The construction of M is given in Fig. 2.14(b). Every path in M from q_1 to f_2 is a path labeled by some string x from q_1 to f_1, followed by the edge from f_1 to q_2 labeled ϵ, followed by a path labeled by some string y from q_2 to f_2. Thus $L(M) = \{xy \mid x \text{ is in } L(M_1) \text{ and } y \text{ is in } L(M_2)\}$ and $L(M) = L(M_1)L(M_2)$ as desired.

CASE 3 $r = r_1^*$. Let $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, \{f_1\})$ and $L(M_1) = r_1$. Construct

$$M = (Q_1 \cup \{q_0, f_0\}, \Sigma_1, \delta, q_0, \{f_0\}),$$

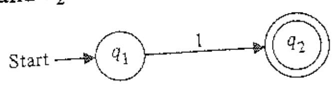
where δ is given by

i) $\delta(q_0, \epsilon) = \delta(f_1, \epsilon) = \{q_1, f_0\}$,

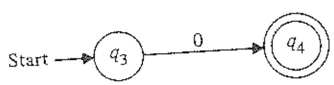
ii) $\delta(q, a) = \delta_1(q, a)$ for q in $Q_1 - \{f_1\}$ and a in $\Sigma_1 \cup \{\epsilon\}$.

The construction of M is depicted in Fig. 2.14(c). Any path from q_0 to f_0 consists either of a path from q_0 to f_0 on ϵ or a path from q_0 to q_1 on ϵ, followed by some number (possibly zero) of paths from q_1 to f_1, then back to q_1 on ϵ, each labeled by a string in $L(M_1)$, followed by a path from q_1 to f_1 on a string in $L(M_1)$, then to f_0 on ϵ. Thus there is a path in M from q_0 to f_0 labeled x if and only if we can write $x = x_1x_2 \cdots x_j$ for some $j \geq 0$ (the case $j = 0$ means $x = \epsilon$) such that each x_i is in $L(M_1)$. Hence $L(M) = L(M_1)^*$ as desired. □

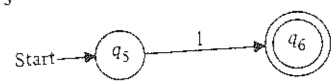
Example 2.12 Let us construct an NFA for the regular expression $\mathbf{01^* + 1}$. By our precedence rules, this expression is really $(\mathbf{0}(\mathbf{1}^*)) + \mathbf{1}$, so it is of the form $r_1 + r_2$, where $r_1 = \mathbf{01}^*$ and $r_2 = \mathbf{1}$. The automaton for r_2 is easy; it is



We may express r_1 as r_3r_4, where $r_3 = \mathbf{0}$ and $r_4 = \mathbf{1}^*$. The automaton for r_3 is also easy:



In turn, r_4 is r_5^*, where r_5 is $\mathbf{1}$. An NFA for r_5 is



Note that the need to keep states of different automata disjoint prohibits us from using the same NFA for r_2 and r_5, although they are the same expression.

To construct an NFA for $r_4 = r_5^*$ use the construction of Fig. 2.14(c). Create states q_7 and q_8 playing the roles of q_0 and f_0, respectively. The resulting NFA for r_4 is shown in Fig. 2.15(a). Then, for $r_1 = r_3r_4$ use the construction of Fig. 2.14(b). The result is shown in Fig. 2.15(b). Finally, use the construction of Fig. 2.14(a) to find the NFA for $r = r_1 + r_2$. Two states q_9 and q_{10} are created to fill the roles of q_0 and f_0 in that construction, and the result is shown in Fig. 2.15(c).

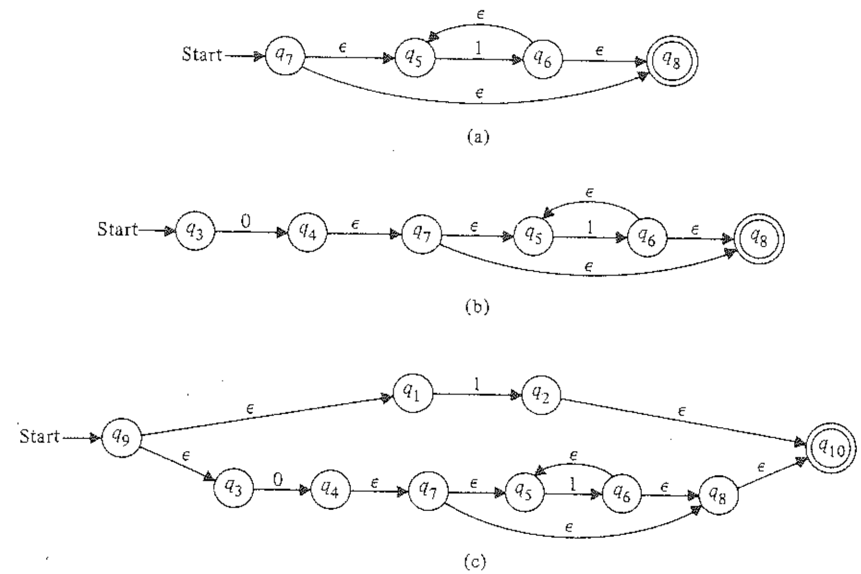


Fig. 2.15 Constructing an NFA from a regular expression. (a) For $r_4 = \mathbf{1}^*$. (b) For $r_1 = \mathbf{01}^*$. (c) For $r = \mathbf{01}^* + \mathbf{1}$.

The proof of Theorem 2.3 is in essence an algorithm for converting a regular expression to a finite automaton. However, the algorithm implicitly assumes that the regular expression is fully parenthesized. For regular expressions without redundant parentheses, we must determine whether the expression is of the form $p + q$, pq, or p^*. This is equivalent to parsing a string in a context-free language, and thus such an algorithm will be delayed until Chapter 5 where it can be done more elegantly.

Now we must show that every set accepted by a finite automaton is denoted by some regular expression. This result will complete the circle shown in Fig. 2.12.

Theorem 2.4 If L is accepted by a DFA, then L is denoted by a regular expression.

Proof Let L be the set accepted by the DFA

$$M = (\{q_1, \ldots, q_n\}, \Sigma, \delta, q_1, F).$$

Let R_{ij}^k denote the set of all strings x such that $\delta(q_i, x) = q_j$, and if $\delta(q_i, y) = q_\ell$, for any y that is a prefix (initial segment) of x, other than x or ϵ, then $\ell \leq k$. That is, R_{ij}^k is the set of all strings that take the finite automaton from state q_i to state q_j without going through any state numbered higher than k. Note that by "going through a state," we mean both entering and then leaving. Thus i or j may be greater than k. Since there is no state numbered greater than n, R_{ij}^n denotes all strings that take q_i to q_j. We can define R^k_{ij} recursively:

$$R^k_{ij} = R^{k-1}_{ik}(R^{k-1}_{kk})^* R^{k-1}_{kj} \cup R^{k-1}_{ij}, \tag{2.1}$$

$$R^0_{ij} = \begin{cases} \{a \mid \delta(q_i, a) = q_j\} & \text{if } i \neq j, \\ \{a \mid \delta(q_i, a) = q_j\} \cup \{\epsilon\} & \text{if } i = j. \end{cases}$$

Informally, the definition of R^k_{ij} above means that the inputs that cause M to go from q_i to q_j without passing through a state higher than q_k are either

1) in R^{k-1}_{ij} (that is, they never pass through a state as high as q_k); or
2) composed of a string in R^{k-1}_{ik} (which takes M to q_k for the first time) followed by zero or more strings in R^{k-1}_{kk} (which take M from q_k back to q_k without passing through q_k or a higher-numbered state) followed by a string in R^{k-1}_{kj} (which takes M from state q_k to q_j).

We must show that for each i, j, and k, there exists a regular expression r^k_{ij} denoting the language R^k_{ij}. We proceed by induction on k.

Basis $(k = 0)$. R^0_{ij} is a finite set of strings each of which is either ϵ or a single symbol. Thus r^0_{ij} can be written as $a_1 + a_2 + \cdots + a_p$ (or $a_1 + a_2 + \cdots + a_p + \epsilon$ if $i = j$), where $\{a_1, a_2, \ldots, a_p\}$ is the set of all symbols a such that $\delta(q_i, a) = q_j$. If there are no such a's, then $\varnothing$ (or ϵ in the case $i = j$) serves as r^0_{ij}.

Induction The recursive formula for R^k_{ij} given in (2.1) clearly involves only the regular expression operators: union, concatenation, and closure. By the induction hypothesis, for each ℓ and m there exists a regular expression $r^{k-1}_{\ell m}$ such that $L(r^{k-1}_{\ell m}) = R^{k-1}_{\ell m}$. Thus for r^k_{ij} we may select the regular expression

$$(r^{k-1}_{ik})(r^{k-1}_{kk})^*(r^{k-1}_{kj}) + r^{k-1}_{ij},$$

which completes the induction.

To finish the proof we have only to observe that

$$L(M) = \bigcup_{q_j \text{ in } F} R^n_{1j}$$

since R^n_{1j} denotes the labels of all paths from q_1 to q_j. Thus $L(M)$ is denoted by the regular expression

$$r^n_{1j_1} + r^n_{1j_2} + \cdots + r^n_{1j_p},$$

where $F = \{q_{j_1}, q_{j_2}, \ldots, q_{j_p}\}$. □

Example 2.13 Let M be the FA shown in Fig. 2.16. The values of r^k_{ij} for all i and j and for $k = 0$, 1, or 2 are tabulated in Fig. 2.17. Certain equivalences among regular expressions such as $(r + s)t = rt + st$ and $(\epsilon + r)^* = r^*$ have been used to simplify the expressions (see Exercise 2.16). For example, strictly speaking, the expression for r^1_{22} is given by

$$r^1_{22} = r^0_{21}(r^0_{11})^* r^0_{12} + r^0_{22} = \mathbf{0}(\epsilon)^*\mathbf{0} + \epsilon.$$

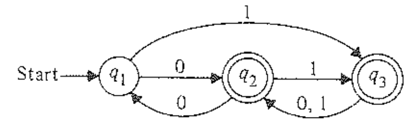


Fig. 2.16 FA for Example 2.13.

	$k = 0$	$k = 1$	$k = 2$
r^k_{11}	ϵ	ϵ	$(\mathbf{00})^*$
r^k_{12}	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}(\mathbf{00})^*$
r^k_{13}	$\mathbf{1}$	$\mathbf{1}$	$\mathbf{0}^*\mathbf{1}$
r^k_{21}	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}(\mathbf{00})^*$
r^k_{22}	ϵ	$\epsilon + \mathbf{00}$	$(\mathbf{00})^*$
r^k_{23}	$\mathbf{1}$	$\mathbf{1} + \mathbf{01}$	$\mathbf{0}^*\mathbf{1}$
r^k_{31}	$\varnothing$	$\varnothing$	$(\mathbf{0} + \mathbf{1})(\mathbf{00})^*\mathbf{0}$
r^k_{32}	$\mathbf{0} + \mathbf{1}$	$\mathbf{0} + \mathbf{1}$	$(\mathbf{0} + \mathbf{1})(\mathbf{00})^*$
r^k_{33}	ϵ	ϵ	$\epsilon + (\mathbf{0} + \mathbf{1})\mathbf{0}^*\mathbf{1}$

Fig. 2.17 Tabulation of r^k_{ij} for FA of Fig. 2.16.

Similarly,

$$r^2_{13} = r^1_{12}(r^1_{22})^* r^1_{23} + r^1_{13} = \mathbf{0}(\epsilon + \mathbf{00})^*(\mathbf{1} + \mathbf{01}) + \mathbf{1}.$$

Recognizing that $(\epsilon + \mathbf{00})^*$ is equivalent to $(\mathbf{00})^*$ and that $\mathbf{1} + \mathbf{01}$ is equivalent to $(\epsilon + \mathbf{0})\mathbf{1}$, we have

$$r^2_{13} = \mathbf{0}(\mathbf{00})^*(\epsilon + \mathbf{0})\mathbf{1} + \mathbf{1}.$$

Observe that $(\mathbf{00})^*(\epsilon + \mathbf{0})$ is equivalent to $\mathbf{0}^*$. Thus $\mathbf{0}(\mathbf{00})^*(\epsilon + \mathbf{0})\mathbf{1} + \mathbf{1}$ is equivalent to $\mathbf{00}^*\mathbf{1} + \mathbf{1}$ and hence to $\mathbf{0}^*\mathbf{1}$.

To complete the construction of the regular expression for M, which is $r^3_{12} + r^3_{13}$, we write

$$\begin{aligned} r^3_{12} &= r^2_{13}(r^2_{33})^* r^2_{32} + r^2_{12} \\ &= \mathbf{0}^*\mathbf{1}(\epsilon + (\mathbf{0} + \mathbf{1})\mathbf{0}^*\mathbf{1})^*(\mathbf{0} + \mathbf{1})(\mathbf{00})^* + \mathbf{0}(\mathbf{00})^* \\ &= \mathbf{0}^*\mathbf{1}((\mathbf{0} + \mathbf{1})\mathbf{0}^*\mathbf{1})^*(\mathbf{0} + \mathbf{1})(\mathbf{00})^* + \mathbf{0}(\mathbf{00})^* \end{aligned}$$

and

$$\begin{aligned} r^3_{13} &= r^2_{13}(r^2_{33})^* r^2_{33} + r^2_{13} \\ &= \mathbf{0}^*\mathbf{1}(\epsilon + (\mathbf{0} + \mathbf{1})\mathbf{0}^*\mathbf{1})^*(\epsilon + (\mathbf{0} + \mathbf{1})\mathbf{0}^*\mathbf{1}) + \mathbf{0}^*\mathbf{1} \\ &= \mathbf{0}^*\mathbf{1}((\mathbf{0} + \mathbf{1})\mathbf{0}^*\mathbf{1})^*. \end{aligned}$$

Hence

$$r^3_{12} + r^3_{13} = \mathbf{0}^*\mathbf{1}((\mathbf{0} + \mathbf{1})\mathbf{0}^*\mathbf{1})^*(\epsilon + (\mathbf{0} + \mathbf{1})(\mathbf{00})^*) + \mathbf{0}(\mathbf{00})^*.$$