Semantics of Programming Languages
Solutions to Assignment 10

Patrick Maier, Jan Schwinghammer
`http://www.ps.uni-sb.de/courses/sem-ws01/`

INFORMATIK

Exercise 10.1 Define $\alpha : L^S \to L^T : f \mapsto g_f$ with $g_f : T \to L : x \mapsto f(x)$.
Let $f \in L^S$ and $g \in L^T$. Then

$$\begin{aligned}
\alpha(f) \sqsubseteq_T g &\Leftrightarrow g_f \sqsubseteq_T g \\
&\Leftrightarrow \forall x \in T : g_f(x) \sqsubseteq g(x) \\
&\Leftrightarrow \forall x \in T : f(x) \sqsubseteq g(x) \\
&\Leftrightarrow \forall x \in S : f(x) \sqsubseteq \begin{cases} g(x) & \text{if } x \in T \\ \top & \text{otherwise} \end{cases} \\
&\Leftrightarrow \forall x \in S : f(x) \sqsubseteq f_g(x) \\
&\Leftrightarrow f \sqsubseteq_S f_g \\
&\Leftrightarrow f \sqsubseteq_S \gamma(g)\ .
\end{aligned}$$

Exercise 10.2 You should have guessed that the program, after running for an infinite amount of time, computes the total function $f : \mathbb{N} \to \mathbb{N}$ with $f(k) = k + 1$ for all $k \in \mathbb{N}$, i. e., the successor function on the natural numbers.

Exercise 10.3 Collecting semantics as system of equations:

$$\begin{aligned}
\mathit{acc}(1) &= \mathit{Store} \\
\mathit{acc}(2) &= \mathit{undefine}_f(\mathit{zero}_n(\mathit{acc}(1))) \\
\mathit{acc}(3) &= \mathit{acc}(2) \cup \mathit{acc}(5) \\
\mathit{acc}(4) &= \mathit{define}_{f(n)}(\mathit{acc}(3)) \\
\mathit{acc}(5) &= \mathit{inc}_n(\mathit{acc}(4))
\end{aligned}$$

where the functions $\mathit{zero}_n, \mathit{inc}_n, \mathit{undefine}_f, \mathit{define}_{f(n)} : 2^{\mathit{Store}} \to 2^{\mathit{Store}}$ are defined as follows:

$$\begin{aligned}
\mathit{zero}_n(S) &= \{s' \in \mathit{Store} \mid \exists s \in S : s'(f) = s(f), s'(n) = 0\} \\
\mathit{inc}_n(S) &= \{s' \in \mathit{Store} \mid \exists s \in S : s'(f) = s(f), s'(n) = s(n) + 1\} \\
\mathit{undefine}_f(S) &= \{s' \in \mathit{Store} \mid \exists s \in S : s'(n) = s(n), \mathit{dom}(s'(f)) = \emptyset\} \\
\mathit{define}_{f(n)}(S) &= \{s' \in \mathit{Store} \mid \exists s \in S : s'(n) = s(n), \mathit{dom}(s'(f)) = \mathit{dom}(s(f)) \cup \{s(n)\}, \\
&\qquad s'(f)(s(n)) = s(n) + 1, \\
&\qquad \forall k \in \mathit{dom}(s(f)) \setminus \{s(n)\} : s'(f)(k) = s(f)(k)\}
\end{aligned}$$

Note that all these functions are monotone.

Exercise 10.4 $F : (2^{Store})^{PP} \to (2^{Store})^{PP}$ is defined as $F(d) = \langle f_1(d), \ldots, f_5(d)\rangle$ where the $f_i : (2^{Store})^{PP} \to 2^{Store}$ are defined as follows:

$$\begin{aligned}
f_1(d) &= Store \\
f_2(d) &= undefine_f(zero_n(d(1))) \\
f_3(d) &= d(2) \cup d(5) \\
f_4(d) &= define_{f(n)}(d(3)) \\
f_5(d) &= inc_n(d(4))
\end{aligned}$$

The f_i are monotone, so F is monotone, so $acc = \operatorname{lfp} F$ exists, and here it is:

$acc(1) = Store$
$acc(2) = \{s \in Store \mid s(n) = 0, dom(s(f)) = \emptyset\}$
$acc(3) = \{s \in Store \mid dom(s(f)) = \{0, \ldots, s(n) - 1\}, \forall k \in dom(s(f)) : s(f)(k) = k + 1\}$
$acc(4) = \{s \in Store \mid dom(s(f)) = \{0, \ldots, s(n)\}, \forall k \in dom(s(f)) : s(f)(k) = k + 1\}$
$acc(5) = \{s \in Store \mid s(n) > 0, dom(s(f)) = \{0, \ldots, s(n) - 1\}, \forall k \in dom(s(f)) : s(f)(k) = k + 1\}$

The collecting semantics does not yield the total successor function on $\mathbb{N}$ but it yields all partial successor functions whose domain is $\{0, \ldots, k\}$ for $k \in \mathbb{N}$.

Exercise 10.5 Define $\gamma : D^\# \to D$ as $\gamma(M) = \{s \in Store \mid M \subseteq dom(s(f))\}$.
Let $S \in D$ and $M \in D^\#$. Then

$$\begin{aligned}
\alpha(S) \sqsubseteq M \Leftrightarrow M \subseteq \alpha(S) \Leftrightarrow M &\subseteq \bigcap\{dom(s(f)) \mid s \in S\} \\
&\Leftrightarrow \forall s \in S : M \subseteq dom(s(f)) \\
&\Leftrightarrow S \subseteq \{s \in Store \mid M \subseteq dom(s(f))\} \Leftrightarrow S \subseteq \gamma(M) \ .
\end{aligned}$$

Hence $\langle D, \subseteq\rangle \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \langle D^\#, \sqsubseteq\rangle$.

Exercise 10.6 Best approximations are:

$$\begin{aligned}
f_1^\#(d) &= \alpha(Store) &&= \emptyset \\
f_2^\#(d) &= \alpha(undefine_f(zero_n(\gamma(d(1))))) &&= undefine_f^\#(zero_n^\#(d(1))) \\
f_3^\#(d) &= \alpha(\gamma(d(2)) \cup \gamma(d(5))) &&= d(2) \sqcup d(5) = d(2) \cap d(5) \\
f_4^\#(d) &= \alpha(define_{f(n)}(\gamma(d(3)))) &&= define_{f(n)}^\#(d(3)) \\
f_5^\#(d) &= \alpha(inc_n(\gamma(d(4)))) &&= inc_n^\#(d(4))
\end{aligned}$$

where the functions $zero_n^\#, inc_n^\#, undefine_f^\#, define_{f(n)}^\# : D^\# \to D^\#$ are as follows:

$$\begin{aligned}
zero_n^\# &= \alpha \circ zero_n \circ \gamma &&= \mathrm{id} \\
inc_n^\# &= \alpha \circ inc_n \circ \gamma &&= \mathrm{id} \\
undefine_f^\# &= \alpha \circ undefine_f \circ \gamma &&= \lambda M.\emptyset \\
define_{f(n)}^\# &= \alpha \circ define_{f(n)} \circ \gamma &&= \mathrm{id}
\end{aligned}$$

The equations for $zero^{\#}_n$, $inc^{\#}_n$ and $undefine^{\#}_f$ are obvious. To see why $define^{\#}_{f(n)} = \mathrm{id}$ note that for every $M \in D^{\#}$ and $s \in \gamma(M)$, the counter n can have any value in $\mathbb{N}$, independent of the domain of the partial function f. Therefore, defining $f(n)$ need not strictly increase the domain of f.

Now, the least fixpoint of the continuous (check this!) function$F^{\#} : {D^{\#}}^{PP} \to {D^{\#}}^{PP} : d \mapsto F^{\#}(d) = \langle f^{\#}_1(d), \ldots, f^{\#}_1(d) \rangle$ is easily computed as $\mathrm{lfp}\, F^{\#} = \langle \emptyset, \ldots, \emptyset \rangle \in {D^{\#}}^{PP}$. We get no information for which natural numbers the program defines f.

Exercise 10.7 Unfolding the loop once means inserting the assignments

$$f := \lambda k.\text{if } k = n \text{ then } n + 1 \text{ else } f(k);\ n := n + 1$$

right before program point 2. This changes the collecting semantics *acc* in such a way that from program point $p \geq 2$ on, $s(f)(0) = 1$ in all stores $s \in acc(p)$. Accordingly, $f^{\#}_2 = \lambda d.\{0\}$ and therefore $\mathrm{lfp}\, F^{\#} = \langle \emptyset, \{0\}, \ldots, \{0\} \rangle$.

In general, unfolding the loop m times has the effect that at program point 2, f is already defined at $\{0, \ldots, m-1\}$. Therfore, $f^{\#}_2 = \lambda d.\{0, \ldots, m-1\}$ and thus $\mathrm{lfp}\, F^{\#} = \langle \emptyset, \{0, \ldots, m-1\}, \ldots, \{0, \ldots, m-1\} \rangle$. Obviously, the analysis for which natural number the program defines f is the more precise (i. e., $\mathrm{lfp}\, F^{\#}$ is the smaller) the larger m is.