

Indecidibilidad y reducciones



UNIVERSIDAD
DE CHILE

Arturo Merino

Plan

■ Recuerdo y plan

- **Recuerdo:**

Recuerdo y plan

- **Recuerdo:**
 - Codificaciones y simulación universal.

Recuerdo y plan

- **Recuerdo:**

- Codificaciones y simulación universal.
- $A_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y acepta } w \}$.

Recuerdo y plan

- **Recuerdo:**

- Codificaciones y simulación universal.
- $A_{\text{MT}} = \{\langle M, w \rangle : M \text{ es una MT y acepta } w\}$.
- A_{MT} es Turing-reconocible e indecidible.

Recuerdo y plan

- **Recuerdo:**

- Codificaciones y simulación universal.
- $A_{\text{MT}} = \{\langle M, w \rangle : M \text{ es una MT y acepta } w\}$.
- A_{MT} es Turing-reconocible e indecidible.

- **Plan:**

Recuerdo y plan

- **Recuerdo:**

- Codificaciones y simulación universal.
- $A_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y acepta } w \}$.
- A_{MT} es Turing-reconocible e indecidible.

- **Plan:**

- Indecidibilidad mediante reducciones.

Recuerdo y plan

- **Recuerdo:**

- Codificaciones y simulación universal.
- $A_{\text{MT}} = \{\langle M, w \rangle : M \text{ es una MT y acepta } w\}$.
- A_{MT} es Turing-reconocible e indecidible.

- **Plan:**

- Indecidibilidad mediante reducciones.
- Historiales de computación.

Recuerdo y plan

- **Recuerdo:**

- Codificaciones y simulación universal.
- $A_{\text{MT}} = \{\langle M, w \rangle : M \text{ es una MT y acepta } w\}$.
- A_{MT} es Turing-reconocible e indecidible.

- **Plan:**

- Indecidibilidad mediante reducciones.
- Historiales de computación.
- El problema de correspondencia de Post.

El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.

El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.
 - $\text{HALT}_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y se detiene sobre } w \}$.

El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.
 - $\text{HALT}_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y se detiene sobre } w \}$.

Teo.

HALT_{MT} no es decidable.

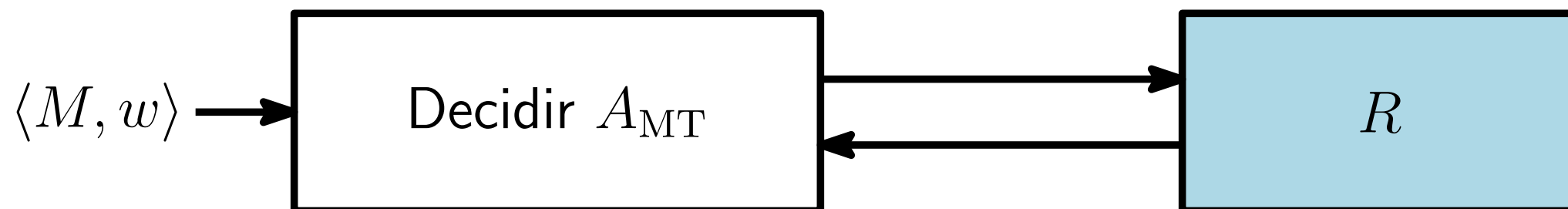
El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.
 - $\text{HALT}_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y se detiene sobre } w \}$.

Teo.

HALT_{MT} no es decidable.

- **Dem:** supongamos que R decide HALT_{MT} .



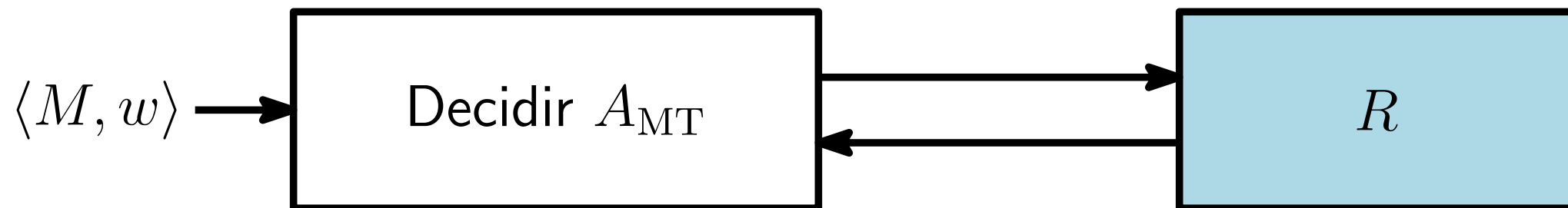
El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.
 - $\text{HALT}_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y se detiene sobre } w \}$.

Teo.

HALT_{MT} no es decidable.

- **Dem:** supongamos que R decide HALT_{MT} .



- ¿Cómo usar R para decidir A_{MT} ?

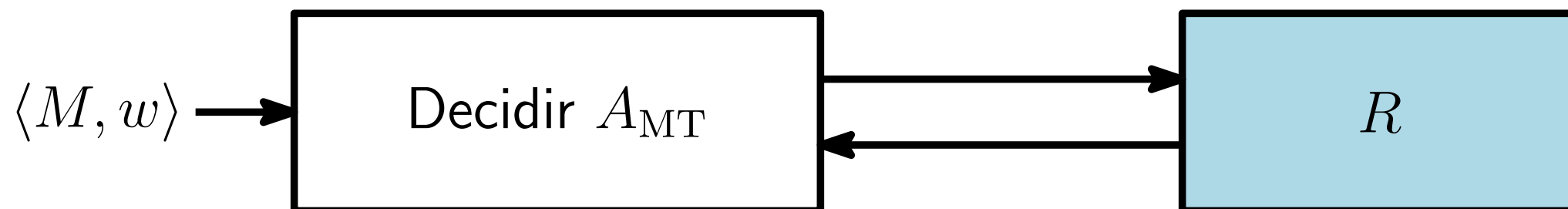
El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.
 - $\text{HALT}_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y se detiene sobre } w \}$.

Teo.

HALT_{MT} no es decidable.

- **Dem:** supongamos que R decide HALT_{MT} .



- ¿Cómo usar R para decidir A_{MT} ?
- **Idea clave:** reducción.

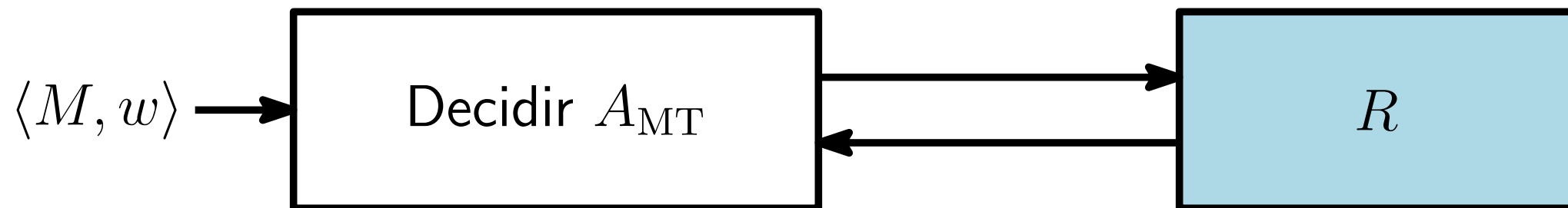
El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.
 - $\text{HALT}_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y se detiene sobre } w \}$.

Teo.

HALT_{MT} no es decidable.

- **Dem:** supongamos que R decide HALT_{MT} .



- ¿Cómo usar R para decidir A_{MT} ?
- **Idea clave:** reducción.
 - A_{MT} se reduce a HALT_{MT} .

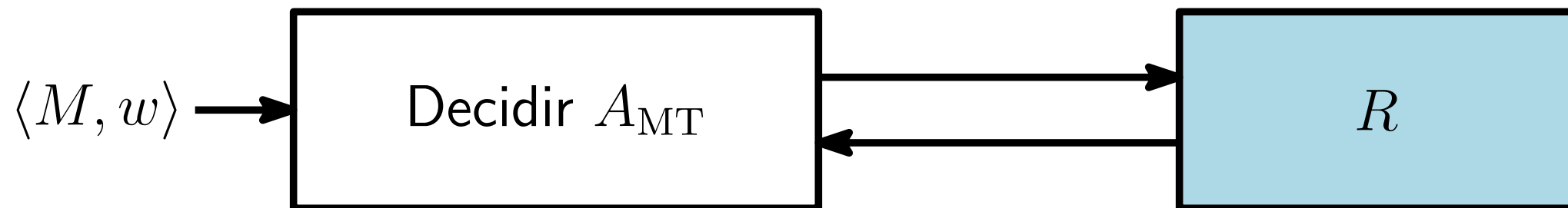
El problema de la detención

- **Halt** $\leftarrow$ chequear si una MT termina en un input.
 - $\text{HALT}_{\text{MT}} = \{ \langle M, w \rangle : M \text{ es una MT y se detiene sobre } w \}$.

Teo.

HALT_{MT} no es decidable.

- **Dem:** supongamos que R decide HALT_{MT} .



- ¿Cómo usar R para decidir A_{MT} ?
- **Idea clave:** reducción.
 - A_{MT} se reduce a HALT_{MT} .
- **Notación:** $A_{\text{MT}} \leq \text{HALT}_{\text{MT}}$.

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.
 - $E_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) = \emptyset\}.$

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.
 - $E_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) = \emptyset\}$.

Teo.

E_{MT} no es decidable.

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.
 - $E_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) = \emptyset\}.$

Teo.

E_{MT} no es decidable.

- **Dem:** reducir A_{MT} .

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.
 - $E_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) = \emptyset\}.$

Teo.

E_{MT} no es decidable.

- **Dem:** reducir A_{MT} .
- **Reg** $\leftarrow$ chequear si una MT acepta un lenguaje regular.

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.

- $E_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) = \emptyset\}.$

Teo.

E_{MT} no es decidable.

- **Dem:** reducir A_{MT} .
- **Reg** $\leftarrow$ chequear si una MT acepta un lenguaje regular.
 - $\text{REGULAR}_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) \text{ es regular}\}.$

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.
 - $E_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) = \emptyset\}$.

Teo.

E_{MT} no es decidable.

- **Dem:** reducir A_{MT} .
- **Reg** $\leftarrow$ chequear si una MT acepta un lenguaje regular.
 - $\text{REGULAR}_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) \text{ es regular}\}$.

Teo.

$\text{REGULAR}_{\text{MT}}$ no es decidable.

Lenguaje vacío

- **Empty** $\leftarrow$ chequear si una MT acepta algún string.

- $E_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) = \emptyset\}.$

Teo.

E_{MT} no es decidable.

- **Dem:** reducir A_{MT} .
- **Reg** $\leftarrow$ chequear si una MT acepta un lenguaje regular.
 - $\text{REGULAR}_{\text{MT}} = \{\langle M \rangle : M \text{ es una MT y } \mathcal{L}(M) \text{ es regular}\}.$

Teo.

$\text{REGULAR}_{\text{MT}}$ no es decidable.

- **Dem:** reducir A_{MT} .

Equivalencia de MT

- **Equality** $\leftarrow$ Chequear que dos MT son equivalentes.

Equivalencia de MT

- **Equality** $\leftarrow$ Chequear que dos MT son equivalentes.

Teo.

$$EQ_{MT} = \{ \langle M_1, M_2 \rangle : M_1, M_2 \text{ son MT y } \mathcal{L}(M_1) = \mathcal{L}(M_2) \}.$$

EQ_{MT} no es decidable.

Equivalencia de MT

- **Equality** $\leftarrow$ Chequear que dos MT son equivalentes.

Teo.

$$EQ_{MT} = \{ \langle M_1, M_2 \rangle : M_1, M_2 \text{ son MT y } \mathcal{L}(M_1) = \mathcal{L}(M_2) \}.$$

EQ_{MT} no es decidable.

- **Dem:** reducir E_{MT} .

Historiales de computación

Historiales de computación

- Recuerdo: configuraciones.

Historiales de computación

- Recuerdo: configuraciones.

Def.

Un *historial de computación aceptador* de M sobre w es una secuencia finita $C_0, \dots, C_t$, con $t \geq 0$, tal que:

C_0 es la configuración inicial de M sobre w .

$C_i \vdash_M C_{i+1}$ para todo $0 \leq i < t$.

C_t es una configuración de aceptación.

Autómatas linealmente acotados

Def.

Una *MT linealmente acotada* (LBA) no mueve el cabezal fuera de las celdas ocupadas por la entrada.

Para la entrada vacía, reservamos una celda en blanco.

Autómatas linealmente acotados

Def.

Una *MT linealmente acotada* (LBA) no mueve el cabezal fuera de las celdas ocupadas por la entrada.

Para la entrada vacía, reservamos una celda en blanco.

- Con q estados, g símbolos y $n \geq 1$ celdas ¿Cuántas configuraciones?

Autómatas linealmente acotados

Def.

Una *MT linealmente acotada* (LBA) no mueve el cabezal fuera de las celdas ocupadas por la entrada.

Para la entrada vacía, reservamos una celda en blanco.

- Con q estados, g símbolos y $n \geq 1$ celdas ¿Cuántas configuraciones?

Teo.

$$A_{\text{LBA}} = \{ \langle B, w \rangle : B \text{ es una LBA y acepta } w \}.$$

A_{LBA} es decidable.

Autómatas linealmente acotados

Def.

Una *MT linealmente acotada* (LBA) no mueve el cabezal fuera de las celdas ocupadas por la entrada.

Para la entrada vacía, reservamos una celda en blanco.

- Con q estados, g símbolos y $n \geq 1$ celdas ¿Cuántas configuraciones?

Teo.

$$A_{\text{LBA}} = \{ \langle B, w \rangle : B \text{ es una LBA y acepta } w \}.$$

A_{LBA} es decidable.

- **Dem:** ¿qué ocurre si una configuración se repite?

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{\langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset\}.$$

E_{LBA} no es decidable.

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{\langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset\}.$$

E_{LBA} no es decidable.

- **Dem:** reducir A_{TM} .

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{ \langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset \}.$$

E_{LBA} no es decidable.

- **Dem:** reducir A_{TM} .
 - Crear $B_{M,w}$ LBA tal que $\mathcal{L}(B) = \emptyset$ ssi $M(w) = 0$.

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{ \langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset \}.$$

E_{LBA} no es decidable.

- **Dem:** reducir A_{TM} .
 - Crear $B_{M,w}$ LBA tal que $\mathcal{L}(B) = \emptyset$ ssi $M(w) = 0$.
- **Similar:**

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{ \langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset \}.$$

E_{LBA} no es decidable.

- **Dem:** reducir A_{TM} .
 - Crear $B_{M,w}$ LBA tal que $\mathcal{L}(B) = \emptyset$ ssi $M(w) = 0$.
- **Similar:**
 - Dada GLC ver si acepta todos los string.

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{\langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset\}.$$

E_{LBA} no es decidable.

- **Dem:** reducir A_{TM} .
 - Crear $B_{M,w}$ LBA tal que $\mathcal{L}(B) = \emptyset$ ssi $M(w) = 0$.
- **Similar:**
 - Dada GLC ver si acepta todos los string.

Teo.

$$\text{ALL}_{\text{GLC}} = \{\langle G \rangle : G \text{ es una GLC y } \mathcal{L}(G) = \Sigma_G^*\}.$$

ALL_{GLC} no es decidable.

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{ \langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset \}.$$

E_{LBA} no es decidable.

- **Dem:** reducir A_{TM} .
 - Crear $B_{M,w}$ LBA tal que $\mathcal{L}(B) = \emptyset$ ssi $M(w) = 0$.
- **Similar:**
 - Dada GLC ver si acepta todos los string.

Teo.

$$\text{ALL}_{\text{GLC}} = \{ \langle G \rangle : G \text{ es una GLC y } \mathcal{L}(G) = \Sigma_G^* \}.$$

ALL_{GLC} no es decidable.

- ¿Consecuencias?

Lenguaje vacío en LBA

Teo.

$$E_{\text{LBA}} = \{ \langle B \rangle : B \text{ es una LBA y } \mathcal{L}(B) = \emptyset \}.$$

E_{LBA} no es decidable.

- **Dem:** reducir A_{TM} .
 - Crear $B_{M,w}$ LBA tal que $\mathcal{L}(B) = \emptyset$ ssi $M(w) = 0$.
- **Similar:**
 - Dada GLC ver si acepta todos los string.

Teo.

$$\text{ALL}_{\text{GLC}} = \{ \langle G \rangle : G \text{ es una GLC y } \mathcal{L}(G) = \Sigma_G^* \}.$$

ALL_{GLC} no es decidable.

- ¿Consecuencias?
 - No podemos decidir si dos GLC son equivalentes.

Correspondencia de Post

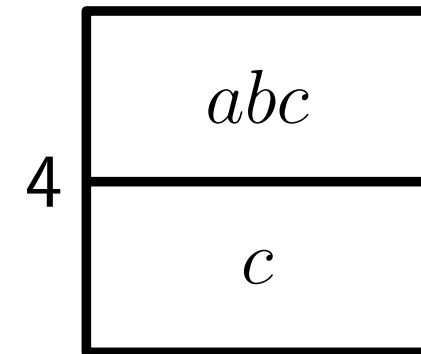
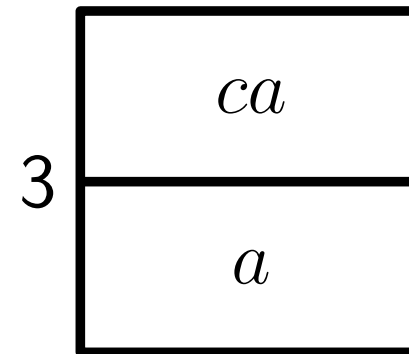
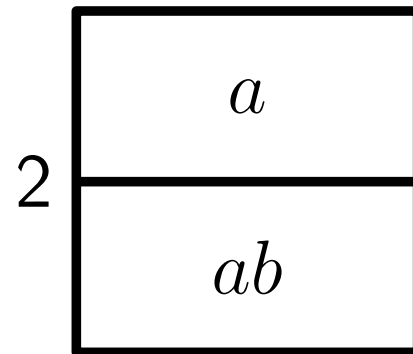
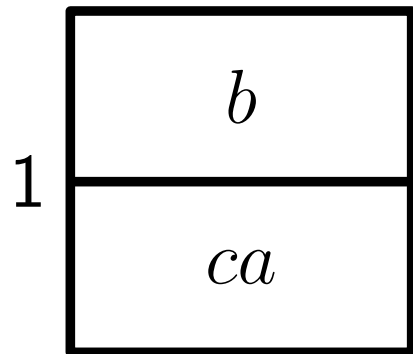
Correspondencia de Post

- **Input:** un conjunto finito de dominós
 $P = \{(t_1, b_1), \dots, (t_k, b_k)\} \subseteq (\Sigma^*)^2$.

Correspondencia de Post

- **Input:** un conjunto finito de dominós

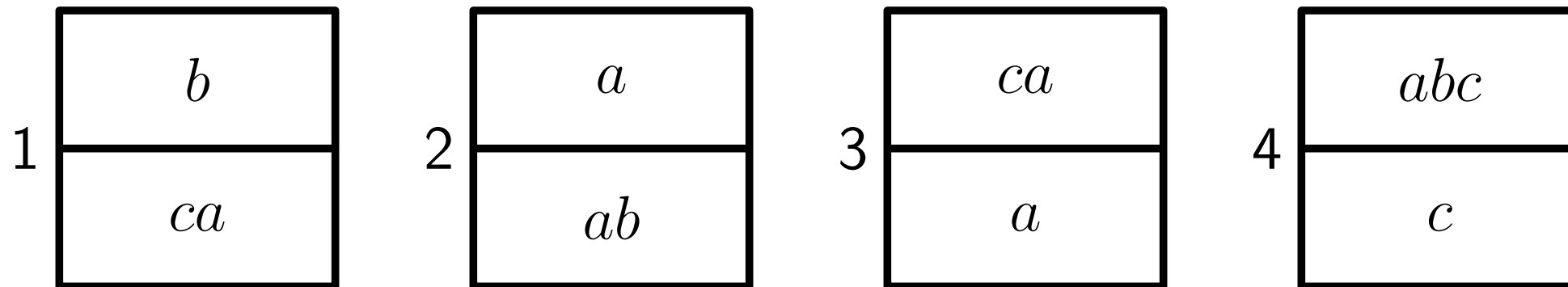
$$P = \{(t_1, b_1), \dots, (t_k, b_k)\} \subseteq (\Sigma^*)^2.$$



Correspondencia de Post

- **Input:** un conjunto finito de dominós

$$P = \{(t_1, b_1), \dots, (t_k, b_k)\} \subseteq (\Sigma^*)^2.$$

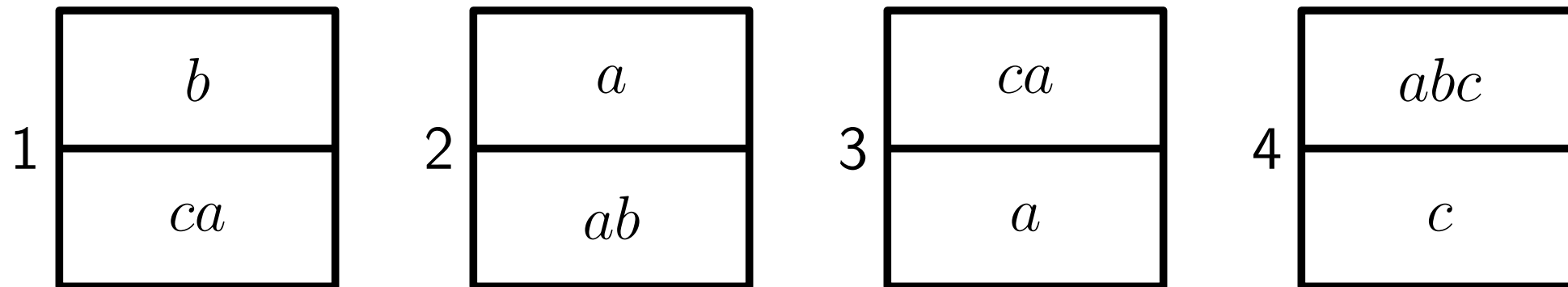


- **Decide:** ¿Hay una secuencia no vacía de dominós con el mismo string arriba y abajo? (Se puede repetir dominós)

Correspondencia de Post

- **Input:** un conjunto finito de dominós

$$P = \{(t_1, b_1), \dots, (t_k, b_k)\} \subseteq (\Sigma^*)^2.$$

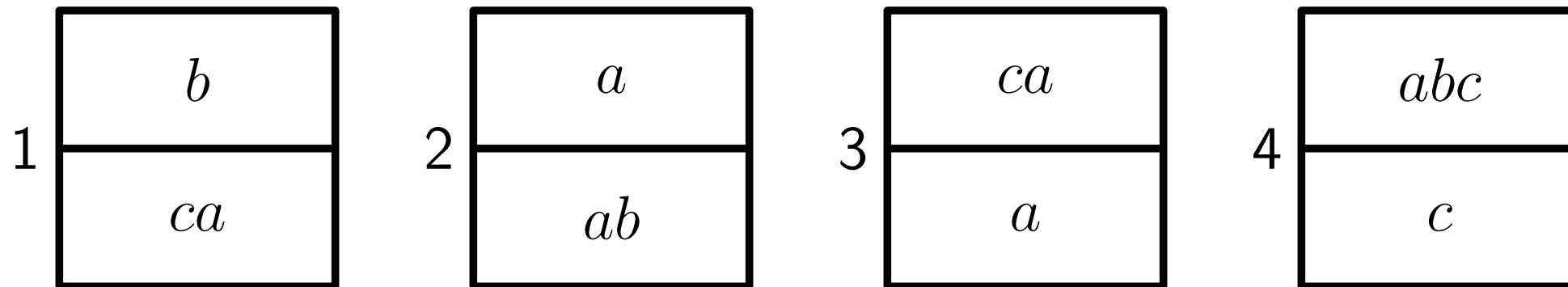


- **Decide:** ¿Hay una secuencia no vacía de dominós con el mismo string arriba y abajo? (Se puede repetir dominós)
- **Ej:** $(2, 1, 3, 2, 4)$: $a b c a a a b c = a b c a a a b c = a b c a a a b c$.

Correspondencia de Post

- **Input:** un conjunto finito de dominós

$$P = \{(t_1, b_1), \dots, (t_k, b_k)\} \subseteq (\Sigma^*)^2.$$



- **Decide:** ¿Hay una secuencia no vacía de dominós con el mismo string arriba y abajo? (Se puede repetir dominós)
- **Ej:** $(2, 1, 3, 2, 4)$: $a b c a a a b c = a b c a a a b c = a b c a a a b c$.

Def.

$PCP = \{\langle P \rangle : \text{hay una secuencia no vacía de dominós de } P \text{ con el mismo string arriba y abajo}\}.$

Indecidibilidad de PCP

Teo.

PCP no es decidable.

Indecidibilidad de PCP

Teo.

PCP no es decidable.

- **Sketch:** reducir A_{MT} .

Indecidibilidad de PCP

Teo.

PCP no es decidable.

- **Sketch:** reducir A_{MT} .
 - Dominós representan los historiales.

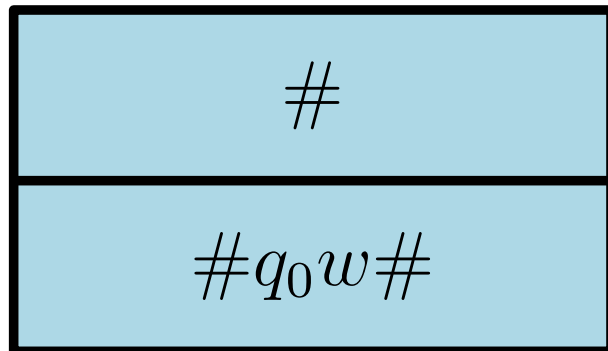
Indecidibilidad de PCP

Teo.

PCP no es decidable.

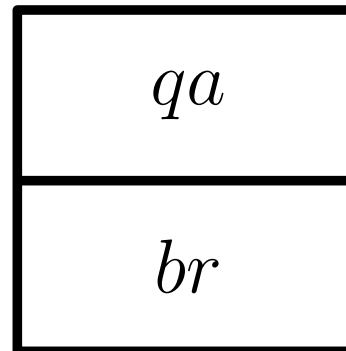
- **Sketch:** reducir A_{MT} .
 - Dominós representan los historiales.

Inicio



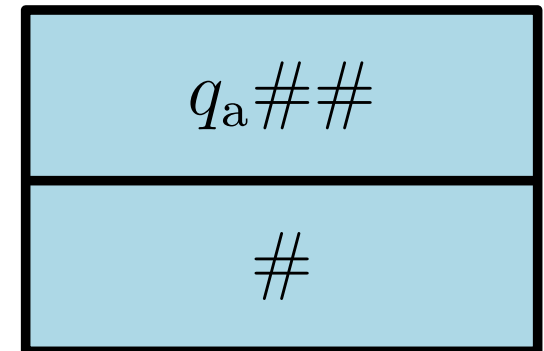
Comenzar aquí,
abajo aparece q_0w .

Transiciones



$\delta(q, a) = (r, b, R)$.
Copiar y actualizar el
cabezal.

Fin



Al aceptar, cerrar.

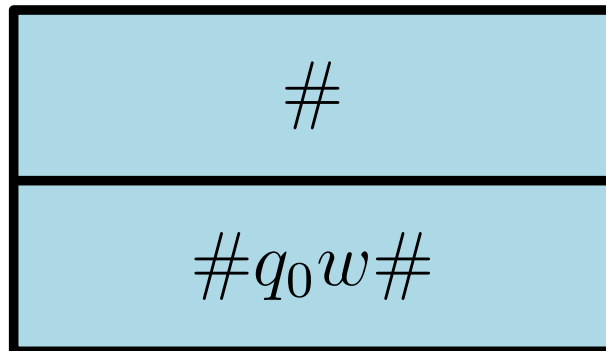
Indecidibilidad de PCP

Teo.

PCP no es decidable.

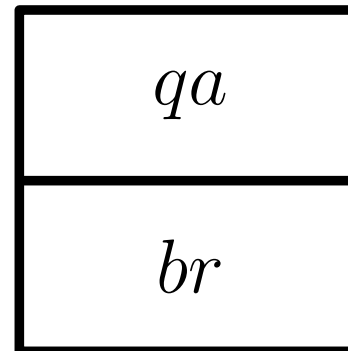
- **Sketch:** reducir A_{MT} .
 - Dominós representan los historiales.

Inicio



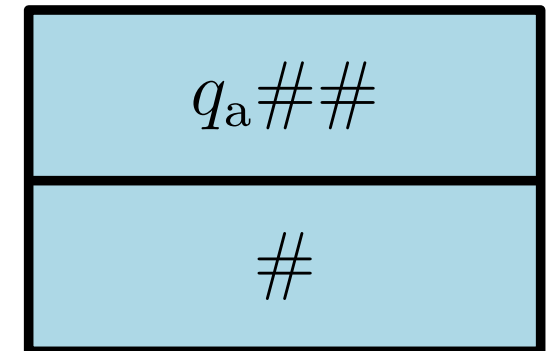
Comenzar aquí,
abajo aparece q_0w .

Transiciones



$\delta(q, a) = (r, b, R)$.
Copiar y actualizar el
cabezal.

Fin



Al aceptar, cerrar.

- Faltan dominós: copias de string, transiciones hacia la izquierda.

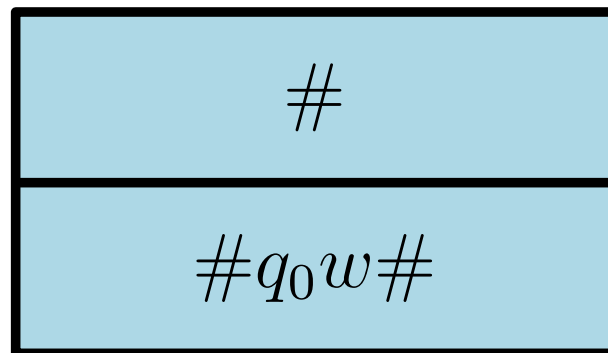
Indecidibilidad de PCP

Teo.

PCP no es decidable.

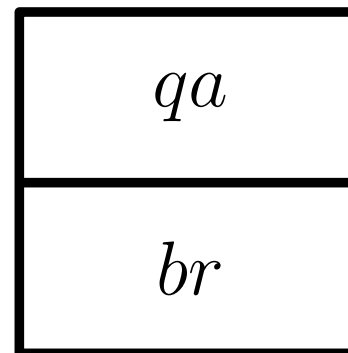
- **Sketch:** reducir A_{MT} .
 - Dominós representan los historiales.

Inicio



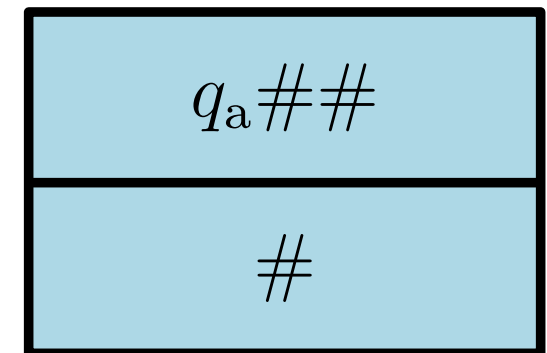
Comenzar aquí,
abajo aparece q_0w .

Transiciones



$\delta(q, a) = (r, b, R)$.
Copiar y actualizar el
cabezal.

Fin



Al aceptar, cerrar.

- Faltan dominós: copias de string, transiciones hacia la izquierda.
- Hay que forzar el inicio de alguna manera.

Dudas/Preguntas