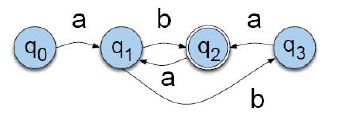
**Homework 2**

1. Regular Expression for given NFSA:



**a(baa|ba)\*(ba|b)**

1. Bigram Probabilities:

**sentence: The president wants to control the board 's control**

1. No Smoothing

P(president|The) = 0.0

P(wants|president) = 0.0

P(to|wants) = 0.5

P(control|to) = 0.0015408320493066256

P(the|control) = 0.0

P(board|the) = 0.1006993006993007

P('s|board) = 0.04644808743169399

P(control|'s) = 0.0

P(sentence) = product of all the above mentioned values

P(sentence) = 0.0

1. Add one Smoothing

P(president|The) = 0.0001736714136853074

P(wants|president) = 0.00017556179775280898

P(to|wants) = 0.0005347593582887701

P(control|to) = 0.000319744204636291

P(the|control) = 0.0001779676098949991

P(board|the) = 0.02060830017055145

P('s|board) = 0.003014065639651708

P(control|'s) = 0.00016691704223001168

P(sentence) = product of all the above mentioned values

P(sentence) = 9.619576655357422e-27

1. Good Turing Discounting based Smoothing

P(president|The) = 0.5160443307757886

P(wants|president) = 0.5160443307757886

P(to|wants) = 3.424561752430911e-05

P(control|to) = 7.418169862907512e-06

P(the|control) = 0.5160443307757886

P(board|the) = 0.0

P('s|board) = 0.00046035805626598467

P(control|'s) = 0.5160443307757886

P(sentence) = product of all the above mentioned values

P(sentence) = 0.0

1. Transformation based POS Tagging:

**sentence: The\_DT president\_NN wants\_VBZ to\_TO control\_??? the\_DT board\_NN 's\_POS control\_???**

1. Brill’s Algorithm

Top rules for NN to VB:

Previous\_tag From\_tag To\_tag Score

MD NN VB 45

TO NN VB 27

Top 5 rules for VB to NN:

Previous\_tag From\_tag To\_tag Score

DT VB NN 17

IN VB NN 12

JJ VB NN 10

NN VB NN 8

CD VB NN 2

Step 1: The word control is assigned tag that is most probable.

From computations: list\_most\_likely\_tag[“control”] = NN

Sentence: The\_DT president\_NN wants\_VBZ to\_TO **control\_NN** the\_DT board\_NN 's\_POS **control\_NN**

Step 2: Applying the rules, we can apply rule 2 for transformation from NN to VB.

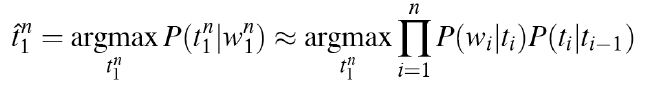
Therefore, the tag of the first control i.e. **… to\_TO control\_NN …**

Changes to VB,

Resulting sentence: The\_DT president\_NN wants\_VBZ to\_TO **control\_VB** the\_DT board\_NN 's\_POS **control\_NN**

1. Naïve Bayesian Classification

**sentence: The\_DT president\_NN wants\_VBZ to\_TO control\_??? the\_DT board\_NN 's\_POS control\_???**



In the corpus the different tags by which, control has been tagged is VB and NN. Hence these are the two possible tagging for each control and we need to find which one has the highest probabilities.

The probability of the entire sequence is given as follows:

P(sentence) = P(president|NN)\*P(NN|DT)  
 \*P(wants|VBZ)\*P(VBZ|NN)  
 \*P(to|TO)\*P(TO|VBZ)  
 \***P(control|X)**\***P(X|TO)** \*P(the|DT)\***P(DT|X)**  
 \*P(board|NN)\*P(NN|DT)  
 \*P('s|POS)\*P(POS|NN)  
 \***P(control|Y)**\***P(Y|POS)**

We saw that the tag of first control X and second control Y can be any of {VB, NN}

In the above equation, only the highlighted probabilities are dependent on the value of tag assigned as other do not contain X or Y.

Let’s compute to find out which tag assignment yields in highest probability.

For the first control, tag X:

X = VB; P(control|VB)\*P(VB|TO)\*P(DT|VB)

= (C(control, VB)/C(VB))\*(C(TO, VB)/C(TO))\*(C(VB, DT)/C(VB))

= (1/1998)\*(986/1551)\*(553/1998)

= 8.806417e-05

X = NN; P(control|NN)\*P(NN|TO)\*P(DT|NN)

= (C(control, NN)/C(NN))\*(C(TO, NN)/C(TO))\*(C(NN, DT)/C(NN))

= (12/11102)\*(51/1551)\*(69/11102)

= 2.2089e-07

We obtain max probability when X = VB. Therefore first control is tagged VB

For the second control, tag = Y:

Y = NN, P(control|NN)\*P(NN|POS) = 0.0010808863267879661 \* 0.43456375838926176

=0.00046971402

Y = VB, P(control|VB)\*P(VB|POS) = 0.0005005005005005005 \* 0 = 0

We obtain max probability for Y = NN, Therefore second control is tagged NN

**The\_DT president\_NN wants\_VBZ to\_TO control\_VB the\_DT board\_NN 's\_POS control\_NN**