COMP4141 Slides Week 2

Brendan Mabbutt

UNSW

February 25, 2026

Admin

Tutorial Solutions

- They will be posted once all the tutes have been held for the week

Quizes

- They start **today**!
- They will be 20 minutes; you will write your answers and I will collect at the end. This will be similar to exam conditions; laptop and phones away. No talking to your fellow students during the quiz.
- Please write down your name, student ID, and the tutorial session on top of the first page.

ϵ-NFAs

ϵ-Nondeterministic Finite Automata

Definition

An ϵ-NFA is an NFA that allows state changes that do not consume input symbols. The only difference in the formal definition is the transition relation:

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \to 2^Q.$$

ϵ-Closure

Definition

For $q \in Q$, define $E(q)$ to be the least set satisfying:

- $q \in E(q)$,
- if $s \in E(q)$ and $t \in \delta(s, \epsilon)$, then $t \in E(q)$.

For a set $S \subseteq Q$, define

$$E(S) = \bigcup_{q \in S} E(q).$$

Proof Constructions

Lemma

Let $L \subseteq \Sigma^*$ be a regular language. Then $\Sigma^* \setminus L$ is also a regular language.

Proof idea: Replace F with $Q \setminus F$.

Lemma

Let $L_1, L_2 \subseteq \Sigma^*$ be regular languages. Then $L_1 \cap L_2$ is regular.

Proof idea: Construct a product automaton with $Q = Q_1 \times Q_2$,

$$\delta((q, q'), a) = (\delta_1(q, a), \delta_2(q', a)), \quad F = F_1 \times F_2 = \{(q, q') : q \in F_1 \text{ and } q' \in F_2\}.$$

Regular Languages

Regular Languages

Regular Expressions (Inductive Definition)

The set of regular expressions over an alphabet Σ is defined inductively:

- If $a \in \Sigma$, then a is a regular expression.
- $\emptyset$ is a regular expression.
- ε is a regular expression.
- If R_1 and R_2 are regular expressions, then:
  - $R_1 \cup R_2$ is a regular expression,
  - $R_1 \cdot R_2$ is a regular expression,
  - R_1^* is a regular expression.

Equivalence with Automata

Kleene's Theorem

Let $L \subseteq \Sigma^*$ be a language. The following are equivalent:

- L is accepted by a DFA,
- L is accepted by an NFA,
- L is accepted by a GNFA,
- $L = L(R)$ for some regular expression R.

From Regular Expressions to Finite Automata (Union)

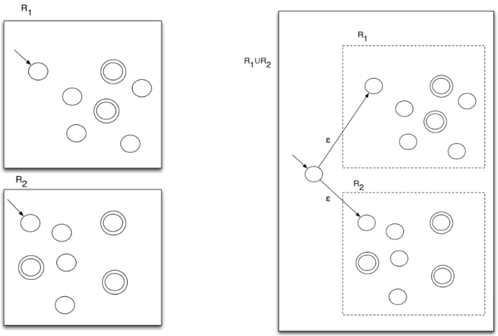


Figure: Proof By Picture (Union)

From Regular Expressions to Finite Automata (Concatenation)

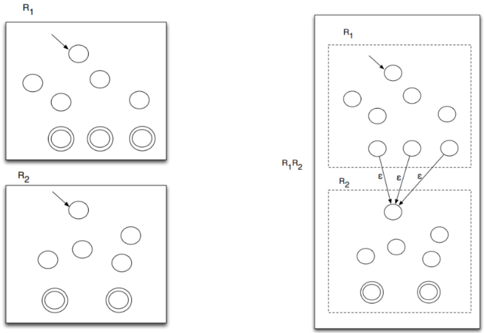


Figure: Proof By Picture (Concatenation)

From Regular Expressions to Finite Automata (Kleene Star)

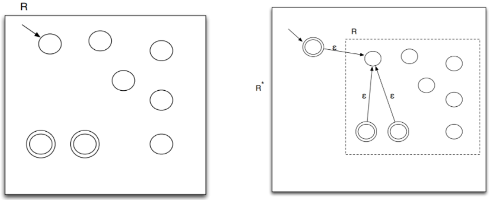


Figure: Proof By Picture (Kleene Star)

Generalised NFAs (GNFAs)

Definition

A **generalised nondeterministic finite automaton (GNFA)** is a 5-tuple $(Q, \Sigma, \delta, q_0, q_F)$ where:

- Q is a finite set of states,
- Σ is the input alphabet,
- $\delta : (Q \setminus \{q_F\}) \times (Q \setminus \{q_0\}) \to \mathrm{RE}_\Sigma$ is the transition function,
- $q_0 \in Q$ is the start state,
- $q_F \in Q$ is the accept state.

Pumping Lemma

Pumping Lemma

If $L \subseteq \Sigma^*$ is regular, then there exists $p \in \mathbb{N}$ (the *pumping length*) such that for every $w \in L$ with $|w| \geq p$, we can write $w = xyz$ satisfying:

1. $|xy| \leq p$,
2. $|y| > 0$,
3. $xy^iz \in L$ for all $i \in \mathbb{N}$.

Myhill–Nerode Theorem

Distinguishability

Let $L \subseteq \Sigma^*$.
We say x and y are *distinguishable by L* if there exists $z \in \Sigma^*$ such that:

$$xz \in L \quad \text{and} \quad yz \notin L$$

(or vice versa).
We write $x \equiv_L y$ if x and y are not distinguishable by L.
The *index* of L is the number of $\equiv_L$-equivalence classes.

Myhill–Nerode Theorem

A language $L \subseteq \Sigma^*$ is regular if and only if the index of L is finite.
Moreover, this index equals the minimum number of states of any DFA that accepts L.

Using Myhill–Nerode

Myhill–Nerode proof format

Find an infinite sequence of words (not necessarily in L), $u_1, u_2, u_3, \ldots,$ and a doubly indexed sequence of words $w_{i,j}$ for $i < j \in \mathbb{N}_0$ such that

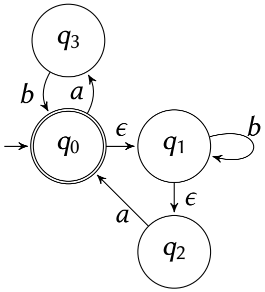
$$u_i w_{i,j} \in L \quad \text{and} \quad u_j w_{i,j} \notin L$$

(or vice versa).

W1 P7 & P8

Problems 7 & 8 (ϵ-Nondeterministic Finite Automata; Week 1)

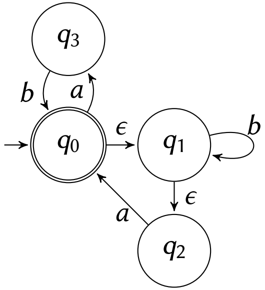
Assume the alphabet $\Sigma = \{a, b\}$. Consider the following ϵ-NFA.



7. List all strings of length ≤ 3 that this automaton accepts.

Problems 7 & 8 (ϵ-Nondeterministic Finite Automata; Week 1)

Assume the alphabet $\Sigma = \{a, b\}$. Consider the following ϵ-NFA.



8. Using the subset construction, build a DFA that accepts the same language. Give the graphical representation of this DFA, clearly indicating the set of NFA states associated to each DFA state. Show only the reachable states.

W2 P1

Problem 1 (Week 2)

Give regular expressions for each of the following subsets of $\{a, b\}^*$.

(a) $\{x : x \text{ contains an even number of } a\text{'s}\}$

Problem 1 (Week 2)

Give regular expressions for each of the following subsets of $\{a, b\}^*$.

(b) $\{x : x$ contains an odd number of b's$\}$

Problem 1 (Week 2)

Give regular expressions for each of the following subsets of $\{a, b\}^*$.

(c) $\{x : x$ contains an even number of a's or an odd number of b's$\}$

W2 P2

Problem 2 (Week 2)

Convert the following regular expressions into DFAs:

(a) $(0 \cup 1) \cdot (0 \cup 1) \cdot (0 \cup 1)$

Problem 2 (Week 2)

Convert the following regular expressions into DFAs:

(b) $0^* \cdot 1^*$

Problem 2 (Week 2)

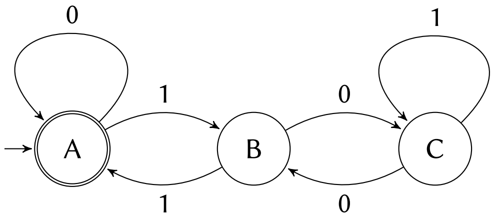
Convert the following regular expressions into DFAs:

(c) $(0 \cup 1^*) \cdot (0^* \cup 1)$

W2 P3

Problem 3 (Week 2)

Give a regular expression that matches the language of the following DFA:



W2 P4

Problem 4 (Week 2)

Show that the following languages are not regular:

(a) $\{0^i 1^j : 0 \leq i \leq j\}$

Problem 4 (Week 2)

Show that the following languages are not regular:

(b) $\{w \in \{0, 1\}^* :$ the number of 0's in w is equal to the number of 1's in $w\}$

Problem 4 (Week 2)

Show that the following languages are not regular:

(c) $\{ww : w \in \{0,1\}^*\}$

Bonus

Discussion

(a) Given an NFA A, how can you determine if $L(A) = \emptyset$?

(b) Using the above process, outline an algorithm that takes two regular expressions E_1 and E_2 and determines if $L(E_1) \subseteq L(E_2)$.

(c) If $|E_1| = n$ and $|E_2| = m$, what is an upper bound (using big-O notation) for the number of states in the DFA that is used in the previous algorithm.