Modeling Computation

Introduction to Formal Languages and Automata
Regular Expressions
Regular Expressions

• Formal specifications of finite automata.

• Any language that is accepted by a finite automaton can be described by a regular expression.

• Such languages are called Regular Languages.

• NFAs and DFAs accept regular languages.
Operations on Languages (languages are sets)

- Union: $L_1 \cup L_2$
- Intersection: $L_1 \cap L_2$
- Complement: $\overline{L}$
- Difference: $L_1 - L_2$
- Concatenation: $L_1L_2$
- Kleene Star: $L^*$
Concatenation $L_1L_2$ of languages $L_1$, $L_2$ is

$$\{uv | u \in L_1, v \in L_2\}$$

Ex: $L_1 = \{a^n | n > 0\}$

$L_2 = \{b^m | m > 0\}$

$L_1L_2 = \{a^n b^m | n > 0, m > 0\}$

$L^n$ denotes $L$ concatenated with itself $n$ times. $L^3 = L(LL)$
Kleene Star $L^*$ of language $L$ is the infinite union

$$\bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \cdots$$

Ex: $L = \{a^3\}$

$L^* = \{\varepsilon, a, aa, aaa, \ldots\} = \{a^n | n \geq 0\}$

Ex: $L = \{oo, la^3\}$

$L^* = \{\varepsilon, oo, la, o000, oola, la00, lala, 00000, 000000, 0000000, \ldots\}$
Theorem: If languages $L$ and $M$ are accepted by finite automata, then so are:

- $L \cup M$ \quad \text{union}
- $L \cap M$ \quad \text{intersection}
- $\overline{L}$ \quad \text{complement} \quad \left( = \Sigma^* - L \right)
- $L - M$ \quad \text{difference}
- $LM$ \quad \text{concatenation}
- $L^*$ \quad \text{Kleene Star}
Proof (by construction)
Suppose L, M are accepted by automata A,B.
Concatenation $LM$
Kleene Star $L^*$
Complement \( \overline{L} \): every favorable state becomes unfavorable, every state becomes favorable.

*MUST convert to DFA first*

Intersection

\[
L \cap M = \overline{L \cup M} \\
\bar{L} \cap M = L \cap \bar{M} = \overline{L \cup M}
\]

Difference

\[
L - M = L \cap \bar{M} = \overline{L \cup M}
\]
A regular expression is a string over the alphabet $\Sigma = \{(,),\varepsilon,\varnothing,\cup,\star\}$.

Recursive definition:

Basis Step: $\emptyset, \varepsilon, \alpha \in \Sigma$ are regular expressions. (ground elements)

Recursive Step: If $\alpha, \beta$ are regular expressions, then so are

- $(\alpha \cup \beta)$
- $(\alpha \beta)$
- $\alpha^*$

Examples:
- $(ab \cup bc) \subseteq \text{easy}$
- $(ab \cup (ab) \cup b) \not\subseteq \emptyset$ (not so much)
$L(\alpha)$ is the language represented by regular expression $\alpha$.

$L(\emptyset) = \emptyset$ \hspace{1cm} $L(\epsilon) = \{ \epsilon \}$

$L(a) = \{a\}$

If $\alpha, \beta$ are regular expressions, then

$L(\alpha \cup \beta) = L(\alpha) \cup L(\beta)$

$L(\alpha\beta) = L(\alpha)L(\beta)$

$L(\alpha^*) = \left( L(\alpha) \right)^*$
Exercise: What language is represented by 
\[(ab^*) \cup (a^*b)\]

\[
L( (ab^*) \cup (a^*b)) \\
L( (ab^*) \cup L(a^*b)) \\
(L((a) L(b^*)) \cup (L(a) L(b))) \\
(L(a) L(b^*)) \cup (L(a) L(b)) \\
(\{a^*b^*\} \cup \{a^*b^*\}) = \{w \mid w \text{ is of the form } ab^n \text{ or } a^nb, n \geq 0\}
\]