Course Topics

• **Computer Theory**
  – Formal Languages
  – Computational Models/Grammars/Regular Expressions
  – Complexity/Intractability
  – Computability/Decidability

• **Applications**
  – Compilers and programming languages
  – Computer design
  – Algorithm analysis
  – Artificial intelligence and human language processing
  – Parsers
Representations

- **Automaton/Turing Machine**
  - Automaton/Turing machine is a computational model to recognize if a string is in a language

- **Grammar**
  - Grammar is a set of rule describing a language

- **Regular Expression**
  - Regular expression describes a set of patterns which form a language.
Formal Proof

- Deductive Proof
- Proof by contradiction
- Counterexamples
- Inductive Proof
- Constructive Proof
- Set Equivalence Proof
Deductive Proof

- P => Q
- If P is true, then Q is true
- Not Q => Not P
Proof by contradiction and with Counterexamples

• Given P => Q and P, prove Q.
• Assume Q is false and derive contradiction such as not P.

Prove a false statement with a counter example.
Inductive Proof

• Induction on integers
  – Basis: prove $P$ is true for a particular integer $i$.
  – Inductive step: for any $n > i$ if $P(n)$ is true, prove $P(n+1)$ is true.

• Structural Induction
  – Prove statements on recursively defined structures such as trees and expressions.
Constructive and Set Equivalence Proof

• Constructive Proof
  – Prove a language is regular by constructing an automaton accepting the language

• Set Equivalence Proof
  – Two sets, S1 and S2 are equivalent if they contain same elements.
  – If e is an element of S1, it is also an element of S2.
  – If e is an element of S2, it is also an element of S1.
Set

• A set is a collection of objects without repetition.
• Set specification
  – list all members: \{0, 1\}
  – use a predicate: \{x | P(x)\} means for all x such that P(x) is true.
    \{i | i is an integer and i mod 2 = 0\} is the set of all even integers.
• A set may have finite or infinite number of members.
• Set relations
  – subset: a set A is a subset of B, denoted by, \( A \subseteq B \) if all members of A are also members of B.
  – Equality: a set A is equal to another set B if they contain the same members.
Set Operations

- Union:
  - $A \cup B = \{x | x \text{ is in } A \text{ or } x \text{ is in } B\}$

- Intersection:
  - $A \cap B = \{x | x \text{ is in } A \text{ and } x \text{ is in } B\}$

- Difference:
  - $A - B = \{x | x \text{ is in } A \text{ and } x \text{ is not in } B\}$

- Complement
  - $\overline{A} = \{x | x \text{ is not in } A\}$
Alphabet

• An alphabet is a finite set of symbols, usually denoted by $\Sigma$.

• Examples:
  
  Binary alphabet $\Sigma = \{0, 1\}$
  $\Sigma = \{a, b\}$
  $\Sigma = \{a, b, c, d, e, \ldots, x, y, z\}$
String

- A string is a finite sequence of symbols of $\Sigma$.
- Examples
  - string
  - cs5104
- Length of a string $w$
  - the number of symbols in the string and denoted by $|w|$  
- Empty string $\varepsilon$
  - the string with no symbol and $|\varepsilon| = 0$
String Operations

- Concatenation:
  \( xy \) is the string with \( x \) followed by \( y \).
  Example: \( x = cs \) and \( y = 510 \),
  \( xy = cs5104 \)

- Power of string
  - \( x^0 = \varepsilon \)
  - \( x^1 = x \)
  - \( x^n = xx^{n-1} \)
  - Example: \( x = 01 \), \( x^4 = 01010101 \)
Substrings

- Substring
  A string $s$ is a substring of a string $x$ if there exist strings $y$ and $z$ such that $x = ysz$.
  $51$ is a substring of $cs5104$ and $cs$ is also a substring of $cs5104$.

- Prefix
  when $x = sz$, $s$ is called a prefix of $x$. $cs$ is a prefix of $cs5104$

- Suffix
  when $x = ys$, $s$ is called a suffix of $x$. $5104$ is a suffix of $cs5104$
String Reversal

- The reversal of $x$, denoted by $x^R$, is defined as follows.
  - $x^R = \varepsilon$
  - $x^R = a(u^R)$ if $x = ua$ and $a$ is symbol and $u$ is a string.
  - Example:
    $(abc)^R = (\overline{ab}c)^R = c(ab)^R = c((a)b)^R = cb(a)^R = cba$
  - For strings $x$ and $y$, $(xy)^R = y^Rx^R$
Power of Alphabets

- Power $n$ of an alphabets is the set of all strings with length $n$.

\[ \Sigma^n = \{ x \mid |x| = n \} \]

\[ \Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots \]

$\Sigma^*$ contains all finite strings over $\Sigma$. $\Sigma^*$ is infinite, but it contains no infinite string.
Language

• Language
  – a language is a set of strings from a given alphabet

• Two special languages
  – empty language: $\emptyset = \{\}$, language with no string
  – language with the empty string $\{\varepsilon\}$

• Examples
  – palindrome over the alphabet $\{a, b\}$
  – strings with equal number of 0s and 1s over the alphabet $\{0, 1\}$

• Formal language
  – languages strictly defined with rules
Language

- Any $L \subseteq \Sigma^*$ is a language. Namely, a language is a subset of all strings of an alphabet.

- Examples:
  - $\emptyset = \{\}$
  - $\{\varepsilon\}$
  - $\Sigma^*$
  - The set of strings containing an equal number of 0s and 1s.
## Formal Language

<table>
<thead>
<tr>
<th>Computation Model</th>
<th>Formal Language</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finite State Machines</td>
<td>Regular Language</td>
<td>Text processing, lexical analysis</td>
</tr>
<tr>
<td>Pushdown Automata</td>
<td>Context Free Language</td>
<td>Compiler parsing</td>
</tr>
<tr>
<td>Turing Machines</td>
<td>Recursive Languages</td>
<td>Undecidability, complexity</td>
</tr>
</tbody>
</table>
Kleene Closure (Star Closure)

• For two languages $L_1$ and $L_2$ over $\Sigma$, $L_1L_2$ is the concatenation of $L_1$ and $L_2$.
  
  \[ L_1L_2 = \{xy|x \text{ is in } L_1 \text{ and } y \text{ is in } L\} \]

• $L^0 = \{\varepsilon\}$, $L^1 = L$, $L^2 = LL$, …, $L^n = L \ L^{n-1}$

• Kleene closure (star closure):
  \[ L^* = L^0 U L^1 U L^2 U \ldots \]

• $L^+ = L^1 U L^2 U \ldots$

• $L^* = L^{**}$
  – Prove: $L^* \subseteq L^{**}$ and $L^{**} \subseteq L^*$
Recursive Definition

• A recursive definition of a set is a method to specify a set. It defines a set by specifying members of the set using previously defined members.

• Recursive definitions
  – base case definition
  – recursive definition
  – no other objects in the set
Recursive Definition Examples

• Odd integers
  – base case: 1 is an odd integer.
  – recursive definition: if n is odd, then n+2 is odd.
  – no other objects are odd.

• All binary strings with even number of symbols
  – base case:
  – recursive definition:
  – no other objects in the set
Problems

• How to define a language formally.
  – regular expression
  – grammars

• Given a language L and string w, how to decide if w is a string of L.
  – Automata