Computabilidad y Complejidad

# Ejercicios I

La máquina de Turing

Mario Campos Mocholí
3CO21

## 1. Bosqueje una MT que reconozca el lenguaje $L = \{x \in \{a, b, c\}^* / |x|_a = |x|_b = |x|_c\}$.

Definimos una Maquina de Turing formada por cuatro cintas potencialmente infinitas en ambos sentidos. Cada cinta representa:

- Cinta de entrada: En ella se escribe la cadena 'x'
- Cinta A: En ella se escribirá tantos '0' como símbolos 'a' haya en 'x'
- Cinta B: En ella se escribirá tantos '0' como símbolos 'b' haya en 'x'
- Cinta C: En ella se escribirá tantos '0' como símbolos 'c' haya en 'x'

Inicialmente se carga en la cinta de entrada la cadena 'x' y se recorre hasta llegar al símbolo auxiliar 'Blanco' añadiendo en la cita correspondiente un '0' según el símbolo leído.
Una vez realizado esto se lee los símbolos a los que cada cabezal de las cintas A, B y C apuntan, tras lo cual nos podemos encontrar en la siguiente casuística:

- Todos los símbolos leídos son '0': se mueven sendos cabezales a la izquierda y se vuelve a leer
- Todos los símbolos son el 'Blanco': para la maquina aceptando L
- Existe algún símbolo en una cinta que es '0' y en otra que es 'Blanco': para la maquina rechazando la cadena de entrada

La computación acaba cuando o todos los símbolos leídos son el 'Blanco' o no todos los símbolos son iguales.

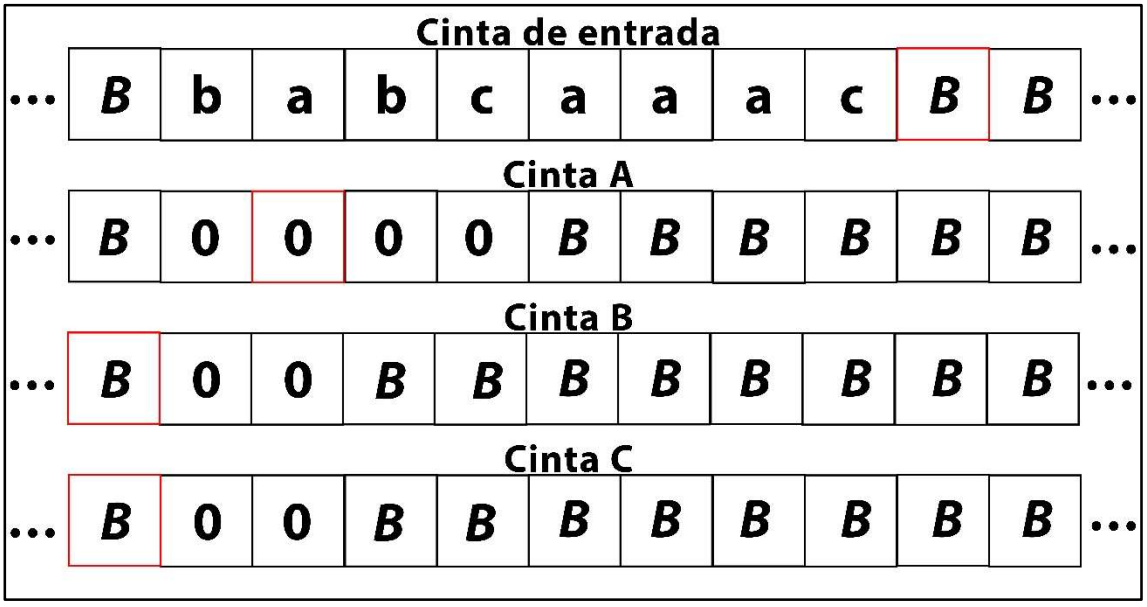


*Imagen 1: Estado final de la computación de la cadena x ='babcaaac', la cual es rechazada*

## 2. Bosqueje una MT que calcule la función $g(n, m) = \sum_{n \le i \le m} i$.

Definimos una Maquina de Turing formada por tres cintas potencialmente infinitas en ambos sentidos. Cada cinta representa:

- Cinta de entrada: En ella se escribe la sucesión de símbolos '0' que representa 'n', un símbolo '1' y la sucesión de '0' que representa 'm'
- Cinta Aux: En ella se escribirá la sucesión de símbolos '0' que representa 'n'
- Cinta de salida: En ella se escribirá la computación de la función g(n, m)

Inicialmente se carga en la cinta de entrada los números 'n' y 'm' tal y como se indica en la definición de la cinta. Entonces:

- Por cada símbolo ‘0’ leído hasta llegar al símbolo ‘1’ se escribe en la cinta Aux un símbolo ‘0’.
- Una vez llegado al símbolo ‘1’ se ignora, a efectos prácticos vuelve a escribir el símbolo ‘1’, y se desplaza a la derecha

Tras llegar a este estado, se sigue recorriendo la cinta de entrada actuando como sigue:

- Si se lee el símbolo ‘Blanco’: para devolviendo la cinta de salida como resultado de la computación
- En caso contrario, por cada símbolo ‘0’ leído en la cinta de entrada, la cinta Aux:
    - Si el cabezal de la cinta Aux se encuentra al principio de la sucesión, es decir, hay un símbolo ‘Blanco’ a la izquierda del cabezal: se desplaza hacia la derecha hasta encontrar un símbolo ‘Blanco’, escribiendo en la cinta de salida tantos símbolos ‘0’ como lea. Finalmente, escribe un símbolo ‘0’ en la cinta Aux
    - Si el cabezal de la cinta Aux se encuentra al final de la sucesión, es decir, hay un símbolo ‘Blanco’ a la derecha del cabezal: se actúa de forma análoga cambiando las direcciones de desplazamiento

La computación acaba cuando se encuentra el primer símbolo ‘Blanco’ en la cinta de entrada.

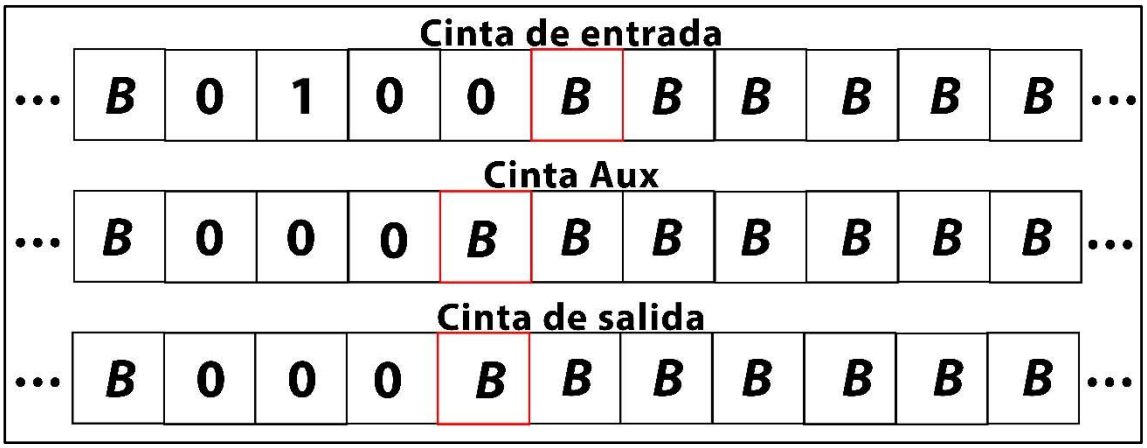


*Imagen 2: Estado final de la computación g(1, 2)*

**3. Sean L y L’ lenguajes, se define la operación binaria ⋄ como sigue L⋄L’ = {z / ∃x ∈ L, ∃y ∈ L’: z = xy ^ |x| = |y|}. Se pregunta:**
**Si L y L’ son lenguajes recursivamente enumerables, entonces ¿lo es también L⋄L’?**

Sí, la operación es cerrada para los lenguajes recursivamente enumerables. Para comprobarlo construimos una máquina de Turing formado por:

- Una maquina desconcatenadora, encargada de dividir la cadena de entrada ‘z’ en dos partes iguales ‘x’ e ‘y’. Si no es capaz de hacerlo, es decir, la longitud de ‘z’ es impar, directamente pararía rechazando. Esta máquina estaría formada por tres cintas: la de entrada, la que contendría a ‘x’ y la que contendría a ‘y’. La máquina iría eliminando un símbolo diferente de ‘Blanco’ del principio, lo añadiría a la cinta X. Tras esto, buscaría uno al final y lo eliminaría añadiéndolo a la cinta Y. Si encuentra un símbolo ‘Blanco’ buscando un símbolo del principio, pararía devolviendo las cadenas; si encuentra un ‘Blanco’ buscando un símbolo al final, pararía rechazando directamente la cadena ‘z’
- Las cadenas ‘x’ e ‘y’ pasarían a las correspondientes aceptadoras del lenguaje de L y de L’, produciendo un símbolo ‘0’ si no pertenece al lenguaje y un símbolo ‘1’ si sí

pertenecen. Como ambas son generadoras de lenguajes recursivamente enumerables, paran ante cualquier entrada valida.

- Finalmente, una última máquina que realizaría la función lógica AND comprueba que se ha obtenido un símbolo '1' en cada aceptadora, aceptando la cadena 'z' en caso afirmativo y rechazándola en el contrario. Esta máquina se podría construir muy simplemente con una cinta de entrada donde se comprobará que todos los símbolos son '1' y siempre pararía

Como las maquinas implementadas siempre paran y las reconocedoras del lenguaje siempre paran si la cadena es del lenguaje, la operación $\diamond$ es cerrada para los lenguajes recursivamente enumerables.

## 4. Sea $M = (\sum, \Gamma, Q, f, B, q_0, F)$ una MT y L(M) un lenguaje recursivo. Para un estado $q \in Q$, se define el lenguaje $L_q(M) = \{x \in L(M) / \exists \alpha, \beta \in \Gamma^*: q_0 x \vdash^* \alpha q \beta\}$. Se pregunta: Si L(M) es un lenguaje recursivo, entonces ¿lo es también $L_q(M)$?

Sí, ya que se podría realizar una maquina de Turing aceptadora de $L_q$ formada por una aceptadora del lenguaje L modificada para aceptar solo aquellas cadenas de entrada que pertenecen a L y que pasen por el estado 'q'. Al ser L un lenguaje recursivo, su aceptadora también lo es, y por lo tanto para para cada entrada. Al ser $L_q$ un subconjunto de L, este también será recursivo y por ende $L_q(M)$ también parará para cada entrada.

## 5. Bosqueje una MT que calcule la función min(n, m) que obtiene el valor mínimo entre n y m *(si n y m tienen el mismo valor entonces devuelve n).*

Definimos una Maquina de Turing formada por tres cintas potencialmente infinitas en ambos sentidos. Cada cinta representa:

- Cinta de entrada: En ella se escribe la sucesión de símbolos '0' que representa 'n', un símbolo '1' y la sucesión de '0' que representa 'm'
- Cinta Aux: En ella se escribirá la sucesión de símbolos '0' que representa 'n'
- Cinta de salida: En ella se escribirá la computación de la función min(n, m)

Inicialmente se recorre la cinta de entrada añadiendo tantos símbolos '0' a la cinta Aux como símbolos '0' se lean en la de entrada. Una vez llegado al símbolo '1' se ignora, a efectos prácticos vuelve a escribir el símbolo '1', y se desplaza a la derecha. Tras esto:

- Se lee un símbolo de la cinta Aux:
  - Si es el símbolo 'Blanco': la máquina para devolviendo la cinta de salida
  - Si es un símbolo '0': desplaza el cabezal a la izquierda y *pasa el control* a la cinta de entrada
- Si se ha leído un símbolo '0' en la cinta de entrada entonces se lee un símbolo en Aux:
  - Si es el símbolo 'Blanco': la máquina para devolviendo la cinta de salida
  - Si es un símbolo '0': escribe un símbolo '0' en la cinta de salida, desplaza el cabezal a la derecha y *devuelve el control* a la cinta de entrada

La computación acaba cuando se encuentre el primer símbolo 'Blanco' en cualquiera de ambas cintas.

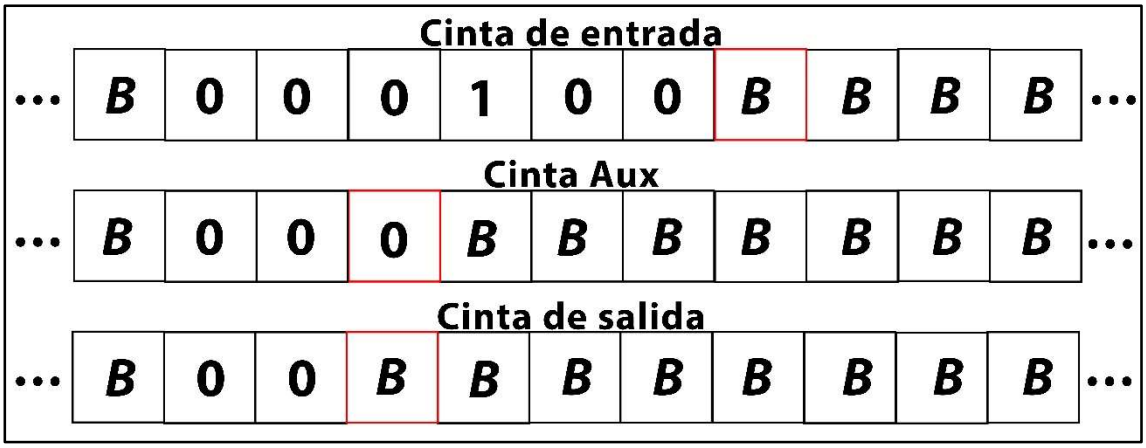

*Imagen 3: Estado final de la computación g(3, 2)*

6. Sean: S un alfabeto, los lenguajes L y L'$\subseteq \sum^*$, y el símbolo # $\notin \sum$. Se define la operación
$\mu(L, L') = \{x\#y / x \in L \wedge y \notin L'\}$. Se pregunta:
Si L y L' son lenguajes recursivamente enumerables ¿lo es también $\mu(L, L')$?
No, la operación no es cerrada para los lenguajes recursivamente enumerables ya que para computar la cadena de entrada se necesitaría comprobar que la subcadena 'y' no pertenece a L' y para ello se necesitaría una maquina aceptadora del lenguaje complemento de L' siendo esta operación no cerrada para los lenguajes recursivamente enumerables, pudiendo no parar para una entrada valida, y por tanto la operación no es cerrada.

7. Para cada lenguaje L se define para cada $n \geq 0$ $P_n(L) = \{x \in L / |x| > 2n\}$. Sea L recursivamente enumerable:

a) ¿Es $P_n(L)$ recursivo para cada $n \geq 0$?

No, no se puede afirmar que $P_n(L)$ sea recursivo ya que al ser L un lenguaje recursivamente enumerable no tiene por qué parar para cada entrada y por ende no ser recursivo.

b) ¿Es $P_n(L)$ recursivamente enumerable para cada $n \geq 0$?

Sí. Para comprobarlo podemos construir una maquina de Turing formada por una maquina reconocedora del lenguaje L y otra maquina encargada de comprobar que la longitud de 'x' sea el doble de 'n' y al ser L recursivamente enumerable también lo será la operación.

c) ¿Es $P_m(L) - P_n(L)$ recursivo para cada $m \geq n \geq 0$?

No, por el mismo razonamiento que el apartado a). Además, la diferencia no es cerrada para los lenguajes recursivamente enumerables.

d) ¿Es $P_m(L) - P_n(L)$ recursivamente enumerable para cada $m \geq n \geq 0$?

Sí, por el mismo razonamiento que el apartado b) y pese que la diferencia no es cerrada para los lenguajes recursivamente enumerables, al ser 'm' mayor o igual que 'n' la operación será cerrada y por tanto el lenguaje que produzca recursivamente enumerable.

8. Se define la operación P sobre lenguajes como sigue *(considere que el alfabeto de definición es {a,b})* $P(L) = \{a^n w b^n : w \in L \wedge |w| = n\}$.

¿Es la clase de los lenguajes recursivos cerrada bajo la operación P?

Sí, la operación P es cerrada para los lenguajes recursivos. Para comprobarlo se construye una maquina de Turing formada por:

- Una máquina que intentara dividir la cadena de entrada en tres partes iguales {x, y, z}. Si no es posible, rechaza directamente la cadena. Esta máquina siempre para ya que la cadena de entrada es finita y se podría recorrer en un conjunto de pasos finitos
- La parte 'x' e 'z' se pasan a otras maquinas que comprueben que todos los símbolos son 'a' o 'b' respectivamente mientras que la parte 'z' se pasa a una aceptadora del lenguaje L
- Finalmente, una última máquina que realizaría la función lógica AND comprueba que se ha obtenido un símbolo '1' en cada aceptadora, aceptando la cadena 'z' en caso afirmativo y rechazándola en el contrario. Esta máquina se podría construir muy simplemente con una cinta de entrada donde se comprobará que todos los símbolos son '1' y siempre pararía

Esta máquina es recursiva ya que acepta el lenguaje generado por la función P y cada máquina por la que está compuesta para.

¿Y la clase de los lenguajes recursivamente enumerables?

Sí, se puede construir una maquina de manera homologa a la del apartado anterior. Esta máquina aceptaría el lenguaje producido por la operación, pero podría no parar si la cadena de entrada no pertenece ya que estaría compuesta por una aceptadora de un lenguaje recursivamente enumerable.

9. Sean L1 y L2 lenguajes. Se define $L1 \diamond L2 = \{xy / x \in L1, y \in L2, |x| < |y|\}$. Se pregunta:

a. Si L1 y L2 son recursivos, ¿lo es también $L1 \diamond L2$?

Sí, es posible crear una máquina de Turing formada por:

- Un generador de pares cuya salida va conectada a dos generadores, uno para cada lenguaje, que generan una cadena 'a' y 'b' respectivamente en orden canónico
- Dichas cadenas se pasan a otra máquina que compara su longitud. Si es la misma, se pasan 'a' y 'b' a otra máquina que las concatena, en caso contrario genera el siguiente par
- Finalmente, se pasa la concatenación 'xy' a otra máquina que comprueba que es igual a 'ab'. En caso afirmativo, para, aceptando la cadena de entrada. En caso negativo, si la longitud de 'ab' sigue siendo igual o menor que la longitud de 'xy' se genera el siguiente par, en caso contrario, rechaza la cadena de entrada

Como un generador de un lenguaje recursivo siempre para y la concatenación es cerrada para estos, podemos afirmar que la operación es cerrada.

b. Si L1 y L2 son recursivamente enumerables, ¿lo es también $L1 \diamond L2$?

Sí, es posible crear una maquina homologa a la creada en el apartado a). Esta máquina es recursivamente enumerable porque acepta el lenguaje generado por la operación, pero puede quedarse indefinidamente generando cadenas.

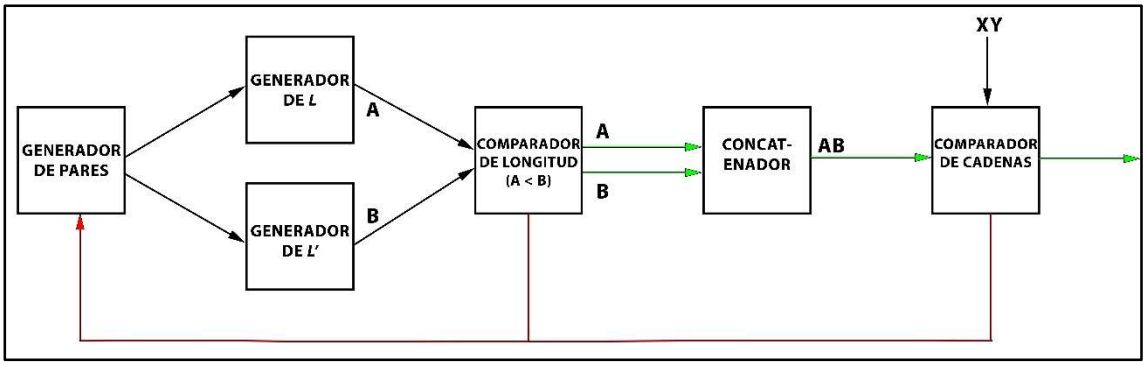


*Imagen 4*

## 10. Bosqueje una MT que calcule la función $g(n) = 2^n$

Definimos una Maquina de Turing formada por tres cintas potencialmente infinitas en ambos sentidos. Cada cinta representa:

- Cinta de entrada: En ella se carga inicialmente la sucesión de símbolos '0' que representa 'n'
- Cinta Aux: En ella, formalmente, se escribirá la computación del instante anterior al actual
- Cinta de salida: En ella se escribirá la computación de la función g(n)

Inicialmente se escribe un símbolo '0' en la cinta Aux. Tras esto se lee el símbolo al que apunta el cabezal de la cinta de entrada:

- Si es el símbolo 'Blanco' y se encuentra en el estado inicial, escribe un símbolo '0' en la cinta de salida y la devuelve; en caso contrario, devuelve la cinta de salida.
- Si es un símbolo '0': la cinta Aux se actualiza como sigue:
    - Por cada símbolo '0' que lea: escribe un símbolo '1' en la posición del '0' y se desplaza hasta encontrar el primer símbolo 'Blanco' escribiendo en su posición un símbolo '1'. Realiza esto hasta que no queden símbolos '0' en la cinta Aux. -Tras esto, vuelve a recorrer la cinta y en la posición de cada símbolo '1' escribe un símbolo '0' escribiendo también un símbolo '0' en la cinta de salida

La computación acaba cuando se lee un símbolo 'Blanco' en la cinta de entrada.

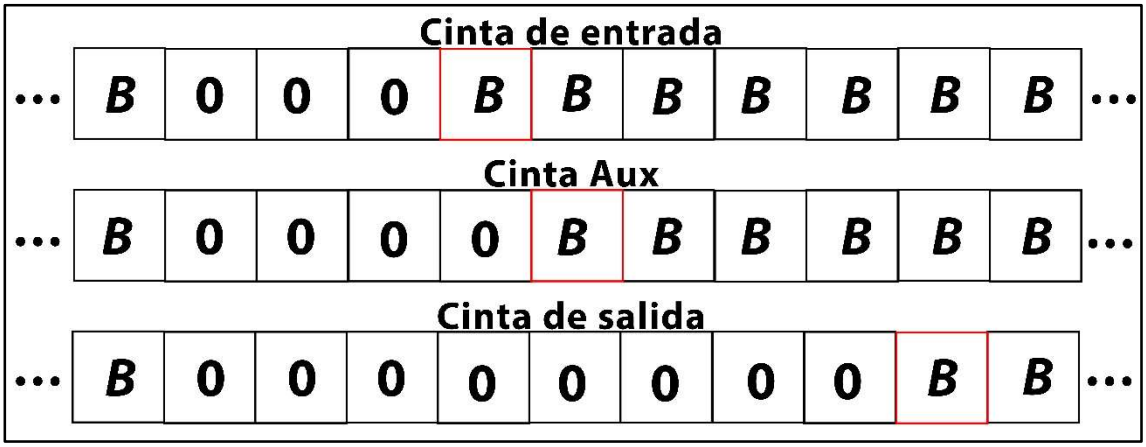


*Imagen 5: Estado final de la computación g(3)*