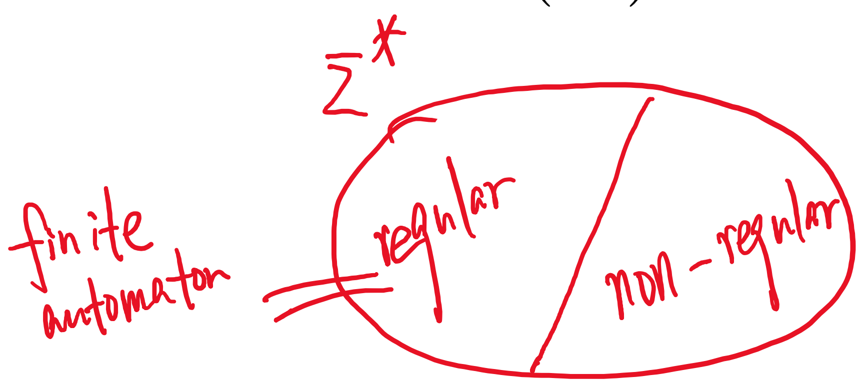
Finite Automaton

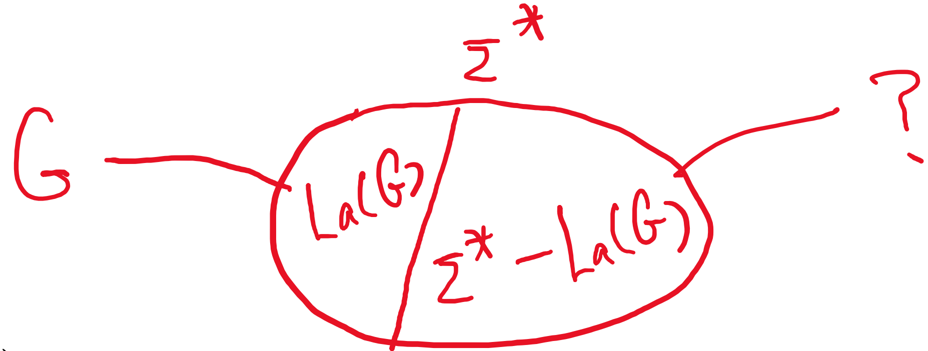
Regular languages

It is not always possible to design
a finite automaton $\mathbf{G}$ to recognize a given L

Defn. A language $L \subseteq \Sigma^*$ is a *regular language*
if L can be recognized by a finite automaton
i.e. there exists $\mathbf{G}$ s.t. $L_a(\mathbf{G}) = L$



Complement



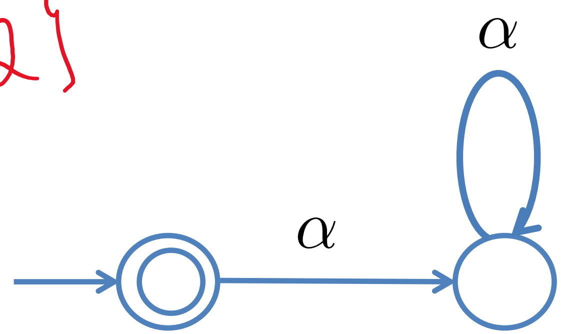
Let $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$ be a finite automaton that recognizes the language $L_a(\mathbf{G}) \subseteq \Sigma^*$.

What is the finite automaton that recognizes $\Sigma^* - L_a(\mathbf{G})$?

Example

$\Sigma = \{a\}$

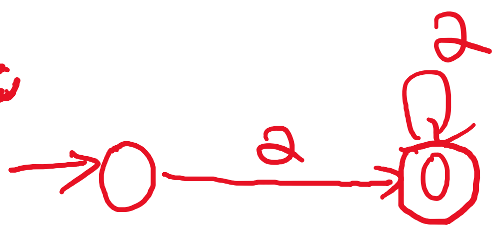
$\mathbf{G}$



$L_a(\mathbf{G}) = \{\varepsilon\}$

$\Sigma^* - L_a(G) = \Sigma^* - \{\varepsilon\} \; (= \Sigma^+)$

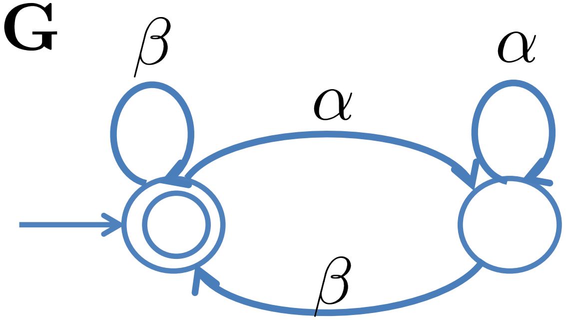
G_c



$L_a(G_c) = \Sigma^* - L_a(G)$

Example

$\mathbf{G}$

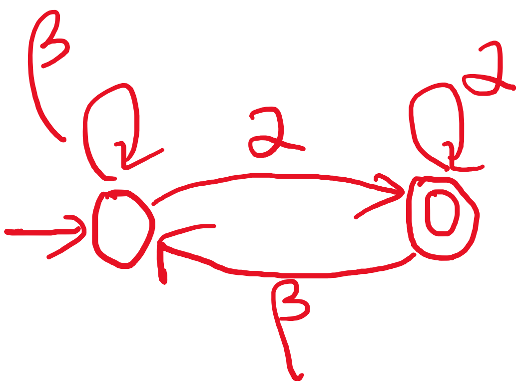


$$L_a(\mathbf{G}) = \{\varepsilon, \beta^*, \beta^*\alpha\alpha^*\beta, \beta^*\alpha\alpha^*\beta\beta^*, \ldots\}$$

$$\Sigma^* - L_a(G)$$

$$= L_a(G_c)$$

G_c



Complement

Let $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_a)$ be a finite automaton that recognizes the language $L_a(\mathbf{G}) \subseteq \Sigma^*$.

What is the finite automaton that recognizes $\Sigma^* - L_a(\mathbf{G})$?

Answer: complement of $\mathbf{G}$:

$$\mathbf{G}_{co} = (Q, \Sigma, \delta, q_0, Q - Q_a)$$

Product of automata

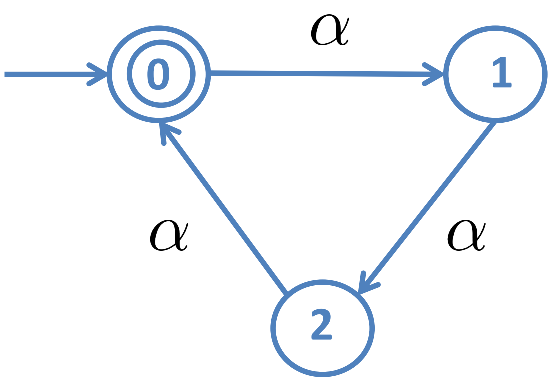
Let $\mathbf{G}_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, Q_{a,1})$ and $\mathbf{G}_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, Q_{a,2})$ be two finite automata that recognize $L_a(\mathbf{G}_1)$, $L_a(\mathbf{G}_2)$.

What is the finite automaton that recognizes $L_a(\mathbf{G}_1) \cap L_a(\mathbf{G}_2)$?

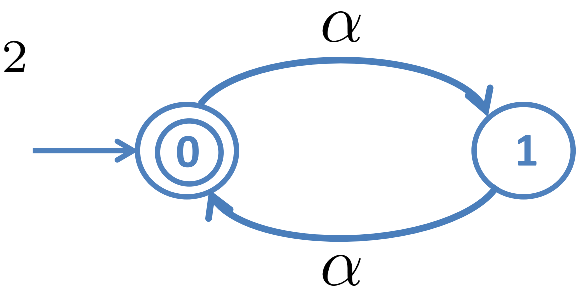
Example

$\Sigma = \{a\}$

$\mathbf{G}_1$



$\mathbf{G}_2$

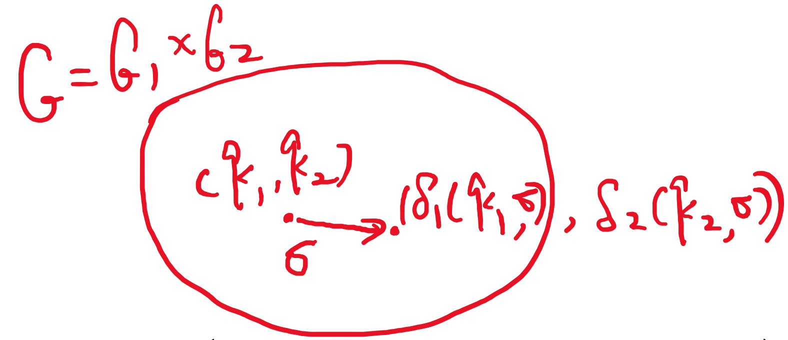


$$L_a(\mathbf{G}_1) = \{\varepsilon, a^3, a^6, \ldots\}$$
$$= \{a^{3n} \mid n = 0, 1, 2, \ldots\}$$

$$L_a(\mathbf{G}_2) = \{a^{2n} \mid n = 0, 1, 2, \ldots\}$$

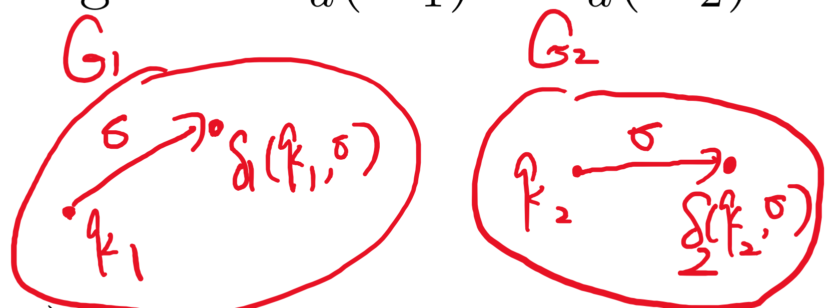
$$L_a(\mathbf{G}_1) \cap L_a(\mathbf{G}_2) = \{a^{6n} \mid n = 0, 1, 2, \ldots\}$$

Product of automata



Let $\mathbf{G}_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, Q_{a,1})$ and $\mathbf{G}_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, Q_{a,2})$ be two finite automata that recognize $L_a(\mathbf{G}_1)$, $L_a(\mathbf{G}_2)$.

What is the finite automaton that recognizes $L_a(\mathbf{G}_1) \cap L_a(\mathbf{G}_2)$?



Answer: product of $\mathbf{G}_1$ and $\mathbf{G}_2$:

$$\boxed{\mathbf{G}_1 \times \mathbf{G}_2 = (Q, \Sigma, \delta, q_0, Q_a)}$$

where $Q = Q_1 \times Q_2$, $q_0 = (q_{0,1}, q_{0,2})$, $Q_a = Q_{a,1} \times Q_{a,2}$

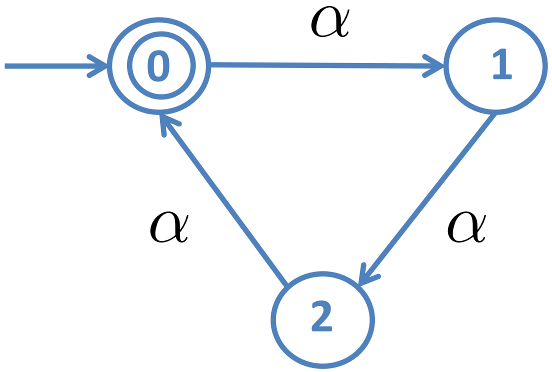
$$\delta : (Q_1 \times Q_2) \times \Sigma \to (Q_1 \times Q_2)$$

$$\delta((q_1, q_2), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)), & \text{if } \delta_1(q_1, \sigma)! \ \& \ \delta_2(q_2, \sigma)! \\ \text{undefined}, & \text{otherwise} \end{cases}$$

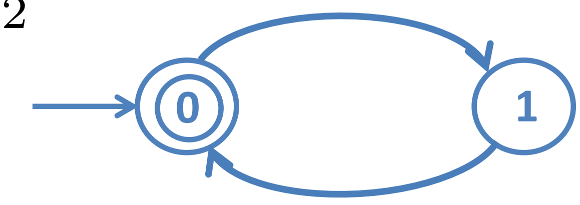
$q_1 \in Q_1$, $q_2 \in Q_2$, $\sigma \in \Sigma$

Example

$\mathbf{G}_1$

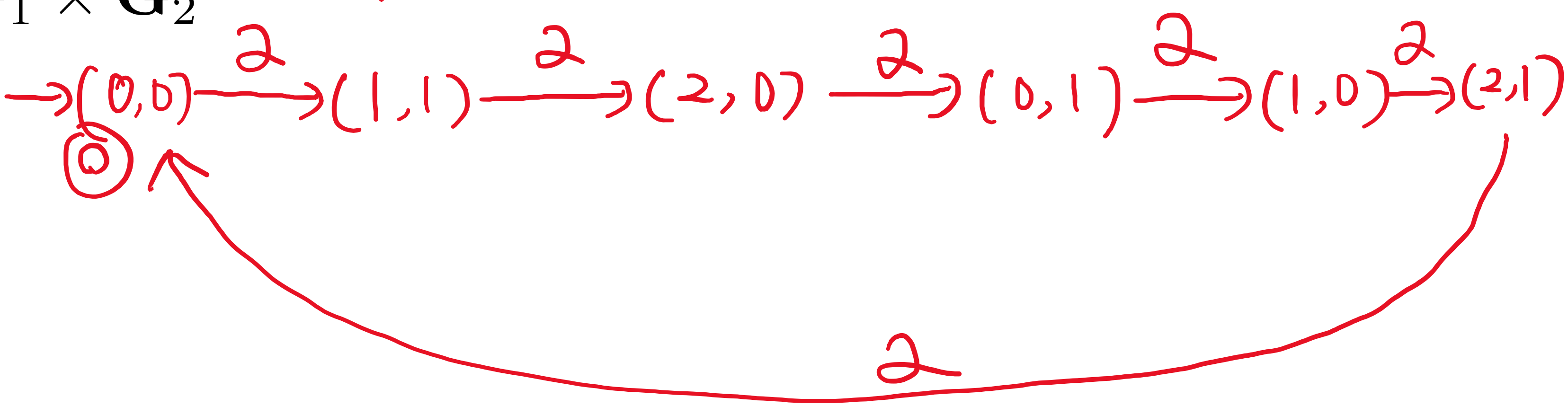


$\mathbf{G}_2$



$$L_a(G_1 \times G_2) = \{\alpha^{6n} \mid n = 0, 1, 2, \dots\} = L_a(G_1) \cap L_a(G_2)$$

$\mathbf{G}_1 \times \mathbf{G}_2$



“Union” of automata

Let $\mathbf{G}_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, Q_{a,1})$ and $\mathbf{G}_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, Q_{a,2})$ be two finite automata that recognize $L_a(\mathbf{G}_1)$, $L_a(\mathbf{G}_2)$.

What is the finite automaton that recognizes $L_a(\mathbf{G}_1) \cup L_a(\mathbf{G}_2)$?

Answer: By De Morgan's Law:

$$L_a(G_1) \cup L_a(G_2) = \Sigma^* - \Big[\underbrace{\left(\Sigma^* - L_a(G_1)\right)}_{\text{complement}} \cap \underbrace{\left(\Sigma^* - L_a(G_2)\right)}_{\text{complement}} \Big]$$

product

complement

Nondeterministic Finite Automaton

So far

In a finite automaton $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_m)$

for each state $q \in Q$ and each symbol $\sigma \in \Sigma$

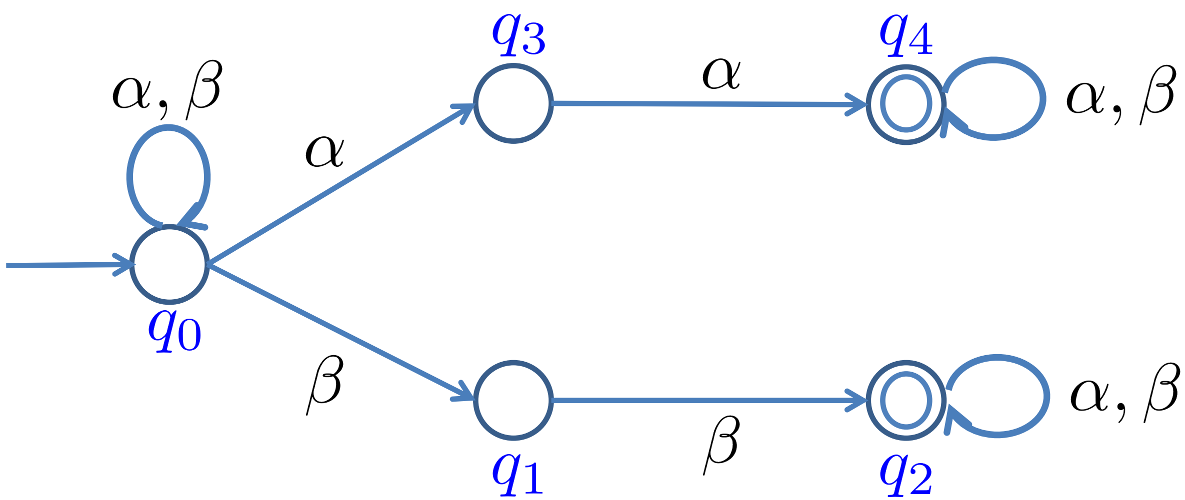
$$q \xrightarrow{\sigma} q'$$

How about:

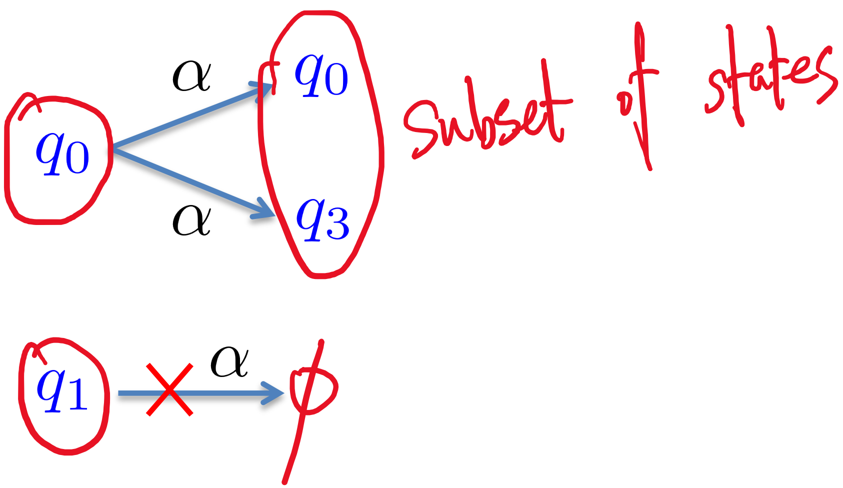
$$\begin{tikzcd} q \arrow[r, "\sigma"] \arrow[dr, "\sigma"'] & q' \\ & q'' \end{tikzcd}$$

$$q \not\xrightarrow{\sigma}$$

Example

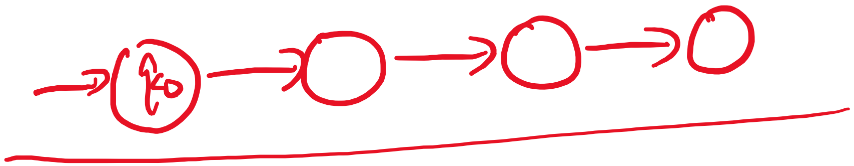


q_3
α
q_4
α, β
α, β
α
q_0
β
q_1
β
q_2
α, β

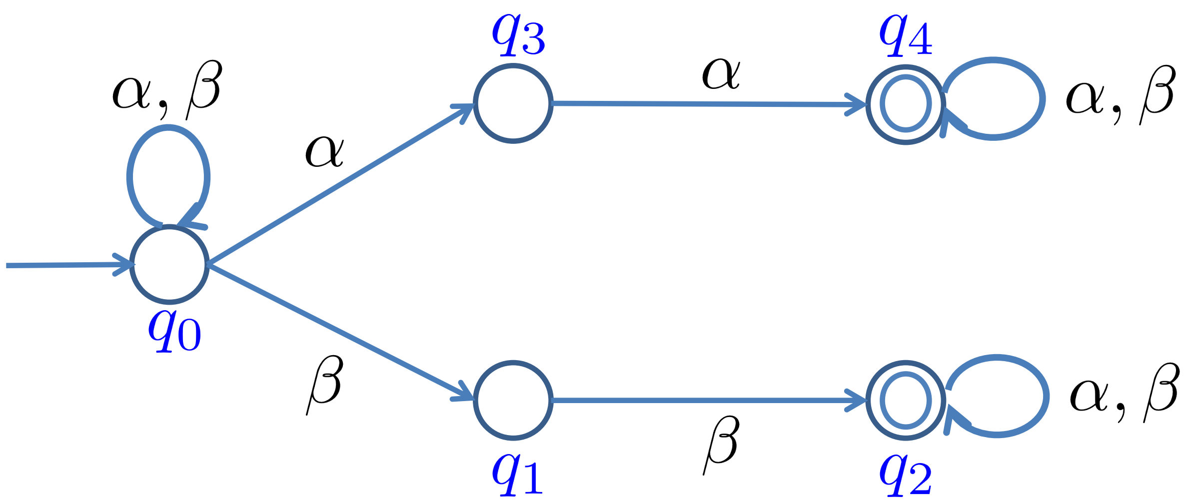


α
q_0
q_0
α
q_3
subset of states
q_1
α
ϕ

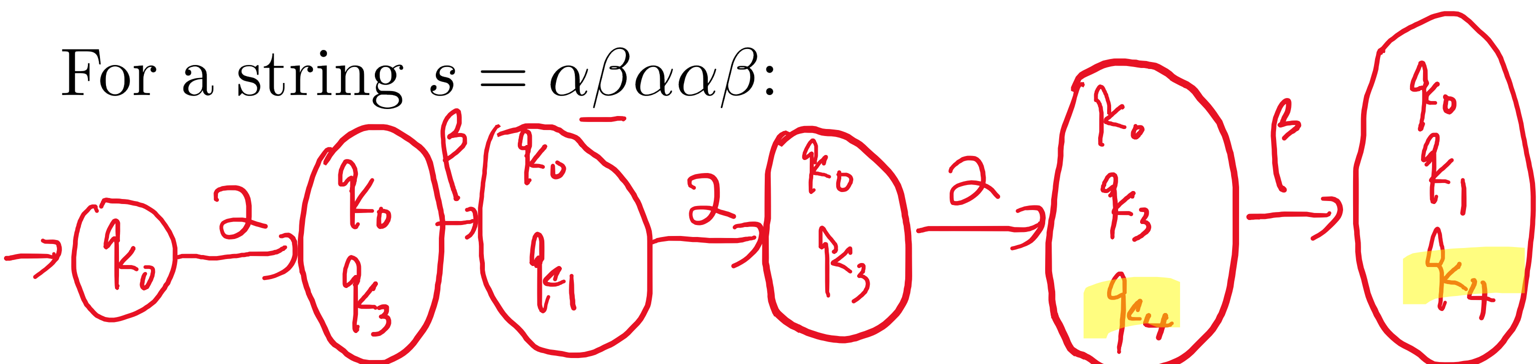
Example



state transition



For a string $s = \alpha\underline{\beta}\alpha\alpha\beta$:

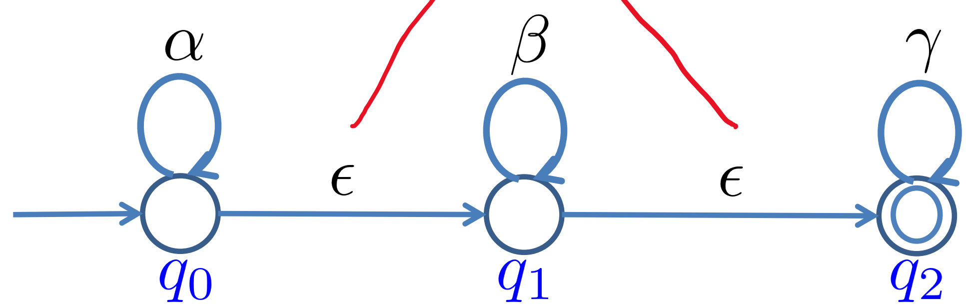


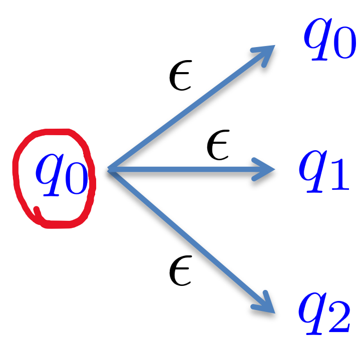
state subset transition

Example

silent transition

ϵ-transitions





$$\text{SILENT}(q_0) = \{q_0, q_1, q_2\}$$

$\text{SILENT}(q_1) = \{q_1, q_2\}$

$\boxed{\delta : Q \times \Sigma \to Q}$ $\quad q_0 \xrightarrow{\delta} q_0'$

Nondeterministic finite automaton

A nondeterministic finite automaton (NFA) $\mathbf{G}$ is a 5-tuple $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_a)$, where

Q : state set; a finite set of states

Σ : alphabet; a finite set of symbols

$\delta : Q \times (\Sigma \cup \{\epsilon\}) \to Pwr(Q)$: state transition function

"Power set"

$q_0 \in Q$: initial state

$Q_a \subseteq Q$: subset of accept states

Call previous automata "deterministic finite automaton" (DFA)

$Pwr(S) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$

Aside: powerset

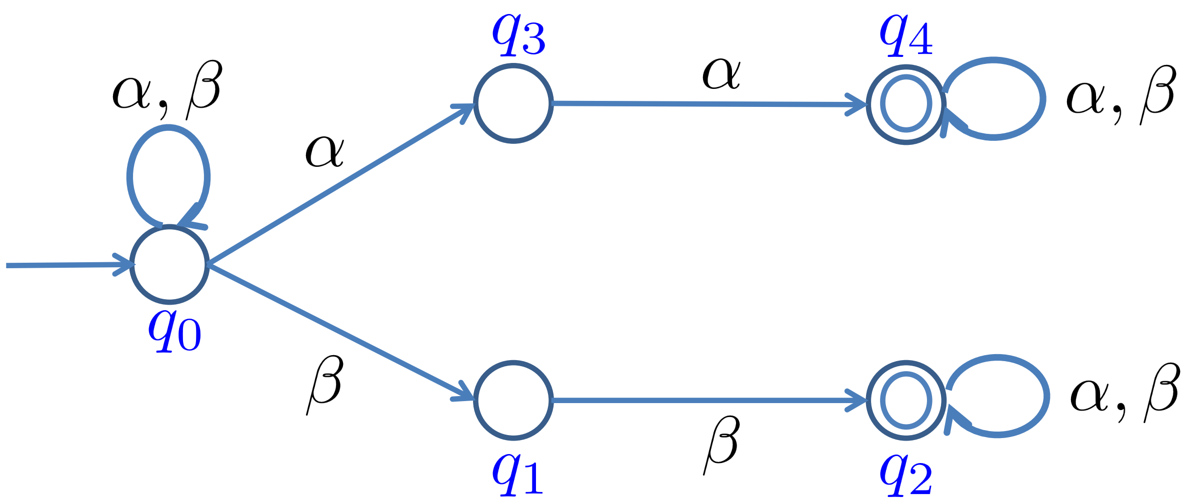
Let $S = \{1, 2, 3\}$.
What is $Pwr(S)$? is the set of subsets of S
How many elements in $Pwr(S)$? $8 = 2^3$

Let S be a fintie set of n elements.
Then $Pwr(S)$ has 2^n elements, thus "powerset".

Example

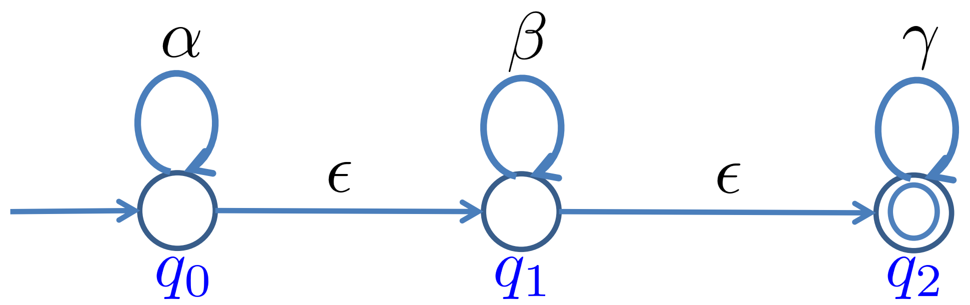
$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow Pwr(Q)$

$\delta(q_0, \beta) = \{q_0, q_1\}$



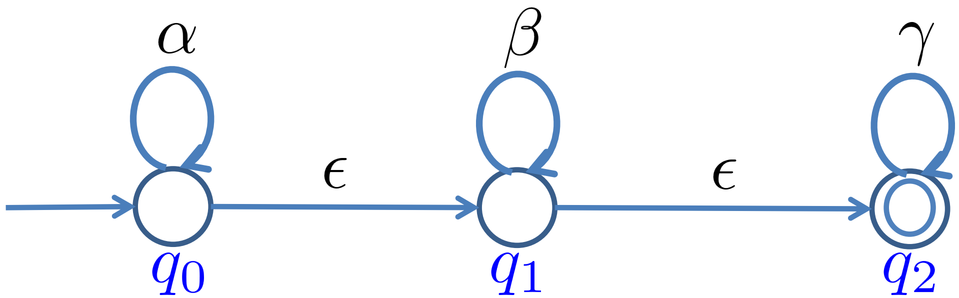
	α	β
q_0		
q_1		
q_2		
q_3		
q_4		

Example



	α	β	γ	ϵ
q_0				
q_1				
q_2				

Example



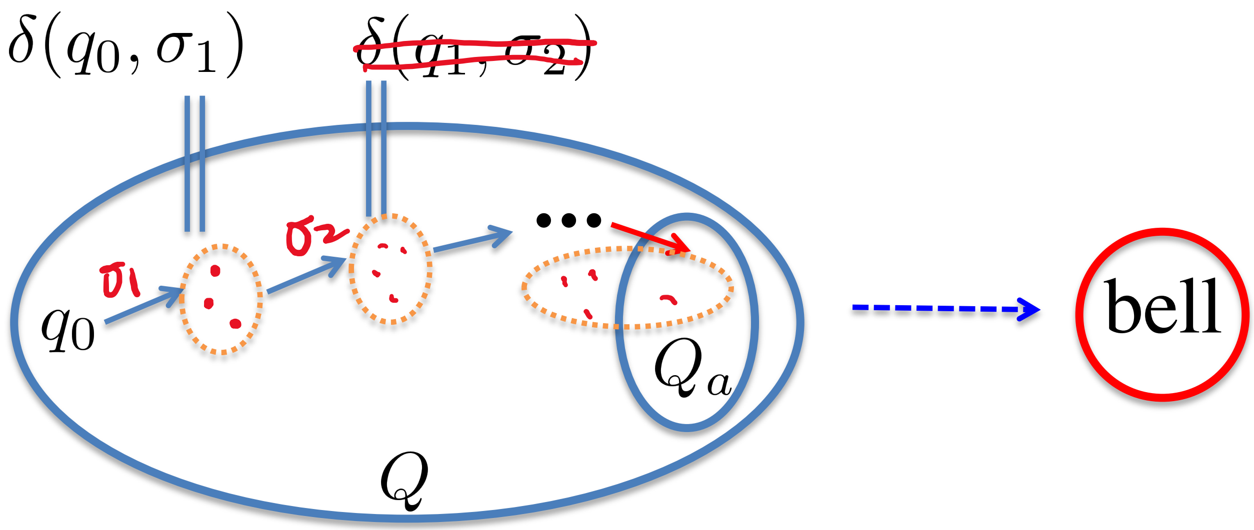
$\text{SILENT}(q_0) = \{q_0, q_1, q_2\}$

$$\text{SILENT}(q) = \{q' \in Q \mid \delta(q, \epsilon) = q'\} \in Pwr(Q)$$

$\subseteq Q$

How does an NFA work

An NFA $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_a)$

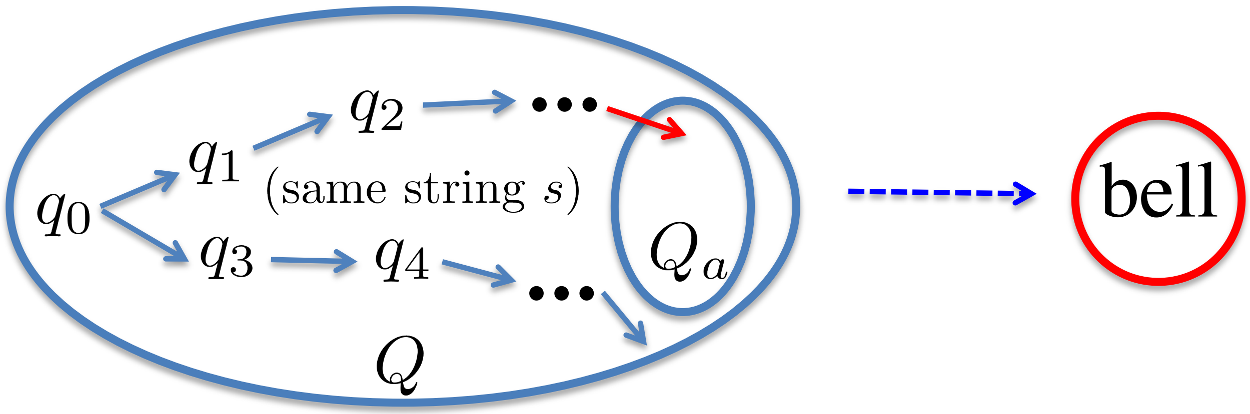


Internal state transitions

Beeps when a transition enters an accept state

How does an NFA work

An NFA $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_a)$



Internal state transitions

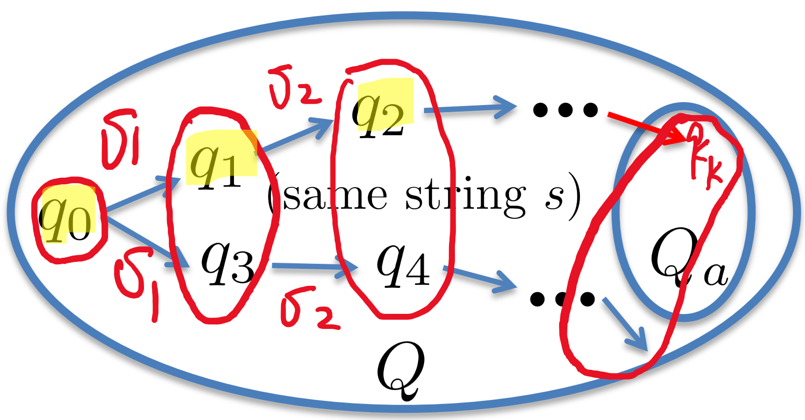
Beeps when a transition enters an accept state

Accepted/recognized strings

Let $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_a)$ be an NFA.

A string $s = \sigma_1 \cdots \sigma_k (\sigma_i \in \Sigma \cup \{\epsilon\})$ is accepted/recognized by $\mathbf{G}$ if there exists states $q_1, \dots, q_k \in Q$ s.t.

$$q_1 \in \delta(q_0, \sigma_1), \dots, q_k \in \delta(q_{k-1}, \sigma_k) \ \& \ q_k \in Q_a$$



Accepted/recognized languages

Let $\mathbf{G} = (Q, \Sigma, \delta, q_0, Q_a)$ be an NFA.

The language accepted/recognized by $\mathbf{G}$ is:

$$L_a(\mathbf{G}) = \{s \in (\Sigma \cup \{\epsilon\})^* \mid s \text{ recognized by } \mathbf{G}\}$$