COSE215: Theory of Computation

Lecture 6 — Regular Expressions and Finite Automata

Hakjoo Oh
2018 Spring

Equivalence between Regular Expressions and Finite Automata

Theorem (From RE to FA)

Every language defined by a regular expression is also defined by a finite automaton.

Theorem (From FA to RE)

Every language defined by some finite automata is also defined by a regular expression.

Conversion From Regular Expression to Finite Automata

Given a regular expression R, we show that $L(R)$ is accepted by an ϵ-NFA such that

- it has exactly one accepting state,
- no arcs into the initial state, and
- no arcs out of the accepting state.

Conversion from Regular Expression to Finite Automata

The conversion is by structural induction on $\boldsymbol{R}$.

Base cases:

- $\boldsymbol{R = \epsilon}$:
- $\boldsymbol{R = \emptyset}$:
- $\boldsymbol{R = a (\in \Sigma)}$:

From Regular Expression to Finite Automata

Inductive cases:

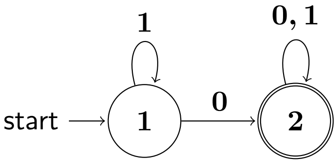
- $\boldsymbol{R = R_1 + R_2}$:
- $\boldsymbol{R = R_1 R_2}$:
- $\boldsymbol{R = R_1^*}$:

Examples

- $\mathbf{0} \cdot \mathbf{1}^*$:
- $(\mathbf{0} + \mathbf{1}) \cdot \mathbf{0} \cdot \mathbf{1}$:
- $(\mathbf{0} + \mathbf{1})^* \cdot \mathbf{1} \cdot (\mathbf{0} + \mathbf{1})$:

From Automata to Regular Expression

Consider DFA D whose states are $\{1, 2, \ldots, n\}$, e.g.,



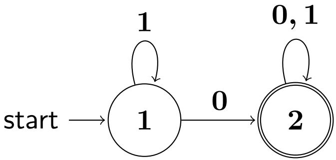
- The idea is to progressively accept more paths in the transition graph.
- Let $R_{ij}^{(k)}$ be the name of a regular expression whose language is the set of strings w such that w is the label of a path from state i to state j in D, and that path has no intermediate node whose number is greater than k.

From Automata to Regular Expressions

When $k = 0$:

1. When $i \neq j$, consider every arc $i \xrightarrow{a} j$ in D.
   1. If there is no such arc, then $R_{ij}^{(0)} = \emptyset$.
   2. If there is exactly one such arc, then $R_{ij}^{(0)} = a$.
   3. If there are multiple arcs $i \xrightarrow{a_1} j$, $i \xrightarrow{a_2} j$, ..., $i \xrightarrow{a_k} j$, then $R_{ij}^{(0)} = a_1 + a_2 + \cdots + a_k$.
2. When $i = j$, consider every arc $i \xrightarrow{a} i$:
   1. If there is no such arc, then $R_{ij}^{(0)} = \epsilon$.
   2. If there is exactly one such arc, then $R_{ij}^{(0)} = \epsilon + a$.
   3. If there are multiple arcs $i \xrightarrow{a_1} i$, $i \xrightarrow{a_2} i$, ..., $i \xrightarrow{a_k} i$, then $R_{ij}^{(0)} = \epsilon + a_1 + a_2 + \cdots + a_k$.

Example



$$R_{11}^{(0)} =$$
$$R_{12}^{(0)} =$$
$$R_{21}^{(0)} =$$
$$R_{22}^{(0)} =$$

From Automata to Regular Expressions

When $k > 0$:

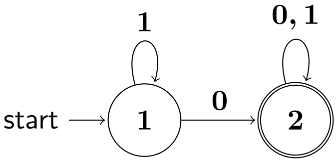
1. When the path does not use state k at all. In this case, the label of the path is in the language of $R_{ij}^{(k-1)}$.
2. When the path goes through state k at least once.

$$R_{ik}^{(k-1)}(R_{kk}^{(k-1)})^* R_{kj}^{(k-1)}$$

By combining the two cases, we have the expression:

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)}(R_{kk}^{(k-1)})^* R_{kj}^{(k-1)}$$

Example



$$
\begin{aligned}
R_{11}^{(1)} &= \\
R_{12}^{(1)} &= \\
R_{21}^{(1)} &= \\
R_{22}^{(1)} &=
\end{aligned}
$$

$$
\begin{aligned}
R_{11}^{(2)} &= \\
R_{12}^{(2)} &= \\
R_{21}^{(2)} &= \\
R_{22}^{(2)} &=
\end{aligned}
$$