



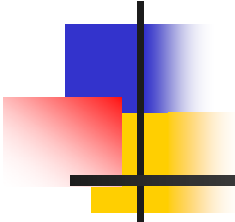
9. Problemas indecidibles

9.1. Lenguaje no recursivamente enumerable: L_d

Fernando Rosa Velardo

Traducción y adaptación de transparencias de Ananth Kalyanaraman
(<http://www.eecs.wsu.edu/~ananth/>)

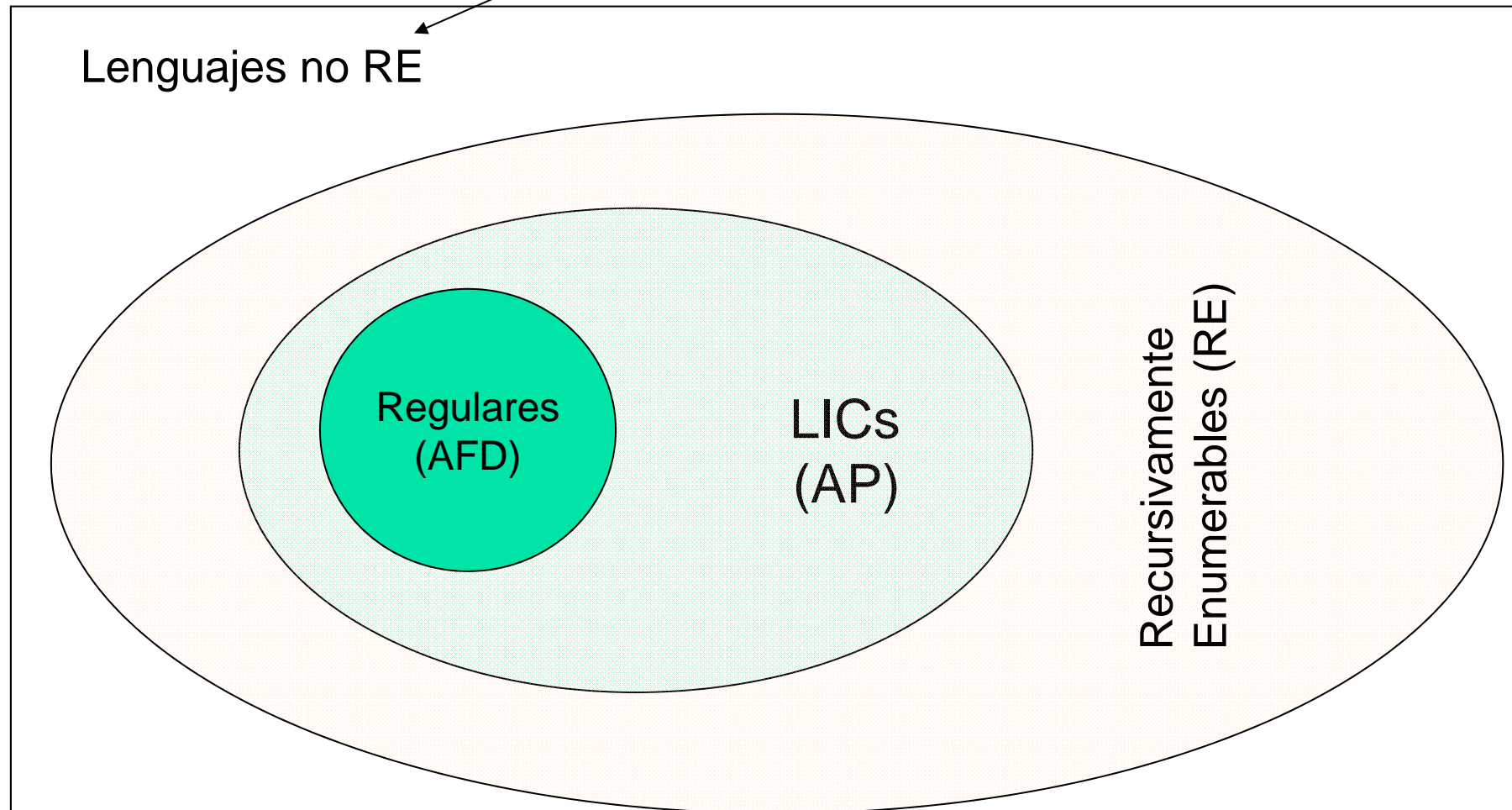
Necesariamente, algunos
lenguajes no son aceptados
por ninguna MT



¿!Pero no podían las MT resolverlo
todo!?

Lenguajes no RE

Necesariamente hay lenguajes aquí



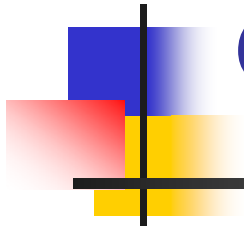


Una explicación:

Hay más lenguajes que MTs

- El conjunto de MTs es infinito numerable
- El conjunto de todos los lenguajes es infinito no numerable
- \Rightarrow Hay lenguajes sin MT (principio del palomar)

L_d : El lenguaje de diagonalización



Ejemplo de lenguaje
no recursivamente enumerable

(es decir, no aceptado por
ninguna MT)



Lenguaje acerca de MTs y su aceptación

- L_u = lenguaje de cadenas $\langle M, w \rangle$ tales que:
 1. M es (la codificación de) una MT
 2. w es una cadena binaria
 3. M acepta w



Enumeración de todas las cadenas binarias

- Sea w una cadena binaria
- Entonces $1w \equiv i$, donde i es un entero (en binario)
 - P.ej., si $w = \varepsilon$ entonces $i = 1$;
 - si $w = 0$ entonces $i = 2$;
 - si $w = 1$ entonces $i = 3$; y así...
- Si $1w \equiv i$ decimos que w es la i -ésima palabra binaria, y escribimos w_i para denotarla.
- \Rightarrow Enumeración de todas las cadenas binarias:
 - $\{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, \dots\}$
 - $\{w_1, w_2, w_3, w_4, \dots, w_i, \dots\}$



Cualquier MT puede ser codificada en binario

- $M = (Q, \{0,1\}, \Gamma, \delta, q_0, B, F)$
 - Le asignamos a cada estado, símbolo y dirección (L y R) un número.
 - Representamos $\delta(q_i, X_j) = (q_k, X_l, D_m)$ como:
 - $\Rightarrow 0^i 1 0^j 1 0^k 1 0^l 1 0^m$
- Resultado: podemos codificar cada MT como una larga cadena binaria
- \Rightarrow Enumeración de las MT:
 - $\{M_1, M_2, M_3, M_4, \dots M_i, \dots\}$

Lenguaje de diagonalización

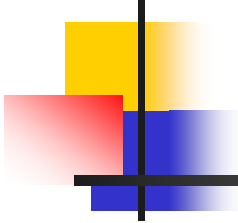
- $L_d = \{ w_i \mid w_i \notin L(M_i) \}$
 - Lenguaje de las cadenas w tales que la MT M con código w no acepta a w .

		j $\xrightarrow{\text{(entrada } w)}$				
		1	2	3	4	...
(MTs) i ↓	1	0	1	0	1	...
	2	1	1	0	0	...
	3	0	1	0	1	...
	4	1	0	0	1	...
	⋮					⋮

diagonal

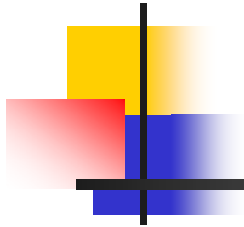
- Tabla: $T[i,j] = 1$, si M_i acepta w_j
 $= 0$, si no.

- Nuevo lenguaje:
 $L_d = \{ w_i \mid T[i,i] = 0 \}$

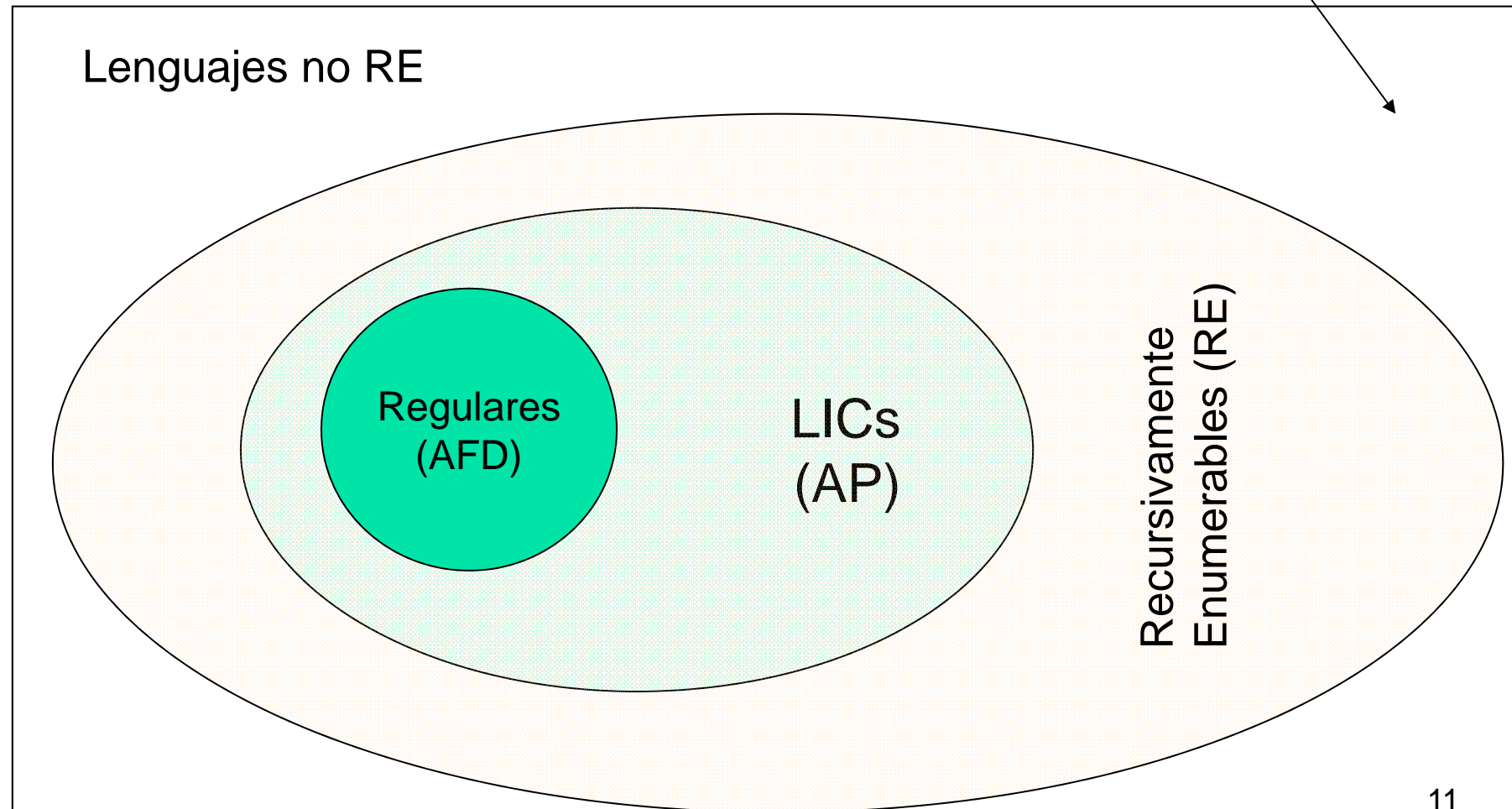


L_d no es RE (no es aceptado por ninguna MT)

- Demostración (reducción al absurdo):
- Sea M la MT que acepta L_d
- $\implies M$ ha de ser alguna M_k
- $\implies L(M_k) = L_d$
- \implies ¿Pertenece w_k a $L(M_k)$?
 1. Si $w_k \in L(M_k) \implies T[k,k]=1 \implies w_k \notin L_d$
 2. Si $w_k \notin L(M_k) \implies T[k,k]=0 \implies w_k \in L_d$
- ¡¡Contradicción en cualquier caso!!



Lenguaje de diagonalización





9. Problemas indecidibles

9.2. Lenguajes recursivos

Fernando Rosa Velardo

Traducción y adaptación de transparencias de Ananth Kalyanaraman
(<http://www.eecs.wsu.edu/~ananth/>)

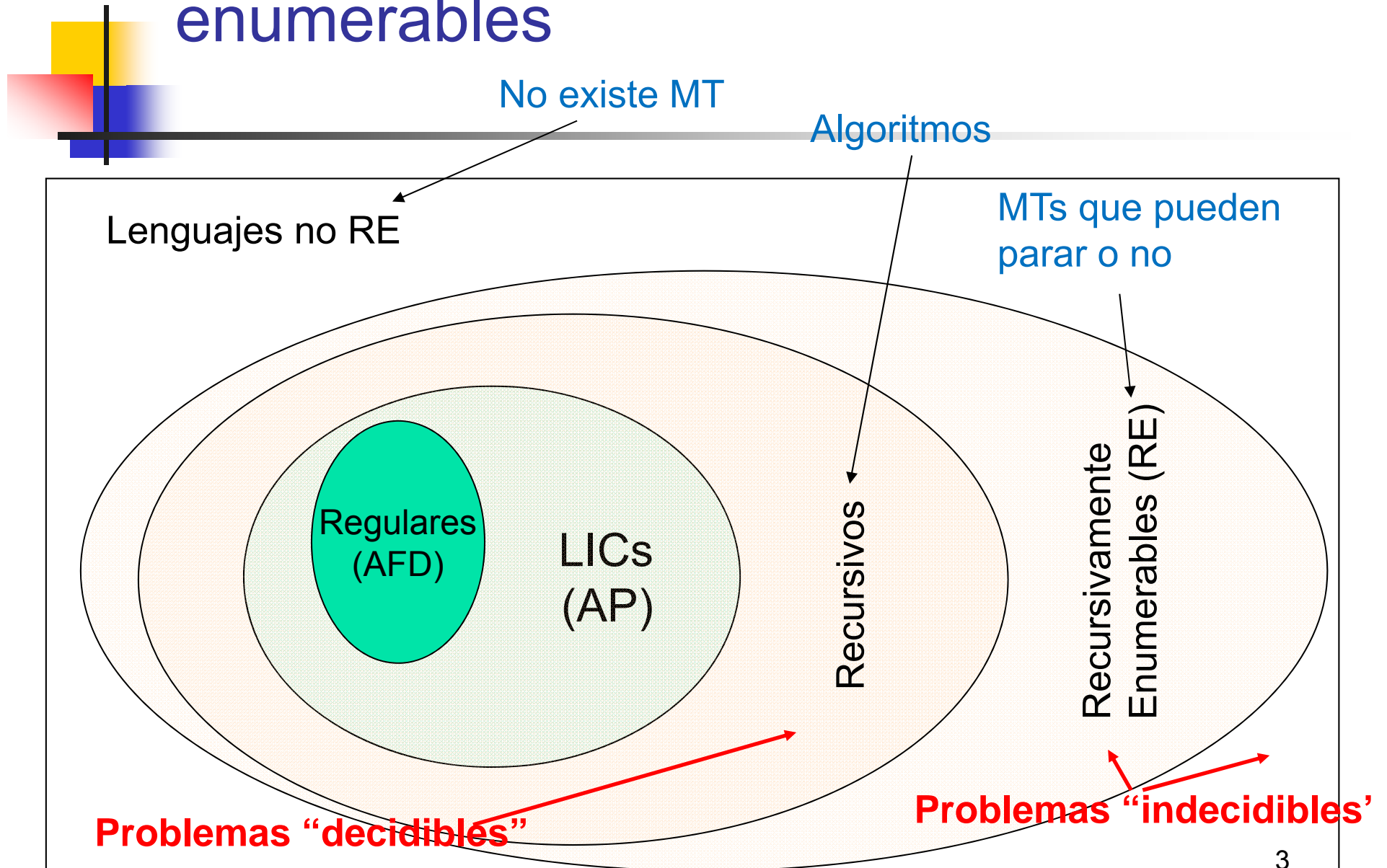
Decidibilidad e indecidibilidad

- Algoritmos: un algoritmo es una MT que *siempre* para, tanto si acepta como si no



- Lenguaje recursivo: lenguaje aceptado por un algoritmo
- **Problemas indecidibles:**
 - *Aquellos para los que no existe ningún algoritmo*

Lenguajes recursivos y recursivamente enumerables





“Lenguajes” y “Problemas”

Un “lenguaje” es un conjunto de cadenas

Cualquier “problema” puede ser expresado como un conjunto de cadenas de la forma:

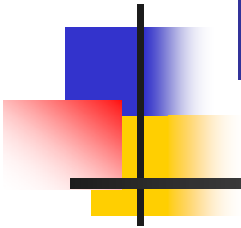
- “<input, output>”

P. ej., problema $(a+b) \equiv$ lenguaje de cadenas de la forma $\{ “a\#b, a+b” \}$

\Rightarrow ¡¡Los problemas son lenguajes!!

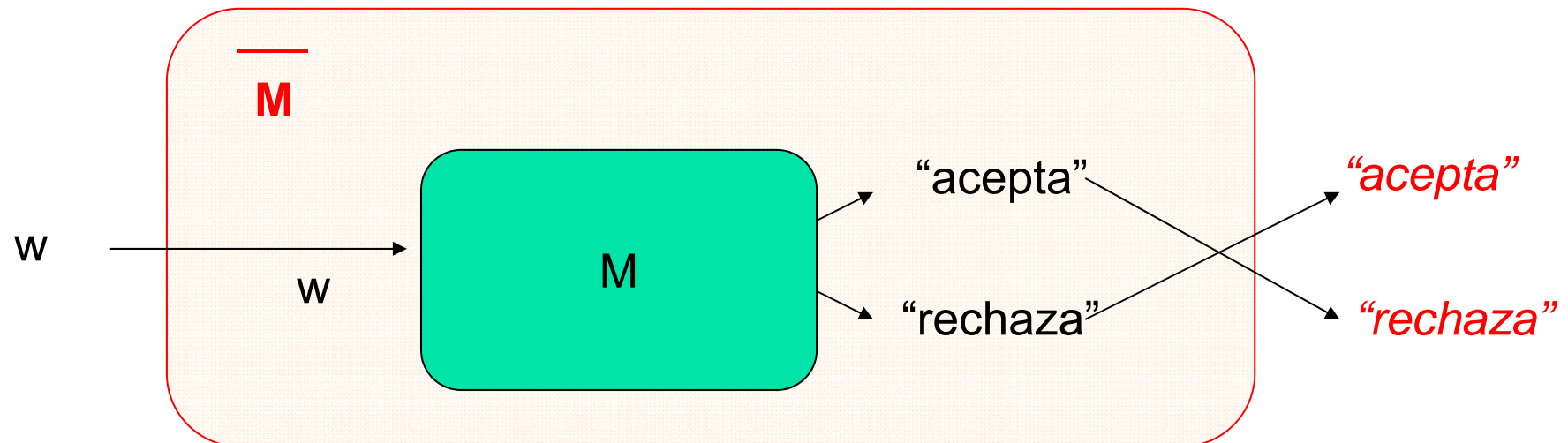
Piensa en el lenguaje de un “problema” \equiv *verificador* del problema

Propiedades de clausura de lenguajes recursivos y RE



Propiedades del complemento

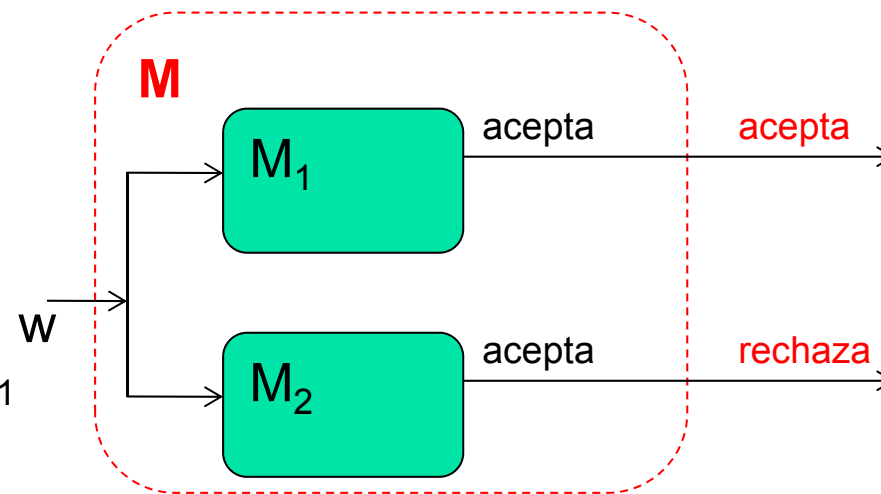
- Los lenguajes recursivos son cerrados para el complemento
 - Si L es recursivo, \overline{L} también lo es.



Propiedades del complemento

(II): L y \bar{L} RE \Rightarrow L recursivo

- Sean M_1 y M_2 MT para L y \bar{L}
- Construimos M_u de 2 cintas, algoritmo:
 1. Copiamos w en las dos cintas
 2. Simulamos *en paralelo* M_1 en la cinta 1 y M_2 en la cinta 2
 3. Si M_1 acepta entonces M para aceptando
 4. Si M_2 acepta entonces M para rechazando.





Otras propiedades de clausura

- Lenguajes recursivos cerrados para:
 - Unión e intersección (y el complemento)
 - Concatenación
 - Clausura de Kleene
- Lenguajes RE cerrados para:
 - Unión, intersección, concatenación, clausura de Kleene

- Lenguajes RE no cerrados para:
 - complemento

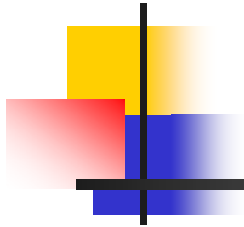


9. Problemas indecidibles

9.3. El Lenguaje Universal: L_u . Indecidibilidad de L_u

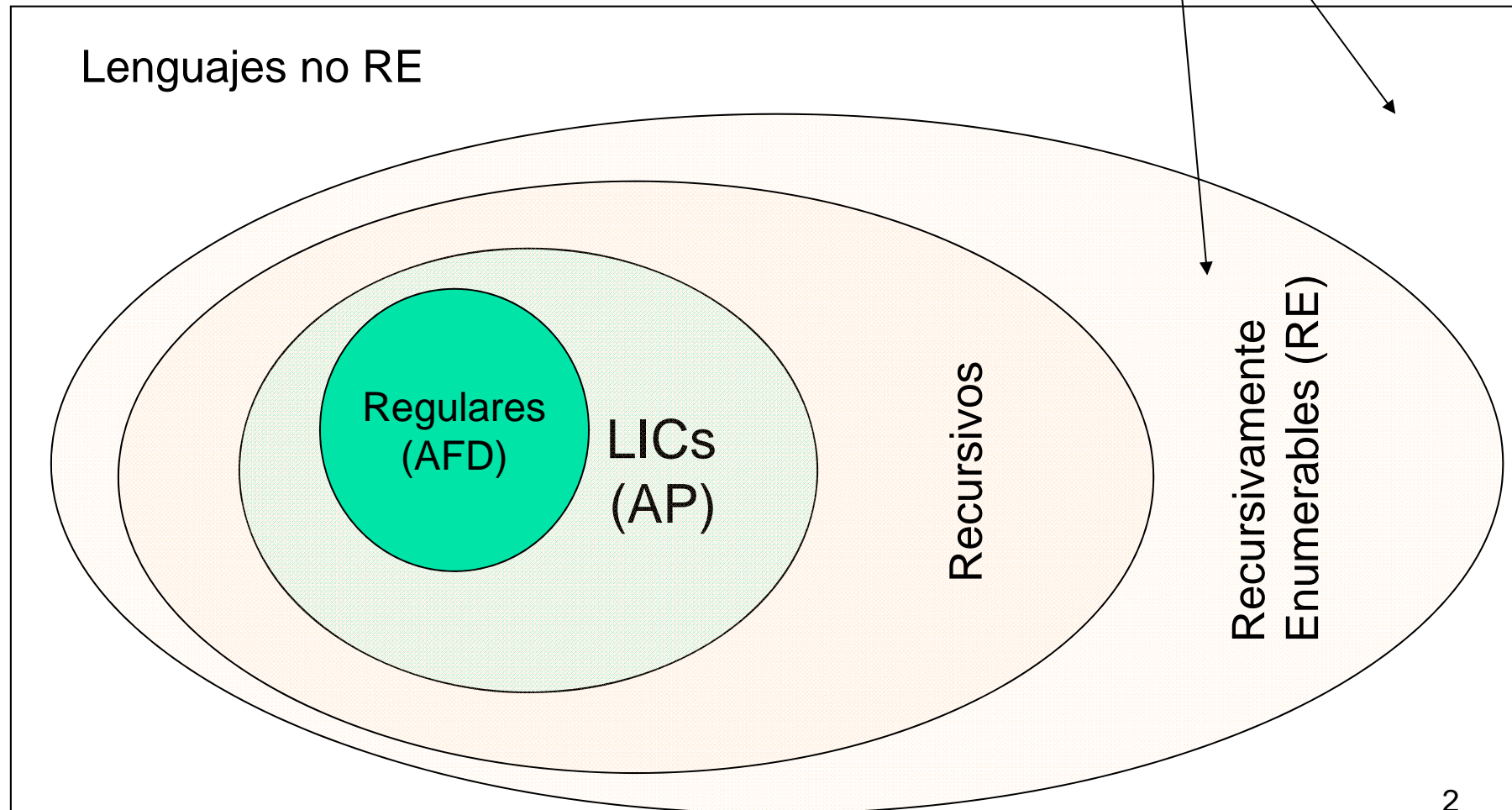
Fernando Rosa Velardo

Traducción y adaptación de transparencias de Ananth Kalyanaraman
(<http://www.eecs.wsu.edu/~ananth/>)



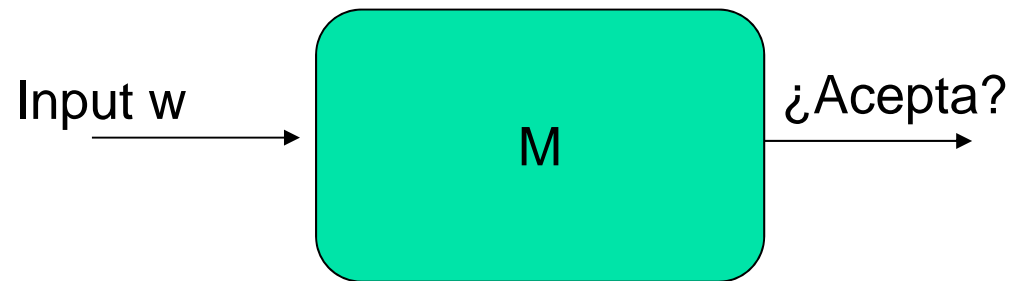
Lenguaje de diagonalización

Lenguaje universal



Lenguaje acerca de MTs y su aceptación

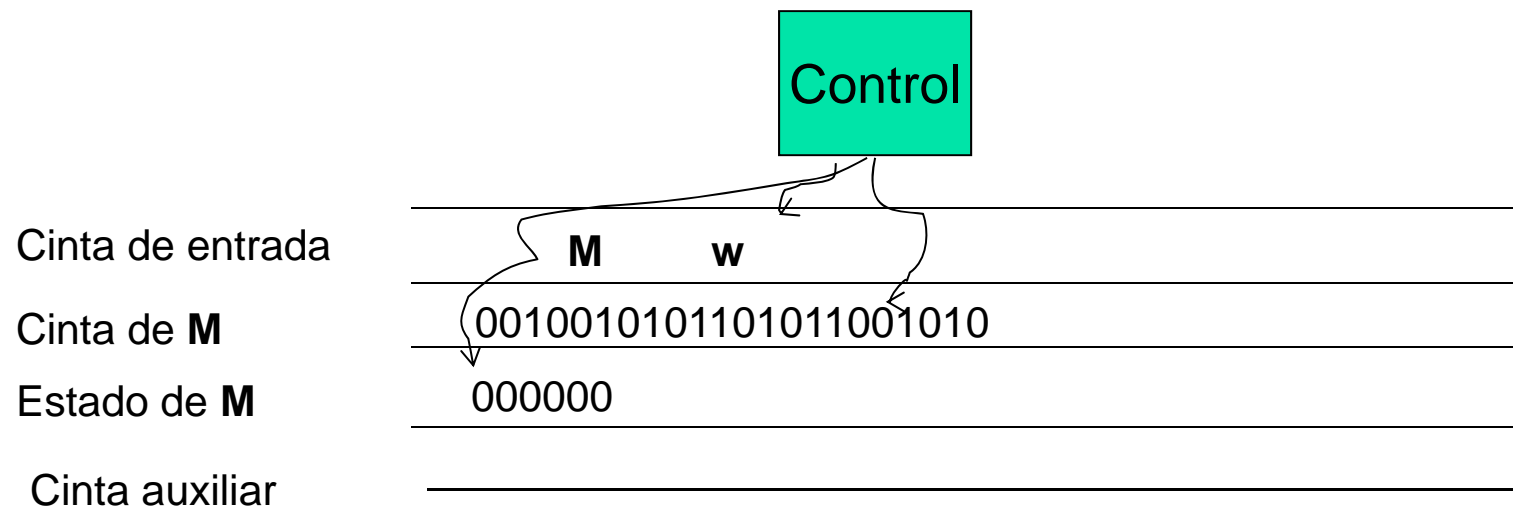
- L_u = lenguaje de cadenas $\langle M, w \rangle$ tales que:
 1. M es (la codificación de) una MT
 2. w es una cadena binaria
 3. M acepta w



L_u es recursivamente enumerable

“Intérprete” de máquinas de Turing

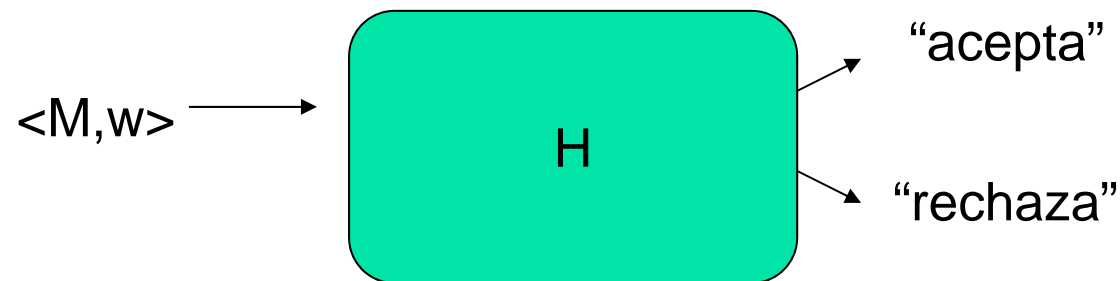
- Construimos U , MT Universal.
- Entrada: (código de) una MT M y w
- Ejecuta M al recibir w como entrada
- Si M acepta entonces U para aceptando





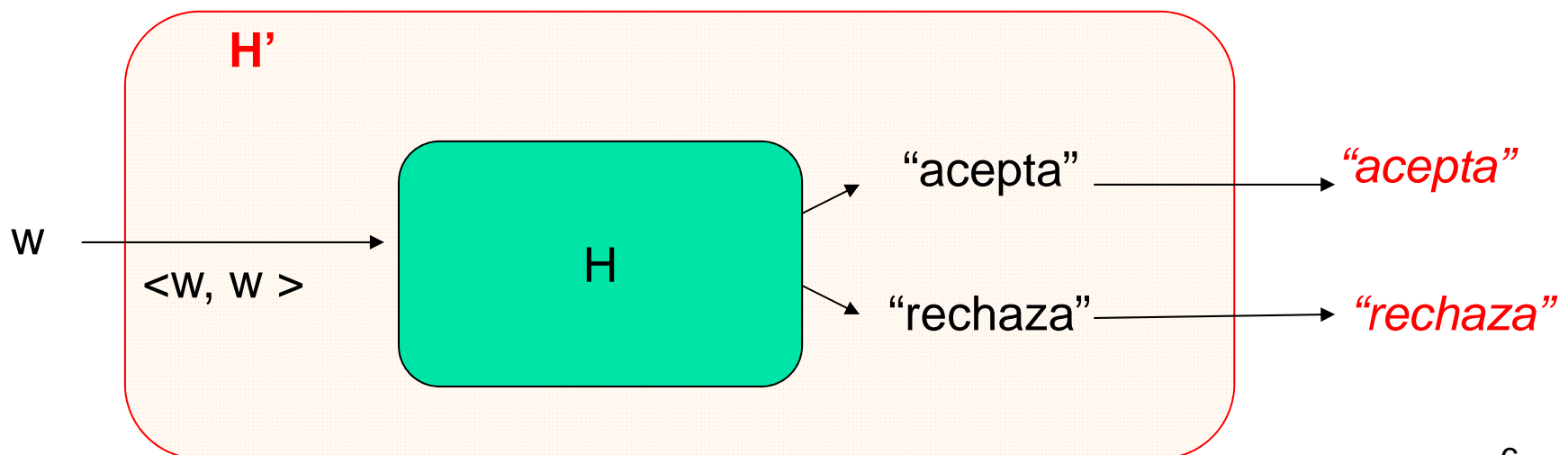
L_u no es recursivo

- Demostración (reducción al absurdo):
 - Supongamos que L_u es recursivo.
 - Entonces \bar{L}_u también lo es.
 - Existe H algoritmo que acepta \bar{L}_u
 - $H \text{ acepta } \langle M, w \rangle \Leftrightarrow M \text{ no acepta } w$



Demostración (cont.)

- Construimos a partir de H un algoritmo H' que acepta L_d (que sabemos que no puede existir):
 - Dada la entrada w , H' ejecuta H sobre $\langle w, w \rangle$;
 - (es decir, la MT correspondiente a w sobre w)
 - H' acepta $w \Leftrightarrow H$ acepta $\langle w, w \rangle \Leftrightarrow w$ pertenece a L_d





9. Problemas indecidibles

9.4. Otros problemas indecidibles

Fernando Rosa Velardo

Traducción y adaptación de transparencias de Ananth Kalyanaraman
(<http://www.eecs.wsu.edu/~ananth/>)



Lenguajes que conocemos de momento...

- *Lenguaje Universal*

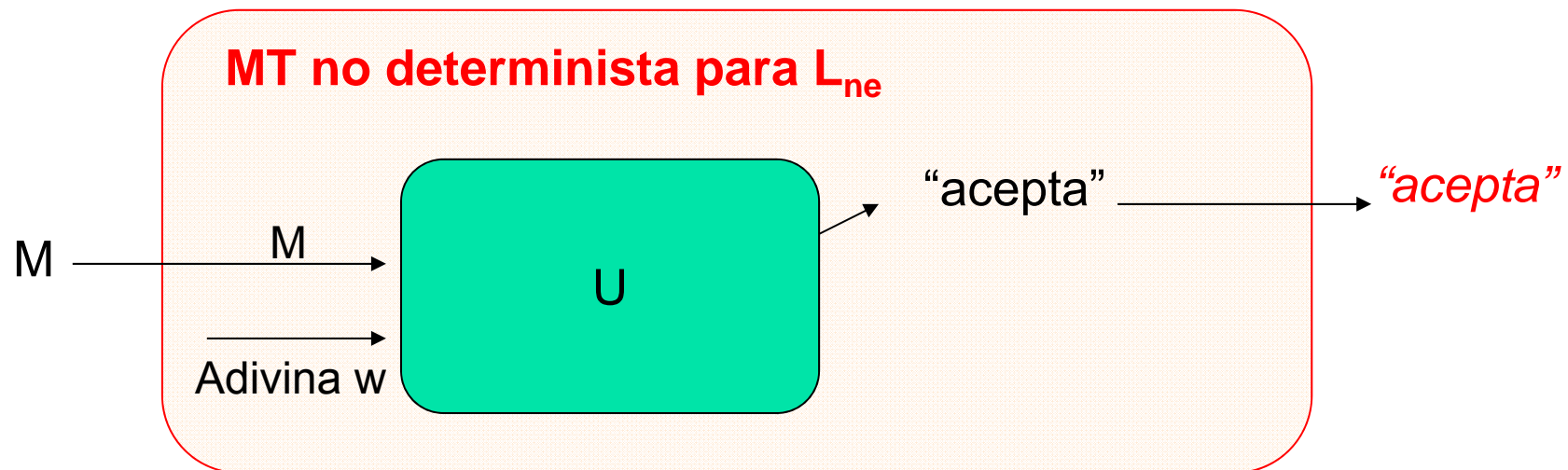
- $L_u = \{ \langle M, w \rangle \mid M \text{ acepta } w \}$
- Resultado: L_u es RE pero no recursivo

- *Lenguaje de diagonalización*

- $L_d = \{ w_i \mid M_i \text{ no acepta } w_i \}$
- Resultado: L_d no es RE

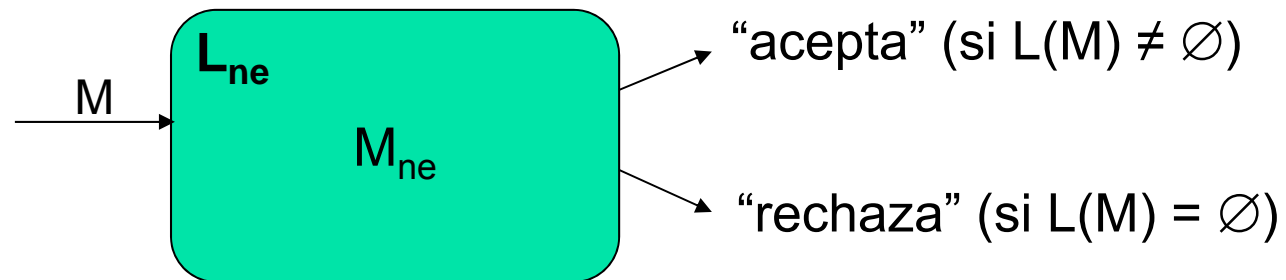
MTs que aceptan lenguajes no vacíos

- $L_{ne} = \{ M \mid L(M) \neq \emptyset \}$
- L_{ne} es RE
- Dem: (construimos MT para L_{ne} usando U)



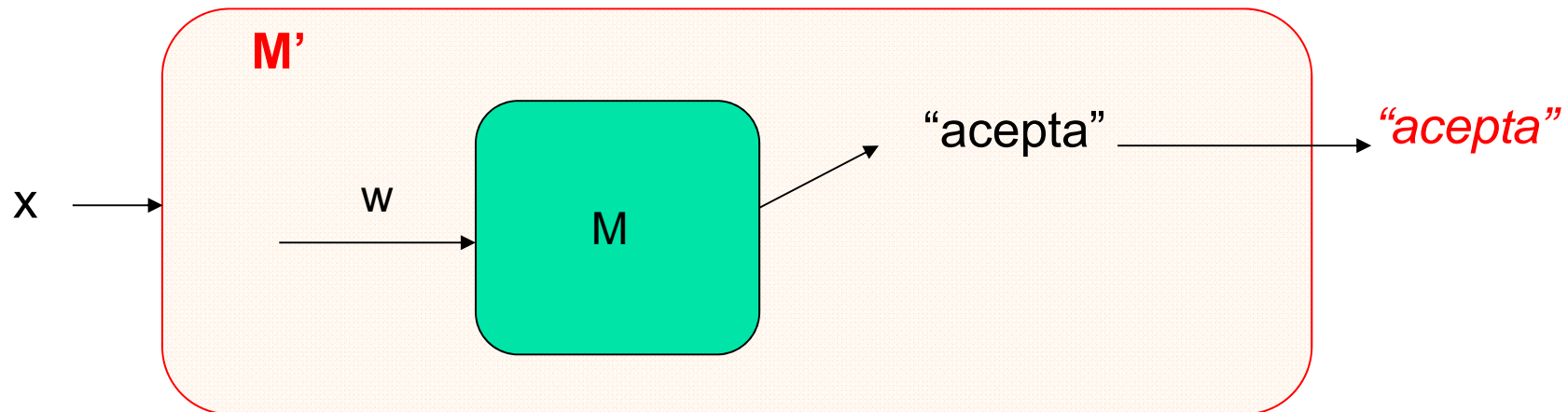
MTs que aceptan lenguajes no vacíos

- L_{ne} es no recursivo
- Dem: Supongamos que L_{ne} es recursivo. Entonces existe un algoritmo M_{ne} que lo acepta



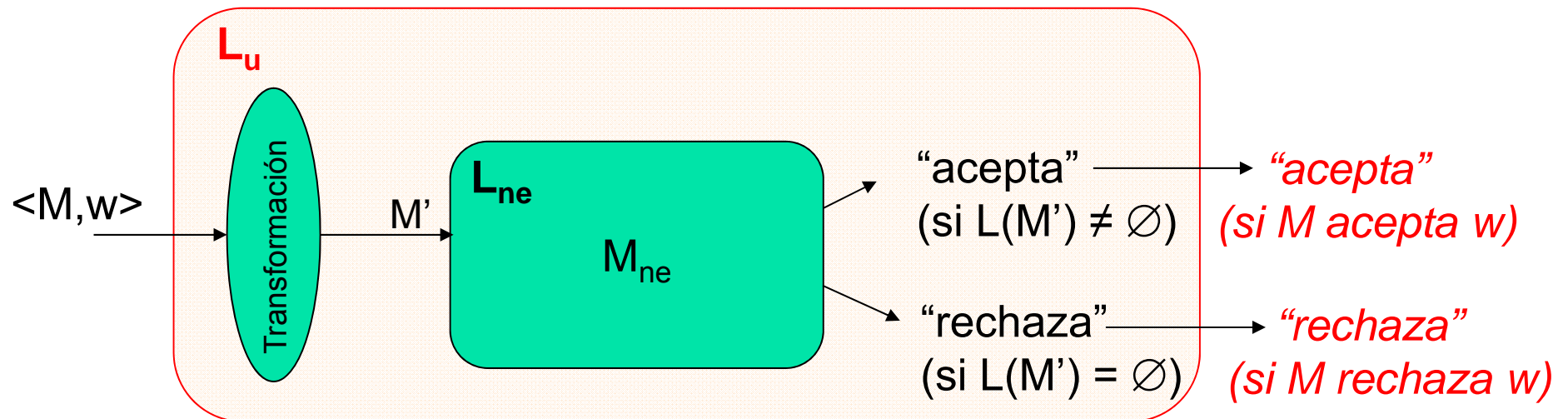
L_{ne} es no recursivo (cont.)

- Dem: (cont.) “*Reducimos*” L_u a L_{ne}
 - Idea: Transformamos $\langle M, w \rangle$ en M' tal que
 $M \text{ acepta } w \Leftrightarrow L(M') \neq \emptyset$



L_{ne} es no recursivo (cont.)

- Dem: (cont.) Usando M_{ne} y la transformación anterior construimos un algoritmo para L_u (¡¡contradicción!!)





L_{ne} y L_e

- $L_{ne} = \{ M \mid L(M) \neq \emptyset \}$
 - Resultado: L_{ne} es RE pero no recursivo
- $L_e = \{ M \mid L(M) = \emptyset \}$
 - Resultado: L_e no es RE



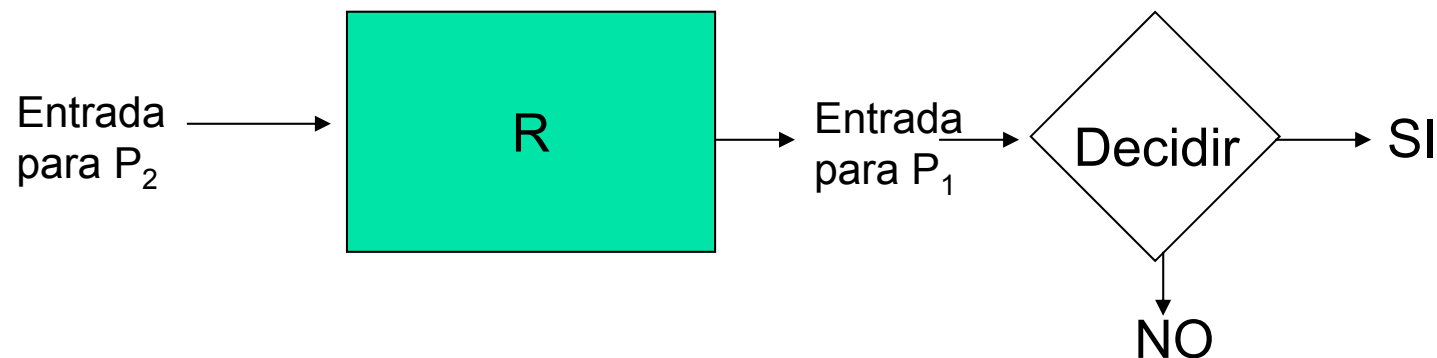
Reducciones

- Para probar: que el problema P_1 es indecidible
- Sabiendo: que el problema P_2 es indecidible
- Idea:
 1. “Reducir” P_2 a P_1 :
 - Convertimos (algorítmicamente) una entrada de P_2 a una entrada de P_1 tal que
 - i) P_2 acepta si y sólo si P_1 acepta
 2. Por lo tanto, si P_1 es decidable P_2 es decidable
 3. Contradicción
 4. Así que P_1 ha de ser indecidible

Reducciones

R = Reducción de P_2 a P_1 :

Obs: distinto
que reducir
 P_1 a P_2



Conclusión: P_2 indecidible $\Rightarrow P_1$ indecidible
 P_2 no RE $\Rightarrow P_1$ no RE
(P_1 es "más difícil" que P_2)