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

23 marzo 2022

Anno Accademico 2021/2022

Linguaggi naturali e linguaggi formali

- **Linguaggi naturali**: italiano, inglese, cinese, ecc.

Linguaggi naturali e linguaggi formali

- **Linguaggi naturali**: italiano, inglese, cinese, ecc.
- **Semiotica** (studio dei "segni" del linguaggio)
  - **Sintassi**: rapporto dei segni tra loro: parole (morfologia) e frasi
  - **Semantica**: rapporto dei segni col significato

Linguaggi naturali e linguaggi formali

- **Linguaggi naturali**: italiano, inglese, cinese, ecc.
- **Semiotica** (studio dei "segni" del linguaggio)
  - **Sintassi**: rapporto dei segni tra loro: parole (morfologia) e frasi
  - **Semantica**: rapporto dei segni col significato
  - **Pragmatica**: rapporto dei segni con la comunità dei parlanti (contesti psichici, ambientali, sociali;)

Linguaggi naturali e linguaggi formali

- **Linguaggi naturali**: italiano, inglese, cinese, ecc.
- **Semiotica** (studio dei "segni" del linguaggio)
  - **Sintassi**: rapporto dei segni tra loro: parole (morfologia) e frasi
  - **Semantica**: rapporto dei segni col significato
  - **Pragmatica**: rapporto dei segni con la comunità dei parlanti (contesti psichici, ambientali, sociali;)
- Ancora sulla semantica:
  - Problema: "capire" il linguaggio naturale
  - Elaborazione del Linguaggio Naturale (*Natural Language Processing*, NLP)

Linguaggi naturali e linguaggi formali

- **Linguaggi naturali**: italiano, inglese, cinese, ecc.
- **Semiotica** (studio dei "segni" del linguaggio)
  - **Sintassi**: rapporto dei segni tra loro: parole (morfologia) e frasi
  - **Semantica**: rapporto dei segni col significato
  - **Pragmatica**: rapporto dei segni con la comunità dei parlanti (contesti psichici, ambientali, sociali;)
- Ancora sulla semantica:
  - Problema: "capire" il linguaggio naturale
  - Elaborazione del Linguaggio Naturale (*Natural Language Processing*, NLP)
  - Ambiguità: "Una vecchia porta la sbarra"

Linguaggi naturali e linguaggi formali

- **Linguaggi naturali**: italiano, inglese, cinese, ecc.
- **Semiotica** (studio dei "segni" del linguaggio)
  - **Sintassi**: rapporto dei segni tra loro: parole (morfologia) e frasi
  - **Semantica**: rapporto dei segni col significato
  - **Pragmatica**: rapporto dei segni con la comunità dei parlanti (contesti psichici, ambientali, sociali;)
- Ancora sulla semantica:
  - Problema: "capire" il linguaggio naturale
  - Elaborazione del Linguaggio Naturale (*Natural Language Processing*, NLP)
  - Ambiguità: "Una vecchia porta la sbarra"
- La teoria dei **linguaggi formali** nasce attorno al 1950.

Linguaggi naturali e linguaggi formali

- **Linguaggi naturali**: italiano, inglese, cinese, ecc.
- **Semiotica** (studio dei "segni" del linguaggio)
  - **Sintassi**: rapporto dei segni tra loro: parole (morfologia) e frasi
  - **Semantica**: rapporto dei segni col significato
  - **Pragmatica**: rapporto dei segni con la comunità dei parlanti (contesti psichici, ambientali, sociali;)
- Ancora sulla semantica:
  - Problema: "capire" il linguaggio naturale
  - Elaborazione del Linguaggio Naturale (*Natural Language Processing*, NLP)
  - Ambiguità: "Una vecchia porta la sbarra"
- La teoria dei **linguaggi formali** nasce attorno al 1950.
- Il primo tentativo di costruire un linguaggio formale risale a **Gottlob Frege**:
  - *Begriffschrift. Un linguaggio in formule del pensiero puro, a imitazione di quello aritmetico* (1879)

Alfabeti

Esempi di alfabeti

- Alfabeto italiano:
  $\{a, b, c, d, e, f, g, h, i, l, m, n, o, p, q, r, s, t, u, v, z\}$
- Alfabeto decimale: $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- Alfabeto binario: $\{0, 1\}$
- Alfabeto di Python:
  $\{a, \ldots, z, A, \ldots, Z, +, -, *, /, =, :, (,), \ldots\}$

Alfabeti

Esempi di alfabeti

- Alfabeto italiano:
  $\{a, b, c, d, e, f, g, h, i, l, m, n, o, p, q, r, s, t, u, v, z\}$
- Alfabeto decimale: $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- Alfabeto binario: $\{0, 1\}$
- Alfabeto di Python:
  $\{a, \ldots, z, A, \ldots, Z, +, -, *, /, =, :, (,), \ldots\}$

Definizione: **Alfabeto**

Un alfabeto Σ è un insieme finito non vuoto di simboli.

Stringhe di un alfabeto

Esempi di stringhe

- Σ è l'alfabeto italiano: $s_1 = buongiorno$, $s_2 = aaabbcddd$, $s_3 = cadcqefvn$
- Σ è l'alfabeto decimale: $s_1 = 23902$, $s_2 = 00778$
- Σ è l'alfabeto binario: $s_1 = 1011$, $s_2 = 0000$

Stringhe di un alfabeto

Esempi di stringhe

- Σ è l'alfabeto italiano: $s_1 = buongiorno$, $s_2 = aaabbcddd$, $s_3 = cadcqefvn$
- Σ è l'alfabeto decimale: $s_1 = 23902$, $s_2 = 00778$
- Σ è l'alfabeto binario: $s_1 = 1011$, $s_2 = 0000$

Definizione: **Stringa**

Sia Σ un alfabeto. Una *stringa* o una *parola* s su Σ è una sequenza finita di simboli di Σ.

Stringhe di un alfabeto

Esempi di stringhe

- Σ è l'alfabeto italiano: $s_1 = buongiorno$, $s_2 = aaabbcddd$, $s_3 = cadcqefvn$
- Σ è l'alfabeto decimale: $s_1 = 23902$, $s_2 = 00778$
- Σ è l'alfabeto binario: $s_1 = 1011$, $s_2 = 0000$

Definizione: **Stringa**

Sia Σ un alfabeto. Una *stringa* o una *parola* s su Σ è una sequenza finita di simboli di Σ.

- La stringa senza simboli è denotata con $s = \epsilon$, detta *stringa vuota*.

Stringhe di un alfabeto

Esempi di stringhe

- Σ è l'alfabeto italiano: $s_1 = buongiorno$, $s_2 = aaabbcddd$, $s_3 = cadcqefvn$
- Σ è l'alfabeto decimale: $s_1 = 23902$, $s_2 = 00778$
- Σ è l'alfabeto binario: $s_1 = 1011$, $s_2 = 0000$

Definizione: **Stringa**

Sia Σ un alfabeto. Una *stringa* o una *parola* s su Σ è una sequenza finita di simboli di Σ.

- La stringa senza simboli è denotata con $s = \epsilon$, detta *stringa vuota*.
- La lunghezza di una stringa s è indicata con $|s|$ (la stringa vuota ha lunghezza 0).

Concatenazione

Definizione: **Concatenazione**

Date due stringhe $s_1 = x_1 x_2 \dots x_k$ e $s_2 = y_1 y_2 \dots y_h$ con $x_i, y_j \in \Sigma$.
La *concatenazione* di s_1 e s_2, scritta $s_1 s_2$ oppure $s_1 \cdot s_2$ è data da

$$s_1 s_2 = x_1 x_2 \dots x_k y_1 y_2 \dots y_h$$

Concatenazione

Definizione: **Concatenazione**

Date due stringhe $s_1 = x_1 x_2 \dots x_k$ e $s_2 = y_1 y_2 \dots y_h$ con $x_i, y_j \in \Sigma$.
La *concatenazione* di s_1 e s_2, scritta $s_1 s_2$ oppure $s_1 \cdot s_2$ è data da

$$s_1 s_2 = x_1 x_2 \dots x_k y_1 y_2 \dots y_h$$

Esempi di concatenazione

- Σ è l'alfabeto italiano: $s_1 = buon$, $s_2 = giorno$, allora
$$s_1 s_2 = buongiorno$$
- Σ è l'alfabeto decimale: $s_1 = 23902$, $s_2 = 08$
$$s_1 s_2 = 2390208$$

Concatenazione

Definizione: **Concatenazione**

Date due stringhe $s_1 = x_1 x_2 \dots x_k$ e $s_2 = y_1 y_2 \dots y_h$ con $x_i, y_j \in \Sigma$.
La *concatenazione* di s_1 e s_2, scritta $s_1 s_2$ oppure $s_1 \cdot s_2$ è data da

$$s_1 s_2 = x_1 x_2 \dots x_k y_1 y_2 \dots y_h$$

Esempi di concatenazione

- Σ è l'alfabeto italiano: $s_1 = buon$, $s_2 = giorno$, allora
$$s_1 s_2 = buongiorno$$
- Σ è l'alfabeto decimale: $s_1 = 23902$, $s_2 = 08$
$$s_1 s_2 = 2390208$$

- La lunghezza di una concatenazione è la somma delle lunghezze delle stringhe di partenza:
$$s = s_1 s_2 \Rightarrow |s| = |s_1| + |s_2|$$

Concatenazione

Proprietà dell'operatore di concatenazione

- Associatività: $s \cdot (s' \cdot s'') = (s \cdot s') \cdot s''$
- Elemento neutro: $s \cdot \epsilon = \epsilon \cdot s = s$

Concatenazione

Proprietà dell'operatore di concatenazione

- Associatività: $s \cdot (s' \cdot s'') = (s \cdot s') \cdot s''$
- Elemento neutro: $s \cdot \epsilon = \epsilon \cdot s = s$

Esempio (proprietà associativa)

$$((abb \cdot b) \cdot ba) = (abbb \cdot ba) = abbbba$$

$$(abb \cdot (b \cdot ba)) = (abb \cdot bba) = abbbba$$

Concatenazione

Proprietà dell'operatore di concatenazione

- Associatività: $s \cdot (s' \cdot s'') = (s \cdot s') \cdot s''$
- Elemento neutro: $s \cdot \epsilon = \epsilon \cdot s = s$

Esempio (proprietà associativa)

$$((abb \cdot b) \cdot ba) = (abbb \cdot ba) = abbbba$$

$$(abb \cdot (b \cdot ba)) = (abb \cdot bba) = abbbba$$

La concatenazione non è commutativa: $s \cdot s' \neq s' \cdot s$.

Concatenazione

Proprietà dell'operatore di concatenazione

- Associatività: $s \cdot (s' \cdot s'') = (s \cdot s') \cdot s''$
- Elemento neutro: $s \cdot \epsilon = \epsilon \cdot s = s$

Esempio (proprietà associativa)

$$((abb \cdot b) \cdot ba) = (abbb \cdot ba) = abbbba$$

$$(abb \cdot (b \cdot ba)) = (abb \cdot bba) = abbbba$$

La concatenazione non è commutativa: $s \cdot s' \neq s' \cdot s$.

Esempio (non commutatività)

$$abb \cdot bba = abbbba$$

$$\neq$$

$$bba \cdot abb = bbaabb$$

Concatenazione

Elevamento a potenza

- Per indicare la ripetizione di simboli in una stringa
- Per concatenare due o più stringhe uguali

Concatenazione

Elevamento a potenza

- Per indicare la ripetizione di simboli in una stringa
- Per concatenare due o più stringhe uguali

Esempio

$$ab^4a = abbbba$$

$$c^2d^4 = ccdddd$$

Se $s = ta$, allora

$$s^3 = tatata$$

$$c^2(ab)^3ac = ccababababac$$

Definizione induttiva di stringhe

Definizione (**Insieme di tutte le stringhe**)

Sia Σ un alfabeto e ϵ la stringa vuota. L'insieme di tutte le stringhe può essere definito induttivamente come segue:

- $\Sigma^0 = \{\epsilon\}$ Unica stringa di lunghezza 0

Definizione induttiva di stringhe

Definizione (**Insieme di tutte le stringhe**)

Sia Σ un alfabeto e ϵ la stringa vuota. L'insieme di tutte le stringhe può essere definito induttivamente come segue:

- $\Sigma^0 = \{\epsilon\}$ — Unica stringa di lunghezza 0
- $\Sigma^1 = \{a | a \in \Sigma\} = \Sigma$ — Stringhe di lunghezza 1

Definizione induttiva di stringhe

Definizione (**Insieme di tutte le stringhe**)

Sia Σ un alfabeto e ϵ la stringa vuota. L'insieme di tutte le stringhe può essere definito induttivamente come segue:

- $\Sigma^0 = \{\epsilon\}$ — Unica stringa di lunghezza 0
- $\Sigma^1 = \{a | a \in \Sigma\} = \Sigma$ — Stringhe di lunghezza 1
- $\Sigma^2 = \{s | s = a \cdot s', a \in \Sigma, s' \in \Sigma^1\}$ — Stringhe di lunghezza 2

Definizione induttiva di stringhe

Definizione (**Insieme di tutte le stringhe**)

Sia Σ un alfabeto e ϵ la stringa vuota. L'insieme di tutte le stringhe può essere definito induttivamente come segue:

- $\Sigma^0 = \{\epsilon\}$ — Unica stringa di lunghezza 0
- $\Sigma^1 = \{a | a \in \Sigma\} = \Sigma$ — Stringhe di lunghezza 1
- $\Sigma^2 = \{s | s = a \cdot s', a \in \Sigma, s' \in \Sigma^1\}$ — Stringhe di lunghezza 2
- $\Sigma^3 = \{s | s = a \cdot s', a \in \Sigma, s' \in \Sigma^2\}$ — Stringhe di lunghezza 3
- ...
- $\Sigma^n = \{s | s = a \cdot s', a \in \Sigma, s' \in \Sigma^{n-1}\}$ — Stringhe di lunghezza n
- ...

Definizione induttiva di stringhe

Definizione (**Insieme di tutte le stringhe**)

Sia Σ un alfabeto e ϵ la stringa vuota. L'insieme di tutte le stringhe può essere definito induttivamente come segue:

- $\Sigma^0 = \{\epsilon\}$ — Unica stringa di lunghezza 0
- $\Sigma^1 = \{a | a \in \Sigma\} = \Sigma$ — Stringhe di lunghezza 1
- $\Sigma^2 = \{s | s = a \cdot s', a \in \Sigma, s' \in \Sigma^1\}$ — Stringhe di lunghezza 2
- $\Sigma^3 = \{s | s = a \cdot s', a \in \Sigma, s' \in \Sigma^2\}$ — Stringhe di lunghezza 3
- ...
- $\Sigma^n = \{s | s = a \cdot s', a \in \Sigma, s' \in \Sigma^{n-1}\}$ — Stringhe di lunghezza n
- ...

Definiamo $\Sigma^* = \bigcup_{(n \geq 0)} \Sigma^n$. Σ^* è l'insieme delle stringhe di lunghezza finita qualsiasi ed è detto *universo linguistico* di Σ.

L'insieme di tutte le stringhe di Σ

Esempio (**Insieme di tutte le stringhe**)

L'insieme di tutte le stringhe di Σ

Esempio (**Insieme di tutte le stringhe**)

- Sia $\Sigma = \{(,)\}$, allora $\Sigma^* = \{\epsilon, (,), (), ((,)),)(, (((, ...\}$;

L'insieme di tutte le stringhe di Σ

Esempio (**Insieme di tutte le stringhe**)

- Sia $\Sigma = \{(,)\}$, allora $\Sigma^* = \{\epsilon, (,), (), ((,)),)(, (((, ...\}$;
- Sia $\Sigma = \{0, 1\}$, allora $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$;

L'insieme di tutte le stringhe di Σ

Esempio (**Insieme di tutte le stringhe**)

- Sia $\Sigma = \{(,)\}$, allora $\Sigma^* = \{\epsilon, (,), (), ((,)),)(, (((, ...\}$;
- Sia $\Sigma = \{0, 1\}$, allora $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$;
- Sia $\Sigma = \{a\}$, allora $\Sigma^* = \{\epsilon, a, aa, aaa, aaaa, aaaaa, aaaaaa, ...\}$.

L'insieme di tutte le stringhe di Σ

Esempio (**Insieme di tutte le stringhe**)

- Sia $\Sigma = \{(,)\}$, allora $\Sigma^* = \{\epsilon, (,), (), ((,)),)(, (((, ...\}$;
- Sia $\Sigma = \{0, 1\}$, allora $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$;
- Sia $\Sigma = \{a\}$, allora $\Sigma^* = \{\epsilon, a, aa, aaa, aaaa, aaaaa, aaaaaa, ...\}$.

Calcolo della lunghezza di una stringa

Definiamo induttivamente la funzione

$$f : \Sigma^* \to \mathbb{N}$$

1. $f(\epsilon) = 0$;
2. $f(a \cdot s) = 1 + f(s)$, $a \in \Sigma$ e $s \in \Sigma^*$.

L'insieme di tutte le stringhe di Σ

Esempio (**Insieme di tutte le stringhe**)

- Sia $\Sigma = \{(,)\}$, allora $\Sigma^* = \{\epsilon, (,), (), ((,)),)(, (((, ...\}$;
- Sia $\Sigma = \{0, 1\}$, allora $\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$;
- Sia $\Sigma = \{a\}$, allora $\Sigma^* = \{\epsilon, a, aa, aaa, aaaa, aaaaa, aaaaaa, ...\}$.

Calcolo della lunghezza di una stringa

Definiamo induttivamente la funzione

$$f : \Sigma^* \to \mathbb{N}$$

1. $f(\epsilon) = 0$;
2. $f(a \cdot s) = 1 + f(s)$, $a \in \Sigma$ e $s \in \Sigma^*$.

Ad esempio, $f(\text{“}casa\text{”}) = 1 + f(\text{“}asa\text{”}) = 1 + 1 + f(\text{“}sa\text{”}) = 1 + 1 + 1 + f(\text{“}a\text{”}) = 1 + 1 + 1 + 1 + f(\epsilon) = 1 + 1 + 1 + 1 + 0 = 4$.

Linguaggi

Definizione (Linguaggio)

Un linguaggio $\mathcal{L}$ su un alfabeto Σ è un sottoinsieme di Σ^*. Gli elementi di $\mathcal{L}$ sono dette le *stringhe ben formate* di $\mathcal{L}$.

Linguaggi

Definizione (**Linguaggio**)

Un linguaggio $\mathcal{L}$ su un alfabeto Σ è un sottoinsieme di Σ^*. Gli elementi di $\mathcal{L}$ sono dette le *stringhe ben formate* di $\mathcal{L}$.

Esempio (**Linguaggi**)

- Sia $\Sigma = \{(,)\}$. L'insieme delle stringhe di parentesi bilanciate è un linguaggio su Σ.
- Sia $\Sigma = \{0, 1\}$. L'insieme delle stringhe che contengono esattamente due 0 è un linguaggio ($\mathcal{L} = \{00, 100, 010, 001, 0011, 0110, 1001, 1100, ...\}$).
- Sia $\Sigma = \{a\}$. $\mathcal{L} = \{\epsilon, a, aa, aaa, aaaa\}$ è un linguaggio.
- Sia $\Sigma = \{a, b\}$. $\mathcal{L} = \{a^n b^n | n \geq 0\}$ è un linguaggio.
- Sono linguaggi anche Σ, Σ^*, $\emptyset$ (linguaggio vuoto), $\{\epsilon\}$.

Operazioni sui linguaggi

Siano $\mathcal{L}_1$ e $\mathcal{L}_2$ due linguaggi su Σ. Definiamo i seguenti operatori:

Definizione (Unione)

$$\mathcal{L}_1 \cup \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ oppure } s \in \mathcal{L}_2\}$$

Operazioni sui linguaggi

Siano $\mathcal{L}_1$ e $\mathcal{L}_2$ due linguaggi su Σ. Definiamo i seguenti operatori:

Definizione (Unione)

$$\mathcal{L}_1 \cup \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ oppure } s \in \mathcal{L}_2\}$$

Definizione (Concatenazione)

$$\mathcal{L}_1 \cdot \mathcal{L}_2 = \{s_1 s_2 | s_1 \in \mathcal{L}_1 \text{ e } s_2 \in \mathcal{L}_2\}$$

Operazioni sui linguaggi

Siano $\mathcal{L}_1$ e $\mathcal{L}_2$ due linguaggi su Σ. Definiamo i seguenti operatori:

Definizione (Unione)

$$\mathcal{L}_1 \cup \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ oppure } s \in \mathcal{L}_2\}$$

Definizione (Concatenazione)

$$\mathcal{L}_1 \cdot \mathcal{L}_2 = \{s_1 s_2 | s_1 \in \mathcal{L}_1 \text{ e } s_2 \in \mathcal{L}_2\}$$

Definizione (Intersezione)

$$\mathcal{L}_1 \cap \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ e } s \in \mathcal{L}_2\}$$

Operazioni sui linguaggi

Siano $\mathcal{L}_1$ e $\mathcal{L}_2$ due linguaggi su Σ. Definiamo i seguenti operatori:

Definizione (Unione)

$$\mathcal{L}_1 \cup \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ oppure } s \in \mathcal{L}_2\}$$

Definizione (Concatenazione)

$$\mathcal{L}_1 \cdot \mathcal{L}_2 = \{s_1 s_2 | s_1 \in \mathcal{L}_1 \text{ e } s_2 \in \mathcal{L}_2\}$$

Definizione (Intersezione)

$$\mathcal{L}_1 \cap \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ e } s \in \mathcal{L}_2\}$$

Definizione (Complementazione)

$$\overline{\mathcal{L}_1} = \{s \in \Sigma^* | s \notin \mathcal{L}_1\} = \Sigma^* \setminus \mathcal{L}_1$$

Operazioni sui linguaggi

Siano $\mathcal{L}_1$ e $\mathcal{L}_2$ due linguaggi su Σ. Definiamo i seguenti operatori:

Definizione (Unione)

$$\mathcal{L}_1 \cup \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ oppure } s \in \mathcal{L}_2\}$$

Definizione (Concatenazione)

$$\mathcal{L}_1 \cdot \mathcal{L}_2 = \{s_1 s_2 | s_1 \in \mathcal{L}_1 \text{ e } s_2 \in \mathcal{L}_2\}$$

Definizione (Intersezione)

$$\mathcal{L}_1 \cap \mathcal{L}_2 = \{s \in \Sigma^* | s \in \mathcal{L}_1 \text{ e } s \in \mathcal{L}_2\}$$

Definizione (Complementazione)

$$\overline{\mathcal{L}_1} = \{s \in \Sigma^* | s \notin \mathcal{L}_1\} = \Sigma^* \setminus \mathcal{L}_1$$

Definizione (Star o stella di Kleene)

$$\mathcal{L}^* = \cup_{n \geq 0} \mathcal{L}_n$$

dove $\mathcal{L}_n$ è definito come $\mathcal{L}_0 = \{\epsilon\}$ e $\mathcal{L}_{n+1} = \mathcal{L}_n \cdot \mathcal{L}$ per $n \geq 0$.

Operazioni sui linguaggi

Esempio (concatenazione)

Siano $\mathcal{L}_1 = \{a^n b^n | n \geq 1\}$ e $\mathcal{L}_2 = \{c^m | m \geq 1\}$ due linguaggi su $\Sigma = \{a, b, c\}$. Allora

- $ab, aabb \in \mathcal{L}_1$, mentre $aab, bbaa \notin \mathcal{L}_1$.
- $c, cccc \in \mathcal{L}_2$, mentre $\epsilon, acc \notin \mathcal{L}_1$.
- $\mathcal{L}_1 \cdot \mathcal{L}_2 =$
- $\mathcal{L}_1 \cdot \mathcal{L}_1 =$
- $\mathcal{L}_2 \cdot \mathcal{L}_2 =$
- $\mathcal{L}_2 \cdot \mathcal{L}_2 \cdot \mathcal{L}_2 =$

Operazioni sui linguaggi

Esempio (concatenazione)

Siano $\mathcal{L}_1 = \{a^n b^n | n \geq 1\}$ e $\mathcal{L}_2 = \{c^m | m \geq 1\}$ due linguaggi su $\Sigma = \{a, b, c\}$. Allora

- $ab, aabb \in \mathcal{L}_1$, mentre $aab, bbaa \notin \mathcal{L}_1$.
- $c, cccc \in \mathcal{L}_2$, mentre $\epsilon, acc \notin \mathcal{L}_1$.
- $\mathcal{L}_1 \cdot \mathcal{L}_2 = \{a^n b^n c^m | n, m \geq 1\}$.
- $\mathcal{L}_1 \cdot \mathcal{L}_1 = \{a^n b^n a^m b^m | n, m \geq 1\}$.
- $\mathcal{L}_2 \cdot \mathcal{L}_2 = \{c^n c^m | n, m \geq 1\} = \{c^n | n \geq 2\}$.
- $\mathcal{L}_2 \cdot \mathcal{L}_2 \cdot \mathcal{L}_2 = \{c^n | n \geq 3\}$.

Operazioni sui linguaggi

Esempi (Calcoliamo $\mathcal{L}^\star$)

- Sia $\mathcal{L} = \{ab, aab\}$.
- Sia $\mathcal{L} = \{a, b\}$.
- Sia $\mathcal{L} = \{aa, bb\}$.
- Sia $\mathcal{L} = \{a^n b^n | n \geq 1\}$.
- Sia $\mathcal{L} = \emptyset$.

Operazioni sui linguaggi

Esempi (Calcoliamo $\mathcal{L}^\star$)

- Sia $\mathcal{L} = \{ab, aab\}$.
  $\mathcal{L}^* = \{\epsilon, ab, aab, abab, abaab, aabab, aabaab, ...\}$.
- Sia $\mathcal{L} = \{a, b\}$. $\mathcal{L}^*$ contiene tutte le stringhe su a e b, inclusa ϵ.
- Sia $\mathcal{L} = \{aa, bb\}$. $\mathcal{L}^*$ contiene tutte le stringhe costituite da coppie di a e coppie di b, inclusa ϵ.
- Sia $\mathcal{L} = \{a^n b^n | n \geq 1\}$.
  $\mathcal{L}^* = \{a^{n_1} b^{n_1} \ldots a^{n_k} b^{n_k} | k \geq 0; n_1 \geq 1; \ldots; n_k \geq 1\}$.
- Sia $\mathcal{L} = \emptyset$. $\mathcal{L}^* = \{\epsilon\}$.

Definizione induttiva dei linguaggi

- Non tutti i linguaggi sono interessanti.
- Considereremo linguaggi con particolari strutture.

Definizione induttiva dei linguaggi

- Non tutti i linguaggi sono interessanti.
- Considereremo linguaggi con particolari strutture.
- Ad esempio:
  - *linguaggio della logica proposizionale*
  - *linguaggio delle espressioni aritmetiche*
  - *linguaggio dei programmi sintatticamente corretti in Python*.

Definizione induttiva dei linguaggi

- Non tutti i linguaggi sono interessanti.
- Considereremo linguaggi con particolari strutture.
- Ad esempio:
  - *linguaggio della logica proposizionale*
  - *linguaggio delle espressioni aritmetiche*
  - *linguaggio dei programmi sintatticamente corretti in Python*.

Esempio (Linguaggio delle parentesi bilanciate)

Definiamo induttivamente il linguaggio delle parentesi bilanciate:

Definizione induttiva dei linguaggi

- Non tutti i linguaggi sono interessanti.
- Considereremo linguaggi con particolari strutture.
- Ad esempio:
  - *linguaggio della logica proposizionale*
  - *linguaggio delle espressioni aritmetiche*
  - *linguaggio dei programmi sintatticamente corretti in Python*.

Esempio (Linguaggio delle parentesi bilanciate)

Definiamo induttivamente il linguaggio delle parentesi bilanciate:

- **Passo base**
  - La stringa () è ben formata.
- **Passo induttivo**
  - Se s è ben formata, allora (s) è ben formata.
  - Se s e s' sono ben formate, allora ss' è ben formata.

Osservazioni sui linguaggi

- Principali approcci *computazionali* per **definire un linguaggio**:
  - **Approccio generativo**: spiegando come si costruisce una stringa del linguaggio
    - definizione algebrica delle stringhe (es. $\{a^n b^n | n > 0\}$)
    - procedura induttiva (**grammatiche generative**) che definisce le regole strutturali da soddisfare (es. linguaggio delle parentesi bilanciate)

Osservazioni sui linguaggi

- Principali approcci *computazionali* per **definire un linguaggio**:
  - **Approccio generativo**: spiegando come si costruisce una stringa del linguaggio
    - definizione algebrica delle stringhe (es. $\{a^n b^n | n > 0\}$)
    - procedura induttiva (**grammatiche generative**) che definisce le regole strutturali da soddisfare (es. linguaggio delle parentesi bilanciate)
  - **Approccio riconoscitivo**: fornendo una **macchina** o **automa** (algoritmo) che ricevendo una stringa di input dice se appartiene o meno al linguaggio

Osservazioni sui linguaggi

- Principali approcci *computazionali* per **definire un linguaggio**:
  - **Approccio generativo**: spiegando come si costruisce una stringa del linguaggio
    - definizione algebrica delle stringhe (es. $\{a^n b^n | n > 0\}$)
    - procedura induttiva (**grammatiche generative**) che definisce le regole strutturali da soddisfare (es. linguaggio delle parentesi bilanciate)
  - **Approccio riconoscitivo**: fornendo una **macchina** o **automa** (algoritmo) che ricevendo una stringa di input dice se appartiene o meno al linguaggio
- Non tutti i linguaggi possono essere riconosciuti mediante algoritmi o definiti in modo generativo:

Osservazioni sui linguaggi

- Principali approcci *computazionali* per **definire un linguaggio**:
  - **Approccio generativo**: spiegando come si costruisce una stringa del linguaggio
    - definizione algebrica delle stringhe (es. $\{a^n b^n | n > 0\}$)
    - procedura induttiva (**grammatiche generative**) che definisce le regole strutturali da soddisfare (es. linguaggio delle parentesi bilanciate)
  - **Approccio riconoscitivo**: fornendo una **macchina** o **automa** (algoritmo) che ricevendo una stringa di input dice se appartiene o meno al linguaggio
- Non tutti i linguaggi possono essere riconosciuti mediante algoritmi o definiti in modo generativo:
  - I possibili **algoritmi** (stringhe) sono una **infinità numerabile**.

Osservazioni sui linguaggi

- Principali approcci *computazionali* per **definire un linguaggio**:
  - **Approccio generativo**: spiegando come si costruisce una stringa del linguaggio
    - definizione algebrica delle stringhe (es. $\{a^n b^n | n > 0\}$)
    - procedura induttiva (**grammatiche generative**) che definisce le regole strutturali da soddisfare (es. linguaggio delle parentesi bilanciate)
  - **Approccio riconoscitivo**: fornendo una **macchina** o **automa** (algoritmo) che ricevendo una stringa di input dice se appartiene o meno al linguaggio
- Non tutti i linguaggi possono essere riconosciuti mediante algoritmi o definiti in modo generativo:
  - I possibili **algoritmi** (stringhe) sono una **infinità numerabile**.
  - I possibili **linguaggi** $\mathcal{L} \subseteq \Sigma^*$ sono un **infinità non numerabile**.

Osservazioni sui linguaggi

- Principali approcci *computazionali* per **definire un linguaggio**:
  - **Approccio generativo**: spiegando come si costruisce una stringa del linguaggio
    - definizione algebrica delle stringhe (es. $\{a^n b^n | n > 0\}$)
    - procedura induttiva (**grammatiche generative**) che definisce le regole strutturali da soddisfare (es. linguaggio delle parentesi bilanciate)
  - **Approccio riconoscitivo**: fornendo una **macchina** o **automa** (algoritmo) che ricevendo una stringa di input dice se appartiene o meno al linguaggio
- Non tutti i linguaggi possono essere riconosciuti mediante algoritmi o definiti in modo generativo:
  - I possibili **algoritmi** (stringhe) sono una **infinità numerabile**.
  - I possibili **linguaggi** $\mathcal{L} \subseteq \Sigma^*$ sono un **infinità non numerabile**.
- Dati in input un linguaggio $\mathcal{L}$ ed una stringa s, possiamo scrivere un algoritmo per decidere se $s \in \mathcal{L}$? NO!