CS 374 HW 1 Problem 1

Aldo Sanjoto, Hieu Huynh, quddus2

TOTAL POINTS

100 / 100

QUESTION 1

1 1A **20 / 20**

✓ **- 0 pts** Correct

- 10 pts Incorrect regular expression

- 10 pts Lack or incorrect explanation

QUESTION 2

2 1B **20 / 20**

✓ **- 0 pts** Correct

- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation

QUESTION 3

3 1C **20 / 20**

✓ **- 0 pts** Correct

- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation

QUESTION 4

4 1D **20 / 20**

✓ **- 0 pts** Correct

- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation

QUESTION 5

5 1E **20 / 20**

✓ **- 0 pts** Correct

- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation

1a.)

$\varepsilon + 0 + 1 + 00 + 01 + 10 + 11 + 001 + 011 + 100 + 110 +$ ~~[illegible]~~ $111 +$

$000 + (0+1)(0+1)(0+1)(0+1)(0+1)^*$

The following solution is brute-fore. All the possible ~~of~~ strings of length 0, 1, 2 are considered and then for strings of length 3 all ~~[illegible]~~ strings are listed except 010 & 101. For strings ~~a~~ of length 4 or more the following expression $(0+1)(0+1)(0+1)(0+1)(0+1)^*$ list all such strings

b.) $(10)(0+1)^*(111)(0+1)^*$

The following solution is correct because all strings must start with 10 so 10 is listed ~~on~~ in the beginning of the expression and since all strings must have the substring 111 the following ~~solution~~ expressio $(0+1)^*(111)(0+1)^*$ considers all strings with the substring 111 and so $(10)(0+1)^*(111)(0+1)^*$ is all strings with the substring 111 which also begin with 10

c.) $(0+11)^*$ the following expression is correct because any time we decide to insert consequtive one's they will be of even length due to the fact that reg expession can chose between 0 or 11 and there is no chance to have a single one which will make the maximal substring of consequtive ones odd

1 1A 20 / 20

✓ - 0 pts Correct

- 10 pts Incorrect regular expression

- 10 pts Lack or incorrect explanation

1a.)

$\varepsilon + 0 + 1 + 00 + 01 + 10 + 11 + 001 + 011 + 100 + 110 + 111 +$

$000 + (0+1)(0+1)(0+1)(0+1)(0+1)^*$

The following solution is brute-fore. All the possible ~~of~~ strings of length 0, 1, 2 are considered and then for strings of length 3 all strings are listed except 010 & 101. For strings of length 4 or more the following expression $(0+1)(0+1)(0+1)(0+1)(0+1)^*$ list all such strings

b.) $(10)(0+1)^*(111)(0+1)^*$

The following solution is correct because all strings must start with 10 so 10 is listed in the beginning of the expression and since all strings must have the substring 111 the following ~~solution~~ expressio $(0+1)^*(111)(0+1)^*$ considers all strings with the substring 111 and so $(10)(0+1)^*(111)(0+1)^*$ is all strings with the substring 111 which also begin with 10

c.) $(0+11)^*$ the following expression is correct because any time we decide to insert consequtive one's they will be of even length due to the fact that reg expression can chose between 0 or 11 and there is no chance to have a single one which will make the maximal substring of consequtive ones odd

2 1B 20 / 20

✓ **- 0 pts** Correct

- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation

1a.)

ε + 0 + 1 + 00 + 01 + 10 + 11 + 001 + 011 + 100 + 110 + [redacted] 111 +

000 + (0+1)(0+1)(0+1)(0+1)(0+1)*

The following solution is brute-fore. All the possible ~~of~~ strings of length 0, 1, 2 are considered and then for strings of length 3 all ~~[illegible]~~ strings are listed except 010 & 101. For strings ~~a~~ of length <u>4 or more</u> the following expression (0+1)(0+1)(0+1)(0+1)(0+1)* list all such strings

b.) (10)(0+1)*(111)(0+1)*

The following solution is correct because all strings must start with 10 so 10 is listed ~~on~~ in the beginning of the expression and since all strings must have the substring 111 the following ~~solution~~ expressia (0+1)*(111)(0+1)* considers all strings with the substring 111 and so (10)(0+1)*(111)(0+1)* is all strings with the substring 111 which also begin with 10

c.) (0+11)* the following expression is correct because any time we decide to insert consequtive one's they will be of even length due to the fact that reg expression can chose between 0 or 11 and there is no chance to have a single one which will make the maximal substring of consequtive ones odd

3 1C 20 / 20

✓ - 0 pts Correct

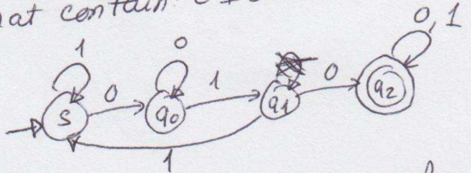
- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation

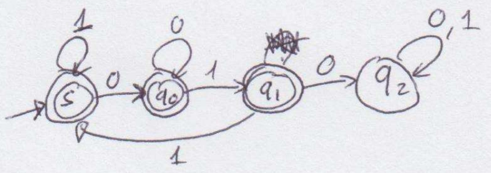
1D) All strings that do not contain substring 010:

$$(1+00^*11)^*(\varepsilon+00^*+00^*1).$$

DFA of strings that contain 010:



⇒ Taking complement of DFA above, we have DFA of all strings that don't contain 010



• From the DFA above, we can see that starting from state S, we can have any number of 1s and still be at state S. Also, starting from state s, the sequence 00^*11 will ~~bring~~ come back to state s.
Therefore, any sequence that has prefix $(00^*11+1)^*$ will end at state S.
From state S, we can go to state q_0 by using the sequence 00^*
From state S, we can go to state q_1 by using the sequence 00^*1
From state S, we can stay in state S by using ε
Putting every thing ~~together~~ together, we have the regular expression: ~~that~~

$$(1+00^*11)^*(\varepsilon+00^*+00^*1)$$

Source:

"https://www.itsalif.info/content/dfa-nfa-regular-expression-without-using-gnfa"

4 1D **20 / 20**

✓ **- 0 pts** Correct

- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation

1e) $1^*0^* + 0^*1^*0^*$

There are 2 cases the regex can have:

① String start with 1: any number of 1s, then any number of 0s.

② String start with 0: any number of 0s, then any number of 1s, then any number of 0s.

Thus, the sequence of 101 will not possible.

[illegible]

5 1E 20 / 20

✓ - 0 pts Correct

- 10 pts Incorrect regular expression

- 10 pts Missing or incorrect explanation