Converting a DFA to a REG EXP

Exposition by William Gasarch

1 The Basic Algorithm

Let $M = (Q, \Sigma, \delta, s, F)$ be a DFA. We can assume $Q = \{1, 2, \ldots, n\}$. We show how to construct a reg expression α that generates the same set the DFA M recognizes.

Let $R(i, j, k)$ be a regular expression for the set of strings x such that if you run M started at state i, only using states $\{1, \ldots, k\}$ (or a subset of them), you end up in state j.

We first show how to find $R(i, j, 0)$. Then, assuming one has $R(i, j, k-1)$ for ALL i, j, we derive $R(i, j, k)$ for ALL i, j.

$R(i, j, 0)$: Note that the only way to NOT use ANY states as intermediaries is to either transition directly from i to j. Hence the following seems reasonable:

$$R(i, j, 0) = \{\sigma \in \Sigma \mid \delta(i, \sigma) = j\}.$$

This IS correct if $i \neq j$. However, if $i = j$ then the empty string also takes you from state i to state i without using any intermediary states. So

$$R(i, i, 0) = \{e\} \cup \{\sigma \in \Sigma \mid \delta(i, \sigma) = j\}.$$

(NOTE: To understand this next equation you really need to be in class.)

$$R(i, j, k) = R(i, j, k-1) \cup R(i, k, k-1)R(k, k, k-1)^*R(k, j, k-1)$$

Hence, by induction on k, all of the $R(i, j, k)$ are regular expressions.

Assume that the start state is 1. The regular expression we seek is

$$\bigcup_{f \in F} R(1, f, n)$$