Linguaggi e Computabilità

UniShare

Davide Cozzi
@dlcgold

Gabriele De Rosa
@derogab

Federica Di Lauro
@f_dila

Indice

Capitolo 1

Introduzione

Questi appunti sono presi a lezione. Per quanto sia stata fatta una revisione è altamente probabile (praticamente certo) che possano contenere errori, sia di stampa che di vero e proprio contenuto. Per eventuali proposte di correzione effettuare una pull request. Link: `https://github.com/dlcgold/Appunti`.
Grazie mille e buono studio!

1.1 Definizioni

- un **linguaggio** è un insieme di stringhe che può essere generato mediante un dato meccanismo con delle date caratteristiche; un linguaggio può essere riconosciuto, ovvero dando in input una stringa un meccanismo può dirmi se appartiene o meno ad un linguaggio. I meccanismi che generano linguaggi si chiamano *grammatiche*, quelli che li riconoscono *automi*. I linguaggi formali fanno parte dell'informatica teorica *(TCS)*
- si definisce **alfabeto** come un insieme finito e non vuoto di simbolo (come per esempio il nostro alfabeto o le cifre da 0 a 9). Solitamente si indica con Σ o Γ
- si definisce **stringa** come una sequenza finita di simboli (come per esempio una parola o una sequenza numerica). La stringa vuota è una sequenza di 0 simboli, e si indica con ε o λ
- si definisce **lunghezza di una stringa** il numero di simboli che la compone (ovviamente contando ogni molteplicità). Se si ha $w \in \Sigma^*$ è una stringa w con elementi da Σ^* (insieme di tutte le stringhe di tutte le lunghezze possibili fatte da Σ), allora $|w|$ è la lunghezza di w, inoltre $|\varepsilon| = 0$.

- si definisce **potenza di un alfabeto** Σ^k come l'insieme di tutte le sequenze (espressi come stringhe e non simboli) di lunghezza $k \in \mathbb{N}$, $k > 0$ ottenibili da quell'alfabeto (se Σ^2 si avranno tutte le sequenza di 2 elementi etc...). Se ho $k = 1$ si ha $\Sigma^1 \neq \Sigma$ in quanto ora ho stringhe e non simboli. Se ho $k = 0$ ho $\Sigma^0 = \varepsilon$. Dato k ho $|\Sigma|$ che è la cardinalità dell'insieme Σ (e non la sua lunghezza come nel caso delle stringhe); sia $w \in \Sigma^k = a_1, a_2, ..., a_k$, $a_i \in \Sigma$ e $|\Sigma| = q$ ora:

$$|\Sigma^k| = q^k$$

- si definisce Σ^* come **chiusura di Kleene** che è l'unione infinita di Σ^k ovvero

$$\Sigma * = \Sigma^0 \cup \Sigma^1 \cup ... \cup \Sigma^k$$

- si ha che Σ^+ è l'unione per $k \geq 1$ di Σ^k ovvero:

$$\Sigma + = \Sigma^1 \cup \Sigma^2 \cup ... \cup \Sigma^k = \Sigma^* - \Sigma^0$$

  per esempio, per l'insieme $\{0, 1\}$ si ha:

$$\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 100, 000, ...\}$$

- quindi un **linguaggio** L è un insieme di stringhe e:

$$L \subseteq \Sigma^*$$

  si hanno sottoinsiemi particolari, come l'insieme vuoto, che resta però un linguaggio, il **linguaggio vuoto** e $\emptyset \in \Sigma^k$, $|\emptyset| = 0$ che è diverso dal linguaggio che contiene la stringa vuota $|\varepsilon| = 1$ (che conta come una stringa). Inoltre $\Sigma^* \subseteq \Sigma^*$ che ha lunghezza infinita. Posso concatenare due stringhe con un punto: $a \cdot b \cdot c = abc$ e $a \cdot \varepsilon = a$. Ovviamente la stringa concatenata è lunga come la somma delle lunghezze delle stringhe che la compongono. Vediamo qualche esempio di linguaggio:

  - il linguaggio di tutte le stringhe che consistono in n 0 seguiti da n 1:

$$\{\varepsilon, 01, 0011, 000111, ...\}$$

  - l'insieme delle stringhe con un uguale numero di 0 e di 1:

$$\{\varepsilon, 01, 10.0011, 0101.1001, ..\}$$

  - l'insieme dei numeri binari il cui valore è un numero primo:

$$\{\varepsilon, 10, 11, 101, 111, 1011, ...\}$$

- Σ^* è un linguaggio per ogni alfabeto Σ
- $\emptyset$, il linguaggio vuoto, e $\{\varepsilon\}$ sono un linguaggio rispetto a qualunque alfabeto

Prendiamo un alfabeto $\Sigma = \{0, 1\}$ con la sua chiusura di Kleen $\Sigma = \{0, 1\}^*$. Quando si ha un input si può avere un problema di decisione, P, che dia come output "si" o "no". Posso avere un problema di decisione (o *membership*) su $w \in \Sigma = \{0, 1\}^*$, con w stringa, che dia in output "si" o "no". Un linguaggio L sarà:

$$L = \{w \in \{0, 1\}^* \mid P(w) = si$$

quindi si ha che:

$$\Sigma^* \backslash L = \{P(w) = no\}$$

Vediamo ora un esempio di *Context Free Language (CFL)*, costruito a partire da una *Context Free Grammar (CFG)*:

Esempio 1. *Sia* $\Sigma = \{0, 1\}$ *e* L_{pal} = "*stringhe palindrome binarie*". *Quindi, per esempio,* $0110 \in L$, $11011 \in L$ *ma* $10010 \notin L$. *Si ha che* ε*, la stringa vuota, appartiene a* L. *Diamo una definizione ricorsiva:*

- ***base:*** ε, 0 $1 \in L_{pal}$
- ***passo:*** *se* w *è palindroma allora* $0w0$ *è palindromo e* $1w1$ *è palindromo*

una variabile generica S *può sottostare alle regole di produzione di una certa grammatica. In questo caso si ha uno dei seguenti:*

$$S \to \varepsilon,\ S \to 0,\ S \to 1,\ S \to 0S0,\ S \to 1S1$$

Si ha che una grammatica G è una quadrupla $G = (V, T, P, S$ con:

- V simboli variabili
- T simboli terminali, ovvero i simboli con cui si scrivono le stringhe alla fine
- P regole di produzione
- S variabile di partenza *start*

riprendiamo l'esempio sopra:

Esempio 2.

$$G_{pal} = (V = \{S\},\, T = \{0, 1\},\, P,\, S)$$

con:

$$P = \{S \rightarrow \varepsilon,\, S \rightarrow 0,\, S \rightarrow 1,\, S \rightarrow 0S0,\, S \rightarrow 1S1\}$$

Si può ora costruire un algoritmo per creare una stringa palindroma a partire dalla grammatica G:

$$\underbrace{S}_{\textit{start}} \quad \underbrace{\rightarrow}_{\textit{applico una regola}} \quad 1S1 \rightarrow 01S10 \rightarrow \quad \underbrace{01010}_{\textit{sostituisco variabile}}$$

con S, $1S1$ *e* $01S10$ *che sono forme sentenziali. Posso così ottenere tutte le possibili stringhe. Esiste anche una forma abbreviata:*

$$S \rightarrow \varepsilon|o|1|0S0|1S1$$

Non si fanno sostituzioni in parallelo, prima una S *e poi un'altra*

Si hanno 4 grammatiche formali:

- **tipo 0:** non si hanno restrizioni sulle regole di produzione, $\alpha \rightarrow \beta$. Sono linguaggi ricorsivamente numerabili e sono rappresentati dalle *macchine di Turing*, deterministiche o non deterministiche (la macchina di Turing è un automa)

- **tipo 1:** il lato sinistro della produzione (*testo*) ha lunghezza uguale a quello destro (*corpo*). Sono grammatiche dipendenti dal contesto (*contestuali*) e come automa hanno *la macchina di Turing che lavora in spazio lineare*:
$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 B \alpha_2$$
con α_1 e α_2 detti *contesto* e α_1, α_2, $\beta \in (V \cup T)^*$

- **tipo 2:** sono quelle libere dal contesto, context free. Come regola ha $A \rightarrow \beta$ con $A \in V$ e $\beta \in V \cup T)^*$ e come automa ha gli *automi a pila non deterministici*

- *tipo 3:* sono le grammatiche *regolari*. Come regole ha $A \rightarrow \alpha B$ e $A \rightarrow \alpha$ con $A, B \in V$ e $\alpha \in T$. Come automi ha gli *automi a stato finito deterministici o non deterministici*