Assignment 2

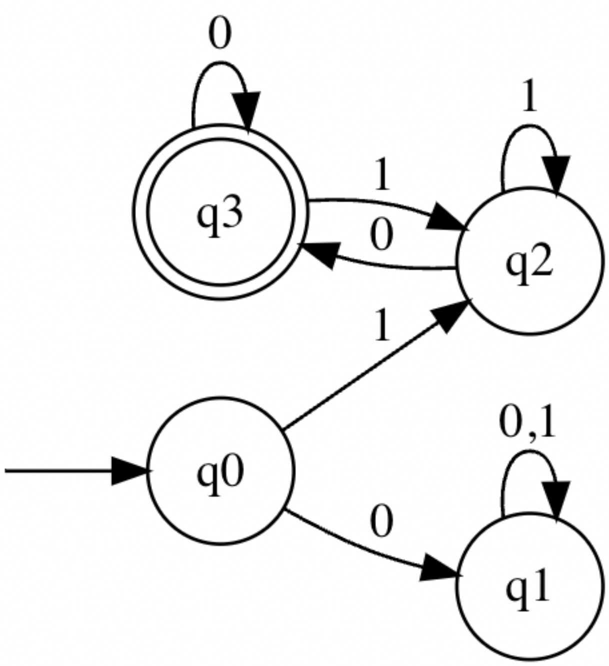
Nishan Dhoj Karki

09/05/2022

1 Exercise 1.6

a. $w|w$ begins with a 1 and ends with a 0

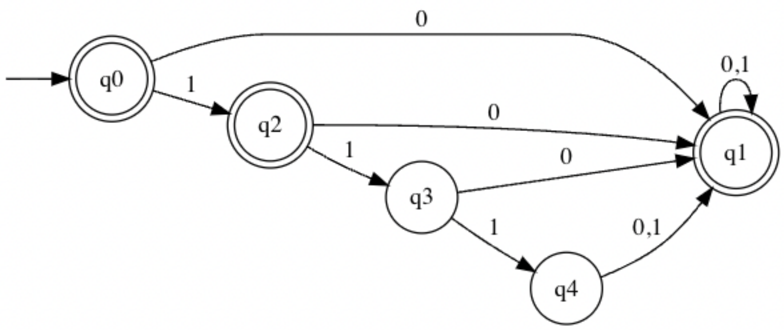
Answer :



1.6 a

h. $w|w$ **is any string except** 11 **and** 111

Answer :

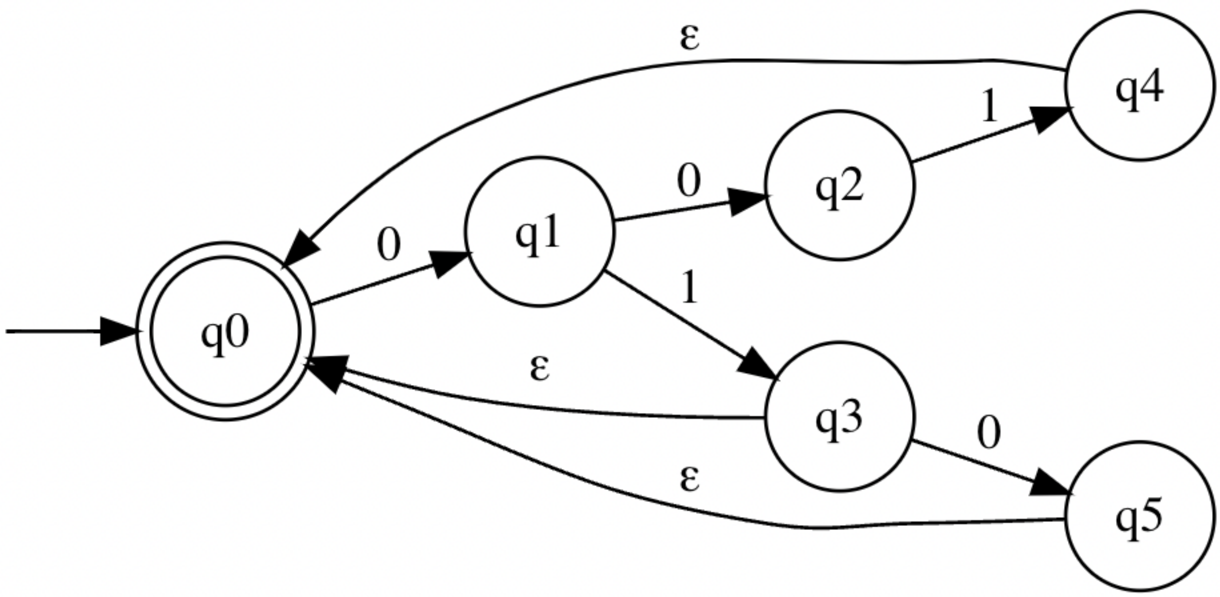


1.6 h

2 Exercise 1.17

a. Give an NFA recognizing the language $(01 \bigcup 001 \bigcup 010)^*$**.**

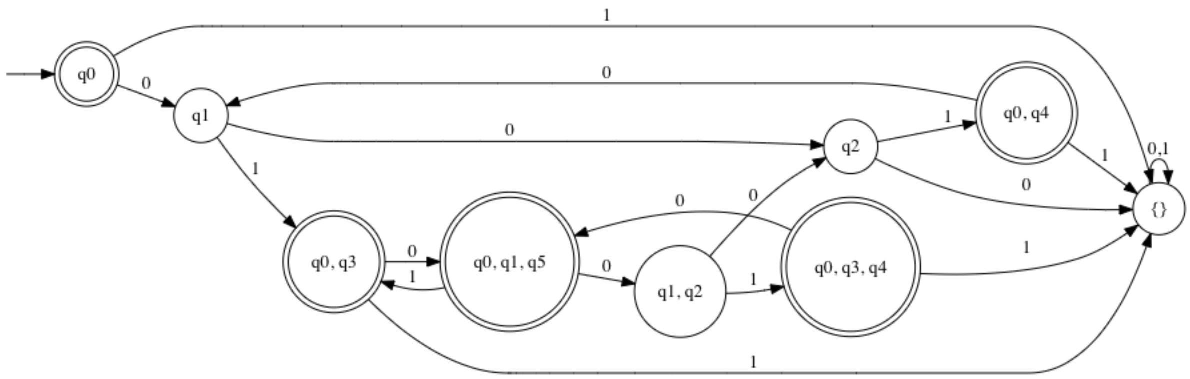
Answer :



1.17 a

b. Convert this NFA to an equivalent DFA. Give only the portion of the DFA that is reachable from the start state.

Answer :



1.17 b

3 Problem 0.13 Show that every graph with two or more nodes contains two nodes that have equal degrees.

Theorem: Every graph with two or more nodes contains two nodes that have equal degree.

Proof sketch:
We know that a node in a graph G with n nodes can have the max degree of $n-1$. For a node to have $n-1$ degree it must be connected to every other node in the graph. If this is the case, then none of the nodes have a degree of 0. Then, there have to be two nodes with the same degree.

Proof by contradiction :
Assumption : Let us assume a graph G with n nodes($n \geq 2$) where all nodes in the graph G have unique degree.

We know that a node in a graph G with n nodes can have the max degree of $n-1$.
For two or more nodes in a graph to not have equal degrees all the nodes must have a unique degree. If the degree of each node of graph G is unique, then the unique degrees must be exactly, {0, 1, 2, ..., n-1}.
Using the pigeonhole principle, we can deduct that it is not possible to have a node of a degree of 0 (connected to no other node) and a node of degree $n-1$, (connected to every other node) simultaneously.

Thus, it is impossible to have a graph with n nodes where one node has a degree of 0 and another has a degree of $n-1$. Hence, we have reached a contradiction and shown that all nodes in a graph cannot have a unique degree, thus two or more nodes must have an equal degree.

4 Problem 1.31

Theorem: For any string $w = w_1w_2 \ldots w_n$, the reverse of w, written as w^R, is the string w in reverse order, $w_n \ldots w_2w_1$. For any language A, let $A^R = \{w^R | w \in A\}$. Show that if A is regular, so is A^R.

Proof sketch:
A language is regular if a DFA or a NFA recognizes it. We approach the proof with an idea to define a DFA or NFA that satisfies A^R. We do this by building a DFA or a NFA that identifies A and reversing the transitions.

Proof by construction:
If A is regular then by the definition of regular languages, there is a DFA and a NFA which recognizes A.

Let us assume, $M = (Q, \sum, \delta, q_0, q_{accept})$ is a DFA that recognizes A. Thus $L(M) = A$.

We know that every DFA has an equivalent NFA.
Constructing a new NFA M^R whose accept state is the start state of M with epsilon closer to all accepting states of M, and all transitions of M are reversed.
$M^R = (Q^R, \sum, \delta^R, q_b, q_0)$
Here,
Q^R is the states,
$Q^R = Q \bigcup q_b$

δ^R is the new transition function such that,
If for M, $\delta : Q \times \sum \rightarrow Q'$ then for $M^R \delta^R : Q' \times \sum \rightarrow Q$. Here, Q and Q' are the states and $\sum$ is the set of alphabets.

q_b is the new start state.
q_0 is the new accept state for M^R. All accept states of the DFA M has epsilon closure to the accept state of M^R. Such that, $q_{accept} \times \epsilon \rightarrow q_0$.

Hence, $L(M^R) = A^R$.
We have defined a NFA that recognizes A^R. Since, A is regular and we have constructed a DFA for A and an NFA for A^R, A^R is regular.