- Case 1: $p(\sigma) = tt$
  - By the definition of F_1, we get $F_1(\sqcup Y)\sigma = \sqcup Y(\sigma) = \sigma'$ for some σ'
  - Therefore, there is a $g \in Y$ such that $g(\sigma) = \sigma'$ (Lemma 3.4)
  - The definition of F_1 gives $\sqcup\{F_1(g)|g \in Y\}\sigma = \sqcup\{g|g \in Y\}\sigma$
  - $g(\sigma) = \sigma'$ implies $\sqcup\{g|g \in Y\}\sigma = \sigma'$ since the least upper bound summarizes all information
- Case 2: $p(\sigma) = ff$
  - Let $F_1(\sqcup Y)\sigma = g_0(\sigma) = \sigma'$
  - For every $g \in Y$ we get $F_1(g)\sigma = g_0(\sigma) = \sigma'$
  - Since the least upper bound summarizes all information, we get $\sqcup\{F_1(g)|g \in Y\}\sigma = \sigma'$

Continuity of $\circ$

Lemma 3.10:

Let g_0 : State $\hookrightarrow$ State and define

$$F_2(g) = g \circ g_0$$

Then F_2 is continuous.

We prove:

1. F_2 is monotone
2. F_2 preserves least upper bounds

Proof: Part 1—Monotonicity

- We have to show that $g_1 \sqsubseteq g_2 \Rightarrow F_2(g_1) \sqsubseteq F_2(g_2)$ where $F_2(g) = g \circ g_0$

- We can assume $g_1(\sigma) = \sigma' \Rightarrow g_2(\sigma) = \sigma'$
- Let $F_2(g_1)\sigma_0 = g_1(g_0(\sigma_0)) = \sigma'$
- There is a σ such that $g_0(\sigma_0) = \sigma$ and $g_1(\sigma) = \sigma'$
- Consequently, $F_2(g_2)\sigma_0 = g_2(g_0(\sigma_0)) = g_2(\sigma) = \sigma'$

Proof: Part 2—Continuity

- Like for $cond$, we have to show that $F_2(\sqcup Y) \sqsubseteq \sqcup\{F_2(g)|g \in Y\}$ where $F_2(g) = g \circ g_0$ and Y is a non-empty chain in State $\hookrightarrow$ State

$$
\begin{array}{ll}
F_2(\sqcup Y)\sigma = \sigma' \Rightarrow & \text{[Definition of } F_2\text{]} \\
\sqcup Y(g_0(\sigma)) = \sigma' \Rightarrow & \text{[Lemma 3.4]} \\
\exists g' \in Y : g'(g_0(\sigma)) = \sigma' \Rightarrow & \text{[Definition of } F_2\text{]} \\
\exists g' \in Y : F_2(g')\sigma = \sigma' \Rightarrow & \\
\exists g'' \in \{F_2(g)|g \in Y\} : g''(\sigma) = \sigma' \Rightarrow & \text{[Lemma 3.4]} \\
\sqcup\{F_2(g)|g \in Y\}\sigma = \sigma' &
\end{array}
$$

Well-Definedness

Theorem 3.11:

> The semantic equations for the denotational semantics define a total function $\mathcal{S}_{DS}$ in $\mathsf{Stm} \to (\mathsf{State} \hookrightarrow \mathsf{State})$

The proof runs by structural induction on the statement

Proof of Well-Definedness

Induction Base:

- Case `skip`: The function id is well-defined
- Case x`:=`e: A function that maps σ to $\sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma]$ is well-defined

Induction Step:

- Case s_1`;`s_2:
  - By the induction hypothesis, $\mathcal{S}_{DS}[\![s_1]\!]$ and $\mathcal{S}_{DS}[\![s_2]\!]$ are well-defined
  - The composition of two well-defined functions is well-defined

Proof of Well-Definedness (cont'd)

- Case `if` b `then` s_1 `else` s_2 `end`:
  - By the induction hypothesis, $\mathcal{S}_{DS}[\![s_1]\!]$ and $\mathcal{S}_{DS}[\![s_2]\!]$ are well-defined
  - $cond$ preserves well-definedness
- Case `while` b `do` s `end`:
  - By the induction hypothesis, $\mathcal{S}_{DS}[\![s]\!]$ is well-defined
  - $F(g) = F_1(F_2(g))$ where $F_1(g) = cond(\mathcal{B}[\![b]\!], g, id)$ and $F_2(g) = g \circ \mathcal{S}_{DS}[\![s]\!]$
  - F_1 and F_2 are continuous (Lemmas 3.9 and 3.10)
  - Lemma 3.7 gives that F is continuous
  - Theorem 3.8 gives that $FIXF$ is well-defined

Example

- The denotational semantics of the factorial statement

$$\mathcal{S}_{DS}[\![\mathsf{y:=1;while\ x\#1\ do\ y:=y*x;x:=x-1\ end}]\!]$$

$$F(g)\sigma = \begin{cases} g(\mathcal{S}_{DS}[\![\mathsf{y:=y*x;x:=x-1}]\!]) & \text{if } \mathcal{B}[\![\mathsf{x\#1}]\!]\sigma = tt \\ \sigma & \text{if } \mathcal{B}[\![\mathsf{x\#1}]\!]\sigma = ff \end{cases}$$

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

Example: Fixed Point Iteration

$$F^0(\bot)\sigma = \text{undefined}$$

$$F^1(\bot)\sigma = \begin{cases} \text{undefined} & \text{if } \sigma(\mathsf{x}) \neq 1 \\ \sigma & \text{if } \sigma(\mathsf{x}) = 1 \end{cases}$$

$$F^2(\bot)\sigma = \begin{cases} \text{undefined} & \text{if } \sigma(\mathsf{x}) \neq 1 \wedge \sigma(\mathsf{x}) \neq 2 \\ \sigma[\mathsf{y} \mapsto \sigma(\mathsf{y}) * 2][\mathsf{x} \mapsto 1] & \text{if } \sigma(\mathsf{x}) = 2 \\ \sigma & \text{if } \sigma(\mathsf{x}) = 1 \end{cases}$$

- If x is 1 or 2, F^2 gives the correct value for y
- For all other values, F^2 is undefined

Pattern of Fixed Point Iteration

- F^n determines the correct value if it can be computed with at most n unfoldings of the loop

$$F^n(\bot)\sigma = \begin{cases} \text{undefined} & \text{if } \sigma(\mathsf{x}) < 1 \vee \sigma(\mathsf{x}) > n \\ \sigma[\mathsf{y} \mapsto \sigma(\mathsf{y}) * j * \ldots * 2 * 1][\mathsf{x} \mapsto 1] & \\ & \text{if } \sigma(\mathsf{x}) = j \wedge 1 \leq j \leq n \end{cases}$$

- Then we have

$$(FIXF)\sigma = \begin{cases} \text{undefined} & \text{if } \sigma(\mathsf{x}) < 1 \\ \sigma[\mathsf{y} \mapsto \sigma(\mathsf{y}) * n * \ldots * 2 * 1][\mathsf{x} \mapsto 1] & \\ & \text{if } \sigma(\mathsf{x}) = n \wedge n \geq 1 \end{cases}$$

Example (cont'd)

- We apply the semantics of the factorial statement to a state σ_0 where x has the value 3.

- We have to compute $FIXF(\sigma_0[\mathsf{y} \mapsto 1])$

$$(FIXF)\sigma = \begin{cases} \text{undefined} & \text{if } \sigma(\mathsf{x}) < 1 \\ \sigma[\mathsf{y} \mapsto \sigma(\mathsf{y}) * n * \ldots * 2 * 1][\mathsf{x} \mapsto 1] & \\ & \text{if } \sigma(\mathsf{x}) = n \wedge n \geq 1 \end{cases}$$

- In the final state, we get $\sigma[\mathsf{y} \mapsto 3 * 2 * 1][\mathsf{x} \mapsto 1]$

Well-Definedness: Summary

Well-definedness of $\mathcal{S}_{DS}$ relies on the following results

1. The set $\text{State} \hookrightarrow \text{State}$ equipped with an appropriate order $\sqsubseteq$ is a ccpo (Lemmas 3.2 and 3.4)
2. Certain functions

$$\Psi : (\text{State} \hookrightarrow \text{State}) \to (\text{State} \hookrightarrow \text{State})$$

   are continuous (Lemmas 3.9 and 3.10)
3. In the definition of $\mathcal{S}_{DS}$ we only apply the fixed point operation to continuous functions (Theorem 3.11)