Linguaggi Liberi dal Contesto

Grammatiche e Linguaggi Liberi da Contesto

- Abbiamo visto che molti linguaggi non sono regolari. Consideriamo allora una classe piu' ampia di linguaggi, i **Linguaggi Liberi da Contesto** (CFL)
- i CFL sono stati usati nello studio dei linguaggi naturali dal 1950, e nello studio dei compilatori dal 1960.
- i CFL si definiscono usando una rappresentazione naturale e ricorsiva, le **grammatiche libere da contesto** (CFG)
- le grammatiche CFG sono la base della sintassi BNF (Backus-Naur-Form) dei linguaggi di programmazione. Nel *parser* viene generata in modo automatica una rappresentazione della struttura del programma (parse tree)
- Altre applicazioni di CGF piu' recenti:formati di documenti o XML, scambio di informazioni sul web

Cosa ci interessa?

- La definizione formale di CFL e CGF (**Linguaggi Liberi da Contesto**)
- la definizione del *albero sintattico* (o parse tree) che costituisce la rappresentazione di una stringa indotta da una grammatica
- gli automi a pila, e le proprieta' di chiusura dei CFL.

Esempio Informale di CGF

Consideriamo $L_{pal} = \{w \in \Sigma^* : w = w^R\}$ il linguaggio della stringhe palindrome
Per esempio: `otto` $\in L_{pal}$, `madamimadam` $\in L_{pal}$.
Sia $\Sigma = \{0, 1\}$,avremo

$$\epsilon, 0, 010, 0110, 01110 \in L_{pal}$$
$$0100, 01111 \notin L_{pal}$$

Applichiamo il Pumping Lemma

Il linguaggio $L_{pal} == \{w \in \Sigma^* : w = w^R\}$ non e' regolare.
Sia $\Sigma = \{0, 1\}$ e supponiamo che L_{pal} sia regolare.
Sia n dato dal pumping lemma. Consideriamo la stringa $w = 0^n 1 0^n \in L_{pal}$. Deve essere $w = xyz$ dove $y \neq \epsilon$ e $|xy| \leq n$.
Allora deve essere y composta sola da 0 (O^j con $j > 0$).
Per pumping lemma, la stringa ottenuta da w cancellando y appartiene al linguaggio, ovvero $xz \in L_{pal}$. Tuttavia $xz = 0^{n-j} 1 0^n$ non e' palindrome.

Una definizione ricorsiva

Tuttavia e' facile dare di L_{pal} una definizione induttiva

- **Base:** $\epsilon, 0$, e 1 sono palindromi.
- **Induzione:** Se w e' una palindrome, anche $0w0$ e $1w1$ lo sono.

Nessun altra stringa e' una palindrome.
Questo ragionamento puo' essere tradotto in una *grammatica formale*, che genera le stringhe appartenenti al linguaggio

La grammatica

Le CFG sono un modo formale per definizioni come quella per L_{pal}.

1. $P \rightarrow \epsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

0 e 1 sono simboli **terminali**
P e' una **variabile** (o **nonterminale**, o *categoria sintattica*)
P e' in questa gramatica anche il **simbolo iniziale**.
1–5 sono **produzioni** (o *regole*)

Definizione Formale

Una **grammatica libera da contesto** e' una quadrupla

$$G = (V, T, P, S)$$

dove
V e' un insieme finito di *variabili*.
T e' un insieme finito di *terminali*.
P e' un insieme finito di *produzioni* della forma $A \longrightarrow \alpha$, dove A e' una variabile e $\alpha \in (V \cup T)^*$
S e' una variabile distinta chiamata il *simbolo iniziale*.

Nota

In una produzione $A \rightarrow \alpha$ chiamiamo A la testa (variabile) e α il corpo della produzione.
Il significato della produzione e' che la variabile A puo' essere rimpiazzata con la stringa α (che e' una stringa di terminali e non terminali o ϵ)

Convenzioni

- A, B denotano non terminali, variabili
- X, Y denotano terminali o non terminali
- α, β denotano stringhe di terminali e non terminali
- x, y denotano stringhe di terminali

Esempio: palindrome

$G_{pal} = (\{P\}, \{0, 1\}, A, P)$, dove

$$A = \{P \to \epsilon, P \to 0, P \to 1, P \to 0P0, P \to 1P1\}$$

Per convenienza possiamo usare la forma

$$A = \{P \to \epsilon | 0 | 1 | 0P0 | 1P1\}$$

raggruppando le produzioni in base alla testa

Esempio: espressioni

Espressioni (semplici) in un tipico linguaggio di programmazione.
Gli operatori sono $+$ e *, e gli operandi sono identificatori, cioe' stringhe in $L((\mathbf{a}+\mathbf{b})(\mathbf{a}+\mathbf{b}+\mathbf{0}+\mathbf{1})^*)$
Le espressioni sono definite dalla grammatica

$$G = (\{E, I\}, T, P, E)$$

dove $T = \{+, *, (,), a, b, 0, 1\}$ e P e' il seguente insieme di produzioni:

$$\begin{array}{ll} E \to I & E \to E + E \\ E \to E * E & E \to (E) \end{array}$$

$$\begin{array}{ll} I \to a & I \to b \\ I \to Ia & I \to Ib \\ I \to I0 & I \to I1 \end{array}$$

Linguaggio definito da una grammatica

- tutte le stringhe di terminali **derivabili dal simbolo iniziale**, applicando le produzioni dalla testa al corpo
- bisogna definire formalmente il *passo di derivazione* e poi applicare la chiusura transitiva

Derivazione

Sia $G = (V, T, P, S)$ una CFG, $A \in V$,
$\{\alpha, \beta\} \subset (V \cup T)^*$, e $A \to \gamma \in P$.
Allora scriviamo

$$\alpha A \beta \underset{G}{\Rightarrow} \alpha \gamma \beta$$

o, se e' ovvia la G,

$$\alpha A \beta \Rightarrow \alpha \gamma \beta$$

e diciamo che da $\alpha A \beta$ *si deriva* $\alpha \gamma \beta$.
Definiamo $\overset{*}{\Rightarrow}$ la chiusura riflessiva e transitiva di $\Rightarrow$, cioe':
Base: Sia $\alpha \in (V \cup T)^*$. Allora $\alpha \overset{*}{\Rightarrow} \alpha$.
Induzione: Se $\alpha \overset{*}{\Rightarrow} \beta$, e $\beta \Rightarrow \gamma$, allora $\alpha \overset{*}{\Rightarrow} \gamma$.

Il Linguaggio definito dalla grammatica

- Se $G = (V, T, P, S)$ e' una CFG, allora il **linguaggio di** G e'

$$L(G) = \{w \in T^* : S \overset{*}{\underset{G}{\Rightarrow}} w\}$$

  cioe' l'insieme delle stringhe su T^* derivabili dal simbolo iniziale.
- Se G e' una CFG, chiameremo $L(G)$ un **linguaggio libero da contesto**.
- Esempio: $L(G_{pal})$ e' un linguaggio libero da contesto.

Esempi: derivazioni

Una derivazione di $w = a * (a + b00)$ da E nella grammatica delle espressioni:

$$E \Rightarrow E * E \Rightarrow I * E \Rightarrow a * E \Rightarrow a * (E) \Rightarrow$$

$$a * (E + E) \Rightarrow a * (I + E) \Rightarrow a * (a + E) \Rightarrow a * (a + I) \Rightarrow$$

$$a * (a + I0) \Rightarrow a * (a + I00) \Rightarrow a * (a + b00)$$

Esempi: diverse derivazioni

Nota: ad ogni passo potremmo avere varie regole tra cui scegliere, ad esempio
$I * E \Rightarrow a * E \Rightarrow a * (E)$, oppure
$I * E \Rightarrow I * (E) \Rightarrow a * (E)$.
Possono esistere diverse derivazioni della stringa $w = a * (a + b00)$ da E appartenente al linguaggio.
Queste due derivazioni per esempio *differiscono solo per l'ordine* di applicazione delle regole.

Esempi: derivazioni

Analogamente, non tutte le scelte corrispondono a derivazioni di una particolare stringa.
Per esempio, usando $E \Rightarrow E + E.$ non possiamo devivare la stringa $w = a * (a + b00)$ da E.
Ai fini della definizione del linguaggio e' chiaro che questo non e' un problema (basta che esista una derivazione di w).
Tuttavia, ai fini dell'applicazione del formalismo in pratica questi problemi sono ovviamente rilevanti.