Guía 2 - Funciones S-computables

Solución de un alumno

Verano 2021

Ejercicio 1

- $V_i \leftarrow k$

```
[A] V <- V - 1
    IF V != 0 GOTO A
    V <- V + 1
    ... (repetimos la instrucción k veces en total)
    V <- V + 1
```

- $V_i \leftarrow V_j + k$

```
[A] Vi <- Vi - 1
    IF Vi != 0 GOTO A
    Vi <- Vi + 1
    ... (repetimos la instrucción k veces en total)
    Vi <- Vi + 1

    Z1 <- Vj
[B] IF Z1 == 0 GOTO E
    Vi <- Vi + 1
    Z1 <- Z1 - 1
    GOTO B
```

- $IF\ V_I = 0\ GOTO\ L$

```
    Z1 <- Vi
    IF Z1 != 0 GOTO A
    Z1 <- 1
    IF Z1 != 0 GOTO L
[A]
```

- $GOTO\ L$

```
    Z1 <- 1
    IF Z1 != 0 GOTO L
```

Ejercicio 2

Ejercicio 3

Ejercicio 4

a)

- S_1 no tiene $V \leftarrow V + 1$

No podemos computar la función $f(x) = x + 1$

Dem: Suponemos que si, entonces el programa va a tener $d_1, d_2, ..., d_m$ estados siendo el d_m el estado del calculo final, y siendo $d_m[Y] = X1 + 1$.

Como el lenguaje solo tiene dos instrucciones, observamos ambas:

Si la instrucción es $V \leftarrow V - 1$ o $IF\ V \neq V'\ GOTO\ L$ entonces:

$$d_{i+1}[Y] \geq d_i[Y]$$

Entonces:

$$d_1[Y] = 0 \geq d_2[Y] \geq ... \geq d_m[Y]$$

Entonces necesariamente:

$$d_m[Y] = 0$$

Entonces demostramos que si el programa termina entonces va a calcular $f(x) = 0$

- S_2 no tiene $IF\ V \neq 0\ GOTO\ L$

No podemos computar el programa $f(x) \uparrow (\forall x)$:

Suponemos que si, entonces vamos a tener un programa de m instrucciones con $d_1, ..., d_m$ estados. Se comienza el programa con la descripción instantánea (1, σ). Como solo tenemos las instrucciones de sumar o restar una variable entonces, en cada instrucción sucede:

$$(n, \sigma_1) \rightarrow (n + 1, \sigma_2)$$

Entonces si se empieza en $(1, \sigma_1)$ luego de m pasos se termina en (m, σ_m) y el programa nunca se cuelga o indefine.

- S_3 no tiene $V \leftarrow V - 1$

No vamos a poder computar la función f(x,y) = (x - y).

---

b) Para demostrarlo podemos simplemente ver que las instrucciones

$$V \leftarrow V' IF\ V \neq V'\ GOTO\ L$$

La podemos realizar en S. Y que la instrucción:

$$V \leftarrow V - 1$$

La podemos hacer en S'.

Veamos lo primero: La primera instrucción puede hacerse:

```
[R] V1 <- V1 - 1
    IF V1 != 0 GOTO R
    IF V' != 0 GOTO A
    GOTO E
[A] V1 <- V1 + 1
    V' <- V' - 1
    IF V' != 0 GOTO A
[E]
```

Ahora veamos la segunda instrucción:

```
    Z1 <- V
    Z2 <- V'
[R] Z1 <- Z1 - 1
    Z2 <- Z2 - 1
    IF Z1 = 0 GOTO B
    IF Z2 = 0 GOTO C
    GOTO R

[B] IF Z2 = 0 GOTO L
    GOTO E

[C] IF Z1 = 0 GOTO L
    GOTO E

[E]
```

Ahora veamos que podemos hacer $V \leftarrow V - 1$ en S':

```
        Z1 <- Z1 + 1
[R] Z1 <- Z1 + 1
        Z2 <- Z2 + 1
        IF Z1 != V GOTO R

        V <- Z2
```

No vemos el caso donde V sea 1 o 0, pero es solo agregar unas guardas.

Ejercicio 5

a)

```
    T <- X_(N+1)
[R] IF p(X_1, ..., X_N, T) = 1 GOTO F
    T <- T + 1
    GOTO R
[F] Y <- T
```

b) La inversa de una función biyectiva existe y también es biyectiva. Podemos definir la inversa como:

$$f^{-1}(x) = min\{t : f(t) = x\}$$

Es total (es decir no se cuelga) porque para todo x existe t tal que $f(t) = x$.

Ejercicio 6

Nos tenemos que fijar que todas las instrucciones sean saltos condicionales y luego nos fijamos que la etiqueta a la que apunten no esté en ninguna instrucción anterior o sea de ella misma.

$$noApareceAntes(n, linea, etiqueta) = (\forall x)_{\leq linea} etiqueta \neq l((n+1)[x])$$

Que nos dice si la etiqueta no se uso ni antes ni en la misma instrucción.

$$inum(x, i) = l(r((x+1)[i]))$$

Nos devuelve el número de la instrucción i.

Entonces podemos escribir $r(x)$ como:

$$r(x) = (\forall i)_{\leq |x+1|} inum(x, i) > 2 \Rightarrow \neg noApareceAntes(x, i, inum(x, i) - 2)$$

Notar que:

$$\times_1 \Rightarrow \times_2 \equiv (\times_1 \wedge \times_2) \vee \neg \times_1$$

Ejercicio 7

damos un programa para cada caso.

f_1)

X1 $\leftarrow x$, X2 $\leftarrow y$

```
    Y <- 1
[B] IF STP(X2, X1, Z1) = 1 GOTO E
    Z1 <- Z1 + 1 //Z1 lo usamos como contador de pasos
    GOTO B
```

Otra forma:

$$f_1(x, y) = (\exists t)(STP(y, x, t) = 1)$$

$$f_2(x) = (\exists < y, t >)(STP(y, x, t) = 1)$$

$$f_3(x, v) = (\exists < v, t >)(STP(v, x, t) = 1 \wedge r(SNAP(v, x, t))[0] = y)$$

$$f_4(x, y) = (\exists < t_1, t_2, v_1, v_2 >)(STP(v_1, x, t_1) = 1 \wedge STP(v_2, y, t_2) = 1 \wedge r(SNAP(v_2, y, t_2))[0] = v_1)$$

Ejercicio 8

Ejercicio 9

$f(x) = (\exists \ < x_1, ..., x_n, t >\in \mathbb{N}) \ (\ x = r(SNAP(x_1, ..., x_n, x, t))[0] \ )$

La función $f(x)$ es pr, entonces es computable.

Notar que podemos escribir $(\exists t_1)(\exists t_2)...(\exists t_n)(f(t_1, ..., t_n))$ como $(\exists < t_1, ..., t_n > \in \mathbb{N})(f(t_1, ..., t_n))$

Notar que $r(SNAP(..., t))[0]$ es el valor de Y luego de t pasos.