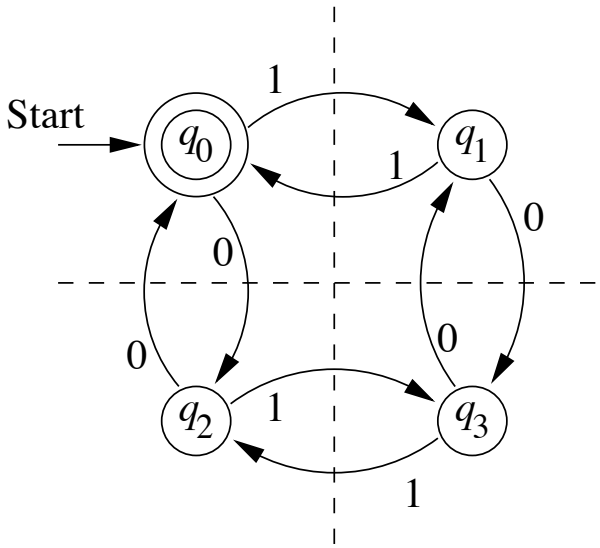
Esempio

Esempio: DFA che accetta tutte e sole le stringhe con un numero pari di zeri e un numero pari di uni (compresa quindi la stringa ϵ)



Esempio

Rappresentazione tabulare dell'automa

		0	1
$\star \rightarrow$	q_0	q_2	q_1
	q_1	q_3	q_0
	q_2	q_0	q_3
	q_3	q_1	q_2

q_0 rappresenta lo stato in cui sia il numero di 0 che di 1 è pari
q_1 rappresenta lo stato in cui il numero di 0 è pari e quello di 1 è dispari
q_2 rappresenta lo stato in cui il numero di 0 è dispari e quello di 1 è pari
q_3 rappresenta lo stato in cui sia il numero di 0 che di 1 è dispari

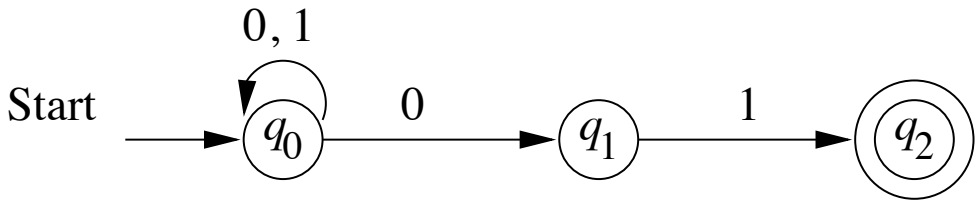
Alcuni esercizi

- DFA per i seguenti linguaggi sull'alfabeto $\{0, 1\}$:
  - Insieme di tutte le stringhe che finiscono con 00
  - Insieme di tutte le stringhe con tre zeri consecutivi
  - Insieme delle stringhe con 011 come sotto-stringa
  - Insieme delle stringhe che cominciano o finiscono (o entrambe le cose) con 01 [Provare a fare prima un automa per le stringhe che cominciano con 01 e uno per quelle che finiscono con 01]

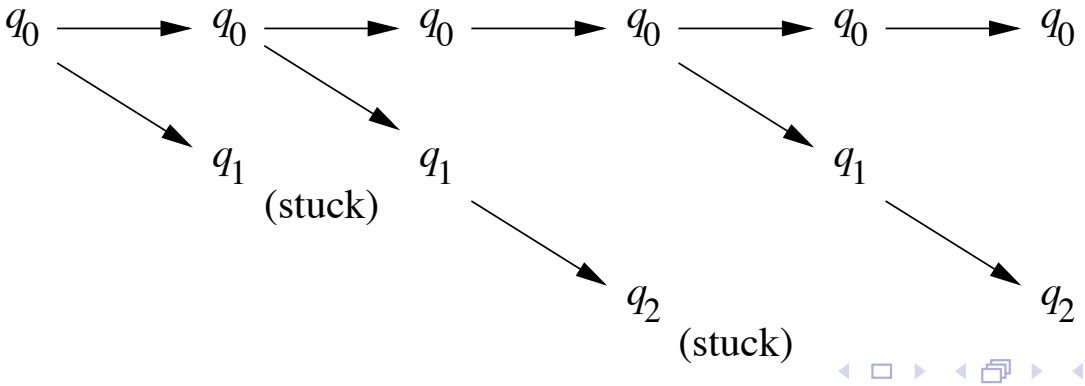
Automi a stati finiti non deterministici (NFA)

- Un NFA può essere in vari stati nello stesso momento, intuitivamente può "scommettere" su quale sarà il prossimo stato.
- Da uno stato, dato un certo input, posso infatti raggiungere un insieme di stati.

Es: automa che accetta tutte e solo le stringhe che finiscono in 01.



Ecco cosa succede quando l'automa elabora l'input 00101 con l'*albero delle alternative*



Definizione formale di NFA

Formalmente, un NFA è una quintupla

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

- Q è un insieme finito di stati
- Σ è un alfabeto finito
- δ è una funzione di transizione da $Q \times \Sigma$ all'insieme dei sottoinsiemi di Q
- $q_0 \in Q$ è lo *stato iniziale*
- $F \subseteq Q$ è un insieme di *stati finali*

Funzione di transizione

- La funzione di transizione δ prende in ingresso uno stato e un simbolo e restituisce un insieme di stati (che può essere anche vuoto).
- Lo stato successivo non è quindi univocamente determinato dallo stato corrente e dall'input. L'automa può *scegliere* tra stati diversi.
- $\delta(q, a) = \{p_1, ..., p_n\}$ denota che ogni stato p_i può essere raggiunto dallo stato q quando si riceve in input il simbolo a

$$q \rightarrow p_i \Leftrightarrow p_i \in \delta(q, a)$$

- Non è detto che per ogni stato sia sempre definito uno stato successivo per ogni simbolo di input.

Esempio

L' NFA di due pagine fa è

$$(\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$

dove δ è la funzione di transizione

	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
$\star q_2$	$\emptyset$	$\emptyset$

Funzione di transizione estesa $\hat{\delta}$ e il linguaggio accettato

Definizione induttiva di $\hat{\delta}$.

Definizione

Base: $\hat{\delta}(q, \epsilon) = \{q\}$

Induzione:

$$\hat{\delta}(q, xa) = \bigcup_{p \in \hat{\delta}(q,x)} \delta(p, a)$$

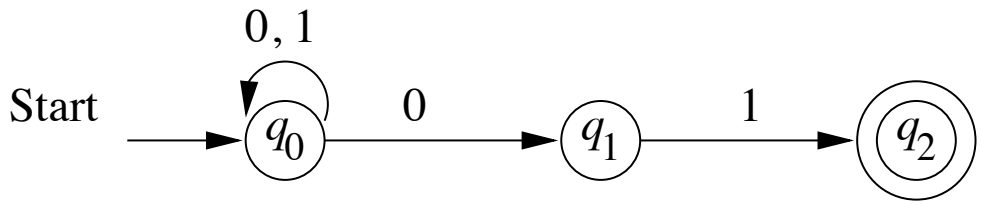
cioè l'unione di tutti gli stati $\delta(p, a)$ per p che appartiene a $\hat{\delta}(q, x)$

- Un NFA *accetta* una stringa w se $\hat{\delta}(q_0, w) \in F$ contiene almeno uno stato finale

Definizione

Il *linguaggio accettato da* A è $L(A) = \{w : \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$

Automi a stati finiti non deterministici (NFA)



Applichiamo la funzione di transizione estesa $\hat{\delta}$ all'input 00101:

1. $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
2. $\hat{\delta}(q_0, 0) = \{q_0, q_1\}$
3. $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
4. $\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$
5. $\hat{\delta}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
6. $\hat{\delta}(q_0, 00101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

con $\{q_0, q_2\} \cap F \neq \emptyset$

Dimostrazioni di equivalenza

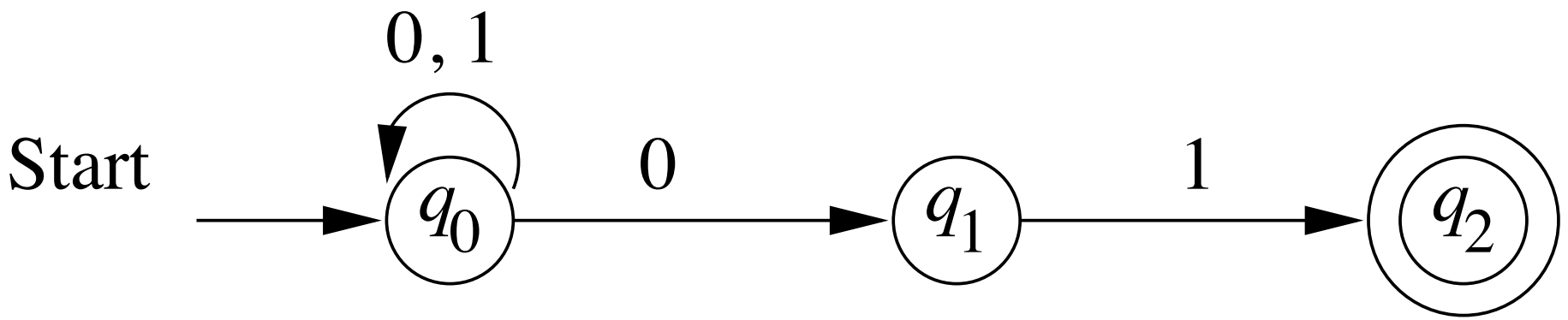
- Spesso ci servirà dimostrare che due descrizioni di insiemi, descrivono lo stesso insieme, ad esempio che il linguaggio accettato dall'automa visto prima coincide con il linguaggio $\{x01 : x \in \Sigma^*\}$
- Per dimostrare che due insiemi S e T sono uguali, si deve dimostrare che $S \subseteq T$ e che $T \subseteq S$:
  - $w \in S \Rightarrow w \in T$, and
  - $w \in T \Rightarrow w \in S$
- Nel nostro esempio dobbiamo dimostrare che

$$w \in L(A) \quad \Leftrightarrow \quad w \in \{x01 : x \in \Sigma^*\}$$

ovvero

$$q_2 \in \hat{\delta}(q_0, w) \quad \Leftrightarrow \quad w \in \{x01 : x \in \Sigma^*\}$$

Per dimostrare formalmente che l'NFA



accetta il linguaggio $\{x01 : x \in \Sigma^*\}$, ci conviene espandere le ipotesi. La dimostrazione si fa per mutua induzione, sulla lunghezza della stringa w, sui tre enunciati seguenti

1. $w \in \Sigma^* \Rightarrow q_0 \in \hat{\delta}(q_0, w)$
2. $q_1 \in \hat{\delta}(q_0, w) \Leftrightarrow w = x0$
3. $q_2 \in \hat{\delta}(q_0, w) \Leftrightarrow w = x01$

Dimostrazione

1. $w \in \Sigma^* \Rightarrow q_0 \in \hat{\delta}(q_0, w)$
2. $q_1 \in \hat{\delta}(q_0, w) \Leftrightarrow w = x0$
3. $q_2 \in \hat{\delta}(q_0, w) \Leftrightarrow w = x01$

Base: Se $|w| = 0$ allora $w = \epsilon$. Allora l'enunciato (1) segue dalla definizione. Per (2) e (3) le ipotesi di entrambi i lati sono false per ϵ e quindi entrambe le implicazioni sono banalmente vere, dato che in logica:

$$(False \Rightarrow False) \equiv True$$

Induzione: Ipotizziamo che $w = xa$, dove $a \in \{0, 1\}$, $|x| = n$ e gli enunciati (1)–(3) valgono per x. Si mostra che gli enunciati valgono per xa (che è lunga $n + 1$).

Dimostrazione (cont.)

Induzione: Ipotizziamo che $w = xa$, dove $a \in \{0, 1\}$, $|x| = n$ e gli enunciati (1)–(3) valgono per x. Si mostra che gli enunciati valgono per xa (che è lunga $n + 1$).

1. $w \in \Sigma^* \Rightarrow q_0 \in \hat{\delta}(q_0, w)$
   Per ipotesi induttiva $q_0 \in \hat{\delta}(q_0, x)$. Dato che $\forall a \in \Sigma . q_0 \in \delta(q_0, a)$, allora $\hat{\delta}(q_0, w) = \delta(\hat{\delta}(q_0, x), a) \ni q_0$

Dimostrazione (cont.)

Induzione: Ipotizziamo che $w = xa$, dove $a \in \{0, 1\}$, $|x| = n$ e gli enunciati (1)–(3) valgono per x. Si mostra che gli enunciati valgono per xa (che è lunga $n + 1$).

2.
   - (**se**) $q_1 \in \hat{\delta}(q_0, w) \Leftarrow w = x0$
     Supponiamo che $w = xa$ finisca per 0 (quindi $a = 0$).
     Per l'enunciato (1) sappiamo che $q_0 \in \hat{\delta}(q_0, x)$ e
     dato che $q_1 \in \delta(q_0, 0)$, possiamo concludere che $q_1 \in \hat{\delta}(q_0, w)$.
   - (**solo se**) $q_1 \in \hat{\delta}(q_0, w) \Rightarrow w = x0$
     Supponiamo che $q_1 \in \hat{\delta}(q_0, w)$. Dal diagramma vediamo che l'unico modo di raggiungere q_1 è che $w = x0$.

Dimostrazione (cont.)

3.
   - (**se**) $q_2 \in \hat{\delta}(q_0, w) \Leftarrow w = x01$
     Supponiamo che $w = xa$ finisca per 01 (quindi $a = 1$ e x finisce per 0).
     Per l'enunciato (2) sappiamo che $q_1 \in \hat{\delta}(q_0, x)$ e
     dato che $q_2 \in \delta(q_1, 1)$, possiamo concludere che $q_2 \in \hat{\delta}(q_0, w)$.
   - (**solo se**) $q_2 \in \hat{\delta}(q_0, w) \Rightarrow w = x01$
     Supponiamo che $q_2 \in \hat{\delta}(q_0, w)$. Dal diagramma vediamo che
     l'unico modo di raggiungere q_2 è che $w = x1$, dove $q_1 \in \hat{\delta}(q_0, x)$.
     Per l'enunciato (2), x finisce per 0 e quindi w finisce per 01.

Equivalenza di DFA e NFA

- Gli NFA sono di solito più facili da "programmare".
- Sorprendentemente, per ogni NFA N c'è un DFA D, tale che $L(D) = L(N)$, e viceversa.
- Questo comporta una *costruzione per sottoinsiemi*, un esempio importante di come un automa B può essere costruito a partire da un altro automa A.
- Dato un NFA

$$N = (Q_N, \Sigma, \delta_N, q_0, F_N)$$

costruiremo un DFA

$$D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$$

tali che

$$L(D) = L(N).$$

I dettagli della costruzione per sottoinsiemi:

- $Q_D = \{S : S \subseteq Q_N\}$.

Nota: $|Q_D| = 2^{|Q_N|}$, anche se la maggior parte degli stati in Q_D sono "garbage", cioè non raggiungibili dallo stato iniziale.

- $F_D = \{S \subseteq Q_N : S \cap F_N \neq \emptyset\}$
- Per ogni $S \subseteq Q_N$ e $a \in \Sigma$,

$$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a)$$

Ad esempio l'insieme di stati $\{q_0, q_1\}$ nel nostro esempio diventa un singolo stato del DFA corrispondente, raggiungibile dallo stato $\{q_0\}$.

Costruiamo δ_D dall'NFA già visto, quello cioè che accetta le stringhe che terminano con 01.

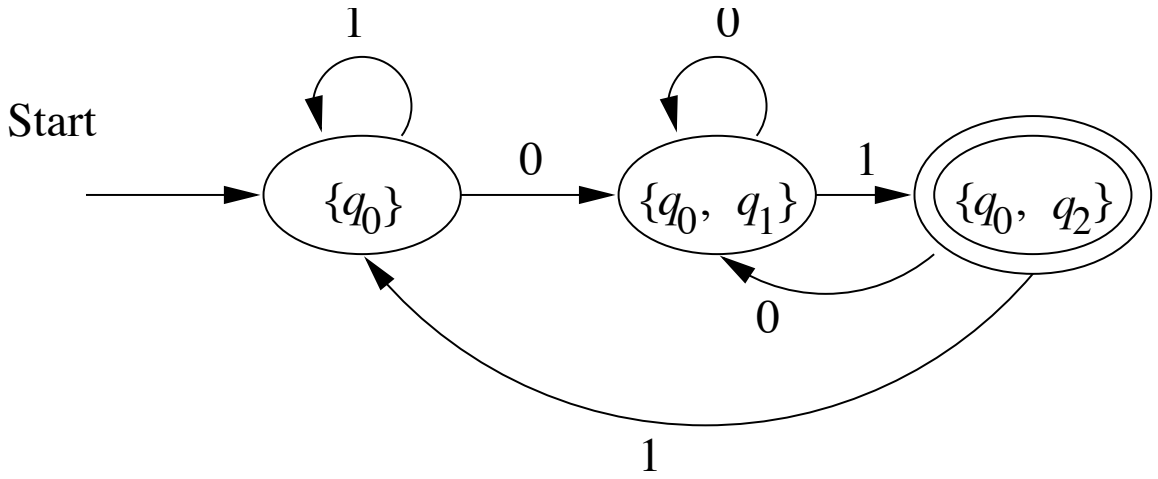
	0	1
$\emptyset$	$\emptyset$	$\emptyset$
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\emptyset$	$\{q_2\}$
$\star\{q_2\}$	$\emptyset$	$\emptyset$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$\star\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\star\{q_1, q_2\}$	$\emptyset$	$\{q_2\}$
$\star\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

Nota: Gli stati di D corrispondono a sottoinsiemi di stati di N, ma potevamo denotare gli stati di D in un altro modo, per esempio $A - F$.

	0	1
A	A	A
$\rightarrow B$	E	B
C	A	D
$\star D$	A	A
E	E	F
$\star F$	E	B
$\star G$	A	D
$\star H$	E	F

Per evitare la crescita esponenziale degli stati può essere utile costruire la tabella di transizione per D solo per gli stati raggiungibili (o accessibili) S come segue:
Base: $S = \{q_0\}$ è raggiungibile in D
Induzione: Se lo stato S è raggiungibile, lo sono anche gli stati in $\bigcup_{a \in \Sigma} \delta_D(S, a)$ raggiungibile a partire da S.

Esempio: Il "sottoinsieme" DFA con i soli stati raggiungibili: B ($\{q_0\}$), E ($\{q_0, q_1\}$) e F ($\{q_0, q_2\}$).



Teorema 2.11:

Teorema

Sia D il DFA ottenuto da un NFA N con la costruzione a sottoinsiemi. Allora $L(D) = L(N)$.

Dimostrazione: Prima mostriamo per induzione su $|w|$ che

$$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$$

N.B. Entrambe le funzioni restituiscono un insieme di stati di Q_N.
Base: $w = \epsilon$. Per definizione $\hat{\delta}_D(\{q_0\}, \epsilon) = \hat{\delta}_N(q_0, \epsilon) = \{q_0\}$.

Induzione: $w = xa$, con $|x| = n$ e $a \in \Sigma$. Per ipotesi induttiva $\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x)$ che supponiamo uguali a $\{p_1, ..., p_k\}$.

$$\begin{aligned}
\hat{\delta}_N(q_0, xa) &\stackrel{\text{def}}{=} \delta_N(\hat{\delta}_N(q_0, x), a) \\
&\stackrel{\text{i.h.}}{=} \delta_N(\{p_1, ..., p_k\}, a) \\
&\stackrel{\text{def}}{=} \bigcup_{p_i \in \hat{\delta}_N(q_0, x)} \delta_N(p_i, a)
\end{aligned}$$

$$\begin{aligned}
\hat{\delta}_D(\{q_0\}, xa) &\stackrel{\text{def}}{=} \delta_D(\hat{\delta}_D(\{q_0\}, x), a) \\
&\stackrel{\text{i.h.}}{=} \delta_D(\{p_1, ..., p_k\}, a) \\
&\stackrel{\text{cst}}{=} \bigcup_{p_i \in \hat{\delta}_N(q_0, x)} \delta_N(p_i, a)
\end{aligned}$$

- Quindi $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$.
- Osservando inoltre che sia D che N accettano w se e solo se $\hat{\delta}_D(\{q_0\}, w)$ e $\hat{\delta}_N(q_0, w)$ contengono uno stato in F_N, ne

Teorema 2.12: Un linguaggio L è accettato da un DFA se e solo se L è accettato da un NFA.

Dimostrazione: La parte "se" è il Teorema 2.11 più la costruzione per sottoinsiemi.

Per la parte "solo se" notiamo che un qualsiasi DFA può essere convertito in un NFA equivalente modificando la δ_D in δ_N secondo la regola seguente:

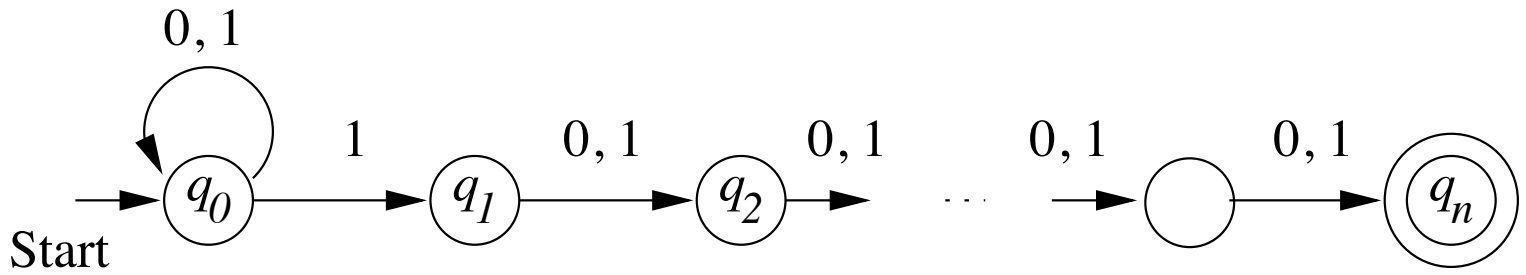
- Se $\delta_D(q, a) = p$, allora $\delta_N(q, a) = \{p\}$.

Per induzione su $|w|$ si può facilmente mostrare che se $\hat{\delta}_D(q_0, w) = p$, allora $\hat{\delta}_N(q_0, w) = \{p\}$.

Di conseguenza la stringa w viene accettata da D se e solo se viene accettata da N e l'enunciato del teorema segue.

Crescita esponenziale degli stati

Esiste un NFA N con $n+1$ stati che non ha nessun DFA equivalente con meno di 2^n stati



$$L(N) = \{x1c_1c_2\cdots c_{n-1} : x \in \{0,1\}^*, c_i \in \{0,1\}\}$$

D deve ricordare gli ultimi n simboli che ha letto.
Dato che ci sono 2^n sequenze di n bit, se esistesse un DFA equivalente con meno di 2^n stati, allora esisterebbe uno stato q e due sequenze $a_1a_2\cdots a_n$ e $b_1b_2\cdots b_n$, con almeno $i : a_i \neq b_i$: $q \in \hat{\delta}_N(q_0, a_1a_2\cdots a_n), q \in \hat{\delta}_N(q_0, b_1b_2\cdots b_n), a_1a_2\cdots a_n \neq b_1b_2\cdots b_n$

Caso 1: $i = 1$

$1a_2 \cdots a_n$

$0b_2 \cdots b_n$

Allora q deve essere sia uno stato di accettazione che uno stato di non accettazione.

Caso 2: $i > 1$

$a_1 \cdots a_{i-1}1a_{i+1} \cdots a_n$

$b_1 \cdots b_{i-1}0b_{i+1} \cdots b_n$

Consideriamo ora lo stato p raggiunto dopo aver letto $i-1$ simboli 0 a partire da q.

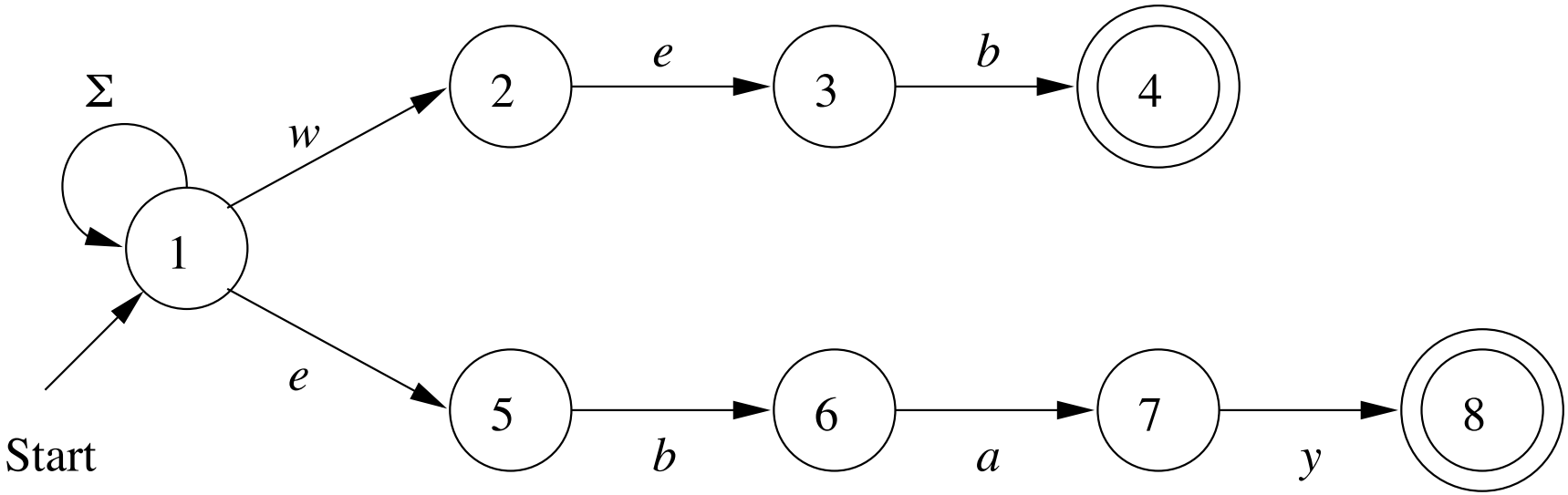
$\hat{\delta}_N(q_0, a_1 \cdots a_{i-1}1a_{i+1} \cdots a_n 0^{i-1}) = \hat{\delta}_N(q_0, b_1 \cdots b_{i-1}0b_{i+1} \cdots b_n 0^{i-1})$

Allora p deve essere sia uno stato di accettazione che uno stato di non accettazione.

$$\hat{\delta}_N(q_0, a_1 \cdots a_{i-1}1a_{i+1} \cdots a_n 0^{i-1}) \in F_D$$
$$\hat{\delta}_N(q_0, b_1 \cdots b_{i-1}0b_{i+1} \cdots b_n 0^{i-1}) \notin F_D$$

NFA per ricerche testuali

Un NFA che accetta l'insieme di parole chiave {`ebay`, `web`}



- Naturalmente l'NFA va poi implementato, passando all'equivalente DFA (che in questo caso ha gli stessi stati).
- Esempio di come derivare un programma (che implementa il DFA) a partire dalla specifica (quella dell'NFA)

Ancora esercizi sui DFA

- Definire un DFA che accetti, sull'alfabeto $\Sigma = \{0, 1\}$, i linguaggi dati dai seguenti insiemi:
  - quello di tutte le stringhe che contengono un numero pari di 0;
  - quello di tutte le stringhe che contengono un numero pari di 1;
  - quello di tutte le stringhe che non contengono mai più di due 0 consecutivi (ovvero che non contengono mai 000 come sottostringa);
- Dimostrare che $\hat{\delta}(q, xy) = \hat{\delta}(\hat{\delta}(q, x), y)$ per qualunque stato q e qualunque coppia di stringhe x e y.
  Si consiglia di usare l'induzione sulla lunghezza della stringa y.
- Dimostrare che $\hat{\delta}(q, ax) = \hat{\delta}(\delta(q, a), x)$ per qualunque stato q, qualunque simbolo a e qualunque stringa x.
  Si consiglia di usare l'esercizio precedente.
- Definire un DFA (con $\Sigma = \{a, b, c, ..., z\}$) che accetti tutte le stringhe in cui le cinque vocali appaiono nell'ordine *a e i o u*
- Per questo come per gli altri argomenti attingere anche agli esercizi sul libro.

Esercizi sugli NFA

- Definire un NFA che accetti, sull'alfabeto $\Sigma = \{a, e, i, o, u\}$, i linguaggi dati dai seguenti insiemi:
  - quello di tutte le stringhe tali che la vocale finale sia apparsa in precedenza;
  - quello di tutte le stringhe tali che la vocale finale non sia apparsa in precedenza
- Definire un NFA che accetti sull'alfabeto $\Sigma = \{a, b\}$, il seguente insieme di stringhe: *abab*, *bab* e *abb*
- Convertire l'NFA ottenuto nel DFA corrispondente
- Definire un NFA che, sull'alfabeto $\Sigma = \{0, 1\}$, riconosca tutte le stringhe in cui compare la sequenza 011
- Dimostrare che l'NFA così definito accetta il linguaggio richiesto