Trabajo práctico N° 6

Mayo 2025

Estudiante Emanuel Nicolás Herrador

Ejercicio 1

En este ejercicio se pretende dar un programa para cada posible comportamiento en LIS con fallas y output. Veamos cada caso posible de forma separada.

Respecto a un programa con *cantidad finita de output y luego divergencia*, podemos considerar:

$$\textbf{while true do skip}$$

Un programa con *cantidad finita de output y luego falla* puede ser:

$$\textbf{fail}$$

Un programa con *cantidad finita de output y luego terminación* puede ser:

$$\textbf{skip}$$

Y, finalmente, un programa con *cantidad infinita de output* puede ser:

$$\textbf{while true do } !1$$

Ejercicio 2

Dado el programa $\textbf{while } \mathbf{x} > 0 \textbf{ do } !\mathbf{x}; c$ se pretende calcular la semántica denotacional para cada uno de los casos dependiendo el programa c. Se verá cada uno por separado.

Item A

Consideramos $c \equiv \textbf{if } \mathbf{x} > 0 \textbf{ then skip else fail}$. Luego, el programa a considerar es:

$$\textbf{while } \mathbf{x} > 0 \textbf{ do } (!\mathbf{x}; \textbf{if } \mathbf{x} > 0 \textbf{ then skip else fail})$$

En base a eso, veamos la semántica del programa. Sea w la semántica del while, entonces:

$$
\begin{aligned}
&[\![\textbf{while } \mathbf{x} > 0 \textbf{ do } (!\mathbf{x}; \textbf{if } \mathbf{x} > 0 \textbf{ then skip else fail})]\!]\,\sigma = \\
&= w\sigma \\
&= \begin{cases} w_*([\![!\mathbf{x}; \textbf{if } \mathbf{x} > 0 \textbf{ then skip else fail}]\!]\,\sigma) & \text{si } [\![\mathbf{x} > 0]\!]\,\sigma \\ \langle\sigma\rangle & \text{si } \neg\,[\![\mathbf{x} > 0]\!]\,\sigma \end{cases} \\
&= \begin{cases} w_*([\![\textbf{if } \mathbf{x} > 0 \textbf{ then skip else fail}]\!]_*\,([\![!\mathbf{x}]\!]\,\sigma)) & \text{si } \sigma\mathbf{x} > 0 \\ \langle\sigma\rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\
&= \begin{cases} w_*([\![\textbf{if } \mathbf{x} > 0 \textbf{ then skip else fail}]\!]\,\langle\sigma\mathbf{x},\ \sigma\rangle) & \text{si } \sigma\mathbf{x} > 0 \\ \langle\sigma\rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\
&= \begin{cases} \langle\sigma\mathbf{x},\ w_*([\![\textbf{if } \mathbf{x} > 0 \textbf{ then skip else fail}]\!]\,\sigma)\rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle\sigma\rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\
&= \begin{cases} \langle\sigma\mathbf{x},\ w_*([\![\textbf{skip}]\!]\,\sigma)\rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle\sigma\rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\
&= \begin{cases} \langle\sigma\mathbf{x}, w\sigma\rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle\sigma\rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\
&= \begin{cases} \langle\sigma\mathbf{x}\rangle \ +\!\!+\ w\sigma & \text{si } \sigma\mathbf{x} > 0 \\ \langle\sigma\rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}
\end{aligned}
$$

Ahora, para calcular w vamos a considerar $F \in (\Sigma \to \Omega) \to (\Sigma \to \Omega)$ tal que:

$$Fw\sigma = \begin{cases} \langle \sigma \mathbf{x} \rangle \;\mathbin{+\!\!+}\; w\sigma & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}$$

Sabemos que como w es un while, entonces F es una función continua. Ahora, para calcular la semántica del while podemos usar TMPF de modo que $w = \bigsqcup_{i \in \mathbb{N}} F^i \bot_{\Sigma \to \Omega}$. Para ello, entonces, se propone la siguiente caracterización para $F^i \bot_{\Sigma \to \Omega}$ con $i \geq 1$:

$$F^i \bot_{\Sigma \to \Omega} = \sigma \mapsto \begin{cases} \langle \overbrace{\sigma\mathbf{x}, \ldots, \sigma\mathbf{x}}^{i \text{ veces}} \rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}$$

Para demostrarlo, vamos a hacer inducción en i. Veamos primero el caso base $i = 1$:

$$\begin{aligned} F^1 \bot_{\Sigma \to \Omega} &= \sigma \mapsto \begin{cases} \langle \sigma \mathbf{x} \rangle \;\mathbin{+\!\!+}\; \bot_{\Sigma \to \Omega}\sigma & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\ &= \sigma \mapsto \begin{cases} \langle \sigma \mathbf{x} \rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \end{aligned}$$

por lo que se cumple. Ahora, como HI suponemos que la caracterización vale para $k \in \mathbb{N}_{\geq 1}$ y queremos ver $k + 1$:

$$\begin{aligned} F^{k+1} \bot_{\Sigma \to \Omega} &= \sigma \mapsto \begin{cases} \langle \sigma \mathbf{x} \rangle \;\mathbin{+\!\!+}\; F^k \bot_{\Sigma \to \Omega}\sigma & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\ &= \sigma \mapsto \begin{cases} \langle \sigma \mathbf{x} \rangle \;\mathbin{+\!\!+}\; \langle \overbrace{\sigma\mathbf{x}, \ldots, \sigma\mathbf{x}}^{k \text{ veces}} \rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\ &= \sigma \mapsto \begin{cases} \langle \overbrace{\sigma\mathbf{x}, \ldots, \sigma\mathbf{x}}^{k+1 \text{ veces}} \rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \end{aligned}$$

Luego, entonces, por TMPF queda claro que la semántica del while es:

$$w = \sigma \mapsto \begin{cases} \langle \overbrace{\sigma\mathbf{x}, \ldots, \sigma\mathbf{x}, \ldots}^{\text{infinitas veces}} \rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}$$

Item B

Ahora consideramos $c \equiv \mathbf{if}\ \mathbf{x} > 0\ \mathbf{then\ fail\ else\ skip}$. Por ello, el programa a considerar es:

$$\mathbf{while}\ \mathbf{x} > 0\ \mathbf{do}\ (!x; \mathbf{if}\ \mathbf{x} > 0\ \mathbf{then\ fail\ else\ skip})$$

En base a eso, veamos la semántica del programa. Sea w la semántica del while, entonces:

$$\begin{aligned} &[\![\mathbf{while}\ \mathbf{x} > 0\ \mathbf{do}\ (!x; \mathbf{if}\ \mathbf{x} > 0\ \mathbf{then\ fail\ else\ skip})]\!]\, \sigma = \\ &= w\sigma \\ &= \begin{cases} w_*([\![!x; \mathbf{if}\ \mathbf{x} > 0\ \mathbf{then\ fail\ else\ skip}]\!]\, \sigma) & \text{si } [\![\mathbf{x} > 0]\!]\, \sigma \\ \langle \sigma \rangle & \text{si } \neg\, [\![\mathbf{x} > 0]\!]\, \sigma \end{cases} \\ &= \begin{cases} w_*([\![\mathbf{if}\ \mathbf{x} > 0\ \mathbf{then\ fail\ else\ skip}]\!]_* \,([\![!x]\!]\, \sigma)) & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\ &= \begin{cases} w_*([\![\mathbf{if}\ \mathbf{x} > 0\ \mathbf{then\ fail\ else\ skip}]\!]) \langle \sigma\mathbf{x}, \sigma \rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \\ &= \begin{cases} \langle \sigma \mathbf{x} \rangle \;\mathbin{+\!\!+}\; w_*([\![\mathbf{if}\ \mathbf{x} > 0\ \mathbf{then\ fail\ else\ skip}]\!]\, \sigma) & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases} \end{aligned}$$

$$= \begin{cases} \langle \sigma\mathbf{x}\rangle \ \text{++}\ w_*([\![\mathbf{fail}]\!]\,\sigma) & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}$$

$$= \begin{cases} \langle \sigma\mathbf{x}\rangle \ \text{++}\ w_*\langle \mathbf{abort}, \sigma\rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}$$

$$= \begin{cases} \langle \sigma\mathbf{x}, \langle \mathbf{abort}, \sigma\rangle\rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}$$

En base a esto, entonces, la semántica del while está dada por:

$$w = \sigma \mapsto \begin{cases} \langle \sigma\mathbf{x}, \langle \mathbf{abort}, \sigma\rangle\rangle & \text{si } \sigma\mathbf{x} > 0 \\ \langle \sigma \rangle & \text{si } \sigma\mathbf{x} \leq 0 \end{cases}$$

Ejercicio 3

En este ejercicio se pretende demostrar o refutar las equivalencias propuestas usando semántica denotacional. Veamos cada una de forma separada.

Item A

Queremos analizar la equivalencia dada por $?x; ?y \equiv ?y; ?x$. Claramente no son equivalentes, por lo que se va a mostrar un contraejemplo. De este modo, sean $\mathbf{x}, \mathbf{y} \in \langle \text{var} \rangle$, entonces se va a ver $?\mathbf{x}; ?\mathbf{y} \equiv ?\mathbf{y}; ?\mathbf{x}$. La semántica de ambos programas es similar, por lo que me voy a concentrar en hacer la de la izquierda:

$$\begin{aligned} [\![?\mathbf{x}; ?\mathbf{y}]\!]\,\sigma &= [\![?\mathbf{y}]\!]_*\,([\![?\mathbf{x}]\!]\,\sigma) \\ &= [\![?\mathbf{y}]\!]_*\,\iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n])) \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ [\![?y]\!]\,[\sigma \mid \mathbf{x} : n]) \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{in}}\,(\lambda m \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n \mid \mathbf{y} : m]))) \end{aligned}$$

Análogamente, para el lado derecho tenemos que

$$[\![?\mathbf{y}; ?\mathbf{x}]\!]\,\sigma = \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{in}}\,(\lambda m \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{y} : n \mid \mathbf{x} : m])))$$

Como claramente no se cumple para todo input posible la equivalencia, entonces no son equivalentes.

Item B

Queremos analizar la equivalencia dada por $?x; z := x \equiv ?z$. Claramente no son equivalentes por lo que daré un contraejemplo. Sea $\mathbf{x}, \mathbf{z} \in \langle \text{var} \rangle$, analicemos $?\mathbf{x}; \mathbf{z} := \mathbf{x} \equiv ?\mathbf{z}$. Veamos primero el lado izquierdo:

$$\begin{aligned} [\![?\mathbf{x}; \mathbf{z} := \mathbf{x}]\!]\,\sigma &= [\![\mathbf{z} := \mathbf{x}]\!]_*\,([\![?\mathbf{x}]\!]\,\sigma) \\ &= [\![\mathbf{z} := \mathbf{x}]\!]_*\,\iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n])) \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ [\![\mathbf{z} := \mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n]) \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n \mid \mathbf{z} : n])) \end{aligned}$$

Mientras que del lado derecho tenemos:

$$[\![?\mathbf{z}]\!]\,\sigma = \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{z} : n]))$$

Luego, entonces, no son equivalentes dado que el programa de la izquierda modifica el valor de $\mathbf{x}$ en el estado mientras que el de la derecha no.

Item C

Queremos analizar la equivalencia dada por $\mathbf{newvar}\ x := e\ \mathbf{in}\ (?x; z := x) \equiv ?z$. Parecería que sí son equivalentes pero en realidad no lo son por un caso particular. Notemos primero que x, z son metavariables, por lo que estas pueden tomar cualquier variable del lenguaje. Teniendo esto en cuenta, para mostrar el contraejemplo vamos a considerar que ambas metavariables toman a la variable $\mathbf{y} \in \langle \text{var} \rangle$ del lenguaje. Con ello en mente, la equivalencia a analizar ahora es $\textbf{newvar } \mathbf{y} := e \textbf{ in } (?\mathbf{y}; \mathbf{y} := \mathbf{y}) \equiv ?\mathbf{y}$. Si vemos el lado izquierdo tenemos:

$$
\begin{aligned}
[\![\textbf{newvar } \mathbf{y} := e \textbf{ in } (?\mathbf{y}; \mathbf{y} := \mathbf{y})]\!]\, \sigma &= (\lambda \sigma' \in \Sigma.\ [\sigma' \mid \mathbf{y} : \sigma \mathbf{y}])_{\dagger}([\![?\mathbf{y}; \mathbf{y} := \mathbf{y}]\!]\, [\sigma \mid \mathbf{y} : [\![e]\!]\, \sigma]) \\
&= (\lambda \sigma' \in \Sigma.\ [\sigma' \mid \mathbf{y} : \sigma \mathbf{y}])_{\dagger}([\![\mathbf{y} := \mathbf{y}]\!]_{*}\, ([\![?\mathbf{y}]\!]\, [\sigma \mid \mathbf{y} : [\![e]\!]\, \sigma])) \\
&= (\lambda \sigma' \in \Sigma.\ [\sigma' \mid \mathbf{y} : \sigma \mathbf{y}])_{\dagger}([\![\mathbf{y} := \mathbf{y}]\!]_{*}\, \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\, ([\sigma \mid \mathbf{y} : n]))) \\
&= (\lambda \sigma' \in \Sigma.\ [\sigma' \mid \mathbf{y} : \sigma \mathbf{y}])_{\dagger} \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ [\![\mathbf{y} := \mathbf{y}]\!]\, [\sigma \mid \mathbf{y} : n]) \\
&= (\lambda \sigma' \in \Sigma.\ [\sigma' \mid \mathbf{y} : \sigma \mathbf{y}])_{\dagger} \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\, ([\sigma \mid \mathbf{y} : n])) \\
&= \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ (\lambda \sigma' \in \Sigma.\ [\sigma' \mid \mathbf{y} : \sigma \mathbf{y}])[\sigma \mid \mathbf{y} : n]) \\
&= \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\, (\sigma))
\end{aligned}
$$

Mientras que si miramos el lado derecho tenemos:

$$[\![?\mathbf{y}]\!] = \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ [\sigma \mid \mathbf{y} : n])$$

Con ello, entonces se muestra el contraejemplo dado que uno mantiene el estado intacto mientras que el otro programa no. Luego, no son programas equivalentes semánticamente.

Ejercicio 4

Queremos analizar la equivalencia dada por $?\mathbf{x}; c; !\mathbf{x} \equiv ?\mathbf{y}; c; !\mathbf{y}$ donde c es un programa que no incluye ni fallas, outputs ni inputs y es tal que $\{\mathbf{x}, \mathbf{y}\} \cap FV(c) = \emptyset$. Como esta equivalencia no es correcta, para refutarla voy a dar un contraejemplo donde $c \equiv \textbf{skip}$. Con ello en mente, se pretende analizar ahora $?\mathbf{x}; \textbf{skip}; !\mathbf{x} \equiv ?\mathbf{y}; c; !\mathbf{y}$. Ambos programas son similares y su cálculo de semántica es análogo. Por ello, me concentraré únicamente en calcularla para el programa de la izquierda:

$$
\begin{aligned}
[\![?\mathbf{x}; \textbf{skip}; !\mathbf{x}]\!]\, \sigma &= [\![\textbf{skip}; !\mathbf{x}]\!]_{*}\, ([\![?\mathbf{x}]\!]\, \sigma) \\
&= [\![\textbf{skip}; !\mathbf{x}]\!]_{*}\, \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\, ([\sigma \mid \mathbf{x} : n])) \\
&= \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ [\![\textbf{skip}; !\mathbf{x}]\!]\, [\sigma \mid \mathbf{x} : n]) \\
&= \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]_{*}\, ([\![\textbf{skip}]\!]\, [\sigma \mid \mathbf{x} : n])) \\
&= \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]_{*}\, \iota_{\text{term}}\, ([\sigma \mid \mathbf{x} : n])) \\
&= \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]\, [\sigma \mid \mathbf{x} : n]) \\
&= \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\, (n,\ \iota_{\text{term}}\, ([\sigma \mid \mathbf{x} : n])))
\end{aligned}
$$

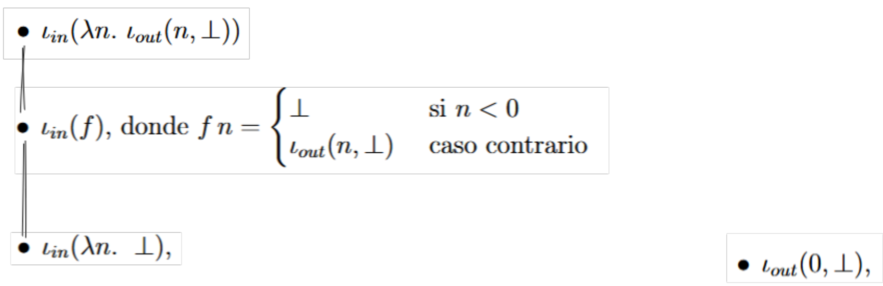
Mientras que del lado derecho llegamos a:

$$[\![?\mathbf{y}; \textbf{skip}; !\mathbf{y}]\!]\, \sigma = \iota_{\text{in}}\, (\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\, (n,\ \iota_{\text{term}}\, ([\sigma \mid \mathbf{y} : n])))$$

Con ello, se muestra que son semánticas distintas porque se llega a un estado diferente debido que en el primero se modifica el valor asignado a $\mathbf{x}$ en el estado, mientras que en el segundo es el de $\mathbf{y}$. Finalmente, entonces, se muestra el contraejemplo de porqué estos dos programas no son equivalentes.

Ejercicio 5

El diagrama de Hasse a considerar es el siguiente:



Ejercicio 6

Se pretende dar un programa de modo que su semántica sea el supremo de la cadena:

$$w_0 = \bot, \qquad w_{i+1} = \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ w_i\right)\right)$$

El programa que doy es **while true do** ?**x**; !**x**. Para ver que cumple con la condición del ejercicio, veamos su semántica:

$$\begin{aligned}
[\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]\,\sigma &= \begin{cases} [\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]_* \left([\![?\mathbf{x}; !\mathbf{x}]\!]\,\sigma\right) & \text{si } [\![\textbf{true}]\!]\,\sigma \\ \iota_{\text{term}}\,(\sigma) & \text{si } \neg\,[\![\textbf{true}]\!]\,\sigma \end{cases} \\
&= [\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]_* \left([\![!\mathbf{x}]\!]_* \left([\![?\mathbf{x}]\!]\,\sigma\right)\right) \\
&= [\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]_* \left([\![!\mathbf{x}]\!]_*\, \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ [\sigma \mid \mathbf{x} : n]\right)\right) \\
&= [\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]_*\, \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n]\right) \\
&= [\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]_*\, \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ [\sigma \mid \mathbf{x} : n]\right)\right) \\
&= \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ [\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n]\right)\right)
\end{aligned}$$

Con ello en mente, entonces, podemos considerar F tal que, sea $w' = [\![\textbf{while true do } ?\mathbf{x}; !\mathbf{x}]\!]$:

$$Fw'\sigma = \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ w'[\sigma \mid \mathbf{x} : n]\right)\right)$$

Sabemos que F es continua al provenir de un while y que por TMPF $w = \bigsqcup_{i \in \mathbb{N}} F^i \bot_{\Sigma \to \Omega}$. Ahora, lo único que resta mostrar es que la cadena $F^0 \bot_{\Sigma\to\Omega} \sqsubseteq F^1 \bot_{\Sigma\to\Omega} \sqsubseteq \ldots$ es la misma que $w_0 \sqsubseteq w_1 \sqsubseteq \ldots$. Esto lo vamos a hacer con inducción. Primero, veamos que para el caso base $F^0 \bot_{\Sigma\to\Omega} = \bot_{\Sigma\to\Omega} = w_0$ por lo que se cumple. Ahora, como HI suponemos que dado un $k \in \mathbb{N}$ se cumple que $F^k \bot_{\Sigma\to\Omega} = w_k$ y queremos ver $k+1$:

$$\begin{aligned}
F^{k+1} \bot_{\Sigma\to\Omega} &= \sigma \mapsto \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ F^k \bot_{\Sigma\to\Omega}[\sigma \mid \mathbf{x} : n]\right)\right) \\
&= \sigma \mapsto \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ w_k[\sigma \mid \mathbf{x} : n]\right)\right)
\end{aligned}$$

Notemos que como $Fw'\sigma = Fw'[\sigma \mid \mathbf{x} : m]$ para $m \in \mathbb{Z}$, entonces se cumple que:

$$F^{k+1} \bot_{\Sigma\to\Omega} = \sigma \mapsto \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ w_k\sigma\right)\right) = \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\left(n,\ w_k\right)\right)$$

Con ello, se demuestra entonces el caso inductivo y que ambas cadenas son la misma. Finalmente, se demuestra que el programa dado cumple con la semántica pedida. $\square$

Ejercicio 7

Queremos ver si los programas de la forma **while true do** $(?x; c)$ cumplen que la cadena $F^i \bot_{\Sigma\to\Omega}\sigma$ de la semántica del while es interesante o no en Ω. Vamos a ver que no lo es por contraejemplo considerando $c \equiv$ **fail**. Sea $w = [\![\textbf{while true do } (?x; \textbf{fail})]\!]$, tenemos:

$$\begin{aligned}
w\sigma &= \begin{cases} w_*([\![?x; \textbf{fail}]\!]\,\sigma) & \text{si } [\![\textbf{true}]\!]\,\sigma \\ \iota_{\text{term}}\,(\sigma) & \text{si } \neg\,[\![\textbf{true}]\!]\,\sigma \end{cases} \\
&= w_*([\![\textbf{fail}]\!]_* \left([\![?x]\!]\,\sigma\right)) \\
&= w_*([\![\textbf{fail}]\!]_*\, \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\left([\sigma \mid \mathbf{x} : n]\right)\right)) \\
&= w_* \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ [\![\textbf{fail}]\!]\,[\sigma \mid \mathbf{x} : n]\right) \\
&= w_* \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\left([\sigma \mid \mathbf{x} : n]\right)\right) \\
&= \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\left([\sigma \mid \mathbf{x} : n]\right)\right)
\end{aligned}$$

Luego, tenemos que $Fw\sigma = \iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\left([\sigma \mid \mathbf{x} : n]\right)\right)$, por lo que $\forall i \geq 1,\ F^i = F^{i+1}$. Por ello, entonces, en este caso no es una cadena interesante porque el supremo de esta cadena es $F^1 \bot_{\Sigma\to\Omega}\sigma$.

Ejercicio 8

Vamos a ver el programa P dado por:

$$P \equiv \textbf{newvar } \mathbf{x} := \mathbf{x} + 1 \textbf{ in } (\textbf{while } \mathbf{x} > 0 \textbf{ do } (?\mathbf{x}; \textbf{if } \mathbf{y} > 0 \textbf{ then fail else } !\mathbf{x}))$$

Y en los siguientes items calcularemos su semántica para los casos donde $\sigma\mathbf{y} > 0$ y $\sigma\mathbf{y} \leq 0$. Primero, veamos en general la semántica para un estado $\sigma \in \Sigma$ cualquiera.

Sea $w = [\![\mathbf{while}\ \mathbf{x} > 0\ \mathbf{do}\ (?\mathbf{x}; \mathbf{if}\ \mathbf{y} > 0\ \mathbf{then}\ \mathbf{fail}\ \mathbf{else}\ !\mathbf{x})]\!]$:

$$\begin{aligned}&[\![\mathbf{newvar}\ \mathbf{x} := \mathbf{x}+1\ \mathbf{in}\ (\mathbf{while}\ \mathbf{x} > 0\ \mathbf{do}\ (?\mathbf{x}; \mathbf{if}\ \mathbf{y} > 0\ \mathbf{then}\ \mathbf{fail}\ \mathbf{else}\ !\mathbf{x}))]\!] = \\ &= (\lambda\sigma' \in \Sigma.\ [\sigma'|\mathbf{x} : \sigma\mathbf{x}])_\dagger(w[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1])\end{aligned} \tag{8.1}$$

Ahora, nos concentremos primero en el while:

$$\begin{aligned}
&w[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1] = \\
&= \begin{cases} w_*([\![?\mathbf{x}; \mathbf{if}\ \mathbf{y} > 0\ \mathbf{then}\ \mathbf{fail}\ \mathbf{else}\ !\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } [\![\mathbf{x} > 0]\!]\,[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1] \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \neg\,[\![\mathbf{x} > 0]\!]\,[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1] \end{cases} \\
&= \begin{cases} w_*([\![\mathbf{if}\ \mathbf{y} > 0\ \mathbf{then}\ \mathbf{fail}\ \mathbf{else}\ !\mathbf{x}]\!]_*\,([\![?\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1])) & \text{si } \sigma\mathbf{x}+1 > 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases} \\
&= \begin{cases} w_*([\![\mathbf{if}\ \mathbf{y} > 0\ \mathbf{then}\ \mathbf{fail}\ \mathbf{else}\ !\mathbf{x}]\!]_*\,\iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n]))) & \text{si } \sigma\mathbf{x}+1 > 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases} \\
&= \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ w_*([\![\mathbf{if}\ \mathbf{y} > 0\ \mathbf{then}\ \mathbf{fail}\ \mathbf{else}\ !\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases} \\
&= \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ w_*([\![\mathbf{fail}]\!]\,[\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \wedge \sigma\mathbf{y} > 0 \\ \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ w_*([\![!\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \wedge \sigma\mathbf{y} \leq 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases} \\
&= \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ w_*\iota_{\text{abort}}\,([\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \wedge \sigma\mathbf{y} > 0 \\ \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ w_*\iota_{\text{out}}\,(n,\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n]))) & \text{si } \sigma\mathbf{x}+1 > 0 \wedge \sigma\mathbf{y} \leq 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases} \\
&= \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\,([\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \wedge \sigma\mathbf{y} > 0 \\ \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ w[\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \wedge \sigma\mathbf{y} \leq 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases}
\end{aligned} \tag{8.2}$$

Obteniendo esto, ahora prosigamos con cada item por separado.

Item A

Aquí se pretende calcular la semántica de P en un estado σ tal que $\sigma\mathbf{y} > 0$. Sea $\sigma \in \Sigma : \sigma\mathbf{y} > 0$, entonces si tomamos desde (8.2) tenemos que:

$$w[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1] = \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\,([\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases}$$

Luego, como en ambos casos se establece una terminación, entonces esta es la semántica del while. Siguiendo con (8.1) tendríamos que:

$$\begin{aligned}
&[\![\mathbf{newvar}\ \mathbf{x} := \mathbf{x}+1\ \mathbf{in}\ (\mathbf{while}\ \mathbf{x} > 0\ \mathbf{do}\ (?\mathbf{x}; \mathbf{if}\ \mathbf{y} > 0\ \mathbf{then}\ \mathbf{fail}\ \mathbf{else}\ !\mathbf{x}))]\!] = \\
&= \begin{cases} (\lambda\sigma' \in \Sigma.\ [\sigma'|\mathbf{x} : \sigma\mathbf{x}])_\dagger(\iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\,([\sigma \mid \mathbf{x} : n]))) & \text{si } \sigma\mathbf{x}+1 > 0 \\ (\lambda\sigma' \in \Sigma.\ [\sigma'|\mathbf{x} : \sigma\mathbf{x}])_\dagger(\iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1])) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases} \\
&= \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\,((\lambda\sigma' \in \Sigma.\ [\sigma'|\mathbf{x} : \sigma\mathbf{x}])[\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x}+1 > 0 \\ \iota_{\text{term}}\,((\lambda\sigma' \in \Sigma.\ [\sigma'|\mathbf{x} : \sigma\mathbf{x}])[\sigma \mid \mathbf{x} : \sigma\mathbf{x}+1]) & \text{si } \sigma\mathbf{x}+1 \leq 0 \end{cases} \\
&= \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{abort}}\,(\sigma)) & \text{si } \sigma\mathbf{x} > -1 \\ \iota_{\text{term}}\,(\sigma) & \text{si } \sigma\mathbf{x} \leq -1 \end{cases}
\end{aligned}$$

por lo que se obtiene la semántica del programa para estados donde $\sigma\mathbf{y} > 0$.

Item B

Ahora, consideramos los estados con $\sigma\mathbf{y} \leq 0$. Es decir, sea $\sigma \in \Sigma : \sigma\mathbf{y} \leq 0$, tenemos que a partir de (8.2):

$$w[\sigma \mid \mathbf{x} : \sigma\mathbf{x} + 1] = \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ w[\sigma \mid \mathbf{x} : n])) & \text{si } \sigma\mathbf{x} + 1 > 0 \\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : \sigma\mathbf{x} + 1]) & \text{si } \sigma\mathbf{x} + 1 \leq 0 \end{cases}$$

Sea $\sigma' = [\sigma \mid \mathbf{x} : \sigma\mathbf{x} + 1] \in \Sigma$, entonces queda claro que la F para la definición del while está dada por:

$$Fw\sigma' = \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ w[\sigma' \mid \mathbf{x} : n])) & \text{si } \sigma'\mathbf{x} > 0 \\ \iota_{\text{term}}\,(\sigma') & \text{si } \sigma'\mathbf{x} \leq 0 \end{cases}$$

Con ello en mente, veamos algunos casos para $F^i \bot_{\Sigma\to\Omega}$:

$$F^0 \bot_{\Sigma\to\Omega} = \bot_{\Sigma\to\Omega}$$

$$\begin{aligned} F^1 \bot_{\Sigma\to\Omega} &= \sigma' \mapsto \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ \bot_{\Sigma\to\Omega}[\sigma' \mid \mathbf{x} : n])) & \text{si } \sigma'\mathbf{x} > 0 \\ \iota_{\text{term}}\,(\sigma') & \text{si } \sigma'\mathbf{x} \leq 0 \end{cases} \\ &= \sigma' \mapsto \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ \bot)) & \text{si } \sigma'\mathbf{x} > 0 \\ \iota_{\text{term}}\,(\sigma') & \text{si } \sigma'\mathbf{x} \leq 0 \end{cases} \end{aligned}$$

$$\begin{aligned} F^2 \bot_{\Sigma\to\Omega} &= \sigma' \mapsto \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ F\bot_{\Sigma\to\Omega}[\sigma' \mid \mathbf{x} : n])) & \text{si } \sigma'\mathbf{x} > 0 \\ \iota_{\text{term}}\,(\sigma') & \text{si } \sigma'\mathbf{x} \leq 0 \end{cases} \\ &= \sigma' \mapsto \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ \iota_{\text{in}}\,(\lambda m \in \mathbb{Z}.\ \iota_{\text{out}}\,(m,\ \bot)))) & \text{si } \sigma' > 0 \wedge n > 0 \\ \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ \iota_{\text{term}}\,([\sigma' \mid \mathbf{x} : n]))) & \text{si } \sigma'\mathbf{x} > 0 \wedge n \leq 0 \\ \iota_{\text{term}}\,(\sigma') & \text{si } \sigma'\mathbf{x} \leq 0 \end{cases} \end{aligned}$$

Claramente se puede notar que la continuación del while depende del valor del input en la iteración para $\mathbf{x}$. Motivo de ello, entonces, no se puede dar una expresión general para este caso del programa.

Ejercicio 9

En este ejercicio se pretende demostrar o refutar las equivalencias propuestas usando semántica denotacional. Veamos cada una de ellas por separado.

Item A

Queremos analizar la equivalencia:

$$?\mathbf{x}; \textbf{while } b \textbf{ do } (!\mathbf{x}; ?\mathbf{x}); !\mathbf{x} \ \equiv\ \textbf{while } b \textbf{ do } ?\mathbf{x}; !\mathbf{x}$$

Vamos a mostrar que no es correcta dando un contraejemplo. Sea $b \equiv \textbf{false}$, $w_1 = [\![\textbf{while false do } (!\mathbf{x}; ?\mathbf{x})]\!]$ y $w_2 = [\![\textbf{while false do } ?\mathbf{x}; !\mathbf{x}]\!]$, entonces:

$$\begin{aligned} [\![?\mathbf{x}; \textbf{while false do } (!\mathbf{x}; ?\mathbf{x}); !\mathbf{x}]\!]\,\sigma &= [\![\textbf{while false do } (!\mathbf{x}; ?\mathbf{x}); !\mathbf{x}]\!]_*\,([\![?\mathbf{x}]\!]\,\sigma) \\ &= [\![\textbf{while false do } (!\mathbf{x}; ?\mathbf{x}); !\mathbf{x}]\!]_*\,\iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n])) \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ [\![\textbf{while false do } (!\mathbf{x}; ?\mathbf{x}); !\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n]) \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]_*\,(w_1[\sigma \mid \mathbf{x} : n])) \\ &= \begin{cases} \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]_*\,(w_{1*}([\![!\mathbf{x}; ?\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n]))) & \text{si } [\![\textbf{false}]\!]\,\sigma \\ \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]_*\,\iota_{\text{term}}\,([\sigma \mid \mathbf{x} : n])) & \text{si } \neg\,[\![\textbf{false}]\!]\,\sigma \end{cases} \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ [\![!\mathbf{x}]\!]\,[\sigma \mid \mathbf{x} : n]) \\ &= \iota_{\text{in}}\,(\lambda n \in \mathbb{Z}.\ \iota_{\text{out}}\,(n,\ [\sigma \mid \mathbf{x} : n])) \end{aligned}$$

Y para el programa de la derecha tenemos que:

$$\begin{aligned} w_2\sigma &= \begin{cases} w_{2*}([\![?\mathbf{x}; !\mathbf{x}]\!]\,\sigma) & \text{si } [\![\textbf{false}]\!]\,\sigma \\ \iota_{\text{term}}\,(\sigma) & \text{si } \neg\,[\![\textbf{false}]\!]\,\sigma \end{cases} \\ &= \iota_{\text{term}}\,(\sigma) \end{aligned}$$

Luego, entonces, con ello llegamos claramente a que para $b \equiv \textbf{false}$ ambos programas tienen una semántica distinta por lo que no son equivalentes. Con ello, se muestra el contraejemplo para refutar esta propuesta de equivalencia.

Item B

Queremos analizar ahora la equivalencia dada por:

$$?\mathbf{x}; \textbf{while } b \textbf{ do } (!\mathbf{x}; ?\mathbf{x}); !\mathbf{x} \ \equiv ?\mathbf{x}; !\mathbf{x}; \textbf{while } b \textbf{ do } ?\mathbf{x}; !\mathbf{x}$$

Es una equivalencia correcta, por lo que hay que demostrarla. Intuyo que es algo larga y con varios casos para ver y explicar, por lo que no lo voy a realizar. En caso que tenga más tiempo, lo hago.