Theory of Automata

Hafiz Tayyeb Javed

Week 1 Lecture-01

Contents

- Instructor
- Outline
- Text & reference Material
- Grading
- Objectives
- Background
- Motivation & Description

Instructor

- Hafiz Tayyeb Javed
  - Assistant Professor
  - Office – 204A
  - Telephone – Ext: 125
  - Email
    - tayyeb.javed@nu.edu.pk
  - Office hours
    - 09:30 hrs to 05:00hrs – Monday to Friday

Course Outline

Finite State Models

- Language definitions preliminaries
- Regular expressions/Regular languages
- Finite automata (FAs)
- Transition graphs (TGs)
- NFAs, kleene's theorem
    - *Making a Scanner*
- Transducers (automata with output)
- Pumping lemma and non regular language

Grammars and PDA

- Context free grammars
- Derivations, derivation trees and ambiguity
- Operations preserving CFLs
- Normal form grammars and parsing

Push Down Automata

- *Making a Parser*
- Pumping lemma and non-context free languages
- Decidability
- Chomsky's hierarchy of grammars

Turning Machines Theory

- Turing machines
- Post machine
    - *Turing Machine project*
- Variations on TM
- TM encoding, Universal Turing Machine
- Context sensitive Grammars
- Defining Computers by TMs

Text and Reference Material

1. *Introduction to Computer Theory*, by Daniel I. Cohen, John Wiley and Sons, Inc., 1991, Second Edition
2. *Introduction to Languages and Theory of Computation*, by J. C. Martin, McGraw Hill Book Co., 1997, Second Edition
3. Class Lecture Slides & Handouts
4. Class Room Sessions & Discussions

Tentative Grading

• Mid-Term Exams (1)	25-30
• Assignments	10
• Quizzes	10
• Final Exams	50-55

– Marks division might change during the semester
– Grading will be absolute:
– Additive or Multiplicative.

Objectives

- The course aims to develop an appreciation of theoretical foundations of computer science through study of mathematical & abstract models of computers and theory of formal languages

Background

- We shall study several **mathematical models** that will describe with varying degrees of accuracy parts of computers, types of computers, and similar machines.
- In particular, the way we shall be studying about computers is to build mathematical models, called **machines**, and then to study their limitations by analyzing the types of inputs on which they can operate successfully.
- The collection of these successful inputs is called the **language** of the machine.

- Every time we introduce a new machine, we will learn its language; and every time we develop a new language, we will try to find a machine that corresponds to it. (Language vis-à-vis Machine)
- This interplay between languages and machines will be our way of investigating problems and their potential solutions by automatic procedures, which we call **algorithms**.
- We will study different types of theoretical machines that are mathematical models for actual physical processes. By considering the possible inputs on which these machines can work, we can analyze their various strengths and weaknesses.

What is mathematical about the models?

We are obliged to **prove** the truth about whatever we discover.

- So consider only question of whether certain tasks can be done at all. Conclusion are of the form “this can be done” or “this can never be done”.
- When we reach conclusions of the second type, we mean not just that techniques for performing these tasks are unknown at the present time, but that such techniques will never exist in the future.

- We will arrive at what we may believe to be the most powerful machine possible. When we do, we will be surprised to find tasks that even such machine cannot perform.
- Our ultimate result is that no matter what machine we build, there will always be questions that are simple to state and that the machine can not answer.
- Limitation of a machine / algorithm / mathematical model / computer
  - Vision/hear/smell/touch - emotions/intuition

What does automata mean?

It is the plural of automaton, and it means "something that works automatically"

What is a Computer?

What is an Abstract Model of a Computer?

What was before Computers?

Search for Algorithms for solving a math problems; What it means?

Which Statement has proof and how can we generate these proofs?

People & Problem

- Alanzo Church,
- Stephen Kleene,
- Jhon Van Neuman and
- Alan Turing
- Noam Chomsky
- An extraordinary simple set of building blocks from which all mathematical algorithms can be comprised.
- They Fashioned (various versions) of a Universal Algorithm Machine; Universal Model for all algorithms

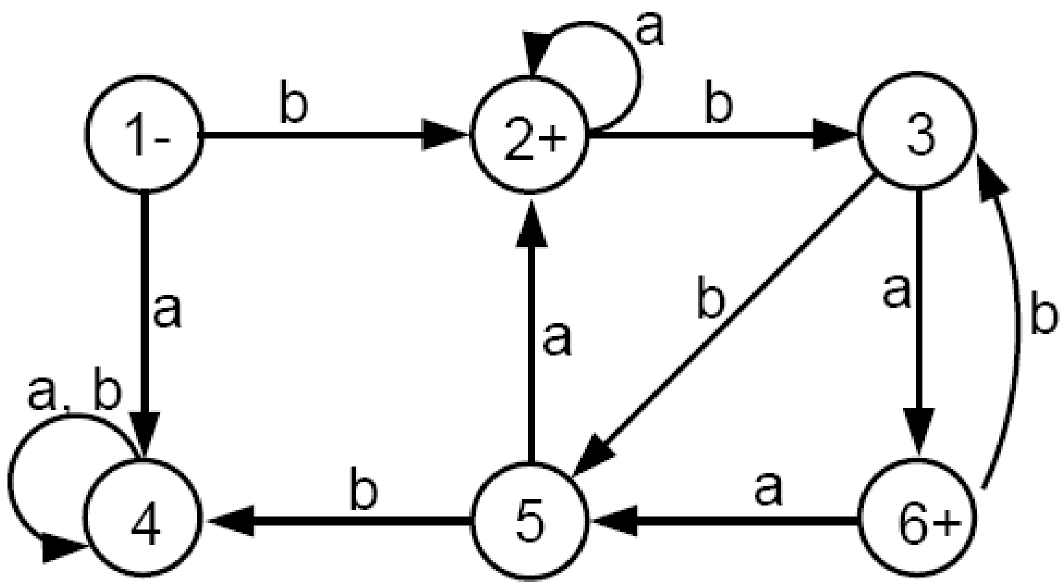
Study of Languages: language vs computational model

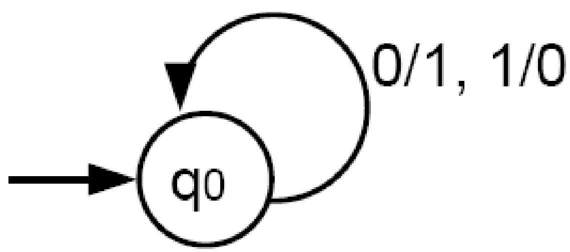
- In independent work in Mathematical Logic, it was noted that
  - There is a natural correspondence between study of models of Computation and the study of natural language Linguistics in an abstract and mathematical sense.
  - **Noam Chomsky** created the subject of mathematical model for description of languages
  - His study grew to the point where it began to shed light on the study of computer languages

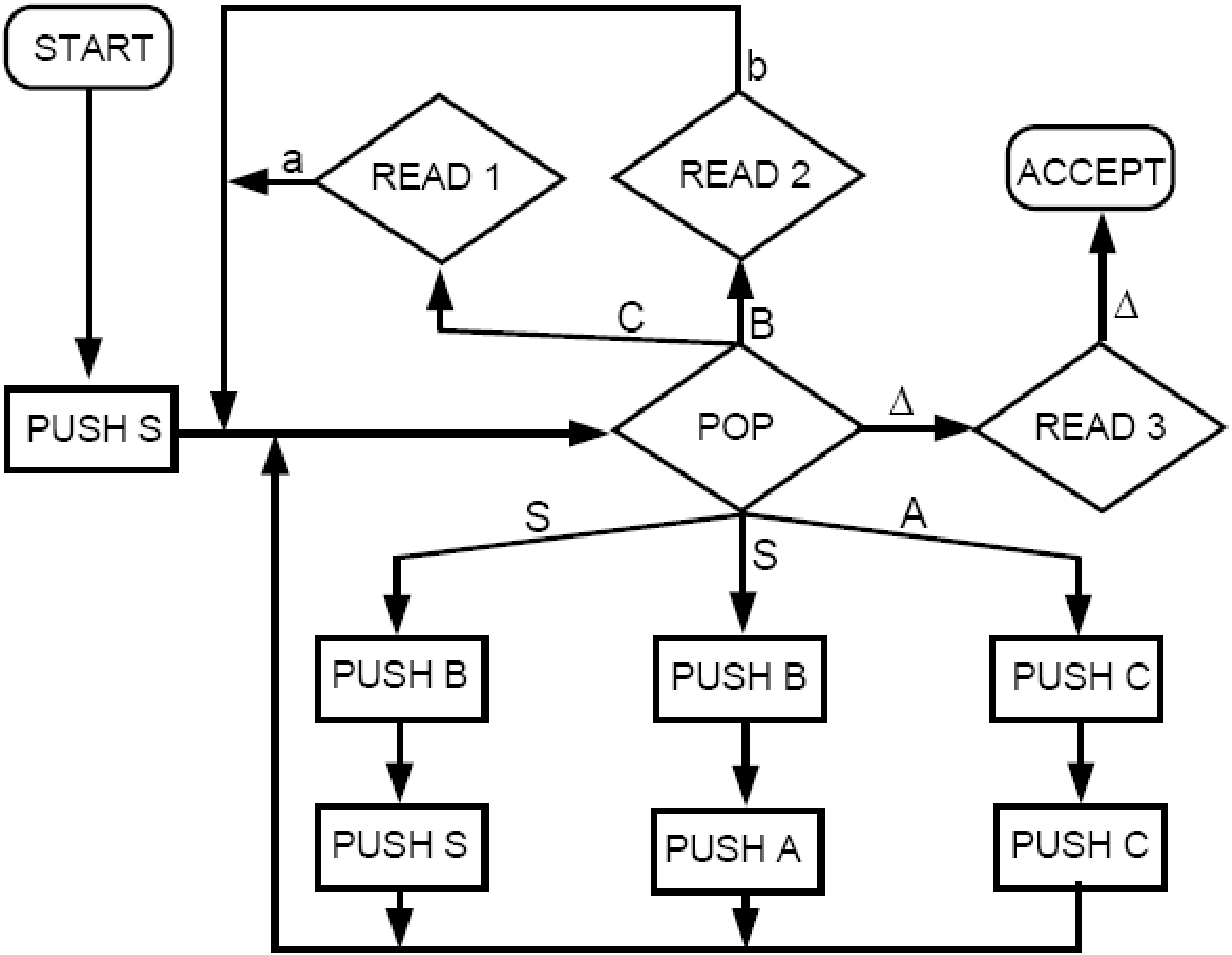
The theory of Formal Languages

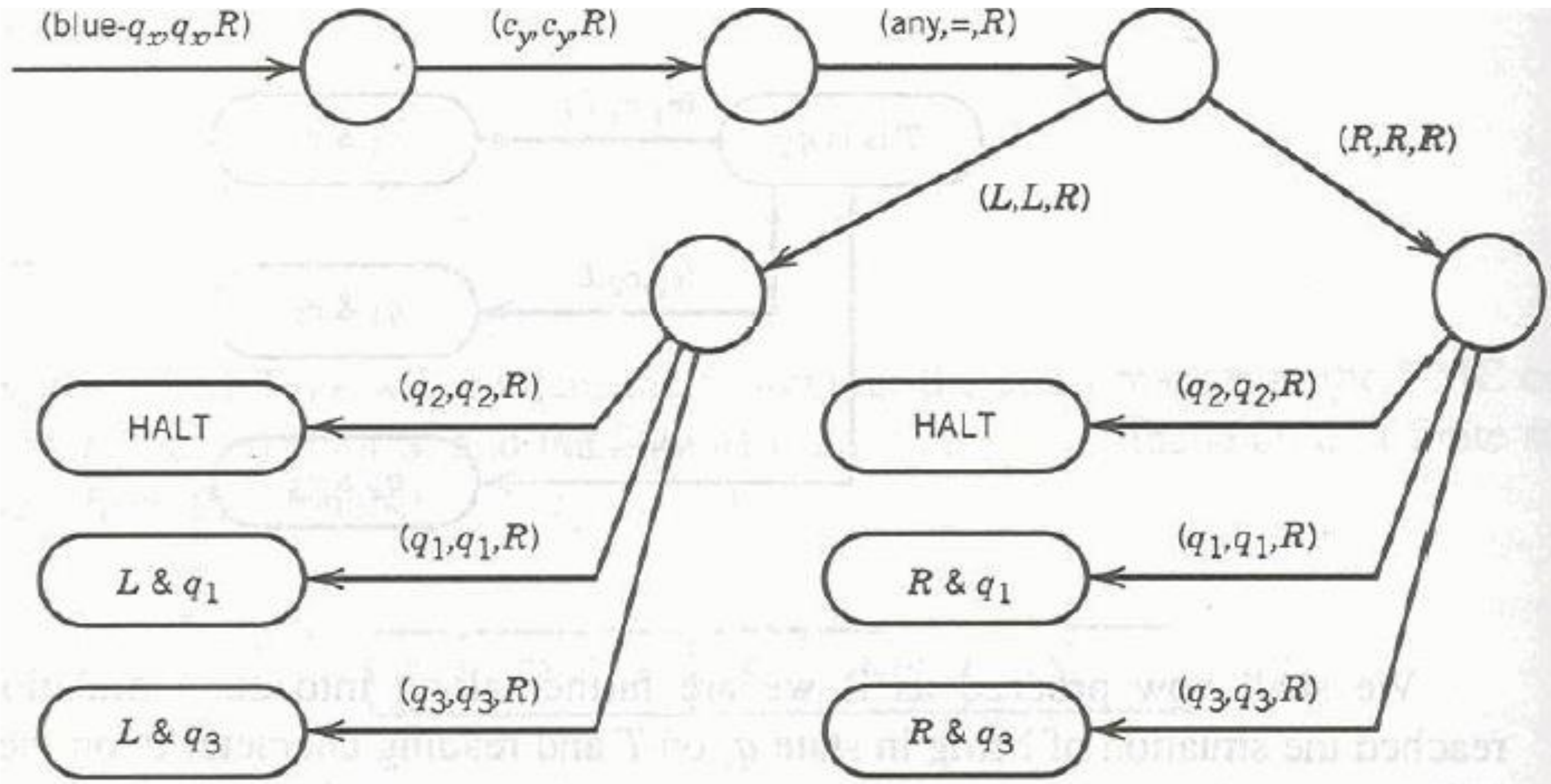
- Regular Languages RE / FSM
- Context free languages CFG / PDA
- Context Sensitive Languages L Bounded TM
- Recursively enumerable Languages TM
- **Alan Turing's** theoretical model for an algorithm machine employing very simple set of mathematical structure held out the possibility that a physical model of the turnings idea could be actually build.
- 1st computer was built and Turing was part of it in Manchester University.

Sample Work






START
PUSH S
a
READ 1
b
READ 2
ACCEPT
C
B
Δ
POP
Δ
READ 3
S
S
A
PUSH B
PUSH B
PUSH C
PUSH S
PUSH A
PUSH C


(blue-q_x,q_x,R)
(c_y,c_y,R)
(any,=,R)
(R,R,R)
(L,L,R)
(q_2,q_2,R)
HALT
(q_1,q_1,R)
L & q_1
(q_3,q_3,R)
L & q_3
(q_2,q_2,R)
HALT
(q_1,q_1,R)
R & q_1
(q_3,q_3,R)
R & q_3

- S → SB | AB
  - A → CC
  - B → b
  - C → a
- $S \rightarrow AB \mid \lambda$
  - $A \rightarrow AB \mid CB \mid a$
  - $B \rightarrow AB \mid b$
  - $C \rightarrow AC \mid c$

- Don’t confuse, we will discuss and explain later

Practical Applications

- FA:
  - Text processing, compilers and hardware design
- CFGs:
  - Programming languages and AI
- Cellular Automata:
  - Biology
- Digital Physics:
  - Whole universe is computed by some sort of a discrete automaton

- General Application:
  - Classical example as a state machine
    - Lift, ATM, bank account (open, active, inactive, closed, blocked), gaming (patrol, search, approach, attack, defend, retreat, die)
    - Mollusks and pines grow by Fibonacci sequence
    - Leopards and snakes can have nearly identical pigmentation patterns, reproducible by two-dimensional automata
    - Cpu/gpu design – sequential logic

- Regular Expressions:
  - Pattern Matching:
    - Email addresses, IP addresses, set of TCP headers, a set of inappropriate words to filter, pattern in genomic data, in text or in a series of bug reports
- CFGs
  - balancing parenthesis
- Turing Machines

No Algorithms

- Vertex coloring
- Graph coloring
- Knapsack problem
- Vertex cover
- Independent set problem
- Travelling sales person problem