ELEMENTI DI INFORMATICA TEORICA

Parte 2:
Linguaggi Formali e Automi

2.1 Linguaggi e grammatiche formali

Giovanni Amendola

Corso di laurea triennale in Informatica
Università della Calabria

2 aprile 2022

Anno Accademico 2021/2022

Linguaggi regolari

Definizione: **Linguaggio regolare**

Sia Σ un alfabeto. Un **linguaggio regolare** $\mathcal{L} \subseteq \Sigma^*$ è definito induttivamente come segue:

- Il linguaggio $\emptyset$ e i linguaggi formati da un solo simbolo $a \in \Sigma$ sono regolari.
- Se $\mathcal{L}_1$ e $\mathcal{L}_2$ sono regolari, allora $\mathcal{L}_1 \cup \mathcal{L}_2$, $\mathcal{L}_1 \cdot \mathcal{L}_2$ e $(\mathcal{L}_1)^*$ sono regolari.

Linguaggi regolari

Definizione: **Linguaggio regolare**

Sia Σ un alfabeto. Un **linguaggio regolare** $\mathcal{L} \subseteq \Sigma^*$ è definito induttivamente come segue:

- Il linguaggio $\emptyset$ e i linguaggi formati da un solo simbolo $a \in \Sigma$ sono regolari.
- Se $\mathcal{L}_1$ e $\mathcal{L}_2$ sono regolari, allora $\mathcal{L}_1 \cup \mathcal{L}_2$, $\mathcal{L}_1 \cdot \mathcal{L}_2$ e $(\mathcal{L}_1)^*$ sono regolari.

- Un linguaggio regolare è **chiuso** sotto unione, concatenazione e stella.

Linguaggi regolari

Definizione: **Linguaggio regolare**

Sia Σ un alfabeto. Un **linguaggio regolare** $\mathcal{L} \subseteq \Sigma^*$ è definito induttivamente come segue:

- Il linguaggio $\emptyset$ e i linguaggi formati da un solo simbolo $a \in \Sigma$ sono regolari.
- Se $\mathcal{L}_1$ e $\mathcal{L}_2$ sono regolari, allora $\mathcal{L}_1 \cup \mathcal{L}_2$, $\mathcal{L}_1 \cdot \mathcal{L}_2$ e $(\mathcal{L}_1)^*$ sono regolari.

- Un linguaggio regolare è **chiuso** sotto unione, concatenazione e stella.
- Il linguaggio $\{\epsilon\}$ è regolare.

Linguaggi regolari

Definizione: **Linguaggio regolare**

Sia Σ un alfabeto. Un **linguaggio regolare** $\mathcal{L} \subseteq \Sigma^*$ è definito induttivamente come segue:

- Il linguaggio $\emptyset$ e i linguaggi formati da un solo simbolo $a \in \Sigma$ sono regolari.
- Se $\mathcal{L}_1$ e $\mathcal{L}_2$ sono regolari, allora $\mathcal{L}_1 \cup \mathcal{L}_2$, $\mathcal{L}_1 \cdot \mathcal{L}_2$ e $(\mathcal{L}_1)^*$ sono regolari.

- Un linguaggio regolare è **chiuso** sotto unione, concatenazione e stella.
- Il linguaggio $\{\epsilon\}$ è regolare.

Teorema: **Linguaggi regolari**

Un linguaggio è regolare sse è generato da una grammatica regolare.

Espressioni regolari

- Le espressioni regolari sono un metodo per rappresentare linguaggi.
- Ogni espressione regolare E corrisponde ad un linguaggio $\mathcal{L}(E)$.

Espressioni regolari

- Le espressioni regolari sono un metodo per rappresentare linguaggi.
- Ogni espressione regolare E corrisponde ad un linguaggio $\mathcal{L}(E)$.

Definizione: Espressione regolare

Un'espressione regolare è definita su $\Sigma \cup \{^*, (,), \cdot, +, \emptyset\}$ in modo induttivo

- Un elemento $a \in \Sigma \cup \{\emptyset\}$ è un'espressione regolare.
- Se E ed F sono espressioni regolari, allora
  - $E + F$ è una espressione regolare ($+$ corrisponde all'unione)
  - $E \cdot F$ è una espressione regolare ($\cdot$ corrisponde alla concatenazione)
  - E^* è una espressione regolare ($*$ corrisponde a star).

Espressioni regolari

- Le espressioni regolari sono un metodo per rappresentare linguaggi.
- Ogni espressione regolare E corrisponde ad un linguaggio $\mathcal{L}(E)$.

Definizione: **Espressione regolare**

Un'espressione regolare è definita su $\Sigma \cup \{^*, (,), \cdot, +, \emptyset\}$ in modo induttivo

- Un elemento $a \in \Sigma \cup \{\emptyset\}$ è un'espressione regolare.
- Se E ed F sono espressioni regolari, allora
  - $E + F$ è una espressione regolare ($+$ corrisponde all'unione)
  - $E \cdot F$ è una espressione regolare ($\cdot$ corrisponde alla concatenazione)
  - E^* è una espressione regolare ($*$ corrisponde a star).

Esempio: **Espressione regolare**

Insieme di tutte le stringhe che iniziano con a, proseguono con un numero arbitrario di b (anche nullo), terminano con la stringa bab o con aba:

Espressioni regolari

- Le espressioni regolari sono un metodo per rappresentare linguaggi.
- Ogni espressione regolare E corrisponde ad un linguaggio $\mathcal{L}(E)$.

Definizione: **Espressione regolare**

Un'espressione regolare è definita su $\Sigma \cup \{^*, (,), \cdot, +, \emptyset\}$ in modo induttivo

- Un elemento $a \in \Sigma \cup \{\emptyset\}$ è un'espressione regolare.
- Se E ed F sono espressioni regolari, allora
  - $E + F$ è una espressione regolare ($+$ corrisponde all'unione)
  - $E \cdot F$ è una espressione regolare ($\cdot$ corrisponde alla concatenazione)
  - E^* è una espressione regolare ($*$ corrisponde a star).

Esempio: **Espressione regolare**

Insieme di tutte le stringhe che iniziano con a, proseguono con un numero arbitrario di b (anche nullo), terminano con la stringa bab o con aba:

$$ab^*(bab + aba)$$

Espressioni regolari

Linguaggi corrispondenti alle espressioni regolari

Espressioni regolari

Linguaggi corrispondenti alle espressioni regolari

- $\mathcal{L}(\emptyset) = \emptyset$.

Espressioni regolari

Linguaggi corrispondenti alle espressioni regolari

- $\mathcal{L}(\emptyset) = \emptyset$.
- Se $a \in \Sigma$, allora $\mathcal{L}(a) = \{a\}$.

Espressioni regolari

Linguaggi corrispondenti alle espressioni regolari

- $\mathcal{L}(\emptyset) = \emptyset$.
- Se $a \in \Sigma$, allora $\mathcal{L}(a) = \{a\}$.
- Se E ed F sono espressioni regolari, allora
  - $\mathcal{L}(E + F) = \mathcal{L}(E) \cup \mathcal{L}(F)$
  - $\mathcal{L}(E \cdot F) = \mathcal{L}(E) \cdot \mathcal{L}(F)$
  - $\mathcal{L}(E^*) = (\mathcal{L}(E))^*$

Espressioni regolari

Linguaggi corrispondenti alle espressioni regolari

- $\mathcal{L}(\emptyset) = \emptyset$.
- Se $a \in \Sigma$, allora $\mathcal{L}(a) = \{a\}$.
- Se E ed F sono espressioni regolari, allora
  - $\mathcal{L}(E + F) = \mathcal{L}(E) \cup \mathcal{L}(F)$
  - $\mathcal{L}(E \cdot F) = \mathcal{L}(E) \cdot \mathcal{L}(F)$
  - $\mathcal{L}(E^*) = (\mathcal{L}(E))^*$

Esempio

Consideriamo $E = a(b^*(bab + aba))$. Calcoliamo $\mathcal{L}(E)$.

Espressioni regolari

Linguaggi corrispondenti alle espressioni regolari

- $\mathcal{L}(\emptyset) = \emptyset$.
- Se $a \in \Sigma$, allora $\mathcal{L}(a) = \{a\}$.
- Se E ed F sono espressioni regolari, allora
  - $\mathcal{L}(E + F) = \mathcal{L}(E) \cup \mathcal{L}(F)$
  - $\mathcal{L}(E \cdot F) = \mathcal{L}(E) \cdot \mathcal{L}(F)$
  - $\mathcal{L}(E^*) = (\mathcal{L}(E))^*$

Esempio

Consideriamo $E = a(b^*(bab + aba))$. Calcoliamo $\mathcal{L}(E)$.

- $\mathcal{L}(E) = \mathcal{L}(a(b^*(bab + aba))) = \mathcal{L}(a) \cdot \mathcal{L}(b^*(bab + aba)) =$
  $= \{a\} \cdot (\mathcal{L}(b^*) \cdot \mathcal{L}(bab + aba)) = \{a\} \cdot ((\mathcal{L}(b))^* \cdot (\mathcal{L}(bab) + \mathcal{L}(aba))) =$
  $= \{a\} \cdot (\{b\}^* \cdot (\{bab\} + \{aba\})) = \{a\} \cdot \{b\}^* \cdot \{bab, aba\}$
- $\mathcal{L}(E) = \{abab, aaba, abbab, ababa, abbbab, abbaba, abbbbab, ...\}$

Espressioni regolari

Proprietà delle operazioni $+$ e $\cdot$

Siano E, F e G espressioni regolari. Allora

- $((E + F) + G) = (E + (F + G))$.
- $((E \cdot F) \cdot G) = (E \cdot (F \cdot G))$.
- $E \cdot (F + G) = E \cdot F + E \cdot G$.

Espressioni regolari

Proprietà delle operazioni $+$ e $\cdot$

Siano E, F e G espressioni regolari. Allora

- $((E + F) + G) = (E + (F + G))$.
- $((E \cdot F) \cdot G) = (E \cdot (F \cdot G))$.
- $E \cdot (F + G) = E \cdot F + E \cdot G$.

Per usare meno parentesi si possono considerare le seguenti precedenze:

$$* \succ \cdot \succ +$$

Espressioni regolari

Proprietà delle operazioni $+$ e $\cdot$

Siano E, F e G espressioni regolari. Allora

- $((E + F) + G) = (E + (F + G))$.
- $((E \cdot F) \cdot G) = (E \cdot (F \cdot G))$.
- $E \cdot (F + G) = E \cdot F + E \cdot G$.

Per usare meno parentesi si possono considerare le seguenti precedenze:

$$* \succ \cdot \succ +$$

Esempi

- $(a + (b(cd))) =$
- $(((aa) + b)c^*) =$
- $(((((ab) + ((ae)f)) + (df))((gh)f)^*)c) =$

Espressioni regolari

Proprietà delle operazioni $+$ e $\cdot$

Siano E, F e G espressioni regolari. Allora

- $((E + F) + G) = (E + (F + G))$.
- $((E \cdot F) \cdot G) = (E \cdot (F \cdot G))$.
- $E \cdot (F + G) = E \cdot F + E \cdot G$.

Per usare meno parentesi si possono considerare le seguenti precedenze:

$$* \succ \cdot \succ +$$

Esempi

- $(a + (b(cd))) = a + bcd$
- $(((aa) + b)c^*) = (aa + b)c^*$
- $(((((ab) + ((ae)f)) + (df))((gh)f)^*)c) = (ab + aef + df)(ghf)^*c$

Espressioni regolari

Proprietà delle operazioni $+$ e $\cdot$

Siano E, F e G espressioni regolari. Allora

- $((E + F) + G) = (E + (F + G))$.
- $((E \cdot F) \cdot G) = (E \cdot (F \cdot G))$.
- $E \cdot (F + G) = E \cdot F + E \cdot G$.

Per usare meno parentesi si possono considerare le seguenti precedenze:

$$* \succ \cdot \succ +$$

Esempi

- $(a + (b(cd))) = a + bcd$
- $(((aa) + b)c^*) = (aa + b)c^*$
- $(((((ab) + ((ae)f)) + (df))((gh)f)^*)c) = (ab + aef + df)(ghf)^*c$

Teorema: **Espressioni regolari**

Un linguaggio è regolare sse è generato da una espressione regolare.

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica per l'espressione regolare ab^*a.

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica per l'espressione regolare ab^*a.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{S, B\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to aBa$
2. $B \to bB \mid \epsilon$

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica per l'espressione regolare ab^*a.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{S, B\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to aBa$
2. $B \to bB \mid \epsilon$

Esempio

Scrivere una grammatica per l'espressione regolare $((a + b)(a + b))^* + a^*ab$.

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica per l'espressione regolare ab^*a.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{S, B\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to aBa$
2. $B \to bB \mid \epsilon$

Esempio

Scrivere una grammatica per l'espressione regolare $((a + b)(a + b))^* + a^*ab$.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{A, B, C, S\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to C \mid Aab$
2. $A \to aA \mid \epsilon$
3. $B \to aa \mid ab \mid ba \mid bb$
4. $C \to BC \mid \epsilon$

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica regolare (lineare destra) per l'espressione regolare ab^*a.

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica regolare (lineare destra) per l'espressione regolare ab^*a.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{S, B\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to aB$
2. $B \to bB \mid a$

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica regolare (lineare destra) per l'espressione regolare ab^*a.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{S, B\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to aB$
2. $B \to bB \mid a$

Esempio

Scrivere una grammatica regolare (lineare destra) per l'espressione regolare $((a + b)(a + b))^* + a^*ab$.

Da Espressioni regolari a Grammatiche

Esempio

Scrivere una grammatica regolare (lineare destra) per l'espressione regolare ab^*a.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{S, B\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to aB$
2. $B \to bB \mid a$

Esempio

Scrivere una grammatica regolare (lineare destra) per l'espressione regolare $((a + b)(a + b))^* + a^*ab$.
Soluzione: $\mathcal{G} = \langle \{a, b\}, \{E_1, E_2, E_3, S\}, S, P \rangle$, con le seguenti produzioni:

1. $S \to aE_1 \mid bE_1 \mid aE_2 \mid \epsilon$
2. $E_1 \to aE_3 \mid bE_3 \mid a \mid b$
3. $E_3 \to aE_1 \mid bE_1$
4. $E_2 \to aE_2 \mid b$

Espressioni regolari e programmazione: **Scanner**

Generare uno scanner: programma che riconosce elementi lessicali in un testo

Scrivere una espressione regolare per riconoscere **numeri decimali** delle seguenti tipologie:

$$+187.6372 \qquad -29.287 \qquad 120.34 \qquad 38$$

$$235. \qquad .0034 \qquad -.120 \qquad +.77$$

Espressioni regolari e programmazione: **Scanner**

Generare uno **scanner**: programma che riconosce elementi lessicali in un testo

Scrivere una espressione regolare per riconoscere **numeri decimali** delle seguenti tipologie:

$$+187.6372 \qquad -29.287 \qquad 120.34 \qquad 38$$

$$235. \qquad .0034 \qquad -.120 \qquad +.77$$

Soluzione:

$$(\epsilon \mid + \mid -)(C^+(\epsilon \mid .C^*) \mid .C^+), \text{ dove } C = (0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9)$$

Nel linguaggio di programmazione **Perl**, la precedente espressione regolare diventa:

```
(\-|\+)?([0-9]+(\.[0-9]*)?|\.[0-9]+)
```

Nota: abbiamo usato il simbolo "|" (come in Perl) al posto di "+" per comporre l'espressione regolare ed evitare ambiguità.

Grammatiche non contestuali: **Parse Tree**

Espressioni aritmetiche e albero di derivazione (parse tree)

Una grammatica per espressioni aritmetiche con operatori somma e prodotto:

- $S \to S + T \mid T$
- $T \to T * V \mid V$
- $V \to (S) \mid \mathit{float}$

La grammatica è $\mathcal{G} = \langle \{\mathit{float}, (,), +, *\}, \{S, T, V\}, S, P \rangle$.

Grammatiche non contestuali: **Parse Tree**

Espressioni aritmetiche e albero di derivazione (parse tree)

Una grammatica per espressioni aritmetiche con operatori somma e prodotto:

- $S \to S + T \mid T$
- $T \to T * V \mid V$
- $V \to (S) \mid float$

La grammatica è $\mathcal{G} = \langle \{float, (,), +, *\}, \{S, T, V\}, S, P \rangle$.

- Esempio: $(float + float) * float$

  $S \Rightarrow T \Rightarrow T * V$
  $\Rightarrow V * V \Rightarrow (S) * V$
  $\Rightarrow (S + T) * V$
  $\Rightarrow (T + T) * V$
  $\Rightarrow (V + T) * V$
  $\Rightarrow (V + V) * V \Rightarrow \ldots$
  $\Rightarrow (float + float) * float$

Grammatiche non contestuali: **Parse Tree**

Espressioni aritmetiche e albero di derivazione (parse tree)

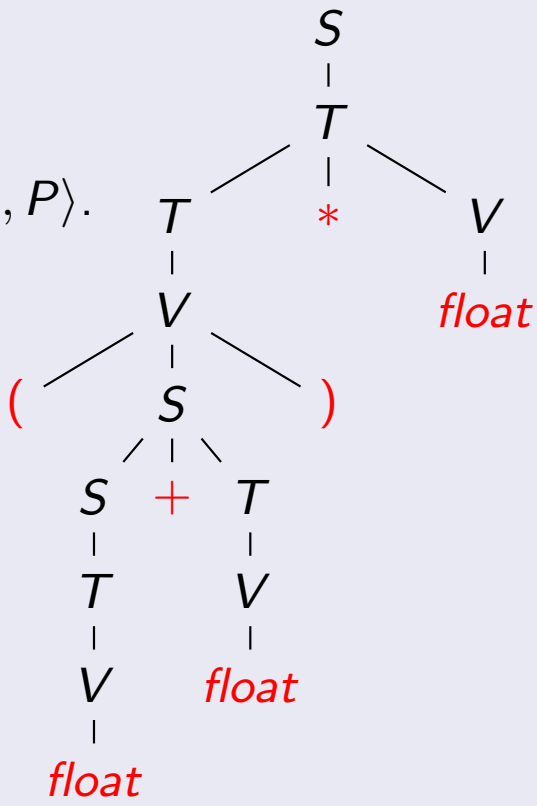
Una grammatica per espressioni aritmetiche con operatori somma e prodotto:

- $S \to S + T \mid T$
- $T \to T * V \mid V$
- $V \to (S) \mid float$

La grammatica è $\mathcal{G} = \langle \{float, (,), +, *\}, \{S, T, V\}, S, P \rangle$.

- Esempio: $(float + float) * float$

$$
\begin{aligned}
S &\Rightarrow T \Rightarrow T * V \\
&\Rightarrow V * V \Rightarrow (S) * V \\
&\Rightarrow (S + T) * V \\
&\Rightarrow (T + T) * V \\
&\Rightarrow (V + T) * V \\
&\Rightarrow (V + V) * V \Rightarrow \dots \\
&\Rightarrow (float + float) * float
\end{aligned}
$$



Grammatiche non contestuali: **Parse Tree**

Espressioni booleane e albero di derivazione (parse tree)

Una grammatica per generare espressioni booleane:

- $S \to (S \wedge S) \mid (S \vee S) \mid (S \to S) \mid (S \leftrightarrow S) \mid (\neg S) \mid A$
- $A \to a \mid b \mid \ldots \mid z$

La grammatica è $\mathcal{G} = \langle \{a, b, \ldots, z, \wedge, \vee, \to, \leftrightarrow, \neg, (,)\}, \{S, A\}, S, P \rangle$.

Grammatiche non contestuali: **Parse Tree**

Espressioni booleane e albero di derivazione (parse tree)

Una grammatica per generare espressioni booleane:

- $S \to (S \wedge S) \mid (S \vee S) \mid (S \to S) \mid (S \leftrightarrow S) \mid (\neg S) \mid A$
- $A \to a \mid b \mid \ldots \mid z$

La grammatica è $\mathcal{G} = \langle \{a, b, \ldots, z, \wedge, \vee, \to, \leftrightarrow, \neg, (,)\}, \{S, A\}, S, P \rangle$.

- Esempio: $((a \vee b) \to (\neg c))$

  $S \Rightarrow (S \to S) \Rightarrow ((S \vee S) \to S) \Rightarrow ((S \vee S) \to (\neg S)) \Rightarrow \ldots \Rightarrow$
  $((A \vee A) \to (\neg A)) \Rightarrow \ldots \Rightarrow ((a \vee b) \to (\neg c))$

Grammatiche non contestuali: **Parse Tree**

Espressioni booleane e albero di derivazione (parse tree)

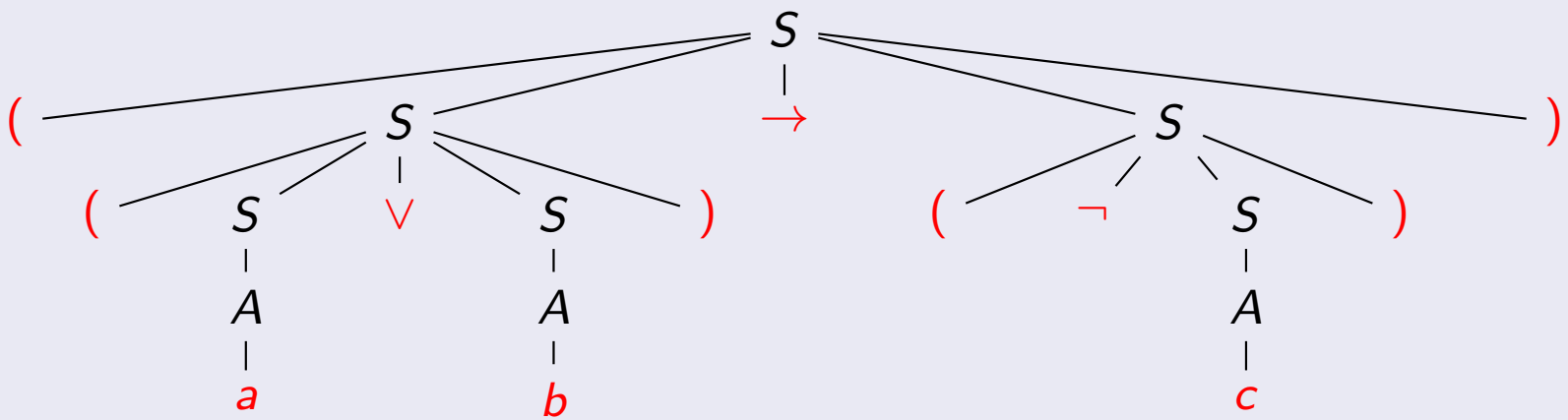
Una grammatica per generare espressioni booleane:

- $S \to (S \wedge S) \mid (S \vee S) \mid (S \to S) \mid (S \leftrightarrow S) \mid (\neg S) \mid A$
- $A \to a \mid b \mid \ldots \mid z$

La grammatica è $\mathcal{G} = \langle \{a, b, \ldots, z, \wedge, \vee, \to, \leftrightarrow, \neg, (,)\}, \{S, A\}, S, P \rangle$.

- Esempio: $((a \vee b) \to (\neg c))$

  $S \Rightarrow (S \to S) \Rightarrow ((S \vee S) \to S) \Rightarrow ((S \vee S) \to (\neg S)) \Rightarrow \ldots \Rightarrow$
  $((A \vee A) \to (\neg A)) \Rightarrow \ldots \Rightarrow ((a \vee b) \to (\neg c))$



Grammatiche non contestuali: **ambiguità**

Esempio di grammatica ambigua

Consideriamo i costrutti **if-then** e **if-then-else** e le seguenti produzioni:

- $S \rightarrow$ **if** C **then** S **else** $S \mid$ **if** C **then** $S \mid a \mid b$
- $C \rightarrow p \mid q$

S e C sono le istruzioni; a e b sono espressioni (aritmetiche o booleane); p e q sono espressioni booleane; e **if**, **then**, **else** sono simboli terminali.

Consideriamo la stringa: **if** q **then if** p **then** a **else** b

Grammatiche non contestuali: **ambiguità**

Esempio di grammatica ambigua

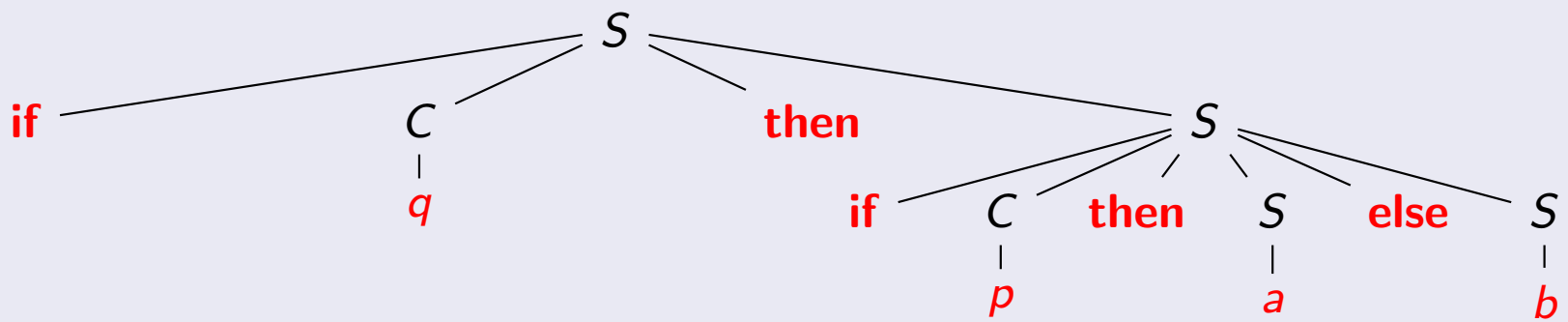
Consideriamo i costrutti **if-then** e **if-then-else** e le seguenti produzioni:

- $S \rightarrow$ **if** C **then** S **else** $S \mid$ **if** C **then** $S \mid a \mid b$
- $C \rightarrow p \mid q$

S e C sono le istruzioni; a e b sono espressioni (aritmetiche o booleane); p e q sono espressioni booleane; e **if**, **then**, **else** sono simboli terminali.

Consideriamo la stringa: **if** q **then if** p **then** a **else** b

(1) **if** q **then** (**if** p **then** a **else** b)



Grammatiche non contestuali: **ambiguità**

Esempio di grammatica ambigua

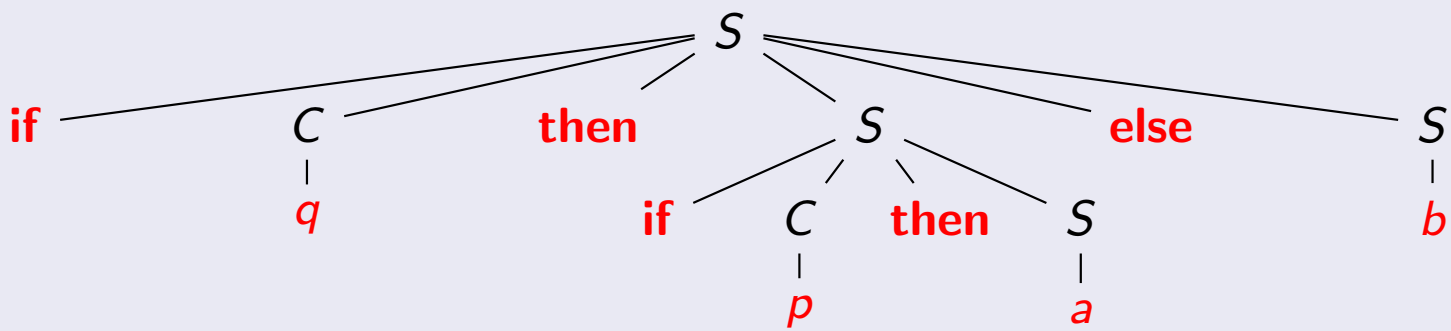
Consideriamo i costrutti **if-then** e **if-then-else** e le seguenti produzioni:

- $S \rightarrow$ **if** C **then** S **else** $S \mid$ **if** C **then** $S \mid a \mid b$
- $C \rightarrow p \mid q$

S e C sono le istruzioni; a e b sono espressioni (aritmetiche o booleane); p e q sono espressioni booleane; e **if**, **then**, **else** sono simboli terminali.

Consideriamo la stringa: **if** q **then if** p **then** a **else** b

(2) **if** q **then** (**if** p **then** a) **else** b



Grammatiche non contestuali: **ambiguità**

Esempio di grammatica ambigua

Consideriamo i costrutti **if-then** e **if-then-else** e le seguenti produzioni:

- $S \to \textbf{if } C \textbf{ then } S \textbf{ else } S \mid \textbf{if } C \textbf{ then } S \mid a \mid b$
- $C \to p \mid q$

S e C sono le istruzioni; a e b sono espressioni (aritmetiche o booleane); p e q sono espressioni booleane; e **if**, **then**, **else** sono simboli terminali.

Consideriamo la stringa: **if** q **then if** p **then** a **else** b

Definizione: **Grammatica ambigua**

Una grammatica non contestuale $\mathcal{G}$ è **ambigua** se qualche stringa $w \in \mathcal{L}(\mathcal{G})$ ha due alberi di derivazione distinti.

Grammatiche non contestuali: **ambiguità**

Esempio di grammatica ambigua

Consideriamo i costrutti **if-then** e **if-then-else** e le seguenti produzioni:

- $S \to$ **if** C **then** S **else** $S \mid$ **if** C **then** $S \mid a \mid b$
- $C \to p \mid q$

S e C sono le istruzioni; a e b sono espressioni (aritmetiche o booleane); p e q sono espressioni booleane; e **if**, **then**, **else** sono simboli terminali.

Consideriamo la stringa: **if** q **then if** p **then** a **else** b

Definizione: **Grammatica ambigua**

Una grammatica non contestuale $\mathcal{G}$ è **ambigua** se qualche stringa $w \in \mathcal{L}(\mathcal{G})$ ha due alberi di derivazione distinti.

Osservazione

Nei linguaggi di programmazione la sintassi deve essere definita attraverso una grammatica non ambigua.

Linguaggi formali e programmazione

Alcuni usi della teoria dei linguaggi formali in informatica:

- Specificare la sintassi dei linguaggi di programmazione.
- Descrivere la struttura dei dati in input di un programma.
- Costruire analizzatori lessicali (**scanner**) e analizzatori sintattici (**parser**) per verificare se un programma è sintatticamente corretto.
- Descrivere la struttura delle frasi del linguaggio naturale (NLP).