

Linguaggi Formali e Traduttori

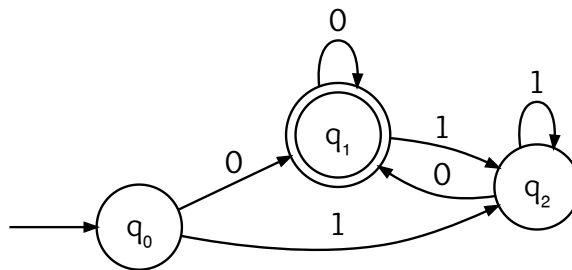
2.7 Minimizzazione di automi a stati finiti deterministici

- Sommario
- Stati (in)distinguibili
- Esempio
- Esempio (dal libro)
- Algoritmo per trovare stati distinguibili
- Esempio
- L'indistinguibilità come equivalenza
- Costruzione dell'automa minimo
- Esempio
- Equivalenza di automi
- Esercizi di minimizzazione
- Esercizi di equivalenza

È proibito condividere e divulgare in qualsiasi forma i materiali didattici caricati sulla piattaforma e le lezioni svolte in videoconferenza: ogni azione che viola questa norma sarà denunciata agli organi di Ateneo e perseguita a termini di legge.

Sommario

Motivazione



- Osservando attentamente questo DFA ci accorgiamo c'è una **ridondanza**
- Una stringa accettata partendo da **q_0** è accettata anche partendo da **q_2** e viceversa
- Possiamo “fondere” **q_0** e **q_2** senza alterare il linguaggio riconosciuto dall'automa

Problema

- In generale, è desiderabile lavorare con DFA “piccoli”
- Avendo a disposizione un DFA, è possibile individuarne gli stati ridondanti e “minimizzarlo”?

In questa lezione

- Definiamo un algoritmo che costruisce il DFA **minimo** (cioè con il più piccolo numero di stati) che riconosce un certo linguaggio regolare.

Stati (in)distinguibili

Definizione

Dato un DFA $A = (Q, \Sigma, \delta, q_0, F)$, diciamo che $p, q \in Q$ sono **indistinguibili** se

$$\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$$

per ogni $w \in \Sigma^*$.

Nota

Affinché p e q siano indistinguibili **non è necessario** che $\hat{\delta}(p, w)$ e $\hat{\delta}(q, w)$ siano lo stesso stato per ogni w , ma solo che siano entrambi finali o entrambi non finali per ogni w .

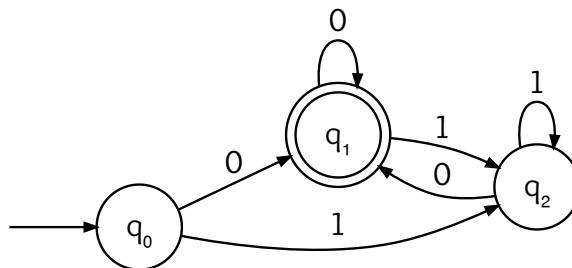
Definizione

Diciamo che due stati $p, q \in Q$ sono **distinguibili** se esiste $w \in \Sigma^*$ tale che solo uno tra $\hat{\delta}(p, w)$ e $\hat{\delta}(q, w)$ appartiene a F . Diciamo che una stringa w con questa proprietà **distingue** p da q .

Nota

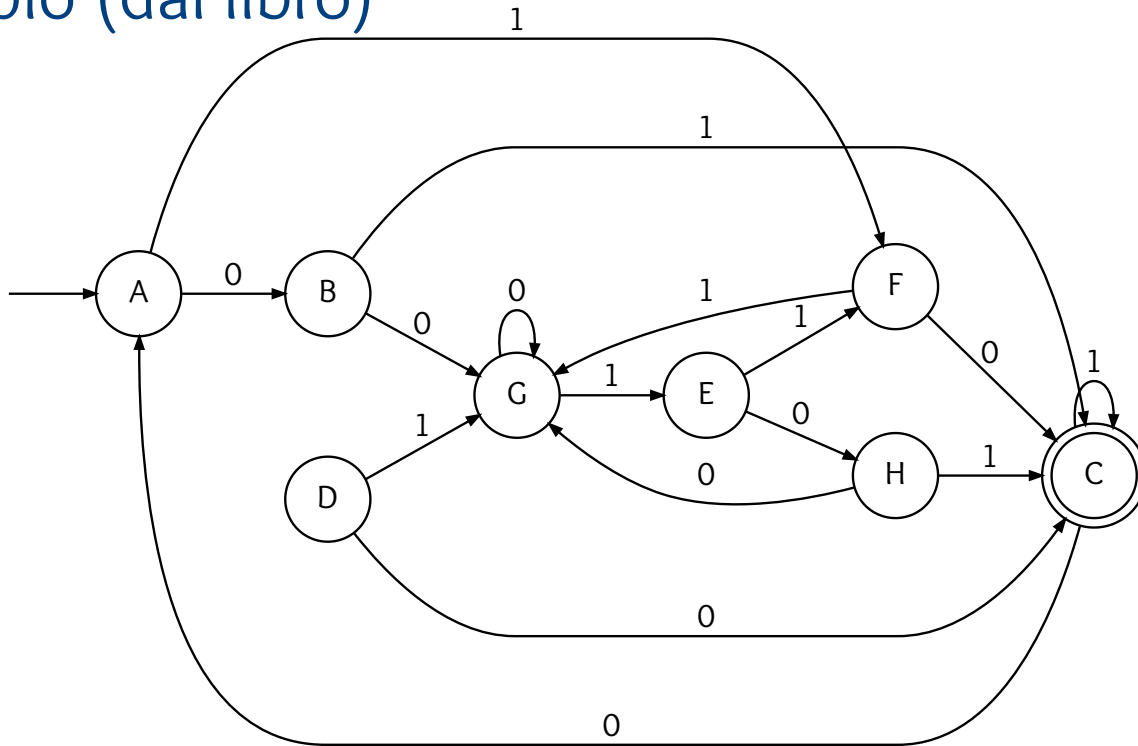
In generale ci possono essere molte stringhe che distinguono due stati.

Esempio



- La stringa ϵ distingue q_0 e q_1 , in quanto q_1 è finale mentre q_0 no.
- Per lo stesso motivo, la stringa ϵ distingue q_1 e q_2 .
- q_0 e q_2 sono indistinguibili, infatti:
 - sono entrambi non finali, dunque ϵ non li distingue;
 - la stringa 0 non li distingue, poiché li porta entrambi in q_1 ;
 - la stringa 1 non li distingue, poiché li porta entrambi in q_2 ;
 - dopo uno 0 o un 1, q_0 e q_2 confluiscono nello stesso stato e da lì in avanti il loro percorso è lo stesso. Dunque, non può esistere alcuna stringa che li distingua.

Esempio (dal libro)



- **A** e **G** sono distinti da 01 ma non da \emptyset né da ϵ
- **A** ed **E** sono indistinguibili (confluiscono dopo 1, 00 e 01)

Algoritmo per trovare stati distinguibili

Input

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

Output

L'insieme di tutte e sole le coppie di stati distinguibili di A .

Algoritmo

1. All'inizio nessuna coppia di stati è marcata come distinguibile.
2. Si marcano come distinguibili tutte le coppie $\{p, q\}$ in cui $p \in F$ e $q \notin F$.
3. Se esistono $p, q \in Q$ e $a \in \Sigma$ tali che $\{\delta(p, a), \delta(q, a)\}$ è marcata come distinguibile, si marca anche $\{p, q\}$ come distinguibile.
4. Si ripete il passo 3 fintantoché vengono marcate **nuove** coppie distinguibili.

Come eseguire l'algoritmo

- Si crea una **tabella triangolare** le cui righe sono etichettate con gli stati dal secondo all'ultimo e le cui colonne sono etichettate con gli stati dal primo al penultimo.
- Si marca con \checkmark ogni casella corrispondente a una coppia distinguibile (passi 2 e 3).
- Si itera il passo 3 considerando tutte le caselle non marcate fintantoché possibile.

Esempio

Seguono le tabelle corrispondenti ai passi 1–3 dell'algoritmo eseguito sul DFA della [slide 5](#).

Esempio

Seguono le tabelle corrispondenti ai passi 1–3 dell'algoritmo eseguito sul DFA della [slide 5](#).

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

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

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

Note

- Iterando ancora una volta il passo 3 dell'algoritmo non si marcano altre coppie come distinguibili, dunque l'algoritmo termina.
- In generale possono servire più iterazioni del passo 3 prima che l'algoritmo termini.

L'indistinguibilità come equivalenza

Proposizione

L'indistinguibilità è una relazione di equivalenza, ovvero è riflessiva, simmetrica e transitiva.

Notazione

Fissato un DFA $A = (Q, \Sigma, \delta, q_0, F)$, introduciamo la seguente notazione:

- Indichiamo con \sim la **relazione di indistinguibilità** tra stati di A , ovvero $p \sim q$ se e solo se $\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$ per ogni $w \in \Sigma^*$.
- Scriviamo $[p]$ per la **classe di equivalenza** di p , ovvero $[p] = \{q \in Q \mid p \sim q\}$.
- Scriviamo X/\sim per l'**insieme quoziente** di X rispetto a \sim , cioè $X/\sim = \{[p] \mid p \in X\}$.

Costruzione dell'automa minimo

Algoritmo

Dato un DFA $(Q, \Sigma, \delta, q_0, F)$, nel quale si assume di aver eliminato gli stati irraggiungibili dallo stato iniziale, l'**automa minimo** corrispondente è l'automa

$$(Q/\sim, \Sigma, \delta', [q_0], F/\sim)$$

in cui

$$\delta'([p], a) = [\delta(p, a)]$$

per ogni $p \in Q$ ed $a \in \Sigma$.

Teorema

Per ogni DFA A , non esiste un DFA equivalente il cui numero di stati è strettamente inferiore a quello dell'automa minimo corrispondente ad A costruito secondo l'algoritmo qui sopra.

Dimostrazione (facoltativa)

Si veda la Sezione 4.4.4 del libro.

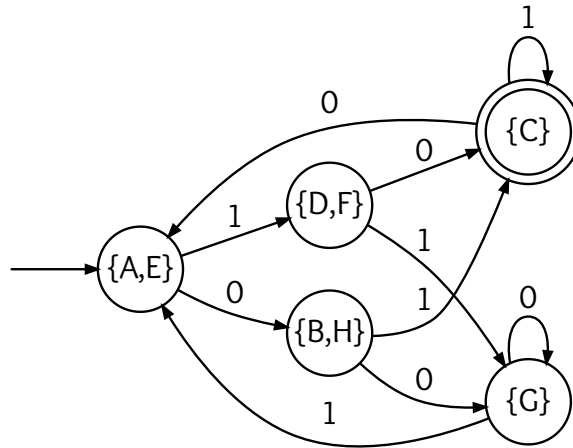
Esempio

Nell'automa della [slide 5](#) gli stati diversi ma indistinguibili sono

$$A \sim E \quad B \sim H \quad D \sim F$$

pertanto l'automa minimo, illustrato sotto, ha come insieme degli stati

$$\{\{A, E\}, \{B, H\}, \{C\}, \{D, F\}, \{G\}\}$$



Equivalenza di automi

L'algoritmo **riempi-tabella** può essere usato per decidere se due automi sono **equivalenti**.

Input

Due DFA $A_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ e $A_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ in cui possiamo assumere, senza perdere in generalità, che $Q_1 \cap Q_2 = \emptyset$.

Output

Vero se $L(A_1) = L(A_2)$ e **falso** altrimenti.

Algoritmo

1. Si crea l'unione dei due DFA $A = (Q_1 \cup Q_2, \Sigma, \delta, q_1, F_1 \cup F_2)$ dove

$$\delta(q, a) = \begin{cases} \delta_1(q, a) & \text{se } q \in Q_1 \\ \delta_2(q, a) & \text{se } q \in Q_2 \end{cases}$$

La scelta di eleggere q_1 come stato iniziale è fatta solo perché uno stato iniziale deve esserci.

2. Si esegue l'algoritmo riempi-tabella su A .
3. A_1 ed A_2 sono equivalenti se e solo se q_1 e q_2 sono indistinguibili in A .

Esercizi di minimizzazione

1. Costruire il DFA minimo equivalente ai seguenti automi (esercizi 4.4.1 e 4.4.2 del libro di testo):

	0	1
→A	B	A
B	A	C
C	D	B
*D	D	A
E	D	F
F	G	E
G	F	G
H	G	D

	0	1
→A	B	E
B	C	F
*C	D	H
D	E	H
E	F	I
*F	G	B
G	H	B
H	I	C
*I	A	E

2. Costruire il DFA minimo equivalente all'espressione regolare a^*b^* .

Esercizi di equivalenza

1. Dimostrare che i seguenti automi sono equivalenti

