UNIVERSITÀ
DEGLI STUDI
DI TORINO

Grammtiche e linguaggi liberi dal contesto (context-free

a.a. 2018-2019

Oltre I linguaggi regolari

I linguaggi regolari sono *troppo semplici*: le espressioni regolari non sono in grado di descrivere, ad esempio, le espressioni aritmetiche parentesizzate o la struttura a blocchi di un linguaggio.

Esempio informale: il linguaggio $\{a^n b^n \mid n > 0\}$

Supponiamo di voler utilizzare uno strumento formale (tipo le e.r.) per generare il lingaggio $L = \{a^n b^n \mid n > 0\}$

Abbiamo già visto che L non è regolare. Introduciamo due nuovi concetti:

- I simboli non terminali che non appartengono al linguaggio ma servono solo come supporto per la generazione delle stringhe corrette

- Una nozione di *regole di riscrittura* per i non terminali, per esempio :

(1) $S \rightarrow a\ S\ b$ (2) $S \rightarrow a\ b$

- Partendo da S quali stringhe (*senza* S) riusciamo a generare (notazione: $\Rightarrow$) applicando successivamente le regole 1 e 2 ? (Nota: qui 2 si può applicare una volta sola …)

Esempio $S \Rightarrow a\ S\ b \Rightarrow a\ a\ S\ b\ b \Rightarrow a\ a\ a\ b\ b\ b$

Si vede facilmente che $\{ x \mid S \Rightarrow^n x \text{ per } n>0\} = \{a^n b^n \mid n > 0\}$

Grammatiche libere dal contesto

Una grammatica **G libera dal contesto** (o **context-free**) e' una quadrupla **<V, Σ, P, S>** tale che:

V e' un insieme di simboli (**non terminali** o **variabili**)

Σ e' un insieme di simboli (**terminali**)

P e' un **insieme di regole di riscrittura** o **produzioni sintattiche**

S $\in$ **V** e' l'**assioma** o **start symbol** della grammatica

le regole sono coppie: $<A, \alpha>$, che scriveremo:

$$\mathbf{A} \rightarrow \alpha$$

dove $A \in V$ (A e' un simbolo non terminale) e α una stringa di simboli terminali e non terminali ($\alpha \in (V \cup \Sigma)^*$)

Ex. 1 : $< \{S\}, \{a, b\}, \{S \rightarrow aSb, S \rightarrow ab\}, S >$

$A \rightarrow \alpha_1 \mid . . . \mid \alpha_n$ abbrevia $A \rightarrow \alpha_1, . . . , A \rightarrow \alpha_n$

Ex 2: $< \{S,P\}, \{a,b,;\}, \{S \rightarrow S;P \mid P \ , P \rightarrow aP \mid bP \mid \varepsilon \}, S >$

Che linguaggio genera?

Derivazioni

In una grammatica G la stringa β **produce** o **si riscrive come** la stringa γ

$$\beta \Rightarrow \gamma$$

se $\beta, \gamma \in (V \cup \Sigma)^*$ e $\beta = \delta A \eta$ e $\gamma = \delta \alpha \eta$ e $A \rightarrow \alpha \in P$

$\beta \Rightarrow^n \gamma$ (β produce γ in n passi) se $\beta = \beta_0 \Rightarrow \beta_1 \Rightarrow \ldots \Rightarrow \beta_{n-1} \Rightarrow \beta_n = \gamma$

$\beta \Rightarrow^* \gamma$ se $\beta \Rightarrow^n \gamma$ per qualche $n \geq 0$ ($\Rightarrow^*$ chiusura riflessiva e transitiva della relazione $\Rightarrow$)

$\beta \Rightarrow^+ \gamma$ se $\beta \Rightarrow^n \gamma$ per qualche $n > 0$ ($\Rightarrow^+$ chiusura transitiva della relazione $\Rightarrow$)

Esempio:

$< \{S, T\}, \{a, b\}, \{S \rightarrow aTb \mid \varepsilon,\ T \rightarrow bSa\}, S >$

$aTb \Rightarrow abSab$ $\quad$ $abSab \Rightarrow abaTbab$

$S \Rightarrow aTb \Rightarrow abSab$

$S \Rightarrow^2 abSab$ $\quad$ $S \Rightarrow^5 abababab$

Linguaggio generato da una grammatica

il **linguaggio generato dal non terminale A in una grammatica** (notazione $L_A(G)$) e' l'insieme delle stringhe di terminali prodotte partendo da A: $L_A(G) = \{x \mid x \in \Sigma^* \;\&\; A \Rightarrow^+ x\}$

il **linguaggio generato da una grammatica** (notazione L(G)) e' l'insieme delle stringhe di terminali prodotte dall'assioma

$$L(G) = \{x \mid x \in \Sigma^* \;\&\; S \Rightarrow^+ x\}$$

Esempio:

$G = <\{S, T\}, \{a, b\}, \{S \rightarrow aTb \mid \varepsilon, T \rightarrow bSa\}, S>$

$T \Rightarrow bSa \Rightarrow ba$

$T \Rightarrow bSa \Rightarrow baTba \Rightarrow babSaba \Rightarrow bababa$

$S \Rightarrow \varepsilon$ $\qquad$ $S \Rightarrow ab$

$S \Rightarrow aTb \Rightarrow abSab \Rightarrow abaTbab \Rightarrow ababSabab \Rightarrow abababab$

$L_T(G) = \{ba, bababa, babababababa, \ldots\} = \{(ba)^{2n+1} \mid n \geq 0\}$

$L(G) = \{\varepsilon, abab, abababab, \ldots\} = \{(ab)^{2n} \mid n \geq 0\}$

Esempi

1. Espressioni regolari su $\Sigma = \{0, 1\}$:

   $G = < \{E\}, \{\Phi, 0, 1, (,), +, *\}, \{E \to \Phi \mid 0 \mid 1 \mid E+E \mid EE \mid E^* \mid (E)\}, E >$

   $E \Rightarrow EE \Rightarrow E(E) \Rightarrow E(E+E) \Rightarrow 1(E+E) \Rightarrow 1(0+E) \Rightarrow 1(0+1)$

   $E \Rightarrow EE \Rightarrow E\,E^* \Rightarrow E(E)^* \Rightarrow 1(E+E)^* \Rightarrow^2 1(0+1)^*$

2. $S \to aSb \mid ab$ $\quad L(G) = \{a^n b^n \,/\, n > 0\}$

3. $S \to aB \mid bA$
   $A \to a \mid aS \mid bAA$
   $B \to b \mid bS \mid aBB$

   $L(G) = \{x \mid x \in \{a, b\}^* \ \& \ |x|_a = |x|_b\}$

ε-produzioni (non terminali annullabili)

Non sono esclude dalla definizione produzioni del tipo $A \rightarrow \varepsilon$

Tali produzioni sono dette *ε-produzioni* e il relativo non terminale non terminale *annullabile.*

ESEMPIO

$S \rightarrow aSb \mid \varepsilon$

Questa grammatica genera il linguaggio $\{a^n b^n \mid n \geq 0\}$

Nota: in questo caso $\varepsilon \in L(G)$.

Esempio: documento HTML

Le cose che *detesto*:

1. le persone che gettano la carta in terra
2. coloro che guidano lentamente nella corsia di sorpasso

<P>Le cose che <EM> detesto </EM>:
<OL>
<LI> le persone che gettano la carta in terra
<LI> coloro che guidano lentamente nella corsia di sorpasso
</OL>

Porzione di grammatica per HTML

<P>Le cose che <EM> detesto </EM>:
<OL>
<LI> le persone che gettano la carta in terra
<LI> coloro che guidano lentamente nella corsia di sorpasso
</OL>

Nota: Text si poteva esprimere anche con una e.r.

- *Char* → a | A | …
- *Text* → ε | *Char Text*
- *Doc* → ε | *Element Doc*
- *Element* → *Text* | <EM> *Doc* </EM> |
- | <P> *Doc* | <OL> *List* </OL>
- *ListItem* → <LI> *Doc*
- *List* → ε | *ListItem List*

Derivazione

Char	→	a | A | …
Text	→	ε | *Char Text*
Doc	→	ε | *Element Doc*
Element	→	*Text* | <EM> *Doc* </EM> | <P> *Doc* | <OL> *List* </OL>
ListItem	→	<LI> *Doc*
List	→	ε | *ListItem List*

Doc ⇒ *Element Doc* ⇒ <P> *Doc Doc* ⇒ <P> *Doc Element Doc* ⇒

⇒ <P> *Element Doc* ⇒ <P> *Text Doc* ⇒ <P> *Text Element Doc* ⇒

⇒ <P> *Text* <EM> *Element Doc* </EM> *Doc* ⇒

⇒ <P> *Text* <EM> *Text Doc* </EM> *Doc* ⇒

⇒ <P> *Text* <EM> *Text* </EM> *Doc* ⇒

⇒ <P> *Text* <EM> *Text* </EM> <OL> *List* </OL> ⇒

⇒ <P> *Text* <EM> *Text* </EM> <OL> *ListItem List* </OL> ⇒

⇒ <P> *Text* <EM> *Text* </EM> <OL> <LI> *Doc List* </OL> ⇒

⇒ <P> *Text* <EM> *Text* </EM> <OL> <LI> *Element Doc List* </OL> ⇒

⇒ <P> *Text* <EM> *Text* </EM> <OL> <LI> *Text Doc List* </OL> ⇒

⇒ <P> *Text* <EM> *Text* </EM> <OL> <LI> *Text List* </OL> ⇒

⇒ <P> *Text* <EM> *Text* </EM> <OL> <LI> *Text ListItem List* </OL> ⇒*

⇒ <P> *Text* <EM> *Text* </EM> <OL> <LI> *Text* <LI> *Text* </OL> ⇒

Linguaggi liberi dal contest (context-free)

Un **linguaggio è libero dal contesto** se esiste una grammatica libera dal contesto che lo genera.

Due grammatiche G e G' sono **debolmente equivalenti** se generano lo stesso linguaggio, cioe' L(G) = L(G')

Esempio:

$G = \langle\{S\}, \{a, b\}, \{S \rightarrow aSb \mid ab\}, S\rangle$

$G' = \langle\{S, A, B\}, \{a, b\}, \{S \rightarrow ASB \mid AB, A \rightarrow a, B \rightarrow b\}, S\rangle$

$L(G) = L(G') = \{ab, aabb, aaabbb, \ldots\} = \{a^n b^n \mid n > 0\}$

Simboli e produzioni inutili

Una grammatica può presentare simboli terminali e non terminali **inutili**, nel senso che **non concorrono a formare il linguaggio** generato.
Sono di due tipi:

- Non terminali **non definiti**, che generano cioè linguaggi vuoti (A e' definito se $L_A(G) \neq \Phi$ ossia se $A \Rightarrow^+ x \in \Sigma^*$)
- Terminali e non terminali **non raggiungibili** dall'assioma, che non occorrono cioè in nessuna derivazione dallo start symbol ($x \in \Sigma \cup V$ e' raggiungibile se $S \Rightarrow^+ \alpha x \beta$)

Simboli e produzioni inutili

Per esempio nella grammatica

$$S \rightarrow AB \mid C , \quad A \rightarrow a, \quad C \rightarrow c, \quad D \rightarrow d$$

B è non definito e D è irraggiungibile.

Le produzioni che contengono simboli inutili possono essere eliminate senza alterare il linguaggio definito dalla grammatica.

Per esempio nella grammatica precedente *B è non definito e D è irraggiungibile*. Quindi si possono eliminare le relative produzioni:

$$S \rightarrow \cancel{AB} \mid C , \quad A \rightarrow a, \quad C \rightarrow c, \quad \cancel{D \rightarrow d}$$

Adesso però anche A diventa *inutile* e si può eliminare

$$S \rightarrow C , \quad \cancel{A \rightarrow a}, \quad C \rightarrow c$$

ottenendo infine $S \rightarrow C , \quad C \rightarrow c$

.

Derivazioni sinistre (left-most) e destre (right-most)

Tra le possibili derivazioni per ottenere una stessa forma sentenziale (in particolare una parola) *due* sono importanti:

- derivazione sinistra (o left-most): $\Rightarrow_{lm}$. Ad *ogni* passo si espande sempre il nonteminale più *a sinistra* nella forma sentenziale

- derivazione destra (o right-most): $\Rightarrow_{rm}$. ad *ogni* passo si espande sempre il nonteminale più *a destra* nella forma sentenziale

Per esempio data la grammatica con le produzioni

$S \rightarrow aB \mid bA$, $A \rightarrow a \mid aS \mid bAA$, $B \rightarrow b \mid bS \mid aBB$

Derivazione sinistra:

$S \Rightarrow_{lm} aB \Rightarrow_{lm} aaBB \Rightarrow_{lm} aabSB \Rightarrow_{lm} aabaBB \Rightarrow_{lm} aababB \Rightarrow_{lm} aababb$

Derivazione destra:

$S \Rightarrow_{rm} aB \Rightarrow_{rm} aaBB \Rightarrow_{rm} aaBb \Rightarrow_{rm} aabSb \Rightarrow_{rm} aabaBb \Rightarrow_{rm} aababb$