5CCS2FC2: Foundations of Computing II

Automata and Turing Machines

Week 1

Dr Christopher Hampson

Department of Informatics

King's College London

Deterministic Finite Automata

(a recap)

Languages

- **Alphabet / Words / Languages**
  - **Alphabet** A finite set of symbols. $\boxed{\Sigma = \{a_0, a_1, \ldots, a_k\}}$

Languages

- **Alphabet / Words / Languages**
  - **Alphabet** A finite set of symbols.

$$\boxed{\boldsymbol{\Sigma = \{a, b, c, \ldots, x, y, z\}}}$$

Languages

- **Alphabet / Words / Languages**
  - **Alphabet** A finite set of symbols. $\boxed{\Sigma = \{0, 1\}}$

Languages

- **Alphabet / Words / Languages**
  - **Alphabet** A finite set of symbols.

$$\boxed{\boldsymbol{\Sigma} = \{\boldsymbol{p}, \boldsymbol{q}, \boldsymbol{r}, \wedge, \vee, \rightarrow, \neg, (\ ,\ )\}}$$

Languages

- **Alphabet / Words / Languages**
  - **Alphabet** A finite set of symbols.

$\Sigma = \{$🧐, 🌍, 🎨, 😵‍💫, 🕴, 🐈$\}$

Languages

- **Alphabet / Words / Languages**
  - **Alphabet** A finite set of symbols. $\boxed{\Sigma = \{a_0, a_1, \ldots, a_k\}}$
  - **Words / Strings** $\boxed{\Sigma^* = \{ \text{ all finite strings from } \Sigma \}}$

    The set of all finite strings comprised of letters from the alphabet Σ.

Languages

- **Alphabet / Words / Languages**
  - **Alphabet** A finite set of symbols. $\boxed{\Sigma = \{a_0, a_1, \ldots, a_k\}}$
  - **Words / Strings** $\boxed{\Sigma^* = \{\text{ all finite strings from } \Sigma \}}$

    The set of all finite strings comprised of letters from the alphabet Σ.
    - **The empty string** $\boxed{\varepsilon \in \Sigma^*}$

Languages

- Alphabet / Words / Languages

  - Languages $\boxed{L \subseteq \Sigma^*}$

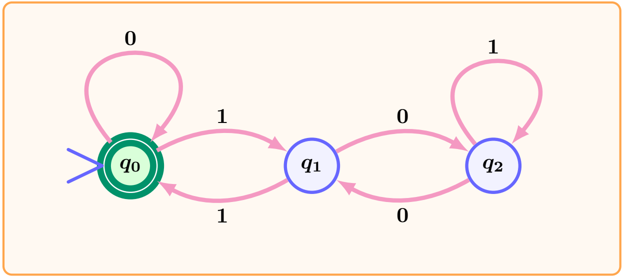
    Any subset (finite of infinite) of words is consider a language.

    - The empty language $\boxed{\emptyset \subseteq \Sigma^*}$

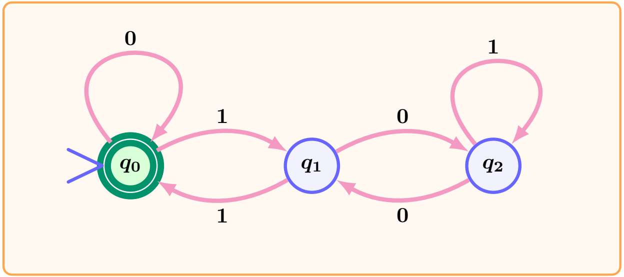
    - All English words

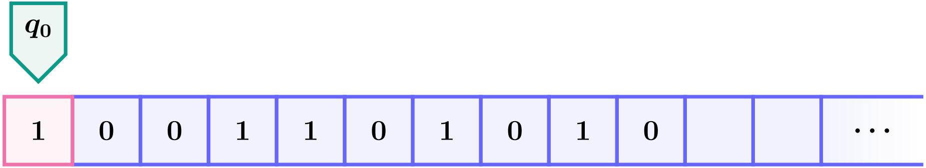
    - All binary strings representing prime numbers *etc.*

Deterministic Finite Automata (DFAs)

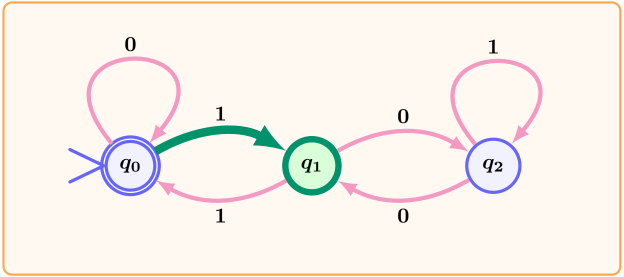


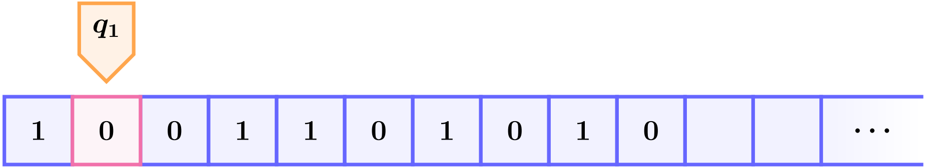
Deterministic Finite Automata (DFAs)



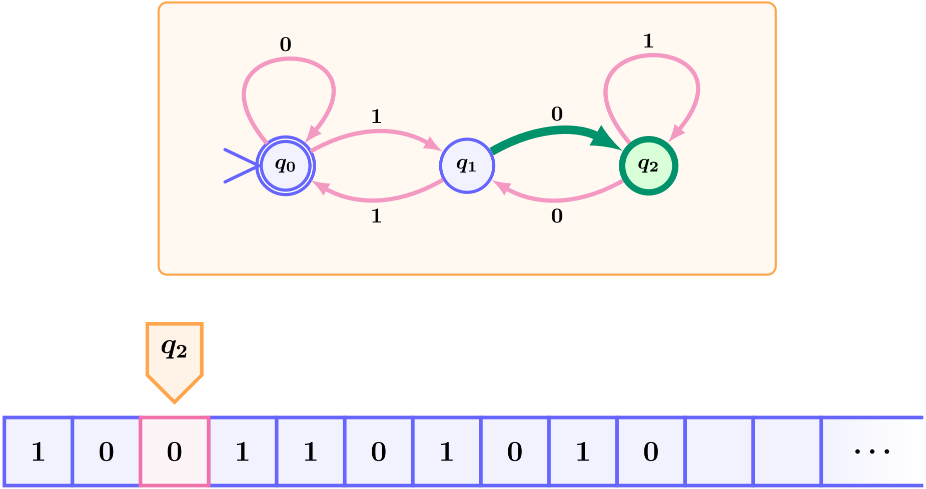


Deterministic Finite Automata (DFAs)

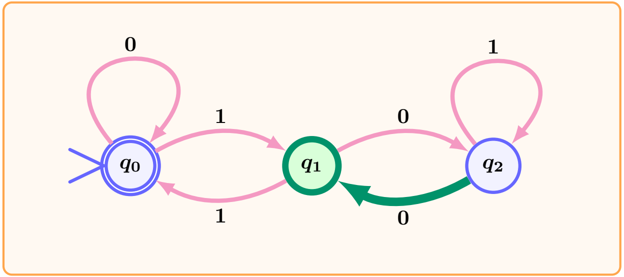


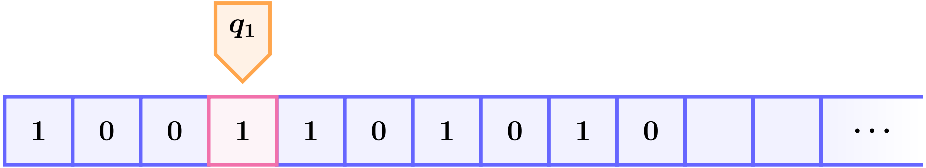


Deterministic Finite Automata (DFAs)

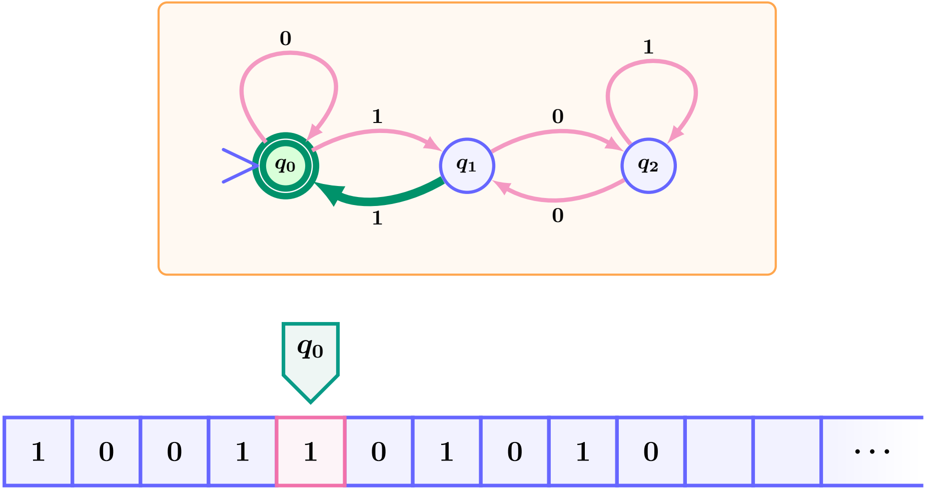


Deterministic Finite Automata (DFAs)

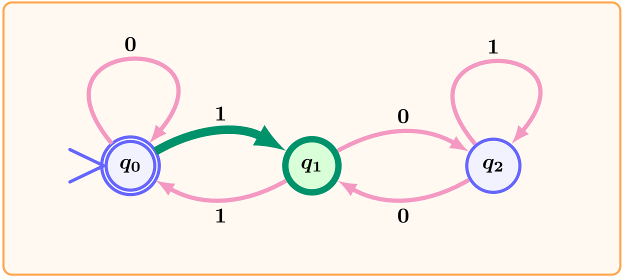


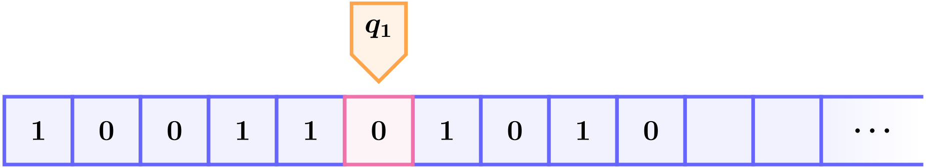


Deterministic Finite Automata (DFAs)

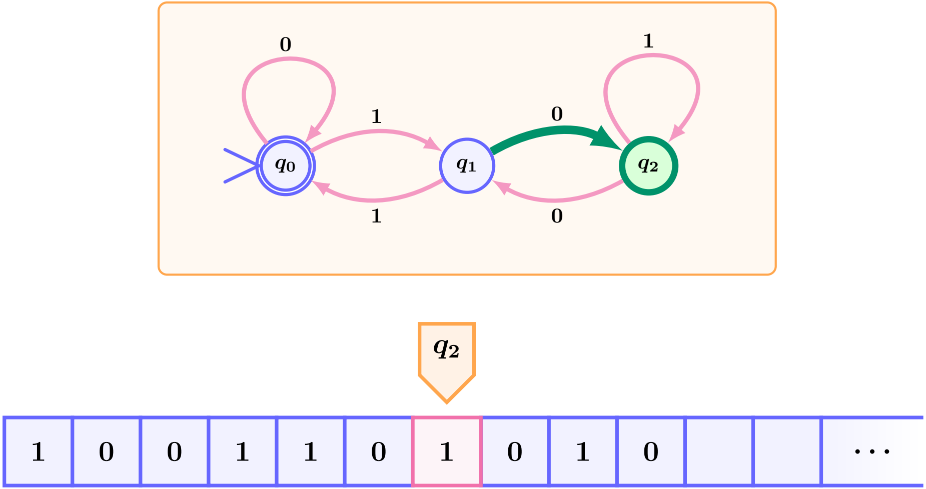


Deterministic Finite Automata (DFAs)

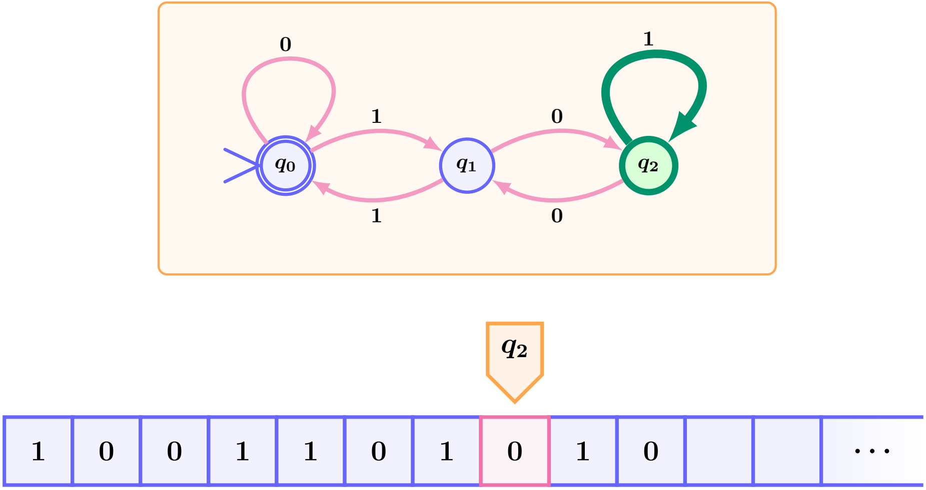




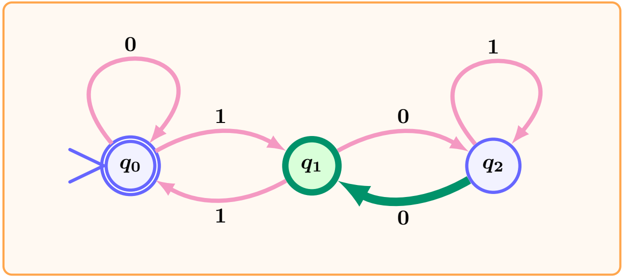
Deterministic Finite Automata (DFAs)

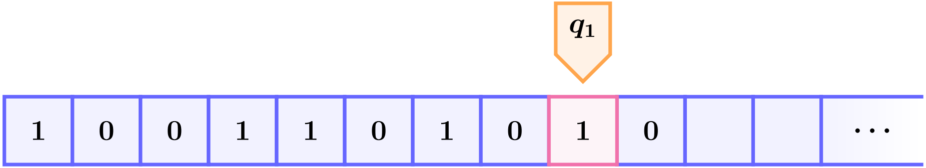


Deterministic Finite Automata (DFAs)

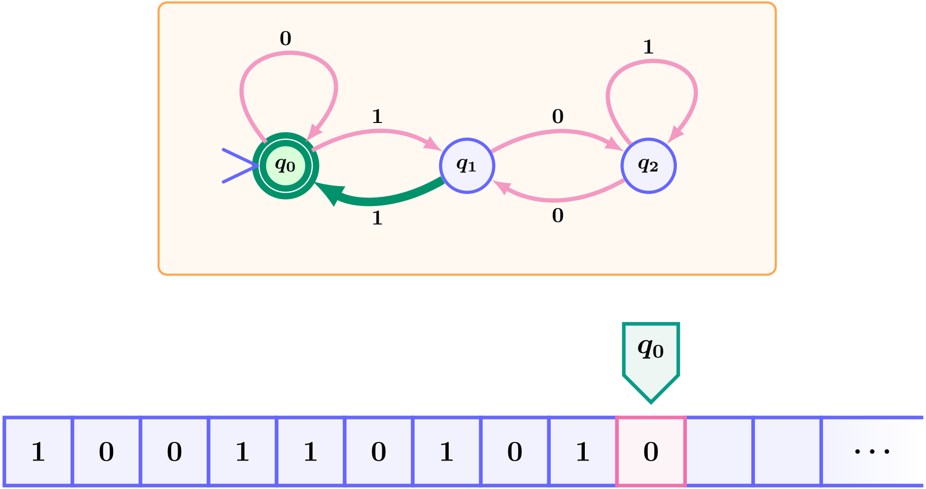


Deterministic Finite Automata (DFAs)

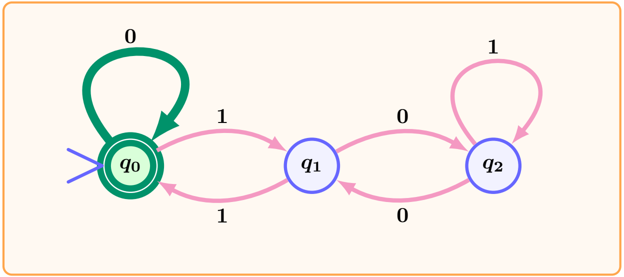


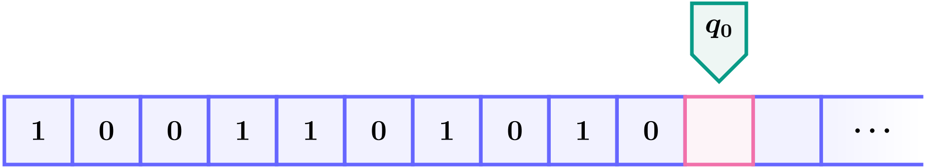


Deterministic Finite Automata (DFAs)



Deterministic Finite Automata (DFAs)





Deterministic Finite Automata (DFAs)

- **Deterministic Finite Automata**

$$\boxed{A \;=\; \langle\, \Sigma,\; Q,\; q_{\mathsf{init}},\; F,\; \delta \,\rangle}$$

Deterministic Finite Automata (DFAs)

- Deterministic Finite Automata

$$\boxed{A \;=\; \langle\, \Sigma,\; Q,\; q_{\mathsf{init}},\; F,\; \delta \,\rangle}$$

  - Alphabet

$$\boxed{\Sigma = \{a_1, \ldots, a_k\}}$$

Deterministic Finite Automata (DFAs)

- Deterministic Finite Automata

$$\boxed{A \;=\; \langle\, \Sigma,\; Q,\; q_{\mathsf{init}},\; F,\; \delta \,\rangle}$$

  - Alphabet

$$\boxed{\Sigma = \{a_1, \ldots, a_k\}}$$

  - Finite set of control states

$$\boxed{Q = \{q_1, \ldots, q_n\}}$$

Deterministic Finite Automata (DFAs)

- Deterministic Finite Automata $\boxed{A = \langle \Sigma, Q, q_{\text{init}}, F, \delta \rangle}$
  - Alphabet $\boxed{\Sigma = \{a_1, \ldots, a_k\}}$
  - Finite set of control states $\boxed{Q = \{q_1, \ldots, q_n\}}$
  - Initial state $\boxed{q_{\text{init}} \in Q}$

Deterministic Finite Automata (DFAs)

- Deterministic Finite Automata $\boxed{A = \langle \Sigma, Q, q_{\text{init}}, F, \delta \rangle}$
    - Alphabet $\boxed{\Sigma = \{a_1, \ldots, a_k\}}$
    - Finite set of control states $\boxed{Q = \{q_1, \ldots, q_n\}}$
    - Initial state $\boxed{q_{\text{init}} \in Q}$
    - Final (accepting) states $\boxed{F \subseteq Q}$

Deterministic Finite Automata (DFAs)

- **Deterministic Finite Automata**

$$\boxed{A \;=\; \langle\, \Sigma,\; Q,\; q_{\mathsf{init}},\; F,\; \delta \,\rangle}$$

  - **Transition function**

$$\boxed{\delta : Q \times \Sigma \to Q}$$

$$\boxed{\delta\,\big(\,\text{current state}\;,\;\text{read symbol}\,\big)\;\mapsto\;\text{new state}}$$

Deterministic Finite Automata (DFAs)

- **Deterministic Finite Automata**

$$\boxed{A \;=\; \langle\, \Sigma,\ Q,\ q_{\text{init}},\ F,\ \delta \,\rangle}$$

  - **Transition function**

$$\boxed{\delta : Q \times \Sigma \to Q}$$

Current State	Read Symbol	New State
q_0	0	q_0
q_0	1	q_1
q_1	0	q_2
q_1	1	q_0
q_2	0	q_1
q_2	1	q_2

Deterministic Finite Automata (DFAs)

- **Configuration** A complete description of the machine at any given time

$$\boxed{(\text{ current state, substring to the right of tape head })}$$

Deterministic Finite Automata (DFAs)

- **Configuration** A complete description of the machine at any given time

$$\boxed{(\text{current state},\ \text{substring to the right of tape head})}$$

- **Computation** The sequence of configurations obtained by running the DFA
  - **Example:**

$$\boxed{(q_0, 1001) \rightarrow (q_1, 001) \rightarrow (q_2, 01) \rightarrow (q_1, 1) \rightarrow (q_0, \varepsilon)}$$

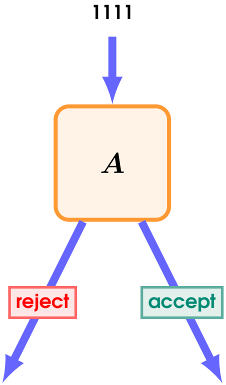
Deterministic Finite Automata (DFAs)

- **Accept Criterion**

A accepts w	$\iff$	The final state of the (unique) computation starting (q_{init}, w) is an accepting state

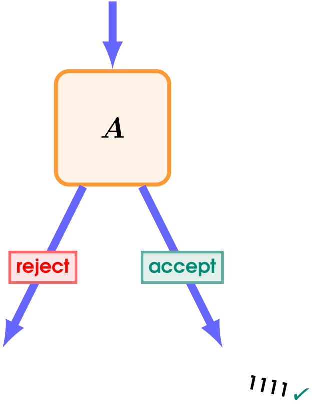
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



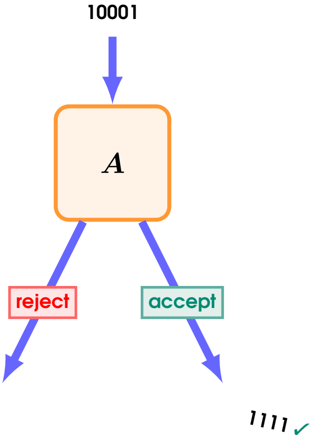
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



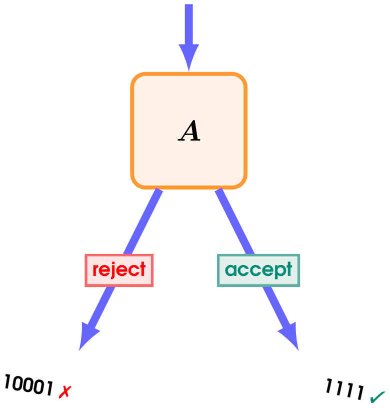
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



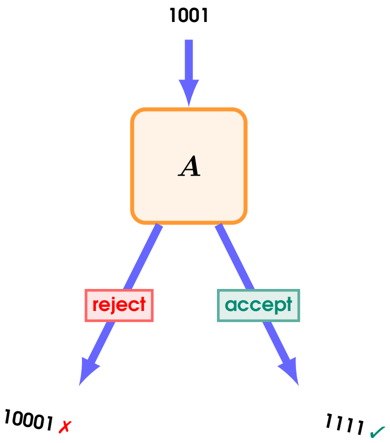
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



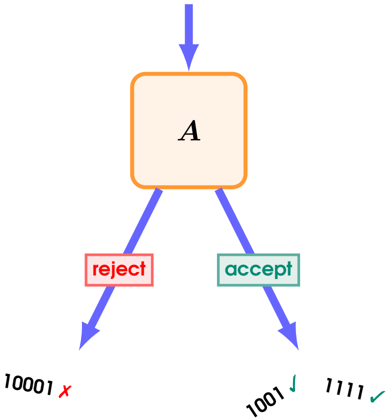
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



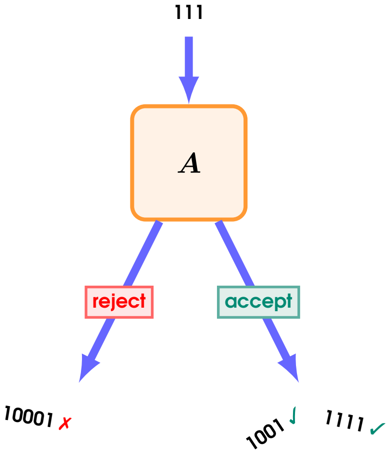
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



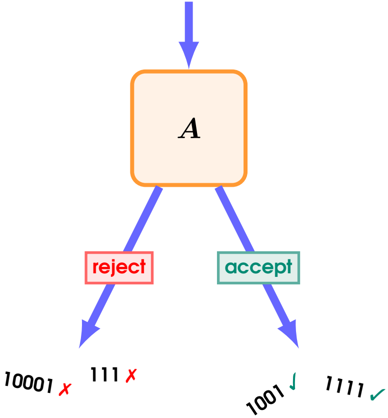
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



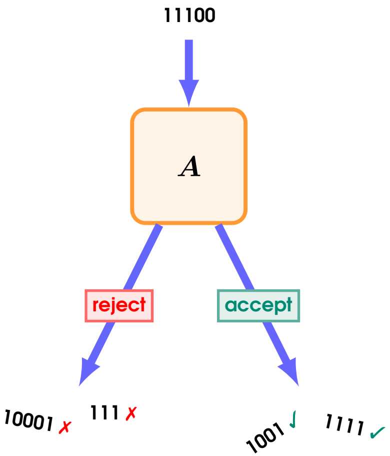
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



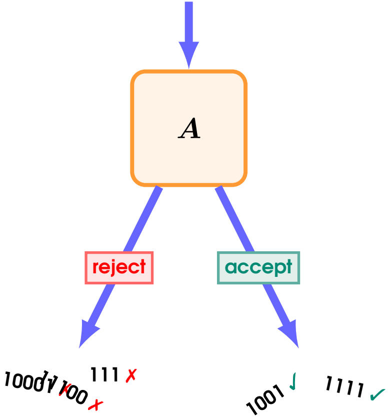
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



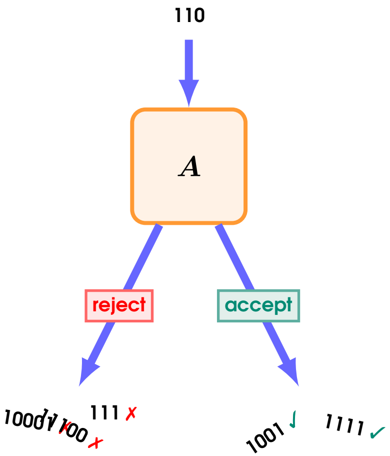
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



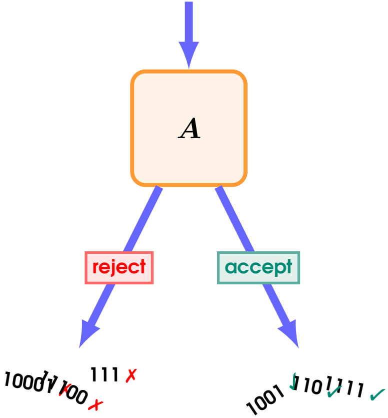
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



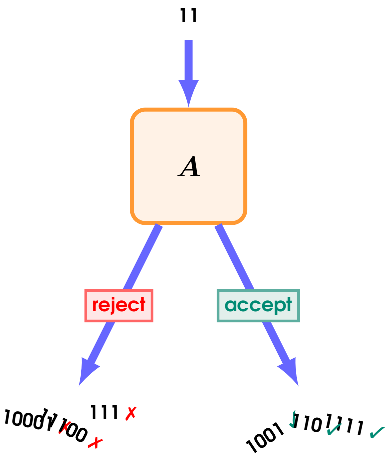
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



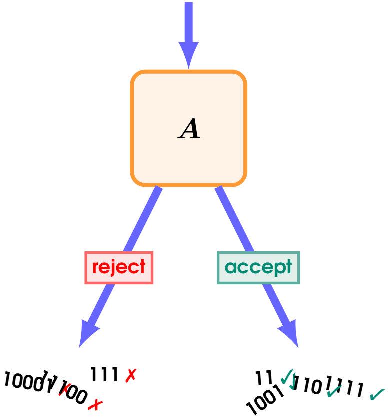
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



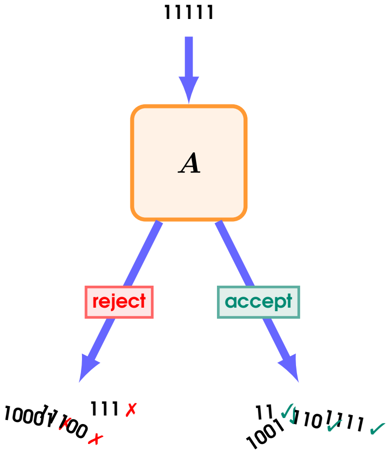
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



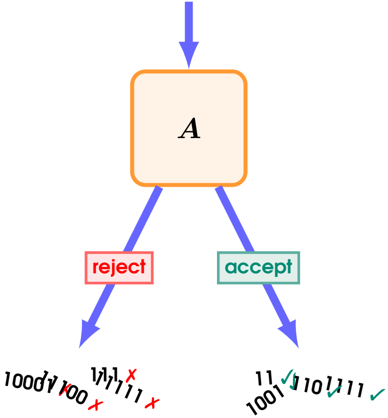
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



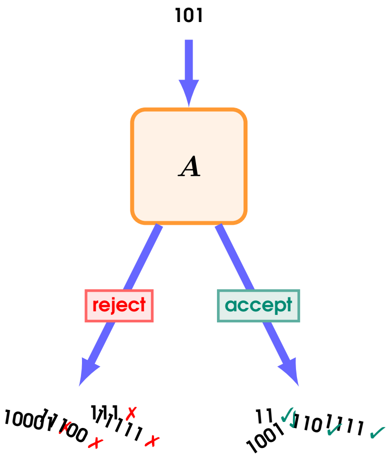
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



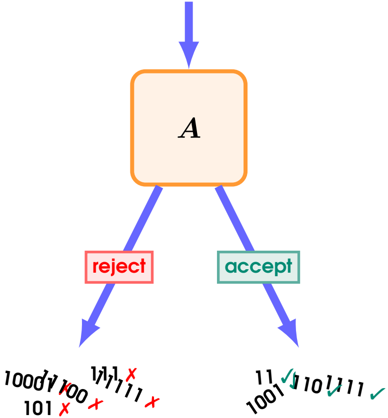
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



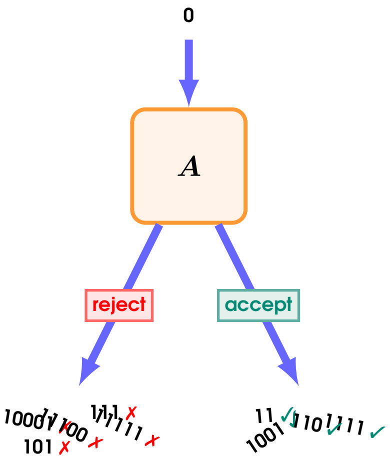
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



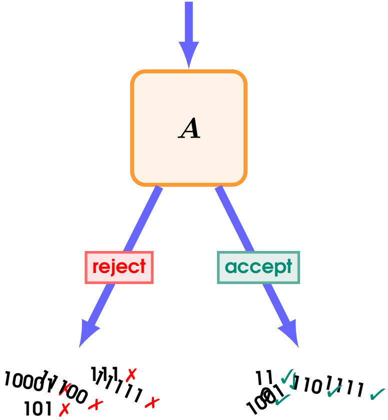
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



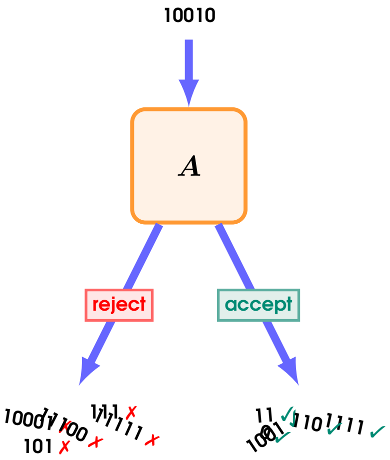
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



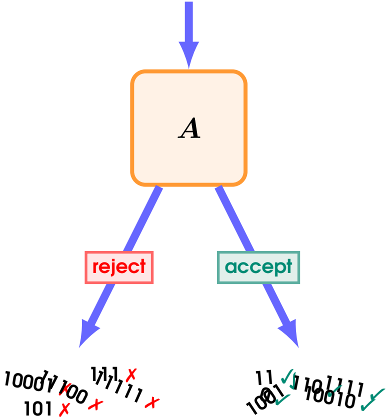
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



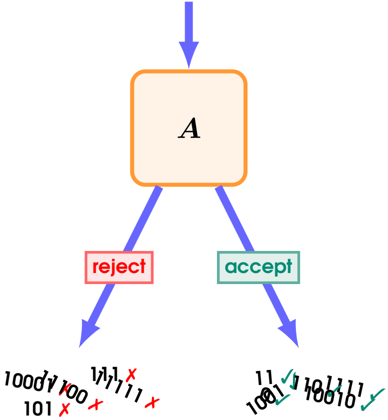
Deterministic Finite Automata (DFAs)

- **Decision Procedure**



Deterministic Finite Automata (DFAs)

- **Decision Procedure**



$$\textbf{Language}\,(A) \;=\; \{\; w \in \Sigma^* \;:\; A \text{ accepts } w \;\}$$

Non-deterministic Finite Automata

(a recap)

Non-deterministic Finite Automata (NFAs)

- Non-deterministic Finite Automata $\boxed{A = \langle \Sigma, Q, q_{\mathsf{init}}, F, \delta \rangle}$

  - Alphabet $\boxed{\Sigma = \{a_1, \ldots, a_k\}}$
  - Finite set of control states $\boxed{Q = \{q_1, \ldots, q_n\}}$
  - Initial state $\boxed{q_{\mathsf{init}} \in Q}$
  - Final (accepting) states $\boxed{F \subseteq Q}$

As before!

Non-deterministic Finite Automata (NFAs)

- **Non-deterministic Finite Automata**

$$\boxed{A = \langle\, \Sigma,\ Q,\ q_{\text{init}},\ F,\ \delta\,\rangle}$$

  - **Transition function**

$$\boxed{\delta : Q \times \Sigma \to \mathcal{P}(Q)}$$

(where $\mathcal{P}(Q)$ denotes the powerset of Q)

$$\boxed{\delta\left(\text{current state}\ ,\ \text{read symbol}\right) \mapsto \left\{\text{all possible new states}\right\}}$$

Non-deterministic Finite Automata (NFAs)

- **Non-deterministic Finite Automata** $\quad A = \langle \Sigma, Q, q_{\text{init}}, F, \delta \rangle$

  - **Transition function** $\quad \delta : Q \times \Sigma \to \mathcal{P}(Q)$

Current State	Read Symbol	New State
s	a	$\{q\}$
s	b	$\{q, f\}$
q	a	$\{f\}$
q	b	$\emptyset$
f	a	$\{f\}$
f	b	$\{s, f\}$

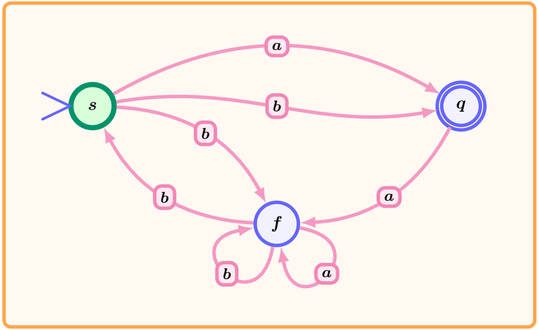
Non-deterministic Finite Automata (NFAs)

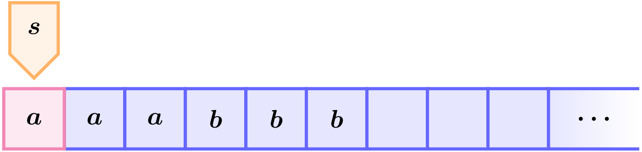
- **Non-deterministic Finite Automata** $\quad A = \langle \Sigma, Q, q_{\text{init}}, F, \delta \rangle$

  - **Transition function** $\quad \delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$

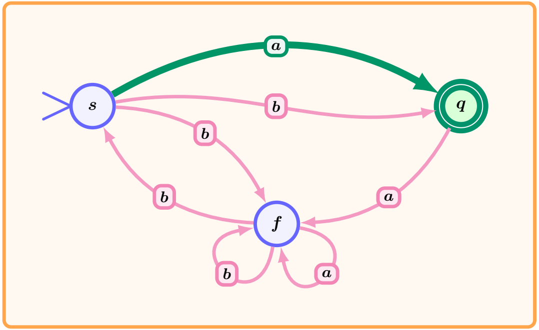
Current State	Read Symbol	New State
s	a	$\{q\}$
s	b	$\{q, f\}$
q	a	$\{f\}$
q	b	$\emptyset$
f	a	$\{f\}$
f	b	$\{s, f\}$

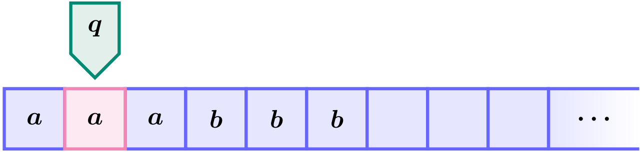
Non-deterministic Finite Automata (NFAs)



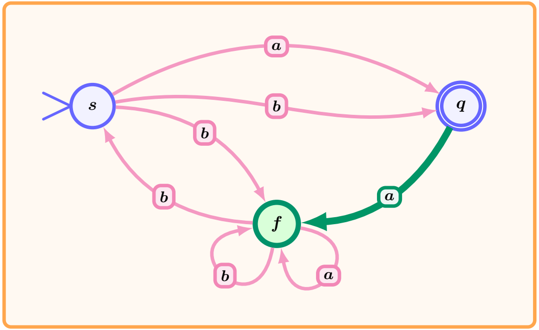


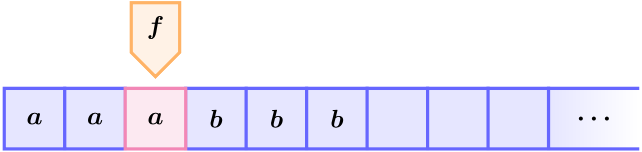
Non-deterministic Finite Automata (NFAs)



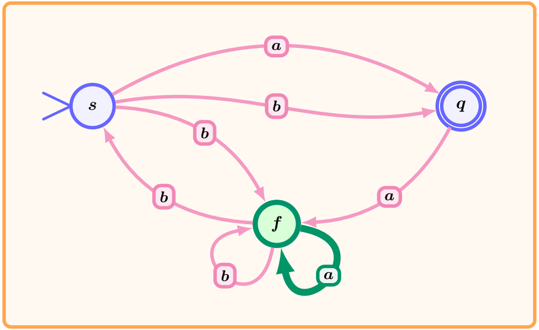


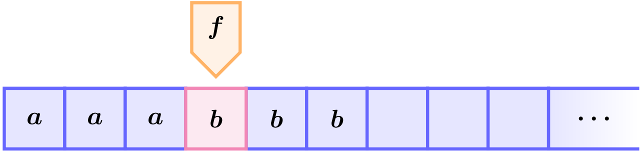
Non-deterministic Finite Automata (NFAs)



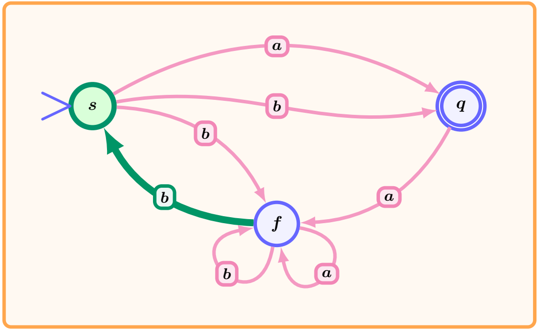


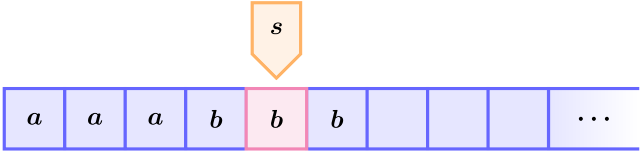
Non-deterministic Finite Automata (NFAs)



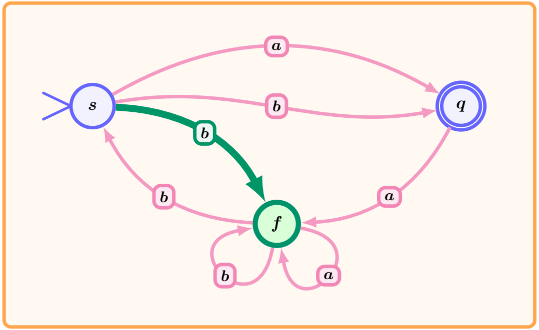


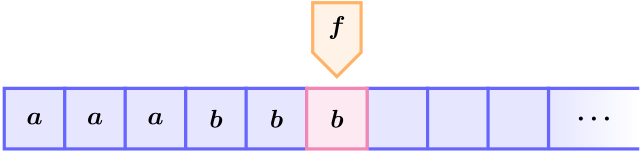
Non-deterministic Finite Automata (NFAs)



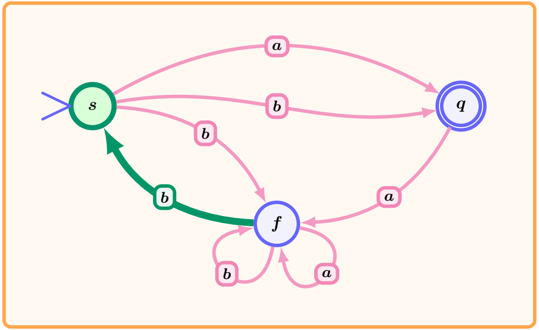


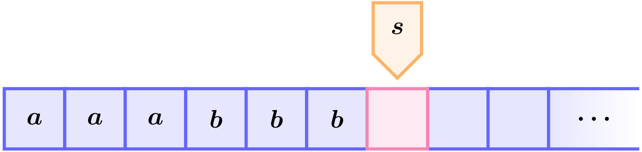
Non-deterministic Finite Automata (NFAs)



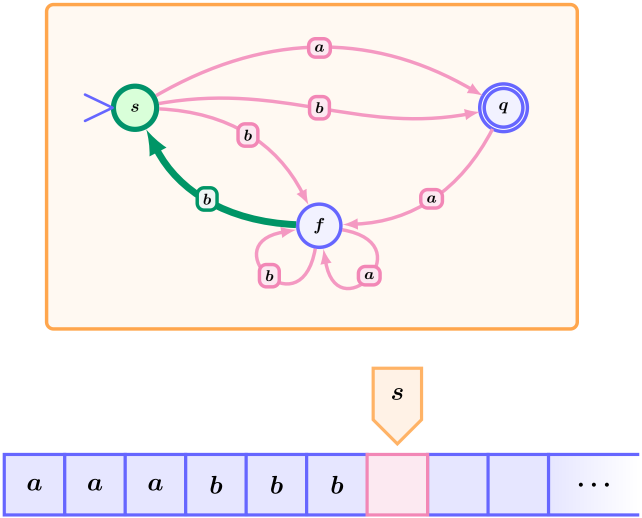


Non-deterministic Finite Automata (NFAs)

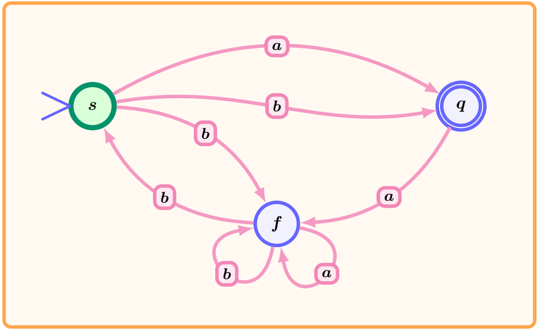


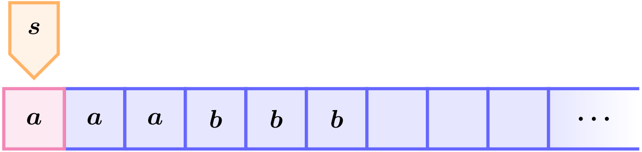


Non-deterministic Finite Automata (NFAs)

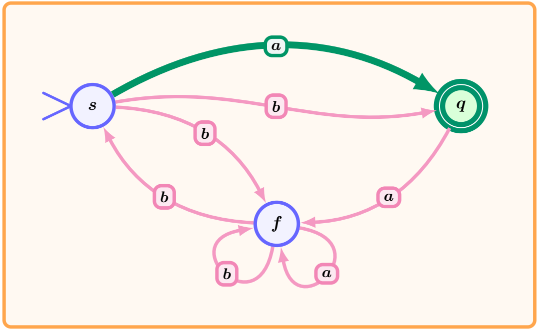


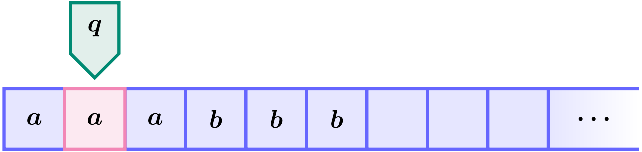
Non-deterministic Finite Automata (NFAs)



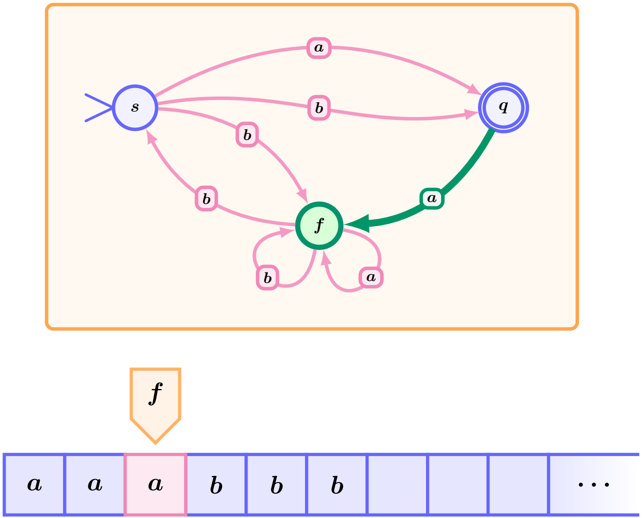


Non-deterministic Finite Automata (NFAs)

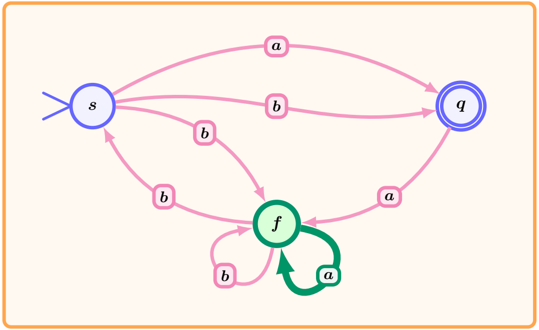


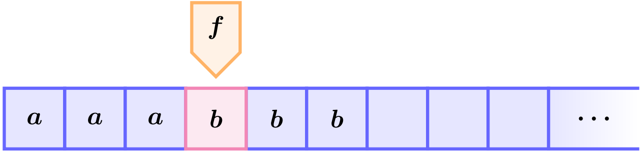


Non-deterministic Finite Automata (NFAs)

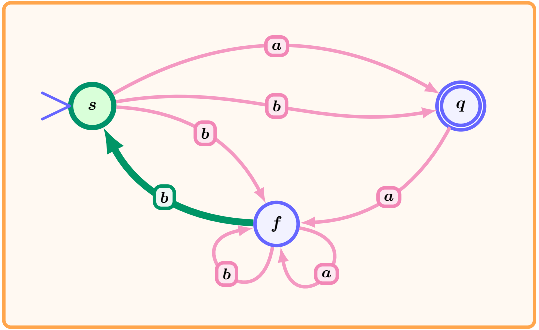


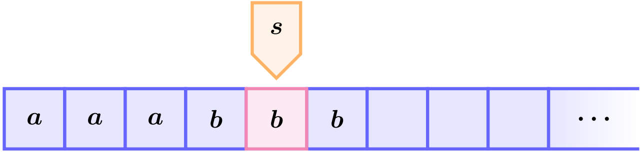
Non-deterministic Finite Automata (NFAs)



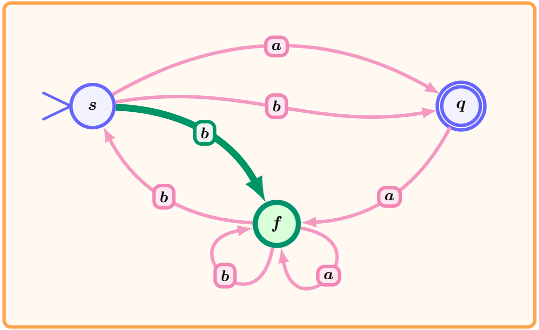


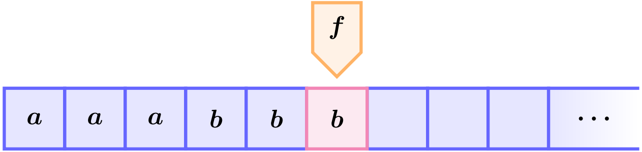
Non-deterministic Finite Automata (NFAs)



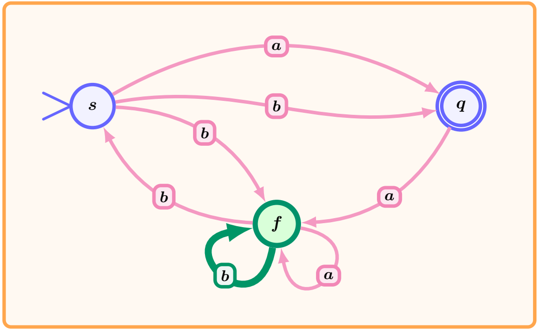


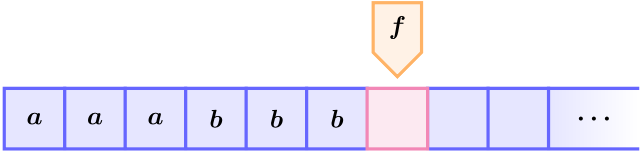
Non-deterministic Finite Automata (NFAs)



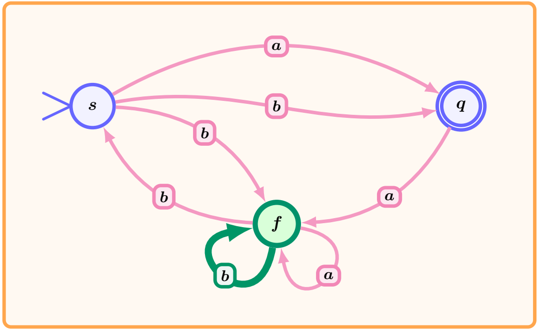


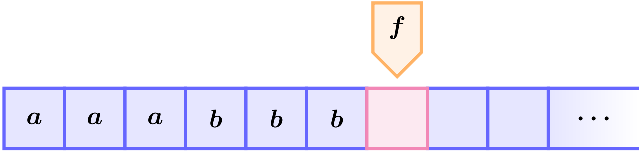
Non-deterministic Finite Automata (NFAs)



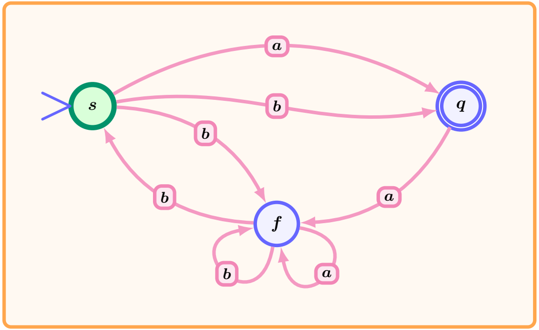


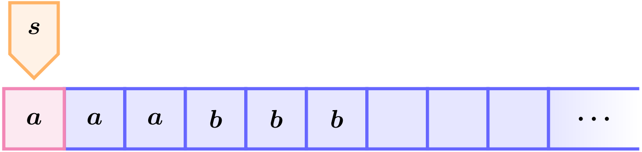
Non-deterministic Finite Automata (NFAs)



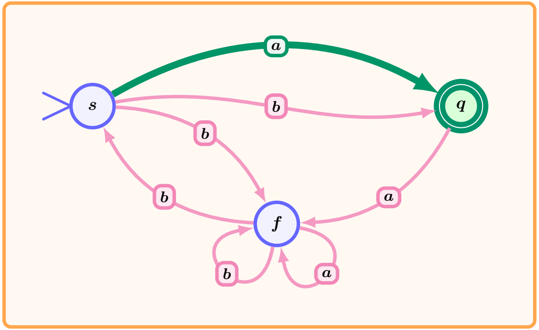


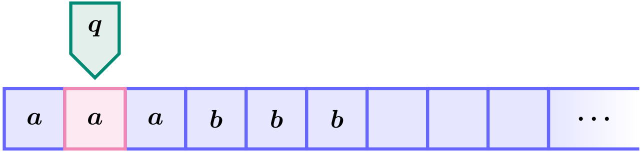
Non-deterministic Finite Automata (NFAs)



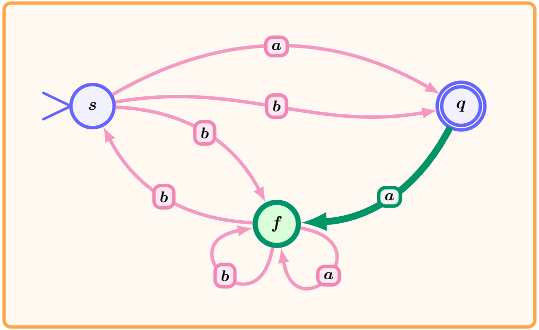


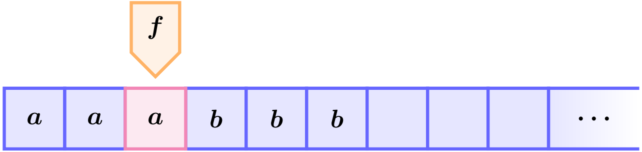
Non-deterministic Finite Automata (NFAs)



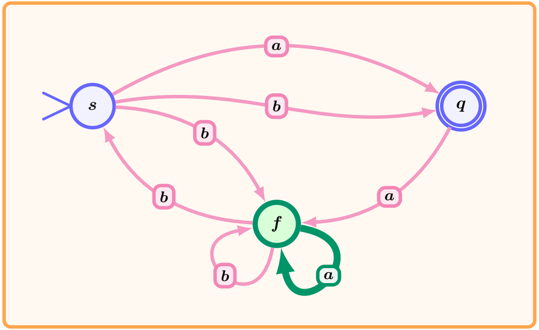


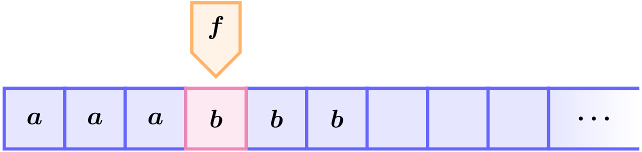
Non-deterministic Finite Automata (NFAs)



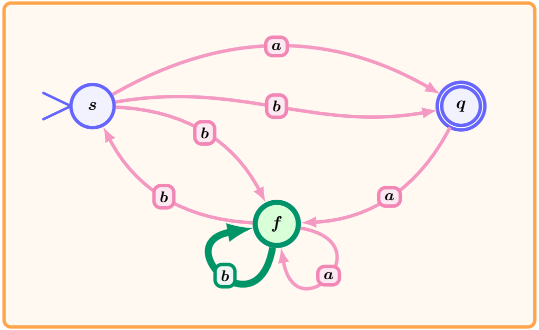


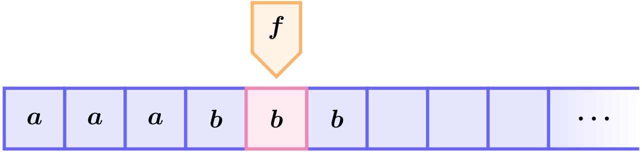
Non-deterministic Finite Automata (NFAs)



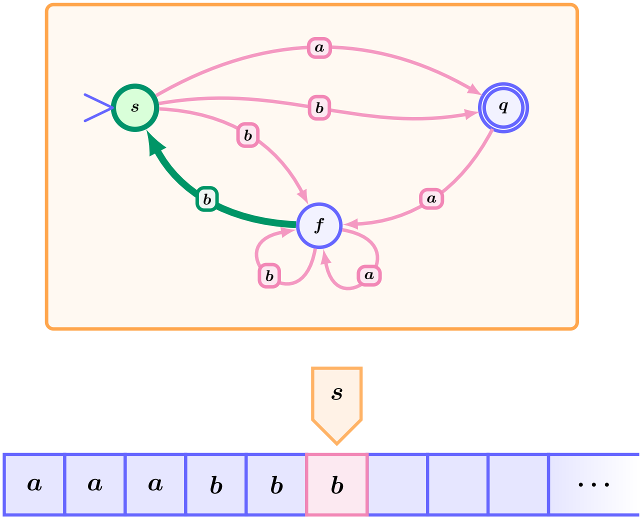


Non-deterministic Finite Automata (NFAs)

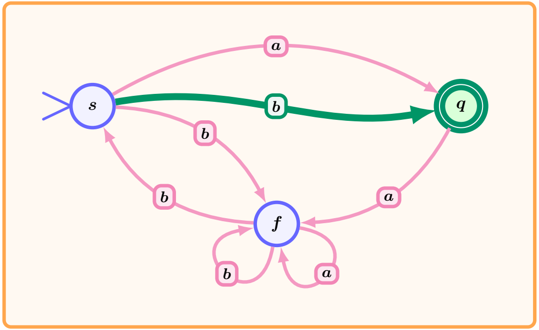


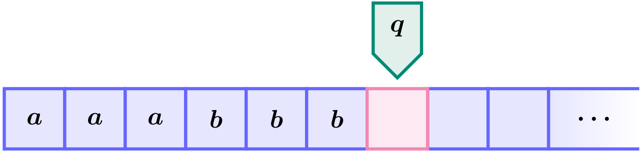


Non-deterministic Finite Automata (NFAs)

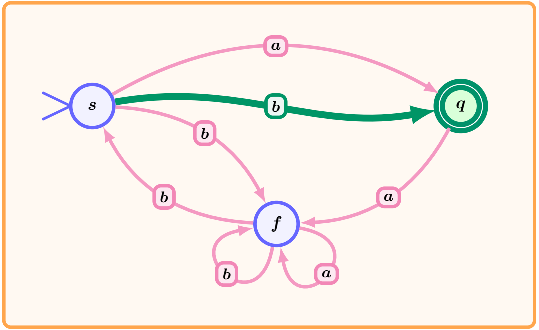


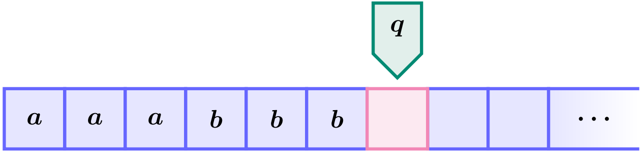
Non-deterministic Finite Automata (NFAs)





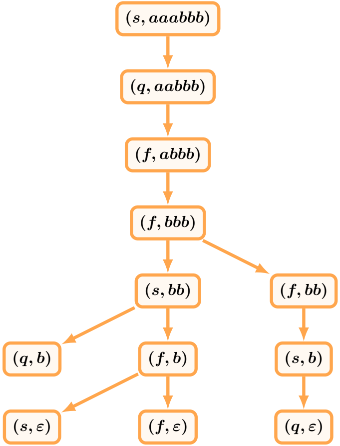
Non-deterministic Finite Automata (NFAs)





Non-deterministic Finite Automata (NFAs)

- Computation



Non-deterministic Finite Automata (NFAs)

- **Accept Criterion for NFAs**

A accepts w	$\iff$	The final state of the **ANY** computation starting (q_{init}, w) is an accepting state

Non-deterministic Finite Automata (NFAs)

> **Theorem** Every NFA can be **converted** to a DFA that accepts the same language.

Proof:

- Recall the **power-set construction** from FC1.

Q.E.D.

Regular Expressions and Regular Languages

Regular Expressions and Regular Languages

- **Regular Expression** Any word that can be constructed using only the following rules:
  - **Concatenation** $\boxed{wv}$
  - **Alternation / Choice** $\boxed{(w \cup v)}$
  - **Kleene Star** $\boxed{w^*}$

Regular Expressions and Regular Languages

- **Regular Expression** Any word that can be constructed using only the following rules:
  - **Concatenation** $\boxed{wv}$
  - **Alternation / Choice** $\boxed{(w \cup v)}$
  - **Kleene Star** $\boxed{w^*}$
- **Regular Language** Any set of words that can be 'represented' by a regular expression.

Regular Expressions and Regular Languages

- **Some Examples:**
  - All binary strings that start with a one and end with a zero,

$$\boxed{1(1 \cup 0)^*0}$$

- **Some Examples:**
  - All binary strings that start with a one and end with a zero,

$$\boxed{1(1 \cup 0)^*0}$$

  - All strings of **even length** containing only $\square$ and $\diamond$,

$$\boxed{((\square \cup \diamond)(\square \cup \diamond))^*}$$

Regular Expressions and Regular Languages

- **Some Examples:**
  - **Less Obviously:** All binary strings representing multiples of three,

$$\boxed{\left(0 \cup 1(01^*0)^*1\right)^*}$$

Regular $\Longrightarrow$ Recognised by some DFA/NFA

Regular $\Leftarrow$ **Recognised by some DFA/NFA**

Regular $\Leftrightarrow$ Recognised by some DFA/NFA

Are all languages regular?

End of Slides!