Automi finiti, Linguaggi ed Espressioni Regolari

Rocco Zaccagnino

Dipartimento di Informatica

Università degli Studi di Salerno

Elementi di Teoria della Computazione: *lezione 6*

Linguaggi regolari (REG)

Relazione tra DFA e Linguaggi

Abbiamo visto..

individuare il linguaggio $L(\mathbb{M})$ riconosciuto dall'automa $\mathbb{M}$

Definizione formale di computazione

Definizione formalmente il linguaggio accettato da un automa attraverso la nozione di

funzione di transizione “estesa” a stringhe

Definizione formale di computazione

Definizione formalmente il linguaggio accettato da un automa attraverso la nozione di

funzione di transizione "estesa" a stringhe

DEF[`funzione di transizione estesa`]

passo base: $\forall q \in Q$, $\hat{\delta}(q, \epsilon) = q$

Definizione formale di computazione

Definizione formalmente il linguaggio accettato da un automa attraverso la nozione di

funzione di transizione "estesa" a stringhe

DEF[`funzione di transizione estesa`]

passo base: $\forall q \in Q$, $\hat{\delta}(q, \epsilon) = q$

passo ricorsivo: $\forall q \in Q$, $w \in \Sigma^*$, $a \in \Sigma$,

$$\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$$

Definizione formale di computazione

DEF[`linguaggio riconosciuto`]

Sia $\mathbb{A} = (Q, \Sigma, \delta, q_0, F)$ un DFA. Il **linguaggio accettato** da A é

$$L(\mathbb{A}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$$

Definizione formale di computazione

DEF[`linguaggio riconosciuto`]

Sia $\mathbb{A} = (Q, \Sigma, \delta, q_0, F)$ un DFA. Il **linguaggio accettato** da A é

$$L(\mathbb{A}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$$

- il primo stato della sequenza è quello iniziale q_0,

Definizione formale di computazione

DEF[`linguaggio riconosciuto`]

Sia $\mathbb{A} = (Q, \Sigma, \delta, q_0, F)$ un DFA. Il **linguaggio accettato** da A é

$$L(\mathbb{A}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$$

- il primo stato della sequenza è quello iniziale q_0,
- l'ultimo stato della sequenza è uno stato finale ($q \in F$),

Definizione formale di computazione

DEF[linguaggio riconosciuto]

Sia $\mathbb{A} = (Q, \Sigma, \delta, q_0, F)$ un DFA. Il **linguaggio accettato** da A é

$$L(\mathbb{A}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$$

- il primo stato della sequenza è quello iniziale q_0,
- l'ultimo stato della sequenza è uno stato finale ($q \in F$),
- la sequenza di stati corrisponde a transizione valide per la stringa w.

Linguaggio regolare

DEF[`linguaggio regolare`]

Un linguaggio L è regolare **se e solo se** esiste un DFA $\mathbb{M}$ che lo riconosce ($L = L(\mathbb{M})$).

Linguaggio regolare

DEF`[linguaggio regolare]`

Un linguaggio L è regolare **se e solo se** esiste un DFA $\mathbb{M}$ che lo riconosce ($L = L(\mathbb{M})$).

- i linguaggi riconosciuti da tutti i DFA formano la classe dei linguaggi regolari ($\mathtt{REG}$),

Linguaggio regolare

DEF[`linguaggio regolare`]

Un linguaggio L è regolare **se e solo se** esiste un DFA $\mathbb{M}$ che lo riconosce ($L = L(\mathbb{M})$).

- i linguaggi riconosciuti da tutti i DFA formano la classe dei linguaggi regolari ($\mathtt{REG}$),
- non tutti i linguaggi sono regolari (ex. $\{a^n b^n \mid n \geq 0\}$).

Osservazione: come possiamo scoprire che un linguaggio è $\mathtt{REG}$?

Linguaggio regolare

DEF[`linguaggio regolare`]

Un linguaggio L è regolare **se e solo se** esiste un DFA $\mathbb{M}$ che lo riconosce ($L = L(\mathbb{M})$).

- i linguaggi riconosciuti da tutti i DFA formano la classe dei linguaggi regolari ($\mathtt{REG}$),
- non tutti i linguaggi sono regolari (ex. $\{a^n b^n \mid n \geq 0\}$).

Osservazione: come possiamo scoprire che un linguaggio è $\mathtt{REG}$?

- progettiamo direttamente un DFA che lo riconosce,

Linguaggio regolare

DEF[`linguaggio regolare`]

Un linguaggio L è regolare **se e solo se** esiste un DFA $\mathbb{M}$ che lo riconosce ($L = L(\mathbb{M})$).

- i linguaggi riconosciuti da tutti i DFA formano la classe dei linguaggi regolari ($\mathtt{REG}$),
- non tutti i linguaggi sono regolari (ex. $\{a^n b^n \mid n \geq 0\}$).

Osservazione: come possiamo scoprire che un linguaggio è $\mathtt{REG}$?

- progettiamo direttamente un DFA che lo riconosce,
- vedremo altri metodi per dimostrare che un linguaggio è regolare.

Esempi di linguaggi regolari

- tutti i **linguaggi finiti**,

Esempi di linguaggi regolari

- tutti i **linguaggi finiti**,
- $\{a^n b \mid n \geq 0\}$ (parole che finiscono per b e contengono esattamente una b),

Esempi di linguaggi regolari

- tutti i **linguaggi finiti**,
- $\{a^n b \mid n \geq 0\}$ (parole che finiscono per b e contengono esattamente una b),
- tutte le stringhe in $\{a, b\}^*$ con prefisso ab,

Esempi di linguaggi regolari

- tutti i **linguaggi finiti**,
- $\{a^n b \mid n \geq 0\}$ (parole che finiscono per b e contengono esattamente una b),
- tutte le stringhe in $\{a, b\}^*$ con prefisso ab,
- tutte le stringhe in $\{O, 1\}^*$ che contengono $OO1$,

Esempi di linguaggi regolari

- tutti i **linguaggi finiti**,
- $\{a^n b \mid n \geq 0\}$ (parole che finiscono per b e contengono esattamente una b),
- tutte le stringhe in $\{a, b\}^*$ con prefisso ab,
- tutte le stringhe in $\{O, 1\}^*$ che contengono $OO1$,
- tutte le stringhe in $\{O, 1\}^*$ con un numero pari di 1.

Da linguaggio a DFA

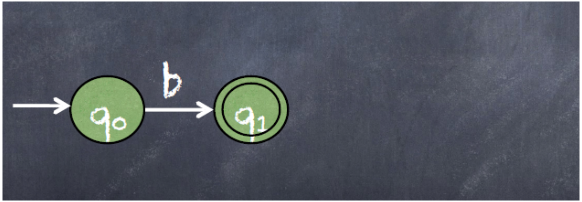
Creare un DFA che riconosce il linguaggio

$$L = \{a^n b \mid n \geq 0\} = \{b, ab, aab, aaab, ...\}$$

Da linguaggio a DFA

Creare un DFA che riconosce il linguaggio

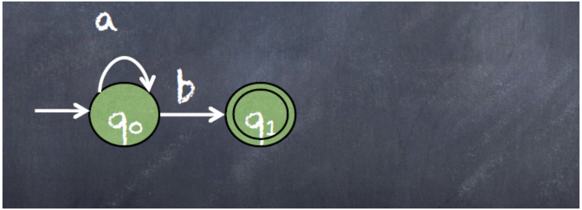
$$L = \{a^n b \mid n \geq 0\} = \{b, ab, aab, aaab, ...\}$$



Da linguaggio a DFA

Creare un DFA che riconosce il linguaggio

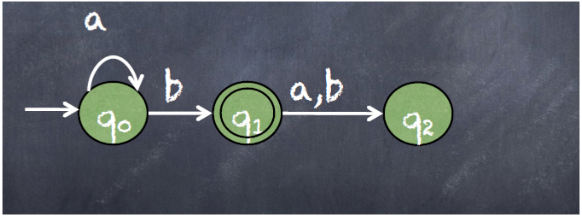
$$L = \{a^n b \mid n \geq 0\} = \{b, ab, aab, aaab, ...\}$$



Da linguaggio a DFA

Creare un DFA che riconosce il linguaggio

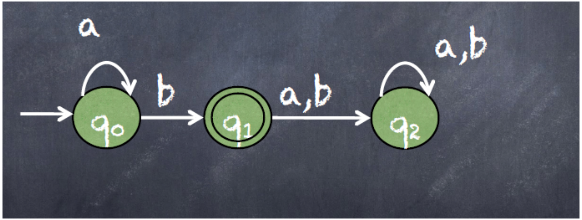
$$L = \{a^n b \mid n \geq 0\} = \{b, ab, aab, aaab, ...\}$$



Da linguaggio a DFA

Creare un DFA che riconosce il linguaggio

$$L = \{a^n b \mid n \geq 0\} = \{b, ab, aab, aaab, ...\}$$



Da linguaggio a DFA

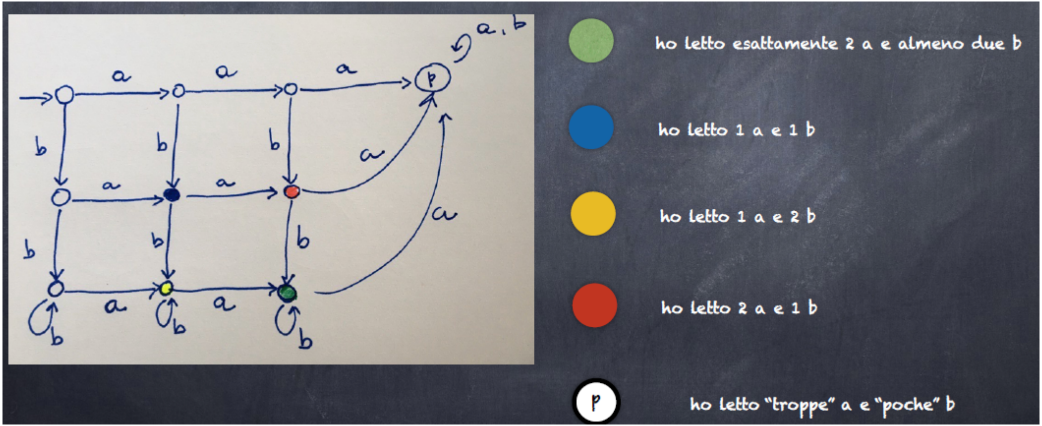
Ma i DFA possono riconoscere anche linguaggi più complessi?

stringhe contenenti esattamente due **a** ed almeno due **b**.

Da linguaggio a DFA

Ma i DFA possono riconoscere anche linguaggi più complessi?

stringhe contenenti esattamente due **a** ed almeno due **b**.



Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Siano L_1 e L_2 linguaggi regolari, allora:

- **unione**: $L_1 \cup L_2$ è regolare

Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Siano L_1 e L_2 linguaggi regolari, allora:

- **unione**: $L_1 \cup L_2$ è regolare
- **concatenazione**: $L_1 \circ L_2$ è regolare

Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Siano L_1 e L_2 linguaggi regolari, allora:

- **unione**: $L_1 \cup L_2$ è regolare
- **concatenazione**: $L_1 \circ L_2$ è regolare
- **star**: L_1^*, L_2^* sono regolari

Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Siano L_1 e L_2 linguaggi regolari, allora:

- **unione**: $L_1 \cup L_2$ è regolare
- **concatenazione**: $L_1 \circ L_2$ è regolare
- **star**: L_1^*, L_2^* sono regolari
- **reversal**: L_1^R, L_2^R sono regolari

Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Siano L_1 e L_2 linguaggi regolari, allora:

- **unione**: $L_1 \cup L_2$ è regolare
- **concatenazione**: $L_1 \circ L_2$ è regolare
- **star**: L_1^*, L_2^* sono regolari
- **reversal**: L_1^R, L_2^R sono regolari
- **complemento**: $\overline{L_1}, \overline{L_2}$ sono regolari

Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Siano L_1 e L_2 linguaggi regolari, allora:

- **unione**: $L_1 \cup L_2$ è regolare
- **concatenazione**: $L_1 \circ L_2$ è regolare
- **star**: L_1^*, L_2^* sono regolari
- **reversal**: L_1^R, L_2^R sono regolari
- **complemento**: $\overline{L_1}, \overline{L_2}$ sono regolari
- **intersezione**: $L_1 \cap L_2$ è regolare

Chiusura dei linguaggi regolari

Che significa chiusura rispetto ad un'operazione?

Chiusura dei linguaggi regolari

Che significa chiusura rispetto ad un'operazione?

Una classe di oggetti è "chiusa" rispetto ad un'operazione se l'applicazione di questa operazione ad elementi della classe restituisce un oggetto ancora della classe.

Chiusura dei linguaggi regolari

Che significa chiusura rispetto ad un'operazione?

Una classe di oggetti è "chiusa" rispetto ad un'operazione se l'applicazione di questa operazione ad elementi della classe restituisce un oggetto ancora della classe.

Ad esempio...

Chiusura dei linguaggi regolari

Che significa chiusura rispetto ad un'operazione?

Una classe di oggetti è "chiusa" rispetto ad un'operazione se l'applicazione di questa operazione ad elementi della classe restituisce un oggetto ancora della classe.

Ad esempio...

- $\mathbb{N}$ è chiuso rispetto a "$+$" e "$\times$": $\forall a, b \in \mathbb{N}$, $a + b \in \mathbb{N}$ and $a \times b \in \mathbb{N}$

Chiusura dei linguaggi regolari

Che significa chiusura rispetto ad un'operazione?

Una classe di oggetti è "chiusa" rispetto ad un'operazione se l'applicazione di questa operazione ad elementi della classe restituisce un oggetto ancora della classe.

Ad esempio...

- $\mathbb{N}$ è chiuso rispetto a "$+$" e "$\times$": $\forall a, b \in \mathbb{N}$, $a + b \in \mathbb{N}$ and $a \times b \in \mathbb{N}$
- $\mathbb{N}$ non è chiuso rispetto a ":", i.e., esistono $a, b \in \mathbb{N}$ tali che $a : b \notin \mathbb{N}$ (ex. $3 : 2 \notin \mathbb{N}$)

Chiusura di REG rispetto all'unione

TEOREMA[`chiusura di REG rispetto all'unione`]

$$L_1 \text{ e } L_2 \text{ regolari} \Longrightarrow L_1 \cup L_2 \text{ regolare}$$

Chiusura di REG rispetto all'unione

TEOREMA[`chiusura di REG rispetto all'unione`]

$$L_1 \text{ e } L_2 \text{ regolari} \implies L_1 \cup L_2 \text{ regolare}$$

Vedremo una dimostrazione **costruttiva**: costruiremo un DFA $\mathbb{M}$ tale che

$$L(\mathbb{M}) = L_1 \cup L_2$$

Chiusura di REG rispetto all'unione

Dimostrazione.

Per ipotesi, L_1 e L_2 **regolari**

Chiusura di REG rispetto all'unione

Dimostrazione.

Per ipotesi, L_1 e L_2 **regolari** $\Longrightarrow \exists$ 2 DFA $\mathbb{M}_1$ e $\mathbb{M}_2$ tali che $L_1 = L(\mathbb{M}_1)$ e $L_2 = L(\mathbb{M}_2)$ e:

$$L_1 \cup L_2 = L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$$

Chiusura di REG rispetto all'unione

Dimostrazione.

Per ipotesi, L_1 e L_2 **regolari** $\Longrightarrow \exists$ 2 DFA $\mathbb{M}_1$ e $\mathbb{M}_2$ tali che $L_1 = L(\mathbb{M}_1)$ e $L_2 = L(\mathbb{M}_2)$ e:

$$L_1 \cup L_2 = L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$$

Potremmo costruire $\mathbb{M}$ combinando $\mathbb{M}_1$ e $\mathbb{M}_2$ in qualche modo...ma come?

Chiusura di REG rispetto all'unione

Dimostrazione.

Per ipotesi, L_1 e L_2 **regolari** $\Longrightarrow \exists$ 2 DFA $\mathbb{M}_1$ e $\mathbb{M}_2$ tali che $L_1 = L(\mathbb{M}_1)$ e $L_2 = L(\mathbb{M}_2)$ e:

$$L_1 \cup L_2 = L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$$

Potremmo costruire $\mathbb{M}$ combinando $\mathbb{M}_1$ e $\mathbb{M}_2$ in qualche modo...ma come?

Affinchè $\mathbb{M}$ possa accettare $L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$, esso dovrebbe accettare una stringa w quando:

Chiusura di REG rispetto all'unione

Dimostrazione.

Per ipotesi, L_1 e L_2 **regolari** $\implies \exists$ 2 DFA $\mathbb{M}_1$ e $\mathbb{M}_2$ tali che $L_1 = L(\mathbb{M}_1)$ e $L_2 = L(\mathbb{M}_2)$ e:

$$L_1 \cup L_2 = L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$$

Potremmo costruire $\mathbb{M}$ combinando $\mathbb{M}_1$ e $\mathbb{M}_2$ in qualche modo...ma come?

Affinchè $\mathbb{M}$ possa accettare $L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$, esso dovrebbe accettare una stringa w quando:

- w è accettata da $\mathbb{M}_1$ ($w \in L(\mathbb{M}_1)$), **oppure**...

Chiusura di REG rispetto all'unione

Dimostrazione.

Per ipotesi, L_1 e L_2 **regolari** $\implies \exists$ 2 DFA $\mathbb{M}_1$ e $\mathbb{M}_2$ tali che $L_1 = L(\mathbb{M}_1)$ e $L_2 = L(\mathbb{M}_2)$ e:

$$L_1 \cup L_2 = L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$$

Potremmo costruire $\mathbb{M}$ combinando $\mathbb{M}_1$ e $\mathbb{M}_2$ in qualche modo...ma come?

Affinchè $\mathbb{M}$ possa accettare $L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$, esso dovrebbe accettare una stringa w quando:

- w è accettata da $\mathbb{M}_1$ ($w \in L(\mathbb{M}_1)$), **oppure**...
- w è accettata da $\mathbb{M}_2$ ($w \in L(\mathbb{M}_2)$).

Chiusura di REG rispetto all'unione

Dimostrazione.

Per ipotesi, L_1 e L_2 **regolari** $\Longrightarrow \exists$ 2 DFA $\mathbb{M}_1$ e $\mathbb{M}_2$ tali che $L_1 = L(\mathbb{M}_1)$ e $L_2 = L(\mathbb{M}_2)$ e:

$$L_1 \cup L_2 = L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$$

Potremmo costruire $\mathbb{M}$ combinando $\mathbb{M}_1$ e $\mathbb{M}_2$ in qualche modo...ma come?

Affinchè $\mathbb{M}$ possa accettare $L(\mathbb{M}_1) \cup L(\mathbb{M}_2)$, esso dovrebbe accettare una stringa w quando:

- w è accettata da $\mathbb{M}_1$ ($w \in L(\mathbb{M}_1)$), **oppure**...
- w è accettata da $\mathbb{M}_2$ ($w \in L(\mathbb{M}_2)$).

dovrebbe simulare $\mathbb{M}_1$ e $\mathbb{M}_2$!

Chiusura di REG rispetto all'unione

Dimostrazione...

Ad esempio, potremmo immaginare che $\mathbb{M}$ simuli prima $\mathbb{M}_1$ su w e poi $\mathbb{M}_2$.

Chiusura di REG rispetto all'unione

Dimostrazione...

Ad esempio, potremmo immaginare che $\mathbb{M}$ simuli prima $\mathbb{M}_1$ su w e poi $\mathbb{M}_2$.

- questo non va bene perchè una volta che i simboli in input sono stati letti e usati per simulare $\mathbb{M}_1$, dopo **non possiamo riavvolgere il nastro** per simulare $\mathbb{M}_2$!

Chiusura di REG rispetto all'unione

Dimostrazione...

Ad esempio, potremmo immaginare che $\mathbb{M}$ simuli prima $\mathbb{M}_1$ su w e poi $\mathbb{M}_2$.

- questo non va bene perchè una volta che i simboli in input sono stati letti e usati per simulare $\mathbb{M}_1$, dopo **non possiamo riavvolgere il nastro** per simulare $\mathbb{M}_2$!

Dobbiamo simulare $\mathbb{M}_1$ e $\mathbb{M}_2$ contemporaneamente!

Chiusura di REG rispetto all'unione

Dimostrazione...

Ad esempio, potremmo immaginare che $\mathbb{M}$ simuli prima $\mathbb{M}_1$ su w e poi $\mathbb{M}_2$.

- questo non va bene perchè una volta che i simboli in input sono stati letti e usati per simulare $\mathbb{M}_1$, dopo **non possiamo riavvolgere il nastro** per simulare $\mathbb{M}_2$!

Dobbiamo simulare $\mathbb{M}_1$ e $\mathbb{M}_2$ contemporaneamente!

Idea: dato la parte di input letto fino ad un dato momento, bisogna portare traccia sia dello stato in cui si troverebbe $\mathbb{M}_1$ che quello in cui si troverebbe $\mathbb{M}_2$.

Chiusura di REG rispetto all'unione

Dimostrazione...

Ad esempio, potremmo immaginare che $\mathbb{M}$ simuli prima $\mathbb{M}_1$ su w e poi $\mathbb{M}_2$.

- questo non va bene perchè una volta che i simboli in input sono stati letti e usati per simulare $\mathbb{M}_1$, dopo **non possiamo riavvolgere il nastro** per simulare $\mathbb{M}_2$!

Dobbiamo simulare $\mathbb{M}_1$ e $\mathbb{M}_2$ contemporaneamente!

Idea: dato la parte di input letto fino ad un dato momento, bisogna portare traccia sia dello stato in cui si troverebbe $\mathbb{M}_1$ che quello in cui si troverebbe $\mathbb{M}_2$.

Ricordare una coppia di stati!

Chiusura di REG rispetto all'unione

Dimostrazione...

Idea: data la parte di input letto fino ad un dato momento, bisogna portare traccia sia dello stato in cui si troverebbe $\mathbb{M}_1$ che quello in cui si troverebbe $\mathbb{M}_2$.

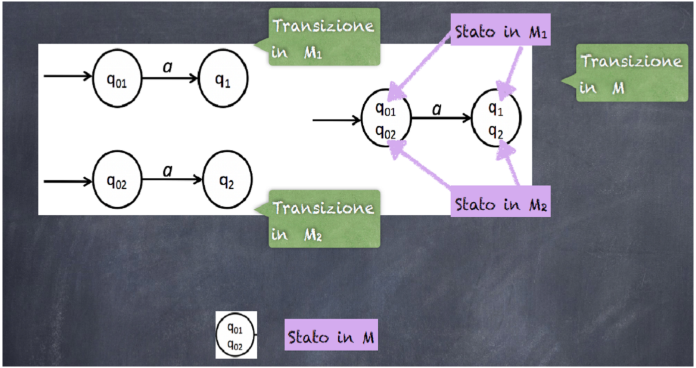
Ricordare una coppia di stati!

Chiusura di REG rispetto all'unione

Dimostrazione...

Idea: data la parte di input letto fino ad un dato momento, bisogna portare traccia sia dello stato in cui si troverebbe $\mathbb{M}_1$ che quello in cui si troverebbe $\mathbb{M}_2$.

Ricordare una coppia di stati!

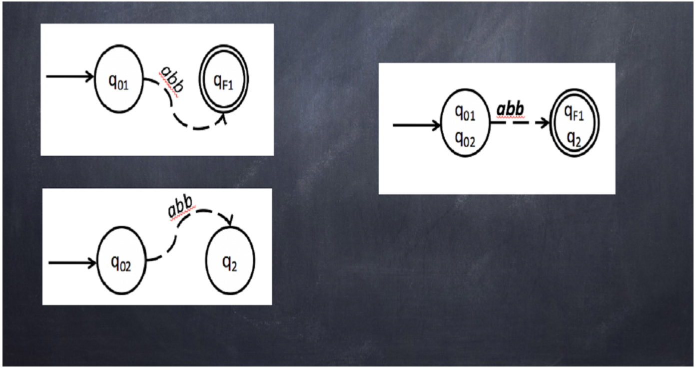


Chiusura di REG rispetto all'unione

Dimostrazione...

Idea: dato la parte di input letto fino ad un dato momento, bisogna portare traccia sia dello stato in cui si troverebbe $\mathbb{M}_1$ che quello in cui si troverebbe $\mathbb{M}_2$.

Ricordare una coppia di stati!



Chiusura di REG rispetto all'unione

Costruzione formale

Chiusura di REG rispetto all'unione

Costruzione formale

Siano $\mathbb{M}_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ e $\mathbb{M}_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$.

Chiusura di REG rispetto all'unione

Costruzione formale

Siano $\mathbb{M}_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ e $\mathbb{M}_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$.

Definiamo $\mathbb{M} = (Q, \Sigma, \delta, q_0, F)$ che riconosce $L_1 \cup L_2$ ($L(\mathbb{M}) = L_1 \cup L_2$), come segue:

- $Q = \{(q_1, q_2) \mid q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$: insieme **prodotto cartesiano** di $Q_1 \times Q_2$.

Chiusura di REG rispetto all'unione

Costruzione formale

Siano $\mathbb{M}_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ e $\mathbb{M}_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$.

Definiamo $\mathbb{M} = (Q, \Sigma, \delta, q_0, F)$ che riconosce $L_1 \cup L_2$ ($L(\mathbb{M}) = L_1 \cup L_2$), come segue:

- $Q = \{(q_1, q_2) \mid q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$: insieme **prodotto cartesiano** di $Q_1 \times Q_2$.
- Σ è lo stesso alfabeto usato in $\mathbb{M}_1$ e $\mathbb{M}_2$: per semplicità assumiamo che $\mathbb{M}_1$ e $\mathbb{M}_2$ usino lo stesso alfabeto, ma continua ad essere vero anche se hanno alfabeti diversi ($\Sigma = \Sigma_1 \cup \Sigma_2$).

Chiusura di REG rispetto all'unione

Costruzione formale

Siano $\mathbb{M}_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ e $\mathbb{M}_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$.

Definiamo $\mathbb{M} = (Q, \Sigma, \delta, q_0, F)$ che riconosce $L_1 \cup L_2$ ($L(\mathbb{M}) = L_1 \cup L_2$), come segue:

- $Q = \{(q_1, q_2) \mid q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$: insieme **prodotto cartesiano** di $Q_1 \times Q_2$.
- Σ è lo stesso alfabeto usato in $\mathbb{M}_1$ e $\mathbb{M}_2$: per semplicità assumiamo che $\mathbb{M}_1$ e $\mathbb{M}_2$ usino lo stesso alfabeto, ma continua ad essere vero anche se hanno alfabeti diversi ($\Sigma = \Sigma_1 \cup \Sigma_2$).
- For each $(q_1, q_2) \in Q$ e ogni $a \in \Sigma$:

$$\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$$

Chiusura di REG rispetto all'unione

Costruzione formale

Siano $\mathbb{M}_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ e $\mathbb{M}_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$.

Definiamo $\mathbb{M} = (Q, \Sigma, \delta, q_0, F)$ che riconosce $L_1 \cup L_2$ ($L(\mathbb{M}) = L_1 \cup L_2$), come segue:

- $Q = \{(q_1, q_2) \mid q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$: insieme **prodotto cartesiano** di $Q_1 \times Q_2$.
- Σ è lo stesso alfabeto usato in $\mathbb{M}_1$ e $\mathbb{M}_2$: per semplicità assumiamo che $\mathbb{M}_1$ e $\mathbb{M}_2$ usino lo stesso alfabeto, ma continua ad essere vero anche se hanno alfabeti diversi ($\Sigma = \Sigma_1 \cup \Sigma_2$).
- For each $(q_1, q_2) \in Q$ e ogni $a \in \Sigma$:

$$\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$$

- $q_0 = (q_1, q_2)$
- F is the set of pairs in which either member is an accept state of $\mathbb{M}_1$ or $\mathbb{M}_2$:

$$F = \{(q_1, q_2) \mid q_1 \in F_1 \textit{ oppure } q_2 \in F_2\}$$

Chiusura di REG rispetto all'intersezione

TEOREMA[`chiusura di REG rispetto all'intersezione`]

$$L_1 \text{ e } L_2 \text{ regolari} \implies L_1 \cap L_2 \text{ regolare}$$

Chiusura di REG rispetto all'intersezione

TEOREMA[`chiusura di REG rispetto all'intersezione`]

$$L_1 \text{ e } L_2 \text{ regolari} \Longrightarrow L_1 \cap L_2 \text{ regolare}$$

...da fare a casa!

to be continued..