Tema 4:
Programas y funciones computables

Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Lógica y Computabilidad
Curso 2010–11

Contenido

El Lenguaje *GOTO*

Es un modelo secuencial y determinista con conjunto de datos: $\mathbb{N}$.

- **Sintaxis**:
  1. Variables: $\begin{cases} \text{De entrada:} & X_1(=X), X_2, X_3, \ldots \\ \text{De salida:} & Y \\ \text{Auxiliares:} & Z_1(=Z), Z_2, Z_3, \ldots \end{cases}$
  2. Etiquetas: $A_1, B_1, C_1, D_1, E_1, A_2, B_2, C_2, D_2, E_2, \ldots$
     - Notación: $A = A_1$, $B = B_1$, $C = C_1$, $D = D_1$, $E = E_1$.
  3. Instrucciones: Para cada variable V y cada etiqueta L:
     - **Incremento**: $V \longleftarrow V + 1$
     - **Decremento**: $V \longleftarrow V - 1$
     - **Condicional**: $IF\ V \neq 0\ GOTO\ L$
     - **Skip**: $V \longleftarrow V$

     Estas instrucciones pueden ser **etiquetadas** con cualquier etiqueta K. Por ejemplo,

$$[K]\ V \longleftarrow V + 1$$
$$[K]\ IF\ V \neq 0\ GOTO\ L$$

Programas GOTO

- Un `programa` *GOTO* (un G–programa) es una sucesión finita de instrucciones del lenguaje *GOTO*, $P = I_1, I_2, .., I_n$, cuya última instrucción NO es $Y \longleftarrow Y$.
- Si $P = I_1, \ldots, I_n$ es un programa *GOTO* el número n se denomina longitud del programa y se denota por $|P|$.
- Caso límite: el *Programa Vacío* (con 0 instrucciones).
- Notación: Denotaremos por $\mathbf{GOTO_P}$ al conjunto de G-programas.
- Ejemplo:

$$p \quad \begin{cases} [A] & X \longleftarrow X - 1 \\ & Y \longleftarrow Y + 1 \\ & \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ A \end{cases}$$

- Cada programa expresa un procedimiento para calcular una cierta función (parcial) de $\mathbb{N}^n$ en $\mathbb{N}$. El modo en que cada programa lleva a cabo esto se determina mediante la **semántica** del lenguaje *GOTO*.

Descripciones instantáneas

Sea P un programa *GOTO*.

- Denotaremos por *VAR* al conjunto de las variables del lenguaje GOTO y mediante $var(P)$ al conjunto de variables que aparecen en P.
- Un **estado** de P es una aplicación $\sigma : VAR \longrightarrow \mathbb{N}$.
- Un estado es, esencialmente, una tabla variable–valor que guarda el valor de las variables del programa en un momento dado. Por ello, a veces, escribiremos $V = m$ en lugar de $\sigma(V) = m$.
- Una **descripción instantánea, (d.i.)** de P, es un par $s = (i, \sigma)$ donde $1 \leq i \leq |P| + 1$ y σ es un estado de P.
- Cuando $i = |p| + 1$, la **d.i.** se denomina **terminal**.
- Si $s = (i, \sigma)$, escribiremos $s(V)$ en lugar de $\sigma(V)$.

Semántica

Sea P un programa y $s = (j, \sigma)$ y $s' = (j', \sigma')$ descripciones instantáneas de P con $j \neq |p| + 1$.
Diremos que s' es la **siguiente** de s en la computación de P, y escribimos $s \vdash_p s'$, si:

- Caso 1: $I_j \equiv V \longleftarrow V + 1$, s(V)=m. Entonces $j' = j + 1$ y
  - σ' es σ sustituyendo $V = m$ por $V = m + 1$.
- Caso 2: $I_j \equiv V \longleftarrow V - 1$, s(V)=m. Entonces $j' = j + 1$ y
  - Si $m > 0$, σ' es σ sustituyendo $V = m$ por $V = m - 1$.
  - Si $m = 0$, $\sigma' = \sigma$
- Caso 3: I_j es de tipo SKIP. Entonces $j' = j + 1$ y $\sigma' = \sigma$.
- Caso 4: $I_j \equiv IF\ V \neq 0\ GOTO\ L$. Entonces $\sigma' = \sigma$ y
  - Si $s(V) = 0$, entonces $j' = j + 1$
  - Si $s(V) \neq 0$
    - Si existe k tal que I_k es la primera instrucción etiquetada con L, entonces $j' = k$
    - Si no existe tal k, $j' = |p| + 1$

Computación de un programa

Una **computación** de un programa P es una sucesión de d.i. de P, $s = s_1, \ldots, s_k, \ldots$ tal que:

- $s_1 = (1, \sigma)$ (y se denomina d.i. **inicial**);
- Para todo k, con s_k no terminal, existe $k+1$ tal que $s_k \vdash_p s_{k+1}$

Una computación s de p es finita si y sólo si existe k tal que s_{k+1} es terminal; en tal caso, escribiremos $P(s)\downarrow$ y diremos que dicha computación para y se realiza en k pasos. En caso contrario, la computación s es infinita; escribiremos $P(s)\uparrow$ y diremos que no para.

Lema. Para todo $P \in GOTO_P$ se verifica:

- Ausencia de bloqueo: Para toda $d.i.$, s, no terminal, existe s' tal que $s \vdash_p s'$.
- Determinismo: Para cada $d.i.$ inicial existe una única computación.

Ejemplo

$[A]$	$IF\ X \neq 0\ GOTO\ B$
	$Z \longleftarrow Z + 1$
	$IF\ Z \neq 0\ GOTO\ E$
$[B]$	$X \longleftarrow X - 1$
	$Y \longleftarrow Y + 1$
	$IF\ X \neq 0\ GOTO\ A$

$$
\begin{aligned}
s_1 &= (1, \{X = 3,\ Y = 0,\ Z = 0\}) \\
s_2 &= (4, \{X = 3,\ Y = 0,\ Z = 0\}) \\
s_3 &= (5, \{X = 2,\ Y = 0,\ Z = 0\}) \\
s_4 &= (6, \{X = 2,\ Y = 1,\ Z = 0\}) \\
s_5 &= (1, \{X = 2,\ Y = 1,\ Z = 0\}) \\
s_6 &= (4, \{X = 2,\ Y = 1,\ Z = 0\}) \\
s_7 &= (5, \{X = 1,\ Y = 1,\ Z = 0\}) \\
s_8 &= (6, \{X = 1,\ Y = 2,\ Z = 0\}) \\
s_9 &= (1, \{X = 1,\ Y = 2,\ Z = 0\}) \\
s_{10} &= (4, \{X = 1,\ Y = 2,\ Z = 0\}) \\
s_{11} &= (5, \{X = 0,\ Y = 2,\ Z = 0\}) \\
s_{12} &= (6, \{X = 0,\ Y = 3,\ Z = 0\}) \\
s_{13} &= (7, \{X = 0,\ Y = 3,\ Z = 0\})
\end{aligned}
$$

Funciones GOTO–computables

Sea P un G–programa. La función de aridad n calculada por P es la función $[\![P]\!]^{(n)} : \mathbb{N}^n \dashrightarrow \mathbb{N}$ definida como sigue:
Dado $\vec{a} = (a_1, \ldots, a_n) \in \mathbb{N}^n$,

1. Sea σ el estado de P dado por:
   - $\sigma(X_i) = a_i$ $(1 \leq i \leq n)$
   - $\sigma(V) = 0$ para toda $V \in VAR \setminus \{X_1, \ldots, X_n\}$.
2. Sea ahora $s_1 = (1, \sigma)$. Entonces

$$[\![P]\!]^{(n)}(\vec{a}) = \begin{cases} s_k(Y) & \texttt{si existe } s_1, \ldots, s_k \texttt{ finita de } p \\ & \texttt{a partir de } s_1 \\ \uparrow & \texttt{e.c.o.c.} \end{cases}$$

Definición. Diremos que una función $f : \mathbb{N}^n \dashrightarrow \mathbb{N}$ es *GOTO*–**computable** ($f \in G\text{–}COMP$), si existe un programa *GOTO*, P tal que $f = [\![P]\!]^{(n)}$.

- Designaremos por $G\text{–}COMP^{(n)}$ al conjunto de funciones computables de aridad n.

Ejemplos

- La función identidad $Id_N : \mathbb{N} \to \mathbb{N}$ es *GOTO*–computable.

$$\begin{array}{ll} & IF\ X \neq 0\ GOTO\ B \\ & Z \longleftarrow Z + 1 \\ & IF\ Z \neq 0\ GOTO\ E \\ [B] & X \longleftarrow X - 1 \\ & Y \longleftarrow Y + 1 \\ & IF\ X \neq 0\ GOTO\ B \end{array}$$

- La función vacía $f_\emptyset$ es *GOTO*–computable:

$$\begin{array}{ll} [A] & Z \longleftarrow Z + 1 \\ & IF\ Z \neq 0\ GOTO\ A \end{array}$$

- La función nula $\mathcal{O} : \mathbb{N} \longrightarrow \mathbb{N} \quad \mathcal{O}(x) = 0$ es *GOTO*–computable.

  La función nula es calculada por cualquier programa que *pare* siempre y en el que no aparezca la variable de salida Y.

Macros

Existen bloques de instrucciones que podemos considerar como "nuevas instrucciones" ya que podemos utilizarlos en cualquier programa para realizar una cierta subtarea. Veamos algunos ejemplos de esto:

- El bloque de instrucciones:

$$Z \longleftarrow Z + 1$$
$$IF\ Z \neq 0\ GOTO\ L$$

  nos permite realizar un salto incondicional a la instrucción etiquetada por L (o terminar la ejecución del programa).

- Podemos poner a cero una variable V mediante el siguiente bloque de instrucciones:

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

Macros (II)

- Es útil introducir abreviaturas para poder usar cómodamente los bloques anteriores en la descripción de un programa.
- Dichas abreviaturas se denominan **macros** y para poder usarlas debemos especificar con precisión la forma en que dichas abreviaturas se reemplazan por verdadero código.

$$\underbrace{GOTO\ L}_{\text{MACRO}} \Longrightarrow \underbrace{\left.\begin{array}{l} Z_k \longleftarrow Z_k + 1 \\ IF\ Z_k \neq 0\ GOTO\ L \end{array}\right\}}_{\text{EXPANSIÓN}}$$

$$\underbrace{V \longleftarrow 0}_{\text{MACRO}} \Longrightarrow \underbrace{\left\{\begin{array}{ll} [K] & V \longleftarrow V - 1 \\ & IF\ V \neq 0\ GOTO\ K \end{array}\right.}_{\text{EXPANSIÓN}}$$

- En el primer caso Z_k debe ser una variable que no aparece ne el programa en que realizamos la expansión. En el segundo caso K es una etiqueta que no aparece en dicho programa.

Macros. Asignación

La macro $V \longleftarrow V'$ asigna a V el valor de V', conservando V' su valor.

$$
\begin{array}{ll}
 & V \longleftarrow 0 \\
[A] & \mathit{IF}\ V' \neq 0\ \mathit{GOTO}\ B \\
 & \mathit{GOTO}\ C \\
[B] & V' \longleftarrow V' - 1 \\
 & V \longleftarrow V + 1 \\
 & Z \longleftarrow Z + 1 \\
 & \mathit{GOTO}\ A \\
[C] & \mathit{IF}\ Z \neq 0\ \mathit{GOTO}\ D \\
 & \mathit{GOTO}\ L \\
[D] & Z \longleftarrow Z - 1 \\
 & V' \longleftarrow V' + 1 \\
 & \mathit{GOTO}\ C \\
[L] & V \longleftarrow V
\end{array}
$$

Copia V' en V y en Z y Pone V' a cero

Copia Z en V' Pone Z a cero

Uso de las macros

La función suma: $Sum : \mathbb{N}^2 \longrightarrow \mathbb{N} \quad Sum(x, y) = x + y$

$$P_+ \left\{ \begin{array}{ll} & Y \longleftarrow X_1 \\ & Z \longleftarrow X_2 \\ [B] & IF\ Z \neq 0\ GOTO\ A \\ & GOTO\ E \\ [A] & Z \longleftarrow Z - 1 \\ & Y \longleftarrow Y + 1 \\ & GOTO\ B \end{array} \right.$$

Uso de macros (II)

La función producto. $Prod : \mathbb{N}^2 \longrightarrow \mathbb{N}$ $Prod(x, y) = x \cdot y$

$$p_{\times} \left\{ \begin{array}{ll} & Z_2 \longleftarrow X_2 \\ [B] & IF\ Z_2 \neq 0\ GOTO\ A \\ & GOTO\ E \\ [A] & Z_2 \longleftarrow Z_2 - 1 \\ & Z_1 \longleftarrow X_1 + Y \\ & Y \longleftarrow Z_1 \\ & GOTO\ B \end{array} \right.$$

Nota: $Z_1 \longleftarrow X_1 + Y$ es una **macro** ejecuta el programa *Suma* y asigna el resultado a la variable Z_1.

Expansión de macros (I)

Sean $f \in GCOMP^{(n)}$ y $P \in GOTO_P$ tales que $[\![P]\!]^{(n)} = f$.
Veamos una expansión para la macro: $W \longleftarrow f(V_1, \ldots, V_n)$.

1. Renombrando las etiquetas de P si fuese necesario, podemos suponer que la única etiqueta de salida de P es E y todas las demás etiquetas de P son $A_1, .., A_r$.
2. Renombrando variables podemos suponer que las únicas variables de entrada que aparecen en P son $X_1, \ldots, X_n$ y las únicas variables auxiliares $Z_1, \ldots, Z_k$.
3. Podemos expresar la situación anterior escribiendo

$$P \equiv P(Y, X_1, \ldots, X_n, Z_1, \ldots, Z_k; E, A_1, \ldots, A_r)$$

4. Dado $m \in \mathbb{N}$ podemos obtener un nuevo programa

$$Q_m \equiv P(Z_m, , Z_{m+1}, .., Z_{m+n}, Z_{m+n+1}, .., Z_{m+n+k}; E_m, A_{m+1}, .., A_{m+r})$$

sustituyendo en P cada variable y cada etiqueta por la correspondiente de Q_m.

Expansión de macros (II)

Para expandir la macro en un programa P', tomamos $m \in \mathbb{N}$ suficientemente grande para que Q_m y P' no tengan variables ni etiquetas comunes. Una vez determinado m reemplazamos la macro por:

$$W \longleftarrow f(V_1, \ldots, V_n) \rightsquigarrow \begin{cases} & Z_m \longleftarrow 0 \\ & Z_{m+1} \longleftarrow V_1 \\ & \vdots \\ & Z_{m+n} \longleftarrow V_n \\ & \vdots \\ & Z_{m+n+1} \longleftarrow 0 \\ & \vdots \\ & Z_{m+n+k} \longleftarrow 0 \\ & Q_m \\ [E_m] & W \longleftarrow Z_m \end{cases}$$

Macros condicionales

Si $P(V_1, \ldots, V_n)$ es un `predicado` G-computable:

$$\left.\begin{array}{c} \texttt{macroexpansión} \\ Z \longleftarrow P(V_1, \ldots, V_n) \\ \mathit{IF}\ Z \neq 0\ \mathit{GOTO}\ L \end{array}\right\} \rightsquigarrow \begin{array}{c} \texttt{macro} \\ \mathit{IF}\ P(V_1, \ldots, V_n) \neq 0\ \mathit{GOTO}\ L \end{array}$$

Ejemplo: $\mathit{IF}\ V = 0\ \mathit{GOTO}\ L$

$$V = 0 \equiv P(v) = \begin{cases} 1 & \texttt{si} \quad v = 0 \\ 0 & \texttt{si} \quad v \neq 0 \end{cases}$$

donde $P(v)$ es G-computable: $\begin{cases} \mathit{IF}\ X \neq 0\ \mathit{GOTO}\ E \\ Y \longleftarrow Y + 1 \end{cases}$