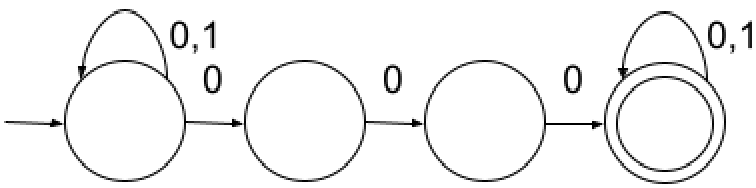
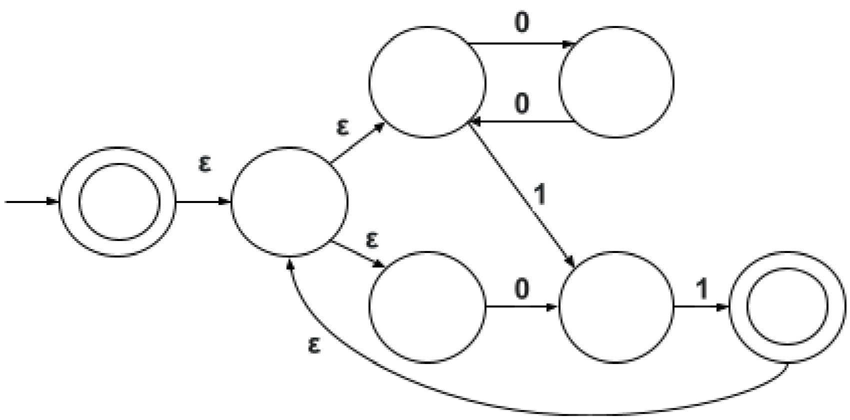
Aughdon Breslin and Isabella Cruz
9/25/20
"I pledge my honor that I have abided by the Stevens Honor System."

Problem 1
For each regular expression below, construct an equivalent NFA.

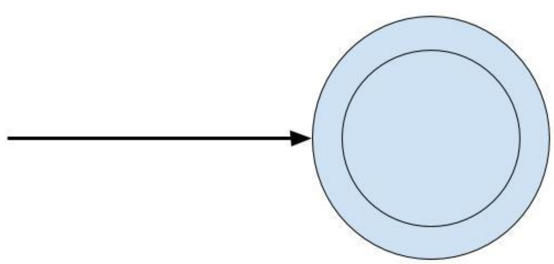
a)$(0 \cup 1)^* 000 (0 \cup 1)^*$



b) $(((00)^* 11) \cup 01)^*$



c)$\phi*$



Problem 2

a) $\{w \in \{a, b\}* : w$ does not end in ba$\}$
   i) ε U a U (a U b)*(aa U b)

b) $\{w \in \{0,1\}* : w = \alpha \circ \beta, \alpha$ has an even number of 1' s and β has an even number of 0'$s\}$
   i) 0* (1 0* 1)* 1*(0 1* 0)*

Problem 3

In some programming languages, comments appear between delimiters such as /# and #/. Let C be the language of all valid delimited comment strings. Each member of C must begin with /# and end with #/ but have no intervening #/. For simplicity, let the alphabet be {a, b, /, #}. Give a regular expression to describe C.

basically any character forever, except after the #s there's gotta be a letter to prevent #/

| once done with slashes, can put as many # as
| you like, also this way can have # w/o a U b after
| |

A: /# (a U b U #*(a U b) U /)* #* #/

Problem 4

Prove that the following languages are regular (your answer can be any one of: DFA/NFA/regular expression). Unless stated otherwise, $\Sigma = \{a, b\}$.

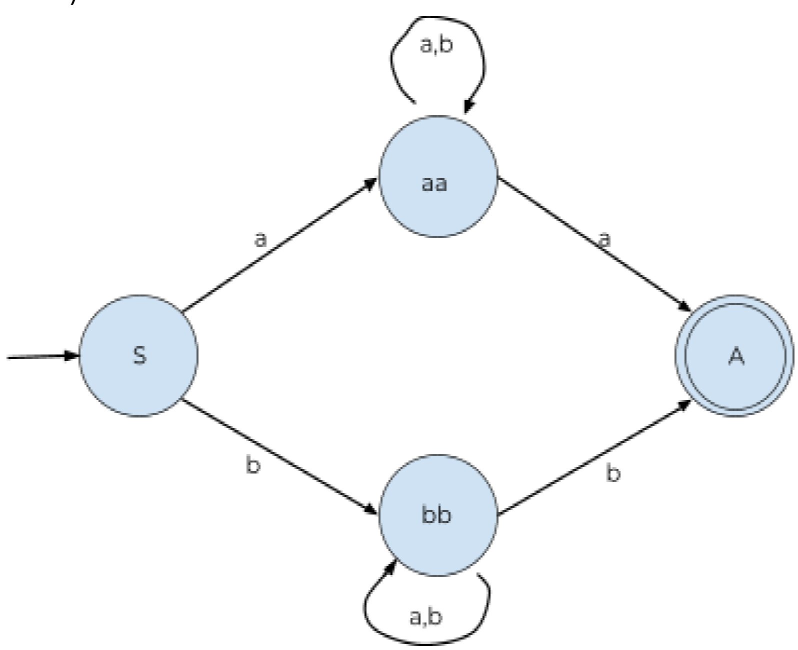
a) $\{w: w$ starts and ends with the same symbol$\}$

b) Let $\Sigma = \{a, b, c, d\}$. The language L consists of all strings in which at least one symbol of Σ is missing.

c) $\{w : w$ has even length and an odd number of $a's\}$

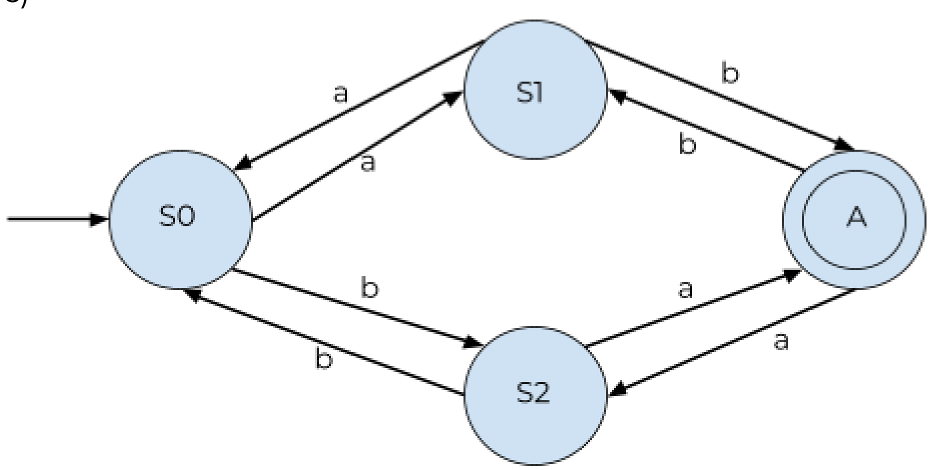
A:

a)



b) ε U a*b*c* U a*b*d* U a*c*d* U b*c*d*

c)



Problem 5
Using regular expressions only (no NFA/DFA involved) prove that the reverse of every regular language is also regular

A: As we learned in lecture, a language is regular if it can be described with a regular expression. We can use the theorem from Problem 2 of Problem Set 2 to show that if we can simplify an expression to a language, we can prove its reverse is regular. If regular expression R describes a regular language, R^R would be:

1. If R = a, a $\in \Sigma$,
$R^R = a^R = a$; so when reversed the language is still regular.

2. If R = ε ,
$R^R = \varepsilon^R$ still contains only the empty string; so it is regular

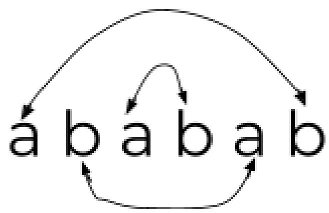
3. If R = Φ,
$R^R = \Phi^R$ still is equal to the empty set; so it is regular

4. If R = Reg1 U Reg2,
R^R = (Reg1 U Reg2)^R would equal the regular expression Reg1^R U Reg2^R. Since R^R can be described using that regular expression, it is a regular language.
ex/ if Σ = {a,b}, Reg1 = a, Reg2 = b; R = Reg1 U Reg2 = a U b, so R^R would accept a^R U b^R; thus R^R is regular because it can be expressed using the regular expression a^R U b^R

5. If R = Reg1 **o** Reg2,
R^R = (Reg1 **o** Reg2)^R would equal Reg2^R **o** Reg1^R. Since R^R can be described using that regular expression, it is a regular language.
ex/ if Σ = {a,b}, Reg1 = a, Reg2 = b; R = Reg1 o Reg2 = a o b, so R^R would accept b^R **o** a^R; thus R^R is regular because it can be expressed using the regular expression b^R **o** a^R

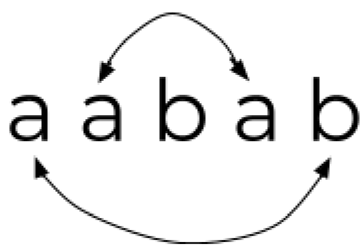
6. If R = R*,
$R^R = (R^R)^*$; so if R is regular, R^R is regular because it can be represented by the regular expression $(R^R)^*$
ex/ if Σ = {a,b} and R = (ab)* then R^R = (ba)*; thus R^R is regular because it can be expressed using a regular expression (ba)*

7. If R = string with an even length,
R^R = the reverse of R. Since R^R can be described using that regular expression(the string reversed), it is a regular language.
ex/ if Σ = {a,b} and R = ababab; R^R would be "bababa" where:



the outermost members of the string will be switched, the the second outermost, so on and so forth until you've switched the middle to members of the string. R^R would be regular because it is described by this new regular expression "bababa".

8. If R = string with an odd length,
R^R = the reverse of R. Since R^R can be described using that regular expression(the string reversed), it is a regular language.
ex/ if Σ = {a,b} and R = aabab; R^R would be "babaa" where:



the outermost members of the string will switch, the then second outermost, so on and so forth until you reach the member at the (string length // 2) + 1 (integer division) index of the string. That member does not move and you have now successfully reversed the string.

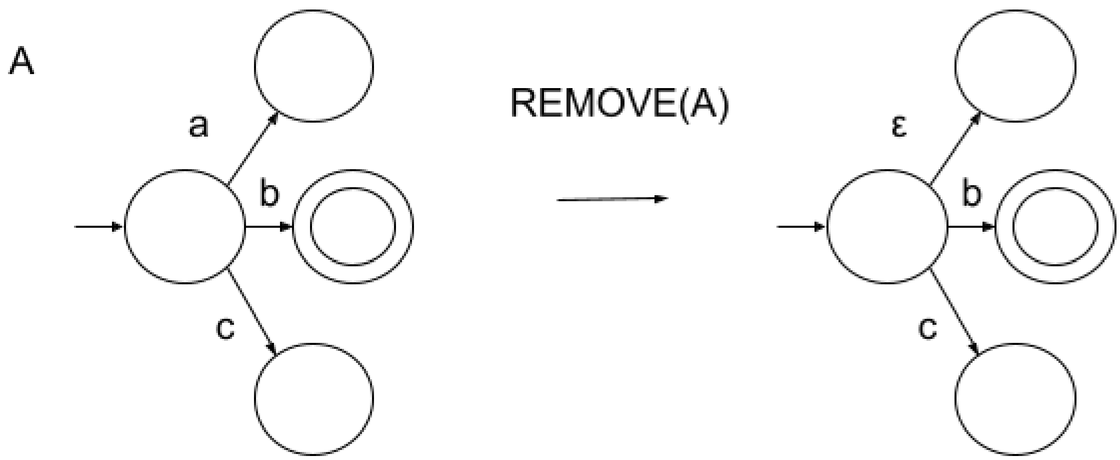
R^R would be regular because it is described by this new regular expression "babaa".

Problem 6

For any language A, define $REMOVE(A)$ to be the language consisting of all strings in A but with exactly one symbol missing. Formally, $REMOVE(A) = \{\alpha\beta : \alpha y\beta \in A,\ y \in \Sigma,\ and\ \alpha, \beta \in \Sigma^*\}$

Note: For convenience, removing a symbol from the empty string results in the empty string.

Construct an NFA to show that if A is regular then so is $REMOVE(A)$. Next, prove this using regular expressions only



.

------------ Side Questions (Answered) ------------

Can you define the rules to change reg ex in english? - Yes, it can be done in english

When it says "All strings but with exactly one symbol missing" does it mean Abcdab turns to abcda_ or to a_cda_? Would you just remove a symbol entirely or one occurrence of one symbol? Just one occurrence

----

A: Given a regular expression that defines the language A, removing one occurrence of a symbol with REMOVE(A) would still leave a regular language.

If the removed symbol fell within an a* (where a $\in A$, a $\in \Sigma$), then the new regular expression would remain the same. Since the symbol was removed from a*, there had to be at least one occurrence of a within a*, and one less occurrence therefore could still be represented using a*.

> Ex: A is represented by a*b. "ab" $\in A$. REMOVE(A) yields just "b". "b" is still rep'd by a*b and REMOVE(A) is still regular. aaaab would yield aaab and is still rep'd by a*b; the language is still regular.

If the removed symbol fell on either side of a concatenation (◦) or union ($\cup$), then the new regular expression would consist of the old regular expression but with an empty string (ε) as replacement of the removed symbol.

Ex1 - Concatenation: A is rep'd by ab. REMOVE(A) would be rep'd by εb, or just b.

Ex2 - Union: A is rep'd by a $\cup$ b. REMOVE(A) would be rep'd by $\varepsilon \cup$ b.

In any case, the removal of exactly one symbol in a regular expression would not destroy the validity of the regular expression under these rules for change. As noted in the problem, removal of a symbol from language A represented by ε would yield a language REMOVE(A) that continues to be represented by ε. Therefore, if A is regular then so is REMOVE(A).

Note: REMOVE(A) can remove any symbol that belongs to the language indiscriminately, I just removed 'a' for simplicity of explanation (and because it reads as "REMOVE A").