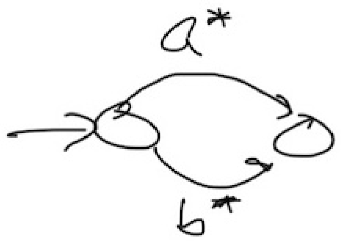
CSC 135 - Ungraded Homework Solutions

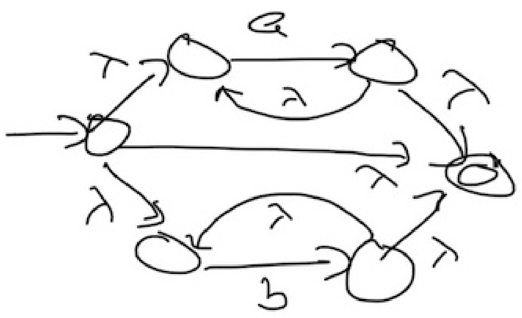
Please let me know if you find any errors.

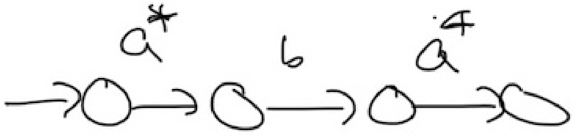
1. The key question is always "If I could be in any of NFA states X and I had to consume input char Y, which states could I be in after".

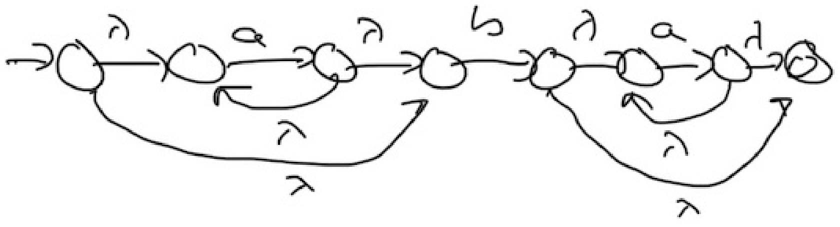
    ```
    #states
    a
    ab
    ac
    abd
    #initial
    a
    #accepting
    abd
    #alphabet
    0
    1
    #transitions
    a:0>ab
    a:1>a
    ab:0>ab
    ab:1>ac
    ac:0>abd
    ac:1>a
    abd:0>ab
    abd:1>ac
    ```

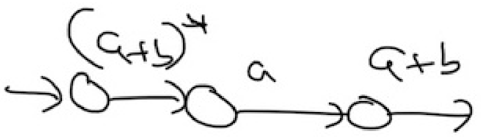
2. Here are the three NFAs. Each has two drawings. The first is a drawing early in the decomposition, and the second is the final.

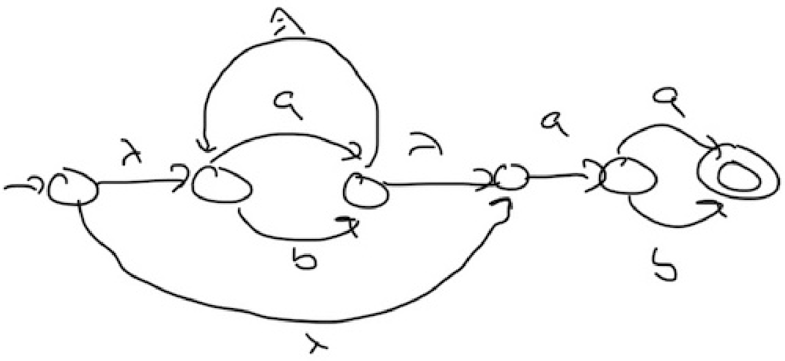












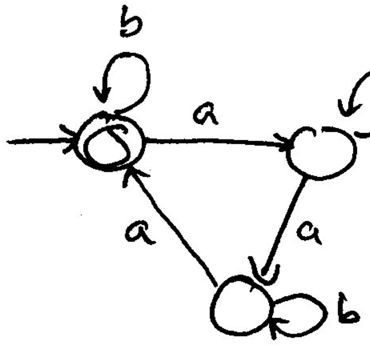
3. Use the pumping lemma to argue that $\{1^n2^n3^n \mid n \geq 0\}$ is not regular.

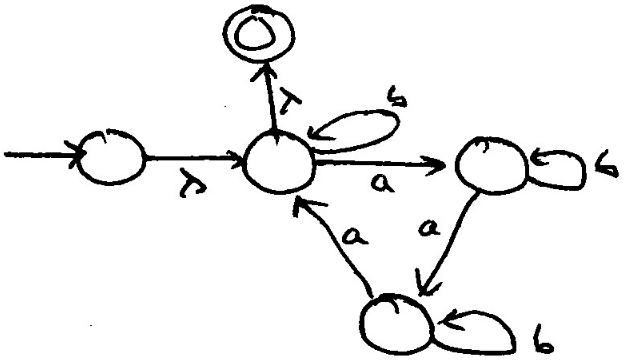
   For purposes of contradiction assume the language $L = \{1^n2^n3^n \mid n \geq 0\}$ is regular.
   Because L is regular there must be a pumping length p.
   Consider the string $1^p2^p3^p$ which is in L.
   The pumping lemma says there exists $xyz = 1^p2^p3^p$ where $|xy| \leq p$, $|y| > 0$, and xy^iz is in L for all $i \geq 0$.
   Because $1^p2^p3^p$ begins with p 1s, x and y must be all 1s. Since $|y| > 0$, xz will have fewer 1s than 2s or 3s and so cannot be in L. This contradicts that the pumping lemma says xz is in L.

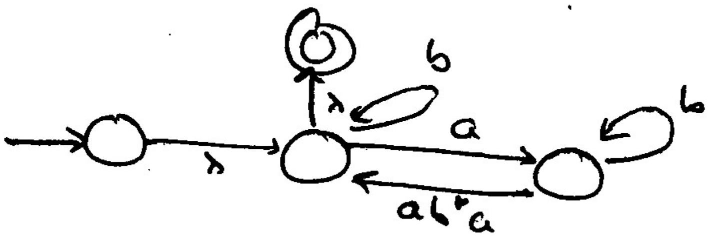
   Remember, we get to pick the string, but it must be in L, must be at least length p, and must be easily shown that pumping it results in a string not in L (which contradicts the pumping lemma which says that it should be in L). We don't get to assign xyz, but a careful choice of string will force x and y to have some structure.

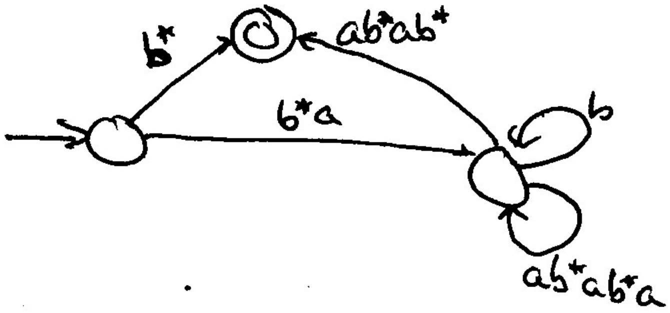
4. See diagrams below. The top-left shows the new start and final states added. The top-middle-left removes the bottom state. The top-middle-right removes the middle state. Since that state had 2 arrows in and 2 arrows out, 4 new arrows replace the 4 in-out combinations. The bottom-middle-left combines parallel edges. The bottom-middle-right removes the right state. And, the final digram combines parallel edges and yields the result.

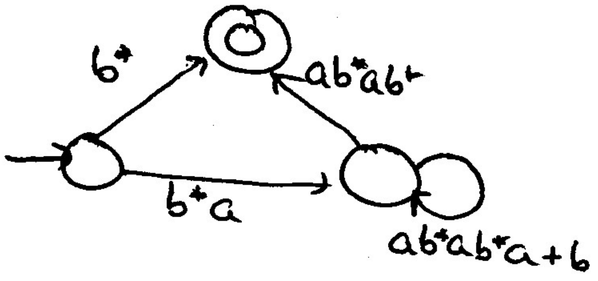
There are multiple possible results to this problem because the candidate states can be removed in any order. For example, if the right state had been removed from the top-middle-left figure, the result would have been (b+ab*ab*a)*, which happens to be much simpler.

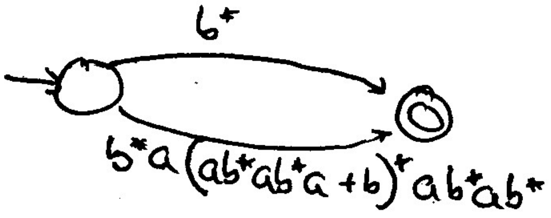


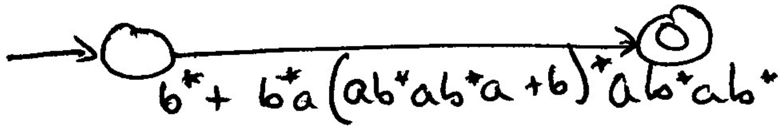




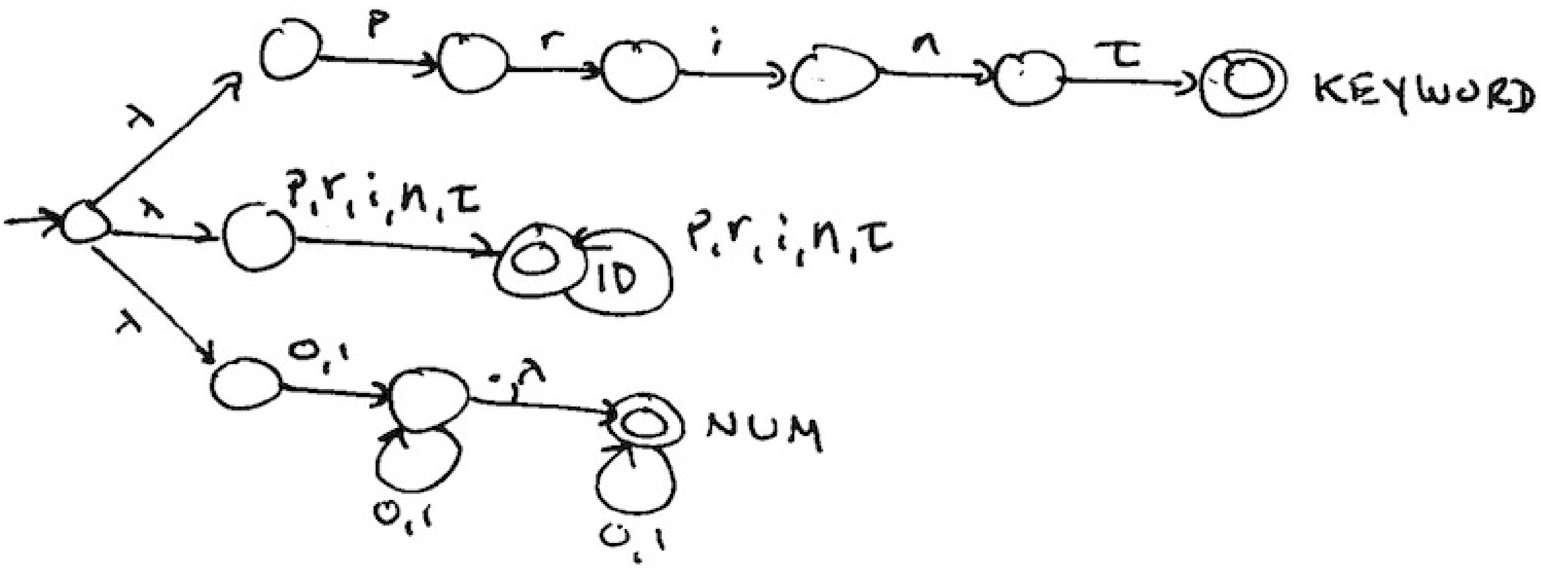








5.



6. Keep doing the following until all the input is gone: (i) skip whitespace, (ii) if text remains, grab (and delete) longest prefix that matches a regular expression, and (iii) return the string that matched and the highest token type for that prefix. Here is the sample input broken into tokens. Note that when the prefix is "print", it matches both ID and KEYWORD, but KEYWORD is specified first and therefore is the type specified.

```
("print", KEYWORD)
("01", NUM)
("10", NUM)
("printt", ID)
("0.1", NUM)
("printprint", ID)
("pri", ID)
("nt", ID)
("i", ID)
```