Complejidad Computacional

Santiago Figueira

Departamento de Computación - FCEN - UBA

clase 1

Clase 1

- Preliminares
- Representaciones en binario
- Problemas de decisión
- Máquinas de Turing
- Configuraciones y cómputos

Organización

- Docentes
  - Santiago Figueira (Prof.)
  - Santiago Cifuentes (JTP)
  - Ayelén Dinkel (AY1)
  - Joaquín Laks (AY2)
  - Belén Loleo Saigos (AY2)
  - Martín Santesteban (AY2)
  - Tomás Spognardi (AY2)
- Miércoles de 17 a 22. Aula 1309 Pab. $0 + \infty$.
- Parciales: 14/5 y 25/6; Recus: 2/7 y 16/7
- Bibliografía:
  - Sanjeev Arora y Boaz Barak. Computational complexity: a modern approach, Cambridge University Press, 2009, `https://theory.cs.princeton.edu/complexity/book.pdf`
  - apunte (en el campus)
  - estas slides

Preliminares

Notación

Dado un alfabeto Γ:

Notación: Conjunto de palabras sobre un alfabeto

Γ^* es el conjunto con todas las palabras formadas sobre Γ

Ejemplo

Si $\Gamma = \{0, 1\}$, entonces $\Gamma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, \dots\}$

Notación: Palabra vacía

ϵ representa la palabra vacía.

Notación: Longitud de una palabra

Si $\sigma \in \Gamma^*$ entonces $|\sigma|$ es la longitud de σ, es decir la cantidad de letras de σ.

Ejemplo

Si $\Gamma = \{0, 1\}$, $\sigma = 101$, $\tau = \epsilon$ entonces $|\sigma| = 3$, $|\tau| = 0$.

Notación

Notación: Conjunto de palabras de largo k

Γ^k es el conjunto con todas las palabras formadas sobre Γ de longitud k. Equivalentemente, a veces las vamos a tomar como tuplas de dimensión k.

Ejemplo

Si $\Gamma = \{0, 1\}$, entonces $\Gamma^2 = \{00, 01, 10, 11\}$.
También puede ser $\Gamma^2 = \{(0, 0), (0, 1), (1, 0), (1, 1)\}$.
Va a quedar claro del contexto cuál usamos.

Notación

Notación: Concatenación de palabras

Si $\sigma, \tau \in \Gamma^*$, $\sigma\tau$ representa la concatenación de σ con τ.

Notación: i-ésimo elemento de una palabra

Si $\sigma \in \Gamma^*$, $i \in \{0, \ldots, |\sigma| - 1\}$, entonces $\sigma(i)$ es la $(i+1)$-ésima letra de σ.

Ejemplo

Si $\Gamma = \{0, 1\}$, y $\sigma = 101$ entonces $\sigma(0) = 1$, $\sigma(1) = 0$, $\sigma(2) = 1$.

Representaciones en binario

Representaciones de números naturales

Buscamos

- fijar un alfabeto; elegimos el binario: $\{0, 1\}$
- representar objetos (números, grafos, etc.) en binario
- trabajar con métodos efectivos que transforman unos objetos en otros

Definición

Si $n \in \mathbb{N}$, notamos $[\boldsymbol{n}]$ a la representación de n en binario y notamos $|\boldsymbol{n}|$ al tamaño de $[n]$, es decir $|n| = \lceil \log n \rceil$.

Representaciones de números naturales

Buscamos

- fijar un alfabeto; elegimos el binario: $\{0, 1\}$
- representar objetos (números, grafos, etc.) en binario
- trabajar con métodos efectivos que transforman unos objetos en otros

Definición

Si $n \in \mathbb{N}$, notamos $[\boldsymbol{n}]$ a la representación de n en binario y notamos $|\boldsymbol{n}|$ al tamaño de $[n]$, es decir $|n| = \lceil \log n \rceil$.

Ejemplos

- $[0] = 0$; $|0| = 1$
- $[5] = 101$; $|5| = 3$

Representaciones autodelimitantes

Definición (codificación autodelimitante de cadenas)

Para $\sigma = b_1 b_2 \ldots b_n$ ($b_i \in \{0, 1\}$), definimos

$$\langle \sigma \rangle = b_1 b_1 \; b_2 b_2 \; \ldots \; b_n b_n \; 10$$

$\langle \sigma \rangle$ es autodelimitante (el 10 marca el final).

Ejemplos

- $\langle 101 \rangle = 11\ 00\ 11\ 10$
- $\langle \epsilon \rangle = 10$

(los espacios no forman parte de la codificación)

Representaciones de listas, tuplas, conjuntos

Definición

Codificamos la lista $L = \sigma_1, \ldots, \sigma_n$ ($\sigma_i \in \{0,1\}^*$) como

$$\langle L \rangle = \langle \sigma_1 \rangle \; \langle \sigma_2 \rangle \; \ldots \; \langle \sigma_n \rangle \; 01$$

(01 marca fin de lista). Codificamos tuplas y conjuntos como listas.

Ejemplos

- $\langle 11, 100, 0001 \rangle = \langle (11, 100, 0001) \rangle = \langle \{11, 100, 0001\} \rangle =$ 111110 11000010 0000001110 01
- $\langle \emptyset \rangle = 01$

(los espacios no forman parte de la codificación)

Representaciones de otros objetos

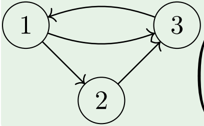
Usaremos misma notación $\langle X \rangle$ para otros objetos matemáticos X a veces sin especificar formalmente su representación.

Ejemplo

$$
\begin{aligned}
A &= \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1k} \\ b_{21} & b_{22} & \dots & b_{2k} \\ & & \dots & \\ b_{\ell 1} & b_{\ell 2} & \dots & b_{\ell k} \end{pmatrix} \qquad (b_{ij} \in \{0,1\}^*) \\
\langle A \rangle &= \langle \langle b_{11}, \dots, b_{1k} \rangle \langle b_{21}, \dots, b_{2k} \rangle \dots \langle b_{\ell 1}, \dots, b_{\ell k} \rangle \rangle \\
|\langle A \rangle| &= 2 \cdot \ell \cdot (k+1) + 2
\end{aligned}
$$

Ejemplo

Si $G = (E, V)$ es un digrafo, podemos representar G con la matriz de adyacencia



$$
\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}
\qquad
\begin{aligned}
\langle G \rangle &= 00111110000011101100001001 \\
|\langle G \rangle| &= 26
\end{aligned}
$$

Representaciones de otros objetos

Notación: codificación de tuplas de objetos

Siempre supondremos que cada objeto O (lista, grafo, número, etc.) tiene una codificación razonable $\langle O \rangle$ en binario.

Muchas veces, vamos a notar la codificación de tuplas $(O_1, \ldots, O_n)$ de objetos simplemente como

$$\langle O_1, \ldots, O_n \rangle$$

en vez de

$$\langle \langle O_1 \rangle, \ldots, \langle O_n \rangle \rangle$$

Del mismo modo, usaremos $|O|$ en vez de $|\langle O \rangle|$ para referirnos al tamaño de la codificación de O.

Todo lo que dijimos hasta acá sobre notación binaria, vale para otros alfabetos finitos Γ.

- a veces vamos a usar alfabetos con más de dos símbolos
- a veces vamos usaremos notación unaria

Problemas de decisión

Problemas de decisión

Nos restringimos a **problemas de decisión**, que se formalizan como funciones booleanas:

$$f : \{0,1\}^* \to \{0,1\}$$

Cada función booleana f representa el lenguaje (conjunto de palabras en $\{0,1\}^*$)

$$\mathcal{L}(f) = \{x \colon f(x) = 1\} \subseteq \{0,1\}^*$$

Y viceversa: cualquier lenguaje $\mathcal{L} \subseteq \{0,1\}^*$ se puede representar por la función booleana

$$\chi_{\mathcal{L}}(x) = \begin{cases} 1 & \text{si } x \in \mathcal{L} \\ 0 & \text{si no} \end{cases}$$

problemas de decisión = funciones booleanas = lenguajes

Ejemplos de problemas de decisión

Ejemplo

El problema de decidir si un número x es par se formaliza con

$$f : \{0,1\}^* \rightarrow \{0,1\}$$
$$f([x]) = \begin{cases} 1 & \text{si } x \text{ es par} \\ 0 & \text{si no} \end{cases}$$

o bien con $\mathcal{L} = \{[x] : x \text{ es par}\}$

Ejemplo

El problema de decidir si un grafo G tiene un k-coloreo

$$f : \{0,1\}^* \rightarrow \{0,1\}$$

$$f(\langle\langle G\rangle, [k]\rangle) = \begin{cases} 1 & \text{si } G \text{ tiene un } k\text{-coloreo} \\ 0 & \text{si no} \end{cases}$$

o bien con

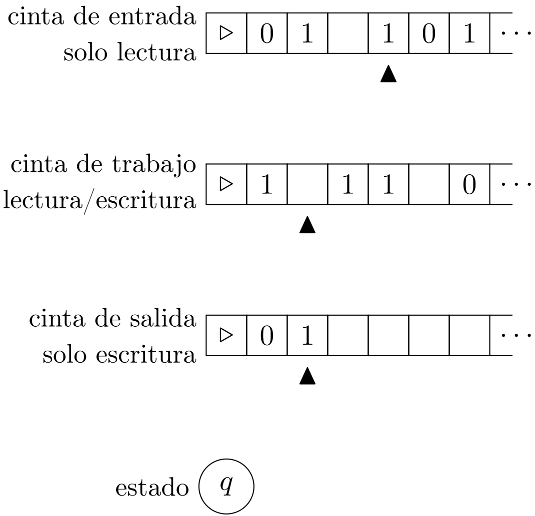
$$\mathcal{L} = \{\langle\langle G\rangle, [k]\rangle : G \text{ tiene un } k\text{-coloreo}\}$$

Notación: $\mathcal{L}$

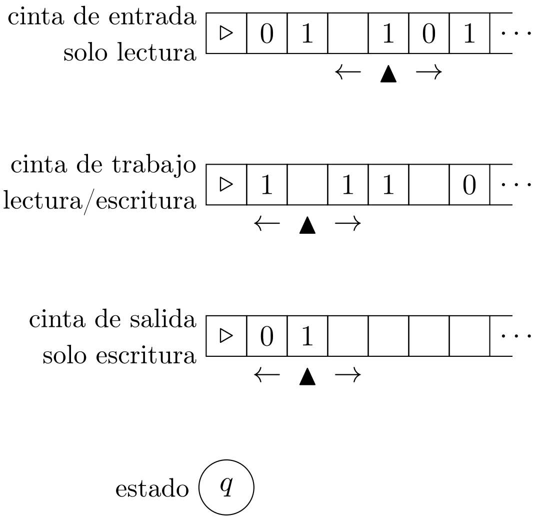
A partir de ahora $\mathcal{L}$ va a ser un lenguaje en $\{0,1\}^*$.

Máquinas de Turing

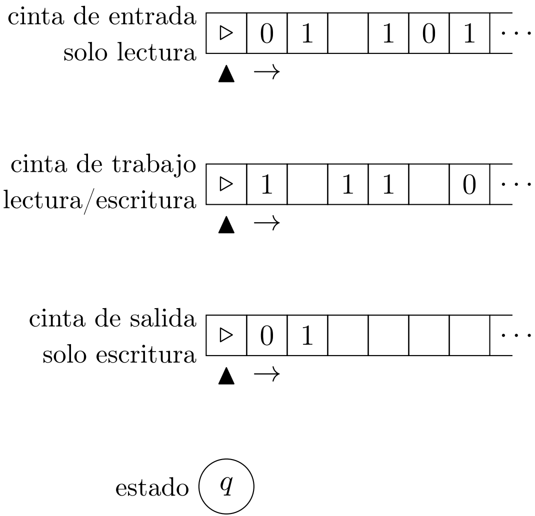
Máquina de Turing con 3 cintas



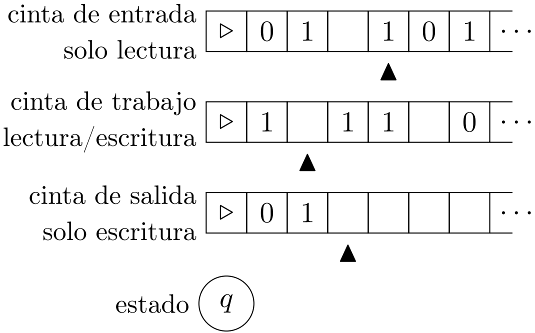
Máquina de Turing con 3 cintas



Máquina de Turing con 3 cintas



Máquina de Turing con 3 cintas



Si

  estado $== q$

  cinta de entrada $== 1$

  cinta de trabajo $== 1$

entonces

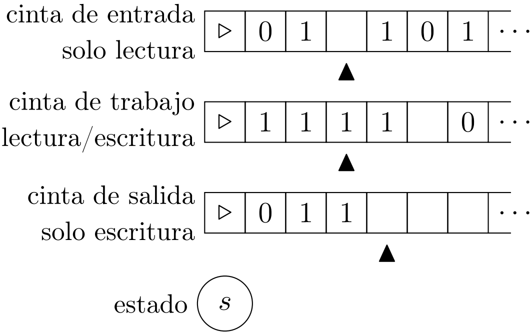
  en cinta de entrada: mover cabeza a L

  en cinta de trabajo: escribir 1 *y mover cabeza a* R

  en cinta de salida: escribir 1 *(y mover a* R*)*

  estado $= s$

Máquina de Turing con 3 cintas



Si

  estado $== q$

  cinta de entrada $== 1$

  cinta de trabajo $== 1$

entonces

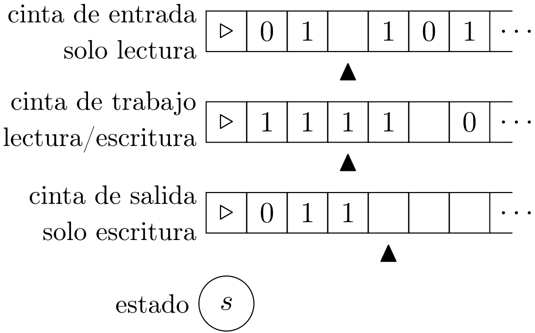
  en cinta de entrada: mover cabeza a L

  en cinta de trabajo: escribir 1 *y mover cabeza a* R

  en cinta de salida: escribir 1 *(y mover a* R*)*

  estado $= s$

Máquina de Turing con 3 cintas



Si

estado $==$ *s*

cinta de entrada $== \square$

cinta de trabajo $== 1$

entonces

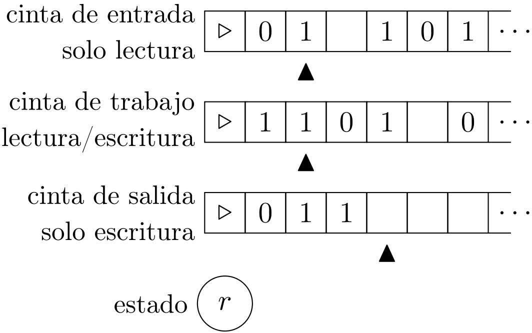
en cinta de entrada: mover cabeza a L

en cinta de trabajo: escribir 0 *y mover cabeza a L*

en cinta de salida: no hacer nada)

estado $= r$

Máquina de Turing con 3 cintas



Si

estado $== s$

cinta de entrada $== \square$

cinta de trabajo $== 1$

entonces

en cinta de entrada: mover cabeza a L

en cinta de trabajo: escribir 0 *y mover cabeza a* L

en cinta de salida: no hacer nada)

estado $= r$

Máquina de Turing con 3 cintas

Definición

Una **máquina de Turing con 3 cintas** (o simplemente **máquina con 3 cintas**) es una tripla (Σ, Q, δ), donde

- Q: es el conjunto finito de **estados**
  - $q_0 \in Q$: estado inicial
  - $q_f \in Q$: estado final
- Σ: es el **alfabeto** (conjunto finito de símbolos)
  - $\triangleright \in \Sigma$: comienzo de cinta
  - $\square \in \Sigma$: blanco

  $\Sigma = \{0, 1, \triangleright, \square\}$ es el alfabeto **estándar**.
- $\delta : Q \times \Sigma \times \Sigma \to \{L, R.S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$ es la **función de transición**
  - L: izquierda (*Left*)
  - R: derecha (*Right*)
  - S: quedarse en el lugar (*Stay*)

La función de transición

INSTRUCCIÓN

si   *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

en cinta de entrada: *mover cabeza a* $L/R/S$
en cinta de trabajo: *escribir* $c \in \Sigma$ *y mover cabeza a* $L/R/S$
en cinta de salida: *escribir* $d \in \Sigma$ *(y mover a* R*) / no hacer nada*
estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

en cinta de entrada: *mover cabeza a* $L/R/S$
en cinta de trabajo: *escribir* $c \in \Sigma$ *y mover cabeza a* $L/R/S$
en cinta de salida: *escribir* $d \in \Sigma$ *(y mover a* R*)* / *no hacer nada*
estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si   *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

en cinta de entrada:	*mover cabeza a* $L/R/S$
en cinta de trabajo:	*escribir* $c \in \Sigma$ *y* *mover cabeza a* $L/R/S$
en cinta de salida:	*escribir* $d \in \Sigma$ *(y mover a* R*)* / *no hacer nada*

estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \rightarrow \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

en cinta de entrada: *mover cabeza a* $L/R/S$
en cinta de trabajo: *escribir* $c \in \Sigma$ *y* *mover cabeza a* $L/R/S$
en cinta de salida: *escribir* $d \in \Sigma$ *(y mover a R) / no hacer nada*
estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si   *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

en cinta de entrada:	*mover cabeza a* $L/R/S$
en cinta de trabajo:	*escribir* $c \in \Sigma$ *y* *mover cabeza a* $L/R/S$
en cinta de salida:	*escribir* $d \in \Sigma$ *(y mover a* R*)* / *no hacer nada*

estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si   *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

en cinta de entrada: *mover cabeza a* $L/R/S$
en cinta de trabajo: *escribir* $c \in \Sigma$ *y*
mover cabeza a $L/R/S$
en cinta de salida: *escribir* $d \in \Sigma$ *(y mover a* R*)* / *no hacer nada*
estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si   *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

en cinta de entrada:	*mover cabeza a* $L/R/S$
en cinta de trabajo:	*escribir* $c \in \Sigma$ *y* *mover cabeza a* $L/R/S$
en cinta de salida:	*escribir* $d \in \Sigma$ *(y mover a* R*)* / *no hacer nada*

estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si $\textit{estado} == q \in Q$

$\textit{cinta de entrada} == a \in \Sigma$

$\textit{cinta de trabajo} == b \in \Sigma$

entonces

en cinta de entrada: *mover cabeza a* $L/R/S$

en cinta de trabajo: *escribir* $c \in \Sigma$ *y*
mover cabeza a $L/R/S$

en cinta de salida: *escribir* $d \in \Sigma$ *(y mover a* R*)* / *no hacer nada*

$\textit{estado} = r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

La función de transición

INSTRUCCIÓN

si   *estado* $== q \in Q$
cinta de entrada $== a \in \Sigma$
cinta de trabajo $== b \in \Sigma$

entonces

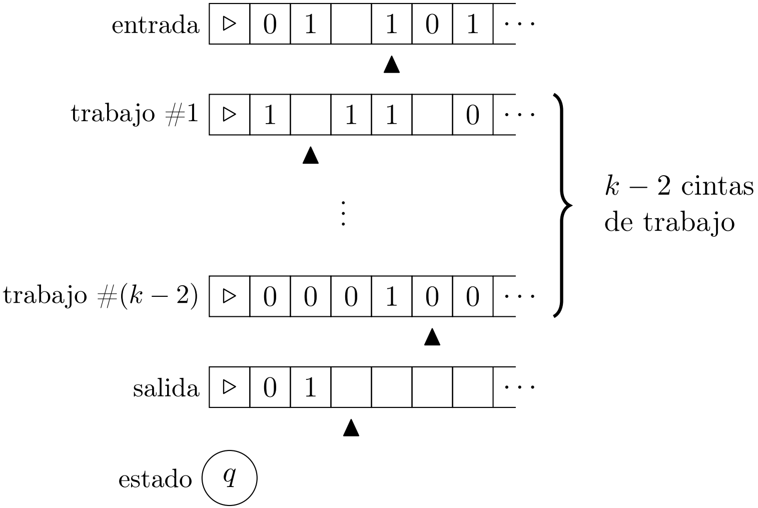
en cinta de entrada: *mover cabeza a* $L/R/S$
en cinta de trabajo: *escribir* $c \in \Sigma$ *y mover cabeza a* $L/R/S$
en cinta de salida: *escribir* $d \in \Sigma$ *(y mover a* R*) / no hacer nada*
estado $= r \in Q$

$$\delta : Q \times \Sigma \times \Sigma \to \{L, R, S\} \times \Sigma \times \{L, R, S\} \times (\Sigma \cup \{S\}) \times Q$$

Hay restricciones sobre δ:

- $\delta(*, \triangleright, *)$ no puede ser de la forma $(L, *, *, *, *)$
- $\delta(*, *, \triangleright)$ no puede ser de la forma $(*, *, L, *, *)$
- $\delta(q_f, a, b)$ solo puede ser de la forma (S, b, S, S, q_f)
  - el estado q_f termina el movimiento

Máquina de Turing con k cintas



Máquina de Turing (general)

Definición

Una **máquina de Turing** (o simplemente **máquina**) es una tripla (Σ, Q, δ), donde

- Q: es el conjunto finito de **estados**
  - $q_0 \in Q$: estado inicial
  - $q_f \in Q$: estado final
- Σ: es el **alfabeto** (conjunto finito de símbolos)
  - $\triangleright \in \Sigma$: comienzo de cinta
  - $\square \in \Sigma$: blanco

  Si no aclaramos, el alfabeto es el **estándar** $\Sigma = \{0, 1, \triangleright, \square\}$
- La **función de transición** es

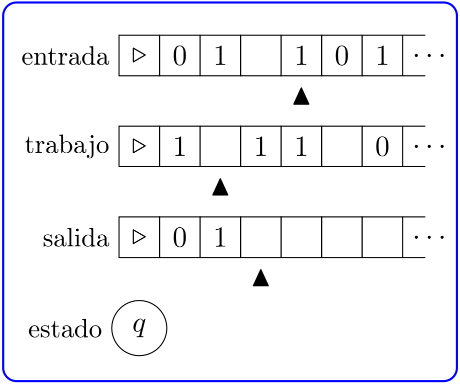
$$\delta : Q \times \Sigma^{k-1} \to \underbrace{\{L, R, S\}}_{\text{entrada}} \times \underbrace{\Sigma^{k-2} \times \{L, R, S\}^{k-2}}_{k-2 \text{ trabajos}} \times \underbrace{(\Sigma \cup \{S\})}_{\text{salida}} \times Q$$

donde $k \geq 3$.

Configuraciones y cómputos

Configuración

Una **configuración** es la fotografía de la máquina en un determinado momento, por ejemplo



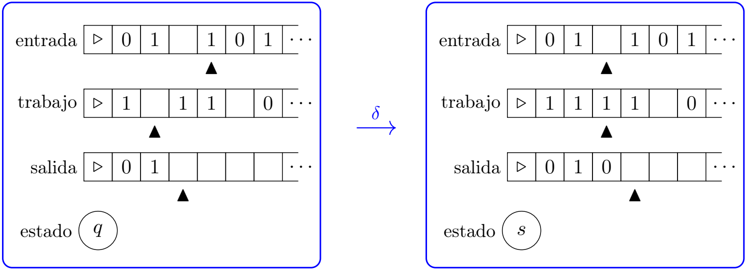
Tiene toda la información sobre:

- contenido de todas las cintas
- posición de cada cabeza
- estado

Alcanza con considerar cintas que solo tienen finitos 0s y 1s (es decir, a partir de una posición i, todas las cintas tienen blancos a la derecha de i).

Evolución de un cómputo

La función de transición δ hace **evolucionar** el cómputo **en un paso** a partir de una dada configuración C:

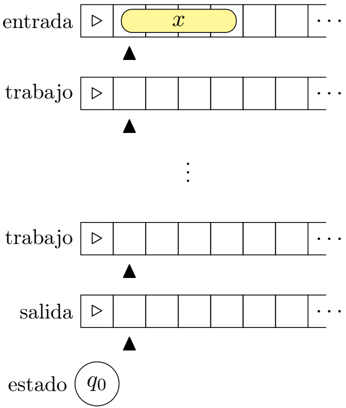


$$\delta(q, 1, \square) = (L, 1, R, 0, s)$$

Si
  estado $== q$
  cinta de entrada $== 1$
  cinta de trabajo $== \square$
entonces
  en cinta de entrada: mover cabeza a L
  en cinta de trabajo: escribir 1 *y mover cabeza a* R
  en cinta de salida: escribir 1 *(y mover a* R*)*
  estado $= s$

Configuración inicial

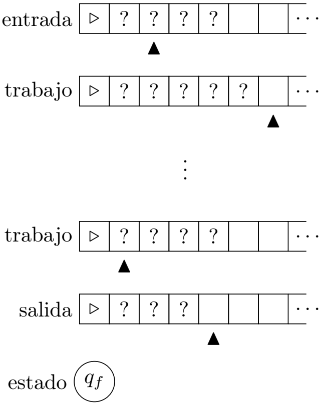
La **configuración inicial** de la máquina $M = (\Sigma, Q, \delta)$ con k cintas para $x \in \Sigma^*$ es:



- la cinta de entrada contiene x y el resto en blanco
- todas las cintas de trabajo en blanco
- la cinta de salida en blanco
- todas las cabezas en la segunda celda
- el estado es q_0 (estado inicial)

Configuración final

Una **configuración final** de la máquina $M = (\Sigma, Q, \delta)$ con k cintas es cualquiera de esta forma:



- el estado es q_f (se congela)
- cualquier cosa escrita en las cintas de trabajo o de salida
- cualquier posición de las cabezas
- decimos que una máquina en una configuración final **paró** o **terminó**
- si en la cinta de salida hay escrito un $\triangleright 1$ seguido de blancos decimos que la configuración final es **aceptadora**

Notación

Notación:

Fijemos un alfabeto finito Γ tal que $\triangleright \notin \Gamma$ y definamos $\Gamma' = \Gamma \cup \{\triangleright, \square\}$.

- las máquinas van a trabajar con el alfabeto Γ', que es lo mismo que Γ pero con el agregado de $\triangleright$ y $\square$.
- las entradas y salidas de las máquinas van a ser palabras sobre Γ

Cómputo

Definición

Un **cómputo** de $M = (\Gamma', Q, \delta)$ a partir de $x \in \Gamma^*$ es una secuencia $C_0, \ldots, C_\ell$ de configuraciones tal que

- C_0 es inicial para x
- C_ℓ es final y
- C_{i+1} es la evolución de C_i en un paso.

Llamamos **longitud** del cómputo a ℓ. Decimos que M con entrada x **termina** si existe un cómputo de M a partir de x.