

# Automati e Linguaggi Formali

## Equivalenza e minimizzazione di automi

A.A. 2014-2015  
Enrico Mezzetti  
emezzett@math.unipd.it



UNIVERSITÀ  
DEGLI STUDI  
DI PADOVA

## Equivalenza tra stati

- Sia  $A = (Q, \Sigma, \delta, q_0, Q_F)$  un DFA con  $\{p, q\} \subseteq Q$

$$p \equiv q \Leftrightarrow \forall w \in \Sigma^* : \hat{\delta}(p, w) \in Q_F \iff \hat{\delta}(q, w) \in Q_F$$

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

- **Distinguibilita'**  $\rightarrow p$  e  $q$  sono distinguibili se e solo se  
 $\exists w : \hat{\delta}(p, w) \in Q_F$  e  $\hat{\delta}(q, w) \notin Q_F$  (o viceversa)

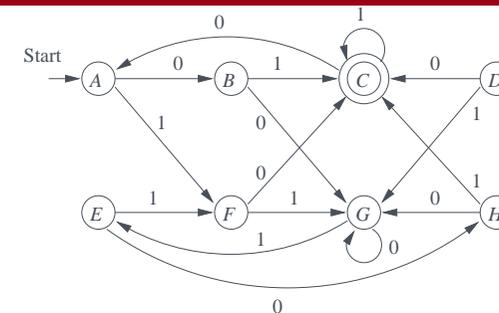


## Proprietà dei linguaggi regolari

- **Pumping Lemma**
  - Ogni linguaggio regolare soddisfa il pumping lemma
  - Se  $L$  non è regolare il pumping lemma mostrerà una contraddizione
- **Proprietà di chiusura**
  - Costruire automi da componenti usando delle operazioni algebriche sui linguaggi
  - E.g., dati  $L$  e  $M$  possiamo costruire un automa per  $L \cap M$
- **Proprietà di decisione**
  - Analisi computazionale degli automi
  - Quanto costa controllare proprietà sugli automi
- **Tecniche di minimizzazione**
  - Risparmiare costruendo automi più piccoli
  - Processo di semplificazione



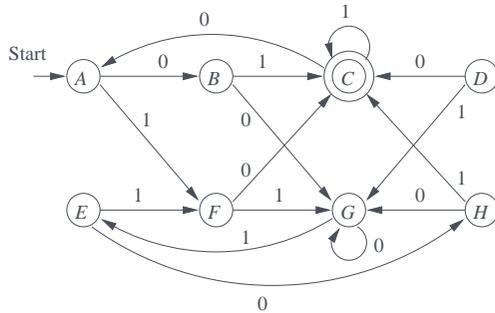
## Esempio di equivalenza



- Alcuni stati ovviamente *distinguibili*
  - $\hat{\delta}(C, \epsilon) \in Q_F, \hat{\delta}(G, \epsilon) \notin Q_F \implies C \not\equiv G$
- Altri meno...
  - $A \stackrel{?}{\equiv} G \rightarrow$  per  $\epsilon, 0, 1$ ?
  - $\hat{\delta}(A, 01) = C \in Q_F, \hat{\delta}(G, 01) = E \notin Q_F \implies A \not\equiv G$



## Esempio di equivalenza - 2



- $\hat{\delta}(A, \epsilon) = A \notin Q_F, \hat{\delta}(E, \epsilon) = E \notin Q_F$
- $\hat{\delta}(A, 1) = F = \hat{\delta}(E, 1)$ 
  - $\hat{\delta}(A, 1x) = \hat{\delta}(E, 1x) = \hat{\delta}(F, x)$
- $\hat{\delta}(A, 00) = G = \hat{\delta}(E, 00)$
- $\hat{\delta}(A, 01) = C = \hat{\delta}(E, 01)$



## Algoritmo induttivo

- Algoritmo di riempimento di una tabella (*TF algorithm*)
  - Trovare coppie di stati *distinguibili*
    - ▶ Coppie residue → equivalenti

**Base** Se  $p \in Q_F$  e  $q \notin Q_F$  allora  $p \neq q$

**Induzione** Se  $\exists a \in \Sigma : \delta(p, a) \neq \delta(q, a)$  allora  $p \neq q$

🔗 Applichiamo a DFA A

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



## Correttezza dell'algoritmo

**Th. 4.20** Se  $p$  e  $q$  non sono distinguibili dall'algoritmo allora anche  $p \equiv q$ .

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

- $\exists w : \hat{\delta}(p, w) \in Q_F, \hat{\delta}(q, w) \notin Q_F$  o viceversa
- l'algoritmo TF non distingue tra  $p$  e  $q$

- Per induzione su  $w = a_1 a_2 \dots a_n$  la stringa *piu' corta* che identifica  $\{p, q\}$  come distinguibili (non colta da TF)

**Base** ( $n = 0$ )  $w = \epsilon$ : TF non puo' non distinguere  $p$  e  $q$

**Induzione** ( $n \geq 1$ ): allora  $\exists r = \delta(p, a_1)$  e  $s = \delta(q, a_1)$

- $p \neq q \Rightarrow r \neq s$  per la stringa  $a_2 \dots a_n$  che e' piu' corta della stringa minima che riconosce una coppia sbagliata
- $\{r, s\}$  non e' una coppia sbagliata e TF deve aver scoperto  $r \neq s$
- per la sua parte induttiva TF deve aver scoperto anche  $p \neq q$



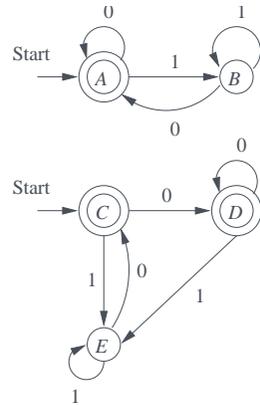
## Testare l'equivalenza tra LR

- Siano  $L$  e  $M$  linguaggi regolari (descritti in qualche forma)
- Per testare se  $L \equiv M$

- 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 TF dice che i due stati iniziali sono distinguibili, allora  $L \neq M$ , altrimenti  $L \equiv M$



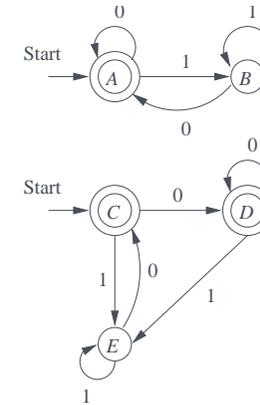
## Esempio test equivalenza LR



- Entrambi accettano  $L(\epsilon + (0 + 1)^*0)$



## Esempio test equivalenza LR

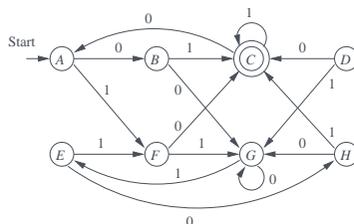


- Entrambi accettano  $L(\epsilon + (0 + 1)^*0)$
- Applichiamo TF
- Complessita'?



## Minimizzazione di DFA

- Equivalenza → **minimizzazione**
- Per ogni DFA che accetta  $L$  possiamo trovare un DFA
  - *Equivalente*
  - *Min* numero di stati rispetto a tutti i DFA che accettano  $L$
  - Tale DFA e' *unico* per  $L$
- Algoritmo TF per clusterizzare stati equivalenti
  - Rimpiazzare  $p$  con  $p/\equiv$  (*classe di equivalenza*)
    - ▶ *Riflessiva, simmetrica e transitiva*
- Esempio



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



## Transitivita'

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

- Prova* Supponiamo *per assurdo* che  $p \not\equiv r$
- $\exists w$  tale che  $\hat{\delta}(p, w) \in Q_F$  e  $\hat{\delta}(r, w) \notin Q_F$
  - $\hat{\delta}(q, w)$  puo' essere o non essere in  $Q_F$ 
    - ▶  $\hat{\delta}(q, w) \in Q_F \rightarrow q \not\equiv r$
    - ▶  $\hat{\delta}(q, w) \notin Q_F \rightarrow p \not\equiv q$
  - Simmetricamente nell'altra ipotesi
  - Concludiamo che  $p \equiv r$



# Algoritmo di minimizzazione

- Per minimizzare un DFA  $A = (Q, \Sigma, \delta, q_0, F)$  costruiamo un

$$DFAB = (Q/\equiv, \Sigma, \gamma, q_0/\equiv, F/\equiv)$$

dove

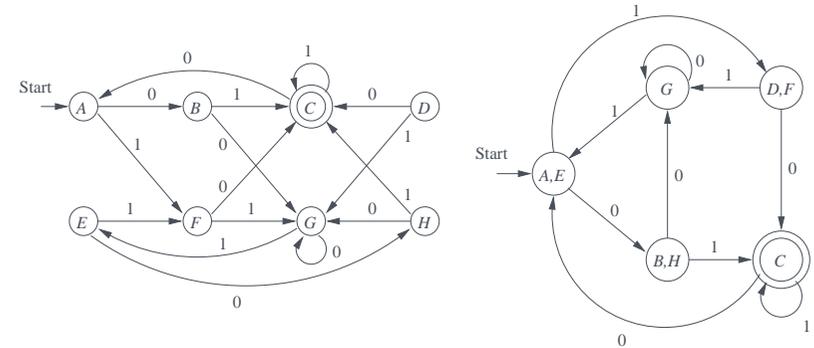
$$\gamma(p/\equiv, a) = \delta(p, a)/\equiv$$

- $B$  ben definito se

$$p \equiv q \Rightarrow \delta(p, a) \equiv \delta(q, a)$$

- In effetti: se  $\delta(p, a) \not\equiv \delta(q, a)$  TF concluderebbe  $p \not\equiv q$
- Notare anche che  $F/\equiv$  contiene tutti e soli gli stati accettanti di  $A$

# Esempio di minimizzazione

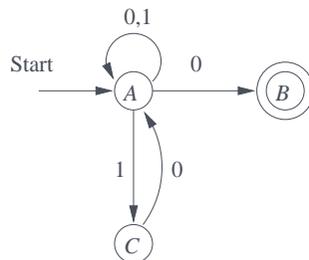


# Minimizzazione di NFA

- Possiamo applicare l'algoritmo di minimizzazione a NFA? **NO**

- Possiamo trovare NFA minimo per  $L$  (per enumerazione esaustiva)
- Non possiamo "raggruppare" gli stati di un NFA (non riduce il numero di stati)

- Esempio



# DFA Minimo

- Sia  $B$  il DFA minimizzato ottenuto su DFA  $A$

- $L(A) = L(B)$
- Può esistere DFA  $C$  con  $L(C) = L(B)$  ma meno stati di  $B$ ?

- La risposta è **NO**

**Prova** Appliciamo TF a  $B$  e  $C$

- Stati iniziali sono indistinguibili poiché  $L(B) \equiv L(C)$
- Se  $\{p, q\}$  sono indistinguibili, lo sono anche i rispettivi successori per ogni  $a \in \Sigma$
- Non esistono stati irraggiungibili (altrimenti non DFA minimo)
- $\forall p$  in  $B, \exists q$  in  $C$  tale che  $p \equiv q$
- Se per assurdo  $C$  avesse meno stati, allora ci sarebbero almeno due stati in  $B$  non distinguibili dallo stesso stato in  $C$  e quindi non distinguibile con un'altro degli stati in  $B$  (impossibile)

