

No determinismo y enumeradores



UNIVERSIDAD
DE CHILE

Arturo Merino



Plan

■ Recuerdo y plan

- **Recuerdo:**

Recuerdo y plan

- **Recuerdo:**
 - Variantes de máquinas de Turing

Recuerdo y plan

- **Recuerdo:**

- Variantes de máquinas de Turing
- Con *stay*, biinfinita, multicinta, RAM, etc...

Recuerdo y plan

- **Recuerdo:**

- Variantes de máquinas de Turing
- Con *stay*, biinfinita, multicinta, RAM, etc...

- **Plan:**

Recuerdo y plan

- **Recuerdo:**

- Variantes de máquinas de Turing
- Con *stay*, biinfinita, multicinta, RAM, etc...

- **Plan:**

- Máquinas de Turing no deterministas

Recuerdo y plan

- **Recuerdo:**

- Variantes de máquinas de Turing
- Con *stay*, biinfinita, multicinta, RAM, etc...

- **Plan:**

- Máquinas de Turing no deterministas
- Enumeradores

Máquinas de Turing no deterministas

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla
 $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla
 $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Γ es el *alfabeto de cinta*, finito, con $\Sigma \cup \{\sqcup\} \subseteq \Gamma$.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Γ es el *alfabeto de cinta*, finito, con $\Sigma \cup \{\sqcup\} \subseteq \Gamma$.

$\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$ es la *función de transición*.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Γ es el *alfabeto de cinta*, finito, con $\Sigma \cup \{\sqcup\} \subseteq \Gamma$.

$\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$ es la *función de transición*.

$q_0 \in Q$ es el *estado inicial*.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Γ es el *alfabeto de cinta*, finito, con $\Sigma \cup \{\sqcup\} \subseteq \Gamma$.

$\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$ es la *función de transición*.

$q_0 \in Q$ es el *estado inicial*.

$q_a \in Q$ es el *estado de aceptación*.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Γ es el *alfabeto de cinta*, finito, con $\Sigma \cup \{\sqcup\} \subseteq \Gamma$.

$\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$ es la *función de transición*.

$q_0 \in Q$ es el *estado inicial*.

$q_a \in Q$ es el *estado de aceptación*.

$q_r \in Q$ es el *estado de rechazo*, con $q_r \neq q_a$.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Γ es el *alfabeto de cinta*, finito, con $\Sigma \cup \{\sqcup\} \subseteq \Gamma$.

$\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$ es la *función de transición*.

$q_0 \in Q$ es el *estado inicial*.

$q_a \in Q$ es el *estado de aceptación*.

$q_r \in Q$ es el *estado de rechazo*, con $q_r \neq q_a$.

- Sin acciones posibles: la computación se detiene sin aceptar.

Máquinas de Turing no deterministas

Def.

Una *MT no determinista (MTND)* es una 7-tupla $N = (Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$, donde

Q es un conjunto finito de *estados*.

Σ es el *alfabeto de entrada*, con $\sqcup \notin \Sigma$.

Γ es el *alfabeto de cinta*, finito, con $\Sigma \cup \{\sqcup\} \subseteq \Gamma$.

$\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$ es la *función de transición*.

$q_0 \in Q$ es el *estado inicial*.

$q_a \in Q$ es el *estado de aceptación*.

$q_r \in Q$ es el *estado de rechazo*, con $q_r \neq q_a$.

- Sin acciones posibles: la computación se detiene sin aceptar.
- **Otros detalles:** (inicio cinta, detención, etc.) cómo en una MT.

Computación

Def.

Sean N una MTND y $w \in \Sigma^*$.

Def.

Sean N una MTND y $w \in \Sigma^*$.

El *árbol de configuraciones* de N sobre w se define recursivamente:

Def.

Sean N una MTND y $w \in \Sigma^*$.

El *árbol de configuraciones* de N sobre w se define recursivamente:

La raíz está etiquetada con la configuración inicial de N con entrada w .

Def.

Sean N una MTND y $w \in \Sigma^*$.

El *árbol de configuraciones* de N sobre w se define recursivamente:

La raíz está etiquetada con la configuración inicial de N con entrada w .

Cada nodo etiquetado con C tiene un hijo nuevo etiquetado con D por cada configuración D alcanzable desde C en un paso de N .

Computación

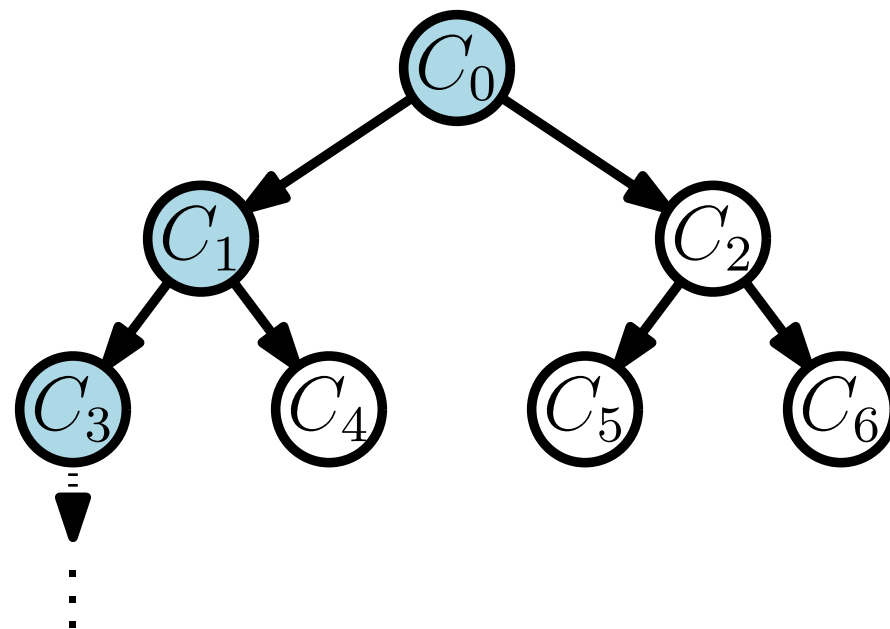
Def.

Sean N una MTND y $w \in \Sigma^*$.

El *árbol de configuraciones* de N sobre w se define recursivamente:

La raíz está etiquetada con la configuración inicial de N con entrada w .

Cada nodo etiquetado con C tiene un hijo nuevo etiquetado con D por cada configuración D alcanzable desde C en un paso de N .



Def.

Sean N una MTND y $w \in \Sigma^*$.

Computación

Def.

Sean N una MTND y $w \in \Sigma^*$.

N *acepta* w si existe un camino desde la raíz del árbol de configuraciones que llega a una configuración con estado q_a .

Def.

Sean N una MTND y $w \in \Sigma^*$.

N *acepta* w si existe un camino desde la raíz del árbol de configuraciones que llega a una configuración con estado q_a .

$$\mathcal{L}(N) = \{w \in \Sigma^* : N \text{ acepta } w\}.$$

Computación

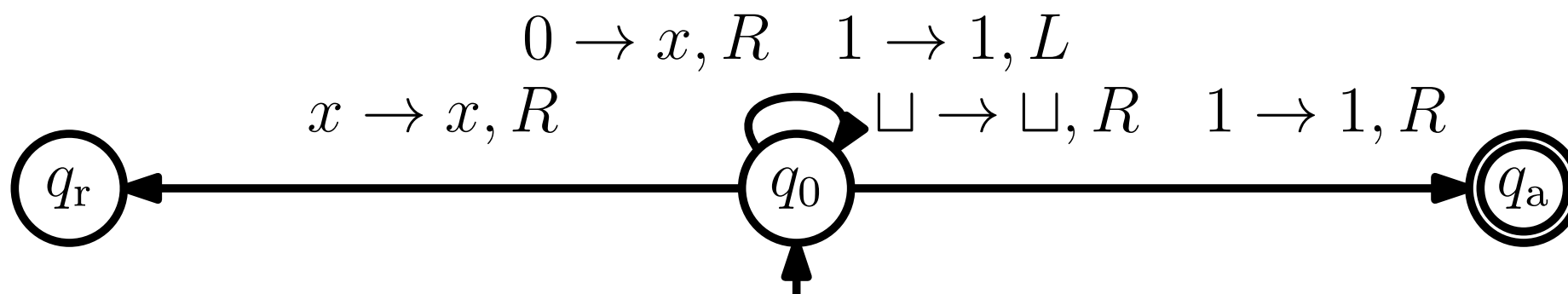
Def.

Sean N una MTND y $w \in \Sigma^*$.

N *acepta* w si existe un camino desde la raíz del árbol de configuraciones que llega a una configuración con estado q_a .

$$\mathcal{L}(N) = \{w \in \Sigma^* : N \text{ acepta } w\}.$$

- **Ej:** N_0 , con $\Sigma = \{0, 1\}$. ¿Cómo computa en ε , 01 y 1?



Diseñando máquinas de Turing no deterministas

- **Ej:** Diseñe una MTND para cada lenguaje.

Diseñando máquinas de Turing no deterministas

- **Ej:** Diseñe una MTND para cada lenguaje.
 - $\{ww : w \in \{0, 1\}^*\}$.

Diseñando máquinas de Turing no deterministas

- **Ej:** Diseñe una MTND para cada lenguaje.
 - $\{ww : w \in \{0, 1\}^*\}$.
 - $\mathcal{L}(M_1) \cup \mathcal{L}(M_2)$, con M_1, M_2 reconocedores sobre Σ .

Expresividad

Teo.

Sea N una MTND.

Existe una MT determinista equivalente a N .

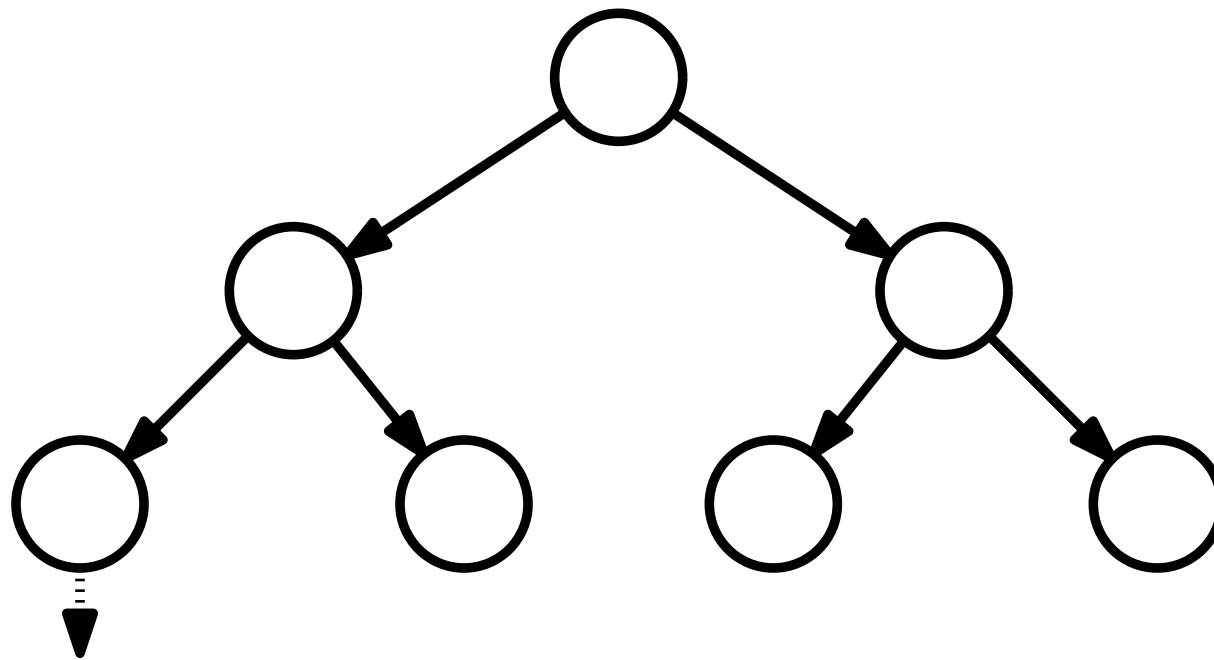
Expresividad

Teo.

Sea N una MTND.

Existe una MT determinista equivalente a N .

- **Dem:** Árbol de cómputo. ¿Cómo recorrerlo?



Decidir un lenguaje

Def.

Sea N una MTND.

Decidir un lenguaje

Def.

Sea N una MTND.

N es un *decididor* si todas sus ramas terminan, para toda entrada.

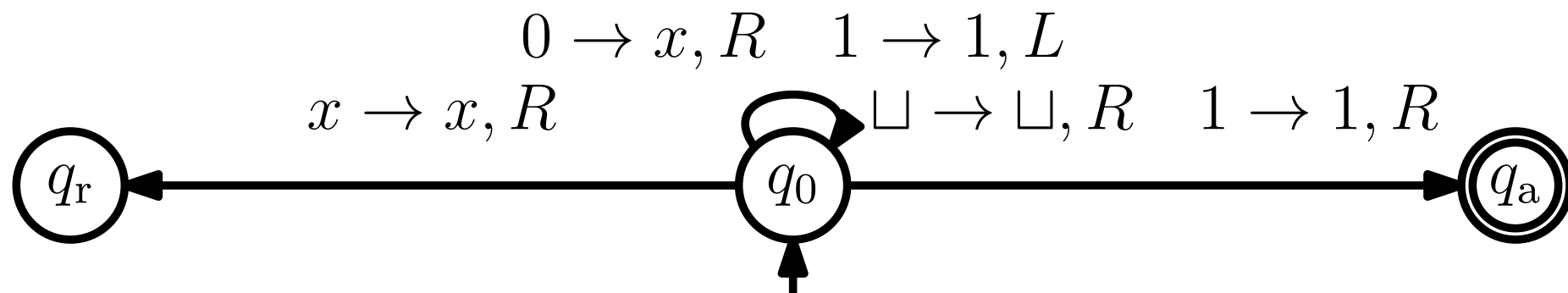
Decidir un lenguaje

Def.

Sea N una MTND.

N es un *decididor* si todas sus ramas terminan, para toda entrada.

- **Ej:** ¿Es N_0 un decididor?



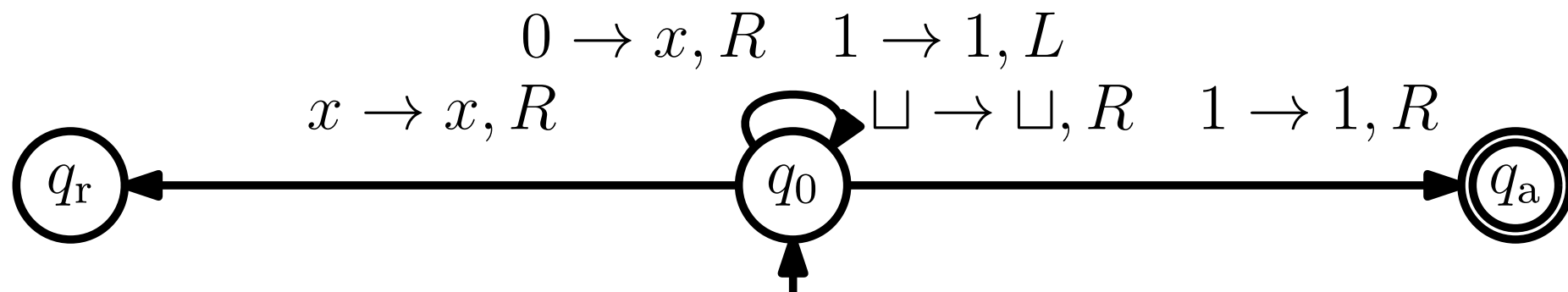
Decidir un lenguaje

Def.

Sea N una MTND.

N es un *decididor* si todas sus ramas terminan, para toda entrada.

- **Ej:** ¿Es N_0 un decididor?



Cor.

Sea $L \subseteq \Sigma^*$.

L es decidable si y solo si algún decididor no determinista reconoce L .

Enumeradores

Enumeradores

Def.

Un *enumerador* E sobre Σ es una MT determinista de dos cintas:

Enumeradores

Def.

Un *enumerador* E sobre Σ es una MT determinista de dos cintas:

Una *cinta de trabajo*, con las convenciones usuales.

Enumeradores

Def.

Un *enumerador* E sobre Σ es una MT determinista de dos cintas:

Una *cinta de trabajo*, con las convenciones usuales.

Una *cinta de salida*, de solo escritura: agrega símbolos de $\Sigma \cup \{\#\}$ al final, con $\# \notin \Sigma$.

Enumeradores

Def.

Un *enumerador* E sobre Σ es una MT determinista de dos cintas:

Una *cinta de trabajo*, con las convenciones usuales.

Una *cinta de salida*, de solo escritura: agrega símbolos de $\Sigma \cup \{\#\}$ al final, con $\# \notin \Sigma$.

E *imprime* un string $w \in \Sigma^*$ si en algún momento agrega $w\#$ como siguiente bloque de la cinta de salida.

Enumeradores

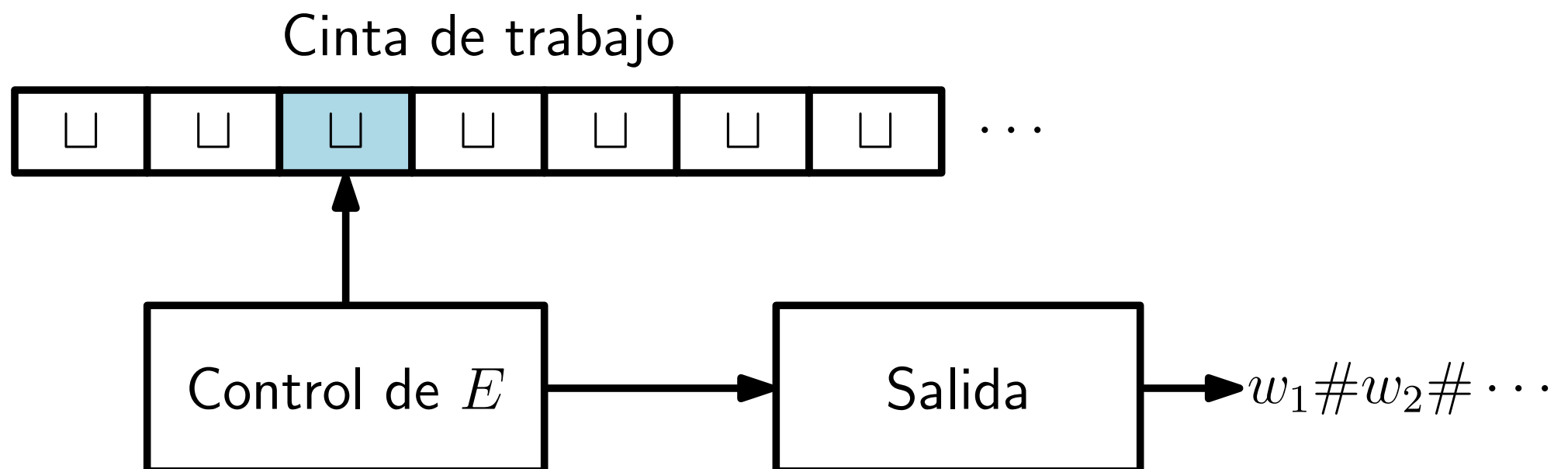
Def.

Un *enumerador* E sobre Σ es una MT determinista de dos cintas:

Una *cinta de trabajo*, con las convenciones usuales.

Una *cinta de salida*, de solo escritura: agrega símbolos de $\Sigma \cup \{\#\}$ al final, con $\# \notin \Sigma$.

E *imprime* un string $w \in \Sigma^*$ si en algún momento agrega $w\#$ como siguiente bloque de la cinta de salida.



Enumeradores

Def.

Sea E un enumerador de strings sobre Σ .

Enumeradores

Def.

Sea E un enumerador de strings sobre Σ .

El *lenguaje enumerado* por E es

$$\mathcal{L}(E) = \{w \in \Sigma^* : E \text{ imprime } w \text{ en algún momento}\}.$$

Enumeradores

Def.

Sea E un enumerador de strings sobre Σ .

El *lenguaje enumerado* por E es

$$\mathcal{L}(E) = \{w \in \Sigma^* : E \text{ imprime } w \text{ en algún momento}\}.$$

- **Ej:** Diseñe enumeradores para:

Enumeradores

Def.

Sea E un enumerador de strings sobre Σ .

El *lenguaje enumerado* por E es

$$\mathcal{L}(E) = \{w \in \Sigma^* : E \text{ imprime } w \text{ en algún momento}\}.$$

- **Ej:** Diseñe enumeradores para:
 - $\{0, 1\}^*$, $\{0^n 1^n : n \geq 0\}$, $\emptyset$.

Enumerar y reconocer

Teo.

Sea $L \subseteq \Sigma^*$.

L es Turing-reconocible si y solo si algún enumerador enumera L .

Enumerar y reconocer

Teo.

Sea $L \subseteq \Sigma^*$.

L es Turing-reconocible si y solo si algún enumerador enumera L .

- **Dem:**

Enumerar y reconocer

Teo.

Sea $L \subseteq \Sigma^*$.

L es Turing-reconocible si y solo si algún enumerador enumera L .

- **Dem:**

- Enumerador $\rightarrow$ reconocedor.

Enumerar y reconocer

Teo.

Sea $L \subseteq \Sigma^*$.

L es Turing-reconocible si y solo si algún enumerador enumera L .

- **Dem:**

- Enumerador $\rightarrow$ reconocedor.
- Reconocedor $\rightarrow$ enumerador.

Dudas/Preguntas