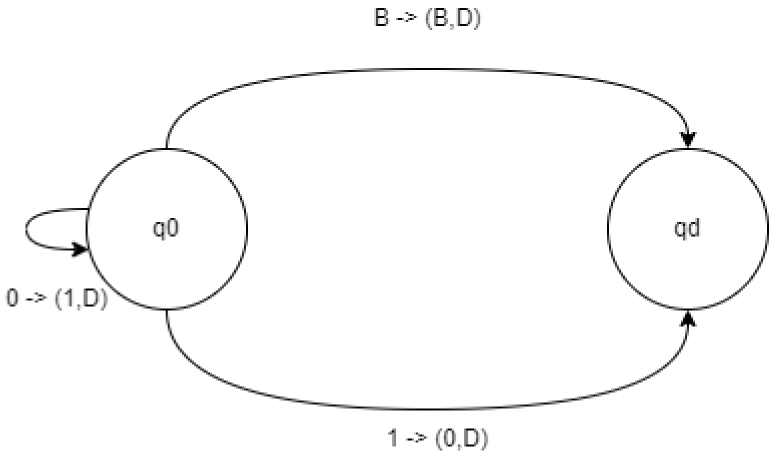
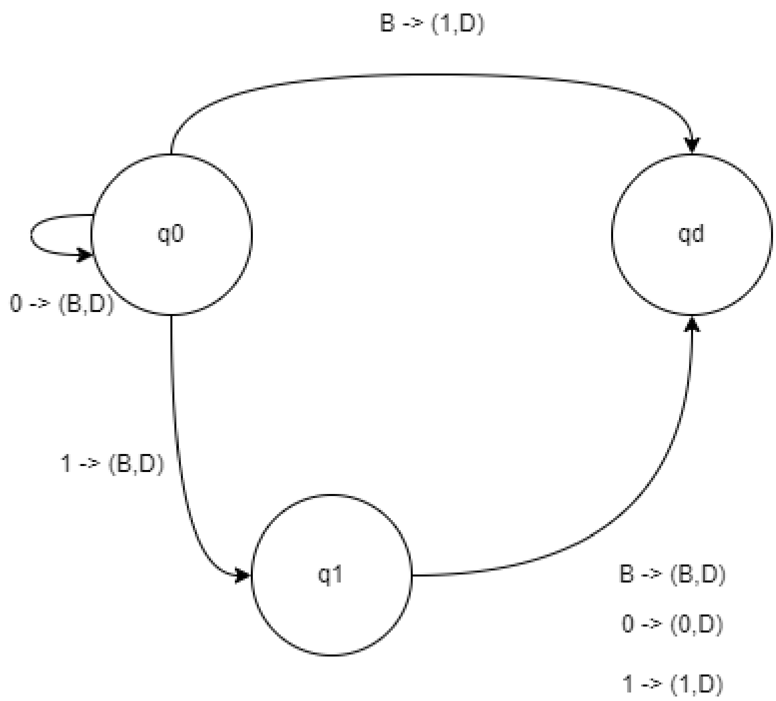
Practica 5

1) Sean L1 y L2, dos lenguajes definidos sobre {0,1}*
   $L1 = \{0^n 1 \mid n \geq 0\}$
   $L2 = \{1^n 0 \mid n \geq 0\}$
   a) Demuestre que existe una reducción (L1 α L2)



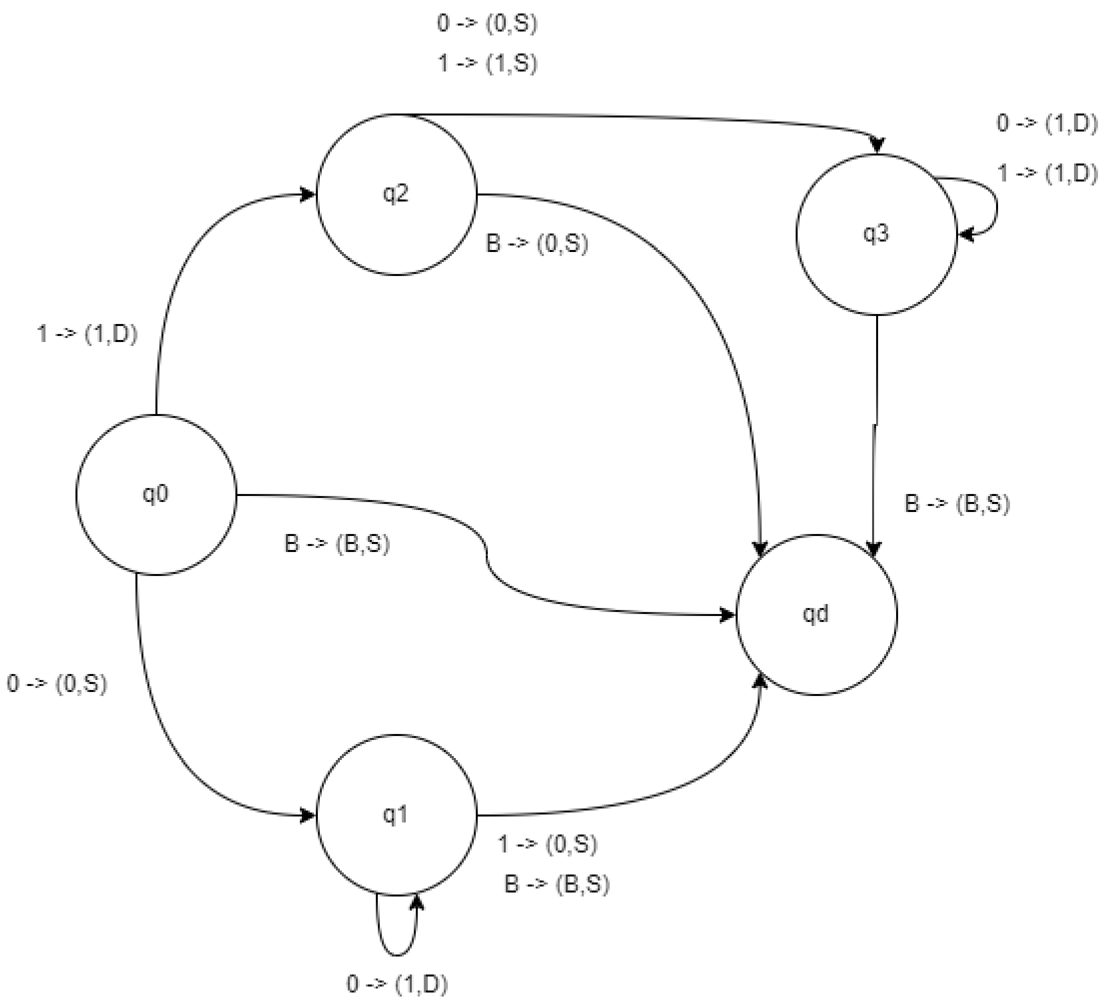
1. Mf siempre se detiene, ya que el alfabeto es finito.
2. w ∈ L1 ⇔ Mf (w) ∈ L2, se puede ver observando la MT
   a. sí w ∈ L1 ⇒ empieza con una cierta cantidad de 0 y tiene solamente un 1 al final ⇒ Mf (w) los 0 pasan a ser 1 y se tiene un 0 al final ⇒ Mf (w) ∈ L2
   b. sí w ∉ L1 ⇒ Mf (w) ∉ L2 como los 0 pasan a ser 1 y si hay un 1 va a pasar a ser 0 si w no pertenecía L1 Mf(w) no va a pertenecer a L2

   b) Idem para L2 = {λ}



1. Mf siempre se detiene, ya que el alfabeto es finito.
2. w $\in$ L1 $\Leftrightarrow$ Mf (w) $\in$ L2, se puede ver observando la MT
   a. sí w $\in$ L1 $\Rightarrow$ empieza con una cierta cantidad de 0 y tiene solamente un 1 al final $\Rightarrow$ Mf (w) los 0 antes del 1 pasan a ser B y el 1 del final pasa a ser B $\Rightarrow$ Mf (w) $\in$ L2
   b. sí w $\notin$ L1 $\Rightarrow$ Mf (w) $\notin$ L2 como los 0 que están antes del 1 pasan a ser B y el primer 1 pasa a ser B, si después de ese 1 sigue habiendo más símbolos (1 o 0), van a quedarse igual, por lo que Mf (w) ≠ λ. En el caso de que w sea λ, se va a agregar un 1 para que Mf (w) = 1 y por lo tanto $\notin$ L

c) Idem para L2 = $\{1^n 0 | n > 0\}$



q0, 0
q1, 0, S

q0, 1
q2, 1, D

q0, B
qd, B, S

q1, 0
q1, 1, D

q1, 1
qd, 0, S

q1, B
qd, B, S

q2, B
qd, 0, S

q2, 1
q3, 1, S

q2, 0
q3, 0, S

q3, 0
q3, 1, D

q3, 1
q3, 1, D

q3, B
qd, B, S

1. Mf siempre se detiene, ya que el alfabeto es finito.
2. w $\in$ L1 $\Leftrightarrow$ Mf (w) $\in$ L2, se puede ver observando la MT
    a. sí w $\in$ L1 $\Rightarrow$ empieza con una cierta cantidad de 0 y tiene solamente un 1 al final $\Rightarrow$ Mf (w) los 0 pasan a ser 1 y el 1 del final pasa a ser 0. En el caso que w = 1, se agregara un 0 atrás $\Rightarrow$ Mf (w) $\in$ L2
    b. sí w $\notin$ L1 $\Rightarrow$ Mf (w) $\notin$ L2
        w $\notin$ L1 puede pasar en 3 situaciones:
        1) sí comienza con 0s, y no tiene un 1 final, esos 0 van a ser transformados a 1 $\Rightarrow$ Mf (w) $\notin$ L2 ya que quedaran siempre todos 1
        2) sí comienza con 0, tiene un 1, pero luego del 1 hay más símbolos, los 0 van que están antes del primer 1 van a ser transformados a 1, lo que esta después del primer 1 se va a ignorar $\Rightarrow$ Mf (w) $\notin$ L2 ya que el primer 0 no será el último símbolo
        3) sí se comienza con un 1, pero después de ese 1 hay más símbolos, todos estos símbolos van a ser pasados a 1 $\Rightarrow$ Mf (w) $\notin$ L2 ya que quedaran siempre todos 1.

(intentar justificar mejor el a y el b)

2) Sean L1 y L2, dos lenguajes tales que existe una reducción (L1 α L2)

    a) Qué se puede afirmar de L1 si se sabe que L2 $\in$ R

    Se pude afirmar que L1 $\in$ R (teorema 1 visto en teoría 10, la demostración está en las diapositivas)

    b) Qué se puede afirmar de L1 si se sabe que L2 $\in$ (CO-RE - RE)

    Se sabe que L2 $\in$ (CO-RE - RE)
    $\Rightarrow$ L2 $\in$ CO-RE ^ L2 $\notin$ RE
    $\Rightarrow$ (definición CO-RE) L2^C $\in$ RE
    $\Rightarrow$ siguiendo el teorema 2 de la teoría 10 L1^C $\in$ RE

⇒ (definición CO-RE) L1 ∈ CO-RE
⇒ Podemos afirmar que L1 ∈ (CO-RE)

c) Qué se puede afirmar de L2 si se sabe que L1 ∈ R

Nada, ya que no sabemos si L2 α L1.

d) Qué se puede afirmar de L2 si se sabe que L1 ∈ (CO-RE – RE)

Si sabemos que L1 ∈ (CO-RE – RE), sabemos que L1 ∉ RE, y por definición L1 ∉ R, por las contrarrecíprocas del teorema 1 y del teorema 2 podemos determinar que

L1 ∉ R ⇒ L2 ∉ R
L1 ∉ RE ⇒ L2 ∉ RE

(consultar)

3) Sean HP y Lu los lenguajes Halting Problem y Lenguaje Universal respectivamente.
HP = {(<M>,w) / M se detiene con input w}
Lu = {(<M>,w) / M acepta w}
Demuestre que existe una reducción HP α Lu

Se construye una MT Mf que computa la función f de reducibilidad
Mf ((<M>,w)) = (<M'>,w)

Mf trabaja de la siguiente manera:

- Si (<M>,w) no es un par válido se deja como esta.
- Si <M> no es un código valido de MT reemplazar <M> en la cinta por una codificación de máquina de Turing que acepta cualquier w. Mf va a construir un <M'> que acepta cualquier w. El w original no se modifica.
- Si (<M>,w) es un par valido y <M> es un código valido de MT, buscar en las quíntuplas de <M> el estado qR y reemplazarlo por qA. Las quintuplas de <M> con estado qA se quedan igual.

1) f es computable ?
Si, dado que Mf siempre se detiene ya que la entrada es finita. En los casos que luego de recorrerla agrega un número de quíntuplas, este número va a ser finito se detendrá.

2) (<M>,w) ∈ HP ⇔ (<M'>,w) ∈ Lu?

   a. (<M>,w) ∈ HP ⇒ (<M'>,w) ∈ Lu
      i. Si <M> es un código valido
      (<M>,w) ∈ HP ⇒ M se detiene
      ⇒ M para en qA o qR
      ⇒ M' para en qA (por construcción)
      ⇒ M' acepta w
      ⇒ (<M>,w) ∈ Lu
      ii. Si <M> no es un código de MT valido

$(<M>,w) \in HP \Rightarrow$ M se detiene inmediatamente
$\Rightarrow$ Por construcción se hace una $<M'>$ donde M' acepta todo w
$\Rightarrow$ M' acepta w
$\Rightarrow (<M>,w) \in Lu$

b. $(<M>,w) \notin HP \Rightarrow (<M'>,w) \notin Lu$
   i. Si $(<M>,w)$ no es un par válido $(<M>,w) \notin HP$ y por construcción $(<M'>,w)$ tampoco lo será (ya que se deja igual) por lo que $(<M'>,w) \notin Lu$
   ii. Si M loopea$\Rightarrow$ M' va a loopear, y por lo tanto rechazara w.

4) Sea HPλ el problema de detención a partir de la cinta en blanco

   $HP\lambda = \{<M> \;/\; M \text{ se detiene con input } \lambda\}$

   Demuestre que existe una reducción HP α HPλ

   Se construye una MT Mf que computa la función f de reducibilidad
   $Mf((<M>,w)) = <M'>$

   Mf trabaja de la siguiente manera:
   - Si $(<M>,w)$ no es un par válido Mf borra la cinta y deja λ
   - Caso contrario Mf va a construir a M' escribiendo las quíntuplas necesarias para que M' chequee que la entrada es λ. Si la entrada no es λ, M' loopeara. Si la entrada es λ escribe w en la cinta (lo hardcodea), posiciona el cabezal al comienzo y simule M sobre w. Si <M> paraba con w <M'> va a parar para entrada λ y si <M> loopeaba con w <M'> tambien va a loopear. Si <M> es un código invalido de MT, <M> va a parar inmediatamente y por lo tanto <M'> al simular su ejecución con w "hardcodeado" también parara inmediatamente.

   1) f es computable ?
      Si, dado que Mf siempre se detiene ya que la entrada es finita y se realizan finitas modificaciones. Mf va a construir una MT finita.

   2) $(<M>,w) \in HP \Leftrightarrow <M'> \in HP\lambda$?

      a. $(<M>,w) \in HP \Rightarrow <M'> \in HP\lambda$

         $(<M>,w) \in HP \Rightarrow$ M se detiene con entrada w
         $\Rightarrow$ M' por construcción se detiene con entrada λ
         $\Rightarrow <M'> \in HP\lambda$

      b. $(<M>,w) \notin HP \Rightarrow <M'> \notin HP\lambda$
         i. Si $(<M>,w)$ no es un par válido $<M'> = \lambda \Rightarrow <M'> \notin HP\lambda$
         ii. Si M loopea $\Rightarrow$ M' loopea $\Rightarrow <M'> \notin HP\lambda$

5) Demuestre que $LV \notin RE$

   $LV=\{(<M>)/L(M) = \emptyset\}$.

   Considere que si <M> es un código inválido de máquina de Turing también pertence a LV (ya que no reconoce ningún string). Así LV es el complemento del lenguaje $LNV=\{(<M>)/L(M) \neq \emptyset\}$

(Ayuda: puede utilizar el complemento de Lu para encontrar una reducción)

Lu^c ={(<M>, w) / (<M>, w) no es un par válido o M rechaza w}

Lu^c α LV
Se construye una MT Mf que computa la función f de reducibilidad
Mf ((<M>,w)) = (<M’>)

Mf trabaja de la siguiente manera:

- Si (<M>, w) no es un par válido Mf va a construir un <M’> que rechace cualquier entrada.
- Si (<M>, w) es un par válido Mf va a construir un <M’> escribiendo las quíntuplas necesarias para que M' borre la entrada y escriba w en la cinta, posicione el cabezal y simule M sobre w. Así M' rechaza cualquier input que M rechaza.

1) f es computable ?
   Si, dado que Mf siempre se detiene ya que la entrada es finita y se realizan finitas modificaciones. Mf va a construir una MT finita.

2) (<M>,w) ∈ Lu^c ⇔ <M’> ∈ LV?

   a. (<M>,w) ∈ Lu^c ⇒ <M’> ∈ LV

      i. (<M>,w) ∈ Lu^c ⇒ (<M>,w) no es un par válido
      ⇒ M’ por construcción rechaza cualquier entrada
      ⇒ <M’> ∈ LV

      ii. (<M>,w)) ∈ Lu^c ⇒ M rechaza entrada w (sucede también si <M> es código invalido de MT)
      ⇒ M’ por construcción rechaza (ya sea loopeando porque M loopea o con qR) cualquier entrada
      ⇒ <M’> ∈ LV

   b. (<M>,w) ∉ Lu^c ⇒ <M’> ∉ LV

      i. Si M acepta w ⇒ M’ acepta cualquier entrada ⇒ <M’> ∉ LV

Queda demostrado que Lu^c α LV.

Como se sabe que Lu^c ∉ RE (demostrado en diapositivas teoría 9) por contrarrecíproca del teorema 2 visto en teoría 10 LV ∉ RE