STRUMENTI FORMALI PER LA BIOINFORMATICA

Stringhe e Linguaggi

Alfabeto

Un **alfabeto** è un insieme finito di elementi (chiamati **lettere** o **simboli** o **caratteri (terminali)**)

Un **alfabeto** è un insieme finito di elementi (chiamati **lettere** o **simboli** o **caratteri (terminali)**)

Esempio: L'alfabeto delle lettere romane minuscole è

$$\Sigma = \{a, b, c, ..., z\}$$

Un **alfabeto** è un insieme finito di elementi (chiamati **lettere** o **simboli** o **caratteri (terminali)**)

Esempio: L'alfabeto delle lettere romane minuscole è

$$\Sigma = \{a, b, c, ..., z\}$$

Esempio: L'alfabeto delle cifre arabe è

$$\Sigma = \{0, 1, \ldots, 9\}$$

Un **alfabeto** è un insieme finito di elementi (chiamati **lettere** o **simboli** o **caratteri (terminali)**)

Esempio: L'alfabeto delle lettere romane minuscole è

$$\Sigma = \{a, b, c, ..., z\}$$

Esempio: L'alfabeto delle cifre arabe è

$$\Sigma = \{0, 1, \ldots, 9\}$$

Esempio: L'alfabeto binario è

$$\Sigma = \{0, 1\}$$

Sia $\Sigma = \{a_1, \dots, a_k\}$ un alfabeto di k simboli, Sia $\Sigma = \{a_1, \ldots, a_k\}$ un alfabeto di k simboli,

ovvero la **cardinalità** di Σ è k.
In simboli: $|\Sigma| = k$.

Sia $\Sigma = \{a_1, \ldots, a_k\}$ un alfabeto di k simboli,
ovvero la **cardinalità** di Σ è k.
In simboli: $|\Sigma| = k$.

Una **stringa** (o **parola**) su un alfabeto è una sequenza (finita) di simboli dell'alfabeto.
Ossia è un insieme ordinato con eventuali ripetizioni di caratteri.
La **stringa vuota** ϵ è la stringa che non contiene nessun simbolo.

- **Esempio:** Sia $\Sigma = \{a, b\}$.
  $aaba$, aaa, $abaa$, b sono stringhe.

- **Esempio:** Sia $\Sigma = \{a, b\}$.
  $aaba, aaa, abaa, b$ sono stringhe.

- **Esempio:** 0131 è una stringa sull'alfabeto $\Sigma = \{0, 1, 2, ..., 9\}$.

- **Esempio:** Sia $\Sigma = \{a, b\}$.
  $aaba$, aaa, $abaa$, b sono stringhe.

- **Esempio:** 0131 è una stringa sull'alfabeto $\Sigma = \{0, 1, 2, ..., 9\}$.

- **Esempio:** 0101 è una stringa sull'alfabeto $\Sigma = \{0, 1\}$.

Definizione

L'insieme Σ^* delle stringhe sull'alfabeto Σ è definito ricorsivamente come segue:

Definizione

L'insieme Σ^* delle stringhe sull'alfabeto Σ è definito ricorsivamente come segue:

PASSO BASE: $\epsilon \in \Sigma^*$ *(dove ϵ è la stringa vuota).*

Definizione

L'insieme Σ^* delle stringhe sull'alfabeto Σ è definito ricorsivamente come segue:

PASSO BASE: $\epsilon \in \Sigma^*$ *(dove ϵ è la stringa vuota).*

PASSO RICORSIVO: *Se $w \in \Sigma^*$ e $x \in \Sigma$, allora $wx \in \Sigma^*$.*

Definizione

L'insieme Σ^* delle stringhe sull'alfabeto Σ è definito ricorsivamente come segue:

PASSO BASE: $\epsilon \in \Sigma^*$ *(dove ϵ è la stringa vuota).*

PASSO RICORSIVO: *Se $w \in \Sigma^*$ e $x \in \Sigma$, allora $wx \in \Sigma^*$.*

- **Nota.** *Se nel passo ricorsivo $w = \epsilon$, porremo $\epsilon x = x$.*

Definizione

Un linguaggio (formale) è un insieme di stringhe su un alfabeto.

Definizione

Un linguaggio (formale) è un insieme di stringhe su un alfabeto.

Esempio: Linguaggi per computer, quali C, C^{++} o Java, sono linguaggi formali con alfabeto

$$\{a, b, ..., z, A, B, ..., Z, 0, 1, 2, ..., 9, >, <, =, +, -, *, /, (,), \cdots\}.$$

Le regole della sintassi definiscono le regole del linguaggio. Ogni programma accettabile (cioè scritto seguendo le regole della sintassi) è una stringa del linguaggio.
Anche l'insieme dei nomi validi di variabili in C (o in C^{++} o Java) è un linguaggio formale.

Esempio: Sia $\Sigma = \{a, b\}$.

$$
\begin{aligned}
L_1 &= \{aa, aaa\} \\
L_2 &= \{aba, aab\} \\
L_3 &= \{w \in \Sigma^* \mid \text{ numero di occorrenze di } a \text{ in } w \\
&\quad = \text{ numero di occorrenze di } b \text{ in } w\}
\end{aligned}
$$

La **cardinalità** di un linguaggio (finito) è il numero delle sue stringhe.

La **cardinalità** di un linguaggio (finito) è il numero delle sue stringhe.

Esempio:

$$|L_2| = |\{aba, aab\}| = 2$$

La **cardinalità** di un linguaggio (finito) è il numero delle sue stringhe.

Esempio:

$$|L_2| = |\{aba, aab\}| = 2$$

Un linguaggio **finito** è un linguaggio che ha un numero finito di stringhe.
Un linguaggio **infinito** è un linguaggio non finito.

La **cardinalità** di un linguaggio (finito) è il numero delle sue stringhe.

Esempio:

$$|L_2| = |\{aba, aab\}| = 2$$

Un linguaggio **finito** è un linguaggio che ha un numero finito di stringhe.
Un linguaggio **infinito** è un linguaggio non finito.

Un esempio di linguaggio finito: il linguaggio vuoto $\emptyset$.
$|\emptyset| = 0$.

- Nota: non solo linguaggi finiti.

  Infatti i linguaggi finiti non sono di solito interessanti.

  Tutti i nostri alfabeti sono finiti, ma la maggior parte dei linguaggi che incontreremo sono infiniti.

Il **numero di occorrenze di un carattere** a in una stringa x viene indicato da $|x|_a$.

Il **numero di occorrenze di un carattere** a in una stringa x viene indicato da $|x|_a$.

Esempio:

$$|aab|_a = 2, \quad |baa|_a = 2, \quad |baa|_b = 1, \quad |baa|_c = 0$$

Il **numero di occorrenze di un carattere** a in una stringa x viene indicato da $|x|_a$.

Esempio:

$$|aab|_a = 2, \quad |baa|_a = 2, \quad |baa|_b = 1, \quad |baa|_c = 0$$

La **lunghezza** di una stringa x è il numero di simboli in x.

Il **numero di occorrenze di un carattere** a in una stringa x viene indicato da $|x|_a$.

Esempio:

$$|aab|_a = 2, \quad |baa|_a = 2, \quad |baa|_b = 1, \quad |baa|_c = 0$$

La **lunghezza** di una stringa x è il numero di simboli in x.

La lunghezza di x è denotata con $|x|$.

Il **numero di occorrenze di un carattere** a in una stringa x viene indicato da $|x|_a$.

Esempio:

$$|aab|_a = 2, \quad |baa|_a = 2, \quad |baa|_b = 1, \quad |baa|_c = 0$$

La **lunghezza** di una stringa x è il numero di simboli in x.

La lunghezza di x è denotata con $|x|$.

Esempio: $|ab| = 2$, $|abaa| = 4$.

Due stringhe

$$x = a_1 a_2 \cdots a_h, \quad y = b_1 b_2 \cdots b_k,$$

con $a_i, b_j \in \Sigma$, $1 \leq i \leq h$, $1 \leq j \leq k$, si dicono **eguali** se

Due stringhe

$$x = a_1 a_2 \cdots a_h, \quad y = b_1 b_2 \cdots b_k,$$

con $a_i, b_j \in \Sigma$, $1 \leq i \leq h$, $1 \leq j \leq k$, si dicono **eguali** se $h = k$ e $a_i = b_i$, per $i = 1, \ldots, h$.

Due stringhe

$$x = a_1 a_2 \cdots a_h, \quad y = b_1 b_2 \cdots b_k,$$

con $a_i, b_j \in \Sigma$, $1 \leq i \leq h$, $1 \leq j \leq k$, si dicono **eguali** se $h = k$ e $a_i = b_i$, per $i = 1, \ldots, h$.

In due stringhe uguali i caratteri letti ordinatamente da sinistra a destra coincidono.

Due stringhe

$$x = a_1 a_2 \cdots a_h, \quad y = b_1 b_2 \cdots b_k,$$

con $a_i, b_j \in \Sigma$, $1 \leq i \leq h$, $1 \leq j \leq k$, si dicono **eguali** se $h = k$ e $a_i = b_i$, per $i = 1, \ldots, h$.

In due stringhe uguali i caratteri letti ordinatamente da sinistra a destra coincidono.

Esempio: $aba \neq baa$, $baa \neq ba$.

Date le stringhe

$$x = a_1 a_2 \cdots a_h, \quad y = b_1 b_2 \cdots b_k,$$

con $a_i, b_j \in \Sigma$, $1 \leq i \leq h$, $1 \leq j \leq k$, la **concatenazione** (di x e y) è definita da

$$x \cdot y = a_1 a_2 \cdots a_h b_1 b_2 \cdots b_k$$

Date le stringhe

$$x = a_1 a_2 \cdots a_h, \quad y = b_1 b_2 \cdots b_k,$$

con $a_i, b_j \in \Sigma$, $1 \leq i \leq h$, $1 \leq j \leq k$, la **concatenazione** (di x e y) è definita da

$$x \cdot y = a_1 a_2 \cdots a_h b_1 b_2 \cdots b_k$$

La concatenazione di due stringhe x e y è spesso denotata xy (invece che $x \cdot y$).

Date le stringhe

$$x = a_1 a_2 \cdots a_h, \quad y = b_1 b_2 \cdots b_k,$$

con $a_i, b_j \in \Sigma$, $1 \leq i \leq h$, $1 \leq j \leq k$, la **concatenazione** (di x e y) è definita da

$$x \cdot y = a_1 a_2 \cdots a_h b_1 b_2 \cdots b_k$$

La concatenazione di due stringhe x e y è spesso denotata xy (invece che $x \cdot y$).

- **Esempio:** $x = vice$, $y = capo$, $z = stazione$ $xy = vicecapo$,

  $yx = capovice \neq xy$

$$(xy)z = vicecapostazione = x(yz)$$

La concatenazione **non è commutativa**, in generale $xy \neq yx$.

La concatenazione **non è commutativa**, in generale $xy \neq yx$.

La concatenazione **è associativa**:

$$(xy)z = x(yz)$$

(possiamo scrivere senza parentesi la concatenazione di tre o più stringhe).

La concatenazione **non è commutativa**, in generale $xy \neq yx$.

La concatenazione **è associativa**:

$$(xy)z = x(yz)$$

(possiamo scrivere senza parentesi la concatenazione di tre o più stringhe).

$$|xy| = |x| + |y|$$

Stringa vuota

La **stringa vuota** ϵ è la stringa che non contiene nessun simbolo.

La **stringa vuota** ϵ è la stringa che non contiene nessun simbolo.

Proprietà della stringa vuota:

$$x\epsilon = \epsilon x = x$$

$$|\epsilon| = 0$$

La **stringa vuota** ϵ è la stringa che non contiene nessun simbolo.

Proprietà della stringa vuota:

$$x\epsilon = \epsilon x = x$$

$$|\epsilon| = 0$$

Nota :

$$\emptyset \neq \epsilon, \quad \emptyset \neq \{\epsilon\}$$

$\emptyset$ è un sottoinsieme di Σ^*, $\epsilon \in \Sigma^*$;
$|\emptyset| = 0 \neq 1 = |\{\epsilon\}|$.

Def. Data una stringa x, una *sottostringa* di x è una qualsiasi sequenza di simboli consecutivi della stringa x. Un *prefisso* di x è una qualsiasi sequenza di simboli consecutivi iniziali della stringa x. Un *suffisso* di x è una qualsiasi sequenza di simboli consecutivi terminali della stringa x.

Def. Data una stringa x, una *sottostringa* di x è una qualsiasi sequenza di simboli consecutivi della stringa x. Un *prefisso* di x è una qualsiasi sequenza di simboli consecutivi iniziali della stringa x. Un *suffisso* di x è una qualsiasi sequenza di simboli consecutivi terminali della stringa x.

Se $x = uyv$ è la concatenazione di stringhe u, y, v (eventualmente vuote) allora:

Def. Data una stringa x, una *sottostringa* di x è una qualsiasi sequenza di simboli consecutivi della stringa x. Un *prefisso* di x è una qualsiasi sequenza di simboli consecutivi iniziali della stringa x. Un *suffisso* di x è una qualsiasi sequenza di simboli consecutivi terminali della stringa x.

Se $x = uyv$ è la concatenazione di stringhe u, y, v (eventualmente vuote) allora:

- y è una **sottostringa** di x,

Def. Data una stringa x, una *sottostringa* di x è una qualsiasi sequenza di simboli consecutivi della stringa x. Un *prefisso* di x è una qualsiasi sequenza di simboli consecutivi iniziali della stringa x. Un *suffisso* di x è una qualsiasi sequenza di simboli consecutivi terminali della stringa x.

Se $x = uyv$ è la concatenazione di stringhe u, y, v (eventualmente vuote) allora:

- y è una **sottostringa** di x,
- u è un **prefisso** di x,

Def. Data una stringa x, una *sottostringa* di x è una qualsiasi sequenza di simboli consecutivi della stringa x. Un *prefisso* di x è una qualsiasi sequenza di simboli consecutivi iniziali della stringa x. Un *suffisso* di x è una qualsiasi sequenza di simboli consecutivi terminali della stringa x.

Se $x = uyv$ è la concatenazione di stringhe u, y, v (eventualmente vuote) allora:

- y è una **sottostringa** di x,
- u è un **prefisso** di x,
- v è un **suffisso** di x

Def. Data una stringa x, una *sottostringa* di x è una qualsiasi sequenza di simboli consecutivi della stringa x. Un *prefisso* di x è una qualsiasi sequenza di simboli consecutivi iniziali della stringa x. Un *suffisso* di x è una qualsiasi sequenza di simboli consecutivi terminali della stringa x.

Se $x = uyv$ è la concatenazione di stringhe u, y, v (eventualmente vuote) allora:

- y è una **sottostringa** di x,
- u è un **prefisso** di x,
- v è un **suffisso** di x

Una sottostringa (prefisso, suffisso) di x è **propria** se non coincide con x.

Esempio: La stringa 472 ha

Esempio: La stringa 472 ha

- prefissi: $\epsilon, 4, 47, 472,$

Esempio: La stringa 472 ha

- prefissi: $\epsilon, 4, 47, 472,$
- suffissi: $\epsilon, 2, 72, 472,$

Esempio: La stringa 472 ha

- prefissi: $\epsilon, 4, 47, 472,$
- suffissi: $\epsilon, 2, 72, 472,$
- sottostringhe: $\epsilon, 4, 7, 2, 47, 72, 472$

Esempio: La stringa 472 ha

- prefissi: $\epsilon, 4, 47, 472,$
- suffissi: $\epsilon, 2, 72, 472,$
- sottostringhe: $\epsilon, 4, 7, 2, 47, 72, 472$
- La stringa 42 non è sottostringa di 472.

Inversa o riflessione di una stringa

Definizione

L'inversa (o reverse o riflessione) $\mathbf{w}^R$ *di una stringa* w *è la stringa ottenuta scrivendo i caratteri di* w *da destra verso sinistra.*

Definizione

L'inversa (o reverse o riflessione) $\mathbf{w}^R$ *di una stringa* w *è la stringa ottenuta scrivendo i caratteri di* w *da destra verso sinistra.*

$\epsilon^R = \epsilon$ e se $w = a_1 \cdots a_n$, con a_j lettere, allora

$$\mathbf{w}^R = a_n a_{n-1} \cdots a_1.$$

Inversa o riflessione di una stringa

Definizione

L'inversa (o reverse o riflessione) $\mathbf{w}^R$ *di una stringa* w *è la stringa ottenuta scrivendo i caratteri di* w *da destra verso sinistra.*

$\epsilon^R = \epsilon$ e se $w = a_1 \cdots a_n$, con a_j lettere, allora

$$\mathbf{w}^R = a_n a_{n-1} \cdots a_1.$$

Esempio: $x = roma$, $x^R = amor$.

Inversa o riflessione di una stringa

Definizione

L'inversa (o reverse o riflessione) $\mathbf{w}^R$ *di una stringa* w *è la stringa ottenuta scrivendo i caratteri di* w *da destra verso sinistra.*

$\epsilon^R = \epsilon$ e se $w = a_1 \cdots a_n$, con a_j lettere, allora

$$\mathbf{w}^R = a_n a_{n-1} \cdots a_1.$$

Esempio: $x = roma$, $x^R = amor$.

Proprietà:

$$(x^R)^R = x, \quad (xy)^R = y^R x^R$$

Definizione ricorsiva dell'inversa di una stringa

PASSO BASE: $\epsilon^R = \epsilon$.

Definizione ricorsiva dell'inversa di una stringa

PASSO BASE: $\epsilon^R = \epsilon$.

PASSO RICORSIVO: Per ogni $x \in \Sigma^*$ e $\sigma \in \Sigma$, $(x\sigma)^R = \sigma x^R$.

Sia $m \geq 1$ un intero non negativo. La *potenza m-esima* di una stringa x è la concatenazione di x con sé stessa $m - 1$ volte. Per convenzione la potenza 0 di una stringa è la stringa vuota.

Sia $m \geq 1$ un intero non negativo. La *potenza m-esima* di una stringa x è la concatenazione di x con sé stessa $m-1$ volte. Per convenzione la potenza 0 di una stringa è la stringa vuota.

Definizione

Sia x una stringa. Poniamo
PASSO BASE: $x^0 = \epsilon$
PASSO RICORSIVO: $x^m = x^{m-1}x$, *per* $m > 0$.

Esempi:

Esempi:

$$
\begin{aligned}
x &= ab \\
x^0 &= \epsilon \\
x^1 &= x = ab \\
x^2 &= (ab)^2 = abab \\
y &= a^2 = aa \\
y^3 &= a^2 a^2 a^2 = a^6 \\
\epsilon^0 &= \epsilon \\
\epsilon^2 &= \epsilon
\end{aligned}
$$

Nota. È necessario racchiudere tra parentesi la stringa da elevare alla potenza se ha lunghezza maggiore di uno.

Nota. È necessario racchiudere tra parentesi la stringa da elevare alla potenza se ha lunghezza maggiore di uno.

$$(ab)^2 = abab \neq abb = ab^2$$

Nota. È necessario racchiudere tra parentesi la stringa da elevare alla potenza se ha lunghezza maggiore di uno.

$$(ab)^2 = abab \neq abb = ab^2$$

L'elevamento a potenza ha *precedenza* rispetto alla concatenazione.

Nota. È necessario racchiudere tra parentesi la stringa da elevare alla potenza se ha lunghezza maggiore di uno.

$$(ab)^2 = abab \neq abb = ab^2$$

L'elevamento a potenza ha *precedenza* rispetto alla concatenazione.

Anche la riflessione ha *precedenza* rispetto alla concatenazione.

Nota. È necessario racchiudere tra parentesi la stringa da elevare alla potenza se ha lunghezza maggiore di uno.

$$(ab)^2 = abab \neq abb = ab^2$$

L'elevamento a potenza ha *precedenza* rispetto alla concatenazione.

Anche la riflessione ha *precedenza* rispetto alla concatenazione.

$b^R = b$, quindi $ab^R = ab$.

$$(ab)^R = ba \neq ab^R = ab$$

Data un'operazione su una stringa, essa si può estendere a tutte le stringhe di un linguaggio.

Data un'operazione su una stringa, essa si può estendere a tutte le stringhe di un linguaggio.

Otteniamo così alcune operazioni sui linguaggi.

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \wedge y \in L\}$$

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \land y \in L\}$$

- **Esempio:**

$$L_1 = \{aa, aaa\}$$

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \wedge y \in L\}$$

- **Esempio:**

$$L_1 = \{aa, aaa\}$$

$$L_1 = L_1^R$$

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \land y \in L\}$$

- **Esempio:**

$$L_1 = \{aa, aaa\}$$

$$L_1 = L_1^R$$

$$L_2 = \{aba, aab\}$$

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \land y \in L\}$$

- **Esempio:**

$$L_1 = \{aa, aaa\}$$

$$L_1 = L_1^R$$

$$L_2 = \{aba, aab\}$$

$$L_2^R = \{aba, baa\}$$

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \wedge y \in L\}$$

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \wedge y \in L\}$$

- **Esempio:**

$$L_3 = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$$

La riflessione di un linguaggio L:

$$L^R = \{x \mid x = y^R \wedge y \in L\}$$

- **Esempio:**

$$L_3 = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$$

$$L_3 \quad = \quad L_3^R$$

L'insieme dei prefissi propri delle stringhe di un linguaggio L:

$$\begin{aligned} \text{Prefissi}(L) &= \{y \mid x = yz \wedge x \in L \wedge z \neq \epsilon\} \\ &= \{y \mid \exists x \in L \text{ tale che } x = yz \text{ con } z \neq \epsilon\} \\ &= \{y \mid y \text{ è prefisso proprio di una stringa in } L\} \end{aligned}$$

Esempio:

L'insieme dei prefissi propri delle stringhe di un linguaggio L:

$$\begin{aligned}
\text{Prefissi}(L) &= \{y \mid x = yz \wedge x \in L \wedge z \neq \epsilon\} \\
&= \{y \mid \exists x \in L \text{ tale che } x = yz \text{ con } z \neq \epsilon\} \\
&= \{y \mid y \text{ è prefisso proprio di una stringa in } L\}
\end{aligned}$$

Esempio:

- $L_1 = \{aa, aaa\}$

L'insieme dei prefissi propri delle stringhe di un linguaggio L:

$$\begin{aligned}\text{Prefissi}(L) &= \{y \mid x = yz \land x \in L \land z \neq \epsilon\} \\ &= \{y \mid \exists x \in L \text{ tale che } x = yz \text{ con } z \neq \epsilon\} \\ &= \{y \mid y \text{ è prefisso proprio di una stringa in } L\}\end{aligned}$$

Esempio:

- $L_1 = \{aa, aaa\}$
  Prefissi$(L_1) = \{\epsilon, a, aa\}$

L'insieme dei prefissi propri delle stringhe di un linguaggio L:

$$\begin{aligned}\text{Prefissi}(L) &= \{y \mid x = yz \wedge x \in L \wedge z \neq \epsilon\} \\ &= \{y \mid \exists x \in L \text{ tale che } x = yz \text{ con } z \neq \epsilon\} \\ &= \{y \mid y \text{ è prefisso proprio di una stringa in } L\}\end{aligned}$$

Esempio:

- $L_1 = \{aa, aaa\}$
  Prefissi$(L_1) = \{\epsilon, a, aa\}$
- $L_2 = \{aba, aab\}$

L'insieme dei prefissi propri delle stringhe di un linguaggio L:

$$
\begin{aligned}
\text{Prefissi}(L) &= \{y \mid x = yz \wedge x \in L \wedge z \neq \epsilon\} \\
&= \{y \mid \exists x \in L \text{ tale che } x = yz \text{ con } z \neq \epsilon\} \\
&= \{y \mid y \text{ è prefisso proprio di una stringa in } L\}
\end{aligned}
$$

Esempio:

- $L_1 = \{aa, aaa\}$

  $\text{Prefissi}(L_1) = \{\epsilon, a, aa\}$
- $L_2 = \{aba, aab\}$

  $\text{Prefissi}(L_2) = \{\epsilon, a, ab, aa\}$

L'insieme dei prefissi propri delle stringhe di un linguaggio L:

$$\begin{aligned}\text{Prefissi}(L) &= \{y \mid x = yz \wedge x \in L \wedge z \neq \epsilon\} \\ &= \{y \mid \exists x \in L \text{ tale che } x = yz \text{ con } z \neq \epsilon\} \\ &= \{y \mid y \text{ è prefisso proprio di una stringa in } L\}\end{aligned}$$

Esempio:

- $L_1 = \{aa, aaa\}$
  Prefissi$(L_1) = \{\epsilon, a, aa\}$
- $L_2 = \{aba, aab\}$
  Prefissi$(L_2) = \{\epsilon, a, ab, aa\}$
- $L_3 = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$

L'insieme dei prefissi propri delle stringhe di un linguaggio L:

$$\begin{aligned}\text{Prefissi}(L) &= \{y \mid x = yz \wedge x \in L \wedge z \neq \epsilon\} \\ &= \{y \mid \exists x \in L \text{ tale che } x = yz \text{ con } z \neq \epsilon\} \\ &= \{y \mid y \text{ è prefisso proprio di una stringa in } L\}\end{aligned}$$

Esempio:

- $L_1 = \{aa, aaa\}$
  Prefissi$(L_1) = \{\epsilon, a, aa\}$
- $L_2 = \{aba, aab\}$
  Prefissi$(L_2) = \{\epsilon, a, ab, aa\}$
- $L_3 = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$
  Prefissi$(L_3) = \{a, b\}^*$

Linguaggi prefissi

Un linguaggio è prefisso se non contiene nessuno dei suoi prefissi propri.

Un linguaggio è prefisso se non contiene nessuno dei suoi prefissi propri.

Un linguaggio L è prefisso se e solo se

$$L \cap \mathrm{Prefissi}(L) = \emptyset$$

Un linguaggio è prefisso se non contiene nessuno dei suoi prefissi propri.

Un linguaggio L è prefisso se e solo se

$$L \cap \mathrm{Prefissi}(L) = \emptyset$$

Importanza pratica: se nella trasmissione dell'informazione una parte finale della stringa viene accidentalmente troncata, l'errore viene individuato.
Nella codifica dell'informazione la decodifica è immediata.

Esempio. $L_1 = \{a^n b^n \mid n \geq 1\}$ è prefisso perché

$$\text{Prefissi}(L_1) = \{a^n b^m \mid n > m \geq 1\} \cup \{a^n \mid n \geq 0\}.$$

Esempio. $L_1 = \{a^n b^n \mid n \geq 1\}$ è prefisso perché

$$\text{Prefissi}(L_1) = \{a^n b^m \mid n > m \geq 1\} \cup \{a^n \mid n \geq 0\}.$$

Esempio. $L_2 = \{a^m b^n \mid m \geq n \geq 1\}$ non è prefisso perché, ad esempio, $a^3 b^2$ e $a^3 b$ sono entrambi in L_2.

Esempio. $L_1 = \{a^n b^n \mid n \geq 1\}$ è prefisso perché

$$\text{Prefissi}(L_1) = \{a^n b^m \mid n > m \geq 1\} \cup \{a^n \mid n \geq 0\}.$$

Esempio. $L_2 = \{a^m b^n \mid m \geq n \geq 1\}$ non è prefisso perché, ad esempio, $a^3 b^2$ e $a^3 b$ sono entrambi in L_2.

Esempio. Ogni linguaggio L che contiene la stringa vuota ϵ e tale che $L \neq \{\epsilon\}$ non è prefisso.

Anche le operazioni su due stringhe possono essere estese a due linguaggi.

Anche le operazioni su due stringhe possono essere estese a due linguaggi.

Prodotto di linguaggi

Anche le operazioni su due stringhe possono essere estese a due linguaggi.

Prodotto di linguaggi

Definizione

Dati due linguaggi L' ed L'' sull'alfabeto Σ, il prodotto (o concatenazione) di L' ed L'' è

$$L'L'' = L' \circ L'' = \{xy \mid x \in L' \wedge y \in L''\}$$

Anche le operazioni su due stringhe possono essere estese a due linguaggi.

Prodotto di linguaggi

Definizione

Dati due linguaggi L' ed L'' sull'alfabeto Σ, il prodotto (o concatenazione) di L' ed L'' è

$$L'L'' = L' \circ L'' = \{xy \mid x \in L' \wedge y \in L''\}$$

$L'L''$ è l'insieme di tutte le stringhe che sono concatenazione di una stringa in L' e di una stringa in L''.

Esempio:

$$\begin{aligned} L_1 &= \{aa, aaa\} \\ L_2 &= \{aba, aab\} \\ L_1 L_2 &= \{aaaba, aaaab, aaaaba, aaaaab\} = \{a^3ba, a^4b, a^4ba, a^5b\} \end{aligned}$$

Esempio. Siano

$$L_1 = \{a^i \mid i \geq 0,\ \text{pari}\}$$

$$L_2 = \{b^j a \mid j \geq 1,\ \text{dispari}\}$$

Esempio. Siano

$$L_1 = \{a^i \mid i \geq 0,\ \text{pari}\}$$

$$L_2 = \{b^j a \mid j \geq 1,\ \text{dispari}\}$$

risulta

$$L_1 L_2 = \{a^i b^j a \mid (i \geq 0,\ \text{pari}) \wedge (j \geq 1,\ \text{dispari})\}$$

Esempio. Siano

$$L_1 = \{a^i \mid i \geq 0,\ \text{pari}\}$$

$$L_2 = \{b^j a \mid j \geq 1,\ \text{dispari}\}$$

risulta

$$L_1 L_2 = \{a^i b^j a \mid (i \geq 0,\ \text{pari}) \wedge (j \geq 1,\ \text{dispari})\}$$

Esempi di stringhe in $L_1 L_2$:

$$ba, a^2 ba, a^4 ba, b^3 a, a^2 b^3 a, a^4 b^3 a$$

Potenza di un linguaggio

Potenza di un linguaggio

Definizione

Sia L un linguaggio sull'alfabeto Σ. Definiamo:

$$
\begin{aligned}
L^0 &= \{\epsilon\}, \\
L^k &= L^{k-1}L, \quad k \geq 1
\end{aligned}
$$

Potenza di un linguaggio

Definizione

Sia L un linguaggio sull'alfabeto Σ. Definiamo:

$$\begin{aligned} L^0 &= \{\epsilon\}, \\ L^k &= L^{k-1}L, \quad k \geq 1 \end{aligned}$$

Nota.
$L^1 = L$
$L^k = \{w_1 w_2 \ldots w_k \mid w_i \in L,\ 1 \leq i \leq k\},\ k \geq 0.$

Esempio:

$$
\begin{aligned}
L_1 &= \{aa, aaa\} \\
L_1^2 &= \{aa, aaa\}\{aa, aaa\} = \{a^4, a^5, a^6\}
\end{aligned}
$$

$$
\begin{aligned}
L_2 &= \{aba, aab\} \\
L_2^2 &= \{aba, aab\}\{aba, aab\} = \{(aba)^2, abaaab, aababa, (aab)^2\}
\end{aligned}
$$

Casi particolari

$$
\begin{aligned}
\emptyset^0 &= \{\epsilon\} \\
L \cdot \emptyset &= \emptyset \cdot L = \emptyset \\
L \cdot \{\epsilon\} &= \{\epsilon\} \cdot L = L
\end{aligned}
$$

Nota. Dato un linguaggio L e un intero $m \geq 2$ consideriamo

Nota. Dato un linguaggio L e un intero $m \geq 2$ consideriamo

- il linguaggio $\{x^m \mid x \in L\}$ che ha come elementi le potenze m-esime degli elementi di L,

Nota. Dato un linguaggio L e un intero $m \geq 2$ consideriamo

- il linguaggio $\{x^m \mid x \in L\}$ che ha come elementi le potenze m-esime degli elementi di L,
- la potenza m-esima L^m di L.

Nota. Dato un linguaggio L e un intero $m \geq 2$ consideriamo

- il linguaggio $\{x^m \mid x \in L\}$ che ha come elementi le potenze m-esime degli elementi di L,
- la potenza m-esima L^m di L.

$$\{x^m \mid x \in L\} \subseteq L^m$$

Nota. Dato un linguaggio L e un intero $m \geq 2$ consideriamo

- il linguaggio $\{x^m \mid x \in L\}$ che ha come elementi le potenze m-esime degli elementi di L,
- la potenza m-esima L^m di L.

$$\{x^m \mid x \in L\} \subseteq L^m$$

In generale il primo è incluso ma è diverso dal secondo.

Nota. Dato un linguaggio L e un intero $m \geq 2$ consideriamo

- il linguaggio $\{x^m \mid x \in L\}$ che ha come elementi le potenze m-esime degli elementi di L,
- la potenza m-esima L^m di L.

$$\{x^m \mid x \in L\} \subseteq L^m$$

In generale il primo è incluso ma è diverso dal secondo.

Esempio

$$L = \{a, b\}, \quad \{x^2 \mid x \in L\} = \{aa, bb\}, \quad L^2 = \{aa, bb, ab, ba\}$$

Nota. Dato un linguaggio L e un intero $m \geq 2$, in generale $\{x^m \mid x \in L\}$ è incluso strettamente in L^m.

Nota. Dato un linguaggio L e un intero $m \geq 2$, in generale $\{x^m \mid x \in L\}$ è incluso strettamente in L^m.

Cosa accade per $m = 0$?

Nota. Dato un linguaggio L e un intero $m \geq 2$, in generale $\{x^m \mid x \in L\}$ è incluso strettamente in L^m.

Cosa accade per $m = 0$?

Cosa accade per $m = 1$?

Nota. Dato un linguaggio L e un intero $m \geq 2$, in generale $\{x^m \mid x \in L\}$ è incluso strettamente in L^m.

Cosa accade per $m = 0$?

Cosa accade per $m = 1$?

Esistono linguaggi L tali che $\{x^m \mid x \in L\} = L^m$ con $m \geq 2$?

L'operatore di potenza permette di definire in modo espressivo il linguaggio delle stringhe di lunghezza non superiore a un intero k.

L'operatore di potenza permette di definire in modo espressivo il linguaggio delle stringhe di lunghezza non superiore a un intero k.

Esempio. Sia $\Sigma = \{a, b\}$ e $k = 3$.

$$\begin{aligned} L &= \{w \in \Sigma^* \mid |w| \leq 3\} \\ &= \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb\} \end{aligned}$$

L'operatore di potenza permette di definire in modo espressivo il linguaggio delle stringhe di lunghezza non superiore a un intero k.

Esempio. Sia $\Sigma = \{a, b\}$ e $k = 3$.

$$\begin{aligned} L &= \{w \in \Sigma^* \mid |w| \leq 3\} \\ &= \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb\} \end{aligned}$$

Altre espressioni per L:

$L = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3$

L'operatore di potenza permette di definire in modo espressivo il linguaggio delle stringhe di lunghezza non superiore a un intero k.

Esempio. Sia $\Sigma = \{a, b\}$ e $k = 3$.

$$\begin{aligned} L &= \{w \in \Sigma^* \mid |w| \leq 3\} \\ &= \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb\} \end{aligned}$$

Altre espressioni per L:

$L = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3$

$L = \{\epsilon, a, b\}^3$

L'operatore di potenza permette di definire in modo espressivo il linguaggio delle stringhe di lunghezza non superiore a un intero k.

Esempio. Sia $\Sigma = \{a, b\}$ e $k = 3$.

$$\begin{aligned} L &= \{w \in \Sigma^* \mid |w| \leq 3\} \\ &= \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb\} \end{aligned}$$

Altre espressioni per L:

$L = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3$

$L = \{\epsilon, a, b\}^3$

$L' = \{w \in \Sigma^* \mid 1 \leq |w| \leq 3\} = \{a, b\}\{\epsilon, a, b\}^2$

Operazioni sui linguaggi - operazioni insiemistiche

I linguaggi sono insiemi. Quindi le operazioni insiemistiche di unione, intersezione, differenza, complemento sono definite anche per i linguaggi in quanto insiemi di stringhe;

Operazioni sui linguaggi - operazioni insiemistiche

I linguaggi sono insiemi. Quindi le operazioni insiemistiche di unione, intersezione, differenza, complemento sono definite anche per i linguaggi in quanto insiemi di stringhe;

come pure le relazioni di inclusione ($\subseteq$), di inclusione stretta ($\subset$) e di eguaglianza ($=$).

Operazioni sui linguaggi - differenza

Se L_1 ed L_2 sono due linguaggi su un alfabeto Σ, la differenza $L_1 - L_2$ di L_1 ed L_2 è

$$L_1 - L_2 = \{w \in L_1 \mid w \notin L_2\}.$$

Se L_1 ed L_2 sono due linguaggi su un alfabeto Σ, la differenza $L_1 - L_2$ di L_1 ed L_2 è

$$L_1 - L_2 = \{w \in L_1 \mid w \notin L_2\}.$$

Notazione alternativa: $L_1 \setminus L_2$.

Operazioni sui linguaggi - differenza

$\Sigma = \{a, b, c\}$

$$\begin{array}{rcl} L_1 & = & \{x \in \Sigma^* \mid |x|_a = |x|_b = |x|_c \geq 0\} \\ L_2 & = & \{x \in \Sigma^* \mid (|x|_a = |x|_b \geq 0) \wedge (|x|_c = 1)\} \end{array}$$

Operazioni sui linguaggi - differenza

$\Sigma = \{a, b, c\}$

$$\begin{aligned} L_1 &= \{x \in \Sigma^* \mid |x|_a = |x|_b = |x|_c \geq 0\} \\ L_2 &= \{x \in \Sigma^* \mid (|x|_a = |x|_b \geq 0) \wedge (|x|_c = 1)\} \end{aligned}$$

$L_1 - L_2 = \{\epsilon\} \cup \{x \in \Sigma^* \mid |x|_a = |x|_b = |x|_c \geq 2\}$

$\Sigma = \{a, b, c\}$

$$\begin{array}{rcl} L_1 & = & \{x \in \Sigma^* \mid |x|_a = |x|_b = |x|_c \geq 0\} \\ L_2 & = & \{x \in \Sigma^* \mid (|x|_a = |x|_b \geq 0) \wedge (|x|_c = 1)\} \end{array}$$

$L_1 - L_2 = \{\epsilon\} \cup \{x \in \Sigma^* \mid |x|_a = |x|_b = |x|_c \geq 2\}$

$L_2 - L_1 = \{x \in \Sigma^* \mid |x|_a = |x|_b \neq |x|_c = 1\}$

Per parlare del complemento di un linguaggio (su un alfabeto Σ) si deve introdurre un *linguaggio universale*, definito come l'insieme di tutte le stringhe su un alfabeto Σ.

Operazioni sui linguaggi - complemento

Per parlare del complemento di un linguaggio (su un alfabeto Σ) si deve introdurre un *linguaggio universale*, definito come l'insieme di tutte le stringhe su un alfabeto Σ.

L'insieme Σ^* di tutte le stringhe su un alfabeto Σ è l'unione di tutte le potenze dell'alfabeto.

$$L_{universale} = \Sigma^* = \cup_{n \geq 0} \Sigma^n$$

Operazioni sui linguaggi - complemento

Se $L \subseteq \Sigma^*$, il complemento $\overline{L}$ di L è

$$\overline{L} = \Sigma^* - L = \{w \in \Sigma^* \mid w \notin L\}$$

Operazioni sui linguaggi - complemento

Se $L \subseteq \Sigma^*$, il complemento $\overline{L}$ di L è

$$\overline{L} = \Sigma^* - L = \{w \in \Sigma^* \mid w \notin L\}$$

Notazione alternativa: $\neg L$.

Operazioni sui linguaggi - complemento

Se $L \subseteq \Sigma^*$, il complemento $\overline{L}$ di L è

$$\overline{L} = \Sigma^* - L = \{w \in \Sigma^* \mid w \notin L\}$$

Notazione alternativa: $\neg L$.

Esempio. Alfabeto: $\{a, b\}$
Linguaggio $L = \{w \in \{a, b\}^* \mid$ la prima lettera di w è $b\}$
$\overline{L}$: ?

$\overline{L}$: insieme delle stringhe su $\{a, b\}$ che non iniziano con b.
N.B.:NON insieme stringhe che iniziano con a (es. stringa vuota $\epsilon \in \overline{L}$)

Il complemento di un linguaggio finito è sempre infinito.

Il complemento di un linguaggio finito è sempre infinito.

Esempio. Alfabeto: $\{a, b\}$
Linguaggio: insieme delle stringhe di una qualsiasi lunghezza tranne che due.

Il complemento di un linguaggio finito è sempre infinito.

Esempio. Alfabeto: $\{a, b\}$
Linguaggio: insieme delle stringhe di una qualsiasi lunghezza tranne che due.
$\overline{\{a, b\}^2} = \{\epsilon\} \cup \{a, b\} \cup (\cup_{n \geq 3} \{a, b\}^n)$

Operazioni sui linguaggi - complemento

Il complemento di un linguaggio finito è sempre infinito.

Esempio. Alfabeto: $\{a, b\}$
Linguaggio: insieme delle stringhe di una qualsiasi lunghezza tranne che due.
$\overline{\{a, b\}^2} = \{\epsilon\} \cup \{a, b\} \cup (\cup_{n \geq 3} \{a, b\}^n)$

Non è detto però che il complemento di un linguaggio infinito sia finito.

Operazioni sui linguaggi - complemento

Il complemento di un linguaggio finito è sempre infinito.

Esempio. Alfabeto: $\{a, b\}$
Linguaggio: insieme delle stringhe di una qualsiasi lunghezza tranne che due.
$\overline{\{a, b\}^2} = \{\epsilon\} \cup \{a, b\} \cup (\cup_{n \geq 3} \{a, b\}^n)$

Non è detto però che il complemento di un linguaggio infinito sia finito.

Esempio. Alfabeto: $\{a\}$
$L = \{a^{2n} \mid n \geq 0\}$
$\overline{L} = \{a^{2n+1} \mid n \geq 0\}$

A eccezione dell'operazione di complemento, le operazioni finora viste non permettono una descrizione finita di linguaggi infiniti. Questo è possibile mediante l'operazione star.

A eccezione dell'operazione di complemento, le operazioni finora viste non permettono una descrizione finita di linguaggi infiniti. Questo è possibile mediante l'operazione star.

Chiusura di Kleene (o Kleene star o star o stella di Kleene)

A eccezione dell'operazione di complemento, le operazioni finora viste non permettono una descrizione finita di linguaggi infiniti. Questo è possibile mediante l'operazione star.

Chiusura di Kleene (o Kleene star o star o stella di Kleene)

Definizione

La chiusura di Kleene (o Kleene star o star) di un linguaggio L è l'unione di tutte le potenze del linguaggio:

$$L^* = \bigcup_{n \in \mathbb{N}, n \geq 0} L^n$$

Nota. L^* è il linguaggio delle stringhe ottenute concatenando un numero qualsiasi di stringhe di L:

Nota. L^* è il linguaggio delle stringhe ottenute concatenando un numero qualsiasi di stringhe di L:

$$L^* = \{w_1 w_2 \ldots w_k \mid k \geq 0,\ w_i \in L,\ 1 \leq i \leq k\}$$

Operazioni sui linguaggi - star di Kleene

Nota. L^* è il linguaggio delle stringhe ottenute concatenando un numero qualsiasi di stringhe di L:

$$L^* = \{w_1 w_2 \dots w_k \mid k \geq 0,\ w_i \in L,\ 1 \leq i \leq k\}$$

Nota. Se $k = 0$, $w_1 w_2 \dots w_k = \epsilon$ è la stringa vuota.

Esempio. Dato $L = \{ab, ba\}$, ogni stringa non vuota in L^* è la concatenazione di parole uguali ad ab o a ba.

$$L^* = \{(ab)^{n_1}(ba)^{m_1} \cdots (ab)^{n_h}(ba)^{m_h} \mid h \geq 0, n_i, m_i \geq 0, i = 1, \ldots, h\}$$

Nota.

$$\emptyset^* = \{\epsilon\}, \quad \{\epsilon\}^* = \{\epsilon\}$$

Nota.

$$\emptyset^* = \{\epsilon\}, \quad \{\epsilon\}^* = \{\epsilon\}$$

I linguaggi $\emptyset, \{\epsilon\}$ sono gli unici tali che la loro chiusura di Kleene è un linguaggio finito. Altrimenti, anche se L è finito, L^* è infinito.

- Se il linguaggio è un alfabeto Σ, la sua chiusura di Kleene Σ^* è il linguaggio universale.

Operazioni sui linguaggi - star di Kleene

- Se il linguaggio è un alfabeto Σ, la sua chiusura di Kleene Σ^* è il linguaggio universale.

- Un linguaggio formale L su un alfabeto Σ è un sottoinsieme di Σ^*: $L \subseteq \Sigma^*$.

Operazioni sui linguaggi - star di Kleene

- Se il linguaggio è un alfabeto Σ, la sua chiusura di Kleene Σ^* è il linguaggio universale.

- Un linguaggio formale L su un alfabeto Σ è un sottoinsieme di Σ^*: $L \subseteq \Sigma^*$.

- A volte L^* coincide con L.

Operazioni sui linguaggi - star di Kleene

- Se il linguaggio è un alfabeto Σ, la sua chiusura di Kleene Σ^* è il linguaggio universale.

- Un linguaggio formale L su un alfabeto Σ è un sottoinsieme di Σ^*: $L \subseteq \Sigma^*$.

- A volte L^* coincide con L.

  Esempio Se $L = \{a^{2n} \mid n \geq 0, n \in \mathbb{N}\}$, allora $L^* = \{a^{2n} \mid n \geq 0, n \in \mathbb{N}\} = L$.

$L \subseteq L^*$ **(monotonicità)**

$L \subseteq L^*$ **(monotonicità)**

$(x \in L^* \wedge y \in L^*) \rightarrow xy \in L^*$ **(chiusura rispetto alla concatenazione)**

$L \subseteq L^*$ **(monotonicità)**

$(x \in L^* \wedge y \in L^*) \rightarrow xy \in L^*$ **(chiusura rispetto alla concatenazione)**

$(L^*)^* = L^*$ **(idempotenza)**

$L \subseteq L^*$ **(monotonicità)**

$(x \in L^* \wedge y \in L^*) \rightarrow xy \in L^*$ **(chiusura rispetto alla concatenazione)**

$(L^*)^* = L^*$ **(idempotenza)**

$(L^*)^R = (L^R)^*$ **(commutatività di star e riflessione)**

Esempio Se $L = \{a^{2n} \mid n \geq 0, n \in \mathbb{N}\}$, allora $L = \{aa\}^*$.

Esempio Se $L = \{a^{2n} \mid n \geq 0, n \in \mathbb{N}\}$, allora $L = \{aa\}^*$.
Quindi

$$\begin{aligned} L^* &= \{a^{2n} \mid n \geq 0, n \in \mathbb{N}\}^* \\ &= (\{aa\}^*)^* \\ &= \{aa\}^* = L \end{aligned}$$

per la proprietà di idempotenza.

Molti linguaggi di programmazione assegnano nomi o identificatori agli oggetti (variabili, sottoprogrammi, ecc.) utilizzati.

Molti linguaggi di programmazione assegnano nomi o identificatori agli oggetti (variabili, sottoprogrammi, ecc.) utilizzati.

Una regola comune a molti linguaggi dice che un identificatore è una stringa che inizia con una lettera in $\{A, B, \ldots, Z\}$ seguita da un numero qualsiasi di lettere e cifre in $\{0, 1, \ldots, 9\}$.

Molti linguaggi di programmazione assegnano nomi o identificatori agli oggetti (variabili, sottoprogrammi, ecc.) utilizzati.

Una regola comune a molti linguaggi dice che un identificatore è una stringa che inizia con una lettera in $\{A, B, \ldots, Z\}$ seguita da un numero qualsiasi di lettere e cifre in $\{0, 1, \ldots, 9\}$.

Esempio SOMMA32A35.

Definiti gli alfabeti

$$\Sigma_{A\ell} = \{A, B, \ldots, Z\}, \quad \Sigma_N = \{0, 1, \ldots, 9\}$$

il linguaggio $I \subseteq (\Sigma_{A\ell} \cup \Sigma_N)^*$ degli identificatori risulta:

Definiti gli alfabeti

$$\Sigma_{A\ell} = \{A, B, \ldots, Z\}, \quad \Sigma_N = \{0, 1, \ldots, 9\}$$

il linguaggio $I \subseteq (\Sigma_{A\ell} \cup \Sigma_N)^*$ degli identificatori risulta:

$$I = \Sigma_{A\ell}(\Sigma_{A\ell} \cup \Sigma_N)^*$$

Sia I_5 il linguaggio degli identificatori di lunghezza al più 5.

Sia I_5 il linguaggio degli identificatori di lunghezza al più 5.

Posto $\Sigma = \Sigma_{A\ell} \cup \Sigma_N$, risulta

Sia I_5 il linguaggio degli identificatori di lunghezza al più 5.
Posto $\Sigma = \Sigma_{A\ell} \cup \Sigma_N$, risulta

$$\begin{aligned} I_5 &= \Sigma_{A\ell}(\Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4) \\ &= \Sigma_{A\ell}(\{\epsilon\} \cup \Sigma \cup \Sigma^2 \cup \Sigma^3 \cup \Sigma^4) \\ &= \Sigma_{A\ell}(\{\epsilon\} \cup \Sigma)^4 \end{aligned}$$

Una operazione utile (ma non indispensabile) è la chiusura positiva (o croce).

Chiusura positiva o croce

Una operazione utile (ma non indispensabile) è la chiusura positiva (o croce).

Chiusura positiva

Una operazione utile (ma non indispensabile) è la chiusura positiva (o croce).

Chiusura positiva

Definizione

Per un linguaggio L sull'alfabeto Σ, definiamo

$$L^+ = \bigcup_{n \in \mathbb{N}, n > 0} L^n$$

Una operazione utile (ma non indispensabile) è la chiusura positiva (o croce).

Chiusura positiva

Definizione

Per un linguaggio L sull'alfabeto Σ, definiamo

$$L^+ = \bigcup_{n \in \mathbb{N}, n > 0} L^n$$

La chiusura positiva si distingue dalla chiusura di Kleene perché nell'unione non compare la potenza di L con esponente zero $L^0 = \{\epsilon\}$.

Valgono le relazioni

Valgono le relazioni

$$L^+ \subseteq L^*$$

Valgono le relazioni

$$L^+ \subseteq L^*$$

$$\epsilon \in L^+ \text{ se e solo se } \epsilon \in L$$

Valgono le relazioni

$$L^{+} \subseteq L^{*}$$

$$\epsilon \in L^{+} \text{ se e solo se } \epsilon \in L$$

$$L^{+} = LL^{*} = L^{*}L$$

Nota. L^+ è il linguaggio delle stringhe ottenute concatenando un numero positivo qualsiasi di stringhe di L:

Nota. L^+ è il linguaggio delle stringhe ottenute concatenando un numero positivo qualsiasi di stringhe di L:

$$L^+ = \{w_1 w_2 \cdots w_k \mid k > 0,\ w_i \in L,\ 1 \leq i \leq k\}$$

Esempio. Dato $L = \{ab, ba\}$, ogni stringa in L^+ è la concatenazione di parole uguali ad *ab* o a *ba*.

$$L^+ = \{(ab)^{n_1}(ba)^{m_1} \cdots (ab)^{n_h}(ba)^{m_h} \mid h > 0, n_i, m_i \geq 0, \exists j\ n_j + m_j \neq 0\}$$

Esempio.

$$\{\epsilon, aa\}^+ = \{a^{2n} \mid n \geq 0, n \in \mathbb{N}\} = \{aa\}^*$$

Esempio.

$$\{\epsilon, aa\}^+ = \{a^{2n} \mid n \geq 0, n \in \mathbb{N}\} = \{aa\}^*$$

Esempio. Le stringhe di lunghezza almeno quattro:

$$\Sigma^4\Sigma^* = (\Sigma^+)^4$$

Le operazioni sui linguaggi finora viste non possono essere utilizzate per accorciare le stringhe del linguaggio/dei linguaggi su cui operano.

Quoziente

Le operazioni sui linguaggi finora viste non possono essere utilizzate per accorciare le stringhe del linguaggio/dei linguaggi su cui operano.

L'operazione di *quoziente (destro)* accorcia una stringa del primo linguaggio cancellandone un suffisso appartenente al secondo.

Quoziente (destro)

Quoziente (destro)

Definizione

Il quoziente (destro) di L' rispetto ad L'' è definito come

$$\begin{aligned} L = L'(L'')^{-1} &= \{y \mid (x = yz \in L') \wedge (z \in L'')\} \\ &= \{y \mid \text{ esiste } z \in L'' \text{ tale che } yz \in L'\} \end{aligned}$$

Quoziente (destro)

Definizione

Il quoziente (destro) di L' rispetto ad L'' è definito come

$$\begin{aligned} L = L'(L'')^{-1} &= \{y \mid (x = yz \in L') \wedge (z \in L'')\} \\ &= \{y \mid \text{ esiste } z \in L'' \text{ tale che } yz \in L'\} \end{aligned}$$

Notazione alternativa: $L'/_D L''$

Esempio:

$$L_2 = \{aba, aab\}$$

$$L_2 a^{-1} = L_2 \{a\}^{-1} = \{ab\}$$

$$L_2 b^{-1} = L_2 \{b\}^{-1} = \{aa\}$$

$$L_1 = \{aa, aaa\}$$

$$L_2 L_1^{-1} = \emptyset = L_1 L_2^{-1}$$

Esempio. Siano

$$L' = \{a^{2n}b^{2n} \mid n \in \mathbb{N}, n > 0\}, \quad L'' = \{b^{2n+1} \mid n \in \mathbb{N}, n \geq 0\}$$

Esempio. Siano

$$L' = \{a^{2n}b^{2n} \mid n \in \mathbb{N}, n > 0\}, \quad L'' = \{b^{2n+1} \mid n \in \mathbb{N}, n \geq 0\}$$

$$L'(L'')^{-1} = \{a^r b^s \mid r, s \in \mathbb{N}, (r \geq 2, \text{ pari}) \wedge (1 \leq s < r, s \text{ dispari})\}$$

Esempio. Siano

$$L' = \{a^{2n}b^{2n} \mid n \in \mathbb{N}, n > 0\}, \quad L'' = \{b^{2n+1} \mid n \in \mathbb{N}, n \geq 0\}$$

$$L'(L'')^{-1} = \{a^r b^s \mid r, s \in \mathbb{N}, (r \geq 2, \text{ pari}) \wedge (1 \leq s < r, s \text{ dispari})\}$$

$$L''(L')^{-1} = \emptyset$$

Esiste un'operazione duale, il *quoziente sinistro* $L'/_S L''$ che accorcia una stringa del primo linguaggio cancellandone un prefisso appartenente al secondo.

Esiste un'operazione duale, il *quoziente sinistro* $L'/_S L''$ che accorcia una stringa del primo linguaggio cancellandone un prefisso appartenente al secondo.

$$L = (L'')^{-1} L' = \{z \mid (x = yz \in L') \wedge (y \in L'')\}$$

Quoziente

Esiste un'operazione duale, il *quoziente sinistro* $L'/_S L''$ che accorcia una stringa del primo linguaggio cancellandone un prefisso appartenente al secondo.

$$L = (L'')^{-1}L' = \{z \mid (x = yz \in L') \wedge (y \in L'')\}$$

Notazione alternativa: $L'/_S L''$