Final Presentation

Bijoy Singh (120050087)
Sai Kiran Mudulkar(120050068)
Manik Dhar(120050006)

Whats in this presentation?

- This presentation does not contain the details of each lab, it only contains the conclusions and results
- It also contains what special and unique things our group did in this lab.
- We have detailed presentations for each lab too.

A* Star Search

Made an abstract implementation in python using object oriented programming.

Implementation of A* Algorithm and storage data structure is independent of the problem at hand, making it abstract for the user.

8-Puzzle Problem

We implemented the 8-puzzle problem over the above API.

Here are the figures:

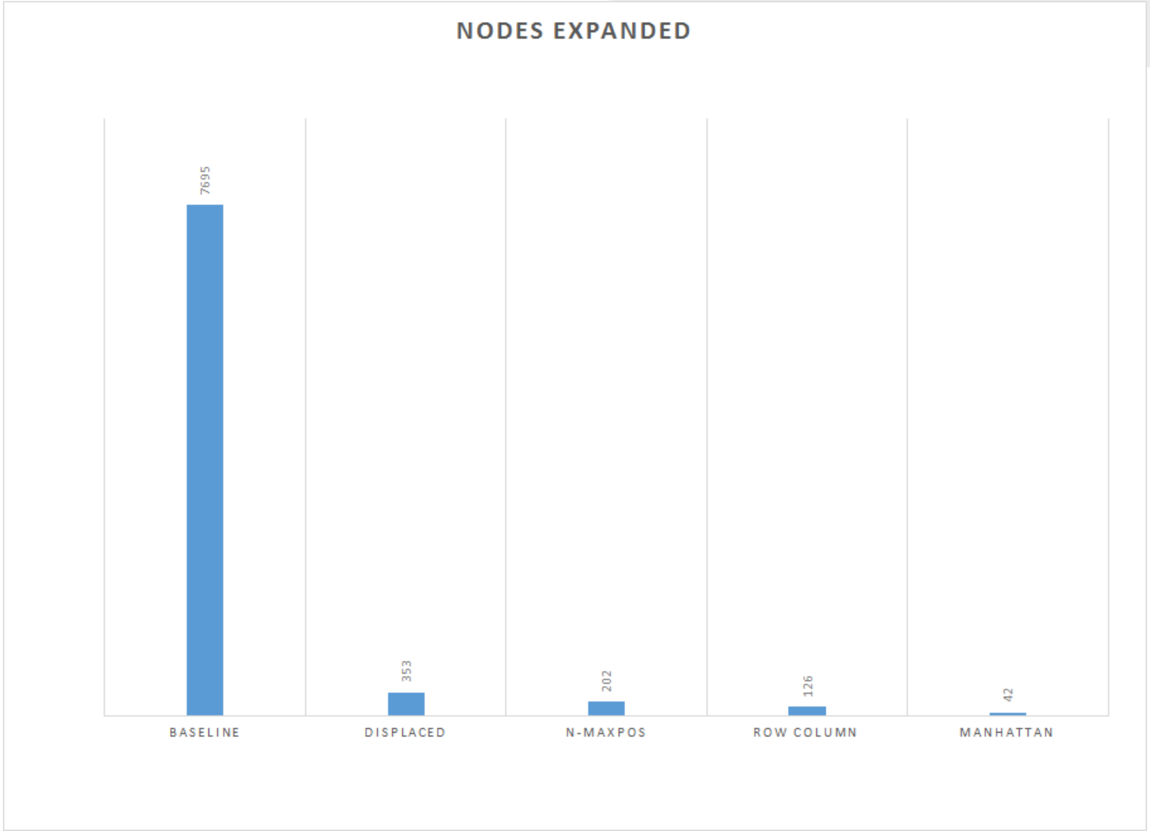
No Heuristic : 7695*

Displaced tiles : 353*

Manhattan distance : 42*

*The figures are subject to the definition of less than operator for f-score. The less than condition we have used take h-score as the next candidate if f-score is same.

Better Heuristics Perform Better



Our Own Heuristics

We implemented 2 heuristics for the lab

1. n-Max Swap : The number of moves to reach final states where in each move you can swap the blank tile with any tile

   Performance : 202 moves

2. RC Distance : The sum of the number of tiles not in correct column and number of tiles not in correct rows

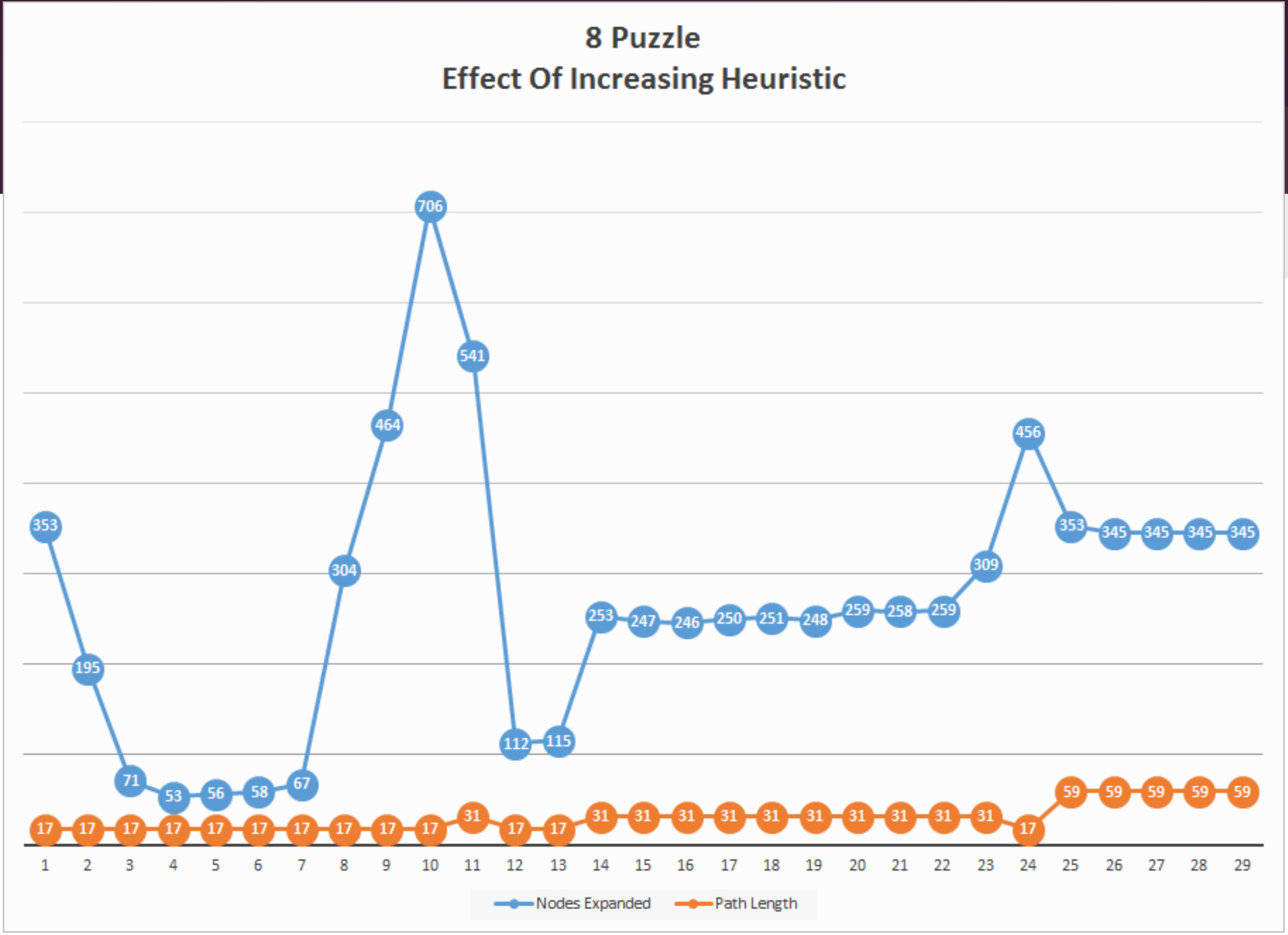
   Performance : 126 moves

Non Admissible Heuristic

We used a simple way of creating a non-admissible is to scale the Displaced Tiles Heuristic by a factor and added 32*

The effect of increasing the heuristic is evident in the following graph

*32 is the maximum number of moves needed to solve the problem.


8 Puzzle
Effect Of Increasing Heuristic
Nodes Expanded: 353, 195, 71, 53, 56, 58, 67, 304, 464, 706, 541, 112, 115, 253, 247, 246, 250, 251, 248, 259, 258, 259, 309, 456, 353, 345, 345, 345, 345
Path Length: 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 31, 17, 17, 31, 31, 31, 31, 31, 31, 31, 31, 31, 31, 17, 59, 59, 59, 59, 59
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
Nodes Expanded
Path Length

Cannibal Missionary Problem

We solved the cannibal missionary problem using the same API we coded.

Results:

No Heuristic : 25 moves

Our Own Heuristics

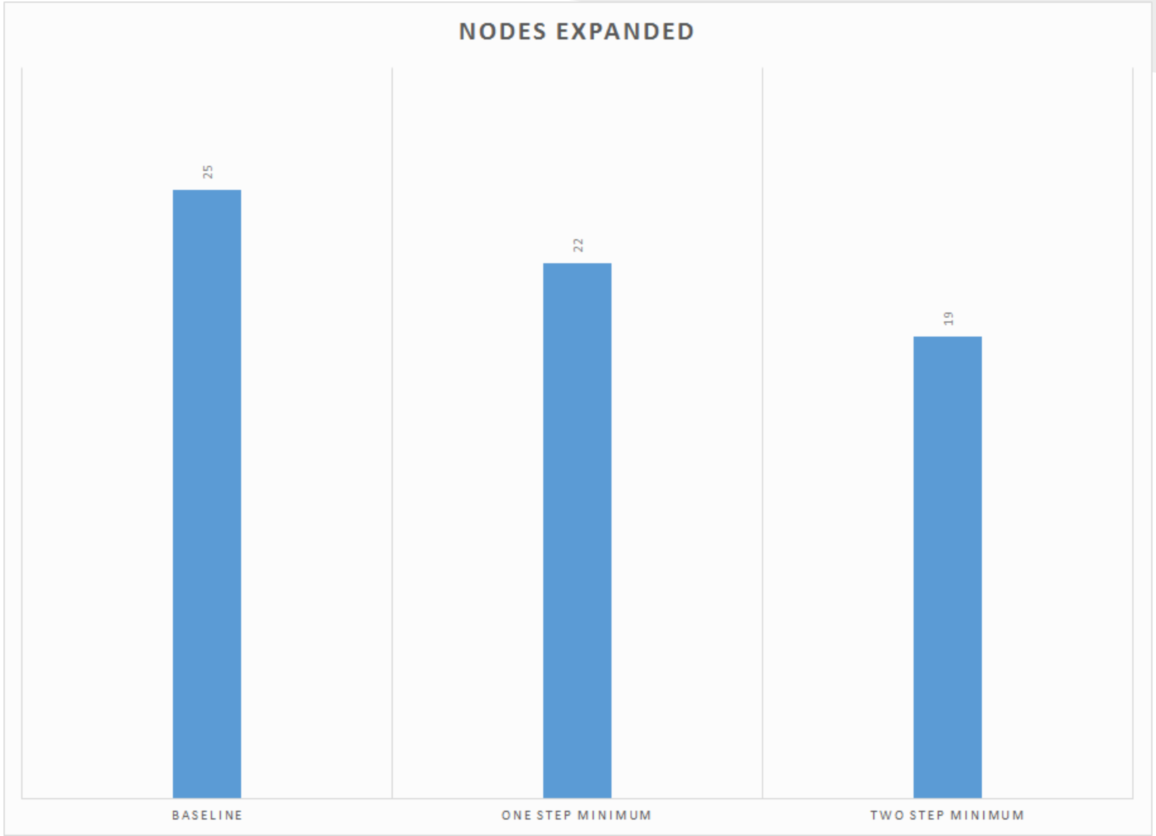
1 Basic One Step heuristic : The minimum times a boat must go back and forth for C people in the first side is (C - 1)

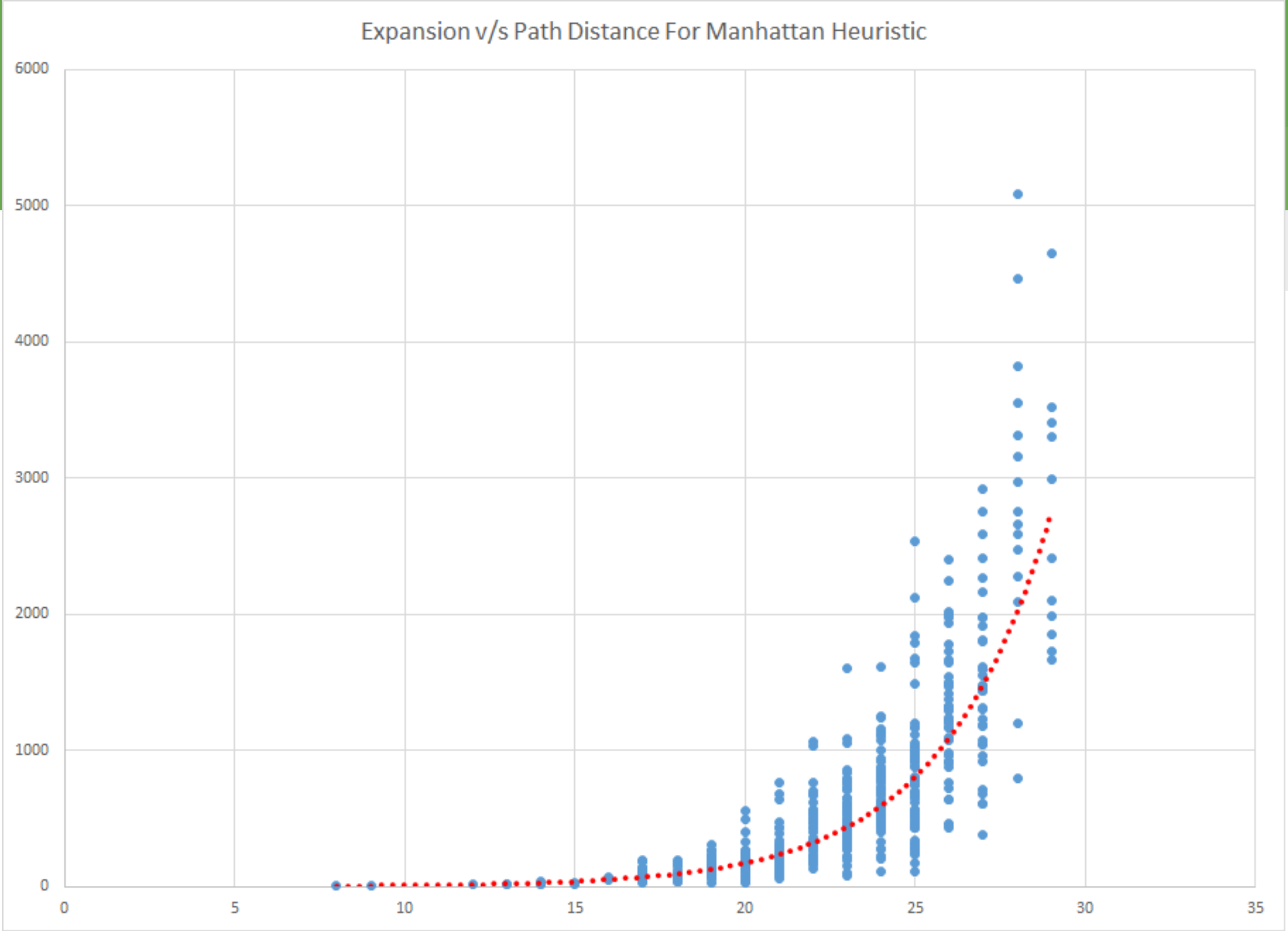
Performance 22 moves

2 Two Step heuristic : A more accurate heuristic, which takes care of which side the boat is and how many people are on first side

Performance 19 moves

Better Heuristic Perform Better



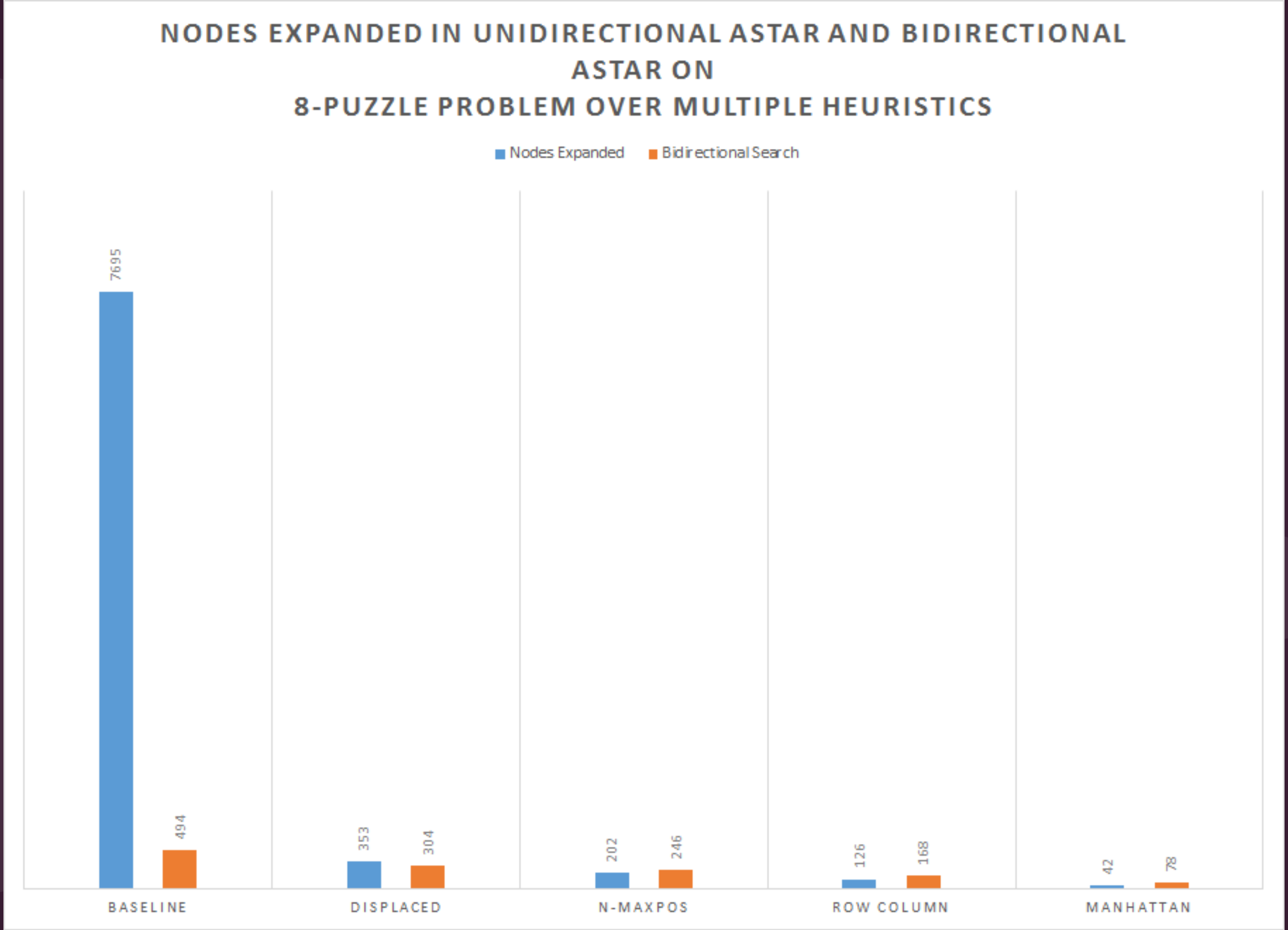

Expansion v/s Path Distance For Manhattan Heuristic
6000
5000
4000
3000
2000
1000
0
0
5
10
15
20
25
30
35

Bidirectional AStar

Implementation:

We use 2 Open Lists and Closed Lists based on the data structures and procedure defined in previous lab.

One open list corresponds to the forward step and the other corresponds to the backward step (from goal to the start)


NODES EXPANDED IN UNIDIRECTIONAL ASTAR AND BIDIRECTIONAL ASTAR ON
8-PUZZLE PROBLEM OVER MULTIPLE HEURISTICS
Nodes Expanded
Bidirectional Search
7695
494
353
304
202
246
126
168
42
78
BASELINE
DISPLACED
N-MAXPOS
ROW COLUMN
MANHATTAN

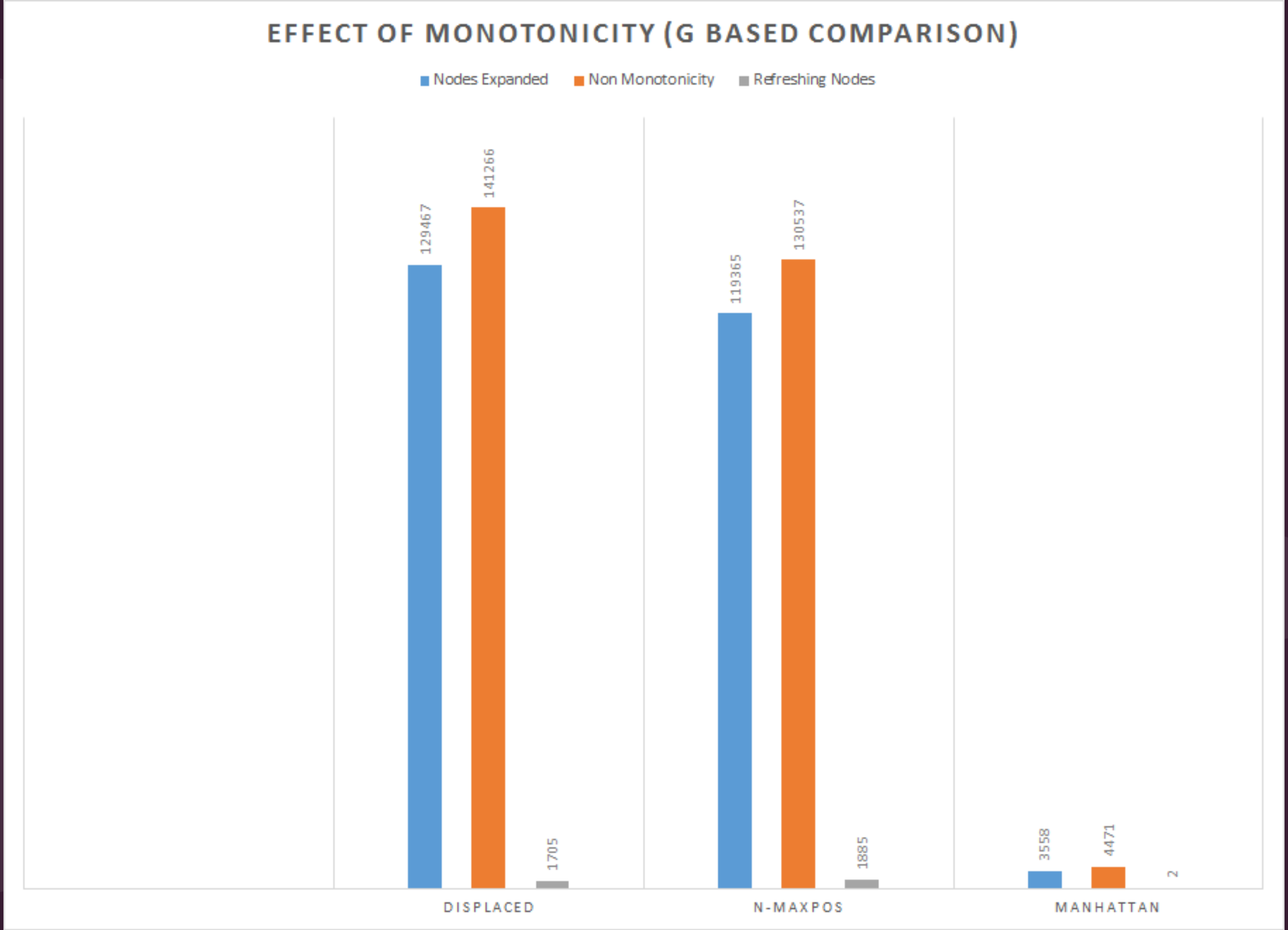
Non Monotonic Heuristic

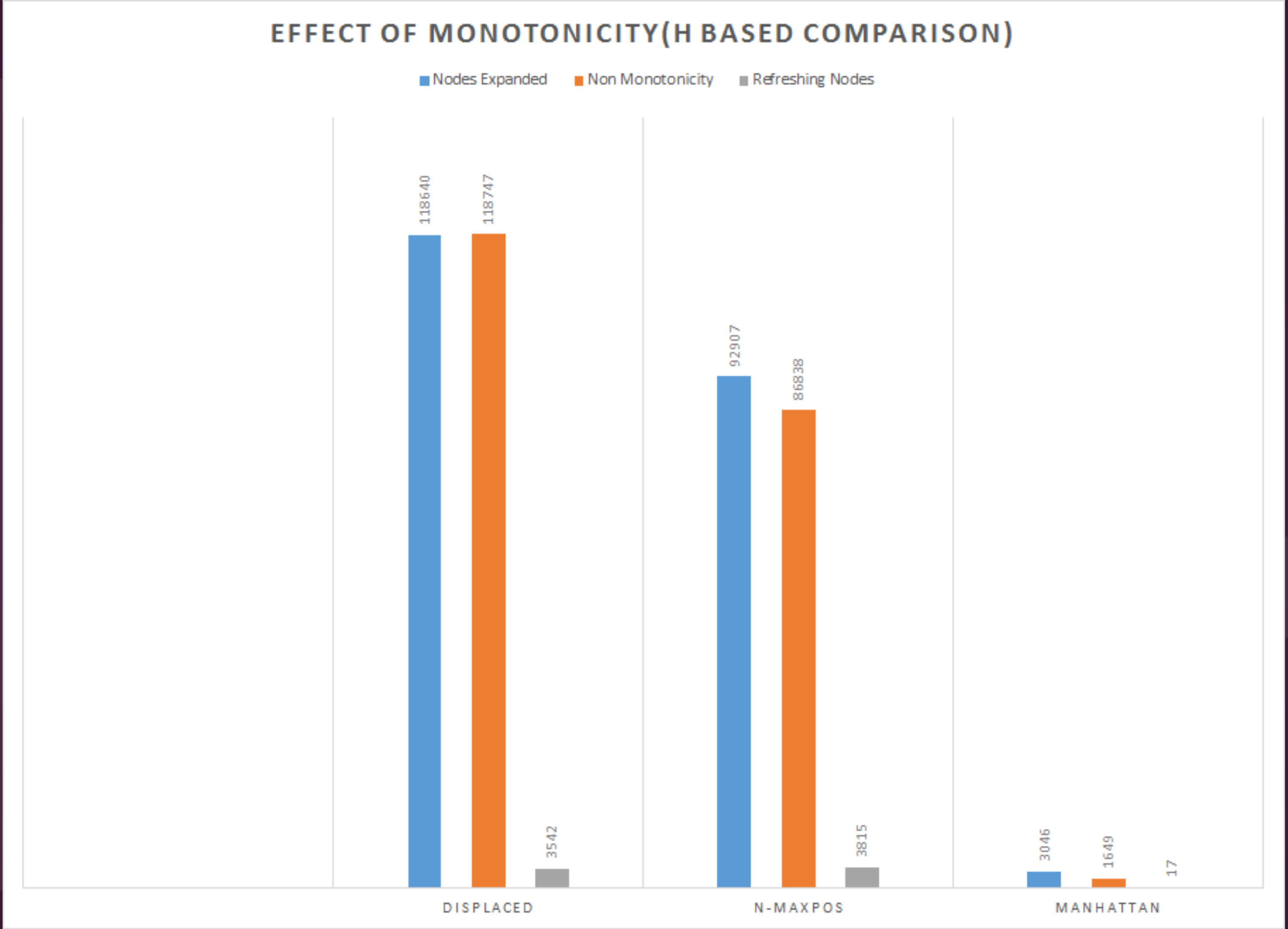
We used the following method to generate a non monotonic heuristic. Let say we have a monotonic heuristic h available to us. We define a new heuristic h’ as follows:

h’ = h if the blank tile is not in the first column

    = 0 otherwise

h’ is a non monotonic heuristic for the 8 puzzle.


EFFECT OF MONOTONICITY (G BASED COMPARISON)
Nodes Expanded
Non Monotonicity
Refreshing Nodes
129467
141266
1705
119365
130537
1885
3558
4471
2
DISPLACED
N-MAXPOS
MANHATTAN


EFFECT OF MONOTONICITY(H BASED COMPARISON)
Nodes Expanded
Non Monotonicity
Refreshing Nodes
118640
118747
3542
92907
86838
3815
3046
1649
17
DISPLACED
N-MAXPOS
MANHATTAN

Theorem Prover

- We have been given a statement in propositional logic.
- The program needs to check whether the statement is a theorem or not.
- The use of semantics is not allowed.
- Only syntax rules and the deduction theorem is used.

The Method

We create the the maximal set of hypothesis

H1, H2,....,Hs

by deduction theorem, contrapositive and modes pones being applied recursively till the information cannot be gathered

The Method (Contd.)

Let us say we are trying to prove K. If K is of the form ((A -> F) -> F) we remove the double not.

We assume each statement to be of the form (LHS -> RHS) if an implication is present.

We search for K in the RHS of all the statements and if we find it, then we attempt to prove the LHS.

The Method (Contd.)

Also if K is the form of (A -> B) we can prove either (~B -> ~A) or B to finish the job.

This because of contrapositives being equivalent to the original statement and Axiom 1 [(B -> (A -> B)) and B give (A -> B)]

We use Axiom 2 in a similar manner. If K is of the form ((A -> B) -> (A -> C)) we try proving (A -> (B -> C)).

The Method (Contd.)

We keep going backwards in this manner until we get stuck at a hypothesis.

Lets say we got stuck at P1, …, Pg

Now we can assume either to finish the proof or trying proving one of them. We ask for the user’s input here to choose one of them.

Now we attempt to prove it.

User Input

If the process gets stuck(no new hypotheses are created) then the program signals an error.

It will ask the user to add a new hypotheses, verify a possible hypotheses or let the program assume a hypothesis and try to solve using that hypothesis

Circuit Verifier

M1 : This method uses prolog logic and the connections that are between various gates are implemented using the variables in Prolog.

M2 : Here we define circuits in a more traditional way by defining components, assigning them values, and attaching these components to each other.

Benefits Of Method 1

1) Smaller and more intuitive code
2) The gates that you define previously are reusable and hence bigger logic circuits will be more modular
3) This method allows Prolog to backtrack.
4) So stronger queries like XOR([A,0],1). can be called giving the output of 1.

Automated Code Generation

- As this process of writing code on prolog with connections is pretty cumbersome and requires a lot of accuracy
- We made a method by which the users can simply add the boolean function and the system will generate the prolog code for that boolean function for testing.

Method 2

- In this method we used dynamic predicates, added using assert and removed using retractall

- The basic facts state what are the inputs of the circuit and what are the outputs.

- Other facts include which connectors are connected to each other and which connectors are connected to gates.

Definition of the circuit:

- Finally we add a predicate which asserts signal values over the inputs, and asserts the type of the gates used.
- The definition of the primitive gates leads to addition of a statement which assigns a value to the output wire.
- In the end we check, if certain constraints are satisfied for eg: No dangling wires, no short circuits,etc.

Grapheme To Phoneme

- Train to HMMs using the CMU dataset.
- One for word to phoneme conversion and the other for phoneme to word.
- As the CMU dataset is tagged, this becomes a supervised learning problem.
- We use the MLE to find HMM parameters and Viterbi to do the actual conversion.

Idea

$P(S\ I \Rightarrow |\ O\ | \Rightarrow S\ II) = n(S\ I \Rightarrow |\ O\ | \Rightarrow S\ II)/n(S\ I)$

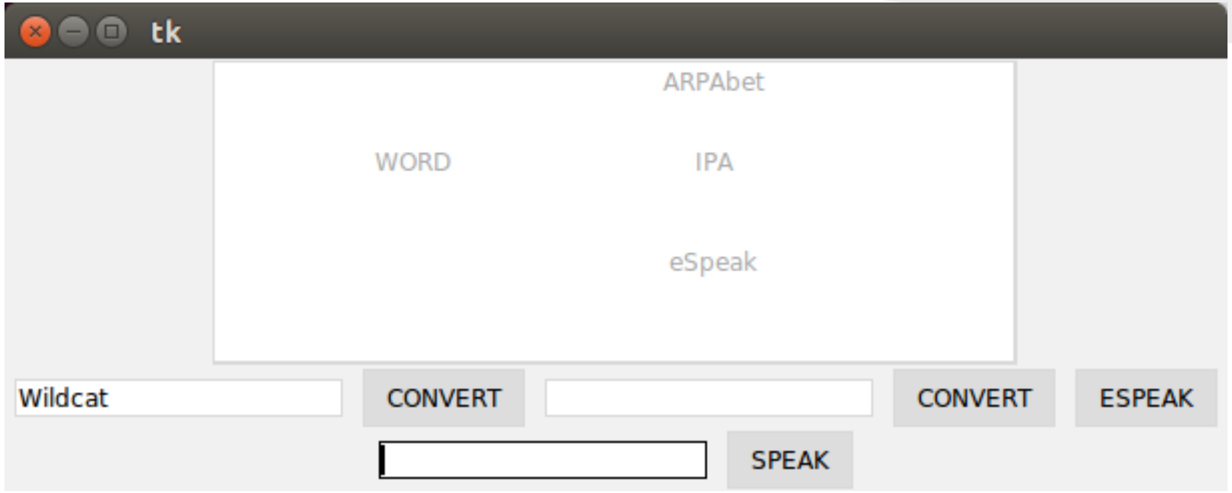
$n(S\ I)$ = no of times S I appears in the data set.

$n(S\ I \Rightarrow |\ O\ | \Rightarrow S\ II)$ = number of times S I is followed by SII in the dataset with O in the observation sequence corresponding to S II.

Implementation

- We store a transition table to store the probabilities which we store in a file to avoid repeated training.
- The phoneme sequence is made using the ARPABET notation. To produce sound we convert the output in ARPABET to ESpeak notation (IPA).
- There is a GUI interface to do the conversion and sound production.

Interface



UI interface and API to allow for easier usage

Interface Properties

Allows for word to phoneme and phoneme to word conversion.

Word can be converted to IPA, ARPAbet or eSpeak notation.

The user can also simply enter a sentence which he or she wants to speak and it will speak it.

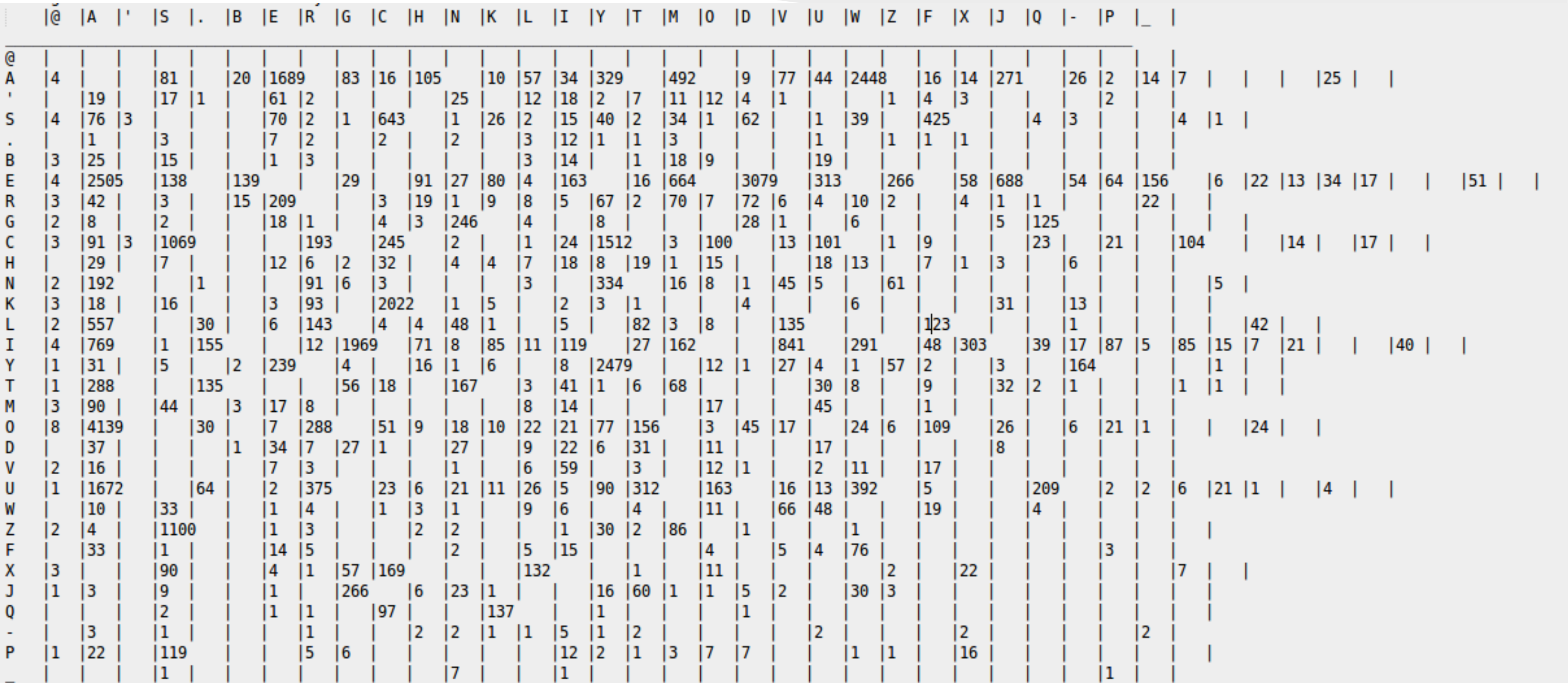
Performance:

We performed 5 fold cross validation after permuting the input set. The results are as follows:

Average Word Conversion Accuracy : 0.7338947118582697

Average Phoneme Conversion Accuracy : 0.8014263311120567

Confusion Matrix (GrGr)



```
      |@    |A    |'    |S    |.    |B    |E    |R    |G    |C    |H    |N    |K    |L    |I    |Y    |T    |M    |O    |D    |V    |U    |W    |Z    |F    |X    |J    |Q    |-    |P    |_    |
__________________________________________________________________________________________________________________________________________________________________________________________
@     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
A     |4    |     |     |81   |     |20   |1689       |83   |16   |105        |10   |57   |34   |329        |492        |9    |77   |44   |2448       |16   |14   |271        |26   |2    |14   |7    |     |     |     |25   |     |
'     |     |19   |     |17   |1    |     |61   |2    |     |     |     |25   |     |12   |18   |2    |7    |11   |12   |4    |1    |     |     |1    |4    |3    |     |     |     |2    |     |
S     |4    |76   |3    |     |     |     |70   |2    |1    |643        |1    |26   |2    |15   |40   |2    |34   |1    |62   |     |1    |39   |     |425        |     |4    |3    |     |     |4    |1    |
.     |     |1    |     |3    |     |     |7    |2    |     |2    |     |2    |     |3    |12   |1    |1    |3    |     |     |     |1    |     |1    |1    |1    |     |     |     |     |     |
B     |3    |25   |     |15   |     |     |1    |3    |     |     |     |     |     |3    |14   |     |1    |18   |9    |     |     |19   |     |     |     |     |     |     |     |     |     |
E     |4    |2505       |138        |139        |     |29   |     |91   |27   |80   |4    |163        |16   |664        |3079       |313        |266        |58   |688        |54   |64   |156        |6    |22   |13   |34   |17   |     |     |51   |     |
R     |3    |42   |     |3    |     |15   |209        |     |3    |19   |1    |9    |8    |5    |67   |2    |70   |7    |72   |6    |4    |10   |2    |     |4    |1    |1    |     |     |22   |     |
G     |2    |8    |     |2    |     |     |18   |1    |     |4    |3    |246        |4    |     |8    |     |     |     |28   |1    |     |6    |     |     |     |5    |125        |     |     |     |     |
C     |3    |91   |3    |1069       |     |     |193        |245        |2    |     |1    |24   |1512       |3    |100        |13   |101        |1    |9    |     |     |23   |     |21   |     |104        |     |14   |     |17   |     |
H     |     |29   |     |7    |     |     |12   |6    |2    |32   |     |4    |4    |7    |18   |8    |19   |1    |15   |     |     |18   |13   |     |7    |1    |3    |     |6    |     |     |
N     |2    |192        |     |1    |     |     |91   |6    |3    |     |     |     |3    |     |334        |16   |8    |1    |45   |5    |     |61   |     |     |     |     |     |     |     |     |5    |
K     |3    |18   |     |16   |     |     |3    |93   |     |2022       |1    |5    |     |2    |3    |1    |     |     |4    |     |     |6    |     |     |     |31   |     |13   |     |     |     |
L     |2    |557        |     |30   |     |6    |143        |4    |4    |48   |1    |     |5    |     |82   |3    |8    |     |135        |     |     |123        |     |     |1    |     |     |     |     |42   |     |
I     |4    |769        |1    |155        |     |12   |1969       |71   |8    |85   |11   |119        |27   |162        |     |841        |291        |48   |303        |39   |17   |87   |5    |85   |15   |7    |21   |     |     |40   |     |
Y     |1    |31   |     |5    |     |2    |239        |4    |     |16   |1    |6    |     |8    |2479       |     |12   |1    |27   |4    |1    |57   |2    |     |3    |     |164        |     |     |1    |     |
T     |1    |288        |     |135        |     |     |56   |18   |     |167        |3    |41   |1    |6    |68   |     |     |     |30   |8    |     |9    |     |32   |2    |1    |     |     |1    |1    |     |
M     |3    |90   |     |44   |     |3    |17   |8    |     |     |     |     |     |8    |14   |     |     |     |17   |     |     |45   |     |     |1    |     |     |     |     |     |     |
O     |8    |4139       |     |30   |     |7    |288        |51   |9    |18   |10   |22   |21   |77   |156        |3    |45   |17   |     |24   |6    |109        |26   |     |6    |21   |1          |     |24   |     |
D     |     |37   |     |     |     |1    |34   |7    |27   |1    |     |27   |     |9    |22   |6    |31   |     |11   |     |     |17   |     |     |     |     |8    |     |     |     |     |
V     |2    |16   |     |     |     |     |7    |3    |     |     |     |1    |     |6    |59   |     |3    |     |12   |1    |     |2    |11   |     |17   |     |     |     |     |     |     |
U     |1    |1672       |     |64   |     |2    |375        |23   |6    |21   |11   |26   |5    |90   |312        |163        |16   |13   |392        |5    |     |     |209        |2    |2    |6    |21   |1    |     |4    |     |
W     |     |10   |     |33   |     |     |1    |4    |     |1    |3    |1    |     |9    |6    |     |4    |     |11   |     |66   |48   |     |     |19   |     |     |4    |     |     |     |
Z     |2    |4    |     |1100       |     |1    |3    |     |     |2    |2    |     |     |1    |30   |2    |86   |     |1    |     |     |1    |     |     |     |     |     |     |     |     |     |
F     |     |33   |     |1    |     |     |14   |5    |     |     |     |2    |     |5    |15   |     |     |     |4    |     |5    |4    |76   |     |     |     |     |     |     |3    |     |
X     |3    |     |     |90   |     |     |4    |1    |57   |169        |     |     |132        |     |1    |     |11   |     |     |     |     |2    |     |22   |     |     |     |     |     |7    |     |
J     |1    |3    |     |9    |     |     |1    |     |266        |6    |23   |1    |     |     |16   |60   |1    |1    |5    |2    |     |30   |3    |     |     |     |     |     |     |     |     |
Q     |     |     |     |2    |     |     |1    |1    |     |97   |     |     |137        |     |1    |     |     |     |1    |     |     |     |     |     |     |     |     |     |     |     |     |
-     |     |3    |     |1    |     |     |     |1    |     |     |2    |2    |1    |1    |5    |1    |2    |     |     |     |     |2    |     |     |     |2    |     |     |     |     |2    |
P     |1    |22   |     |119        |     |     |5    |6    |     |     |     |     |     |12   |2    |1    |3    |7    |7    |     |     |1    |1    |     |16   |     |     |     |     |     |     |
_     |     |     |     |1    |     |     |     |     |     |     |     |7    |     |     |1    |     |     |     |     |     |     |     |     |     |     |     |     |     |     |1    |     |
```

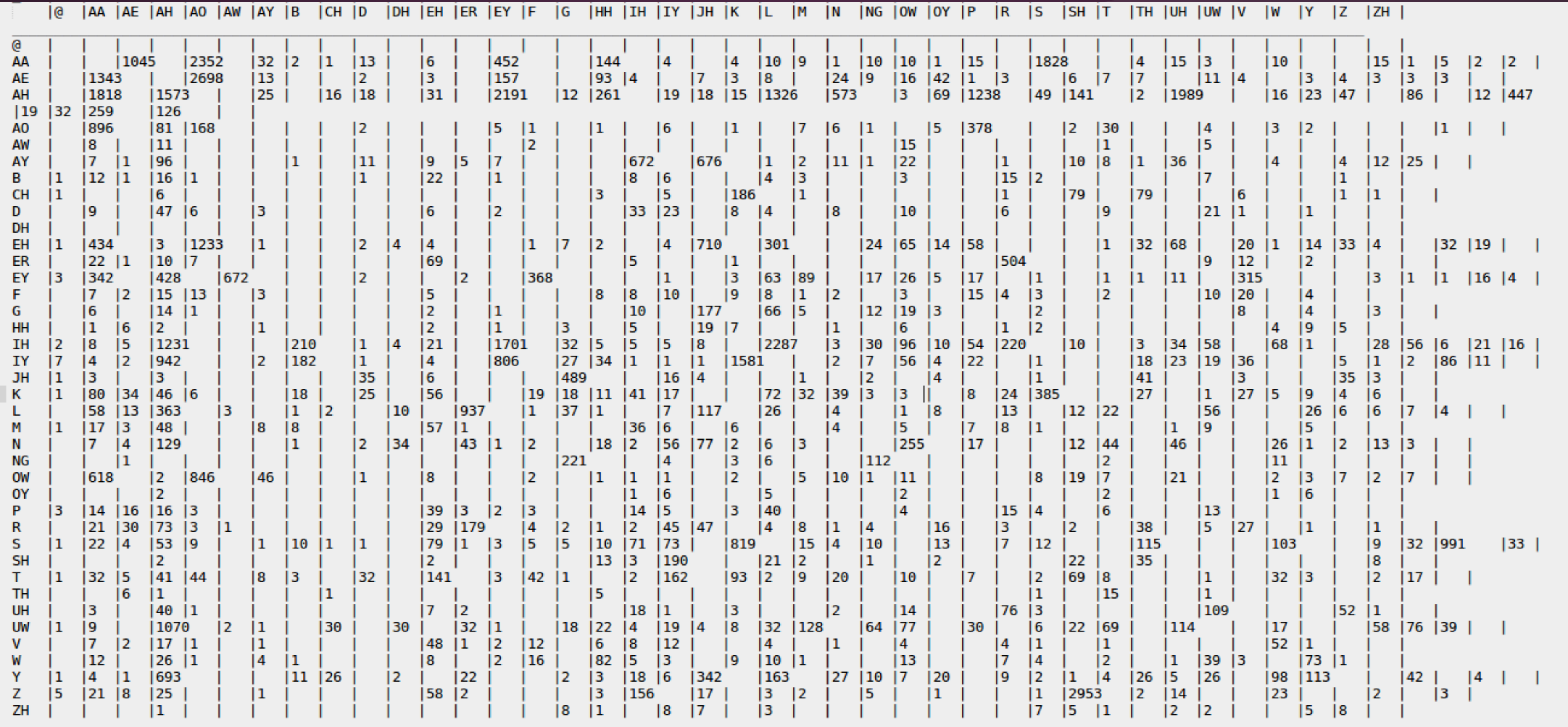
Confusion Matrix (GrGr) conclusions

We can see that some alphabets are characteristic of being confused to others

A↔E,A↔O,S↔C, E↔I etc.

These confusion are legitimate as even english has these issues.

Confusion Matrix(PhPh)



```
      |@   |AA  |AE  |AH  |AO  |AW  |AY  |B   |CH  |D   |DH  |EH  |ER  |EY  |F   |G   |HH  |IH  |IY  |JH  |K   |L   |M   |N   |NG  |OW  |OY  |P   |R   |S   |SH  |T   |TH  |UH  |UW  |V   |W   |Y   |Z   |ZH  |
________________________________________________________________________________________________________________________________________________________________________________________________________
@     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
AA    |    |    |1045     |2352     |32  |2   |1   |13  |    |6   |    |452      |    |144      |4   |    |4   |10  |9   |1   |10  |10  |1   |15  |    |1828     |    |4   |15  |3   |    |10  |    |    |15  |1   |5   |2   |2   |
AE    |    |1343     |    |2698     |13  |    |    |2   |    |3   |    |157      |    |93  |4   |    |7   |3   |8   |    |24  |9   |16  |42  |1   |3   |    |6   |7   |7   |    |11  |4   |    |3   |4   |3   |3   |3   |    |
AH    |    |1818     |1573     |    |25  |    |16  |18  |    |31  |    |2191     |12  |261      |19  |18  |15  |1326     |573      |3   |69  |1238     |49  |141      |2   |1989     |    |16  |23  |47  |    |86  |    |12  |447
|19   |32  |259      |126      |    |
AO    |    |896      |81  |168      |    |    |    |2   |    |    |    |5   |1   |    |1   |    |6   |    |1   |    |7   |6   |1   |    |5   |378      |    |2   |30  |    |    |4   |    |3   |2   |    |    |    |1   |    |    |
AW    |    |8   |    |11  |    |    |    |    |    |    |    |    |    |    |2   |    |    |    |    |    |    |    |    |    |    |15  |    |    |    |    |    |1   |    |    |5   |    |    |    |    |    |
AY    |    |7   |1   |96  |    |    |    |1   |    |11  |    |9   |5   |7   |    |    |    |672      |676      |1   |2   |11  |1   |22  |    |    |1   |    |10  |8   |1   |36  |    |    |4   |    |4   |12  |25  |    |    |
B     |1   |12  |1   |16  |1   |    |    |    |    |1   |    |22  |    |1   |    |    |    |8   |6   |    |    |4   |3   |    |    |3   |    |    |15  |2   |    |    |    |    |7   |    |    |    |1   |    |    |
CH    |1   |    |    |6   |    |    |    |    |    |    |    |    |    |    |    |    |3   |    |5   |    |186      |1   |    |    |    |    |    |1   |    |79  |    |79  |    |    |6   |    |    |1   |1   |    |    |
D     |    |9   |    |47  |6   |    |3   |    |    |    |    |6   |    |2   |    |    |    |33  |23  |    |8   |4   |    |8   |    |10  |    |    |6   |    |    |9   |    |    |21  |1   |    |1   |    |    |    |
DH    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
EH    |1   |434      |3   |1233     |1   |    |    |2   |4   |4   |    |    |1   |7   |2   |    |4   |710      |301      |    |24  |65  |14  |58  |    |    |    |1   |32  |68  |    |20  |1   |14  |33  |4   |    |32  |19  |    |    |
ER    |    |22  |1   |10  |7   |    |    |    |    |    |    |69  |    |    |    |    |    |5   |    |    |1   |    |    |    |    |    |    |    |504      |    |    |    |    |9   |12  |    |2   |    |    |    |    |
EY    |3   |342      |428      |672      |    |    |2   |    |    |2   |    |368      |    |    |1   |    |3   |63  |89  |    |17  |26  |5   |17  |    |1   |    |1   |1   |11  |    |315      |    |    |3   |1   |1   |16  |4   |    |
F     |    |7   |2   |15  |13  |    |3   |    |    |    |    |5   |    |    |    |    |8   |8   |10  |    |9   |8   |1   |2   |    |3   |    |15  |4   |3   |    |2   |    |    |10  |20  |    |4   |    |    |    |
G     |    |6   |    |14  |1   |    |    |    |    |    |    |2   |    |1   |    |    |    |10  |    |177      |66  |5   |    |12  |19  |3   |    |    |2   |    |    |    |    |    |8   |    |4   |    |3   |    |    |
HH    |    |1   |6   |2   |    |    |1   |    |    |    |    |2   |    |1   |    |3   |    |5   |    |19  |7   |    |    |1   |    |6   |    |    |1   |2   |    |    |    |    |    |    |4   |9   |5   |    |    |
IH    |2   |8   |5   |1231     |    |    |210      |1   |4   |21  |    |1701     |32  |5   |5   |5   |8   |    |2287     |3   |30  |96  |10  |54  |220      |10  |    |3   |34  |58  |    |68  |1   |    |28  |56  |6   |21  |16  |
IY    |7   |4   |2   |942      |    |2   |182      |1   |    |4   |    |806      |27  |34  |1   |1   |1   |1581     |    |2   |7   |56  |4   |22  |    |1   |    |    |18  |23  |19  |36  |    |    |5   |1   |2   |86  |11  |    |    |
JH    |1   |3   |    |3   |    |    |    |    |    |35  |    |6   |    |    |    |489      |    |16  |4   |    |    |1   |    |2   |    |4   |    |    |1   |    |    |41  |    |    |3   |    |    |35  |3   |    |    |
K     |1   |80  |34  |46  |6   |    |    |18  |    |25  |    |56  |    |    |19  |18  |11  |41  |17  |    |    |72  |32  |39  |3   |3   |    |8   |24  |385      |    |27  |    |1   |27  |5   |9   |4   |6   |    |    |
L     |    |58  |13  |363      |3   |    |1   |2   |    |10  |    |937      |1   |37  |1   |    |7   |117      |26  |    |4   |    |1   |8   |    |13  |    |12  |22  |    |    |56  |    |    |26  |6   |6   |7   |4   |    |    |
M     |1   |17  |3   |48  |    |    |8   |8   |    |    |    |57  |1   |    |    |    |    |36  |6   |    |6   |    |    |4   |    |5   |    |7   |8   |1   |    |    |    |1   |9   |    |    |5   |    |    |    |
N     |    |7   |4   |129      |    |    |1   |    |2   |34  |    |43  |1   |2   |    |18  |2   |56  |77  |2   |6   |3   |    |    |255      |17  |    |    |12  |44  |    |46  |    |    |26  |1   |2   |13  |3   |    |    |
NG    |    |    |1   |    |    |    |    |    |    |    |    |    |    |    |    |221      |    |4   |    |3   |6   |    |    |112      |    |    |    |    |    |2   |    |    |    |    |11  |    |    |    |    |    |    |
OW    |    |618      |2   |846      |46  |    |    |1   |    |8   |    |    |2   |    |1   |1   |1   |    |2   |    |5   |10  |1   |11  |    |    |    |8   |19  |7   |    |21  |    |    |2   |3   |7   |2   |7   |    |    |
OY    |    |    |    |2   |    |    |    |    |    |    |    |    |    |    |    |    |    |1   |6   |    |    |5   |    |    |    |2   |    |    |    |    |    |2   |    |    |    |    |1   |6   |    |    |    |
P     |3   |14  |16  |16  |3   |    |    |    |    |    |    |39  |3   |2   |3   |    |    |14  |5   |    |3   |40  |    |    |    |4   |    |    |15  |4   |    |6   |    |    |13  |    |    |    |    |    |    |
R     |    |21  |30  |73  |3   |1   |    |    |    |    |    |29  |179      |4   |2   |1   |2   |45  |47  |    |4   |8   |1   |4   |    |16  |    |3   |    |2   |    |38  |    |5   |27  |    |1   |    |1   |    |    |
S     |1   |22  |4   |53  |9   |    |1   |10  |1   |1   |    |79  |1   |3   |5   |5   |10  |71  |73  |    |819      |15  |4   |10  |    |13  |    |7   |12  |    |    |115      |    |    |103      |    |9   |32  |991      |33  |
SH    |    |    |    |2   |    |    |    |    |    |    |    |2   |    |    |    |    |13  |3   |190      |    |21  |2   |    |1   |    |2   |    |    |    |22  |    |35  |    |    |    |    |    |    |8   |    |    |
T     |1   |32  |5   |41  |44  |    |8   |3   |    |32  |    |141      |3   |42  |1   |    |2   |162      |93  |2   |9   |20  |    |10  |    |7   |    |2   |69  |8   |    |    |1   |    |32  |3   |    |2   |17  |    |    |
TH    |    |    |6   |1   |    |    |    |    |1   |    |    |    |    |    |    |    |5   |    |    |    |    |    |    |    |    |    |    |    |    |1   |    |15  |    |    |1   |    |    |    |    |    |    |
UH    |    |3   |    |40  |1   |    |    |    |    |    |    |7   |2   |    |    |    |    |18  |1   |    |3   |    |    |2   |    |14  |    |    |76  |3   |    |    |    |    |109      |    |    |52  |1   |    |    |
UW    |1   |9   |    |1070     |2   |1   |    |30  |    |30  |    |32  |1   |    |18  |22  |4   |19  |4   |8   |32  |128      |64  |77  |    |30  |    |6   |22  |69  |    |114      |    |17  |    |    |58  |76  |39  |    |    |
V     |    |7   |2   |17  |1   |    |1   |    |    |    |    |48  |1   |2   |12  |    |6   |8   |12  |    |    |4   |    |1   |    |4   |    |    |4   |1   |    |1   |    |    |    |    |52  |1   |    |    |    |
W     |    |12  |    |26  |1   |    |4   |1   |    |    |    |8   |    |2   |16  |    |82  |5   |3   |    |9   |10  |1   |    |    |13  |    |    |7   |4   |    |2   |    |1   |39  |3   |    |73  |1   |    |    |
Y     |1   |4   |1   |693      |    |    |11  |26  |    |2   |    |22  |    |    |2   |3   |18  |6   |342      |163      |27  |10  |7   |20  |    |9   |2   |1   |4   |26  |5   |26  |    |98  |113      |    |42  |    |4   |    |    |
Z     |5   |21  |8   |25  |    |    |1   |    |    |    |    |58  |2   |    |    |    |3   |156      |17  |    |3   |2   |    |5   |    |1   |    |    |1   |2953     |2   |14  |    |    |23  |    |    |2   |    |3   |
ZH    |    |    |    |1   |    |    |    |    |    |    |    |    |    |    |    |8   |1   |    |8   |7   |    |3   |    |    |    |    |    |    |    |7   |5   |1   |    |2   |2   |    |    |5   |8   |    |    |
```

Confusion Matrix (PhPh)

conclusions

We can see that some phonemes are characteristic of being confused to others

AA↔AE,AA↔AH,OW↔AA, AH↔AE etc.

These confusion are legitimate as the sounds are close and their occurrences are in generally similar surrounding context.

Optimisation Attempts

1) Non Zero probability initially
2) Trigram Model
3) Neural Networks(Later)

Non Zero initial probability

The logic behind this is that many possibilities of the state diagram is nullified by the 0 probabilities existing due to no data seen for these transitions.

Instead of 0 we add 10-10 to allow for some weightage. We see not too much improvement of 0.01%(not much huh!)

Average Word Conversion Accuracy : 0.7430759192440042

Average Phoneme Conversion Accuracy : 0.8014293446848637

Trigram:

We changed the state machine to work with trigrams. The results were as follows:

Average Word Conversion Accuracy : 0.6153025931351043

Average Phoneme Conversion Accuracy : 0.7398425392381858

Trigram Output Analysing

The performance of the machine reduces with the trigram assumption(besides it takes significantly more time to solve), this is expected for the reason that the new state transistion ((S1,S2),S3,O) requires more training to get trainined and that the english model for equi-phone-grapheme words may not be too much effected by this assumption.

Neural Networks

Neural Networks

Bijoy Singh (120050087)
Sai Kiran Mudulkar(120050068)
Manik Dhar(120050006)

Perceptron Model

We implemented the perceptron model in the general form.

- **θ** was a part of the weights with constant input of -1
- Both the models for deciding the value of the output were allowed(step threshold and sigmoidal threshold)

Also we made the schema general enough to use directly as well in a Neural Network

Perceptron Training

Perceptron training for problems just one perceptron uses the PTA algorithm.

It takes a truth table modifies it to create appropriate classes and inverts 0 class.

It now trains using the PTA algorithm which is guaranteed to converge if the function is separable.

Perceptron Performance

A perceptron can solve the problems as one would expect:

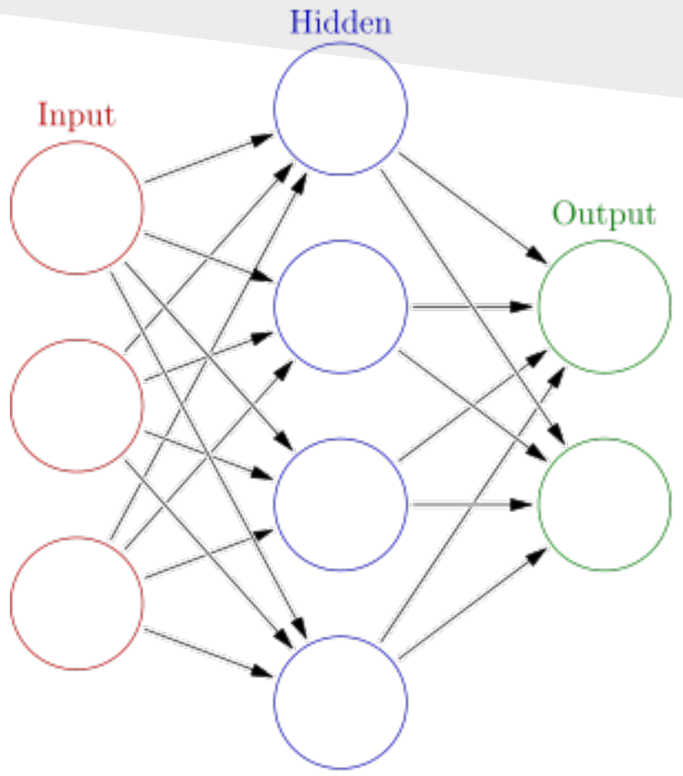
- AND
- OR
- MAJORITY

but cannot to

- PALINDROME (if it could it would be able to solve XOR, which we know it cannot)
- XOR

Neural Networks

We now used our existing infrastructure of perceptrons and using the sigmoid threshold neuron model, we used the to make a neural network



Neural Network

The user can specify

- The number of inputs, number of neurons on each row, the number of outputs, η(for Δw_{ij}) and the accuracy anticipated.
- This way the user can control the learning rate(through η) and the final result accuracy.

Neural Network Training

- We use the standard back propagation algorithm with the parameters η and accuracy(ε).
- For the smaller boolean circuits, we have trained the network until the ε becomes very small(0.1) this can be modified by the user
- The algorithm stops when the training is complete

Neural Network Performance

The neural network was tested on the following circuits and it performed as expected

3 input PALINDROME

4 input PALINDROME

XOR

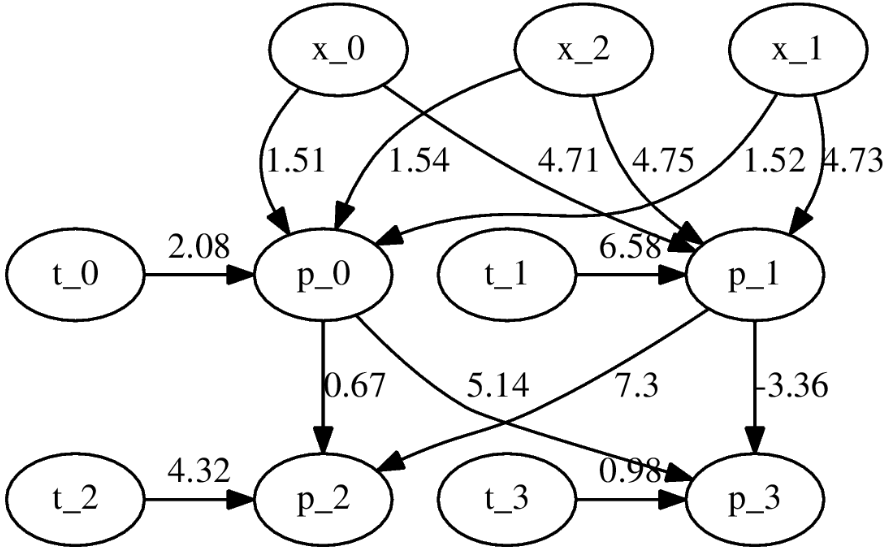
FULL ADDER

Visual Results of Neural Training

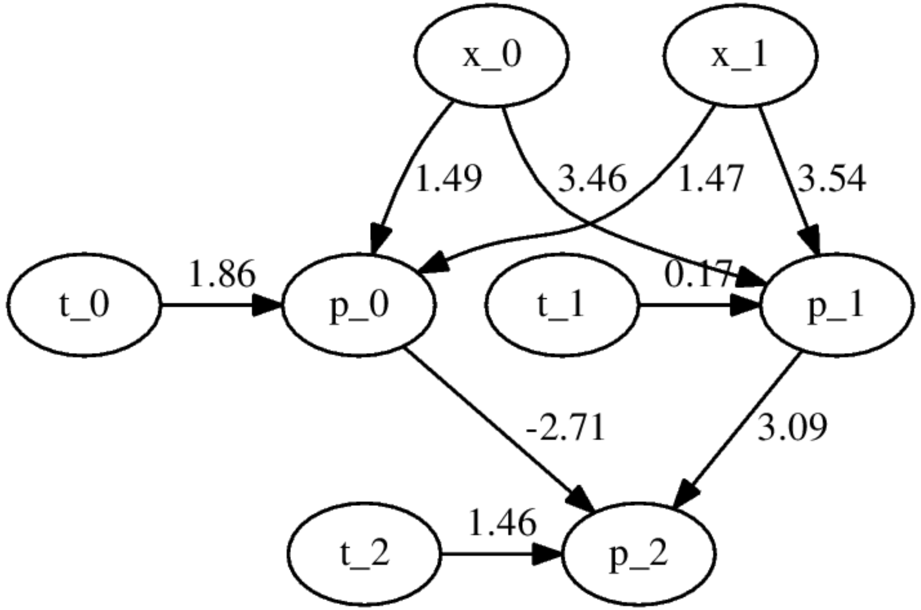
Now the results of these neural networks are a bit difficult to visualise from text. Hence we made a module to allow for UI output for the neurons

The outputs of the various neural networks are displayed

Visual Results of Neural Training

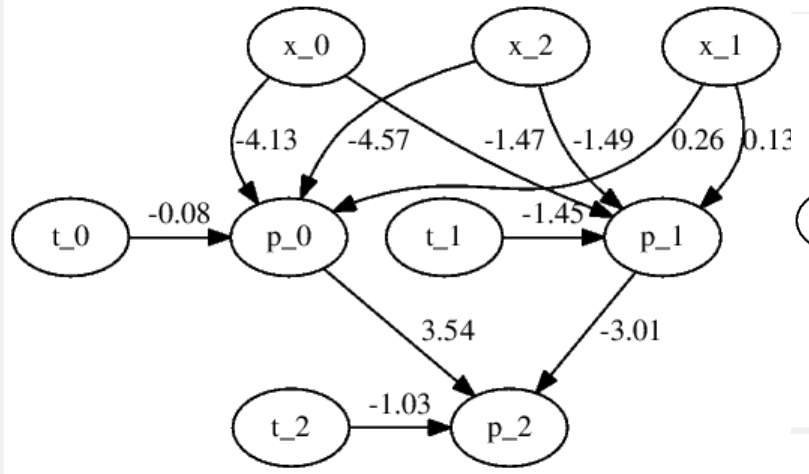


FULL ADDER

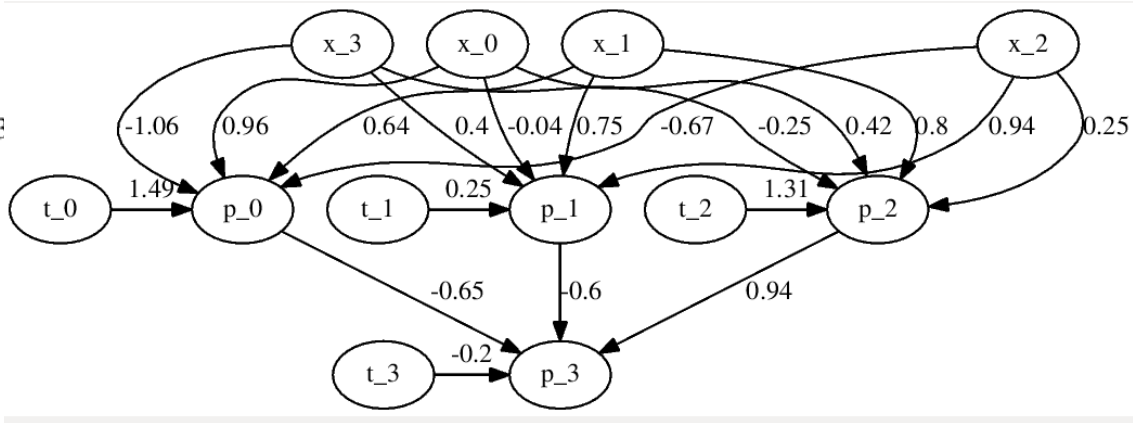


XOR

Visual Results of Neural Training



3 PALINDROME



4 PALINDROME

Grapheme To Phoneme using NN

- We encoded the graphemes and phonemes using bit sequences by giving each grapheme and a phoneme a bit representation as given in the question
- For padding we used 0000...
- For testing we used this bit sequence

Training Adjustment

- As the neural network is large we cannot expect it to train over the entire data till convergence(which is not even known if it will happen),
- We iterate over a large cycles to let it train the data. The user can specify this using a command line argument

Neural Network Performance

The neural network performance is mixed* **(we'll analyse this later in this presentation and see why this answer isn't the best measure what our system could achieve)**

- Measures to look at:
- Word Length
- Test Cases
- Training Duration
- Character Accuracy
- Bit Accuracy

THE RESULTS

	Test Case 1	Test Case 2	Test Case 3	Test Case 4	Tes
Test Case	Word Length 4 Test cases 10	Word Length 4 Test cases 10	Word Length 6 Test cases 100	Word Length 8 Test cases 1000	Word Length 10 Test cases 10000
Training Duration	10000 iterations **1m 45.074s**	100000 iterations **9m40.104s**	20000 iterations **~10 hours**	20000 iterations **5 days 22hours!!!!!** *(Yes we actually ran that on a server for 6 days just to see what happens)*	1000 iterations **115m7.984s**
Accuracy	70.0%	82.5%	32.66%	54.72%	23.9986%
Word Efficiency	97.1428%	98.5714%	83.11%	89.6719%	73.463%

Performance Issues and Improvements

The performance on smaller data being so high indicate that the mechanism has the potential of performing well,

However but as the amount of data becomes bigger, the amount of time to train(to the same level) increases exponentially.

Hence the increased requirement in time but reduced accuracy

How we can improve the accuracy

1) Increase the time to train (exponential)
2) Use more sparse numbering of the input grapheme bits
3) Dont use padding and work with only a standard size

THANK YOU