Scott Hansen
Professor Alan Labouseur

Lab 2

Crafting a compiler

3.3: Write regular expressions that define the strings recognized by the FAs in Figure 3.33 on page 107.

Exercises

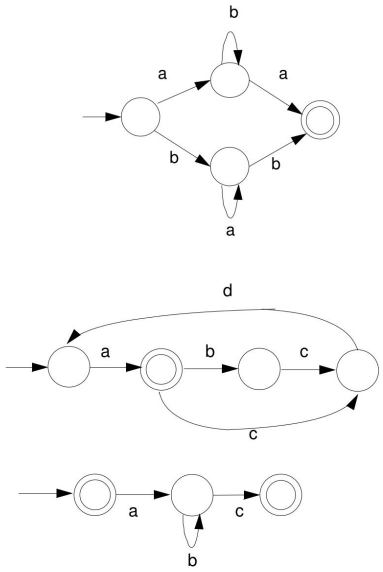
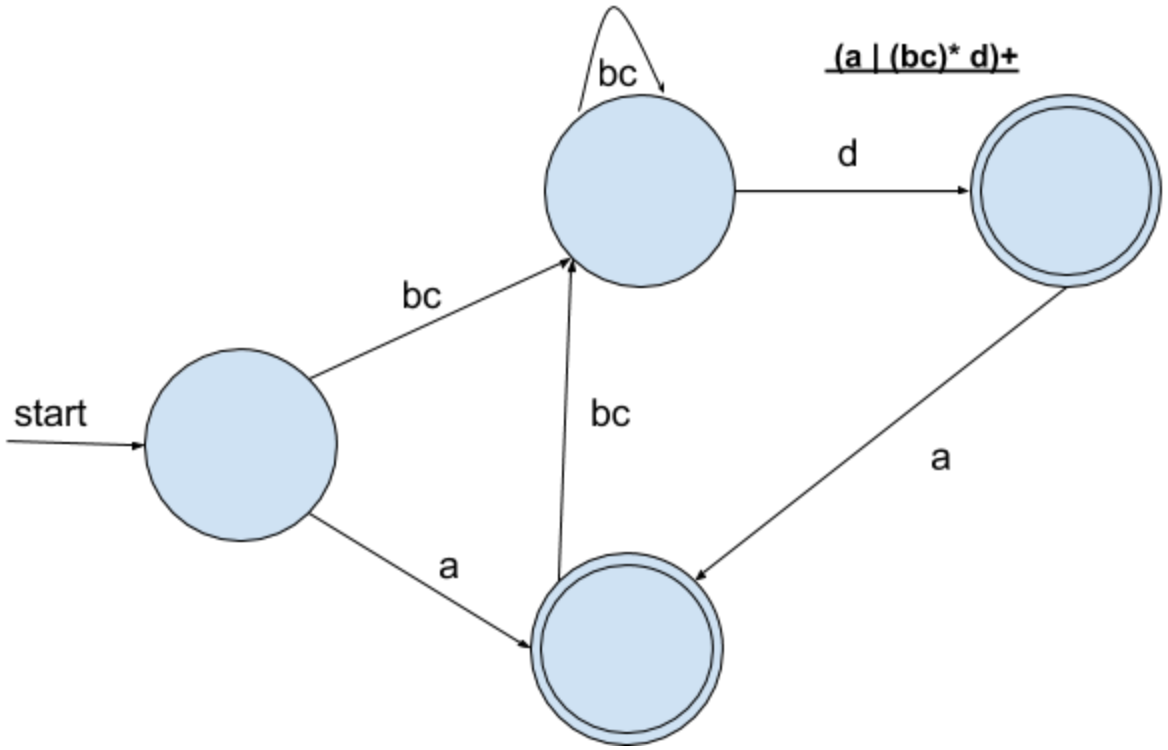


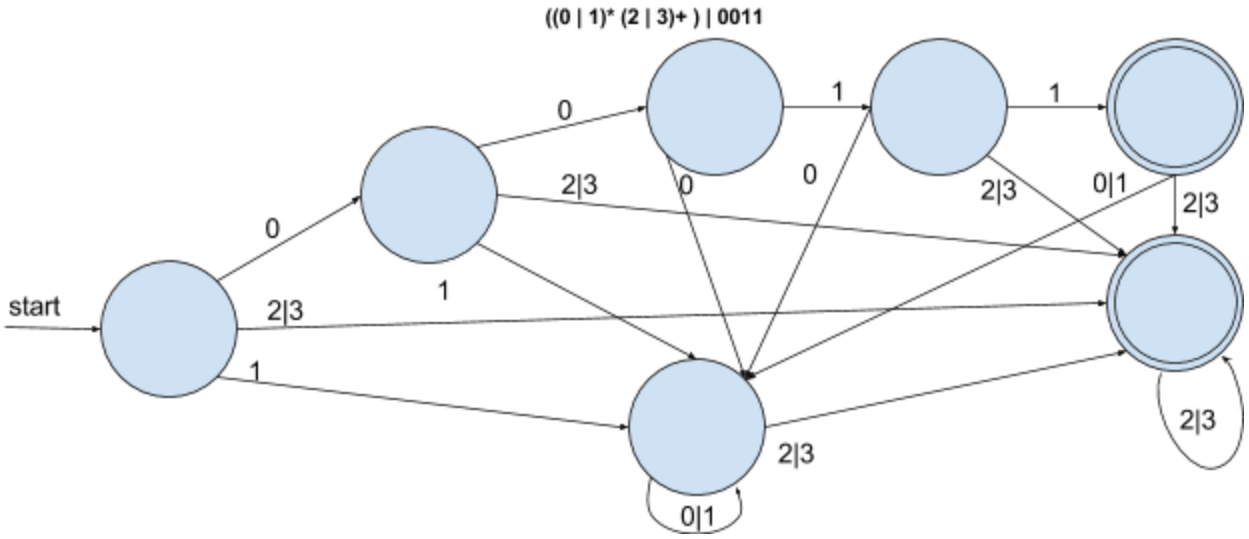
Figure 3.33: FA for Exercise 3.

1. (a(b)* a | b(a)* b)
2. (a | ((bc) | c) da)
3. (ε | (a (b)* c)

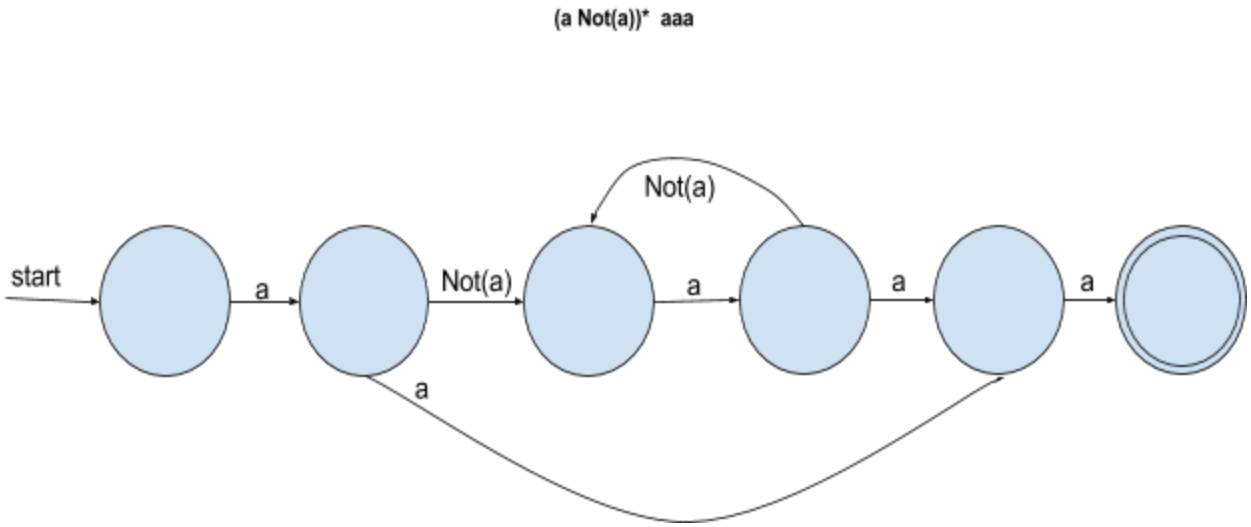
3.4: Write DFAs that recognize the tokens defined by the following regular expressions:
(a) (a | (bc)* d)+



(b) ((0 | 1)* (2 | 3)+) | 0011

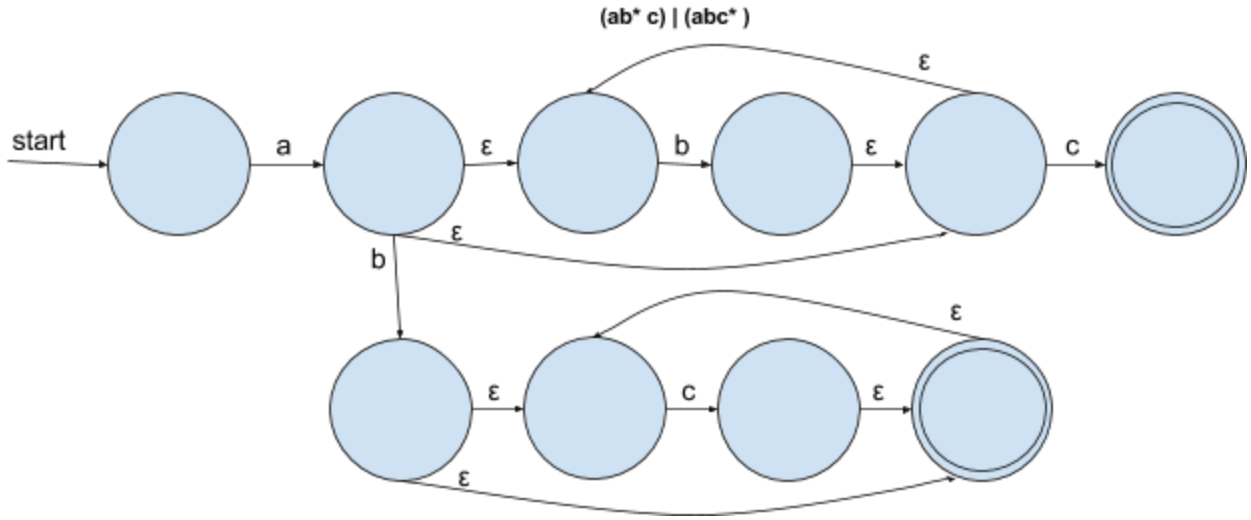


(c) (a Not(a))* aaa

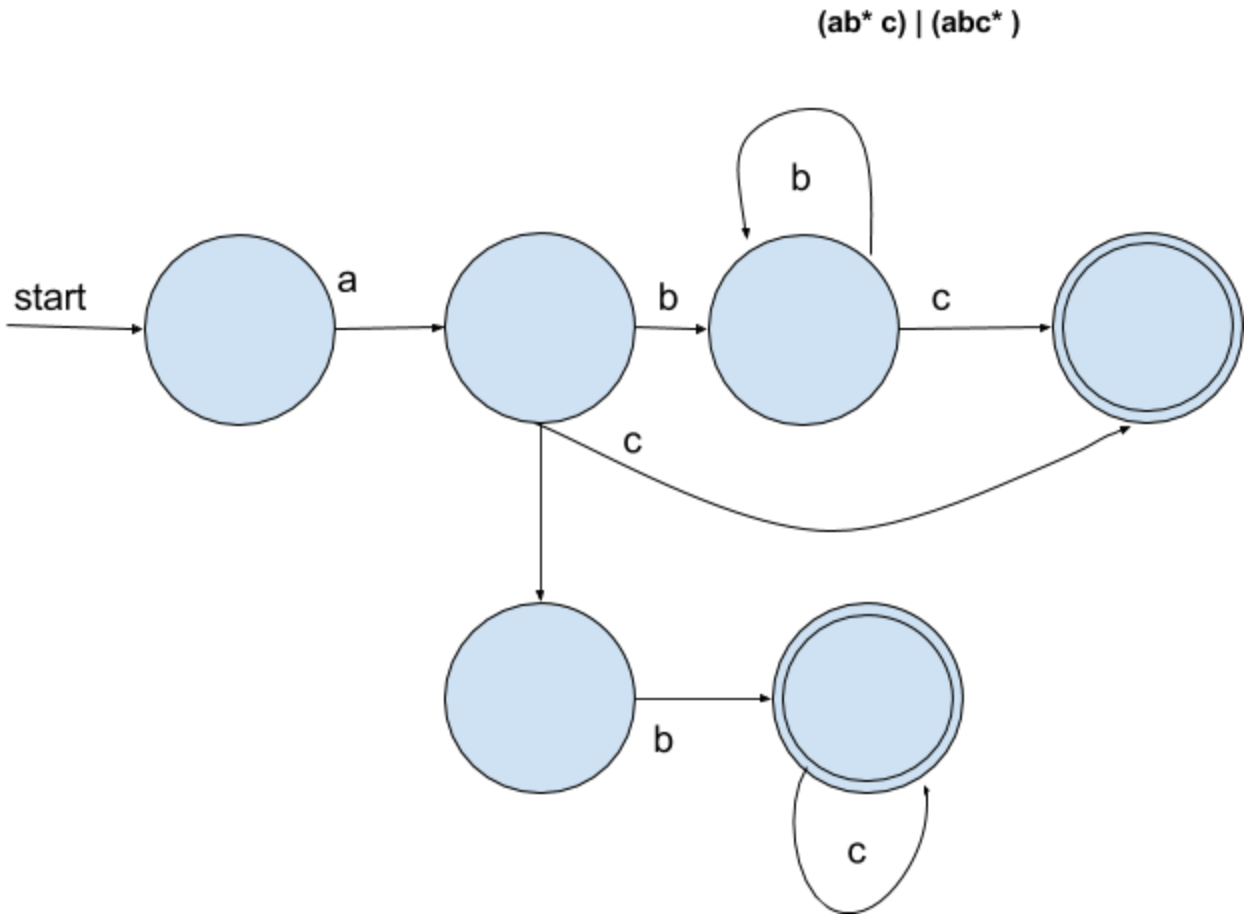


3.15: Show the NFA that would be created for the following expression using the techniques of Section 3.8:

(ab* c) | (abc*)



Using *MakeDeterministic*, translate the NFA into a DFA. Using the techniques of Section 3.8.3, optimize the DFA you created into a minimal state equivalent.



Dragon Book
3.3.4: : Most languages are case sensitive, so keywords can be written only one way, and the regular expressions describing their lexeme is very simple. However, some languages, like SQL, are case insensitive, so a keyword can be written either in lowercase or in uppercase, or in any mixture of cases. Thus, the SQL keyword SELECT can also be written s e l e c t , Select, or sElEcT, for instance. Show how to write a regular expression for a keyword in a case-insensitive language. Illustrate the idea by writing the expression for "select" in SQL.

[S | s] [E | e] [L | l] [E | e] [C | c] [T | t]