Basic Regular Expressions

A Regular Expression (RE in short) is a string of symbols that describes a regular Language.

Three base cases:
• Let $\Sigma$ be an alphabet. For each $a \in \Sigma$, the symbol $a$ is an RE representing the set $\{a\}$.
• The symbol $\varepsilon$ is an RE representing the set $\{\varepsilon\}$. (The set containing the empty string).
• The symbol $\phi$ is an RE representing the empty set.

Some Useful Notation

Let $R$ be a regular expression:
• The string $R^+$ represents $RR^*$, and it also holds that $R^* \cup \{\varepsilon\} = R^*$.
• The string $R^k$ represents $\underbrace{RR\ldots R}_{k \text{ times}}$.
• The symbol $\Sigma$ represents $\{a_1, a_2,\ldots, a_k\}$.
• The Language represented by $R$ is denoted by $L(R)$.

Three Recursive Constructions

Let $R_1$ and $R_2$ be two regular expressions representing languages $L_1$ and $L_2$, resp.
• The string $(R_1 \cup R_2)$ is a regular expression representing the set $L_1 \cup L_2$.
• The string $(R_1R_2)$ is a regular expression representing the set $L_1 \circ L_2$.
• The string $(R_1)^*$ is a regular expression representing the set $L_1^*$.

Precedence Rules

• The star (*) operation has the highest precedence.
• The concatenation (\circ) operation is second on the preference order.
• The union (\cup) or (+) operation is the least preferred.
• Parentheses can be omitted using these rules.
Examples

- $0^*10^* = \{ w \mid w \text{ contains a single } 1 \}$
- $\Sigma^*1\Sigma^* = \{ w \mid w \text{ has at least a single } 1 \}$
- $\Sigma^*(101)\Sigma^* = \{ w \mid w \text{ contains } 101 \text{ as a substring} \}$
- $1^*(01^*)^* = \{ w \mid \text{ every } 0 \text{ in } w \text{ is followed by at least a single } 1 \}^*$
- $(\Sigma\Sigma)^* = \{ w \mid w \text{ is of even length} \}$.

Equivalence With Finite Automata

Regular expressions and finite automata are equivalent in their descriptive power. This fact is expressed in the following Theorem:

**Theorem**

A set is regular if and only if it can be described by a regular expression.

The proof has two parts.

Examples

- $0\Sigma^*0\cup1\Sigma^*1\cup0\cup1^*$ - all words starting and ending with the same letter.
- $(0\cup\varepsilon)1^* = 01^*\cup1^*$ - all strings of forms $1,1,\ldots,1$ and $0,1,1,\ldots1$.
- $R\emptyset = \emptyset$ - A set concatenated with the empty set yields the empty set.
- $\emptyset^* - \emptyset^* = \{ \varepsilon \}$.

Part 1 ->

If a language $L$ can be described by regular expression then $L$ is regular.
Part 2 <-

If a language $L$ is regular then $L$ can be described by some regular expression.

The textbook proof:

The proof follows the following stages:
1. Define Generalized Nondeterministic Finite Automaton (GNFA in short). Instead of $\delta:Q \times \Sigma \rightarrow Q$, we use $\delta:Q \times \text{RE} \rightarrow Q$.
2. Show how to convert any DFA to an equivalent GNFA.
3. Show an algorithm to convert any GNFA to an equivalent GNFA with 2 states.
4. Convert a 2-state GNFA to an equivalent RE.

Right-linear Equations over RE

- Right-linear Equation over RE:
  $-X = r_1X + r_2$, where $X$ is a variable and $r_1$ and $r_2$ are regular expressions.
- The solution of the above equation is $-X = (r_1)^* r_2$.
- Example: given $X = 01X + 0$, then $X = (01)^*0$.

From DFA to System of Right-linear Equations over RE

- Given a DFA $M = (Q, \Sigma, \delta, q_0, F)$
- For each state $q$, define $X_q$ to be the set of strings which run from $q$ and end with a final state:
  $-X_q = \{ w \mid w \in \Sigma^*, \delta(q, w) \text{ is in } F \}.$
- Obviously, $L(M) = X_{q_0}$.
- Suppose $\delta(q, a_i) = p$, for $\Sigma = \{a_1, a_2, \ldots, a_k\}$.
  - If $q$ is not in $F$, then
    $-X_q = a_1X_{p_1} + a_2X_{p_2} + \ldots + a_kX_{p_k}$
  - If $q$ is in $F$, then
    $-X_q = a_1X_{p_1} + a_2X_{p_2} + \ldots + a_kX_{p_k} + \varepsilon.$
Solving System of Equations

- By substitution and simplification,
  - \( r_1 + r_2 = r_2 + r_1 \),
  - \( r_1 \varepsilon = \varepsilon r_1 = r_1 \),
  - \( r_1 (r_2 r_3) = (r_1 r_2) r_3 = r_1 r_2 r_3 \),
  - \( r_1 (r_2 + r_3) = r_1 r_2 + r_1 r_3, (r_2 + r_3) r_1 = r_2 r_1 + r_3 r_1 \),
  - \( r_1^+ = r_1 r_1^*, r_1^* = r_1^+ + \varepsilon \),
- and the solution \( X = (r_1)^* r_2 \) of \( X = r_1 X + r_2 \).
- The solution of \( Xq_0 \) is the RE we looked for.