CSE 105 Discussion Week 1 Friday

<table>
<thead>
<tr>
<th>Definitions</th>
<th>Notes</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbol is an element of the alphabet.</td>
<td>Symbols are single characters. Thus, $\varepsilon \notin \Sigma$.</td>
<td>a, b, c</td>
</tr>
<tr>
<td>Alphabet is a non-empty finite set of symbols.</td>
<td>$\Sigma^*$ is the set of all strings over $\Sigma$.</td>
<td>$\Sigma = {a, b, c}$</td>
</tr>
<tr>
<td>String is a sequence of symbols.</td>
<td>A string over $\Sigma$ has all its symbols in $\Sigma$.</td>
<td>Strings abc, abba, $\varepsilon$ are over $\Sigma$.</td>
</tr>
<tr>
<td>A language over $\Sigma$ is a set of strings over $\Sigma$.</td>
<td>Language over $\Sigma \subseteq \Sigma^*$.</td>
<td>{abc, abba, $\varepsilon$} is a language over $\Sigma$.</td>
</tr>
<tr>
<td>A regular expression over alphabet $\Sigma$ is a syntactic expression</td>
<td>You can think of regex as sequences of symbols that specify a match</td>
<td>(ab)* (a $\cup$ b)</td>
</tr>
<tr>
<td>that can describe a language over $\Sigma$.</td>
<td>pattern, except $\emptyset$ and $\varepsilon$.</td>
<td></td>
</tr>
</tbody>
</table>

Regular Expressions:
- **Base Step**: $\varepsilon$, $\emptyset$, and $a$ are regular expressions, where $a$ is a symbol in $\Sigma$.
- **Recursive Step**: $(R_1 \cup R_2)$, $(R_1 \circ R_2)$, $(R_1^*)$ are regular expressions, where $R_1$ and $R_2$ are regular expressions.

Language described by regular expression:
- $L(\varepsilon) = \{\varepsilon\}$
- $L(\emptyset) = \emptyset$
- $L(x) = \{x\}$, for any symbol $x$.
- $L(R_1 \cup R_2) = L(R_1) \cup L(R_2) = \{s \mid s \in L(R_1) \text{ or } s \in L(R_2)\}$
- $L(R_1 \circ R_2) = L(R_1) \circ L(R_2) = \{uv \mid u \in L(R_1) \text{ and } v \in L(R_2)\}$
- $L(R_1^*) = (L(R_1))^* = \{w_1 \ldots w_k \mid k \geq 0 \text{ and each } w_i \in L(R_1)\}$

Example:
- $L(ab^*) = \{a, ab, abb, abbb, \ldots \}$
- $L((ab)^*) = \{\varepsilon, ab, abab, ababab, \ldots \}$
1.1 M_1 

![Diagram of M_1](image)

(a) Start state has an arrow pointing from nowhere to it.
  - M_1: \( q_1 \)
  - M_2: \( q_1 \)

(b) Accept states have double circles.
  - M_1: \{q_2, q_3\}
  - M_2: \{q_4\}

(c) Input is \( aabb \)
  - M_1: \( q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3 \xrightarrow{b} q_1 \)
  - States: \( q_1, q_2, q_3, q_4, q_1 \) = Answer
  - M_2: \( q_1 \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_4 \)
  - States: \( q_1, q_1, q_1, q_2, q_4 \) = Answer

(d) The final state on M_1 is \( q_1 \), which is not an accept state. M_1 does not accept \( aabb \).

The final state on M_2 is \( q_4 \), which is an accept. M_2 does accept \( aabb \).

(e) M_1: No. The final state for empty string \( \epsilon \) is \( q_1 \), which is not an accept state.
  M_2: Yes. The final state for \( \epsilon \) is \( q_2 \), which is an accept state.
A finite automaton is a 5-tuple 
\((Q, \Sigma, S, q_0, F)\), where 
\(Q\) is states, \(\Sigma\) is alphabet, 
\(S\) is transition function, \(q_0 \in Q\) is the 
start state and \(F \subseteq Q\) is set of accept 
states.

DFA \(\delta : Q \times \Sigma \rightarrow Q\), 
NFA \(\delta : Q \times \Sigma \varepsilon \rightarrow P(Q)\), 
where \(\Sigma \varepsilon = \Sigma \cup \{\varepsilon\}\), 
\(P(Q)\) is power set of \(Q\) 
(Non-determinism will be covered next week).

1.2 \(M_1 = (\{q_1, q_2, q_3, 3\}, \{a, b, 3\}, S_1, \) 
\(q_1, \{q_1, q_2, q_3\}\), where

\[
\begin{array}{c|cc}
S_1 & a & b \\
\hline
q_1 & q_2 & q_1 \\
q_2 & q_3 & q_2 \\
q_3 & q_2 & q_1 \\
\end{array}
\]

\(M_2 = (\{q_1, q_2, q_3, q_4, q_4\}, \{a, b, 3\}, S_2, q_1, \{q_1, q_2, q_3, q_4\}\)

\[
\begin{array}{c|cc}
S_2 & a & b \\
\hline
q_1 & q_1 & q_2 \\
q_2 & q_3 & q_4 \\
q_3 & q_2 & q_1 \\
q_4 & q_3 & q_4 \\
\end{array}
\]
1.3 \[ M = \{ \{q_1, q_2, q_3, q_4, q_5\}, S, q_3, \{q_3\}\} \]

\[
\begin{array}{|c|c|c|}
\hline
S & u & d \\
q_1 & q_1 & q_2 \\
q_2 & q_1 & q_3 \\
q_3 & q_2 & q_4 \\
q_4 & q_3 & q_5 \\
q_5 & q_4 & q_5 \\
\hline
\end{array}
\]

1.18 Let \( L_0 = \{ w \mid w \text{ begins with } 1 \text{ and ends with } 0^3 \} \)
\( \Sigma = \{0, 1\} \)

Let \( R = 1 \Sigma^* 0 \)

We want language over regular expression \( R \) to be the same as \( L_0 \).

\( L(R) = L_0 \)
\( L(R) = \{10, 100, 110, 1000, \ldots\} \)
(b) \[ R = \Sigma^* \{ \Sigma^* | \Sigma^* \} \Sigma^* \]
\[ L = \{ w \mid w \text{ contains at least three } 1s \} \]

(c) \[ R = \Sigma^* 0101 \Sigma^* \]
\[ L = \{ w \mid w \text{ contains substring } 0101 \} \]

(d) \[ R = \Sigma \Sigma 0 \Sigma \]
\[ L = \{ w \mid |w| \geq 3 \text{ and third symbol is } 0 \} \]

(e) \[ L = \{ w \mid w \text{ doesn't contain the substring } 110^2 \} \]
\[ R = \{ 0^* (10^+ \} \}^* \]

(Answer changed since discussion section)

(f) \[ L = \{ w \mid w \text{ is any string except } \text{ and } \text{ and } \text{ and } \} \]
\[ R = (\varepsilon \cup \{ 0 \} \cup \{ 1 \} \cup \{ 0 \} \Sigma^* \cup \{ 1 \} \Sigma^* \cup \{ 11 \} \Sigma^* \cup \{ 1 \} \Sigma^* \} \]

1.23 \[ B^+ = BB^+ \text{ (Solution given back of the chapter)} \]

Case 1: Assume \( B = B^+ \)
\[ B^+ = BB^+ \]
\[ \Rightarrow BB \subseteq BB^+ = B^+ = B \]
\[ \Rightarrow BB \subseteq B \]

\[ \therefore B = B^+ \Rightarrow BB \subseteq B \]

Case 2: Assume \( BB \subseteq B \)
\[ B \subseteq BB^* = B^+ \]
\[ \Rightarrow B \subseteq B^+ \]
Suppose \( w \in B^+ \).
\[ \exists x_1, x_2, \ldots, x_k \in B \text{ such that } \]
\[ w = x_1 x_2 \ldots x_k, \text{ for some } k \geq 1. \]

\[ x_1, x_2 \in B \]
\[ \Rightarrow x_1 x_2 \in BB \]
\[ \Rightarrow x_1 x_2 \in B \]

\[ x_1 x_2, x_3 \in B \]
\[ \Rightarrow x_1 x_2 x_3 \in BB \]
\[ \Rightarrow x_1 x_2 x_3 \in B \]

\[ \cdots \]
\[ \Rightarrow w = x_1 x_2 \ldots x_k \in B \]

Thus, \( \forall w \in B^+, w \in B \)
\[ \Rightarrow B^+ \subseteq B \]

\( B^+ \subseteq B \) and \( B \subseteq B^+ \)
\[ \Rightarrow B^+ = B \]

Therefore, \( BB \leq B \Rightarrow B^+ = B \)

Proved. \( BB \leq B \) iff \( B^+ = B \)