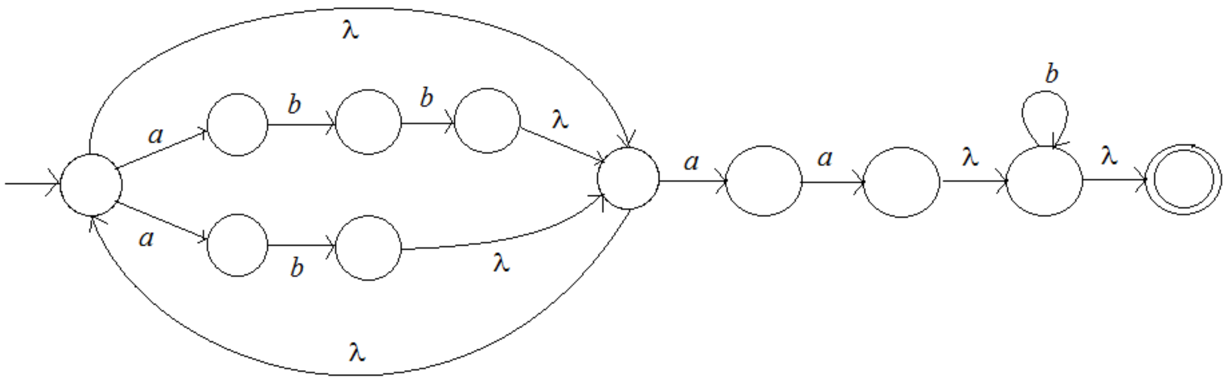
Cosc 362, Spring 2017
Test #2 – Take-Home
Solutions

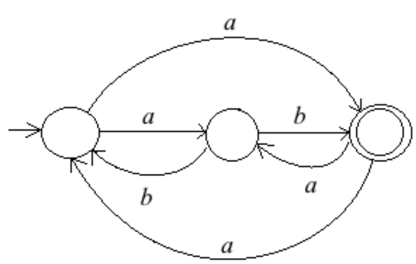
1. Use the construction in Theorem 3.1 to find an nfa that accepts the language $L(r)$, where

$$r = (abb + ab)^* aab^*.$$

Solution: An nfa that can be constructed following the guidelines of Theorem 3.1 is as follows:



2. Use the procedure demonstrated in class (or, if you prefer, the "nfa-to-rex" procedure in the text) to find a regular expression that generates the language accepted by the following nfa:



Solution: One possible solution is $(ab)^*(ab + a)\big(ab + (ab + a)(ab)^*(ab + a)\big)^*$. Other, equivalent solutions are possible, depending on the sequence in which the steps of the conversion algorithm are carried out. Click here for a diagram outlining the steps by which we obtained our solution.

3. Show there exists an algorithm that, given any two regular languages L_1 and L_2, determines whether there exists a string, w, such that $w \in L_1$ and $w^R \in L_2$. Give a thorough explanation, citing theorems or examples from the text as needed.

Solution: The set of strings such that $w^R \in L_2$ is simply L_2^R. So, we're looking for an algorithm to determine whether $L_1 \cap L_2^R$ is non-empty. The family of regular languages is closed under intersection (Theorem 4.1.1) and reversal (Theorem 4.1.2); therefore, $L_1 \cap L_2^R$ is regular if L_1 and L_2 are both regular. Therefore, by Theorem 4.2.2 (check this), there exists an algorithm to determine whether $L_1 \cap L_2^R$ is non-empty (namely, construct a dfa for $L_1 \cap L_2^R$, and then inspect all paths in the transition diagram to determine whether a path exists from the initial state to a final state).

4. a) Find an s-grammar for $L = \{a^n b^{n+2} : n \geq 2\}$.
   b) Based on your grammar from part (a), give the derivation tree for the string $aaaabbbbbb$.

Solution:

a. An s-grammar that generates L is as follows:

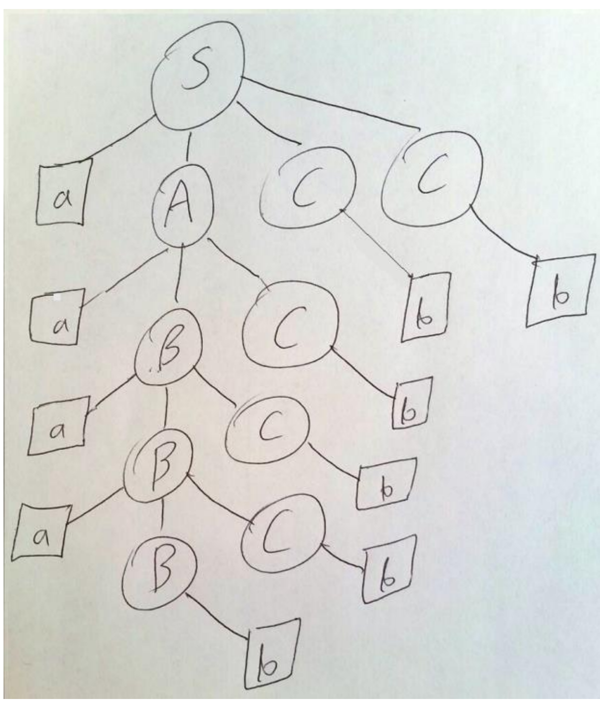
$$S \to aACC$$
$$A \to aBC$$
$$B \to aBC \mid b$$
$$C \to b$$

We can summarize all possible derivations in this grammar as follows:

$$S \to aACC \to aACC \to aaBCCC \underbrace{\to}_{\substack{B \to aBC \\ k\ times, k \geq 0}} aaa^k BC^k CCC \underbrace{\to}_{\substack{B \to b \\ C \to b}} a^{k+2} b^{k+4}$$

So, $w \in L(G)$ iff $w = a^{k+2} b^{k+4}$ for some $k \geq 0$, or equivalently iff $w = a^n b^{n+2}$ for some $n \geq 2$.

b.



5. Let L be the language consisting of all strings of even length whose two <u>middle</u> letters are aa. (For example L contains aa; $a\boldsymbol{aa}a, a\boldsymbol{aa}b, b\boldsymbol{aa}a, b\boldsymbol{aa}b$; all six-letter strings whose third and fourth letters are aa; and so on.)

a) Prove that this language is *not* regular.
b) Show that this language *is* context-free.

(Note: For #5, a more formal definition of L would be $\{w_1 aa w_2 : w_1, w_2 \in \{a,b\}^*, |w_1| = |w_2|\}$)

a. Assume (for a contradiction) that L is regular. Let m be the constant as defined in the Pumping Lemma, and let $w = b^m aab^m$. Then, $w \in L$, and, by the Pumping Lemma, w must have some decomposition $w = xyz$ such that $|xy| \leq m, |y| \geq 1$, and $xy^i z \in L$ for all $i \geq 0$.

While we can't specify what substrings x, y, and z, our choice of w guarantees that x and y both consist entirely of $b's$. (Since $|xy| \leq m$, the substring xy is entirely contained in the prefix b^m.) This means $x = b^j, y = b^k$, and $z = b^{m-j-k}aab^m$, for some j, k where $j + k \leq m$ and $k \geq 1$.

The Pumping Lemma now guarantees that $xy^i z \in L$ for all non-negative integers i. In particular, we can choose $i = 0$, which means $xz \in L$. This shortens the opening string of $b's$ by at least 1; in particular, $xz = b^j\, b^{(m-j-k)}aab^m = b^{m-k}aa\, b^m \in L$. Since this string starts with fewer b's than it ends with, the substring aa is not in the middle of the string; thus, we have a string in L whose middle two letters are not aa. But this contradicts the definition of language L. The contradiction is induced by our assumption that L is a regular language; therefore, L must not be regular.

b. To show a language is context-free, simply find a context-free grammar that generates it. One such grammar for L is as follows:

$$S \to ASA \mid aa$$
$$A \to a \mid b$$

This grammar generates all strings of the form $w_1 aa w_2$, where w_1 and w_2 are of the same length. (The common length of w_1 and w_2 is equal to the number of times the rule $S \to ASA$ is followed.)