Gradiance Online Accelerated Learning

Zayd

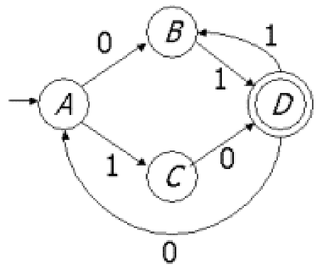
- Home Page
- Assignments Due
- Progress Report
- Handouts
- Tutorials
- Homeworks
- Lab Projects
- Log Out

Help

Submission number:	60386
Submission certificate:	AH183242
Submission time:	2014-02-10 18:55:30 PST (GMT - 8:00)

Number of questions:	5
Positive points per question:	4.0
Negative points per question:	0.0
Your score:	20

1. Examine the following DFA:


$$\begin{tikzcd}
 & B \arrow[dr, "1"] & \\
\arrow[r] A \arrow[ur, "0"] \arrow[dr, "1"'] & & D \arrow[ul, bend right, "1"'] \arrow[ll, bend left=60, "0"] \\
 & C \arrow[ur, "0"'] &
\end{tikzcd}$$

Identify in the list below the string that this automaton accepts.

a) 010011

b) 0110

c) 1011011

d) 1011

Answer submitted: **d)**

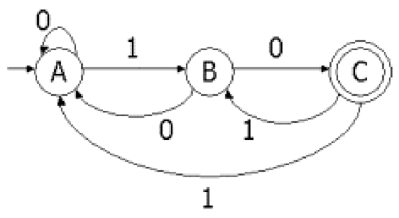
You have answered the question correctly.

Question Explanation:

To test whether a string is accepted by a deterministic finite automaton, we start at the start state, and process each symbol of the string, in order from the beginning. Each symbol causes a state transition, from the state *q* we are in to the state at the head of the arrow whose tail is at *q* and whose label is the symbol in question. If there is no such arrow, we assume the automaton goes to a dead state, from which it can never accept.

If this simulation leads to an accepting state, then the string is accepted; otherwise it is not. In our example, A is the start state, and the only accepting state is D. For example, the input string 0111010 takes us through the sequence of states A,B,D,B,D,A,C,D. Since we wind up in accepting state D, 0111010 is accepted.

2. The following nondeterministic finite automaton:



accepts which of the following strings?

a) 1011101

b) 00110100

c) 0010010

d) 1110100

Answer submitted: **c)**

You have answered the question correctly.

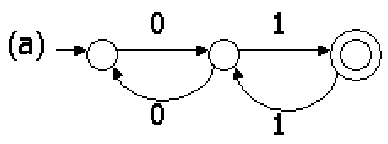
Question Explanation:

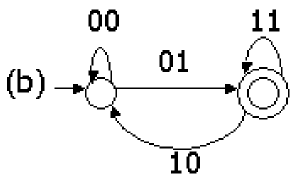
The easiest way to solve this question is to convert the NFA to an equivalent DFA. Here is the transition table for the DFA:

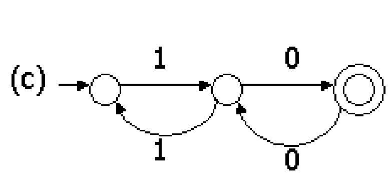
	0	**1**
{A}	{A}	{B}
{B}	{A,C}	φ
{A,C}	{A}	{A,B}
φ	φ	φ
{A,B}	{A,C}	{B}

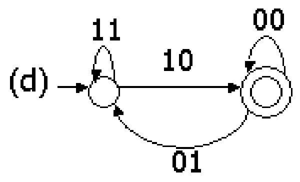
Notice that only {A,C} contains an accepting state of the NFA, so only that DFA state is accepting. We therefore can identify input strings accepted by the NFA by seeing whether the DFA goes from its start state {A} to {A,C} on that input.

3. Which automata define the same language?









Note: (b) and (d) use transitions on strings. You may assume that there are nonaccepting intermediate states, not shown, that are in the middle of these transitions, or just accept the extension to the conventional finite automaton that allows strings on transitions and, like the conventional FA accepts strings that are the concatenation of labels along any path from the start state to an accepting state.

a) a and b

b) a and c

c) a and d

d) b and c

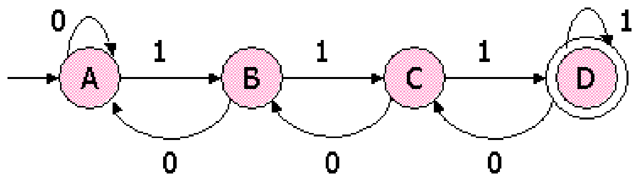
Answer submitted: **a)**

You have answered the question correctly.

Question Explanation:

Clearly (a) and (b) accept only strings that end in 1, and (c) and (d) accept only strings that end in 0. Thus, there are no possible equivalences except possibly a-b and c-d. But notice that (b) is constructed from (a) by eliminating the middle state, and (d) is constructed in the same way from (c). Thus, (a) and (b) are equivalent and so are (c) and (d).

4. Examine the following DFA:



This DFA accepts a certain language L. In this problem we shall consider certain other languages that are defined by their tails, that is, languages of the form (0+1)*w, for some particular string *w* of 0's and 1's. Call this language L(w). Depending on *w*, L(w) may be contained in L, disjoint from L, or neither contained nor disjoint from L (i.e., some strings of the form *xw* are in L and others are not).

Your problem is to find a way to classify *w* into one of these three cases. Then, use your knowledge to classify the following languages:

1. L(1111001), i.e., the language of regular expression (0+1)*1111001.
2. L(11011), i.e., the language of regular expression (0+1)*11011.
3. L(110101), i.e., the language of regular expression (0+1)*110101.
4. L(00011101), i.e., the language of regular expression (0+1)*00011101.

a) L(00011101) is neither disjoint from L nor contained in L.

b) L(00011101) is disjoint from L.

c) L(11011) is disjoint from L.

d) L(11011) is contained in L.

Answer submitted: **d)**

You have answered the question correctly.

Question Explanation:

It helps to imagine two "explorers" walking the graph of the automaton in response to the same input string. If explorer 1 starts out in state A and explorer 2 starts out in state B, then after reading a sequence of 0's and 1's, either the two explorers wind up in the same state, or explorer 1 is one state to the left of explorer 2. The proof is a simple induction on the length of the input string.

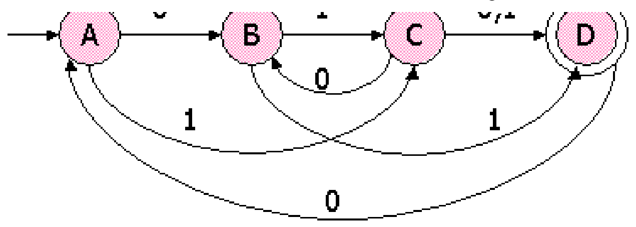
Similarly, if the explorers start out in states B and C we can draw the same conclusion --- either they are in the same state or the explorer that started out in B is one state to the left of the other explorer. And if the two explorers start out in C and D, the conclusion is the same. Thus, if input *w* takes you from state A to state D, then *w* also takes B, C, and D to D. we conclude that if *w* takes A to D, then L(w) is contained in L; i.e., any string *xw* takes A to D, regardless of where *x* takes you.

On the other hand, if *w* does not take A to D, then there is at least one string in L(w), namely *w* itself, that is in L(w) but not in L. In those cases, we have to decide whether or not L(w) and L are disjoint. By our arguments about the two explorers, we can also observe that if *w* does not take D to D, then *w* cannot take any state to D. In that case, L(w) and L are disjoint. However, if *w* takes D but not A to D, then L(w) is neither disjoint from L nor contained in L.

Here are the conclusions about the four strings in question:

- 1111001 takes neither A nor D to D. L(1111001) is disjoint from L.
- 11011 takes both A and D to D. L(11011) is contained in L.
- 110101 takes D to A but A to C. Thus, L(110101) is neither contained in L nor disjoint from L.
- 00011101 takes both A and D to D. L(00011101) is contained in L.

5. Here is a nondeterministic finite automaton:



Some input strings lead to more than one state. Find, in the list below, a string that leads from the start state A to three different states (possibly including A).

a) 1000

b) 01010

c) 10010

d) 11010

Answer submitted: **b)**

You have answered the question correctly.

Question Explanation:

One way to approach this problem is to construct the equivalent DFA. Here is its transition table:

	0	**1**
→{A}	{A,B}	{C}
{A,B}	{A,B}	{C,D}
{C}	{B,D}	{D}
*{C,D}	{A,B,D}	{D}
*{B,D}	{A}	{C,D}
*{D}	{A}	{D}
*{A,B,D}	{A,B}	{C,D}

Notice that there is only one DFA state accessible from {A} that contains three states: {A,B,D}. Thus, the problem can be solved by simulating this DFA on each of the strings in the list of choices. The one that gets to state {A,B,D} is the correct choice.