1

Apellido y Nombre: Renison Iván
email (@mi.unc.edu.ar): ivan.renison@mi.unc.edu.ar
Nota:

Lenguajes y Compiladores **1er Parcial 2024 - 26 de abril de 2024**

1. Sea $\mathbb{N}_{\downarrow}$ el poset $(\mathbb{N}, \sqsubseteq)$ de los naturales con la relación $\geqslant$; es decir $x \sqsubseteq y$ si y sólo si $x \geqslant y$. Por ejemplo $4 \sqsubseteq 2$ porque $4 \geqslant 2$.
   (a) ¿Es $\mathbb{N}_{\downarrow}$ un predominio?
   (b) Considerá el mapeo $f \doteq x \mapsto x$ que lo podemos ver como una función en $\mathbb{N}_{\downarrow} \to \mathbb{N}^{\infty}$. ¿Es f una función monótona?
   (c) Definí una función monótona en $\mathbb{N}_{\downarrow} \to \mathbb{N}^{\infty}$ que no sea constante.
   (d) ¿Hay funciones monótonas en $\mathbb{N}_{\downarrow} \to \mathbb{N}^{\infty}$ que no sean continuas?

2. Considerá la siguiente ecuación recursiva:

$$f(x) = \begin{cases} 0 & \text{si } x < 10 \\ 1 + f(x/10) & \text{si } x \geqslant 10 \end{cases}$$

   Sea $F\colon (\mathbb{N} \to \mathbb{N}_{\perp}) \to (\mathbb{N} \to \mathbb{N}_{\perp})$ el funcional asociado a esa ecuación.
   (a) ¿Cuál es el menor $i \in \mathbb{N}$ para el cual existe $x \in \mathbb{N}$ tal que $F^i \perp x = 4$? Proponé, además, un x que cumpla esa propiedad.
   (b) ¿Es $F^i \perp$ una cadena interesante?
   (c) ¿Cuál es el supremo de esa cadena?

3. Considerá el lenguaje imperativo simple con IO y la siguiente ecuación recursiva:

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

   (a) ¿Hay algún elemento mayor a la menor solución de g?
   (b) ¿Puede un programa de la forma $?\mathsf{w}; c'$ tener como semántica la menor solución de g?
   (c) Proponé un programa cuya semántica coincida con la menor solución de g. Justificá que efectivamente sea así.

4. Probá o refutá los siguientes enunciados. Justificá tu respuesta.
   (a) Sean x y e tales que $\mathsf{x} \notin FV(e)$, entonces $\mathsf{x} := e \equiv \mathsf{x} := e; \mathsf{x} := e$.
   (b) **while** b **do** $(c;$ **fail**$) \equiv$ **fail**.
   (c) **while** b **do** $c \equiv$ (**while** b **do** c); **while** b **do** c.

Iván Reinson

Parcial 1
Lenguajes y compiladores

1)

a)

Si es un predominio porque todas las cadenas son aburridas ✓

Todas las cadenas son aburridas porque si una cadena quisiera crecer siempre, eventualmente llegaría a 0 que es el $\sqsubseteq$-máximo ✓

b)

f no es monotona ya que: ✓

$2 \sqsubseteq 1$

Pero

$f(2) = 2 \not\leq 1 = f(1)$

c)

$g : N_\downarrow \to N^\infty$

$g\ n = \begin{cases} n = 0 \to 1 \\ sino \to 0 \end{cases}$ ✓

d)

No hay

Sea h una función monótona de $N_\downarrow \to N^\infty$, vemos que h es continua

Sea $x_0, x_1, \ldots$ una cadena de $N_\downarrow$

Hay que ver $h(\bigsqcup_{i \in \mathbb{N}} x_i) = \bigsqcup_{i \in \mathbb{N}} h\, x_i$

Por lo visto en 2 $x_0, \dots$ es abortida

Sea $j \in \mathbb{N}$ talque $x_j = x_{j+1} = x_{j+2} \dots$ (*)

$\Rightarrow$

Tenemos:

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

$= h(x_j)$

$= \bigsqcup_{i \in \mathbb{N}} h(x_i)$

$\left.\begin{array}{l} \bigsqcup_{i \in \mathbb{N}} x_i = x_j \\ (*) \Rightarrow h x_j = h x_{j+1} = \dots \end{array}\right\}$ y h es monotona $\Rightarrow h x_0 \leq h x_1 \leq \dots \leq h x_j$

Por ende h es continua ✓

2)

$$F\ g\ x = \begin{cases} x < 10 \to 0 \\ \text{sino} \to 1 + g(x/10) \end{cases}$$

3)

$i = 5, \quad x = 10^1$

Tenemos que

$F^5 \perp 10^1 = F\ (F^4 \perp)\ 10^1$

$F\ (F \perp) = 1 + F^1 \perp 10^3$

$= 1 + F\ (F^3 \perp)\ 10^3$

$= 1 + 1 + F^3 \perp 10^2$

$= 2 + F\ (F^2 \perp)\ 10^2$

$= 2 + 1 + F^2 \perp 10$

$= 3 + F\ (F \perp)\ 10$

Con $i = 1$ o menos no alcanza porque la llamada a $F^{i-1} \perp$ tiene que hacerse 1 veces (y en $i = 1$ se hace máximo 3) ✓

Ivan
Renison

$= 3 + 1 + F\,\bot\,1$

$= 4 + 0$

$= 4$ ✓

b)

Tenemos que:

$F^i\,\bot\,x = \begin{cases} x < 10^i \to \lfloor \log_{10} x \rfloor \\ \text{sino} \to \bot \end{cases}$ excepto que $\log_{10} 0$ no está definido (*)

La cadena es interesante porque cada $F^i\,\bot$ da $\neq \bot$ en más valores que el anterior

c)

$\left(\bigcup_{i \in N} F^i\,\bot\right) x = \lfloor \log_{10} x \rfloor$ ✓

Esto ya que $\forall x \in N.\ \exists i \in N.\ x < 10^i$ y por ende siempre hay alguna $F^i\,\bot$ en la que se cumple la primera guarda

(*) Demostración:

El caso base es cierto porque $x < 10 \Rightarrow \lfloor \log_{10} x \rfloor = 0$

En el caso inductivo:

$F\,(F^i\,\bot)\,x = \begin{cases} x < 10 \to 0 \\ \text{sino} \to 1 + F^i\,\bot\,(x/10) \end{cases}$

$= \begin{cases} x < 10 \to \lfloor \log_{10} x \rfloor \\ \text{sino} \to 1 + \begin{cases} x/10 < 10^i \to \lfloor \log_{10} (x/10) \rfloor \\ \text{sino} \to \bot \end{cases} \end{cases}$

$= \begin{cases} x < 10^{i+1} \to \lfloor \log_{10} x \rfloor \\ \text{sino} \to \bot \end{cases}$

a)

No no hay porque no hay ningún ¿$\perp$? que convertir en $> \perp$, ya que todos los constructores que se usan (los c) no tienen un constructor mayor y por ende un elemento mayor tendría que ser el mismo o con un argumento mayor

b)

No porque ?w modificaría w para el caso en el que el programa termina ✓

c)

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

→ pero queremos "contar" los negativos en n y los positivos en p.

c

c'

$[\![?v; c']\!]\sigma = \mathrm{fix}(\lambda k. [\![c']\!][\sigma \mid v : k])$

$= \mathrm{fix}\left(\lambda k. \begin{cases} k = 0 \to [\sigma \mid v : k] \\ \text{sino} \to [\![c']\!]_*([\![c]\!][\sigma \mid v : k]) \end{cases}\right)$

si $\neg[\![b]\!]\sigma \Rightarrow [\![\text{while } b \text{ do } c]\!]\sigma = [\![c; \text{while } b \text{ do } c]\!]\sigma$

$= \mathrm{fix}\left(\lambda k. \begin{cases} k = 0 \to [\sigma \mid v : k] \\ k > 0 \to \mathrm{fout}(1, [\![p := p+1; ?v; c']\!][\sigma \mid v : k]) \\ \text{sino} \to \mathrm{fout}(-1, [\![p := p+1; ?v; c']\!][\sigma \mid v : k]) \end{cases}\right)$

$= \mathrm{fix}\left(\lambda k. \begin{cases} k = 0 \to [\sigma \mid v : k] \\ k > 0 \to \mathrm{fout}(1, [\![?v; c']\!][\sigma \mid v : k, p : \sigma p + 1]) \\ \text{sino} \to \mathrm{fout}(-1, [\![?v; c']\!][\sigma \mid v : k, p : \sigma p + 1]) \end{cases}\right)$

Esto muestra que $[\![?v; c']\!]$ satisface g, y es lo menor por ∂

Iván
Renison

1)

a) verdadero, demostración:

$[\![x := e; x := e]\!]\sigma$
$= [\![x := e]\!]_*([\![x := e]\!]\sigma)$
$= [\![x := e]\!]_*[\sigma \mid x : [\![e]\!]\sigma]$
$= [\![x := e]\!][\sigma \mid x : [\![e]\!]\sigma]$
$= [[\sigma \mid x : [\![e]\!]\sigma] \mid x : [\![e]\!][\sigma \mid x : [\![e]\!]\sigma]]$
$= [\sigma \mid x : [\![e]\!][\sigma \mid x : [\![e]\!]\sigma]]$ ← Teorema de coincidencia
$= [\sigma \mid x : [\![e]\!]\sigma]$
$= [\![x := e]\!]\sigma$

b) Falso, por ejemplo:
$b = true$, $c = x := 0$
Tenemos que:
$[\![\text{while } b \text{ do } c; \text{fail}]\!]\sigma$
$= [\![(c; \text{fail}); \text{while } b \text{ do } c; \text{fail}]\!]\sigma$
$= [\![\text{while } b \text{ do } c; \text{fail}]\!]_*([\![c; \text{fail}]\!]\sigma)$
$= [\![\text{while } b \text{ do } c; \text{fail}]\!]_*([\![\text{fail}]\!]_*([\![c]\!]\sigma))$
$= [\![\text{while } b \text{ do } c; \text{fail}]\!]_*([\![\text{fail}]\!][\sigma \mid x : 0])$
$= [\![\text{while } b \text{ do } c; \text{fail}]\!]_* \langle \text{abort}, [\sigma \mid x : 0] \rangle$
$= \langle \text{abort}, [\sigma \mid x : 0] \rangle$

Y:
$[\![\text{fail}]\!]\sigma = \langle \text{abort}, \sigma \rangle$

(trabajo en Lis solo)

c) verdadero, demostración llamemos $c' \equiv$ while b do c

$[\![(c'); c']\!]\sigma = [\![c']\!]\sigma$

$\Leftrightarrow$

$[\![c']\!]_*([\![c']\!]\sigma) = [\![c']\!]\sigma$

Si $[\![c']\!]\sigma = \bot$ es cierto porque:

$[\![c']\!]_* \bot = \bot$

Si no sea $\sigma' = [\![c']\!]\sigma$:

$[\![c']\!]_* \sigma'$

$= [\![c']\!]\sigma'$

$= \bigsqcup_{i \in \mathbb{N}} F^i \bot \sigma'$ Por ejercicio del práctico $\neg[\![b]\!]\sigma'$ (*)

$= \bigsqcup_{i \in \mathbb{N}}' \sigma'$ $\Rightarrow$ simple se entra en la guarda de F que devuelve σ'

$= \sigma'$

(*):

Si $\sigma' = [\![\text{while } b \text{ do } c]\!]\sigma \neq \bot$

entonces $\neg[\![b]\!]\sigma'$