1 Parcial 1: 13/04/2022

(1) Determinar si las siguientes son verdaderas o falsas, justificando acabadamente.

(a) Sea

$$p := \forall x.\,(1 < x \vee \exists y.z = y + 1) \wedge \exists y.\,((\forall z.z + x = y) \Rightarrow z > y)$$

y sea $\delta = [z \to z + 1, y \to x, x \to x + y]$. Entonces al hacer p/δ, en todos los cuantificadores $\forall v.p$ se puede tomar $v = v_{new}$.

(b) Para toda g, la cadena $f_0, f_1, \ldots$ es interesante, donde $f_i \in \mathbb{Z} \to \mathbb{Z}_\perp$ se define mediante

$$f_i\ n = \begin{cases} g\ n & -i \leq n \leq i \\ \perp & c.c. \end{cases}$$

(c) Si $c_w =$ **while true do** c, entonces el lenguaje imperativo con fallas satisface $[\![c_w]\!] = [\![c; c_w]\!]$.

(d) Si $\mathbb{Z}$ tiene el orden discreto, entonces $\mathbb{Z} \to \mathbb{Z}_\perp$ tiene infinitos elementos maximales.

(a) El enunciado es verdadero. La sustitución de v

(b) Sea g la función constantemente bottom. Se sigue que para todo n, $f_i\ n = \perp$. Por lo tanto, $f_i = g = \perp_{\mathbb{Z} \to \mathbb{Z}_\perp}$. Luego, la cadena $f_0, f_1, \ldots$ es no interesante. $\therefore$ El enunciado es falso.

(c) El enunciado es falso. Observemos que $[\![c_w]\!]\sigma = \bigsqcup_{i \in \mathbb{N}} F^i\ \perp$ con

$$F\ f\ \sigma = f_*([\![c]\!]\sigma)$$

Claramente, $F \perp \sigma = \perp$, y $F^2 \perp \sigma = \perp$, etc. Es decir, $\{F^i\ \perp\}$ es no interesante con supremo $\perp$. $\therefore [\![c_w]\!]\sigma = \perp$.

Sin embargo, tome por ejemplo $c =$ **fail**. Entonces

$$\begin{aligned}
[\![c; c_w]\!]\sigma &= [\![c_w]\!]_* ([\![\mathbf{fail}]\!]\sigma) \\
&= [\![c_w]\!]_* (\langle \mathbf{abort}, \sigma \rangle) \\
&= \langle \mathbf{abort}, \sigma \rangle \\
&\neq \bot
\end{aligned}$$

(d) Como $\mathbb{Z}$ tiene el orden discreto, $\mathbb{Z}_\bot$ es el orden llano, que es un dominio. Entonces $\mathbb{Z} \to \mathbb{Z}_\bot$ es un dominio y su mínimo es la función constantemente $\bot$.

Sea $z_0 \in \mathbb{Z}_\bot$ un elemento distinto de $\bot$. Por ser $\mathbb{Z}_\bot$ llano, se sigue que z_0 es maximal. Sea $f(x) = z_0$ una función constante. Necesariamente, f es maximal, porque si existiere alguna $g \neq f$ tal que $f \leq g$ deberíamos tener $f(x) \leq g(x) \equiv z_0 \leq g(x)$ para todo x en que ambas funciones estén definidas. Pero esto no es posible a menos que $g(x) = z_0$ para toda x, lo cual por hipótesis es falso.

Como el z_0 elegido es arbitrario, esto se cumple para cualquier $z \in \mathbb{Z}$. Por lo tanto, toda función constante en $\mathbb{Z} \to \mathbb{Z}_\bot$ es maximal. $\therefore$ Existen infinitos maximales.

Sea $h \in \mathbb{Z} \to \mathbb{Z}_\perp$. Considere:

$$f\ n = \begin{cases} h\ n & n \leq 0 \vee n \text{ impar} \\ f(n-2) & n > 0 \wedge n \text{ par} \end{cases}$$

Sea F_h la función asociada a la ecuación recursiva, cuyo menor punto fijo es la solución. (a) Escriba la definición explícita de F_h. (b) Defina $h_0, h_1 \in \mathbb{Z} \to \mathbb{Z}_\perp$ tales que

$$F_{h_0}\ h_0 = h_0, \qquad F_{h_1}\ h_1 \neq h_1$$

(c) Calcule $F_h^3 \perp_{\mathbb{Z} \to \mathbb{Z}_\perp}$.

(d) Sin necesidad de calcularla, proponga una expresión para la función $\bigsqcup_{i \in \mathbb{N}} F_h^i\ \perp$. Describa informalmente qué denota este supremo en términos de h.

(e) En el cálculo de (d), ¿es el único punto fijo o hay más?

(a) La definición explícita de F es

$$F\ f = n \mapsto \begin{cases} h\ n & n \leq 0 \vee n \text{ impar} \\ f(n-2) & n > 0 \wedge n \text{ par} \end{cases}$$

(b) Observemos qué hace $f\ n$, o bien qué tipo de funciones solucionan la ecuación definida por F. Dado $n \leq 0$ e impar, se deriva el control a una función h sin una llamada recursiva. Si n es positivo y par, se ejecuta $f(n-2) = f(n-4) = \ldots = f(0) = h(0)$. Podemos interpretar f como una función definida recursivamente sobre los pares positivos, con caso base $h\ 0$ extendido sobre el resto del dominio. En general, toda función en la cual $h(0)$ coincida con el valor dado a los pares positivos satisface la ecuación.

Por ejemplo, la siguiente ecuación es punto fijo:

$$h_0\ n = \begin{cases} n & n \leq 0 \vee n \text{ impar} \\ 0 & c.c. \end{cases}$$

Esto puede verse:

$$
\begin{aligned}
F\ h_0\ n &= \begin{cases} h_0\ n & n \leq \vee n \text{ impar} \\ h_0(n-2) & n \in \{2,4,6,\ldots\} \end{cases} \\
&= \begin{cases} n & n \leq 0 \vee n \text{ impar} \\ h_0(2-2) & n = 2 \\ h_0(n) & n \in \{4,6,8,\ldots\} \end{cases} \\
&= \begin{cases} n & n \leq 0 \vee n \text{ impar} \\ 0 & n = 2 \\ 0 & n \in \{4,6,8,\ldots\} \end{cases} \\
&= \begin{cases} n & n \leq 0 \vee n \text{ impar} \\ 0 & c.c. \end{cases} \\
&= h_0(n)
\end{aligned}
$$

Una función que no es punto fijo es, por ejemplo, $h_1(n) = n+1$. Es fácil ver que esto resulta en

$$
F\ h_1\ n = \begin{cases} n+1 & n \leq 0 \vee n \text{ impar} \\ n-1 & c.c. \end{cases} \qquad \neq h_1(n)
$$

(c) Calculemos

$$
\begin{aligned}
F\bot = n \mapsto & \begin{cases} h\ n & n \leq 0 \vee n \text{ impar} \\ \bot & c.c. \end{cases} \\
F^2\ \bot = n \mapsto & \begin{cases} h\ n & n \leq 0 \vee n \text{ impar} \\ (F\ \bot)(n-2) & n \in \{2,4,6,\ldots\} \end{cases} \\
= n \mapsto & \begin{cases} h\ n & n \leq 0 \vee n \text{ impar} \\ h(n-2) & n \in \{2,4,6,\ldots\} \wedge \Big((n-2) \leq 0\ \vee (n-2) \text{ impar}\Big) \\ \bot & (n-2) \in \{2,4,6,\ldots\} \wedge \Big(n-2 > 0 \wedge n-2 \text{ par}\Big) \end{cases} \\
= n \mapsto & \begin{cases} h\ n & n \leq 0 \vee n \text{ impar} \\ h(n-2) & n = 2 \\ \bot & n \in \{4,6,\ldots\} \end{cases} \\
= n \mapsto & \begin{cases} h\ n & n \leq 0 \vee n \text{ impar} \\ h(0) & n = 2 \\ \bot & n \in \{4,6,\ldots\} \end{cases}
\end{aligned}
$$

Por último,

$$
\begin{aligned}
F^3\ \bot = n \mapsto & \begin{cases} h(n) & n \leq 0 \vee n \text{ impar} \\ h(n-2) & (n > 0 \wedge n \text{ par}) \wedge (n-2 \leq 0 \vee n-2 \text{ impar}) \\ h(0) & (n > 0 \wedge n \text{ par}) \wedge n - 2 = 2 \\ \bot & n - 2 \in \{4, 6, \ldots\} \end{cases} \\
= n \mapsto & \begin{cases} h(n) & n \leq 0 \vee n \text{ impar} \\ h(n-2) & n = 2 \\ h(0) & n = 4 \\ \bot & n \in \{6, 8, \ldots\} \end{cases} \\
= n \mapsto & \begin{cases} h(n) & n \leq 0 \vee n \text{ impar} \\ h(0) & n \in \{2, 4\} \\ \bot & n \in \{6, 8, \ldots\} \end{cases}
\end{aligned}
$$

(d) Proponemos

$$
\bigsqcup_{i \in \mathbb{N}} F^i\ \bot = n \mapsto \begin{cases} h(n) & n \leq 0 \vee n \text{ impar} \\ h(0) & n > 0 \wedge n \text{ par} \end{cases}
$$

Dicho supremo denota la *menor* función que satisface la ecuación inducida por F, donde *menor* se entiende por "la que provee la menor cantidad posible de información".

(c) No es el único punto fijo, de hecho en a dimos h_0 otro posible punto fijo. Pero sí es el *menor* punto fijo.

(3) Analice la equivalencia del comando **skip** con el comando

$$c := \textbf{newvar } x := 0 \textbf{ in catchin } (\textbf{while } x = 0 \textbf{ do fail}) \textbf{ with skip}$$

Primero, pensemos qué hace c. Inicializa x localmente en cero, ejecuta un **while** cuya guarda necesariamente será verdadera, dentro del cual se ejecuta **fail**, y esta falla es "catcheada" con **skip**. Con lo cual, intuitivamente, la equivalencia será verdadera. Demostrémoslo. Denotemos con $\mathscr{R}$ la función de restauración asociada a x y el estado σ utilizado abajo:

$$\begin{aligned}
&[\![\textbf{newvar } x := 0 \textbf{ in catchin } (\textbf{while } x = 0 \textbf{ do fail}) \textbf{ with skip}]\!]\sigma \\
=&\mathscr{R}_\dagger\,([\![\textbf{catchin } (\textbf{while } x = 0 \textbf{ do fail}) \textbf{with skip}]\!][\sigma \mid x : 0]) \\
=&\ \mathscr{R}_\dagger\,([\![\textbf{skip}]\!]_+\,([\![\textbf{while } x = 0 \textbf{ do fail}]\!][\sigma \mid x : 0])) \\
=&\ \mathscr{R}_\dagger\,([\![\textbf{skip}]\!]_+\,(\langle \textbf{abort}, [\sigma \mid x : 0]\rangle)) && (\star) \\
=&\ \mathscr{R}_\dagger\,([\![\textbf{skip}]\!][\sigma \mid x : 0]) \\
=&\ \mathscr{R}_\dagger\,(\sigma \mid x : 0) \\
=&\ \sigma \\
=&\ [\![\textbf{skip}]\!]\sigma
\end{aligned}$$

El paso $(\star)$ está justificado porque definiendo

$$F\ f\ \sigma = \begin{cases} \sigma & \sigma\ x \neq 0 \\ f * ([\![\textbf{fail}]\!]\sigma) & c.c. \end{cases}$$

tenemos que

$$[\![\textbf{while } x = 0 \textbf{ do fail}]\!] = \bigsqcup_{i \in \mathbb{N}} F^i\ \bot = \sigma \mapsto \begin{cases} \sigma & \sigma\ x \neq 0 \\ \langle \textbf{abort}, \sigma\rangle & c.c. \end{cases}$$

Y en la línea marcada por $\star$, la semántica del **while** está de hecho recibiendo un estado donde la variable x vale 0.

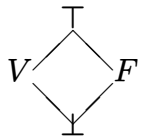
Enunciar el teorema de coincidencia para LIS con fallas.

(1) Sea c un comando y sean σ, σ' estados tales que $\sigma\ w = \sigma'\ w$ para toda $w \in FV(c)$. Entonces, o bien $[\![c]\!]\sigma = [\![c]\!]\sigma' = \bot$, o bien $[\![c]\!]\sigma\ w = [\![c]\!]\sigma'\ w$ para toda $w \in FV(c)$.

(2) Sea c un comando. Entonces, para toda $w \notin FV(c)$, se cumple que para todo $\sigma \in \Sigma$, $[\![c]\!]\sigma\ w = \sigma\ w$, siempre que $[\![c]\!]\sigma \neq \bot$.

2 Parcial 1: 12/06/2024

1. Sea $\mathbb{B}_\perp^\top$ el siguiente reticulado:



   (a) Determine la validez de la siguiente afirmación: "Toda función monótona en $\mathbb{B}_\perp^\top \to \mathbb{B}_\perp^\top$ tiene un punto fijo".

   (b) Determine la validez de la siguiente afirmación: "Existe una función no monótona (y por lo tanto no continua) en $\mathbb{B}_\perp^\top \to \mathbb{B}_\perp^\top$ que tiene un punto fijo".

   (c) Proponga una función $F\colon (\mathbb{Z} \to \mathbb{B}^\top) \to (\mathbb{Z} \to \mathbb{B}^\top)$ que NO sea NI constante NI la identidad y que SÍ sea continua. Dé su menor punto fijo.

2. Considere el lenguaje imperativo simple con fallas. Sea c el programa siguiente

$$\text{while } x \neq 0 \text{ do if } x < 0 \text{ then } d := 1 + d\,;\, x := x + 3 \text{ else fail}$$

   (a) Escriba de la forma más sencilla posible la ecuación para $F(f)(\sigma)$ donde F es el funcional asociado al ciclo de ese programa.

   (b) Proponga un valor negativo para x en σ para que $[[c]]\sigma = \langle \text{abort}, [\sigma \mid d : 3][x : 2]\rangle$? Justificá tu respuesta calculando la semántica de c, también tendrás que elegir un valor para d en σ.

(1) (a) Como el reticulado es finito, toda cadena es no-interesante y toda función monótona es continua. Por lo tanto,

$$x_0 = \bot, x_1 = f(\bot), x_2 = f(f(\bot)), x_3 = f(f(f(\bot))), \ldots$$

es una cadena no interesante. Como es no interesante, la cadena $x_0, x_1, \ldots$ tiene un elemento x_n que se repite indefinidamente. Es decir, existe un n tal que $x_{n+1} = x_n$. Pero $x_{n+1} = f(x_n)$, con lo cual tenemos $f(x_n) = x_n$. Luego existe un punto fijo.

(b) Definamos

$$f(x) = \begin{cases} \bot & x = \top \\ \top & x = \bot \\ V & x = V \\ F & x = F \end{cases}$$

Claramente, no es monótona porque $\bot \leq \top$, mientras que $f(\bot) \leq f(\top) \equiv \top \leq \bot \equiv \textbf{False}$. Sin embargo, tiene puntos fijos.

(c) Una tal función continua es la que mapea $V \mapsto F, F \mapsto V$, y deja lo demás intacto. Probemos que es continua.

Claramente, $\bot \leq a$ y $f(\bot) = \bot \leq f(a)$ para todo a. $V \leq \top, F \leq \top$ implican inmediatamente que $f(V) \leq f(\top) = \top, f(F) \leq f(\top) = \top$. Obviamente $f(\top) = \top \leq \top$. Su menor punto fijo, trivialmente, es $\bot$.

(2) Considere la siguiente ecuación recursiva.

$$g(x) = \begin{cases} 1 & \text{si } 0 \leq x \leq 3 \\ 1 + g(x-4) & \text{si } x < 0 \vee 3 < x \end{cases}$$

Calcule la menor solución para esa ecuación en $\mathbb{Z} \to \mathbb{Z}_\perp$.

De acuerdo con el teorema del menor punto fijo, definimos

$$F\ f\ x = \begin{cases} 1 & 0 \leq x \leq 3 \\ 1 + f(x-4) & x < 0 \vee 3 < x \end{cases}$$

Encontremos el menor punto fijo de F. Claramente,

$$F\ \perp = x \mapsto \begin{cases} 1 & 0 \leq x \leq 3 \\ \perp & x < 0 \vee 3 < x \end{cases}$$

Entonces

$$\begin{aligned} F^2\ \perp = x \mapsto & \begin{cases} 1 & 0 \leq x \leq 3 \\ 1 + (F\ \perp)(x-4) & c.c. \end{cases} \\ = x \mapsto & \begin{cases} 1 & 0 \leq x \leq 3 \\ 1+1 & (x < 0 \vee 3 < x) \wedge (0 \leq x-4 \leq 3) \\ \perp & (x < 0 \vee 3 < x) \wedge (x-4 < 0 \vee 3 < x-4) \end{cases} \\ = & \begin{cases} 1 & 0 \leq x \leq 3 \\ 2 & (x < 0 \vee 3 < x) \wedge (4 \leq x \leq 7) \\ \perp & (x < 0 \vee 3 < x) \wedge (x < 4 \vee 7 < x) \end{cases} \\ = & \begin{cases} 1 & 0 \leq x \leq 3 \\ 2 & (4 \leq x \leq 7) \\ \perp & (x < 0 \vee 3 < x) \wedge (x < 4 \vee 7 < x) \end{cases} \end{aligned}$$

Además,

$$F^3\ \bot = x \mapsto \begin{cases} 1 & 0 \leq x \leq 3 \\ 1 + (F^2\ \bot)(x-4) & x < 0 \vee 3 < x \end{cases}$$

$$x \mapsto = \begin{cases} 1 & 0 \leq x \leq 3 \\ 1+1 & (x < 0 \vee 3 < x) \wedge (0 \leq x-4 \leq 3) \\ 1+2 & (x < 0 \vee 3 < x) \wedge (4 \leq x-4 \leq 7) \\ \bot & (x < 0 \vee 3 < x) \wedge (x-4 < 0 \vee 3 < x-4) \wedge (x-4 < 4 \vee 7 < x-4) \end{cases}$$

$$= x \mapsto \begin{cases} 1 & 0 \leq x \leq 3 \\ 2 & 4 \leq x \leq 7 \\ 3 & 8 \leq x \leq 11 \\ \bot & c.c. \end{cases}$$

Asumamos como HI que

$$F^k \bot = x \mapsto \begin{cases} 1 & 0 \leq x \leq 3 \\ 2 & 4 \leq x \leq 7 \\ \vdots & \\ k & 4(k-1) \leq x \leq 4k-1 \\ \bot & c.c. \end{cases}$$

Veamos entonces que

$$F^{k+1}\ \bot = x \mapsto \begin{cases} 1 & 0 \leq x \leq 3 \\ 1 + (F^k\ \bot)(x-4) & c.c. \end{cases}$$

$$= x \mapsto \begin{cases} 1 & 0 \leq x \leq 3 \\ 1+1 & (x < 0 \vee 3 < x) \wedge 0 \leq x-4 \leq 3 \\ 1+2 & (x < 0 \vee 3 < x) \wedge 4 \leq x-4 \leq 7 \\ \vdots & \\ 1+k & (x < 0 \vee 3 < x) \wedge 4(k-1) \leq x-4 \leq 4k-1 \\ \bot & c.c. \end{cases}$$

$$= x \mapsto \begin{cases} 1 & 0 \leq x \leq 3 \\ 2 & 4 \leq x \leq 7 \\ 3 & 8 \leq x \leq 11 \\ \vdots & \\ 1+k & 4k \leq x \leq 4(k+1)-1 \\ \bot & c.c. \end{cases}$$

Que es lo que se quería demostrar. Es evidente entonces que

$$\bigsqcup_{i \in \mathbb{N}} F^i \ \bot = x \mapsto x/4$$

donde $\cdot/\cdot$ es la divisón entera.

(3) Considerá LIS con fallas. Sea

$$c := \textbf{while } x \neq 0 \textbf{ do if } x < 0 \textbf{ then } d := 1 + d; x := x + 3 \textbf{ else fail}$$

(a) Escribí de la forma más sencilla la ecuación para $F\ f\ \sigma$.

(b) Proponé un valor negativo para x en σ para que

$$[\![c]\!]\sigma = \langle \textbf{abort}, [\sigma \mid d : 3 \mid x : 2] \rangle$$

Justificá calculando la semántica de c.

La ecuación para $F\ f\ \sigma$ toma la forma

$$F\ f\ \sigma = \begin{cases} \sigma & x = 0 \\ f_*\Big([\![\textbf{if } x < 0 \textbf{ then } d := 1 + d; x := x + 3 \textbf{ else fail}]\!]\sigma\Big) & c.c. \end{cases}$$

y para simplificarla debemos simplificar la semántica del **if**. Pero claramente

$$[\![\textbf{if } x < 0 \textbf{ then } d := d + 1; x := x + 3 \textbf{ else fail}]\!]\sigma = \begin{cases} [\sigma \mid d : \sigma\ d + 1 \mid x :\ \ x + 3] & x < 0 \\ \textbf{fail} & x \geq 0 \end{cases}$$

con lo cual la forma más simplificada es

$$F\ f\ \sigma = \begin{cases} \sigma & x = 0 \\ f\left([\sigma \mid d : \sigma\ d + 1 \mid x : \sigma\ x + 3]\right) & x < 0 \\ \langle \textbf{abort}, [\sigma \mid d : \sigma\ d + 1 \mid x : \sigma\ x + 3] \rangle & x > 0 \end{cases}$$

Claramente, si $\sigma\ x = -1, \sigma\ x = -2$, el while se ejecutará una vez poniendo $\sigma'\ x = 2, \sigma'\ d = 3$, y abortando en dicho estado.

(4) Propone un ejemplo de un programa que satisfaga:

$$[\![\text{while } b \text{ do } c]\!] = F^3\ \bot_{\Sigma \to \Sigma_\bot}$$

Proponemos

$$\textbf{while } x < 2 \textbf{ do if } x < 0 \textbf{ then } x := 1 \textbf{ else } x := x + 1$$

En el peor caso, este while se ejecuta dos veces, con lo cual la cadena $F^i\ \bot$ es no interesante y su supremo es $F^3\ \bot$, correspondiendo a la segunda iteración.

(5) ¿Puede haber un programa de la forma `while b do c` tal que para cualquier σ, $[\![\text{while b do c}]\!]\sigma = \iota_{\text{in}}(g)$, donde $g\ n = [\sigma \mid v : 2 \mid x : n]$?

Formulemos la pregunta en términos intuitivos. ¿Existe un while que pida un input y cuyo resultado siempre sea dejar la variable v en 2 y la variable x en el input? La respuesta es no. La razón es que *primero* debemos pedir el input con el comando $?x$.

Si pusiéramos el comando $?x$ dentro del cuerpo de c, ¿qué guarda utilizamos para marcar el ingreso al while? Si usamos la guarda $v \neq 2$, dejarmos sin alterar los estados en que $v = 2$, y por ende no pedimos ningún input.

$$\textbf{while } v \neq 2 \textbf{ do } c$$

3 Parcial 1: 21/06/2023

(2) Decidí V o F.

(a) Si D es dominio sin cadenas interesantes, entonces $D \to D$ tampoco tiene cadenas interesantes.

(b) Sea $f : P \to P'$ una función monótona entre predominios. Entonces, para cualquier cadena $\{x_i\}_{i \in \mathbb{N}}$ vale

$$\bigsqcup_{i \in \mathbb{N}} (f\ x_i) \leq f\left(\bigsqcup_{i \in \mathbb{N}} x_i\right)$$

(a) La afirmación es falsa. Tomemos como contraejemplo $D = \mathbb{N}_\perp$ el orden llano. Sabemos que no tiene cadenas interesantes. Sin embargo, considere

$$f_i\ n = \begin{cases} n & n \leq i \\ \perp & c.c. \end{cases}$$

Sabemos que $f_1 \leq f_2 \leq \ldots$ es cadena interesante cuyo supremo es la función identidad de $\mathbb{N}_\perp \to \mathbb{N}_\perp$.

$$f_1 \leq f_2 \leq \ldots$$

(b) La afirmación es verdadera.

Sea $x_1, x_2, \ldots$ una cadena en P. Entonces $f(x_1) \leq f(x_2) \leq \ldots$ es cadena en P'. Ahora bien, $x_j \leq \bigsqcup_{i \in \mathbb{N}} x_i$, y por lo tanto, por ser f monótona, $f(x_j) \leq f(\bigsqcup_{i \in \mathbb{N}} x_i)$. Por lo tanto, $f\left(\bigsqcup_{i \in \mathbb{N}} x_i\right)$ es cota superior de $f(x_1), f(x_2), \ldots$ Como el supremo de dicha cadena es la menor de las cotas superiores,

$$\bigsqcup_{i \in \mathbb{N}} f(x_i) \leq f\left(\bigsqcup_{i \in \mathbb{N}} x_i\right)$$

que es lo que dice la afirmación.

(4) (a) Proponé un programa que divida x por y si y solo si $y \neq 0$. En caso que $y = 0$ debe generar un error.

(b) Hacé otro programa c' que use c para dividr x por y, pero que si c falla solo cambie v por -1. En c' no podés usar condicionales ni ciclos, pero está bien usar c incluso si c tiene condicionales o ciclos.

(a) Proponemos

$$c := \textbf{if } y \neq 0 \textbf{ then } x/y \textbf{ else fail}$$

(b) El programa propuesto es

$$\textbf{catchin } c \textbf{ with } v := -1$$

4 Parcial 1 : 26/04/2024

(1) Sea $\mathbb{N}_\star$ el poset $(\mathbb{N}, \preceq)$ con $\preceq$ el orden tal que $a \preceq b \iff b \leq a$. (a) ¿Es un predominio? (b) Considerá $f = x \mapsto x$ visto como función en $\mathbb{N}_\star \to \mathbb{N}^\infty$. ¿Es monótona? (c) Definí una función monótona no constante en el espacio de funciones mencionado en (b). (d) ¿Hay funciones monótonas que no sean continuas en dicho espacio?

(a) Un predominio es un domino con un elemento mínimo. Observemos que toda cadena en $\mathbb{N}_\star$ es no-interesante y por ende se trata de un predominio. Sin embargo, no existe elemento mínimo. Por ende, no es un dominio.

(b) Sean $x, y \in \mathbb{N}_\star$ tales que $x \preceq y$. Si sigue por def. de $\preceq$ que $f(x) \leq f(y)$, donde $\leq$ es el orden usual que rige $\mathbb{N}^\infty$. Por lo tanto, f no es monótona.

(c) Sea

$$f\ n := \begin{cases} \infty & n = 0 \\ 0 & c.c. \end{cases}$$

Claramente no es constante. Probemos que es monótona. Para ello, tomemos $a, b \in \mathbb{N}$ tales que $a \preceq b$. Si ambos son distintos de cero, obtenemos $f(a) \leq f(b) \equiv 0 \leq 0 \equiv \textbf{True}$. Si ambos son cero, es fácil ver que el orden se preserva. Si $a = 0$ necesariamente se tiene $b = 0$, porque 0 es elemento máximo, con lo cual este caso está contemplado en el anterior. Si $a \neq 0, b = 0$, se tiene $f(a) \leq f(b) \equiv 0 \leq \infty \equiv \textbf{True}$. Esto basta para probar la monotonía de f.

(d) Sea $f \in \mathbb{N}_\star \to \mathbb{N}^\infty$ monótona. Como $\mathbb{N}_\star$ no tiene cadenas interesantes, toda cadena $f(x_1) \leq f(x_2) \leq \ldots$ derivada de una cadena $x_1 \preceq x_2 \preceq \ldots$ es no interesante. Y en particular, si x_k es el elemento que se repite en la cadena original, $f(x_k)$ será el elemento que se repita en la cada derivada por f. Es decir que toda cadena $x_1 \leq x_2 \leq \ldots$ con supremo x_k tiene una cadena asociada $f(x_1) \leq f(x_2) \leq \ldots$ con supremo $f(x_k)$. $\therefore$ La función f es continua.

Por lo tanto, toda f monótona es continua en el espacio dado.

Considerá el lenguaje imperativo con IO y la siguiente ecuación para $g \in \Sigma \to \Omega$:

$$g\ \sigma = \iota_{\text{in}} \left(\lambda\ k \in \mathbb{Z}. \begin{cases} \iota_{\text{term}}[\sigma \mid v : 0] & k = 0 \\ \iota_{\text{out}}\ \left\langle 1, g\left[\sigma \mid p : \sigma\ p + 1 \mid v : k\right]\right\rangle & k > 0 \\ \iota_{\text{out}}\left\langle -1, g\left[\sigma \mid n : \sigma\ n + 1 \mid v : k\right]\right\rangle & k < 0 \end{cases} \right)$$

(a) ¿Hay algún elemento mayor a la menor solución de g?

(b) ¿Puede un programa de la forma $?w; c'$ tener como semántica la menor solución de g?

(c) Proponé un programa cuya semántica coincide con la menor solución de g, justificando.

Primero, describamos qué hace el programa recursivo asociado a g. Claramente, toma un input y opera en función de si dicho input es igual, menor o mayor a cero. Si es igual a cero, la recursión termina y se devuelve σ transformado con $v = 0$. Si es menor a cero, se imprime 1 y se hace una llamada recursiva sobre el estado donde a p se le ha sumado 1 y a v se le ha asignado el input. Si es mayor a cero, se imprime -1 y se hace una llamada recursiva sobre el estado en que a n se le ha sumado 1 y a v se le ha asignado el input.

Podemos pensar que la función iterativamente cuenta cuántos inputs negativos se dieron (con la variable p), cuántos positivos (con la variable n), y termina cuando se da como input 0.

(a) Como la definición de g está definida para todos los casos posibles del input k, y para todos los estados posibles, no existe una función que sea equivalente pero provea más información. Por lo tanto, la menor solución de g es maximal.

(b) No, porque el programa cuya semántica es la menor solución de g cambia el valor de v en cada iteración (no el de w).

(c) El programa sería:

$?v$; **while** $y \neq 0$ **do**
  if $y > 0$ **then** $!1; p := p + 1$
  else $!(-1); n := n + 1;$
  $?v$

(4) Probar o refutar.

(a) Sean x, e tales que $x \notin FV(e)$. Entonces $x := e \equiv x := e; x := e$.

(b) **while** b **do** $(c; \textbf{fail}) \equiv \textbf{fail}$

(c) **while** b **do** $c \equiv$ **while** b **do** $c;$ **while** b **do** c.

(a) Es verdadero. Como x no es variable libre de e, cambiar el valor de x no modifica el valor de e. Es decir,

$$[\![e]\!]\,([\![x := e]\!]\sigma) = [\![e]\!]\sigma$$

Entonces

$$\begin{aligned}
[\![x := e; x := e]\!] &= [\![x := e]\!]_*\,([\![x := e]\!]\sigma) \\
&= [\![x := e]\!]_*\,([\sigma \mid x : [\![e]\!]\sigma]) \\
&= [[\sigma \mid x : [\![e]\!]\sigma] \mid x : [\![e]\!]([\sigma \mid x : [\![e]\!]\sigma])] \\
&= [\sigma \mid x : [\![e]\!]\sigma] \\
&= [\![x := e]\!]\sigma
\end{aligned}$$

(b) Falso porque en la primera iteración del **while** podría suceder que $c = \bot$, con lo cual $[\![c; \textbf{fail}]\!] = \bot$, con lo cual el **while** completo daría $\bot \neq [\![\textbf{fail}]\!]$.

(c) Estoy cansado señores.

5 Más práctica con los prácticos

(1) Decida si $H : (\Sigma \to \Sigma'_\perp) \to (\Sigma \to \Sigma'_\perp)$ es monótona y si es continua.

$$H\ f\ \sigma = \begin{cases} \sigma & f = \perp \\ \perp & c.c. \end{cases}$$

Sea ψ una función arbitraria in $\Sigma \to \Sigma'_\perp$ distinta de $\perp$. Claramente, $H\ \psi\ \sigma = \perp$ y $H\ \perp\ \sigma = \sigma$. Sin embargo, $\perp \leq \psi$ y $\perp \leq \sigma$. Es decir, no se preserva el orden. La función no es monótona. Por lo tanto no es continua.

Calcule la semántica de **while** $x > 0$ **do** $!x; c$ en los casos:

(a) $c \equiv$ **if** $x > 0$ **then** *skip* **else fail**

(b) $c \equiv$ **if** $x > 0$ **then fail else skip**

(a) En este caso,

$$[\![\textbf{while } x > 0 \textbf{ do } !x; c]\!]\sigma = [\![c]\!]_*\Big([\![\textbf{while } x > 0 \textbf{ do } !x]\!]\sigma\Big)$$

Veamos que

$$F\ f\ \sigma = \begin{cases} \sigma & \sigma\ x \leq 0 \\ f_*\Big([\![!x]\!]\sigma\Big) & \sigma\ x > 0 \end{cases}$$

tiene

$$F\ \bot\ \sigma = \begin{cases} \sigma & \sigma\ x \leq 0 \\ \bot & \sigma\ x > 0 \end{cases}$$

y

$$\begin{aligned} F^2\ \bot\ \sigma &= \begin{cases} \sigma & \sigma\ x \leq 0 \\ (F\ \bot)\Big([\![!x]\!]\sigma\Big) & \sigma\ x > 0 \end{cases} \\ &= \begin{cases} \sigma & \sigma\ x \leq 0 \\ [\![!x]\!]\sigma & x > 0 \wedge [\![!x]\!]\sigma\ x \leq 0 \\ \bot & \textit{c.c.} \end{cases} \\ &= \begin{cases} \sigma & x \leq 0 \\ \iota_{\text{out}}(\sigma\ x, \iota_{\text{term}}\ \sigma) & \sigma\ x > 0 \wedge \iota_{\text{out}}(\sigma\ x, \iota_{\text{term}}\ \sigma)\ x \leq 0 \\ \bot & \textit{c.c.} \end{cases} \\ &= \begin{cases} \sigma & \sigma\ x \leq 0 \\ \langle \sigma\ x, \sigma \rangle & x > 0 \wedge \sigma\ x \leq 0 \\ \bot & \textit{c.c.} \end{cases} \\ &= \begin{cases} \sigma & \sigma\ x \leq 0 \\ \bot & \textit{c.c.} \end{cases} \end{aligned}$$

Se aprecia lo que intuitivamente esperamos: es una cadena no interesante y por ende el supremo es el resultado antedicho. Por lo tanto, obtenemos

$$
\begin{aligned}
[\![\textbf{while } x > 0 \textbf{ do } !x; c]\!]\sigma &= [\![c]\!]_*\Big([\![\textbf{while } x > 0 \textbf{ do } !x]\!]\sigma\Big) \\
&= \begin{cases} [\![c]\!]_*\big(\sigma\big) & \sigma\ x \leq 0 \\ [\![c]\!]_*\bot & \sigma\ x > 0 \end{cases} \\
&= \begin{cases} [\![c]\!]\sigma & \sigma\ x \leq 0 \\ \bot & \textit{c.c.} \end{cases} \\
&= \begin{cases} [\![\textbf{if } x < 0 \textbf{ then skip else fail}]\!]\sigma & \sigma\ x \leq 0 \\ \bot & \textit{c.c.} \end{cases} \\
&= \begin{cases} [\![\textbf{skip}]\!]\sigma\ \sigma & \sigma\ x < 0 \\ [\![\textbf{fail}]\!]\sigma & \sigma\ x = 0 \\ \bot & \textit{c.c.} \end{cases} \\
&= \begin{cases} \sigma & \sigma\ x < 0 \\ \langle \textbf{abort}, \sigma \rangle & \sigma\ x = 0 \\ \bot & \textit{c.c.} \end{cases}
\end{aligned}
$$

Demostrar o refutar.

(a) $?x;?y \equiv ?y;?x$.

(b) $?x; z := x \equiv ?z$.

(c) **newvar** $x := e$ **in** $(?x; z := x) \equiv ?z$.

(a) Veamos que

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

El mismo desarrollo nos da

$$[\![?y;?x]\!]\sigma = \iota_{\text{in}}\Big(\lambda\ k \in \mathbb{Z}.\Big(\iota_{\text{in}}\left(\lambda n \in \mathbb{Z}.\left(\iota_{\text{term}}([\sigma \mid y : k \mid x : n])\right)\right)\Big)\Big)$$

No son la misma función.

(c) Veamos que

$$\begin{aligned}
[\![\textbf{newvar}\ x := e\ \textbf{in}\ (?x; z := x)]\!]\sigma &= \mathscr{R}_\dagger^{\sigma,x}\Big([\![?x; z := x]\!][\sigma \mid x : [\![e]\!]\sigma]\Big) \\
&= \mathscr{R}_\dagger^{\sigma,x}\Big([\![z := x]\!]_*\Big([\![?x]\!][\sigma \mid x : [\![e]\!]\sigma]\Big)\Big) \\
&= \mathscr{R}_\dagger^{\sigma,x}\Big([\![z := x]\!]_*\Big(\iota_{\text{in}}\Big(\lambda\ k \in \mathbb{Z}.\iota_{\text{term}}[[\sigma \mid x : [\![e]\!]\sigma] \mid x : k]\Big)\Big)\Big) \\
&= \mathscr{R}_\dagger^{\sigma,x}\Big([\![z := x]\!]_*\Big(\iota_{\text{in}}\Big(\lambda\ k \in \mathbb{Z}.\iota_{\text{term}}[\sigma \mid x : k]\Big)\Big)\Big) \\
&= \mathscr{R}_\dagger^{\sigma,x}\Big(\iota_{\text{in}}\Big(\lambda\ k \in \mathbb{Z}.[\![z := x]\!]_*\iota_{\text{term}}[\sigma \mid x : k]\Big)\Big) \\
&= \mathscr{R}_\dagger^{\sigma,x}\Big(\iota_{\text{in}}\Big(\lambda k \in \mathbb{Z}.\iota_{\text{term}}[\sigma \mid x : k \mid z : k]\Big)\Big) \\
&= \iota_{\text{in}}\Big(\lambda k \in \mathbb{Z}.\mathscr{R}_\dagger^{\sigma,x}\iota_{\text{term}}[\sigma \mid x : k \mid z : k]\Big) \\
&= \iota_{\text{in}}\Big(\lambda k \in \mathbb{Z}.\iota_{\text{term}}[\sigma \mid z : k]\Big) \\
&= [\![?z]\!]\sigma
\end{aligned}$$