

# Turing vence a von Neumann

Agustín Curto

FaMAF

2017

**Notación:** dados  $x_1, \dots, x_n \in \omega$  y  $\alpha_1, \dots, \alpha_m \in \Sigma^*$ , con  $n, m \in \omega$ , usaremos:

$$\|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\|$$

para denotar el estado

$$((x_1, \dots, x_n, 0, \dots), (\alpha_1, \dots, \alpha_m, \varepsilon, \dots))$$

Notese que por ejemplo:

$$\|x\| = ((x, 0, \dots), (\varepsilon, \dots)) \text{ Para } n = 1, m = 0$$

$$\|\diamond\| = ((0, \dots), (\varepsilon, \dots)) \text{ Para } n = m = 0$$

Ademas es claro que:

$$\|x_1, \dots, x_n, \alpha_1, \dots, \alpha_m\| = \|x_1, \dots, x_n, \overbrace{0, \dots, 0}^i, \alpha_1, \dots, \alpha_m, \overbrace{\varepsilon, \dots, \varepsilon}^j\|$$

cualesquiera sean  $i, j \in \omega$ .

**Probaremos:** Si  $f$  es una función  $\Sigma$ -mixta que es computada por un programa  $\mathcal{P} \in \text{Pro}^\Sigma$ , entonces existe una máquina de Turing determinística con unit  $M$  la cual computa a  $f$ .

**Definición:** Dado  $\mathcal{P} \in \text{Pro}^\Sigma$ , definamos:

$$N(\mathcal{P}) = \text{menor } k \in \mathbf{N} \text{ tal que las variables que ocurren en } \mathcal{P} \\ \text{están todas en la lista } N1, \dots, N\bar{k}, P1, \dots, P\bar{k}$$

Ejemplo: Sea  $\Sigma = \{\&, \#\}$ , si  $\mathcal{P}$  es el siguiente programa:

```
L1  N4  $\leftarrow$  N4 + 1  
    P1  $\leftarrow$  P1.&  
    IF N1  $\neq$  0 GOTO L1
```

entonces tenemos  $N(\mathcal{P}) = 4$

Sea  $\mathcal{P}$  un programa y sea  $k$  fijo y mayor o igual a  $N(\mathcal{P})$ . A continuación describiremos como puede construirse una maquina de Turing la cual simulara a  $\mathcal{P}$ . La construccion de la maquina simuladora dependera de  $\mathcal{P}$  y de  $k$ . Notese que cuando  $\mathcal{P}$  se corre desde algun estado de la forma

$$\|x_1, \dots, x_k, \alpha_1, \dots, \alpha_k\|$$

los sucesivos estados por los que va pasando son todos de la forma

$$\|y_1, \dots, y_k, \beta_1, \dots, \beta_k\|$$

es decir en todos ellos las variables con indice mayor que  $k$  valen 0 o  $\varepsilon$ . La razon es simple: ya que en  $\mathcal{P}$  no figuran las variables

$$\begin{aligned} &\overline{Nk+1}, \overline{Nk+2}, \dots \\ &\overline{Pk+1}, \overline{Pk+2}, \dots \end{aligned}$$

estas variables quedan con valores 0 y  $\varepsilon$ , respectivamente a lo largo de toda la computacion.

Necesitaremos tener alguna manera de representar en la cinta los diferentes estados por los cuales se va pasando, a medida que corremos a  $\mathcal{P}$ . Esto lo haremos de la siguiente forma: al estado

$$\|x_1, \dots, x_k, \alpha_1, \dots, \alpha_k\|$$

lo representaremos en la cinta de la siguiente manera

$$B \mid^{x_1} \dots B \mid^{x_k} B \alpha_1 \dots B \alpha_k B B B B \dots$$

Por ejemplo consideremos el programa  $\mathcal{P}$  mostrado recién y fijemos  $k = 6$ . Entonces al estado

$$\|3, 2, 5, 0, 4, 2, \&, \&\&, \varepsilon, \#\&, \#, \#\#\#\|$$

lo representaremos en la cinta de la siguiente manera

$$B \mid\mid\mid B \mid\mid B \mid\mid\mid\mid BB \mid\mid\mid B \mid\mid B \& B \&\& BB \# \& B \# B \# \# \# B B B B B \dots$$

A lo que queda entre dos blancos consecutivos (es decir que no hay ningún blanco entre ellos) lo llamaremos "bloque", por ejemplo en la cinta de arriba tenemos que los primeros 12 bloques son

III II IIII  $\varepsilon$  IIII II & &&  $\varepsilon$  #& # ###

y después los bloques siguientes (que son infinitos ya que la cinta es infinita hacia la derecha) son todos iguales a  $\varepsilon$ .

Una observación importante es que esta forma de representación de estados en la cinta depende del  $k$  elegido, es decir si tomáramos otro  $k$ , por ejemplo  $k = 9$ , entonces el estado anterior se representaría de otra forma en la cinta.

Armaremos la máquina simuladora como concatenación de máquinas, cada una de las cuales simularán, vía la representación anterior, el funcionamiento de las distintas instrucciones de  $\mathcal{P}$ . Para esto, a continuación describiremos para los distintos tipos de instrucciones posibles de  $\mathcal{P}$ , sus respectivas máquinas asociadas. Asumiremos que en  $\mathcal{P}$  no hay instrucciones de la forma  $\text{GOTO } L\bar{m}$  ni de la forma  $L\bar{n} \text{ GOTO } L\bar{m}$ . Esto simplificará un poco la construcción de la máquina simuladora y de hecho lo podemos hacer ya que toda función  $\Sigma$ -computable puede ser computada por un programa sin este tipo de instrucciones, tal como lo veremos en un lema más adelante.

En esta etapa solo describiremos que propiedades tendrá que tener cada máquina simuladora de cada tipo posible de instrucción, y más adelante mostraremos como pueden ser construidas efectivamente dichas máquinas. Todas las máquinas descritas tendrán a  $\sqcup$  como unit y a  $B$  como blanco, tendrán a  $\Sigma$  como su alfabeto terminal y su alfabeto mayor será  $\Gamma = \Sigma \cup \{B, \sqcup\} \cup \{\tilde{a} : a \in \Sigma \cup \{\sqcup\}\}$ . Además tendrán uno o dos estados finales con la propiedad de que si  $q$  es un estado final, entonces  $\delta(q, \sigma) = \emptyset$ , para cada  $\sigma \in \Gamma$ . Esta propiedad es importante ya que nos permitirá concatenar pares de dichas máquinas identificando algún estado final de la primera con el inicial de la segunda.

- Para  $1 \leq i \leq k$ , sea  $M_i^+$  una máquina tal que:

$$\begin{array}{ccc}
 B_{|^{x_1}} B \dots B_{|^{x_k}} B\alpha_1 B \dots B\alpha_k & \vdash^* & B_{|^{x_1}} B \dots B_{|^{x_{i-1}}} B_{|^{x_i+1}} B_{|^{x_{i+1}}} \dots \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$

- Para  $1 \leq i \leq k$ , sea  $M_i^-$  una máquina tal que:

$$\begin{array}{ccc}
 B_{|^{x_1}} B \dots B_{|^{x_k}} B\alpha_1 B \dots B\alpha_k & \vdash^* & B_{|^{x_1}} B \dots B_{|^{x_{i-1}}} B_{|^{x_i-1}} B_{|^{x_{i+1}}} \dots \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$

- Para  $1 \leq i \leq k$  y  $a \in \Sigma$ , sea  $M_i^a$  una máquina tal que:

$$\begin{array}{ccc}
 B_{|^{x_1}} B \dots B_{|^{x_k}} B\alpha_1 B \dots B\alpha_k & \vdash^* & B_{|^{x_1}} B \dots B_{|^{x_k}} B\alpha_1 B \dots B\alpha_{i-1} B\alpha_i \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$



- Para  $1 \leq i \leq k$ , sea  $M_i^{\curvearrowright}$  una máquina tal que:

$$\begin{array}{ccc} B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \stackrel{*}{\vdash} & B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_{i-1} B^{\curvearrowright} \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

- Para  $j = 1, \dots, k$ , y  $a \in \Sigma$ , sea  $IF_j^a$  una máquina con dos estados finales  $q_{si}$  y  $q_{no}$  tal que:  
Si  $\alpha_j$  comienza con  $a$ , entonces:

$$\begin{array}{ccc} B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \stackrel{*}{\vdash} & B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{si} \end{array}$$

- Caso contrario:

$$\begin{array}{ccc} B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \stackrel{*}{\vdash} & B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{no} \end{array}$$

Análogamente, para  $j = 1, \dots, k$ , sea  $IF_j$  una máquina tal que:

- Si  $x_j \neq 0$ , entonces:

$$\begin{array}{ccc} B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \vdash^* & B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{si} \end{array}$$

- Si  $x_j = 0$ , entonces:

$$\begin{array}{ccc} B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \vdash^* & B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k \\ \uparrow & & \uparrow \\ q_0 & & q_{no} \end{array}$$

- Para  $1 \leq i, j \leq k$ , sea  $M_{i \leftarrow j}^*$  una máquina tal que:

$$\begin{array}{ccc} B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \vdash^* & B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_{i-1} B \alpha_i \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

- Para  $1 \leq i, j \leq k$ , sea  $M_{i \leftarrow j}^\#$  una máquina tal que:

$$\begin{array}{ccc}
 B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \stackrel{*}{\vdash} & B \mid^{x_1} B \dots B \mid^{x_{i-1}} B \mid^{x_j} B \mid^{x_{i+1}} B \dots B \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$

- Para  $1 \leq i \leq k$ , sea  $M_{i \leftarrow 0}$  una máquina tal que:

$$\begin{array}{ccc}
 B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \stackrel{*}{\vdash} & B \mid^{x_1} B \dots B \mid^{x_{i-1}} B B \mid^{x_{i+1}} B \dots B \mid^{x_k} \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$

- Para  $1 \leq i \leq k$ , sea  $M_{i \leftarrow \varepsilon}$  una máquina tal que:

$$\begin{array}{ccc}
 B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k & \stackrel{*}{\vdash} & B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_{i-1} B B \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$

A continuacion veremos un ejemplo de como se arma la maquina simuladora de un programa dado. Sea  $\Sigma = \{\&, \#\}$  y sea  $\mathcal{P}$  el siguiente programa

```
L3  N4  $\leftarrow$  N4 + 1  
    P1  $\leftarrow$   $\neg$ P1  
    IF P1 BEGINS & GOTO L3  
    P3  $\leftarrow$  P3.#
```

Tomemos  $k = 5$ . Es claro que  $k \geq N(\mathcal{P}) = 4$ . A la maquina que simulara a  $\mathcal{P}$  respecto de  $k$ , la llamaremos  $M_{sim}$  y sera la siguiente:

Figura 8

Veamos con un ejemplo como  $M_{sim}$  simula a  $\mathcal{P}$ . Supongamos que corremos  $\mathcal{P}$  desde el estado

$$\|2, 1, 0, 5, 3, \#\&\#\#, \varepsilon, \&\&, \#\&, \#\|$$

Tendremos entonces la siguiente sucesion de descripciones instantaneas:

$$(1, \|2, 1, 0, 5, 3, \#\&\#\#, \varepsilon, \&\&, \#\&, \#\|)$$

$$(2, \|2, 1, 0, 6, 3, \#\&\#\#, \varepsilon, \&\&, \#\&, \#\|)$$

$$(3, \|2, 1, 0, 6, 3, \&\#\#, \varepsilon, \&\&, \#\&, \#\|)$$

$$(1, \|2, 1, 0, 6, 3, \&\#\#, \varepsilon, \&\&, \#\&, \#\|)$$

$$(2, \|2, 1, 0, 7, 3, \&\#\#, \varepsilon, \&\&, \#\&, \#\|)$$

$$(3, \|2, 1, 0, 7, 3, \#\#, \varepsilon, \&\&, \#\&, \#\|)$$

$$q_0B \mid {}^2B \mid BB \mid {}^5B \mid {}^3B \# \& \# \# BB \& \& B \# \& B \# B$$

$$q_1B \mid {}^2B \mid BB \mid {}^6B \mid {}^3B \# \& \# \# BB \& \& B \# \& B \# B$$

$$q_2B \mid {}^2B \mid BB \mid {}^6B \mid {}^3B \# \& \# \# BB \& \& B \# \& B \# B$$

$$q_3B \mid {}^2B \mid BB \mid {}^6B \mid {}^3B \& \# \# BB \& \& B \# \& B \# B$$

$$q_4B \mid {}^2B \mid BB \mid {}^6B \mid {}^3B \& \# \# BB \& \& B \# \& B \# B$$

$$q_{si}B \mid {}^2B \mid BB \mid {}^6B \mid {}^3B \& \# \# BB \& \& B \# \& B \# B$$

$$q_0B \mid {}^2B \mid BB \mid {}^6B \mid {}^3B \& \# \# BB \& \& B \# \& B \# B$$

$$q_1B \mid {}^2B \mid BB \mid {}^7B \mid {}^3B \& \# \# BB \& \& B \# \& B \# B$$

$$q_2B \mid {}^2B \mid BB \mid {}^7B \mid {}^3B \& \# \# BB \& \& B \# \& B \# B$$

$$q_0B \mid {}^2B \mid BB \mid {}^7B \mid {}^3B \# \# \& \& B \# \& B \# B$$

Ahora describiremos en general como puede armarse la maquina simuladora de  $\mathcal{P}$ , respecto de  $k$ . Supongamos que  $\mathcal{P} = I_1 \dots I_n$ . Para cada  $i = 1, \dots, n$ , llamaremos  $M_i$  a la maquina que simulara el efecto que produce la instruccion  $I_i$ , es decir tomemos



## Proof.

- $M_i = M_{j,k}^+$ , si  $Bas(l_i) = N\bar{j} \leftarrow N\bar{j} + 1$
- $M_i = M_{j,k}^-$ , si  $Bas(l_i) = N\bar{j} \leftarrow N\bar{j} - 1$
- $M_i = M_{j,k}^a$ , si  $Bas(l_i) = P\bar{j} \leftarrow P\bar{j}.a$
- $M_i = M_{j,k}^\cap$ , si  $Bas(l_i) = P\bar{j} \leftarrow \cap P\bar{j}$
- $M_i = M_{j \leftarrow m}^{\#,k}$ , si  $Bas(l_i) = N\bar{j} \leftarrow N\bar{m}$
- $M_i = M_{j \leftarrow m}^{*,k}$ , si  $Bas(l_i) = P\bar{j} \leftarrow P\bar{m}$
- $M_i = M_{j \leftarrow 0}^k$ , si  $Bas(l_i) = N\bar{j} \leftarrow 0$
- $M_i = M_{j \leftarrow \varepsilon}^k$ , si  $Bas(l_i) = P\bar{j} \leftarrow \varepsilon$
- $M_i = M_{\text{SKIP}}$ , si  $Bas(l_i) = \text{SKIP}$
- $M_i = IF_{j,k}$ , si  $Bas(l_i) = \text{IF } N\bar{j} \neq 0 \text{ GOTO } L\bar{m}$ , para algun  $m$
- $M_i = IF_{j,k}^a$ , si  $Bas(l_i) = \text{IF } P\bar{j} \text{ BEGINS } a \text{ GOTO } L\bar{m}$ , para algun  $m$



Ya que la maquina  $M_i$  puede tener uno o dos estados finales, la representaremos de la siguiente manera

Figura 5

entendiendo que en el caso en que  $M_i$  tiene un solo estado final, este esta representado por el circulo de abajo a la izquierda y en el caso en que  $M_i$  tiene dos estados finales, el estado final representado con lineas punteadas (es decir, el de la derecha) corresponde al estado  $q_{si}$  y el otro al estado  $q_{no}$ .

Para armar la maquina que simulara a  $\mathcal{P}$  hacemos lo siguiente. Primero unimos las maquinas  $M_1, \dots, M_n$  de la siguiente manera

Nueva Figura 6

Luego para cada  $i$  tal que  $Bas(l_i)$  es de la forma  $\alpha GOTO L\bar{m}$ , ligamos con una flecha de la forma

$$\xrightarrow{B, B, K}$$

el estado final  $q_{si}$  de la  $M_i$  con el estado inicial de la  $M_h$ , donde  $h$  es tal que  $l_h$  es la primer instruccion que tiene label  $L\bar{m}$ .

## Lemma

Sea  $\mathcal{P} \in \text{Pro}^\Sigma$  y sea  $k \geq N(\mathcal{P})$ . Supongamos que en  $\mathcal{P}$  no hay instrucciones de la forma GOTO  $L\bar{m}$  ni de la forma  $L\bar{n}$  GOTO  $L\bar{m}$ . Para cada  $a \in \Sigma \cup \{1\}$ , sea  $\tilde{a}$  un nuevo simbolo. Sea  $\Gamma = \Sigma \cup \{B, 1\} \cup \{\tilde{a} : a \in \Sigma \cup \{1\}\}$ . Entonces hay una maquina de Turing deterministica con unit  $M = (Q, \Gamma, \Sigma, \delta, q_0, B, 1, \{q_f\})$  la cual satisface

- (1)  $\delta(q_f, \sigma) = \emptyset$ , para cada  $\sigma \in \Gamma$ .
- (2) Cualesquiera sean  $x_1, \dots, x_k \in \omega$  y  $\alpha_1, \dots, \alpha_k \in \Sigma^*$ , el programa  $\mathcal{P}$  se detiene partiendo del estado

$$\|x_1, \dots, x_k, \alpha_1, \dots, \alpha_k\|$$

sii  $M$  se detiene partiendo de la descripcion instantanea

$$[q_0 B \mid^{x_1} B \dots B \mid^{x_k} B \alpha_1 B \dots B \alpha_k B]$$

- (3) Si  $x_1, \dots, x_k \in \omega$  y  $\alpha_1, \dots, \alpha_k \in \Sigma^*$  son tales que  $\mathcal{P}$  se detiene partiendo del estado

En esta etapa nos dedicaremos a explicar como construir las distintas maquinas simuladoras de instrucciones.

Para poder hacer concretamente las maquinas recién descritas deberemos diseñar antes algunas maquinas auxiliares. Para cada  $j \geq 1$ , sea  $D_j$  la maquina siguiente

Figura 1

Notese que

$$\begin{array}{ccc}
 \alpha B \beta_1 B \beta_2 B \dots B \beta_j B \gamma & \stackrel{*}{\vdash} & \alpha B \beta_1 B \beta_2 B \dots B \beta_j B \gamma \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$

siempre que  $\alpha, \gamma \in \Gamma^*$ ,  $\beta_1, \dots, \beta_j \in (\Gamma - \{B\})^*$ . Es decir la maquina  $D_j$  lo unico que hace es mover el cabezal desde el blanco de la izquierda de un bloque determinado, exactamente  $j$  bloques a la derecha

Analogamente  $I_j$  sera una maquina que desplaza el cabezal  $j$  bloques a la izquierda del blanco que esta escaneando. Es decir  $I_j$  cumplira que

$$\begin{array}{ccc}
 \alpha B \beta_j B \dots B \beta_2 B \beta_1 B \gamma & \stackrel{*}{\vdash} & \alpha B \beta_j B \dots B \beta_2 B \beta_1 B \gamma \\
 \uparrow & & \uparrow \\
 q_0 & & q_f
 \end{array}$$

siempre que  $\alpha, \gamma \in \Gamma^*$ ,  $\beta_1, \dots, \beta_j \in (\Gamma - \{B\})^*$ . Dejamos al lector la manufactura de esta maquina.

Para  $j \geq 1$ , sea  $TD_j$  una maquina con un solo estado final  $q_f$  y tal que

$$\begin{array}{ccc} \alpha B \gamma & \overset{*}{\vdash} & \alpha B B \gamma \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

cada vez que  $\alpha, \gamma \in \Gamma^*$  y  $\gamma$  tiene exactamente  $j$  ocurrencias de  $B$ . Es decir la maquina  $TD_j$  corre un espacio a la derecha todo el segmento  $\gamma$  y agrega un blanco en el espacio que se genera a la izquierda. Por ejemplo, para el caso de  $\Sigma = \{\&, \#\}$  podemos tomar  $TD_3$  igual a la siguiente maquina

Nueva Figura 3



Analogamente, para  $j \geq 1$ , sea  $TI_j$  una maquina tal que

$$\begin{array}{ccc} \alpha B \sigma \gamma & \overset{*}{\vdash} & \alpha B \gamma \\ \uparrow & & \uparrow \\ q_0 & & q_f \end{array}$$

cada vez que  $\alpha \in \Gamma^*$ ,  $\sigma \in \Gamma$  y  $\gamma$  tiene exactamente  $j$  ocurrencias de  $B$ . Es decir la maquina  $TI_j$  corre un espacio a la izquierda todo el segmento  $\gamma$  (por lo cual en el lugar de  $\sigma$  queda el primer simbolo de  $\gamma$ ). Dejamos al lector la construccion de por ejemplo  $TI_3$  para  $\Sigma = \{\&, \#\}$

Teniendo las maquinas auxiliares antes definidas podemos combinarlas para obtener las maquinas simuladoras de instrucciones. Por ejemplo  $M_{i,k}^a$  puede ser la siguiente maquina

Figura 4

Una posible forma de diseñar la maquina  $IF_{i,k}^a$  es la siguiente

Figura 2

Una posible forma de diseñar la maquina  $M_{i \leftarrow j}^{*,k}$  para el caso  $\Sigma = \{\&, \#\}$  y  $i < j$ , es la siguiente

Nueva Figura 7

En lo que sigue probaremos que el paradigma computacional de Turing es por lo menos tan expresivo como el paradigma imperativo dado por el lenguaje  $\mathcal{S}^\Sigma$ , es decir probaremos que toda función  $\Sigma$ -computable es  $\Sigma$ -Turing computable. Antes un lema

### Lemma

*Si  $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow \Sigma^*$  es  $\Sigma$ -computable, entonces hay un programa  $Q$  el cual computa a  $f$  y el cual cumple con las siguientes propiedades*

- (1) En  $Q$  no hay instrucciones de la forma GOTO  $L\bar{i}$  ni de la forma  $L\bar{j}$  GOTO  $L\bar{i}$*
- (2) Cuando  $Q$  termina partiendo de un estado cualquiera dado, el estado alcanzado es tal que las variables numericas tienen todas el valor 0 y las alfabeticas tienen todas exepcto P1 el valor  $\varepsilon$ .*

## Proof.

Sea  $\mathcal{P}$  un programa que compute a  $f$ . Sea  $r \in \mathbf{N}$  tal que  $r \geq N(\mathcal{P}), n, m$ .  
Sea  $\tilde{\mathcal{P}}$  el resultado de reemplazar en  $\mathcal{P}$  cada instrucción de la forma

$$\alpha \text{GOTO } L\bar{i}$$

con  $\alpha \in \{\varepsilon\} \cup \{L\bar{j} : j \in \mathbf{N}\}$  por  $\alpha \text{IF } N\bar{r} \neq 0 \text{ GOTO } L\bar{i}$ . Ahora sea  $\mathcal{Q}$  el siguiente programa

$$N\bar{r} \leftarrow N\bar{r} + 1$$

$$\tilde{\mathcal{P}}$$

$$N1 \leftarrow 0$$

$$\vdots$$

$$N\bar{r} \leftarrow 0$$

$$P2 \leftarrow \varepsilon$$

$$\vdots$$

$$P\bar{r} \leftarrow \varepsilon$$

Es facil ver que  $\mathcal{Q}$  tiene las propiedades (1) y (2).



Por supuesto, hay un lema analogo para el caso en que  $f$  llega a  $\omega$  en lugar de llegar a  $\Sigma^*$

Ahora si, el anunciado del teorema:

### Theorem

*Si  $f : D_f \subseteq \omega^n \times \Sigma^{*m} \rightarrow O$  es  $\Sigma$ -computable, entonces  $f$  es  $\Sigma$ -Turing computable.*

## Proof.

Supongamos  $O = \Sigma^*$ . Por el Lema ?? existe  $\mathcal{P} \in \text{Pro}^\Sigma$  el cual computa  $f$  y tiene las propiedades (1) y (2). Sea  $k = \max\{n, m, N(\mathcal{P})\}$  y sea  $M_{sim}$  la maquina de Turing con unit que simula a  $\mathcal{P}$  respecto de  $k$ . Como puede observarse, la maquina  $M_{sim}$ , no necesariamente computara a  $f$ . Sea  $M_1$  la maquina siguiente

Figura 9

(Cuando  $n = 0$  debemos interpretar que

$D_0 = (\{q_0, q_f\}, \Gamma, \Sigma, \delta, q_0, B, \vdash, \{q_f\})$ , con  $\delta(q_0, B) = \{(q_f, B, K)\}$  y  $\delta = \emptyset$  en cualquier otro caso).

Notese que  $M_1$  cumple que para cada  $(\vec{x}, \vec{\alpha}) \in \omega^n \times \Sigma^{*m}$ ,

$$\lfloor q_0 B \mid^{x_1} B \dots B \mid^{x_n} B \alpha_1 B \dots B \alpha_m B \rfloor \vdash^* \lfloor q_f B \mid^{x_1} B \dots B \mid^{x_n} B^{k-n} B \alpha_1 B \dots B \alpha_m B \rfloor$$

Notese que en la confeccion de  $M_1$ , para el caso  $m > 0$  podriamos haber usado directamente la  $TD_m$  en lugar de usar  $TD_m$ .

Sea  $M_2$  la siguiente maquina

Figura 10