Natural Language Processing Homework 6 README

Zixuan Wang, Mou Zhang

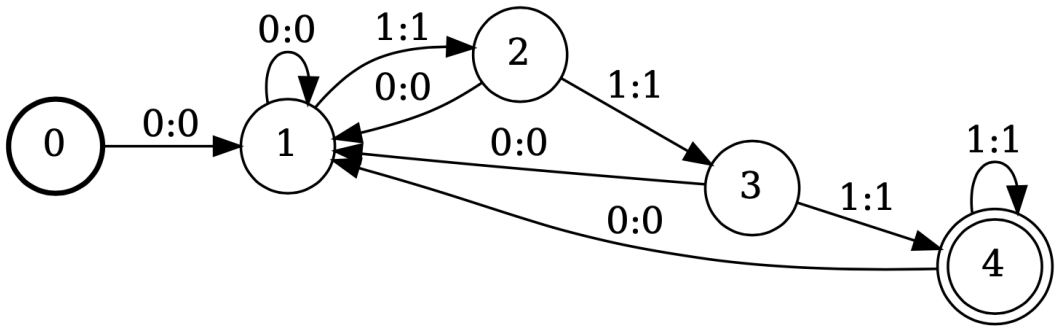
November 25, 2019

1 Question 1

(a) i If the typed string matches the FST, it will print itself. If not, it will print "Rewrite failed". .

ii First will accept any binary string that starts with 1 zero and ends with 3 ones. You can put any numbers of zeros or ones between the 1 zero and 3 ones.
Basic string FSTs are defined by text enclosed by double quotes. 0s and 1s should be enclosed by double quote because they are input strings instead of numerical values.

iii There are 5 states and 9 arcs.



(b) i In this step, we added Second and Disagreements to the binary.grm.

ii Disagreements should not accept any string and any input string will prompt "Rewrite failed". We can conclude this by showing that the state and arc of Disagreements are both 0, which indicates that it doesn't accept any input string.

```
fst type                                vector
arc type                                standard
input symbol table                      **Byte symbols
output symbol table                     **Byte symbols
# of states                             0
# of arcs                              0
initial state                           -1
# of final states                      0
# of input/output epsilons             0
# of input epsilons                    0
# of output epsilons                   0
input label multiplicity                0
output label multiplicity               0
# of accessible states                  0
# of coaccessible states                0
# of connected states                   0
# of connected components               0
# of strongly conn components           0
input matcher                           y
output matcher                          y
input lookahead                         n
output lookahead                        n
expanded                                y
mutable                                 y
error                                   n
acceptor                                y
input deterministic                     y
output deterministic                    y
input/output epsilons                   n
input epsilons                          n
output epsilons                         n
input label sorted                      y
output label sorted                     y
weighted                                n
cyclic                                  n
cyclic at initial state                 n
top sorted                              y
accessible                              y
coaccessible                            y
string                                  y
weighted cycles                         n
```

(c) i Fist will have 20 states and 25 arcs. Second will have 13 states and 16 arcs. Disagreement will have 88 states and 113 arcs.

ii Because in the expression, it has (First - Second) and (Second - First), there should be 2 branches corresponding to those 2 expressions in order to calculate separately. In the end, the expression Union those two branches together, which is why they merged in the end.

iii The result should be the same with the optimized version. Because the only difference between optimized and unoptimized is the ϵ transaction. This transaction will not make any change to the result of the output because people cannot type ϵ in the query.

iv We get 0 state and 0 arc for optimizing Disagreements which is the same as doing optimize after optimize First and Second. The reason why we get this because optimize works really well in reducing all the states. When no possible final state exists, it will generate 0 state and 0 arc in the end.

2 Question 2

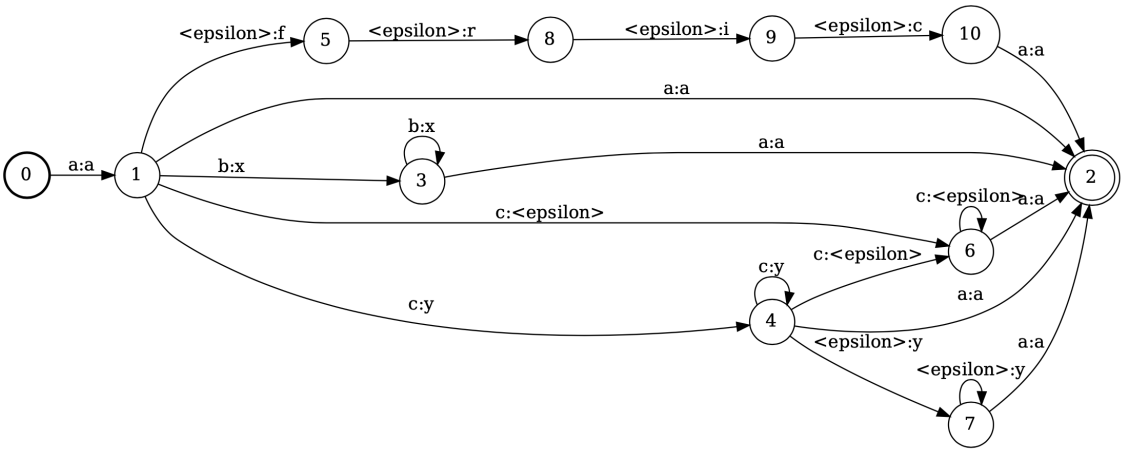
3 Question 3

(a) The input language should be a(b*|c+| ϵ)a

(b) 0 output: zzzzz
1 output: aba
2 output : aa
zz more than 2 outputs: aca

(c) We don't need to answer this question.

(d) Yes, it consists with our answer above. There are 11 states and 20 arcs in the optimized version of Cross.



4 Question 4

(a) Please see BitFlip1 in rewrite.grm

(b) Please see BitFlip2 in rewrite.grm

(c) We don't think ϵ is a valid binary number. We treat ϵ as neither an even number nor an odd number because in real life it is neither of them.

(d) Please see Parity2 in rewrite.grm

(e) If we use CDRewrite, it will treat ϵ as an empty string because Bit* would contain empty string and it is between [BOS] and [EOS]. It is a valid input and produce an empty string as output.

(f) This FST would only accept 2 inputs: "0" and "1". If you type "0", it will generate many binary even numbers and if you type "1", it will generate many binary odd numbers. If you type ϵ, it will treat it as an empty string and produce an output as empty string.

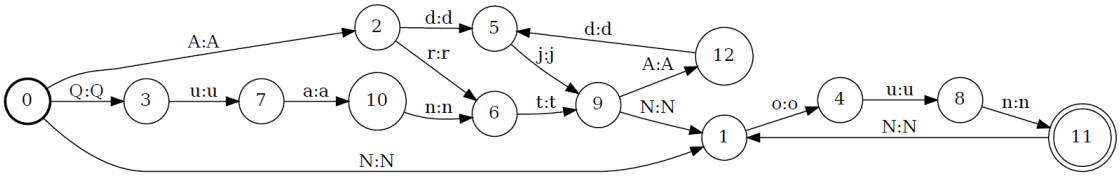
(g) Please see Split in rewrite.grm

(h) Please see SplitThree in rewrite.grm

5 Question 5

(a) i You should put double quotation marks on those tags. The fixed version should be the following one: export NP = Optimize [("Art"|"Quant")? ("Adj")* ("Noun")+];
It will accept strings like "ArtAdjNoun", "QuantAdjNoun" and "AdjNounNoun" etc.

ii There are 13 states and 17 arcs for NP.fst. The machine use every single character as the input between the states and reuse the same character for different word. For exmaple, "t" is used for both "Art" and "Quant".



(b)

(c) i This composition is taking everything that satisfies NP and use it to do MakeNmod to get the result. If the initial input is not NP, it will fail the transducer.

ii The result of ArtAdjNounNounNoun is ArtAdjNmodNmodNoun
The input AdjNounNounNounVerb will fail the transducer.

iii There are 16 states and 20 arcs in TransformNP. There are 25 states and 40 arcs in MakeNmod.

iv The topology of TransformNP is pretty similar to the topology of NP. The only difference is that in TransformNP, there is one more step to change every non-last Noun to Nmod. It is like a loop before reaching the final Noun.

(d) Brackets1 will show results which allow to leave some replaceable substrings unreplaced, which would usually have more results in the end.
Brackets2 has a obligatory ('obl') mode which means unreplaced regions may not contain any more replaceable substrings, which would usually have less results in the end.

(e) Please see BracketTransform in chunker.grm

(f) Please see BracketResults in chunker.grm

(g) Please see TranformNP2 in chunker.grm

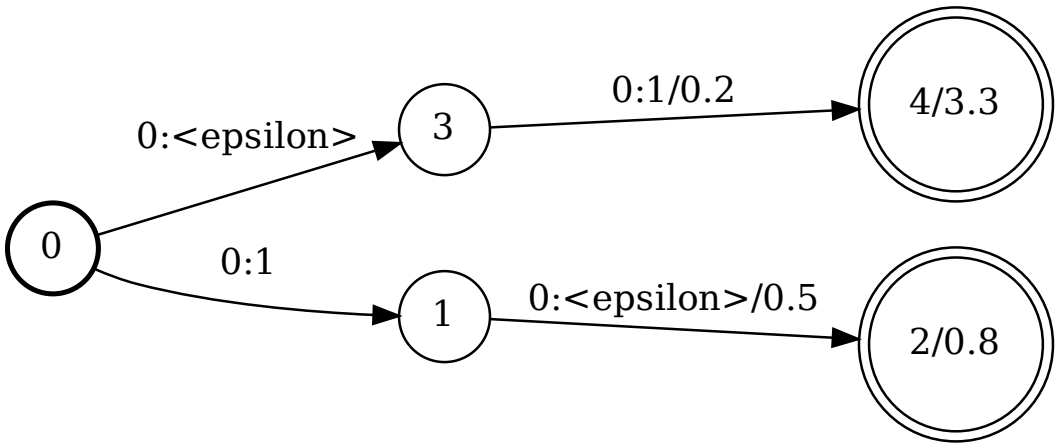
6 Question 6

(a) i The minimum-weight string accepted by this FSA is ZeroBitOne and its weight is 1 + 0.2 + 0.5 = 1.7.

ii The minimum-weight pair of strings should also be ZeroBitOne and its weight should be 1 + 0.2 + 0.75 = 1.95.

(b) Please see WFlip in Binary.grm

(c) i We used (x,y) = (00, 1) as an exmaple. Out FST only takes 00 as input and 1 as output. We randomly assign different weights to those two different paths and finally get the minimized result. The weight of the first paths is 3.5 and the weight of the second path 1.3. Please see the diagram for more information.



7 Question 7

8 Question 8

9 Question 9