# INT201 Decision, Computation and Language

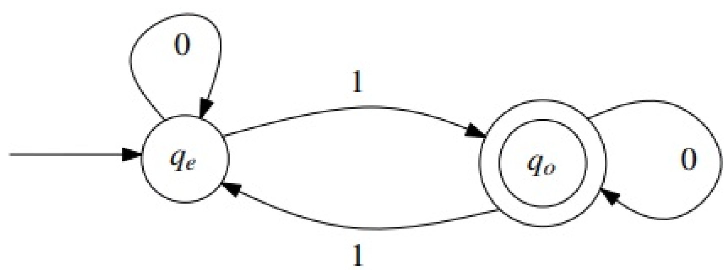
Lecture 3 – Nondeterministic Finite Automata

Dr Yushi Li

Xi'an Jiaotong-Liverpool University
西交利物浦大學

# Nondeterministic Finite Automata (NFA)

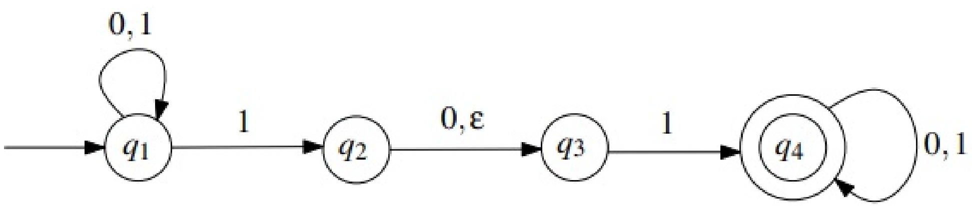
A finite automata is **deterministic**, if the next state the machine goes to on any given symbol is uniquely determined.




- DFA has exactly one transition leaving each state for each symbol.

# Nondeterministic Finite Automata (NFA)

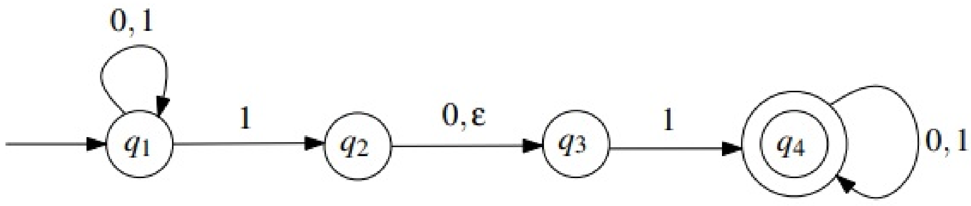
A finite automata is **nondeterministic**, if the machine allows for several or no choices to exist for the next state on a given symbol.




For a state $q$ and symbol $s \in \Sigma$, NFA can have:

- Multiple edges leaving $q$ labelled with the same symbol $s$;
- No edge leaving $q$ labelled with symbol $s$;
- Edge leaving $q$ labelled with $\epsilon$ (without reading any symbol).

# Nondeterministic Finite Automata (NFA)

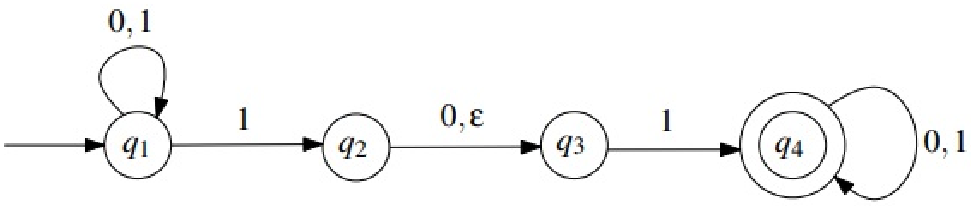



This NFA is in a state with multiple ways to proceed, e.g. state $q_1$ has two transition paths with $1$.

The machine splits into multiple copies of itself (threads):

- Each copy proceeds with computation independently of others.
- NFA may be in a set of states, instead of a single state.
- NFA follows all possible computation paths in parallel.
- If a copy is in a state and next input symbol doesn't appear on any outgoing edge from the state, then the copy dies or crashes.

# Nondeterministic Finite Automata (NFA)

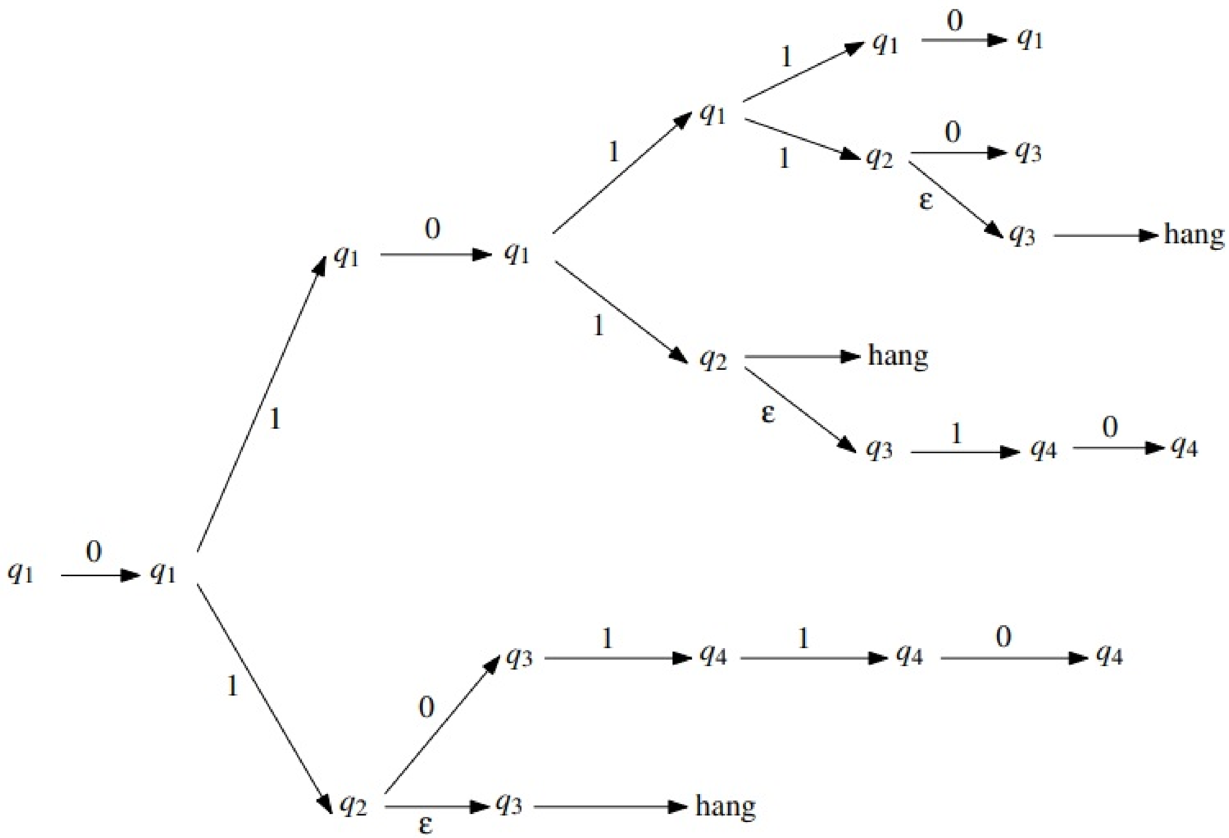



The NFA accepts the input string, if any copy ends in an accept state after reading the entire string.

The NFA rejects the input string, if no copy ends in an accept state after reading the entire string.

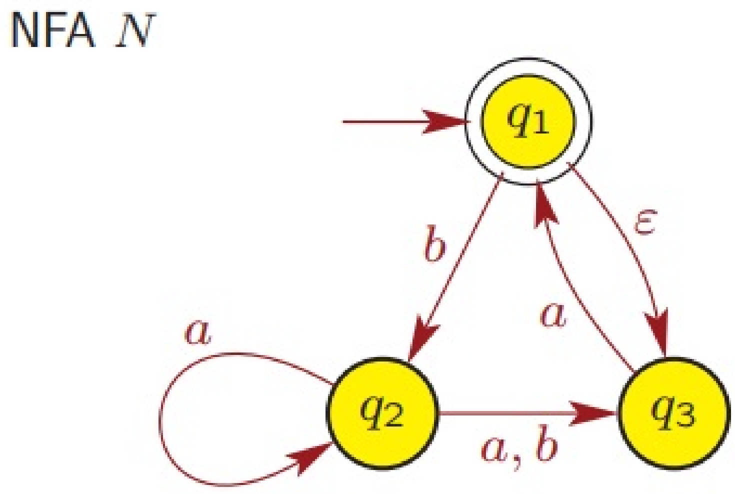
# Nondeterministic Finite Automata (NFA)

What can this automaton do when it gets the string 010110 as input?



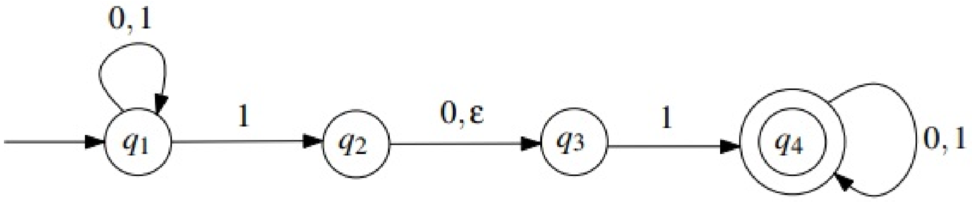
# Nondeterministic Finite Automata (NFA)

**Example**



# Nondeterministic Finite Automata (NFA)

**Question**




How about 010?

# Formal Definition of NFA

For any alphabet $\Sigma$, we define $\Sigma_\epsilon$ to be the set:

$$\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$$

Recall the notion of a power set: For any set $Q$, the power set of $Q$, denoted by $P(Q)$, is the set of all subsets of $Q$:

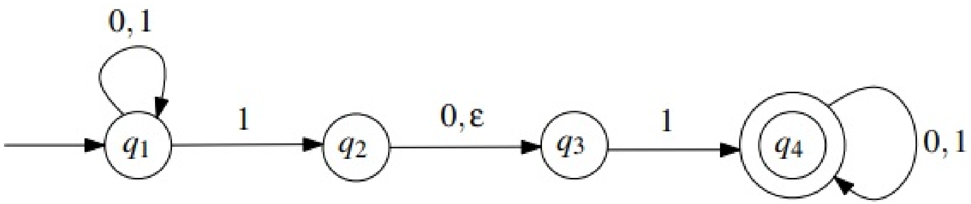
$$P(Q) = \{R : R \subseteq \mathrm{Q}\}$$

A **nondeterministic finite automaton** (NFA) is a 5-tuple $\mathrm{M} = (Q, \Sigma, \delta, q, F)$, where

1. $Q$ is a finite set of **states**,
2. $\Sigma$ is a finite set of symbols, called the **alphabet** of the automaton,
3. $\delta : Q \times \Sigma_\epsilon \rightarrow P(Q)$ is a function, called the **transition function**,
4. $q \in Q$ is called the **initial/start state**,
5. $F \subseteq Q$ is a set of **accepting/terminal states**.

# Formal Definition of NFA

**Example**




Formal description of above NFA M = ($Q$, $\Sigma$, $\delta$, $q$, $F$)

1. $Q = \{q_1, q_2, q_3, q_4\}$
2. $\Sigma = \{0, 1\}$
3. $\delta : Q \times \Sigma_\epsilon \to P(Q)$
4. $q_1$ is the start state
5. $F = \{q_4\}$ is a set of accepting states.

# Formal Definition of NFA

Let $M = (Q, \Sigma, \delta, q, F)$ be an NFA, and let $w \in \Sigma^*$ . We say that $M$ accepts $w$, if $w$ can be written as $w = y_1\, y_2 \ldots\, y_m$ where $y_i \in \Sigma_\epsilon$ for all $i$ with $1 \leq i \leq m$, and there exists a sequence of states $r_1, r_2, \ldots, r_m$ in $Q$, such that:

- $r_0 = q$
- $r_{i+1} \in \delta(r_i, y_{i+1})$, for $i = 0, 1, \ldots, m - 1$
- $r_m \in F$

Otherwise, we say that $M$ rejects the string $w$.

# Difference between DFA and NFA

- DFA has transition function $\delta : Q \times \Sigma \rightarrow Q$
- NFA has transition function $\delta : Q \times \Sigma_\epsilon \rightarrow P(Q)$

✓ Returns a set of states rather than a single state.

✓ Allows for $\epsilon$-transition because $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$.

✓ Note that every DFA is also an NFA.

# Formal Definition of NFA

Extend the map $\delta$ to a map $Q \times \Sigma^* \rightarrow P(Q)$ by defining:

$$\delta(q, \epsilon) = \{q\} \quad \text{for all } q \in Q$$

$$\delta(q, wa) = \cup_{p \in \delta(q,\, w)} \delta(p, a) \quad \text{for all } q \in Q;\ w \in \Sigma^*;\ a \in \Sigma$$

Thus $\delta(q, w)$ is the set of all possible states that can arise when the input $w$ is received in the state $q$. $w$ is accepted provided that $\delta(q, w)$ contains an accepting state.

**Notation: accepting/rejecting paths**

Suppose, in a DFA, we can get from state $p$ to state $q$ via transitions labelled by letters of a word $w$. Then we say that the states $p$ and $q$ are connected by a path with label $w$.

If $w = abc$ and the 2 intermediate states are $r_1$ and $r_2$ we could write this as:

$$p \xrightarrow{a} r_1 \xrightarrow{b} r_2 \xrightarrow{c} q$$

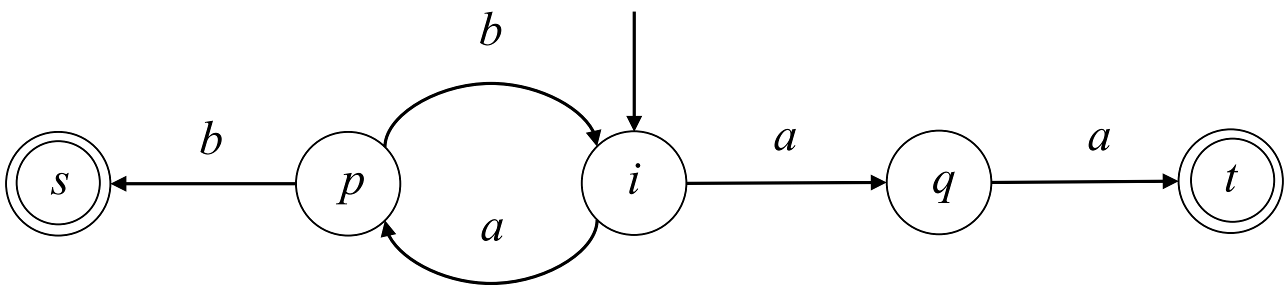
In a NFA, if $\delta(p, a) = \{q, r\}$ we could write:

$$\{p\} \xrightarrow{a} \{q, r\}$$

and this would be an **accepting path** if any state on the RHS is an accepting state, otherwise it would be **rejecting path**.

# Nondeterministic Finite Automata (NFA)

**Exercise**




Can input string *abaa* be accepted by this NFA?

Possible paths of this string?

The accepting path of this string?

# Language accepted by NFA

Let $M = (Q, \Sigma, \delta, q, F)$ be an NFA. The language $L(M)$ accepted by $M$ is defined as

$$L(M) = \{w \in \Sigma^* : M \text{ accepts } w\}.$$

**Example**

Let $A$ be the language $A = \{w \in \{0, 1\}^* : \mathrm{w}$ has a $1$ in the third position from the right$\}$ , design $M : L(M)$.

# Equivalence of DFAs and NFAs)

Two machines (of any type) are **equivalent** if they recognize the same language.

DFA is a restricted form of NFA:

- Every NFA has an equivalent DFA.
- We can convert an arbitrary NFA to a DFA that accepts the same language.
- DFA has the same power as NFA

# DFA to NFA

The formal conversion of a DFA to an NFA is done as follows: Let $M = (Q, \Sigma, \delta, q, F)$ be a DFA. Recall that $\delta$ is a function $\delta : Q \times \Sigma \rightarrow Q$. We define the function $\delta' : Q \times \Sigma_\epsilon \rightarrow P(Q)$ as follows. For any $r \in Q$ and for any $a \in \Sigma_\epsilon$ ,

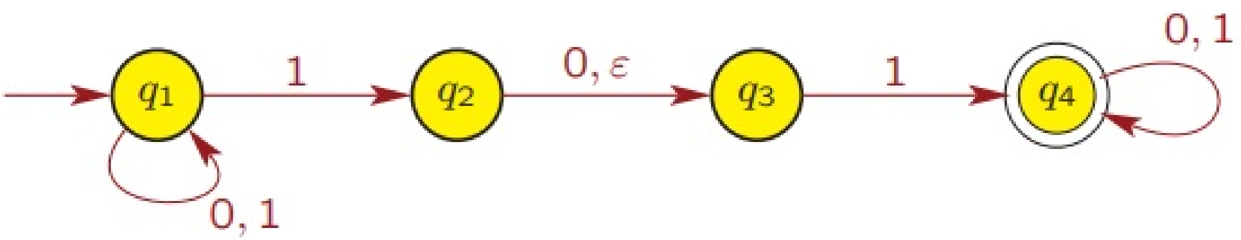
$$\delta'(r, a) = \begin{cases} \{\delta(r, a)\} & if\ a \neq \epsilon \\ \emptyset & if\ a = \epsilon \end{cases}$$

Then $N = (Q, \Sigma, \delta', q, F)$ is an NFA, whose behavior is exactly the same as that of the DFA $M$; the easiest way to see this is by observing that the state diagrams of $M$ and $N$ are equal. Therefore, we have $L(M) = L(N)$.

# NFA to DFA

**Example**

Convert NFA into equivalent DFA



# NFA to DFA

Conversion step

# NFA to DFA

Conversion step

# NFA to DFA

Conversion step

# NFA to DFA

Conversion step

# NFA to DFA

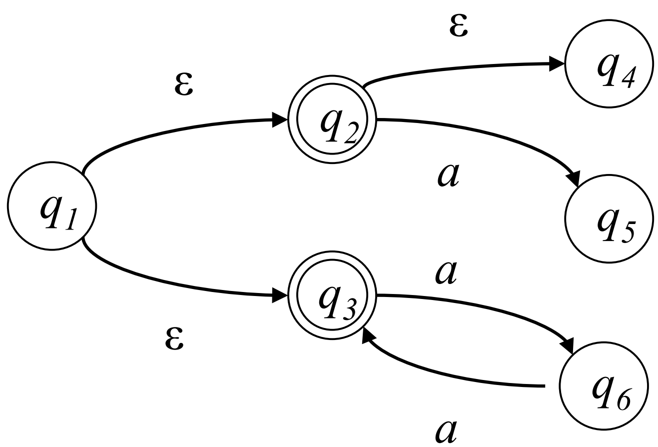
The $\varepsilon$-closure of a set of states $\mathrm{R} \subseteq Q$:

$\mathrm{E(R)} = \{\ q \mid q$ can be reached from $R$ by travelling over zero or more $\varepsilon$ transitions $\}$.

**Example**

$$\mathrm{E}(\{q_1, q_2\}) = \{q_1, q_2, q_3\}.$$

**Question**



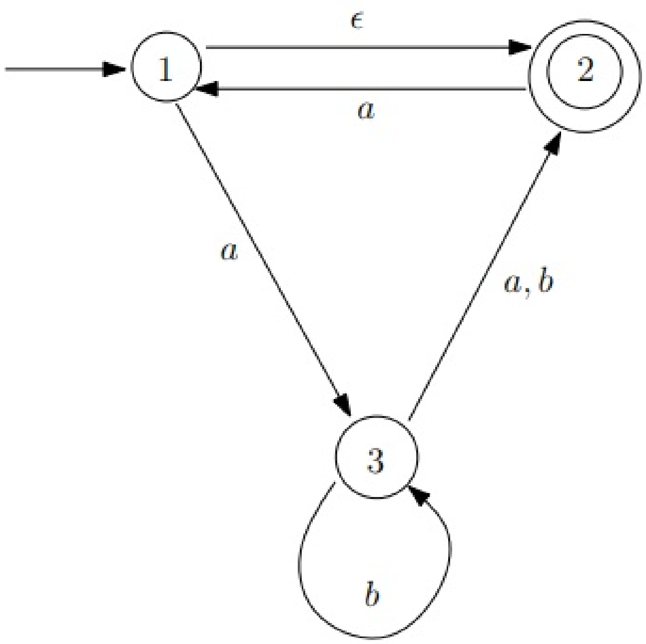

$\mathrm{E}(q_1) = ?\quad \mathrm{E}(q_2) = ?$

# NFA to DFA

## Example

Consider the NFA $M = (Q, \Sigma, \delta, q, F)$, where $Q = \{1, 2, 3\}$, $\Sigma = \{a, b\}$, $q = 1$, $F = \{2\}$, and $\delta$ is given by the following table:

| | $a$ | $b$ | $\epsilon$ |
|---|---|---|---|
| 1 | $\{3\}$ | $\emptyset$ | $\{2\}$ |
| 2 | $\{1\}$ | $\emptyset$ | $\emptyset$ |
| 3 | $\{2\}$ | $\{2,3\}$ | $\emptyset$ |



# NFA to DFA

**Example**

How can we convert the above NFA to a DFA?

# NFA to DFA

**Example**

How can we convert the above NFA to a DFA?

# NFA to DFA

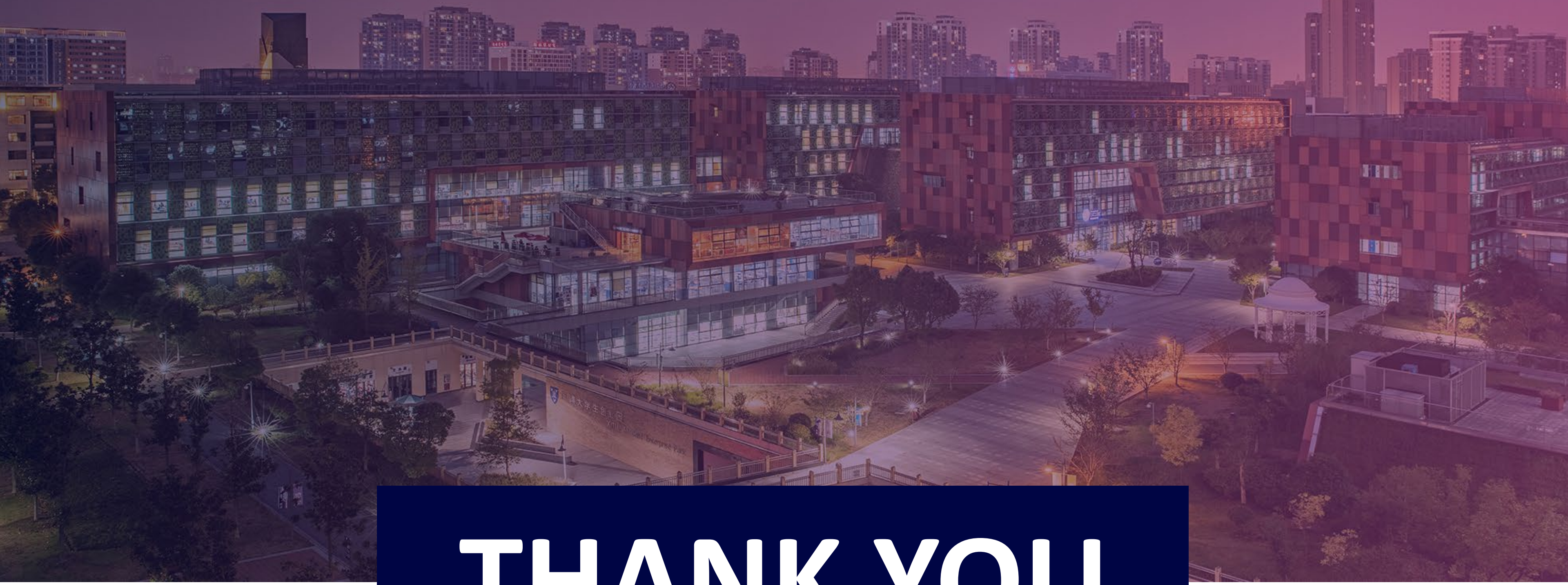
**Example**

How can we convert the above NFA to a DFA?

# NFA to DFA

**Example**

How can we convert the above NFA to a DFA?



# THANK YOU

Xi'an Jiaotong-Liverpool University
西交利物浦大学

XJTLU | SCHOOL OF FILM AND TV ARTS