CSE 322: Midterm Review

Basic Concepts (Chapter 0)

Sets

Notation and Definitions
- \( A = \{ x \mid \text{rule about } x \} \), \( x \in A \), \( A \subseteq B \), \( A = B \)
- \( \exists \) (“there exists”), \( \forall \) (“for all”)

Finite and Infinite Sets
- Set of natural numbers \( \mathbb{N} \), integers \( \mathbb{Z} \), reals \( \mathbb{R} \) etc.
- Empty set \( \emptyset \)

Set operations: Know the definitions for proofs
- Union: \( A \cup B = \{ x \mid x \in A \text{ or } x \in B \} \)
- Intersection: \( A \cap B = \{ x \mid x \in A \text{ and } x \in B \} \)
- Complement: \( \overline{A} = \{ x \mid x \notin A \} \)

Set operations (cont.)
- Power set of \( A = \text{Pow}(A) \) or \( 2^A = \text{set of all subsets of } A \)
  - E.g. \( A = \{0,1\} \rightarrow 2^A = \{ \emptyset, \{0\}, \{1\}, \{0,1\} \} \)
- Cartesian Product: \( A \times B = \{ (a,b) \mid a \in A \text{ and } b \in B \} \)

Functions:
- \( f: \text{Domain} \rightarrow \text{Range} \)
  - \( \text{Add}(x,y) = x + y \rightarrow \text{Add: } \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z} \)
  - Definitions of 1-1 and onto (bijection if both)
Strings

- Alphabet $\Sigma = \text{finite set of symbols, e.g. } \Sigma = \{0, 1\}$
- String $w = \text{finite sequence of symbols } \in \Sigma$
  - $w = w_1w_2\ldots w_n$
- String properties: Know the definitions
  - Length of $w = |w|$ (if $w = w_1w_2\ldots w_n$)
  - Empty string = $\varepsilon$ (length of $\varepsilon = 0$)
  - Substring of $w$
  - Reverse of $w = w^R = w_n\ldots w_1$
  - Concatenation of strings $x$ and $y$ (append $y$ to $x$)
  - $y^k = \text{concatenate } y \text{ to itself to get string of } k \text{ } y's$
  - Lexicographical order = order based on length and dictionary order within equal length

Languages and Proof Techniques

- Language $L = \text{set of strings over an alphabet } (\text{i.e. } L \subseteq \Sigma^*)$
  - E.g. $L = \{0^n1^n | n \geq 0\}$ over $\Sigma = \{0, 1\}$
  - E.g. $L = \{p \mid p \text{ is a syntactically correct C++ program}\}$ over $\Sigma = \text{ASCII characters}$
- Proof Techniques: Look at lecture slides, handouts, and notes
  1. Proof by counterexample
  2. Proof by contradiction
  3. Proof of set equalities ($A = B$)
  4. Proof of “iff” ($X \iff Y$) statements (prove both $X \Rightarrow Y$ and $X \Leftarrow Y$)
  5. Proof by construction
  6. Proof by induction
  7. Pigeonhole principle
  8. Dovetailing to prove a set is countably infinite E.g. $\mathbb{Z}$ or $\mathbb{N} \times \mathbb{N}$
  9. Diagonalization to prove a set is uncountable E.g. $2^\mathbb{N}$ or Reals
Languages and Machines (Chapter 1)

- Language = set of strings over an alphabet
  - Empty language = language with no strings = $\emptyset$
  - Language containing only empty string = $\{\varepsilon\}$

- DFAs
  - Formal definition $M = (Q, \Sigma, \delta, q_0, F)$
  - Set of states $Q$, alphabet $\Sigma$, start state $q_0$, accept ("final") states $F$, transition function $\delta: Q \times \Sigma \rightarrow Q$
  - $M$ recognizes language $L(M) = \{w \mid M$ accepts $w\}$
  - In class examples:
    - E.g. DFA for $L(M) = \{w \mid w$ ends in $0\}$
    - E.g. DFA for $L(M) = \{w \mid w$ does not contain $00\}$
    - E.g. DFA for $L(M) = \{w \mid w$ contains an even # of $0$’s and an odd number of $1$’s$\}$
    - Try: DFA for $L(M) = \{w \mid w$ contains an even # of $0$’s and an odd number of $1$’s$\}$
Languages and Machines (cont.)

- **Regular Language** = language recognized by a DFA
- **Regular operations**: Union $\cup$, Concatenation $\circ$ and star $^*$
  - Know the definitions of $A \cup B$, $A \circ B$ and $A^*$
  - $\Sigma = \{0,1\} \Rightarrow \Sigma^* = \{\varepsilon, 0, 1, 00, 01, \ldots\}$
- **Regular languages are closed under the regular operations**
  - Means: If $A$ and $B$ are regular languages, we can show $A \cup B$, $A \circ B$ and $A^* \text{ (and also } B^*\text{)}$ are regular languages
  - Cartesian product construction for showing $A \cup B$ is regular by simulating DFAs for $A$ and $B$ in parallel
- **Other related operations**: $A \cap B$ and complement $\overline{A}$
  - Are regular languages closed under these operations?

NFAs, Regular expressions, and GNFAs

- **NFAs vs DFAs**
  - DFA: $\delta(\text{state}, \text{symbol}) = \text{next state}$
  - NFA: $\delta(\text{state}, \text{symbol or } \varepsilon) = \text{set of next states}$
    - Features: Missing outgoing edges for one or more symbols, multiple outgoing edges for same symbol, $\varepsilon$-edges
  - Definition of: NFA $N$ accepts a string $w \in \Sigma^*$
  - Definition of: NFA $N$ recognizes a language $L(N) \subseteq \Sigma^*$
  - E.g. NFA for $L = \{w \mid w = x1a, x \in \Sigma^* \text{ and } a \in \Sigma\}$
- **Regular expressions**: Base cases $\varepsilon$, $\emptyset$, $a \in \Sigma$, and $R_1 \cup R_2$, $R_1 \circ R_2$ or $R_1^*$
- **GNFAs** = NFAs with edges labeled by regular expressions
  - Used for converting DFAs to regular expressions
Main Results and Proofs

- **L is a Regular Language iff**
  - L is recognized by a DFA iff
  - L is recognized by an NFA iff
  - L is recognized by a GNFA iff
  - L is described by a Regular Expression

**Proofs:**
- NFA \( \Rightarrow \) DFA: subset construction (1 DFA state = subset of NFA states)
- Reg Exp \( \Rightarrow \) NFA: combine NFAs for base cases with \( \varepsilon \)-transitions
- DFA \( \Rightarrow \) GNFA \( \Rightarrow \) Reg Exp: Repeat two steps:
  1. Collapse two parallel edges to one edge labeled \((a \cup b)\), and
  2. Replace edges through a state with a loop with one edge labeled \((ab*\varepsilon)\)

Other Results

- Using NFAs to show that Regular Languages are closed under:
  - Regular operations \( \cup, \cdot \) and *
- Are Regular Languages closed under:
  - intersection?
  - complement (Exercise 1.10)?
- Are other operations that regular languages are closed under?
What about the \textit{reversal} operation?

What about the \textit{icannotact} operation?

What about the \textit{subset} operation?

Other Results

- Are Regular Languages closed under:
  - reversal (Problem 1.24)?
  - subset $\subseteq$?
  - superset $\supseteq$?
  - no-prefix (Problem 1.32a)?
    
    no-prefix($A$) = \{ $w \in A$ | no proper prefix of $w$ is in $A$ \}
  - no-extend (Problem 1.32b)?
    
    no-extend($A$) = \{ $w \in A$ | $w$ is not a proper prefix of any string in $A$ \}
Pumping Lemma

- **Pumping lemma in plain English (sort of):** If $L$ is regular, then there is a $p$ (= number of states of a DFA accepting $L$) such that any string $s$ in $L$ of length $\geq p$ can be expressed as $s = xyz$ where $y$ is not null ($y$ is the loop in the DFA), $|xy| \leq p$ (loop occurs within $p$ state transitions), and any “pumped” string $xy^iz$ is in $L$ for all $i \geq 0$ (go through the loop 0 or more times).

- **Pumping lemma in plain Logic:**
  
  $L$ regular $\Rightarrow \exists p$ s.t. $(\forall s \in L \text{ s.t. } |s| \geq p \ (\exists x, y, z \in \sum^* \text{ s.t. } (s = xyz) \text{ and } (|y| \geq 1) \text{ and } (|xy| \leq p) \text{ and } (\forall i \geq 0, xy^iz \in L)))$

- Is the other direction $\Leftarrow$ also true?
  
  **No! See Problem 1.37 for a counterexample**

Proving Non-Regularity using the Pumping Lemma

- **Proof by contradiction to show $L$ is not regular**
  1. Assume $L$ is regular
  2. Let $p$ be some arbitrary number (“pumping length”)
  3. Choose a long enough string $s \in L$ such that $|s| \geq p$
  4. Let $x, y, z$ be strings such that $s = xyz$, $|y| \geq 1$, and $|xy| \leq p$
  5. Pick an $i \geq 0$ such that $xy^iz \not\in L$ (for all $x, y, z$ as in 4)

  This contradicts the pump. lemma. Therefore, $L$ is not regular

- **Examples:** $\{0^n1^n \mid n \geq 0\}$, $\{ww \mid w \in \sum^*\}$, $\{0^n \mid n \text{ is prime}\}$, ADD = $\{x=y+z \mid x, y, z \text{ are binary numbers and } x \text{ is sum of } y \text{ and } z\}$

- **Can sometimes also use closure under $\cap$ (and/or complement)**
  - E.g. If $L \cap B = L_1$, and $B$ is regular while $L_1$ is not regular, then $L$ is not regular (if $L$ was regular, $L_1$ would have to be regular)
Some Applications of Regular Languages

- Pattern matching and searching:
  - E.g. In Unix:
    - `ls *.c`
    - `cp /myfriends/games/*.c /mydir/`
    - `grep 'Spock' *trek.txt`

- Compilers:
  - `id ::= letter (letter | digit)*`
  - `int ::= digit digit*`
  - `float ::= d d*.d* (ε|E d d*)`
  - The symbol `|` stands for “or” (= union)

Good luck on the midterm on monday!

- You can bring one 8 1/2” x 11” review sheet
- The questions sheet will have space for answers. I will also bring extra blank sheets for those of you who balk at brevity.

  Don’t sweat it!

  • Go through the homeworks, lecture slides, and examples in the text (Chapters 0 and 1 only)
  • Do the practice midterm on the website and avoid being surprised!