Linguaggi di Programmazione (corso A)

Docente: Giovanni Semeraro

Capitolo 7 – Linguaggi regolari, espressioni regolari e teorema di Kleene.

Descrizione algebrica dei linguaggi lineari destri

<table>
<tr><th>Grammatiche generative</th><th>Automi a stati finiti accettori</th><th>Espressioni regolari</th></tr>
<tr><td>Generatori</td><td>Riconoscitori</td><td rowspan="3"></td></tr>
<tr><td colspan="2">procedure effettive per</td></tr>
<tr><td>Costruire stringhe che appartengono al linguaggio</td><td>Stabilire se una stringa appartiene o no ad un linguaggio</td></tr>
<tr><td colspan="2">Definizioni operazionali</td><td>Definizione denotazionale</td></tr>
</table>

Linguaggi regolari ed espressioni regolari

- Definizione di linguaggio regolare

  Sia *X* un alfabeto. Un *linguaggio* $L \subseteq X^*$ è *regolare* se:

  - *L* è finito (| *L* | = *k*, *k* intero)

  oppure

  - *L* può essere ottenuto per induzione (*sulla costruzione*) utilizzando una delle seguenti operazioni:
    - $L = L_1 \cup L_2,\ \text{con}\ L_1, L_2\ \text{regolari};$
    - $L = L_1 \cdot L_2,\ \text{con}\ L_1, L_2\ \text{regolari};$
    - $L = L_1^*,\ \text{con}\ L_1\ \text{regolare}.$

  Si noti che $\varnothing$ e $\{\lambda\}$ sono linguaggi regolari.

  Denotiamo con $\mathcal{L}_{REG}$ la classe dei linguaggi regolari.

Espressioni regolari

- Definizione di espressione regolare
  Sia *X* un alfabeto. Una stringa *R* di alfabeto $X \cup \{ \lambda, +, *, \cdot, \varnothing, (,) \}$ (con $X \cap \{ \lambda, +, *, \cdot, \varnothing, (,) \} = \varnothing$) è una *espressione regolare* di alfabeto *X* se e solo se vale una delle seguenti condizioni:
  - (i) $R = \varnothing$
  - (ii) $R = \lambda$
  - (iii) $R = a,$ per ogni $a \in X$
  - (iv) $R = (R_1 + R_2),$ *con* R_1, R_2 espressioni regolari di alfabeto X
  - (v) $R = (R_1 \cdot R_2),$ *con* R_1, R_2 espressioni regolari di alfabeto X
  - (vi) $R = (R_1)^*,$ *con* R_1 espressione regolare di alfabeto X

Espressioni regolari e linguaggi regolari

- Ad ogni espressione regolare *R* corrisponde un linguaggio regolare ***S*(*R*) definito** nel modo seguente:

Espressione regolare	***Linguaggio regolare corrispondente***
$\varnothing$	$\varnothing$
λ	$\{\lambda\}$
a	$\{a\}$
$(R_1 + R_2)$	$S(R_1) \cup S(R_2)$
$(R_1 \cdot R_2)$	$S(R_1) \cdot S(R_2)$
$(R_1)^*$	$\left(S(R_1)\right)^*$

Espressioni regolari e linguaggi regolari

- Da una espressione regolare si possono eliminare le coppie di parentesi superflue, tenuto conto che le operazioni ai punti (iv), (v) e (vi) sono elencate in ordine crescente di *priorità*.

Proposizione

- Un linguaggio su *X* è regolare se e solo se corrisponde ad una espressione regolare su *X*.

  Quindi, denotato con $\mathfrak{R}$ l'insieme delle espressioni regolari di alfabeto *X*, definiamo la funzione:

$$S : \mathfrak{R} \rightarrow 2^{X^*}$$

  che ad ogni espressione regolare *R* associa il corrispondente linguaggio regolare *S*(*R*). Si ha dunque:

$$\mathcal{L}_{REG} = \left\{ L \in 2^{X^*} \;\middle|\; \exists R \in \mathfrak{R} \,.\, L = S(R) \right\}$$

Esempio

Osservazione

- Un linguaggio regolare può essere descritto da più di una espressione regolare, ossia $S : \Re \rightarrow 2^{X^*}$ non è una funzione iniettiva (come d'altra parte L(*G*) e T(*M*)).

Esempio

- Il linguaggio costituito da tutte le parole su $X = \{a, b\}$ con *a* e *b* che si alternano (e che cominciano e terminano con *b*) può essere descritto sia dall'espressione regolare

$$b \cdot (ab)^*$$

sia da

$$(ba)^* b$$

Definizione di espressioni regolari equivalenti

- Due espressioni regolari R_1 e R_2 su X sono *equivalenti* (per abuso di notazione, scriviamo $R_1 = R_2$) se e solo se $S(R_1) = S(R_2)$.

  Relativamente all'esempio precedente, si ha, pertanto:

$$b \cdot (ab)^* = (ba)^* b$$

Proprietà delle espressioni regolari

- Siano R_1, R_2 ed R_3 espressioni regolari di alfabeto X, risulta:

1. $(R_1 + R_2) + R_3 = R_1 + (R_2 + R_3) = R_1 + R_2 + R_3$ Prop. associativa
2. $R_1 + R_2 = R_2 + R_1$ Prop. commutativa
3. $R_1 + \varnothing = \varnothing + R_1 = R_1$ $\varnothing$ è l'elemento neutro rispetto all'operazione "+" definita sulle espressioni regolari
4. $R_1 + R_1 = R_1$ Idempotenza
5. $(R_1 \cdot R_2) \cdot R_3 = R_1 \cdot (R_2 \cdot R_3) = R_1 \cdot R_2 \cdot R_3$ Prop. associativa
6. $R_1 \cdot R_2 \neq R_2 \cdot R_1$ In generale
7. $R_1 \cdot \lambda = \lambda \cdot R_1 = R_1$ λ è l'elemento neutro rispetto all'operazione "$\cdot$ " definita sulle espressioni regolari

Proprietà delle espressioni regolari

- Siano R_1, R_2 ed R_3 espressioni regolari di alfabeto X, risulta:

8. $R_1 \cdot \varnothing = \varnothing \cdot R_1 = \varnothing$ — $\varnothing$ è l'elemento assorbente rispetto all'operazione "."
9. $R_1 \cdot (R_2 + R_3) = (R_1 \cdot R_2) + (R_1 \cdot R_3)$ — Proprietà distributiva di "·" rispetto all'operazione "+" (distributività sinistra).
10. $(R_1 + R_2) \cdot R_3 = (R_1 \cdot R_3) + (R_2 \cdot R_3)$ — Proprietà distributiva di "·" rispetto all'operazione "+" (distributività destra).
11. $(R_1)^* = (R_1)^* \cdot (R_1)^* = \left((R_1)^*\right)^* = (\lambda + R_1)^*$
12. $(\varnothing)^* = (\lambda)^* = \lambda$

Proprietà delle espressioni regolari

- Siano R_1, R_2 ed R_3 espressioni regolari di alfabeto X, risulta:

13. $(R_1)^* = \lambda + R_1 + R_1^{\,2} + \ldots + R_1^{\,n} + (R_1^{\,n+1} \cdot R_1^{\,*})$

    Caso particolare:

    $(R_1)^* = \lambda + R_1 \cdot R_1^{\,*} = \lambda + R_1^{\,*} \cdot R_1$

14. $(R_1 + R_2)^* = \left(R_1^{\,*} + R_2^{\,*}\right)^* = \left(R_1^{\,*} \cdot R_2^{\,*}\right)^* =$

    $= \left(R_1^{\,*} \cdot R_2\right)^* \cdot R_1^{\,*} = R_1^{\,*} \cdot \left(R_2 \cdot R_1^{\,*}\right)^*$

15. $(R_1 + R_2)^* \neq R_1^{\,*} + R_2^{\,*}$ In generale

16. $R_1^{\,*} \cdot R_1 = R_1 \cdot R_1^{\,*}$

17. $R_1 \cdot (R_2 \cdot R_1)^* = (R_1 \cdot R_2)^* \cdot R_1$

Proprietà delle espressioni regolari

- Siano R_1, R_2 ed R_3 espressioni regolari di alfabeto X, risulta:

18. $\left(R_1^* \cdot R_2\right)^* = \lambda + (R_1 + R_2)^* \cdot R_2$

19. $\left(R_1 \cdot R_2^*\right)^* = \lambda + R_1 \cdot (R_1 + R_2)^*$

20. Supponiamo che: $\lambda \notin S(R_2)$

$$R_1 = R_2 \cdot R_1 + R_3 \quad \text{se e solo se} \quad R_1 = R_2^* \cdot R_3$$

$$R_1 = R_1 \cdot R_2 + R_3 \quad \text{se e solo se} \quad R_1 = R_3 \cdot R_2^*$$

Dimostrazione proprietà espressioni regolari

- Per esercizio
  - Aiuto: le 1) - 5) e le 7) - 14) si dimostrano ricorrendo alla funzione *S*;
  - comunque la maggior parte delle 1) - 20) può essere provata con una tecnica generale, detta *dimostrazione mediante riparsificazione*.

Dimostrazione mediante riparsificazione

- Illustriamo questa tecnica dimostrando la proprietà 17)

$$R_1 \cdot (R_2 \cdot R_1)^* = (R_1 \cdot R_2)^* \cdot R_1$$

Si consideri una qualunque parola:

$$w \in S\left(R_1 \cdot (R_2 \cdot R_1)^*\right),\ \ w = r_1^{\,0}\left(r_2^{\,1} \cdot r_1^{\,1}\right) \cdot \left(r_2^{\,2} \cdot r_1^{\,2}\right) \cdot \ldots \cdot \left(r_2^{\,n} \cdot r_1^{\,n}\right),\ \ n \geq 0$$

ove:

$$r_1^{\,i} \in S(R_1), \quad i = 0,1,2,\ldots,n$$

$$r_2^{\,j} \in S(R_2), \quad j = 0,1,2,\ldots,n$$

Riparsificando w ed utilizzando la proprietà associativa della concatenazione, si ha: $w = \left(r_1^{\,0} \cdot r_2^{\,1}\right) \cdot \left(r_1^{\,1} \cdot r_2^{\,2}\right) \cdot \ldots \cdot \left(r_1^{\,n-1} \cdot r_2^{\,n}\right) \cdot r_1^{\,n}$

dunque: $w \in S\left((R_1 \cdot R_2)^* \cdot R_1\right)$

da cui: $S\left(R_1 \cdot (R_2 \cdot R_1)^*\right) \subseteq S\left((R_1 \cdot R_2)^* \cdot R_1\right)$

In modo analogo si dimostra: $S\left(R_1 \cdot (R_2 \cdot R_1)^*\right) \supseteq S\left((R_1 \cdot R_2)^* \cdot R_1\right)$

Dimostrazione proprietà espressioni regolari

- Un’altra tecnica comune per dimostrare tali proprietà è semplicemente quella di utilizzare proprietà già note.
- Mostreremo ora come si usano le proprietà delle espressioni regolari per provare l’equivalenza di espressioni regolari.

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Dimostrare la seguente equivalenza:

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Dimostrare la seguente equivalenza:

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) =$$

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Dimostrare la seguente equivalenza:

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

$$(b + aa^*b)\lambda + (b + aa^*b) \cdot [(a + ba^*b)^* \cdot (a + ba^*b)] =$$
$$\overset{9)}{=} (b + aa^*b) \cdot [\lambda + (a + ba^*b)^*(a + ba^*b)]$$

9) $R_1 \cdot (R_2 + R_3) = (R_1 \cdot R_2) + (R_1 \cdot R_3)$
$R_1 = (b + aa^*b)$
$R_2 = \lambda$, $R_3 = (a + ba^*b)^* \cdot (a + ba^*b)$

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Dimostrare la seguente equivalenza:

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) =$$
$$\overset{9)}{=} (b + aa^*b) \cdot [\lambda + (a + ba^*b)^*(a + ba^*b)] =$$
$$\overset{13)}{=} (b + aa^*b) \cdot (a + ba^*b)^* =$$

9) $R_1 \cdot (R_2 + R_3) = (R_1 \cdot R_2) + (R_1 \cdot R_3)$

13) $(R_1)^* = \lambda + R_1 \cdot R_1{}^* = \lambda + R_1{}^* \cdot R_1$

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Dimostrare la seguente equivalenza:

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) =$$
$$\overset{9)}{=} (b + aa^*b) \cdot [\lambda + (a + ba^*b)^*(a + ba^*b)] =$$
$$\overset{13)}{=} (b + aa^*b) \cdot (a + ba^*b)^* =$$
$$\overset{10)}{=} (\lambda + aa^*) \cdot b \cdot (a + ba^*b)^* =$$

9) $R_1 \cdot (R_2 + R_3) = (R_1 \cdot R_2) + (R_1 \cdot R_3)$

13) $(R_1)^* = \lambda + R_1 \cdot {R_1}^* = \lambda + {R_1}^* \cdot R_1$

10) $(R_1 + R_2) \cdot R_3 = (R_1 \cdot R_3) + (R_2 \cdot R_3)$

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Dimostrare la seguente equivalenza:

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) =$$
$$\overset{9)}{=} (b + aa^*b) \cdot [\lambda + (a + ba^*b)^*(a + ba^*b)] =$$
$$\overset{13)}{=} (b + aa^*b) \cdot (a + ba^*b)^* =$$
$$\overset{10)}{=} (\lambda + aa^*) \cdot b \cdot (a + ba^*b)^* =$$

9) $R_1 \cdot (R_2 + R_3) = (R_1 \cdot R_2) + (R_1 \cdot R_3)$

13) $(R_1)^* = \lambda + R_1 \cdot {R_1}^* = \lambda + {R_1}^* \cdot R_1$

10) $(R_1 + R_2) \cdot R_3 = (R_1 \cdot R_3) + (R_2 \cdot R_3)$
$R_3 = b$
$R_1 = \lambda, R_2 = aa^*$

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Dimostrare la seguente equivalenza:

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) \overset{9)}{=}$$
$$\overset{9)}{=} (b + aa^*b) \cdot [\lambda + (a + ba^*b)^*(a + ba^*b)] =$$
$$\overset{13)}{=} (b + aa^*b) \cdot (a + ba^*b)^* =$$
$$\overset{10)}{=} (\lambda + aa^*) \cdot b \cdot (a + ba^*b)^* =$$
$$\overset{13)}{=} a^*b \cdot (a + ba^*b)^*$$

9) $R_1 \cdot (R_2 + R_3) = (R_1 \cdot R_2) + (R_1 \cdot R_3)$

13) $(R_1)^* = \lambda + R_1 \cdot {R_1}^* = \lambda + {R_1}^* \cdot R_1$

10) $(R_1 + R_2) \cdot R_3 = (R_1 \cdot R_3) + (R_2 \cdot R_3)$

13) $(R_1)^* = \lambda + R_1 \cdot {R_1}^* = \lambda + {R_1}^* \cdot R_1$

Esercizio: applicazione delle proprietà per semplificare espressioni regolari

- Abbiamo dimostrato l'equivalenza:

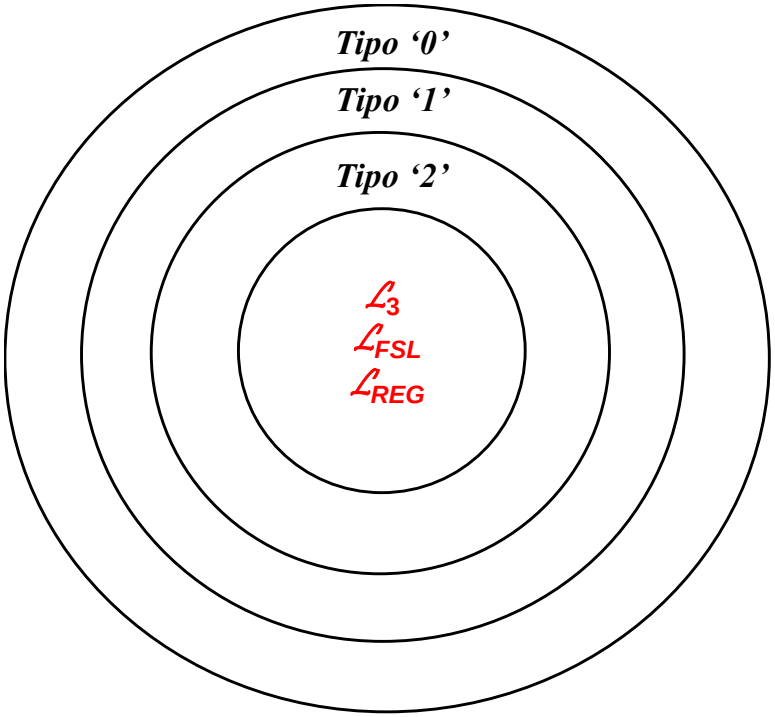
$$(b + aa^*b) + (b + aa^*b) \cdot (a + ba^*b)^* \cdot (a + ba^*b) = a^*b \cdot (a + ba^*b)^*$$

che ha indicato un possibile utilizzo delle proprietà delle espressioni regolari per semplificarle, ossia ridurle a espressioni più semplici

Teorema di Kleene

- $\mathcal{L}_3 \equiv \mathcal{L}_{FSL} \equiv \mathcal{L}_{REG}$

  Equivalenza tra: linguaggi lineari destri, linguaggi a stati finiti, linguaggi regolari



Teorema di Kleene

- $\mathcal{L}_3 \equiv \mathcal{L}_{FSL} \equiv \mathcal{L}_{REG}$

Dimostrazione

Lo schema della dimostrazione è il seguente:

- $\mathcal{L}_3 \subset \mathcal{L}_{FSL}$ $(\mathcal{L}_{FSL} \subset \mathcal{L}_3)$
- $\mathcal{L}_{FSL} \subset \mathcal{L}_{REG}$
- $\mathcal{L}_{REG} \subset \mathcal{L}_3$

Dimostrazione teorema di Kleene

- $\mathcal{L}_3 \subset \mathcal{L}_{FSL}$ *def*
  Sia $L \in \mathcal{L}_3 \Leftrightarrow \exists\, G = (X, V, S, P)$, *G di tipo '3': L(G) = L.*
  Vogliamo costruire un automa a stati finiti
  $M = (Q, \delta, q_0, F)$ tale che $T(M) = L(G)$.

  Allo scopo si fornisce il seguente algoritmo per la costruzione di un automa a stati finiti non deterministico che riconosce il linguaggio generato da una fissata grammatica lineare destra.

Algoritmo: Costruzione di un automa a stati finiti non deterministico equivalente ad una grammatica lineare destra

- Data una grammatica lineare destra:

$$G = (X, V, S, P)$$

l'automa accettore a stati finiti equivalente ($T(M) = L(G)$) viene costruito come segue:

$$M = (Q, \delta, q_0, F)$$

- (I) X come alfabeto di ingresso;
- (II) $Q = V \cup \{q\},\ q \notin V$
- (III) $q_0 = S$
- (IV) $F = \{q\} \cup \{B \mid B \to \lambda \in P\}$
- (V) $\delta : Q \times X \to 2^Q$ э′ $V.a \quad \forall B \to aC \in P,\ C \in \delta(B,a)$

$$V.b \quad \forall B \to a \in P,\ q \in \delta(B,a)$$

Algoritmo: Costruzione di un automa a stati finiti non deterministico equivalente ad una grammatica lineare destra

- L'algoritmo può generare un automa *non deterministico* per effetto dei passi V.a. e V.b. Si può facilmente constatare che, se: $w = x_1 x_2 ... x_n \in L(G)$
  w può essere generata da una derivazione del tipo:
  $S \Rightarrow x_1 X_2 \Rightarrow x_1 x_2 X_3 \Rightarrow ... \Rightarrow x_1 x_2 ... x_{i-1} X_i \Rightarrow x_1 x_2 ... x_i$
  Dalla definizione data, l'automa *M*, esaminando la stringa $w = x_1 x_2 ... x_n$compie una serie di mosse (o transizioni) che lo portano dallo stato *S* ad $X_1, X_2, ..., X_i$ e *q*; pertanto $L(G) \subseteq T(M)$.
  In modo del tutto analogo, ogni *w* in *T*(*M*) comporta una sequenza di mosse dell'automa a cui corrisponde una derivazione in *G*, e pertanto $T(M) \subseteq L(G)$.
  Se ne deduce che: *L(G) = T(M)* *c.v.d.*

Algoritmo: Costruzione di un automa a stati finiti non deterministico equivalente ad una grammatica lineare destra

- Sebbene non sia strettamente necessario per la dimostrazione del Teorema di Kleene, per il suo interesse pratico si riporta di seguito l'algoritmo per la costruzione di una grammatica lineare destra che genera il linguaggio accettato da un automa a stati finiti.
- Tale algoritmo costituisce una dimostrazione costruttiva del seguente risultato:

$$\mathcal{L}_{FSL} \subset \mathcal{L}_3$$

Algoritmo: Costruzione di una grammatica lineare destra equivalente ad un automa accettore a stati finiti

- Sia dato un automa accettore a stati finiti:

$$M = (Q, \delta, q_0, F)$$

con alfabeto di ingresso X.
La grammatica lineare destra G equivalente a M, ossia tale che $L(G) = T(M)$, si costruisce come segue:

- (I) X = alfabeto di ingresso di M
- (II) $V = Q$;
- (III) $S = q_0$;
- (IV) $P = \{q \to xq' \mid q' \in \delta(q,x)\} \cup \{q \to x \mid \delta(q,x) \in F\} \cup \{q_0 \to \lambda \mid q_0 \in F\}$

Pumping Lemma per i linguaggi regolari

- Sia $M = (Q, \delta, q_0, F)$ un automa accettore a stati finiti con n stati ($|Q|=n$) e sia $z \in T(M)$, $|z| \geq n$.
  Allora z può essere scritta come uvw, e $uv^*w \subset T(M)$ (ossia $\forall i,\ i \geq 0 : uv^i w \in T(M)$).
- Una formulazione alternativa è la seguente:

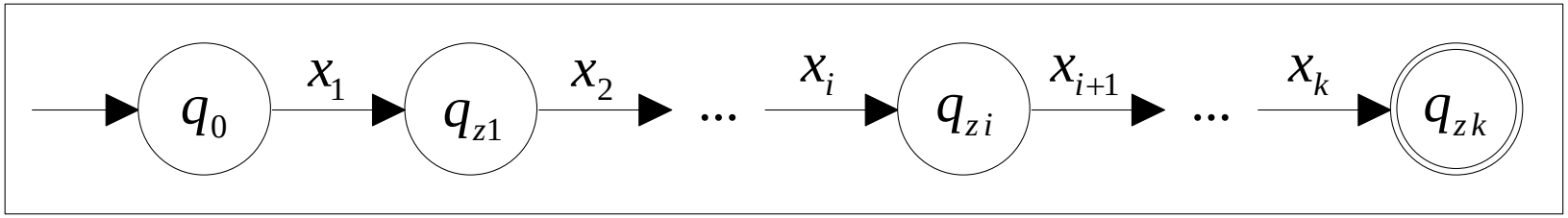
  Sia $L = T(M)$ un linguaggio regolare con $M = (Q, \delta, q_0, F)$ un automa accettore a stati finiti.
  Allora $\exists n = |Q|$ t.c. $\forall z \in L,\ |z| \geq n : z = uvw$ e:
  - $|uv| \leq n$
  - $v \neq \lambda$
  - $uv^i w \in L,\ \forall i,\ i \geq 0$

Pumping Lemma per i linguaggi regolari

- Dimostrazione

  Sia $z = x_1 x_2 \ldots x_k$, $z \in T(M)$.

  Possiamo rappresentare il comportamento dell'automa M, con ingresso z, come segue:



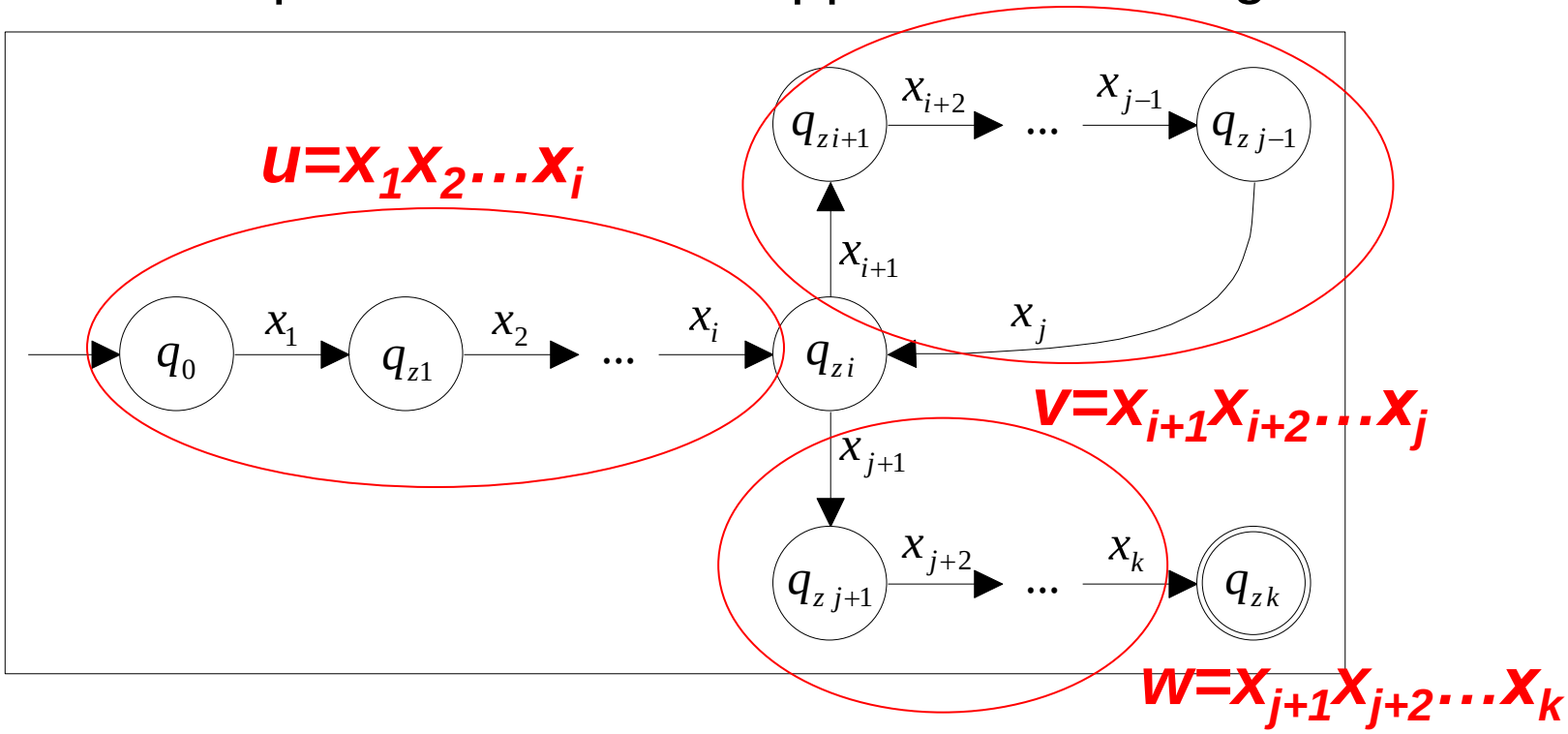
  Se si ha: $|z| \geq n$, nella figura devono comparire almeno n+1 stati ma, poiché M ha solo n stati distinti ($|Q| = n$) almeno una stato tra q_0, q_{z1}, q_{z2}, …, q_{zk} deve comparire due volte.

  Supponiamo che si abbia $q_{zi} = q_{zj}$, $i < j$.

Pumping Lemma per i linguaggi regolari

- Dimostrazione

  Si ha dunque la situazione rappresentata in figura:



Possiamo scrivere *z* nella forma: *z* = *uvw*

Pumping Lemma per i linguaggi regolari

- Dimostrazione

  Poiché $z \in T(M)$, l'automa M, per effetto dell'ingresso di $z = uvw$, si porta per definizione in uno stato finale ($\delta^*(q_0, z) \in F$).

  Ma è immediato osservare che tale stato è lo stesso in cui M si porta per effetto dell'ingresso delle parole uw, uv^2w,… uv^iw, $i \geq 0$.

  Dunque si ha: $uv^iw \in T(M)$, $i \geq 0$. *c.v.d.*

Proprietà decidibili dei linguaggi regolari

- Dato $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X:

- T(M) = ∅ è $decidibile$
- Dimostrazione

  Provo per tutte le parole $w \in X^*$: $|w| < |Q|$.

- T(M) = ∞ è $decidibile$
- Dimostrazione

  Provo per tutte le parole $w \in X^*$: $|Q| \leq |w| < 2^*|Q|$.

Proprietà decidibili dei linguaggi regolari

- Dati $M_1 = (Q', \delta', q'_0, F')$ e $M_2 = (Q'', \delta'', q''_0, F'')$ con alfabeto di ingresso X:

- $T(M_1) = T(M_2)$ è $decidibile$
- Dimostrazione

  Discende dalla prima proprietà decidibile in quanto

  $T(M_1) \setminus T(M_2) = \emptyset$ è $decidibile$

  $T(M_2) \setminus T(M_1) = \emptyset$ è $decidibile$

  e la classe dei linguaggi regolari è chiusa rispetto alle operazioni di intersezione e complemento.

Esercizi

Esercizi

Riferimenti

- Semeraro, G., Elementi di Teoria dei Linguaggi Formali, ilmiolibro.it, 2017 (http://ilmiolibro.kataweb.it/libro/informatica-e-internet/317883/elementi-di-teoria-dei-linguaggi-formali/).
  - Capitolo 7