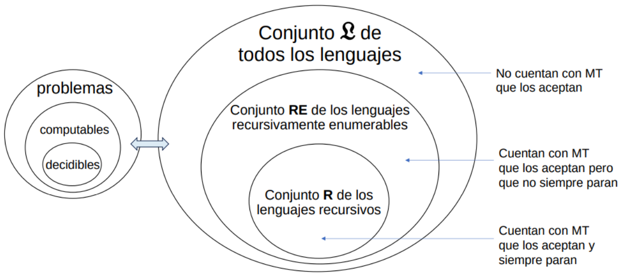
Clase 2

sábado, 15 de marzo de 2025 13:17

Posibilidades:

- MT que siempre paran
  - Computable decidible
  - L es recursivo
  - R es el conjunto de estos lenguajes
- A veces en casos negativos loopean, no paran
  - Computable no decidible (o semidecidible)
  - Recursivamente enumerable. Lo puedo generar
  - Re es el conjunto
- En casos positivos loopea. El minimo que pido es que en casos positivos pare
  - No computable
  - L no es recursivamente enumerable



Formalizando las definiciones

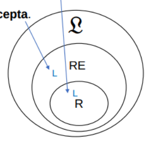
- **Σ** es el alfabeto universal de todos los símbolos: Σ = {a, b, …, 1, 2, …, +, −, …}
- **Σ*** es el conjunto universal de todas las cadenas de símbolos de Σ: Σ* = {λ, a, b, 1, …, aa, ab, a1, …, aaa, …}
- **𝓛** es el conjunto universal de todos los lenguajes con alfabeto Σ: conjunto de todos los subconjuntos de Σ*.
- Un lenguaje L es **recursivo** (L ∈ R) sii existe una MT M que lo **acepta y siempre para (lo decide).**

  Es decir, para toda cadena w de Σ*:

  - Si w ∈ L, entonces M a partir de w para en su estado q_A
  - Si w ∉ L, entonces M a partir de w para en su estado q_R
- Un lenguaje L es **recursivamente enumerable** (L ∈ RE) sii existe una MT M que lo **acepta**.

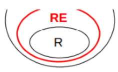
  Es decir, para toda cadena w de Σ*:

  - Si w ∈ L, entonces M a partir de w para en su estado q_A
  - Si w ∉ L, entonces M a partir de w para en su estado q_R o no para



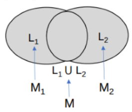
Se cumple por definición que **R ⊆ RE ⊆ 𝔏** (ejercicio)
Las inclusiones son estrictas: R ⊂ RE ⊂ 𝔏 (lo probamos en la próxima clase)

06



Propiedad 3. Si L_1 ∈ RE y L_2 ∈ RE, entonces $L_1 \cup L_2$ ∈ RE.

Es decir, si existe una MT M_1 que acepta L_1,
y existe una MT M_2 que acepta L_2,
también existe una MT M que acepta $L_1 \cup L_2$.



Prueba.

Idea general. ¿Construir una MT M que ejecute secuencialmente las MT M_1 y M_2 y acepte sii M_1 o M_2 aceptan?

No funca porque puede pasar que m2 acepte y m1 no pare a partir de w entonces M que deberia aceptar w no lo hace
Habria que hacerlo en paralelo(ejecutar un paso de m1 y uno de m2 alternadamente)
Si una dijo que no y la otra loopea, M no rechaza, pq puede ser que eventualmente terine y sea un si. Sigue loopeando
SI las dos dicen que no, rechazo
Con que una diga que si, ya está.
Si hago la intersección, sirve hacerlas secuencial.. Hay que cambiar cuando dice que si y cuando que no nomás

CORE: complementos de re (L RE sii LC CO-RE)

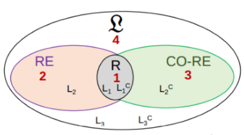
Conclusión: si un problema y el problema contrario son computables, entonces ambos son decidibles.

Tengo un lenguaje que está en re.

Región 1 (lenguajes aceptados por MT que siempre paran)
Conjunto R.
Si L está en R, entonces L^C está en R

Región 2 (lenguajes aceptados por MT que no siempre paran)
Conjunto RE – R.
Si L está en RE, entonces L^C está en CO-RE

Región 3 (lenguajes no aceptados por MT, con complementos aceptados por MT que no siempre paran)
Conjunto CO-RE – R.
Si L está en CO-RE, entonces L^C está en RE



Las cuatro regiones de la jerarquía, con grado de dificultad creciente con grado de dificultad creciente.

Región 4 (lenguajes no aceptados por MT, con complementos no aceptados por MT)

Conjunto $\mathfrak{L}$ – (RE U CO-RE).

Si L está en $\mathfrak{L}$ – (RE U CO-RE), entonces L^C está en $\mathfrak{L}$ – (RE U CO-RE)

Los lenguajes de co-re tienen algunas mt que las aceptan (solo los de r). El caso más difícil es el de la región 4. ninguno tiene una máquina que lo reconoce.

Del 4 no podemos saber nada. No se pueden resolver con mt. EN el 3 no los podemos saber pero si el completento.