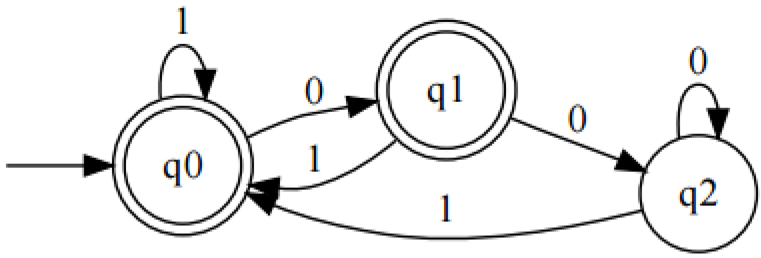
1.

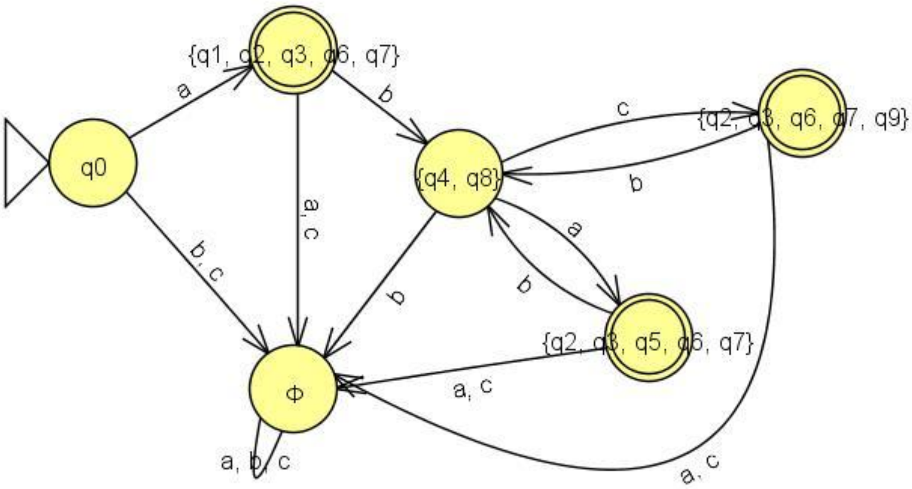
	a	b	ε
Q0	{Q1}	{Q2}	{Q1}
Q1	φ	{Q1, Q3}	φ
Q2	φ	φ	{Q3}
Q3	{Q0, Q3}	{Q2}	φ

2.

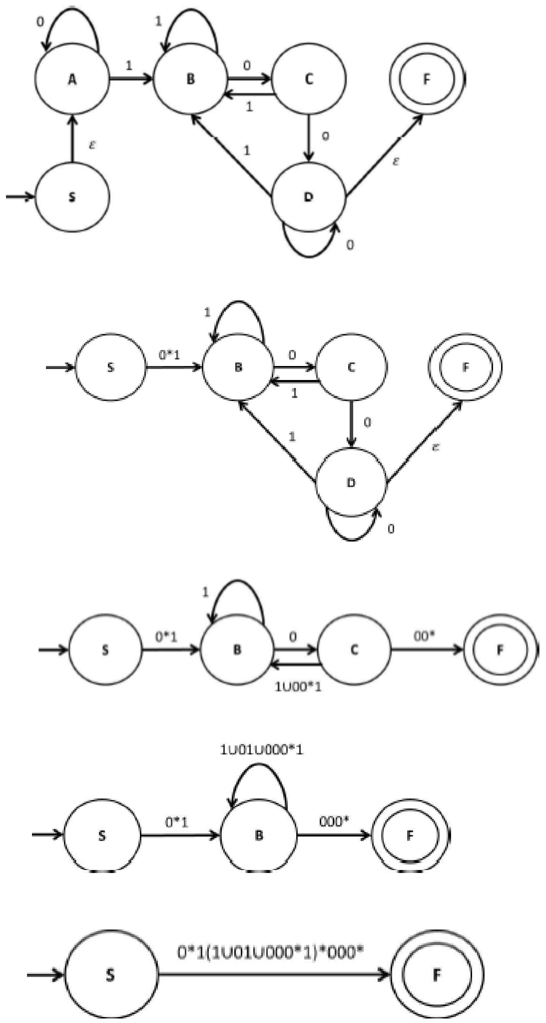
ε + 0 + 1 + (0 + 1)* (01 + 10 + 11)



3.

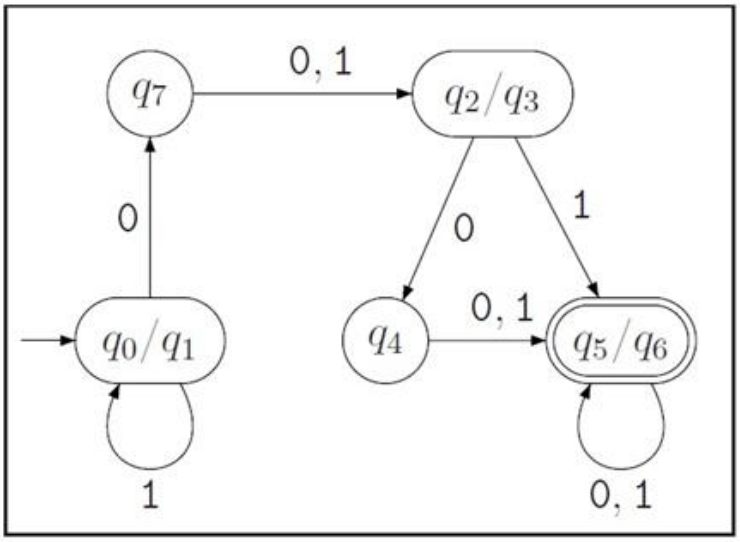


4.



A
B
C
D
F
S
ε
0*1
00*
1∪00*1
1∪01∪000*1
000*
0*1(1∪01∪000*1)*000*

5.



q_7
0, 1
q_2/q_3
0
1
q_0/q_1
q_4
q_5/q_6

6.

By contradiction.

Assume language L is regular, and p is pumping length,

and give a string s, $s \in L$. By pumping lemma, s can be divided to xyz, and s is satisfied for each $i \geqq 0$, $xy^iz \in L$, $|y|>0$, and $|xy| \leqq p$.

Let string $s=a^pb^pc^{2p} \in L$, and by definition of pumping lemma, $|y|>0$, and $|xy| \leqq p$.

Thus, y can only contain one kind alphabet, 'a'.

Once we pump i to 0, the string $xz = a^{p-|y|}b^pc^{2p} \notin L$ because $p-|y|+p \neq 2p$.

There is a contradiction.

7.

a)

Let w’’ be the longest string among $w \in L(M)$ where $n \le |w| < 2n$.

Since $|w''| \ge n$, by pumping lemma, w’’ can be divided into three pieces xyz where $|y|>0$, $|xy| \le n$ and $xy^iz \in L(M)$ for i=0,1,…

Let $w'=xy^2z$.

Since $|y|>0$, we have $|w'|>|w''|$. If $|w'|<2n$, we will obtain a contradiction that, by assumption, w’’ must be the longest string among all $w \in L(M)$ where $n \le |w| < 2n$. Thus, we have $|w'| \ge 2n$.

b)

Let w’’ be the shortest string among $w \in L(M)$ where $|w| \ge 2n$.

Since $|w''| \ge 2n$, by pumping lemma, w’’ can be divided into three pieces xyz where $|y|>0$, $|xy| \le n$ and $xy^iz \in L(M)$ for i=0,1,…

Let w’=xz.

Since $|y|>0$, we have $|w'|<|w|$. If $|w'| \ge 2n$, we will obtain a contradiction that, by assumption, w’’ must be the shortest string among all $w \in L(M)$ where $|w| \ge 2n$. Thus, we have $|w'|<2n$.

If $|w'|<n$, we have $|y|=|w''|-|w'|>n$ and we will obtain a contradiction that $|xy| \le n$.

Thus, we have $|w'| \ge n$.

In summary, we have $n \le |w'| < 2n$.

8.

Let $D_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ and $D_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ be two DFAs that recognize A and B, respectively. Here, we shall construct a DFA $D = (Q, \Sigma, \delta, q, F)$ that recognizes the perfect shuffle of A and B.

The key idea is to design D to alternately switch from running D_A and running D_B after each character is read. Therefore, at any time, D needs to keep track of (i) the current states of D_A and D_B and (ii) whether the next character of the input string should be matched in D_A or in D_B. Then, when a character is read, depending on which DFA should match the character, D makes a move in the corresponding DFA accordingly. After the whole string is processed, if both DFAs are in the accept states, the input string is accepted; otherwise, the input string is rejected.

Formally, the DFA D can be defined as follows:

(a) $Q = Q_A \times Q_B \times \{A, B\}$, which keeps track of all possible current states of D_A and D_B, and which DFA to match.

(b) $q = (q_A, q_B, A)$, which states that D starts with D_A in q_A, D_B in q_B, and the next character read should be in D_A.

(c) $F = F_A \times F_B \times \{A\}$, which states that D accepts the string if both D_A and D_B are in accept states, and the next character read should be in D_A (i.e., last character was read in D_B).

(d) δ is as follows:

  i. $\delta((x, y, A), a) = (\delta_A(x, a), y, B)$, which states that if current state of D_A is x, the current state of D_B is y, and the next character read is in D_A, then when a is read as the next character, we should change the current state of A to $\delta_A(x, a)$, while the current state of B is not changed, and the next character read will be in D_B.

  ii. Similarly, $\delta((x, y, B), b) = (x, \delta_B(y, b), A)$.