LENGUAJE S

1. Elementos del Lenguaje

Un programa en lenguaje S es una secuencia finita de instrucciones. Se escribe una instrucción debajo de la otra. Para definir las mismas, es necesario primero definir variables y etiquetas. No hay invocación de programas, ni pasaje de parámetros.

1.1. Variables. Las variables no se declaran. El tipo de datos es $\mathbb{N}$. Se tienen 3 tipos de variables:

1. Variables de entrada:

$$X_1,\ X_2,\ X_3,\ X_4,\ \cdots$$

2. Variables temporales:

$$Z_1,\ Z_2,\ Z_3,\ Z_4,\ \cdots$$

3. Variable de salida:

$$Y$$

Las variables auxiliares y la de salida comienzan inicializadas en 0.

1.2. Etiquetas. Las etiquetas se representan de la siguiente manera:

$$A_1,\ B_1,\ C_1,\ D_1,\ E_1,\ A_2\ \cdots$$

1.3. Instrucciones. A cada instrucción la puede anteceder o no una etiqueta, la cual se escribe entre corchetes:

$$[L]\ Instruccion$$

Se tienen 3 tipos de instrucciones:

1. Sea V una variable,

$$V \leftarrow V + 1$$

Si la variable V tiene el valor $n \in \mathbb{N}$, luego de ejecutarse la instrucción va a tener el valor $n + 1$.

2. Sea V una variable,

$$V \leftarrow V - 1$$

Si la variable V tiene el valor $n \in \mathbb{N}_{\geq 1}$, luego de ejecutarse la instrucción va a tener el valor $n - 1$. Si la variable V tiene el valor 0, luego de ejecutarse la instrucción queda con el mismo valor.

3. Sea V una variable y L una etiqueta,

$$IF \quad V \neq 0 \quad GOTO \quad L$$

Si la variable V tiene el valor 0, se pasa a la próxima instrucción, si la variable V tiene un valor distinto de 0, se pasa a la primer instrucción del programa a la cual la antecede la etiqueta L.

2. Ejemplos de programas

Ejemplo 2.1. Dado el siguiente programa

$$\begin{gathered}[A_1] \quad X_1 \leftarrow X_1 - 1 \\ Y \leftarrow Y + 1 \\ IF\ X \neq 0\ \ GOTO\ \ A_1\end{gathered}$$

Cuando $X_1 = 0$ el programa termina, porque no hay una cuarta instrucción. Este programa computa la función $f : \mathbb{N} \to \mathbb{N}$,

$$f(x) = \begin{cases} 1, & x = 0; \\ x, & x \neq 0. \end{cases}$$

Ejemplo 2.2. Dado el siguiente programa

$$Y \leftarrow Y - 1$$

Este programa computa la función

$$f : \mathbb{N} \to \mathbb{N}, \quad f(x) = 0$$

Notemos que el siguiente programa, computa la misma función:

$$\begin{gathered}X_1 \leftarrow X_1 - 1 \\ Y \leftarrow Y - 1\end{gathered}$$

Conclusión: Dos programas distintos pueden computar la misma función.

Ejemplo 2.3. Dado el siguiente programa

$$\left.\begin{gathered}Y \leftarrow Y + 1 \\ Y \leftarrow Y + 1 \\ \vdots \\ Y \leftarrow Y + 1\end{gathered}\right\} k \text{ instrucciones}$$

Este programa computa la función $f : \mathbb{N} \to \mathbb{N}$, $f(x) = k$.

Ejemplo 2.4. Dado el siguiente programa

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

Este programa computa la función $f : \mathbb{N} \to \mathbb{N}$,

$$f(x) = \begin{cases} \uparrow, & x \neq 0; \\ 0, & x = 0. \end{cases}$$

donde $\uparrow$ significa que no está definida la función en dicho caso el programa no termina.

Ejemplo 2.5. Dado el siguiente programa

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

Como la etiqueta E_1 no antecede a ninguna instrucción, cuando busca dicha etiqueta el programa termina.
Este programa computa la función identidad $f : \mathbb{N} \to \mathbb{N}$, $f(x) = x$.

3. Macros

Las macros son pseudoinstrucciones que representan segmentos de programas. Cada vez que en un programa P aparece una macro, hay que reemplazarla por el segmento de programa que representa, dicho segmento de programa se denomina expansión de la macro. Hay que tener el cuidado de que las variables auxiliares y las etiquetas que se utilicen para expandir una macro, no aparezcan en otras instrucciones del programa.

3.1. Ejemplos de macros.

Ejemplo 3.1. La macro *salto incondicional* se representa como

$$GOTO\ \ L$$

siendo L una etiqueta. Y se expande de la siguiente manera:

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

Ejemplo 3.2. La macro *asignación de cero* se representa como

$$V \leftarrow 0$$

Y se expande de la siguiente manera:

$$[L]\ \ V \leftarrow V - 1$$
$$IF\ \ V \neq 0\ \ GOTO\ \ L$$

Ejemplo 3.3. La macro *asignación de variable* se representa como

$$V_1 \leftarrow V_2$$

Y se expande de la siguiente manera:

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

Recordemos que las etiquetas que aparecen en esta macro: A_1, B_1, C_1, D_1, E_1 tienen que ser etiquetas que no se usen en el programa donde se va a usar la macro, lo mismo tiene que pasar con la variable auxiliar Z_1.
Notemos que si cambiamos V_1 por Y y V_2 por X_1 y vemos la macro como un programa completo, el mismo computaría la función identidad, pero a diferencia de como lo hicimos antes, resguardaría l valor de entrada.

4. Ejemplos de programas con macros

Ejemplo 4.1. Función suma
El siguiente programa computa la función: $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N},\ \ f(x_1, x_2) = x_1 + x_2$.

$$Y \leftarrow X_1$$
$$Z_1 \leftarrow X_2$$
$$[C_1]\ IF\ \ Z_1 \neq 0\ \ GOTO\ \ B_1$$
$$GOTO\ E_1$$
$$[B_1]\ Z_1 \leftarrow Z_1 - 1$$
$$Y \leftarrow Y + 1$$
$$GOTO\ C_1$$

Ejemplo 4.2. Función producto
El siguiente programa computa la función: $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N},\ \ f(x_1, x_2) = x_1 x_2$.

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

Ejemplo 4.3. Función sucesor
El siguiente programa computa la función: $f : \mathbb{N} \to \mathbb{N},\ \ f(x) = x + 1$.

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

Ejemplo 4.4. Función proyección
Basta con cambiarle al programa que computa la función identidad X_1 por la variable X_i, si es la i-ésima variable la que se quiere proyectar.

5. Descripciones de programas

Estado de un programa
Un estado de un programa $\mathcal{P}$ es una lista de ecuaciones de la forma $V = m$, donde V es una variable y m un número. Hay una única ecuación para cada variable que aparece en $\mathcal{P}$.

Ejemplo 5.1. Dado el siguiente programa $\mathcal{P}$:

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

Los siguientes son estados de $\mathcal{P}$:
$X_1 = 2,\ Y = 1$
$X_1 = 3,\ Y = 1,\ Z_1 = 0$
$X_1 = 5,\ Y = 1,\ Z_1 = 6$
Notemos que no es necesario que los estados sean alcanzados.

Veamos ahora ejemplos de secuencias de ecuaciones que no son estados:
$X_1 = 2$. No es un estado porque falta una ecuación asociada a la variable Y.
$X_1 = 2,\ Z_1 = 3$. No es un estado porque falta una ecuación asociada a la variable Y.
$Y = 1,\ X_1 = 6,\ Y = 2$. No es un estado porque hay dos ecuaciones asociadas a la variable Y.

Descripción instantánea de un programa o Snapshot

Supongamos que un programa $\mathcal{P}$ tiene longitud n, es decir consta de n instrucciones: $I_1,\ I_2,\ \cdots, I_n$. Para un estado σ de $\mathcal{P}$ y un $i \in \{1,\ 2,\ \cdots, n+1\}$ tenemos qie el par $(i,\ \sigma)$ es una descripción instantánea de $\mathcal{P}$, la cual se llama terminal si i es igual a $n+1$. Para una $(i,\ \sigma)$ no terminal se puede definir su sucesor $(j,\ \tau)$:

1. Si la i-ésima instrucción de $\mathcal{P}$ es $V \leftarrow V+1$, entonces $j = i+1$, τ es σ salvo que $V = m$ se reemplaza por $V = m+1$.
2. Si la i-ésima instrucción de $\mathcal{P}$ es $V \leftarrow V-1$, entonces $j = i+1$, τ es σ salvo que $V = m$ se reemplaza por $V = m-1$ si m no es cero.
3. Si la i-ésima instrucción de $\mathcal{P}$ es $IF\ V \neq\ 0\ GOTO\ L$, entonces τ es σ y si en σ, $V = 0$, entonces $j = i+1$ y si $V \neq 0$ entonces $j = \begin{cases} min\{k : I_k\ tiene\ etiqueta\ L\}, & \text{si existe } [L]\ I_k \\ n+1, & \text{si no existe } [L]\ I_k \end{cases}$

Cómputo

Un cómputo de un programa $\mathcal{P}$ a partir de una descripción instantánea d_1 es una lista $d_1,\ d_2, \cdots,\ d_k$ de descripciones instantáneas de $\mathcal{P}$ tal que d_{i+1} es el sucesor de d_i para $i \in \{1, \cdots, k-1\}$, siendo d_k terminal.

Ejemplo 5.2. Para el siguiente programa $\mathcal{P}$:

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

el cómputo a partir de $d_1 = \{1,\ X_1 = 2,\ Y = 0\}$ es:
$d_2 = \{2,\ X_1 = 1,\ Y = 0\}$
$d_3 = \{3,\ X_1 = 1,\ Y = 1\}$
$d_4 = \{1,\ X_1 = 1,\ Y = 1\}$
$d_5 = \{2,\ X_1 = 0,\ Y = 1\}$
$d_6 = \{3,\ X_1 = 0,\ Y = 2\}$
$d_7 = \{4,\ X_1 = 0,\ Y = 2\}$

Estado inicial

Sea $\mathcal{P}$ un programa y sean $u_1,\ \cdots,\ u_m$ números dados. El estado inicial de $\mathcal{P}$ para $u_1,\ \cdots,\ u_m$ es el estado σ_1 que tiene $X_1 = u_1$, $X_2 = u_2$, $\cdots$, $X_m = u_m$, $Y = 0$ junto a $V = 0$ para toda V temporal que aparezca en $\mathcal{P}$, o cualquier otra variable que aparezca en $\mathcal{P}$ y no hayamos mencionado. Por la tanto la descripción inicial de $\mathcal{P}$ para $u_1,\ \cdots,\ u_m$ es $(1,\ \sigma_1)$

Cómputo a partir del estado inicial

Sea $\mathcal{P}$ un programa y sean $u_1, \cdots, u_m$ números dados y σ_1 el estado inicial para ellos. Existen dos posibilidades:

1. Hay un cómputo de $\mathcal{P}$, $d_1, \cdots, d_k$ siendo $d_1 = (1, \sigma_1)$.

   Notamos $\Psi^m_{\mathcal{P}}(u_1, \cdots, u_m)$ al valor de Y en la descripción instantánea d_k.

2. No hay tal cómputo, es decir existe una secuencia infinita $d_1, d_2, \cdots$ siendo $d_1 = (1, \sigma_1)$ y d_{i+1} el sucesor de d_i.

   Decimos que $\Psi^m_{\mathcal{P}}(u_1, \cdots, u_m)$ está indefinido, lo notamos $\Psi^m_{\mathcal{P}}(u_1, \cdots, u_m) = \uparrow$

Función computable

Una función f es parcialmente computable si existe un programa $\mathcal{P}$ tal que $f(u_1, \cdots, u_m) = \Psi^m_{\mathcal{P}}(u_1, \cdots, u_m)$ para todo $(u_1, \cdots, u_m) \in \mathbb{N}^m$.

Una función f es computable si es parcialmente computable y total. Total quiere decir que el dominio de f es $\mathbb{N}^m$.

Observación:

Notemos que el mismo programa puede servir para computar funciones de 1, 2, 3, $\cdots$, variables. Por lo tanto si en $\mathcal{P}$ aparece X_n y no aparece X_i con $i > n$ hay dos opciones:

1. Si solo se especifican $m < n$ valores de entrada, entonces $X_{m+1}, \cdots, X_n$ toman el valor 0.
2. Si se especifican $m > n$ valores de entrada, $\mathcal{P}$ ignorará $u_{n+1}, \cdots, u_m$.

Ejemplo 5.3. Sea el programa $\mathcal{P}$:

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

Notemos que $\Psi^1_{\mathcal{P}}(x) = \begin{cases} 0, & x = 0; \\ \uparrow, & x \neq 0. \end{cases}$

La función que computa es parcialmente computable pues su dominio es $\{0\}$.