

# COMPUTABILIDAD Y COMPLEJIDAD (CMC)

**BOLETIN DE EJERCICIOS – LA MÁQUINA DE TURING**

TOMÁS A. PÉREZ

JOSÉ M. SEMPERE

## La máquina de Turing

### Ejercicios propuestos y ejercicios resueltos

#### (I) Ejercicios propuestos

1. Sea el alfabeto  $\Delta = \{1,0\}$  y sea la enumeración 0,1. Bosqueje un generador de Turing que enumere  $\Delta^*$  en el orden canónico especificado.
2. Se define la operación  $P$  sobre palabras como sigue:  $P(x) = x^r$  si la longitud de  $x$  es impar y  $P(x) = xx$  en otro caso. La operación  $P$  se extiende a lenguajes de la forma habitual. ¿Es la clase de los lenguajes recursivos cerrada para  $P$ ? ¿Y la clase de los lenguajes recursivamente enumerables?

3. Sean  $L$  y  $L'$  lenguajes. Se define la operación

$$L \setminus L' = \{x \mid \exists y \in L': yx \in L\}$$

Se pregunta:

- (a) Si  $L$  y  $L'$  son lenguajes recursivamente enumerables, entonces ¿lo es también  $L \setminus L'$ ?
- (b) Si  $L$  es un lenguaje recursivo y  $L'$  es un lenguaje finito, entonces ¿es  $L \setminus L'$  recursivo?

3. Sean  $L$  y  $L'$  dos lenguajes de modo que:

- $L \cup L'$  es recursivamente enumerable, y
- $L'$  es recursivo, y
- $L \cap L'$  es finito.

Se pregunta: ¿Es  $L$  recursivamente enumerable?

4. Sea  $L \subseteq \Sigma^*$  un lenguaje recursivamente enumerable no recursivo. Se define la función sobre cadenas  $f_L$  de forma que  $f_L(x) = x$  si  $x \in L$ , y  $f_L(x) = \lambda$  si  $x \notin L$ . ¿Es  $f_L$  Turing-computable?
5. Bosqueje una máquina de Turing que compute la función  $g(n, m) = n \cdot m$ .
6. Bosqueje una máquina de Turing que compute la función  $g(n, m) = n/m$ , donde  $/$  es el operador de división entera.
7. Bosqueje una máquina de Turing que compute la función  $g(n, m) = n \% m$ , donde  $\%$  es el resto de la división entera.
8. Bosqueje una máquina de Turing que compute la siguiente función  $g: \mathbb{N} \rightarrow \mathbb{N}$ , estando definida como sigue:  
$$g(n) =$$
  - $\lfloor lg_2 n \rfloor$ , si  $n \geq 1$
  - 0, en otro caso
9. Bosqueje una MT que calcule la función  $f(n) = 2^n$ .

10. Bosqueje una máquina de Turing que compute el homomorfismo  $h: \{0,1\}^* \rightarrow \{0,1\}^*$  con  $h(0) = 00$  y  $h(1) = 11$ .

11. Bosqueje una máquina de Turing que compute la función  $f: \{a,b\}^* \rightarrow \{a,b\}^*$  de modo que

- $f(\lambda) = \lambda$
- $\forall e \in \{a,b\}: f(e) = e$
- $\forall e, e' \in \{a,b\}, \forall x \in \{a,b\}^*: f(exe') = e'f(x)e$

12. Bosqueje una MT que calcule la función  $\max(n, m)$  que retorna el valor máximo entre  $n$  y  $m$ .

## (II) Ejercicios resueltos

1. Sean dos lenguajes  $L_1, L_2 \subseteq \Sigma^*$ , se define la operación P del modo que sigue:

$$x \in P(L_1, L_2) \Leftrightarrow \exists u, v, w \in \Sigma^*: x = uvw \wedge u \in L_1 \wedge w \in L_2$$

Se pregunta

- ¿Es la familia de los lenguajes recursivos cerrada para la operación P?
- ¿Es la familia de los lenguajes recursivamente enumerables cerrada para la operación P?

### Solución

El lenguaje definido por  $P(L_1, L_2)$  se caracteriza porque sus palabras tienen un prefijo en  $L_1$ , un sufijo en  $L_2$ , y el segmento restante en  $\Sigma^*$ . Así  $P(L_1, L_2) = L_1 \Sigma^* L_2$ .

Demostraremos que las clases de a) y b) son cerradas para la operación P demostrando que son cerradas para la concatenación, ya que  $\Sigma^*$  es un lenguaje regular y por tanto recursivo y recursivamente enumerable.

- a) La clase de los lenguajes recursivos es cerrada para la concatenación.

Sean  $L$  y  $L'$ , dos lenguajes recursivos; existen dos máquinas de Turing  $M$  y  $M'$  que se detienen para cada entrada con  $L(M) = L$  y  $L(M') = L'$ . Seguidamente definimos una máquina de Turing  $M''$  que se detiene para cada entrada y  $L(M'') = LL'$ .

$M''$  opera como sigue. Tiene una cinta de entrada con dos sectores, en uno de ellos se escribe la palabra de entrada  $x$  y en el otro un símbolo de marcaje  $\sim$ . Este símbolo recorrerá los símbolos de  $x$  (hasta el blanco B inmediatamente a la derecha de  $x$ ). Para una posición del símbolo  $\sim$ , en el rango especificado, se factorizará a  $x$  como  $uv$ , siendo  $u$  el prefijo de  $x$  que termina en el símbolo anterior al marcado y  $v$  el sufijo de  $x$  restante.

Si  $M''$  recibe como entrada  $x = \lambda$ , aplicará la entrada  $\lambda$  a  $M$ , si  $M$  rechaza  $M''$  también rechaza, si  $M$  acepta aplicará  $\lambda$  a  $M'$ , aceptando o rechazando según ésta lo haga.

Si  $M''$  recibe como entrada  $x \neq \lambda$ , pondrá  $\sim$  sobre el primer símbolo de  $x$  y aplicará la entrada  $u$  a  $M$ , si  $M$  acepta aplicará  $v$  a  $M'$ ; si  $M'$  acepta,  $M''$  acepta a  $x$ . Si se produce rechazo, bien por parte de  $M$  o de  $M'$ , se desplaza la marca  $\sim$  una posición a la derecha y vuelve a repetir el proceso, siempre que la marca está dentro del rango de posiciones. Si la marca se sale del rango de posiciones  $M''$  termina rechazando a  $x$ .

Claramente  $M''$  se detiene para cualquier entrada y  $L(M'') = LL'$ .

- b) La clase de los lenguajes recursivamente enumerables es cerrada para la concatenación.

Sean  $L$  y  $L'$ , dos lenguajes recursivamente enumerables; existen dos máquinas de Turing  $M$  y  $M'$  con  $L(M) = L$  y  $L(M') = L'$ . Seguidamente definimos una máquina de Turing  $M''$  con  $L(M'') = LL'$ .

$M''$  opera como sigue.  $M''$  integra dos máquinas  $M$  y  $M'$ , que reciben como entrada una palabra  $z$  y un contador  $j$  que limita el número máximo de movimientos que cada una puede realizar al procesar  $z$  simulando, respectivamente, a  $M$  y a  $M'$ .

Si  $M''$  recibe como entrada  $x = \lambda$ , entonces en una cinta aparte inicializa el contador  $j$  a 1, y aplica  $\lambda$  y  $j$  a  $M$ , si ésta rechaza incrementa el contador  $j$  en uno y vuelve a repetir la operación; si acepta, aplica  $\lambda$  y  $j$  a  $M'$ , si ésta acepta  $M''$  acepta, si no incrementa el contador  $j$  en uno y se vuelve a repetir la operación completa con el nuevo valor de  $j$ .

Si  $M''$  recibe como entrada  $x \neq \lambda$ , entonces arranca un generador triangular controlado de pares  $(i, j)$ , que inicialmente genera el par  $(1, 1)$  y luego genera los pares sucesivos, de uno en uno, cada vez que recibe la señal de control de avance. Ante un par  $(i, j)$ , si  $i > |x| + 1$ , envía al generador la señal de control, en otro caso factoriza  $x$  como  $uv$  donde  $u$  es el prefijo de  $x$  de longitud  $i - 1$ , y  $v$  es el sufijo de  $x$  restante; seguidamente aplica  $u$  y  $j$  a  $M$ , si ésta rechaza envía al generador la señal de control y vuelve a repetirse la operación con el nuevo par  $(i, j)$ ; si acepta, aplica  $v$  y  $j$  a  $M'$ , si ésta acepta  $M''$  acepta, si no se envía al generador la señal de control y vuelve a repetirse la operación completa con el nuevo par  $(i, j)$ .

Claramente  $L(M'') = LL'$ :

- $x \in LL' \Leftrightarrow (\exists u \in L)(\exists v \in L')(x = uv)$
- $u \in L \Leftrightarrow u \in L(M) \Leftrightarrow (\exists n)(M \text{ acepta a } u \text{ en } n \text{ pasos})$
- $v \in L' \Leftrightarrow v \in L(M') \Leftrightarrow (\exists m)(M' \text{ acepta a } v \text{ en } m \text{ pasos})$

Sea  $k = \max(n, m)$ , entonces  $x \in LL'$  si y sólo si en  $M''$  se genera el par  $(|u| + 1, k)$ . Dado que el par  $(|u| + 1, k)$  se genera en alguna etapa se concluye que  $x \in LL'$  si y sólo si  $x \in L(M'')$ .

2. Sea  $L \subseteq \{a, b\}^*$ , se define la operación  $P(L) = \{x \in L : (\exists y \in L)(|y| = |x|_a)\}$ .

Se pregunta

- a) ¿Es la operación  $P$  de cierre en la clase de los lenguajes recursivos?
- b) ¿Es la operación  $P$  de cierre en la clase de los lenguajes recursivamente enumerables?

### Solución

- a) La operación  $P$  es de cierre en la clase de los lenguajes recursivos.

Si  $L$  es recursivo, entonces también lo es  $P(L)$ . En primer lugar, dado que  $P(L) \subseteq L$ , se deduce que, si  $L$  es finito, entonces también lo es  $P(L)$ , y en consecuencia, también es recursivo.

Supóngase ahora que  $L$  es infinito. Para establecer que  $P(L)$  es recursivo definiremos una máquina de Turing  $M$  que lo reconozca y que se detenga para cada entrada.

Puesto que  $L$  es recursivo existe una máquina de Turing  $M_L$  que reconoce a  $L$  y se detiene para cada entrada y también existe un generador  $G$  que genera a  $L$  en orden canónico. La máquina  $M$  opera del siguiente modo. Para una entrada  $x$  la aplica a la máquina  $M_L$ , si ésta rechaza entonces  $M$  también lo hace. Si  $M_L$  acepta, entonces se cuenta el número de veces que en la cadena  $x$  aparece el símbolo  $a$ , sea este número  $n$ , y se arranca el generador  $G$ ; cada vez que  $G$  genera una palabra se comprueba si su longitud coincide con  $n$ . Si coincide se acepta  $x$ . Si no, entonces se genera la siguiente palabra. En estas condiciones ocurrirá

## Computabilidad y Complejidad (CMC)

### La máquina de Turing

---

que, en alguna etapa de la generación, o bien aparece una palabra de longitud  $n$  (en cuyo caso se acepta  $x$ ) o de longitud mayor que  $n$  en cuyo caso se rechaza  $x$ .

Así, la máquina  $M$  acepta al lenguaje  $P(L)$  y se detiene para cada entrada. Por tanto,  $P(L)$  es recursivo.

b) La operación  $P$  es de cierre para la clase de los lenguajes recursivamente enumerables.

Si  $L$  es recursivamente enumerable, entonces también lo es  $P(L)$ . Para establecer que  $P(L)$  es recursivamente enumerable definiremos una máquina de Turing  $M$  que lo reconozca. Puesto que  $L$  es recursivamente enumerable existe una máquina de Turing  $M_L$  que reconoce a  $L$  y también existe un generador  $G$  que genera a  $L$ . La máquina  $M$  opera del siguiente modo. Para una entrada  $x$  la aplica a la máquina  $M_L$ , si ésta acepta, entonces, y sólo entonces, se cuenta el número de veces que en la cadena  $x$  aparece el símbolo  $a$ , sea este número  $n$ , y se arranca el generador  $G$ ; cada vez que  $G$  genera una palabra se comprueba si su longitud coincide con  $n$ . Si coincide se acepta  $x$ . Si no, se genera la siguiente palabra.

Así la máquina  $M$  acepta al lenguaje  $P(L)$ . Por tanto,  $P(L)$  es recursivamente enumerable.

3. Sea  $\Sigma$  un alfabeto y  $L \subseteq \Sigma^*$ , se define

$$P(L) = \{x \in \Sigma^* : (\exists y \in \Sigma^*)((xy \in L) \wedge (|x| = |y|))\}$$

Se pregunta

- a) Si  $L$  es recursivo ¿lo es también  $P(L)$ ?
- b) Si  $L$  es recursivamente enumerable ¿lo es también  $P(L)$ ?

#### Solución

a) Si  $L$  es recursivo entonces también lo es  $P(L)$

Para su demostración construiremos una máquina de Turing  $M'$  que lo reconozca y que se detenga para cada entrada. Puesto que  $L$  es recursivo existe una máquina de Turing  $M$  que lo reconoce y que se detiene para cada entrada. La máquina  $M'$  opera como sigue. Cuando se le aplica una entrada  $x$ , arranca un generador canónico  $G_x$  de todas las palabras de longitud  $|x|$  de  $\Sigma^*$ , este generador comienza generando la primera palabra en el orden canónico de longitud  $|x|$  y seguidamente continuará la generación, de palabra en palabra, cada vez que reciba una señal de control  $\sim$ . Cuando se le solicite mediante esta señal que genere la siguiente palabra y ya no haya más indicará a la máquina  $M'$  que se detenga rechazando. Cada vez que  $G_x$  genere una palabra  $y$ , ésta se concatenará a  $x$  formando la palabra  $z = xy$  que se aplicará a la máquina  $M$ ; si  $M$  acepta, entonces  $M'$  se detendrá aceptando. Si  $M$  rechaza se enviará a  $G_x$  una señal  $\sim$ . En estas condiciones la máquina  $M'$ , por construcción, se detiene para cada entrada y reconoce a  $P(L)$ .

b) Si  $L$  es recursivamente enumerable entonces  $P(L)$  también lo es.

Para su demostración construiremos un generador de Turing  $M'$  que lo genere. Puesto que  $L$  es recursivamente enumerable existe un generador de Turing  $M$  que lo genera.

El generador  $M'$  opera como sigue. Comienza arrancando el generador  $M$  que actuará, controlado por una señal de continuación  $\sim$ , de modo similar al generador  $G_x$  del apartado anterior. Cada vez que  $M$  genera una palabra  $z$ , ésta, si se puede (longitud par), se divide en dos mitades, y se enumera la primera en la cinta de salida, enviándose en cualquier caso (tanto si se puede dividir como si no) la señal  $\sim$  al generador  $M$ . La división de  $z$ , en  $xy$  con

$|x| = |y|$  puede realizarse utilizando una cinta infinita en ambos sentidos con dos sectores uno para  $z$  y el otro para marcar conveniente con dos marcas (por ejemplo \$ y &), la primera y segunda mitad. (También puede hacerse directamente sin el uso de marcas.) Así, por construcción,  $G(M') = P(L)$ .

4. Sea  $P$  la operación sobre lenguajes definida como sigue: para cada palabra  $x$  del lenguaje, si  $x$  contiene un número par de símbolos  $b$ , entonces cada símbolo  $a$  de  $x$  pasa a ser  $aa$ ; si la palabra  $x$  tiene un número impar de símbolos  $b$ , entonces queda como está. Por ejemplo, si  $x = babaa$ , entonces  $P(x) = baabaaaa$ ; si  $x = baa$ , entonces  $P(x) = baa$ . ¿Es la familia de los lenguajes recursivamente enumerables cerrada respecto de la operación  $P$ ?

#### Solución

La familia de los lenguajes recursivamente enumerables es cerrada bajo  $P$ . Sea  $L_{pb} = (a^*ba^*b)^*a^*$  (el lenguaje formado por todas aquellas palabras  $x$  tales que  $|x|_b$  es un número par) y sea  $L_{ib}$  su complementario. Ambos son lenguajes regulares y, por tanto, recursivamente enumerables. Sea  $h$  el homomorfismo tal que  $h(a) = aa$  y  $h(b) = b$ .

Entonces,  $L = (L \cap L_{pb}) \cup (L \cap L_{ib})$  y  $P(L) = h(L \cap L_{pb}) \cup (L \cap L_{ib})$ .

Así, puesto que la clase de los lenguajes recursivamente enumerables es cerrada bajo homomorfismos, unión e intersección, si  $L$  es recursivamente enumerable también lo será  $P(L)$ ; por lo que la clase es cerrada bajo  $P$ .

5. Sean  $L_1$  y  $L_2$  lenguajes. Se define la operación  $\&$  como sigue:

$$L_1 \& L_2 = \{x \in L_1 : (\exists y \notin L_2)(|x| = |y|)\}.$$

¿Es la clase de los lenguajes recursivos cerrada bajo la operación  $\&$ ?

#### Solución

La clase de los lenguajes recursivos sí es cerrada bajo  $\&$ .

Supóngase que el lenguaje complementario de  $L_2$  (en adelante  $L_2^c$ ) es finito, dado que  $L_1 \& L_2 = \{x \in L_1 : (\exists y \in L_2^c)(|x| = |y|)\}$ , entonces  $L_1 \& L_2$  también es finito y, por ser finito, es recursivo.

Supóngase en lo que sigue que  $L_2^c$  es infinito. Sean  $L_1$  y  $L_2$  dos lenguajes recursivos. El lenguaje  $L_2^c$  será también recursivo y, por tanto, existirán máquinas de Turing  $M_1$  y  $G_2^c$  tales que  $M_1$  acepta a  $L_1$  y se detiene para cada entrada y  $G_2^c$  genera a  $L_2^c$  en orden canónico.

A partir de  $M_1$  y  $G_2^c$  construimos una máquina  $M$  que funciona de la siguiente manera: Cuando  $M$  recibe una entrada  $x$  simula en primer lugar a  $M_1$  sobre  $x$ . Si  $M_1$  rechaza a  $x$ ,  $M$  se detiene rechazando. En caso contrario,  $M$  activa al generador  $G_2^c$  e irá, por tanto, generando las palabras de  $L_2^c$  en orden canónico deteniéndose momentáneamente (en una pausa) cada vez que genere una palabra. Así cada vez que  $G_2^c$  genera una palabra  $z$ ,  $M$  comparará  $|x|$  con  $|z|$ . Si  $|z| < |x|$ ,  $M$  activa de nuevo a  $G_2^c$  que genera la siguiente palabra repitiéndose este proceso. Si  $|z| = |x|$ ,  $M$  se detiene aceptando. Si  $|z| > |x|$ ,  $M$  se detiene rechazando (puesto que habrá verificado que no hay ninguna palabra en  $L_2^c$  de la misma longitud que  $x$ ). Por tanto, la máquina  $M$  así construida acepta el lenguaje  $L_1 \& L_2$  y se detiene para cada entrada. En consecuencia,  $L_1 \& L_2$  es recursivo.

6. Sean  $L \subseteq \Sigma^*$  y  $a \in \Sigma$ . Se define la operación  $Insertar(a, L) = \{xay : xy \in L\}$ .
- Si  $L$  es recursivo ¿lo es también  $Insertar(a, L)$  ?
  - Si  $L$  es recursivamente enumerable ¿lo es también  $Insertar(a, L)$  ?

Solución

a) Si  $L$  es recursivo también lo es  $Insertar(a, L)$ .

Sea  $M$  una MT que reconoce a  $L$  y que se detiene para cada entrada. Definimos una MT  $M'$  que actúa de acuerdo al siguiente algoritmo de cómputo:

Operando a partir del comienzo de  $x$ :

Si existe una nueva ocurrencia de  $a$ , todavía no considerada,  
ENTONCES eliminarla y aplicar la palabra resultante  $x'$  a  $M$ , si ésta se detiene  
aceptando hacer lo mismo, en otro caso restaurar  $x$  y volver a proceder;  
EN OTRO CASO detenerse rechazando.

Para realizar este cómputo se utiliza una cinta con varios sectores con:

- Un sector para la palabra de entrada  $x$ .
- Un sector para marcar la primera posición de la cinta para, al operar, no salirse de la misma.
- Un sector en la cinta para realizar el pertinente marcaje que controla el procesamiento de las  $a$ 's. Marca en cada etapa el símbolo  $a$  eliminado para dar lugar a  $x'$ .
- Dos sectores, con los que se opera en consonancia, para procesar la correspondiente entrada  $x'$  obtenida a partir de  $x$  con el correspondiente símbolo  $a$  (en cada caso) eliminado. En uno de ellos se realiza el procesamiento de la entrada  $x'$ , en el otro se lleva un marcaje (^) que señala la posición más a la derecha alcanzada en el sector de procesamiento de  $x'$ . De este modo, cuando (si ha lugar) se ha de procesar una nueva entrada  $x'$  (con excepción de la primera) antes de hacerlo, como paso previo, lo primero que se realiza es limpiar, poniendo a blancos, el sector de procesamiento de  $x'$  utilizando el marcaje del otro sector (como delimitador del tramo de cinta a considerar), modificando su marcaje de modo que se ajuste inicialmente a la entrada  $x'$  (lo que inicialmente se hace cuando se procesa la primera  $x'$ ).

Claramente  $M'$  reconoce a  $Insertar(a, L)$  y se detiene para cada entrada, por tanto  $Insertar(a, L)$  es recursivo.

b) Si  $L$  es recursivamente enumerable también lo es  $Insertar(a, L)$ .

Sea  $G$  un generador de  $L$ , seguidamente definimos un generador  $G'$  para  $Insertar(a, L)$ . El generador  $G'$  simula a  $G$  de modo que cuando  $G$  genera una palabra  $x$ ,  $G'$  genera las  $|x|+1$  palabras obtenidas a partir de  $x$  insertando un símbolo  $a$  en cada una de las posiciones admisibles.



7. Sean  $L$  y  $L'$  dos lenguajes, se define la operación  $F$  de modo que

$$F(L, L') = \{x \in L : x^r \notin L'\}.$$

Si  $L$  y  $L'$  son lenguajes recursivos ¿lo es también  $F(L, L')$  ?

#### Solución

Si  $L$  y  $L'$  son recursivos, entonces también lo es  $F(L, L')$ . Para demostrarlo definimos una MT  $F$  que lo reconoce y que se detiene para cada entrada. Sea  $M$  una MT que reconoce a  $L$  y que se detiene para cada entrada y  $M'$  una MT que reconoce a  $L'$  y que también se detiene para cada entrada.  $F$  opera como sigue: dada una entrada  $x$  la aplica a  $M$  (copiándola previamente en otra cinta asociada a  $M$ ), si  $M$  rechaza  $F$  se detiene rechazando; en otro caso obtiene el reverso de  $x$  y lo aplica a  $M'$ , si  $M'$  rechaza  $F$  se detiene aceptando, en otro caso se detiene rechazando. Para obtener el reverso de la entrada  $x$  puede recorrerla de derecha a izquierda e ir escribiendo los símbolos, en la secuencia leída, en otra cinta, asociada a  $M'$ , avanzando sobre ella de izquierda a derecha.

8. ¿Son los lenguajes recursivamente enumerables cerrados para los homomorfismos inversos? ¿Y los lenguajes recursivos?

#### Solución

Sea un homomorfismo  $h: \Sigma^* \rightarrow \Gamma^*$  y sea  $L \subseteq \Gamma^*$  un lenguaje recursivamente enumerable. Demostramos que también lo es  $h^{-1}(L)$ .

Recuérdese que  $h^{-1}(L) = \{x \in \Sigma^* : h(x) \in L\}$ .

Puesto que  $h$  admite una descripción finita puede codificarse en la función de transición (en el control finito) de una MT. Sea  $M$  una MT tal que  $L(M) = L$ . Seguidamente definimos una MT  $M'$  con  $L(M') = h^{-1}(L)$ .  $M'$  tiene  $h$  embebido en su función de transición y opera como sigue: dada una entrada  $x$ ,  $M'$  obtiene  $h(x)$  y se la aplica a  $M$  dejándole a ésta el resto de la computación. Por tanto,  $M'$  acepta a  $x$  si y sólo si  $M$  acepta a  $h(x)$ , esto es

$$L(M') = \{x \in \Sigma^* : h(x) \in L\} = h^{-1}(L).$$

Para el caso de los lenguajes recursivos se procede de modo similar partiendo de una máquina de Turing  $M$  que reconozca a  $L$  y se detenga para cada entrada.

9. ¿Son los lenguajes recursivamente enumerables cerrados para los homomorfismos?

#### Solución

Sea un homomorfismo  $h: \Sigma^* \rightarrow \Gamma^*$  y sea  $L \subseteq \Sigma^*$  un lenguaje recursivamente enumerable. Demostramos que también lo es  $h(L)$ .

Recuérdese que  $h(L) = \{h(x) : x \in L\}$ . Puesto que  $h$  admite una descripción finita, el homomorfismo puede codificarse en la función de transición de un generador de Turing. Sea  $M$  un generador de Turing tal que  $G(M) = L$ . Seguidamente definimos un generador de Turing  $M'$  con  $G(M') = h(L)$ .  $M'$  tiene  $h$  embebido en su función de transición y opera como sigue: simula a  $M$  de modo que cada vez que  $M$  genera una palabra  $x$ ,  $M'$  genera la palabra  $h(x)$ . Por tanto,  $M'$  genera el lenguaje  $G(M') = \{h(x) : x \in L\} = h(L)$ , por lo que  $h(L)$  es un lenguaje recursivamente enumerable.

10. Sean  $L_1, L_2$  y  $L$  lenguajes no vacíos. Sea  $\mathcal{L}_{REN}$  la clase de los lenguajes recursivamente enumerables. Pruebe o refute la siguiente implicación:

$$(\forall L_1, L_2, L) [(L_1 \in \mathcal{L}_{REN} \wedge L_2 \in \mathcal{L}_{REN} \wedge L_1 \cap L_2 = \emptyset \wedge L \cap L_2 = \emptyset) \Rightarrow L \in \mathcal{L}_{REN}]$$

#### Solución

El enunciado es falso. Sea  $L$  un lenguaje no recursivamente enumerable.  $L^c$  es un lenguaje infinito y por tanto existen dos palabras diferentes que pertenecen al mismo; sean  $x_1$  y  $x_2$  dos palabras en estas condiciones. Sean ahora  $L_1 = \{x_1\}$  y  $L_2 = \{x_2\}$ . Se tiene que  $L_1 \in \mathcal{L}_{REN} \wedge L_2 \in \mathcal{L}_{REN} \wedge L_1 \cap L_2 = \emptyset \wedge L \cap L_2 = \emptyset$  y, sin embargo,  $L \notin \mathcal{L}_{REN}$ .

11. Se define la siguiente operación sobre palabras:  $P(x) = 0^n x 0^n$  donde  $n = |x|_0$ . Esta operación se extiende del modo usual a lenguajes, esto es:  $P(L) = \{P(x) : x \in L\}$ . Pruebe o refute las siguientes implicaciones:

- a)  $L$  es recursivo  $\Rightarrow P(L)$  es recursivo.
- b)  $P(L)$  es recursivo  $\Rightarrow L$  es recursivo.

#### Solución

a) Veamos que  $P(L)$  es recursivo cuando  $L$  lo es.

Puesto que  $L$  es recursivo existe una máquina de Turing  $M$  que se detiene para cada entrada y que reconoce a  $L$ . Sea  $M'$  la máquina de Turing constituida, y que opera, como sigue:  $M'$  tiene un módulo  $M''$  que cuando recibe la entrada  $z$  (de  $M'$ ) examina si su número de ceros es (o no) múltiplo de 3, ya que si pertenece a  $P(L)$  necesariamente ha de ser así. [Esto se puede llevar a cabo utilizando tres estados:  $p_0, p_1$  y  $p_2$ ; se inicia el proceso en  $p_0$  y cada vez que se lee un cero se transita al siguiente estado (operando consecuentemente en módulo 3), esto es, se transita de  $p_0$  a  $p_1$ , de  $p_1$  a  $p_2$  y de  $p_2$  a  $p_0$ . Si al terminar se está en el estado  $p_0$ , entonces el número de ceros es múltiplo de 3, en cualquier otro caso no lo es.] Si no es múltiplo de 3,  $M'$  se detiene sin aceptar. Si el número de ceros es múltiplo de 3, entonces  $M''$  lo divide por 3 [este cociente puede obtenerse incrementando un contador, originalmente a cero, cada vez que se realiza la transición de  $p_2$  a  $p_0$ ] y examina si existe en la entrada  $z$  un prefijo y un sufijo de ceros con esta longitud. Si no los hay  $M'$  se detiene sin aceptar. Si los hay los elimina y emite la palabra  $z'$  resultante, continuándose como sigue. Seguidamente arranca la máquina  $M$  y le suministra la entrada  $z'$ . Procediendo a partir de este punto tal y como lo hace  $M$ .

Así  $M'$  es una máquina de Turing que se detiene para cada entrada y que reconoce a  $P(L)$ .

b) Veamos que  $L$  es recursivo cuando  $P(L)$  lo es.

Del apartado anterior se deriva que la operación  $P$  es inyectiva, en consecuencia, podemos proceder como sigue. Puesto que  $P(L)$  es recursivo existe una máquina de Turing  $M_1$  que se detiene para cada entrada y que lo reconoce. Sea  $M'_1$  la máquina de Turing definida como sigue: Cuando recibe una entrada  $x$  calcula su número de ceros  $n$  y le añade el prefijo y el sufijo  $0^n$ ; la palabra  $x'$ , así formada,  $x' = 0^n x 0^n$ , se aplica a  $M_1$  procediendo, a partir de aquí, igual que  $M_1$ . Así  $M'_1$  es una máquina de Turing que se detiene para cada entrada y que reconoce a  $L$ .

12. Sea  $\Sigma$  un alfabeto y  $R \subseteq \Sigma^*$  un lenguaje recursivo dado. Para  $L, L' \subseteq \Sigma^*$  se define  $P(L, L') = \{x \in L : (\exists y \in L') (prefijos(x) \cap sufijos(y) \cap R \neq \emptyset)\}$ . Si  $L$  y  $L'$  son dos lenguajes recursivamente enumerables ¿lo es también  $P(L, L')$ ?

#### Solución

Veremos que en las condiciones dadas el lenguaje  $P(L, L')$  es recursivamente enumerable. Puesto que  $R$  es un lenguaje recursivo existe una máquina de Turing que se detiene para cada entrada y que reconoce a  $R$ , sea  $M_R$  una máquina en estas condiciones.

Puesto que  $L$  es un lenguaje recursivamente enumerable existe una máquina de Turing  $M_L$  que lo reconoce. Puesto que  $L'$  es un lenguaje recursivamente enumerable existe un generador de Turing  $G_{L'}$  que lo genera.

En lo que sigue vamos a definir una máquina de Turing  $M$  que reconozca a  $P(L, L')$ .  $M$  está constituida y opera como sigue. Cuando se le aplica una entrada  $x$  a  $M$  ésta le aplica  $x$  a  $M_L$ , si  $M_L$  reconoce a  $x$ , y sólo en este caso, se arranca el generador  $G_{L'}$  de modo que inicie la generación de cada palabra cada vez que se le aplique una señal de continuación. Cada vez que  $G_{L'}$  genera una palabra  $y$ , se le aplica, junto con  $x$ , a una máquina PS que obtendrá uno por uno todos los prefijos (de  $x$ ) y sufijos (de  $y$ ) comunes, mientras sea posible, cada vez que se le aplique una señal de operación. (Para llevar a cabo esta tarea se puede disponer de dos cintas, una para  $x$  y otra para  $y$ , cada una con dos sectores para mantener una marca que especifique el prefijo y sufijo considerado cada vez; esta marca se va desplazando a medida que se realiza el cálculo.) De este modo el módulo PS obtiene el primer prefijo-sufijo común ( $\lambda$ ) y se le aplica a  $M_R$  si ésta acepta, la palabra  $x$  es aceptada por  $M$ , en otro caso se envía a PS la señal de operación para genere el siguiente prefijo-sufijo común y se vuelve a proceder, cuando ya no haya más prefijo-sufijos comunes se envía la señal a  $G_{L'}$  para que continúe con el proceso de generación. De este modo el lenguaje que acepta la máquina  $M$  es  $P(L, L')$ .

13. Dados un alfabeto  $\Sigma$ ,  $L \subseteq \Sigma^*$  y  $x \in \Sigma^*$ , se define la operación cociente  $x^{-1}L = \{y \in \Sigma^* : xy \in L\}$ .

- Si  $L$  es un lenguaje recursivo ¿lo es también  $x^{-1}L$  para cada  $x \in \Sigma^*$ ?
- Si para  $x \in \Sigma^*$ ,  $x^{-1}L$  es un lenguaje recursivo ¿lo es también  $L$ ?
- Si  $L$  es un lenguaje recursivamente enumerable ¿lo es también  $x^{-1}L$  para cada  $x \in \Sigma^*$ ?

#### Solución

- a) Si  $L$  es un lenguaje recursivo entonces para cada  $x \in \Sigma^*$ ,  $x^{-1}L$  también es recursivo

Si  $L$  es un lenguaje recursivo existe una máquina de Turing  $M$  que lo reconoce y que se detiene para cada entrada. Seguidamente definiremos una máquina de Turing  $M'$  que se detendrá para cada entrada y que reconocerá a  $x^{-1}L$ . La máquina  $M'$  está constituida, de modo que tiene a la palabra  $x$  integrada en su función de transición, y opera como sigue. Cuando se le aplica una palabra  $y$  se la concatena a  $x$  (que se encuentra almacenada en el control finito) obteniendo la palabra  $xy$  que seguidamente se le aplica a  $M$  comportándose a partir de aquí como ésta. Así se tiene que  $L(M') = x^{-1}L$ .

b) Si para  $x \in \Sigma^*$ ,  $x^{-1}L$  es un lenguaje recursivo ¿lo es también  $L$ ?

No necesariamente. Sea el alfabeto  $\Sigma = \{0,1,\$\}$ , y sea  $L \subseteq \{0,1\}^*$  un lenguaje recursivamente enumerable pero no recursivo. En estas condiciones:  $\$^{-1}L = \emptyset$ . Así  $\$^{-1}L$  es recursivo sin serlo  $L$ .

c) Si  $L$  es un lenguaje recursivamente enumerable ¿lo es también  $x^{-1}L$  para cada  $x \in \Sigma^*$ ?

Sí. Como  $L$  es un lenguaje recursivamente enumerable existe un generador  $M$  que lo genera, esto es,  $G(M) = L$ . Seguidamente definimos una máquina de Turing  $M'$  que genera a  $x^{-1}L$ . La máquina  $M'$  está constituida, de modo que tiene a la palabra  $x$  integrada en su función de transición, y opera como sigue. Simula al generador  $M$  de modo que cuando éste genera una palabra  $z$  (momentáneamente se detiene esperando una señal de continuación para continuar con la generación de  $L$ ),  $M'$  comprueba si  $z$  tiene como prefijo a  $x$ , en cuyo caso elimina  $x$  de  $z$  obteniendo el correspondiente sufijo  $z'$  que es generado, en otro caso no se realiza ninguna generación relativa a la palabra  $z$ . Seguidamente se envía a  $M$  la señal para que continúe con la generación de las palabras de  $L$ .

14. Sea  $L$  un lenguaje, se define  $P(L) = \{x : (\exists u, v)(x = uv \wedge vu \in L)\}$ .

- a) Si  $L$  es un lenguaje recursivamente enumerable ¿lo es también  $P(L)$ ?
- b) Si  $L$  es un lenguaje recursivo ¿lo es también  $P(L)$ ?

#### Solución

a) Si  $L$  es un lenguaje recursivamente enumerable lo es también lo es  $P(L)$ .

Para establecerlo demostraremos que existe un generador de Turing que genera  $P(L)$ . Puesto que  $L$  es recursivamente enumerable existe un generador de Turing  $M$  tal que  $G(M)=L$ . A partir de  $M$  definiremos un generador  $M'$  tal que  $G(M') = P(L)$ .  $M'$  está definido para operar como sigue: utiliza el generador  $M$  para generar palabras de  $L$ ; inicialmente opera hasta generar la primera, luego seguirá operando cuando reciba una señal de generación. Cada vez que  $M$  genera una palabra  $z$  ésta se aplica a un módulo  $F$  que factoriza secuencialmente la misma en todos los pares  $(v, u)$  tales que  $z = vu$ , generando seguidamente, a su vez, en cada caso, la palabra  $uv$ . Cuando  $F$  termina este proceso envía a  $M$  una señal de generación para que continúe con la tarea de generación de  $L$ . Así se tiene que  $G(M') = P(L)$ . Para llevar a cabo estas factorizaciones en el módulo  $F$  puede utilizarse una cinta con dos sectores: el inferior para  $z$  y el superior para llevar mediante una marca,  $\sim$ , el control de donde termina  $v$  y donde comienza  $u$ ; por ejemplo, puede tomarse que  $u$  comience en la posición marcada. Esta marca se avanza una posición, si es posible, cada vez que se recibe una señal de factorización después de haber generado la anterior palabra  $uv$  correspondiente a la  $z$  en curso; cuando ya no se puede avanzar la marca se envía al generador la señal de generación anteriormente mencionada. Se comienza colocándola sobre el primer símbolo de  $z$  ( $v = \lambda$  y  $u = z$ ) y se termina en la celdilla, en blanco, siguiente al último símbolo de  $z$  ( $v = z$  y  $u = \lambda$ ). Para el caso  $z = \lambda$  se toma directamente como única factorización  $u = v = \lambda$ .

b) Si  $L$  es un lenguaje recursivo, entonces también lo es  $P(L)$ .

Para establecerlo demostraremos que existe una máquina de Turing  $M_P$  que se detiene para cada entrada y que reconoce a  $P(L)$ . Puesto que  $L$  es recursivo existe una máquina de

Turing  $M$  que se detiene para cada entrada con  $L(M) = L$ . La máquina  $M_P$  está definida para operar como sigue: cuando se le aplica una entrada  $z$  ésta, a su vez, se aplica internamente a un módulo de factorización  $F'$  que, en condiciones similares al  $F$ , obtiene un par  $(u, v)$  con  $z = uv$  cada vez que, con excepción de la primera, se le envía una señal de factorización. Cuando se le suministra  $z$  obtiene automáticamente el primer par. Si recibe una señal de factorización y ya ha realizado todas las posibles factorizaciones detiene todo el proceso y rechaza la entrada. Cuando factoriza un par  $(u, v)$  proporciona como salida la palabra  $t = vu$  que se le aplica seguidamente a la máquina  $M$ , si ésta acepta  $t$ , entonces  $M_P$  acepta  $z$ , en otro caso, si rechaza  $t$ , se envía al módulo  $F'$  una señal de factorización para si es posible obtenga la siguiente factorización. Claramente, en estas condiciones,  $M_P$  se detiene para cada entrada y  $L(M_P) = P(L)$ .

15. Sea  $\Sigma$  un alfabeto, para  $x, y \in \Sigma^*$  se define la operación  
 $P(x, y) = \{z : (\exists u, v, w)(x = uv \wedge y = vw \wedge z = uvw)\}$ .

Esta operación se extiende a lenguajes  $L, L' \subseteq \Sigma^*$  de la manera habitual, esto es,

$$P(L, L') = \bigcup_{\substack{x \in L \\ y \in L'}} P(x, y)$$

Si  $L$  y  $L'$  son lenguajes recursivos ¿lo es también  $P(L, L')$ ?

#### Solución

Si  $L$  y  $L'$  son lenguajes recursivos, entonces también lo es  $P(L, L')$ . Para establecerlo demostraremos que existe una máquina de Turing  $M_P$  que se detiene para cada entrada y que reconoce a  $P(L)$ . Puesto que  $L$  y  $L'$  son recursivos existen dos máquinas de Turing  $M$  y  $M'$  que se detienen para cada entrada y  $L(M) = L$  y  $L(M') = L'$ .

La máquina  $M_P$  está definida para operar como sigue: cuando se le aplica una entrada  $z$  ésta se pasa a un módulo de factorización  $T$  que la factoriza en ternas  $(u, v, w)$  tales que  $z = uvw$  cada vez que recibe una señal de factorización, excepto en la primera ocasión que lo hace automáticamente, y proporciona como salida las palabras  $x = uv$  e  $y = vw$ . Si al recibir la señal de factorización no hay ninguna nueva terna que obtener detiene todo el proceso y rechaza la entrada  $z$ . La salida  $x$  se le aplica ahora a la máquina  $M$  si ésta rechaza se envía una señal de factorización a  $T$ , si acepta se le aplica la palabra  $y$  a  $M'$  si ésta rechaza se envía una señal de factorización a  $T$ , en otro caso, si acepta, se acepta la entrada  $z$ . Para factorizar, el módulo  $T$  (sin entrar en todos los detalles precisos) puede utilizar una cinta con tres sectores: el inferior para  $z$  y los otros dos para albergar, en cada uno una marca,  $\checkmark_1$  y  $\checkmark_2$ , que permitan controlar donde se ubican los segmentos  $u, v$  y  $w$ . Así, por ejemplo, la marca del sector intermedio ( $\checkmark_1$ ) puede señalar la celdilla donde termina  $u$  y la del sector superior ( $\checkmark_2$ ) donde comienza  $w$ . Cada vez que recibe una señal de factorización mueve una posición, hacia la derecha, la marca del sector superior, salvo si ya ha sobrepasado el final de  $z$  en cuyo caso avanza, si es posible, una posición la marca del sector intermedio y sobre la celdilla contigua se coloca la marca del sector superior ( $v = \lambda$ ). Inicialmente la marca intermedia se encuentra en la celdilla en blanco inmediatamente anterior al primer símbolo de  $z$  y la marca superior sobre el primer símbolo de  $z$  ( $u = v = \lambda$  y  $w = z$ ). En la última factorización la marca intermedia se encuentra sobre el último símbolo de  $z$  y la superior sobre la celdilla inmediatamente a la derecha ( $u = z$  y  $v = w = \lambda$ ). En el caso en que  $z = \lambda$  sólo produce la factorización  $u = v = w = \lambda$ . Claramente, en estas condiciones, la máquina  $M_P$  se detiene para cada entrada y  $L(M_P) = P(L, L')$ .

16. Sea  $\Sigma$  un alfabeto y  $L \subseteq \Sigma^*$ . Sea  $R \subseteq \Sigma^*$  un lenguaje recursivo. Se define la operación  $P_R(L) = \{x \in L : (x \notin R) \wedge (x = x^r)\}$ .

- a) Si  $L$  es un lenguaje recursivamente enumerable ¿lo es también  $P_R(L)$ ?
- b) Si  $L$  es un lenguaje recursivamente enumerable ¿lo es también  $L - P_R(L)$ ?

#### Solución

Sea el lenguaje  $I = \{x \in \Sigma^* : x = x^r\}$ . En primer lugar, estableceremos que este lenguaje es recursivo; para este fin definiremos una máquina de Turing  $M_I$  que se detenga para cada entrada y que lo reconozca. La máquina  $M_I$  tiene dos cintas y opera como sigue: cuando recibe una entrada  $x$  la recorre copiándola en la otra cinta, luego ubica el cabezal de una de las cintas al principio de la palabra (y lo desplaza de izquierda a derecha) y en la otra al final (y lo desplaza de derecha a izquierda) comenzando a comparar los símbolos correspondientes que encuentra cuando recorre las dos cintas. Si en este proceso encuentran los cabezales un blanco acepta, si en algún momento los símbolos leídos por los dos cabezales difieren rechaza. Por lo tanto,  $L(M_I) = I$ .

En lo sigue, y como ya es habitual, el complementario de un lenguaje  $L$  lo representaremos como  $L^c$ .

- a) Si  $L$  es un lenguaje recursivamente enumerable, entonces  $P_R(L)$  también lo es.

Sea  $W = R^c \cap I$ , donde  $R^c$  (por lo dicho) es el complementario de  $R$ , y es un lenguaje recursivo por ser el complemento una operación de cierre en la familia lenguajes recursivos; así  $W$  es también un lenguaje recursivo por ser la intersección otra operación de cierre. Basta ahora ver, inmediatamente a partir del enunciado, que  $P_R(L) = L \cap W$ . Puesto que  $W$  por ser recursivo es también recursivamente enumerable y ser la intersección una operación de cierre en los lenguajes recursivamente enumerables se sigue que  $P_R(L)$  es un lenguaje recursivamente enumerable.

- b) Si  $L$  es un lenguaje recursivamente enumerable, entonces también lo es  $L - P_R(L)$ .

Basta notar que, a partir de lo anterior,

$$\begin{aligned} L - P_R(L) &= L - (L \cap W) = L \cap (L \cap W)^c = L \cap (L^c \cup W^c) = \\ &= (L \cap L^c) \cup (L \cap W^c) = \emptyset \cup (L \cap W^c) = L \cap W^c \end{aligned}$$

Nótese que  $W^c$  es un lenguaje recursivo.

17. Sean  $L_1, L_2, L_3$  lenguajes. Se define la operación  $P(L_1, L_2, L_3)$  que define el lenguaje formado por las palabras que pertenecen al menos a dos de sus argumentos.

- a) ¿Es la operación  $P$  una operación de cierre en la familia de los lenguajes recursivos?
- b) ¿Es la operación  $P$  una operación de cierre en la familia de los lenguajes recursivamente enumerables?

#### Solución

A partir de la definición puede verse que

$$P(L_1, L_2, L_3) = (L_1 \cap L_2) \cup (L_1 \cap L_3) \cup (L_2 \cap L_3).$$

Puesto que la intersección y la unión son operaciones de cierre tanto en los lenguajes recursivos como en los lenguajes recursivamente enumerables se tiene que la operación  $\mu$  es de cierre en ambas familias de lenguajes.

18. Sea  $\Sigma$  un alfabeto y sea  $<$  una relación de orden canónico definida sobre  $\Sigma^*$ . Sean dos lenguajes  $L, L' \subseteq \Sigma^*$ , se define la operación  $\mu(L, L')$  del modo que sigue:

$$\mu(L, L') = \{x \in L : (\exists y \in L')(y \leq x)\}.$$

- 1) Si  $L$  y  $L'$  son lenguajes recursivamente enumerables, entonces ¿lo es también  $\mu(L, L')$  ?
- 2) Si  $L$  y  $L'$  son lenguajes recursivos, entonces ¿lo es también  $\mu(L, L')$  ?

### Solución

Veamos que la operación binaria  $\mu$  es de cierre en ambos casos. Lo estableceremos mediante dos caminos diferentes: uno, utilizando propiedades de cierre y el otro, asociando las correspondientes máquinas de Turing.

Utilizando las propiedades de cierre procedemos como sigue. Si  $L' = \emptyset$ , entonces  $\mu(L, L') = \emptyset$ , que es tanto un lenguaje recursivo como recursivamente enumerable. Si  $L' \neq \emptyset$ , entonces sea  $\min_{\leq}(L')$  el mínimo elemento de  $L'$  de acuerdo al orden canónico dado  $<$ . Sea  $\Theta_{L'} = \{x \in \Sigma^* : x < \min_{\leq}(L')\}$  que, claramente, es un lenguaje finito y, en consecuencia, es también un lenguaje recursivo. Así, su complementario  $\Theta_{L'}^c = \Sigma^* - \Theta_{L'} = \{x \in \Sigma^* : \min_{\leq}(L') \leq x\}$  también es recursivo y, por tanto, también es recursivamente enumerable. Ahora puede verse que  $\mu(L, L') = L \cap \Theta_{L'}^c$ , puesto que

$$x \in \mu(L, L') \Leftrightarrow (x \in L) \wedge (\min_{\leq}(L') \leq x) \Leftrightarrow x \in L \cap \Theta_{L'}^c,$$

en consecuencia, puesto que los lenguajes recursivos, y también los recursivamente enumerables, son cerrados para la intersección, se tiene que la operación  $\mu(L, L')$  es de cierre tanto en la clase de los lenguajes reursivos como en la clase de los lenguajes recursivamente enumerables.

Otra manera de resolver la cuestión es definir las máquinas de Turing correspondientes en cada caso.

Para el caso 1) sea  $M$  una máquina de Turing tal que  $L(M) = L$ , y sea  $M'$  un generador de Turing tal que  $G(M') = L'$ . Definimos seguidamente una máquina de Turing  $M_\mu$  tal que  $L(M_\mu) = \mu(L, L')$ . Esta máquina opera como sigue. Cuando se le suministra una palabra de entrada  $x$  ésta se aplica a la máquina  $M$  y sólo en el caso de que ésta acepte se continúa arrancando el generador  $M'$  que estará controlado mediante una señal de continuación. Este generador cuando se arranca comienza el proceso generador y cuando genera la primera palabra se suspende hasta que recibe la señal de continuación en cuyo momento continuará el proceso generativo y seguirá así operando, mientras sea necesario, de acuerdo a este esquema. Cada vez que se genera una palabra y se comprueba si  $y \leq x$ ; en caso afirmativo se termina el proceso aceptando  $x$ , en caso negativo se envía al generador la señal de continuación. El test  $y \leq x$  se lleva a cabo en un módulo que siempre responde sí o no y opera como sigue: se examina en primer lugar si  $y = x$ , si son iguales, entonces se retorna sí, en otro caso se arranca un generador canónico de  $\Sigma^*$  asociado al orden  $<$  y se ve qué palabra aparece primero, si lo hace  $y$ , entonces  $y < x$  y se retorna sí; en otro caso, si primero aparece  $x$ , entonces  $y > x$  y se retorna no. En consecuencia, la máquina  $M_\mu$  así



definida cumple que  $L(M_\mu) = \mu(L, L')$  y, por tanto,  $\mu(L, L')$  es un lenguaje recursivamente enumerable.

Para el caso 2) sean  $A$  y  $A'$  dos máquinas de Turing que se detienen para cada entrada y que además cumplen que  $L(A) = L$  y  $L(A') = L'$ . Definimos seguidamente una máquina de Turing  $A_\mu$  que se detiene para cada entrada y tal que  $L(A_\mu) = \mu(L, L')$ . Esta máquina opera como sigue. Cuando se le suministra una palabra de entrada  $x$  ésta se aplica a la máquina  $A$ , si ésta rechaza entonces la palabra  $x$  se rechaza, y si ésta acepta se continúa arrancando un generador canónico de  $\Sigma^*$  con las mismas características que en el caso anterior. Cada vez que genera una palabra  $y$  se aplica a  $A'$ , si esta máquina acepta y entonces se acepta  $x$  (siempre que  $y < x$ ), si rechaza e  $y \neq x$  se envía al generador la señal de continuación; si  $y > x$  entonces se termina rechazando  $x$ .

19. Se define la operación  $P: \{a, b\}^* \rightarrow \mathcal{P}(\{a, b\}^*)$  del modo que sigue:

- I.  $P(\lambda) = \{\lambda\}$
- II.  $P(x) = a^*x$ , si  $x = by$
- III.  $P(x) = b^*x$ , si  $x = ay$

¿Es la operación  $P$  de cierre dentro de la clase de los lenguajes recursivos?

#### Solución

Veamos que la operación  $P$  es de cierre en la clase de los lenguajes recursivos. Sea  $\Sigma = \{a, b\}$ . Un modo inmediato de hacerlo es observar que

$$P(L) = (L \cap \{\lambda\}) \cup \{a\}^*(L \cap \{b\}\Sigma^*) \cup \{b\}^*(L \cap \{a\}\Sigma^*)$$

y que al ser  $\{\lambda\}, \{b\}\Sigma^*, \{a\}\Sigma^*, \{a\}^*\{b\}^*$  lenguajes recursivos (de hecho son regulares y todos los lenguajes regulares son lenguajes recursivos, ya que cada autómata finito determinista puede simularse, de modo directo, mediante una máquina de Turing cuyo cabezal nunca retrocede y que en consecuencia se detiene para cada entrada), y ser éstos cerrados para la unión, la intersección y la concatenación resulta que  $P(L)$  también es recursivo.

20. Dado un lenguaje  $L$  se define el lenguaje  $\mu(L) = \{x : (x = x^r) \wedge (x \in \text{prefijos}(L))\}$ . Si  $L$  es recursivamente enumerable, entonces ¿lo es también  $\mu(L)$ ?

#### Solución

Si el lenguaje  $L$  es recursivamente enumerable, entonces también lo es  $\mu(L)$ . Para demostrarlo, describimos una máquina de Turing con varias cintas  $M_L$  que lo reconoce. Puesto que  $L$  es recursivamente enumerable existe un generador de Turing  $G_L$  que lo genera y que se encuentra controlado durante la generación, en el sentido habitual, mediante una señal de control, de continuación, que le permite generar las palabras de una en una, con las correspondientes pausas, de acuerdo a esta señal.

La máquina  $M_L$  opera como sigue. Cuando se le aplica una entrada  $x$  comprueba primeramente que  $x = x^r$ , para ello copia la palabra  $x$  en una cinta auxiliar y vuelve al principio de  $x$  en la cinta de entrada recorriendo seguidamente  $x$  y  $x^r$  síncronamente en sentidos contrarios, comprobando que los respectivos símbolos examinados son iguales; si en ambas cintas encuentra simultáneamente un blanco, y sólo en estas circunstancias, da por satisfecha la condición, esto es:  $x = x^r$ .



Seguidamente, si esta condición se cumple, arranca el generador  $G_L$  que comienza la generación; siempre que genera una palabra se suspende hasta que recibe la señal de continuación; con la última palabra generada  $z$  comprueba que  $x$  es un prefijo de la misma, para ello escribe  $z$  en una cinta auxiliar y recorre desde el principio sincronamente  $x$  y  $z$  comprobando que los símbolos examinados coincidan; mientras esta coincidencia se dé continúa la comprobación que finaliza con la aceptación si se encuentra un blanco al examinar  $x$  (tanto si en  $z$  aparece un blanco como si no), en otro caso, si esta coincidencia de símbolos no se produce, se envía la señal de control al generador volviéndose a repetir esta fase.