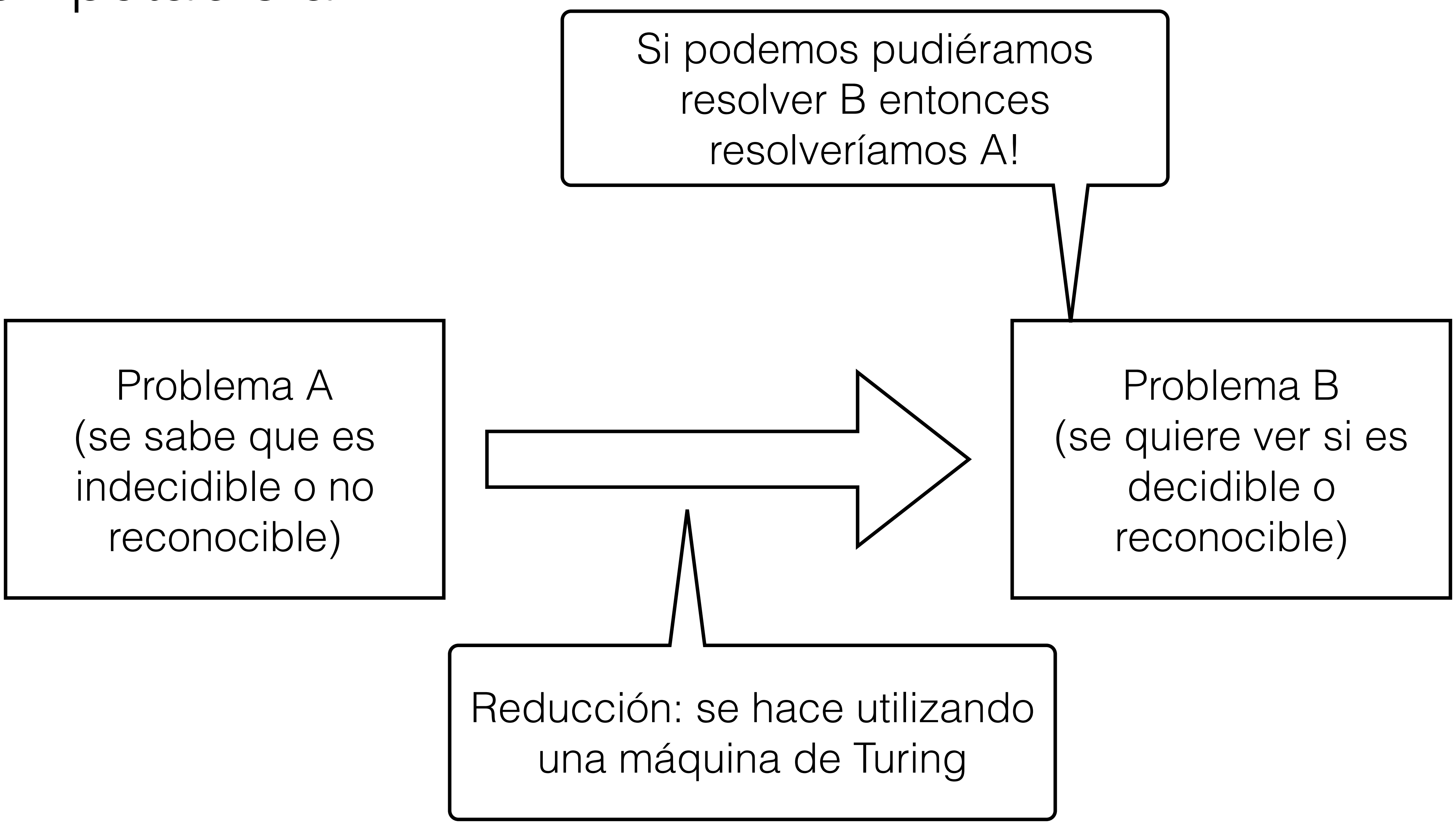
Reducciones

Pablo Castro - UNRC

Reducciones

Para demostrar que un problema A no es computable, lo que generalmente hacemos es reducir un problema no computable a A:



Ejemplo

Recordemos el lenguaje: $A_{TM} = \{\langle M, w\rangle \mid M \text{ acepta } w\}$

Consideremos el lenguaje:

$$H_{TM} = \{\langle M, w\rangle \mid M \text{ termina con la entrada } w\}$$

Veamos que H_{TM} no es decidible.

Supongamos que H_{TM} es decidible por una MT H, entonces definimos M' tal que:

En la entrada $\langle M, w\rangle$, M' actúa de la siguiente forma:

- Ejecuta H con $\langle M, w\rangle$:
  - Si no termina se rechaza,
  - Si termina, se simula M con w.

Esta máquina decide A_{TM} lo cual es una contradicción.

Otro Ejemplo

Consideremos el lenguaje:

$$E_{TM} = \{\langle M \rangle \mid \mathscr{L}(M) = \emptyset\}$$

Para ver que no es decidible podemos reducir A_{TM} a este problema.

Supongamos que E decide E_{TM} entonces diseñamos M' de la siguiente forma:

M' decide A_{TM}

Cuando recibe la entrada $\langle M, w \rangle$:

- Definimos una máquina S que rechaza todas las cadenas diferentes de w, en w corre M.
- Corremos E con la entrada $\langle S, w \rangle$, si acepta acetamos, sino rechazamos.

Más Ejemplos

Consideremos el siguiente lenguaje:

$$Reg = \{M \mid M \text{ acepta un lenguaje regular}\}$$

Supongamos que R decide Reg, podemos reducir A_{TM} a este problema.

Construimos una MT M de la siguiente forma:

Dada la entrada $\langle M, w \rangle$, se construye una MT M':

- M' acepta el lenguaje $\{0^n 1^n \mid n \geq 0\}$
- Para $x \notin \{0^n 1^n \mid n \geq 0\}$, M' corre M

Corremos R con M' si acepta, aceptamos, sino rechazamos

Notar que M' reconoce Σ^* cuando M acepta w. Sino reconoce un lenguaje no regular.

Equivalencia de MTs

Veamos que el problema de ver si dos MTs reconocen el mismo lenguaje es indecidible

$$Eq_{TM} = \{\langle M, M' \rangle \mid \mathscr{L}(M) = \mathscr{L}(M')\}$$

Podemos reducir E_{TM} a este problema. Asumamos que hay una MT R que decide Eq_{TM} entonces, construimos E que decide E_{TM}:

En la entrada $\langle M \rangle$, corremos Eq_{TM} con $\langle M, M' \rangle$

MT que rechaza todas las palabras.

- Si R acepta, entonces aceptamos,
- Sino se rechaza

Teorema de Rice

El teorema de Rice es importante porque dice que cualquier propiedad no trivial sobre MTs es indecidible:

Un conjunto $P \subseteq \{\langle M \rangle \mid M \text{ es una máquina de Turing}\}$ se dice **propiedad** si:

Si $\mathscr{L}(M) = \mathscr{L}(M')$ entonces: $M \in P$ ssi $M' \in P$

Se dice no-trivial si:

1. $P \neq \varnothing$
2. $P \neq \{\langle M \rangle \mid M \text{ es una máquina de Turing }\}$

Teorema de Rice

Teorema: Cualquier propiedad no trivial sobre máquinas de Turing es indecidible.

Prueba:

Sea P una propiedad no trivial sobre MTs. Asumamos M_P que decida P. Sea $T \in P$, definimos la MT S de la siguiente forma:

En la entrada $\langle M, w \rangle$, utilizaremos la MT M_w:

- M_w con input x, simula M con input w :
  - Si M termina y rechaza, M_w termina y rechaza
  - Si M acepta, entonces M_w simula T con x. Si esta acepta, entonces acepta.
- Usamos M_P para saber si $\langle M_w \rangle \in P$

Soluciona A_{TM}

Notar que M_w simula T cuando M acepta w

Historia de Ejecución

Dada una MT un historia de ejecución es una secuencia:

$$C_0, C_1, \ldots, C_k$$

Conf. inicial

mov. legales

Conf. final

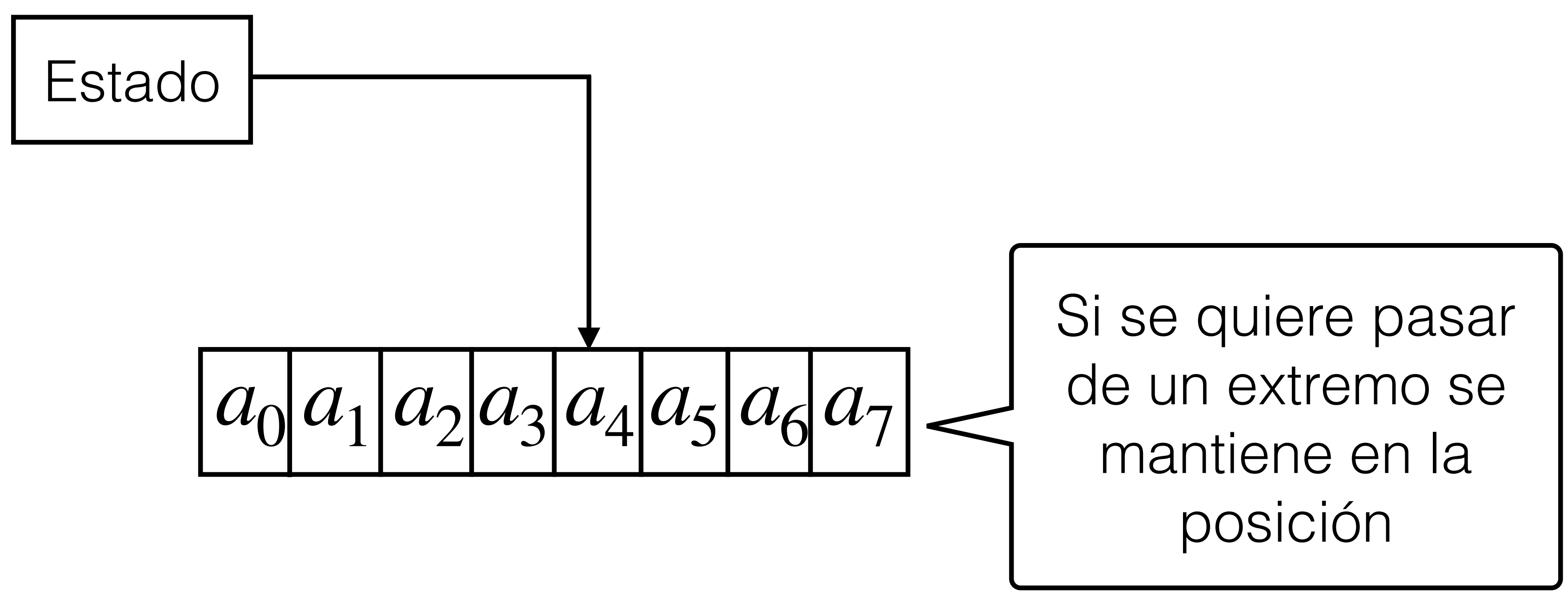
Las máquinas de Turing deterministicas tienen a lo sumo una ejecución para cada entrada

Las ejecuciones que aceptan son las que terminan en q_A

Las ejecuciones que rechazan son las que terminan en q_R

Autómatas Linearmente Acotados

Los autómatas linearmente acotados (LBAs) son MTs, que solo pueden usar la cadena de entrada de memoria:



La posible cantidad de configuraciones es:

$$\#q * n * \#\Gamma^n$$

LBAs

Muchos problemas para LBAs se vuelven decidibles:

$$A_{LBA} = \{\langle A, w\rangle \mid \text{LBA } A \text{ acepta } w\}$$

Problema de aceptación en LBA es decidible

Para decidir este lenguaje tenemos que detectar loops:

$$C_0 \rightarrow C_1 \rightarrow C_2 \rightarrow \cdots \rightarrow C_k \rightarrow$$

Si se hacen más de $\#q * n * \#\Gamma^n$ pasos entramos en un ciclo

En la entrada $\langle A, w\rangle$:

Se simula A en w por $\#q * n * \#\Gamma^n$ pasos

Si se acepta, se acepta

Si se rechaza, se rechaza

si se hacen más pasos, se rechaza

Problemas Indecidibles sobre LBAs

También podemos encontrar problemas indecidibles sobre LBAs:

$$E_{LBA} = \{\langle A \rangle \mid \mathscr{L}(A) = \varnothing\}$$

Este problema es indecidible. Podemos reducir A_{TM} a E_{LBA}

Dada $\langle M, w \rangle$ podemos considerar la LBA E que chequea todas las posibles configuraciones de M empezando con w

Es decir, si E empieza con $C_0 C_1 \ldots C_k$ y $C_0 = q_0 w$ acepta si esta es la ejecución de M con w, sino rechaza.

Luego usamos A que resuelve E_{LBA} para chequear si este lenguaje es vacío o no.

El Problema de Correspondencia de Post

Dados dominos del estilo:

$$\frac{b}{ca} \quad \frac{a}{ab} \quad \frac{ca}{a} \quad \frac{abc}{c}$$

Se tiene que encontrar un matching (con posibles repeticiones)

Por ejemplo:

$$\frac{a}{ab} \quad \frac{b}{ca} \quad \frac{ca}{a} \quad \frac{a}{ab} \quad \frac{abc}{c}$$

Es una posible solución.

El Problema de Post

Este problema es indecidible. Idea:

Introducir dominos para simular las ejecuciones de M con la entrada w

$$\frac{q_i a}{b q_j} \qquad \text{Si } \delta(a, q_i) = (b, q_j, R)$$

$$\frac{a q_i b}{q_j b c} \qquad \text{Si } \delta(b, q_i) = (c, q_j, L)$$

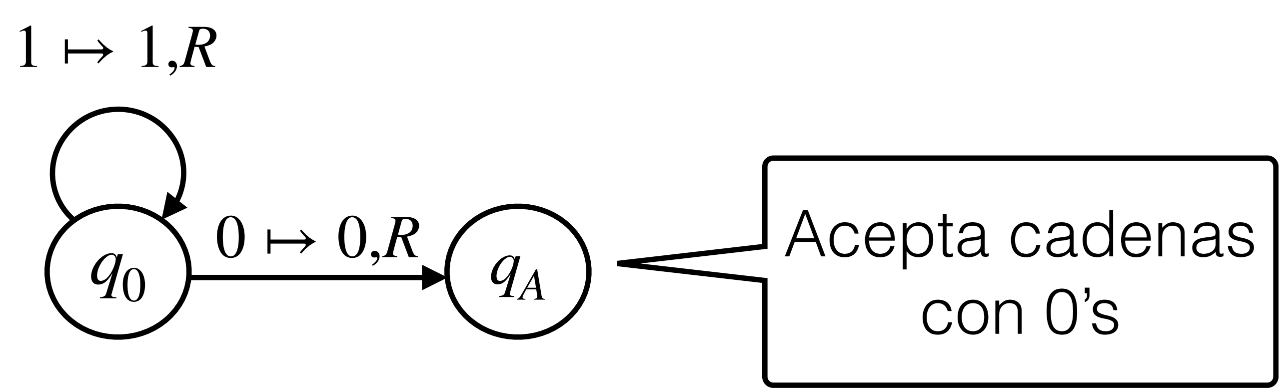
Permiten codificar δ

El Problema de Post

Se agregan dominos adicionales:

$$\frac{a}{a} \quad \text{para cualquier símbolo}$$

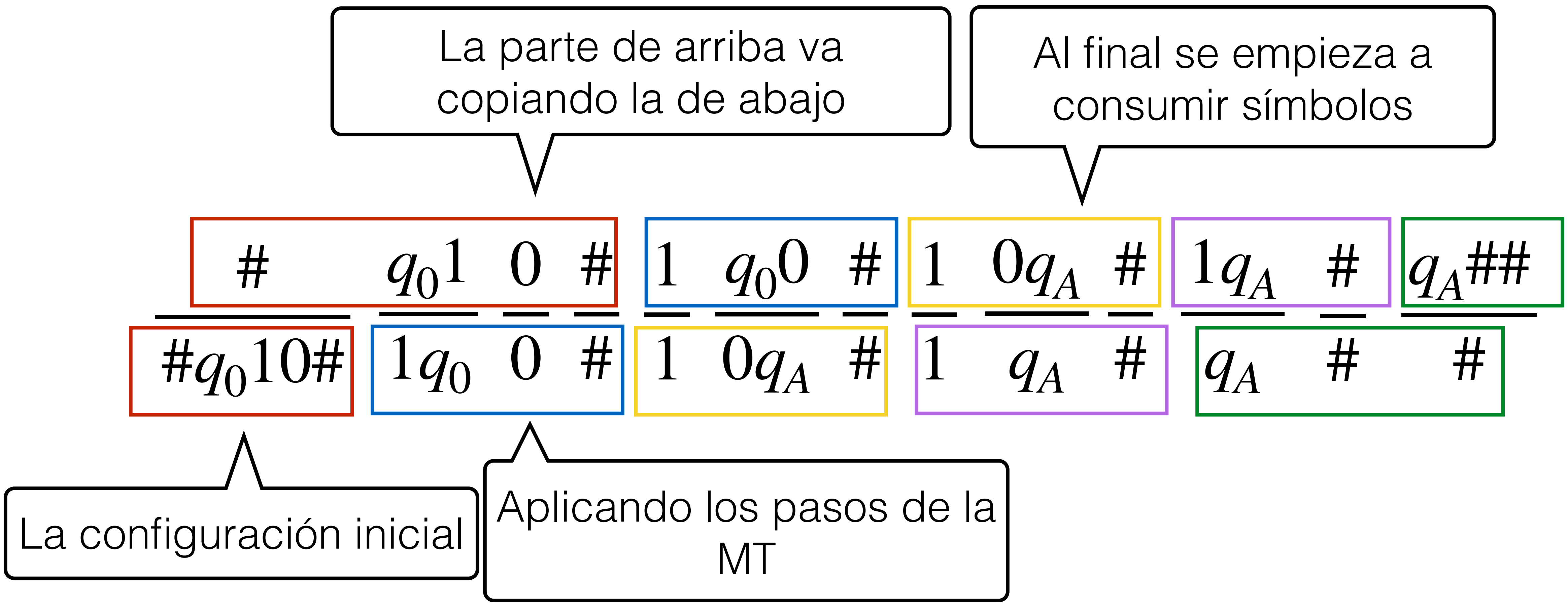
Veamos un ejemplo:



$$\frac{q_0 1}{1 q_0} \quad \frac{q_0 0}{0 q_A} \quad \frac{0}{0} \quad \frac{1}{1} \quad \frac{\#}{\#} \quad \frac{q_A \#\#}{\#} \quad \frac{0 q_A}{q_A} \quad \frac{1 q_A}{q_A} \quad \frac{q_A 0}{q_A} \quad \frac{q_A 1}{q_A}$$

Juguemos el Juego

Una ejecución de M acepta w ssi hay una matching



Si Post fuera decidible, también A_{TM} sería decidible.

La demostración consiste en codificar las ejecuciones de M

Funciones Computables

Para caracterizar la noción de función computable:

Una función $f : \Sigma^* \to \Sigma^*$ se dice computable si existe una máquina de Turing M tal que empezando con w termina con $f(w)$ en la cinta.

Ejemplo: Las funciones aritméticas son computables, por ejemplo podemos diseñar una máquina de Turing que con entrada $\langle m, n \rangle$ devuelva $m + n$.

Ejemplo: Podemos considerar los procedimientos que cambian máquinas de Turing, como funciones que transforman textos en textos.

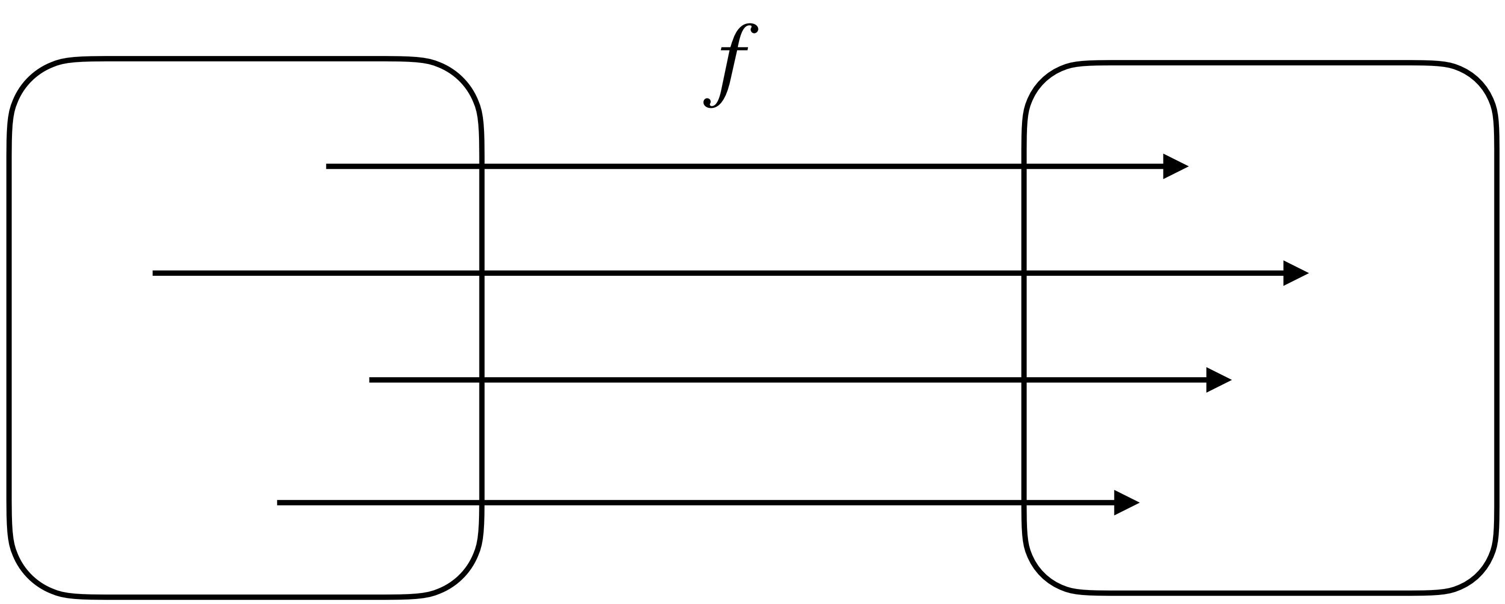
La Noción de Reducibilidad

Definamos cuando un lenguaje es reducible a otro:

El lenguaje A se dice reducible al lenguaje B, $A \leq B$ si hay una **función computable** tal que para todo w:

$$w \in A \Leftrightarrow f(w) \in B$$

La función f es llamada una reducción de A a B



Reducibilidad

Tenemos la siguiente propiedad:

Si $A \leq B$ y B es decidible, entonces A es decidible

Prueba:

Si B es decidible, tenemos la máquina M_B que lo decide
Podemos decidir A de la siguiente forma:

En la entrada w:

- computar $f(w)$
- correr M_B en la entrada $f(w)$ y devolver lo mismo

Reducidibilidad

La misma clase de propiedad se cumple para reconocedores:

Propiedad: Si $A \leq B$ y B es reconocible, entonces A es reconocible.

Si B es reconocible, tenemos la máquina M_B que lo reconoce. Podemos reconocer A de la siguiente forma:

En la entrada w:

- computar $f(w)$
- correr M_B en la entrada $f(w)$ y devolver lo mismo

Ejemplo: Halt

Veamos la reducción de A_{TM} a $Halt$:

Tenemos que describir una función $f : \Sigma^* \rightarrow \Sigma^*$ computable tal que:

$$\langle M, w \rangle \in A_{TM} \qquad \text{ssi} \qquad f(\langle M, w \rangle) \in Halt$$

La función computable f en la entrada $\langle M, w \rangle$ hace lo siguiente:

Con la entrada $\langle M, w \rangle$ f construye la MT M':

M' corre M con w:
- Si M acepta, M' acepta,
- Si M rechaza, M' hace un loop y no termina

Se cumple la condición para la reducción

Ejercicios:

1. Encontrar un matching para el siguiente problema de Post.

$$\frac{ab}{abab}, \frac{b}{a}, \frac{aba}{b}, \frac{aa}{a}$$

2.Si $A \leq B$ y B es un lenguaje regular, esto implica que A es y un lenguaje regular? Fundamentar.

3. Demostrar que $\leq$ es transitiva.

4.Demostrar que si A es reconocible y $A \leq \overline{A}$ entonces A es decidible.

5.Demostrar que A es reconocible sii $A \leq A_{TM}$

6.Usar el teorema de Rice, para demostrar la indecibilidad del siguiente lenguaje:

$$All = \{\langle M \rangle \mid \mathscr{L}(M) = \Sigma^*\}$$