Linguaggi ricorsivi e ricorsivamente enumerabili

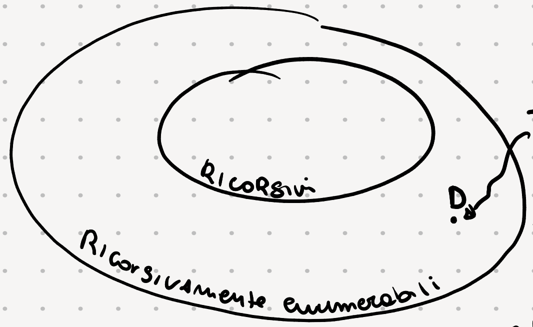
<u>Def</u>: Un linguaggio L é ricorsivo quando esiste un algoritmo w t.c.

$$F_w(x) = \begin{cases} 1 & \text{se } x \in L \\ 0 & \text{se } x \notin L \end{cases}$$

<u>Def</u>: Un linguaggio L é ricorsivamente enumerabile quando esiste una procedura w t.c.

$$F_w(x) = \begin{cases} 1 & \text{se } x \in L \\ \uparrow & \text{se } x \notin L \end{cases}$$

GRAZIE ALL'ESISTENZA DI D
LE DUE CLASSI NON COINCIDONO



- D é ricorsivamente enum.
- D non é ricorsivo

• D^c

<u>Def</u>: L'INTERPRETE é un programma "u" $\in \{0,1\}^*$

Input: passiamo ad u la coppia
(programma, dato)

Output: il risultato del "programma" su quel dato

x passare + input

$$F_u(w\$x) = \begin{cases} F_w(x) & \text{se } w \text{ é programma} \\ \perp & \text{se } w \text{ \underline{non} é programma} \end{cases}$$

interprete — prog — dato

dove $w\$x \in \{0,1\}^*$

Definizioni:

$$D = \{ x \in \{0,1\}^* \mid F_u(x\$x)\downarrow \}$$

$x\$x \in \{0,1\}^*$ è quindi lecito passarlo ad "u"

D è detto Linguaggio dell'arresto ristretto

$$D^c = \{ x \in \{0,1\}^* \mid F_u(x\$x)\uparrow \}$$

D^c è il complemento del linguaggio dell'arresto ristretto

<u>Teorema</u>: 1) D è un linguaggio ricorsivamente enumerabile

2) D non è ricorsivo

3) D^c non è ricorsivamente enumerabile

<u>Dimostrazione</u> 1) Devo esibire per D una procedura P t.c.

$$F_P(x) = \begin{cases} 1 & \text{se } x \in D \ ☺ \\ \uparrow & \text{se } x \notin D \ ☹ \end{cases}$$

☺ ovvero quando $F_u(x\$x)\downarrow$

☹ ovvero quando $F_u(x\$x)\uparrow$

Posso costruire la seguente procedura

```
Procedura RICNUM (x ∈ {0,1}*)
{
    y = F_u(x$x)
    return 1
}
```

u esiste sempre, quindi fattibile

Vediamo che RICNUM è proprio la procedura P che cercavamo dim. di correttezza di RICNUM:

$x \in D \Rightarrow F_u(x \$ x)\downarrow \Rightarrow$ viene fatta l'istr. $y = \dots$

$\Rightarrow$ viene poi eseguito return 1

$\Rightarrow$ RICNUM$(x) = 1$

$x \notin D \Rightarrow F_u(x \$ x)\uparrow \Rightarrow$ loop

$\Rightarrow$ return (1) non viene eseguita

$\Rightarrow$ RICNUM$(x)\uparrow$

$\Downarrow$

RICNUM(x) é la procedura p che cercavamo

$\Downarrow$

D è ricorsivamente enumerabile

Dimostrazione 2) D non é ricorsivo

Si dimostra per ASSURDO,

supponendo che D sia ricorsivo

posso costruire un programma che mi porterà ad una CONTRADDIZIONE

posso farlo perché

$\in \{0,1\}^*$

```
Procedura AssurdoA (x ∈ {0,1}*)
{
   if (x ∈ D) then return(1 - F_u(x $ x))
              else return(0)
}
```

Tutte le istruzioni sono implementabili

In particolare $x \in D$ lo é perché sotto ipotesi che D é ricorsivo

$\Downarrow$

esiste "e" codice binario

ASSURDOA(x)

Passo "e" codice di ASSURDOA(x)
la parola binaria e, mi chiedo:
quanto vale ASSURDOA(e)?

-> Secondo alcune considerazioni ottengo
la seguente risposta:

$$(\#)\quad ASSURDOA(e) \underset{\text{per def. di } e}{=} F_e(e) \underset{\text{per def. di } u}{=} F_u(e\$e)$$

quindi ho ottenuto

$$AssurdoA(e) = F_u(e\$e)$$

-> Esiste un altro modo per mostrare
quanto vale ASSURDOA(e)

Considerando ora il codice di
ASSURDOA ottengo un altro risultato:

$$\boxed{\text{caso } e \in D} \underset{\text{per def. } D}{\Rightarrow} F_u(e\$e)\downarrow \begin{cases} 1 \\ 0 \end{cases}$$

$$F_u(e\$e) \underset{\text{per eq. }\#}{=} ASSURDOA(e) \underset{\text{per def. ASSURDOA}}{=} 1 - F_u(e\$e)$$

ottengo così

$$F_u(e\$e) = 1 - F_u(e\$e)$$

ma siamo nel caso $F_u(e\$e)\downarrow \begin{cases} 1 \\ 0 \end{cases}$

ottengo

$$\text{contraddizione}\begin{cases} 1 = 1-1 = 0 & -F_u(e\$e) = 1 \\ 0 = 1-0 = 1 & -F_u(e\$e) = 0 \end{cases}$$

$\boxed{\text{caso } e \notin D} \Rightarrow F_u(e\$e)\uparrow$

def. D

$F_u(e\$e) = \text{ASSURDOA}(e) = 0$

per def. ASSURDA

ma siamo nel caso in cui $F_u(e\$e)\uparrow$

per cui si ottiene

$\uparrow = 0$ } contraddizione

$\Downarrow$

allora e non può esistere

perché non sta in D e nemmeno

fuori D

$\Downarrow$

Non posso implementare "$x \in D$?"

$\Downarrow$

D non è ricorsivo!