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 N, integers Z, reals 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\}

Basic Concepts (cont.)

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

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

Languages and Proof Techniques

- Language \( L \) = set of strings over an alphabet (i.e. \( L \subseteq \Sigma^* \))
  - E.g. \( L = \{0^n1^n \mid n \geq 0\} \) over \( \Sigma = \{0,1\} \)
  - E.g. \( L = \{p \mid p \) is a syntactically correct C++ program\} over \( \Sigma \) = 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. \( Z \) or \( N \times N \)
  9. Diagonalization to prove a set is uncountable E.g. \( 2^N \) or Reals
Languages and Machines (Chapter 1)

- **Language** = set of strings over an alphabet
  - Empty language = language with no strings = ∅
  - Language containing only empty string = \{ε\}

- **DFAs**
  - Formal definition \(M = (Q, \Sigma, δ, q_0, F)\)
  - Set of states \(Q\), alphabet \(\Sigma\), start state \(q_0\), accept (“final”) states \(F\), transition function \(δ: Q \times Σ \rightarrow Q\)
  - \(M\) recognizes language \(L(M) = \{w | M\) accepts \(w\}\)
  - In class examples:
    - E.g. DFA for \(L(M) = \{w | w\) ends in 0\}
    - E.g. DFA for \(L(M) = \{w | w\) does not contain 00\}
    - E.g. DFA for \(L(M) = \{w | w\) contains an even # of 0’s\}
  - Try: DFA for \(L(M) = \{w | w\) 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 \( A^* \) (and also \( B^* \)) 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: \( \delta(\text{state}, \text{symbol} \text{ or } \varepsilon) = \text{set of next states} \)
    ➔ 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 \( R1 \cup R2, R1 \circ R2 \) or \( R1^* \)
✦ GNFAs = NFAs with edges labeled by regular expressions
  ➔ Used for converting NFAs/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 → DFA: subset construction (1 DFA state = subset of NFA states)
  - DFA → GNFA → 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 \((ab^*c)\)
  - Reg Exp → NFA: combine NFAs for base cases with \(\epsilon\)-transitions

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 there other operations that regular languages are closed under?
What about the reversal operation?

What about the \textit{icannotact} operation?

What about the subset operation?

Other Results

- Are Regular languages closed under:
  - reversal?
  - subset $\subseteq$ ?
  - superset $\supseteq$ ?
  - MAX?
  \[
  \text{MAX}(L) = \{ w \in L \mid w \text{ is not a proper prefix of any string in } L \}\]
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. ($\forall s \in L$ s.t. $|s| \geq p$ ($\exists x,y,z \in \Sigma^*$ s.t. ($s = xyz$) and ($|y| \geq 1$) and ($|xy| \leq p$) 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 $xy^iz \notin L$ (for all $x,y,z$ as in 4)
  
  This contradicts the pump. lemma. Therefore, $L$ is not regular

- Examples: $\{0^n1^n|n \geq 0\}$, $\{ww| w \in \Sigma^*\}$, $\{0^n|n$ is prime$\}$, ADD = $\{x=y+z | x, y, z$ are binary numbers and $x$ is sum of $y$ 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/**.* /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. We 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!