Grammatiche formali e riconoscitori, gerarchia di Chomsky

Lezione 20 di Fondamenti di informatica

Docente: Giuseppe Scollo

Università di Catania
Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica, I livello, AA 2008-09

Indice

algebra delle stringhe

rivediamo alcuni concetti noti e introduciamo **convenzioni notazionali** comuni nella teoria dei linguaggi formali:

- Σ : un **alfabeto** (insieme di simboli), di solito **finito**
- Σ^* : il monoide delle stringhe su Σ
- ε : la stringa vuota, elemento neutro del monoide
- $\_\_$: concatenazione di stringhe (operatore binario infisso "invisibile")
- $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$: il semigruppo delle stringhe non vuote su Σ
- $|\_|$: lunghezza di una stringa (definibile per induzione... su che cosa?)
- Σ^n : l'insieme delle stringhe su Σ di lunghezza n

altre operazioni e proprietà facilmente definibili (ad es. per induzione sulla lunghezza):

- **inversa** x^R di una stringa x
- **prefisso** x di una stringa xy, **proprio** se $y \neq \varepsilon$
- **suffisso** y di una stringa xy, **proprio** se $x \neq \varepsilon$
- **sottostringa** y di una stringa xyz, **propria** se $xz \neq \varepsilon$

linguaggi formali

un **linguaggio** L sull'alfabeto Σ è un insieme di stringhe su Σ: $L \subseteq \Sigma^*$

esempi: $\varnothing$, Σ, Σ^*, Σ^+, $\{\varepsilon\}$, $\{ x \in \Sigma^* \mid x \text{ è palindroma} \}$, ...

operazioni sui linguaggi (su un alfabeto Σ):

- le **operazioni booleane** standard di ogni algebra di insiemi
  unione, intersezione, complemento, differenza, ...
- **concatenazione:** $LL' = \{ xy \mid x \in L, y \in L' \}$
  (se gli alfabeti di L, L' sono diversi, quello di LL' ne è l'unione)
- **chiusura di Kleene:** $L^* = \{ x \mid x \in L^n, n \geq 0 \}$
  dove le iterate della concatenazione sono induttivamente definite:
  $L^0 = \{ \varepsilon \}$, $L^{n+1} = L^n L = L L^n$
- **chiusura positiva:** $L^+ = L^* \setminus L^0$

operazioni e proprietà dei linguaggi

perché estendere le operazioni delle stringhe ai linguaggi?

> ciò è utile nella trattazione algebrica delle grammatiche, dove un simbolo non terminale può essere visto come la designazione di un **insieme di stringhe** (cioè un linguaggio) sull'alfabeto terminale, mentre un simbolo terminale designa il singoletto che lo contiene

le algebre di interesse a tal fine sono note come **algebre di Kleene**

proprietà di frequente interesse di linguaggi, ad es. nella codifica dell'informazione, sono le seguenti, raccolte in una sola definizione:

> un linguaggio L gode della **proprietà prefissa (suffissa)** se nessuna stringa del linguaggio è un prefisso (suffisso) proprio di un'altra stringa del linguaggio

esempio:

> il linguaggio $L = \{a^i b \mid i \geq 0\}$ gode della proprietà prefissa, ma non della proprietà suffissa

grammatiche formali

si tratta di strutture matematiche atte alla **definizione generativa** di linguaggi formali

> si è già vista la definizione di grammatica libera $G_L = (V_T, V_N, P, s_0)$, quale struttura di generazione del linguaggio L

una classe più generale di grammatiche, che generano una più ampia classe di linguaggi, si ha dalla definizione di grammatica libera ammettendo nella parte sinistra delle produzioni, invece che solo un simbolo non terminale, una stringa sull'alfabeto $V_T \cup V_N$ che contenga almeno un simbolo non terminale

> resta valida la condizione che il simbolo iniziale s_0 sia un elemento di V_N, nonché la definizione di ***L, linguaggio generato da*** G_L, quale sottoinsieme di V_T^* costituito dalle stringhe terminali ***derivabili da*** s_0 con le produzioni P

quest'ultima proprietà è definita come segue: siano $\alpha, \beta, \gamma, \delta$ stringhe di $V_T \cup V_N$:

derivazione diretta $\Rightarrow_G$: $\alpha\beta\gamma \Rightarrow_G \alpha\delta\gamma$ se $\beta \to \delta \in P$

derivazione $\Rightarrow_G^*$: chiusura riflessiva e transitiva della derivazione diretta

un esempio stimolante

convenzioni notazionali: d'ora innanzi designiamo con
- $\mathcal{T}$: l'alfabeto (o vocabolario) terminale
- $\mathcal{N}$: l'alfabeto (o vocabolario) non terminale
- lettere maiuscole A,B,C,...: simboli non terminali
- lettere minuscole iniziali a,b,c,...: simboli terminali
- lettere minuscole finali u,v,w,x,y,z,...: stringhe su $\mathcal{T}$
- lettere minuscole greche $\alpha,\beta,\gamma,\delta,\ldots$: stringhe su $\mathcal{T}\cup\mathcal{N}$

problema: trovare una grammatica che generi il linguaggio $\{a^n b^n c^n \mid n \geq 0\}$

soluzione: con $\mathcal{N}=\{A,B,C\}$ e $s_0=A$, la grammatica che ha le seguenti produzioni:

A	→	aABC
A	→	aBC
CB	→	BC
aB	→	ab
bB	→	bb
bC	→	bc
cC	→	cc

fatto: non è generabile da una grammatica libera (come si vedrà più avanti)

esempio: derivazione della stringa aabbcc

A → aABC → aaBCBC → aaBBCC → aabBCC
→ aabbCC → aabbcC → aabbcc

gerarchia di Chomsky

classificazione di Chomsky delle grammatiche, per restrizioni crescenti sulla forma delle produzioni $\alpha\to\beta$:
- **tipo 0, o *unrestricted* :** nessuna restrizione
- **tipo 1, o *context-sensitive* :** $|\alpha| \leq |\beta|$
- **tipo 2, o *context-free* :** $|\alpha| = 1$
- **tipo 3, o *right-linear* :** $|\alpha| = 1,\ \beta \in \mathcal{T}^* \cup \mathcal{T}^*\mathcal{N}$

si rammenti che α deve sempre contenere almeno un simbolo di $\mathcal{N}$
- ne consegue che ε non è derivabile con grammatiche di tipo 1, mentre può esserlo con quelle degli altri tipi

se si escludono le produzioni aventi $\beta=\varepsilon$, le suddette classi di grammatiche sono ordinate da inclusione propria, così come lo sono le corrispondenti classi di linguaggi da esse generabili, ai quali si applica la stessa terminologia
- due grammatiche si dicono **equivalenti** se generano lo stesso linguaggio

le grammatiche di tipo 1 devono il loro nome al fatto di avere sempre una grammatica equivalente le cui produzioni sono della forma $\gamma A\delta\to\gamma\beta\delta$, con $A\in\mathcal{N}$ e β non vuota

automi e riconoscitori

data una grammatica G, il **problema di decisione del linguaggio** L_G consiste nel determinare se, data una stringa x sul suo alfabeto terminale, $x \in L_G$

un **riconoscitore** è (un modello matematico di) una procedura di calcolo per risolvere il problema di decisione di un linguaggio

costituenti essenziali dei riconoscitori:

Q : un insieme (finito) di **stati**, fra i quali:

$q_0 \in Q$: uno **stato iniziale**

$F \subseteq Q$: un insieme di stati **finali**, o **accettanti**

δ : una **dinamica**, che determina le transizioni di stato ed eventuali altre azioni del riconoscitore in base all'input e allo stato corrente

un **automa a stati finiti (FSA)** non ha altri costituenti, e la sua dinamica è limitata alle transizioni di stato e alla lettura sequenziale dell'input

un FSA è **deterministico (DFA)** se la sua dinamica assegna al più uno stato (di arrivo) alla transizione determinata dallo stato corrente e da un simbolo in input, altrimenti è **non deterministico (NFA)**

gerarchia di riconoscitori

ulteriori costituenti, e altre possibilità di azione oltre alle transizioni di stato, caratterizzano classi di riconoscitori più sofisticati degli automi:

possibilità di **spostamento bidirezionale** del dispositivo di lettura sul supporto dell'input ("nastro")

possibilità di **scrittura sul nastro**

disponibilità di una **memoria ausiliaria**

un riconoscitore **accetta una stringa** x in input se la sua dinamica permette una sequenza di transizioni dallo stato iniziale a un stato accettante

il **linguaggio definito da un riconoscitore** è l'insieme delle stringhe che accetta

le classi di linguaggi definite nella gerarchia di Chomsky possono essere caratterizzate in base alle caratteristiche di corrispondenti classi di riconoscitori:

tipo 3: automi a stati finiti

tipo 2: automi con pila (*pushdown automata (PDA)*

tipo 1: automi *linear bounded*

tipo 0: macchine di Turing