

# Equivalenza tra ASF, RG e RE

Corso di Fondamenti di Informatica - modulo 1  
Corso di Laurea in Informatica  
Università di Roma "Tor Vergata"  
a.a. 2020-2021

Giorgio Gambosi

## 1 Grammatiche regolari

### ASF e grammatiche di tipo 3

Per ogni grammatica regolare  $\mathcal{G} = \langle V_T, V_N, P, S \rangle$ , esiste un ASFND  $\mathcal{A}_N = \langle \Sigma, Q, \delta_N, q_0, F \rangle$  che riconosce il linguaggio che essa genera.

Viceversa, per ogni ASFND  $\mathcal{A}_N$  esiste una grammatica regolare che genera il linguaggio che esso riconosce.

### ASF e grammatiche di tipo 3

Sia  $\mathcal{G} = \langle V_T, V_N, P, S \rangle$  una grammatica di tipo 3, con al più la sola  $\varepsilon$ -produzione  $S \rightarrow \varepsilon$ .

Definiamo una procedura che partendo da  $\mathcal{G}$  produca un ASFND  $\mathcal{A}_N = \langle \Sigma, Q, \delta_N, q_0, F \rangle$  equivalente (che accetta tutte e sole stringhe prodotte da  $\mathcal{G}$ ).

### Da $\mathcal{G}$ a $\mathcal{A}_N$

$$\Sigma = V_T$$

$$Q = \{q_I \mid I \in V_N\} \cup \{q_F\}$$

$$q_0 = q_S$$

$$F = \begin{cases} \{q_0, q_F\} & \text{se } S \rightarrow \varepsilon \in P \\ \{q_F\} & \text{altrimenti} \end{cases}$$

### Da $\mathcal{G}$ a $\mathcal{A}_N$

Per ogni coppia  $a \in V_T$  e  $B \in V_N$ ,

$$\delta_N(q_B, a) = \begin{cases} \{q_C \mid B \rightarrow aC \in P\} \cup \{q_F\} & \text{se } B \rightarrow a \in P \\ \{q_C \mid B \rightarrow aC \in P\} & \text{altrimenti.} \end{cases}$$

L'automa è, in generale, non deterministico.

### Da $\mathcal{G}$ a $\mathcal{A}_N$ . Equivalenza di $\mathcal{G}$ e $\mathcal{A}_N$

Per dimostrare l'equivalenza tra  $\mathcal{G}$  e  $\mathcal{A}_N$ , dobbiamo mostrare che per ogni  $x \in \Sigma^*$  si ha che

$$S \xRightarrow[\mathcal{G}]{} x \quad \text{se e solo se} \quad \bar{\delta}_N(q_S, x) \cap F \neq \emptyset$$

Questo è chiaramente vero se  $x = \varepsilon$ , in quanto  $\bar{\delta}_N(q_0, \varepsilon) = q_0 \in F$ , se e solo se  $S \rightarrow \varepsilon \in P$ , per costruzione.

Nel caso  $x \in \Sigma^+$  mostriamo, per induzione sulla lunghezza di  $x$ , la proprietà più generale

$$S \xRightarrow{} xZ \quad \text{se e solo se} \quad q_Z \in \bar{\delta}_N(q_S, x)$$

### Da $\mathcal{G}$ a $\mathcal{A}_N$ . Equivalenza di $\mathcal{G}$ e $\mathcal{A}_N$

Iniziamo da

$$S \xRightarrow{} xZ \quad \text{implica} \quad q_Z \in \bar{\delta}_N(q_S, x)$$

Passo base:  $|x| = 1$ , per cui  $x = a$ , con  $a \in \Sigma$ . Allora abbiamo che  $S \Rightarrow aZ$  se e solo se  $S \rightarrow aZ \in P$  e quindi se e solo se, per costruzione dell'automa,  $q_Z \in \delta_N(q_S, a)$ .

**Da  $\mathcal{G}$  a  $\mathcal{A}_N$ . Equivalenza di  $\mathcal{G}$  e  $\mathcal{A}_N$**

Passo induttivo:  $|x| > 1$ , per cui  $x = ya$ , con  $|y| = n \geq 1$  e  $a \in \Sigma$ .

Per l'ipotesi induttiva il risultato si assume valido per  $y$ , quindi

$$S \Rightarrow yZ \quad \text{se e solo se} \quad q_Z \in \bar{\delta}_N(q_S, y)$$

Osserviamo che  $S \Rightarrow xZ'$  se e solo se esiste  $Z \in V_N$  tale che  $S \Rightarrow yZ \Rightarrow yaZ' = xZ'$ . Ne deriva che

- $q_Z \in \bar{\delta}_N(q_S, y)$  per induzione
- $Z \rightarrow aZ' \in P$ , e quindi  $q_{Z'} \in \delta_N(a, Z)$  per costruzione

Quindi,  $q_{Z'} \in \bar{\delta}_N(q_S, ya) = \bar{\delta}_N(q_S, x)$

**Da  $\mathcal{G}$  a  $\mathcal{A}_N$ . Equivalenza di  $\mathcal{G}$  e  $\mathcal{A}_N$**

Abbiamo verificato che  $S \Rightarrow xZ$  se e solo se  $q_Z \in \bar{\delta}_N(q_S, x)$ .

Osserviamo ora che  $S \Rightarrow x$  se e solo se esistono  $Z \in V_N$ ,  $y \in \Sigma^*$  e  $Z \rightarrow a \in P$  tali che  $x = ya$  e  $S \Rightarrow yZ \Rightarrow ya = x$ .

Da quanto visto sopra, ciò è vero se e solo se  $q_Z \in \bar{\delta}_N(q_S, y)$  e  $q_F \in \delta_N(q_Z, a)$ , e quindi se e solo se  $q_F \in \bar{\delta}_N(q_S, ya) = \bar{\delta}_N(q_S, x)$ .

In conclusione, per ogni linguaggio regolare (generato da una grammatica di tipo 3) esiste un ASFND che lo accetta (e quindi anche un ASFD che lo decide).

**ASF e grammatiche di tipo 3**

Sia  $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$  un ASFD.

Definiamo una procedura che partendo da  $\mathcal{A}$  produca una grammatica di tipo 3  $\mathcal{G} = \langle V_T, V_N, P, S \rangle$  equivalente (che genera tutte e sole stringhe accettate da  $\mathcal{A}$ ).

**Da  $\mathcal{A}$  a  $\mathcal{G}$**

Se  $q_0 \notin F$ :

$$V_T = \Sigma$$

$$V_N = \{A_i \mid \text{per ogni } q_i \in Q\}$$

$$S = A_0$$

per ogni regola di transizione  $\delta(q_i, a) = q_j$  esiste  $A_i \rightarrow aA_j \in P$ , e se  $q_j \in F$  esiste anche  $A_i \rightarrow a \in P$

**Da  $\mathcal{A}$  a  $\mathcal{G}$**

Se  $q_0 \in F$ :

$$V_T = \Sigma$$

$$V_N = \{A_i \mid \text{per ogni } q_i \in Q\} \cup \{A'_0\}$$

$$S = A'_0$$

per ogni regola di transizione  $\delta(q_i, a) = q_j$  esiste  $A_i \rightarrow aA_j \in P$ , e se  $q_j \in F$  esiste anche  $A_i \rightarrow a \in P$  (tutte le precedenti). Inoltre, per ogni  $\delta(q_0, a) = q_j$  esiste  $A'_0 \rightarrow aA_j \in P$ , e se  $q_j \in F$  esiste anche  $A'_0 \rightarrow a \in P$  ( $A'_0$  ha tutte le produzioni di  $A_0$ ), infine, esiste  $A'_0 \rightarrow \varepsilon \in P$ .

**Da  $\mathcal{A}$  a  $\mathcal{G}$ . Equivalenza di  $\mathcal{G}$  e  $\mathcal{A}$**

Come prima, per dimostrare l'equivalenza tra  $\mathcal{G}$  e  $\mathcal{A}_N$ , dobbiamo mostrare che per ogni  $x \in \Sigma^*$  si ha che

$$\bar{\delta}(q_0, x) \in F \quad \text{se e solo se} \quad S \xRightarrow[\mathcal{G}]{*} x$$

Questo è chiaramente vero se  $x = \varepsilon$ , in quanto in tal caso necessariamente  $q_0 \in F$  e, per costruzione, l'assioma di  $\mathcal{G}$  è  $A'_0$  e  $A'_0 \rightarrow \varepsilon \in P$ .

Nel caso  $x \in \Sigma^+$  mostriamo, per induzione sulla lunghezza di  $x$ , entrambe le proprietà

$$A_i \xRightarrow{*} xA_j \quad \text{se e solo se} \quad \bar{\delta}(q_i, x) = q_j$$

$$A_i \xRightarrow{*} x \quad \text{se e solo se} \quad \bar{\delta}(q_i, x) \in F$$

### Da $\mathcal{A}$ a $\mathcal{G}$ . Equivalenza di $\mathcal{G}$ e $\mathcal{A}$

Passo base:  $|x| = 1$ , ad esempio  $x = a$ . Abbiamo allora che

Per costruzione,  $A_i \rightarrow aA_j \in P$  (e quindi  $A_i \Rightarrow aA_j$ ) se e solo se  $\delta(q_i, a) = q_j$  (e quindi  $\bar{\delta}(q_i, a) = q_j$ )

e inoltre che, per costruzione,

$A_i \rightarrow a \in P$  (e quindi  $A_i \Rightarrow a$ ) se e solo se  $q_j \in F$

### Da $\mathcal{A}$ a $\mathcal{G}$ . Equivalenza di $\mathcal{G}$ e $\mathcal{A}$

Passo induttivo:  $|x| = n > 1$ .

Sia  $x = ya$ , con  $|y| = n - 1$ : per l'ipotesi induttiva, la proprietà è valida per  $y$ , e quindi

$$A_i \Rightarrow^* yA_k \quad \text{se e solo se} \quad \bar{\delta}(q_i, y) = q_k$$

Supponiamo  $A_i \Rightarrow^* xA_j = yaA_j$ : ciò è possibile se e solo se esiste  $A_k$  tale che  $A_i \Rightarrow^* yA_k$  e  $A_k \rightarrow aA_j \in P$

### Da $\mathcal{A}$ a $\mathcal{G}$ . Equivalenza di $\mathcal{G}$ e $\mathcal{A}$

Per l'ipotesi induttiva,  $A_i \Rightarrow^* yA_k$  se e solo se  $\bar{\delta}(q_i, y) = q_k$ .

Per costruzione,  $A_k \rightarrow aA_j \in P$  se e solo se  $\delta(q_k, a) = q_j$ .

Ne consegue che

$$A_i \Rightarrow^* yA_k \Rightarrow^* yaA_j = xA_j$$

se e solo se

$$q_j = \delta(q_k, a) = \delta(\bar{\delta}(q_i, y), a) = \bar{\delta}(q_i, ya) = \bar{\delta}(q_i, x)$$

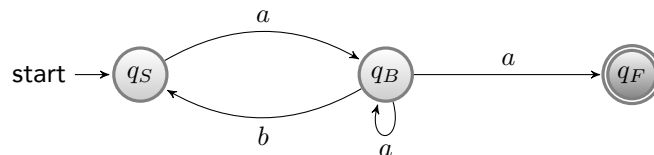
### Esempio

Il linguaggio rappresentato da  $a(a + ba)^*a$  è generato dalla grammatica

$$S \rightarrow aB$$

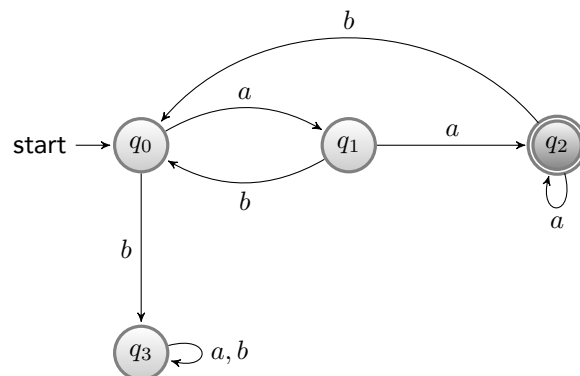
$$B \rightarrow aB \mid bS \mid a.$$

ed è riconosciuto dall'ASFND



### Esempio

A partire dall'ASFND è possibile derivare un ASFD equivalente



### Esempio

E da questo una grammatica di tipo 3 equivalente a quella iniziale, dove  $S = A_0$

$$A_0 \rightarrow aA_1$$

$$A_1 \rightarrow bA_0 \mid aA_2 \mid a$$

$$A_2 \rightarrow aA_2 \mid bA_0 \mid a$$

Per costruzione, questa grammatica ha, per ogni coppia  $X \in V_N$  e  $c \in V_T$ , al più un  $Y \in V_N$  tale che  $X \rightarrow cY \in P$ .

### Esercizio

Si consideri la grammatica regolare avente le seguenti produzioni:

$$S \rightarrow 0A \mid 1B \mid 0S$$

$$A \rightarrow aB \mid bA \mid a$$

$$B \rightarrow bA \mid aB \mid b.$$

Si derivino un ASFND e un ASFD che riconoscono il linguaggio generato da tale grammatica. A partire dall'automa deterministico, derivare poi una grammatica di tipo 3 equivalente.

## 2 Espressioni regolari

### Espressioni regolari

**Teorema 1.** *Tutti i linguaggi definiti da espressioni regolari sono regolari.*

#### Espressioni regolari e grammatiche di tipo 3

**Teorema 2.** *Data una grammatica  $\mathcal{G}$  di tipo 3, esiste una espressione regolare  $r$  tale che  $L(\mathcal{G}) = \mathcal{L}(r)$ , che descrive cioè il linguaggio generato da  $\mathcal{G}$ .*

#### Espressioni regolari e grammatiche di tipo 3

Consideriamo una grammatica  $\mathcal{G}$  di tipo 3 ed il linguaggio  $L$  da essa generato, che per semplicità assumiamo non contenga la stringa vuota  $\varepsilon$ .

Se così non fosse, applichiamo le considerazioni seguenti al linguaggio  $L - \{\varepsilon\}$ , anch'esso regolare: una volta derivata un'espressione regolare  $r$  che lo definisce, l'espressione regolare che definisce  $L$  sarà chiaramente  $r + \varepsilon$ .

#### Espressioni regolari e grammatiche di tipo 3

Alla grammatica  $\mathcal{G}$  possiamo far corrispondere un sistema di equazioni su espressioni regolari.

Estensione del linguaggio delle espressioni regolari con variabili  $A, \dots, Z$ , associando una variabile ad ogni non terminale in  $\mathcal{G}$ .

Tali variabili potranno assumere valori nell'insieme delle espressioni regolari.

#### Espressioni regolari e grammatiche di tipo 3

Raggruppamento di tutte le produzioni che presentano a sinistra lo stesso non terminale. Per ogni produzione del tipo

$$A \rightarrow a_1B_1 \mid a_2B_2 \mid \dots \mid a_nB_n \mid b_1 \mid \dots \mid b_m$$

equazione del tipo

$$A = a_1B_1 + a_2B_2 + \dots + a_nB_n + b_1 + \dots + b_m.$$

#### Espressioni regolari e grammatiche di tipo 3

Da una grammatica regolare si ottiene un sistema di *equazioni lineari destre*, in cui ogni monomio contiene una variabile a destra di simboli terminali.

### Espressioni regolari e grammatiche di tipo 3

Risoluzione del sistema di equazioni su espressioni regolari estese:

individuazione dei valori (espressioni regolari normali, prive delle variabili che definiscono a loro volta espressioni regolari) che, una volta sostituiti alle variabili, soddisfano il sistema di equazioni.

### Espressioni regolari e grammatiche di tipo 3

$$A \longrightarrow aA \mid bB$$

$$B \longrightarrow bB \mid c$$

corrisponde al sistema di equazioni

$$\begin{cases} A = aA + bB \\ B = bB + c. \end{cases}$$

### Espressioni regolari e grammatiche di tipo 3

Per risolvere il sistema è possibile utilizzare, le trasformazioni algebriche applicabili sulle operazioni di unione e concatenazione (distributività, fattorizzazione, ecc.), oltre alle seguenti due regole.

### Espressioni regolari e grammatiche di tipo 3

Sostituzione di una variabile con un'espressione regolare estesa.

Con riferimento all'esempio precedente abbiamo

$$\begin{cases} A = aA + b(bB + c) = aA + bbB + bc \\ B = bB + c. \end{cases}$$

### Espressioni regolari e grammatiche di tipo 3

Eliminazione della ricursione.

L'equazione  $B = bB + c$  si risolve in  $B = b^*c$ . Infatti, sostituendo a destra e sinistra abbiamo

$$b^*c = b(b^*c) + c = b^+c + c = (b^+ + \varepsilon)c = b^*c.$$

Più in generale abbiamo che un'equazione del tipo

$$A = \alpha_1 A + \alpha_2 A + \dots + \alpha_n A + \beta_1 + \beta_2 + \dots + \beta_m$$

si risolve in

$$A = (\alpha_1 + \alpha_2 + \dots + \alpha_n)^*(\beta_1 + \beta_2 + \dots + \beta_m),$$

dove  $\alpha_1, \dots, \alpha_n, \beta_1, \dots, \beta_m$  sono espressioni regolari estese.

### Espressioni regolari e grammatiche di tipo 3

Grammatica regolare

$$A'_0 \longrightarrow \varepsilon \mid aA_1 \mid a$$

$$A_0 \longrightarrow aA_1 \mid a$$

$$A_1 \longrightarrow bA_3 \mid bA_2$$

$$A_2 \longrightarrow aA_2 \mid bA_0 \mid b$$

$$A_3 \longrightarrow bA_3 \mid aA_2.$$

da cui si ottiene il seguente sistema lineare.

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = aA_1 + a \\ A_1 = bA_3 + bA_2 \\ A_2 = aA_2 + bA_0 + b \\ A_3 = bA_3 + aA_2 \end{cases}$$

Da cui si ottiene:

### Espressioni regolari e grammatiche di tipo 3

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = aA_1 + a \\ A_1 = bA_3 + bA_2 \\ A_2 = aA_2 + bA_0 + b \\ A_3 = b^*aA_2 \end{cases}$$

per eliminazione della ricursione su  $A_3$

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = aA_1 + a \\ A_1 = bA_3 + bA_2 \\ A_2 = a^*(bA_0 + b) \\ A_3 = b^*aA_2 \end{cases}$$

per eliminazione della ricursione su  $A_2$ .

### Espressioni regolari e grammatiche di tipo 3

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = aA_1 + a \\ A_1 = bA_3 + bA_2 \\ A_2 = a^*(bA_0 + b) \\ A_3 = b^*aa^*(bA_0 + b) \end{cases}$$

per sostituzione di  $A_2$  nell'equazione relativa ad  $A_3$

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = aA_1 + a \\ A_1 = b(b^*aa^*(bA_0 + b)) + b(a^*(bA_0 + b)) \\ A_2 = a^*(bA_0 + b) \\ A_3 = b^*aa^*(bA_0 + b) \end{cases}$$

per sostituzione di  $A_2$  e  $A_3$  nell'equazione relativa ad  $A_1$ .

### Espressioni regolari e grammatiche di tipo 3

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = aA_1 + a \\ A_1 = b(b^*aa^* + a^*)(bA_0 + b) \\ A_2 = a^*(bA_0 + b) \\ A_3 = b^*aa^*(bA_0 + b) \end{cases}$$

per fattorizzazione nell'equazione relativa ad  $A_1$ .

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = a(b(b^*aa^* + a^*)(bA_0 + b)) + a \\ A_1 = b(b^*aa^* + a^*)(bA_0 + b) \\ A_2 = a^*(bA_0 + b) \\ A_3 = b^*aa^*(bA_0 + b) \end{cases}$$

per sostituzione di  $A_1$  nell'equazione relativa ad  $A_0$

### Espressioni regolari e grammatiche di tipo 3

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = ab(b^*aa^* + a^*)bA_0 + ab(b^*aa^* + a^*)b + a \\ A_1 = b(b^*aa^* + a^*)(bA_0 + b) \\ A_2 = a^*(bA_0 + b) \\ A_3 = b^*aa^*(bA_0 + b) \end{cases}$$

per fattorizzazione nell'equazione relativa ad  $A_0$

$$\begin{cases} A'_0 = \varepsilon + aA_1 + a \\ A_0 = (ab(b^*aa^* + a^*)b)^*(ab(b^*aa^* + a^*)b + a) \\ A_1 = b(b^*aa^* + a^*)(bA_0 + b) \\ A_2 = a^*(bA_0 + b) \\ A_3 = b^*aa^*(bA_0 + b) \end{cases}$$

per eliminazione della ricursione su  $A_0$ .

### Espressioni regolari e grammatiche di tipo 3

$$\begin{cases} A'_0 &= a(b(b^*aa^* + a^*)(bA_0 + b)) + a + \varepsilon \\ A_0 &= (ab(b^*aa^* + a^*)b)^*(ab(b^*aa^* + a^*)b + a) \\ A_1 &= b(b^*aa^* + a^*)(bA_0 + b) \\ A_2 &= a^*(bA_0 + b) \\ A_3 &= b^*aa^*(bA_0 + b) \end{cases}$$

per sostituzione di  $A_1$  nell'equazione relativa ad  $A'_0$

### Espressioni regolari e grammatiche di tipo 3

$$\begin{cases} A'_0 &= ab(b^*aa^* + a^*)(b((ab(b^*aa^* + a^*)b)^*(ab(b^*aa^* + a^*)b + a)) \\ &\quad + b) + a + \varepsilon \\ A_0 &= (ab(b^*aa^* + a^*)b)^*(ab(b^*aa^* + a^*)b + a) \\ A_1 &= b(b^*aa^* + a^*)(bA_0 + b) \\ A_2 &= a^*(bA_0 + b) \\ A_3 &= b^*aa^*(bA_0 + b) \end{cases}$$

per sostituzione di  $A_0$  nell'equazione relativa ad  $A'_0$

### Espressioni regolari e grammatiche di tipo 3

$$\begin{cases} A'_0 &= ((ab(b^*a^* + \varepsilon)a^*b)^* + \varepsilon)(ab(b^*a + \varepsilon)a^*b + a) + \varepsilon \\ A_0 &= (ab(b^*aa^* + a^*)b)^*(ab(b^*aa^* + a^*)b + a) \\ A_1 &= b(b^*aa^* + a^*)(bA_0 + b) \\ A_2 &= a^*(bA_0 + b) \\ A_3 &= b^*aa^*(bA_0 + b) \end{cases}$$

per fattorizzazione nell'equazione relativa ad  $A'_0$

### Espressioni regolari e grammatiche di tipo 3

$$S \longrightarrow aS \mid bA \mid \varepsilon$$

$$A \longrightarrow aA \mid bS \mid \varepsilon$$

### Espressioni regolari e grammatiche di tipo 3

Eliminazione della produzione  $A \longrightarrow \varepsilon$ :

$$S \longrightarrow aS \mid bA \mid \varepsilon \mid b$$

$$A \longrightarrow aA \mid bS \mid a.$$

### Espressioni regolari e grammatiche di tipo 3

$$S = aS + bA + b + \varepsilon$$

$$A = aA + bS + a$$

### Espressioni regolari e grammatiche di tipo 3

$$S = aS + bA + b + \varepsilon$$

$$A = a^*(bS + a)$$

### Espressioni regolari e grammatiche di tipo 3

$$S = aS + ba^*(bS + a) + b + \varepsilon$$

$$A = a^*(bS + a)$$

### Espressioni regolari e grammatiche di tipo 3

$$S = (a + ba^*b)S + ba^*a + b + \varepsilon$$

$$A = a^*(bS + a)$$

### Espressioni regolari e grammatiche di tipo 3

$$S = (a + ba^*b)^*(ba^*a + b + \varepsilon)$$

$$A = a^*(bS + a)$$

### Esercizio

Si consideri la seguente grammatica:

$$A \longrightarrow aB \mid bC \mid a$$

$$B \longrightarrow aA \mid bD \mid b$$

$$C \longrightarrow ab \mid aD \mid a$$

$$D \longrightarrow aC \mid bB \mid b$$

che genera le stringhe contenenti un numero dispari di  $a$  o un numero dispari di  $b$ .

Si costruisca l'espressione regolare corrispondente.

### Espressioni regolari e ASF

Dato un ASF  $\mathcal{A}$ , esiste una espressione regolare  $r$  tale che  $L(\mathcal{A}) = \mathcal{L}(r)$ , che descrive cioè il linguaggio riconosciuto da  $\mathcal{A}$ .

### Espressioni regolari e ASF

Sia  $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$  un ASF e sia  $L$  il linguaggio da esso riconosciuto. Assumiamo  $F = \{q_F\}$ .

Sia  $n = |Q|$  e sia  $\langle q_0, \dots, q_{n-1} \rangle$  un qualunque ordinamento degli stati tale che  $q_{n-1} = q_F$ .

### Espressioni regolari e ASF

Definiamo ora come

$$R_{ij}^k \quad 0 \leq i, j \leq n-1; k \geq \max(i, j)$$

l'insieme delle stringhe tali da portare  $\mathcal{A}$  da  $q_i$  a  $q_j$  senza transitare per nessuno stato  $q_h$  con  $h \geq k$ .

Abbiamo cioè che  $x = a_1, \dots, a_m \in R_{ij}^k$  se e solo se:

1.  $\bar{\delta}(q_i, x) = q_j$ ;
2. se  $\bar{\delta}(q_i, a_1 \dots a_l) = q_{i_l}$  allora  $i_l < k$ , per  $1 \leq l \leq m-1$ .

### Espressioni regolari e ASF

Per  $k = 1$  si ha:

$$R_{ij}^1 = \begin{cases} \bigcup \{a\} & \text{tali che } \delta(q_i, a) = q_j, \text{ se ne esiste almeno uno;} \\ \emptyset & \text{altrimenti.} \end{cases}$$

### Espressioni regolari e ASF

Per  $k > 1$ , se  $x \in R_{ij}^{k+1}$  è una stringa che conduce da  $q_i$  a  $q_j$  senza transitare per nessuno stato  $q_h$  con  $h \geq k+1$ , possono verificarsi due casi:

1.  $x$  conduce da  $q_i$  a  $q_j$  senza transitare per  $q_k$ , dal che deriva che  $x \in R_{ij}^k$ .
2.  $x$  conduce da  $q_i$  a  $q_j$  transitando per  $q_k$

### Espressioni regolari e ASF

Nel secondo caso la sequenza degli stati attraversati può essere divisa in varie sottosequenze:

1. una prima sequenza, da  $q_i$  a  $q_k$  senza transitare per nessuno stato  $q_h$  con  $h > k$ , la corrispondente sottostringa di  $x$  appartiene quindi a  $R_{ik}^k$ ;
2.  $r \geq 0$  sequenze, ognuna delle quali inizia e termina in  $q_k$  senza transitare per nessuno stato  $q_h$  con  $h \geq k$ , le corrispondenti sottostringhe di  $x$  appartengono quindi ciascuna a  $R_{kk}^k$ ;
3. una sequenza finale, da  $q_k$  a  $q_j$  senza transitare per nessuno stato  $q_h$  con  $h \geq k$ , la corrispondente sottostringa di  $x$  appartiene quindi a  $R_{kj}^k$ .

In conseguenza, ne deriva la relazione

$$R_{ij}^{k+1} = R_{ij}^k \cup R_{ik}^k \circ (R_{kk}^k)^* \circ R_{kj}^k$$



### Espressioni regolari e ASF

Dalle osservazioni precedenti deriva che è possibile costruire tutti gli insiemi  $R_{ij}^k$  a partire da  $k = 1$  e derivando poi man mano i successivi.

Osserviamo anche che  $L = R_{0(n-1)}^n$

### Espressioni regolari e ASF

Ogni insieme di stringhe  $R_{ij}^k$  può essere descritto per mezzo di una opportuna espressione regolare  $r_{ij}^k$ , infatti abbiamo che, per  $k = 1$ ,

$$r_{ij}^1 = \begin{cases} a_{i_1} + \dots + a_{i_l} & \text{dove } \delta(q_i, a_{i_k}) = q_j, k = 1, \dots, l; \\ \varepsilon & \text{se } l = 0. \end{cases}$$

### Espressioni regolari e ASF

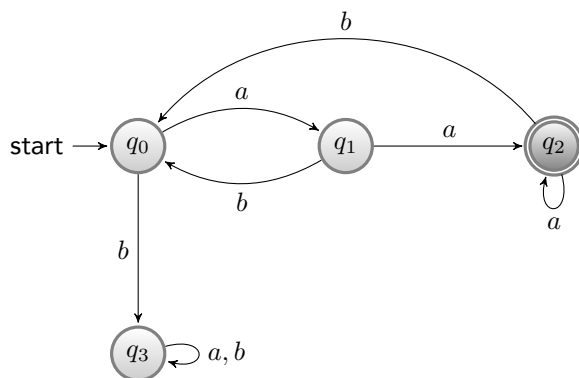
Per  $k > 1$ , abbiamo che, dalla relazione tra  $R_{ij}^{k+1}$ ,  $R_{ik}^k$ ,  $R_{kk}^k$  e  $R_{kj}^k$ , deriva che

$$r_{ij}^{k+1} = r_{ij}^k + r_{ik}^k (r_{kk}^k)^* r_{kj}^k$$

Quindi, il linguaggio  $L$  sarà descritto dall'espressione regolare

$$r_{0(n-1)}^n$$

### Espressioni regolari e ASF



### Espressioni regolari e ASF

Assumiamo l'ordinamento  $q_1 = q_0, q_2 = q_1, q_3 = q_3, q_4 = q_2$  tra gli stati. Allora:

$$\begin{aligned} r_{00}^0 &= \emptyset; r_{01}^0 = a; r_{02}^0 = b; r_{03}^0 = \emptyset; \\ r_{10}^0 &= b; r_{11}^0 = \emptyset; r_{12}^0 = \emptyset; r_{13}^0 = a; \\ r_{20}^0 &= \emptyset; r_{21}^0 = \emptyset; r_{22}^0 = a + b; r_{23}^0 = \emptyset; \\ r_{30}^0 &= b; r_{31}^0 = \emptyset; r_{32}^0 = \emptyset; r_{33}^0 = a; \end{aligned}$$

### Espressioni regolari e ASF

$$\begin{aligned} r_{00}^1 &= r_{00}^0 (r_{00}^0)^* r_{00}^0 = \emptyset (\emptyset)^* \emptyset = \emptyset \\ r_{01}^1 &= r_{00}^0 (r_{00}^0)^* r_{01}^0 = \emptyset (\emptyset)^* a = a \\ r_{02}^1 &= r_{00}^0 (r_{00}^0)^* r_{02}^0 = \emptyset (\emptyset)^* b = b \\ r_{03}^1 &= r_{00}^0 (r_{00}^0)^* r_{03}^0 = \emptyset (\emptyset)^* \emptyset = \emptyset \\ r_{10}^1 &= r_{10}^0 (r_{00}^0)^* r_{00}^0 = b (\emptyset)^* \emptyset = b \\ r_{11}^1 &= r_{10}^0 (r_{00}^0)^* r_{01}^0 = b (\emptyset)^* a = ba \\ r_{12}^1 &= r_{10}^0 (r_{00}^0)^* r_{02}^0 = b (\emptyset)^* b = bb \\ r_{13}^1 &= r_{10}^0 (r_{00}^0)^* r_{03}^0 = b (\emptyset)^* \emptyset = b \\ &\dots \end{aligned}$$

## Espressioni regolari e ASF

$$\begin{aligned} r_{00}^1 &= \emptyset; r_{01}^1 = a; r_{02}^1 = b; r_{03}^1 = \emptyset; \\ r_{10}^1 &= b; r_{11}^1 = ba; r_{12}^1 = bb; r_{13}^1 = b; \\ r_{20}^1 &= \emptyset; r_{21}^1 = \emptyset; r_{22}^1 = a + b; r_{23}^1 = \emptyset; \\ r_{30}^1 &= b; r_{31}^1 = ba; r_{32}^1 = bb; r_{33}^1 = a; \end{aligned}$$

## Espressioni regolari e ASF

$$\begin{aligned} r_{00}^2 &= a(ba)^*b; r_{01}^2 = a + a(ba)^*ba; r_{02}^2 = b + a(ba)^*bb; r_{03}^2 = a(ba)^*a; \\ r_{10}^2 &= b + ba(ba)^*b; r_{11}^2 = ba + ba(ba)^*ba; r_{12}^2 = bb + ba(ba)^*bb; r_{13}^2 = a + ba(ba)^*a; \\ r_{20}^2 &= \emptyset; r_{21}^2 = \emptyset; r_{22}^2 = a + b; r_{23}^2 = \emptyset; \\ r_{30}^2 &= b + ba(ba)^*b; r_{31}^2 = ba + ba(ba)^*ba; r_{32}^2 = bb + ba(ba)^*bb; r_{33}^2 = a + ba(ba)^*a; \end{aligned}$$

Il linguaggio accettato dall'automa sarà descritto dall'espressione regolare

$$r_{03}^4$$

## State elimination

Procedura iterativa di eliminazione degli stati su un automa non deterministico *generalizzato* equivalente, in cui:

1. la funzione di transizione è definita su  $Q \times E$ , dove  $E$  è l'insieme delle espressioni regolari su  $\Sigma$ , per cui gli archi sono etichettati con e.r.
2. lo stato iniziale non ha archi entranti, per cui  $\nexists q \in Q, e \in E : q_0 \in \delta_N(q, e)$
3. esiste un solo stato finale  $q_F$  senza archi uscenti, per cui  $\nexists e \in E : \delta_N(q_F, e) \neq \emptyset$

## State elimination

Dato un qualunque automa  $\mathcal{A}$  non deterministico, un automa *generalizzato*  $\mathcal{A}'$  equivalente può essere immediatamente ottenuto:

1. mantenendo gli stati di  $\mathcal{A}$
2. introducendo, per ogni arco del grafo di transizione di  $\mathcal{A}$  etichettato con l'insieme  $a_1, \dots, a_k$ , un arco nel grafo di transizione di  $\mathcal{A}'$  etichettato  $a_1 + \dots + a_k$
3. se lo stato iniziale  $q_0$  di  $\mathcal{A}$  ha archi entranti, introducendo in  $\mathcal{A}'$  un nuovo stato iniziale  $\bar{q}_0$  senza archi entranti, e la  $\varepsilon$ -transizione  $\delta'_N(\bar{q}_0, \varepsilon) = \{q_0\}$
4. se esistono più stati finali in  $F$ , o se il solo stato finale ha archi uscenti, introducendo un ulteriore stato  $q_F$ , ponendo  $F' = q_F$  e introducendo la  $\varepsilon$ -transizione  $\delta'_N(q, \varepsilon) = \{q_F\}$  per ogni  $q \in F$

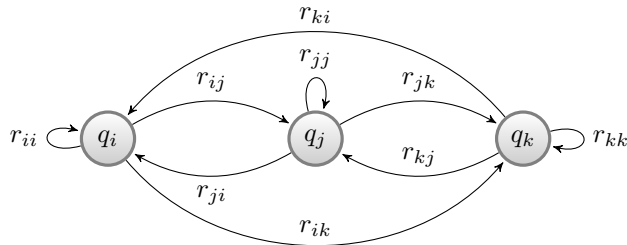
## State elimination

- Dato un automa nondeterministico (con  $\varepsilon$ -transizioni)  $\mathcal{A}$  con insieme di stati  $Q$ , e dato uno stato  $q$  non iniziale né finale, è possibile ottenere un automa generalizzato equivalente  $\mathcal{A}'$  con stati  $Q - \{q\}$  effettuando una opportuna operazione di *eliminazione dello stato*
- L'eliminazione dello stato viene effettuata considerando tutti i possibili cammini di lunghezza 3 passanti per  $q$  (sequenze  $q_i, q, q_j$  per le quali esistono archi da  $q_i$  a  $q$  e da  $q$  a  $q_j$ )

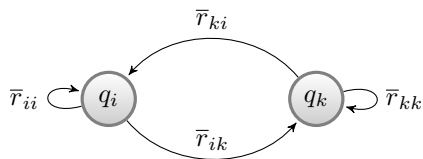
- Per ogni cammino, le etichette degli archi interessati vengono modificate come mostrato di seguito
- Al termine, rimangono lo stato iniziale e quello finale, collegati da un arco, la cui etichetta fornisce l'espressione regolare cercata

### State elimination

Da

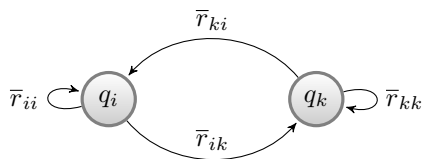


a



### State elimination

Le espressioni regolari risultanti possono comunque essere complesse



$$\bar{r}_{ii} = r_{ij}(r_{jj} + r_{jk}r_{kk}^*r_{kj})^*r_{ji} + r_{ij}(r_{jj} + r_{jk}r_{kk}^*r_{kj})^*r_{jk}r_{kk}^*r_{ki} + r_{ik}(r_{kj}r_{jj}^*r_{jk} + r_{kk})^*r_{kj}r_{jj}^*r_{ji} + r_{ik}r_{ki}$$

$$\bar{r}_{kk} = r_{kj}(r_{jj} + r_{kj}r_{ii}^*r_{ij})^*r_{jk} + r_{kj}(r_{jj} + r_{ji}r_{ii}^*r_{ij})^*r_{ji}r_{ii}^*r_{ik} + r_{ki}(r_{ij}r_{jj}^*r_{ji} + r_{ii})^*r_{ij}r_{jj}^*r_{jk} + r_{ki}r_{ik}$$

$$\bar{r}_{ik} = r_{ik} + r_{ij}r_{jj}^*r_{jk}$$

$$\bar{r}_{ki} = r_{ki} + r_{kj}r_{jj}^*r_{ji}$$

In effetti, se esistono  $n$  cammini  $q_i q_j q_h$  ( $h = k_1, \dots, k_n$ ), allora si ha che

$$\bar{r}_{ik} = r_{ik_1} + r_{ij}r_{jj}^*r_{jk_1} + r_{ik_2} + r_{ij}r_{jj}^*r_{jk_2} + \dots + r_{ik_n} + r_{ij}r_{jj}^*r_{jk_n}$$

lo stesso, evidentemente, vale per  $\bar{r}_{ki}$