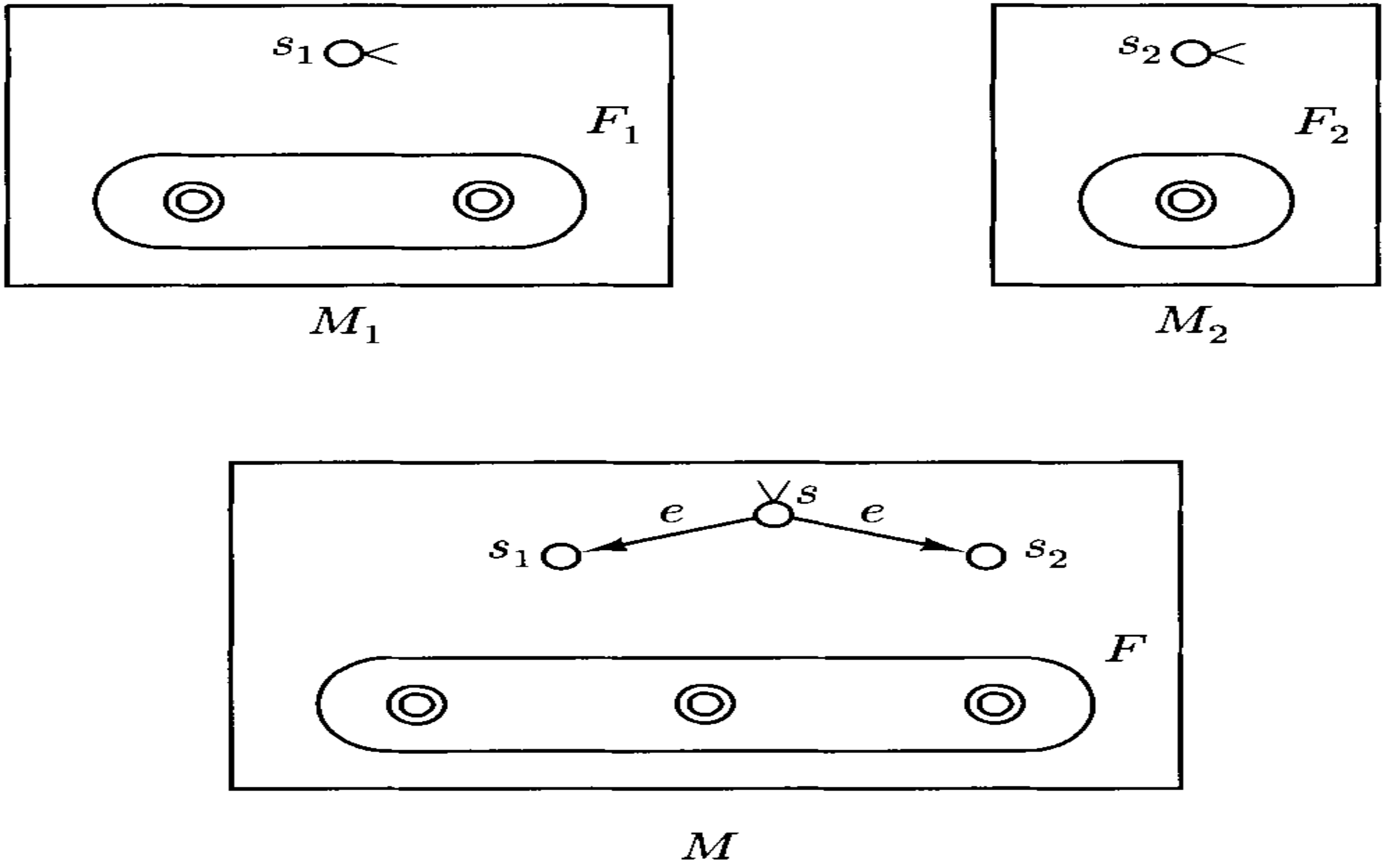
Theorem 2.3.1: *The class of languages accepted by finite automata is closed under*

(a) *union;*
(b) *concatenation;*
(c) *Kleene star;*
(d) *complementation;*
(e) *intersection.*

(a) Union

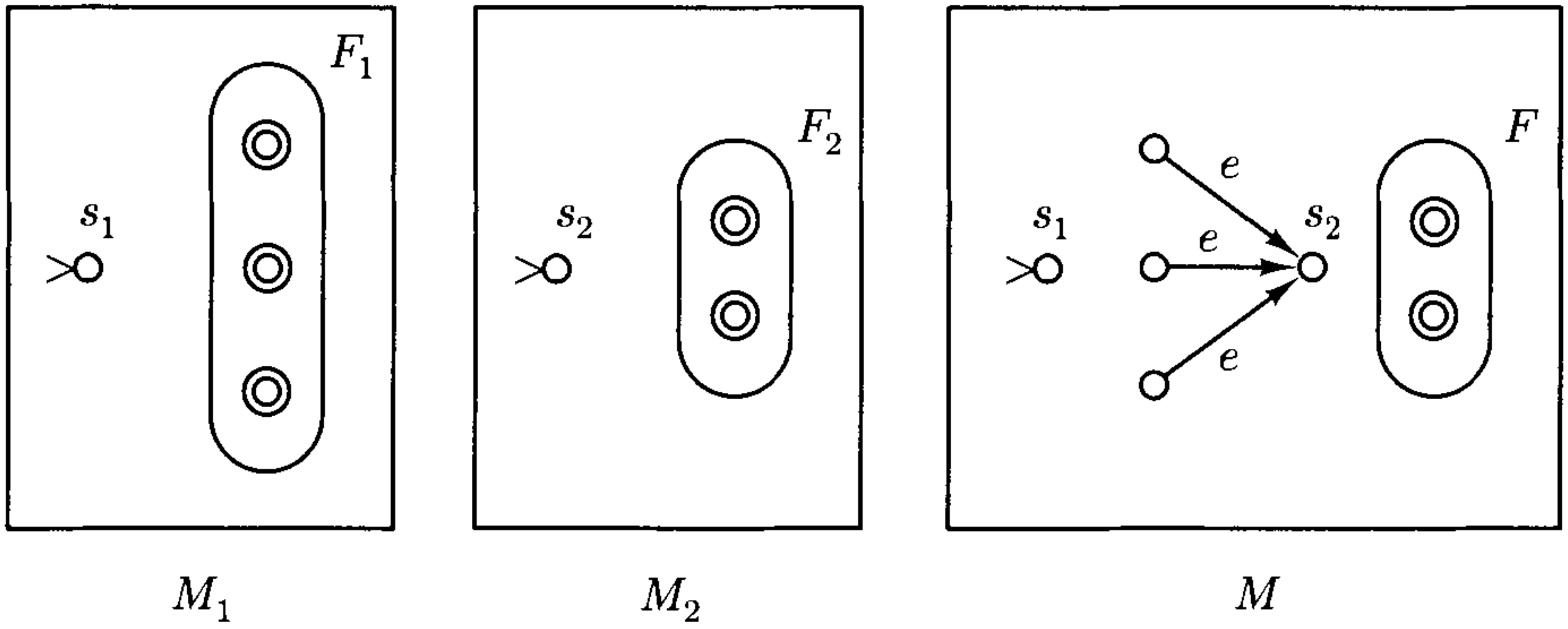


Formal Description of M_{union}

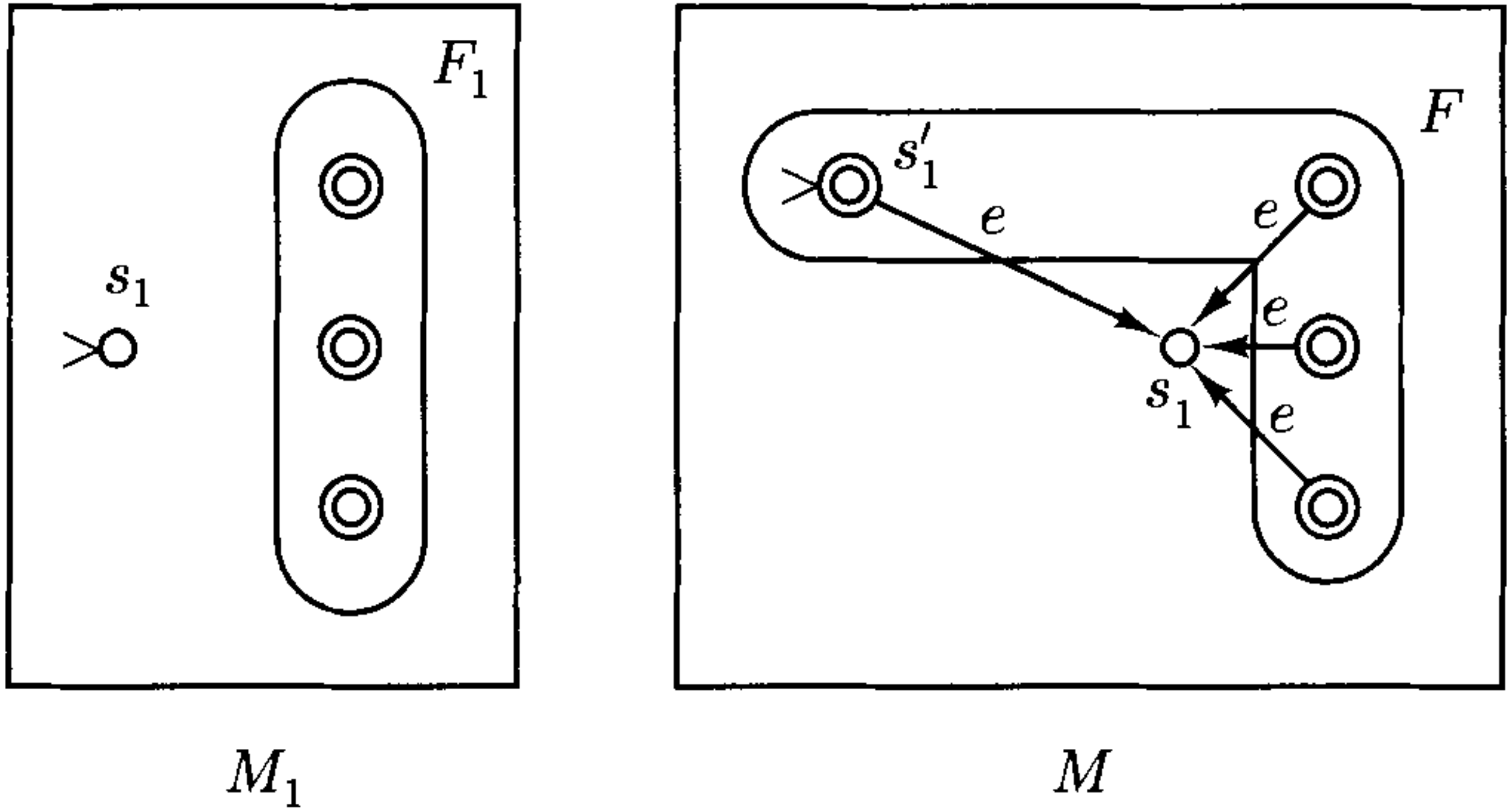
$M = (K, \Sigma, \Delta, s, F)$, where s is a new state not in K_1 or K_2,

$$K = K_1 \cup K_2 \cup \{s\},$$
$$F = F_1 \cup F_2,$$
$$\Delta = \Delta_1 \cup \Delta_2 \cup \{(s, e, s_1), (s, e, s_2)\}.$$

(b) Concatenation



(c) Kleene Star



(d) *Complementation.* Let $M = (K, \Sigma, \delta, s, F)$ be a *deterministic* finite automaton. Then the complementary language $\overline{L} = \Sigma^* - L(M)$ is accepted by the deterministic finite automaton $\overline{M} = (K, \Sigma, \delta, s, K - F)$. That is, $\overline{M}$ is identical to M except that final and nonfinal states are interchanged.

(e) *Intersection.* Just recall that

$$L_1 \cap L_2 = \Sigma^* - ((\Sigma^* - L_1) \cup (\Sigma^* - L_2)),$$

and so closedness under intersection follows from closedness under union and complementation ((a) and (d) above). ∎

Ex: *Design a FSA for L= (ab U aab)**

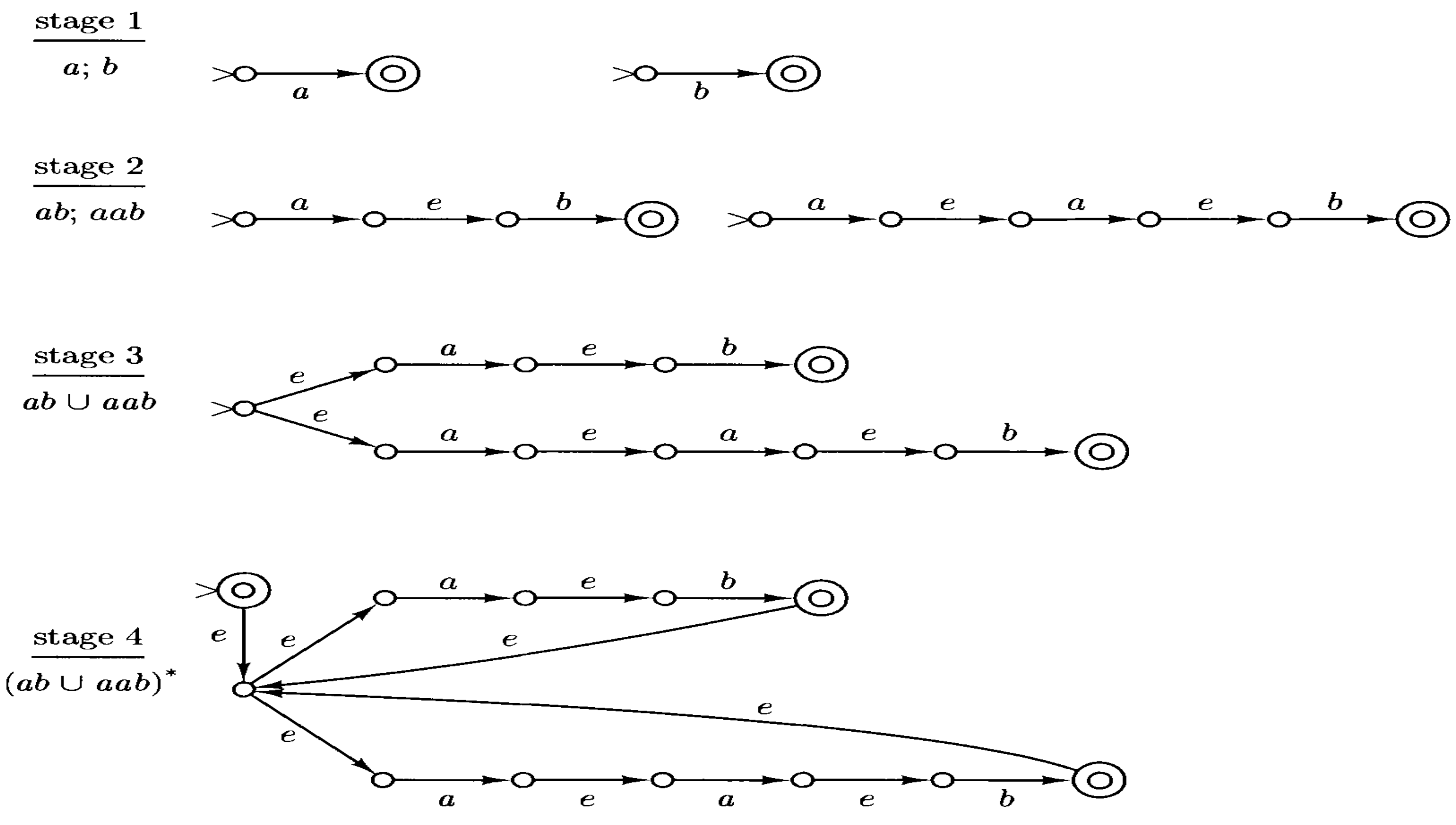


Figure 2-14

Theorem 2.3.2: *A language is regular if and only if it is accepted by a finite automaton.*

Proof: *Only if.* Recall that the class of regular languages is the smallest class of languages containing the empty set $\emptyset$ and the singletons a, where a is a symbol, and closed under union, concatenation, and Kleene star. It is evident (see Figure 2-14) that the empty set and all singletons are indeed accepted by finite automata; and by Theorem 2.3.1 the finite automaton languages are closed under union, concatenation, and Kleene star. Hence every regular language is accepted by some finite automaton.

If. Let $M = (K, \Sigma, \Delta, s, F)$ be a finite automaton (not necessarily deterministic). We shall construct a regular expression R such that $L(R) = L(M)$. We shall represent $L(M)$ as the union of many (but a finite number of) simple languages. Let $K = \{q_1, \ldots, q_n\}$ and $s = q_1$. For $i, j = 1, \ldots, n$ and $k = 0, \ldots, n$, we define $R(i, j, k)$ as the set of all strings in Σ^* that may drive M from state q_i to state q_j without passing through any *intermediate* state numbered $k + 1$ or greater —the endpoints q_i and q_j are allowed to be numbered higher than k. That is, $R(i, j, k)$ is the set of strings spelled by all paths from q_i to q_j *of rank* k (recall the similar maneuver in the computation of the reflexive-transitive closure of a relation in Section 1.6, in which we again considered paths with progressively higher and higher-numbered intermediate nodes). When $k = n$, it follows that

$$R(i, j, n) = \{w \in \Sigma^* : (q_i, w) \vdash_M^* (q_j, e)\}.$$

Therefore

$$L(M) = \bigcup\{R(1, j, n) : q_j \in F\}.$$

The crucial point is that *all of these sets* $R(i, j, k)$ *are regular*, and hence so is $L(M)$.

The proof that each $R(i, j, k)$ is regular is by induction on k. For $k = 0$, $R(i, j, 0)$ is either $\{a \in \Sigma \cup \{e\} : (q_i, a, q_j) \in \Delta\}$ if $i \neq j$, or it is $\{e\} \cup \{a \in \Sigma \cup \{e\} : (q_i, a, q_j) \in \Delta\}$ if $i = j$. Each of these sets is finite and therefore regular.

For the induction step, suppose that $R(i, j, k-1)$ for all i, j have been defined as regular languages for all i, j. Then each set $R(i, j, k)$ can be defined combining previously defined regular languages by the regular operations of union, Kleene star, and concatenation, as follows:

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

This equation states that to get from q_i to q_j without passing through a state numbered greater than k, M may either

(1) go from q_i to q_j without passing through a state numbered greater than $k - 1$; or
(2) go (a) from q_i to q_k; then (b) from q_k to q_k zero or more times; then (c) from q_k to q_j; in each case without passing through any intermediate states numbered greater than $k - 1$.

Therefore language $R(i, j, k)$ is regular for all i, j, k, thus completing the induction. ■

Carrying out explicitly the construction of the proof of the *if* part can be very tedious (in this simple case, thirty-six regular expressions would have to be constructed!). Things are simplified considerably if we assume that the nondeterministic automaton M has two simple properties:

(a) It has a single final state, $F = \{f\}$.

(b) Furthermore, if $(q, u, p) \in \Delta$, then $q \neq f$ and $p \neq s$; that is, there are no transitions into the initial state, nor out of the final state.

This "special form" is not a loss of generality, because we can add to any automaton M a new initial state s and a new final state f, together with e-transitions from s to the initial state of M and from all final states of M to f (see Figure 2-16(a), where the automaton of Figure 2-15 is brought into this "special form"). Number now the states of the automaton $q_1, q_2, \ldots, q_n$, as required by the construction, so that $s = q_{n-1}$ and $f = q_n$. Obviously, the regular expression sought is $R(n-1, n, n)$.

We shall compute first the $R(i,j,0)$'s, from them the $R(i,j,1)$'s, and so on, as suggested by the proof. At each stage we depict each $R(i,j,k)$'s as a label on an arrow going from state q_i to state q_j. We omit arrows labeled by $\emptyset$, and self-loops labeled $\{e\}$. With this convention, the initial automaton depicts the correct values of the $R(i,j,0)$'s —see Figure 2-16(a). (This is so because in our initial automaton it so happens that, for each pair of states (q_i, q_j) there is at most one transition of the form (q_i, u, q_j) in Δ. In another automaton we might have to combine by union all transitions from q_i to q_j, as suggested by the proof.)

Now we compute the $R(i,j,1)$'s; they are shown in Figure 2-16(b). Notice immediately that state q_1 need not be considered in the rest of the construction; all strings that lead M to acceptance passing through state q_1 have been considered and taken into account in the $R(i,j,1)$'s. We can say that state q_1 has been *eliminated*. In some sense, we have transformed the finite automaton of Figure 2-16(a) to an equivalent *generalized finite automaton*, with transitions that may be labeled not only by symbols in Σ or e, but by entire *regular expressions*. The resulting generalized finite automaton has one less state than the original one, since q_1 has been eliminated.

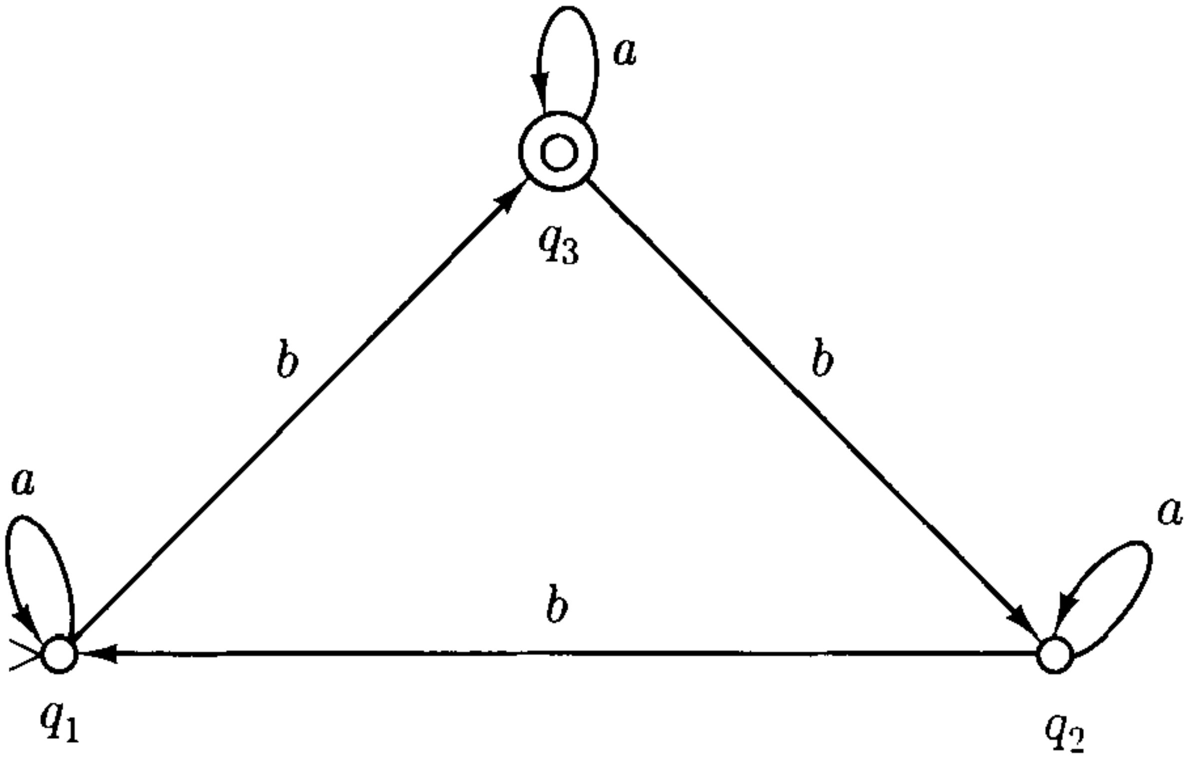


Figure 2-15

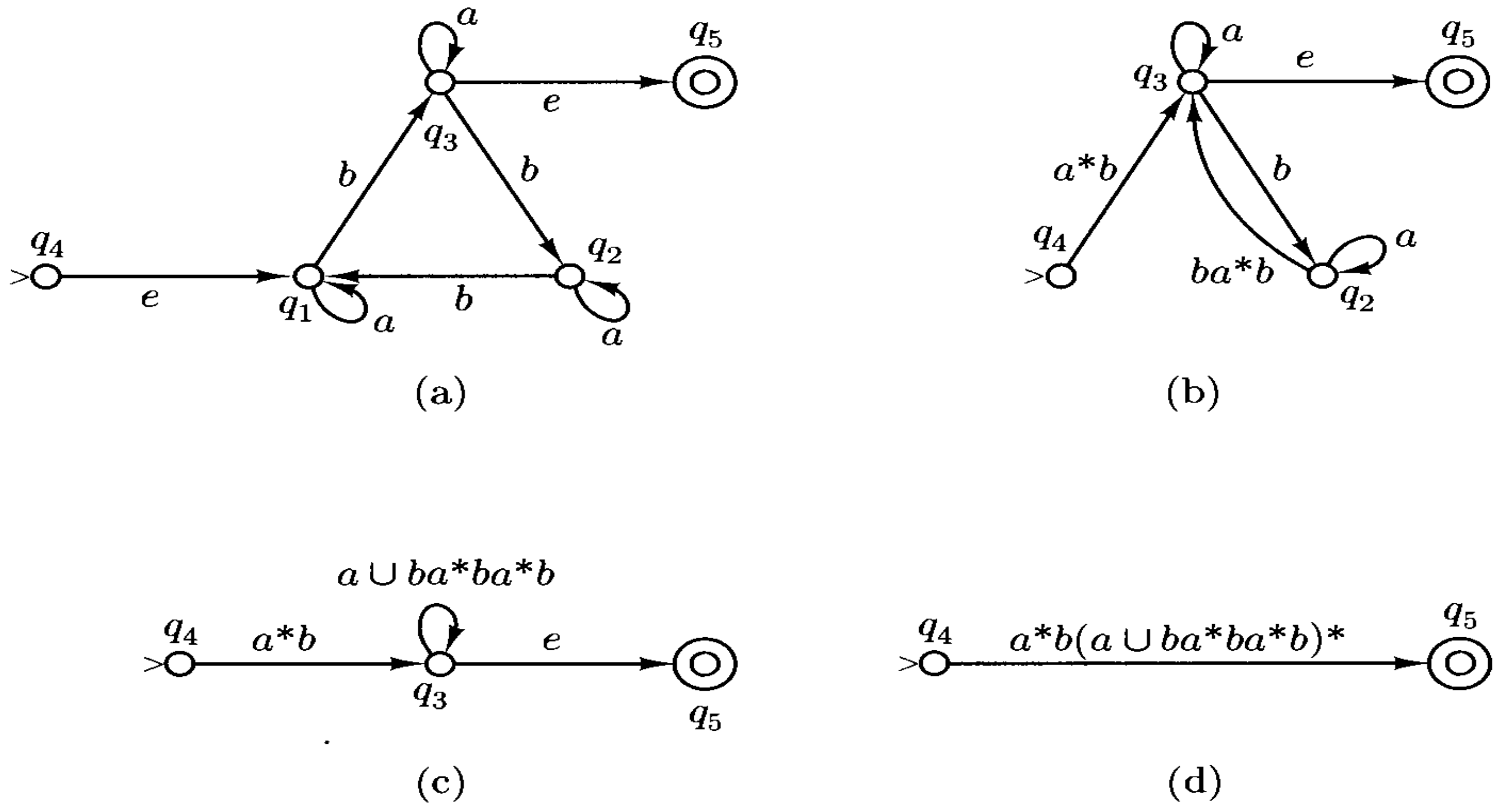


Figure 2-16

$$R = R(4,5,5) = R(4,5,3) = a^*b(ba^*ba^*b \cup a)^*$$

Let us examine carefully what is involved in general in eliminating a state q (see Figure 2-17). For each pair of states $q_i \neq q$ and $q_j \neq q$, such that there is an arrow labeled α from q_i to q and an arrow labeled β from q to q_j, we add an arrow from q_i to q_j labeled $\alpha\gamma^*\beta$, where γ is the label of the arrow from q to itself (if there is no such arrow, then $\gamma = \emptyset$, and thus $\gamma^* = \{e\}$, so the label becomes $\alpha\beta$). If there was already an arrow from q_i to q_j labeled δ, then the new arrow is labeled $\delta \cup \alpha\gamma^*\beta$.

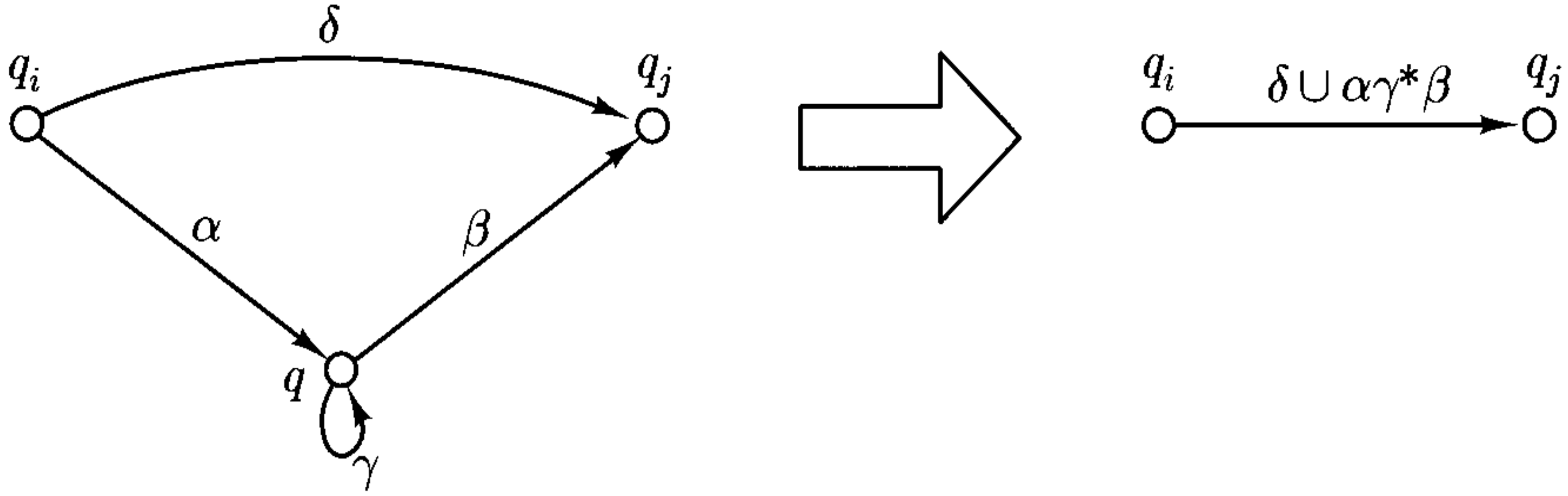


Figure 2-17