**CS3331 – Assignment 1**
**due Oct. 8, 2019 (latest to submit: Oct. 11)**

1. (60pt) For each of the following languages, prove whether it is regular or not. If it is, then

   - construct a NDFSM for it
   - convert the NDFSM into a DFSM (Note that you do not have to include trap/dead states)
   - minimize the DFSM
   - convert one of the machines into a regular expression (whichever gives a simpler regular expression)

   Show your work.

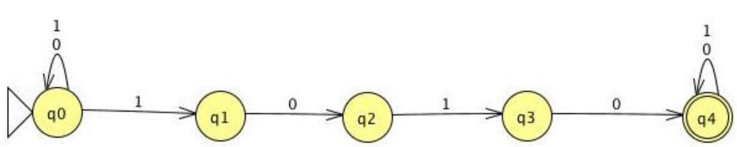
   Note 1: If you can give directly a DFSM, then you don't have to provide a NDFSM. If you provide directly the minimal DFSM, you still need to argue why it is minimal.

   Note 2: Horribly looking regular expressions from JFLAP are acceptable only when no obvious simpler ones can be found. Usually, JFLAP gives better looking regular expressions from "smaller" machines, deterministic or not.

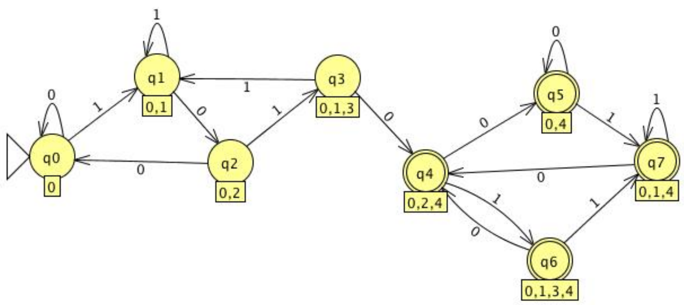
   (a) $\{w^R w w^R \mid w \in \{a,b\}^*\}$.
   (b) $\{w \in \{0,1\}^* \mid w$ has 1010 as substring$\}$.
   (c) $\{w \in \{0,1\}^* \mid w$ does not have 1010 as substring$\}$.
   (d) $\{w \in \{a,b\}^* \mid$ every $b$ in $w$ is immediately preceded and followed by $a\}$.
   (e) $\{w \in \{a,b,c\}^* \mid$ the third and second from the last characters are $b$'s$\}$.
   (f) $\{w \in \{a,b\}^* \mid (\#_a(w) + 2\#_b(w)) \equiv 0 \pmod 4\}$. ($\#_a(w)$ is the number of $a$'s in $w$).
   (g) $\{w \in \{a,b\}^* \mid \#_a(w) - 2\#_b(w) = 0\}$.
   (h) $\{w \in \Sigma^* \mid w$ is a C comments$\}$, where $\Sigma$ is the keyboard alphabet; C comments are of two types:
   `/* ... comment ... */`
   `// ... comment ... \n`

   **Solution:**

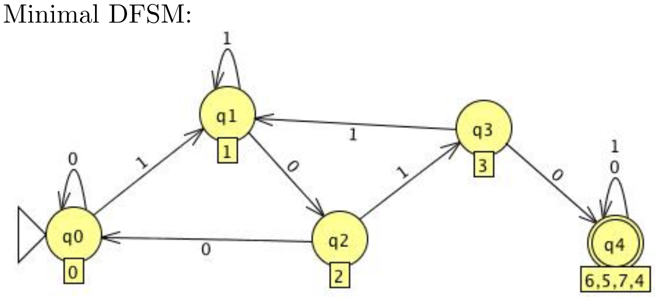
   (a) Not regular. Use pumping lemma for $a^k bba^{2k}b$.
   (b) NDFSM:




   DFSM:

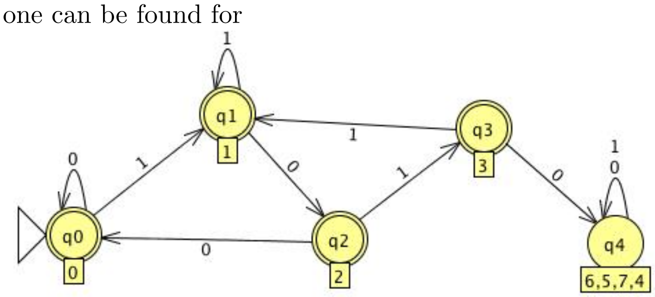



Minimal DFSM:



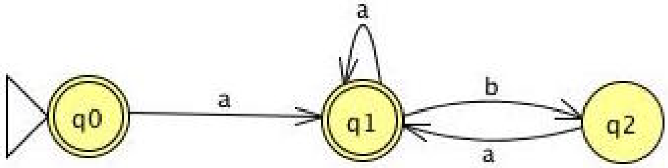

Regular expression: $(0+1)^*1010(0+1)^*$ (from NDFSM).

(c) The machines are easy; just complement the minimal DFSM from (b). This gives indeed the minimal DFSM. First, it clearly gives a DFSM. Second, it is minimal, as otherwise a smaller one can be found for



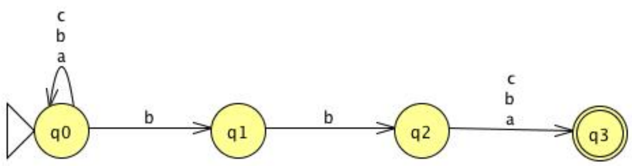

For the regular expression, JFLAP's is acceptable as there is no obvious simpler one:
$(0 + 11^*00 + 11^*01(11^*01)^*11^*00)^*(\varepsilon + 11^* + 11^*0 + 11^*01(11^*01)^*(\varepsilon + 11^* + 11^*0))$.

(d) The first machine I built turned out to be the minimal DFSM, as $q_0$ and $q_1$ clearly cannot be merged:



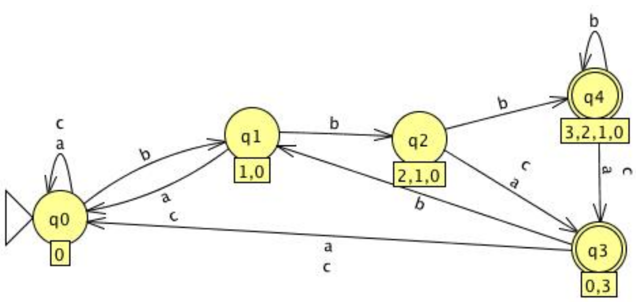
Regular expression: $\varepsilon + aa^* + aa*b(aa*b)^*aa^*$. (I skipped the steps of the algorithm. You have to show them.)

(e) NDFSM:

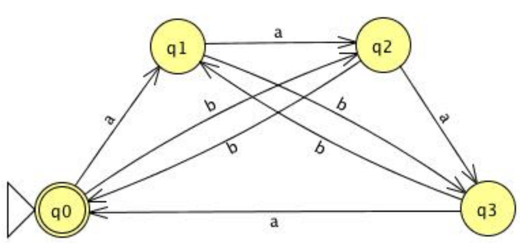



Regular expression: $(a+b+c)^*bb(a+b+c)$ (Note that JFLAP gives this from the NDFSM and something quite horrible from the DFSM.)

Minimal DFSM:

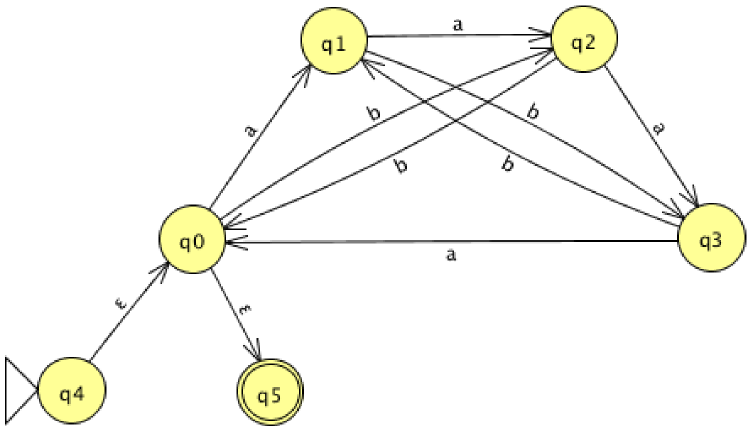



(f) The first machine I built turned out to be the minimal DFSM. The idea is essential: use one state for each possible remainder modulo 4: strings that lead the machine to $q_i$ have $\#_a(w) + 2\#_b(w)) \equiv i \pmod 4$. Each $a$ changes the remainder by 1 whereas each $b$ changes it by 2.

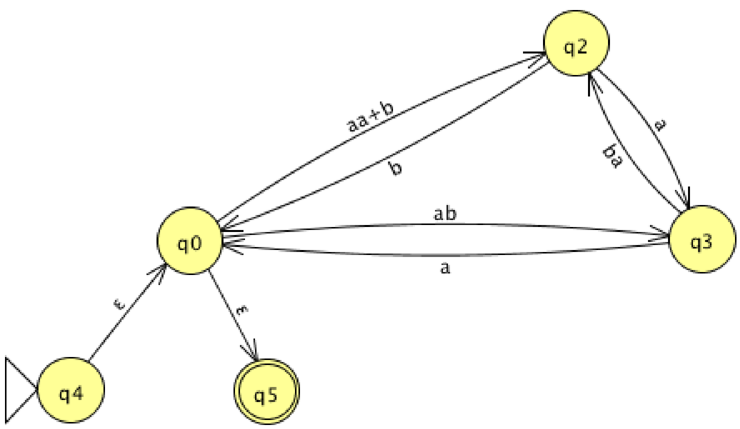


For the regular expression, we need to work.

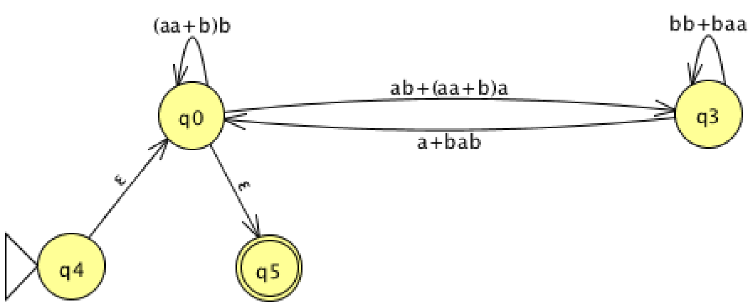
Standardize the machine (I do not add $\emptyset$-transitions for simplicity):



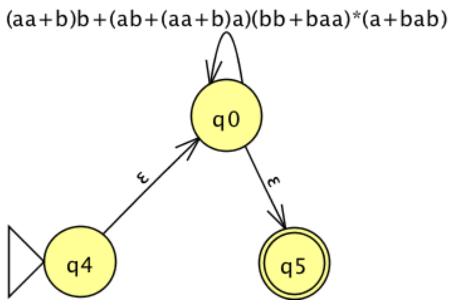

- eliminate $q_1$:



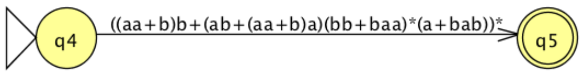

- eliminate $q_2$:




- eliminate $q_3$:



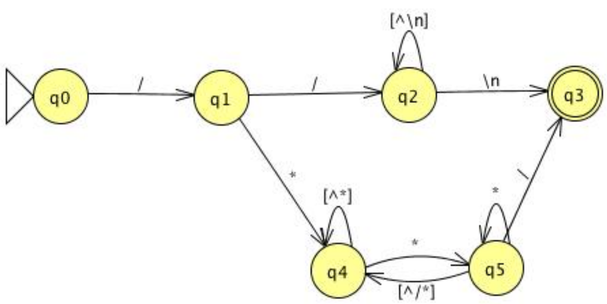

- eliminate $q_0$:




Regular expression: $((aa+b)b+(ab+(aa+b)a)(bb+baa)^*(a+bab))^*$.

(g) Not regular. Use pumping theorem for the string $a^{2k}b^k$.

(h) The first machine I built is the minimal DFSM because there is one way to do it:




The regular expression is (make distinction between the character $*$ and the operation $^*$):

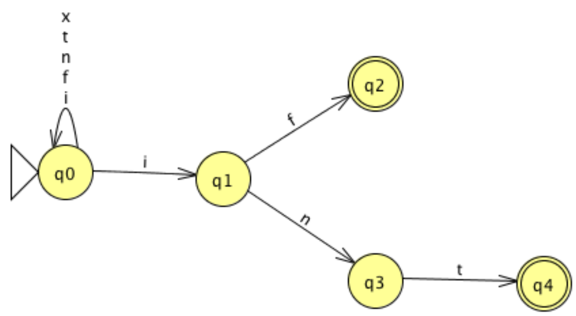
$$/ * (\Sigma - \{*\})^* * (* + (\Sigma - \{\backslash, *\})(\Sigma - \{*\})^* *)^* / + //(\Sigma - \{\backslash n\})^* \backslash n$$

(I skipped the steps of the algorithm. You have to show them.)

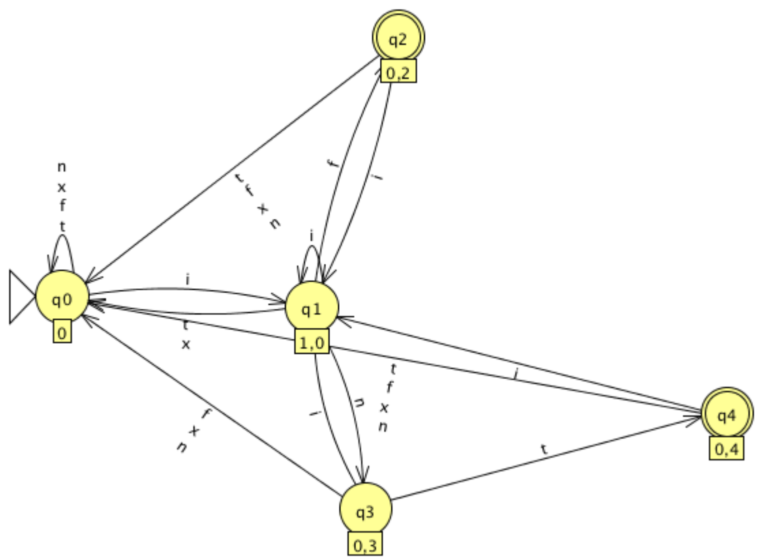
2. (20pt) Recall the Multi-Pattern Searching problem is: Given several patterns $p_1, p_2, \ldots, p_k \in \Sigma^*$ and a text $T \in \Sigma^*$, find all occurrences of $p_i$'s in $T$. It can be solved in linear time by constructing a DFSM for the regular expression $\Sigma^*(p_1 \cup p_2 \cup \cdots \cup p_k)$ and then run the text $T$ through it; every time the machine is in an accepting state, we report the end of an occurrence of the patters.

   Assume $\Sigma = \{\mathtt{i}, \mathtt{f}, \mathtt{n}, \mathtt{t}, \mathtt{x}\}$ ($\mathtt{x}$ stands for any character different from $\mathtt{i}, \mathtt{f}, \mathtt{n}, \mathtt{t}$.) Construct the minimal DFSM to solve the multi-pattern searching problem for the patterns $p_1 = \mathtt{if}, p_2 = \mathtt{int}$. (This is used for keyword identification.) Show your work. You are allowed to use Thomson's construction or directly build an NDFSM.

   **Solution:** NDFSM (much smaller than Thompson's, yet intuitively clear):




DFSM (subset construction):



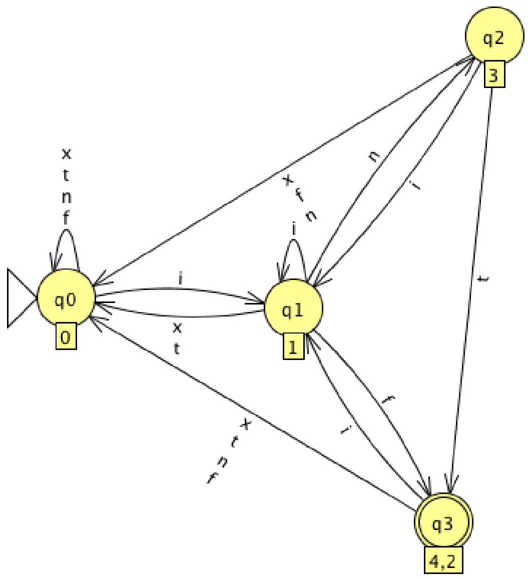

Minimal DFSM; successive state partitioning:

$\{[0,1,3],[2,4]\}$ - use `t`: $3 \xrightarrow{t} 4$, $0 \xrightarrow{t} 0$, $1 \xrightarrow{t} 0$

$\{[0,1],[3],[2,4]\}$ - use `n`: $1 \xrightarrow{n} 3$, $0 \xrightarrow{n} 0$

$\{[0],[1],[3],[2,4]\}$

States 2 and 4 have the same behaviour, they cannot be split so they will be merged:




3. (20pt) Show that the following problem is decidable:

   Given $\Sigma = \{a, b\}$ and $\alpha$ a regular expression, is it true that $L(\alpha)$ contains only non-empty even-length strings in $\Sigma^*$ and no string consisting only of $b$'s?

   You are allowed to use any of the following:

   – closure properties: union, concatenation, Kleene star, complement, intersection, difference

   – conversion algorithms between DFSM, NDFSM, regular expressions, and regular grammars (see the last slide of Ch.7: Conversions)

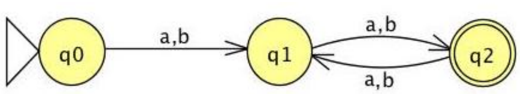
   – decision algorithms: membership, emptiness, finiteness, totality, equivalence, minimality.

   Explain which closure property and algorithm you have used. Any other construction or algorithm should be described in the assignment.

   **Solution:**

   (a) Build a DFSM $M_\alpha$ for $L(\alpha)$ (regular expression to NDFSM, NDFSM to DFSM)

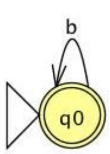
   (b) Build a DFSM, $M_1$, for the language $\{w \in \Sigma^* \mid w \neq \varepsilon, |w| \text{ even}\}$:


   (c) Build a DFSM, $M_2$, for the language $L(M_\alpha) - L(M_1)$ (difference)

   (d) If $L(M_2) \neq \emptyset$, then return **no** (emptiness)

   (e) Build a DFSM, $M_2$, for the language $b^*$:


   (f) Build a DFSM, $M_3$, for the language $L(M_\alpha) \cap L(M_2)$ (intersection)

   (g) If $L(M_3) \neq \emptyset$, then return **no** (emptiness)

(h) return **yes**

**Note:** Submit your solution as a single pdf file on `owl.uwo.ca`. Solutions should be typed but high quality hand written solutions are acceptable. Make sure you submit everything as a single pdf file.

**Note:** You are allowed to use JFLAP to solve the assignment. But remember that JFLAP will not be allowed during the midterm exam!