LOS PROGRAMAS WHILE: CONSTANTES, FUNCIONES Y PREDICADOS NECESARIOS

- la constante $\varepsilon \in \Sigma^*$ (palabra vacía)
- la función cdr: $\Sigma^* \to \Sigma^*$

$$cdr\ (x) = \begin{cases} v & \exists s \in \Sigma\ (x = s \bullet v) \\ \varepsilon & \text{c.c.} \end{cases}$$

- para cada $s \in \Sigma$ la función $cons_s$: $\Sigma^* \to \Sigma^*$

$$cons_s\ (x) = s \bullet x$$

- para cada $s \in \Sigma$ el predicado $car_s?$: $\Sigma^* \to B$

$$car_s?\ (x) = \begin{cases} true & \exists v \in \Sigma^*\ (x = s \bullet v) \\ false & \text{c.c.} \end{cases}$$

- el predicado nonem?: $\Sigma^* \to B$

$$nonem?\ (x) = \begin{cases} true & x \neq \varepsilon \\ false & \text{c.c.} \end{cases}$$

SINTAXIS DE LOS PROGRAMAS WHILE

- PROGRAMAS SIMPLES:

  * XI := ε;

  * XI := $cons_s$ (XJ);

  * XI := cdr (XJ);

  Donde $I, J \in N$ y $s \in \Sigma$

- PROGRAMAS COMPUESTOS:

  * P Q

  * if car_s? (XI) then P end if;

  * while nonem?(XI) loop P end loop;

  donde $I \in N$, $s \in \Sigma$ y $P, Q \in W$

EJEMPLOS DE PROGRAMAS-WHILE

- PI

```
X0:= ε;
X0:= cons_a(X0);
X0:= cons_b(X0);
X0:= cons_b(X0);
X0:= cons_a(X0);
X0:= cons_a(X0);
```

- PII

```
X0:= cons_a(X0);
while nonem?(X0) loop
      X0:= cons_a (X0);
end loop;
```

- PIII

```
X0:= cons_a(X5);
if car_a?(X2) then
      X0:= cons_a(X0);
      while nonem?(X2) loop
            if car_b?(X2) then
                  X2:= cons_b(X2);
            end if;
            X2:= cdr(X2);
      end loop;
end if;
if car_b?(X2) then
      X0:= cons_b(X4);
end if;
```

ESTADOS DE CÓMPUTO

- ÚLTIMA VARIABLE de un programa es la de índice más alto que aparece en el mismo
  - Siempre existe (no hay programas-while sin variables)
  - No indica cuántas variables usa en realidad el programa

  Si en Q aparecen las variables X3, X41, X0 y X12, su última variable es X41

- INSTRUCCIONES de un programa-while son:
  - Sus ASIGNACIONES:

    $XI := \varepsilon$ $XI := cons_s(XJ)$ $XI := cdr(XJ)$

  - Sus CONDICIONES:

    $car_s?(XI)$ $nonem?(XI)$

- ESTADO DE CÓMPUTO de un programa P cuya última variable es XK e $\overline{y} = (y_0, y_1, \ldots y_k)$ es un vector de k+1 palabras de Σ^*.
  - El estado de cómputo refleja el contenido de las variables de P en un momento cualquiera de su ejecución.
  - El estado de cómputo también incluye el contenido de variables que, en realidad, no son utilizadas por el programa

  Un estado de cómputo de Q incluye 42 variables, aunque en Q sólo aparecen 4

CÓMPUTOS

- CÓMPUTO para un programa P es una secuencia alterna de estados de cómputo e instrucciones que reflejan la ejecución del programa de principio a fin

$$(\varepsilon, abbb, bc, caba, \varepsilon, bba)$$

$$X5 := \varepsilon$$

$$(\varepsilon, abbb, bc, caba, \varepsilon, \varepsilon)$$

$$car_b?(X2)$$

$$(\varepsilon, abbb, bc, caba, \varepsilon, \varepsilon)$$

$$X3 := cdr(X1)$$

$$(\varepsilon, abbb, bc, bbb, \varepsilon, bba)$$

  - La ejecución de una asignación modifica la variable afectada en el estado de cómputo
  - La evaluación de una condición no altera el estado de cómputo

- Un cómputo para P empieza y queda determinado por su ESTADO INICIAL

- Si el programa termina, lo hace en un ESTADO FINAL y es un CÓMPUTO CONVERGENTE

- Si el programa no termina, el cómputo es infinito y se llama CÓMPUTO DIVERGENTE

FUNCIÓN COMPUTADA

- FUNCIÓN j-aria COMPUTADA por $P \in W$

$$[\varphi_P^j]$$

  - $\varphi_P^j: \Sigma^{*j} \longrightarrow \Sigma^*$ describe la *relación entre datos y resultado* que establecen los cómputos de P (cuya última variable es XK) cuando se le suministran j entradas

- Definimos $\varphi_P^j(\overline{y}) \cong \varphi_P^j(y_1, y_2, \ldots, y_j)$ según:

  - Estado inicial:

  a) si $k \geq j$:

$$V_0 = (\varepsilon, \overline{y}, \varepsilon, \ldots, \varepsilon)$$

  b) si $k \leq j$:

$$V_0 = (\varepsilon, y_1, y_2, \ldots, y_k)$$

  - Comportamiento de P:

  a) si el cómputo asociado a P y V_0 es *convergente*, con estado de cómputo final $V_1 = \overline{z} = (z_0, z_1, \ldots, z_j)$:

$$\varphi_P^j(\overline{y}) = z_0$$

  b) si el cómputo asociado a P y V_0 es *divergente*:

$$\varphi_P^j(\overline{y})\uparrow$$

EJEMPLO DE FUNCIÓN COMPUTADA POR UN PROGRAMA-WHILE

```
while nonem?(X1) loop
    if car_a?(X1) then X2:= cons_a(X2); end if;
    if car_b?(X1) then X2:= cons_b(X2); end if;
    X1:=cdr(X1);
end loop;
if car_a?(X2) then
    X0:= cons_a(X0);
    while nonem?(X2) loop
        if car_b?(X2) then X0:= cons_b(X0); end if;
        X2:= cdr(X2);
    end loop;
end if;
if car_b?(X2) then
    X0:= cons_b(X0);
    while nonem?(X2) loop
        if car_a?(X2) then X0:= cons_a(X0); end if;
        X2:= cdr(X2);
    end loop;
end if;
```

- Para un solo argumento:

$$\varphi_P^1(x) \cong \begin{cases} \varepsilon & x = \varepsilon \\ b^n a & \exists v\,(x = v \bullet a) \wedge |\,x\,|_b = n \\ a^n b & \exists v\,(x = v \bullet b) \wedge |\,x\,|_a = n \end{cases}$$

- Para dos argumentos:

$$\varphi_P^2(x, y) \cong \begin{cases} \varepsilon & x = \varepsilon \wedge y = \varepsilon \\ b^n a & x = \varepsilon \wedge \exists v\,(y = a \bullet v) \wedge |\,y\,|_b = n \\ a^n b & x = \varepsilon \wedge \exists v\,(y = b \bullet v) \wedge |\,y\,|_a = n \\ b^n a & \exists v\,(x = v \bullet a) \wedge |\,y\,|_b + |\,x\,|_b = n \\ a^n b & \exists v\,(x = v \bullet b) \wedge |\,y\,|_a + |\,x\,|_a = n \end{cases}$$

- Para tres (o más) argumentos:

$$\varphi_P^3(x, y, z) \cong \varphi_P^2(x, y)$$

WHILE-COMPUTABILIDAD

- La función $\Psi: \Sigma^{*j} \rightarrow \Sigma^*$ es WHILE-COMPUTABLE si existe $P \in W$ que cumpla $\varphi_P^j \cong \Psi$
  - Buscar P no es una tarea trivial
  - Puede que P no exista
  - Si existe P, NUNCA es único
- El predicado P: $\Sigma^{*j} \rightarrow B$ es WHILE-DECIDIBLE si su *función característica*:

$$C_P(\overline{y}) = \begin{cases} 'a_1' & P(\overline{y}) \\ \varepsilon & c.c. \end{cases}$$

es while-computable

- El conjunto $A \subseteq \Sigma^{*j}$ es WHILE-DECIDIBLE si su *función característica*:

$$C_A(\overline{y}) = \begin{cases} 'a_1' & (\overline{y}) \in A \\ \varepsilon & c.c. \end{cases}$$

es while-computable