Scribe: based on notes by Reba Schuller and Yuval Gabay Lecturer: Andrew Myers

1 Review

Goal: We want to define $\mathcal{C}[\![\textbf{while}\, b\, \textbf{do}\, c]\!]$ to be the least fixed point of F where

$$F = \lambda f \in \Sigma \to \Sigma_\perp.\, \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \mathit{strict}(f, \mathcal{C}[\![c]\!]\sigma)$$

where we define *strict* as follows:

$$\mathit{strict} \stackrel{def}{=} \lambda f \in \Sigma \to \Sigma_\perp, \bar{\sigma} \in \Sigma_\perp.\, \mathit{if}\ \bar{\sigma} = \perp\ \mathit{then}\ \perp\ \mathit{else}\ f(\bar{\sigma})$$

Since the functions $F^n(\perp)$ approximate the desired result arbitrarily closely, we suspect that the following denotation might be what we want:

$$\mathcal{C}[\![\textbf{while}\, b\, \textbf{do}\, c]\!] = \bigsqcup_{n\in\omega} F^n(\perp)$$

Problems:

- The least upper bound might not exist. $\Sigma \to \Sigma_\perp$ is a cpo, which ensures that every chain has a LUB. What if the functions $F^n(\perp)$ do not form a chain?
- Assuming the LUB exists, it might not be a fixed point of F, let alone the least one.

Solution: We introduce some conditions under which the LUB exists and equals the least fixed point, and then we show that these conditions hold for our F.

2 Monotonicity and Continuity

Definition: Let $(D, \sqsubseteq)$ be a cpo, $F : D \to D$ a function. F is *monotonic* if

$$\forall x, y \in D \quad x \sqsubseteq y \to F(x) \sqsubseteq F(y).$$

Claim: If $(D, \sqsubseteq, \perp)$ is a pointed cpo and $F : D \to D$ is monotonic then the elements $F^n(\perp)$ form an increasing chain in D:

$$\perp \sqsubseteq F(\perp) \sqsubseteq F^2(\perp) \sqsubseteq \ldots$$

Proof: Since $\perp$ is the least element of D, we have

$$\perp \sqsubseteq F(\perp).$$

Monotonicity of F gives

$$\forall n \in \omega \quad F^n(\perp) \sqsubseteq F^{n+1}(\perp) \Rightarrow F^{n+1}(\perp) \sqsubseteq F^{n+2}(\perp).$$

The result follows by induction.

Notice that if $F : D \to D$ is monotonic and $x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \ldots$ is a chain in D, then $F(x_0) \sqsubseteq F(x_1) \sqsubseteq F(x_2) \sqsubseteq \ldots$ is also a chain in D. This permits the following definition.

Definition: Let $(D, \sqsubseteq)$ be a cpo, $F : D \to D$ a monotonic function. F is *continuous* if for every chain

$$x_0 \sqsubseteq x_1 \sqsubseteq x_2 \sqsubseteq \ldots$$

in D, F preserves the LUB operator:

$$\bigsqcup_{n \in \omega} F(x_n) \quad = \quad F(\bigsqcup_{n \in \omega} x_n).$$

3 The Fixed Point Theorem

We will now show that the properties of monotonicity and continuity allow us to compute the least fixed point as desired.

Claim: Let $(D, \sqsubseteq)$ be a cpo, and let $F : D \to D$ be a monotonic continuous function. Then $\bigsqcup_{n \in \omega} F^n(\bot)$ is a fixed point of F.

Proof: By continuity of F,

$$F(\bigsqcup_{n \in \omega} F^n(\bot)) = \bigsqcup_{n \in \omega} F(F^n(\bot))$$

Applying F,

$$= \bigsqcup_{n \in \omega} F^{n+1}(\bot)$$

Reindexing,

$$= \bigsqcup_{n=1,2,\ldots} F^n(\bot)$$

By definition of $\bot$,

$$= \bot \sqcup \bigsqcup_{n=1,2,\ldots} F^n(\bot)$$

And, finally, absorbing the join with $\bot$ into the big join,

$$= \bigsqcup_{n \in \omega} F^n(\bot)$$

We now know that monotonicity and continuity guarantee that $\bigsqcup_{n \in \omega} F^n(\bot)$ is a fixed point of F. We also want $\bigsqcup_{n \in \omega} F^n(\bot)$ to be the *least* fixed point of F. To show this, we must prove that $y = F(y) \Rightarrow \bigsqcup_{n \in \omega} F^n(\bot) \sqsubseteq y$. We can actually prove something even stronger.

Definition: Let $(D, \sqsubseteq)$ be a cpo, $F : D \to D$ a function. $x \in D$ is a *prefixed* point of F if $F(x) \sqsubseteq x$.

Notice that every fixed point of F is also a prefixed point. As a consequence, if a fixed point of F is the least prefixed point of F, it is also the least fixed point of F.

Claim: Let $(D, \sqsubseteq, \bot)$ be a pointed cpo. For any monotonic continuous function, $F : D \to D$, $\bigsqcup_{n \in \omega} F^n$ is the least prefixed point of F.

Proof: Suppose y is a prefixed point of F. By definition of $\bot$,

$$\bot \sqsubseteq y$$

Taking F of both sides,

$$F(\bot) \sqsubseteq F(y) \sqsubseteq y$$

Inductively, for all $n \geq 0$,

$$F^n(\bot) \sqsubseteq y$$

Because y is an upper bound for all the $F^n(\bot)$, it must be at least as large as their least upper bound:

$$\bigsqcup_{n \in \omega} F^n(\bot) \sqsubseteq y$$

We have now proven:

The Fixed Point Theorem: Let $(D, \sqsubseteq, \bot)$ be a pointed cpo. For any monotonic continuous function, $F : D \to D$, $\bigsqcup_{n \in \omega} F^n$ is the least fixed point of F.

We have actually encountered an instance of the fixed point theorem before. Recall lecture 6, when we defined the set of all elements derivable in some rule system to be the least fixed point of the rule operator, R. Our proof in that case was an instantiation of the fixed point theorem on the CPO consisting of all subsets of a set, ordered by set inclusion:

$$R = F$$

$$\emptyset = \bot$$

$$\bigcup = \bigsqcup$$

$$\subseteq = \sqsubseteq$$

The tricky part of the earlier proof corresponded to showing that R is a continuous operator, which was true because we only allow inference rules with a finite number of premises.

4 Reinterpreting $\to$

Because we are mostly interested in continuous functions in this course, we will from now on interpret the notation $D \to E$ as denoting the continuous functions from D to E rather than all functions from D to E (which can be written as E^D, as before).

5 Back to **while**

Now we can finally complete the denotational semantics of IMP. We define the meaning of a while statement as:

$$\mathcal{C}[\![\textbf{while } b \textbf{ do } c]\!] = \mathit{fix}_{\Sigma \to \Sigma_\bot}\ \lambda f \in \Sigma \to \Sigma_\bot.\, \lambda \sigma \in \Sigma.\, \mathit{if}\ \neg \mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \mathit{strict}(f, \mathcal{C}[\![c]\!]\sigma)$$

where the least fixed point operator on domain D, fix_D, is defined as follows:

$$\mathit{fix} \stackrel{def}{=} \lambda f \in D \to D.\, \bigsqcup_{n \in \omega} f^n(\bot)$$

To check this definition, consider the denotation of **while true do skip**, which we expect to be $\lambda\sigma \in \Sigma.\, \bot$:

$$
\begin{aligned}
\mathcal{C}[\![\textbf{while true do skip}]\!] &= \mathit{fix}_{\Sigma\to\Sigma_\bot}\ \lambda f \in \Sigma \to \Sigma_\bot.\, \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \mathit{strict}(f, \mathcal{C}[\![c]\!]\sigma) \\
&= \mathit{fix}_{\Sigma} \to \Sigma_\bot\ \lambda f \in \Sigma \to \Sigma_\bot.\, \lambda\sigma \in \Sigma.\, \mathit{strict}(f, \mathcal{C}[\![c]\!]\sigma) \\
&= \mathit{fix}_{\Sigma\to\Sigma_\bot}\ \lambda f \in \Sigma \to \Sigma_\bot.\, \lambda\sigma \in \Sigma.\, f(\sigma) \\
&= \mathit{fix}_{\Sigma\to\Sigma_\bot}\ \lambda f \in \Sigma \to \Sigma_\bot.\, f
\end{aligned}
$$

The function we are taking the fixed point of is the identity function, and its least fixed point is $\bot_{\Sigma\to\Sigma_\bot}$, which is what *fix* will give us.

Now let us verify that our definition of the meaning of **while** works by checking that F is continuous. First of all, F must be monotonic. Consider an arbitrary pair of functions f, f' where $f \sqsubseteq f'$. We want to show that $F(f) \sqsubseteq F(f')$, which is true iff for all σ, $F(f)(\sigma) \sqsubseteq F(f')(\sigma)$. Consider an arbitrary σ. If $\neg\mathcal{B}[\![b]\!]\sigma$ or $\mathcal{C}[\![c]\!]\sigma = \bot$, the required relationship holds trivially. Otherwise, we require $f(\mathcal{C}[\![c]\!]\sigma) \sqsubseteq f'(\mathcal{C}[\![c]\!]\Sigma)$, which follows from $f \sqsubseteq f'$.

For F to be continuous, we also must have $\bigsqcup_n F(f_n) = F(\bigsqcup_n f_n)$ for all chains f_n. This also follows straightforwardly:

$$
\begin{aligned}
\bigsqcup F(f_n) &= \bigsqcup \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \mathit{strict}(f_n, \mathcal{C}[\![c]\!]\sigma) \\
&= \lambda\sigma \in \Sigma.\, \bigsqcup \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \mathit{strict}(f_n, \mathcal{C}[\![c]\!]\sigma) \\
&= \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \bigsqcup \mathit{strict}(f_n, \mathcal{C}[\![c]\!]\sigma) \\
&= \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \bigsqcup \mathit{strict}(f_n, \mathcal{C}[\![c]\!]\sigma) \\
&= \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \bigsqcup (\mathit{if}\ \mathcal{C}[\![c]\!]\sigma = \bot\ \mathit{then}\ \bot\ \mathit{else} f_n(\mathcal{C}[\![c]\!]\sigma)) \\
&= \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \bigsqcup (\mathit{if}\ \mathcal{C}[\![c]\!]\sigma = \bot\ \mathit{then}\ \bot\ \mathit{else} f_n(\mathcal{C}[\![c]\!]\sigma)) \\
&= \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ (\mathit{if}\ \mathcal{C}[\![c]\!]\sigma = \bot\ \mathit{then}\ \bot\ \mathit{else} \bigsqcup f_n(\mathcal{C}[\![c]\!]\sigma)) \\
&= \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \mathit{if}\ \mathcal{C}[\![c]\!]\sigma = \bot\ \mathit{then}\ \bot\ \mathit{else} \bigsqcup f_n(\mathcal{C}[\![c]\!]\sigma) \\
&= \lambda\sigma \in \Sigma.\, \mathit{if}\ \neg\mathcal{B}[\![b]\!]\sigma\ \mathit{then}\ \sigma\ \mathit{else}\ \mathit{if}\ \mathcal{C}[\![c]\!]\sigma = \bot\ \mathit{then}\ \bot\ \mathit{else}\ (\bigsqcup f_n)(\mathcal{C}[\![c]\!]\sigma) \qquad \text{(defn of join on fcns)} \\
&= F(\bigsqcup f_n)
\end{aligned}
$$

This process is simple but tedious. However, if we later decide to change any part of our definitions of any commands (c) or boolean expressions (b), we would have to prove that the resulting modified F still satisfied these properties. So, a more general method is desirable. Next time, we will define a set of constructs that always yield continuous functions.