Theory of Automata

Lecture 1: Introduction

Dr. Sabina Akhtar

1

About me

- MS(Computer Science)
  - FAST-NU Islamabad (2002-2005)

- MS(Computer Science)
  - Université de Lorraine, Nancy, France (2007-2008)

UNIVERSITÉ DE LORRAINE

2

About me

- Ph.d (Formal Verification)
  - Researched at INRIA Nancy
  - Université de Lorraine, Nancy, France (2008-2012)

3

Contact details

- Email: sabina.buic@bahria.edu.pk
- Office hours:
  - To be decided.

4

Some Rules

- Everyone must wear the mask
- Attendance during first 10 minutes for each hour
- After that you will be marked as absent
- Raise your hand before asking any question and then WAIT for the permission
- Must wear your ID card in the class

5

Some Rules

- Submission must be done on time
- No compensation for the missed quizzes & assignments
- Keep mobile phones on silent mode in the class
- Whatever you do, please do not create disturbance in the class

6

Dishonesty, Plagiarism

- Students involved in any kind of cheating in any exam (Quizzes/Assignments) will get **0 (Zero)** in that exam OR get F in the course
- In case of quizzes with cheating case, their weightage will be highest.

7

Pre-Requisites

- None required!!
- But you must ask yourself
  - From where we got the idea of building machine that computes,
  - How a computer works,

8

Tentative Evaluation Breakdown

Total	**100**
Quizzes	10
Assignments	20
Mid-Term Examination	20
Final Examination	50

9

Quizzes and Assignments

- 4 Quizzes
  - Will always be a surprise
- 4 Assignments
  - No late submissions will be accepted!!!!

10

Course outline

Week #	Topic
1	Introduction to Finite Automata
2	Deterministic/Nondeterministic Finite Automata
3	Equivalence of NFA, DFA
4	Regular Expressions & Languages
5	Pumping Lemma & Closure Properties for RL
6	Transducers
7	Introduction to Context Free Grammar
8	Derivations & Ambiguous Grammars
9	**Midterms**

11

Course outline

Week #	Topic
10	Pumping lemma & Closure properties for CFL
11	Introduction to Pushdown Automata
12	Parsing & PDA
13	Introduction to Turing Machine
14	Examples of TM and Variations of TM
15	TM encoding, Universal TM & Computer vs TM
16	Decidability
17	Revision
18	**Final Exams**

12

Books

- Text Book:
  - Introduction to Automata Theory, Languages, and Computation, 3/e by John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, Pearson Ed., 2009.
  - Introduction to the Theory of Computation by Michael Sipser, Third edition, 2013 Cengage Learning.
  - An introduction to formal languages and automata, fifth edition by Peter Linz, 2011.
- Web Reference:
  - Lectures from Stanford University by D. Ullman. Link: http://infolab.stanford.edu/~ullman/ialc/spr10/spr10.html#LECTURE%20NOTES
  - Lectures from Washington State Universityhttp://www.eecs.wsu.edu/~ananth/CptS317/Lectures/

13

13

What is Automata? And Why study?

14

Example: Finite automata for a switch

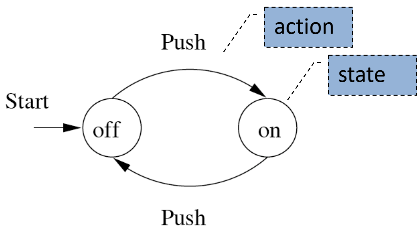
- Specifications:
  - A switch can only have 2 possible states
  - Initially it’s in “Off” state
  - When the it is pushed it turns the power on and moves to the “On” state
  - If it is pushed at the “On” state then it turns the power off and moves to “Off” state
- How can we model it using a finite automata?

15

16

Finite Automata : Examples

- Design automata for On/Off switch



17

Example

- If you were to design a system that
  - Takes any two numbers
  - Performs addition
  - and, stores the result
- Questions:
  - There are certain states for your system, what are they?
  - How would you model your system?

18

19

Informal Explanation

- Finite automata are finite collections of states with transition rules that take you from one state to another.
- Original application was sequential switching circuits, where the “state” was the settings of internal bits.
- Today, several kinds of software can be modeled by FA.

20

Finite Automata

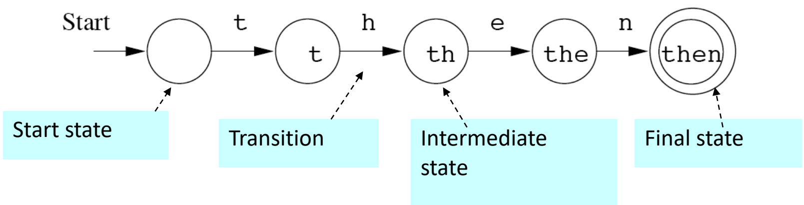
- Some Applications
  - Software for designing and checking the behavior of digital circuits
  - Lexical analyzer of a typical compiler
  - Software for scanning large text (e.g., web pages) for pattern finding
  - Software for verifying systems of all types that have a finite number of states (e.g., distributed systems, communication/network protocol,…)

Representing FA

- Simplest representation is often a graph.
  - Nodes = states.
  - Arcs indicate state transitions.
  - Labels on arcs tell what causes the transition.

Finite Automata : Examples

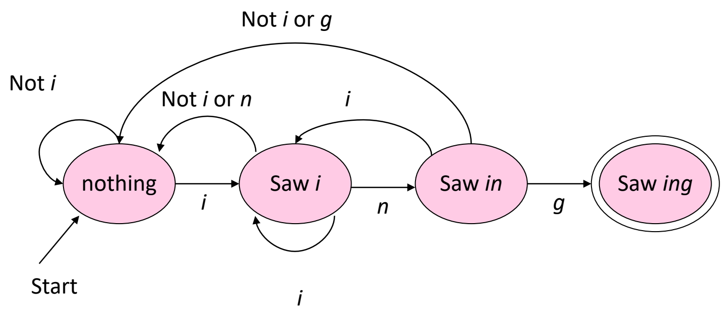
- Modeling recognition of the word *"then"*



23

Class Activity:

- Design an automata recognizing Strings Ending in "ing"



24

Automata to Code

- In C/C++, make a piece of code for each state. This code:
  1. Reads the next input.
  2. Decides on the next state.
  3. Jumps to the beginning of the code for that state.

25

Example: Automata to Code

```
2: /* i seen */
  c = getNextInput();
  if (c == 'n') goto 3;
  else if (c == 'i') goto 2;
  else goto 1;
3: /* "in" seen */
  . . .
```

26

Automata to Code – Thoughts

- How would you do this in Java, which has no goto?
- You don't really write code like this.
- Rather, a code generator takes a "regular expression" describing the pattern(s) you are looking for.
  - Example: `.*ing` works in grep

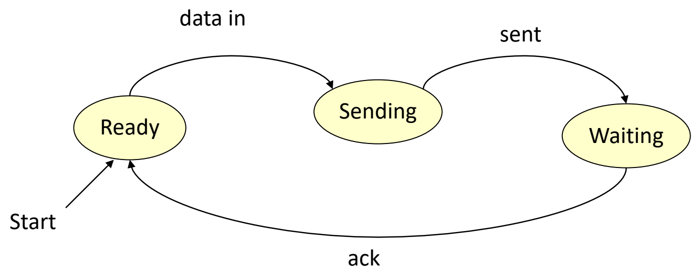
27

Class Activity

- Draw a finite automata for the following protocol
  - System is in Ready state at the start.
  - When it receives some data to send, it moves to Sending state and starts sending the data.
  - It moves to Waiting state when it has sent the data to receive an acknowledgement
  - Once the acknowledgement is received it moves back to Ready state.

28

Solution



29

Terminologies

- Symbol
- Alphabet
- String
- Length of a string
- Language
- Kleene's closure
- Positive closure

30

Alphabet

- An *alphabet* is any finite set of symbols. Represented as **Σ.**
- Examples:
  - ASCII,
  - {0,1} (*binary alphabet*),
  - {a,b,c}
  - {a,b,c,…,z}

31

String

- A string is a finite sequence of symbols chosen from some alphabet **Σ.**
  - E.g., 01101 is a string from $\Sigma=\{0,1\}$
  - Strings shown with no commas, e.g., abc.
- Σ^* denotes all the set of strings.
- ϵ stands for the *empty string* (string of length 0).
- |S| : Length of a string S

32

Example: Strings

- $\{0,1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, . . . \}$
- Subtlety: 0 as a string, 0 as a symbol look the same.
  - Context determines the type.

33

Powers of an Alphabet

Let Σ be an alphabet.

- Σ^k = the set of all strings of length k
- $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots$
- $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$

Are these sets finite or infinite?

34

Example

Example 1.24: Note that $\Sigma^0 = \{\epsilon\}$, regardless of what alphabet Σ is. That is, ϵ is the only string whose length is 0.

If $\Sigma = \{0, 1\}$, then $\Sigma^1 = \{0, 1\}$, $\Sigma^2 = \{00, 01, 10, 11\}$,

- What is Σ^3?

$$\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$$

35

Class Activity

- If $\Sigma = \{a,b,c\}$ then provide
  - Σ^0
  - Σ^1
  - Σ^2
  - Σ^3

36

Kleene's closure

- Given Σ, then the Kleene's closure of an alphabet Σ, denoted by Σ^*, is the collection of all strings defined over Σ, including ϵ
- Examples
  - Σ = {0}
    Σ^* = {ϵ, 0, 00, 000, 0000, ...}
  - Σ = {0,1}
    Σ^* ={ϵ,0,1,00,01,10,11,....}
  - Σ = {a, b, c}
  - Σ^* = {ϵ, a, b, c, aa, ab, ac, bc, ...}

37

Positive closure

- Given Σ, then the Kleene's closure of an alphabet Σ, denoted by Σ^+, is the collection of all strings defined over Σ, excluding ϵ
- Examples
  - Σ = {0}
    Σ^+ = {0, 00, 000, 0000, ...}
  - Σ = {0,1}
    Σ^+ ={0,1,00,01,10,11,....}
  - Σ = {a, b, c}
  - Σ^+ = {a, b, c, aa, ab, ac, bc, ...}

38

Language

- A *language* is a subset of Σ* for some alphabet Σ.

  ➔ this is because ∑* is the set of all strings (of all possible length including 0) over the given alphabet ∑

- Example: The set of strings of 0's and 1's with no two consecutive 1's.
- L = {ϵ, 0, 1, 00, 01, 10, 000, 001, 010, 100, 101, 0000, 0001, 0010, 0100, 0101, 1000, 1001, 1010, . . . }

39

Class Activity

What is the subset? Provide atleast 8 possible strings.

- Let L be *the* language of <u>all strings consisting of *n* 0's followed by *n* 1's.</u>
- Let L be *the* language of <u>all strings of with equal number of 0's and 1's</u>:

- L = {ε,01,0011,000111,…}
- L = {ε,01,10,0011,1100,0101,1010,1001,…}

40

Other language examples

- θ, the empty language, is a language over any alphabet
- {ε}, the language consisting of only the empty string, is also a language over any alphabet
  - NOTE: θ ≠ {ε} since θ has no strings and {ε} has one
- {w | w consists of an equal number of 0 and 1}
- $\{0^n 1^n \mid n \geq 1\}$
- $\{0^i 1^j \mid 0 \leq i \leq j\}$

Write a java program to distinguish between θ and {ε}

41

The Membership Problem

*Given a string w ∈ ∑*and a language L over ∑, decide whether or not w ∈L.*

Example:

Let w = 100011

Q) Is w ∈ the language of strings with equal number of 0s and 1s?

42

42

Class Activity

- Give possible set of strings for the following languages over the alphabet {0, 1}.
  1. The set of all strings ending in 00.
  2. The set of all strings with three consecutive 0's (not necessarily at the end).
  3. The set of strings with 011 as a substring.

43

DETERMINISTIC FINITE AUTOMATA

44

Deterministic Finite Automata

A formalism for defining languages, consisting of:

1. A finite set of *states* (Q, typically).
2. An *input alphabet* (Σ, typically).
3. A *transition function* (δ, typically).
4. A *start state* (q_0, in Q, typically).
5. A set of *final states* ($F \subseteq Q$, typically).
   - "Final" and "accepting" are synonyms.

45

45

The Transition Function

- Takes two arguments: a state and an input symbol.
- δ(q, a) = the state that the DFA goes to when it is in state *q* and input *a* is received.

46

46

Graph Representation of DFA's

- Nodes = states.
- Arcs represent transition function.
  - Arc from state p to state q labeled by all those input symbols that have transitions from p to q.
- Arrow labeled "Start" to the start state.
- Final states indicated by double circles.

47

47

Example #1

Example 2.1: Let us formally specify a DFA that accepts all and only the strings of 0's and 1's that have the sequence 01 somewhere in the string. We can write this language L as:

$$\{w \mid w \text{ is of the form } x01y \text{ for some strings } x \text{ and } y \text{ consisting of 0's and 1's only}\}$$

48

Example #1

- Build a DFA for the following language:
  - L = {w | w is a binary string that contains 01 as a substring}
- Steps for building a DFA to recognize L:
  - ∑ = {0,1}
  - Decide on the states: Q
  - Designate start state and final state(s)
  - δ: Decide on the transitions:
- "Final" states == same as "accepting states"
- Other states == same as "non-accepting states"

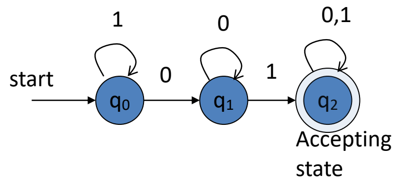
49

49

Solution

DFA for strings containing 01

- What makes this DFA deterministic?



Formal description

- $Q = \{q_0, q_1, q_2\}$
- $\sum = \{0,1\}$
- start state = q_0
- $F = \{q_2\}$
- Transition fucntion?

symbols

states

δ	0	1
→ q_0	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_2	q_2

50

51

Class Activity

Provide DFA for the following languages over the alphabet {0, 1}.

1. The set of all the strings ending in 00.
2. The set of all the strings with three consecutive 0's (not necessarily at the end).
3. The set of all the strings with 011 as a substring.
4. The set of all the strings whose 3^{rd} symbol from the left is 0.

52

Extended Transition Function

- We describe the effect of a string of inputs on a DFA by extending δ to a state and a string.
- Induction on length of string.
- Basis: $\hat{\delta}(q, \epsilon) = q$
- Induction: $\hat{\delta}(q,wa) = \delta(\hat{\delta}(q,w),a)$
  - w is a string; a is an input symbol.

53

53

Example

	0	1
$* \rightarrow q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Show the processing of extended delta for input 110101 using the above transition table.

54

Example

	0	1
$* \to q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

55

Example: Solution

	0	1
$* \to q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

- $\hat{\delta}(q_0, \epsilon) = q_0$.

56

Example: Solution

	0	1
$* \rightarrow q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

- $\hat{\delta}(q_0, \epsilon) = q_0$.
- $\hat{\delta}(q_0, 1) = \delta\big(\hat{\delta}(q_0, \epsilon), 1\big) = \delta(q_0, 1) = q_1$.

57

Example: Solution

	0	1
$* \rightarrow q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

- $\hat{\delta}(q_0, \epsilon) = q_0$.
- $\hat{\delta}(q_0, 1) = \delta\big(\hat{\delta}(q_0, \epsilon), 1\big) = \delta(q_0, 1) = q_1$.
- $\hat{\delta}(q_0, 11) = \delta\big(\hat{\delta}(q_0, 1), 1\big) = \delta(q_1, 1) = q_0$.

58

Example: Solution

	0	1
$* \to q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

- $\hat{\delta}(q_0, \epsilon) = q_0$.
- $\hat{\delta}(q_0, 1) = \delta\big(\hat{\delta}(q_0, \epsilon), 1\big) = \delta(q_0, 1) = q_1$.
- $\hat{\delta}(q_0, 11) = \delta\big(\hat{\delta}(q_0, 1), 1\big) = \delta(q_1, 1) = q_0$.
- $\hat{\delta}(q_0, 110) = \delta\big(\hat{\delta}(q_0, 11), 0\big) = \delta(q_0, 0) = q_2$.

59

Example: Solution

	0	1
$* \to q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

- $\hat{\delta}(q_0, \epsilon) = q_0$.
- $\hat{\delta}(q_0, 1) = \delta\big(\hat{\delta}(q_0, \epsilon), 1\big) = \delta(q_0, 1) = q_1$.
- $\hat{\delta}(q_0, 11) = \delta\big(\hat{\delta}(q_0, 1), 1\big) = \delta(q_1, 1) = q_0$.
- $\hat{\delta}(q_0, 110) = \delta\big(\hat{\delta}(q_0, 11), 0\big) = \delta(q_0, 0) = q_2$.
- $\hat{\delta}(q_0, 1101) = \delta\big(\hat{\delta}(q_0, 110), 1\big) = \delta(q_2, 1) = q_3$.
- $\hat{\delta}(q_0, 11010) = \delta\big(\hat{\delta}(q_0, 1101), 0\big) = \delta(q_3, 0) = q_1$.
- $\hat{\delta}(q_0, 110101) = \delta\big(\hat{\delta}(q_0, 11010), 1\big) = \delta(q_1, 1) = q_0$.

60

Delta-hat

- In book, the extended δ has a "hat" to distinguish it from δ itself.
- Not needed, because both agree when the string is a single symbol.
- $\hat{\delta}(q, a) = \delta(\hat{\delta}(q, \epsilon), a) = \delta(q, a)$

Extended deltas

61

61

Example: Extended Delta

	0	1
→ A	A	B
B	A	C
*C	C	C

Show the processing of extended delta for input 011 using the above transition table.

$\hat{\delta}(A,011) = ?$

62

Class Activity

	0	1
→ A	A	B
B	A	C
*C	C	C

Show the processing of extended delta using the above transition table.

$\hat{\delta}(B,0010110) = ?$

63

Language of a DFA

- Automata of all kinds define languages.
- If A is an automaton, L(A) is its language.
- For a DFA A, L(A) is the set of strings labeling paths from the start state to a final state.
- Formally: L(A) = the set of strings w such that $\delta(q_0, w)$ is in F.
- *i.e.,* $L(A) = \{ w \mid \hat{\delta}(q_0,w) \in F \}$

64

64

What does a DFA do on reading an input string?

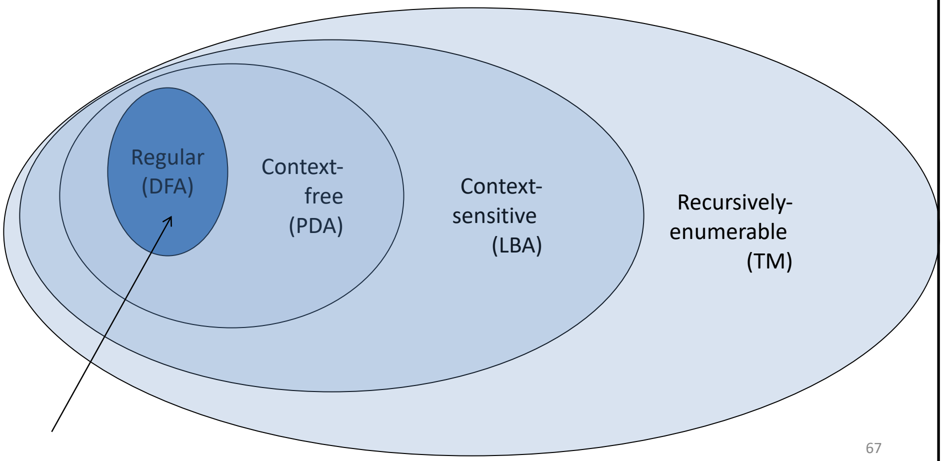
- Input: a word w in ∑*
- Question: Is w acceptable by the DFA?
- Steps:
  - Start at the "start state" q_0
  - For every input symbol in the sequence w do
    - Compute the next state from the current state, given the current input symbol in w and the transition function
  - If after all symbols in w are consumed, the current state is one of the accepting states (F) then *accept w;*
  - Otherwise, *reject w.*

Regular Languages

- Let L(A) be a language *recognized* by a DFA A.
  - Then L(A) is called a "*Regular Language".*
- Locate regular languages in the Chomsky Hierarchy

The Chomsky Hierachy

- A containment hierarchy of classes of formal languages



67

67

Example #2

Clamping Logic:

- A clamping circuit waits for a "1" input, and turns on forever. However, to avoid clamping on spurious noise, we'll design a DFA that waits for *two consecutive 1s* in a row before clamping on.

- Build a DFA for the following language:

  L = { w | w is a bit string which contains the substring 11}

- State Design:
  - q_0 : start state (initially off), also means the most recent input was not a 1
  - q_1: has never seen 11 but the most recent input was a 1
  - q_2: has seen 11 at least once

68

68

Example #3

- Build a DFA for the following language:
  L = { w | w is a binary string that has even number of 1s}
- ?

Example #4

- Build a DFA for the following language:
  L = { w | w is a binary string that has odd number of 0s}
- ?

References

- Book Section Chapter 1 and 2
- Lectures from Stanford University
  – http://infolab.stanford.edu/~ullman/ialc/spr10/spr10.html#LECTURE%20NOTES
- Lectures from Washington State University
  – http://www.eecs.wsu.edu/~ananth/CptS317/Lectures/

71