Lecture 2

Last time:
- Finite Automata

Today:
- Finite Automata
- Operations on languages
- Nondeterminism

Tomorrow: Homework 0 due
Homework 1 out

Sofya Raskhodnikova
A language is **regular** if it is recognized by a finite automaton

\[ L = \{ w \mid w \text{ contains 001} \} \text{ is regular} \]
\[ L = \{ w \mid w \text{ has an even number of 1s} \} \text{ is regular} \]

**Many interesting programs recognize regular languages**

- NETWORK PROTOCOLS
- COMPILERS
- GENETIC TESTING
- ARITHMETIC
Let $\text{TCPS} = \{ w \mid w \text{ is a complete TCP Session} \}$

Theorem. $\text{TCPS}$ is regular.
Arithmetic

Let \( \Sigma_3 = \{ [0], [0], [0], [0], [0], [0], [1], [1] \} \)

- A string over \( \Sigma_3 \) has three ROWS
- Each ROW \( b_0b_1b_2\ldots b_N \) represents the integer \( b_0 + 2b_1 + \ldots + 2^Nb_N \).
- Let \( \text{ADD} = \{ S \mid \text{ROW}_1 + \text{ROW}_2 = \text{ROW}_3 \} \)

**Theorem.** \( \text{ADD} \) is regular.
COMMENTS:
- Are delimited by /* */
- Cannot have NESTED /* */
- Must be CLOSED by */
- */ is ILLEGAL outside a comment

COMMENTS = \{\text{strings over \{0,1, /, *\} with legal comments}\}

Theorem. COMMENTS is regular.
DNA SEQUENCES are strings over the alphabet \{A,C,G,T\}.

A GENE \(g\) is a special substring.

A GENETIC TEST searches a DNA SEQUENCE for a gene.

\[
\text{GeneticTest}_g = \{\text{strings over } \{A,C,G,T\} \text{ containing a copy of } g\}
\]

Theorem. GeneticTest\(_g\) is regular for every gene \(g\).
Regular Operations on languages

Complement: \( \neg A = \{ w \mid w \notin A \} \)

Union:

Intersection:

Reverse:

Concatenation:

Star:
\[
= \{ \epsilon \} \cup A \cup AA \cup AAA \cup AAAAA \cup \ldots
\]
THEOREM. The class of regular languages is **closed** under all 6 operations.

If A and B are regular, applying any of these operation yields a regular language.
Complement: $\neg A = \{ w \mid w \notin A \}$

**Theorem.** The complement of a regular language is also a regular language.

**Proof:**
Closure properties

Complement: \( \neg A = \{ w \mid w \notin A \} \)

Union: \( A \cup B = \{ w \mid w \in A \text{ or } w \in B \} \)
**Theorem.** The union of two regular languages is also a regular language.

Proof: Consider two regular languages \( L_1 \) and \( L_2 \). Prove that \( L_1 \cup L_2 \) is regular.

Let \( M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1) \) be finite automaton for \( L_1 \) and \( M_2 = (Q_2, \Sigma, \delta_2, q_0, F_2) \) be finite automaton for \( L_2 \).

Construct a finite automaton \( M = (Q, \Sigma, \delta, q_0, F) \) that recognizes \( L = L_1 \cup L_2 \).
Example

M = ?

Sofya Raskhodnikova; based on slides by Nick Hopper
Proof (continued)

Idea: Run both $M_1$ and $M_2$ at the same time!

$Q = \text{pairs of states, one from } M_1 \text{ and one from } M_2$

$= \{ (q_1, q_2) \mid q_1 \in Q_1 \text{ and } q_2 \in Q_2 \}$

$= Q_1 \times Q_2$

$q_0 = (q_0^1, q_0^2)$

$F = \{ (q_1, q_2) \mid q_1 \in F_1 \text{ or } q_2 \in F_2 \}$

$\delta( (q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))$
Example (continued)

M \_\_\_\_ intersection

M_1

M_2

$q_0, p_0$

$q_1, p_0$

$q_0, p_1$

$q_1, p_1$
Closure properties

Complement: \( \overline{A} = \{ w | w \notin A \} \)

Union: \( A \cup B = \{ w | w \in A \text{ or } w \in B \} \)

Intersection: \( A \cap B = \{ w | w \in A \text{ and } w \in B \} \)

Reverse: \( A^R = \{ w_1 \ldots w_k | w_k \ldots w_1 \in A \} \)
Theorem. The reverse of a regular language is also a regular language.

Proof: Let $L$ be a regular language and $M$ be a finite automaton that recognizes it. Construct a finite automaton $M'$ recognizing $L^R$.

Idea: Define $M'$ as $M$ with the arrows reversed. Swap start and accept states.
Closure under reverse

\[ M' \text{ IS NOT ALWAYS A DFA!} \]

It may have many start states.

Some states may have too many outgoing edges, or none.
Example
What happens with 100?

Nondeterministic Finite Automaton (NFA) accepts if there is a way to make it reach an accept state.
Nondeterminism

Nondeterministic Finite Automaton (NFA) accepts if there is a way to make it reach an accept state.
Example

$L(M) = \{1, 00\}$
Example

\[ L(M) = \{ w \mid w \text{ contains } 101 \text{ or } 11 \} \]
What is the language of this NFA?

\( (0^k \text{ means } 00\ldots0) \)

A. \( \{0^k \mid k \text{ is a multiple of } 2\} \).
B. \( \{0^k \mid k \text{ is a multiple of } 3\} \).
C. \( \{0^k \mid k \text{ is a multiple of } 6\} \).
D. \( \{0^k \mid k \text{ is a multiple of } 2 \text{ or } 3\} \).
E. None of the above.
Formal Definition

- An **NFA** is a 5-tuple \( M = (Q, \Sigma, \delta, q_0, F) \)
  - \( Q \) is the set of states
  - \( \Sigma \) is the alphabet
  - \( \delta : Q \times \Sigma_\varepsilon \rightarrow P(Q) \) is the transition function
  - \( q_0 \in Q \) is the start state
  - \( F \subseteq Q \) is the set of accept states
- \( P(Q) \) is the set of subsets of \( Q \) and \( \Sigma_\varepsilon = \Sigma \cup \{\varepsilon\} \)
- **M** accepts a string \( w \) if there is a path from \( q_0 \) to an accept state that \( w \) follows.
Example

\[ N = (Q, \Sigma, \delta, q_0, F) \]

\[ Q = \{q_0, q_1, q_2, q_3\} \]

\[ \Sigma = \{0, 1\} \]

\[ F = \{q_3\} \]

\[ \delta(q_0, 0) = \{q_0\} \]

\[ \delta(q_0, 1) = \{q_0, q_1\} \]

\[ \delta(q_1, \varepsilon) = \{q_1, q_2\} \]

\[ \delta(q_2, 0) = \emptyset \]
Nondeterminism

Ways to think about nondeterminism
- parallel computation
- tree of possible computations
- guessing and verifying the “right” choice

Deterministic Computation
- accept or reject

Nondeterministic Computation
- accept
- reject

9/7/2017
Sofya Raskhodnikova; based on slides by Nick Hopper
A DFA that recognizes the language \{1\}:

An NFA that recognizes the language \{1\}:
Theorem. Every DFA for language \{1\} must have at least 3 states.

Proof: