Ch 1.3 Regular Expressions (S)

Say that R is a **regular expression** if R is
1. a for some a in the alphabet Σ,
2. ϵ,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions,
5. $(R_1 \circ R_2)$, where R_1 and R_2 are regular expressions, or
6. (R_1^*), where R_1 is a regular expression.

▼ *Remark 1*

In items 1 & 2, the regular expressions a and ϵ represent the languages $\{a\}$ and $\{\epsilon\}$, respectively. In item 3, the regular expression $\emptyset$ represents the empty language. In items 4, 5, and 6, the expressions represent the languages obtained by taking the union or concatenation of the languages R_1 and R_2, or the star of the language R_1, respectively.

▼ *Remark 2*

The expression ϵ represents the language containing a single string — namely, the empty string — whereas $\emptyset$ represents the language that doesn't contain any strings.

❓Example 1.53

Equivalence with Finite Automata

Any regular expression can be converted into a finite automaton that recognizes the language it describes, and vice versa.

Theorem A language is regular *iff* some regular expression describes it.

Lemma If a language is described by a regular expression, then it is regular.

Example 1.58 **Convert regular expression to an NFA**

Lemma If a language is regular, then it is described by a regular expression.

A **generalized nondeterministic finite automaton** is a 5-tuple, $(Q, \Sigma, \delta, q_{\text{start}}, q_{\text{accept}})$, where
1. Q is the finite set of states,
2. Σ is the input alphabet,
3. $\delta : (Q - \{q_{\text{accept}}\}) \times (Q - \{q_{start}\}) \longrightarrow \mathcal{R}$ is the transition function,
4. q_{start} is the start state, and
5. q_{accept} is the accept state.