02-linguaggio-e-grammatiche-24

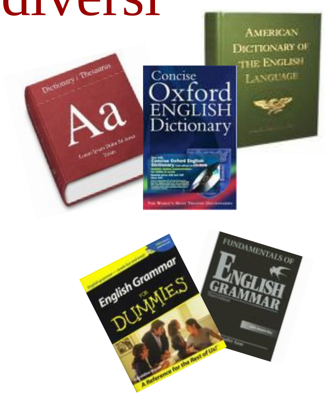
Linguaggi e Grammatiche

Linguaggi e informatica

- ubiquitari nelle applicazioni
  - linguaggi di programmazione
    - compilatori ed interpreti
  - linguaggi di comunicazione
    - protocolli per il dialogo tra entità omologhe
  - linguaggi per intefacce
    - specifica di sequenze di operazioni
- paradigmatici nella teoria
  - molti importanti problemi teorici sono riconducibili a quello dell'appartenenza di una stringa ad un linguaggio

02-linguaggio-e-grammatiche-24

Tre approcci diversi



- approccio insiemistico
  - utile per determinare le proprietà elementari dei linguaggi
- approccio generativo
  - grammatiche formali
- approccio riconoscitivo
  - automi riconoscitori

3

3

Concetti matematici di base

- Insiemi
- Relazioni
- Funzioni

4

4

02-linguaggio-e-grammatiche-24

Insiemi

- consideriamo insiemi *finiti* e insiemi *infiniti*
- $|A|$ = cardinalità dell'insieme (finito) A
- alcuni insiemi infiniti di numeri:

N	naturali (contiene zero)	Q	razionali relativi
N^+	naturali positivi	Q^+	razionali positivi
		Q^-	razionali negativi
Z	interi relativi		
Z^+	interi positivi	R	reali
Z^-	interi negativi	R^+	reali positivi
		R^-	reali negativi

5

5

Sottoinsiemi e insiemi uguali

- dati due insiemi A e B, se
  $x \in B \Rightarrow x \in A$
  allora B è *sottoinsieme* di A, e si scrive $B \subseteq A$
- ogni insieme è sottoinsieme di se stesso
- l'insieme vuoto $\varnothing$ è sottoinsieme di ogni insieme
- se A e B sono finiti, allora $B \subseteq A \Rightarrow |B| \leq |A|$
- A e B *insiemi uguali*
  $A=B \Leftrightarrow (x \in A \Leftrightarrow x \in B)$
  si può scrivere anche
  $A=B \Leftrightarrow (A \subseteq B \wedge B \subseteq A)$
- A è *sottoinsieme proprio* di B ($A \subset B$) se
  $(A \subseteq B) \wedge (A \neq B)$

6

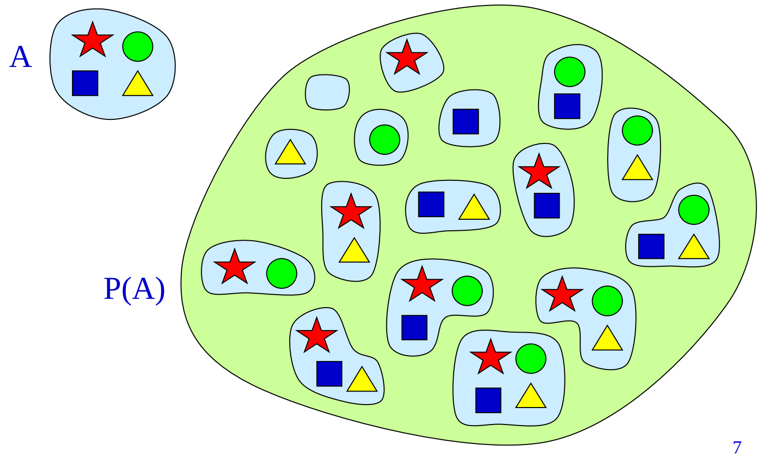
6

02-linguaggio-e-grammatiche-24

Insieme delle parti

l'insieme dei sottoinsiemi di A è detto l'*insieme delle parti* di A e si indica con P(A) o 2^A

se A è finito e $|A| = n$ allora $|P(A)| = 2^n$

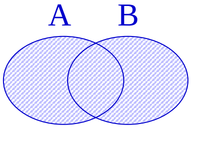


7

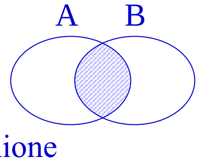
7

Operazioni tra insiemi

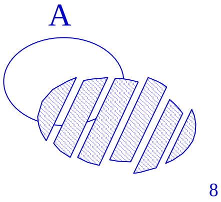
- *unione* $C = A \cup B$
  - se A e B sono finiti $|C| \leq |A|+|B|$
  - commutativa e associativa



- *intersezione* $C=A \cap B$
  - se A e B sono finiti $|C| \leq \min\{|A|, |B|\}$
  - commutativa e associativa
  - l'intersezione è distributiva rispetto all'unione



- *partizione* di A
  - insieme di n sottoinsiemi di A tali che
    $A_1 \cup A_2 \cup \ldots \cup A_n = A$
    $i \neq j \Rightarrow A_i \cap A_j = \varnothing$



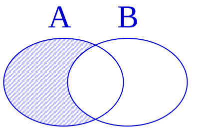
8

8

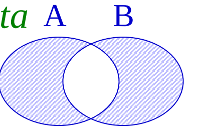
02-linguaggio-e-grammatiche-24

Operazioni tra insiemi

- *complemento* di B rispetto ad A
  $C = A\text{-}B = \{x \mid x \in A \wedge x \notin B\}$



- *differenza simmetrica* o *somma disgiunta*
  $A+B=A \cup B\text{-}(A \cap B)$



- *prodotto cartesiano* $C=A \times B$
  $C= \{<x,y> \mid x \in A \wedge y \in B\}$
  - insieme di tutte le possibili coppie ordinate
  - il prodotto cartesiano è associativo ma non commutativo

9

9

Relazioni

- siano $A_1, A_2, \ldots, A_n$ n insiemi (non necessariamente distinti)
- una *relazione n-aria* è un sottoinsieme di

$$A_1 \times A_2 \times \ldots \times A_n$$

$$R \subseteq A_1 \times A_2 \times \ldots \times A_n$$

esempio:
- la relazione "minore di" definita sui naturali è l'insieme $R \subseteq N \times N = N^2$, dove $R=\{<x,y> \mid x<y\}$

10

10

02-linguaggio-e-grammatiche-24

Relazione d'ordine

- $R \subseteq A^2 = A \times A$ è una *relazione d'ordine* se valgono le seguenti proprietà:
  1. *riflessività*
     $\langle x,x \rangle \in R$
  2. *antisimmetria*
     $\langle x,y \rangle \in R \wedge \langle y,x \rangle \in R \Rightarrow x=y$
  3. *transitività*
     $\langle x,y \rangle \in R \wedge \langle y,z \rangle \in R \Rightarrow \langle x,z \rangle \in R$

un insieme su cui è definita una relazione d'ordine si dice parzialmente ordinato o *poset* ("partially ordered set")

esempio: la relazione "$\leq$" è una relazione d'ordine su N

11

11

Relazione d'ordine totale

- una relazione d'ordine $R \subseteq A^2$ è detta *totale* se
  $\langle x,y \rangle \in A^2 \Rightarrow \langle x,y \rangle \in R \vee \langle y,x \rangle \in R$

esempio:

la relazione "$\leq$" è una relazione d'ordine totale su N
$1 \leq 2 \leq 3 \leq 4 \leq 5 \leq 6 \leq 7 \leq 8 \ldots$

12

12

02-linguaggio-e-grammatiche-24

Relazione di equivalenza

- $R \subseteq A^2 = A \times A$ è una *relazione di equivalenza* se valgono le seguenti proprietà:
  1. *riflessività*
     $\langle x,x \rangle \in R$
  2. *simmetria*
     $\langle x,y \rangle \in R \Rightarrow \langle y,x \rangle \in R$
  3. *transitività*
     $\langle x,y \rangle \in R \wedge \langle y,z \rangle \in R \Rightarrow \langle x,z \rangle \in R$

esempio: la relazione "=" è una relazione di equivalenza su R

13

13

Relazione di equivalenza

- un insieme A su cui è definita una relazione di equivalenza si può partizionare in sottoinsiemi massimali di equivalenza, detti *classi di equivalenza*
- l'insieme delle classi di equivalenza di A è detto *insieme quoziente* e si denota A/R
- un elemento di A/R si denota con [a]
- il numero di classi di A/R si chiama *indice* di R

14

14

02-linguaggio-e-grammatiche-24

Esempio di relazione di equivalenza

- consideriamo la relazione E_k su N
  $n \equiv_k m$
  se esistono q, q′, r (con r<k) tali che
  $n=qk+r$ e $m=q'k+r$
- E_k è una relazione di equivalenza
- le sue classi sono le classi resto rispetto alla divisione per k

15

15

Operazioni su relazioni

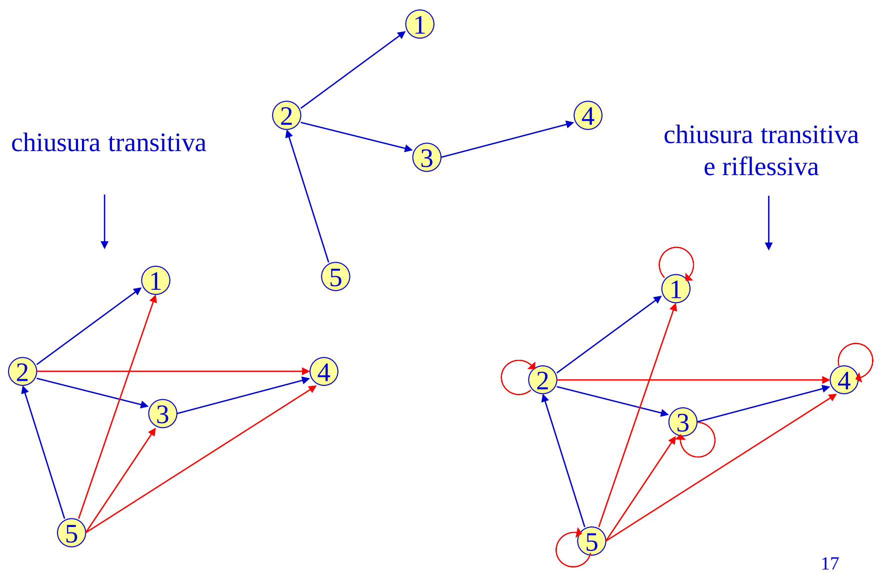
- *unione*
  $R_1 \cup R_2=\{<x,y> \mid <x,y> \in R_1 \vee <x,y> \in R_2\}$
- *complementazione*
  $\underline{R}=\{<x,y> \mid <x,y> \notin R\}$
- *chiusura transitiva*
  $R^+ = \{<x,y> \mid$
  $\exists y_1, \ldots, y_n \in A,\ n \geq 2,\ y_1=x,\ y_n=y$
  tali che
  $<y_i, y_{i+1}> \in R,\ i=1, \ldots, n-1\}$
- *chiusura transitiva e riflessiva*
  $R^*=R^+ \cup \{<x,x> \mid x \in A\}$

16

16

02-linguaggio-e-grammatiche-24

Chiusure di relazioni



17

17

Funzioni

$$R \subseteq X_1 \times \ldots \times X_n$$

è una *relazione funzionale* se

$$\forall \langle x_1, \ldots, x_{n-1} \rangle \in X_1 \times \ldots \times X_{n-1}$$

esiste al più un elemento $x_n \in X_n$ tale che

$$\langle x_1, \ldots, x_{n-1}, x_n \rangle \in R$$

si chiama *funzione* la legge che associa $\langle x_1, \ldots, x_{n-1} \rangle$ ad x_n

$$f(x_1, \ldots, x_{n-1}) = x_n$$

$$f: X_1 \times \ldots \times X_{n-1} \to X_n$$

$X_1 \times \ldots \times X_{n-1}$ è il *tipo* della funzione

18

18

02-linguaggio-e-grammatiche-24

Funzioni: dominio codominio

dom(f) = *dominio* di f

sottoinsieme di $X_1 \times \ldots \times X_{n-1}$

$\mathrm{dom}(f) = \{<x_1, \ldots, x_{n-1}> \in X_1 \times \ldots \times X_{n-1} \mid$

$\exists\, x_n \in X_n \;\; f(x_1, \ldots, x_{n-1}) = x_n\}$

cod(f) = *codominio* di f

sottoinsieme di X_n

$\mathrm{cod}(f) = \{x_n \in X_n \mid$

$\exists <x_1, \ldots, x_{n-1}> \in X_1 \times \ldots \times X_{n-1}$

$f(x_1, \ldots, x_{n-1}) = x_n\}$

19

19

Funzioni: fibra

dato un x_n

$f^{-1}(x_n)$ = *controimmagine* o *fibra* di x_n

sottoinsieme di $X_1 \times \ldots \times X_{n-1}$

$f^{-1}(x_n) = \{<x_1, \ldots, x_{n-1}> \in X_1 \times \ldots \times X_{n-1} \mid$

$<x_1, \ldots, x_{n-1}> \in \mathrm{dom}(f)$

$\wedge$

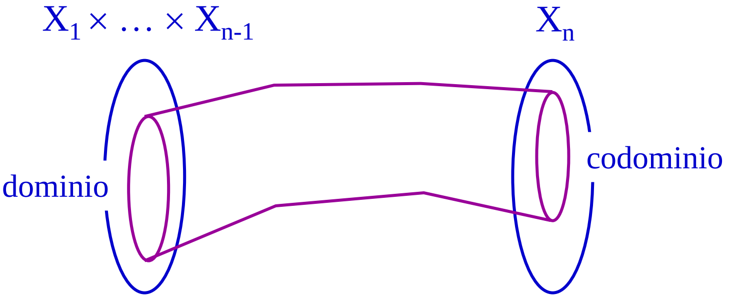
$f(x_1, \ldots, x_{n-1}) = x_n\}$

20

20

02-linguaggio-e-grammatiche-24

Funzione

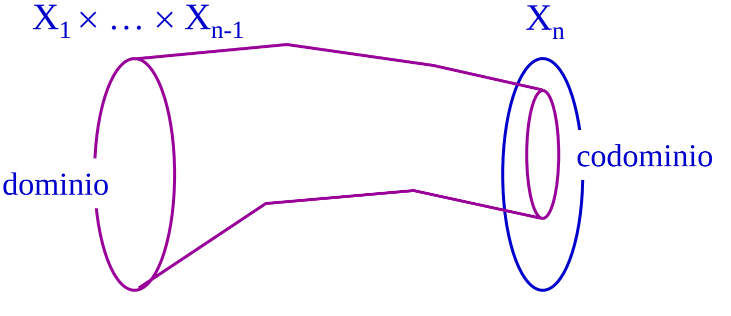


21

21

Funzione totale

- una funzione f è *totale* se dom(f) = $X_1, \ldots, X_{n-1}$



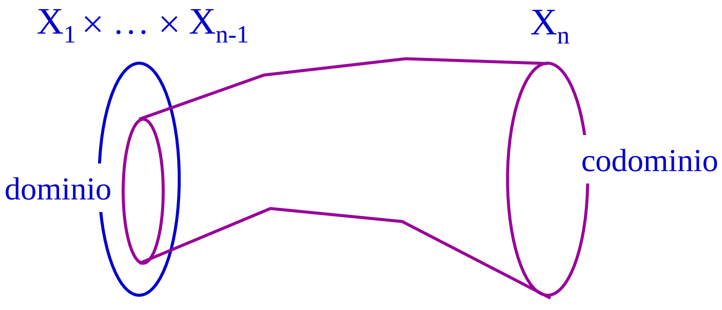
22

22

02-linguaggio-e-grammatiche-24

Funzione suriettiva

- una funzione f è *suriettiva* se cod(f) = X_n

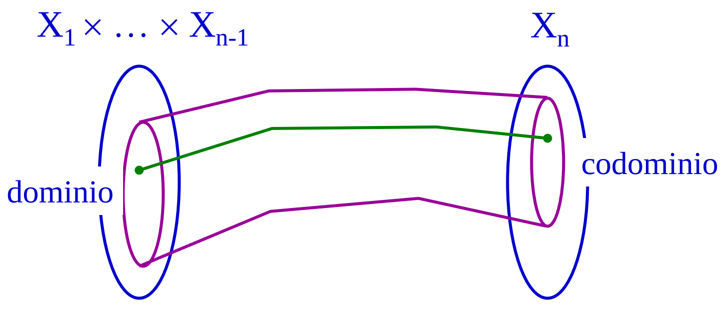


23

23

Funzione iniettiva

- una funzione f è *iniettiva* se $|f^{-1}(x_n)|=1$



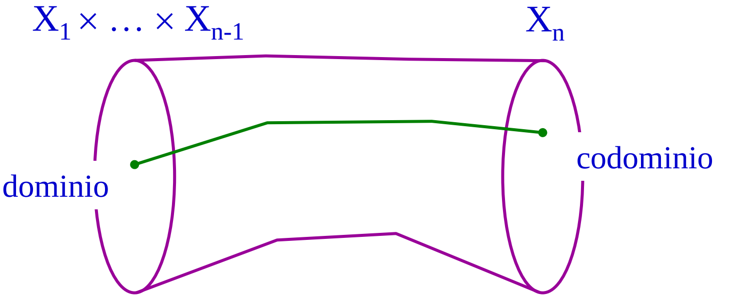
24

24

02-linguaggio-e-grammatiche-24

Funzione biiettiva

- una funzione f è *biiettiva* (biiezione) se è iniettiva, suriettiva e totale



25

25

Alfabeto

- un *alfabeto* è un insieme finito non vuoto di simboli (caratteri)
- esempi:

  {'**M**', '**C**', '**L**', '**X**', '**I**', '**V**'}

  {'**0**', '**1**'}

  {'**0**', '**1**', '**2**', '**3**', '**4**', '**5**', '**6**', '**7**', '**8**', '**9**'}

  {'**a**', '**b**', '**c**', '**d**', '**e**', '**f**', '**g**', '**h**', '**i**', '**l**', '**m**', '**n**', '**o**', '**p**', '**q**', '**r**', '**s**', '**t**', '**u**', '**v**', '**z**'}

26

26

02-linguaggio-e-grammatiche-24

Concatenazione

- dato un alfabeto Σ, definiamo l'operazione binaria *concatenazione* (denotata con "$\circ$")

$$\mathtt{a} \circ \mathtt{b} = \mathtt{ab} \text{ con } \mathtt{a}, \mathtt{b} \in \Sigma$$

- indichiamo con $\mathtt{a}^n$ la concatenazione di **a** con se stessa n volte

  esempio: $\mathtt{a}^4 = \mathtt{a} \circ \mathtt{a} \circ \mathtt{a} \circ \mathtt{a} = \mathtt{aaaa}$
- l'operazione "$\circ$" è associativa ma non commutativa
- dati Σ e "$\circ$" definiamo Σ^+ come l'insieme delle stringhe (parole) di lunghezza finita
- se a Σ^+ aggiungiamo la stringa vuota ε = "" otteniamo Σ^*

27

27

Linguaggio

- un *linguaggio* è un sottoinsieme di Σ^*
- Σ^* è detto *linguaggio universale*
- il linguaggio vuoto Λ non contiene stringhe (nota che Λ coincide con l'insieme vuoto $\varnothing$)
  - attenzione:

$$\Lambda \equiv \varnothing$$

$$\Lambda \neq \{\varepsilon\}$$

28

28

02-linguaggio-e-grammatiche-24

Operazioni sui linguaggi

L_1 e L_2 linguaggi

- *unione*

  $L_1 \cup L_2 = \{x \in \Sigma^* \mid x \in L_1 \vee x \in L_2\}$

  $L_1 \cup \Lambda = L_1$

- *intersezione*

  $L_1 \cap L_2 = \{x \in \Sigma^* \mid x \in L_1 \wedge x \in L_2\}$

  $L_1 \cap \Lambda = \Lambda$

- *complementazione*

  $\underline{L}_1 = \{x \in \Sigma^* \mid x \notin L_1\}$

29

29

Operazioni sui linguaggi

L_1 e L_2 linguaggi

- *concatenazione* o *prodotto*

  $L_1 \circ L_2 = \{x \in \Sigma^* \mid$
  $\exists x_1 \in L_1 \wedge \exists x_2 \in L_2$ tali che $x = x_1 \circ x_2\}$

  $L \circ \{\varepsilon\} = \{\varepsilon\} \circ L = L$

  esempio:

  $L_1 = \{\mathtt{a}^n \mid n \geq 1\}$; $L_2 = \{\mathtt{b}^m \mid m \geq 1\}$; $L_1 \circ L_2 = \{\mathtt{a}^n \mathtt{b}^m \mid n,m \geq 1\}$

- *potenza* L^h di un linguaggio L

  $L^0 = \{\varepsilon\}$

  $L^h = L \circ L^{h-1}$, per $h \geq 1$

30

30

02-linguaggio-e-grammatiche-24

Operatore di Kleene

- *chiusura riflessiva e transitiva* di un linguaggio

$L^* = \bigcup_{h=0}^{\infty} L^h$

$\varepsilon \in L^*$ $\qquad \Lambda^* = \{\varepsilon\}$

esempio: $L=\{aa\}$ $\qquad L^*=\{a^{2n} | n\geq 0\}$

- *chiusura transitiva* (non riflessiva) di un linguaggio

$L^+ = \bigcup_{h=1}^{\infty} L^h$

esempio: $L=\{aa\}$ $\qquad L^+=\{a^{2n} | n\geq 1\}$

$L^*=L^+ \cup \{\varepsilon\}$

31

31

Espressioni regolari

- è uno strumento per descrivere linguaggi (vedremo nel seguito quali)
- dato un alfabeto Σ, si definisce *espressione regolare* ogni stringa r

$$r \in (\Sigma \cup \{+, *, (,), \circ, \varnothing\})^+$$

- tale che:
  1. $r=\varnothing$ oppure
  2. $r\in\Sigma$ oppure
  3. $r=(s+t)$ oppure $r=(s\circ t)$ oppure $r=s^*$, con s e t espressioni regolari

semantica

espressione	linguaggio
$\varnothing$	Λ
$a\in\Sigma$	$\{a\}$
$(s+t)$	$L(s)\cup L(t)$
$(s\circ t)$	$L(s)\circ L(t)$
s^*	$L(s)^*$

32

32

02-linguaggio-e-grammatiche-24

Espressioni regolari

esempio:
(**a**+**b**)* rappresenta L=({**a**}∪{**b**})*

esempio:
(**a**+**b**)***a** rappresenta L={x|x∈{**a**,**b**}* ∧ "x termina con **a**"}

forma sintetica
st è forma sintetica di **s**∘**t**

forma sintetica
espressioni sintetiche si ottengno definendo delle precedenze tra gli operatori: * > ∘ > +

esempio:
(**a**+(**b**(**cd**))) = **a**+**bcd**

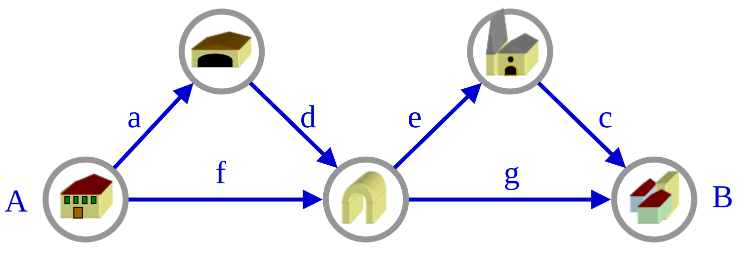
i linguaggi rappresentabili con espressioni regolari sono una interessante sottoclasse

33

33

Esercizio

data una mappa stradale, scrivere un'espressione regolare che definisca tutti i percorsi possibili tra A e B



adec+**adg**+**fec**+**fg** =

= **ad**(**ec**+**g**) + **f**(**ec**+**g**) =

= (**ad**+**f**)(**ec**+**g**)

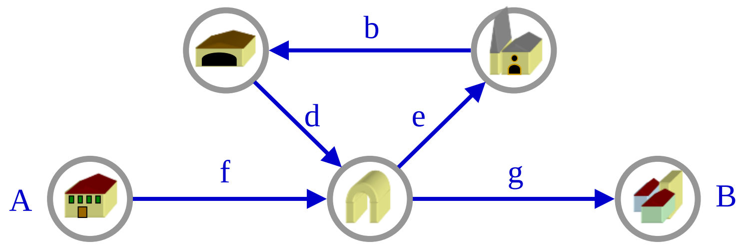
34

34

02-linguaggio-e-grammatiche-24

Esercizio

data una mappa stradale, scrivere un'espressione regolare che definisca tutti i percorsi possibili tra A e B



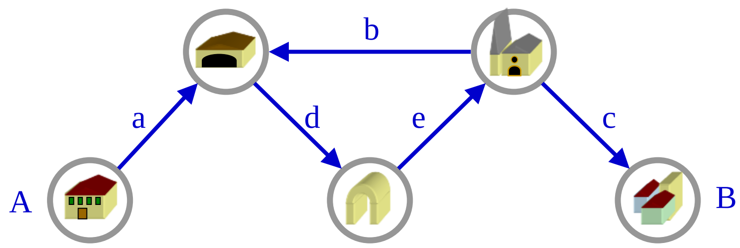
$f(ebd)^*g$

35

35

Esercizio

data una mappa stradale, scrivere un'espressione regolare che definisca tutti i percorsi possibili tra A e B



$a(deb)^*dec$

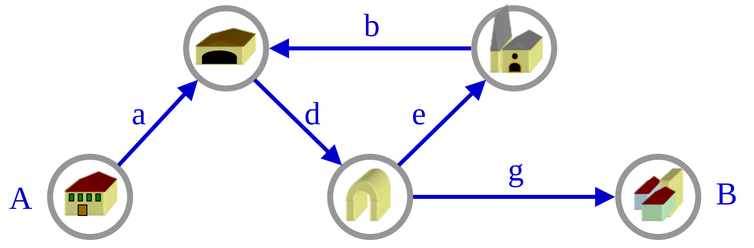
36

36

02-linguaggio-e-grammatiche-24

Esercizio

data una mappa stradale, scrivere un'espressione regolare che definisca tutti i percorsi possibili tra A e B



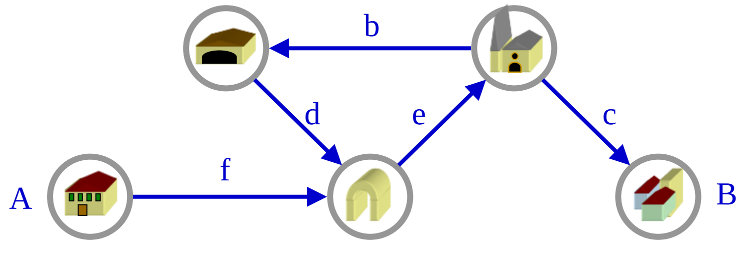
a(deb)*dg

37

37

Esercizio

data una mappa stradale, scrivere un'espressione regolare che definisca tutti i percorsi possibili tra A e B



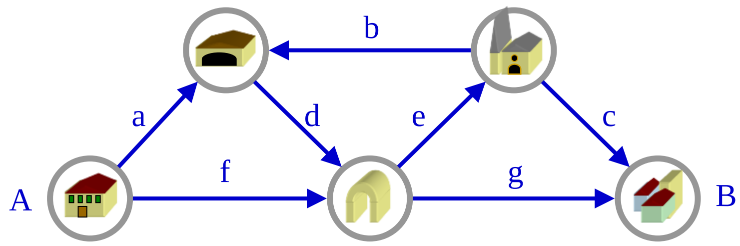
f(edb)*ec

38

38

Esercizio

data una mappa stradale, scrivere un'espressione regolare che definisca tutti i percorsi possibili tra A e B



$$f(ebd)^*g + f(ebd)^*ec + a(deb)^*dec + a(deb)^*dg =$$

$$= f(ebd)^*(g+ec) + a(deb)^*d(g+ec) =$$

$$= (f(ebd)^*+a(deb)^*d)\ (g+ec) =$$

$$= (f+ad)(ebd)^*(g+ec)$$

39

39

Le grammatiche formali

- approccio generativo alla descrizione dei linguaggi
- metodo di costruzione delle stringhe basato sulla riscrittura

1940 Post e Markof, riscrittura e derivazione di stringhe

1950 Chomsky, classificazione delle grammatiche nell'ambito degli studi sul linguaggio naturale

1960 Backus Naur Form per descrivere Algol

40

40

02-linguaggio-e-grammatiche-24

Grammatiche formali

- grammatiche di Chomsky
- ε-produzioni
- riconoscimento di linguaggi

41

41

Grammatiche di Chomsky

una *grammatica* è una quadrupla

$$G=<V_T, V_N, P, S>$$

- $V_T \subseteq \Sigma$ è l'alfabeto (finito) di *simboli terminali*
- V_N è un insieme (finito) di *simboli non terminali*, o *categorie sintattiche*, tale che $V_N \cap \Sigma = \emptyset$
- P, detto insieme delle *produzioni*, è una relazione binaria finita su

$$(V_T \cup V_N)^* \circ V_N \circ (V_T \cup V_N)^* \times (V_T \cup V_N)^*$$

forma sintetica

$<\alpha,\beta> \in P$ si indica generalmente con $\alpha \rightarrow \beta$

- $S \in V_N$ è *l'assioma*

42

42

02-linguaggio-e-grammatiche-24

Esempio

una grammatica definisce implicitamente tutte le stringhe di terminali generabili a partire dall'assioma tramite una sequenza di riscritture

esempio:

G=<{a, b, c}, {S, B, C}, P, S>, con P composto da:

❶ $S \to aS$ ❷ $S \to B$ ❸ $B \to bB$

❹ $B \to bC$ ❺ $C \to cC$ ❻ $C \to c$

genera $L(G) = \{a^n b^m c^h \mid n \geq 0,\ m,\ h \geq 1\}$

forma sintetica

$\alpha \to \beta_1$ $\alpha \to \beta_2$ … $\alpha \to \beta_n$

viene anche indicato con

$\alpha \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$

43

43

Linguaggio generato

forma sintetica

$V_T \cup V_N$ viene talvolta indicato con V

- *derivazione diretta*: relazione su

$$(V^* \circ V_N \circ V^*) \times V^*$$

$<\varphi,\psi>$ appartiene alla relazione (si scrive $\varphi \Rightarrow \psi$) se $\exists \alpha \in V^* \circ V_N \circ V^*$ ed $\exists \beta,\gamma,\delta \in V^*$ t.c. $\varphi = \gamma\alpha\delta$ $\psi = \gamma\beta\delta$ e $\alpha \to \beta \in P$

φ e ψ sono dette *forme di frase*

- *derivazione*: chiusura riflessiva e transitiva della derivazione diretta, si rappresenta con $\Rightarrow^*$
- il *linguaggio generato* da G è $L(G) = \{x \mid x \in V_T^* \wedge S \Rightarrow^* x\}$
- due *grammatiche* G_1 e G_2 sono *equivalenti* se $L(G_1)=L(G_2)$

forma sintetica

talvolta $\Rightarrow$ al posto di $\Rightarrow^*$

44

44

02-linguaggio-e-grammatiche-24

Grammatiche formali

esempio: generazione di $\{a^n b^n c^n | n \geq 1\}$

grammatica G=<{a, b, c},{S,B,C,F,G},P,S>

con P composto da

❶ $S \rightarrow aSBC$ ❷ $CB \rightarrow BC$

❸ $SB \rightarrow bF$ ❹ $FB \rightarrow bF$

❺ $FC \rightarrow cG$ ❻ $GC \rightarrow cG$

❼ $G \rightarrow \varepsilon$

per generare aabbcc

$S \Rightarrow^{1} aSBC \Rightarrow^{1} aaSBCBC \Rightarrow^{2} aaSBBCC$

$\Rightarrow^{3} aabFBCC \Rightarrow^{4} aabbFCC \Rightarrow^{5} aabbcGC$

$\Rightarrow^{6} aabbccG \Rightarrow^{7} aabbcc$

45

45

Grammatiche formali

osservazione: non è detto che una sequenza di derivazioni porti ad una stringa del linguaggio

esempio:

la grammatica G=<{a, b, c}, {S, A}, P, S> con

$S \rightarrow aSc \mid A$

$A \rightarrow bAc \mid \varepsilon$

genera $\{a^n b^m c^{n+m} | n,m \geq 0\}$

esempio:

la grammatica G=<{a, b, c}, {S, A}, P, S> con

$S \rightarrow Ab$

$A \rightarrow Sa$

genera Λ

46

46

02-linguaggio-e-grammatiche-24

Grammatiche di Chomsky

- di tipo 0, non limitate
- di tipo 1, context sensitive, contestuali
- di tipo 2, context free (CF), non contestuali
- di tipo 3, lineari destre (RL), regolari

47

47

Grammatiche di Chomsky di tipo 0, non limitate

- sono le meno restrittive
- produzioni del tipo

$$\alpha \to \beta,\ \alpha \in V^* \circ V_N \circ V^*,\ \beta \in V^*$$

ammettono anche derivazioni che accorciano stringhe

linguaggi di tipo 0

esempio:

il linguaggio $\{a^n b^n | n \geq 1\}$ è di tipo 0 in quanto generato da

$S \to aAB$ $\quad$ $B \to b$

$aA \to aaAb$ $\quad$ $aAb \to ab$

$aAA \to aA$

48

48

02-linguaggio-e-grammatiche-24

Grammatiche di Chomsky di tipo 1, context sensitive, contestuali

- produzioni che non riducano la lunghezza delle forme di frase

$$\alpha \rightarrow \beta,\ |\alpha| \leq |\beta|,\ \alpha \in V^* \circ V_N \circ V^*,\ \beta \in V^*$$

linguaggi di tipo 1

esempio:

il linguaggio $\{a^n b^n c^n | n \geq 1\}$ è di tipo 0 in quanto generato da

$S \rightarrow aSBC$ $\quad CB \rightarrow BC$

$SB \rightarrow bF$ $\quad FB \rightarrow bF$

$FC \rightarrow cG$ $\quad GC \rightarrow cG$

$cG \rightarrow c$

ma è anche di tipo 1, infatti è generato anche da

$S \rightarrow aSBc \mid aBc$

$cB \rightarrow Bc$

$bB \rightarrow bb$

$aB \rightarrow ab$

49

49

Generazione di stringhe di $a^n b^n c^n$

(1) $S \rightarrow aSBc \mid aBc$ $\quad$ (2) $cB \rightarrow Bc$

(3) $bB \rightarrow bb$ $\quad$ (4) $aB \rightarrow ab$

$S \Rightarrow aSBc$
$\Rightarrow aaSBcBc$
$\Rightarrow aaaSBcBcBc$
$\Rightarrow aaaaBcBcBcBc$
$\Rightarrow aaaaBcBcBBcc$
$\Rightarrow aaaaBcBBcBcc$
$\Rightarrow aaaaBcBBBccc$
$\Rightarrow aaaaBBcBBccc$
$\Rightarrow aaaaBBBcBccc$
$\Rightarrow aaaaBBBBcccc$
$\Rightarrow aaaabBBBcccc$
$\Rightarrow aaaabbBBcccc$
$\Rightarrow aaaabbbBcccc$
$\Rightarrow aaaabbbbcccc$

50

50

02-linguaggio-e-grammatiche-24

Grammatiche di Chomsky di tipo 2, context free (CF), non contestuali

- produzioni del tipo

$$A\rightarrow\gamma,\ A\in V_N,\ \gamma\in V^+$$

linguaggi di tipo 2

esempio:

il linguaggio $\{a^n b^n | n\geq 1\}$ è di tipo 0 in quanto generato da

$S \rightarrow aAb$

$aA \rightarrow aaAb$

$A \rightarrow \varepsilon$

ma è anche di tipo 2, infatti è generato anche da

$S \rightarrow aSb \mid ab$

51

51

Esempi di linguaggi di tipo 2

linguaggio delle espressioni aritmetiche con la variabile i (come per esempio "`i*i+(i*i+(i))*i*i`", oppure "`((i+i)*i)`"). L'assioma è E.

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow i \mid (E)$

grammatica delle parentesi ben bilanciate (esempio "`((()))()`")

$S \rightarrow () \mid SS \mid (S)$

da quale sequenza di produzioni è generata "`()((()()))`" ?

grammatica delle stringhe palindrome (esempio "elle", "ereggere")

52

52

02-linguaggio-e-grammatiche-24

Grammatiche di Chomsky di tipo 3, lineari destre (RL), regolari

- produzioni del tipo

$$A \rightarrow \delta,\ A \in V_N,\ \delta \in (V_T \circ V_N) \cup V_T$$

- linguaggi di tipo 3

esempio:

il linguaggio $\{a^n b | n \geq 0\}$ è di tipo 3 in quanto generato da

$S \rightarrow aS$

$S \rightarrow b$

si possono anche definire grammatiche lineari sinistre (LL) con

$$A \rightarrow \delta,\ A \in V_N,\ \delta \in (V_N \circ V_T) \cup V_T$$

esempio: il linguaggio $\{a^n b | n \geq 0\}$ è anche generato da

$S \rightarrow Tb \mid b$

$T \rightarrow a \mid Ta$

teorema: i linguaggi generati da grammatiche LL e RL coincidono

53

53

Grammatiche di Chomsky

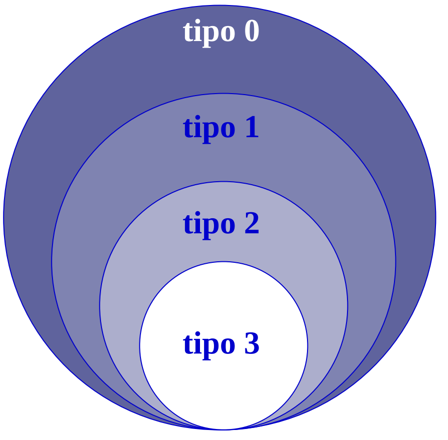
un linguaggio è *strettamente di tipo n* se esiste una grammatica di tipo n che lo genera e non esiste una grammatica di tipo m>n che lo genera

esempio: il linguaggio $\{a^n b^n | n \geq 1\}$ è generato da una grammatica di tipo 2 e non è generato da nessuna grammatica di tipo 3

54

54

Grammatiche di Chomsky



contenimento tra i linguaggi

55

55

Grammatiche di Chomsky

tipo	produzioni	vincoli
tipo 0 non limitate	$\alpha \rightarrow \beta$	$\alpha \in V^* \circ V_N \circ V^*$, $\beta \in V^*$
tipo 1 contestuali	$\alpha \rightarrow \beta$	$\lvert\alpha\rvert \leq \lvert\beta\rvert$ $\alpha \in V^* \circ V_N \circ V^*$, $\beta \in V^*$
tipo 2 non contestuali	$A \rightarrow \gamma$	$A \in V_N$, $\gamma \in V^+$
tipo 3 regolari	$A \rightarrow \delta$	$A \in V_N$, $\delta \in (V_T \circ V_N) \cup V_T$

quadro riassuntivo della classificazione delle grammatiche secondo Chomsky

56

56

02-linguaggio-e-grammatiche-24

ε-produzioni

- con grammatiche di tipo 1, 2, 3 non è possibile generare la stringa vuota ε
  - per generare ε occorre una produzione α→ε che viene detta ε-produzione
  - per Chomsky tutti i linguaggi che contengono ε-produzioni sono linguaggi di tipo 0
- qual è l'impatto sui corrispondenti linguaggi delle ε-produzioni nelle grammatiche?
  - se ammettiamo ε-produzioni dobbiamo fare attenzione, altrimenti rischiamo di snaturare la gerarchia di Chomsky

57

57

ε-produzioni: variazione della gerarchia

con le seguenti modifiche, i linguaggi generati dale diverse tipologie di grammatiche rimangono inalterati, salvo per la possibilità di generare la stringa vuota

tipo	ε-produzioni ammesse
0	tutte (per definizione)
1	solo sull'assioma quando quest'ultimo non compare mai a destra di una produzione
2	tutte
3	tutte

58

58

02-linguaggio-e-grammatiche-24

Esempi di grammatiche

- il linguaggio $\{w \circ w \mid w \in (a+b)^*\}$
- è generato dalla grammatica contestuale

(1)	(2)	(3)	(4)
$S \to T \mid \varepsilon$	$Aa \to aA$	$AA_0 \to A_0a$	$A_0 \to a$
$T \to aAT \mid bBT \mid A_0a \mid B_0b$	$Ab \to bA$	$BA_0 \to A_0b$	$B_0 \to b$
	$Ba \to aB$	$AB_0 \to B_0a$	
	$Bb \to bB$	$BB_0 \to B_0b$	

- le (1) generano insieme caratteri della prima e della seconda stringa; A_0 (B_0) è l'ultimo carattere della prima stringa
- le (2) e le (3) separano la prima stringa dalla seconda
- le (4) chiudono la generazione, se sono applicate troppo presto il processo diverge

59

59

Esempi di grammatiche

- il linguaggio $\{(x\#)^* \mid x = \text{permutazione di } (a,b,c)\}$ (che contiene per esempio le stringhe ε, **abc#**, **acb#**, **bac#cab#** , ecc)
  è generato dalla grammatica contestuale:

$S \to S' \mid \varepsilon$	$AB \to BA$	$A \to a$
$S' \to ABC\#$	$AC \to CA$	$B \to b$
$S' \to ABC\#S'$	$BC \to CB$	$C \to c$

- ma è generato anche dalla grammatica CF:

  $S \to E\#S \mid \varepsilon \qquad E \to abc \mid acb \mid cba \mid cab \mid bac \mid bca$
- ed anche dalla grammatica regolare:

  $S \to aX \mid bY \mid cZ \mid \varepsilon \qquad R \to \#S$

$X \to bX' \mid cX''$	$Y \to aY' \mid cY''$	$Z \to aZ' \mid bZ''$
$X' \to cR$	$Y' \to cR$	$Z' \to bR$
$X'' \to bR$	$Y'' \to aR$	$Z'' \to aR$

60

60

02-linguaggio-e-grammatiche-24

Forma normale di Backus

- la BNF è una notazione CF con alcuni accorgimenti sintattici che ne aumentano la leggibilità

esempio

```
<sequenza istruzioni>::= <istruzione>;
     {<istruzione>;}
```

può essere riscritto:

$Q \rightarrow I\ \texttt{;}\ |\ I\ \texttt{;}\ Q$

```
<istruzione if> ::= if ( <condizione> )
     <istruzione> [else <istruzione>]
```

può essere riscritto:

$F \rightarrow \texttt{if (}\ C\ \texttt{)}\ I\ \texttt{else}\ I\ |\ \texttt{if (}\ C\ \texttt{)}\ I$

61

61

Riconoscimento dei linguaggi

problema:

stabilire se una stringa appartiene ad un dato linguaggio

- esistono linguaggi a cui non corrisponde alcun algoritmo di decisione
- i linguaggi di tipo 3 sono riconosciuti da dispositivi con memoria costante in tempo lineare (automi a stati finiti)
- i linguaggi strettamente di tipo 2 sono riconosciuti da dispositivi non deterministici con pila in tempo lineare (automi a pila non deterministici)

62

62

02-linguaggio-e-grammatiche-24

Riconoscimento dei linguaggi

- i linguaggi strettamente di tipo 1 sono riconosciuti da dispositivi non deterministici con memoria che cresce linearmente con la lunghezza della stringa da esaminare (automi non deterministici "linear bounded")
- i linguaggi strettamente di tipo 0 sono riconosciuti da macchine di Turing con memoria e tempo illimitati
  - è possibile che non esista un algoritmo di decisione ma un processo semidecisionale, in cui, se la stringa non fa parte del linguaggio non è detto che la computazione termini

63

63