

# LINGUAGGI FORMALI e AUTOMI

DOCENTE:

PALANO BEATRICE

homepage:

palano.di.unimi.it

email:

palano@di.unimi.it

riavvimento:

SU APPUNTAMENTO

4011 IV piano via Celoria 18

PROGRAMMA DEL CORSO:

cosa Tratteremo?

Partiamo dal titolo del corso:

LINGUAGGI FORMALI E AUTOMI



GRAMMATICHE

LINGUAGGIO: insieme di frasi per la comunicazione tra entità diverse

- Esempi:
- i linguaggi naturali: italiano, inglese,  
Tra le persone  
....
  - i linguaggi di programmazione:  
java, c, c++, python, go ...  
Tra uomo e macchina
  - CODICE MORSE: una parola viene codificata con punti e linee

Per conoscere un linguaggio lo bisogna:

- dei vocaboli, o parole  
es: per l'italiano, il vocabolario italiano  
per il C, le parole chiavi del C
- sintassi: regole per costruire  
es: per l'italiano, le frasi  
per il C, i programmi

FORMALI: consideriamo questi concetti in  
maniera precisa usando la  
MATEMATICA

VANTAGGIO: uso di dispositivi elettronici  
per automatizzare operazioni

Esempi positivi:

compilatori



interpreti



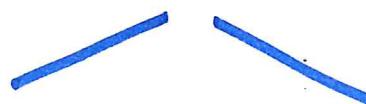
per la Traduzione  
dei programmi

per l'esecuzione  
dei programmi

Esempi negativi:

Traduzioni di linguaggi naturali

linguaggio parlato  
Segnale continuo



linguaggio scritto  
caratteri di spaziatura

caso particolare: il linguaggio dei messaggi  
nelle stazioni ferroviarie < O.K.

ma per l'intero linguaggio abbiamo un problema:  
**AMBIGUITÀ**

Per specificare un linguaggio ho bisogno di  
**UN SISTEMA FORMALE**

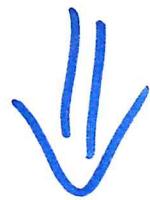
Sistemi generativi  
vocaboli + regole  
**GRAMMATICHE**

Sistemi riconoscitivi  
macchine a stati finiti  
**AUTOMI**

Altra visione:

possiamo pensare ad un linguaggio come:

LINGUAGGIO = UN PROBLEMA  
FORMALE DI DECISIONE  
SI / NO



Programma

① Concetti base della Teoria dei linguaggi

Elementi della Teoria della calcolabilità

- linguaggi ricorsivi
- linguaggi ricorsivamente enumerabili

Esistenza di un problema non trattabile

GRAMMATICHE: classificazione di Chomsky

②

Linguaggi regolati:

G di tipo 3, automi a stati finiti  
e espressioni regolari  
minimizzazione di automi  
monotermismo

③

Linguaggi acointestuali:

G di tipo 2, automi a pila  
ambiguità  
pumping lemma

# Concetti centrali nella Teoria dei ling. formali

Alfabeto: insieme finito di simboli

$$\Sigma = \{a_1, a_2, \dots, a_k\}$$

Parola su  $\Sigma$ : sequenza finita di simboli

Esempio: Parole

$$\Sigma = \{a\} \quad a, aa, aaaa, \dots$$

$$\Sigma = \{0, 1\} \quad 0, 1, 000, 010110$$

$$\Sigma = \{A, C, G, T\}$$

basi orotate

$10^6$   
↓  
DNA

ADENINA, CITOSINA, GUANINA, TIMINA

Caso particolare :  $\epsilon$   
parola vuota

Lunghezza di una parola :

numero di simboli nella parola

Parola	Lunghezza
$w$	$ w $

Esempio

001011  $|001011| = 6$

Qual è la lunghezza di  $\epsilon$  ?

$|\epsilon| = 0$

$\Sigma^*$  = insieme delle parole su  $\Sigma$   
compresa  $\epsilon$

$\Sigma^+$  = insieme delle parole su  $\Sigma$   
esclusa  $\epsilon$

$$\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$$

Concatenazione  $\sigma$  PRODOTTO  
di  
GIUSTA PPOSIZIONE

Date  $x, y \in \Sigma^*$  si dice prodotto  
la parola

$$z = x \cdot y = x_1 x_2 \dots x_k y_1 y_2 \dots y_{k'}$$

dove  $x = x_1 x_2 \dots x_k$  e  $y = y_1 y_2 \dots y_{k'}$

Proprietà del  $\cdot$ :

- chiuso rispetto a  $\Sigma^*$
- associativo

$$(x \cdot y) \cdot z = x \cdot (y \cdot z)$$

- elemento neutro

Domande:

① <sup>qual</sup> cosa è la lunghezza del prodotto?

$$|x \cdot y| = |x| + |y|$$

② chi è l'elemento neutro?

e.t.c.

$$e \cdot x = x \cdot e = x$$

$$e = \epsilon$$

Allora  $(\Sigma^*, \cdot, \varepsilon)$

è un monoidale dove

$\Sigma^*$  = insieme di parole

$\cdot$  = operazione binaria associativa

$\varepsilon$  = l'elemento neutro

Esempio :

$(\mathbb{N}, +, 0)$

$(\mathbb{N}, \times, 1)$

differenza?

Risp.

i due monoidi indicati

Sono entrambi commutativi

$$\forall x, y \quad x \text{ op } y = y \text{ op } x$$

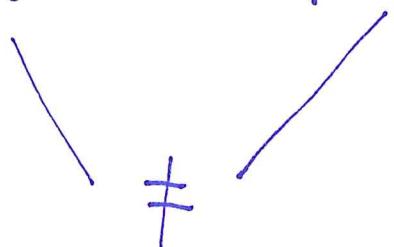
mentre

$$(\Sigma^*, ., \varepsilon) \quad \text{no}$$

Infatti

$$x = 12 \qquad \qquad y = 80$$

$$x \cdot y = 1280 \qquad y \cdot x = 8012$$



# Scomposizione di parole

esempio:



Definizioni formali:  $w, x \in \Sigma^*$

• prefisso:

$w$ ,  $x$  è prefisso di  $w$ ?

Si dice che  $x$  è prefisso di  $w$   
quando

$$w = x \cdot y$$

per una qualche parola  $y$ . ( $y \in \Sigma^*$ )

- suffisso

Si dice che  $x$  è suffisso di  $w$  quando

$$w = y \cdot x$$

per una qualche parola  $y$ . ( $y \in \Sigma^*$ )

- fattore

Si dice che  $x$  è fattore di  $w$  quando

$$w = y \cdot x \cdot z$$

per qualche  $y, z$ . ( $y, z \in \Sigma^*$ )

Caso particolare:

Gisano parole che sono contemporanea M.  
prefisso, suffisso, e fattore?

Risp. Data  $w \in \Sigma^*$  le parole:

- $\epsilon$
- $w$

Sono contemporaneamente

prefisso, suffisso, e fattore di  $w$

Definizione di linguaggio formale

Un linguaggio  $L$  è un qualunque  
sottoinsieme di  $\Sigma^*$ :

$$L \subseteq \Sigma^*$$

casi particolari:

- linguaggio vuoto:  $L = \emptyset$
  - linguaggio della parola vuota:  $L = \{\epsilon\}$

# Linguaggi

$$L \subset \Sigma^*$$

FINITI

INFINITI

Esempi di L finiti:

- $L = \emptyset$  .  $|L| = 0$
  - $L = \{\varepsilon\}$  .  $|L| = 1$

$$- \Sigma = \{a\}$$

$$L_n = \{\varepsilon, a, aa, aaa, \dots, \underbrace{a \cdots a}_{\downarrow \downarrow} \}$$

n lettere a  
 $a^n$

$$|L_n| = n+1$$

## - VOCABOLARIO DELL' ITALIANO

$$\Sigma = \{a, b, c, \dots, z\}$$

$$V = \{a, abac0, \dots, zuzzerellone, zzz\}$$

$$|V| \approx 200\cdot 000$$

Esempi di linguaggi INFINITI:

- $I = \{a\}$

$$L = \{\varepsilon, a, aa, \dots, a^n, \dots\} = \Sigma^* = \{a\}^*$$

$$\{a\}^* = \{a^n \mid n \in \mathbb{N}\}$$

dove:  $a^0 = \varepsilon$

- Espressioni booleane E

$$\Sigma = \{0, 1, \wedge, \vee, \neg, (, )\}$$

E è così definito:

- $0, 1 \in E$

- Se  $x, y \in E$  allora:

$$\left. \begin{array}{l} - (x \wedge y) \\ - (x \vee y) \\ - \neg x \end{array} \right\} \in E$$

- nient' altro appartiene ad  $E$

Esempi :

- parole in  $E$ :

$0, 1, (0 \wedge 1)$

$((0 \wedge 1) \vee 0)$

$\neg((0 \wedge 1) \vee 0)$

- parole non in  $E$ :

$\varepsilon, (,$  (0  $\wedge$  1

$(0 \wedge$  )

$((\dots) \dots$

$(00 \wedge 10)$

# OPERAZIONI SUI LINGUAGGI

•) Insiemistiche:  $A, B \subseteq \Sigma^*$

- UNIONE

$$A \cup B = \{ \omega \in \Sigma^* \mid \omega \in A \vee \omega \in B \}$$

- INTERSEZIONE

$$A \cap B = \{ \omega \in \Sigma^* \mid \omega \in A \wedge \omega \in B \}$$

- COMPLEMENTO

$$A^c = \{ \omega \in \Sigma^* \mid \omega \notin A \}$$

Esercizi :

- $A = \text{numeri binari escluso lo zero}$   
 $= \text{parole binarie con prefisso 1}$

$$A = 1\{0,1\}^*$$

$$B = 0\{0,1\}^*$$

$$C = \{0,1\}^* 0$$

$$- A \cup B = 1\{0,1\}^* \cup 0\{0,1\}^*$$

$$A \cup B \stackrel{?}{=} \Sigma^* = \{0,1\}^*$$

no, manca  $\epsilon$

$$A \cup B = \Sigma^+ = \{0,1\}^+$$

- $A \cap B = \emptyset$
  - $A \cap C = 1\{0,1\}^* \cap \{0,1\}^* 0$   
 $= 1\{0,1\}^* 0$
  - $A^c = (1\{0,1\}^*)^c$   
 $= 0\{0,1\}^* \cup \{\varepsilon\} = B \cup \{\varepsilon\}$
  - $A = a^* = \{a^n \mid n \in \mathbb{N}\} = \{a\}^*$
- $A^c = ?$
- $\Sigma = \{a\}$
- $A^c = \{w \in \{a\}^* \mid w \notin A\} = \emptyset$

$$\Sigma = \{a, b\}$$

$$A^c = \{ w \in \{a, b\}^* \mid w \notin A \}$$

= { parole su  $\{a, b\}$  che hanno almeno una  $b$  }

$$b^* \cup a \cap \dots ?$$

$$b \{a, b\}^* \cup \{a, b\}^* b [aba \text{ è esclusa}]$$

$$= \boxed{\{a, b\}^* \cdot b \cdot \{a, b\}^*}$$

) operazioni tipiche sui linguaggi:

- PRODOTTO

$$A \cdot B = \{ xy \in \Sigma^* \mid x \in A \text{ e } y \in B \}$$

- POTENZA

$$L \subseteq \Sigma^*$$

$$L^K = \underbrace{L \cdot L \cdot L \cdot \dots \cdot L}_{K\text{-volte}}$$

$$\left\{ \begin{array}{l} L^0 = \{\epsilon\} \\ L^K = L \cdot L^{K-1} \end{array} \right.$$

$\leftarrow$  def  
RICORSIVA

- CHIUSURA DI KLEENE : \*, +

$$L^* = L^0 \cup L^1 \cup L^2 \cup \dots \cup L^k \cup \dots$$

$$= \bigcup_{k=0}^{\infty} L^k$$

$$L^+ = L^1 \cup L^2 \cup \dots \cup L^k \cup \dots$$

$$= \bigcup_{k=1}^{\infty} L^k$$

ESERCIZIO

$$L^+ \stackrel{?}{=} L^* \setminus \{\varepsilon\}$$

Esercizi :

$$- A = \{ \text{mo, se} \}$$

$$B = \{ \text{ra, re} \}$$

$$A \cdot B = \{ \text{mora, more, sera, sere} \}$$

$$B \cdot A = \{ \text{ramo, rase, remo, rese} \}$$

$$A \cdot B \neq B \cdot A$$

- numeri binari:

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

$$A = \{1\} \quad \text{e} \quad B = \{0,1\}^*$$

$$A \cdot B = \{1\} \cdot \{0,1\}^* = 1 \{0,1\}^*$$

-  $L = \{a, b\}$

$$L^K = L \cdot L \cdot L \cdot \dots \cdot L$$

$$= \left\{ w \in \{a, b\}^* \mid |w| = K \right\}$$

$\Sigma^*$  ?  
 $\Sigma^* \doteq$  insieme delle parole su  $\Sigma$  con  $\epsilon$

$$\hookrightarrow = \bigcup_{k=0}^{\infty} \Sigma^k$$

$\Sigma^k$  = parole su  $\Sigma$  di lunghezza  $k$

$$\hookrightarrow = \underbrace{\Sigma^0}_{\{\epsilon\}} \cup \underbrace{\Sigma^1}_{\sum} \cup \underbrace{\Sigma^2}_{|w|=2} \cup \dots \cup \underbrace{\Sigma^k}_{|w|=k} \cup \dots$$

= O.K.

- $A = \{bb\}$        $\Sigma = \{b\}$
- $$A^* = \left\{ w \in \{b\}^* \mid |w| = 2n, n \geq 0 \right\}$$
- $$= \left\{ b^{2n} \mid n \in \mathbb{N} \right\}$$

- $A = \{b, bb\}$

$$A^* = \{b\}^*$$

$$A^+ = \{b\}^+$$

OSSERVAZIONE:

$bbb$   
 $\downarrow$   
decomposizione  
in  $A$

$\left\{ \begin{array}{l} b \cdot b \cdot b \\ bb \cdot b \\ b \cdot bb \end{array} \right.$

$A$   
NON  
é  
un codice!

## Definizione:

Un linguaggio  $L$  è un codice quanolo:

ogni parola in  $L^+$   
è decomponibile in un unico modo in parole di  $L$

Esempio:

$$L = \{ aa, ab, b \}$$

$$ab|aa|b|aa|ab \in L^+$$

$L$  è un codice

Definizione :

$L$  è un codice PREFISSO

o ISTANTANEO quando :

- è un codice

- ogni parola di  $L$  non è prefisso di altre parole di  $L$

Proprietà dei codici prefissi :

Ammettono un algoritmo di decodifica on-line (istantaneo)

Esempi :

{aa, ab, b}



PREFISSO

{0, 01}



NON PREFISSO

## CODICE ASCII ESTESO

codifica ogni carattere della Tastiera:

$$\{ A, \dots, Z, a, \dots, z, 0, 1, \dots, 9, \dots, \{, \}, \dots \}$$

in sequenze di 8 bit:

$$C_A = \{ x \in \{0,1\}^* \mid |x| = 8 \}$$

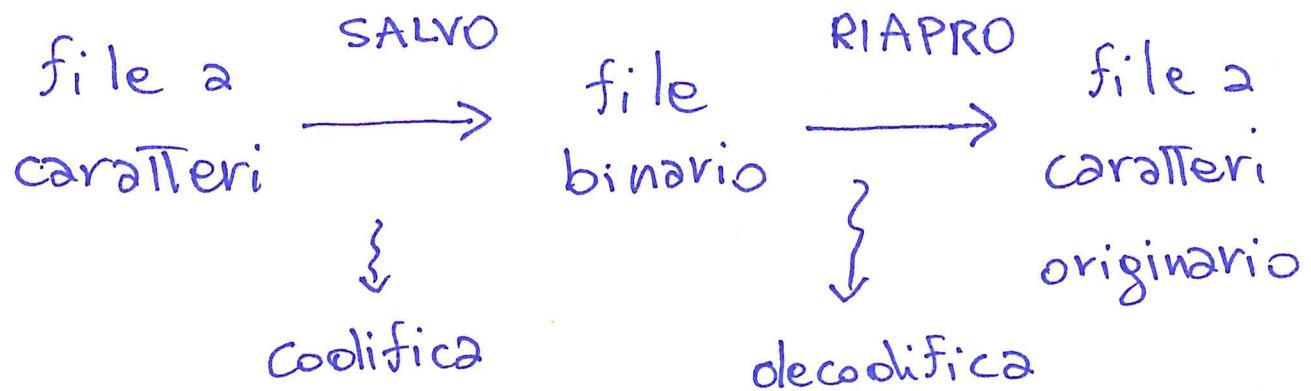
chi è  $C_A^+$ ?

$C_A^+ = \text{"file binari"}$

Perché è importante che

$C_A$  sia un codice?

Risp.



- se CA non fosse un codice non otterei il file a caratteri originario
- ancora CA è prefisso:  
algoritmo di decodifica on-line:  
taglia il file binario ogni 8 bit!

# LINGUAGGIO = PROBLEMA

Definizione di Linguaggi:

•)  $L = \{ w_1, w_2, \dots, w_n \}$

solo se  $L$  è finito

ESTENSIVO

•)  $L = \{ w \in \Sigma^* \mid P(w) = 1 \}$

o dove  $P$  è una proprietà

INTENSIVO

vale per ogni  $L$

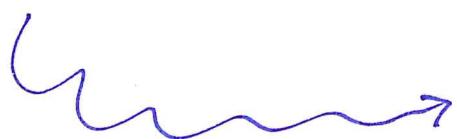
anche infinito

Fatto:

Ad ogni  $L$  è associato il problema  $P_L$

Linguaggio

$$L = \{w \in \Sigma^* \mid P(w) = 1\}$$



ASSOCIAZIONE

Problema  $P_L$ :

[ INPUT:  $x \in \Sigma^*$   
OUTPUT:  
 $x$  soddisfa  $P$ ? ]

↓  
si      ↓  
NO

{  
è un problema  
di DECISIONE

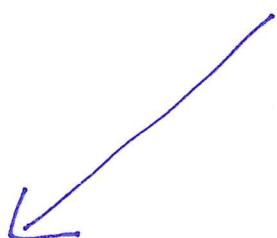
Dato il problema  $P_L$  siamo interessati:

2 :

- 1) sapere se  $P_L$  ammette una soluzione automatica.
- 2) Se  $P_L$  ammette un algoritmo trovare quello migliore.

Tutto questo si traduce in :

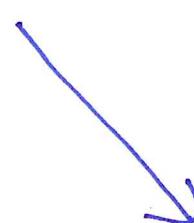
- 1) sapere se  $L$  ammette un sistema formale



sistema generativo



genero le parole  
di  $L$



sistema riconoscitivo



stabilisco  
 $w \in L$

2) Se  $L$  ammette un sistema riconoscitivo, allora trovare quello migliore

Esempio

-  $I = \text{linguaggio degli indirizzi internet}$

$$= \left\{ x \in \{0, 1, *\}^* \mid x = x_1 \cdot x_2 \cdot x_3 \cdot x_4 \right.$$

$$\text{dove } x_i \in \{0, 1\}^* \text{ e } |x_i| = 8 \}$$

importanti software in rete  
deve stabilire la correttezza  
degli indirizzi internet



TROVARE  
UN

$x \in I ?$  = SISTEMA  
RICONOSCITIVO  
PER I

- $\hat{P}$  = linguaggio delle password chi  
un siTo Web
- $$= \left\{ w \in \{a, \dots, z, A, \dots, Z, 0, 1, \dots, 9\}^* \mid \begin{array}{l} |w|=8 \wedge \exists i \quad w_i \in \{A, \dots, Z\} \\ \wedge \exists j \quad w_j \in \{0, \dots, 9\} \end{array} \right\}$$

si richiede che le password  
siano generate in maniera sistematica



TROVARE UN SISTEMA  
GENERATIVO PER  $\hat{P}$

Domanda:

tutti i problemi di decisione  
ammettono soluzione automatica?

Risultato indipendente dalla Tecnologia

Teoria della calcolabilità  
( concetti base )

procedura: sequenza finita di istruzioni  
che possono portare ad un risultato

algoritmo: è una procedura che  
termina su ogni input

Cos'è un programma?

- 
- ① ASPECTO SINTATTICO
  - ② ASPECTO SEMANTICO

① Un programma è una parola binaria:

$w \in \{0,1\}^*$  grazie al codice  
ASCII

② È risultato ottenuto dall'esecuzione  
una funzione:

$F_w$   $\rightsquigarrow$  la semantica di  $w$

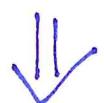
$F_w(x)$  = risultato del programma  
 $w$  su input  $x$

### OSSERVAZIONE

L'input è una parola binaria

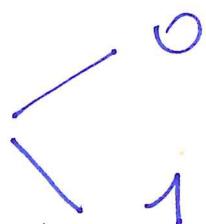
$x \in \{0,1\}^*$

ma anche  $w \in \{0,1\}^*$



quindi anche  $w$  può essere  
un input

Limitazione sui programmi:

- σ non terminano
- σ se terminano mi danno  
in uscita 

Notazione:

$$F_w(x) \uparrow$$

w su input x  
non termina

$$F_w(x) \downarrow$$

w su input x  
termina

$$F_w(x) = 1 \quad \text{se l'output è 1}$$

$$F_w(x) = 0 \quad \text{se l'output è 0}$$

Esempio:

Problema:

calcolare la parità dei numeri binari positivi

$$(x \in 1\{0,1\}^*)$$

Programma:

bin-parità ( $x = x_1 x_2 \dots x_n \in \{0,1\}^*$ )

```
if (x1=1) then return 1-xn;  
" "  
    -> xn  
loop;  
}
```

$$F_{\text{par}}(x) = \begin{cases} 1 & \text{se } x \text{ è numero binario pari} \\ 0 & \text{se } x \text{ è numero binario dispari} \end{cases}$$

$F_{\text{par}}(x) \uparrow$  se  $x$  è non binario

$$F_v(x) = \begin{cases} 1 & \text{se } x \in 1\{0,1\}^* 0 \\ 0 & \text{se } x \in 1\{0,1\}^* 1 \cup \{1\} \\ \perp & \text{se } x \in 0\{1,0\}^* \cup \{\varepsilon\} \end{cases}$$

$F_v \leftarrow$  semantica del programma  $v$   
 è una funzione:

$$F_v : \{0,1\}^* \rightarrow \{0,1,\perp\}$$

$\uparrow$   $\uparrow$   
 INPUT OUTPUT

Definizione:

La funzione caratteristica di  $L$  è:

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

esiste sempre per ogni  $L$ !

Definizione:

Un linguaggio  $L$  è detto RICORSIVO quando

esiste un paio algoritmo  $w$  t.c.:

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

Inoltre:

Se  $L$  è ricorsivo allora:

- $P_L$  è oggetto decidibile
- $L$  ammette un sistema riconoscitivo

ESEMPI

problemi decidibili:

- numeri pari
- numeri primi

linguaggi ricorsivi:

- $a^* b^*$
- $\{a^m b^m \mid m > 0\}$

## Definizione:

Un linguaggio  $L$  è RICORSIVAMENTE ENUMERABILE

quando

esiste una procedura  $w$  t.c. :

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

Se  $L$  è ricorsivam. enum. allora:

- $P_L$  è detto semidecidibile
- $L$  ammette un SISTEMA GENERATIVO

Relazione:

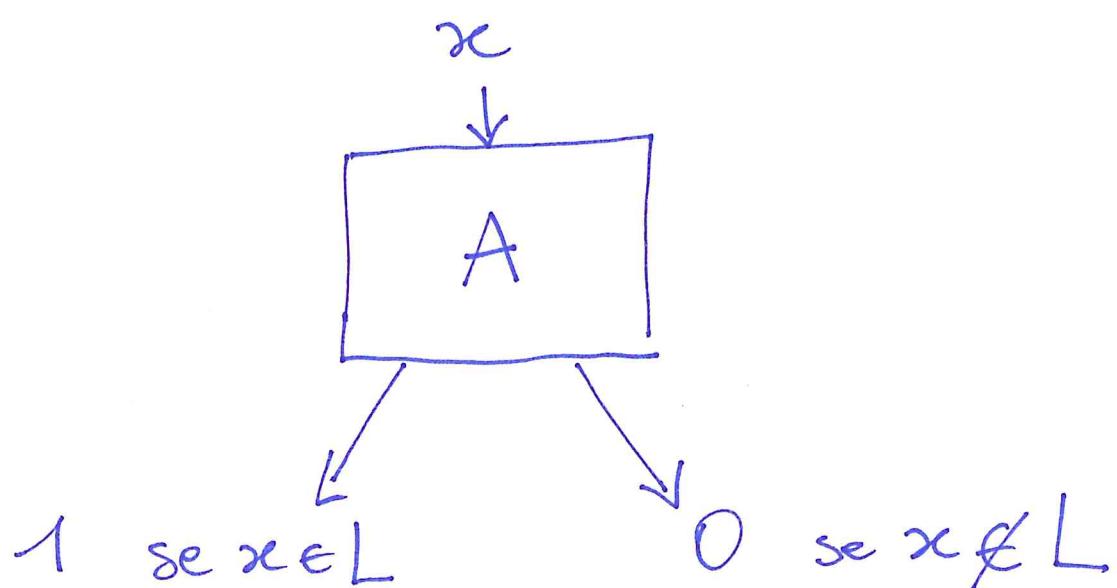


### Teorema

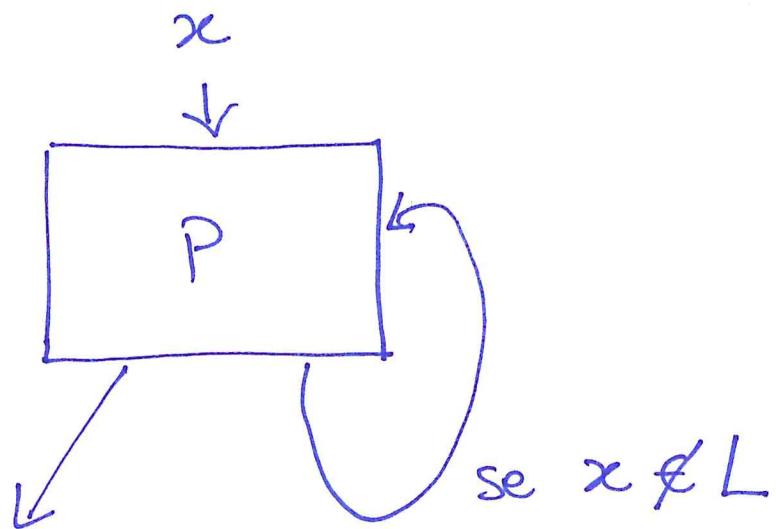
Se  $L$  è RICORSIVO  $\Rightarrow L$  è RICORSIV. ENUMER

dim.

Per ipotesi esiste un algoritmo  $A$  per  $L$ :



devo dimostrare l'esistenza di  
una procedura  $P$ :



1 se  $x \in L$

Procedura  $P(x)$

{      $y = A(x); \quad // A \text{ esiste}$   
      perché  $L$  è ricorsivo

if ( $y = 1$ ) Then return (1);

loop;

}

dim. di correttezza:

$$\underbrace{x \in L}_{\text{true}} \Rightarrow A(x) = 1 \Rightarrow y = 1 \Rightarrow \underbrace{P(x) = 1}_{\text{true}}$$

$$\underbrace{x \notin L}_{\text{true}} \Rightarrow A(x) = 0 \Rightarrow y = 0 \Rightarrow \underbrace{P(x)}_{\text{true}} \uparrow$$

segue che

$L$  è ricorsivamente enumerabile

Domanda:

È vero il viceversa?

Risposta: No,

intuitivamente non sempre  
si può trasformare una  
procedura in un algoritmo

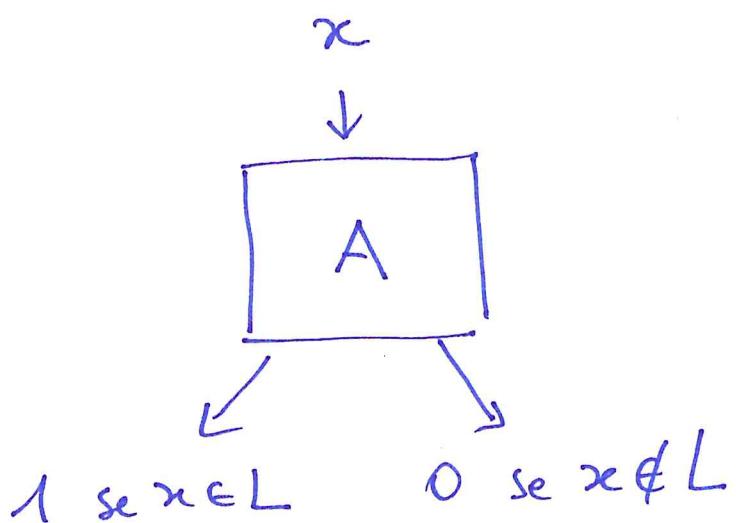
Proprietà dei linguaggi ricorsivi:

Teorema:

Se  $L$  è ricorsivo  $\Rightarrow L^c$  è ricorsivo

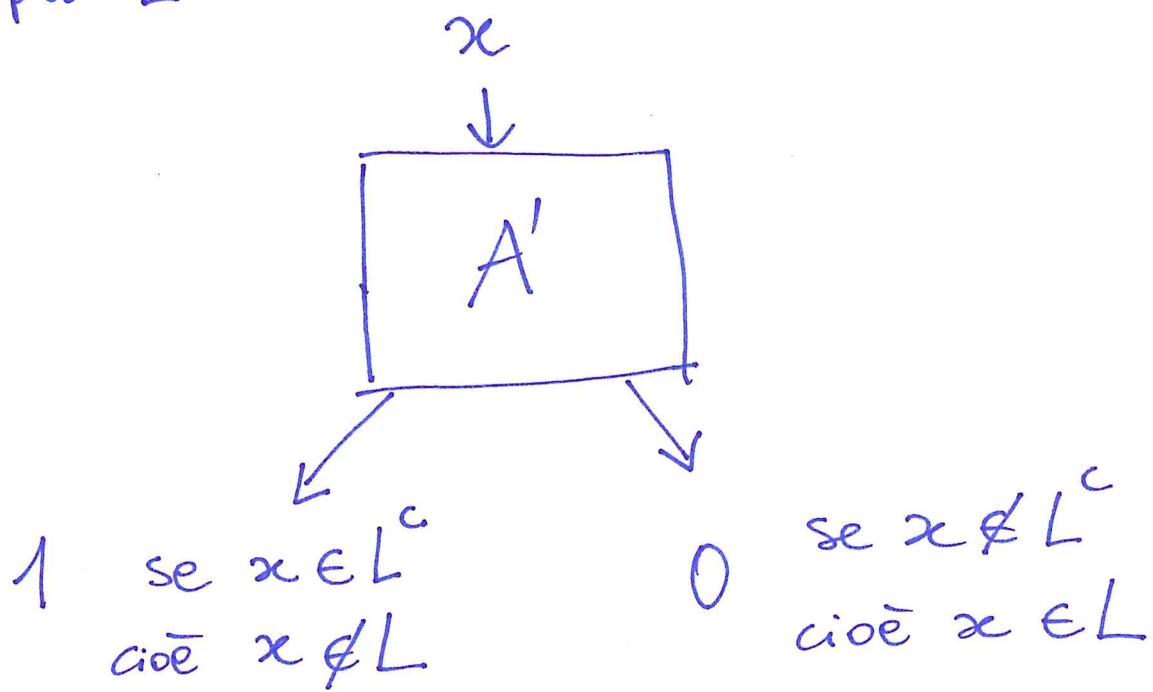
ohimè.

Per ipotesi ho:

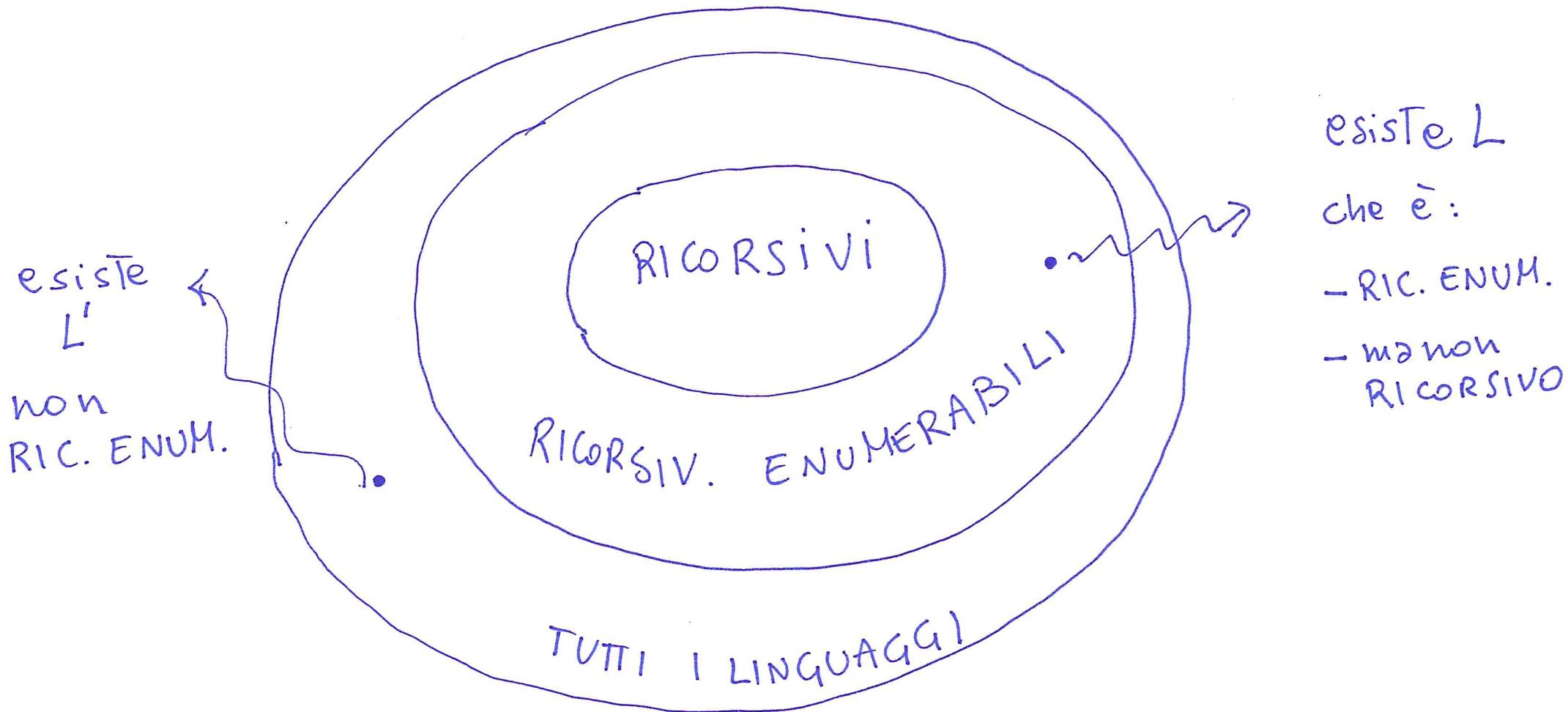


dovendo costruire un algoritmo  $A'$

per  $L^c$



# Risultati importanti della Teoria della calcolabilità



Conseguenze:

•) non è possibile verificare per via automatica la correttezza semantica dei programmi

Teorema di Rice  
"1950"

oppure

non è possibile verificare per via automatica l'equivalenza di due programmi cioè:

dati  $v$  e  $w$  verificare che

$$\forall x \in \{0,1\}^* \quad F_v(x) = F_w(x)$$

•) non è possibile verificare per via automatica la terminazione dei programmi cioè:

(\*) → Problema dell'arresto:

INPUT: programma  $w$ , dato  $x$

OUTPUT:  $F_w(x) \downarrow ?$

questo problema è indecidibile!  
"TURING 1936"

(FATTI VERI - FRASI VERE)

.) ci sono "teoremi" matematici  
non dimostrabili

Risultato di incompletezza di GOEDEL  
nel "1930"

Interprete, programma " $\mu$ "

in input passiamo ad  $\mu$  la coppia

(programma, dato)

in uscita ho il risultato dell'esecuzione  
del "programma" sul "dato"

$$F_\mu(w \$ x) = \begin{cases} \overline{F_w}(x) & \text{se } w \\ & \text{è programma} \\ \perp & \text{altrimenti} \end{cases}$$

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