

Semantics and Verification of Software

Lecture 13: Denotational Semantics of Procedures

Thomas Noll

Lehrstuhl für Informatik 2
RWTH Aachen University
`noll@cs.rwth-aachen.de`

<http://www-i2.informatik.rwth-aachen.de/i2/svsw/>

Summer semester 2007

1 Denotational Semantics of Blocks and Procedures

- As before: statements denote **storage transformations**
- New: **dependence on environments**

$$\mathfrak{C}[\![\cdot]\!] : Cmd \rightarrow (VEnv \times PEnv \rightarrow (Sto \rightarrow Sto))$$

- Variable environment** obtained as before:

$$VEnv := \{\rho \mid \rho : Var \rightarrow Loc\}$$

- New **declaration functional** takes role of update functions:

$$\mathfrak{D}_v[\![\cdot]\!] : VDec \rightarrow (VEnv \rightarrow VEnv)$$

$$\mathfrak{D}_v[\![\text{var } x; v]\!]\rho := \mathfrak{D}_v[\![v]\!]\rho[x \mapsto \min\{l \in \mathbb{N} \mid \rho(l) = \perp\}]$$

$$\mathfrak{D}_v[\![\varepsilon]\!]\rho := \rho$$

- Procedures now interpreted as **storage transformations**:

$$PDec := \{\pi \mid \pi : PVar \rightarrow (Sto \rightarrow Sto)\}$$

- Recursive procedure calls **involve fixpoints**:

$$\mathfrak{D}_p[\![\cdot]\!] : PDec \rightarrow (VEnv \times PEnv \rightarrow PEnv)$$

$$\mathfrak{D}_p[\![\text{proc } P \text{ is } c; p]\!](\rho, \pi) := \mathfrak{D}_p[\![p]\!](\rho, \pi[P \mapsto \text{fix}(\Phi)])$$

$$\mathfrak{D}_p[\![\varepsilon]\!](\rho, \pi) := \pi$$

where

$$\Phi : (Sto \rightarrow Sto) \rightarrow (Sto \rightarrow Sto)$$

$$\Phi(f) := \mathfrak{C}[\![c]\!](\rho, \pi[P \mapsto f])$$

- Fixpoint approach motivated by equation

$$f = \mathfrak{C}[[c]](\rho, \pi[P \mapsto f])$$

- $(Sto \rightarrow Sto, \sqsubseteq)$ is a **chain-complete partial order**
(cf. Lemma 5.5 for $(\Sigma \rightarrow \Sigma, \sqsubseteq)$)
- Φ is **continuous**
(cf. Lemma 6.6 for $\Phi(f) := \text{cond}(\mathfrak{B}[[b]], f \circ \mathfrak{C}[[c]], \text{id}_\Sigma)$)

$\Rightarrow$ Well-definedness of $\mathfrak{D}_p[[\cdot]]$

- In particular for a non-recursive procedure P (i.e., no P -call in c):

$$\mathfrak{D}_p[[\text{proc } P \text{ is } c]](\rho, \pi) = \pi[P \mapsto \mathfrak{C}[[c]](\rho, \pi)]$$

Definition 13.1 (Denotational semantics of statements)

The (denotational) semantic functional for statements,

$$\mathfrak{C}[\![\cdot]\!] : Cmd \times VEnv \times PEnv \rightarrow (Sto \rightarrow Sto),$$

is given by:

$$\begin{aligned}\mathfrak{C}[\![\text{skip}]\!](\rho, \pi) &:= \text{id}_{Sto} \\ \mathfrak{C}[\![x := a]\!](\rho, \pi)(\sigma) &:= \sigma[\rho(x) \mapsto \mathfrak{A}[\![a]\!](\sigma \circ \rho)] \\ \mathfrak{C}[\![c_1; c_2]\!](\rho, \pi) &:= \mathfrak{C}[\![c_2]\!](\rho, \pi) \circ \mathfrak{C}[\![c_1]\!](\rho, \pi) \\ \mathfrak{C}[\![\text{if } b \text{ then } c_1 \text{ else } c_2]\!](\rho, \pi) &:= \text{cond}(\mathfrak{B}[\![b]\!] \circ \rho, \mathfrak{C}[\![c_1]\!](\rho, \pi), \mathfrak{C}[\![c_2]\!](\rho, \pi)) \\ \mathfrak{C}[\![\text{while } b \text{ do } c]\!](\rho, \pi) &:= \text{fix}(\Phi) \\ \mathfrak{C}[\![\text{call } P]\!](\rho, \pi) &:= \pi(P) \\ \mathfrak{C}[\![\text{begin } v \text{ p } c \text{ end}]\!](\rho, \pi) &:= \mathfrak{C}[\![c]\!](\mathfrak{D}_v[\![v]\!]\rho, \mathfrak{D}_p[\![p]\!](\mathfrak{D}_v[\![v]\!]\rho, \pi))\end{aligned}$$

where

$$\Phi : (Sto \rightarrow Sto) \rightarrow (Sto \rightarrow Sto) : f \mapsto \text{cond}(\mathfrak{B}[\![b]\!] \circ \rho, f \circ \mathfrak{C}[\![c]\!](\rho, \pi), \text{id}_{Sto})$$

Example: Non-Recursive Case

Example 13.2

Let c be given by

```
begin
  var x;           } v
  proc P is y := x; } p
  x := 1;          } c1
  begin            }
    var x;         } c2
    x := 2;
    call P
  end
end.
```

Let $\rho_0 := \rho_\emptyset[y \mapsto 0] \in VEnv$,
 $\rho_1 := \rho_0[x \mapsto 1] \in VEnv$,
 $\pi_1 := \pi_\emptyset[P \mapsto \mathfrak{C}[[y := x]](\rho_1, \pi_\emptyset)]$
 $\in PEnv$.

Then $\mathfrak{D}_v[[v]]\rho_0 = \rho_1$
 $\mathfrak{D}_p[[p]](\rho_1, \pi_\emptyset) = (\rho_1, \pi_1)$

Thus, for every $\sigma \in Sto$,

$$\begin{aligned}\pi_1(P)(\sigma) &= \sigma[\rho_1(y) \mapsto \sigma(\rho_1(x))] \\ &= \sigma[0 \mapsto \sigma(1)].\end{aligned}$$

Altogether, for any $\sigma_0 \in Sto$,

$$\begin{aligned}\mathfrak{C}[[c]](\rho_0, \pi_\emptyset)(\sigma_0) &= \mathfrak{C}[[c_1; c_2]](\rho_1, \pi_1)(\sigma_0) \\ &= \mathfrak{C}[[c_2]](\rho_1, \pi_1)(\mathfrak{C}[[c_1]](\rho_1, \pi_1)(\sigma_0)) \\ &= \mathfrak{C}[[c_2]](\rho_1, \pi_1)(\underbrace{\sigma_0[1 \mapsto 1]}_{=: \sigma_1})\end{aligned}$$

Decomposing c_2 yields

$$\mathfrak{D}_v[[\text{var } x;]]\rho_1 = \underbrace{\rho_1[x \mapsto 2]}_{=: \rho_2}$$

which gives the overall result

$$\begin{aligned}\mathfrak{C}[[c_2]](\rho_1, \pi_1)(\sigma_1) &= \mathfrak{C}[[\text{call } P]](\rho_2, \pi_1)(\underbrace{\sigma_1[2 \mapsto 2]}_{=: \sigma_2}) \\ &= \sigma_2[0 \mapsto \sigma_2(1)] \\ &= \sigma_0[0 \mapsto 1, 1 \mapsto 1, 2 \mapsto 2]\end{aligned}$$

Example 13.3

The semantics of procedure declaration p

```
proc Mult is
  begin
    if x > 0 then z := z+y; x := x-1; call Mult
    else skip
  }c
end
```

with respect to variable environment $\rho_0 := \rho_\emptyset[x \mapsto 0, y \mapsto 1, z \mapsto 2]$ is given by

$$\mathcal{D}_p[p](\rho_0, \pi_\emptyset) = \pi_\emptyset[\text{Mult} \mapsto \text{fix}(\Phi)]$$

where, for every $f : \text{Sto} \rightarrow \text{Sto}$ and $\sigma \in \text{Sto}$,

$$\begin{aligned}\Phi(f)(\sigma) &= \mathcal{C}[c](\rho_0, \pi_\emptyset[\text{Mult} \mapsto f])(\sigma) \\ &= \begin{cases} f(\sigma[2 \mapsto \sigma(2) + \sigma(1), 0 \mapsto \sigma(0) - 1]) & \text{if } \sigma(0) > 0 \\ \sigma & \text{otherwise} \end{cases}\end{aligned}$$

Example 13.3 (continued)

Computation of $\text{fix}(\Phi) = \bigsqcup_{n \in \mathbb{N}} \Phi^n(f_\emptyset)$ by fixpoint iteration (Theorem 7.1):

$$\begin{aligned} f_2(\sigma) &:= \Phi(f_1)(\sigma) \\ f_0(\sigma) &:= f_\emptyset(\sigma) &= \begin{cases} \perp & \text{if } \sigma(0) > 1 \\ \sigma[2 \mapsto \sigma(2) + \sigma(1), 0 \mapsto 0] & \text{if } \sigma(0) = 1 \\ \sigma & \text{if } \sigma(0) \leq 0 \end{cases} \\ f_1(\sigma) &:= \Phi(f_0)(\sigma) &f_3(\sigma) &:= \Phi(f_2)(\sigma) \\ &= \begin{cases} \perp & \text{if } \sigma(0) > 0 \\ \sigma & \text{if } \sigma(0) \leq 0 \end{cases} &= \begin{cases} \perp & \text{if } \sigma(0) > 2 \\ \sigma[2 \mapsto \sigma(2) + 2 * \sigma(1), 0 \mapsto 0] & \text{if } \sigma(0) = 2 \\ \sigma[2 \mapsto \sigma(2) + \sigma(1), 0 \mapsto 0] & \text{if } \sigma(0) = 1 \\ \sigma & \text{if } \sigma(0) \leq 0 \end{cases} \\ &\vdots \\ f_n(\sigma) &= \begin{cases} \perp & \text{if } \sigma(0) \geq n \\ \sigma[2 \mapsto \sigma(2) + \sigma(0) * \sigma(1), 0 \mapsto 0] & \text{if } 0 < \sigma(0) < n \\ \sigma & \text{if } \sigma(0) \leq 0 \end{cases} \end{aligned}$$

In the limit we obtain

$$\text{fix}(\Phi)(\sigma) = \begin{cases} \sigma[2 \mapsto \sigma(2) + \sigma(0) * \sigma(1), 0 \mapsto 0] & \text{if } \sigma(0) > 0 \\ \sigma & \text{if } \sigma(0) \leq 0 \end{cases}$$