Esercizi di Informatica Teorica

Linguaggi non contestuali: proprietà e forme normali

a cura di

Luca Cabibbo e Walter Didimo

Sommario

- proprietà delle grammatiche non contestuali
- pumping lemma
- forme normali

notazioni sul livello degli esercizi: (*) facile, (**) non difficile
(***) media complessità, (****) difficile, (*****) quasi impossibile

Grammatiche non contestuali

grammatica non contestuale (CFG o tipo 2):

$\alpha \rightarrow \beta$ con $\alpha \in V_N$, $\beta \in (V_T \cup V_N)+$

osservazione: è possibile estendere una CFG con ε-produzioni ed inoltre, per ogni CFG con ε-produzioni esiste una CFG equivalente in cui al più solo l'assioma ha una ε-produzione e l'assioma non compare mai a destra

proprietà di chiusura: i linguaggi non contestuali sono chiusi rispetto all'unione, concatenazione ed iterazione; non sono chiusi rispetto ad intersezione e complementazione

Proprietà di chiusura di linguaggi di tipo 2

siano G_1 e G_2 due grammatiche non constestuali e siano S_1 e S_2 i rispettivi assiomi; siano inoltre L_1 ed L_2 i linguaggi riconosciuti da G_1 e G_2:
le grammatiche che riconoscono i linguaggi unione, concatenazione ed iterazione di L_1 ed L_2 sono ottenibili da G_1 e G_2 ridefinendo l'assioma S e le sue produzioni al modo:

- unione: $S \rightarrow S_1 \mid S_2$
- concatenzaione: $S \rightarrow S_1 S_2$
- iterazione (di L_1): $S \rightarrow S_1 S \mid \varepsilon$

esercizio: fare un esempio di linguaggi L_1 ed L_2 di tipo 2 la cui intersezione non è un linguaggio di tipo 2

Pumping lemma per linguaggi di tipo 2

pumping lemma: se L è un linguaggio non contestuale allora $\exists\, n > 0$ tale che $\forall z \in L$ con $|z| \geq n$ $\exists$ u,v,w, x, y :

1) z = uvwxy
2) $|vwx| \leq n$
3) $|vx| \geq 1$
4) $z_i = uv^iwx^iy \in L \quad \forall i \in \mathbf{N}$ (cioè $i = 0, 1, 2, \ldots$)

osservazioni:

1. n dipende da L (viene fissato una volta per tutte sulla base di L)
2. u, v, w, x, y dipendono da z e da n
3. u, w, y possono anche essere stringhe vuote
4. una delle due stringhe v ed x può anche essere vuota
5. poiché può anche essere i = 0, la stringa z_0 = uwy deve appartenere ad L affinché la proprietà 4 del lemma sia soddisfatta

Esercizi svolti sul pumping lemma

Esercizio 1(***) verificare che il pumping lemma vale per i seguenti linguaggi non contestuali:

- $L = \{a^h\, b^k\, a^k\, b^h : h, k \geq 0\}$
- $L = \{s \in \{a,b,c\}^* : \#c = \#a + \#b\}$

Soluzione

- $L = \{a^h\, b^k\, a^k\, b^h : h, k \geq 0\}$

il pumping lemma è valido per ogni stringa (non vuota) di L; infatti, se $k, h > 0$ basta suddividere la stringa in modo che v sia formata soltanto dall'ultima 'b' del primo gruppo di 'b', w sia vuota, ed x sia formata soltanto dalla prima 'a' del secondo gruppo di 'a'

$$z = \underbrace{aa\ldots aabb\ldots b}_{u}\underbrace{ba}_{vx}\underbrace{a\ldots aabb\ldots bb}_{y}$$

Esercizi svolti sul pumping lemma

se $k = 0$ o $h = 0$, allora la stringa z è del tipo a..ab..b oppure b...ba...a, dove il numero di 'a' è uguale al numero di 'b'; in tal caso basta scegliere v ed x come l'ultimo ed il primo simbolo rispettivamente del primo e del secondo gruppo di simboli.

• L = {s∈ {a,b,c}*: #c = #a + #b }
anche in questo caso si può applicare il pumping lemma ad ogni stringa (non vuota) z di L; infatti, in z esiste almeno una 'c' che è adiacente o ad una 'a' o ad una 'b'; supponiamo, per fissare le idee, che esista una 'c' adiacente ad una 'a' e che tale 'a' si trovi alla sua destra; allora è sufficiente scegliere v uguale alla sola 'c', w vuota, ed x uguale alla sola 'a' (gli altri casi sono analoghi)

$$z = \underbrace{abc}_{u}\underbrace{c}_{v}\underbrace{a}_{x}\underbrace{abcccca}_{y}$$

Esercizi svolti sul pumping lemma

Esercizio 2(***) dimostrare, utilizzando il pumping lemma, che i seguenti linguaggi non sono di tipo 2:
• L = $\{a^h b^k a^h b^k : h, k \geq 1\}$
• L = {s∈ {a,b,c}+: #a = #b = #c}

Soluzione

• L = $\{a^h b^k a^h b^k : h, k \geq 1\}$
supponiamo che valga il pumping lemma; allora è possibile fissare un n tale che per tutte le stringhe z di L di lunghezza maggiore o uguale ad n riesce z = uvwxy ($|vwx| \leq n$, $|vx| \geq 1$) e $z_i = uv^iwx^iy \in L \ \ \forall i \in \mathbf{N}$; ma se scegliamo una stringa $z = a^h b^k a^h b^k$ tale che $h, k > n$ si osserva che z ha lunghezza maggiore di n ma non ammette suddivisioni valide; infatti:

Esercizi svolti sul pumping lemma

v ed x devono essere formate o da sole ‘a’ o da sole ‘b’;
– inoltre, per mantenere il bilanciamento, v ed x devono contenere delle ‘a’ (o delle ‘b’) di gruppi diversi (es. v nel primo gruppo di ‘a’ ed x nel secondo gruppo di ‘a’)
– tuttavia, ciò non è possibile dovendo essere $|vwx| \leq n$ ed essendo $h, k > n$ (cioè la distanza minima tra due gruppi di simboli uguali è superiore ad n)

• $L = \{s \in \{a,b,c\}^+ : \#a = \#b = \#c\}$
supponiamo che valga il pumping lemma e che n sia una costante tale che per tutte le stringhe z di L di lunghezza maggiore o uguale ad n riesce $z = uvwxy$ ($|vwx| \leq n$, $|vx| \geq 1$) e $z_i = uv^iwx^iy \in L \;\; \forall i \in \mathbf{N}$;
consideriamo allora la seguente stringa z di lunghezza maggiore di n:

Esercizi svolti sul pumping lemma

$z = a^k b^k c^k$ con $k > n$; comunque proviamo a scegliere una suddivisione “valida” per z, poiché deve essere $|vwx| \leq n$, ed essendo $k > n$, non è mai possibile fare in modo che v ed x prendano uno stesso numero di ‘a’, di ‘b’ e di ‘c’ (vedi l’esempio in figura)

$$z = \text{aaa...aa}\underbrace{\text{a}}_{v}\underbrace{\text{bbb...bb}}_{w}\underbrace{\text{bc}}_{x}\text{cc...ccc}$$

$$> n$$

d’altro canto, suddivisioni di altro tipo sbilancerebbero la stringa, cioè pompando non si avrebbe che $\#a = \#b = \#c$.

Esercizio 3(****) dire se i seguenti linguaggi sono non contestuali, giustificando le risposte:
• $L = \{ss : s \in \{a,b\}^*\}$
• $L = \{ss^R : s \in \{a,b\}^*\}$

Esercizi svolti sul pumping lemma

Soluzione

• $L = \{ss : s \in \{a,b\}^*\}$

L non è un linguaggio non contestuale e si può dimostrare utilizzando il pumping lemma; supponiamo per assurdo che il pumping lemma valga, e sia dunque n una costante tale che per tutte le stringhe z di L di lunghezza maggiore o uguale ad n riesce $z = uvwxy$ ($|vwx| \leq n$, $|vx| \geq 1$) e $z_i = uv^iwx^iy \in L \quad \forall i \in \mathbf{N}$;

consideriamo $z = a^k b^k a^k b^k$ con $k > n$; mostriamo che z, pur avendo lunghezza maggiore di n, non può essere suddivisa opportunamente:

– v ed x non possono prendere solo la prima metà della stringa (cioè il primo gruppo di 'a' e/o di 'b') perché allora $z_0 = uwy$ non sarebbe della forma ss (verificare formalmente!); analogamente v ed x non possono prendere solo la seconda metà della stringa;

Esercizi svolti sul pumping lemma

– allora v ed x devono essere prese a cavallo del centro della stringa, e poiché deve essere $|vwx| \leq n$, allora risulta $vwx = b^i a^j$, con $i, j > 0$; ma allora, se ancora una volta consideriamo la stringa $z_0 = uwy$, essa avrà la forma: $z_0 = a^k b^t a^r b^k$, che non ha la forma ss;

– da ciò l'assurdo

• $L = \{ss^R : s \in \{a,b\}^*\}$ è un linguaggio non contestuale; infatti si tratta dell'insieme delle stringhe palindrome su {a,b} di lunghezza pari; tale linguaggio è per esempio generato dalla seguente grammatica non contestuale:

$S \to \varepsilon$	$S \to X$
$X \to aXa$	$X \to bXb$
$X \to aa$	$X \to bb$

Esercizi da svolgere sul pumping lemma

Esercizio 4(****) dire, giustificando la risposta, se il seguente linguaggio è non contestuale: L = $\{a^i b^j c^k : i = 0 \text{ o } j=k\}$
dire inoltre se L è regolare oppure no.

Esercizio 5(****) mostrare un esempio di linguaggio non di tipo 2 per cui valga il pumping lemma per linguaggi di tipo 2
(suggerimento: sfruttare l'idea dell'Esercizio 4)

Esercizio 6(****) dimostrare, usando il pumping lemma per linguaggi non contestuali, che il seguente linguaggio non è di tipo 2:
L = $\{a^k : k$ è un numero primo$\}$

Grammatiche in forma ridotta

una grammatica G non contestuale è in forma ridotta se:
- G non contiene ε-produzioni, se non sull'assioma, ed in tal caso l'assioma non compare a destra di nessuna produzione;
- G non contiene simboli inutili, cioè:
  - simboli non fecondi (cioè dai quali non sono generabili stringhe di soli terminali)
  - simboli non generabili dall'assioma
- G non contiene produzioni unitarie (cioè del tipo $A \to B$)

teorema: ogni grammatica non contestuale si può scrivere in forma ridotta

Grammatiche in forma ridotta

algoritmo:

• input: una CFG G

• output: una CFG G' equivalente a G ed in forma ridotta

1 - portare eventuali ε-produzioni solo sull'assioma, e se l'assioma compare a destra, introdurre un nuovo assioma ($S' \rightarrow S$, $S' \rightarrow \varepsilon$) ed una serie di produzioni che si ottengono da quelle esistenti sostituendo ε ad S (a destra)

2 - rimuovere le produzioni che contengono simboli non fecondi

3 - rimuovere le produzioni che contengono simboli non generabili dall'assioma

4 - per ogni produzione unitaria $A \rightarrow B$ applicare una tra le due regole seguenti:

(I) per ogni $B \rightarrow ...C...$ ⇒ introdurre $A \rightarrow ...C...$ ed eliminare $A \rightarrow B$

(II) per ogni $C \rightarrow ...A...$ ⇒ introdurre $C \rightarrow ...B...$ ed eliminare $A \rightarrow B$; inoltre eliminare anche $C \rightarrow ...A...$ se A non ha altre produzioni

Esercizi svolti sulla forma ridotta

Esercizio 7(***) portare in forma ridotta la seguente grammatica non contestuale:

$S \rightarrow AB \mid CAB \mid ACE$

$A \rightarrow B \mid BA \mid SAAB$

$B \rightarrow b \mid CAb \mid CED$

$C \rightarrow aC \mid CaD \mid BaD$

$D \rightarrow Ca \mid BEC$

$E \rightarrow e \mid Be$

Soluzione

• non ci sono ε-produzioni

• i simboli non fecondi sono: C, D; rimuovendo dunque le produzioni che li contengono la grammatica diventa:

Esercizi svolti sulla forma ridotta

$S \rightarrow AB$
$A \rightarrow B \mid BA \mid SAAB$
$B \rightarrow b$
$E \rightarrow e \mid Be$

• i simboli non generabili dall'assioma sono: E; rimuovendo le produzioni che contengono E si ha dunque:
$S \rightarrow AB$
$A \rightarrow B \mid BA \mid SAAB$
$B \rightarrow b$

• l'unica produzione unitaria è $A \rightarrow B$ e poiché $B \rightarrow b$, possiamo introdurre la produzione $A \rightarrow b$ e rimuovere $A \rightarrow B$

Esercizi svolti sulla forma ridotta

la grammatica in forma ridotta è dunque:
$S \rightarrow AB$
$A \rightarrow b \mid BA \mid SAAB$
$B \rightarrow b$

nota: per eliminare la produzione unitaria $A \rightarrow B$ dalla grammatica
$S \rightarrow AB$
$A \rightarrow B \mid BA \mid SAAB$
$B \rightarrow b$
potevamo applicare la regola (II) anziché la (I) al modo:
$S \rightarrow AB \mid BB$
$A \rightarrow BA \mid SAAB \mid BB \mid SBBB$
$B \rightarrow b$

Esercizi svolti sulla forma ridotta

Esercizio 8(***) portare in forma ridotta la seguente grammatica non contestuale:

$S \rightarrow \varepsilon$

$S \rightarrow SS$

$S \rightarrow (S)$

Soluzione

• l'assioma ha una ε-produzione e compare anche a destra di altre produzioni; dobbiamo quindi introdurre un nuovo assioma S' ed aggiungere le produzioni che si ottengono sostituendo ε ad S in quelle preesistenti:

$S' \rightarrow \varepsilon \mid S$

$S \rightarrow SS \mid (S) \mid ()$ (la produzione $S \rightarrow S$ è banale e non va messa)

Esercizi svolti sulla forma ridotta

• rimane da eliminare le produzioni unitarie, in quanto non vi sono simboli inutili; l'unica produzione unitaria è $S' \rightarrow S$, e la grammatica in forma ridotta è la seguente (applico la regola (I)):

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

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

Grammatiche in forma normale di Chomsky

una grammatica non contestuale è in forma normale di Chomsky (CNF) se tutte le sue produzioni sono della forma $A \to BC$ o $A \to a$

teorema: ogni grammatica non contestuale G tale che $\varepsilon \notin L(G)$ può scriversi in forma normale di Chomsky

algoritmo
- portare in forma ridotta
- sostituire ogni terminale 'a' con un non terminale X_a in tutte le produzioni in cui compare 'a' ed introdurre la produzione $X_a \to a$ (la forma ottenuta a questo punto si chiama "quasi CFN")
- sostituire ricorsivamente ogni produzione del tipo: $A \to BC\alpha$ con le seguenti: $A \to BD$, $D \to C\alpha$, dove D è un nuovo non terminale

Esercizi svolti sulla CNF

Esercizio 9(**) portare in CNF la seguente grammatica non contestuale:
$S \to SS \mid (S) \mid ()$

Soluzione
- la grammatica è già in forma ridotta
- aggiungiamo due nuovi simboli non terminali A e Z, dove A è associato al simbolo '(' e Z è associato al simbolo ')'; risulta:

$S \to SS \mid ASZ \mid AZ$
$A \to ($ $Z \to)$

- spezziamo le produzioni con più di tre simboli a destra:

$S \to SS \mid AD \mid AZ$
$D \to SZ$
$A \to ($ $Z \to)$

Esercizi svolti sulla CNF

Esercizio 10(***) portare in CNF la seguente grammatica non contestuale:

$S \rightarrow aAa \mid aa$
$A \rightarrow aAa \mid B$
$B \rightarrow bBb \mid bb$

Soluzione

• portiamo la grammatica in forma ridotta

$S \rightarrow aAa \mid aa$
$A \rightarrow aAa \mid bBb \mid bb$
$B \rightarrow bBb \mid bb$

• aggiungiamo un non terminale per 'a' ed uno per 'b' al modo:

Esercizi svolti sulla CNF

$S \rightarrow X_a\, A\, X_a \mid X_a\, X_a$
$A \rightarrow X_a\, A\, X_a \mid X_b\, B\, X_b \mid X_b\, X_b$
$B \rightarrow X_b\, B\, X_b \mid X_b\, X_b$
$X_a \rightarrow a$
$X_b \rightarrow b$

• spezziamo le produzioni con più di tre simboli a destra:

$S \rightarrow X_a\, C \mid X_a\, X_a$
$A \rightarrow X_a\, C \mid X_b\, D \mid X_b\, X_b$
$B \rightarrow X_b\, D \mid X_b\, X_b$
$X_a \rightarrow a$
$X_b \rightarrow b$

$C \rightarrow A\, X_a$
$D \rightarrow B\, X_b$

Grammatiche in forma normale di Greibach

una grammatica non contestuale è in forma normale di Greibach (GNF) se tutte le sue produzioni sono della forma $A \rightarrow a\beta$, dove β è una sequenza (eventualmente vuota) di non terminali

teorema: ogni grammatica non contestuale G tale che $\varepsilon \notin L(G)$ può scriversi in forma normale di Greibach

algoritmo

- portare in CNF o in quasi CNF
- fissare un ordinamento dei non terminali: $A_1, A_2, \ldots, A_m$
- portare tutte le produzioni nella forma: $A_i \rightarrow A_j\alpha$ con $i < j$, oppure $A_i \rightarrow a\gamma$ con 'a' simbolo terminale, usando la seguente procedura:

Grammatiche in forma normale di Greibach

per $k = 1,\ldots,m$ applicare le due regole seguenti nell'ordine:

– sostituzione: $A_k \rightarrow A_j\alpha\ ,\ A_j \rightarrow \beta \Rightarrow A_k \rightarrow \beta\alpha\ ,\ A_j \rightarrow \beta\ \ (\forall j = 1,..,k-1)$

– eliminazione ricorsione sinistra: $A_k \rightarrow A_k\alpha_1 \mid \ldots \mid A_k\alpha_n \mid \beta_1 \mid \ldots \mid \beta_s \Rightarrow$

$A_k \rightarrow \beta_1 B_k \mid \ldots \mid \beta_s B_k \mid \beta_1 \mid \ldots \mid \beta_s$ e $B_k \rightarrow \alpha_1 B_k \mid \ldots \mid \alpha_n B_k \mid \alpha_1 \mid \ldots \mid \alpha_n$

- applicare la sostituzione a "ritroso" $(i = m-1,..,1)$ prima sui non terminali A_i e poi a ritroso sui non terminali B_j (questa fase ci garantisce che tutte le produzioni avranno la parte destra che inizia con un simbolo terminale)

nota pratica: è utile scegliere bene l'ordinamento iniziale dei non terminali per semplificare il calcolo

Esercizi svolti sulla GNF

Esercizio 11(***) portare in GNF la seguente grammatica non contestuale:

$S \rightarrow SS \mid AZ \mid AX$

$X \rightarrow SZ$

$A \rightarrow ($ $\qquad$ $Z \rightarrow)$

Soluzione

- la grammatica è in CNF, quindi è anche in quasi CNF
- scegliamo un ordinamento vantaggioso dei non terminali;

si osserva che S "dipende" da A e che X "dipende" da S, quindi scegliamo il seguente ordinamento: $X < S < A < Z$

- mettiamo tutte le produzioni nella forma $C \rightarrow D\alpha$ con $C < D$ oppure nella forma $C \rightarrow a\gamma$, dove 'a' è un simbolo terminale; dobbiamo considerare i non terminali nell'ordine crescente assegnato

Esercizi svolti sulla GNF

– per X non dobbiamo fare niente, perché $X < S$

– per S dobbiamo solo eliminare la ricorsione sinistra nella produzione $S \rightarrow SS$; le nuove produzioni per S sono:

$S \rightarrow AZB \mid AXB \mid AZ \mid AX$

$B \rightarrow SB \mid S$

– per A e Z non dobbiamo fare niente

la grammatica ottenuta fin qui è dunque:

$S \rightarrow AZB \mid AXB \mid AZ \mid AX$

$B \rightarrow SB \mid S$

$X \rightarrow SZ$

$A \rightarrow ($ $\qquad$ $Z \rightarrow)$

- facciamo ora la sostituzione dei non terminali originali nell'ordine inverso di crescita

Esercizi svolti sulla GNF

– per Z ed A non si deve fare niente
– per S si ha: $S \rightarrow (ZB \mid (XB \mid (Z \mid (X$
– per X si ha: $X \rightarrow (ZBZ \mid (XBZ \mid (ZZ \mid (XZ$

ripetute, quindi si possono togliere

• ora effettuiamo le sostituzioni per B:
$B \rightarrow (ZBB \mid (XBB \mid (ZB \mid (XB \mid (ZB \mid (XB \mid (Z \mid (X$

• quindi, la grammatica in GNF è la seguente:
$S \rightarrow (ZB \mid (XB \mid (Z \mid (X$
$X \rightarrow (ZBZ \mid (XBZ \mid (ZZ \mid (XZ$
$B \rightarrow (ZBB \mid (XBB \mid (ZB \mid (XB \mid (Z \mid (X$
$A \rightarrow ($ $\qquad Z \rightarrow)$

inutile, quindi si può togliere

Esercizi svolti sulla GNF

Esercizio 12(***) portare in GNF la seguente grammatica non contestuale:

$S \rightarrow AC \mid CA$
$A \rightarrow a \mid CAA$
$C \rightarrow b \mid c \mid ACC$

Soluzione
• la grammatica è già in quasi CNF
• scegliamo il seguente ordinamento dei non terminali: $S < A < C$
• effettuiamo le sostituzioni e le eliminazioni della ricorsione sinistra nell'ordine crescente assegnato per i non terminali:
– per S non si deve fare niente

Esercizi svolti sulla GNF

– per A non si deve fare niente;
– per C si ha:
  – $C \rightarrow b \mid c \mid aCC \mid CAACC$ (sostituzione)
  – $C \rightarrow bB \mid cB \mid aCCB \mid b \mid c \mid aCC$ (eliminazione risorsione sin.)
  $B \rightarrow AACCB \mid AACC$

• sostituiamo a ritroso:

$C \rightarrow bB \mid cB \mid aCCB \mid b \mid c \mid aCC$

$A \rightarrow a \mid bBAA \mid cBAA \mid aCCBAA \mid bAA \mid cAA \mid aCCAA$

$S \rightarrow aC \mid bBAAC \mid cBAAC \mid aCCBAAC \mid bAAC \mid cAAC \mid aCCAAC \mid bBA \mid cBA \mid aCCBA \mid bA \mid cA \mid aCCA$

$B \rightarrow aACCB \mid bBAAACCB \mid cBAAACCB \mid aCCBAAACCB \mid bAAACCB \mid cAAACCB \mid aCCAAACCB \mid aACC \mid bBAAACC \mid cBAAACC \mid aCCBAAACC \mid bAAACC \mid cAAACC \mid aCCAAACC$