Formal Languages Homework 4

Liam Dillingham

February 25, 2019

1 Problem 3.2.2

Here is a transition table for a DFA:

	0	1
$\to q_1$	q_2	q_3
q_2	q_1	q_3
$*q_3$	q_2	q_1

1.1 a) Give all the regular expressions $R_{ij}^{(0)}$. Note: Think of state q_i as if it were the state with integer number i.

$R_{ij}^{(0)}$	q_1	q_2	q_3
$\to q_1$	ϵ	0	1
q_2	0	ϵ	1
$^*q_3$	1	0	ϵ

1.2 b) Give all the regular expressions $R_{ij}^{(1)}$. Try to simplify the expressions as much as possible

$R_{ij}^{(1)} = R_{ij}^{(0)} + R_{i1}^{(0)}(R_{11}^{(0)})^* R_{1j}^{(0)}$

$R_{ij}^{(1)}$	q_1	q_2	q_3
$\to q_1$	ϵ	0	1
q_2	0	$\epsilon + 00$	$1 + 01$
$^*q_3$	1	$0 + 00$	$\epsilon + 1$

1.3 c) Give all the regular expressions $R_{ij}^{(2)}$. Try to simplify the expressions as much as possible

$R_{ij}^{(2)} = R_{ij}^{(1)} + R_{i2}^{(1)}(R_{22}^{(1)})^* R_{2j}^{(1)}$

$R_{ij}^{(1)}$	q_1	q_2	q_3
$\rightarrow q_1$	$\epsilon + 0(\epsilon + 00)^*0$	$0 + 0(\epsilon + 00)^*(\epsilon + 00)$	$1 + 0(\epsilon + 00)^*(1 + 01)$
q_2	$0 + (\epsilon + 00)(\epsilon + 00)^*0$	$\epsilon + 00 + (\epsilon + 00)(\epsilon + 00)^*(\epsilon + 00)$	$1 + 01 + (\epsilon + 00)(\epsilon + 00)^*(1 + 01)$
$^*q_3$	$\epsilon + 1 + (0 + 00)(\epsilon + 00)^*0$	$0 + 00 + (0 + 00)(\epsilon + 00)^*(\epsilon + 00)$	$\epsilon + 1 + (0 + 00)(\epsilon + 00)^*(1 + 01)$

1.4 d) Give a regular expression for the language of the automaton

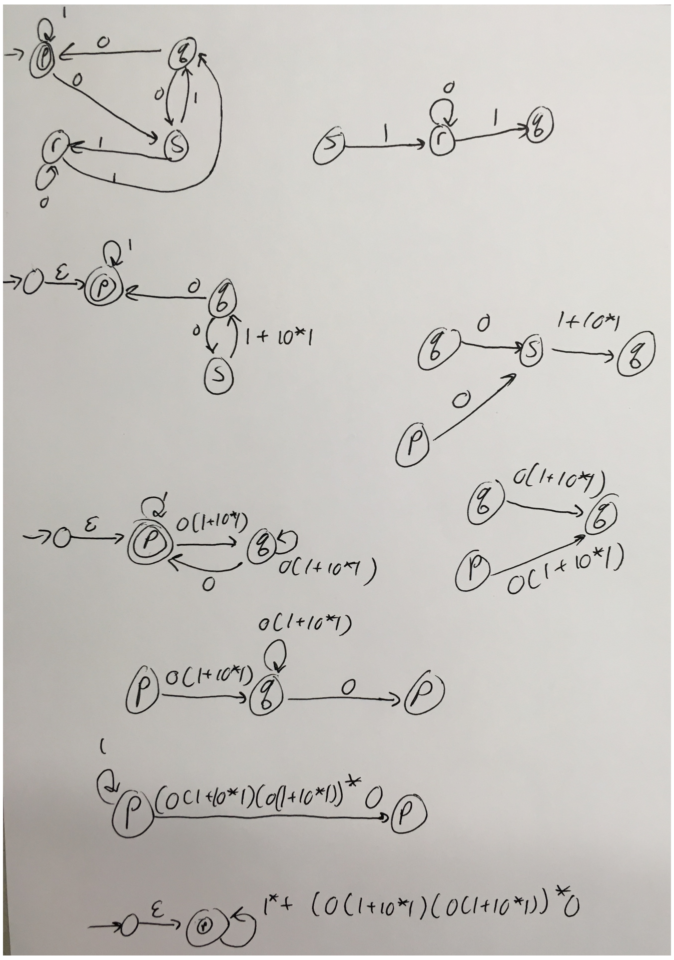
final answer: $1 + 0(00)^*(01 + 0)((10 + 1)(00)^*(01 + 0) + 11)^*$.

2 Problem 3.2.3

Convert the following DFA to a regular expression, using the state-elimination technique of Section 3.2.2

	0	1
$\rightarrow * p$	s	p
q	p	s
r	r	q
s	q	r

I wanted to include the scratchwork I used to calculate the regular expression:

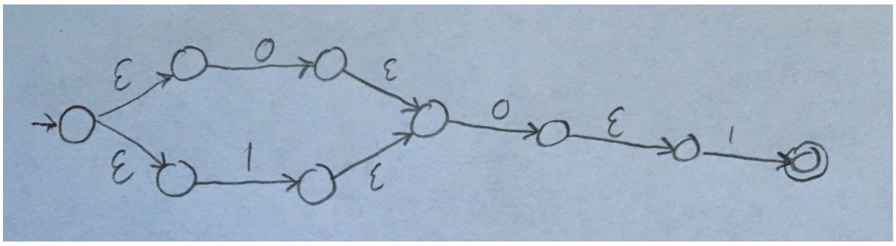


Final Answer:
$1^* + (0(1 + 10^*1)(0(1 + 10^*1))^*0$

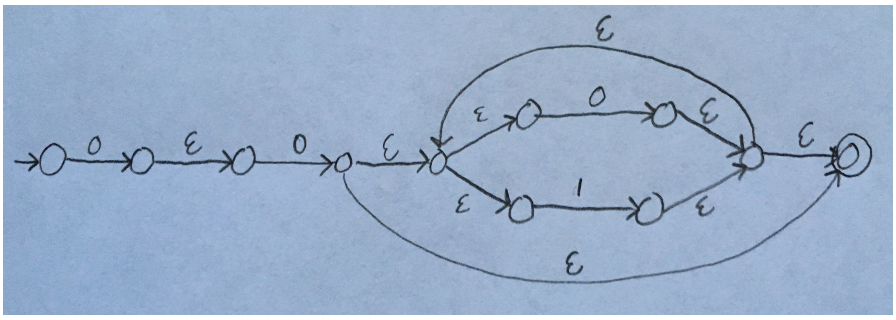
3 Problem 3.2.4

Convert the following regular expressions to NFA's with ϵ-transitions.

3.1 b) $(0+1)01$



3.2 c) $00(0+1)^*$



4 Problem 3.4.1

Verify the following identities involving regular expressions.

4.1 b) $(R + S) + T = R + (S + T)$

Let $r \in R, s \in S, t \in T$ be representative elements from their respective regular expressions. Starting on the left hand side, we have $r + s = \{r, s\}$. Then $\{r, s\} + t = \{r, s, t\}$. For the right hand side, we have $s + t = \{s, t\}$, and then $r + \{s, t\} = \{r, s, t\}$. Thus the two sides are equal.

4.2 c) $(RS)T = R(ST)$

Let $r \in R, s \in S, t \in T$ be representative elements from their respective regular expressions. rs produces the set $\{rs\}$, with t gives us $\{rst\}$. Similarly, st gives us $\{st\}$, and with r we have $\{rst\}$. Both sides are equal.

4.3 h) $(R^*S^*)^* = (R + S)^*$

Let $r \in R, s \in S$ be representative elements from their respective regular expressions. Note on the right hand side, we have the set $\{r, s\}$. Then to compute the kleene closure, we pick an element from this set and append it to our word. We get a set like this: $\{\epsilon, r, s, rs, sr, ssr, rrs, srs, rsr, rrr, ..\}$. That is, every combination of s's and r's for any length. For the right hand side, we have $R^* = \{\epsilon, r, rr, rrr, ...\}$, and $S^* = \{\epsilon, s, ss, sss, ...\}$. Then we compute the closure of the concatenation of these two sets. Then we dot these two sets together, giving us a set with an arbitrary number of r's on the left, and s's on the right. With this, we compute the kleene closure, where we pick an arbitrary number of elements from our set and concatenate them. To prove this simply, note that both r and s are in this set, that is $\{r, s\} \subset R^*S^*$. Then, by simply selecting from this small subset, we can build a string of arbitrary length of any combination of s's and r's. Thus, the two sets are equal.