Learning Outcomes

By the end of this session you should be able to...

- ... *define* nondeterministic finite automata (NFA)

- ... *prove* that DFA and NFA are equivalent

- ... *define* NFA with $\epsilon$ moves ($\epsilon$-NFA)
Today

- Nondeterminism
- Nondeterministic finite automata (NFAs)
- Subset construction
- Epsilon moves
Determinism vs Nondeterminism

- **Determinism**
  - next state uniquely determined by current state and input

- **Nondeterminism**
  - given a current state and input, there can be any number of next states (including 0!)
Nondeterministic transitions

- we have states and transitions as before, but:
  - there are *two* $b$-labelled transitions from 0;
  - there are *no* transitions from 1.
Recall: DFAs (formally)

- a DFA \( M = (Q, \Sigma, \delta, s, F) \):
  - \( Q \) is a finite set of States;
  - \( \Sigma \) is an alphabet;
  - \( \delta : Q \times \Sigma \to Q \) is the transition function;
    - we will write \( q \overset{a}{\rightarrow} q' \) for \( \delta(q, a) = q' \);
  - \( s \) is the start state;
  - \( F \subseteq Q \) is the set of final states;
- which part of the definition guarantees determinism?
Nondeterministic finite automata (NFA)

- an NFA \( M \overset{\text{def}}{=} (Q, \Sigma, \Delta, s, F) \):
  - \( Q \) is a finite set of States;
  - \( \Sigma \) is an alphabet;
  - \( \Delta : Q \times \Sigma \rightarrow 2^Q \) is the transition function;
    - \( \star \) we will write \( q \overset{\sigma}{\rightarrow} q' \) for \( q' \in \Delta(q, \sigma) \);
  - \( s \) is the start state;
  - \( F \subseteq Q \) is the set of final states;
Formal presentation

\( Q = \{0, 1\}, \Sigma = \{a, b\} \)

\( s = 0, F = \{1\} \)

\( \Delta(0,a) = \{0\}, \Delta(0,b) = \{0, 1\} \)

\( \Delta(1,a) = \emptyset, \Delta(1,b) = \emptyset \)
Nondeterministic acceptance

- an automaton accepts some strings (and rejects those that it doesn’t accept.

- diagrammatically: place a pebble on the initial state, move it by choosing a transition according to the symbols in the string. Accept there is a way of choosing transitions in such a way that we end in a final state, otherwise reject.

  \[
  abbab, \epsilon, babbbbab, abba
  \]
Nondeterministic acceptance

- for $\sigma_1, \sigma_2 \in \Sigma$, write $q \xrightarrow{\sigma_1\sigma_2} q'$ when there is a $q''$ such that $q \xrightarrow{\sigma_1} q''$ and $q'' \xrightarrow{\sigma_2} q'$, and similarly for longer length strings. Write $q \xrightarrow{\epsilon} q'$ when $q = q'$.

- a string $x \in \Sigma^*$ determines a (possibly empty) set of states $q$ such that $s \xrightarrow{x} q$. $M$ accepts $x$ when there exists $f \in F$ such that $s \xrightarrow{x} f$. 
• Any DFA is a special kind of NFA:
  ◦ compose \( \delta : Q \times \Sigma \rightarrow Q \) with the *singleton* function \( \eta : Q \rightarrow 2^Q \) defined \( \eta(q) = \{q\} \) to obtain a function \( \delta ; \eta : Q \times \Sigma \rightarrow 2^Q \);
  ◦ (exercise!) the acceptance conditions coincide;
  ◦ so NFAs are *at least as powerful* as DFAs (accept at least the languages accepted by DFAs).
The power of nondeterminism

Q. Are NFAs more ‘powerful’ than DFAs?
   i.e. is there a language accepted by an NFA that cannot be accepted by a DFA?

A. No! Any NFA can be converted to a DFA that accepts the same language. The conversion method is known as the subset construction.
Subset construction

- Let $M = (Q, \Sigma, \Delta, s, F)$ be an NFA
- we will construct a DFA $M'$ over $\Sigma$ with:
  - set of states $Q' \overset{\text{def}}{=} 2^Q$;
  - starting state $s' \overset{\text{def}}{=} \{s\}$;
  - end states $F' \overset{\text{def}}{=} \{X \mid \exists f \in F. f \in X\}$;
  - transition function $\delta'(X, \sigma) \overset{\text{def}}{=} \bigcup_{q \in X} \Delta(q, \sigma)$
The states that correspond to subsets $\emptyset$ and $\{1\}$ are called **unreachable**: there is no path from the start state that ends in such a state;

Some people call state $\emptyset$ the **error** state.
Conclusions

- Any DFA is an NFA, so regular languages are included in the class of languages accepted by NFAs.
- Any NFA can be turned into a DFA that accepts the same language, so reverse inclusion holds. Hence:

  NFAs accept precisely the regular languages.
Epsilon moves

• We have seen that nondeterminism can be tamed
  ◦ the price: possibly exponential increase in number of states...

• We need one more useful feature: $\epsilon$-moves

• Idea: in the pebble game you can always take an $\epsilon$-labelled transition without "consuming a symbol"
Nondeterministic finite automata with $\epsilon$-moves

• an $\epsilon$NFA $M \overset{\text{def}}{=} (Q, \Sigma, \theta, s, F)$:
  
  ○ $Q$ is a finite set of States;
  ○ $\Sigma$ is an alphabet;
  ○ $\theta : Q \times (\Sigma + \{\epsilon\}) \rightarrow 2^Q$ is the transition function;
    
    ★ we will write $q \xrightarrow{\sigma} q'$ for $q' \in \theta(q, \sigma)$;
  ○ $s$ is the start state;
  ○ $F \subseteq Q$ is the set of final states;
Acceptance in $\epsilon$NFAs

- write $q \xrightarrow{\epsilon} q'$ if either $q = q'$ or there exists a sequence of $\epsilon$-moves $q \xrightarrow{\epsilon} \cdots \xrightarrow{\epsilon} q'$;
- write $q \xrightarrow{\sigma} q'$ (for $\sigma \in \Sigma$) if there exists a sequence $q \xrightarrow{\epsilon} \xrightarrow{\sigma} \xrightarrow{\epsilon} q'$;
- write $q \xrightarrow{\sigma \tau} q'$ if we have $q \xrightarrow{\epsilon} \xrightarrow{\sigma} \xrightarrow{\epsilon} \xrightarrow{\tau} \xrightarrow{\epsilon} q'$, and similarly for longer strings.
- a string $x$ is **accepted** if there exists $f \in F$ such that $s \xrightarrow{x} f$. 

COMP 2210: Lecture 2 – p. 18
• What is the language accepted by this $\epsilon$NFA?
The power of $\epsilon$-moves

- Any NFA is a special kind of $\epsilon$NFA (one that has no $\epsilon$-transitions)

- Q. Are $\epsilon$NFAs more powerful than NFAs?
- A. No! Any $\epsilon$NFA can be turned into an NFA. (see next slide)

- So $\epsilon$NFAs accept precisely the regular languages.
From $\epsilon$NFAs to NFAs

- Let $M$ be an $\epsilon$NFA. Let $M'$ be an NFA with the same states but:

  \[ \Delta'(q, \sigma) \overset{\text{def}}{=} \{ q' \mid q \xrightarrow{\epsilon} q' \}, \quad F' \overset{\text{def}}{=} \{ q \mid \exists f \in F. q \xrightarrow{\epsilon} f \} \]

- Example:

![Diagram of NFAs before and after transformation]

- Claim: $L(M) = L(M')$
Learning Outcomes

You should be able to . . .

• . . . define nondeterministic finite automata (NFA)

• . . . prove that DFA and NFA are equivalent

• . . . define NFA with \( \varepsilon \) moves (\( \varepsilon \)-NFA)