Fundamentos de Algoritmos y Computabilidad

* Gramáticas no restringidas
* Clasificación de problemas
* Decidibilidad

Máquinas de Turing

Tipo	Lenguajes	Tipo de máquina	Normas para la gramática
0	Recursivamente enumerables	Máquina de Turing	No restringida
1	Sensibles al contexto	Autómata lineal acotado	$\alpha \rightarrow \beta$, $\lvert\alpha\rvert \leq \lvert\beta\rvert$
2	Independientes del contexto	Autómata de pila	$A \rightarrow \gamma$
3	Regulares	Autómata finito	$A \rightarrow aB$ $A \rightarrow a$

¿Cómo es la gramática de un lenguaje aceptado por una máquina de Turing?

Máquinas de Turing

Gramáticas no restringidas

Una gramática no restringida es una colección de 4 elementos $G=(\Sigma,N,S,P)$ donde:

- Σ es un alfabeto de terminales
- N es un alfabeto de no terminales
- $S \in N$ es el símbolo inicial
- P es un conjunto de producciones de la forma $\alpha \rightarrow \beta$, donde $\alpha \in (N \cup \Sigma)^+$ y $\beta \in (N \cup \Sigma)^*$

Máquinas de Turing

Gramáticas no restringidas

S→aBSc|abc

A→aBA | a

B→bb | BaB

S→aBSc|aAbcA

aBa→aB | a

aAa→bb | A

El lado izquierdo puede tener varios símbolos

Máquinas de Turing

Gramáticas no restringidas

S→aBSc|abc

Ba→aB

Bb→bb

Máquinas de Turing

Gramáticas no restringidas

S→aBSc|abc

Ba→aB

Bb→bb

S → aB<u>S</u>c

→ a<u>Ba</u>bcc

→ aa<u>Bb</u>cc

→ aabbcc

Máquinas de Turing

Gramáticas no restringidas

$S \rightarrow aBSc \mid abc$

$Ba \rightarrow aB$

$Bb \rightarrow bb$

- $L(G)=\{a^n b^n c^n \mid n \geq 1\}$

Máquinas de Turing

Tesis de Church-Turing

- Todo algoritmo es Turing-computable

El problema del MCD

- El problema consiste en:

Calcular el Máximo Común Divisor entre dos números enteros positivos

El problema del MCD

$r_0 \leftarrow a$ $\quad$ $r_1 \leftarrow b$
$i \leftarrow 1$
mientras $r_i \neq 0$ haga
    $r_{i+1} \leftarrow r_{i-1}$ mod r_i
    $i \leftarrow i+1$
$mcd(a,b) = r_{i-1}$

Máquinas de Turing

El problema del MCD

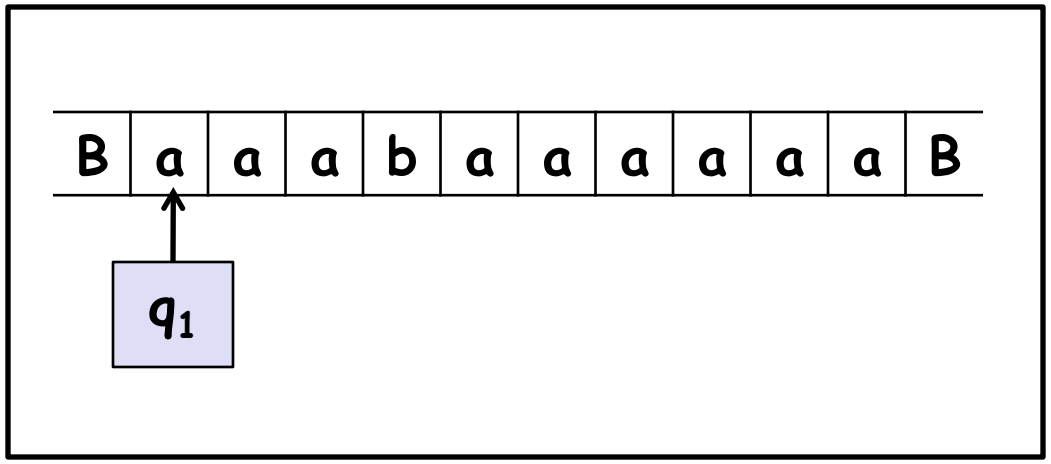
$r_0 \leftarrow a$ $r_1 \leftarrow b$
$i \leftarrow 1$
mientras $r_i \neq 0$ haga
    $r_{i+1} \leftarrow r_{i-1}$ mod r_i
    $i \leftarrow i+1$
$mcd(a,b) = r_{i-1}$

Es posible diseñar un algoritmo que resuelva el **problema del MCD**

Este es un problema decidible o resoluble

Máquinas de Turing

El problema del MCD



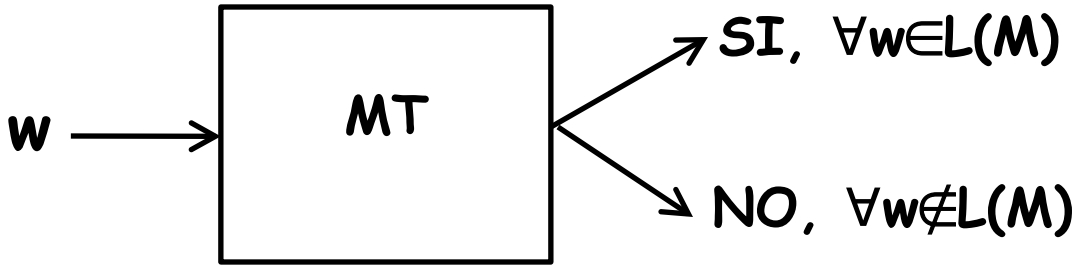
Es posible diseñar una MT que resuelve el **problema del MCD**

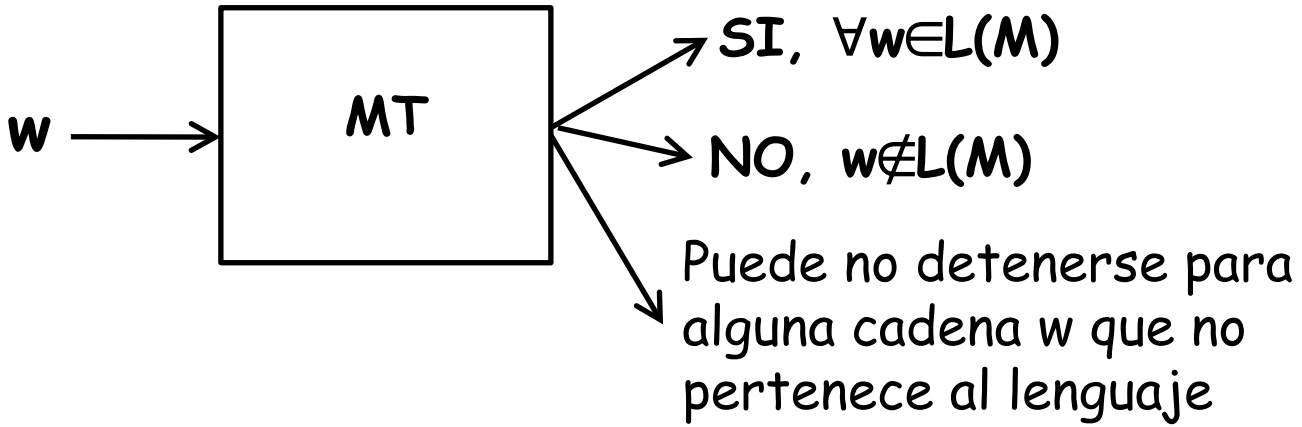
Máquinas de Turing

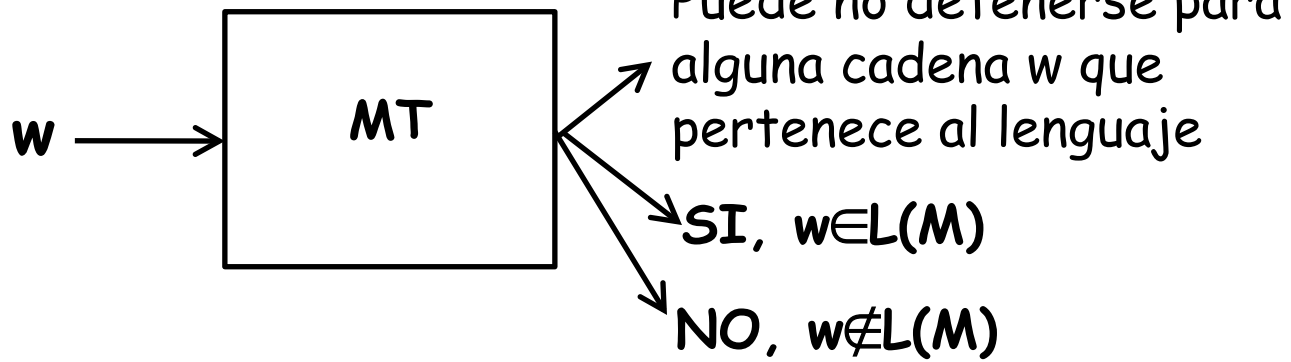
Clasificación de problemas

Se tienen tres clases de problemas:

- **Problemas decidibles o resolubles**
- **Problemas parcialmente decidibles**
- **Problemas no decidibles o irresolubles**


w
MT
SI, ∀w∈L(M)
NO, ∀w∉L(M)


w
MT
SI, ∀w∈L(M)
NO, w∉L(M)
Puede no detenerse para alguna cadena w que no pertenece al lenguaje


w
MT
Puede no detenerse para alguna cadena w que pertenece al lenguaje
SI, w∈L(M)
NO, w∉L(M)

Máquinas de Turing

- Aquellos para los que existe un algoritmo, es decir, una MT que se detiene tanto cuando se acepta la entrada como cuando no. **Problemas decidibles o resolubles**
- Aquellos problemas para los que existe una MT que se detiene cuando se acepta la entrada y corre indefinidamente para algunas entradas que no deben ser aceptadas. **Problemas parcialmente decidibles**
- Aquellos que solo se pueden resolver por una MT que cuando se debe aceptar la entrada, es posible que continúe ejecutándose indefinidamente. **Problemas indecidibles o irresolubles**

Máquinas de Turing

El problema de la parada de Lagarias, 1985

- Dado el siguiente algoritmo:

```
Entrar X
Mientras X≠1 hacer:
      si X es par, X=X/2
      sino, X=3X+1
Parar
```

indicar si para cualquier X se detiene o no

Máquinas de Turing

El problema de la parada de Lagarias, 1985

- Dado el siguiente algoritmo:

```
Entrar X
Mientras X≠1 hacer:
        si X es par, X=X/2
        sino, X=3X+1
Parar
```

Si el valor inicial de X es 7, tomaría los siguientes valores: 7,22,11,34,17,52,26,13,40,20,10,5,16,8,4,2,1

Máquinas de Turing

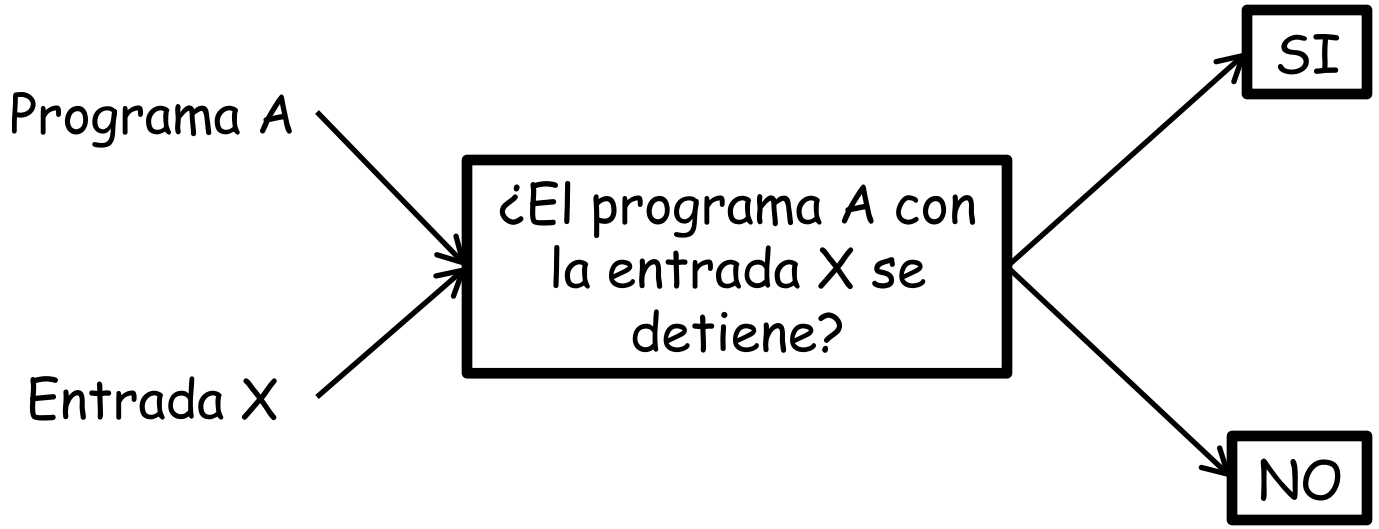
El problema de la parada de Lagarias, 1985

- Dado el siguiente algoritmo:

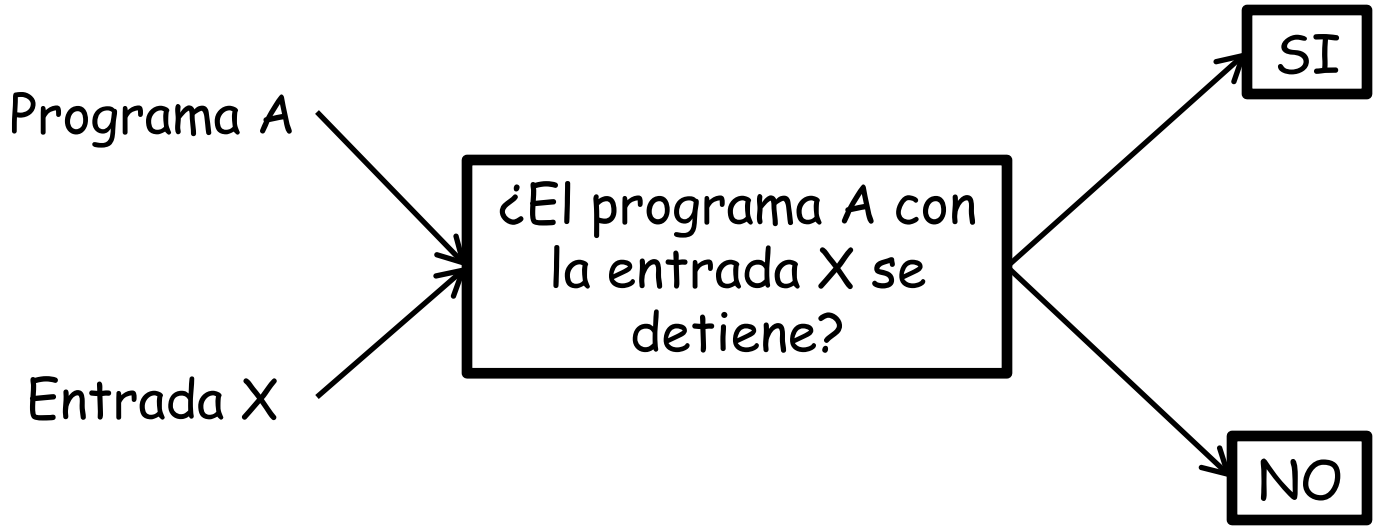
```
Entrar X
Mientras X≠1 hacer:
        si X es par, X=X/2
        sino, X=3X+1
Parar
```

No hay manera de decidir, en general, si la ejecución se detiene o no

El problema de la parada



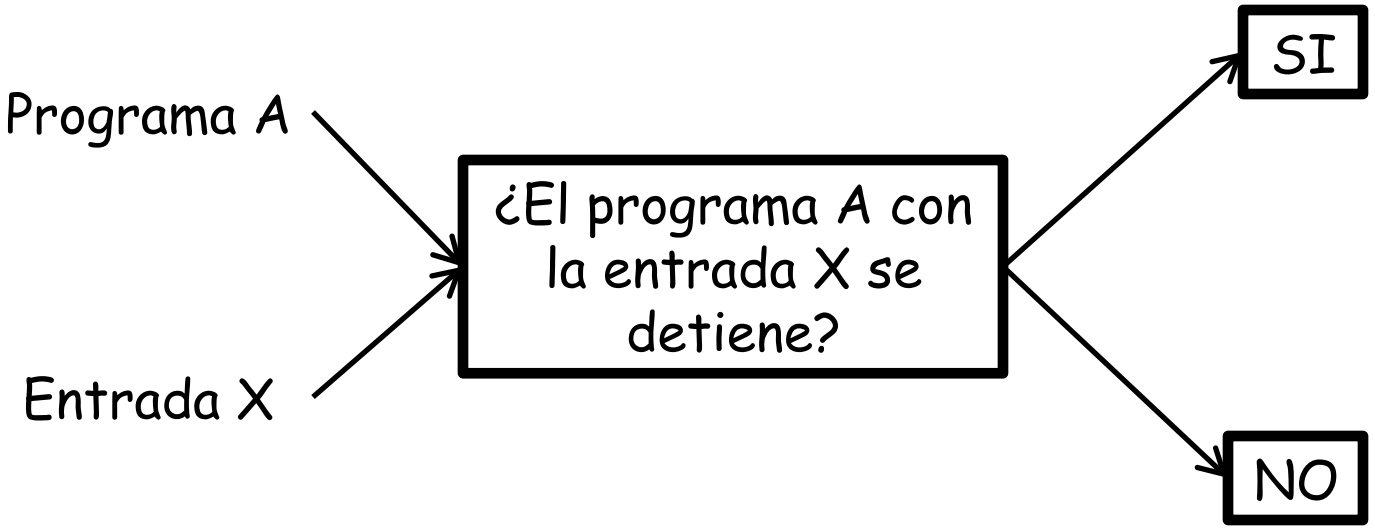
El problema de la parada



El **problema de la parada** es no decidible o no resoluble

Máquinas de Turing

El problema de la parada



ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING

Máquinas de Turing

El problema de la equivalencia

- Dados dos programas A y B, calculan lo mismo?

Máquinas de Turing

El problema de la equivalencia

- Dados dos programas A y B, calculan lo mismo?

El **problema de la equivalencia** es no decidible o no resoluble

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles

Tratables
Duros o Intratables

Problemas no decidibles o no resolubles

No computables
Fuertemente no computables

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles

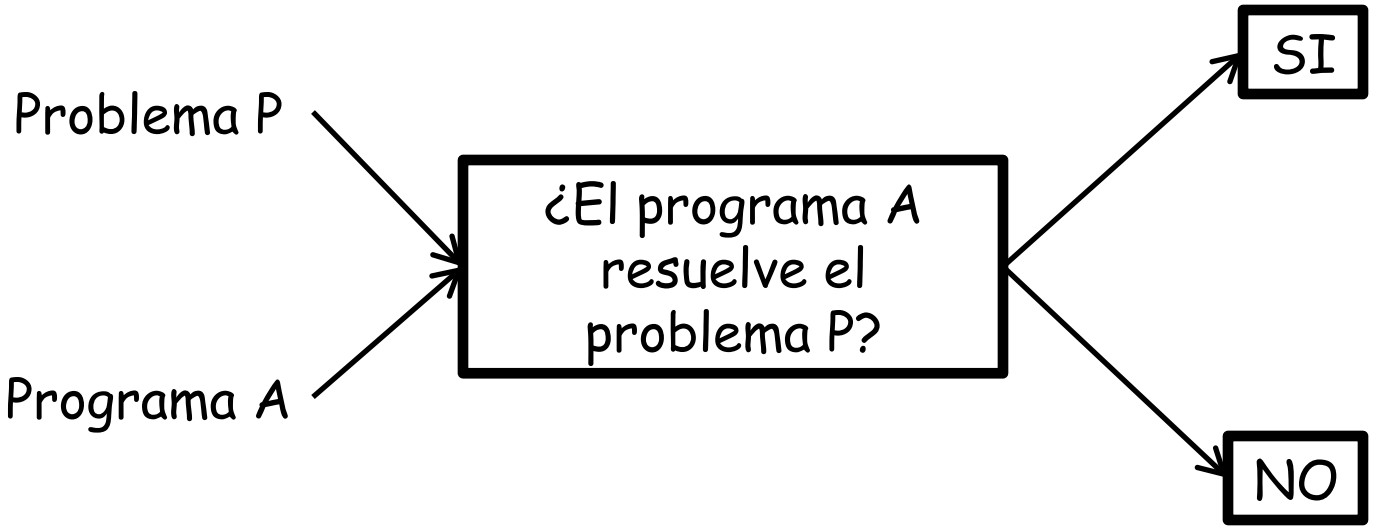
Tratables
Duros o Intratables

Problemas no decidibles o no resolubles

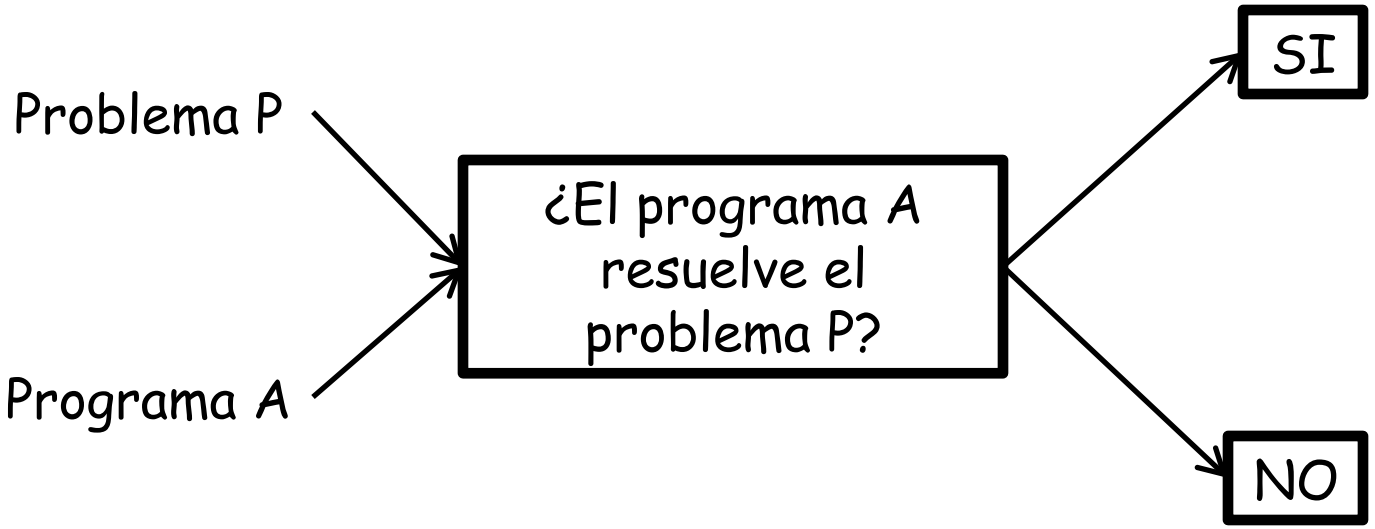
No computables
Fuertemente no computables

El **problema de la parada** es no decidible no computable

El problema de la verificación



El problema de la verificación



El **problema de la verificación** es fuertemente no computable

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles

Tratables
Duros o Intratables

Problemas no decidibles o no resolubles

No computables
Fuertemente no computables

El **problema de la verificación** es no decidible fuertemente no computable

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles*

Tratables
Duros o Intratables

Problemas no decidibles o no resolubles

No computables
Fuertemente no computables

Máquinas de Turing

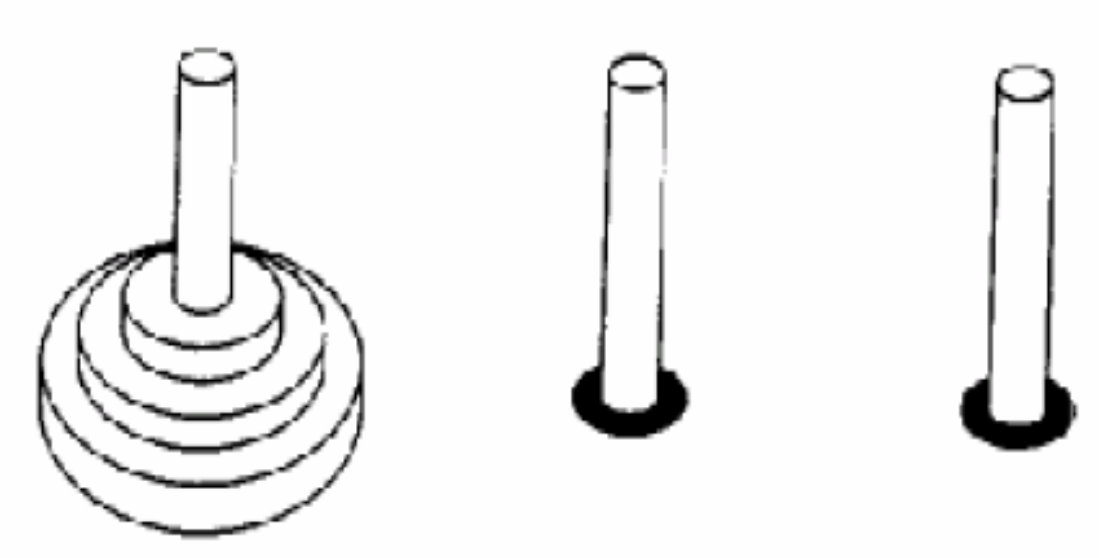
El problema de las torres de Hanoi

- El problema consiste en:

Mover los N anillos de la torre A a la torre B o C, usando la otra como ayuda, pero sin que haya un disco de mayor diámetro sobre otro de menor

Máquinas de Turing

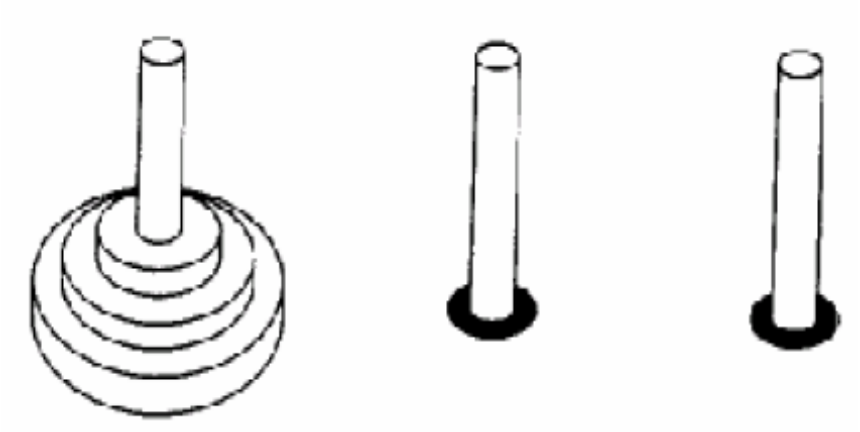
El problema de las torres de Hanoi



(A) (B) (C)

Máquinas de Turing

El problema de las torres de Hanoi

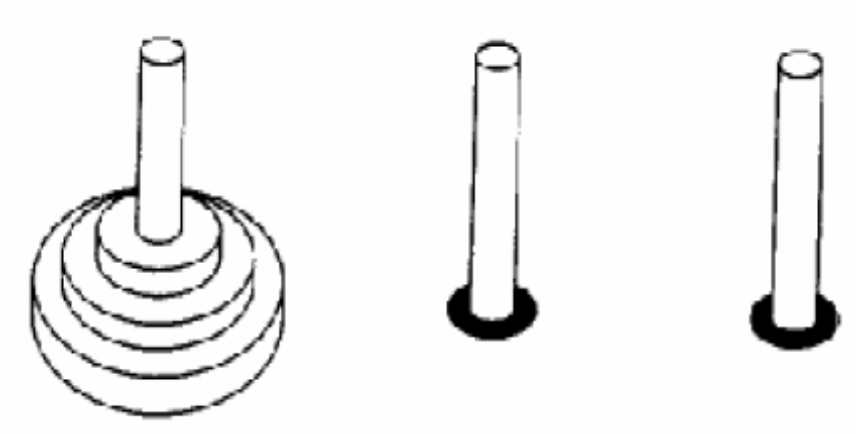


- La cota mínima de complejidad para este problema es 2^N

N	Operaciones
10	1.024
20	1.048.576
50	1.125.899.906.842.624

Máquinas de Turing

El problema de las torres de Hanoi



- Un computador capaz de hacer 1 millón de operaciones por segundo, tardaría 1 ms en resolver el juego con 10 anillos y casi 36 años si se colocan 50 anillos

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles

Tratables
Duros o Intratables

Problemas no decidibles o irresolubles

No computables
Fuertemente no computables

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles

Tratables
Duros o Intratables

Problemas no decidibles o irresolubles

No computables
Fuertemente no computables

El problema de las **torres de Hanoi** es decidible intratable

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles

Tratables
Duros o Intratables

Problemas no decidibles o irresolubles

No computables
Fuertemente no computables

Clasifique el problema del MCD

Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles

Tratables
Duros o Intratables

Problemas no decidibles o irresolubles

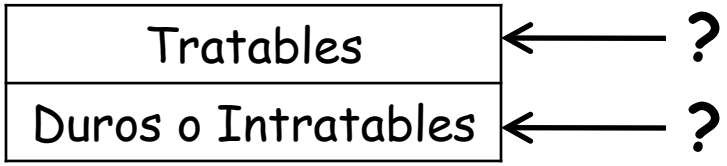
No computables
Fuertemente no computables

El **problema del MCD** es decidible tratable

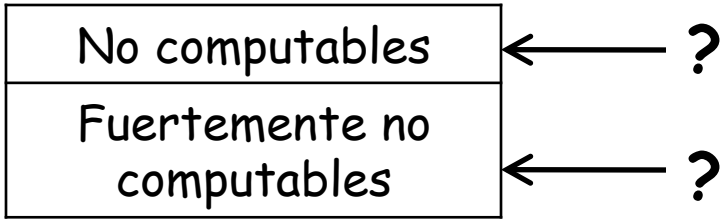
Máquinas de Turing

Clasificación de problemas

Problemas decidibles o resolubles



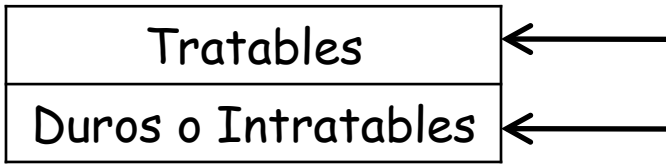
Problemas no decidibles o irresolubles



Máquinas de Turing

Clasificación de problemas

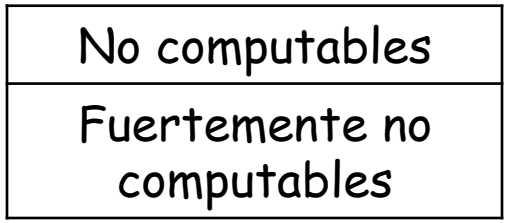
Problemas decidibles o resolubles



Problema del MCD

Problema de las torres de Hanoi

Problemas no decidibles o irresolubles



Problemas de la parada, de la teselación, de la serpiente

Problema de la verificación

Máquinas de Turing

Problemas resolubles

- Problemas P
- Problemas NP
- Problemas NP-completos

Máquinas de Turing

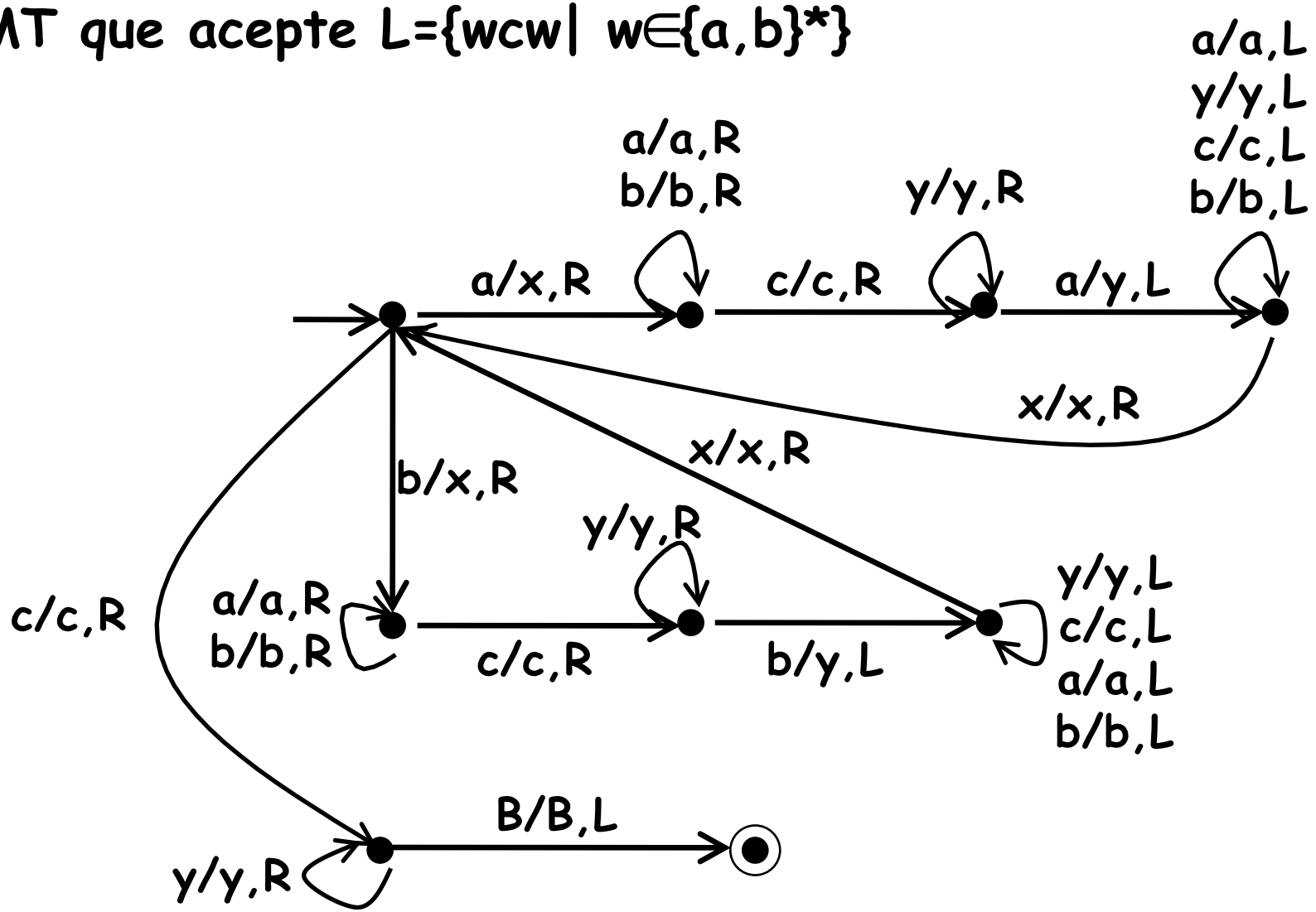
Problemas resolubles

- Problemas P
- Problemas NP
- Problemas NP-completos

Esta subclasificación de problemas reúne dos criterios, el determinismo de la máquina y el tiempo que toma alcanzar la solución

Máquinas de Turing

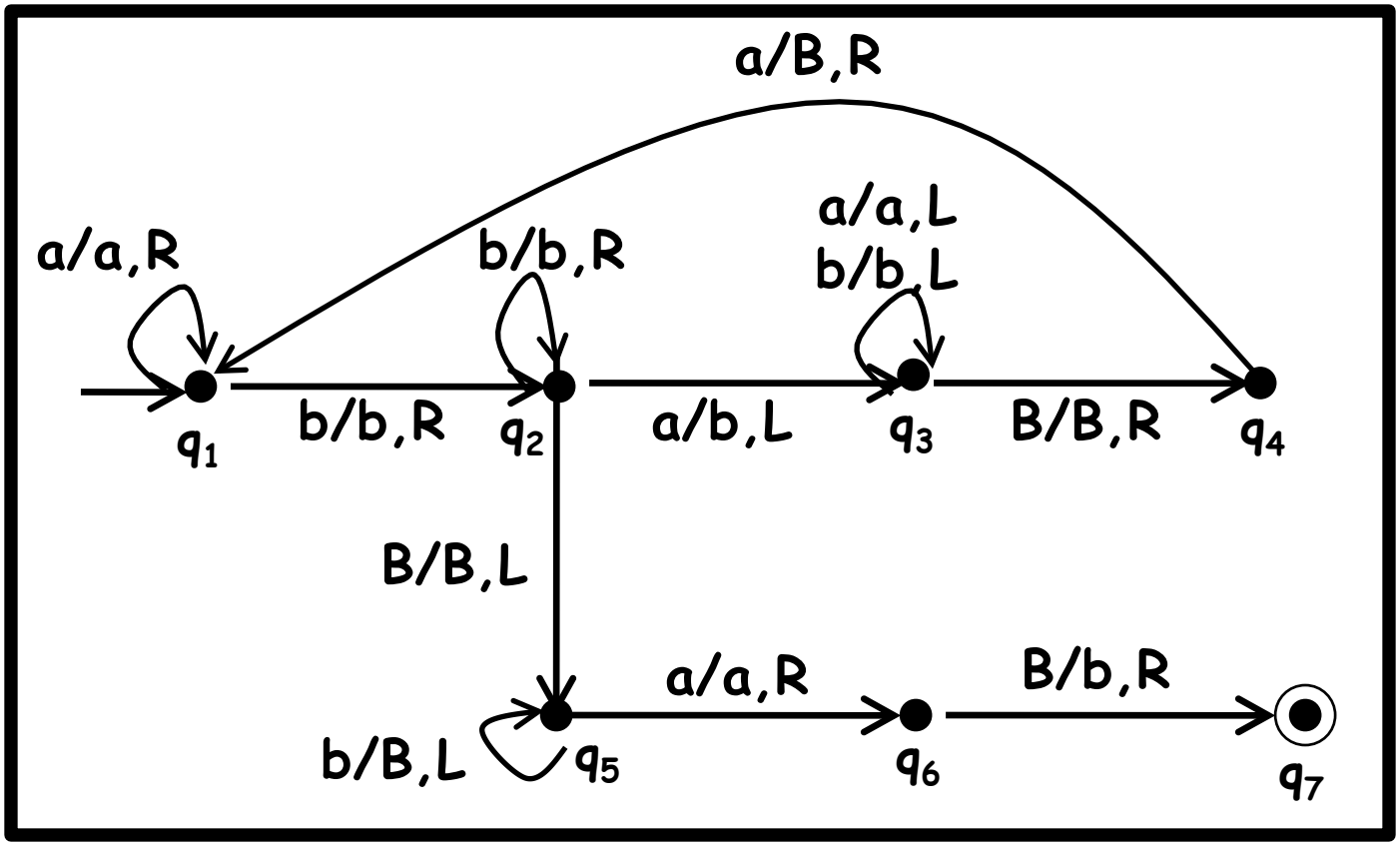
MT que acepte L={wcw| w∈{a,b}*}



Máquinas de Turing

- **Problemas P.** Aquellos problemas que pueden ser resueltos por medio de una MT determinista en tiempo polinomial

Máquinas de Turing

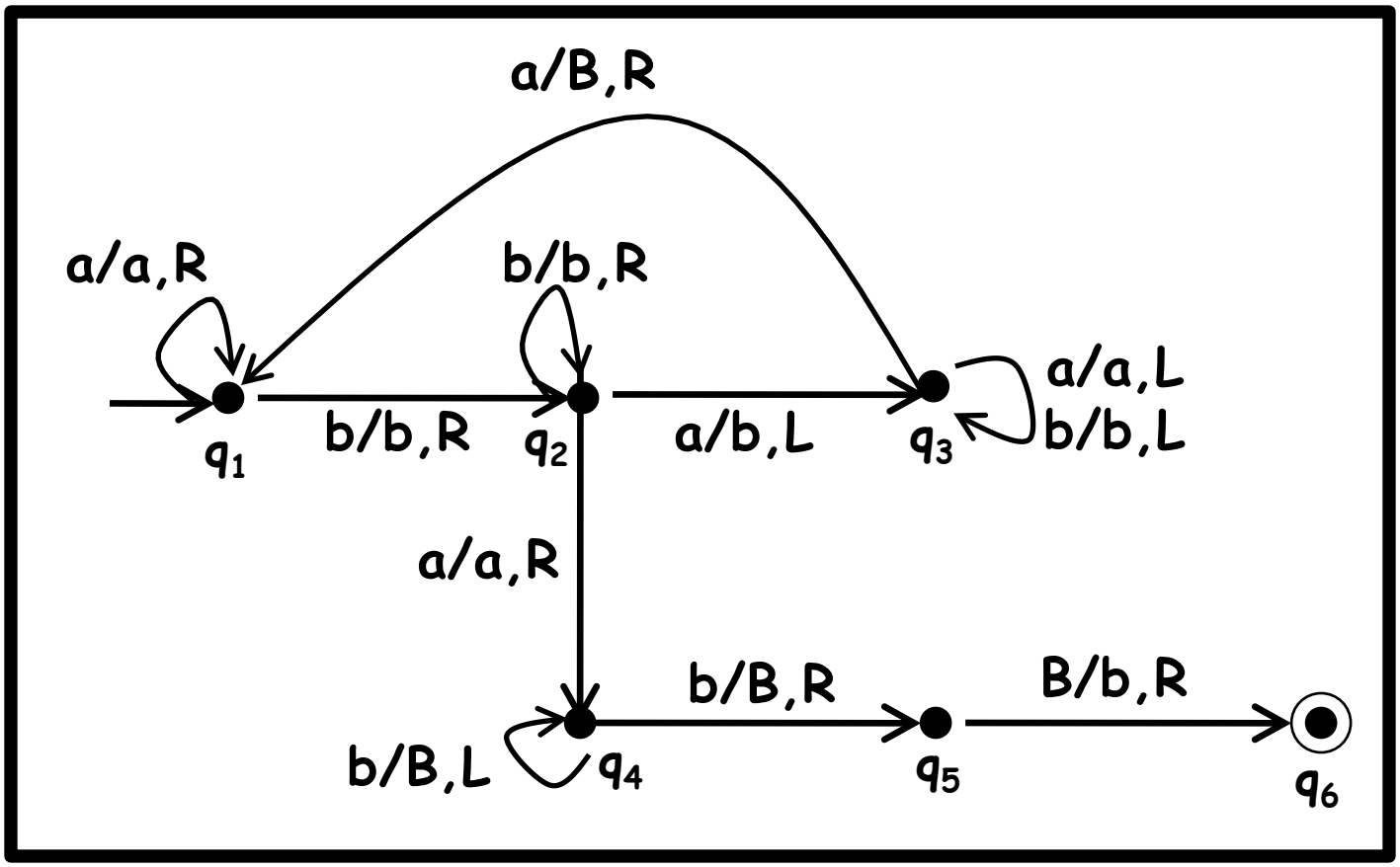


- Máquina determinista que resuelve el problema **f(a,b)=a-b** en tiempo polinomial
- El problema **f(a,b)=a-b** es **P**

Máquinas de Turing

- **Problemas NP.** *Aquellos problemas que sólo pueden ser resueltos por medio de una MT no determinista en tiempo polinomial (Non-deterministic Polynomial)*

Máquinas de Turing



Máquina no determinista

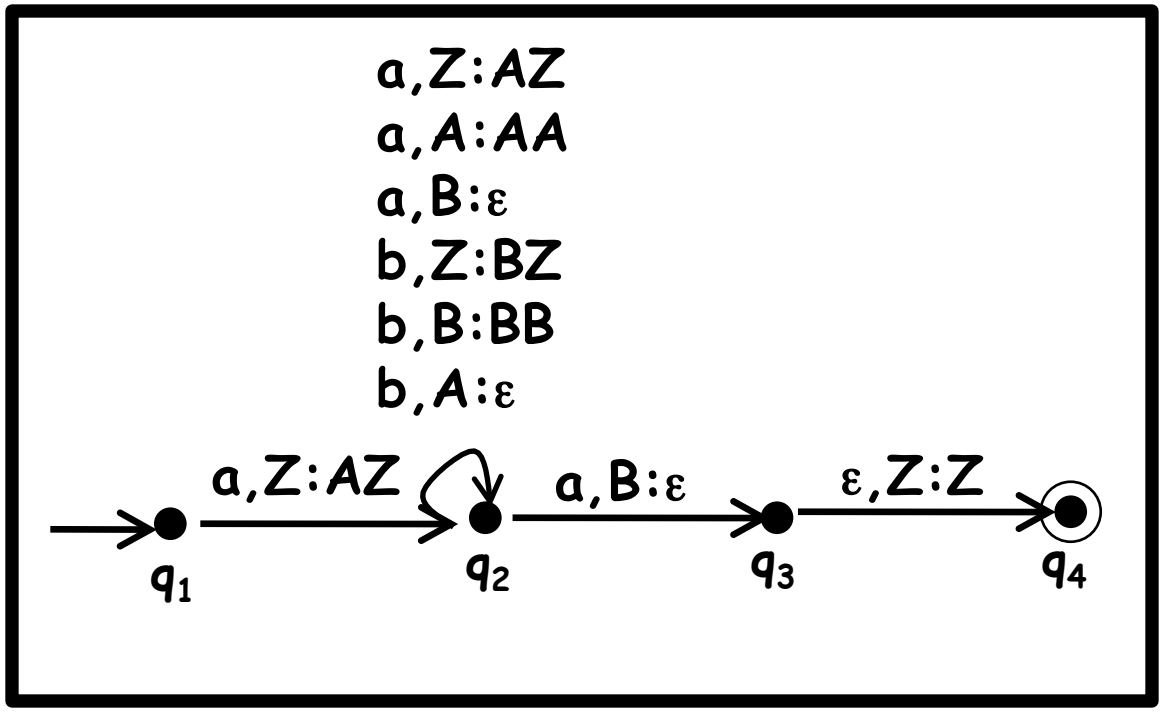
$(q_2,a)=\{(q_3,b,L),(q_4,a,R)\}$

$(q_4,b)=\{(q_4,b,L),(q_5,B,R)\}$

Máquinas de Turing

Autómata no determinista

L={w∈{a,b}*| w tiene la misma cantidad de a's que b's e inician y terminan en a}



Máquinas de Turing

Un problema es NP-completo si:

- Una solución al problema puede ser verificada rápidamente (en tiempo polinomial)
- Si se pudiera resolver el problema en tiempo polinomial, lo mismo ocurriría con los otros problemas NP-completos

Máquinas de Turing

Un problema es NP-completo si:

- Una solución al problema puede ser verificada rápidamente (en tiempo polinomial)
- Si se pudiera resolver el problema en tiempo polinomial, lo mismo ocurriría con los otros problemas NP-completos

Dado un problema NP-completo, se puede verificar una solución en tiempo polinomial, pero no se ha podido encontrar una solución rápida para el problema

Problemas NP-completos

Algunos de los más famosos son:

- Problema de satisfacibilidad booleana (SAT)
- Problema del agente viajero
- Problema del ciclo hamiltoniano
- Problema de la mochila (knapsack)

Máquinas de Turing

Stephen Cook

• Recibió el Premio Turing en 1982 por su descubrimiento:

"Por su avance en nuestra comprensión de la complejidad computacional de un modo significativo y profundo. Su artículo pionero, *The Complexity of Theorem Proving Procedures*, 1971, sentó los cimientos de la teoría de NP-completitud."

(1939 -)

Teorema de Cook

- El Teorema de Cook establece lo siguiente:

> El Problema de satisfactibilidad booleana (SAT) es NP-completo

Máquinas de Turing

- **El problema de satisfactibilidad booleana (SAT)**

Dada una expresión booleana saber si tiene asociada una asignación de valores de verdad que hacen que la expresión sea verdadera

Máquinas de Turing

- **El problema de satisfactibilidad booleana (SAT)**

Dada una expresión booleana saber si tiene asociada una asignación de valores de verdad que hacen que la expresión sea verdadera

$$(p \wedge \neg q) \rightarrow \neg s$$

Máquinas de Turing

- **El problema de satisfactibilidad booleana (SAT)**

Dada una expresión booleana saber si tiene asociada una asignación de valores de verdad que hacen que la expresión sea verdadera

$$(p \wedge \neg q) \rightarrow \neg s$$

¿Es `{p=True,q=False,s=False}` una solución?

Máquinas de Turing

- **El problema de satisfactibilidad booleana (SAT)**

Dada una expresión booleana saber si tiene asociada una asignación de valores de verdad que hacen que la expresión sea verdadera

$$(p \wedge \neg q) \rightarrow \neg s$$

La asignación `{p=True,q=False,s=False}` hace que la expresión sea verdadera

Máquinas de Turing

- **El problema de satisfactibilidad booleana (SAT)**

Dada una expresión booleana saber si tiene asociada una asignación de valores de verdad que hacen que la expresión sea verdadera

$$((p \wedge \neg q) \vee \neg s) \rightarrow (\neg p \wedge s)$$

¿Existe una asignación de valores de verdad que hacen la expresión verdadera?

Máquinas de Turing

- **El problema de satisfactibilidad booleana (SAT)**

Dada una expresión booleana saber si tiene asociada una asignación de valores de verdad que hacen que la expresión sea verdadera

$$((p \wedge \neg q) \vee \neg s) \rightarrow (\neg p \wedge s)$$

¿Existe una asignación de valores de verdad que hacen la expresión verdadera?

Dado un problema NP-completo, se puede verificar una solución en tiempo polinomial, pero no se ha podido encontrar una solución rápida para el problema

Máquinas de Turing

- **El problema de satisfactibilidad booleana (SAT)**

Dada una expresión booleana saber si tiene asociada una asignación de valores de verdad que hacen que la expresión sea verdadera

$$((p \wedge \neg q) \vee \neg s) \rightarrow (\neg p \wedge s)$$

¿Existe una asignación de valores de verdad que hacen la expresión verdadera?

No existe una mejor forma que evaluar todas las posibles asignaciones de valores de verdad. **Orden exponencial**

Problemas NP-completos

Algunos de los más famosos son:

- Problema de satisfacibilidad booleana (SAT)
- Problema del agente viajero
- Problema del ciclo hamiltoniano
- Problema de la mochila (knapsack)