Relacion-2-MAC.pdf

patriciacorhid

Modelos Avanzados de Computacion

4º Doble Grado en Ingeniería Informática y Matemáticas

Facultad de Ciencias
Universidad de Granada

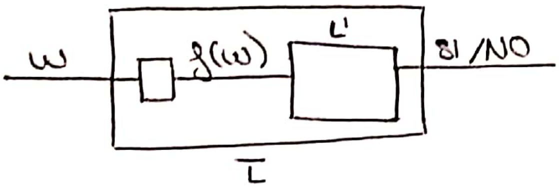
Patricia Córdoba Hidalgo

Relación 2

① L lenguaje r.e. pero no recursivo. Consideremos $L' = \{0w \mid w \in L\} \cup \{1w \mid w \notin L\}$. ¿$L'$ y su complementario son recursivos, r.e o no r.e?

$\overline{L}$ no puede ser recursivamente enumerable, ya que si no L sería recursivo, luego $\overline{L}$ es no recursivamente numerable.

Veamos que L' es no recursivamente numerable reduciendo $\overline{L}$ a L'.



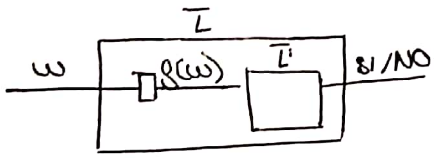
Dada una palabra $w \in A^*$, $f(w) = 1w$. f es una función totalmente calculable, ya que existe una MT que la calcula y siempre para.

- Si $w \in \overline{L}$, es decir, $w \notin L \Rightarrow f(w) = 1w \in L'$
- Si $w \notin \overline{L}$, es decir, $w \in L \Rightarrow f(w) = 1w \notin L'$

Por tanto $\overline{L}$ se reduce a L'. Como $\overline{L}$ es no recursivamente enumerable, L' es no recursivamente enumerable.

$\overline{L'} = \{0w \mid w \notin L\} \cup \{1w \mid w \in L\} \cup \{$palabras que empiezan por caracteres diferentes a 0 o 1$\}$.

Veamos que $\overline{L'}$ no es recursivamente numerable reduciendo $\overline{L}$ a $\overline{L'}$.



Dada una palabra $w \in A^*$, $f(w) = 0w$. f es una función totalmente calculable.

- Si $w \in \overline{L}$, es decir, $w \notin L \Rightarrow f(w) = 0w \in \overline{L'}$
- Si $w \notin \overline{L}$, es decir, $w \in L \Rightarrow f(w) = 0w \notin \overline{L'}$

Por tanto $\overline{L}$ se reduce a $\overline{L'}$. Como $\overline{L}$ es no recursivamente enumerable, $\overline{L'}$ no es recursivamente enumerable.

Por tanto L' y $\overline{L'}$ son no r.e.

② Determinar si los siguientes problemas son decidibles, semidecidibles o indecidibles.

a) Dada una MT determinar si acepta, al menos, dos palabras distintas.

Aceptar al menos dos palabras distintas es equivalente a decir que $L(M)$ está formado por al menos dos palabras.

"Estar formado por al menos dos palabras" es una propiedad no trivial del lenguaje, ya que el lenguaje vacío no tiene más de dos y el formado por todas las palabras del alfabeto sí.

Por tanto, por el Teorema de Rice, es no decidible.

② a) Es semidecidible:

Dada una entrada M, una MT, elijo dos palabras, w_1 y w_2, de forma no determinista.

Si M acepta las dos ⇒ acepto

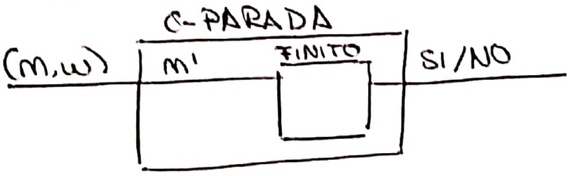
En caso contrario ⇒ rechazo

Como acepta en los casos positivos, es semidecidible.

b) Dada una MT, determinar si acepta un nº finito de palabras.

Reducimos el problema complementario a la parada a éste.

Como el problema de la parada es semidecidible pero no decidible ⇒ el complementario es no semidecidible.



Dada una entrada (M,w) del problema complementario a la parada, construimos $f(M) = M'$ una MT de forma que:

Dada una entrada x, M' ejecuta los |x| primeros pasos de M con w.

Si M para con w en |x| pasos ⇒ acepta x

Si M no para con w en |x| pasos ⇒ rechaza x

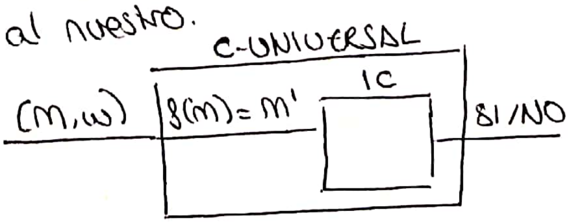
De esta manera se tiene que:

- Si (M,w) es una entrada positiva de C-PARADA, es decir, si M no para con w, $\mathcal{L}(M') = \emptyset$, que es finito.
- Si (M,w) es una entrada negativa de C-PARADA, es decir, si M para con w en N pasos, M' acepta todas las palabras de longitud mayor o igual que N, que es un nº infinito de palabras.

Como C-PARADA es no semidecidible, nuestro problema es no semidecidible.

c) Dada una MT, determinar si el lenguaje aceptado es independiente del contexto.

Reducimos el problema C-UNIVERSAL, que es no semidecidible, al nuestro.



Dada (M,w) una entrada del problema C-UNIVERSAL construimos $g(M) = M'$ una máquina de Turing.

② c) M' tiene como entrada palabras de la forma $0^n 1^n 0^n$.
Si M acepta w en n pasos ⇒ M' acepta $0^n 1^n 0^n$
Si M no acepta w en n pasos ⇒ M' rechaza $0^n 1^n 0^n$

Así:

- Si (M, w) es una entrada positiva de C-UNIVERSAL, es decir, M no acepta w ⇒ $L(M') = \emptyset$ que es un lenguaje independiente del contexto.
- Si (M, w) es una entrada negativa de C-UNIVERSAL, es decir, M acepta w en N pasos ⇒ M' acepta las palabras de la forma $0^n 1^n 0^n$ $\forall n \geq N$ que no es un lenguaje independiente del contexto.

Como C-UNIVERSAL es no semidecidible, determinar si un lenguaje es independiente del contexto es no semidecidible.

d) Dada una MT, saber si para una entrada 0011 no va a usar más de 10 casillas de la cinta.

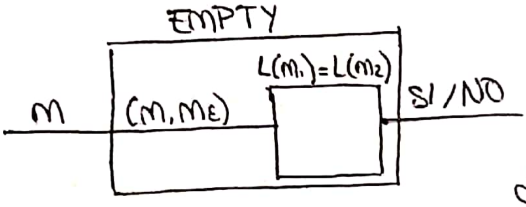
Las configuraciones de una MT son de la forma (q, u, w) con $q \in Q$, $u, w \in B^*$. Como no vamos a usar más de 10 casillas de la cinta en una entrada positiva del problema $|u| + |w| \leq 10$. Acotamos burdamente: $|u| \leq 10$, $|w| \leq 10$. El conjunto de posibilidades es n^{20} donde n es el nº de símbolos de B, un nº finito. Por tanto, si m es el nº de estados de la máquina de turing, en menos de $m \cdot n^{20}$ configuraciones se repetirá alguna de la obtenida anteriormente, por lo que si en $m \cdot n^{20}$ configuraciones no se ha salido de las 10 casillas ejecutando la entrada 0011, ya no lo hará. Este problema es decidible, y la MT que lo resuelve es:

Dada una entrada M, ejecutamos 0011 en M:

- Si en $m \cdot n^{20}$ pasos no se ha salido de las 10 casillas ⇒ acepto.
- Si se sale de las 10 casillas ⇒ rechazo.

e) Dadas dos MT's, saber si aceptan el mismo lenguaje.

Reducimos el problema de determinar si $L(M)=\emptyset$ a este.



Dada una entrada del problema EMPTY M, una MT, construimos la pareja (M, M_ε), donde M_ε es una máquina de Turing sin transiciones, es decir,

$L(M_\varepsilon)=\emptyset$.

- Si $L(M)=\emptyset \Rightarrow L(M)=L(M_\varepsilon)$
- Si $L(M)\neq\emptyset \Rightarrow L(M)\neq L(M_\varepsilon)$

Luego hemos reducido el problema EMPTY al nuestro. Como el problema EMPTY es no semidecidible, el nuestro también lo es.

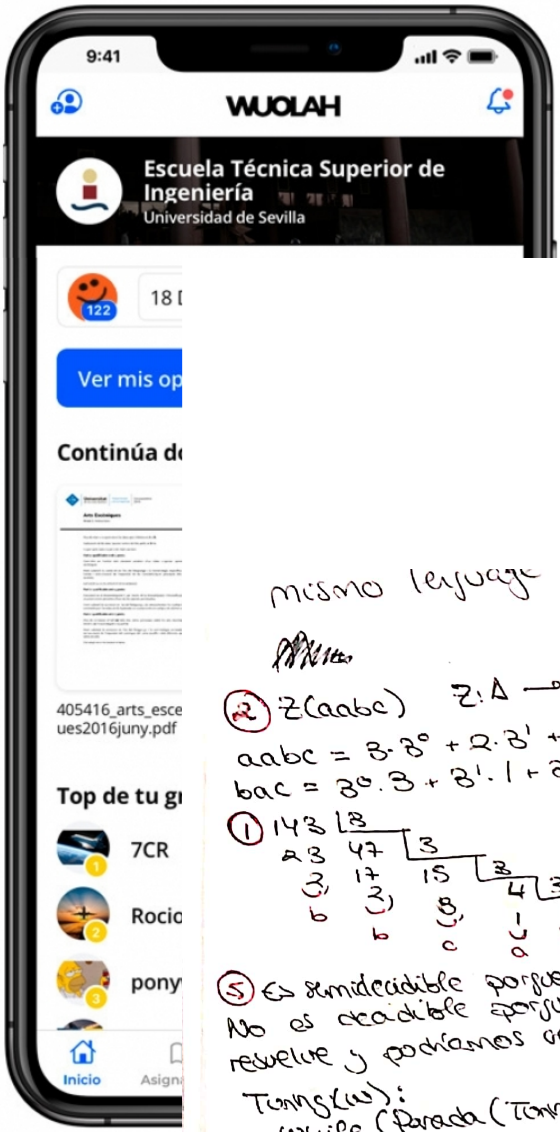
g) Dada una MT M y una palabra u, saber si la MT acepta la palabra u en un nº de pasos menor o igual a $|u|$

Dada la entrada (M,u) ejecutamos los primeros $|u|$ pasos de M con la entrada u:

Si M acepta u en esos pasos $\Rightarrow$ acepto

Si M no acepta u en esos pasos $\Rightarrow$ rechazo.

Por tanto, es decidible.



mismo lenguaje

Rel 2

(2) $Z(aabc)$ $\quad Z: \Delta \rightarrow \mathbb{N}$ $\quad a \rightarrow 1 \quad b \rightarrow 2 \quad c \rightarrow 3$

$1\cdot 3^0 + 1\cdot 3^1 + 1\cdot 3^2 + 1\cdot 3^3 = 3 + 6 + 9 + 27 = 45.$

$aabc = 3\cdot 3^0 + 2\cdot 3^1 + 1\cdot 3^2 + \ldots$

$bac = 3^0\cdot 3 + 3^1\cdot 1 + 3^2\cdot 2 = 3 + 3 + 18 = 24.$

(1) $143 \,|\, 3$: $143 = 3\cdot 47 + 2$ → b; $47 = 3\cdot 15 + 2$ → b; $15 = 3\cdot 4 + 3$ → c; $4 = 3\cdot 1 + 1$ → a; 1 → a

aacbb

$100 \,|\, 3$: $100 = 3\cdot 33 + 1$ → a; $33 = 3\cdot 10 + 3$ → c; $10 = 3\cdot 3 + 1$ → a; 3 → c

caca

(5) Es semidecidible porque existe un algoritmo que termina con los casos positivos. No es decidible porque sino habría un programa, llamémosle parada, que lo resuelve y podríamos crear:

Turing(w):
    while (Parada(Turing, w)) {}

y si llamamos a Turing(Turing) tendríamos un programa que cicla cuando no cicla y no cicla cuando cicla.

(6). a) A partir de la suma implementamos la multiplicación y vamos multiplicando un nº (dividir por todos los nº menores que él) que determine si por sí mismo.

b) Hacer una subrutina determinista que determine si es primo y si lo es, se copia.

(7) El complementario de L_i es $\bigcup_{j=1, j\neq i}^{k} L_j$

La unión ~~de lenguajes~~ finita de lenguajes recursivamente enumerables es recursivamente enumerable, ya que haciendo una MT que ejecute cada máquina de las otras en una cinta y acepta si alguna de ellas acepta.

Como L_i y $\bigcup_{j=1, j\neq i}^{k} L_j$ son r.e $\Rightarrow L_i$ es recursivo.

OPCIÓN 2:

Construyo una máquina de Turing que ejecute la de cada lenguaje en una cinta. en paralelo

Siempre para porque la unión es Δ^* y acepta si lo acepta el L_i y rechaza si lo acepta L_j con $j\neq i$.

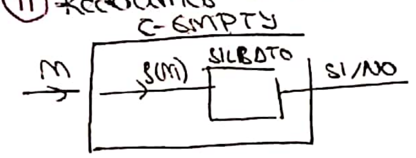
⑧ $\overline{L}$ no es r.e. (si no, L sería ~~decidible~~ recursivo)
Para ver que L' no es r.e, reducimos $\overline{L}$ a L'. Si existiese una MT que ~~acepta~~ acepta L', podemos construir una que acepta $\overline{L}$ (le pega un Δ al inicio) ABSURDO.
Luego L' no es r.e.
$\overline{L'} = \{0w \mid w \notin L\} \cup \{1w \mid w \in L\} \cup \{resto\}$. Tanto L' como $\overline{L'}$ son r.e.
Hago la reducción añadiendo un 0 a la palabra de $\overline{L}$.

⑨

	Unión	Intersección	Concatenación	Clausura	Homomorf.	Homomorf. inverso
recursivos	$\cancel{\checkmark}$	$\cancel{\checkmark}$	✓ (1)	✓ (2)	✓ (3)	✓ (4)
recursiv. enum	✓	(no numerable)	✓	✓	✓ (3)	✓ (1)

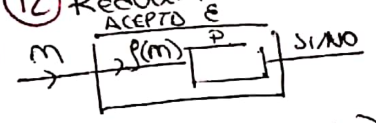
(1) Pones separador de forma no determinista
(2) Como concatenación, pero infinita, de un mismo lenguaje consigo mismo.
(3) Tomo w de forma no determinista. Si pertenece a L, ~~nnn~~ calculo su imagen y veo si coincide con mi palabra.
(4) $g(w)$ hace que pertenecer a L (se comprueba).

⑪ Reducimos C-EMPTY al problema de ver si la MT entrará en estado silbato.



Nos pasan como entrada M una MT y queremos ver si M acepta alguna palabra. $f(M)$ es una MT donde cambiamos los estados finales de M por estados silbato y los no finales por ~~estados~~ campana. Esa se la pasemos a la MT que resuelve el problema del silbato y devuelve sí si hay una palabra ~~para la que toca~~ aceptada por M y no en caso contrario.
Como C-EMPTY es semidecidible y no decidible ⇒ el problema es no decidible.

⑫ Reducimos el problema de que un lenguaje acepte la palabra vacía a éste:

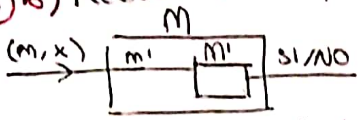


Nos pasan una MT, M, a $f(M)$ es cambiar los 1 del alfabeto de M por 2 (para que no tengan las palabras que los estados finales escriban un 1 el Δ) y luego hacer que los estados finales escriban un 1 como último símbolo al pasarle la palabra
M acepta ε si $f(M)$ escribe un 1 y no lo acepta en caso contrario
Como ACEPTAR ε se reduce a nuestro problema y aceptar ε es una propiedad no trivial del lenguaje ⇒ es no decidible ⇒ nuestro problema es no decidible

⑬ a) No decidible (⇐ T^{ma} Rice.) ~~semidecidible~~: Cojo de forma no determinista 2 palabras. Si acepta las 2 FIN.
semidecidible: Cojo (⇐ T^{ma} Rice) ¿?
b) No decidible (⇐ T^{ma} Rice)
c) Por T^{ma} Rice, es no decidible.

⑭ - Semidecidible:
elijo de forma no determinista k palabras y acepto si M_1 y M_2 las aceptan.
· No decidible:
El problema $L(A) \cap L(B) = \emptyset$ es indecidible si A y B son gramáticas libres de contexto. Como una gramática libre de contexto puedo pasarla a un autómata pi y un autómata con pila la puedo pasar a MT, si hubiese sol. para mi ~~habría sol para~~ le paso (M_A, M_B, Δ) y me da sol de problema inicial, $L(A) \cap L(B) = \emptyset$.

(13) b) Reducimos el problema complementario a la parada a éste



M: Devuelve Sí si m no para con x
M': " " " m' acepta lenguaje finito

Dada una entrada (m, x), construyo una máquina m' tal que
m'(w) ejecuta los |w| primeros pasos de m(x).
si para ⇒ acepta.
si no para ⇒ rechaza.

~~Si m' acepta un lenguaje infinito~~ si m para con x ~~...~~ en un nº de pasos n_0, $\forall n \geq n_0$
acepta m' es infinito.
m para con x en n pasos, luego el lenguaje que acepta m' es infinito.
si m no para con x, m' no acepta ninguna palabra ⇒ acepta un lenguaje finito.
luego M' responde sí cuando M responde sí y ~~...~~ con no igual.
Como el complementario a la parada no es semidecidible (si lo fuese el de la parada sería ~~...~~ decidible), nuestro problema no es semidecidible.

(15) a) • Escribe símbolo no blanco con palabra ε:
Dada una entrada M construyo M' tal que sus ~~únicos estados~~ únicas transiciones que van a estados finales son todas las de M que escribían símbolo no blanco en la cinta. ~~...~~
Aceptemos M si M' acepta ε.
Es semidecidible, ya que aceptemos en los casos positivos.
• Sólo escribir símbolos blancos con ε: ~~construyo M' tal que si es~~ la ejecuto con la palabra ε.
Dada una entrada M si la cinta ≠ # rechazo. Acepto cuando pasemos si escribe un símbolo a la cinta en por algún estado por el que ya habíamos pasado dejando la cinta en blanco.
Esto para siempre ⇒ es decidible.

Lo importante

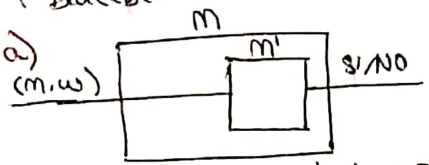
rechaza a ...

(15). b) Veremos que el complementario es ~~decidible~~ semidecidible:
Elijo una palabra w de forma no determinista.
Si M se mueve a la izq. en algún momento ⇒ acepto.
Es semidecidible, dice sí en casos positivos.
Veamos que nuestro problema es semidecidible.

c) •Lee una casilla más de una vez:
Las marco a todas con un * y si paso por una con *, acepto.

•Lee una casilla solo una vez:
Lee o solo izq o solo dcha.
Si leo dos veces la misma, rechazo.
Si no he leído 2 veces la misma casilla y lee un blanco, si hace K+1 pasos mas con K el no de estados. y no repito ⇒ acepto.
Decidible

(16) a)

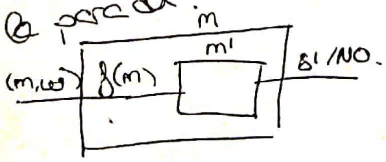


M' acepta si la entrada para para cualquier entrada.
M acepta si m no acepta w (complementario a parada)
$f(m) = m'$ es una MT tal que para una entrada x ejecuta |x| pasos de m con w.
Si m ~~acepta~~ se para con w en |x| pasos ⇒ m' cicla ⇒ M' devuelve NO
~~[illegible]~~
Si m no se para con w en |x| pasos ⇒ m' se para ⇒ M' devuelve SI
(ya que ∀x, m no para con w (cicla con w) y m' para con todas las entradas). NO SEMIDECIDIBLE.

b) el complementario es semidecidible (c)), si este fuese semi ⇒ sería decidible.
Veamos que no es semidecidible reduciendo el problema complementario a la parada a este.
Dada una instancia (m,w) del problema complementario a la parada, $f(m)=m'$ ignora la entrada y procesa w. Si cicla con w la m, m' cicla para cualquier entrada y si m no cicla con w, m' no cicla para ninguna entrada.

c) Semidecidible
Dada una entrada M, elijo de forma no determinista una palabra w.
Si m para con w ⇒ acepto.
En caso contrario ⇒ rechazo.

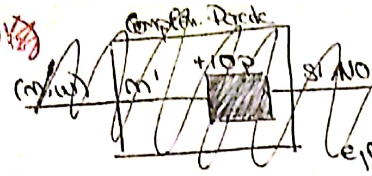
d) No puede ser decidible por ser su complementario es no semidecidible.
Veamos que es no semidecidible reduciendo la al complementario del problema de la parada.



$f(m)=m'$ es una MT que ignora su entrada y ejecuta w.
Si m cicla con w ⇒ m' cicla para todas sus entradas ⇒ devuelve SI
Si m no cicla con w ⇒ m' no cicla para ninguna entrada ⇒ devuelve no.

(17). a) FALSO. b) POSIBLE c) POSIBLE d) FALSO e) FALSO f) VERDADERO g) CIERTO h) [redacted]. VERDADERO.

(18) Por el teorema de Rice no es decidible.
Reducimos el problema de que $L(M)=\emptyset$ a este.
Dada una entrada M del problema $L(M)=\emptyset$, construyo $f(M)=M'$ una máquina a la que le añado el símbolo * a su alfabeto (no estaba antes) y 100 estados nuevos finales para que acepte 100 palabras formadas por secuencias de *.
Si M' acepta 100 palabras, es porque ~~M acepta~~ $L(M)=\emptyset$
Si acepta más, es porque $L(M)\neq\emptyset$.

(19) 

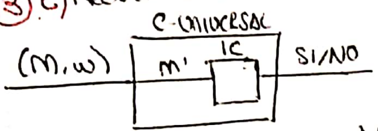
~~Si M cicla con w ⇒ acepta +10~~
~~Si M no cicla con w ⇒ acepta +10~~
~~M' es una máquina que, dada una entrada x, ejecuta |x| pasos de w en M.~~
~~Si M para en |x| ejecutando w ⇒ acepta ⇒ M' acepta inf. palabras~~
~~Si M no para en |x| ejecutando w ⇒ ... $L(M) = \emptyset$~~

b) Por el T^ma^ Rice, no es decidible.
Dada una entrada M una MT, elijo 11 palabras de forma no determinista.
Si las acepta ⇒ acepto la MT M
Si no ⇒ rechazo.

c) No es decidible por no serlo $L(G_1) = L(G_2)$.

d) ~~Deci~~ Decidible; ejecuto |w| pasos. Si acepta ⇒ acepto y si no rechazo.

(13) c) Reducir C-Universal al nuestro. C-Universal: Dado (M,w) ver que M no acepta w.



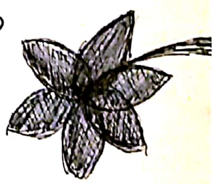
Si M no acepta w ⇒ M' acepta lenguaje IC.
Si M acepta w ⇒ M' acepta lenguaje NO IC.

Sea M' una MT que tiene como entrada $0^n 1^n 0^n$.
Si M acepta w en n pasos ⇒ M' acepta $0^n 1^n 0^n$.
Si M no acepta w en n pasos ⇒ M' rechaza $0^n 1^n 0^n$.
Si (M,w) es una entrada positiva de C-Universal, $L(M') = \emptyset$, que es IC
Si (M,w) " " " negativa de ", $L(M') = 0^n 1^n 0^n$ con $n \geq n_0$ (nº de pasos que necesita M para aceptar w), que es NO IC.

(20) a) Semidecidible:
Tomo una palabra w de forma no determinista. Si ambas máquinas aceptan ⇒ acepto
No decidible: Como $L(G_1) \cap L(G_2) = \emptyset$ es no decidible ⇒ $L(M_1) \cap L(M_2) = \emptyset$ es no decidible.

Probar con palabras de longitud 5 o menos todas en paralelo; si para alguna vez +5 mov ⇒ rechazo, si no acepto.

b) Complementario es semi. ¿d?
d) El complementario es semi pero no decidible (por T^ma^ Rice) ⇒
⇒ este es no semidecidible.



(21) ~~a) FALSO b)~~ c) VERDADERO d) VERDADERO e) POSIBLE

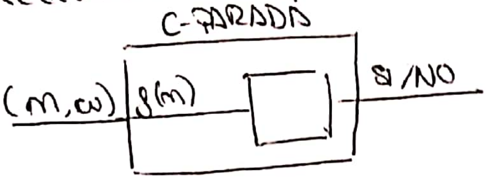
(22) a) Reducimos el lenguaje vacío a este problema.
Cogemos M_1 y M_E donde M_E no tiene transiciones (lenguaje vacío)
Si tiene el mismo lenguaje ⇒ $L(M_1) = \mathcal{E}$
Si no " " " ⇒ $L(M_1) \neq \mathcal{E}$
Como vemos que $L(M_1) = \mathcal{E}$ es no semidecidible ⇒ es no semidecidible.

(24) c) Reducimos el problema de ver si un lenguaje es finito o infinito a este, hacemos intersección con máquina que acepta todas las palabras. La intersección es la máquina original y veo si su lenguaje es finito o no.

(25) a) puede ser lo que quiera b) L_2 no es recursivo c) L_2 no es recursivamente enumerable
d) L_1 es recursivo.

(26) a) lenguaje a regular: aceptado por un autómata determinista
Ej lenguaje regular 1^n0^n con $n \leq 5$ Ej no regular. 1^n0^n.

Reducimos el problema C-PARADA a este ~~f(x)~~ $f(m) = m'$
~~m' reconoce 1^n0^n acepta palabras~~
~~m' acepta palabras de la forma 1^n0^n~~



- Si M para con w en $|x|$ pasos, m' acepta $1^{|x|}0^{|x|}$
- Si m no para con w en $|x|$ pasos m' no acepta $1^{|x|}0^{|x|}$.

- (M, w) no para ⟹ m' acepta (lenguaje vacío (regular))
- (M, w) para ⟹ m' acepta 1^n0^n con $n \geq n_0$ (n_0 nº pasos necesarios para que m pare con w) que es no regular

Como C-PARADA es no r.e ⟹ este es no r.e.

b) Semi: coges un palíndromo de forma no determinista y miras si las 2 la aceptan.
No decidible: Reducimos aceptar un palíndromo a eso.
Dada una MT, hacemos el problema con una MT que acepte todas las palabras. Si MT acepta palíndromo ⟹ la intersección ~~tam~~ tmb. Si no ⟹ intersección tmp

c) Reducimos $L(M) \neq \varepsilon$ a este:
Dada una entrada M, M' es una tal que a cualquier estado que no sea final le hago una transición para que sea accesible.
~~Si acepta~~ Si $L(M) \neq \varepsilon$ ⟹ el estado final es accesible.
Si $L(M) = \varepsilon$ ⟹ " " " no es accesible.

Por tanto es no decidible.

Es semi:
Cojo tantas palabras como estados de forma no determinista (a lo mejor todos los estados son accesibles pero no hay una palabra que pase por todos).
Las ejecuto en paralelo. Si paso por todos los estados ⟹ acepto.
Si no ⟹ rechazo.

d) Miras la configuración. Si tiene + de 5 ⟹ rechazo. Si no ⟹ acepto.

(27) A cada bloque le asigno un nº entero correspondiente a ocurrencias del símbolo arriba − ocurrencias del símbolo abajo.
Si algún bloque tiene asignado el 0 ⟹ acepto
Si hay un bloque positivo y otro negativo ⟹ acepto
En caso contrario ⟹ rechazo
(Si tengo 37 y −13, sumo 13 veces el 37 y resto 37 veces el 13).

aa
a

2-1 = 1

aaa
aaaa

3-4 = -1