Turing Machines

Homework

- Homework
  - From textbook:
    - Exercise 8.2.1 b,c
    - Exercise 8.2.2 a,b (use JFLAP)
    - Exercise 8.2.5 a,b (hint: try running on JFLAP)
    - Exercise 8.4.1 a,b (implement using JFLAP)
  - Additional problems:
    - Draw a TM that computes the following function. Assume that the TM represents n as 0^n (use JFLAP)
      - $F(x) = 2^x$

Just a reminder

- Next Tuesday
  - Exam 2

Before We Start

- Any questions?

Languages

- The $64,000 Question
  - What is a language?
  - What is a class of languages?

Now our picture looks like

We’re going to start to look at languages out here
**The Turing Machine**

- We investigate the next classes of languages by first considering the machine
  - Turing Machine
    - Developed by Alan Turing in 1936
    - More than just recognizing languages
    - Foundation for modern theory of computation

---

**More about Turing**

- “Breaking the Code”
  - Movie about the personal life of Alan Turing
    - Death was by cyanide poisoning (some say suicide)
  - Turing worked as a code breaker for the Allies during WWII.
  - Turing eventually tried to build his machine and apply it to mathematics, code breaking, and games (chess).
    - Was beat to the punch by vonNeumann

---

**Theory Hall of Fame**

- Alan Turing
  - 1912 – 1954
  - PhD – Princeton (1938)
  - Research
    - Cambridge and Manchester U.
    - National Physical Lab, UK
  - Creator of the Turing Test

---

**The Turing Machine**

- Some history
  - Created in response to Kurt Godel’s 1931 proof that formal mathematics was incomplete
    - There exists logical statements that cannot be proven by using formal deduction from a set of rules
      - Good Reading: “Godel, Escher, Bach” by Hofstadter
    - Turing set out to define a process by which it can be decided whether a given mathematical can be proven or not.

---

**Theory Hall of Fame**

- Kurt Godel
  - 1906 -- 1978
  - b. Brünn, Austria-Hungary
  - PhD – University of Vienna (1929)
  - Research
    - Princeton University
  - Godel’s Incompleteness Theorem

---

**The Turing Machine**

- Motivating idea
  - Build a theoretical a “human computer”
  - Likened to a human with a paper and pencil that can solve problems in an algorithmic way
  - The theoretical machine provides a means to determine:
    - If an algorithm or procedure exists for a given problem
    - What that algorithm or procedure looks like
    - How long would it take to run this algorithm or procedure.
The Church-Turing Thesis (1936)

- Any algorithmic procedure that can be carried out by a human or group of humans can be carried out by some Turing Machine"  
  - Equating algorithm with running on a TM  
  - Turing Machine is still a valid computational model for most modern computers.

Theory Hall of Fame

- **Alonso Church**  
  - 1903 – 1995  
  - b. Washington D.C.  
  - PhD – Princeton (1927)  
  - Mathematics Prof (1927 – 1967)  
  - Advisor to both Turing and Kleene

Turing Machine

- A Machine consists of:
  - A state machine  
  - An input tape  
  - A movable r/w tape head  
- A move of a Turing Machine  
  - Read the character on the tape at the current position of the tape head  
  - Change the character on the tape at the current position of the tape head  
  - Move the tape head  
  - Change the state of the machine based on current state and character read

Turing Machines

- Let’s formalize this  
  - A Turing Machine M is a 7-tuple:  
    - M = (Q, Σ, Γ, δ, q₀, B, F) where  
      - Q = a finite set of states  
      - Σ = input alphabet (strings to be used as input)  
      - Γ = tape alphabet (chars that can be written onto the tape. Includes symbols from Σ)  
      - q₀ = start state  
      - B = the blank symbol (B ∈ Γ, B ∉ Σ)  
      - F = set of final states  
      - δ = transition function
- Transition function:  
  - δ: Q x Γ → Q x Γ x {R, L}

  - Input:  
    - Current state  
    - Tape symbol read at current position of tape head  
  - Output:  
    - State in which to move the machine  
    - Tape symbol to write at current position of tape head  
    - Direction in which to move the tape head (R = right, L = left)
Turing Machines

- Transition Function
  - Symbol at current tape head position
  - Symbol to write at the current head position
  - Direction in which to move the tape head

Turing Machine

- Configuration of a TM
  - Gives the current "configuration" of a TM

Turing Machine

- Initial configuration:
  - To run an input string x on a TM,
    - Start in the starting state
    - Place the string on the tape
    - Place the head at the start of this string:

Turing Machine

- Accepting a string
  - A string x is accepted by a TM, if
    - Starting in the initial configuration
    - With x on the input tape
    - The machine eventually ends up in an accepting state.
  - I.e.
    - \( q_0x \rightarrow^* \alpha \rho \beta \) and \( \rho \in F \)

Turing Machine

- TMs and halting
  - We say that a TM halts if
    - The machine has nowhere to go (at a state, reading a symbol where no transition is defined)
  - Without loss of generality, we can assume that a TM will always halt when in an accepting state.
  - Note that the TM can halt in a non-accepting state!
Turing Machine

- Running a Turing Machine
  - The execution of a TM can result in 3 possible cases:
    - The machine “halts” in an accepting state (ACCEPT)
    - The machine “halts” in a non-accepting state (REJECT)
    - The machine goes into an “infinite loop” (REJECT but keeps us guessing!)

TMs and Regular Languages

- Example
  - L = \{ x \in \{ a, b \}^* | x \text{ contains the substring } aba \}\}

TMs and Regular Languages

- Example
  - L = \{ x \in \{ a, b \}^* | x \text{ contains the substring } aba \}\}

  - Build a TM that mimics the FA that accepts this language

TMs and Regular Languages

- Do you think that JFLAP can handle TMs?
  - You bet!
Theory Hall of Fame

• **Susan Rodgers**
  – PhD – Purdue (1985)
  – CS Prof
  • RPI (1989-1994)
  • Duke (1995 – present)
  – Creator and keeper of JFLAP

TMs and Regular Languages

• **Example**
  – Observations
    • Like FAs TM tape head will always move to the right
    • Like FAs, TM will not write new chars onto the tape
    • Can enter final state even before the machine reads all the characters of x.

TMs and CFLs

• **Example**
  – L = \{ x \in \{0,1\}^* \mid x = 0^i1^i \}
  – Basic idea:
    • Find leftmost 0 and change it to an X
    • Find the rightmost 1 and change to a Y
    • Continue to match until all 0s have been changed to Xs
      – If every X has a matching Y, accept
      – If can’t find a matching 1 reject
      – If is leftover, reject

TMs and Context Free Language

– L = \{ x \in \{0,1\}^* \mid x = 0^i1^i \}
– States:
  • q_0 – at the leftmost blank
  • q_1 – read leftmost 0, looking for rightmost 1
  • q_2 – found and matched rightmost 1 / go back to leftmost 0
  • q_3 – no more 0’s, all 0s matched with 1s / see if we get to end of string without reading a 1
  • q_4 – accept and halt

TMs and Context Free Language

– JFLAP…go to work!!
TMs and Context Free Language

• Another?

• Let’s try our old friend palindrome
  – \( L = \{ x \in \{a, b\}^* \mid x = x^r \} \)

TMs and Context Free Language

• Example
  – \( L = \{ x \in \{a, b\}^* \mid x = x^r \} \)
  – Basic idea:
    • Compare the first character with the last character.
    • If they match compare the second character with the second to last character
    • If they match, compare the 3rd character with the 3rd to last character
    • And so on...
    • For \( x \) in \( \text{pal} \), eventually we will end up with 0 or 1 unmatched characters.

TMs and Context Free Language

• Example
  – \( L = \{ x \in \{a, b\}^* \mid x = x^r \} \)
  – How to compare characters?
    • Read a character and replace it with blanks.
    • Move across the tape to first blank character
    • Check the character to the left
      – If it’s the character that you initially read in, replace it with a blank, move the tape head left until you reach the first blank character and so on.

TMs and Context Free Language

• Example
  – \( L = \{ x \in \{a, b\}^* \mid x = x^r \} \)
  – Halting condition:
    • If when you moved left/right after finding the first blank, the character found is a blank, we have found a palindrome!

TMs and Context Free Language

• Example
  – \( L = \{ x \in \{a, b\}^* \mid x = x^r \} \)
  – States:
    • \( q_1 \) – at the leftmost blank
    • \( q_2 \) – read an a, move right until you find a blank
    • \( q_3 \) – looking for an a, look left after finding rightmost blank
    • \( q_4 \) – matched first character read, move left till you find the leftmost blank
    • \( q_5 \) – read an b, move right until you find a blank
    • \( q_6 \) – looking for an b, look left after finding rightmost blank
TMs and Context Free Language

• Example
  – \( L = \{ x \in \{a, b\}^* | x = x^c \} \)

  – Let’s go to the video tape

Let’s try a non-context free language

• Example
  – \( L = \{ xx | x \in \{a, b\}^* \} \)
  
  – Basic idea
    • Find and mark the middle of the string
    • Compare characters starting from the start of the string with characters starting from the middle of the string.

Let’s try a non-context free language

• Example
  – \( L = \{ xx | x \in \{a, b\}^* \} \)

  – Finding the middle of the string
    • Convert first character to it’s upper case equiv.
    • Move all the way to the right to the last lower case character and change it to upper case.
    • Move all the way back left to the first lower case character and change it to upper case
    • And so on.

Let’s try a non-context free language

• Once you’ve found the middle,
  – Convert the 1st half of the string back to lower case.
  – Start from the left of the tape
    • Match upper case chars in 1st half with lower case chars in the 2nd,
    • Replace a matched upper case char with blanks
    • Repeat until the 1st half of the string is all blank.

Let’s try a non-context free language

• Let’s go to the video tape
Turing Machines

- Questions?

- Let's break.