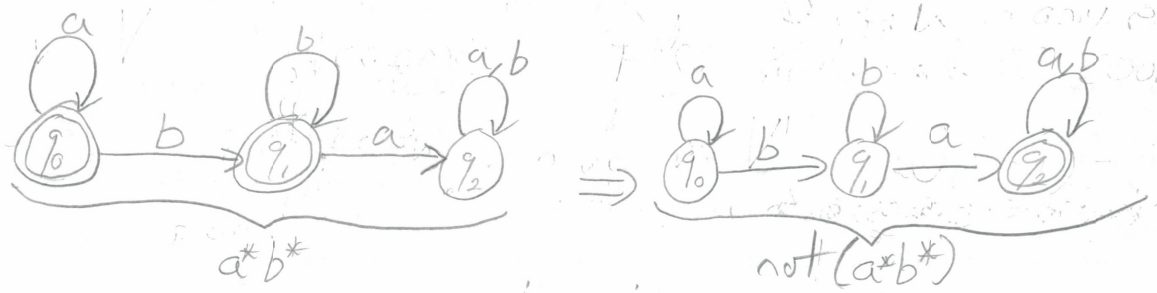
Brandon Smith

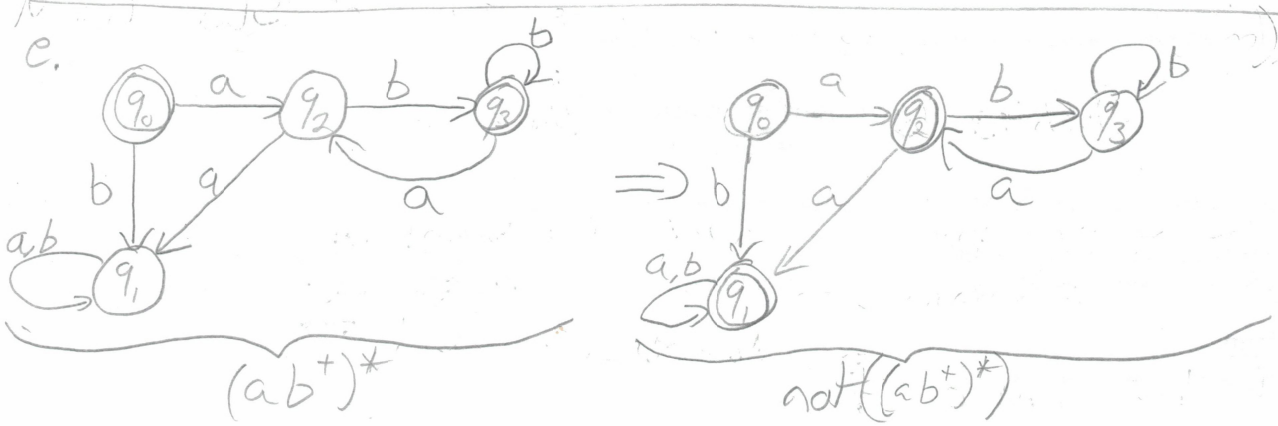
October 1st, 2015

COMP 330: Assignment 1

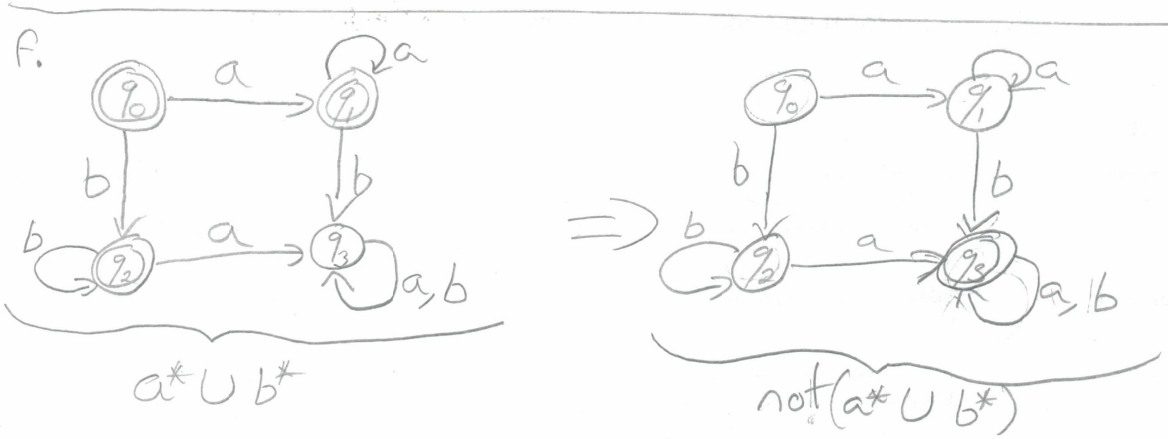
1.5 d.



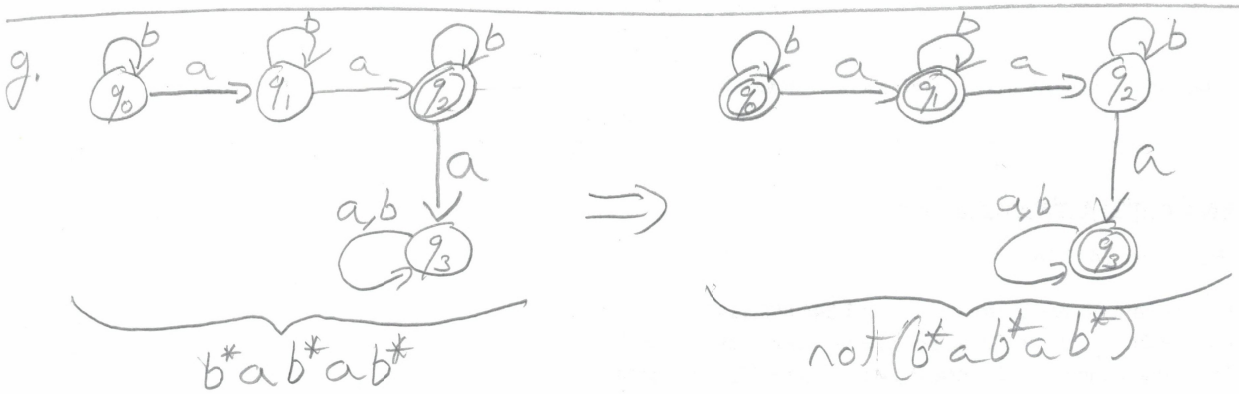
e.



f.

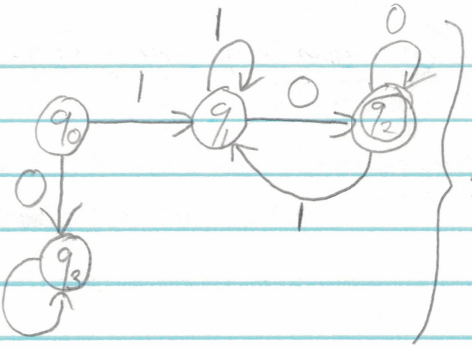


g.



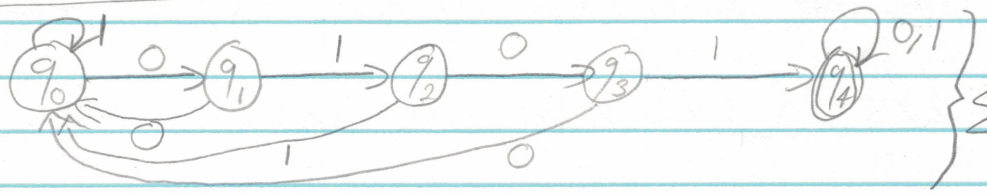
1.6

a.



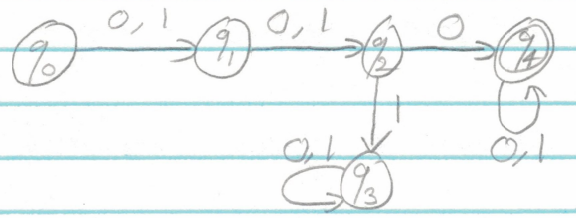
$1\Sigma^*0$

c.



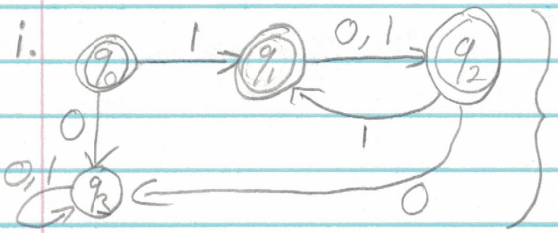
$\Sigma^*0101\Sigma^*$

d.



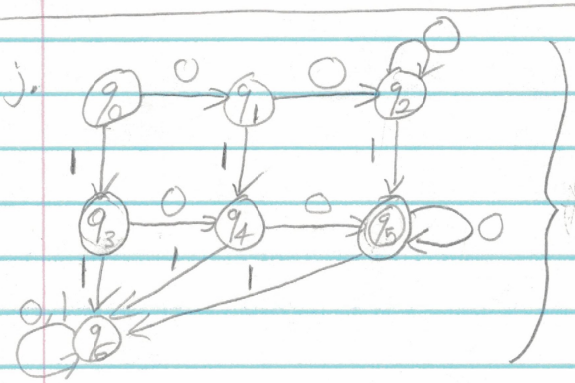
$(0\cup1)(0\cup1)01\Sigma^*$

i.



$(1(1\cup0))^*$

j.



$(1000^*)\cup(0100^*)\cup(0010^*)$

m.

$\emptyset$

n.

$(0\cup1)\Sigma^*$

1.14

a. Given $M \Rightarrow B$, show $\overline{M} \Rightarrow \overline{B}$

$M = (Q, \Sigma, \delta, q_0, F)$

$\overline{M} = (Q, \Sigma, \delta, q_0, Q-F)$

$\overline{M}$ guaranteed to be deterministic because it has the same δ.

Let e be the ending state s.t. $e \in Q$,

Given $w \notin B \Rightarrow e \notin F \Rightarrow e \in Q-F$

so $w \notin B$ is accepted by $\overline{M}$.

Given $w \in B$, $e \in F \Rightarrow w \in B$ is

Not accepted by $\overline{M}$. Therefore

$\overline{M}$ accepts only what is NOT in B

which is $\overline{B}$.

$\Rightarrow$ "all ... not a part of B"

$\Rightarrow$ If a string in $\overline{B}$ is parsed by M,

$\Rightarrow$ then M will finish in a non-accepting

state by the assumption that M

recognizes B. Thus, if we construct

$\Rightarrow$ an identical DFA but swap the accept

and non-accept states, labeling it $\overline{M}$,

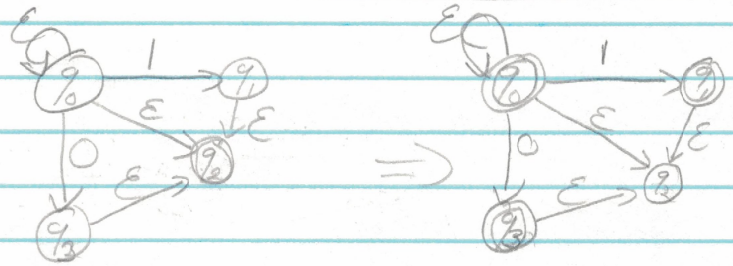
then parsing $\overline{B}$ through $\overline{M}$ will be guaranteed

$\Rightarrow$ to accept because the only non-accepting

states are those that recognize B.

$\Rightarrow$ if $M \Rightarrow B$ then $\overline{M} \Rightarrow \overline{B}$ ✓ Yes Regular languages are closed under comp

b.



This accepts 1, 0, and ε

This also accepts 1, 0, and ε

By the theorem 1.39, there exists a DFA that recognizes the language C recognized by the NFA. So, if we have a DFA M recognizing C then by part a. $\overline{M}$ recognizes $\overline{C}$ which means that the class of languages recognized by NFA's are closed under complement.

Probability Book

Sept 2, 2011

1.20

a. Members: aaaabbbb, aaaaabbb
Not Members: aabbaaa, bbbbbba

b. Members: abababab, ababababab
Not Members: bbbbbbbb, aaaaaaaa

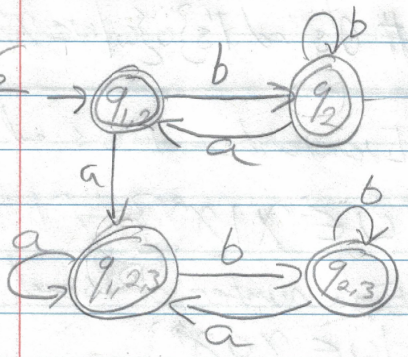
c. Members: aaaaaaaa, bbbbbbbb
Not Members: aaaaaab, bbbbbba

e. Members: ababbba, bbabbbbbaba
Not Members: bbbbbbbb, aaaaaaaa

f. Members: None of size ≥ 7, aba, bab
Not Members: ababábab, babababa

g. Members: None of size ≥ 7, b, ab
Not Members: bbbbbbb, abaaaaa

1.16



A nice Automata that accepts everything.

1.31 Thm. 1.54 ⇒ A language is regular iff some regular expression describes it

⇒ R = regular expression describing A.
⇒ By def. 1.16 that ∃D (a DFA) which recognizes R and A.

IP we then reverse δ in D, that is reverse all edge transitions

in D, we will be left with D^R. D^R will be an NFA acd by Thm. 1.39 There must exist a DFA which also describes D^R.

Therefore, by definition 1.16, the language described by D^R is regular. Thus proving that if A is regular, A^R must also be regular.

1.46

a. In order to describe $0^n1^m0^n$ using a DFA, we would need $2n+m+1$ edge transitions. m and n are only restricted by a lower bound $(m,n \geq 0)$ so the upper bound on m, n is countably infinite. We have thus created a DFA with $2n+m+2$ states which has index $K = 2n+m+2$ by the Myhill Nerode Theorem. Thus K is bounded by ∞ so by the Myhill Nerode Theorem $\{0^n1^m0^n \mid n,m \geq 0\}$ is not a regular language.

---

c. $\{w \mid w \in \{0,1\}^* \text{ Not a palindrome}\}$ can be written differently as: $\overline{\{w \mid w \in \{0,1\}^* \text{ A palindrom}\}}$ $x \Rightarrow x^R$ (reversed)

Which can again be written differently as:

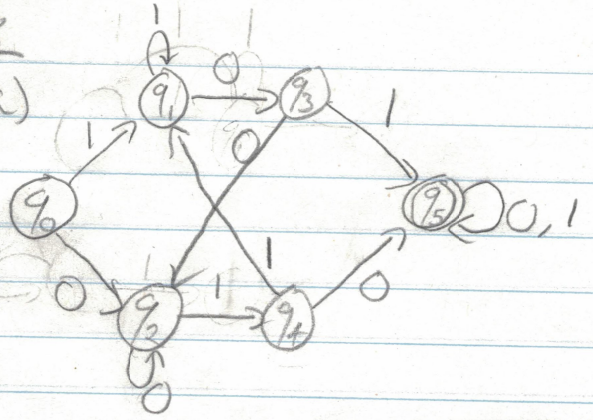
$$\overline{\left(\{w \mid w \text{ has even length}\}0\{w \mid \text{Even \# 0's and 1's}\}0\{w \mid w \in x x^R\}\right)}$$

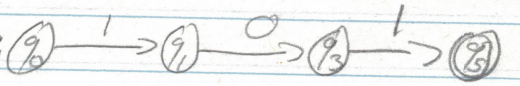
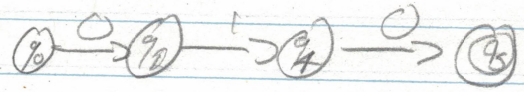
$$\cup\Big(\{w \mid w \text{ has odd length}\}0\Big(\{w \mid \text{Even number of 0's and odd 1's}\}0\{w \mid w \in x 1 x^R\}\Big) \cup \Big(\{w \mid \text{Even number of 1's and odd 0's}\}0\{w \mid w \in x 0 x^R\}\Big)\Big)$$

I was attempting to decompose the original language description into sub-parts such that one or more sub-parts would be non-regular, and by the knowledge that regular languages are closed under union, complement, and intersection we could thus declare that the original language is not regular.

1.98

a)



b) Base Case: 1:  2:  } Accepts strings 101 & 010

I.H.:

My intuition for this question was to show that if it works for the sub-strings by themselves, then by adding any additional character at any point in the string (that does not break the sub-string) then the DFA would still accept the string. Then I would show that if you insert a character in such a way that it breaks the sub-string (and doesn't create a new sub-string 101 or 010) then the DFA no longer accepts the string, therefore proving that the DFA works IF AND ONLY IF the input string has a sub-string 101 or 010.