CS 310, Assignment 3

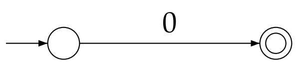
Answers

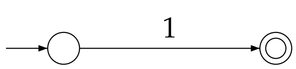
1. Using one of the method described in class and/or textbook (Section 9.1) convert the following regular expression into a state transition diagram:

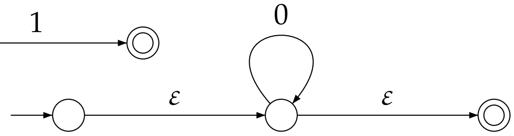
$$(0^*1 + 1^*0)^*(1 + 0)^*$$

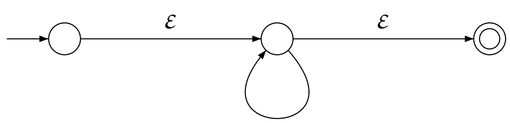
Indicate in your answer how did you arrive at the result as follows: Write down all the state transition diagrams that you constructed for all the subexpressions and clearly indicate which diagram corresponds to which expression. Do *not* simplify any state transition diagram.

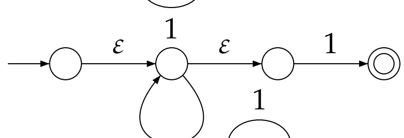
ANSWER:

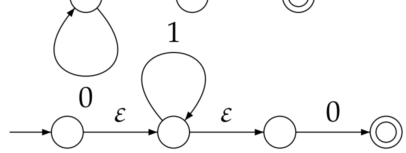
(a) Automaton for 0: 

(b) Automaton for 1: 

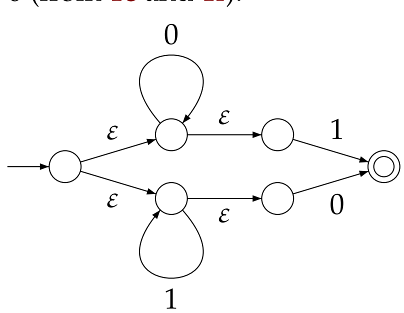
(c) Automaton for 0^* (from 1a): 

(d) Automaton for 1^* (from 1b): 

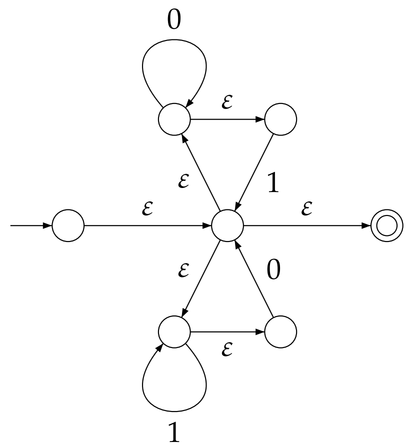
(e) Automaton for 0^*1 (from 1b and 1c): 

(f) Automaton for 1^*0 (from 1a and 1d): 

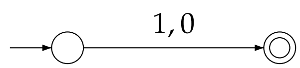
(g) Automaton for $0^*1 + 1^*0$ (from 1e and 1f):



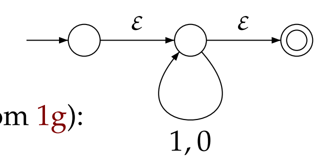
(h) Automaton for $(0^*1 + 1^*0)^*$ (from 1g):



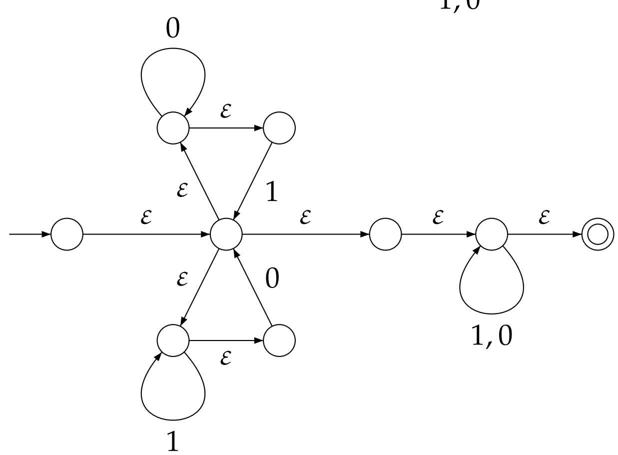
(i) Automaton for $1 + 0$ (from 1a and 1b):



(j) Automaton for $(1 + 0)^*$ (from 1i):



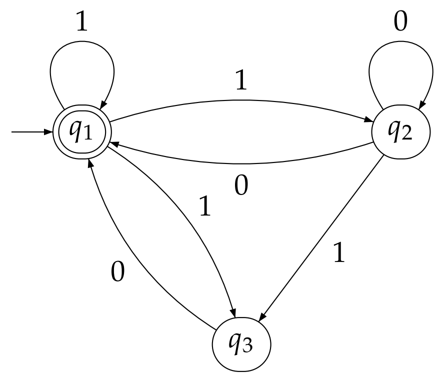
(k) Automaton for $(0^*1 + 1^*0)^*(1 + 0)^*$ (from 1g):



Note that I used the technique that merges states. Using ε-transitions instead would have been equally fine, but the resulting automaton would have been considerably larger.

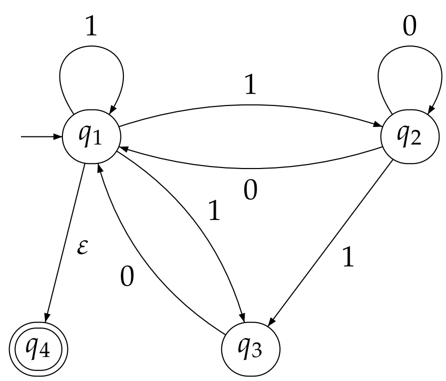
---

2. Consider the following state transition diagram over $\Sigma = \{0, 1\}$:



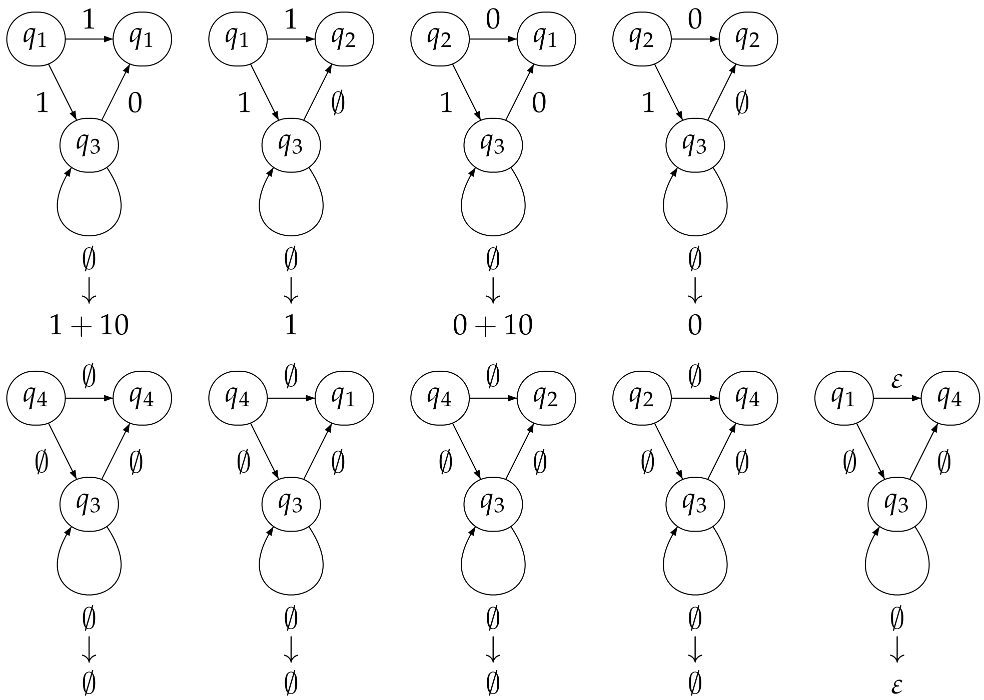
Using the method described in class and in the textbook (Section 9.2) convert the diagram into an equivalent regular expression. Include all the intermediate steps in your answer.

ANSWER: We have a single accepting state which happens to be identical to the initial state. We can live with that and eliminate all the other stated, ending with a single state and a loop transition as shown in class. Being the sucker for punishment I am going to do it the hard way though (following the algorithms from the textbook) and separate the initial and accepting state by introducing a new accepting state:

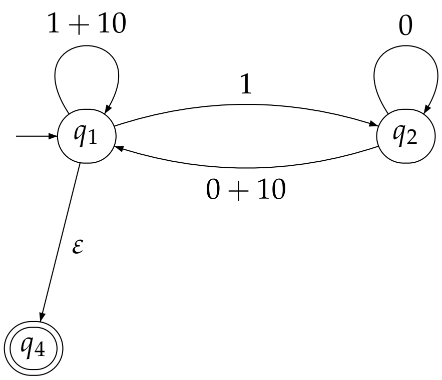


It is worth noting that the resulting regular expression is going to be the same whether we introduce this kind of a separate accepting state or not.

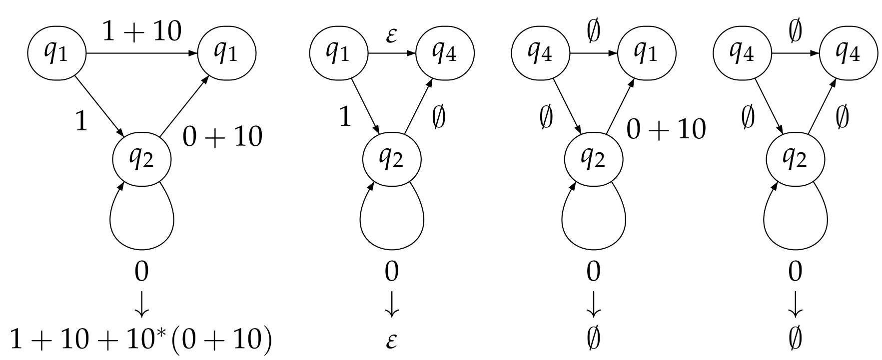
We now have two states that need to be eliminated. We start by eliminating q_3 (just a random choice, we could have eliminated q_2 instead). We have the following "triangles" with their respective regular expressions, simplified according to the facts that $\emptyset^* = \varepsilon$ and $\emptyset L = L\emptyset = \emptyset$:



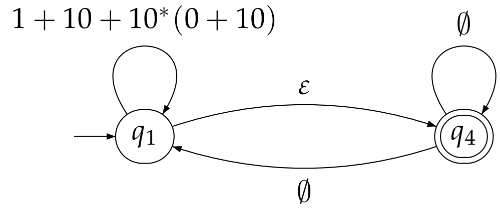
This in turn results in the following generalized state transition diagram:



We then eliminate the remaining state which is neither initial nor accepting namely, q_2:



We end up with the following generalized transition diagram:



The regular expression equivalent to the original transition diagram is therefore:

$$(1 + 10 + 10^*(0 + 10))^*\varepsilon(\emptyset^* + \emptyset(1 + 10 + 10^*(0 + 10))^*\varepsilon)^*$$

Given that $\emptyset^* = \varepsilon$, $\varepsilon^* = \varepsilon$, ε is an identity for concatenation, and $\emptyset$ is a zero for concatenation the expression can be easily simplified to the following:

$$(1 + 10 + 10^*(0 + 10))^*$$

which in turn can be converted in the following possibly cleaner form:

$$(1 + 10 + 10^*0 + 10^*10)^*$$

---

3. Are the languages L_1 and L_2 below over the alphabet $\Sigma = \{a, b, c\}$ regular or non-regular? Justify your answer carefully.

   (a) $L_1 = \{a^{2i}b^jc^i : i \geq 0, j > 2\}$

   ANSWER: L_1 is not regular and we will prove it so using the pumping lemma.

   Assume therefore that L_1 is regular. Note that both i and j are arbitrarily large, so there exists a string $w = a^{2i}b^jc^i \in L_1$ that is longer than the threshold n and so the pumping lemma applies to it. In fact we will take $i = n$ (so that w is *much* longer than n) that is, $w = a^{2n}b^jc^n$ for some $j > 2$.

   From the pumping lemma we have that $w = xyz$ such that $xy^2z \in L_1$. We furthermore have $|xy| < n$. It follows that y only contains a symbols that is, $y = a^m$ for some $m > 0$ (indeed, $|xy| < n$ so xy must come from the first n symbols of w which are all a).

   If this is the case, then xy^2z has $2n + m$ a's (we have increased their number since we pumped y) and n c's (we have not touched those). Since $xy^2z \in L_1$ it follows that the numbers a's is twice as much as the number of c's that is, $2n + m = 2n$ which is equivalent to $m = 0$. This contradicts the fact that $m > 0$ and so our initial assumption (that L_1 is regular) must be false. $\square$

   What happens with the b's you ask? We have not touched those simply because we were able to come up with a string w long enough so that the b's do not enter the picture. This is fortunate, since the b's could have been pumped liberally and so would have spoiled our day.

(b) $L_2 = \{a^i b^{2j+1} : i, j \geq 0\} \cap \{a^{2k+1} b^{2n} c^{2p} : k, n, p \geq 0\}$

ANSWER: We can show that L_2 is regular by noting that $L_2 = (L_{21}L_{22}) \cap (L_{23}L_{24}L_{25})$, where $L_{21} = \{a^i : i \geq 0\} = a^*$, $L_{22} = \{b^{2j+1} : j \geq 0\} = b(bb)^*$, $L_{23} = \{a^{2k+1} : k \geq 0\} = a(aa)^*$, $L_{24} = \{b^{2n} : n \geq 0\} = (bb)^*$, and $L_{25} = \{c^{2p} : p \geq 0\} = (cc)^*$.

All the languages L_{2j} are regular, $1 \leq j \leq 5$; indeed, I just gave above the respective regular expressions. Thus L_2 is a combination of concatenating and intersecting regular languages (see above), and regular languages are closed under both concatenation and intersection. It follows that L_2 is regular. □

A more direct (but less general) way to show the same thing is to note that L_2 cannot contain any c (since c's do not appear in the first term of the intersection) and so a string in L_2 has the form $a^x b^y$ for some $x, y \geq 0$. Furthermore x must be odd (because of the second term in the intersection). More interestingly, y must also be odd (this time because of the first term) but at the same time it must be even (because of the second term). That is, y does not have any valid value and so $L_2 = \emptyset$, which is clearly regular. □

Note in passing that this is not the language I meant to give you; the b^{2n} term was meant to be b^n and so the language should have been $a(aa)^* b(bb)^*$. This is obviously immaterial for this assignment but is worth noting since I will do my best for the future questions not to include trivial languages like this.
