1 Week 9 revision

1. [Source: Sipser 1.16] Convert the following nondeterministic FSM (NFSM) to an equivalent deterministic finite automata (DFSM).

\[\text{Solution:}\]

1. Convert the following nondeterministic FSM (NFSM) to an equivalent deterministic finite automata (DFSM).
2. [Source: Sipser 1.16] Convert the following nondeterministic FSM (NFSM) to an equivalent deterministic finite automata (DFSM).

![NFSM Diagram]

**Solution:**

How does this work?
When considering the starting state, note that since 1 can reach 1 or 2 by epsilon only, the initial state is given as the set state 1,2 (we lose the original start state 1). Neither 2 or 3 have any epsilon transitions leaving them. Technically this is called epsilon closure of the states. Here is the state transition table starting from the starting state 1,2.

**Solution:** (solution, cont’d)
1,2 a 1,2,3
1,2 b ∅ neither 1 or 2 have a transition for
b in the NFSM
1,2,3 a 1,2,3 1a3, 3a2, 2a1, 2ae2 in the NFSM
1,2,3 b 2,3 1 and 2 have no b transition but
3b2 and 3b3 in the NFSM
2,3 a 1,2 2a1, 2ae2, 3a2
2,3 b 2,3 2 has no b, but 3b3 and 3b2
∅ a,b ∅ This is the non-accept state with
no way out

Note that any set of states with a 2 in it is an accepting state of
the DFSM because 2 is accepting in the NFSM. So 1,2,3 and
2,3 are all accepting states.

2 Regular expressions and languages

1. Give a regular expression for the following sets:
   a) all strings of 0s and 1s beginning with 1 and ending with 1
   b) strings containing exactly two 1s
   c) strings of 0s and 1s having an odd number of 1s
   d) the set of all strings of 0s and 1s containing at least one 0
   e) the set of all strings of 0s and 1s where each 0 is followed by two 1s
   f) the set of all strings of 0s and 1s containing exactly three 0s

Solution:

(a) 1(0 + 1)*1 + 1
    That is, a 1 followed by any number of 0s or 1s, finishing with
    a 1, or just a single 1
(b) 0*10*10*
    That is, two 0s with 0,1 or more 1s before, after and between
    them.
(c) 0*1(0*10*1)*0*
That is, at least 1 occurrence of 1 and then sequences with 2 1s repeated, with 0 or more 0s in between any of these.

(d) \((0 + 1)^*0(0 + 1)^*\)
That is, any binary string, a 0 and then any binary string.

(e) \((011 + 1)^*\) or \((1^*(011)^*1^*)^*\)
Allow for multiple occurrences of 011 which may be separated by any length sequences of 0 or more 1s. For example test this string: 111 011 1 011 011 111 (the spaces are just for readability - not part of the string). remember that the \(^*\) operator means 0 or more repetitions. So \(1^*(011)\) can be 011 or 1011 or 1111111011 etc. Also, make sure that sequences with no 0s are accepted too.

(f) \(1^*01^*01^*01^*\)
That is three 1s with 0,1 or more 0s before, after and between them.

Other answers are possible.

2. Find a regular expression for the language \(L\) consisting of all strings over \(\{0, 1\}\) with no consecutive zeros (that is, any string containing 00 is \(\text{not}\) in the language).

\[
\text{Solution:} \\
1^*(011^*)^*(\epsilon + 0)
\]

3. Does the string 01110111 belong to the regular set \((1^*01)^*(11 + 0^*)\) ?
Justify your answer.

\[
\text{Solution:} \\
\text{Yes. The first part 011101 matches } (1^*01)^* \text{ by looping twice. The second part 11 matches } (11 + 0^*) \text{ by selecting the first branch.}
\]

4. Does the string 011100101 belong to the regular set \(01^*0^*(11^*0)^*\) ?
Justify your answer.
SOLUTION:
No. 01*10* matches the first part 011100 but the remaining 101 does not match (11*0)* which must either be empty or contain two 1s.

5. Prove that if $A$ is a regular set with alphabet $I$, then the language defined by taking the set difference $I^* - A$ is also regular.

SOLUTION:
$I^* - A$ is the set of all possible words from the alphabet except those strings in $A$.
Suppose we have a DFSM $M$ that recognises $A$. So it has accepting states for every string in $A$.
Now create an FSM $P$ with the same structure as $M$ but with the accepting and non accepting states - That is, $F_P = Q - F_M$.
The new FSM $P$ will recognise $I^* - A$. Therefore, since we have created a FSM to recognise the language $I^* - A$, that language is regular. QED

6. Simplify the following regular expression as much as possible
$(((a^*)^*)(\epsilon + b)c(c + (\epsilon + \epsilon)))^*$

Explain your reasons for each simplification step.

SOLUTION:
$(((a^*)^*)^*$ is $a^{****}$ which is the same as $a^*$ (that is, $a$ repeated 0 or more times).
$(\epsilon + \epsilon)$ is the same as $\epsilon$: that is, the empty symbol.
So the reg exp simplifies to $a^*(\epsilon + b)c(c + \epsilon)^*$
And $(c+\epsilon)^*$ is the same as $c^*$, so it simplifies further to: $a^*(\epsilon+b)cc^*$.
Note that $cc^*$ generates 1 or more cs while $c^*$ generates 0 or more, so you can’t simplify the two cs any more.
Finally we have the simplified expression:
$a^*(b + \epsilon)cc^*$