Proprieta' dei linguaggi regolari

Proprieta' dei Linguaggi regolari

- **Pumping Lemma**
  Ogni linguaggio regolare soddisfa il Pumping Lemma. Per dimostrare che un linguaggio non e' regolare useremo il pumping lemma e arriveremo a una contraddizione.

- **Proprieta' di chiusura**
  Se due linguaggi L e M sono regolari, anche i linguaggi $L \cap M$, $L \cup M$, $\bar{L}$, $L \setminus M$, L^R, L^*, $L.M$ sono regolari

- **Tecniche di minimizzazione**
  Possiamo risparmiare costruendo automi piu' piccoli.

Teorema 4.1: Il Pumping Lemma per Linguaggi Regolari

Sia L un linguaggio regolare.
Allora $\exists n, \forall w \in L : |w| \geq n \Rightarrow w = xyz$ tale che:

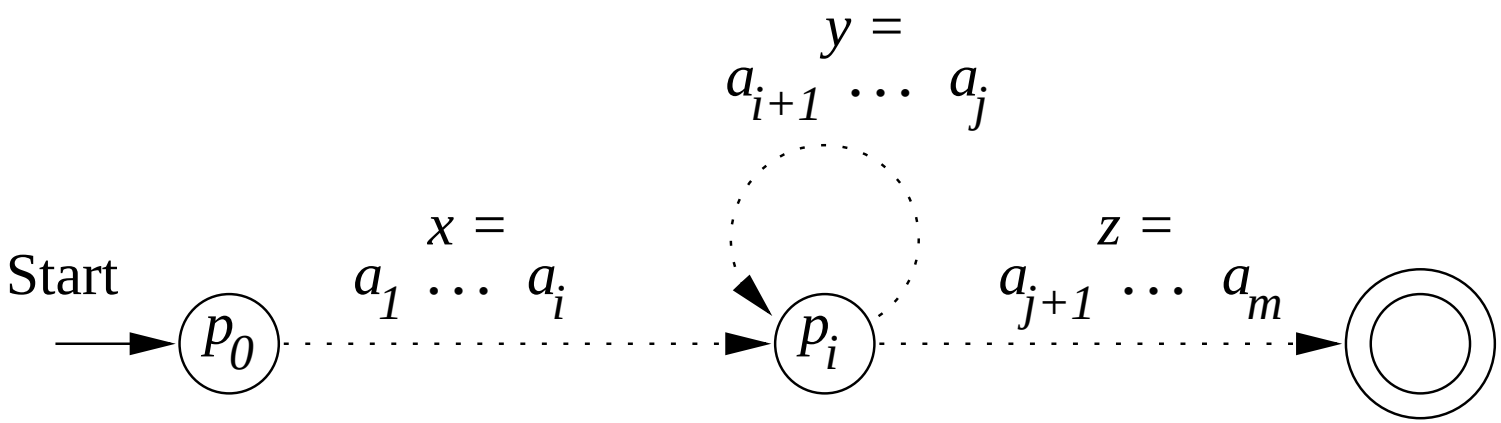
1. $y \neq \epsilon$
2. $|xy| \leq n$
3. $\forall k \geq 0,\ xy^k z \in L$

Prova del Pumping Lemma

- Supponiamo che L sia regolare.
- Allora L e' riconosciuto da un DFA A con, ad esempio, n stati.
- Sia $w = a_1 a_2 \dots a_m \in L$, $m \geq n$.
- Per $i = 0, \dots, n$ sia $p_i = \hat{\delta}(q_0, a_1 a_2 \cdots a_i)$
  p_i è lo stato in cui si trova A dopo aver letto i simboli.
  Si noti che $p_0 = q_0$.
- Dato che ci sono solo n stati in A, gli $n+1$ stati p_i, per $i = 0, \dots, n$ non possono essere tutti distinti
  $\Rightarrow \exists i < j : p_i = p_j$

Possiamo scomporre $w = xyz$:

1. $x = a_1 a_2 \cdots a_i$ (x porta da q_0 a p_i, x puo' essere vuota)
2. $y = a_{i+1} a_{i+2} \cdots a_j$ (y porta da p_i a p_j, y non puo' essere vuota)
3. $z = a_{j+1} a_{j+2} \ldots a_m$ (z porta da p_j ad uno stato accettante, z puo' essere vuota)



Quindi anche $xy^k z \in L$, per ogni $k \geq 0$.

Tecnica dei due giocatori

- Possiamo interpretare l'applicazione del Pumping Lemma come **gioco a due giocatori G1 e G2**
  - giocatore G1 vuol dimostare che L non e' regolare
  - giocatore G2 sceglie n (ma non deve comunicarlo a G1)
  - giocatore G1 sceglie $w \in L$, che puo' dipendere da n, tale che $|w| \geq n$
  - giocatore G2 sceglie la partizione $w = xyz$, con $|xy| \leq n$ e $y \neq \epsilon$ (ma non deve comunicarla a G1)
  - giocatore G1 sceglie k, che puo' dipendere da x, y, z e n, tale che $xy^k z \notin L$, cosi' viola il Pumping lemma.
  - giocatore G1 conclude che L non e' regolare.

Esercizio 1

- Provare con il Pumping Lemma che L non e' regolare,
  L= linguaggio delle stringhe su $\{0,1\}$ con ugual numero di 0 e 1.

- Soluzione:
  - giocatore G1 vuol dimostrare che L non e' regolare
  - giocatore G2 sceglie n
  - giocatore G1 sceglie $w = 0^n1^n \in L$, con $|w| \geq n$
  - giocatore G2 sceglie la partizione $w = xyz$ con $|xy| \leq n$ e $y \neq \epsilon$
  - giocatore G1 sceglie k tale che $xy^kz \notin L$.
    G1 sceglie $k = 0$. G1 deve dimostrare che $xy^0z = xz \notin L$.
    - Poiche' $|xy| \leq n$ allora, sia x che y sono composti solo da zeri poiche' abbiamo scelto $w = 0^n1^n$. Quindi $xz = 0^{n-|y|}1^n$.
    - Poiche' $y \neq \epsilon$, allora $|y| \neq 0$. Quindi $n - |y| \neq n$.
    - Percio' $xz = 0^{n-|y|}1^n \notin L$ poiche' il numero di zeri e' minore del numero di uni. G1 conclude che L non e' regolare.

Esercizio 2

- Provare con il Pumping Lemma che L non e' regolare:
  $L = \{vv^R,$ dove v^R è la stringa v rovesciata e $v \in \{0,1\}^*\}$

- Soluzione:
  - giocatore G1 vuol dimostrare che L non e' regolare
  - giocatore G2 sceglie n
  - giocatore G1 sceglie $w = 0^n1^n1^n0^n \in L$, con $|w| \geq n$
  - giocatore G2 sceglie la partizione $w = xyz$ con $|xy| \leq n$ e $y \neq \epsilon$
  - giocatore G1 sceglie $k = 0$ e deve dimostrare che $xy^0z = xz \notin L$.
    - Poiche' $|xy| \leq n$ allora sia x che y sono composti solo da zeri, poiche' abbiamo scelto $w = 0^n1^n1^n0^n$. Quindi $\underline{xz = 0^{n-|y|}1^n1^n0^n}$.
    - Poiche' $y \neq \epsilon$, allora $|y| \neq 0$. Quindi $n - |y| \neq n$.
    - Percio' $\underline{xz = 0^{n-|y|}1^n1^n0^n \notin L}$, quindi L non e' regolare.

Esercizio 3

- Provare con il Pumping Lemma che il linguaggio delle palindromi di lunghezza dispari su alfabeto $\{0, 1\}$ non è regolare.

- Soluzione:
  - giocatore G1 vuol dimostrare che L non e' regolare
  - giocatore G2 sceglie n
  - giocatore G1 sceglie $w = 0^n 1 0^n \in L$, con $|w| \geq n$
  - giocatore G2 sceglie la partizione $w = xyz$, $|xy| \leq n$ e $y \neq \epsilon$
  - giocatore G1 sceglie $k = 0$ e deve dimostrare che $xy^0z = xz \notin L$.
    - Poiche' $|xy| \leq n$ allora sia x che y sono composti solo da zeri, poiche' abbiamo scelto $w = 0^n 1 0^n$. Quindi $\underline{xz = 0^{n-|y|} 1 0^n}$.
    - Poiche' $y \neq \epsilon$, allora $|y| \neq 0$. Quindi $n - |y| \neq n$.
    - Percio' $\underline{xz = 0^{n-|y|} 1 0^n \notin L}$, quindi L non e' regolare.

Esercizi

- Esercizi per casa
  - Dimostrare con il Pumping Lemma che il linguaggio $L = \{v\bar{v},$ dove $\bar{v}$ è la stringa v in cui ogni 0 e' sostituito con 1 e ogni 1 e' sostituito con 0, $v \in \{0,1\}^*\}$ non è regolare.
  - Dimostrare con il Pumping Lemma che il linguaggio $L = \{v1^n,\ v \in \{0,1\}^*$ con $|v| = n\}$ non è regolare.
  - Esercizi delle dispense sul Pumping Lemma

Proprieta' di chiusura dei linguaggi regolari

Siano L e M due linguaggi regolari. Allora i seguenti linguaggi sono regolari:

- *Unione:* $L \cup M$
- *Intersezione:* $L \cap M$
- *Complemento:* $\overline{L}$
- *Differenza:* $L \setminus M$
- *Inversione:* $L^R = \{w^R : w \in L\}$
- *Chiusura:* L^*.
- *Concatenazione:* $L.M$

Chiusura rispetto a unione e complemento

Teorema 4.4. Per ogni coppia di linguaggi regolari L e M, $L \cup M$ e' regolare.

Prova. Poiche' L e M sono linguaggi regolari, sono descritti da espressioni regolari. Sia $L = L(E)$ e $M = L(F)$ dove E e F sono espressioni regolari. Allora $L \cup M = L(E) \cup L(F) = L(E + F)$ per definizione dell'operatore $+$ delle espressioni regolari.

Teorema 4.5. Se L e' un linguaggio regolare su Σ, allora che $\overline{L} = \Sigma^* \setminus L$ e' regolare.

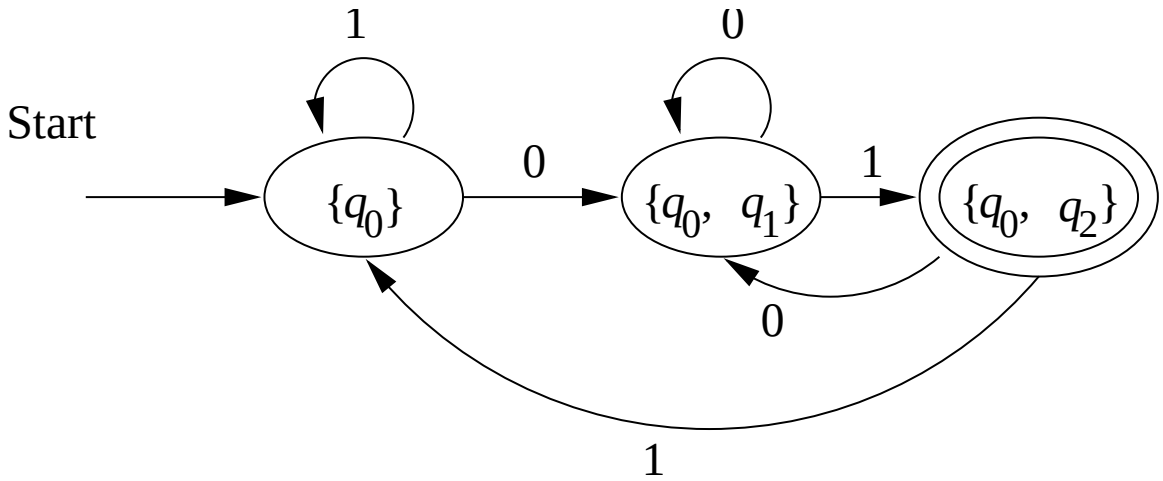
Prova. Sia L riconosciuto da un DFA

$$A = (Q, \Sigma, \delta, q_0, F).$$

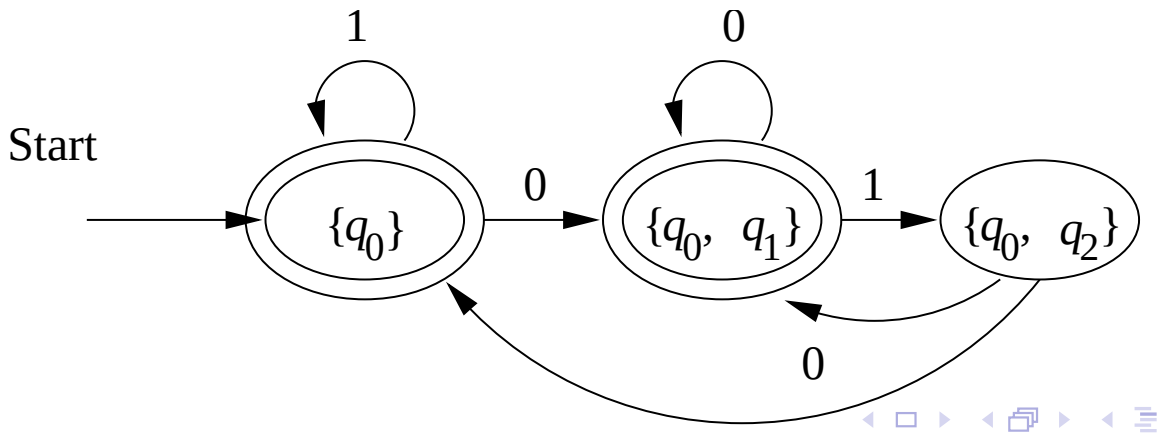
Sia $B = (Q, \Sigma, \delta, q_0, Q \setminus F)$. Allora $L(B) = \overline{L}$.

Esempio

Sia L riconosciuto dal DFA qui sotto:



Allora $\overline{L}$ e' riconosciuto da:



Chiusura rispetto all'intersezione

Teorema 4.8. Se L e M sono regolari, allora anche $L \cap M$ e' regolare.

Prova 1. Per la legge di DeMorgan, $L \cap M = \overline{\overline{L} \cup \overline{M}}$. Sappiamo gia' che i linguaggi regolari sono chiusi sotto il complemento e l'unione.

Chiusura rispetto alla differenza

Teorema 4.10 Se L e M sono linguaggi regolari, allora anche $L \setminus M$ e' regolare.

Prova. Osserviamo che $L \setminus M = L \cap \overline{M}$. Sappiamo gia' che i linguaggi regolari sono chiusi sotto il complemento e l'intersezione.

Chiusura rispetto al "reverse"

L^R e' il linguaggio formato dalle inversioni di tutte le stringhe di L. L'inversione di una stringa $a_1a_2...a_{n-1}a_n$ e' la stringa scritta al contrario $a_na_{n-1}\ldots a_2a_1$

Teorema 4.11 Se L e' un linguaggio regolare, allora anche L^R e' regolare.

Prova 1: Sia L riconosciuto da un FA A. Modifichiamo A per renderlo un FA per L^R:

1. Giriamo tutti gli archi.
2. Rendiamo il vecchio stato iniziale l'unico stato finale.
3. Creiamo un nuovo stato iniziale p_0, con $\delta(p_0, \epsilon) = F$ (i vecchi stati finali).

Esercizi

- Esercizio (Problema 2 - I compitino 2013)
  Commentare la veridicità delle seguenti affermazioni, giustificando la risposta:
  1. L'intersezione di un linguaggio non regolare L_1 e di un linguaggio regolare L_2 puo' essere non regolare;
  2. L'intersezione di un linguaggio non regolare e di un linguaggio finito è sempre regolare;
  3. Ogni sottoinsieme di un linguaggio non regolare è non regolare.

- Esercizi per casa
  - Esercizi sulle proprieta' dei linguaggi delle dispense

Esercizio

- Esercizio (Problema 2 - I compitino 2013)
  Commentare la veridicità delle seguenti affermazioni:

  1. L'intersezione di un linguaggio non regolare L_1 e di un linguaggio regolare L_2 puo' essere non regolare;
     Soluzione: VERO. Infatti, se ad esempio $L_2 = \Sigma^*$, allora $L_1 \cap L_2 = L_1$ che e' non regolare.

  2. L'intersezione di un linguaggio non regolare e di un linguaggio finito è sempre regolare;
     Soluzione: VERO. Infatti l'intersezione di un qualunque linguaggio con un linguaggio finito e' sempre un linguaggio finito e ogni linguaggio finito e' regolare.

  3. Ogni sottoinsieme di un linguaggio non regolare è non regolare.
     Soluzione: FALSO. Infatti $\emptyset$ e' sottoinsieme di tutti i linguaggi e quindi anche di quelli non regolari e $\emptyset$ e' regolare.

Equivalenza e minimizzazione di automi

Stati equivalenti

Sia $A = (Q, \Sigma, \delta, q_0, F)$ un DFA, e $\{p, q\} \subseteq Q$. Definiamo

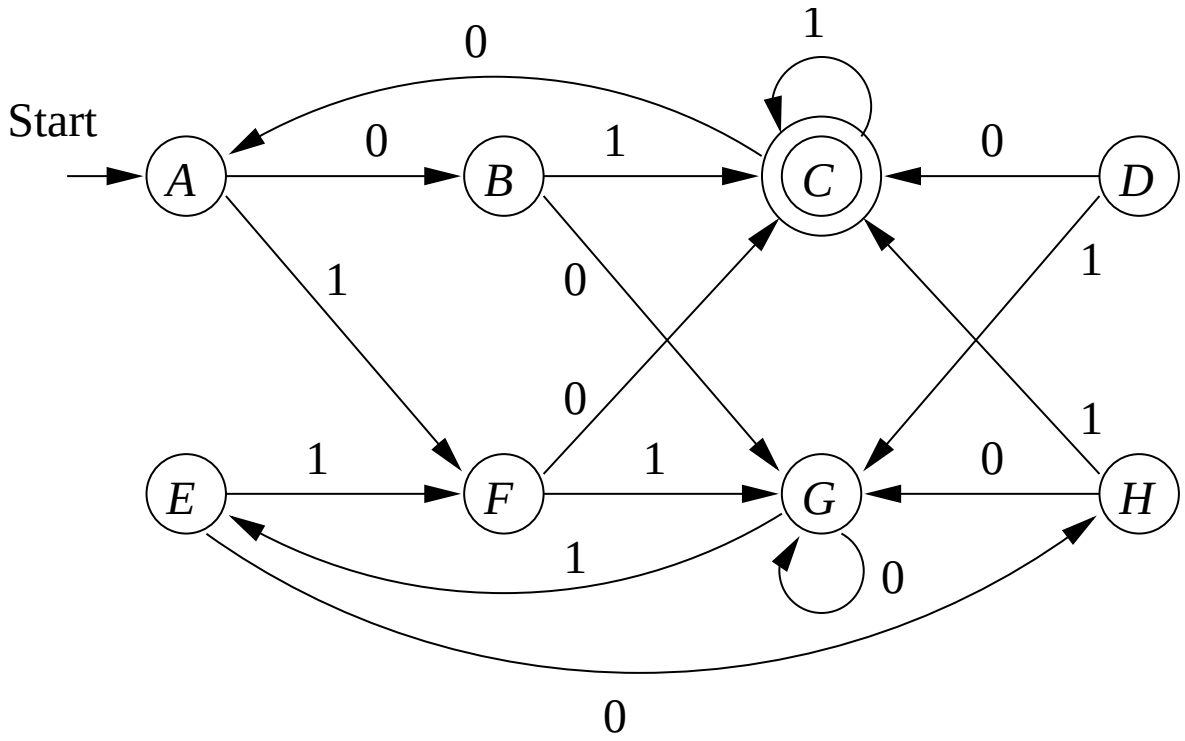
$$p \equiv q \Leftrightarrow \underline{\forall w \in \Sigma^*} : \hat{\delta}(p, w) \in F \text{ se e solo se } \hat{\delta}(q, w) \in F$$

- Se $p \equiv q$ diciamo che p e q sono equivalenti
- Se $p \not\equiv q$ diciamo che p e q sono distinguibili

In altre parole: p e q sono distinguibili se e solo se

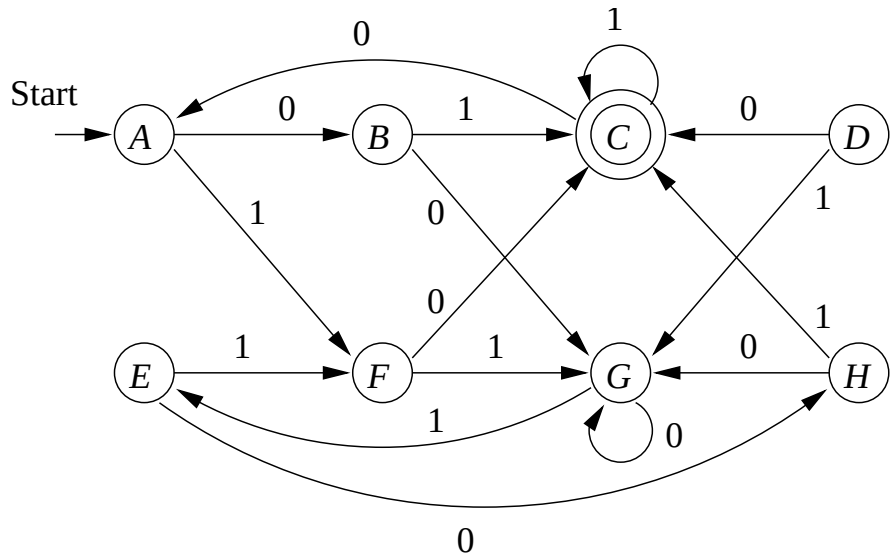
$$\exists w : \hat{\delta}(p, w) \in F \text{ e } \hat{\delta}(q, w) \notin F, \text{ o viceversa}$$

Esempio



$\hat{\delta}(C, \epsilon) \in F, \hat{\delta}(G, \epsilon) \notin F \Rightarrow C \not\equiv G$
$\hat{\delta}(A, 01) = C \in F, \hat{\delta}(G, 01) = E \notin F \Rightarrow A \not\equiv G$

Cosa si puo' dire su A e E?



$\hat{\delta}(A, \epsilon) = A \notin F, \;\; \hat{\delta}(E, \epsilon) = E \notin F$
$\hat{\delta}(A, 0) = B \notin F, \;\; \hat{\delta}(E, 0) = H \notin F$
$\hat{\delta}(A, 1) = F = \hat{\delta}(E, 1)$
Quindi $\hat{\delta}(A, 1x) = \hat{\delta}(E, 1x) = \hat{\delta}(F, x)$
$\hat{\delta}(A, 00) = G = \hat{\delta}(E, 00)$
$\hat{\delta}(A, 01) = C = \hat{\delta}(E, 01)$
Conclusione: $A \equiv E$

Algoritmo induttivo

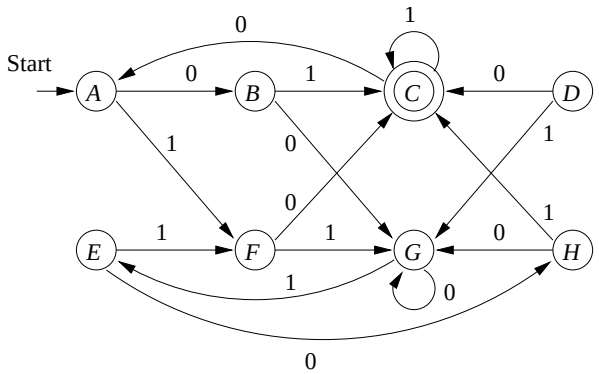
Possiamo calcolare coppie di stati distinguibili con il seguente metodo induttivo (algoritmo riempi-tabella):

Base: Se $p \in F$ e $q \notin F$, allora $p \not\equiv q$.
Induzione: Se $\exists a \in \Sigma : \delta(p,a) \not\equiv \delta(q,a)$, allora $p \not\equiv q$.

Esempio: Applichiamo l'algoritmo all'automa precedente:

B	x					
C	x	x				
D	x	x	x			
E		x	x	x		
F	x	x	x		x	
G	x	x	x	x	x	x
H	x		x	x	x	x	x
	A	B	C	D	E	F	G

Algoritmo riempi-tabella sull'esempio



- Poiche' $C \in F$ e $A, B, D, E, F, H \notin F$,
  $C \not\equiv A$, $C \not\equiv B$, $C \not\equiv D$, $C \not\equiv E$, $C \not\equiv F$, $C \not\equiv G$, $C \not\equiv H$
- Poiche' $C \not\equiv H$,
  - visto che $\delta(F, 0) = C$ e $\delta(E, 0) = H$, allora $F \not\equiv E$
  - visto che $\delta(D, 0) = C$ e $\delta(E, 0) = H$, allora $D \not\equiv E$
- Poiche' $F \not\equiv E$,
  - visto che $\delta(A, 1) = F$ e $\delta(G, 1) = E$, allora $A \not\equiv G$
  - visto che $\delta(E, 1) = F$ e $\delta(G, 1) = E$, allora $E \not\equiv G$
- .... (Completarlo per casa)

Correttezza dell'algoritmo

Teorema 4.20: Se p e q non sono distinguibili dall'algoritmo, allora $p \equiv q$.

Prova: Supponiamo per assurdo che esista una coppia "sbagliata" $\{p, q\}$, tale che

1. $p \not\equiv q : \exists w : \hat{\delta}(p, w) \in F, \hat{\delta}(q, w) \notin F$, o viceversa
2. L'algoritmo non distingue tra p e q.

Sia $w = a_1 a_2 \cdots a_n$ la stringa piu' corta che identifica la coppia "sbagliata" $\{p, q\}$.
Allora $w \neq \epsilon$ perche' altrimenti l'algoritmo distinguerebbe p da q (caso base). Quindi $n \geq 1$.

- Consideriamo gli stati $r = \delta(p, a_1)$ e $s = \delta(q, a_1)$.
- r e s sono distinti dalla stringa $a_2, \ldots, a_n$, poiche' li porta rispettivamente in $\hat{\delta}(p, w) \in F$ e $\hat{\delta}(q, w) \notin F$.
- Ma la stringa che distingue r e s e' piu' corta di ogni stringa che distingue una coppia "sbagliata"
- Allora $\{r, s\}$ non possono essere una coppia "sbagliata".
- Quindi l'algoritmo riempi-tabella deve avere scoperto che r and s sono distinguibili
- Ma allora l'algoritmo distinguerebbe p da q nella parte induttiva.
- Quindi non ci sono coppie "sbagliate" e il teorema e' vero.

L'algoritmo riempi tabella e' utile per

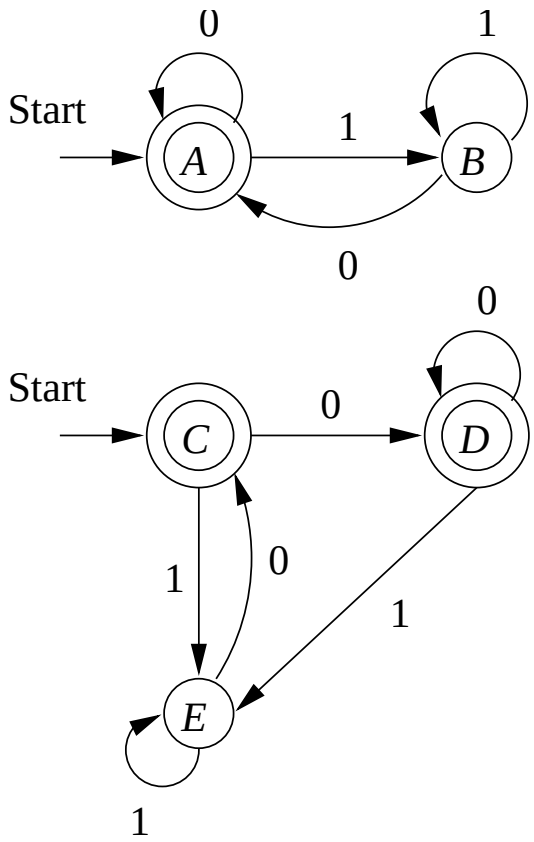
- Testare l'equivalenza di linguaggi regolari
- Costruire il DFA minimo

Testare l'equivalenza di linguaggi regolari

Siano L e M linguaggi regolari (descritti in qualche forma).
Per testare se $L = M$

1. convertiamo sia L che M in DFA.
2. Immaginiamo il DFA che e' l'unione dei due DFA (non importa se ha due stati iniziali)
3. Se l'algoritmo dice che i due stati iniziali sono distinguibili, allora $L \neq M$, altrimenti $L = M$.

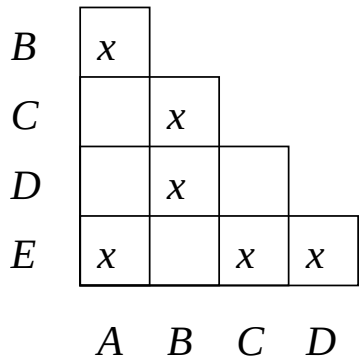
Esempio



Possiamo vedere che entrambi i DFA accettano $L(\epsilon + (\mathbf{0} + \mathbf{1})^*\mathbf{0})$.

Esempio

Il risultato dell'algoritmo e'



Poiche' A e C sono equivalenti, i due automi sono equivalenti.

Minimizzazione di DFA

Algoritmo di minimizzazione di un DFA:

1. Per prima cosa si eliminano gli stati irraggiungibili dallo stato iniziale
2. Si applica l'algoritmo riempi-tabella solo per gli stati rimanenti: tali stati vengono ripartiti in blocchi in modo che tutti gli stati di uno stesso blocco siano tutti equivalenti e due stati in blocchi diversi non siano mai equivalenti

Minimizzazione di DFA

- Possiamo usare l'algoritmo riempi-tabella per minimizzare un DFA mettendo insieme tutti gli stati equivalenti.
  Cioe' rimpiazzando p con $p/_{\equiv}$.

- Esempio: Il DFA di prima ha le seguenti classi di equivalenza: $\{\{A, E\}, \{B, H\}, \{C\}, \{D, F\}, \{G\}\}$.

- Notare: affinche' $p/_{\equiv}$ sia una classe di equivalenza, la relazione $\equiv$ deve essere una relazione di equivalenza (riflessiva, simmetrica, e transitiva).

Transitivita'

Teorema 4.23: Se $p \equiv q$ e $q \equiv r$, allora $p \equiv r$.

Prova: Supponiamo per assurdo che $p \not\equiv r$.

- Allora $\exists w$ tale che $\hat{\delta}(p, w) \in F$ e $\hat{\delta}(r, w) \notin F$, o viceversa.
- Lo stato $\hat{\delta}(q, w)$ e' o di accettazione o no.
- *Caso 1*: $\hat{\delta}(q, w)$ e' di accettazione. Allora $q \not\equiv r$.
- *Caso 2*: $\hat{\delta}(q, w)$ non e' di accettazione. Allora $p \not\equiv q$.
- Il caso contrario puo' essere provato simmetricamente.
- Quindi deve essere $p \equiv r$.

Minimizzazione di automi

Per minimizzare un DFA $A = (Q, \Sigma, \delta, q_0, F)$
costruiamo un DFA $B = (Q/_{\equiv}, \Sigma, \gamma, q_0/_{\equiv}, F/_{\equiv})$, dove

$$\gamma(p/_{\equiv}, a) = \delta(p, a)/_{\equiv}$$

Affinche' B sia ben definito, dobbiamo mostrare che

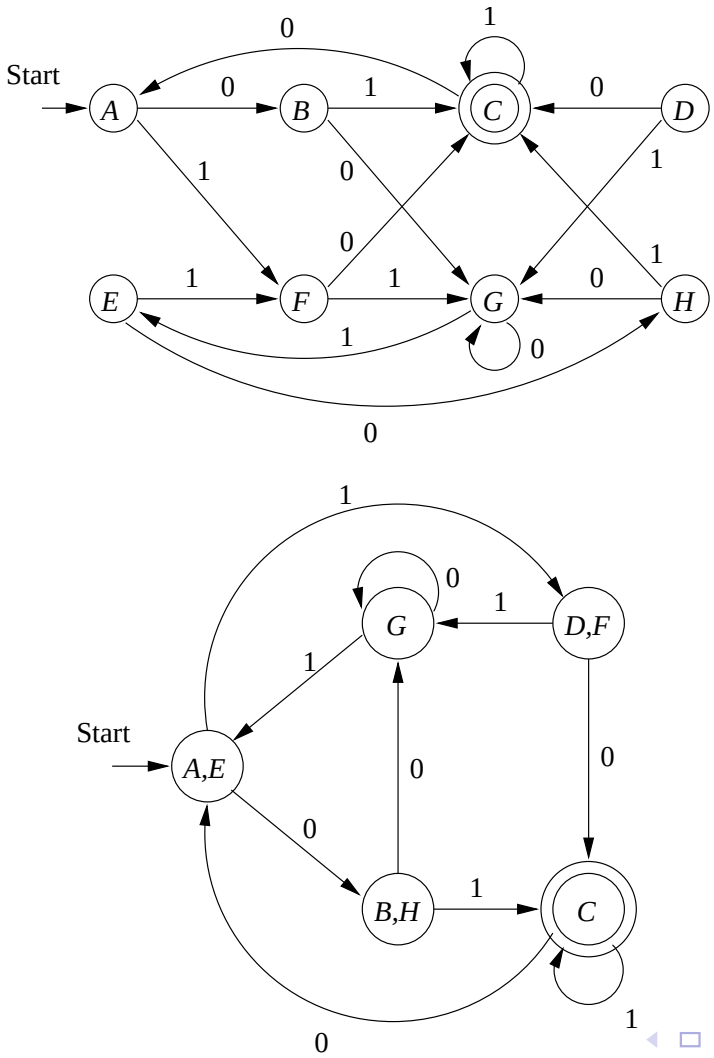
$$\text{Se } p \equiv q \text{ allora } \delta(p, a) \equiv \delta(q, a)$$

Se $\delta(p, a) \not\equiv \delta(q, a)$, allora l'algoritmo concluderebbe $p \not\equiv q$, quindi B e' ben definito.
Notare anche che $F/_{\equiv}$ contiene tutti e soli gli stati accettanti di A.

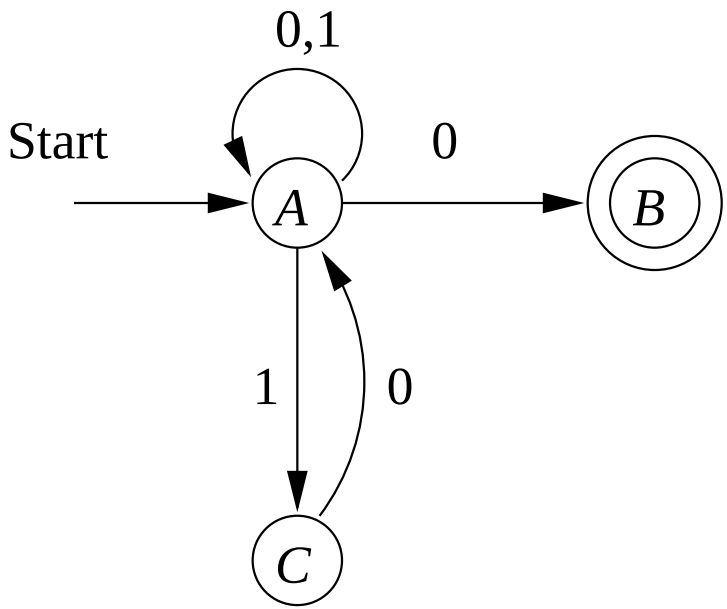
Il DFA minimo non puo' essere migliorato

Esempio

Costruiamo il DFA minimo dell'automa seguente



Notare: Non possiamo applicare l'algoritmo a NFA.
Per esempio, per minimizzare



rimuoviamo lo stato C e otteniamo un automa piu' piccolo
Ma $A \not\equiv C$ e quindi l'algoritmo riempi-tabella non ci avrebbe consentito di rimuovere C.

Esercizio

- Esercizio per casa: Problema 2 - II appello 2013
  1. Dare la definizione di stati equivalenti.
  2. Costruire il **diagramma** di transizione dell'automa che minimizza l'automa seguente spiegando **sinteticamente** il procedimento seguito.
     $Q = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{0, 1\}, \delta, q_0, \{q_3, q_5\})$ dove δ è descritta dalla seguente tabella di transizione:

	0	1
$\rightarrow q_0$	q_1	q_3
q_1	q_0	q_3
q_2	q_1	q_4
$*q_3$	q_5	q_5
q_4	q_3	q_3
$*q_5$	q_5	q_5