CSCI-2200
FOUNDATIONS OF COMPUTER SCIENCE

Spring 2015

April 30, 2015
Announcements

• Homework 8 is due now.
• Homework 9 is due next Thursday at 10am (in class).
• If you need to make alternate arrangements for the final exam (and have an excused absence), please let me know asap.
Regular Languages

Context-Free Languages

\{a^n b^n\} \quad \{ww^R\}

Recognized by FAs

Recognized by Pushdown Automata
Review: Context-Free Languages

- A context free language is a language that can be recognized by a new type of computational model – a pushdown automata
REVIEW: CFGs

• A context-free language can be described by a context-free grammar (CFG).

• Design a CFG that generates the language of strings that are palindromes over the alphabet $\Sigma = \{0,1\}$.
  • Assume the empty string is a palindrome.
Fact: The language \( \{a^n b^n c^n : n \geq 0\} \) is not context-free

(This can be can proven using a pumping lemma for context-free languages)
Languages accepted by Turing Machines

- \( a^n b^n c^n \)
- \( a^n b^n \)
- \( a^* b^* \)
- \( ww \)
- \( ww^R \)
TURING MACHINES
Finite Automaton (FA)

- Read input string from tape, one character at a time
- Change state based on input
- When input ends, output “accept” or “reject”
Pushdown Automata

- Read input string from tape, one character at a time
- Can also push and pop from stack
- Change state based on input and (optionally) top of stack
- When input ends, output “accept” or “reject”
A Turing Machine has the “highest” computational power

Turing Machine

Input

String

Automaton

Output

“Accept” or “Reject”

Random Access Memory
Alan Turing (1912 – 1954)
Alan Turing

- Widely recognized as the father of theoretical computer science
- Born in London in 1912.
- Studied quantum mechanics, probability, logic at Cambridge (undergrad).
- Published landmark paper, 1936
  - Proposed most widely accepted formal model for algorithmic computation – the Turing Machine
  - Proved the existence of computationally unsolvable problems
- Did PhD in logic, algebra, number theory at Princeton, 1936–38.
- Worked on UK cryptography program, 1939-1945
  - Built the bombe to help break the Enigma – shortened the length of the war.
Alan Turing (cont.)

• Contributed to design and construction of UK’s first digital computers, 1946 - 1949
• Proposed Turing Test for artificial intelligence in 1950
• In 1952, Turing was arrested and convicted under a British law that prohibited "acts of gross indecency between men, in public or private."
  • Sentence was chemical castration
• Turing committed suicide in 1954 by eating a cyanide laced apple.
• Turing received a royal pardon on December 24, 2013.

• Interesting podcast: 
  http://www.radiolab.org/story/193037-turing-problem/
Turing’s Thesis

• (Church-Turing Thesis 1936) Any computation carried out by mechanical means can be performed by a Turing Machine

• There is no known “realistic” model of computation more powerful than Turing Machines

• When we say “there exists an algorithm”, we mean “there exists a Turing Machine”.
Turing Machine

Input

String

Automaton

Random Access Memory

Output

“Accept” or “Reject”
A Turing Machine

Tape

......

......

......

Read-Write head

Control Unit

- The Read-Write head may move both left and right over the input tape.
- A TM may both read from and write to its input tape.
- A TM halts immediately when it reaches an accept or reject state.
The Tape

Infinite length

Input string

Blank symbol

head
Read-Write head

The head at each time step:

1. Reads a symbol
2. Writes a symbol
3. Moves Left or Right
   - Cannot move Left past start of input. If it tries, it stays in place
States & Transitions

Read

Write

Move Left

Move Right

$q_1 \xrightarrow{a \rightarrow b, L} q_2$

$q_1 \xrightarrow{a \rightarrow b, R} q_2$
Determinism

Turing Machines are (for now) deterministic

**Allowed**

\[ a \rightarrow b, R \]

\[ b \rightarrow d, L \]

**Not Allowed**

\[ a \rightarrow b, R \]
Accepting and Rejecting Inputs

- A TM has an accept state $q_{accept}$ and a reject state $q_{reject}$
  - Note: $q_{accept} \neq q_{reject}$
  - These states have no outgoing transitions.

- The computation continues until the machine enters the accept or reject state
  - If machine enters one of these states, it outputs “accept” or “reject” immediately.
  - Doesn’t read any more input.

- The machine may also loop – meaning it does not halt.
  - The computation goes on forever.
  - In this case, the input is not accepted.
Languages Accepted by Turing Machine

• The set of strings accepted (recognized) by a Turing machine is the language accepted by the Turing machine.
• A language is called Turing-recognizable if some Turing Machine recognizes it.

• A Turing machine T is said to decide a language L if and only if it outputs “accept” and halts if the string is in L and outputs “reject” and halts if the string is not in L.
  • It must not enter an infinite loop.
• A language is called Turing-decidable if some Turing Machine decides it.
TM Example 1

- Create a TM that recognizes the language $aa^*$ over the alphabet $\Sigma=\{a\}$
TM Example 2

Create a TM that recognizes the language $aa^*$ over the alphabet $\Sigma=\{a,b\}$. 
Create a TM that recognizes the language $aa^*$ over the alphabet $\Sigma = \{a, b\}$ but does not decide the language.
TM Example 3

• Create a Turing Machine over the alphabet \( \{a,b\} \) that recognizes the language \( L = \{w \mid w \text{ begins with } b \text{ or } w = aa^* \} \)
Formal Definition of a TM

- A Turing Machine is defined by a 7-tuple $M=(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$
  - $Q$: finite set of states
  - $\Sigma$: finite input alphabet (does not contain blank symbol)
  - $\Gamma$: finite tape alphabet, contains blank symbol and elements of $\Sigma$
  - Transition function $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$
    - * partial function
  - $q_0$ is the start state
  - $q_{accept}$ is the accept state
  - $q_{reject}$ is the reject state, where $q_{reject} \neq q_{accept}$
High Level Description of a TM

- We can describe a TM *informally* by describing what the head does in response to certain input and state.

- Example: Describe (informally) a TM that recognizes the language \{a^n b^n \mid n \geq 1\} (where the alphabet is \{a,b\})

On input string w:

1. Check if string is of the form \(a^i b^k\) (can be done with a DFA)
   
   If not, reject.

2. Zig zag across tape, checking if every a has a corresponding b
   
   Cross off symbols (write X) to keep track of which symbols correspond.

3. If all a’s are crossed off and there are remaining b’s, then reject
   
   If for some a, no corresponding b is found, then reject.

   Otherwise, accept.
High Level Description of a TM

Example: Describe (informally) a TM that recognizes the language \( \{a^n b^n \mid n \geq 1\} \) (where the alphabet is \( \{a,b\}\))

• What is the running time?
  • Running time counts number of state transitions.
  • Use big-O notation.
Another High Level Description of a TM

• Give a TM decides the language $B = \{w#w \mid w \in \{0,1\}^*\}$

• $M = \text{On input string } w:$
  1. Zig-zag across the tape, corresponding to positions on either side of the # symbol to check whether these positions contain the same symbol.
     • If they do not, or if no # is found, “reject”
     • Cross off symbols as they are checked to keep track of which symbols correspond
  2. When all symbols to left of # have been crossed off, check for any remaining symbols to right of #.
     • If any remain, “reject”
     • Else “accept”