Lecture 3:

• Deterministic Finite Automata
• Non-deterministic FAs

Reading:
Sipser Ch 1.1-1.2

Mark Bun
February 1, 2021
Last Time

• Parts of a theory of computation: Model for machines, model for problems, theorems relating machines and problems

• Strings: Finite concatenations of symbols

• Languages: Sets $L$ of strings

• Computational (decision) problem: Given a string $x$, is it in the language $L$?
Deterministic Finite Automata
A (Real-Life?) Example

• Example: Kitchen scale

• \( P = \) Power button (\textit{ON} / \textit{OFF})

• \( U = \) Units button (cycles through \( g / oz / lb \))

  Only works when scale is \textit{ON}, but units remembered when scale is \textit{OFF}

• Starts \textit{OFF} in \( g \) mode

• A computational problem: Does a sequence of button presses in \{\( P, U \}\}^* leave the scale \textit{ON} in \textit{oz} mode?
Machine Models

• **Finite Automata (FAs):** Machine with a finite amount of unstructured memory

  Input: $P\, U\, P\, U \ldots$

  Control scans left-to-right

  1. Control can take on finite # of "states"

  2. Control "transitions" between states

  "state diagram"
A DFA for the Kitchen Scale Problem
A DFA Recognizing Parity

The language recognized by a DFA is the set of inputs on which it ends in an “accept” state.

Parity: Given a string consisting of $a$’s and $b$’s, does it contain an even number of $a$’s?

$\Sigma = \{a, b\} \quad L = \{w \mid w \text{ contains an even number of } a\text{'s}\}$

Which state is reached by the parity DFA on input $aabab$?

a) “even”
b) “odd”
Anatomy of a DFA

States
transitions
start states
accept states (final states)
Some Tips for Thinking about DFAs

Given a DFA, what language does it recognize?
- Try experimenting with it on short strings. Do you notice any patterns?
- What kinds of inputs cause the DFA to get trapped in a state?

Given a language, construct a DFA recognizing it
- Imagine you are a machine, reading one symbol at a time, always prepared with an answer
- What is the essential information that you need to remember? Determines set of states.
What language does this DFA recognize?

\[ \exists w \mid w \text{ contains substring } 001 \overline{1} \]
Practice!

- Lots of worked out examples in Sipser
- Tomorrow’s discussion section
- Automata Tutor: https://automata-tutor.model.in.tum.de/
Formal Definition of a DFA

A finite automaton is a 5-tuple \( M = (Q, \Sigma, \delta, q_0, F) \)

- \( Q \) is the set of states
- \( \Sigma \) is the alphabet
- \( \delta: Q \times \Sigma \rightarrow Q \) is the transition function
- \( q_0 \in Q \) is the start state
- \( F \subseteq Q \) is the set of accept states
A DFA for Parity

Parity: Given a string consisting of $a$’s and $b$’s, does it contain an even number of $a$’s?

$\Sigma = \{a, b\}$  \quad $L = \{w \mid w \text{ contains an even number of } a$’s$\}$

State set $Q = \{q_0, q_1, q_2\}$
Alphabet $\Sigma = \{a, b\}$
Transition function $\delta$

\[
\begin{array}{c|cc}
\delta & a & b \\
\hline
q_0 & q_1 & q_0 \\
q_1 & q_0 & q_1 \\
\end{array}
\]

Start state $q_0$
Set of accept states $F = \{q_0, q_2\}$
Formal Definition of DFA Computation

A DFA $M = (Q, \Sigma, \delta, q_0, F)$ accepts a string $w = w_1 w_2 \cdots w_n \in \Sigma^*$ (where each $w_i \in \Sigma$) if there exist $r_0, \ldots, r_n \in Q$ such that

1. $r_0 = q_0$
2. $\delta(r_i, w_{i+1}) = r_{i+1}$ for each $i = 0, \ldots, n - 1$, and
3. $r_n \in F$

$L(M) = \text{the language of machine } M$

= set of all strings machine $M$ accepts

$M$ recognizes the language $L(M)$
Example: Computing with the Parity DFA

Let \( w = abba \)

Does \( M \) accept \( w \)?

What is \( \delta(r_2, w_3) \)?

a) \( q_0 \)

b) \( q_1 \)

A DFA \( M = (Q, \Sigma, \delta, q_0, F) \) accepts a string \( w = w_1 w_2 \cdots w_n \in \Sigma^* \) (where each \( w_i \in \Sigma \)) if there exist \( r_0, \ldots, r_n \in Q \) such that

1. \( r_0 = q_0 \)
2. \( \delta(r_i, w_{i+1}) = r_{i+1} \) for each \( i = 0, \ldots, n - 1 \), and
3. \( r_n \in F \)
Regular Languages

**Definition:** A language is regular if it is recognized by a DFA

\[ L = \{ w \in \{a, b\}^* \mid w \text{ has an even number of } a\text{'s} \} \text{ is regular} \]

\[ L = \{ w \in \{0, 1\}^* \mid w \text{ contains 001} \} \text{ is regular} \]

\[ \text{Not regular: } \exists a^n b^n \mid n \neq 0 \]

Many interesting programs recognize regular languages

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

**Theorem.** TCPS is regular
Comments:
Are delimited by /* */
Cannot have nested /* */
Must be closed by */
*/ is illegal outside a comment

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

Theorem. COMMENTS is regular.
Genetic Testing

DNA sequences are strings over the alphabet \( \{A, C, G, T\} \).

\[
W = A A G C T G C T
\]

A gene \( g \) is a special substring over this alphabet.

\[
g = G C T
\]

A genetic test searches a DNA sequence for a gene.

Is \( g \) a substring of \( W \)?

\[
\text{GENETICTEST}_g = \{ \text{strings over } \{A, C, G, T\} \text{ containing } g \text{ as a substring} \}
\]

Theorem. GENETICTEST\(_g\) is regular for every gene \( g \).
Arithmetic

LET \( \Sigma = \{ \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \} \)

- A string over \( \Sigma \) has three ROWS (ROW\(_1\), ROW\(_2\), ROW\(_3\))
- Each ROW \( b_0 b_1 b_2 \ldots b_N \) represents the integer \( b_0 + 2b_1 + \ldots + 2^N b_N \).
- Let ADD = \( \{ S \in \Sigma^* \mid \text{ROW}_1 + \text{ROW}_2 = \text{ROW}_3 \} \)

**Theorem.** ADD is regular.
Nondeterministic Finite Automata
Non-determinism

In a DFA, the machine is always in exactly one state upon reading each input symbol.

In a non-deterministic FA, the machine can try out many different ways of reading the same string:
- Next symbol may cause an NFA to “branch” into multiple possible computations.
- Next symbol may cause NFA’s computation to fail to enter any state at all.

\(\text{(NFA)}\)
Nondeterminism

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

Example: Does this NFA accept the string 1100?  YES
Some special transitions

- Multiple branching: A circle connected to more than one other circle.

- No branching: A circle connected to another circle with a single arrow, indicating no further branching.

- Epsilon (ε) transition: A circle connected to another circle without any arrow label, indicating a transition without consuming any input.
Example

\[ L(M) = \{ w \mid w \text{ ends in 1 or } w \text{ in 00} \} \]