UNIVERSITÀ
DEGLI STUDI DI TRIESTE

Computabilità, Complessità e Logica

Prof. Adriano Peron

Computabilità: Linguaggi liberi dal contesto

Linguaggi liberi dal contesto

- **La classe dei linguaggi liberi dal contesto estende propriamente la classe dei linguaggi regolari**
  - Include tutti i linguaggi regolari
  - Comprende linguaggi che non sono regolari
  - Ad esempio il linguaggio L = $\{a^n \cdot b^n : n \geq 0\}$
  - Non è un linguaggio regolare
  - E' un linguaggio libero dal contesto.
- **Due vie alternative per definire i linguaggi liberi dal contesto**
  - Utilizzando **grammatiche** che li **generano.**
  - Utilizzando **macchine con pila** che li **riconoscono.**

Grammatiche libere dal contesto

- **Un metodo per definire linguaggi.**
- **Utilizzate originariamente nello studio del linguaggio**
  - **Permettono di stabilire le relazioni tra parti in cui una frase del linguaggio è strutturata (sostantivi, verbi, articoli, etc)**

- **Una applicazione importante ha storicamente riguardato la compilazione dei linguaggi di programmazione.**
  - **Permettono di definire la sintassi formale di un linguaggio di programmazione**
  - **Permettono di validare la correttezza sintattica di un programma (parsing)**

Grammatiche libere dal contesto

- **Esempio.**

<FRASE> → <SOGGETTO> <VERBO> <OGGETTO>

<SOGGETTO> → <ARTICOLO> <SOSTANTIVO>

<OGGETTO> → <ARTICOLO> <SOSTANTIVO>

<FRASE> → <FRASE> <CONGIUNZIONE> <FRASE>

- Il topo mangia il formaggio e il gatto mangia il topo

<ARTICOLO> → il

<CONGIUNZIONE> → e

<VERBO> → mangia

<soggetto> → gatto | topo

Grammatiche libere dal contesto

- **Due tipi di elementi.**
- **Variabili** (consentono di definire la struttura)

  <FRASE>, <ARTICOLO>, <VERBO> . . . .

- **Simboli terminali** (elementi informativi)

  il, cane, gatto, mangia, ...

- **Una parola accettata usa solo simboli terminali senza occorrenze di variabili**
  - Il topo mangia il formaggio e il gatto mangia il topo
- **Caratteristiche. Definizioni ricorsive.**

<FRASE> → <FRASE> <CONGIUNZIONE> <FRASE>

Grammatiche libere dal contesto

- **Generazione della frase:**

<FRASE>

<FRASE> <CONGIUNZIONE> <FRASE>

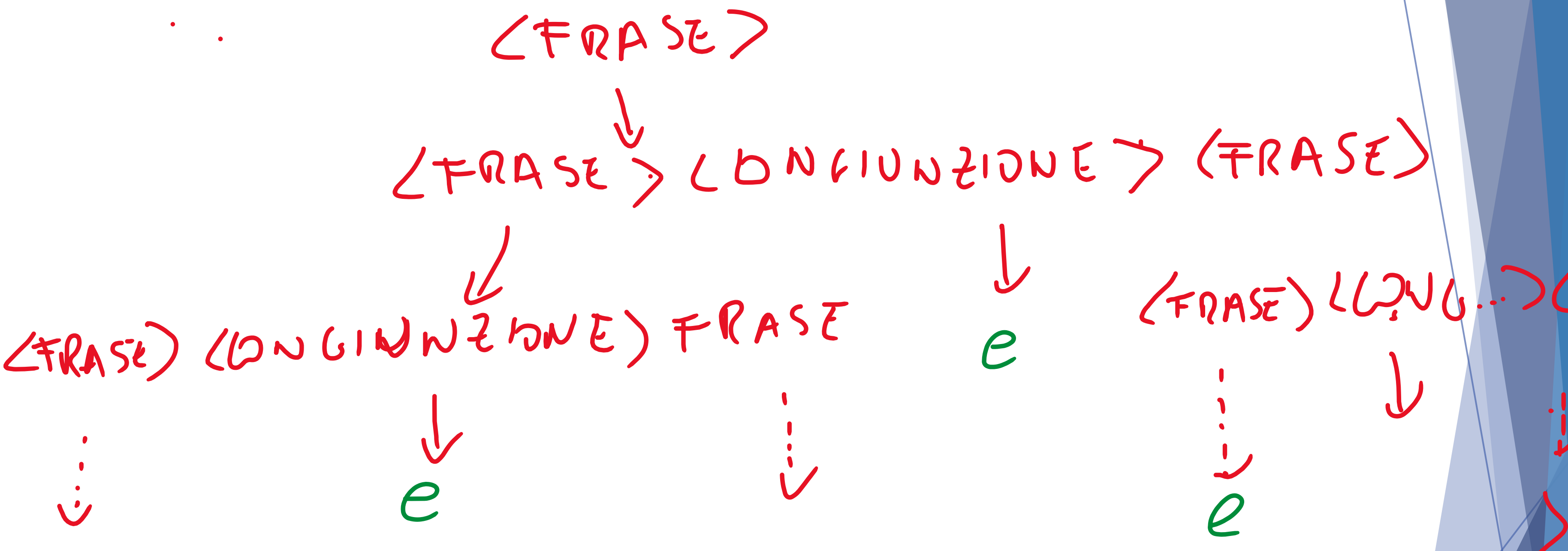
<SOGGETTO> <VERBO> <OGGETTO> e ....

<ARTICOLO> <SOSTANTIVO> mangia <ARTICOLO> <SOSTANTIVO>

il topo il formaggio

Grammatiche libere dal contesto

- **Ricorsione:**



Grammatiche libere dal contesto

Esempio: una DTD in XML

```
<!DOCTYPE PcSpecs [
    <!ELEMENT PCS (PC*)>
    <!ELEMENT PC (MODEL, PRICE, PROCESSOR, RAM, DISK+)>
    <!ELEMENT MODEL (\#PCDATA)>
    <!ELEMENT PRICE (\#PCDATA)>
    <!ELEMENT PROCESSOR (MANF, MODEL, SPEED)>
    <!ELEMENT MANF (\#PCDATA)>
    <!ELEMENT MODEL (\#PCDATA)>
    <!ELEMENT SPEED (\#PCDATA)>
    <!ELEMENT RAM (\#PCDATA)>
    <!ELEMENT DISK (HARDDISK | CD | DVD)>
    <!ELEMENT HARDDISK (MANF, MODEL, SIZE)
    <!ELEMENT SIZE (\#PCDATA)>
    <!ELEMENT CD (SPEED)>
    <!ELEMENT DVD (SPEED)>
]>
```

Grammatiche libere dal contesto

Esempio: una DTD in XML

```
<PCS>
    <PC>
        <MODEL>4560</MODEL>
        <PRICE>$2295</PRICE>
        <PROCESSOR>
            <MANF>Intel</MANF>
            <MODEL>Pentium</MODEL>
            <SPEED>800MHz</SPEED>
        </PROCESSOR>
        <RAM>256</RAM>
        <DISK><HARDDISK>
            <MANF>Maxtor</MANF>
            <MODEL>Diamond</MODEL>
            <SIZE>30.5Gb</SIZE>
        </HARDDISK></DISK>
        <DISK><CD>
            <SPEED>32x</SPEED>
        </CD></DISK>
    </PC>
    <PC>
        ...
    </PC>
</PCS>
```

Grammatiche libere dal contesto

Esercizio: convertire la DTD in una grammatica

```
<!DOCTYPE CourseSpecs [
    <!ELEMENT COURSES (COURSE+)>
    <!ELEMENT COURSE (CNAME, PROF, STUDENT*, TA?)>
    <!ELEMENT CNAME (#PCDATA)>
    <!ELEMENT PROF (#PCDATA)>
    <!ELEMENT STUDENT (#PCDATA)>
    <!ELEMENT TA (#PCDATA)> ]>
```

Grammatiche libere dal contesto: definizione

- **Una grammatica è una tupla G=**$< \boldsymbol{V}, \boldsymbol{\Sigma}, \boldsymbol{R}, \boldsymbol{S} >$
- $\boldsymbol{V}$ è l'insieme delle variabili
- $\boldsymbol{\Sigma}$ è l'insieme dei simboli terminali
- $\boldsymbol{R}$ è l'insieme delle regole della forma $A \longrightarrow w$ con $A \epsilon V, w \in (V \cup \Sigma)^*$
- $\boldsymbol{S \epsilon V}$ è la variabile iniziale.
- Si osservi che:
  - La parte sinistra della regola è costituita da una sola variabile
  - La parte destra da una sequenza di simboli terminali o di variabili (o entrambi).

Grammatiche libere dal contesto: semantica

- **Idea:**
- Si parte con il simbolo iniziale
- Si riscrive la stringa di caratteri corrente applicando iterativamente le regole
- Una riscrittura consiste nella sostituzione di una variabile nella stringa corrente con la parte destra di una regola per la variabile.
- Il processo di riscrittura termina quando la stringa di caratteri corrente non ha più variabili.
- Esempio $< \boldsymbol{V}, \boldsymbol{\Sigma}, \boldsymbol{R}, \boldsymbol{S} >$
- $\boldsymbol{V} = \{\mathbf{S}, \mathbf{A}, \mathbf{B}\}, \boldsymbol{\Sigma} = \{\mathbf{a}, \mathbf{b}, \#\}, \mathbf{R} = \{\mathbf{S} \longrightarrow \boldsymbol{A}, \boldsymbol{A} \longrightarrow \boldsymbol{aAb}, \boldsymbol{A} \longrightarrow \boldsymbol{B}, \boldsymbol{B} \longrightarrow \#\}$

$$S \Rightarrow A \Rightarrow aAb \Rightarrow aaAbb \Rightarrow aa\underbrace{aAb}_{A \rightarrow aAb}bb$$

$$\Rightarrow aaa\#bbb$$

Grammatiche libere dal contesto: semantica

- **Più formalmente:**
- Sia $w, w' \epsilon (V \cup \Sigma)^*$, w′è una **riscrittura** di w se
  - $w = vAv'\,per\ qualche\ v, v' \epsilon (V \cup \Sigma)^*$
  - esiste una regola $A \longrightarrow z \in R$
  - $w' = vzv'$
- In questo caso si scriverà w$\Longrightarrow w'$
- Sia $w, w' \epsilon (V \cup \Sigma)^*$, w′è una **derivazione** di w se
- $w = w'\,oppure$ se esiste una sequenza di riscritture da w a w'

$$w = w_1 \Longrightarrow \ \dots \ \Longrightarrow w_n = w'$$

- In questo caso si scriverà w$\Longrightarrow^* w'$
- Il linguaggio generato dalla grammatica G, è l'insieme delle parole

$$L(G) = \{w\epsilon\Sigma^* : S \Longrightarrow^* w\}$$

- Si osservi che fanno parte del linguaggio solo le parole di simboli terminali derivabili dalla variabile iniziale.

Esempi di linguaggi liberi dal contesto

- **L =** $\{a^n \cdot b^n : n \geq 0\}$

$$\text{G=}< V, \Sigma, R, S >$$
$$V = \{\mathbf{S}, \mathbf{A}\}, \Sigma = \{\mathbf{a}, \mathbf{b}\}$$
$$\mathbf{S} \longrightarrow A \mid \varepsilon,$$
$$A \longrightarrow aAb|\varepsilon$$

- **L linguaggio delle parentesi {} ben accoppiate**

$$\text{G=}< V, \Sigma, R, S >$$
$$V = \{\mathbf{S}, \mathbf{A}\}, \Sigma = \{\{,\}\}$$
$$\mathbf{S} \longrightarrow A \mid \varepsilon,$$
$$A \longrightarrow \{A\}|\varepsilon|AA$$

Esempi di linguaggi liberi dal contesto

- **L linguaggio delle parole palindrome**

$$G = < V, \Sigma, R, S >$$
$$V = \{S, A\}, \Sigma = \{a, b\}\}$$
$$S \longrightarrow A \mid \varepsilon,$$
$$A \longrightarrow aAa|bAb|\varepsilon\ |a|b$$

L linguaggio delle parole che hanno lo stesso numero di a e di b

$$G = < V, \Sigma, R, S >$$
$$V = ?\ , \Sigma = \{a, b\}$$
$$R = ?$$

Generazione dei linguaggi regolari

- Se L è un linguaggio regolare esiste una grammatica G che lo genera.
- Sia A=$< Q, \Sigma, q_0, \delta, F >$ un automa tale che L(A)=L
- Definiamo una grammatica

$$G=< V, \Sigma, R, S >$$

tale che L(G)=L(A)=L

Idea.

- Le variabili sono gli stati, S lo stato iniziale

$$V = Q, S= q_0$$

- Per ogni transizione $(q, a, q') \in \delta$ aggiungo una regola

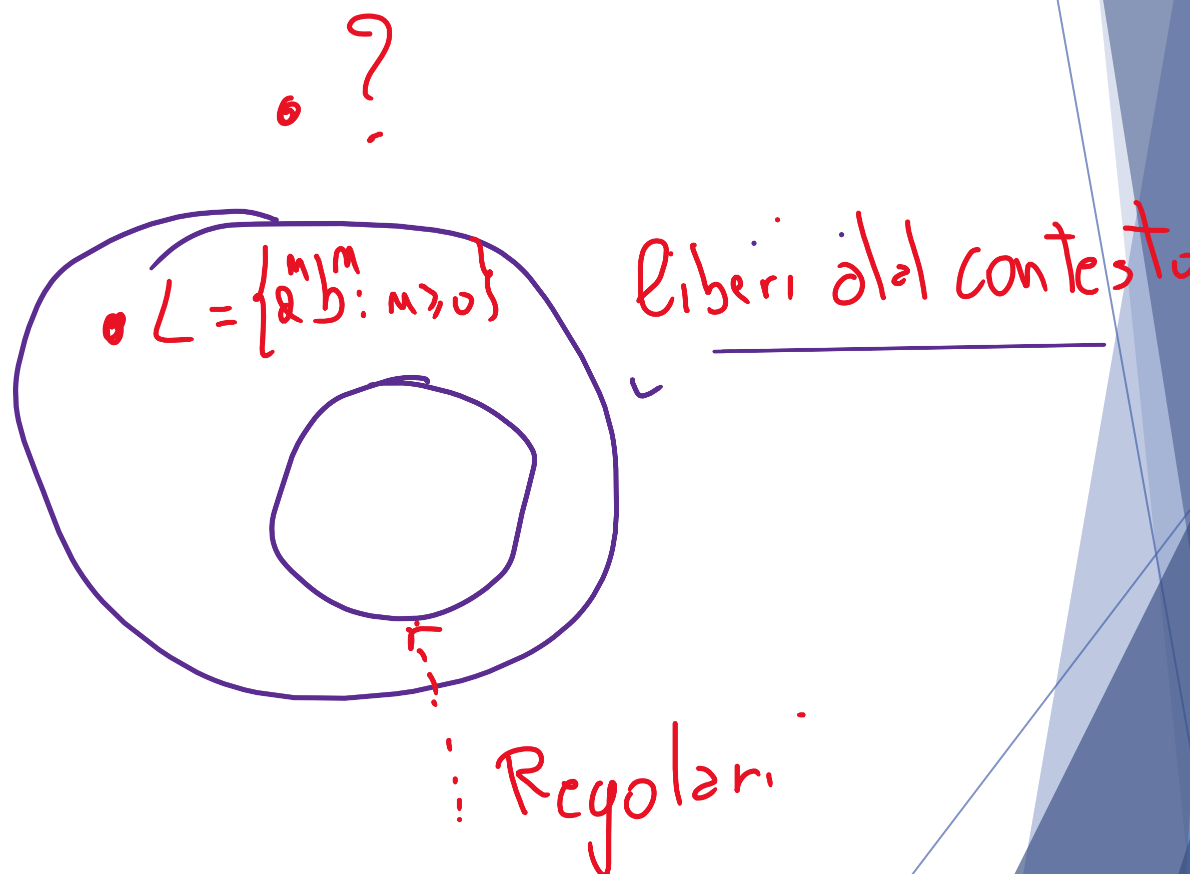
$$q \longrightarrow aq'$$

- Per ogni stato finale q$\in F$ aggiungo una regola

$$q \longrightarrow \varepsilon$$

Espressività

- **Teorema. I linguaggi regolari sono un sottoinsieme proprio dei linguaggi liberi dal contesto.**



Grammatiche in forma normale di Chomsky

- È possible restringere il formato delle regole senza alterare la capacità espressiva delle grammatiche libere dal contesto.

- **Forma normale di Chomsky.** Una grammatica G=$< V, \Sigma, R, S >$ è $\boldsymbol{in\ forma\ normale\ di\ Chomsky}$ se le regole rispettano il seguente formato:
  - $\boldsymbol{A \longrightarrow BC,}$
  - $\boldsymbol{A \longrightarrow a,}$
  - A, B e C sono variabili diverse dalla variabile iniziale S ed a è un simbolo terminale
  - Solo la variabile inziale può esprimere una regola del tipo S $\longrightarrow \boldsymbol{\varepsilon}$
- **Teorema.** **Per ogni grammatica libera dal contesto G esiste una grammatica in forma normale di Chomsky G' tale che L(G)=L(G')**

Grammatiche in forma normale di Chomsky

- Buona proprietà computazionale della forma normale di Chomsky
- Se G=$< V, \Sigma, R, S >$ è $\boldsymbol{in\ forma\ normale\ di\ Chomsky}$ e $w \in L(G)$ con |w|=n allora esiste una derivazione per w lunga $2n-1$.
- Intuizione:
- Con n-1 passi di derivazione genero una parola di n variabili (ogni passo incrementa di 1 la lunghezza della parola)
- Con n passi tasformo n variabili in n simboli terminali.

Applicazione: per cercare una derivazione di una parola di lunghezza n non serve cercare derivazioni di lunghezza superiore a $2n-1$

Automi con pila

- **I linguaggi liberi del contesto possono essere equivalente espressi mediante una estensione delle macchine a stati finiti**
- **le macchine a stati finiti sono estese con l'aggiunta di una struttura dati a pila (stack):**
- **Le macchine a Stati finiti possono memorizzare informazione solo nello stato**
- **L'uso della pila permette di memorizzare informazione in modo illimitato anche nella struttura dati oltre che nello stato**

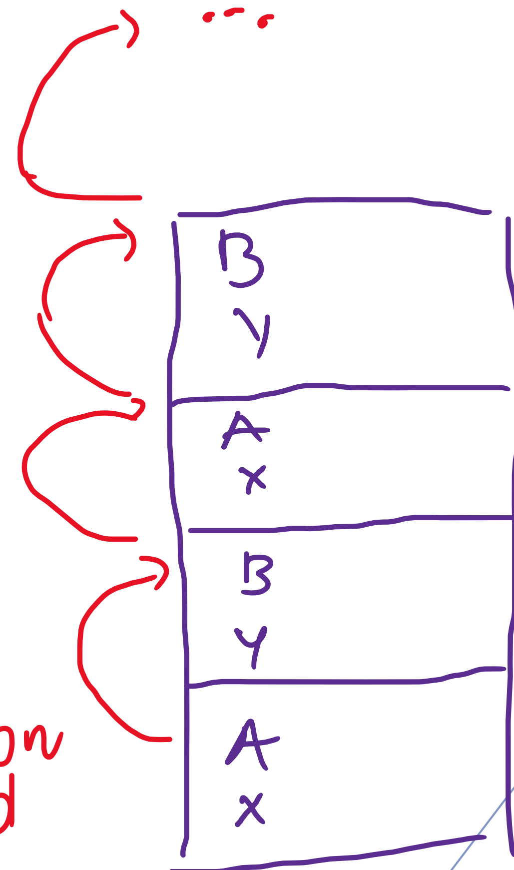
Automi con pila

- **La pila è una struttura dati illimitata che permette di memorizzare e recuperare valori tramite una politica LIFO (Last In First Out).**
- **Operazioni:**
- **inserire un elemento in testa alla pila (push); aumenta di una unità l'altezza della pila**
- **Rimuovere un elemento dalla testa della pila (pop) se la pila non è vuota; decresce di una unità l'altezza della pila.**
- **Struttura dati indispensabile nell'implementazione dei linguaggi di programmazionen che supportano le chiamate ricorsive**

Pila

call stack

Procedure A	Procedure B
(…	(…
x int	y int
…	…
BEGIN	BEGIN
…..	…..
B(x)	A(y)
…..	…..
END	END



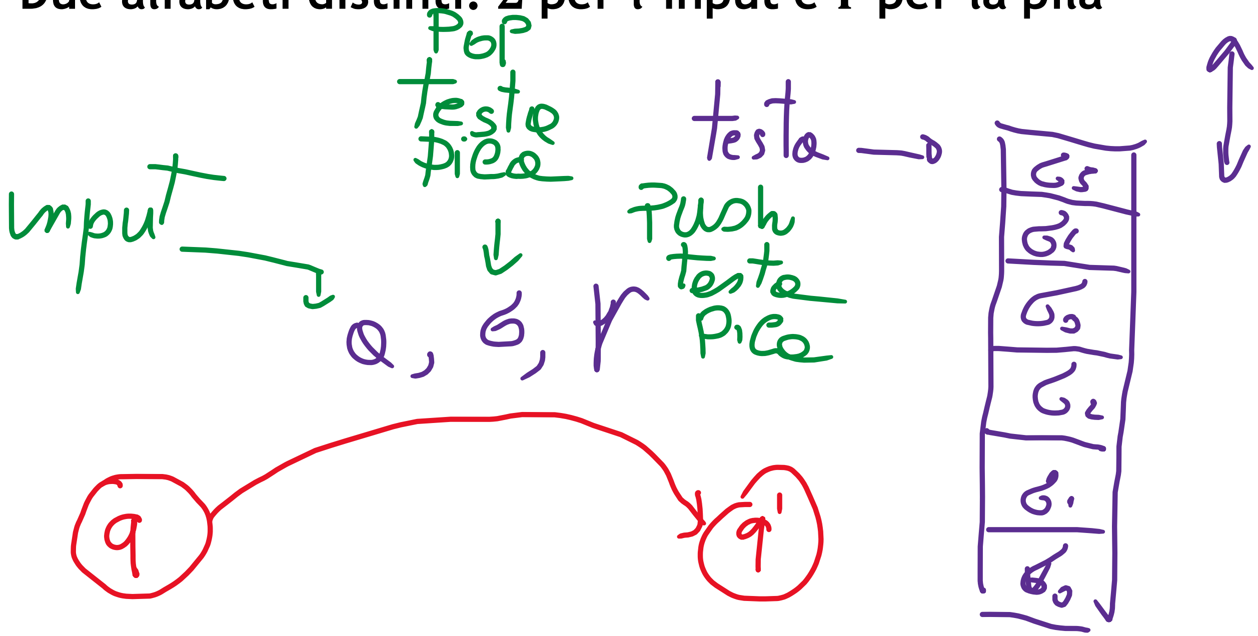
call A

call B

activation record

Automi con pila

- Una transizione è regolata non solo dallo stato di partenza e dal simbolo dell’alfabeto ma anche vdal valore deositato sulla testa della pila
- Una transizione oltre determinare un cambio di stato può svolgere anche una operazione di pop o push sulla pila.
- Due alfabeti distinti: Σ per l’input e Γ per la pila



Automi con pila

- **Per codificare le operazioni sulla pila si usa l'alfabeto $\Gamma \cup \{\varepsilon\}$**

- Transizioni senza uso della pila

no pop

no push

$$q \xrightarrow{a,\ \varepsilon,\ \varepsilon} q'$$

- Transizioni con operazioni di pop sulla pila

pop σ

no push $\sigma \neq \varepsilon$

$$q \xrightarrow{a,\ \sigma,\ \varepsilon} q'$$

Automi con pila

- Per codificare le operazioni sulla pila si usa l'alfabeto $\Gamma \cup \{\varepsilon\}$

- Transizioni con operazioni di push sulla pila

no pop

push

$\gamma \neq \varepsilon$

$$q \xrightarrow{a,\,\varepsilon,\,\gamma} q'$$

- Transizioni con operazioni di pop e push sulla pila

pop σ

push

$\sigma \neq \varepsilon$

$\gamma \neq \varepsilon$

$$q \xrightarrow{a,\,\sigma,\,\gamma} q'$$

swap del contenuto della testa della pila

Automi con pila

- **Un automa a pila che usa solo transizioni etichettate con triple della forma $< a, \varepsilon, \varepsilon >$ è un automa regolare.**
- **Per ogni automa regolare A esiste un automa a pila B tale che L(A) = L(B)**

Automi con pila: sintassi

- **Sia Σ l'alfabeto di input (Σ_ε denota l'insieme $\Sigma \cup \{\varepsilon\}$)**
- **Sia Γ l'alfabeto della pila (Γ_ε denota l'insieme $\Gamma \cup \{\varepsilon\}$)**

- **Un automa a pila B è una tupla $< Q, \Sigma, \Gamma, \delta, q_0, F >$ dove**
  - **Q è l'insieme finite degli stati**
  - **$q_0 \epsilon Q$ è lo stato iniziale**
  - **$\delta \subseteq Q \times \Sigma \times \Gamma \times \Gamma \times Q$ è la relazione di transizione**
  - **$F \subseteq Q$ è l'insieme degli stati finali.**

Automi con pila: semantica

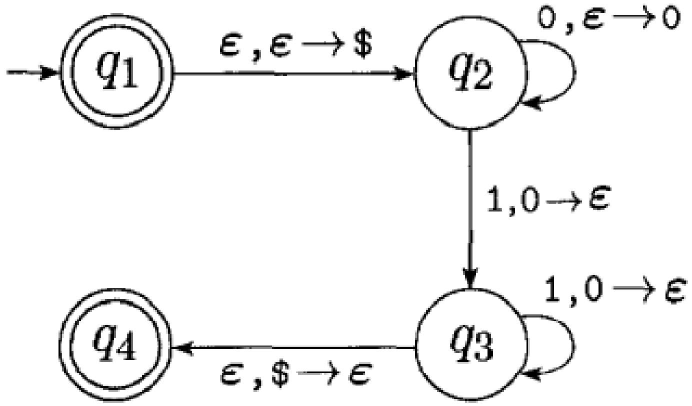
- Lo stato di un automa a pila è dato da due elementi (q, σ):
  - $q \epsilon Q$ è lo stato di controllo corrente dell'automa;
  - $\sigma \epsilon \Gamma^*$ è il contenuto della pila (il contenuto della pila è una stringa di simboli dell'alfabeto della pila).
- Una computazione per una parola $w_0\ w_1 w_2\ \dots w_n \epsilon\, \Sigma^*$ è una sequenza di stati dell'automa della forma

  $(q_0, a_0, \sigma_0),\ (q_1, a_1, \sigma_1),\ (q_2, a_2, \sigma_2),\ \dots\ (q_m, a_m, \sigma_m)$ tale che
- $m \geq n$, e $a_i\ \epsilon \Sigma\ \cup \{\varepsilon\},\ a_0 \cdot a_1 \cdot \cdot \cdot a_m = w_0\ w_1 w_2\ \dots w_n$
- q_0 è lo stato di controllo iniziale
- $\sigma_0 = \varepsilon$ è il contenuto iniziale della pila (pila vuota)
- Per ogni $i > 0$ esiste una transizione $(q_{i-1}, a_{i-1}, \gamma_1, \gamma_2, q_i)$ tale che
- Se $\sigma_{i-1} = \sigma \cdot \gamma$ per qualche $\gamma\ \epsilon\ \Gamma_\varepsilon$ e $\sigma \epsilon \Gamma^*$ allora $\gamma = \gamma_1$

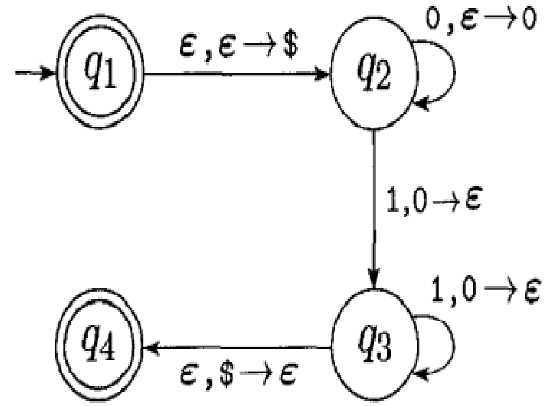
  e $\sigma_i = \sigma \cdot \gamma_2$
- La computazione è accettante se $q_n \epsilon\ F$

Automi con pila: esempio

- Automa per riconoscere il linguaggio **L = $\{o^n \cdot 1^n : n \geq 0\}$**
- **(Il simbolo di stack \$ è usato per marcare lo stack vuoto)**



Automi con pila: esempio



- **Computazione per la parola 000111**

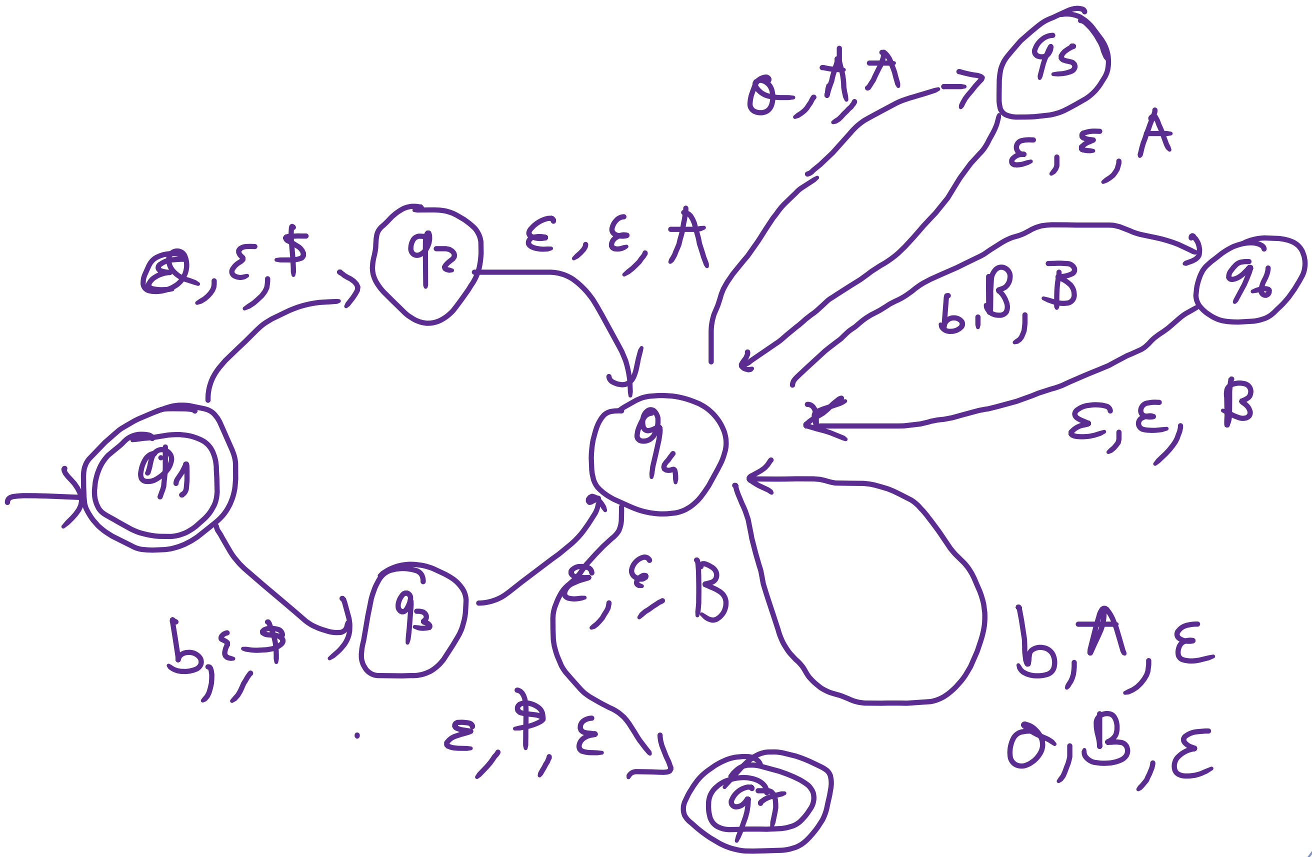
$\Sigma = \{0, 1\}$ $\Gamma = \{\$, 0, 1\}$

input		ε	0	0	0	1	1	1	ε
stato	q_1	q_2	q_2	q_2	q_2	q_3	q_3	q_3	q_4
pila	ε	\$	\$0	\$00	\$000	\$00	\$0	\$	\$

computazione accettante
q_4 è finale

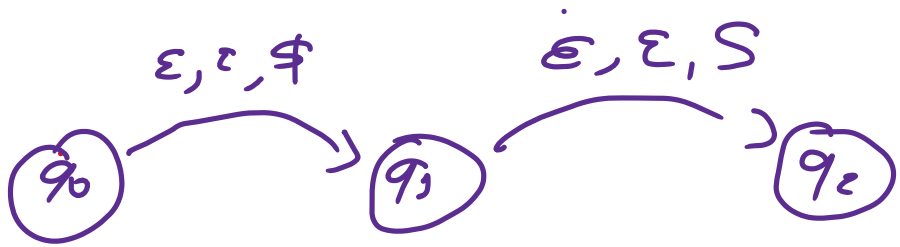
Automi con pila: esempio

- Automa per riconoscere il linguaggio **L delle parole con uguale numero di occorrenze di a e di b.**
- (Il simbolo di stack $ è usato per marcare lo stack vuoto)



Espressività degli automi a pila

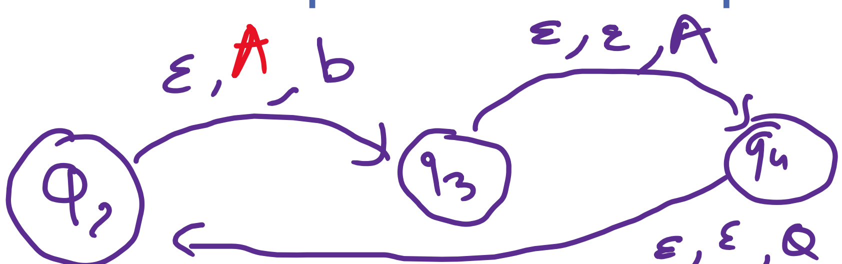
- Si inizia la marcatura di pila vuota e poi la variabile inziale della grammatica



- Si riscrive la testa della pila utilizzando le regole della grammatica.

Ci sono due casi:

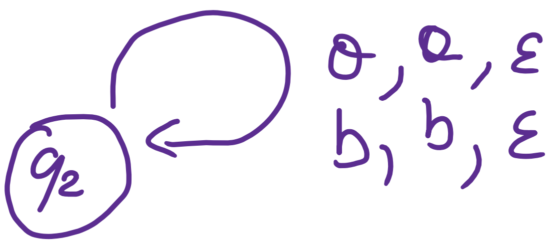
Se in testa alla pila c'è una variabile si sceglie una regola per la variabile e si carica la sua parte destra nella pila.



con la regola
$A \to aAb$
in ordine inverso

Espressività degli automi a pila

Se in testa alla pila c'è un simbolo dell'alfabeto dell'input si fa un passo di lettura nella parola di input leggendo quel simbolo.

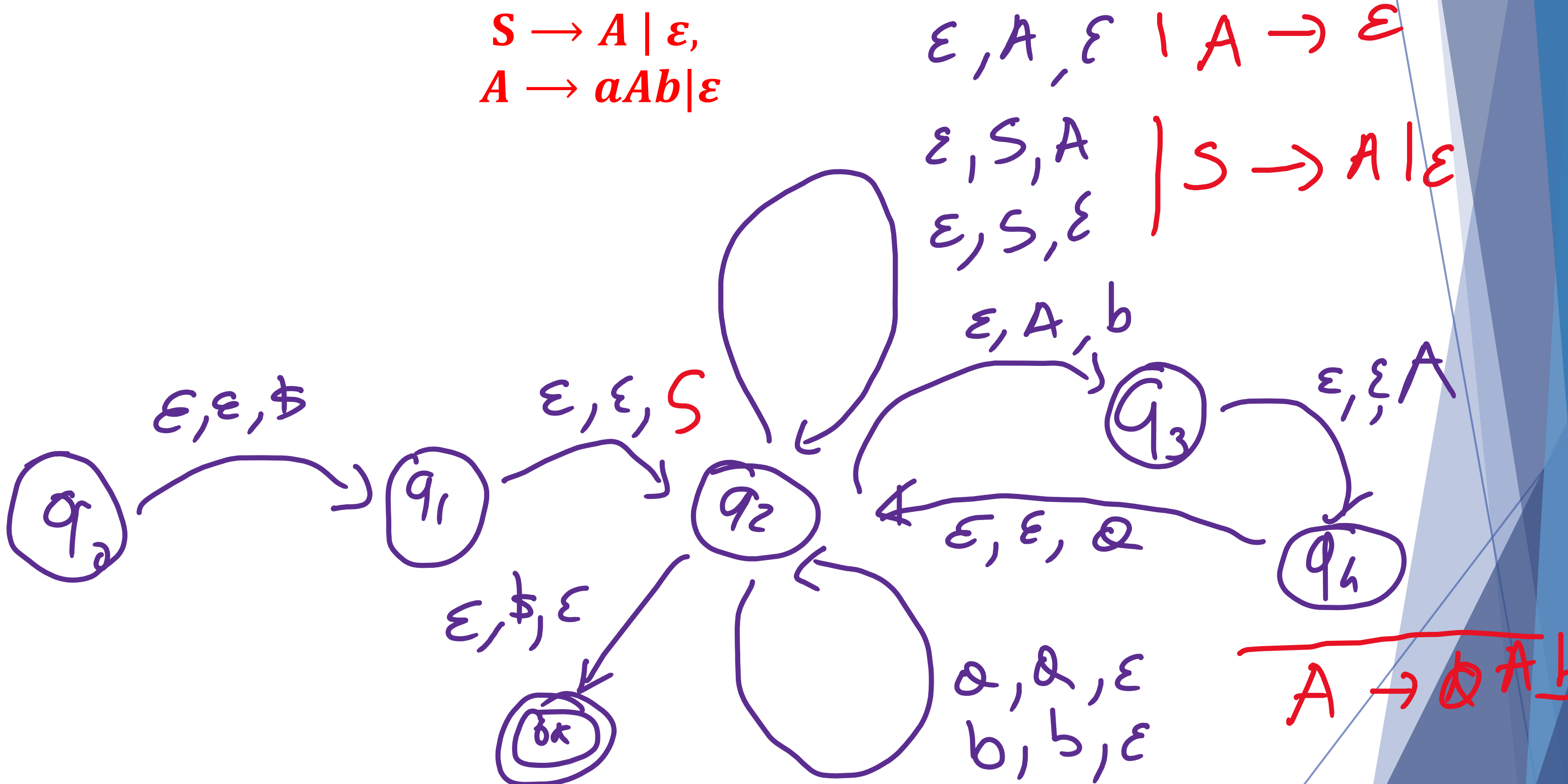


Quando la pila è vuota si va in uno stato di accettazione

Espressività degli automi a pila

- Esempio. Costruzione dell'automa per la grammatica che riconosce il linguaggio **L =** $\{a^n \cdot b^n : n \geq 0\}$

$$G=< V, \Sigma, R, S >$$
$$V = \{S, A\}, \Sigma = \{a, b\}$$
$$S \longrightarrow A \mid \varepsilon,$$
$$A \longrightarrow aAb|\varepsilon$$



Espressività degli automi a pila

Esempio. Scrivere l'automa per la grammatica G=$< V, \Sigma, R, S >$

$$V = \{\mathbf{S}, \mathbf{A}\}, \boldsymbol{\Sigma} = \{\{,\}\}$$

$$\mathbf{S} \longrightarrow A \mid \varepsilon,$$

$$A \longrightarrow \{A\}|\varepsilon|AA$$

Espressività degli automi a pila

- Vale anche l'implicazione opposta

Lemma. Per ogni automa B=$< Q, \Sigma, \Gamma, \delta, q_0, F >$ esiste una grammatica G=$< V, \Sigma, R, S >$ che genera il linguaggio L accettato da B.

- **Automi a pila e grammatiche libere dal contesto sono equivalenti dal punto di vista espressivo.**
- **Possono entrambi essere indifferentemente utilizzati per la definizione dei linguaggi liberi dal contesto.**

Pumping Lemma per linguaggi contrext free definizione

- Teorema. Sia L un linguaggio context-free. Esiste un numero p tale che per ogni parola $w \in L$ con $|w| \geq p$ (p è detto periodo di pumping) la parola w può essere divisa in 5 parti
- $w = u \cdot v \cdot x \cdot y \cdot z$
- e vale che
- $|v \cdot y|>0$ (o v o y diverso da ϵ)
- $|v \cdot x \cdot y| \leq p$
- Per ogni $i \geq 1, u \cdot v^i \cdot x \cdot y^i \cdot z \in L$ (punmping)

Linguaggi non regolari: applicazione del pumping lemma

- **Per accertare che un linguaggio L NON è context-free si sfrutta il pumping lemma.**

1. **Si assume per ipotesi che L sia context-free**
2. **Si applica il pumping lemma per derivare una contraddizione.**

Ad esempio sia L = $\{\boldsymbol{a^n \cdot b^n \cdot c^n : n \geq 0}\}$

1. **Assumiamo che L sia context-free.**
2. **Allora esiste un periodo di pumping p per il linguaggio.**
3. **Prendiamo la parola** $\boldsymbol{w = a^p \cdot b^p \cdot c^p}$
4. **Possiamo frammentare la parola** $\boldsymbol{w} = u \cdot v \cdot x \cdot y \cdot z$ **con** $|\boldsymbol{v \cdot x \cdot y}| \leq \boldsymbol{p}$
5. **se** $|\boldsymbol{v \cdot x \cdot y}| \leq \boldsymbol{p}$ **allora ci possono essere vari casi:**
6. **A)** $\boldsymbol{v \cdot x \cdot y}$ ***incluso in*** $\boldsymbol{a^p}$ ***(oppure*** $\boldsymbol{b^p}$ ***o in*** $\boldsymbol{c^p}$***) .***
7. ***B)*** $\boldsymbol{v \cdot x \cdot y}$ ***incluso in*** $\boldsymbol{a^p \cdot b^n}$***(oppure in*** $\boldsymbol{b^p \cdot c^p}$***) .***
8. *C) non è possibile che* $v \cdot x \cdot y$ *si estenda sulle tre parte simultaneamente*

Linguaggi non regolari: applicazione del pumping lemma

- Caso A)

$\overline{v \cdot x \cdot y} \Rightarrow$ per $i > 1$

$|v^i x \cdot y^i| > |v \cdot x \cdot y|$

$\Rightarrow w'_i = a^{p+k} \cdot b^p \cdot c^p \notin L \qquad K \geq 1$

$\Rightarrow w'_i = a^p b^p c^{p+k} \in L \qquad \overline{v \cdot x \cdot y}$

Linguaggi non regolari: applicazione del pumping lemma

- Caso B)

$a \quad a^p b \quad b^p c \quad c^p$

$u \cdot x \cdot y$

$w_i = a^{p+j} \cdot b^{p+k} \cdot c^p$ con $k+j \geq 1$

$\Rightarrow w_i \notin L$

e

$a \quad a^p b \quad b^p c \quad c$

$u \cdot x \cdot y$

$w_i = a^p \cdot b^{p+j} \cdot c^{p+k}$

con $k+j \geq 1 \Rightarrow w_i \notin L$

Linguaggi non regolari: applicazione del pumping lemma

Ad esempio sia L = $\{w\#w : w \in \{a, b\}^*\}$

1. **Assumiamo che L sia context-free.**
2. **Allora esiste un periodo di pumping p per il linguaggio.**
3. **Prendiamo la parola $w = a^p \cdot b^p \cdot \# \cdot a^p \cdot b^p$**
4. **Possiamo frammentare la parola $w = u \cdot v \cdot x \cdot y \cdot z$ con $|v \cdot x \cdot y| \leq p$**
5. **se $|v \cdot x \cdot y| \leq p$ allora ci possono essere vari casi:**
6. **A) $v \cdot x \cdot y$ *incluso prima di #.***
7. ***B) $v \cdot x \cdot y$ incluso dopo #.***
8. ***C) $v \cdot x \cdot y$ a cavallo di #.***
9. *Si verifichi che in tutti i casi si ottiene una contraddizione!*

Linguaggi non regolari: applicazione del pumping lemma

Provare che i seguenti linguaggi non sono regolari

1. **L il linguaggio su $\Sigma = \{a, b\}$ dove le parole hanno un ugual numero di occorrenze di a e di b**
2. **L il linguaggio su $\Sigma = \{a\}$ dove le parole hanno lunghezza pari a una potenza di due**
3. **L il linguaggio su Σ delle parole speculari: $L = \{w \cdot w^R : w \epsilon \Sigma^*\}$**

Automi a pila e non-determinismo

Attenzione. Gli automi cosi come sono definiti sono non-deterministici.

- E' possible definire una versione deterministica degli automi a pila.
- Gli automi **deterministici** a pila sono meno espressivi degli automi non-deterministici a pila
- **Determinizzazione**: In generale, **non** è possible trovare per ogni automa non-deterministico a pila un automa deterministico a pila ad esso equivalente.

Espressività degli automi a pila

Panoramica sull'espressività

$\{w\#w : w \in \Sigma^*\}$

$\{a^n b^n c^n : n \geq 0\}$

Automi a pila non-det.
=
Grammatiche libere dal contesto

Automi a pila determ.

Linguaggi regolari
- Automi det
- Automi non-det
- Espressioni regolari

Automi a pila e chiusure

Teorema. Gli automi a pila (**I linguaggi context free**) sono chiusi rispetto alle seguenti operazioni:

- Unione.
- Concatenazione.
- * di Kleene.

(Si può dimostrare con costruzioni simili a quelle viste per gli automi regolari)

Risultati negativi.

Teorema. Gli automi a pila (**I linguaggi context free**) **non** sono chiusi rispetto alle seguenti operazioni:

- Intersezione.
- Complemento.

Automi a pila e chiusure

Teorema. Gli automi a pila (I linguaggi context free) non sono chiusi rispetto all'intersezione.

Sia $L_1 = \{a^n b^n c^i : n \geq 0, i \geq 0\}$

$$L_2 = \{a^i b^n c^n : n \geq 0, i \geq 0\}$$

Sia L_1 sia L_2 sono linguaggi liberi dal contesto.

$$L_1 \cap L_2 = \{a^n b^n c^n : n \geq 0\}.$$

Si può dimostare che $\{a^n b^n c^n : n \geq 0\}$ non è un linguaggio libero dal contesto.

Automi a pila e chiusure

Teorema. Gli automi a pila **(I linguaggi context free)** sono chiusi rispetto all'intersezione con i linguaggi regolari.

Sia L un linguaggio acettato dall'automa con pila B e R un linguaggio accettato dall'automa regolare R.

$$L \cap R$$

È un linguaggio libero dal contesto.

Idea.

- Si può costruire un automa a pila che riconosce L sincronizzato con un automa regolare che riconosce R (stessa costruzione fatta per gli automi regolari)
- L'automa che riconosce R ovviamente non usa la pila.
- Perché la stessa costruzione non si può usare per sincronizzare due automi a pila?

Automi a pila e chiusure

Teorema. Gli automi a pila **(I linguaggi context free) non** sono chiusi rispetto alla complementazione.

Prova per assurdo.

- Sia L un linguaggio libero dal contesto e si assuma **per assurdo** che $\overline{L}$ sia un linguaggio libero dal contesto.
- Se L_1 e L_2 sono linguaggi liberi dal contesto allora
$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L}_2}$$
- poiché I linguaggi liberi dal contesto sono chiusi per l'operazione di unione questo implicherebbe che $L_1 \cap L_2$ è un linguaggio libero dal contesto.
- Assurdo per quanto precedentemente dimostrato.

Automi a pila e chiusure

Teorema. Gli automi a pila (I linguaggi context free) non sono chiusi rispetto alla complementazione.

Prova alternativa.

- **Sia $\boldsymbol{L_1}$il linguaggio $\{x\#\varkappa : x\epsilon\{\boldsymbol{a}, b\}^*\}$**
- **Sia $\boldsymbol{L_2}$il linguaggio $\{x\#\boldsymbol{y} : x, \boldsymbol{y}\ \epsilon\{\boldsymbol{a}, b\}^*, \boldsymbol{x} \neq \boldsymbol{y}\}$**

- **$\boldsymbol{L_1}$non è un linguaggio libero dal contesto**
- **$\boldsymbol{L_2}$ è un linguaggio libero dal contesto**
- **R =$\{\boldsymbol{w}\#\boldsymbol{w}' : \boldsymbol{w}, \boldsymbol{w}' \in \{\boldsymbol{a}, b\}^*\}$ è un linguaggio regolare**

Si può osservare che $\boldsymbol{L_1} = \bar{\boldsymbol{L}}_{\boldsymbol{2}} \cap R$

- **Poiché i linguaggi liberi dal contesto sono chiusi per intersezione con i linguaggi regolari si ha $\bar{\boldsymbol{L}}_{\boldsymbol{2}}$ non è un linguaggio libero dal contesto.**