CS310

Turing Machines

Section 3.1

November 5, 2010

Alan Turing

- English mathematician
- Helped break German codes during WWII
- Helped formalize the concept of an algorithm
  - Represented by the Turing Machine
  - A TM is a precise way to discuss/reason about algorithms
  - Data: any data can be encoded as a string of 0s and 1s

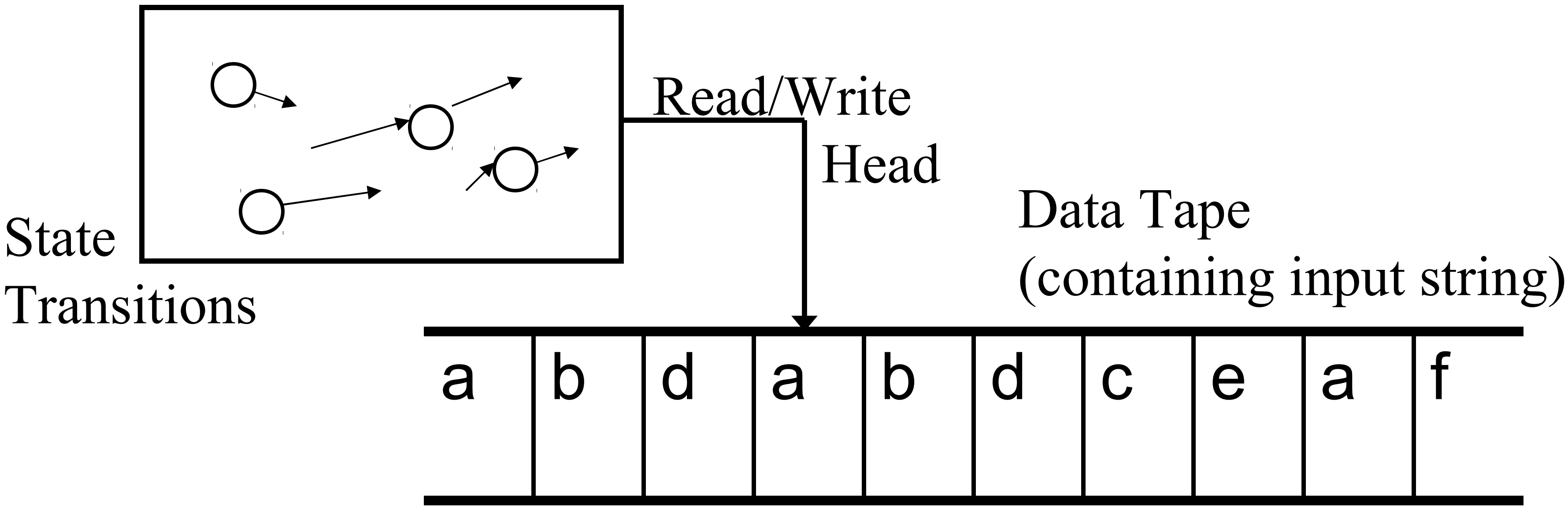
Turing Machines

- Similar to Finite Automata
  - unlimited and unrestricted memory
    - random access
  - more accurate model of modern computer
- Problems that cannot be solved by a Turing Machine cannot be solved by a "real" digital computer
  - theoretical limits of computation

What are the fundamental capabilities and limitations of computers?
Computer Science is really the science of computation, not of computers.

Turing Machine

- State Transitions plus infinite “data tape”
  - read/write tape
  - move around on tape



Notes

- Deterministic
- May make multiple passes over input

- Reject string by entering reject configuration or looping forever
  - hard to tell if a machine will loop forever
  - Halting problem

Differences with FA

- TM can read and write from tape
  - FA can only read

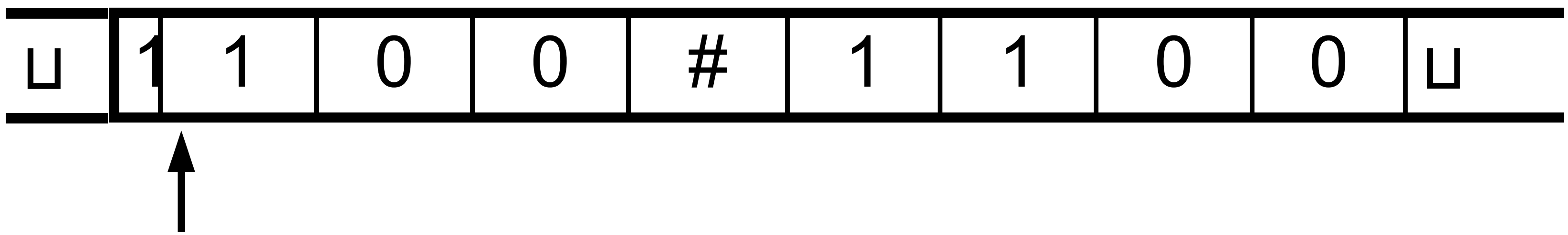
- Read/Write head can move left or right
  - FA can only move one direction, one step
  - must move*

- TM tape is infinite
- TM accept and reject states take effect *immediately*

Simple Machine
$\{w\#w \mid w \in \{1,0\}^* \}$

Example

- L = { w#w | w $\in$ { 0,1} * }
- Conceptually, we want to do what?
- input string:



Formal Definition (7 Tuple)

- $\{Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject}\}$
- Q: set of states

- Σ: input alphabet, not containing the blank character: ⊔

  Γ: tape alphabet, $\sqcup \in \Gamma$ and $\Sigma \subseteq \Gamma$

  δ: $Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$: transition function

  $q_0 \in Q$: start state

  $q_{accept} \in Q$: accept state

  $q_{reject} \in Q$: reject state, $q_{accept} \neq q_{reject}$

Operation

- Start configuration of M on input w is: q_0w
- Accepting configuration: q_{accept}

- Rejecting configuration: q_{reject}

- Yield: uaq_ibv yields uq_nacv

  if $\delta(q_i, b) \rightarrow (q_n, c, L)$
- Accepting and Rejecting configurations are called *halting* configurations
  - the TM stops operating
  - otherwise, loops forever

Definition of Computing

- A TM, M, accepts a string, w, if there exists a sequence of configurations, $c_0, c_1, \ldots, c_n$, such that:
  - c_0 is the start configuration
  - c_i yields c_{i+1} for all i
  - c_n is an accept configuration
- The set of strings M accepts is L(M)
  - language of M

Definitions

- Turing recognizable
  - a language is Turing Recognizable if some TM recognizes it (accepts all valid strings)

- Turing decidable
  - a language is Turing decidable if some TM decides it
  - halts on all inputs
    - hard to tell if a looping machine is really going to reject the string

Church-Turing Thesis

- Turing Model is and always will be the most powerful model
  - it can simulate other models: D/NFA, PDA
  - variations do not provide more power
    - extra tape
    - nondeterminism
    - extra read/write heads
    - (but may make it easier to build)

Build a Machine!

Often, you write an algorithm for the machine rather than a set of transitions.

- $L = \{ \Sigma\Sigma0\Sigma \}$ $\Sigma = \{0, 1\}$
- $L = \{a^n b^n \mid n >= 0\}$
- $L = \{a^n b^n c^n \mid n >= 0\}$
- $L = \{a^n b^m c^p \mid n, m, p > 0, p = n - m\}$
- $L = \{ w \mid |w| \text{ is even} \}$, $\Sigma = \{1\}$
- $L = \{ w \mid |w| \text{ a power of } 2 \}$, $\Sigma = \{1\}$
- $L = \{ w \mid |w| \text{ is prime} \}$, $\Sigma = \{1\}$

Transducer

- TM produces output (to the tape)
- A function F with domain D is Turing-

  Computable if there exists a TM, M, such that the configuration q_0w yields q_{accept}, $F(w)$ for all $w \in D$.
- x = number in base 1, F(x) = 2x

  x = 111 2x = 111111

Transducer

- x, y positive integers in base 1
- design TM that computes x+y