

# INF155: Informática Teórica

## Clase 23: Reducción de Problemas

Aldo Berrios Valenzuela

Jueves 2 de Junio de 2016

### 1. Reduccion de Problemas

**Definición 1.1** (Problema). Lenguaje  $\subseteq \Sigma^*$ , determinar si  $\sigma$  pertenece o no. Interesan lenguajes infinitos (no tiene gracia una simple lista finita).

**Definición 1.2** (Reducción). Algoritmo que transforma  $\alpha \in \Sigma^*$  (el problema es ¿es  $\alpha \in A$ ?) en  $\beta \in \Gamma^*$  (problema es ¿es  $\beta \in B$ ?) tal que  $\alpha \in A \Leftrightarrow \beta \in B$ . Si hay una reducción se anota  $A \leq B$  (Figura 1).

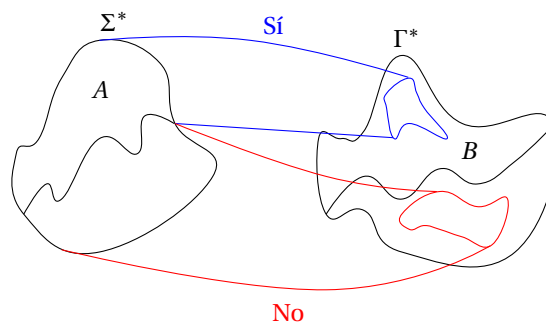


Figura 1: Representación gráfica de la reducción de problemas. En este caso, reducimos el problema A a instancias del problema B, donde B es a lo menos, tan difícil como A.

**Teorema 1.1.** Si  $P_1 \leq P_2$ , entonces:

- Si  $P_1$  no es decidible,  $P_2$  no es decidible.
- Si  $P_1$  no es recursivamente enumerable,  $P_2$  no es recursivamente enumerable.

**Demostración.** Por contradicción. Ejercicio: realmente entender esto. □

Un par de lenguajes útiles:

$$\left. \begin{array}{l} L_e = \{M : \mathcal{L}(M) = \emptyset\} \\ L_{ne} = \{M : \mathcal{L}(M) \neq \emptyset\} \end{array} \right\} \text{ Notar que } L_c = \overline{L_{ne}} \text{ (} e \rightsquigarrow \text{empty, } ne \rightsquigarrow \text{not empty)}$$

**Teorema 1.2.**  $L_{ne}$  es R.E.

**Demostración.** Usamos  $L_u$  (también es R.E.) (lenguaje aceptado por una máquina de Turing universal) en nuestra construcción (Figura 2). □

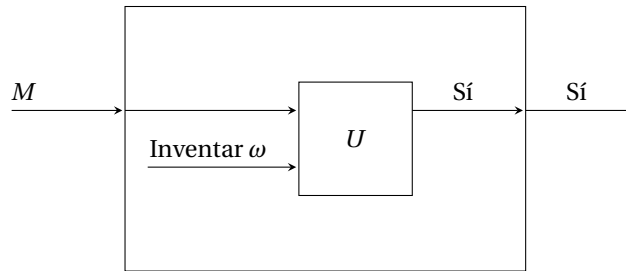


Figura 2: Inventamos una palabra  $\omega$  tal que  $U$  la acepte. Nótese que este  $\omega$  inventado representa nuestro no-determinismo.

**Teorema 1.3.**  $L_{ne}$  no es recursivo (=  $L_e$  no es recursivamente enumerable)

**Demostración.** Reducimos  $L_u$  a  $L_{ne}$  (la reducción de problemas también se aplica para deducir si un lenguaje es recursivamente enumerable o no). Sabemos que  $L_u$  es R.E., no recursivo

En castellano, dado  $\langle M, \omega \rangle$  en  $L_u$ , debo construir una TM que acepte el lenguaje vacío si  $M$  no acepta  $\omega$  y un lenguaje no vacío (por ejemplo  $\Sigma^*$ ) si  $M$  acepta  $\omega$ . Dados  $M$  y  $\omega$  puedo construir  $M'$  (Figura 3).

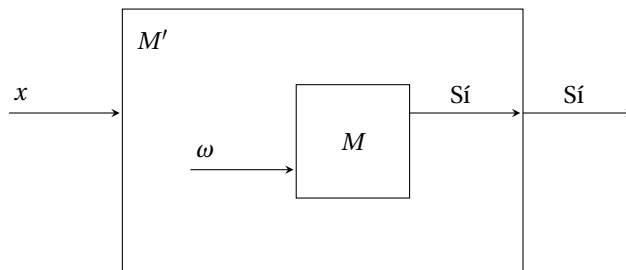


Figura 3: /\* Agregar alguna leyenda \*/

Esto es  $L_u \leq L_{ne}$ ;  $L_u$  no recursivo  $\Rightarrow L_{ne}$  no recursivo.

□

**Definición 1.3.** Una propiedad de los lenguajes es un conjunto de lenguajes.

Por ejemplo, “es de contexto libre” es el conjunto de lenguajes generados por CFG. “Es vacío” es el conjunto  $\{\emptyset\}$ .

**Definición 1.4.** Una propiedad se llama no trivial si hay lenguajes con la propiedad y lenguajes sin la propiedad.

Representamos un lenguaje recursivamente enumerable por una TM que lo acepta.

**Teorema 1.4 (Rice).** Toda propiedad no trivial de los lenguajes R.E. es no decidible. Un ejemplo: detectar si un lenguaje es regular, de contexto libre, etc. es no decidible

Si  $P = \{M : \mathcal{L}(M) \text{ tiene la propiedad } P\}$ , entonces  $L$  no es recursivo.