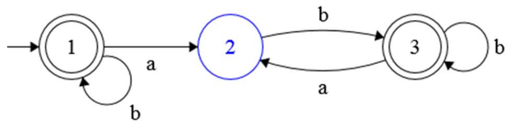
CSE 34151 Spring 2018 Exam 1

- SIGN YOUR NAME ON THE TOP OF ALL PAGES!!!!!!
- Sit so there is an empty chair on both sides of you.
- Do all problems and return all sheets.
- This is an in-class exam, with only open book and this year's class notes permitted. If you have an e-book you may use your computer for that but under the ND Honor Code you may not use the computer for any other purpose, i.e. computer searches, emails, texts, cell phones, or communications with or help from others. All other aspects of the ND Honor code apply. Note you CANNOT include materials such as answer keys from non-class or prior years sources.
- Show all you work in the spaces supplied, including auxiliary equations or definitions that you may have used. **You may use the space below and the blank page at the end for spill-over. Indicate on the original problem page if you use either.**
- Short, clear pictures and diagrams are fine - you need not add lots of explanations.

Problem	Max Points	Points Off
1	20	
2	20	
3	20	
4	20	
5	20	
Total	100	



1. (20pt) For the following DFA M with $\Sigma = \{a, b\}$:
   (a) (5pt) Describe the language L accepted by M in words or a regex. Words are simpler if done right. Ensure you include all possible strings in the language.
   (b) (5pt) Describe the language L_R that is **the reverse** of this language; i.e. if $w = w_1 ... w_n$ is in L, then $w^R = w_n ... w_1$ is in L_R.
   (c) (5pt) Design NFA N_R that accepts L_R. Show all components. Table or state diagram is ok. Do not bother with trap states.
   (d) (5pt) Describe briefly how, given *any* DFA $(Q, \Sigma, \delta, q_0, F\}$ that accepts some L, you would build an $NFA = (Q_R, \Sigma_R, \delta_R, q_R, F_R\}$ that accepts the reverse of L. You do not need to "prove" it but show formally how each component of the NFA would be constructed. Do not bother with trap states. **What does this tell you about properties of the reverse of any regular language.**

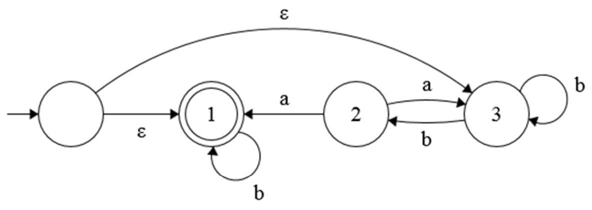
Solution:

Part a: In words: $\{w|$ w from Σ^* with 0 or more a's, and every a is followed by at least one b$\}$ or all strings from Σ^* that don't end with an "a" or contain "aa", or is ε or starts with a b and does not contain aa

As a regex: $b^* \cup b^*ab(ab \cup b)^*$ or more simply $(ab \cup b)^*$ or $b^* \ \cup \ (b^*abb^*)^*$

Part b: $\{w|w$ from Σ^* 0 or more a's, and every a is preceded by at least one b$\}$

Part c: if you reverse the given machine (alternatives may be simpler)



Part d: In words, add a new start state with an ε branch to each of the prior final states. Make the final state the initial starting state. Reverse all arcs.

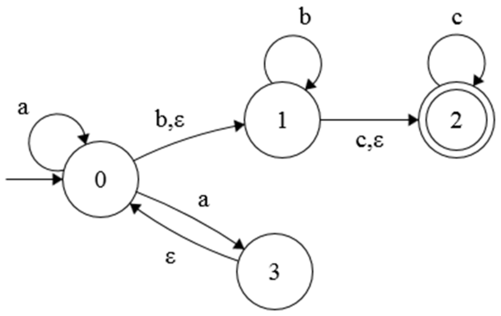
- $Q_R = Q \cup \{q_R\}$, q_R is a new state
- $\Sigma_R = \Sigma$
- $F_R = \{q_0\}$
- q_R is start state (new)
- $\delta_R(q_i, a) = q_j$ where $\delta(q_j, a) = q_i$
- $\delta(r_R, \varepsilon) = F$

Thus for any regular language, the reverse is also regular (regular languages are closed under reversal)

Scoring:

- -0.5 if F not a set
- added ε to Σ
- -1 if not formal
- -1 if no comment about property of reversed languages
- -1 if approach does not handle multiple accepting states in original DFA

2. (20pt) NFA to DFA: Convert the following NFA N to a DFA D. **Clearly list all components** of D. Start by listing E(q) for all states q. For δ, either a state diagram or transition table is sufficient. A table may be faster. Note, you need only show states reachable from the start state.



Solution:

- $E(0) = \{0, 1, 2\}$
- $E(1) = \{1, 2\}$
- $E(2) = \{2\}$
- $E(3) = \{0, 1, 2, 3\}$

- $\Sigma = \{a, b, c\}$
- $Q = P(\{0, 1, 2, 3\})$
- start = $\{0, 1, 2\}$
- F is all states including 2.

State	a	b	c
$\{0, 1, 2\}$	$\{0, 1, 2, 3\}$	$\{1, 2\}$	$\{2\}$
$\{0, 1, 2, 3\}$	$\{0, 1, 2, 3\}$	$\{1, 2\}$	$\{2\}$
$\{1, 2\}$	$\emptyset$	$\{1, 2\}$	$\{2\}$
$\{2\}$	$\emptyset$	$\emptyset$	$\{2\}$

Since state 2 is in all states above, all are in F.

Scoring:

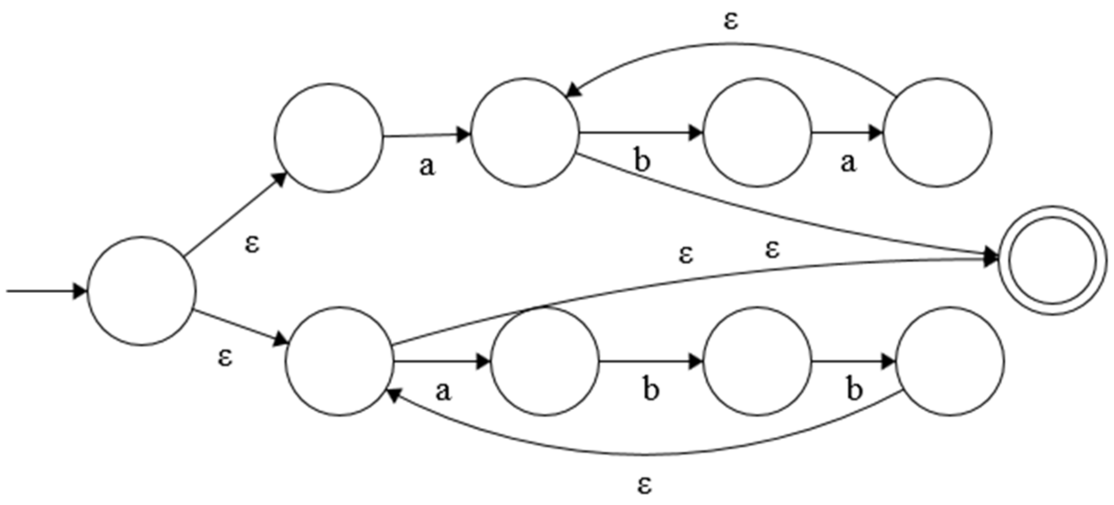
- 4 points for 4 E values
- 1 point each for Σ, Q, start, F
- 12 points for table -0.5 for $\{\emptyset\}$

3. (20pt) Use the construction process in the proof of Lemma 1.55 to convert the following regex to an NFA (a DFA if you can manage it but don't stress out about it):

   $a(ba)^* \ \cup \ (abb)^*$

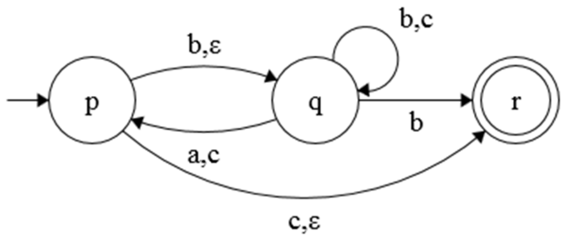
   Show your NFA as a state diagram. You need not show each step of the constructions, and you need not be totally faithful to putting all the ε edges and intermediate states specified by the algorithm where they are redundant, but your diagram should still be correct and reflect a solid understanding of the construction process. This may best be shown by annotating which parts of the diagram come from which major terms of the regex. Do not bother with trap states.

   Solution:

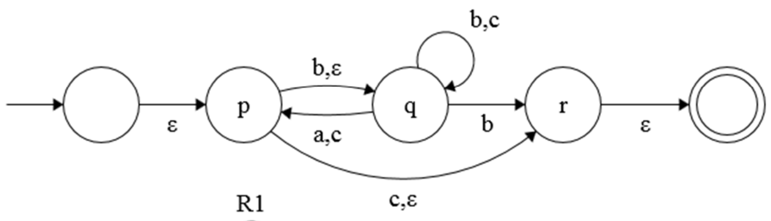


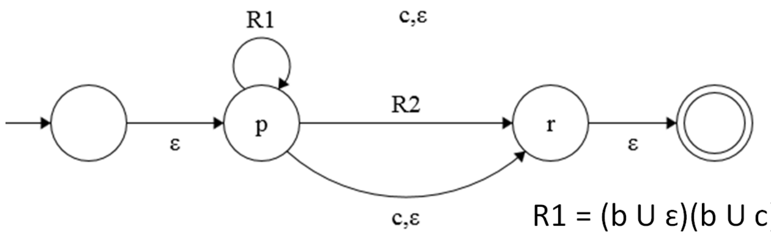
- 6 points handling concatenations correctly
- * 3 points each
- $\cup$ 6 points - needs start state
- 2 points for final state

4. (20pt) Determine what regular expression the following NFA is equivalent to, using the GNFA approach. Show each step. To simplify grading, remove states in order q,p,r. It is strongly recommended that you use variables like $R_1 = ...$ to simplify your edge labels. You can use the blank sheet following this if you need more room. Feel free to add states to the figure below for your first step.

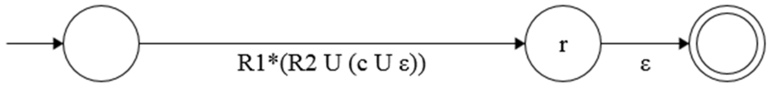


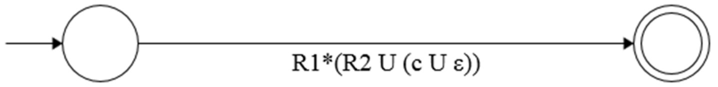
Solution:





R1 = (b U ε)(b U c)*((a U c)
R2 = (b U ε)(b U c)*b





- 4 points per step,
- 4 points for final expression,
- Typically points taken off for first occurrence of an error, but rest of problem then graded as if error was correct. Typical errors:
  - Writing εx instead of just x. (0.5 points)
  - Not following the order of state removal: A,B,C,D,E
  - Dropping an edge
  - Listing a path thru states as union, not concatenation
  - Not handling self-loops properly
  - Not removing accepting state status from initial accepting states
  - Not adding new start or accept state
  - not writing x,y as $x \cup y$
- -2 if you did not rip in specified order
- -3 altogether missing self-loop

5. (20pt) Assume P is the language of matching parenthesis, i.e. $P = \{w | w = w_1 ... w_i ... w_{2n} | n \geq 0, \Sigma = \{(,)\}$, where in any prefix substring $w_1 ... w_k$ there are never more ")" than "(", and in the whole string the numbers of "(" and ")" are equal (n each).$\}$. Examples include "(()()(()))" but not ")()(" or "(()". Prove P is not regular. Be specific as to why pumping is not possible, i.e. what is in x, y, and z and what are all possible choices.

Solution:

Choose $s = (^p)^p, \ |s| = n2p > p$

Since $|xy| \leq p$, x and y must be all ('s. Assume $x = (^a$, $0 \leq a \leq p-1$ and $y = (^b$ where $1 \leq b \leq p-a$. Then $z = (^{p-(a-b)}(^{p-a-b})^p$

Now $s = xyz = (^a(^b z$. If regular then all string of the form $(^a(^{ib}(^{p-a-b})^p = (^{p-b+ib})^p$ must be in P.

Either of the following now shows that this can't be true.

If $i = 0$ then $(^{p-b})^p$ must be in P. not enough (

If $i = 1$ then $(^{p+b})^p$ must be in P, too many)

Scoring:

- 6 points for a reasonable string
- 6 points for defining x and y well
- 8 points for choosing and demonstrating values of i

This is a blank sheet for your use. Label which problem you are using it for.