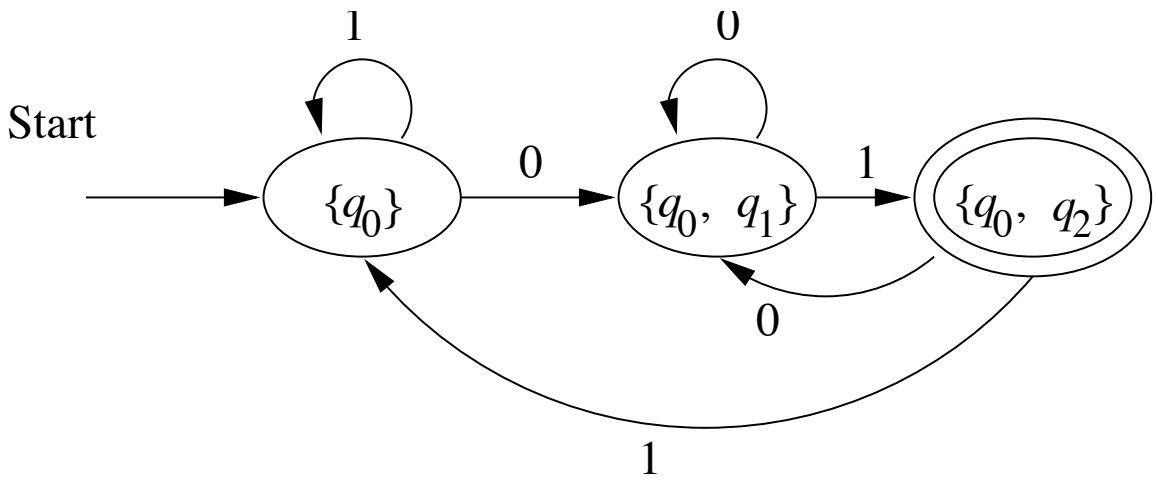
Per evitare la crescita esponenziale degli stati può essere utile costruire la tabella di transizione per D solo per gli stati raggiungibili (o accessibili) S come segue:
Base: $S = \{q_0\}$ è raggiungibile in D
Induzione: Se lo stato S è raggiungibile, lo sono anche gli stati in $\bigcup_{a \in \Sigma}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

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:

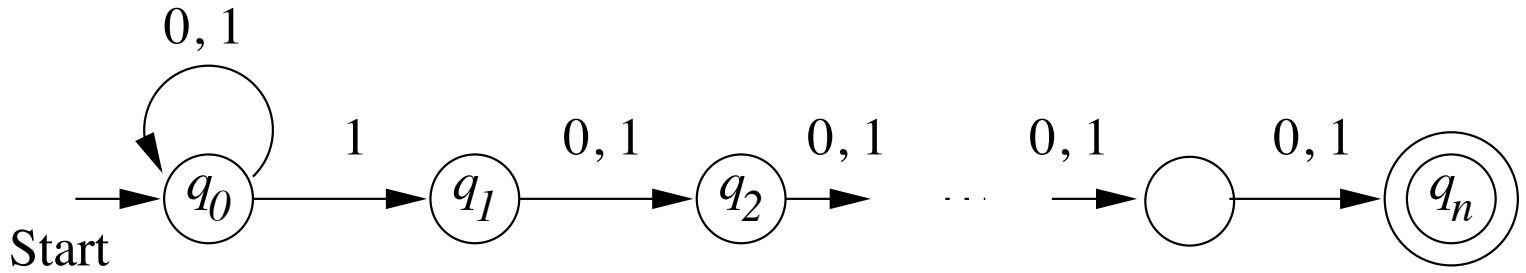
- Se $\delta_D(q, a) = p$, allora $\delta_N(q, a) = \{p\}$.

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.

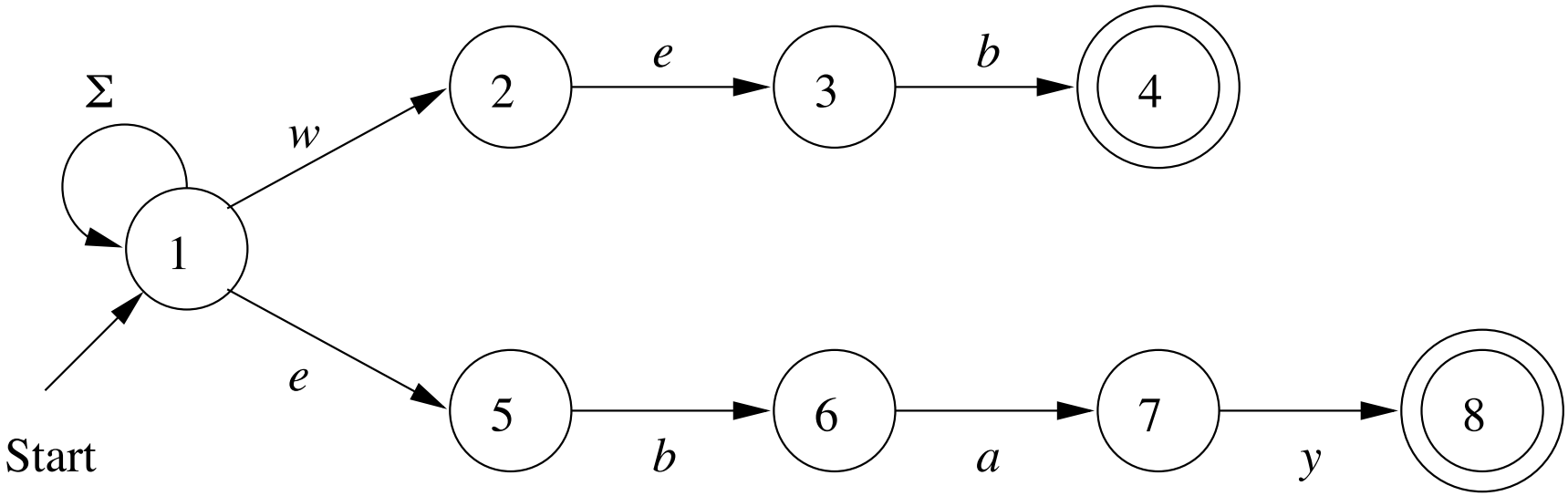
$\hat{\delta}_N(q_0, a_1 \cdots a_{i-1}1a_{i+1} \cdots a_n 0^{i-1}) = \hat{\delta}_N(q_0, b_1 \cdots b_{i-1}0b_{i+1} \cdots b_n 0^{i-1})$

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)

NFA per riconoscere numeri decimali

Vogliamo un NFA che accetta numeri decimali. Un numero decimale consiste di:

1. Un segno + o -, **opzionale**
2. Una stringa di cifre decimali
3. un punto decimale
4. un'altra stringa di cifre decimali

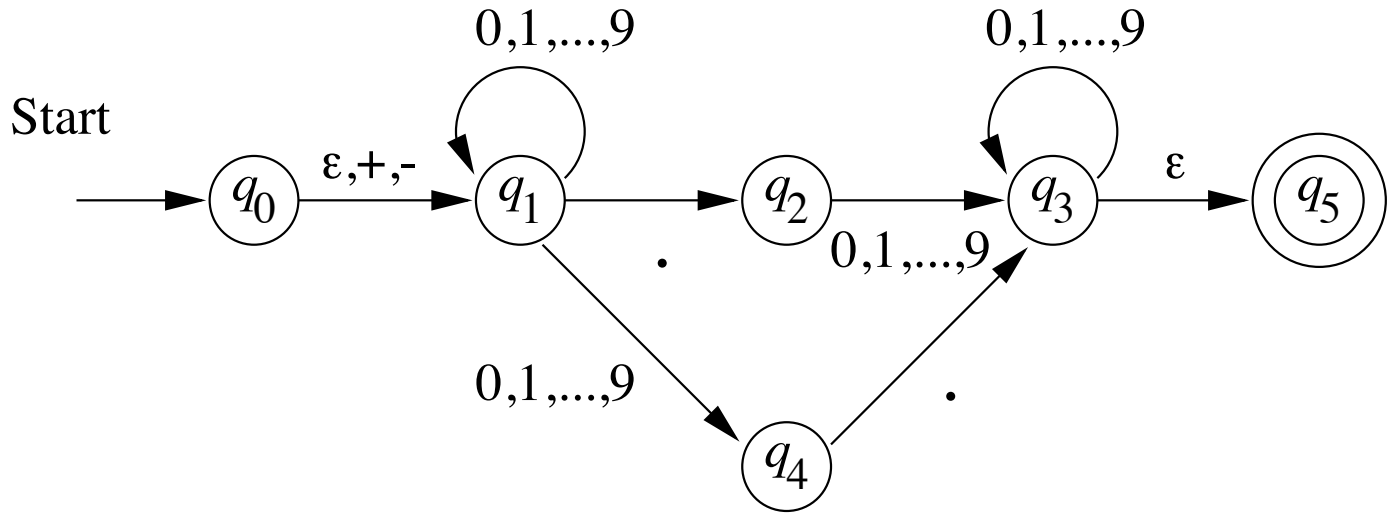
Una delle stringhe (2) e (4), ma non entrambi, può essere vuota. Cosa succede se non compare il segno?

NFA per riconoscere numeri decimali

Vogliamo un NFA che accetta numeri decimali consiste di:

1. Un segno + o -, **opzionale**
2. Una stringa di cifre decimali
3. un punto decimale
4. un'altra stringa di cifre decimali

Una delle stringhe (2) e (4), ma non entrambi, può essere vuota.
Cosa succede se non compare il segno? Uso un ϵ-NFA.



Definizione ed esempio

Un ϵ-NFA è una quintupla $(Q, \Sigma, \delta, q_0, F)$ dove:

- $\delta : Q \times \Sigma \cup \{\epsilon\} \to 2^Q$ è una funzione da $Q \times \Sigma \cup \{\epsilon\}$ all'insieme dei sottoinsiemi di Q.

Esempio: L'ϵ-NFA della pagina precedente

$$E = (\{q_0, q_1, \ldots, q_5\}, \{., +, -, 0, 1, \ldots, 9\}\ \delta, q_0, \{q_5\})$$

dove la tabella delle transizioni (con una colonna per ϵ) per δ è

	ϵ	+,-	.	$0, \ldots, 9$
$\to q_0$	$\{q_1\}$	$\{q_1\}$	$\emptyset$	$\emptyset$
q_1	$\emptyset$	$\emptyset$	$\{q_2\}$	$\{q_1, q_4\}$
q_2	$\emptyset$	$\emptyset$	$\emptyset$	$\{q_3\}$
q_3	$\{q_5\}$	$\emptyset$	$\emptyset$	$\{q_3\}$
q_4	$\emptyset$	$\emptyset$	$\{q_3\}$	$\emptyset$
$\star q_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Epsilon-chiusura

ECLOSE(q) = gli stati raggiungibili da q tramite una sequenza di ϵ-transizioni

Definizione induttiva di ECLOSE(q)

Base:

$q \in$ ECLOSE(q)

Induzione:

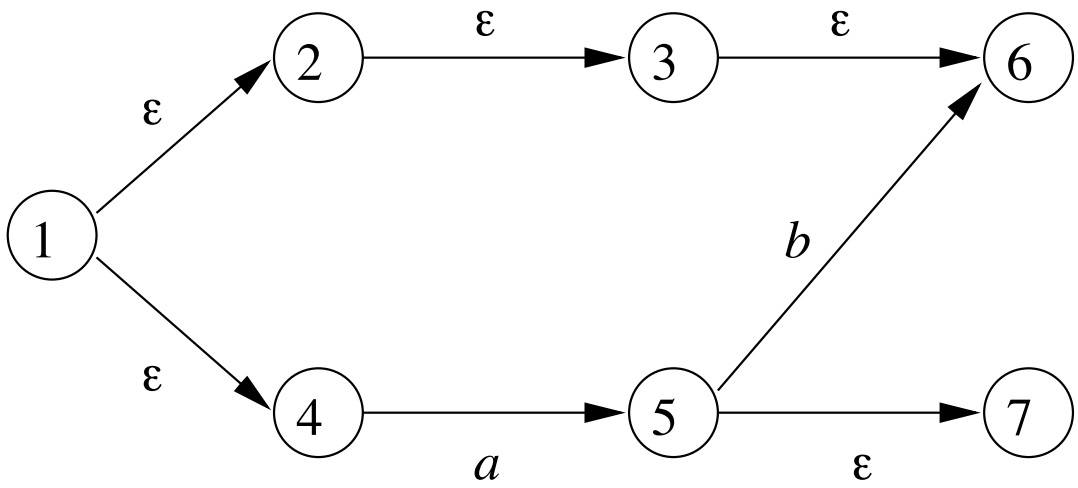
$p \in$ ECLOSE(q) and $r \in \delta(p, \epsilon) \Rightarrow r \in$ ECLOSE(q)

Nell'esempio di prima ogni stato coincide con la propria epsilon-chiusura, tranne ECLOSE(q_0) = $\{q_0, q_1\}$ e ECLOSE(q_3) = $\{q_3, q_5\}$ che corrispondono alle due ϵ-transizioni.

La stessa operazione si può applicare ad un insieme di stati:

$$\text{ECLOSE}(S) = \bigcup_{q \in S} \text{ECLOSE}(q)$$

Esempio di epsilon-chiusura



Per esempio,

$$\text{ECLOSE}(1) = \{1, 2, 3, 4, 6\}$$

Ognuno di questi stati può essere raggiunto a partire da 1 attraverso ϵ-transizioni.

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

Intuitivamente $\hat{\delta}(q, w)$ è l'insieme degli stati raggiungibili da q seguendo il percorso w (con possibili ϵ-transizioni)

- Definizione induttiva di $\hat{\delta}$ per automi ϵ-NFA

Base:

$$\hat{\delta}(q, \epsilon) = \text{ECLOSE}(q)$$

Induzione: $a \neq \epsilon$

Se

- $\hat{\delta}(q, x) = \{p_1, ..., p_k\}$;
- $\bigcup_{p_i \in \hat{\delta}(q,x)} \delta(p_i, a) = \{r_1, ..., r_m\}$

$$\hat{\delta}(q, xa) = \text{ECLOSE}(\bigcup_{p_i \in \hat{\delta}(q,x)} \delta(p_i, a)) = \text{ECLOSE}\{r_1, ..., r_m\}$$

Il linguaggio di un ϵ-NFA E è dato da:

$$L(E) = \{w | \hat{\delta}(q_0, w) \cap F_E \neq \emptyset\}$$

Da ϵ-NFA a DFA

Procedimento simile alla costruzione per sottoinsiemi: vanno incorporate le ϵ-transizioni, usando le ϵ-chiusure.
Dato un ϵ-NFA

$$E = (Q_E, \Sigma, \delta_E, q_0, F_E)$$

costruiremo un DFA

$$D = (Q_D, \Sigma, \delta_D, q_D, F_D)$$

tale che

$$L(D) = L(E)$$

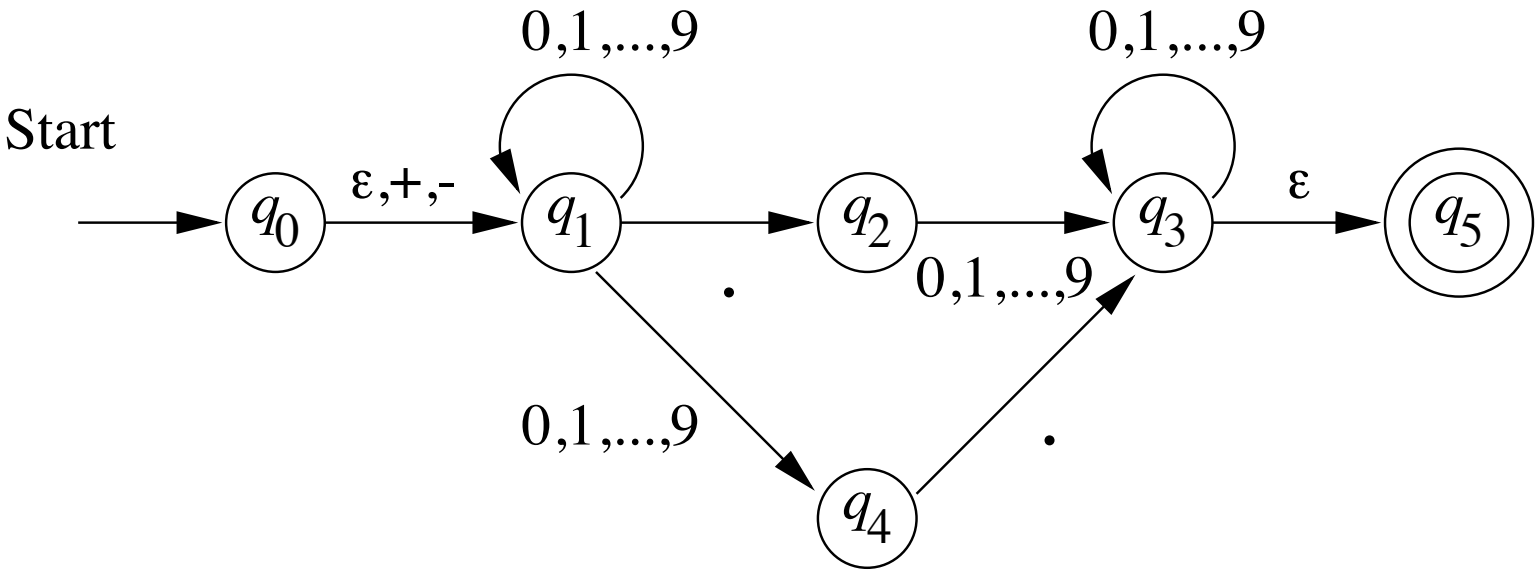
Dettagli della costruzione:

- $Q_D = \{S : S \subseteq Q_E \text{ e } S = \text{ECLOSE}(S)\}$
- $q_D = \text{ECLOSE}(q_0)$
- $F_D = \{S : S \in Q_D \text{ e } S \cap F_E \neq \emptyset\}$
- $\delta_D(S, a) =$

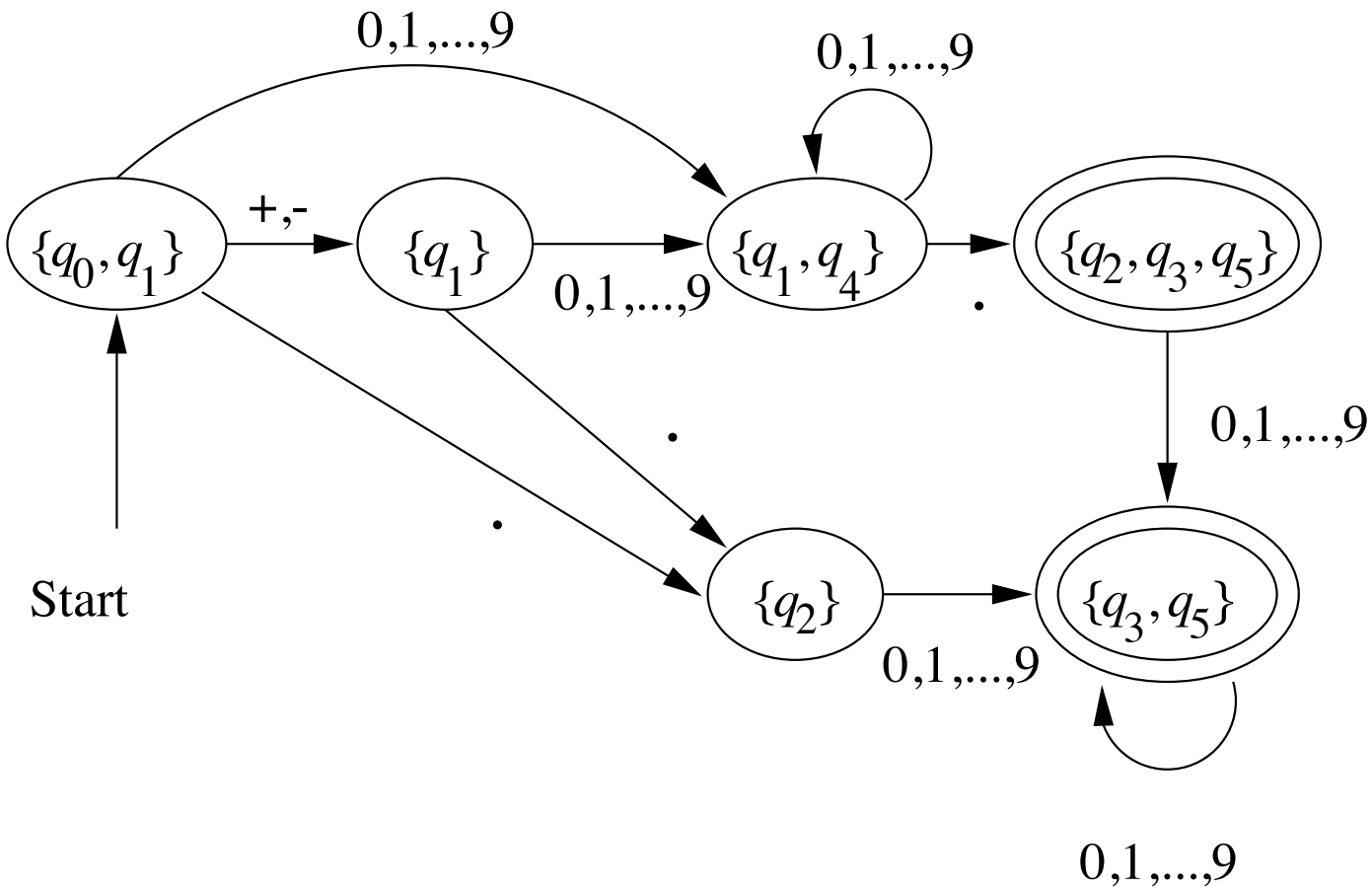
$$\bigcup \{\text{ECLOSE}(p) : p \in \delta(t, a) \text{ per alcuni } t \in S\}$$

Esempio

ϵ-NFA E per riconoscere numeri decimali in notazione anglo-sassone



DFA D corrispondente ad E



Teorema 2.22: Un linguaggio L è accettato da un ϵ-NFA E se e solo se L è accettato da un DFA.
Dimostrazione: Usiamo D costruito come sopra e mostriamo per induzione che $\hat{\delta}_E(q_0, w) = \hat{\delta}_D(q_D, w)$

Base: $\hat{\delta}_E(q_0, \epsilon) = \text{ECLOSE}(q_0) = q_D = \hat{\delta}(q_D, \epsilon)$

Induzione: $a \neq \epsilon$ Sia $w = xa$ con l'ipotesi valida per x, ovvero $\hat{\delta}_E(q_0, x) = \hat{\delta}_D(q_0, x) = \{p_1, ..., p_k\}$. Per la definizione di $\hat{\delta}_E$, $\hat{\delta}_E(q_0, w)$ si calcola come segue:

- sia $\bigcup_{p_i \in \hat{\delta}_E(q_0, x)} \delta(p_i, a) = \{r_1, ..., r_m\}$,
- allora

$$\hat{\delta}_E(q, xa) = \text{ECLOSE}\Big(\bigcup_{p_i \in \hat{\delta}(q, x)} \delta(p_i, a)\Big) = \text{ECLOSE}\,\{r_1, ..., r_m\}$$

Ricapitolando

- Tutti i tipi di automi visti, DFA, NFA e ϵ-NFA, accettano lo stesso insieme di linguaggi: i linguaggi regolari.
- Solo i DFA possono tuttavia essere implementati!

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