CS131 Compilers: Writing Assignment 1
Due Tuesday, March 27, 2018 at 23:55

Zhiqiang Xie - 77892769

This assignment asks you to prepare written answers to questions on regular languages, finite automata, and lexical analysis. Each of the questions has a short answer. You may discuss this assignment with other students and work on the problems together. However, your write-up should be your own individual work and you should indicate in your submission who you worked with, if applicable. You should use the Latex template provided at the course web site to write your solution and use the *tikz* package to draw automata.

I worked with: (Name,ID), (Name,ID)...

1. ($2 \times 3 = 6$ **pts**) For each of the follow prompts, write any non-empty sentence:

   (a) Name one reason why you would like to learn in this class.
   I want to have a comprehensive understanding towards how a program worked from coding level to hardware level. So that I can find my interest from the whole computer science.
   Besides, I'm interested in some topics will be covered in this course: intermediate representation, some basic ideas to process formal language (since I have some research experience in data-driven processing approach, I think there are some good ideas can be found from the classical methods)

   (b) Write a question you would like the professor to answer on any topic, from personal opinions to the class material.
   What do you think about the key ideas in compilers?
   We all know there are some great ideas like cache, intermediate layer in many fields of computer science. what's that in compilers?

   (c) What do you expect from this class.
   More about IR and some more general approach instead of focusing on string and tokens processing too much. Or there's some great general ideas hidden in it, that's the question above.

2. ($2 \times 4 = 8$ **pts**) Write regular expressions for the following languages over the alphabet $\Sigma = \{0, 1\}$:

   (a) L_1: The set of all finite strings containing only three $1's$.

   $$L_1 = (0^*1)\{3\}0^*$$

   (b) L_2: The set of all finite strings containing at least three $1's$ and the third character from beginning is 1.

   $$L_2 = 111[01]^*|(10|01)1(0^*10^*)^+|001(0^*1)\{2,\}0^*$$

   (c) L_3: The set of all finite strings containing at most three $0's$ and at least two $1's$.

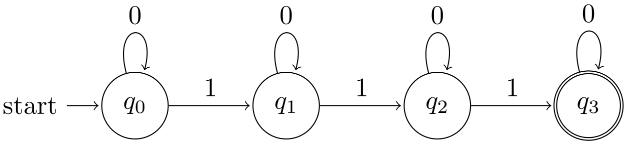
   $$\begin{aligned} L_3 =&0001\{2,\}|00101^+|001\{2,\}(01^*)?|01001^+|0101^+(01^*)?|01\{2,\}(01^*)\{0,2\}|10001^+ \\ &|1001^+(01^*)?|101^+(01^*)\{0,2\}|1\{2,\}(01^*)\{0,3\} \end{aligned}$$

   (d) L_4: The set of all finite strings which does not contain subsequence 100.

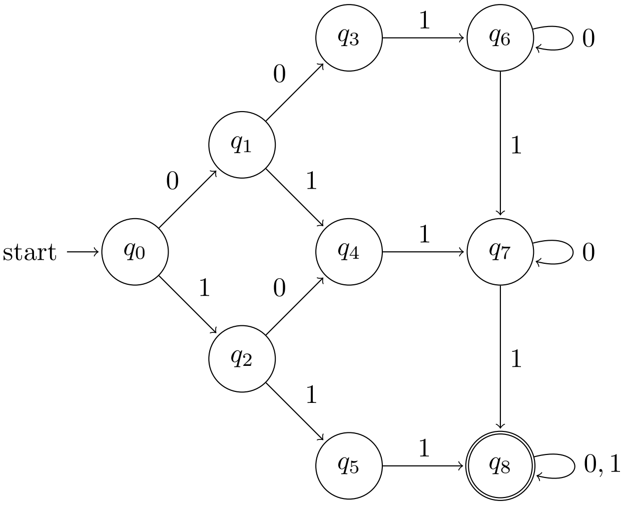
   $$L_4 = 0^*(1^+0)^*1^*$$

3. ($2 \times 4 = 8$ **pts**) Draw DFA's for each of the languages L_1, L_2, L_3 and L_4 from Question 1.

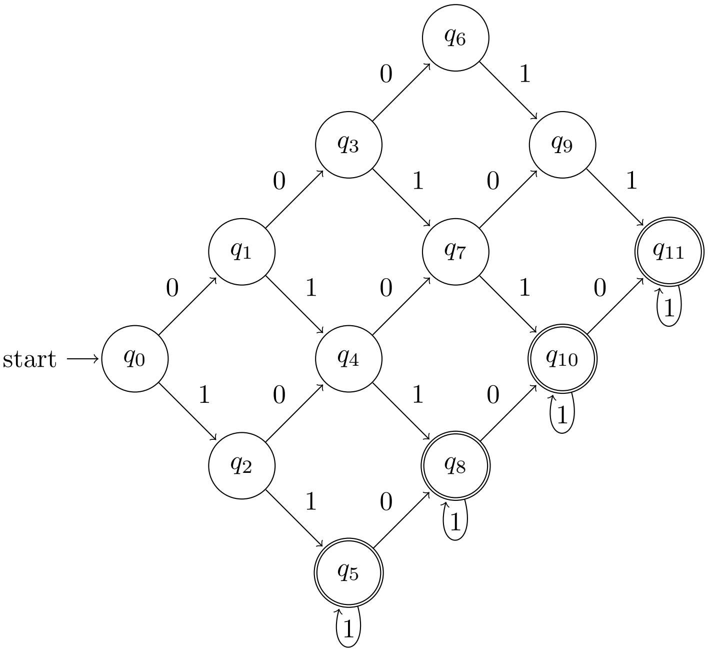
(a) L_1.



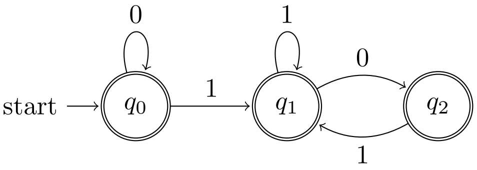
(b) L_2.



(c) L_3.

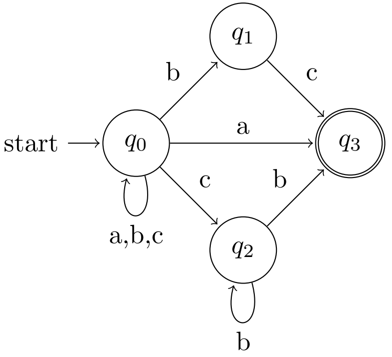


(d) L_4.

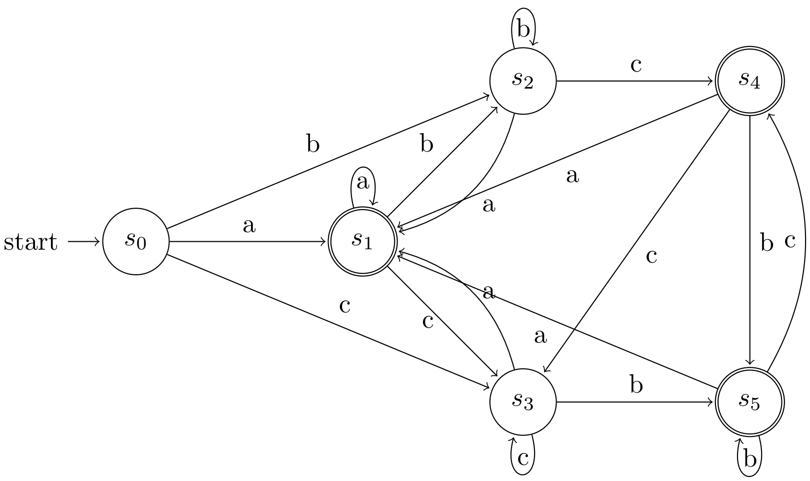


4. ($5 \times 3 = 15$ **pts**) Using the techniques covered in class, transform the following NFAs with ϵ-transitions over the given alphabet Σ into DFAs. Note that a DFA must have a transition defined for every state and symbol pair, whereas a NFA need not. You must take this fact into account for your transformations. Hint: Is there a subset of states the NFA transitions to when fed a symbol for which the set of current states has no explicit transition?

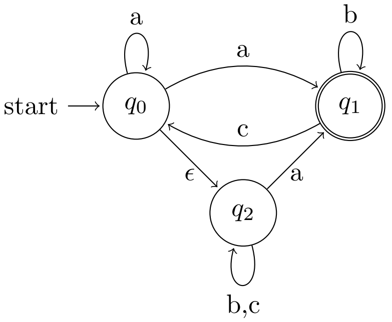
(a) Original NFA, $\Sigma = \{a, b, c\}$:



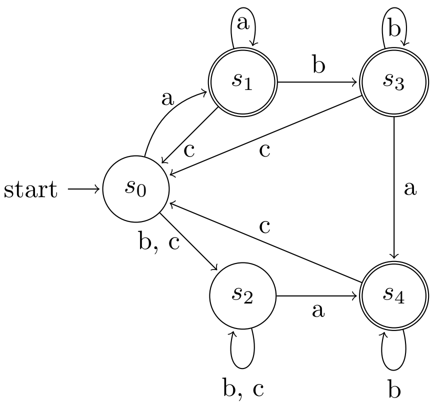
DFA:



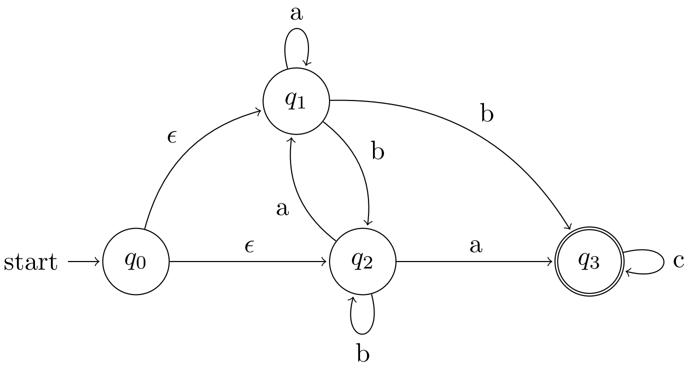
(b) Original NFA, $\Sigma = \{a, b, c\}$:



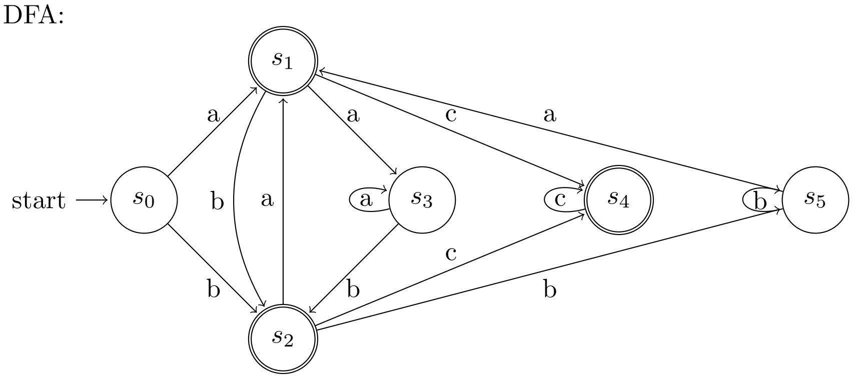
DFA:



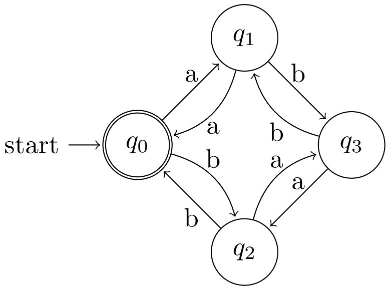
(c) Original NFA, $\Sigma = \{a, b, c\}$:



DFA:



5. (13 **pts**) Draw the NFA for the set of all strings over the alphabet $\Sigma = \{a, b\}$, where both a and b occur even times. Examples of strings that should be accepted by this NFA: abbabbbbaa, baabaaabaaaaab. Examples of strings that should **not** be accepted: ababb, abbbabba.



where q_0 denotes even a and even b, q_1 denotes odd a and even b, q_2 denotes even a and odd b, q_2 denotes odd a and odd b,

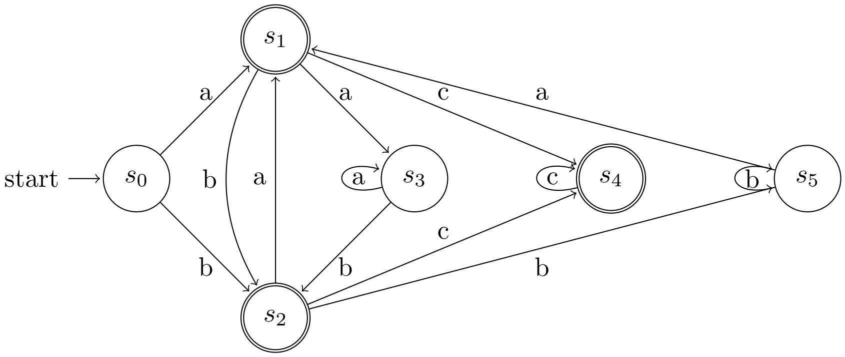
6. (5 **pts**) Consider the following tokens and their associated regular expressions, given as a **flex** scanner specification:

```
%%
(if)                     {printf("IF");}
(0*1|1*0)                {printf("PS");}
[0-9]+                   {printf("NUM");}
[a-zA-Z0-9]+             {printf("ID");}
[ ]                      {}
```

Give an input to this scanner such that the output string is $(\texttt{NUM}^2\texttt{IF}^2\texttt{ID}^3\texttt{PS})^2$, where $\texttt{A}^{\texttt{i}}$ denotes `A` repeated `i` times. (And, of course, the parentheses are not part of the output.) You may use similar shorthand notation in your answer.

$$((2\ )^2(if\ )^2(a\ )^3 1\ )^2$$

7. (5 **pts**) Draw the minimal DFA of the DFA constructed in Question 4(c).



It's already a minimal DFA.