Brendon Boldt
February 7, 2016
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.

```
1. (ab*a)|(ba*b)
2. a(b?cda)*
3. (ab*c)?
```

Where "?" is zero or one occurrences of the preceding character or group

3.5 Write a regular expression that defines a C-like, fixed-decimal literal with no superfluous leading or trailing zeros. That is, 0.0, 123.01, and 123005.0 are legal, but 00.0, 001.000, and 0023455.1000 are illegal.

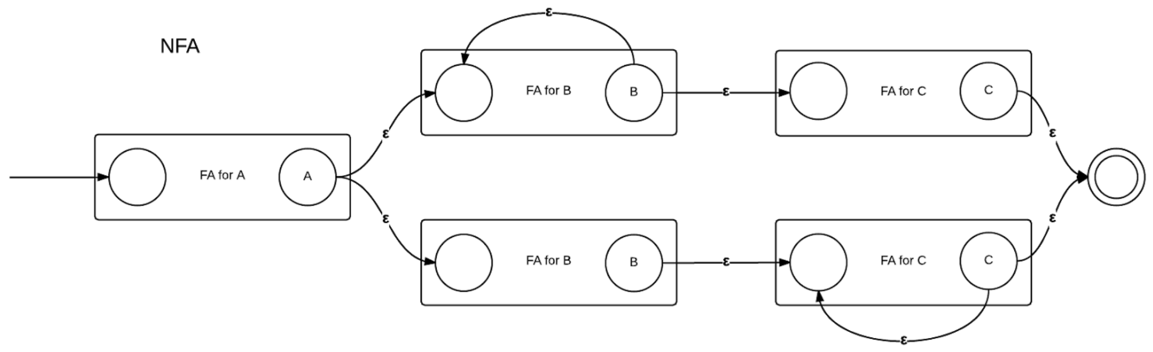
```
\b(0|([1-9][0-9]*))\.[0-9][1-9]*
```

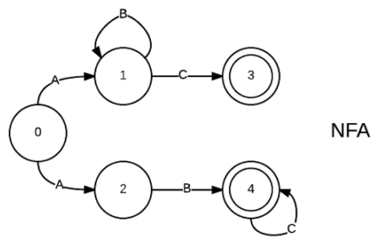
"\b" denotes word boundary, so the above matches "0.0" but not "b0.0" or "0.0c"

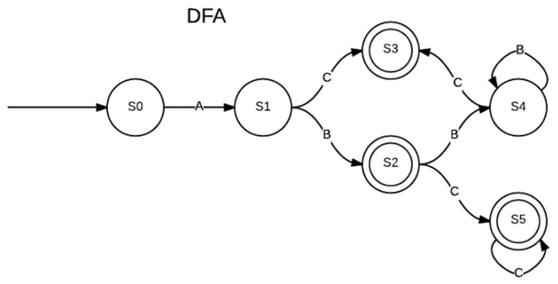
3.15 Show the NFA that would be created for the following expression using the techniques of Section 3.8

```
(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.







{0} = S0
S0, A = {1,2} = S1
S1, B = {1,4} = S2
S1, C = {3} = S3
S2, B = {1} = S4
S2, C = {3,4} = S5
S4, B = {1}
S4, C = {3}
S5, C = {4} = S6
S6, C = {4}

Dragon

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 select, 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)
```