Teorema: $L_1, L_2 \subseteq \Sigma^*$ y existe la reducción $L_1 \alpha L_2$, si $L_2 \in R \Rightarrow L_1 \in R$.

Dem.: $L_2 \in R \Rightarrow$ existe una máquina de Turing M_2 tq $L_2 = L(M_2)$ y M_2 siempre se detiene. Se construye M_1 que hace:

1) Simula M_f sobre w y obtiene f(w)

2) Simula M_2 sobre f(w) y acepta sii M_2 acepta

a) ¿$L_1 = L(M_1)$? porque $L_1 \alpha L_2$

$w \in L_1 \Rightarrow f(w) \in L_2 \Rightarrow M_2$ para en q_A con input $f(w) \Rightarrow M_1$ para en q_A con input $w \Rightarrow$

$w \in L(M_1)$

porque $L_1 \alpha L_2$

$w \notin L_1 \Rightarrow f(w) \notin L_2 \Rightarrow M_2$ para en q_R con input $f(w) \Rightarrow M_1$ para en q_R con input $w \Rightarrow$

$w \notin L(M_1)$

Por lo tanto, $L_1 = L(M_1)$

b) ¿M_1 se detiene siempre? Sí, pues M_f y M_2 se detienen siempre por hipótesis.

De a) y b) $L_1 \in R$.

Teorema: Sean L_1, $L_2 \subseteq \Sigma^*$ y la reducción $L_1 \alpha L_2$, se cumple que $L_2 \in RE \Rightarrow L_1 \in RE$.

Se demuestra de manera similar a la demostración del teorema anterior.

Corolario: Sean L_1, $L_2 \subseteq \Sigma^*$ y la reducción $L_1 \alpha L_2$, se cumplen que:

$$\left.\begin{array}{l} \text{Si } L_1 \notin R \Rightarrow L_2 \notin R \\ \\ \text{Si } L_1 \notin RE \Rightarrow L_2 \notin RE \end{array}\right\} \quad \text{Por las contrarrecíprocas}$$

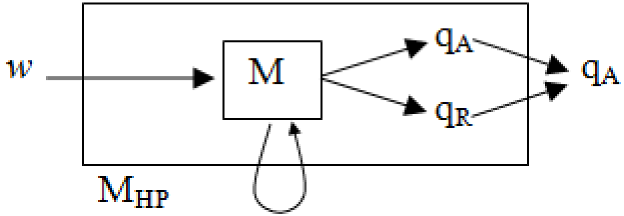
Ejercicio: sea el lenguaje Halting Problem

HP = {(<M>, w) tq M se detiene con input w}

demostrar que HP $\in$ (RE-R).

I) HP $\in$ RE.

Se construye una MT M_{HP} tq $L(M_{HP})$ = HP. M_{HP} chequea sintácticamente la entrada (<M>, w), si es un par inválido para en q_R, si es un par válido pero <M> es un código de MT inválido para en q_A, si ambos son válidos simula M sobre w, si M para (en q_A o en q_R) M_{HP} para en q_A, si M no para, M_{HP} no para.



II) HP $\notin$ R.

Se puede probar que existe la reducción $L_u \; \alpha$ HP, y como $L_u \notin$ R será cierto HP $\notin$ R.

Dem. Sea la siguiente máquina de Turing M_f que computa la función f de reducibilidad

$$M_f((\langle M\rangle,w))=(\langle M'\rangle,w)$$

M_f trabaja de la siguiente manera:

Si <M> no es un código válido de MT o (<M>,w) no es un par válido borra la cinta (deja λ como salida), de lo contrario busca en las quíntuplas de <M> el estado q_R y lo reemplaza por un nuevo estado q. Luego agrega las quíntuplas (q,0,q,0,S); (q,1,q,1,S); (q,B,q,B,S) para hacerla loopear. Es decir que la máquina construida por M_f loopea cuando M para en q_R.

Hay que demostrar que M_f es una máquina que computa la función de reducibilidad

1) f es computable pues M_f siempre se detiene pues la entrada es finita y luego de recorrerla agrega un número finito de quíntuplas y se detiene.

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

   a) si $(<M>,w) \in L_u \Rightarrow$ M acepta w $\Rightarrow$ M para en q_A. $\Rightarrow$ M' para en $q_A \Rightarrow$ M' se detiene con input w $\Rightarrow (<M'>,w) \in HP$

b) si $(<M>,w) \notin L_u \Rightarrow$

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

o bien

ii) M rechaza w $\Rightarrow$ M loopea o para en q_R con input w $\Rightarrow$M’ loopea con input w $\Rightarrow (<M'>,w) \notin HP$

Por a) y b) se tiene que $(<M>,w) \in L_u \Leftrightarrow (<M'>,w) \in HP$

dónde $(<M'>,w)$ es $M_f((<M>,w))$

de 1) y 2) se tiene que M_f computa la función de reducibilidad buscada y por lo tanto queda demostrado que $L_u \;\alpha\; HP$.

Como $L_u \notin R \Rightarrow HP \notin R$

De I) y II) $HP \in (RE\text{-}R)$.

Ejercicio: Probar que $L_{\Sigma*}$ = {<M> / L(M) = Σ*} no es recursivo.

Mostraremos que existe una reducción $L_u \, \alpha \, L_{\Sigma*}$

Hay que encontrar una función total computable tal que

$$(\langle M\rangle,w) \in L_u \Leftrightarrow f\,(\langle M\rangle,w) \in L_{\Sigma*}$$

Sea M_f la máquina de Turing que computa la función f (<M>,w) = <M'> y trabaja de la siguiente manera:

Si (<M>,w) no es un par válido o <M> no es un código de MT válido, M_f borra la cinta (deja λ como salida), de lo contrario M_f construye <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' para en q_A para cualquier input ⇔ M acepta w

1) $f((<M>,w))$ es computable? Claramente sí, pues M_f para luego de realizar una cantidad finita de acciones.

2) $(<M>,w) \in L_u \Leftrightarrow <M'> \in L_{\Sigma*}$?

a) Sea $(<M>,w) \in L_u \Rightarrow$ M para en q_A con input w $\Rightarrow$M' para en q_A con cualquier input $\Rightarrow <M'> \in L_{\Sigma*}$

b) Sea $(<M>,w) \notin L_u \Rightarrow$

i. Si $(<M>,w)$ no es un par válido o $<M>$ no es un código de MT válido $\Rightarrow <M'> = \lambda \Rightarrow <M'> \notin L_{\Sigma*}$

ii. M rechaza w $\Rightarrow$ M' rechaza todo input $\Rightarrow <M'> \notin L_{\Sigma*}$

De a) y b) se tiene que $(<M>,w) \in L_u \Leftrightarrow <M'> \in L_{\Sigma*}$

De 1) y 2) se tiene que $L_u \,\alpha\, L_{\Sigma*}$

Por lo tanto $L_{\Sigma*} \notin$ a R (porque $L_u \notin$ a R)

Nota: Puede demostrarse también que existe una reducción $\overline{L}_u \,\alpha\, L_{\Sigma*}$ y como $\overline{L}_u \notin$ RE se tiene que $L_{\Sigma*} \notin$ a RE

Para practicar. Demostrar que existe una reducción de $\overline{L}_u \,\alpha\, L_{\varnothing}$

con $L_{\varnothing} = \{<M>/\ L(M) = \varnothing\}$

Teorema. $L_{EQ} = \{(<M_1>,<M_2>) \,/\, L(M_1)=L(M_2)\} \notin RE$

Prueba: $L_{\Sigma*} \;\alpha\; L_{EQ}$

La función *f* de reducibilidad que computa M_f es

$f\,(<M>)=(<M>,<M_{\Sigma*}>)$ Siendo $<M_{\Sigma*}>$ el código de una máquina de Turing que acepta Σ*

Por ejemplo la δ de transición de $M_{\Sigma*}$ puede ser la siguiente:

$\delta(q_0,0)=(q_A,0,S);\ \delta(q_0,1)=(q_A,1,S);\ \delta(q_0,B)=(q_A,B,S)$

Claramente M_f se detiene, por lo tanto *f* es computable

Además:

$<M> \in L_{\Sigma*} \Leftrightarrow L(M) = \Sigma^* \Leftrightarrow L(M) = L(M_{\Sigma*}) \Leftrightarrow (<M>,<M_{\Sigma*}>) \in L_{EQ}$