Functional Programs and their Semantics

The next class of programs we will consider are programs which are presented as recursive functional definitions. For example, the two functions presented by the following definitions:

\[
F(x) \leftarrow \text{if } x = 0 \text{ then } 1 \text{ else } x \cdot F(x - 1)
\]
\[
G(x) \leftarrow \text{if } x > 100 \text{ then } x - 10 \text{ else } G(G(x + 11))
\]

represent, respectively, the \textit{factorial} function and the \textit{91} function.
Semantics of Functional Programs

The first question to be asked is what is the precise semantics of a functional program, presented by a definition such as \( F(x) \leftarrow E(x, F) \). Similar to the case of imperative programs, we have to define a notion equivalent to a computation, so that the various notions of correctness could be formulated.

Altogether, there are two dual approaches to the definition of the semantics of a functional program.
The Operational Approach

According to the operational approach, in order to find out the value of $F(a)$ for some $a \in D$, we form an evaluation sequence

$$F(a) = t_0, t_1, t_2, \ldots,$$

where each $t_{i+1}$ is derived from its predecessor $t_i$ by one of the following transformations:

- **Replacing** one or more occurrences of the form $F(e)$ by $E(e, F)$.
- Applying a **simplification** step to the term $t_i$.

For example, we may form the following evaluation sequence to the term $F(3)$ to the factorial function definition. This yields the following sequence:

$$F(3) \xrightarrow{\text{rewrite}} \begin{cases} \text{if } 3 = 0 \text{ then } 1 \text{ else } 3 \cdot F(3 - 1) \xrightarrow{\text{simplify}} 3 \cdot F(2) \xrightarrow{\text{rewrite}} \\ \text{if } \text{false } \text{ then } 1 \text{ else } 3 \cdot F(2) \xrightarrow{\text{simplify}} 3 \cdot F(2) \xrightarrow{\text{rewrite}} \\ 3 \cdot (\text{if } 2 = 0 \text{ then } 1 \text{ else } 2 \cdot F(2 - 1)) \xrightarrow{\text{simplify}} 3 \cdot 2 \cdot F(0) \xrightarrow{\text{simplify}} 6 \cdot F(0) \xrightarrow{\text{rewrite}} \\ 6 \cdot (\text{if } 0 = 0 \text{ then } 1 \text{ else } 0 \cdot F(2 - 1)) \xrightarrow{\text{simplify}} 6 \cdot 1 \xrightarrow{\text{simplify}} 6 \end{cases}$$

If there exists an evaluation sequence such that $t_n = b \in D$ is a constant, we write $F(a) = b$.

Some questions remain open in this definition:

- Which occurrences can be selected in a **rewrite** step?
- Which simplifications can be applied?
The Declarative Approach

In the declarative approach we view the definition $F(x) \iff E(x, F)$ as an equation $F(x) = E(x, F)$, and would like to adopt its solution as the function defined by the functional program.

The drawback to this approach is that there may be more than a single solution to the equation. For example, the equation

$$F(x) = \text{if } x = 0 \text{ then } 1 \text{ else } x \cdot F(x - 1)$$

has the family of functions

$$G_a(x) = \text{if } x \geq 0 \text{ then } x! \text{ else } (-1)^{-x} \frac{a}{(-x-1)!}$$

for each value of $a$ as a possible solution. Thus, the naive version of this approach does not yield a unique definition.

In the following we will refine the declarative approach obtaining a unique definition. Later we will investigate which versions of the operatorional approach are compatible with the declarative definition.
Ordered Domains

Central to the definition of the semantics of functional programs is the notion of one function being “less defined” than another. Consequently, we introduce a partial ordering relation $\sqsubseteq$, where we write $a \sqsubseteq b$ to denote that $a$ is less defined than $b$.

Given a concrete domain $D$ (such as the integers), we denote by $D^\perp = D \cup \{ \perp \}$ the domain obtained by augmenting $D$ with the undefined value $\perp$. Ordering on $D^\perp$ is defined by

$$a \sqsubseteq b \iff a = \perp \lor a = b$$

For example, following is the domain $\mathbb{N}^\perp$:

$\bullet \quad \bullet \quad -2 \quad -1 \quad 0 \quad +1 \quad +2 \quad \bullet \quad \bullet \quad \bullet$

Given ordered domains $D_1, \ldots, D_k$, we define their Cartesian product $D = D_1 \times \cdots \times D_k$ as the ordered domain consisting of $k$-tuples, whose ordering is given by

$$(a_1, \ldots, a_k) \sqsubseteq (b_1, \ldots, b_k) \iff a_1 \sqsubseteq b_1 \land \cdots \land a_k \sqsubseteq b_k$$
Monotonic Functions

A function \( f : (D_1^\downarrow)^n \mapsto D_2^\downarrow \) is said to be monotonic if \( \vec{a} \leq \vec{b} \) implies \( f(\vec{a}) \leq f(\vec{b}) \).

We denote by \([(D_1^\downarrow)^n \mapsto (D_2^\downarrow)^m]\) the set of all monotonic functions from \((D_1^\downarrow)^n\) to \((D_2^\downarrow)^m\).

Examples:

- The identity function \( I = \lambda x : x \) mapping each tuple \( \vec{x} \) to itself is monotonic.
- Every constant function \( \lambda x : c \) mapping any arguments to the constant \( c \) is monotonic. A special case is \( \Omega = \lambda x : \bot \), the totally undefined function.

Properties of monotonic functions

Let \( f : D_1^\downarrow \mapsto D_2^\downarrow \) be a unary function. Then \( f \) is monotonic iff

Either \( f(\bot) = \bot \)

Or \( f = \lambda x : c \), i.e., \( f \) is a constant function.

Let \( f : (D_1^\downarrow)^n \mapsto D_2^\downarrow \) be an \( n \)-ary function for \( n > 1 \). If \( f \) is monotonic then,

Either \( f(\bot, \ldots, \bot) = \bot \)

Or \( f = \lambda \vec{x} : c \), i.e., \( f \) is a constant function.

Note that the function \( f(x, y) = \text{if } x = \bot \text{ then } \bot \text{ else if } y = \bot \text{ then } 0 \text{ else } x \div y \) satisfies the condition \( f(\bot, \bot) = \bot \), yet it is not monotonic.
Natural Extension

Let $f : (D_1)^n \rightarrow (D_2)^m$ be a function over unordered domains. A straightforward way of extending $f$ into a monotonic function in $[(D_1^\perp)^n \rightarrow (D_2^\perp)^m]$ is the natural extension $f^+$ defined by

$$f^+(x_1, \ldots, x_n) = \text{if } x_1 = \perp \lor \cdots \lor x_n = \perp \text{ then } \perp^m \text{ else } f(x_1, \ldots, x_n)$$

**Claim 31. [The natural extension lemma]**

*Every naturally extended function is monotonic.*

**Proof:** Consider two tuples $(a_1, \ldots, a_n) \sqsubseteq (b_1, \ldots, b_n)$. We have to show that $f^+(\vec{a}) \sqsubseteq f^+(\vec{b})$. There are two cases to consider. If $a_i = \perp$ for some $i = 1, \ldots, n$, then $f^+(\vec{a}) = \perp^m \sqsubseteq f^+(\vec{b})$.

The other case is that $a_1 = b_1, \ldots, a_n = b_n$. In this case $f^+(\vec{a}) = f^+(\vec{b})$ which implies $f^+(\vec{a}) \sqsubseteq f^+(\vec{b})$. \qed

From now on, we will restrict our attention to base functions which are naturally extended with the exceptions of constant functions $\lambda \vec{x} : c$ and the *if-then-else* function which is defined as follows:

- $\text{if } \perp \text{ then } a \text{ else } b = \perp$
- $\text{if } \text{true } \text{ then } a \text{ else } b = a$
- $\text{if } \text{false } \text{ then } a \text{ else } b = b$

for every $a, b \in D^\perp$. Note that *if-then-else* is monotonic even though it is not naturally extended.
Composition of Monotonic Functions

Let \( f : (D_1^\perp)^n \rightarrow (D_2^\perp)^m \) and \( g : (D_2^\perp)^m \rightarrow (D_2^\perp)^k \) be two monotonic functions. Then, their composition \( f \circ g \) given by \( \lambda \vec{x} : g(f(\vec{x})) \) is monotonic.

Consider two tuples \( \vec{a} \subseteq \vec{b} \). Since \( f \) is monotonic, we have that \( f(\vec{a}) \subseteq f(\vec{b}) \). As \( g \) is monotonic, we have

\[
(f \circ g)(\vec{a}) = g(f(\vec{a})) \subseteq g(f(\vec{b})) = (f \circ g)(\vec{b})
\]

A function \( f : (D_1^\perp)^n \rightarrow D_2^\perp \) is called strict if \( f(a_1, \ldots, a_{i-1}, \perp, a_{i+1}, \ldots, a_n) = \perp \) for every \( i = 1, \ldots, n \) and \( a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in D_1^\perp \).
Distribution of Monotonic Functions over Conditionals

The set of monotonic functions \([\left(D_1^+\right)^n \rightarrow \left(D_2^+\right)^m]\) can be viewed as an ordered domain, where for each \(f_1, f_2 : \left(D_1^+\right)^n \rightarrow \left(D_2^+\right)^m\), we write \(f_1 \sqsubseteq f_2\) to denote that \(\forall \vec{x} : \left(D_1^+\right)^n : f_1(\vec{x}) \sqsubseteq f_2(\vec{x})\).

Consider the functions \(f_1\) and \(f_2\) defined as

\[
\begin{align*}
  f_1(\vec{x}) &= g(\text{if } p(\vec{x}) \text{ then } h_1(\vec{x}) \text{ else } h_2(\vec{x})) \\
  f_2(\vec{x}) &= \text{if } p(\vec{x}) \text{ then } g(h_1(\vec{x})) \text{ else } g(h_2(\vec{x}))
\end{align*}
\]

where, \(p\), \(g\), \(h_1\), and \(h_2\) are monotonic functions. Both \(f_1\) and \(f_2\) are monotonic. One can show that always \(f_2 \sqsubseteq f_1\). In the case that \(g(\bot) = \bot\) which covers the cases that \(g\) is monotonic but not a constant, we also have an inclusion in the other direction, leading to \(f_1 = f_2\).

Both inclusions can be proven by considering the three cases that \(p(\vec{x}) = \text{true}\), \(p(\vec{x}) = \text{false}\), and \(p(\vec{x}) = \bot\). For these cases, \(f_1(\vec{x})\) assumes the values \(g(h_1(\vec{x}))\), \(g(h_2(\vec{x}))\), and \(g(\bot)\), respectively, while \(f_2(\vec{x})\) assumes the values \(g(h_1(\vec{x}))\), \(g(h_2(\vec{x}))\), and \(\bot\). This shows that always \(f_2 \sqsubseteq f_1\) and if \(g(\bot) = \bot\) then also \(f_1 = f_2\).
Least Upper Bound

Let $\mathcal{D}$ be a partially ordered domain, and $S \subseteq \mathcal{D}$ a set of elements of $\mathcal{D}$. An element $a \in \mathcal{D}$ is said to be an upper bound of $S$, if $b \sqsubseteq a$ for every $b \in S$. We write $S \sqsubseteq a$ to denote that $a$ is an upper bound of $S$.

Not every set has an upper bound. For example, the set $\{0, 1\} \subseteq \mathbb{N}^\perp$ has no upper bound.

Element $a \in \mathcal{D}$ is called a least upper bound of set $S$ if $a$ is an upper bound of $S$, and $a \sqsubseteq c$ for every $c$ a (possibly different) upper bound of $S$. Note that if there exists a least upper bound for a set $S$ it must be unique. We denote by $\text{lub}(S)$ the least upper bound of set $S$ if it exists.

A set $S \subseteq \mathcal{D}$ is called directed if for every $a, b \in S$, $a \sqsubseteq b$ or $b \sqsubseteq a$. A special case of a directed set is a chain consisting of a finite or infinite sequence $a_0 \sqsubseteq a_1 \sqsubseteq a_2 \sqsubseteq \cdots$. A partially ordered domain $\mathcal{D}$ is called a complete partial order (CPO) if every directed set in $\mathcal{D}$ has a least upper bound.

An ordered domain $\mathcal{D}$ is said to be of bounded height if there exists an integer $n$ such that every directed set of pairwise distinct elements is of length not exceeding $n$. For example, the domain $(\mathcal{D}^\perp)^n$ is of bounded height $n + 1$. It is not difficult to see that every domain of bounded height is a CPO.
Monotonic functions form a CPO

Claim 32. [Monotonic functions form a CPO]
The domain \([(D_1^+)^n \rightarrow (D_2^+)^m]\) is a CPO.

Proof:
Let \(S \subseteq [(D_1^+)^n \rightarrow (D_2^+)^m]\) be a directed set. We define a function \(h : (D_1^+)^n \rightarrow (D_2^+)^m\) by letting \(h(\bar{a})\), for \(\bar{a} \in (D_1^+)^n\) equal

\[h(\bar{a}) = \text{lub}\{f(\bar{a}) \mid f \in S\}\]

Observing that the set \(\{f(\bar{a}) \mid f \in S\}\) is a directed set belonging to \((D_2^+)^m\), its lub is always defined, which shows that \(h\) is well formed. Note that, for every \(\bar{a} \in (D_1^+)^n\), there exists a function \(g \in S\), such that \(h(\bar{a}) = g(\bar{a})\).

We will show first that \(h\) is an upper bound of \(S\). Let \(g \in S\). For every \(\bar{a} \in (D_1^+)^n\), \(g(\bar{a}) \in \{f(\bar{a}) \mid f \in S\}\). Hence \(g(\bar{a}) \subseteq \text{lub}\{f(\bar{a}) \mid f \in S\} = h(\bar{a})\). It follows that \(g \subseteq h\).

Next, we will show that \(h\) is the least upper bound of \(S\). Let \(r\) be any other upper bound of \(S\). Consider any \(\bar{a} \in (D_1^+)^n\). Since \(r\) is an upper bound of \(S\), it follows that \(\{f(\bar{a}) \mid f \in S\} \subseteq r(\bar{a})\). Consequently, \(h(\bar{a}) = \text{lub}\{f(\bar{a}) \mid f \in S\} \subseteq r(\bar{a})\). We conclude that \(h \subseteq r\).

Finally, we will show that \(h\) is monotonic. Let \(\bar{a} \subseteq \bar{b}\). According to a previous observation, there exists a function \(g \in S\), such that \(h(\bar{a}) = g(\bar{a})\). It follows that

\[h(\bar{a}) \quad = \quad g(\bar{a}) \quad \subseteq \quad g(\bar{b}) \quad \subseteq \quad \text{lub}\{f(\bar{b}) \mid f \in S\} \quad = \quad h(\bar{b})\]

We conclude that \(h\) is monotonic.
More About \( \text{lub}'s \)

As an example, consider the infinite chain \( f_0 \subseteq f_1 \subseteq f_2 \subseteq \cdots \) in \([\mathbb{N}^+ \rightarrow \mathbb{N}^+]\) defined by

\[
f_i(x) : \text{if } x < i \text{ then } x! \text{ else } \bot
\]

by Claim 32, this chain has an \( \text{lub} \) which is given by \( g(x) = x! \).

**Corollary 33. [Lifting the \( \text{lub} \) operator]**

Let \( S \subseteq [(D_1^+)^n \rightarrow (D_2^+)^m] \) be a directed set, and \( \bar{a} \in (D_1^+)^n \). Then

\[
\text{lub}(\{f \mid f \in S\})(\bar{a}) = \text{lub}(\{f(\bar{a}) \mid f \in S\}).
\]

This corollary is a direct consequence of the definition of \( h = \text{lub}(\{f \mid f \in S\}) \).

**Claim 34. [Identity at \( k \) points]**

Let \( S \subseteq [(D_1^+)^n \rightarrow (D_2^+)^m] \) be a directed set, and let \( h = \text{lub}(S) \). For any set of points \( \bar{a}_1, \ldots, \bar{a}_k \in (D_1^+)^n \), there exists a function \( g \in S \) such that \( g \) agrees with \( h \) on their values at arguments \( \bar{a}_1, \ldots, \bar{a}_k \).

**Proof:**

We prove the claim by induction on \( k \). For \( k = 1 \), the observation made within the proof of Claim 32 guarantees the existence of \( g \) such that \( h(\bar{a}_1) = g(\bar{a}_1) \).

Assume that the claim has been proven for some \( k > 0 \). We will show that it also holds for \( k + 1 \). By the induction hypothesis, there exists a function \( g_k \in S \), such that \( h \) agrees with \( g_k \) on their values for arguments \( \bar{a}_1, \ldots, \bar{a}_k \). By the above mentioned observation, there also exists a function \( g \in S \) such that \( h(\bar{a}_{k+1}) = g(\bar{a}_{k+1}) \). Since \( S \) is a directed set, we must have \( g_k \subseteq g \) or \( g \subseteq g_k \). Let \( g_{k+1} \) denote the bigger of the two, i.e., \( g_{k+1} = \text{lub}(\{g_k, g\}) \). It is not difficult to show that \( g_{k+1} \) agrees with \( h \) on their values for arguments \( \bar{a}_1, \ldots, \bar{a}_k, \bar{a}_{k+1} \).
Continuous Functionals

A functional is a function $\mathcal{T} : [(D_1^+)^n \rightarrow D_2^+] \rightarrow [(D_1^+)^n \rightarrow D_2^+]$ mapping one monotonic function in $[(D_1^+)^n \rightarrow D_2^+]$ to another.

- A functional $\mathcal{T}$ is monotonic if $f \sqsubseteq g$ implies $\mathcal{T}(f) \sqsubseteq \mathcal{T}(g)$.
- Functional $\mathcal{T}$ is continuous if, for every directed set $S \subseteq [(D_1^+)^n \rightarrow D_2^+]$,

$$\mathcal{T}(\text{lub}(S)) = \text{lub}(\{\mathcal{T}(f) \mid f \in S\})$$

which also implies the existence of the right-hand side lub.

Note that a continuous functional is also monotonic. To see this, consider $f \sqsubseteq g$ and the directed set $S = \{f, g\}$. Obviously, $\text{lub}(S) = g$. We therefore have

$$\mathcal{T}(g) = \mathcal{T}(\text{lub}(S)) = \text{lub}(\{\mathcal{T}(f), \mathcal{T}(g)\})$$

Since $\mathcal{T}(g)$ has been identified as the lub of $\{\mathcal{T}(f), \mathcal{T}(g)\}$, it follows that $\mathcal{T}(f) \sqsubseteq \mathcal{T}(g)$, establishing the monotonicity of $\mathcal{T}$.

In our applications here we will be considering functionals of the form $\mathcal{T} = \lambda F : \mathcal{E}(F)$, where $\mathcal{E}(F)$ is an expression obtained by the composition of monotonic functions and the function symbol $F$. We refer to such functionals as constructible.
Constructible Functionals are Continuous

Claim 35.
Every constructible functional of the form $\mathcal{T} = \lambda F : \mathcal{E}(F)$ is continuous.

Proof:
We prove the claim by induction on the structure of the expression $\mathcal{E}(F)$.

First, consider the case that $\mathcal{E}(F) = r$ for some monotonic function $r$ which is independent of $F$. Thus, $\mathcal{T}(f) = r$ for every monotonic function $f$. Let $S$ be a directed set of monotonic functions such that $h = \text{lub}(S)$. We have

$$\mathcal{T}(\text{lub}(S)) = \mathcal{T}(h) = r = \text{lub}\{\{r\}\} = \text{lub}(\{\mathcal{T}(f) \mid f \in S\})$$

establishing that $\lambda F : r$ is a continuous functional.

Next, consider the case that $\mathcal{T} = \lambda F : r(\mathcal{T}_1(F), \ldots, \mathcal{T}_n(F))$, where $r$ is a monotonic function and $\mathcal{T}_1(F), \ldots, \mathcal{T}_n(F)$ are continuous functionals. Let $S$ be a directed set of monotonic functions such that $h = \text{lub}(S)$. We will show that $\text{lub}(\{\mathcal{T}(f) \mid f \in S\}) = \mathcal{T}(h)$ by establishing inequalities in the two directions. Let $f \in S$. Since $h = \text{lub}(S)$, $f \subseteq h$. Due to the monotonicity of $\mathcal{T}_1, \ldots, \mathcal{T}_n$ and $r$, we have that $\mathcal{T}$ is monotonic in $F$ and, therefore, $\mathcal{T}(f) \subseteq \mathcal{T}(h)$. As $\mathcal{T}$ is monotonic, the set $\mathcal{T}(S) = \{\mathcal{T}(f) \mid f \in S\}$ is directed and has an lub. As $\mathcal{T}(h)$ is an upper bound of $\mathcal{T}(S)$, we have that $\text{lub}(\mathcal{T}(S)) \subseteq \mathcal{T}(h)$. 

Proof Continued

In the other direction, consider the evaluation of \( \mathcal{T}(h)(\bar{a}) \). When we evaluate this expression we are called to evaluate the function \( h \) on arguments \( \bar{a}_1, \ldots, \bar{a}_k \), one of which may equal \( \bar{a} \). By Claim 34, there exists a function \( g_k \in S \) such that \( h \) agrees with \( g_k \) on their values for arguments \( \bar{a}_1, \ldots, \bar{a}_k \). It follows that \( \mathcal{T}(h)(\bar{a}) = \mathcal{T}(g_k)(\bar{a}) \). Since \( g_k \) is only one of the functions belonging to \( S \), we conclude

\[
\mathcal{T}(h)(\bar{a}) = \mathcal{T}(g_k)(\bar{a}) \subseteq \text{lub}(\{\mathcal{T}(f)(\bar{a}) \mid f \in S\}) = \text{lub}(\{\mathcal{T}(f) \mid f \in S\})(\bar{a})
\]

The last equality in this chain follows from Corollary 33. Abstracting away the arbitrary argument \( \bar{a} \), we conclude \( \mathcal{T}(h) \subseteq \text{lub}(\mathcal{T}(S)) \).

Finally, we consider the case that \( \mathcal{T} = \lambda F : F(\mathcal{T}_1(F), \ldots, \mathcal{T}_n(F)) \). This case is treated in a way identical to the treatment of the case \( \mathcal{T} = \lambda F : r(\mathcal{T}_1(F), \ldots, \mathcal{T}_n(F)) \), where the only difference may be in the number of arguments at which the function \( h \) should be evaluated, when we compute \( \mathcal{T}(h)(\bar{a}) \).
Examples

- The functional $\mathcal{T}$ over $[\mathbb{N}^\perp \mapsto \mathbb{N}^\perp]$ defined by:

\[
\mathcal{T}[F](x) : \text{ if } x = 0 \text{ then } 1 \text{ else } F(x + 1)
\]

is continuous, since it is constructible.

- The functional over $[\mathbb{N}^\perp \mapsto \mathbb{N}^\perp]$ defined by:

\[
\mathcal{T}[F](x) : \text{ if } \forall y \in \mathbb{N} : F(y) = y \text{ then } F(x) \text{ else } \perp
\]

is monotonic but not continuous. Consider, for example the infinite chain $f_0 \sqsubseteq f_1 \sqsubseteq f_2 \sqsubseteq \cdots$, where

\[
f_i(x) : \text{ if } x < i \text{ then } x \text{ else } \perp
\]

Then $\mathcal{T}[f_i] = \Omega$ (the identically undefined function $\lambda x : \perp$) for any $i \geq 0$, so that $\text{lub}\{\mathcal{T}(f_i)\} = \Omega$ whereas $\text{lub}\{f_i\} = \mathcal{T}(\text{lub}\{f_i\}) = \lambda x : x \neq \Omega$.

- The functional over $[\mathbb{N}^\perp \mapsto \mathbb{N}^\perp]$ defined by:

\[
\mathcal{T}[F](x) : \text{ if } F(X) = \perp \text{ then } 0 \text{ else } \perp
\]

is not even monotonic. For let $Z = \lambda x : 0$ denote the zero function. Then $\mathcal{T}[Z] = \Omega$ and $\mathcal{T}[\Omega] = Z$; Thus while $\Omega \sqsubseteq Z$, we have

\[
\mathcal{T}[\Omega] = Z \not\sqsubseteq \Omega = \mathcal{T}[Z]
\]

Note that the functional $\mathcal{T}'[F](x) : \text{ if } F(x) = \perp \perp \text{ then } 0 \text{ else } \perp$ is continuous. In particular, $\mathcal{T}'[f] = \Omega$ for every $f : [\mathbb{N}^\perp \mapsto \mathbb{N}^\perp]$. 
Fix-points of Functionals

Let $\mathcal{T} : [(D_1^\perp)^n \rightarrow D_2^\perp] \rightarrow [(D_1^\perp)^n \rightarrow D_2^\perp]$ be a functional. A function $f : [(D_1^\perp)^n \rightarrow D_2^\perp]$ is called a fix-point of $\mathcal{T}$ if it satisfies the equality $f = \mathcal{T}(f)$.

Function $f$ is called a least fix-point of $\mathcal{T}$ if it is a fix-point, and $f \subseteq g$ for every other $g$ a fix-point of $\mathcal{T}$. Note that if there exists a least fix-point, it is unique.

Examples:

- Consider the following functional over $[\mathbb{N}^\perp \rightarrow \mathbb{N}^\perp]$:

$$\mathcal{T}_1[F](x) : \begin{cases} \text{if } x = 0 \text{ then } 1 \text{ else } x \cdot F(x - 1) \end{cases}$$

Since the function $\lambda x : x!$ satisfies the equality

$$\mathcal{T}_1[x!](x) = \begin{cases} \text{if } x = 0 \text{ then } 1 \text{ else } x \cdot (x - 1)! \end{cases} = x!$$

we conclude that $x!$ is a fix-point of $\mathcal{T}_1$. 
Example: $f_{91}$

Consider the following functional over $[\mathbb{Z}^\perp \mapsto \mathbb{Z}^\perp]$ ($\mathbb{Z}$ is the set of integers):

$$\mathcal{T}[F](x) : \text{ if } x > 100 \text{ then } x - 10 \text{ else } F(F(x + 11))$$

we will show that the function $f_{91}(x) : \text{ if } x > 100 \text{ then } x - 10 \text{ else } 91$ is a fix-point of $\mathcal{T}$. To show this we need to establish that

$$\text{if } x > 100 \text{ then } x - 10 \text{ else } f_{91}(f_{91}(x + 11)) = \text{if } x > 100 \text{ then } x - 10 \text{ else } 91$$

$$\mathcal{T}[f_{91}](x) = f_{91}(x)$$

To show this equality, we consider 4 cases:

- $x > 100$. In this case $\mathcal{T}[f_{91}](x) = x - 10 = f_{91}(x)$.

- $89 < x \leq 100$. In this case $\mathcal{T}[f_{91}](x) = f_{91}(f_{91}(x + 11)) = f_{91}(x + 1)$. The constraint on $x$ implies $90 < x + 1 \leq 101$. We consider two sub-cases: if $x + 1 = 101$ then $f_{91}(x + 1) = (x + 1) - 10 = 91$. Otherwise, $x + 1 \leq 100$ and $f_{91}(x + 1) = 91$. Thus, in both sub-cases, $\mathcal{T}[f_{91}](x)$ equals 91 as is $f_{91}(x)$ because $x \leq 100$.

- $x \leq 89$. In this case $\mathcal{T}[f_{91}](x) = f_{91}(f_{91}(x + 11)) = f_{91}(91) = 91 = f_{91}(x)$ since both $x + 11$ and $x$ do not exceed 100.

- The final case is $x = \bot$ in which both sides yield $\bot$. 
The Fix-Point Theorem

Let $\mathcal{T}$ be a continuous functional. We form the sequence $\mathcal{T}^0[\Omega], \mathcal{T}^1[\Omega], \mathcal{T}^2[\Omega], \ldots$, defined as follows: $\mathcal{T}^0[\Omega] = \Omega$, and $\mathcal{T}^{i+1}[\Omega] = \mathcal{T}[\mathcal{T}^i[\Omega]]$, for every $i \geq 0$.

Since $\Omega \subseteq \mathcal{T}[\Omega]$, and $\mathcal{T}$ is monotonic, we can prove by induction that $\mathcal{T}^i[\Omega] \subseteq \mathcal{T}^{i+1}[\Omega]$. Thus, the sequence of functions forms a chain $\mathcal{T}^0[\Omega] \subseteq \mathcal{T}^1[\Omega] \subseteq \mathcal{T}^2[\Omega] \subseteq \cdots$, which has an lub.

**Theorem 36. [First recursion theorem, Kleene]**

*Every continuous functional $\mathcal{T}$ has a least fix-point, denoted $f_\mathcal{T}$, which is given by $f_\mathcal{T} = \text{lub}\{\mathcal{T}^i[\Omega]\}$.*

**Proof:**

We take $f_\mathcal{T} = \text{lub}\{\mathcal{T}^i[\Omega]\}$. We show first that $f_\mathcal{T}$ is a fix-point of $\mathcal{T}$. Since $\mathcal{T}$ is continuous, we have

$$\mathcal{T}[f_\mathcal{T}] = \mathcal{T}[\text{lub}\{\mathcal{T}^i[\Omega]\}] = \text{lub}\{\mathcal{T}^{i+1}[\Omega]\} = \text{lub}\{\mathcal{T}^i[\Omega]\} = f_\mathcal{T}$$

Thus, $\mathcal{T}[f_\mathcal{T}] = f_\mathcal{T}$.

Next, assume that $g$ is another fix-point of $\mathcal{T}$. We show by induction that $\mathcal{T}^i[\Omega] \subseteq g$. For the base step, we observe that $\Omega \subseteq g$. Assuming $\mathcal{T}^i[\Omega] \subseteq g$, we may apply the monotonic $\mathcal{T}$ to both sides and obtain $\mathcal{T}^{i+1}[\Omega] = \mathcal{T}[\mathcal{T}^i[\Omega]] \subseteq \mathcal{T}[g] = g$, establishing the induction hypothesis for $i + 1$. Since $g$ is an upper bound for all $\mathcal{T}^i[\Omega]$ it is greater or equal to the least upper bound of this chain, which is $f_\mathcal{T}$. Thus, we have that $f_\mathcal{T} \subseteq g$, which shows that $f_\mathcal{T}$ is the least fix-point of $\mathcal{T}$.  

Examples

- Consider the functional

\[ T_1[F](x) : \text{ if } x = 0 \text{ then } 1 \text{ else } x \cdot F(x - 1) \]

Computing the chain \( T_1^i[\Omega] \), we obtain

\[ T_1^i[\Omega](x) : \text{ if } x < i \text{ then } x! \text{ else } \bot \]

Obviously, the \textit{lub} of this chain is \( x! \).

- Next, consider the functional

\[ T[F]_{91}(x) : \text{ if } x > 100 \text{ then } x - 10 \text{ else } F(F(x + 11)) \]

Computing the chain \( T_{91}^i[\Omega](x) \), we obtain

\[ \Omega \quad \text{ for } i = 0 \]
\[ \text{if } x > 100 \text{ then } x - 10 \text{ else if } x > 101 - i \text{ then } 91 \text{ else } \bot \quad \text{ for } 1 \leq i \leq 11 \]
\[ \text{if } x > 100 \text{ then } x - 10 \text{ else if } x > 90 - 11(i - 11) \text{ then } 91 \text{ else } \bot \quad \text{ for } 11 \leq i \]

Taking the \textit{lub} of this chain, we obtain

\[ f_{91}(x) : \text{ if } x > 100 \text{ then } x - 10 \text{ else } 91 \]
Based on the above discussion, we consider functional programs of the form

\[ F \leftarrow \mathcal{T}[F] \]

where \( \mathcal{T}[F] \) is a continuous functional. The semantics of such recursive definition is given by the least fix-point \( f_\mathcal{T} \) of the functional \( \mathcal{T} \).

Two questions remain:

- What is the operational semantics which is compatible with the above declarative semantics?
- How can we use these definitions and concepts in order to verify the correctness of such functional programs?