Intro to Theory of Computation

Lecture 3

Last time:
- DFAs and NFAs
- Operations on languages

Today:
- Nondeterminism
- Equivalence of NFAs and DFAs
- Closure properties of regular languages

Sofya Raskhodnikova

Sofya Raskhodnikova; based on slides by Nick Hopper
What is the language of this NFA?

\( \{0^k \mid k \text{ is a multiple of } 2\} \).

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.
Nondeterministic Finite Automaton (NFA) accepts a string $w$ if there is a way to make it reach an accept state on input $w$. 
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
  - \( \delta : Q \times \Sigma \epsilon \rightarrow P(Q) \) is the transition function
  - \( q_0 \in Q \) is the start state
  - \( F \subseteq Q \) is the set of accept states
- \( \Sigma \epsilon = \Sigma \cup \{\epsilon\} \) and \( P(Q) \) is the set of subsets of \( Q \)
- \( M \) **accepts** a string \( w \) if there is a path from \( q_0 \) to an accept state that \( w \) follows.

Sofya Raskhodnikova; based on slides by Nick Hopper
Example

Let $N = (Q, \Sigma, \delta, q_0, F)$ be a deterministic finite automaton (DFA), where:

- $Q = \{q_0, q_1, q_2, q_3\}$
- $\Sigma = \{0, 1\}$
- $F = \{q_3\}$
- $\delta(q_0, 0) = \{q_0\}$
- $\delta(q_0, 1) = \{q_0, q_1, q_2\}$
- $\delta(q_1, \varepsilon) = \{q_1, q_2\}$
- $\delta(q_2, 0) = \emptyset$
Nondeterminism

Deterministic Computation

accept or reject

Nondeterministic Computation

• parallel computation
• tree of possible computations
• guessing and verifying the “right” choice

Ways to think about nondeterminism

accept
reject

Sofya Raskhodnikova; based on slides by Nick Hopper
NFAs ARE SIMPLER THAN DFAs

A DFA that recognizes the language \{1\}: 

\[
\begin{array}{c}
\text{state 1} \\
\text{state 2} \\
\text{state 3} \\
\text{state 4}
\end{array}
\]

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

Proof:
Theorem. Every NFA has an equivalent DFA.

Corollary: A language is regular iff it is recognized by an NFA.
NFA to DFA Conversion

Input: \( N = (Q, \Sigma, \delta, q_0, F) \)

Output: \( M = (Q', \Sigma, \delta', q_0', F') \)

Intuition: Do the computation in parallel, maintaining the set of states where all threads are.

Idea:
\[ Q' = P(Q) \]
NFA to DFA Conversion

**Input:** \( N = (Q, \Sigma, \delta, q_0, F) \)

**Output:** \( M = (Q', \Sigma, \delta', q_0', F') \)

\[ Q' = P(Q) \]
\[ \delta' : Q' \times \Sigma \rightarrow Q' \]
\[ \delta'(R, \sigma) = \text{for all } R \subseteq Q \text{ and } \sigma \in \Sigma. \]

\[ q_0' = \]
\[ F' = \]
Example: NFA to DFA

1) 

\[
\begin{align*}
& a \\
\rightarrow & 1 \\
\rightarrow & b
\end{align*}
\]
Examples NFA to DFA

2)

Diagram:

- States: 1, 2, 3
- Transitions:
  - 1 to 2 on ε
  - 2 to 3 on 0
  - 2 to 1 on 1

Note: The diagram is a visual representation of a non-deterministic finite automaton (NFA) and its deterministic finite automaton (DFA) equivalent.
NFA to DFA Conversion

Input: \( N = (Q, \Sigma, \delta, q_0, F) \)
Output: \( M = (Q', \Sigma, \delta', q_0', F') \)

For \( R \subseteq Q \), let \( E(R) \) be the set of states reachable by \( \varepsilon \)-transitions from the states in \( R \).

\[ Q' = P(Q) \]
\[ \delta' : Q' \times \Sigma \rightarrow Q' \]
\[ \delta'(R, \sigma) = \bigcup_{r \in R} (\delta(r, \sigma)) \quad \text{for all } R \subseteq Q \text{ and } \sigma \in \Sigma. \]
\[ q_0' = \{ \{q_0\} \} \]
\[ F' = \{ R \in Q' \mid R \text{ contains some accept state of } N \} \]
Complement: \( \overline{A} = \{ w \mid w \notin A \} \)

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

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

Reverse: \( A^R = \{ w_1 \ldots w_k \mid w_k \ldots w_1 \in A \} \)

Concatenation: \( A \circ B = \{ vw \mid v \in A \text{ and } w \in B \} \)

Star: \( A^* = \{ w_1 \ldots w_k \mid k \geq 0 \text{ and each } w_i \in A \} \)
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.
Palindromes

A palindrome is a word or a phrase that reads the same forward and backward.

Examples

• mom
• madam
• Never odd or even.
• Stressed? No tips? Spit on desserts!
Exercise

Let $L$ be the set of words in English. Then $L \cap L^R$ is

A. The set of English words in alphabetical order, followed by the same words in reverse alphabetical order.

B. $\{w \mid w$ is an English word or an English word written backwards$\}$.

C. $\{w \mid w$ is an English word that is a palindrome$\}$.

D. None of the above.
Theorem. The reverse of a regular language is also regular.

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

- Define $M'$ as $M$ with the arrows reversed.
- Make the start state of $M$ be the accept state in $M'$.
- Make a new start state that goes to all accept states of $M$ by $\varepsilon$-transitions.
New construction for $A \cup B$

Construct an NFA $M$:

$L(M_A) = A$

$L(M_B) = B$
Concatenation operation

Concatenation: \(A \circ B = \{ vw \mid v \in A \text{ and } w \in B \}\)

**Theorem.** If \(A\) and \(B\) are regular, \(A \circ B\) is also regular.

**Proof:** Given DFAs \(M_1\) and \(M_2\), construct NFA by connecting all accept states in \(M_1\) to the start state in \(M_2\).
Concatenation operation

**Concatenation**: \( A \circ B = \{ vw \mid v \in A \text{ and } w \in B \} \)

**Theorem.** If \( A \) and \( B \) are regular, \( A \circ B \) is also regular.

**Proof:** Given DFAs \( M_1 \) and \( M_2 \), construct NFA by connecting all accept states in \( M_1 \) to the start state in \( M_2 \).

- Make all states in \( M_1 \) non-accepting.

\[ L(M_1) = A \]
\[ L(M_2) = B \]
Star operation

Star: $A^* = \{ w_1 \ldots w_k \mid k \geq 0 \text{ and each } w_i \in A \}$

**Theorem.** If $A$ is regular, $A^*$ is also regular.
The class of regular languages is closed under

**Regular operations**

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

**Concatenation:** \( A \circ B = \{ vw \mid v \in A \text{ and } w \in B \} \)

**Star:** \( A^* = \{ w_1 \ldots w_k \mid k \geq 0 \text{ and each } w_i \in A \} \)

**Other operations**

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

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

**Reverse:** \( A^R = \{ w_1 \ldots w_k \mid w_k \ldots w_1 \in A \} \)