CS 172: Computability and Complexity

Turing Machines & The Church-Turing Hypothesis

Sanjit A. Seshia
EECS, UC Berkeley

Acknowledgments: L.von Ahn, L. Blum, M. Blum

Notes about this Lecture

- This lecture was done on the whiteboard
- We include here a synopsis of the notes written on the board, plus the slides used in class
  - Review this alongside Sections 3.1 and 3.3 of Sipser

Notes - 1

- Informal description of Turing machine (TM)
- What's different from a DFA
  - Input tape is infinite
  - Input head can both READ and WRITE
  - Input head can move LEFT and RIGHT
  - Has both ACCEPT and REJECT states ***and accepts/rejects rightaway*** (does not need to reach end of input)

Notes - 2

- Let $L = \{ w\#w \mid w \in \{0,1\}^* \}$
- Give a high-level description of a TM that accepts an input if it is in L, and rejects if not.
- See Sipser 3.1 for details

Notes - 3

- Formal definition of Turing Machine: a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$
- Main points:
  - $\Sigma \subset \Gamma$
  - Blank symbol is in Γ, but not in Σ
  - $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$
  - Special case: when rd/wr head is at left end of the input tape

Notes - 4

- Configuration

  u q v -- TM is in state q with uv on the tape and the head on the first symbol of v

  C_i yields C_j if δ takes the TM from config C_i to C_j

  Start, accepting, rejecting configurations

- Acceptance: TM accepts w if $\exists$ sequence of configurations $C_1, C_2, \ldots, C_k$ where
  1. C_1 is the start configuration
  2. C_i yields C_{i+1} $(1 \leq I \leq k-1)$
  3. C_k is an accepting configuration

A TM recognizes a language if it *accepts* all and only those strings in the language

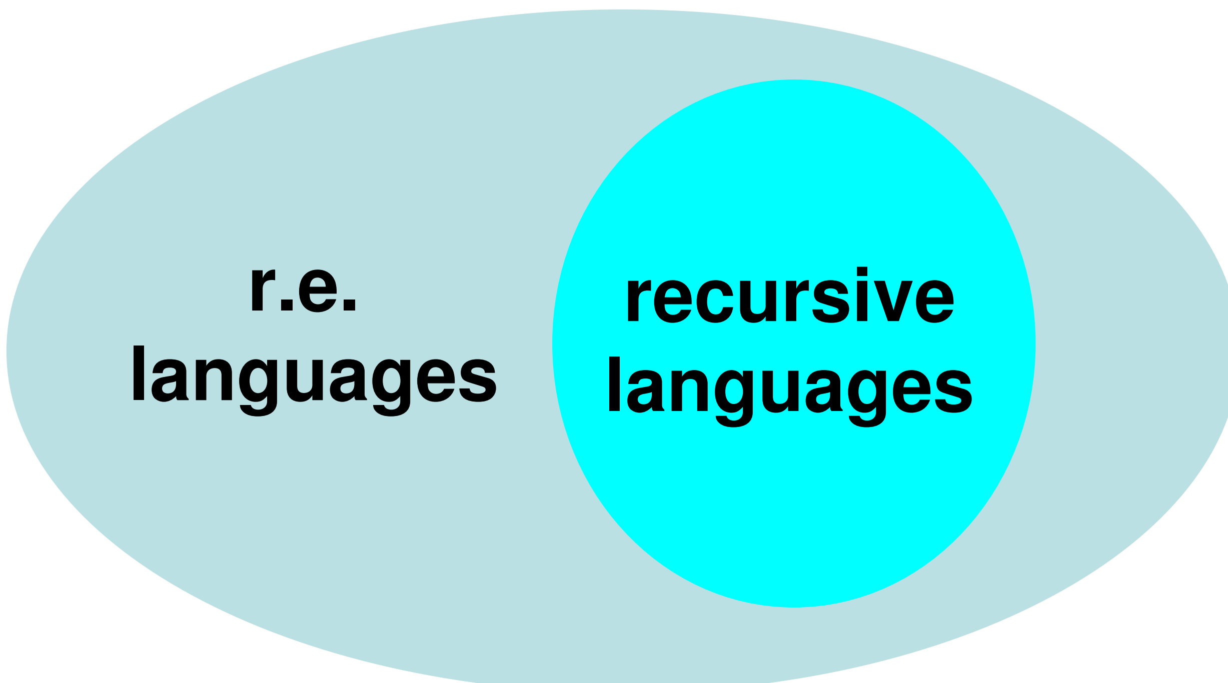
A language is called Turing-recognizable or *recursively enumerable* if some TM recognizes it

A TM decides a language if it *accepts* all strings in the language and *rejects* all strings not in the language

A language is called decidable or *recursive* if some TM decides it

A language is called **Turing-recognizable or recursively enumerable (r.e.)** if some TM recognizes it

A language is called **decidable or recursive** if some TM decides it

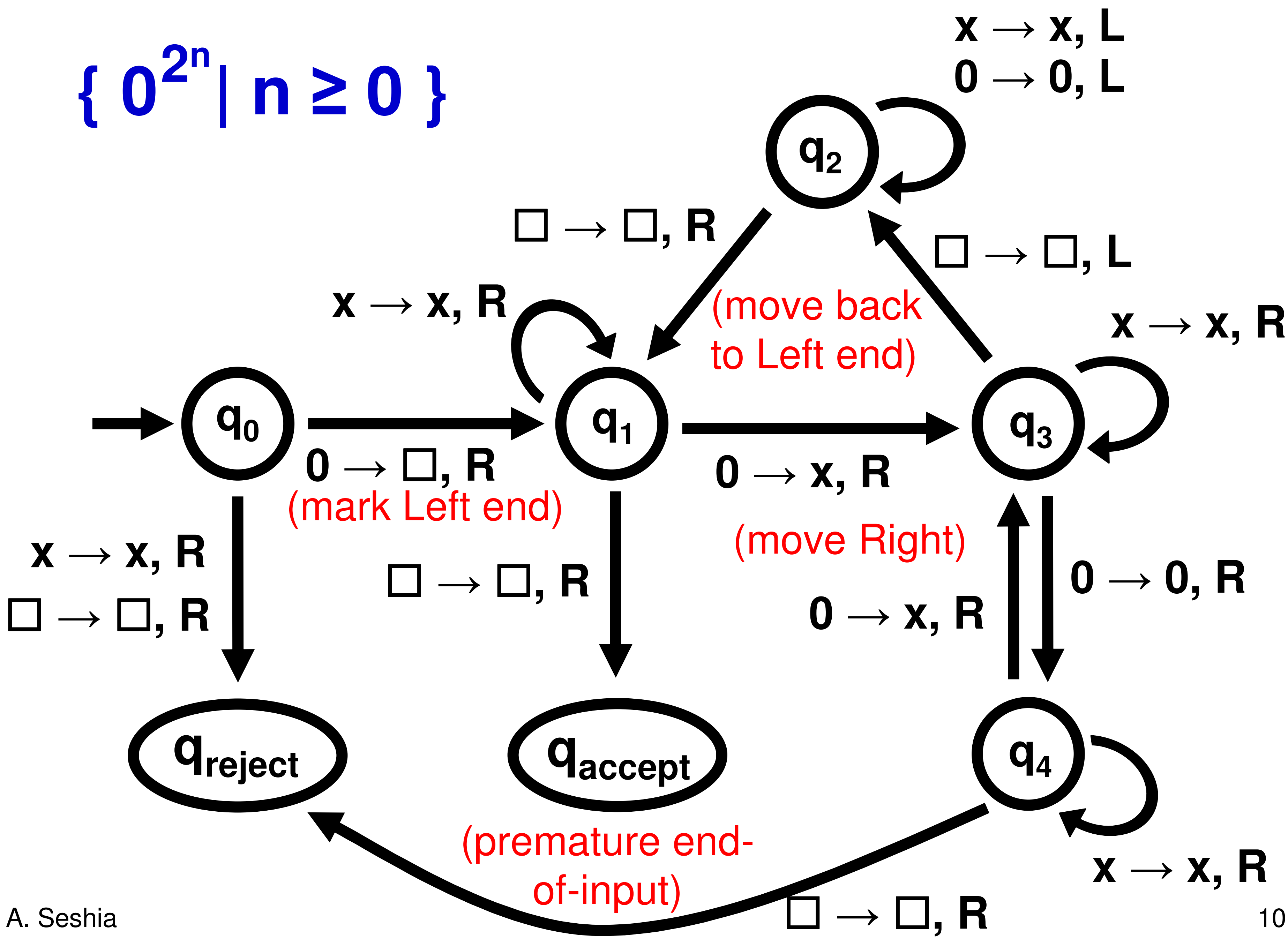


Design a TM that decides $L = \{ 0^{2^n} \mid n \geq 0 \}$

Note: Elements of L encode powers of 2 in UNARY!

• Think about how you would decide whether a given number is a power of 2 (in decimal)
• Then translate that procedure over to work in UNARY
• Main idea: "repeated division by 2"

$\{ 0^{2^n} \mid n \geq 0 \}$



Do this at Home

$\mathbf{L = \{ w\#w \mid w \in \{0, 1\}^* \}}$

Construct the TM that decides L

(check with Sipser Example 3.9)

Church-Turing Hypothesis

- Intuitive notion of algorithms

= Turing machine algorithms

- "Any process which could be naturally called *an effective procedure* can be realized by a Turing machine"

Hilbert and his 10th Problem

- In 1900: Posed 23 "challenge problems" in Mathematics
- The 10th problem:
  Devise an algorithm to decide if a given polynomial has an integral root.
- We now know: This is undecidable!
  - Needed a definition of "algorithm", which was given by Church and Turing (independently)

For Next Class

- Read Sipser 3.1, 3.2, 3.3
  - Practice writing down *high-level descriptions* of Turing machines, as he does