CS3063 Theory of Computing

Semester 4 (20 Intake), Feb – Jun 2023

Lecture 11

Turing Machines: Session 2

Sanath Jayasena

Announcements

- Assignment 2: **Due 5th June**

- No interactive session, No Quiz 10 on 5th June
  - (as per student request)
  - Next (and last) session, with Quiz 10: **12th June**

Review of Previous Lecture

Turing Machines - 1

- Turing Machine (TM) Model
- Definitions, Etc
  - Configuration of a TM
  - Acceptance
- Examples
- Computing a (partial) Function

Outline:

Lecture 11

Turing Machines - 2

- Review Exercises
- Combining TM’s
- Variations of TM’s
- Non-deterministic TM’s
- Universal TM’s
- Church-Turing Thesis
- Characteristic Functions

PART 1

Outline:

Lecture 11

Turing Machines - 2

- Review Exercises
- Combining TM’s
- Variations of TM’s
- Non-deterministic TM’s
- Universal TM’s
- Church-Turing Thesis
- Characteristic Functions

Review Exercises

1. Design a TM to accept the language L = {1^m : m is odd} for ∑={1}

2. Design a TM to accept the language L = {0^m1^m : $m > 0$} for ∑={0,1}

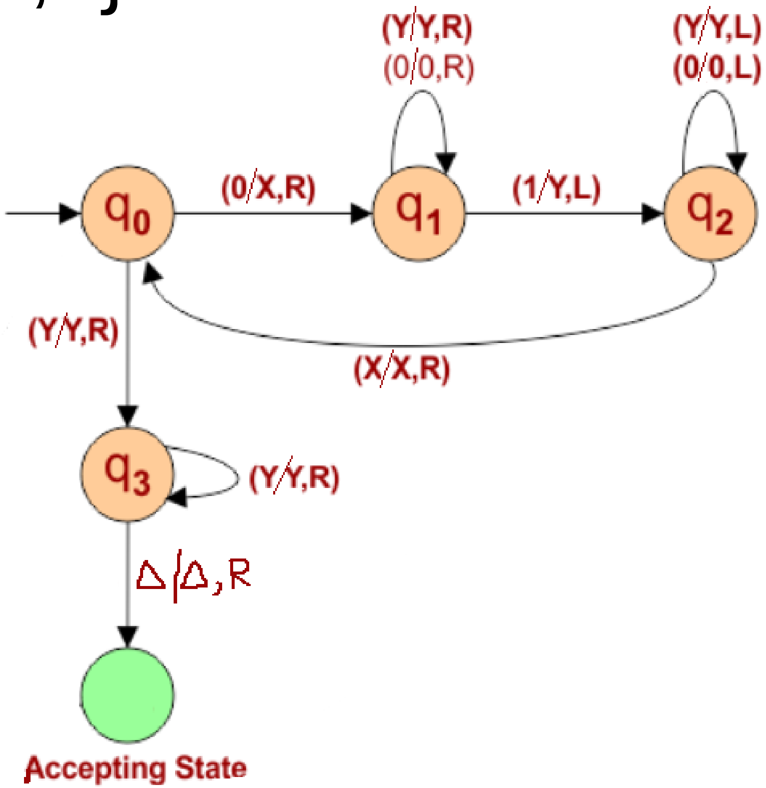
3. TM accepting language, L= {ss | s in $\{a , b\}^*$) for ∑={a,b}? (homework last week)

Review Exercise 1

- Design a TM to accept the language L = {1^m : m is odd} for $\sum$={1}

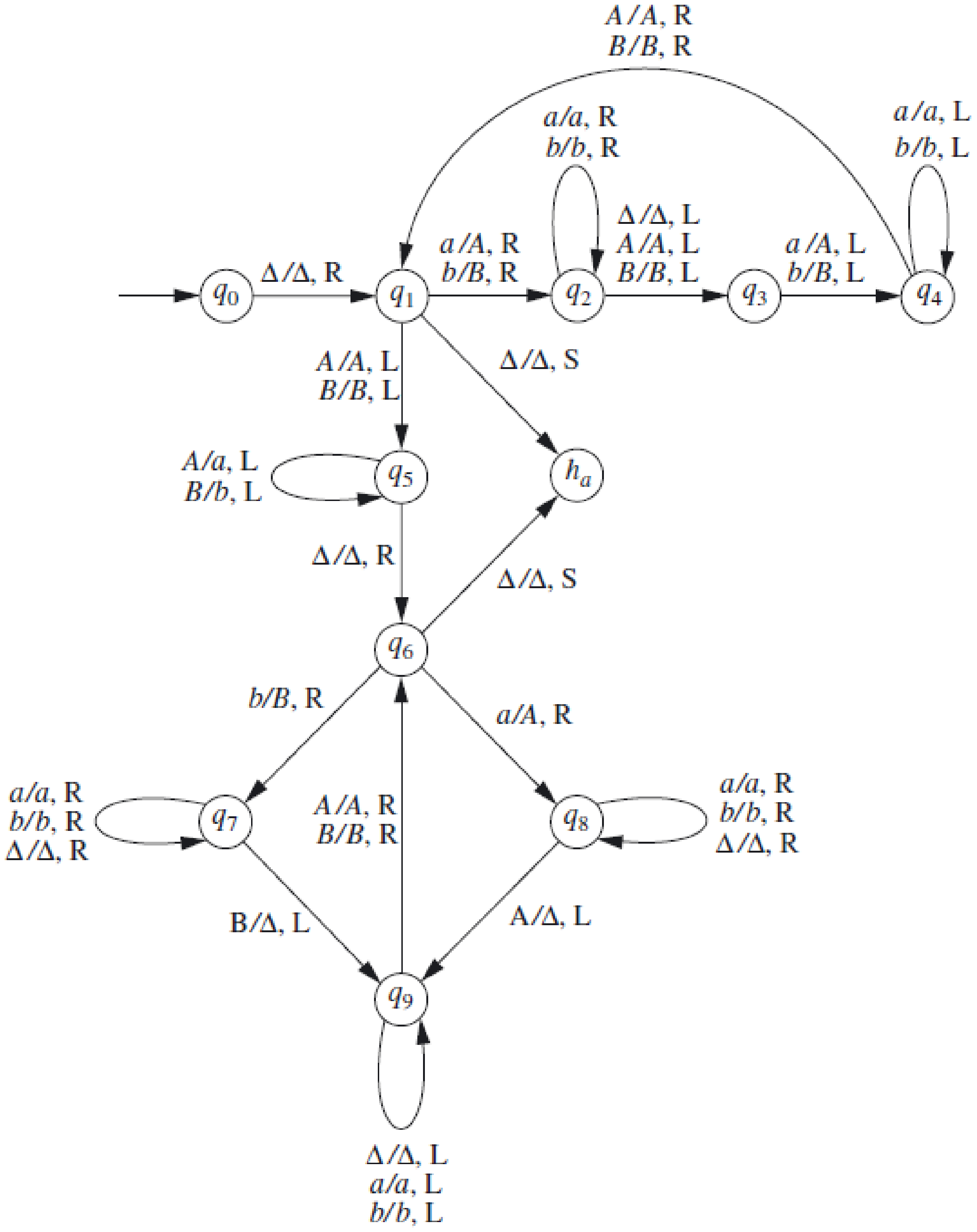
Review Exercise 2

- Design a TM to accept the language L = {0^m1^m : $m > 0$} for $\sum$={0,1}



Review Exercise 3

- TM accepting language, $L = \{ss \mid s \text{ in } \{a , b\}^*)$ for $\sum=\{a,b\}$?



TM accepting language
$L = \{ss \mid s \text{ in } \{a, b\}^*)$

Combining Turing Machines

- Natural way to build a complicated TM
  - Build from simpler, reusable components
- E.g., If T_1 and T_2 are TM's (with disjoint non-halting states and transition functions)
  - T_1T_2 denotes the composite TM in which we execute T_1 first and then T_2
  - T_1T_2 begins in the initial state of T_1
  - For any move that halts in accepting state of T_1, T_1T_2 moves to the initial state of T_2

Combining Turing Machines ...contd

- E.g., ...contd
  - From there, the moves of T_1T_2 are those of T_2
  - If T_1 or T_2 rejects, then T_1T_2 does also
  - T_1T_2 accepts when T_2 accepts
  - We can write $T_1 \rightarrow T_2$

Variations of TM

- Minor variations to the basic model
  - Tape head always moves either to right or left
  - A move can include writing a symbol or moving the tape head, but not both
- Consider another: *multi-tape TM*
  - Easier to describe algorithm implementations
  - Different data items on various tapes
  - But no change in ultimate computing power

Variations of TM ...contd

- What do we mean by computing power of a TM?
  - Can two TM's solve the same problems and get the same answers?
  - (Speed, efficiency, convenience ignored)

- A TM gives an answer by
  - Accepting or rejecting
  - Producing an output (when halts)

Variations of TM ...contd

- Head on each tape can be independent
- Can define an *n*-tape TM formally
  - Transition function and configuration of the TM defined considering all tapes
  - Use 1st tape for input, others for work-space

- Can prove: a 1-tape TM is as powerful as an *n*-tape TM

Non-deterministic TM's

- A non-deterministic TM (or NTM) is defined exactly the same way as a (deterministic) TM, except values of transition function are subsets
- We ignore output; consider acceptance
- Can prove:
  - For a given NTM, T_1, there is a deterministic TM, T_2, with $L(T_1)=L(T_2)$

PART 2

Outline:

Lecture 11

Turing Machines - 2

- Review Exercises
- Combining TM's
- Variations of TM's
- Non-deterministic TM's
- **Universal TM's**
- **Church-Turing Thesis**
- **Characteristic Functions**

Universal Turing Machines

- Previous TMs executed specific algorithms
  - A different TM needed for a different algorithm
  - Or, re-wire the machine
- Turing (in 1936) foresaw the *stored-program computer*
  - Flexibility to execute different algorithms
- Turing describes a Universal TM

Universal TM's ...contd

- A **Universal TM**, $\mathbf{T_u}$, has as input
  - (a) a program
  - (b) a data set

  for it to process
- The program is expressed as a string that specifies another special purpose TM, $\mathbf{T_1}$
- Data set is a string **w**; it is input to $\mathbf{T_1}$
- $\mathbf{T_u}$ simulates the processing of **w** by $\mathbf{T_1}$

Church-Turing Thesis

- "**A TM is a general model of computation**"
  - Means: *any algorithmic procedure that can be carried out (by a human or a computer) can be carried out by a TM*
- First formulated by Alonzo Church (1936)
  - Referred to as Church's thesis also
  - Not a precise statement because "algorithmic procedure" not defined → cannot prove
  - Considered as a conjecture too

Church-Turing Thesis ...contd

- The thesis is generally accepted, because
  - Nature of model indicates all steps crucial to human computation can be carried out
  - Various enhancements do not enhance the computing power
  - Other theoretical models have been shown to be equivalent to a TM
  - No one has proposed an “algorithmic procedure” that cannot be implemented in TM

Characteristic Functions

- Characteristic functions defined for sets
- For any language L in Σ^*, the CF of L is the function from Σ^* to $\{0,1\}$ defined as:

$$\chi_L(x) = \begin{cases} 1 & \text{if } x \in L \\ 0 & \text{otherwise} \end{cases}$$

- Computing the CF can be made similar to accepting the language
- (Assumption here: each string is either accepted or rejected)

Characteristic Functions ...contd

- A TM, T_1, can distinguish between strings in L and not in L (by accepting or rejecting)

- Also, a TM, T_2, may accept every input and distinguish the 2 types by ending up:
  - In configuration $(h_a, \underline{\Delta}1)$ (for strings accepted)
  - In configuration $(h_a, \underline{\Delta}0)$ (for strings rejected)

Characteristic Functions ...contd

- If a TM T_1 exists to accept a language L such that T_1 halts for every string in L → a TM T_2 that computes the CF can be constructed

- But T_1 may still loop forever for some strings not in the language accepted by it
  - In this case, not clear how to obtain corresponding T_2

Accept, Recognize, Decide?

- A TM, T, with input alphabet Σ ***accepts*** a language L in Σ^* if $L(T) = L$
- A TM, T, ***decides*** L if T computes its characteristic function
  - That is: T decides L if T halts in state h_a for every string x in Σ^*, producing output 1 if x is in L and output 0 otherwise

---

- **Recognize**: use with care
  - For some authors recognize ≡ accept while for some others recognize ≡ decide

L11: Conclusion

- Today we discussed
  - – Combining TM’s and Variations of TM’s
  - – Non-deterministic TM’s
  - – Universal TM’s
  - – Church-Turing Thesis
  - – Characteristic Functions
    - Accepting a language
    - Deciding a language