Lenguaje S

DEFINICION 1: Variables

1. Variables de entrada: $X_1, X_2, X_3, \dots$
2. Variable de salida: Y (una! variable de salida)
3. Variables temporales/auxiliares: $Z_1, Z_2, Z_3, \dots$

Obs: variables auxiliares y la de salida inicializadas en 0

⚠ Tipo de datos = $\mathbb{N}$ (No hay otro tipo)

DEFINICION 2: Etiquetas

Las etiquetas se representan $A_1, B_1, C_1, D_1, E_1, A_2, B_2, \dots$

DEFINICION 3: Instruccion

A cada instruccion le puede anteceder o no una etiqueta ⇒ [L] Instruccion / L = etiqueta

1. Sea V variable ⇒ V ← V + 1 i.e. Si $V = n \in \mathbb{N}$, luego de ejecutarse la instruccion, $V = n+1$.
2. Sea V variable ⇒ V ← V - 1 i.e. Si $V = n \in \mathbb{N}$, luego $V = n-1$ (⚠ Si $V = 0$, luego queda ci el mismo valor $V = 0$).
3. Sea V variable y L etiqueta ⇒ IF V ≠ 0 GOTO L i.e. Si $V \neq 0$, se pasa a la 1er instruccion ci etiqueta L y si $V = 0$, se pasa a la proxima instruccion.

DEFINICION 4: Programa

Un programa es una lista finita de instrucciones $I_1, I_2, \dots, I_n$ q se escriben una debajo de la otra.

EJEMPLOS

EJEMPLO 1

[A_1] $X_1 \leftarrow X_1 - 1$
$Y \leftarrow Y + 1$
IF $X \neq 0$ GOTO A_1

⇒ este programa computa la funcion $f: \mathbb{N} \to \mathbb{N} / f(x) = \begin{cases} 1 & \text{si } x = 0 \\ x & \text{si } x \neq 0 \end{cases}$

Veamos que pasa en funcion de X_1:

X_1	2	1	0
Y	0	1	2

X_1	1	0
Y	0	1

X_1	0
Y	1

Obs: $X \leftarrow X+1$ es eq a $Y \leftarrow Y-1$
$Y \to Y-1$

⇒ dos programas ≠ pueden computar la misma funcion

EJEMPLO 2

$Y \leftarrow Y - 1$ ⇒ este programa computa la funcion $f: \mathbb{N} \to \mathbb{N} / f(x) = 0$

EJEMPLO 3

$Y \leftarrow Y + 1$
$Y \leftarrow Y + 1$
⋮ } k veces
$Y \leftarrow Y + 1$

⇒ este programa computa $f: \mathbb{N} \to \mathbb{N} / f(x) = k$

⚠ Y siempre esta PERO si no aparece queda $Y = 0$

EJEMPLO 4

[A_1] IF $X_1 \neq 0$ GOTO A_1 ⇒ este programa computa $f: \mathbb{N} \to \mathbb{N} / f(x) = \begin{cases} 0 & \text{si } x = 0 \\ f & \text{si } x \neq 0 \end{cases}$ dnd f = no definida

Obs: esta mal definida la funcion i.e. el dominio debera ser cero

EJEMPLO 5

[A_1] IF $X \neq 0$ GOTO B_1
$Z_1 \leftarrow Z_1 + 1$
IF $Z_1 \neq 0$ GOTO E_1

⇒ Z nunca vale 0 i.e. es eq a mandarlo directo a E_1

Obs: como la etiqueta E_1 no antecede ninguna instruccion, cdo busca dicha etiqueta el programa termina en 0

[B_1] $X_1 \leftarrow X_1 - 1$
$Y \leftarrow Y + 1$
$Z_1 \leftarrow Z_1 + 1$
IF $Z_1 \neq 0$ GOTO A_1

⇒ este programa computa la funcion identidad $f: \mathbb{N} \to \mathbb{N} / f(x) = x$

DEFINICION 5: Macros

Los macros son pseudoinstrucciones q representan un segmento de programa. Cada vez q en un programa P aparece una macro, hay q reemplazarla x el segmento de programa q representa, dicho segmento de programa se denomina expansion de la macro.

EJEMPLOS

EJEMPLO 1 Salto incondicional: GOTO L siendo L una etiqueta

Expansion: $V \leftarrow V + 1$
IF $V \neq 0$ GOTO L

⚠⚠ Variables auxiliares y etiquetas q uso pi expandir macros no deben aparecer en otras instrucciones del programa

EJEMPLO 2 Asignacion de cero: $V \leftarrow 0$

Expansion: [L] V ← V - 1
IF V ≠ 0 GOTO L

EJEMPLO 3 Asignacion de variable: $V_1 \leftarrow V_2$

Obs: no puedo usar la funcion identidad xq en ese programa X queda en 0 PERO en este programa queremos que X conserve su valor

⚠ La variable Z_1 NO debe aparecer en el resto del programa xq quiero q empiece en cero.

Expansion: $V_1 \leftarrow 0$
$[A_1]$ IF $V_2 \neq 0$ GOTO B_1
GOTO C_1
$[B_1]$ $V_2 \leftarrow V_2 - 1$
$V_1 \leftarrow V_1 + 1$
$Z_1 \leftarrow Z_1 + 1$
$[C_1]$ IF $Z \neq 0$ GOTO D_1
GOTO E_1
$[D_1]$ $Z_1 \leftarrow Z_1 - 1$
$V_2 \leftarrow V_2 + 1$
GOTO C_1

EJEMPLOS: Programas con macros

EJEMPLO 1 Funcion suma $f: \mathbb{N}^2 \to \mathbb{N} / f(x_1, x_2) = x_1 + x_2$

$Y \leftarrow X_1$
$Z_1 \leftarrow X_2$
} asignacion de variable

$[C_1]$ IF $Z_1 \neq 0$ GOTO B_1
GOTO E_1 ⇒ salto incondicional
$[B_1]$ $Z_1 \leftarrow Z_1 - 1$
$Y \leftarrow Y + 1$
GOTO C_1 ⇒ salto incondicional

EJEMPLO 2 Funcion producto $f: \mathbb{N}^2 \to \mathbb{N} / f(x_1, x_2) = x_1 . x_2$

$Z_2 \leftarrow X_2$
$[B_1]$ IF $Z_2 \neq 0$ GOTO A_1
GOTO E_1
$[A_1]$ $Z_2 \leftarrow Z_2 - 1$
$Z_1 \leftarrow X_1 + Y$
$Y \leftarrow Z_1$
GOTO B_1

EJEMPLO 3 Funcion sucesor $f: \mathbb{N} \to \mathbb{N} / f(x) = x + 1$

$Y \leftarrow X_1$
$Y \leftarrow Y + 1$

Ej.
- $f_1(x_1, x_2) = x_1$
- $f_2(x_1, x_2) = x_2$
- $f_2(x_1, x_2, x_3) = x_2$

EJEMPLO 4 Funcion proyeccion $f_j: \mathbb{N}^k \to \mathbb{N} / f_j(x_1, \dots, x_k) = x_j$ cl $1 \leq j \leq k$

Basta con cambiar el programa q computa la funcion identidad x_1 por la variable x_j si es la j-esima variable la que se quiere proyectar.

EJEMPLO 5

Sean $f, g: \mathbb{N} \to \mathbb{N} / \exists$ un programa para f y otro pl g ⇒ probar que existe un programa pl f+g.

⇒ en estas dos primeras macros vuelvo a inicializar Y en 0

$Z_1 \leftarrow f(X_1)$ ⇒ si expando la macro seria hacer el programa pl f y redireccionar la salida a Z_1
$Z_2 \leftarrow g(X_1)$ ⇒ macro q redirecciona la salida a Z_2
$Y \leftarrow Z_1 + Z_2$

DEFINICION 6: Estado de un programa

Un estado de un programa P es una lista de ecuaciones de la forma $V = m$, dnd V es una variable y m un numero. Hay una unica ecuacion/igualdad pl cada variable q aparece en P

Obs: es una lista finita

EJEMPLO

$[A_1]$ $X_1 \leftarrow X_1 - 1$
$Y \leftarrow Y + 1$
IF $X_1 \neq 0$ GOTO A_1

Estados de P:

(1) $X_1 = 2, Y = 8$
(2) $X_1 = 3, Y = 1, Z_2 = 10$
(3) $X_1 = 6, Y = 1, Z_1 = 0$
} NO es necesario q los estados sean alcanzados

Obs: el estado (2) NO es alcanzable

⚠ Las variables que aparecen en el programa deben estar en el estado.

(1) $X_1 = 2$ NO es estado pues falta Y

(2) $Y = 1, X = 2, Y = 3$ NO es estado pues hay 2 ecuaciones asociadas a la variable Y

DEFINICION 7: Descripcion instantanea (o Snapshot)

Sup q un programa P tiene longitud n i.e. consta de n instrucciones $I_1, I_2, \dots, I_n$. P/ un estado σ de P y un $i \in \{1, 2, \dots, n, n+1\}$ tenemos que el par (i, σ) es una descrip° instantanea de P, la cual llamamos terminal si $i = n+1$ dnd i dice a q instruccion apunta antes de ser ejecutada y en σ estan los valores de las variables antes de ejecutar la instruccion i.

Para una (i, σ) no terminal, se define a su sucesor (j, τ):

1. Si la i-esima instruccion es $V \leftarrow V+1 \Rightarrow j = i+1$ y $\tau = \sigma$ salvo q $V = m$ se reemplaza por $V = m+1$.

Obs: si $V = 0$, V no cambia y $\tau = \sigma$

2. Si la i-esima instruccion es $V \leftarrow V-1 \Rightarrow j = i+1$ y $\tau = \sigma$ salvo q $V = m$ se reemplaza por $V = m-1$.
3. Si la i-esima instruccion es IF $V \neq 0$ GOTO L $\Rightarrow \tau = \sigma$ y $j = i+1$ si $V = 0$ y $j = \min\{k / I_k \text{ tiene etiqueta } L\}$ si $V \neq 0$ PERO si $\nexists$ [L], $j = n+1$ i.e. termina el programa.

Obs: Si el programa esta bien, no tendrian q haber 2+ instrucciones c/ la misma etiqueta PERO en caso de q esto ocurra Sabemos a donde mandar a j

DEFINICION 8: Computo

Un computo de un programa P a partir de un snapshot d_1 es una lista finita $d_1, d_2, \dots, d_k$ de snapshots siendo d_{j+1} el sucesor de d_j y d_k es la terminal.

EJEMPLO

[A_1] $X_1 \leftarrow X_1 - 1$ $\Rightarrow$ Hallar un computo de P a partir de $d_1 = (1, X_1 = 2, Y = 0)$

$Y \leftarrow Y + 1$

IF $X_1 \neq 0$ GOTO A_1

- $d_2 = (2, X_1 = 1, Y = 0)$
- $d_3 = (3, X_1 = 1, Y = 1)$
- $d_4 = (1, X_1 = 1, Y = 1)$
- $d_5 = (2, X_1 = 0, Y = 1)$
- $d_6 = (3, X_1 = 0, Y = 2)$
- $d_7 = (4, X_1 = 0, Y = 2)$ $\Rightarrow$ termina el programa

DEFINICION 9: Estado inicial

Sea P un programa y sean $u_1, u_2, \dots, u_m$ numeros naturales. El estado inicial de P p/ dichos valores es:

$\Rightarrow X_j$ y Z_j si aparecen en el programa

$\sigma_1 = (X_1 = u_1, X_2 = u_2, \dots, X_m = u_m, X_j = 0 \ \forall j > m, Y = 0, Z_j = 0)$. Luego, la descripcion inicial es $(1, \sigma_1)$.

DEFINICION 10: Computo a partir del estado inicial

Sean P un programa, $u_1, u_2, \dots, u_m$ numeros naturales y σ_1 el estado inicial para ellos.

Existen dos posibilidades:

1. "Programa termina ante dichas entradas" $\Rightarrow \exists$ un computo de P a partir de $d_1 = (1, \sigma_1)$ i.e. $\exists d_1, d_2, \dots, d_k$ dnd d_k terminal $\Rightarrow \Psi_P^m(u_1, u_2, \dots, u_m) =$ valor de Y en d_k

2. "El programa no termina" $\Rightarrow \nexists$ un computo de P a partir de $d_1 = (1, \sigma_1) \Rightarrow \Psi_P^m(u_1, u_2, \dots, u_m) = \uparrow$ i.e. esta indefinido

$\Rightarrow$ ciclo infinito

DEFINICION 11: Funcion computable

Una funcion $f: \mathbb{N}^k \to \mathbb{N}$ es parcialmente computable si $\exists$ un programa P/ $f(u_1, u_2, \dots, u_k) = \Psi_P^k(u_1, \dots, u_k)$ i.e. Si $x = (x_1, \dots, x_k) \in \text{Dom } f \Rightarrow f(x_1, \dots, x_k) = \Psi_P^k(x_1, \dots, x_k)$ y si $x \notin \text{Dom } f \Rightarrow \Psi_P^k(x_1, \dots, x_k) = \uparrow$.

Una funcion $f: \mathbb{N}^k \to \mathbb{N}$ es computable si es parcialmente computable y total i.e. $\text{Dom } f = \mathbb{N}^k$.

EJEMPLO

[A_1] IF $X_1 \neq 0$ GOTO A_1

Notemos que $\Psi_P^1(x) = \begin{cases} 0 & \text{si } x = 0 \\ \uparrow & \text{si } x \neq 0 \end{cases}$ $\Rightarrow$ la funcion q computa es parcialmente computable pues $\text{Dom } f = \{0\}$

BIBLIOGRAFIA

1. Teoria de Automatas, HopCroft
2. Teoria basica de la computacion, Booksnear