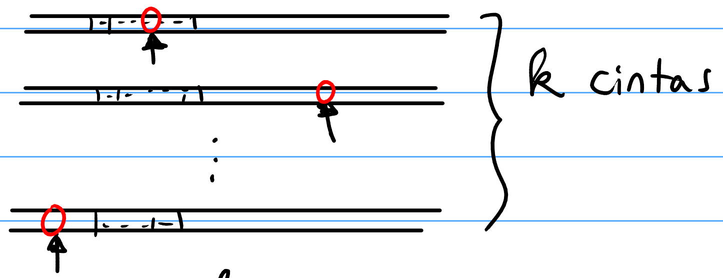
Recordemos

Una M.T. (determinística, de 1 cinta.) es una 7-upla $(\Sigma, \Gamma, Q, \delta, q_0, q_{accept}, q_{reject})$

$$\Sigma \subseteq \Gamma$$
$$\sqcup \in \Gamma$$

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}.$$

Máquinas de Turing multi-cinta



Es una 7-upla

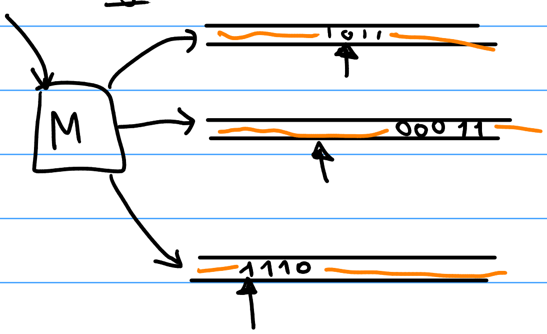
$(\Sigma, \Gamma, Q, \delta, q_0, q_{accept}, q_{reject})$

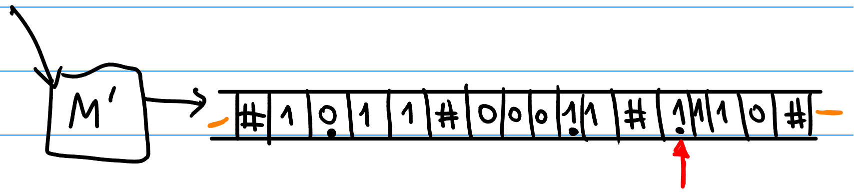
$$\delta : Q \times \Gamma^k \longrightarrow Q \times \Gamma^k \times \{L, R, S\}^k$$

Teorema. Si M es una M.T. multi-cinta que semi-decide un lenguaje $L \subseteq \Sigma^*$, existe una M.T. M' de una sola cinta que semi-decide L.

Dem. (Idea).

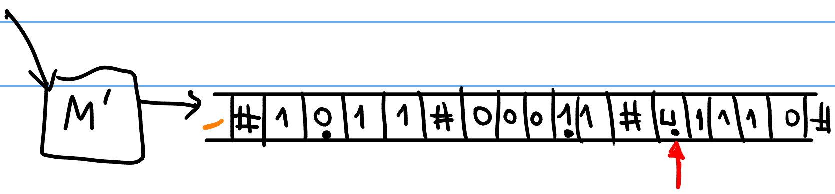
Ej. $k=3$





$\Gamma = \{0, 1, \sqcup\}$

$\Gamma' = \{0, 1, \sqcup, \#, \dot{0}, \dot{1}, \dot{\sqcup}\}$



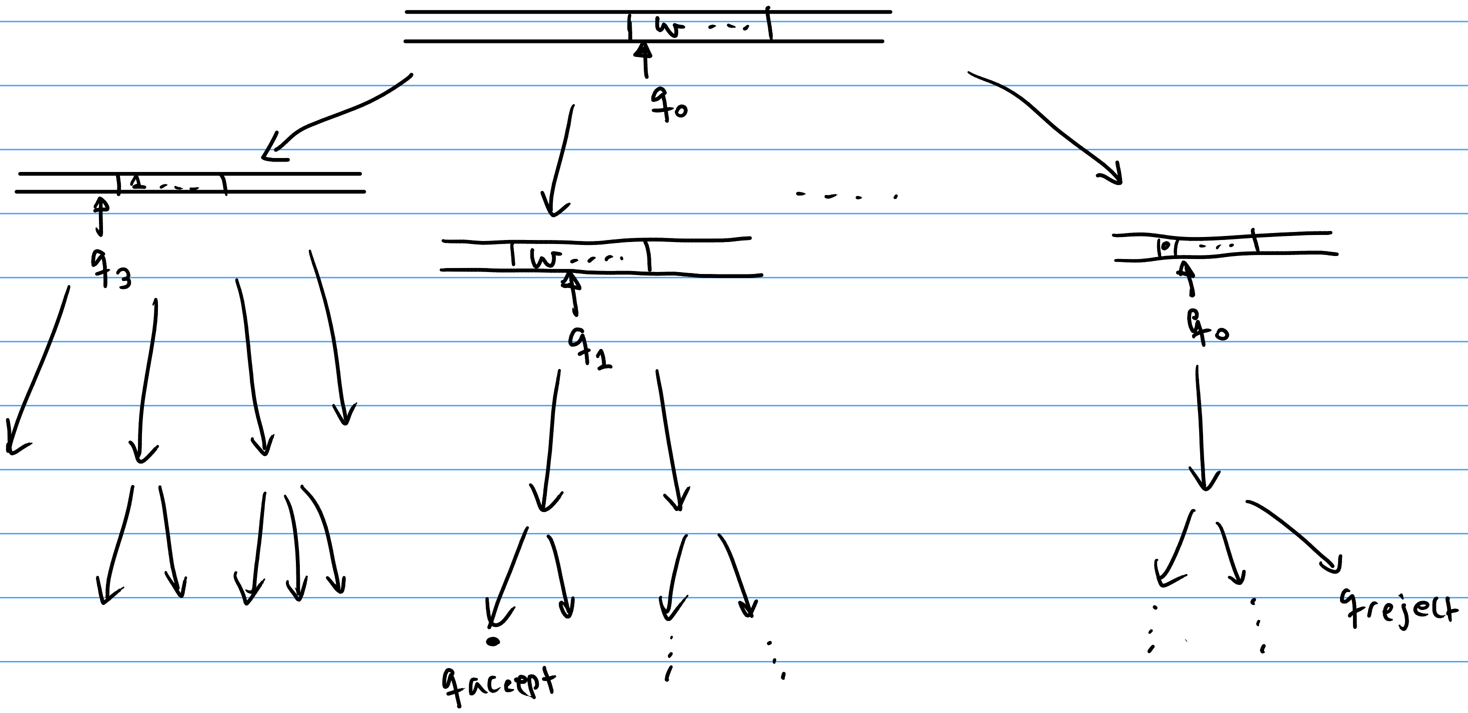
Corolario. Las máquinas de Turing de una cinta tienen el mismo poder expresivo que las máquinas de Turing multi-cinta.

Semi-deciden los mismos lenguajes.

Máquinas de Turing no determinísticas

Def. Una M.T. no determinística es una 7-upla $(\Sigma, \Gamma, Q, \delta, q_0, q_{accept}, q_{reject})$.

$$\delta : Q \times \Gamma \longrightarrow \mathcal{P}(Q \times \Gamma \times \{L, R\})$$

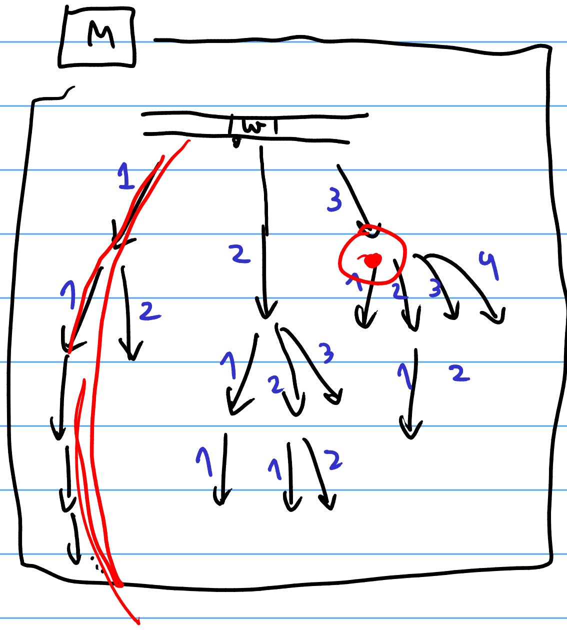


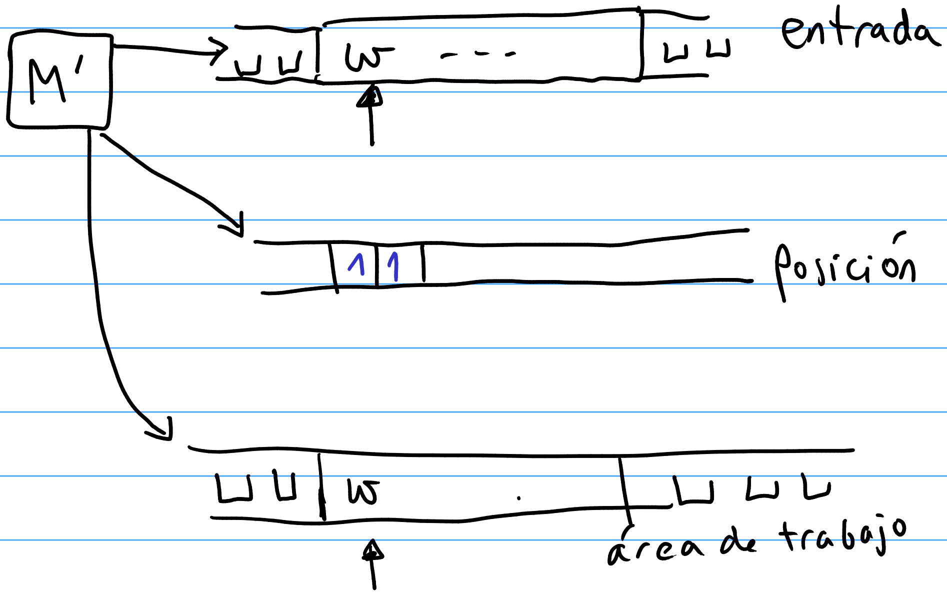
- Decimos que una M.T. no determinística acepta una cadena si existe un camino que llega al estado q_{accept}.

- Decimos que una M.T. no determinística rechaza una cadena si todos los caminos llegan al estado q_{reject}.

Teorema. Si una MT no determinística M semi-decide un lenguaje L, existe una MT determinística M' que semi-decide L.

Dem. (Idea).
Recorrido BFS.





Corolario. Las MT determinísticas tienen el mismo poder expresivo que las MT no determinísticas.

Def. Un enumerador es como una M.T., con una instrucción "print".

$$(\Sigma, \Gamma, Q, \delta, q_0)$$

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R, \underset{\text{print}}{P}\}$$

- El enumerador comienza con la cinta en blanco.

Def. Un lenguaje $L \subseteq \Sigma^*$ es computablemente enumerable si existe un enumerador E que imprime todas las palabras de L, y ninguna otra.

(no importa el orden)

(puede haber repeticiones).

Teorema. Un lenguaje es computablemente enumerable (por algún enumerador E) si y sólo si es semi-decidible. (por alguna M.T. M).

Dem. ($\Longrightarrow$) Sea $L \subseteq \Sigma^*$ un lenguaje computablemente enumerable, es decir, existe un enumerador E que imprime todas y solamente las palabras de L.
Veamos que L es semi-decidible.
Construyamos una M.T. M así:

- Cuando M recibe una palabra $w \in \Sigma^*$ como entrada, M simula el comportamiento del enumerador E.
- Cada vez que el enumerador hace un "print", imprime una palabra $v \in \Sigma^*$.
Comparamos v con w. Si son iguales, llegamos a un estado de aceptación.

si no son iguales, se sigue simulando el comportamiento de E.

$$M(w) = \begin{cases} \text{acepta} & \text{si } w \in L \\ \text{no acepta} & \text{si } w \notin L \end{cases}$$

Por lo tanto M semi-decide el lenguaje L.
Por lo tanto L es semi-decidible.

($\Leftarrow$) Sup. que $L \subseteq \Sigma^*$ es semi-decidible.

Es decir, existe una M.T. M tq. $\qquad \Sigma = \{a, b\}$

$$\forall w \in \Sigma^*, \quad M(w) = \text{acepta} \iff w \in L.$$

¿Cómo podemos enumerar todas las palabras de L?

- Consideremos una enumeración de todas las palabras de Σ^*:
$\{w_1, w_2, w_3, w_4, \ldots\}$.

ε
a
b
aa
ab
ba
bb
aaa
aab
⋮

- Construimos un enumerador E de la siguiente manera:

```
i := 1
loop {
    para cada palabra w ∈ {w_1, ..., w_i} {
        si la máquina M acepta la palabra
        w en a lo sumo i pasos, Print(w).
    }
    i := i+1
}
```

Se verifica:

w_1	w_1	w_2	w_1	w_2	w_3	w_1	w_2	w_3	w_4 ...
(1)	(2)	(2)	(3)	(3)	(3)	(4)	(4)	(4)	(4)

Observemos que:

E imprime una palabra w si y sólo si $M(w) = \text{acepta}$.

Si $M(w) = \text{acepta}$, $w = w_i$ y $M(w_i)$ termina en j pasos.
Entonces en la iteración $\max\{i, j\}$, E imprime w_i.

E imprime todas y solamente las palabras de L.

$\boxed{L \text{ es computablemente enumerable.}}$

Funciones recursivas primitivas

Intuición: Funciones que se pueden calcular usando "for".

```
s := 0
for i in 1..n {
    s := s + i
}
```

Sin usar "while".

Def. Una función $f : \mathbb{N}^k \longrightarrow \mathbb{N}$ es recursiva primitiva si se construye de alguna de las cinco maneras siguientes:

1) La función $\mathrm{cero} : \mathbb{N}^1 \longrightarrow \mathbb{N}$
$$\mathrm{cero}(n) = 0$$
es rec. prim.

2) La función $\mathrm{suc} : \mathbb{N}^1 \longrightarrow \mathbb{N}$
$$\mathrm{suc}(n) = n+1$$
es rec. prim.

3) Las funciones proyectoras $U_i^k : \mathbb{N}^k \longrightarrow \mathbb{N}$
$$U_i^k(x_1, x_2, \dots, x_k) = x_i$$
son rec. prim. $\forall k \ \forall i \in 1..k$.

4) Si $f_1 : \mathbb{N}^k \to \mathbb{N}, \dots, f_n : \mathbb{N}^k \longrightarrow \mathbb{N}$ son rec. prim.

y $g : \mathbb{N}^n \to \mathbb{N}$ es rec. prim.

entonces $h : \mathbb{N}^k \longrightarrow \mathbb{N}$
$$h(x_1, \dots, x_k) = g(f_1(x_1, \dots, x_k), \dots, f_n(x_1, \dots, x_k))$$
es rec. prim.

5) Si $f: \mathbb{N}^k \to \mathbb{N}$ es rec. prim.

$g: \mathbb{N}^{k+2} \to \mathbb{N}$ es rec. prim.

entonces

$$h: \mathbb{N}^{k+1} \to \mathbb{N}$$

$$h(0, x_1, \dots, x_k) = f(x_1, \dots, x_k)$$

$$h(n+1, x_1, \dots, x_k) = g(n, h(n, x_1, \dots, x_k), x_1, \dots, x_k)$$

es rec. prim.

Ej. $\text{Suma}: \mathbb{N}^2 \to \mathbb{N}$

$\text{Suma}(x, y) = x + y$ es rec. prim.

Porque se construye así (por el caso 5):

$$\text{Suma}(0, y) = f(y) = y$$

$$\text{Suma}(n+1, y) = g(n, \text{Suma}(n, y), y) = \text{suma}(n, y) + 1$$

donde f, g son rec. prim.

$f: \mathbb{N} \to \mathbb{N}$ es la identidad

$f(x) = x$ es rec. prim.

Ej. $\text{mult}(x, y) = x * y$ es rec. prim.

$$\text{mult}(0, y) = 0$$

$$\text{mult}(n+1, y) = \text{mult}(n, y) + y$$

Ej. $\text{Pred}: \mathbb{N} \to \mathbb{N}$

$\text{pred}(x) = x \dot{-} 1$ es rec. prim.

$$\begin{bmatrix} 2 \dot{-} 1 = 1 \\ 1 \dot{-} 1 = 0 \\ 0 \dot{-} 1 = 0 \end{bmatrix}$$

$$\text{pred}(0) = 0$$

$$\text{pred}(n+1) = n$$

Ej. $\text{resta}: \mathbb{N}^2 \to \mathbb{N}$

$\text{resta}(x,y) = x \dot{-} y$ es rec. prim.

$\text{resta}(x, 0) = x$

$\text{resta}(x, n+1) = \text{pred}(\text{resta}(x,n))$

· Una función $f: \mathbb{N}^k \to \mathbb{N}$ es un <u>predicado</u> si siempre devuelve 0 ó 1.

· Ej.

$\text{mayorQue}\emptyset : \mathbb{N} \to \mathbb{N}$

$\text{mayorQue}\emptyset(0) = 0$

$\text{mayorQue}\emptyset(n+1) = 1$

$\text{mayor}: \mathbb{N}^2 \to \mathbb{N}$

$\text{mayor}(x,y) = \text{mayorQue}\emptyset(x \dot{-} y)$

$\text{and}: \mathbb{N}^2 \to \mathbb{N}$

$\text{and}(x,y) = x * y$

x	y	
0	0	0
0	1	0
1	0	0
1	1	1

es rec. prim.

$\text{or}: \mathbb{N}^2 \to \mathbb{N}$

$\text{or}(x,y) = 1 \dot{-} ((1 \dot{-} x) * (1 \dot{-} y))$

x	y	$1 \dot{-} x$	$*$	$1 \dot{-} y$	
0	0	1	1	1	0
0	1	1	0	0	1
1	0	0	0	1	1
1	1	0	0	0	1

es rec. prim.

$n = m \qquad n \leq m$

Si $p: \mathbb{N} \to \mathbb{N}$ es un predicado rec. prim

entonces

$$q(n) = \exists_{i=0}^{n-1} p(i) \qquad q \text{ es rec. prim.}$$

$p(0) \vee p(1) \vee \dots \vee p(n-1)$

$q(0) = 0$

$$q(n+1) = \underbrace{q(n)}_{\left(\exists_{i=0}^{n-1} p(i)\right)} \vee p(n)$$

$$\left(\exists_{i=0}^{n-1} p(i)\right) \vee p(n) = \exists_{i=0}^{n} p(i)$$

$$r(n) = \forall_{i=0}^{n-1} p(i)$$ r es rec. prim.

$= p(0) \wedge p(1) \wedge \dots \wedge p(n)$

$$r(0) = 1$$

$$r(n+1) = r(n) \wedge p(n)$$

$$\text{esPrimo} : \mathbb{N}^2 \to \mathbb{N}$$

$$\text{esPrimo}(n) = \neg \exists_{a=0}^{n-1} \exists_{b=0}^{n-1} (a * b = n) \wedge (a > 1) \wedge (b > 1)$$

es rec. prim.

$$\text{neg} : \mathbb{N} \to \mathbb{N}$$

$$\text{neg}(x) = 1 \dot{-} x$$

. Todas las funciones rec. prim. son computables.

Teorema. Hay funciones computables que no son rec. prim.

Dem. Las funciones rec. prim. $f : \mathbb{N} \to \mathbb{N}$ se pueden enumerar.

$$f_1, f_2, f_3, \ldots$$

- Entonces la función

$$g : \mathbb{N} \longrightarrow \mathbb{N}$$
$$g(n) = f_n(n)$$ es computable.

- Afirmo: g no es recursiva primitiva.

Supongamos, por el absurdo que g es rec. prim.

Si g fuera rec. prim., podríamos definir:

$$h : \mathbb{N} \longrightarrow \mathbb{N}$$
$$h(n) = g(n) + 1$$

y h sería rec. prim.

Como h es rec. prim. existe un $i \in \mathbb{N}$ tq. $h = f_i$.

Entonces:

$$h(i) = f_i(i)$$
$$\shortparallel$$
$$g(i)+1 = f_i(i)+1$$

Absurdo.

Un ejemplo particular de función computable pero no rec. prim.

Función de Ackermann.

$$A(0,n) = n+1$$
$$A(m+1,0) = A(m,1)$$
$$A(m+1,n+1) = A(m, A(m+1,n))$$

- $n+1$
- $n+m = n+\underbrace{1+\dots+1}_{m \text{ veces}}$
- $n * m = \underbrace{n+n+\dots+n}_{m \text{ veces}}$
- $n^m = \underbrace{n \cdot n \cdot \dots \cdot n}_{m \text{ veces}}$
- $\vdots$

A es computable pero no es rec. prim.

Lema. Si $f: \mathbb{N}^k \longrightarrow \mathbb{N}$ es una función rec. prim. entonces existe un $M \in \mathbb{N}$ tal que $\forall x_1,\dots,x_k \in \mathbb{N}$

$$f(x_1,\dots,x_k) < A(M, \text{máx}\{x_1,\dots,x_k\})$$

Dem. (omitida).

Teorema. A no es rec. prim.

Dem. Por el absurdo, sup. que A fuera rec. prim. Entonces existe un $M \in \mathbb{N}$ tal que $\forall m,n \in \mathbb{N}$ $A(m,n) < A(M, \text{máx}\{m,n\})$.

En particular tomando $m=n=M$, tendríamos que:

$$\underset{\mathbb{N}}{\overset{\cap}{A(M,M)}} < \underset{A(M,M)}{\overset{\parallel}{A(M, \text{máx}\{M,M\})}}.$$

Absurdo.

Obs. Si $p : \mathbb{N} \to \mathbb{N}$ es un predicado rec. prim.

$q : \mathbb{N} \to \mathbb{N}$

$$q(n) = \underset{i=0..n-1}{\operatorname{argmín}} \; p(i) \qquad \text{es rec. prim.}$$

$$[\underbrace{p(0)}_{0}, \underbrace{p(1)}_{0}, \underbrace{p(2)}_{0}, \ldots, \underbrace{p(i)}_{1}, \ldots \; p(n-1)]$$

$$q(0) = 0$$

$$q(n+1) = \neg\left(\exists_{i=0}^{n-1} p(i)\right) * p(n) * n + q(n)$$

Entonces q es rec. prim.

$$\mu = \underset{i \geq 0}{\operatorname{argmín}} \; p(i)$$

$$\underset{0}{p(0)}, \underset{0}{p(1)}, \underset{0}{p(2)}, \ldots, \underset{1}{p(i)}, \ldots$$

$$\Downarrow$$
$$i$$

Def. Una función $f : \mathbb{N}^k \to \mathbb{N}$

es <u>recursiva general</u>

si se construye usando

las reglas de funciones rec. prim.

(cero, suc, U_i^k, composición, recursión)

ó <u>minimización</u>:

si $p : \mathbb{N}^{k+1} \to \mathbb{N}$ es un predicado rec. general

entonces $q : \mathbb{N}^k \to \mathbb{N}$

$$q(x_1, \ldots, x_k) = \underset{n \geq 0}{\operatorname{argmín}} \; p(n, x_1, \ldots, x_k)$$

es rec. general

Siempre y cuando
$\forall x_1, \ldots, x_k.\ \exists n.\ p(n, x_1, \ldots, x_k) = 1.$