CS321
Theory of Computation
Exam I, Fall 2008

Name:

1. (a) [2pt] What is a language?

   A set of strings.

   (b) [4pt] Given a language L, define the *decision problem* that is associated with L. Do this by specifying what the possible problem inputs are and what the desired output is for a given input.

   The input to the decision problem is any finite string. The desired output is 'accept' if the input string is in L and 'reject' otherwise.

   (c) [8pt] Consider a DFA $M = (Q, \Sigma, \delta, q_0, F)$.

   i. What is the domain and range of the transition function δ? (The domain of a function is the set of possible inputs. The range is the set of possible outputs.)

   The domain is $Q \times \Sigma$. The range is Q.

   ii. Give a formal definition of L(M) in set notation. (You will want to make use of the extended transition function δ^*.)

$$L(M) = \{w : \delta^*(q_0, w) \in F\}$$

   (d) [8pt] Consider an NFA $M = (Q, \Sigma, \delta, q_0, F)$.

   i. What is the domain and range of the transition function δ?
   The domain is $Q \times \Sigma \cup \{\lambda\}$. The range is 2^Q, i.e. the set of all subsets of Q, or the powerset of Q.

   ii. Give a formal definition of L(M) in set notation. (You will want to make use of the extended transition function δ^*.)

$$L(M) = \{w : \delta^*(q_0, w) \cap F \neq \emptyset\}$$

   (e) [3pt] Given an NFA N and a regular expression r, what does it mean to say that N and r are equivalent?

   This means that they define the same language. That is, $L(N) = L(r)$.

2. Consider the following language

$$L = \{w_1 w_2 w_3 : w_1 \in \{aa\}^+, w_2 \in \{ba\}^*, w_3 \in \{bb\}^+\}$$

(a) [4pt] Give an informal description of L in your own words.

The language contains all strings that begin with a strictly positive even number of a's, followed by zero or more repetitions of ba, followed by a strictly positive even number of b's.

(b) [5pt] Circle the strings in the following list that are in L.

$$\lambda \qquad \underline{aabb} \qquad \underline{aababbabb} \qquad aaababbabb \qquad abaababb$$

(c) [8pt] Consider the following language over alphabet $\Sigma = \{a, b\}$

$$L_1 = \{w_1 b a w_2 : w_1 = a^n, w_2 = b^m, n \geq 0, m \geq 0\}$$

Describe L_1^R in set notation.

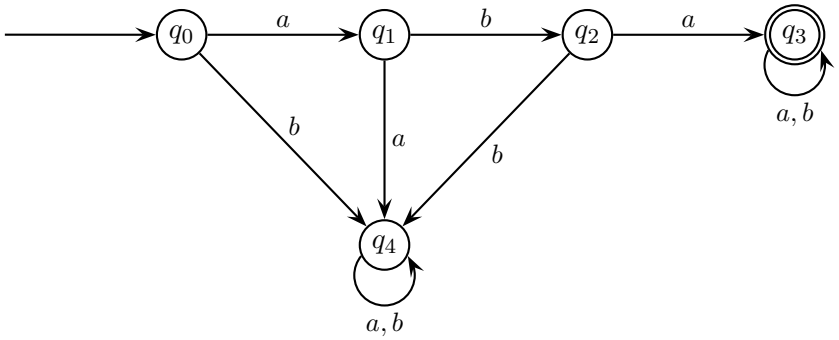
$$L_1 = \{w_1 a b w_2 : w_1 = b^m, w_2 = a^n, n \geq 0, m \geq 0\}$$

(d) [8pt] Let $L_2 = \{b\}^*\{a\}^*$. Describe the language $L_1 \cap L_2$ in set notation, where L_1 is defined as in part (c).

The only string in L_1 that consists of b's followed by a's is when $n = m = 0$ leaving just the middle ba. So we have that

$$L_1 \cap L_2 = \{ba\}$$

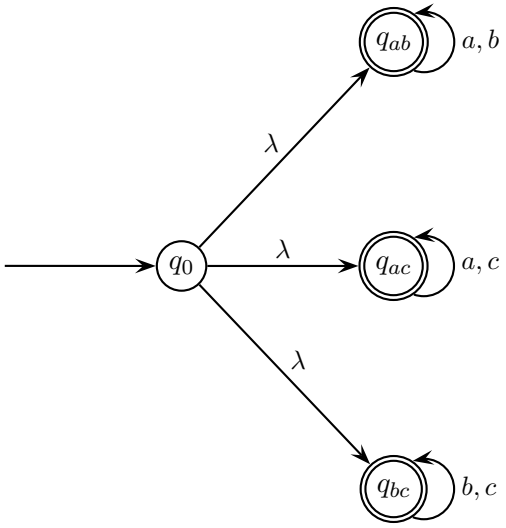
3. (a) [8pt] Draw a DFA M such that $L(M) = \{w : w = abaw_1, w_1 \in \{a, b\}^*\}$.



(b) [8pt] Draw an NFA N such that

$$L(N) = \{w : w \in \{a, b, c\}^*, w \text{ does } \textit{not} \text{ contain all of the characters in } \{a, b, c\}\}$$

. So for example aa, bb, cc, abba, bbc, and ca are in $L(N)$, but abc and baacc are not.



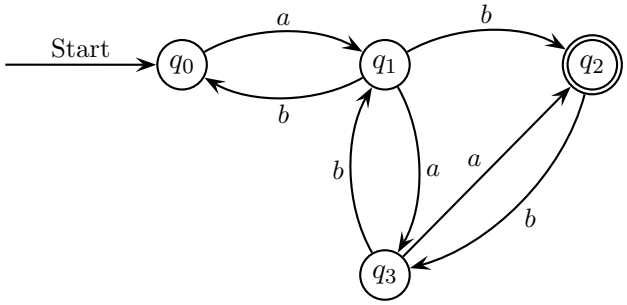
(c) [9pt] Write a regular expression r such that

$$L(r) = \{w : w \in \{a, b\}^*, n_a(w) \bmod 2 = 0 \text{ or } n_b(w) \bmod 3 = 0\}$$

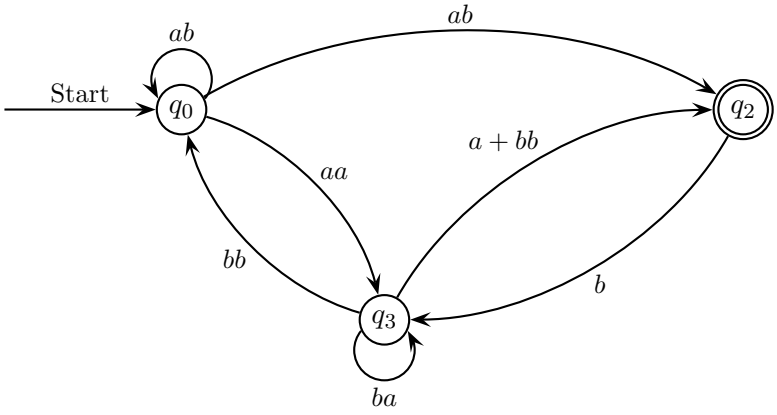
.

$$r = (b^*ab^*ab^*)^* + (a^*ba^*ba^*ba^*)^*$$

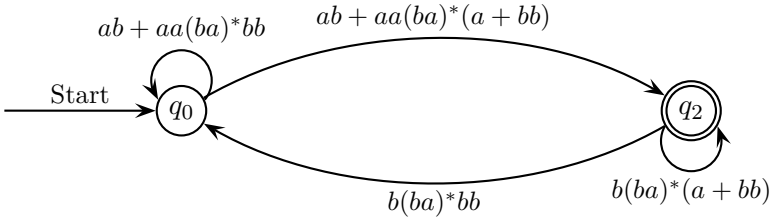
4. [8pt] Consider the following NFA N.



(a) [8pt] Draw an equivalent generalized transition graph G_1 that results by removing q_1 from N.



(b) [8pt] Draw an equivalent generalized transition graph G_2 that results by removing q_3 from G_1.



(c) [9pt] Write the regular expression that is equivalent to the generalized transition graph G_2.

There equivalent regular expression is

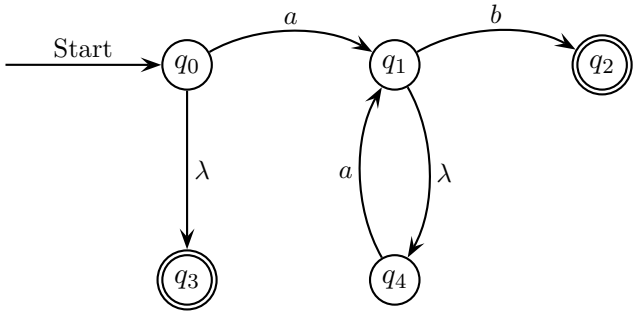
$$r = r_1^* r_2 (r_3 + r_4 r_1^* r_2)^*$$

where $r_1 = ab + aa(ba)^*bb$, $r_2 = ab + aa(ba)^*(a + bb)$, $r_3 = b(ba)^*(a + bb)$, $r_4 = b(ba)^*bb$.

5. (a) [5pt] Let DFA_L be the set of languages accepted by DFAs and NFA_L be the set of languages accepted by NFAs. Is it true that $\text{DFA}_L \subseteq \text{NFA}_L$? Give a brief argument in support of your answer.

   This is true. This relationship says that every language that can be accepted by some DFA can be accepted by some NFA. This is trivially true since a DFA is a special case of an NFA, which proves that for every DFA there is an equivalent NFA, mainly the DFA itself.

   (b) [10pt] Convert the following NFA to an equivalent DFA. Following the procedure outlined in the book or in class.



   The following DFA is equivalent to the above NFA.