CSE 105:Introduction to the Theory of Computation, Fall 1999

Homework 1

For many of the problems there are multiple possible solutions. The answers presented below outline one such solution for each problem. In some of the diagrams, E, represents an empty transition.

PROBLEM 1 a) See figure 1.

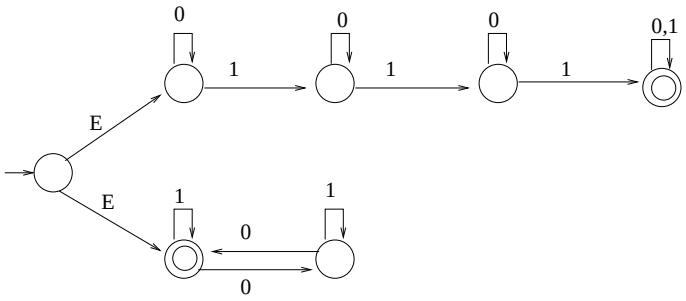

Fig. 1. Solution for 1 a.

PROBLEM 1 b) See figure 2.

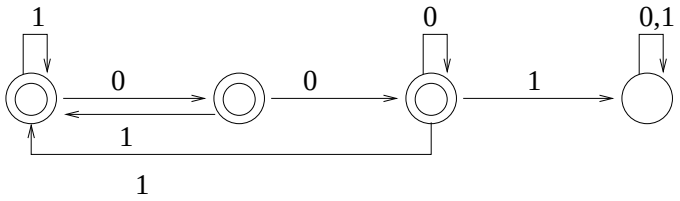

Fig. 2. Solution for 1 b.

PROBLEM 1 c) See figure 3.

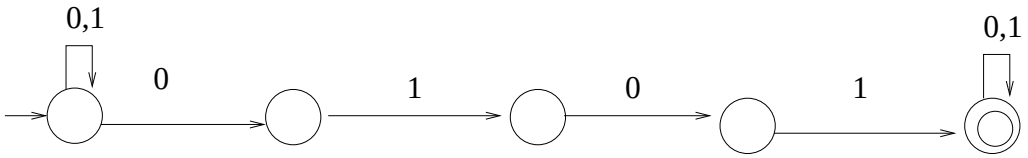

Fig. 3. Solution for 1 c.

PROBLEM 2 (1.6a) See figure 4.

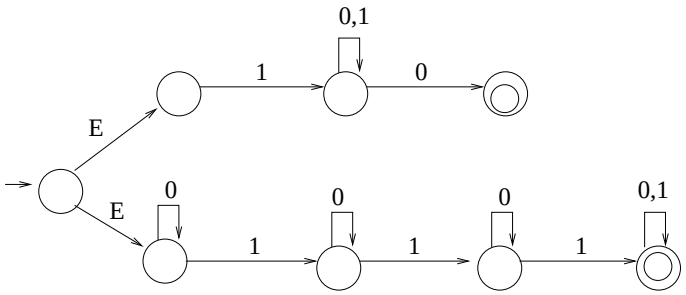

Fig. 4. Solution for 2.

PROBLEM 3 (1.7a) See figure 5.

PROBLEM 4a (1.10 a)

Let M be some DFA. Since M is a DFA, we know that for any input string S there is exactly one state that M is in after the entire string S is processed. Let this state be known as F.

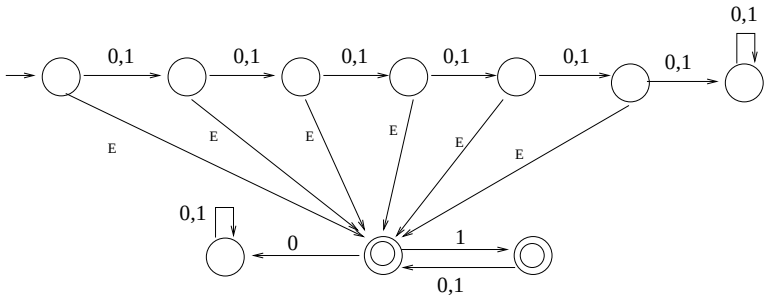

Fig. 5. Solution for 3.

Let M' be a DFA, formed by copying the states in M, and toggling the accept staus of each state such that accept states become non-accept, and non-accept states become accept states.

Since no transitions have changed, it is clear that string S will still reach state F. If state F is accepted by M, it will not be accepted by M'. If state F is not accepted by M, it will be accepted by M'. Therefore, for any DFA M, there is some DFA M' that accepts a complement language of M. As a language is regular iff it can be written as a DFA, this indicates that regular languages are closed under complement.

PROBLEM 4b (1.10 b)

In figure 6, M will accept string string "1". M' will also accept the string M. Therefore, swapping the accept and non-accept status of states does not yield a complement of M in this case.

The class of languages recognized by NFAs is still closed under complement, however. The reason is that a language can be described by an NFA iff the language can also be described by a DFA. Thus, using the result derived in problem 4a showing that DFAs are closed under complement, we know that NFAs are closed under complement.

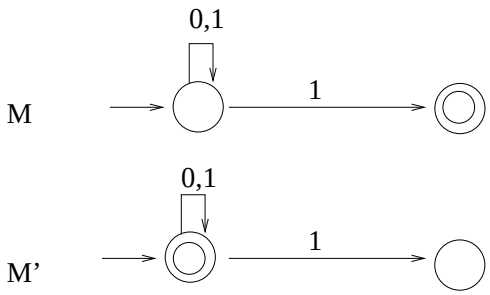

Fig. 6. Solution for 4b.

PROBLEM 5 (1.12a) See figure 7.

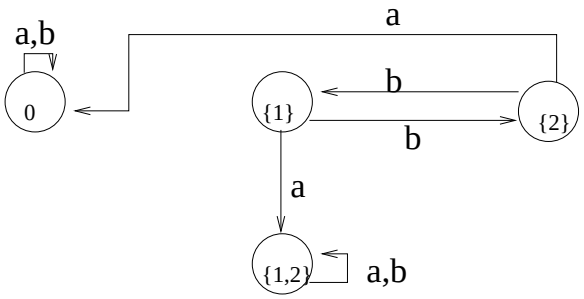

Fig. 7. Solution for 5.

PROBLEM 6 (1.16b) See figure 8.

PROBLEM 7 (1.17a)

$A = \{0^n 1^n 2^n | n \geq 0\}$

1. Assume A is regular. Then, the pumping lemma holds.
2. Let p be pumping length of A, given by the pumping lemma.
3. Let $s = 0^p 1^p 2^p \in L$. Note that s is a string in language L of length $geqp$.

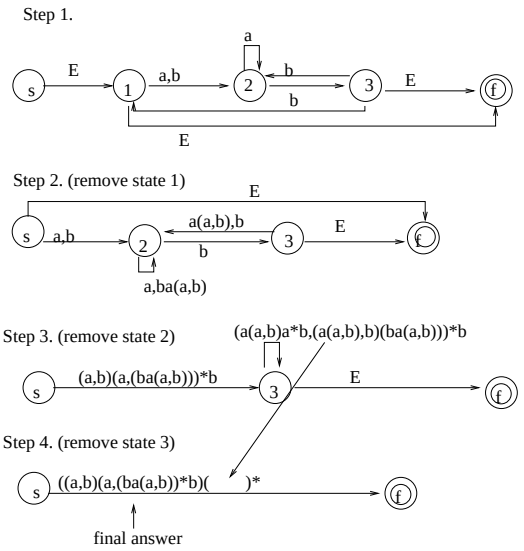

Fig. 8. Solution for 6.

4. Consider all cases s can be diviided into x, y, and z, satisfying the property $|y| > 0$ and $|xy| \leq p$. Clearly, xy must contain only 0's (else $|xy| > p$). When pumped to xy^2z, the resulting string will have more than p zeros. The number of 1s and 2s, however, will still be p. Thus, the resulting string is not in A.

5. This contradicts the pumping lemma. Therefore, one of our assumptions was incorrect. As our only assumption was that A was regular, A must not be regular.

PROBLEM 8 (1.24)

If A is a regular language, there is some DFA a that describes it. The NFA a^r is constructed as follows:

1. Copy a. Add a new start state, with $\in$ transitions to each of the original final states.
2. make the original start state an accept state instead.
3. without changing the labels on transisitions, reverse the arrow direction of each state transition.

It can be seen that a accepts a string w if and only if the reverse of w is accepted by a^r. This is because all ending states in the original DFA are now starting states, all starting states in the original DFA are now ending states, and all transitions from original starting states toward original ending states are now transitions from original ending states toward original starting states. Thus, for any language A there is an NFA that accepts A^r.

As there exists an NFA (a^r that accepts the reverse of any language A^r, if A is regular, the reverse of A must be regular.

PROBLEM 9 (1.25) Figure 9 shows an NFA for B^R. Since B^R is regular, by problem 1.24 B is regular.

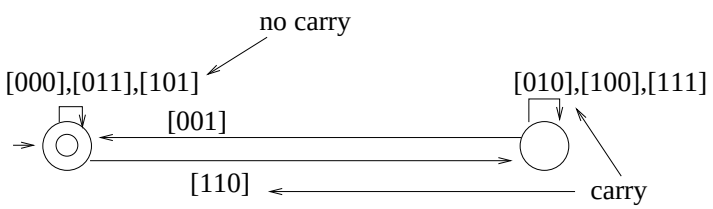

Fig. 9. Solution for 9.

PROBLEM 10.

1. Assume S is a regular language containing a string s iff string s has balanced prenthesis. As S is regular, the pumping lemma must hold.
2. Let p be the pumping length if s given by the pumping lemma.
3. Let $s1$ be the string $(^p)^p$, contained by the langage S. Note that $s1$ is of lenth $\geq p$.

4. In all cases where $s1$ is divided into x, y, and z, where $|y| > 0$ and $|xy| \leq p$, y contains at least one (and 0).
5. When $s1$ is pumped to $xyyz$, the resulting string is unbalanced, thus by the pumping lemma, S is not regular. This creates a contradiction, indicating one of the assumptions was wrong. The only assumption was that S was a regular language containing only strings of balanced parenthesis. Therefore, There is no regular language that describes strings of balanced parenthesis.

PROBLEM 11 (1.16b) As can be seen in figure 10, the minimal DFAs for $(a \cup b)*$ and $(a * \cup b*)*$ are equivalent. Therefore, the two regular expressions are equivalent.

type = byte,short,int,long
opt = public,private,static,final,transient,volatile, E
name = ({A-Z,a-z},_,$)({A-Z,a-z},_,$,{0-9})*

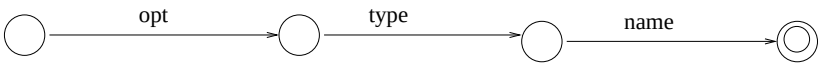

Fig. 10. Solution for 11.

PROBLEM 12 See figure 11

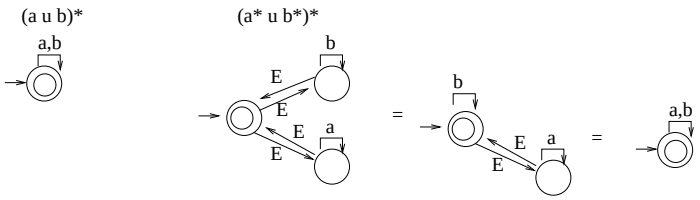

Fig. 11. Solution for 12.