SIL 618
Computer Architecture
Assignment 2
Report
2021-22

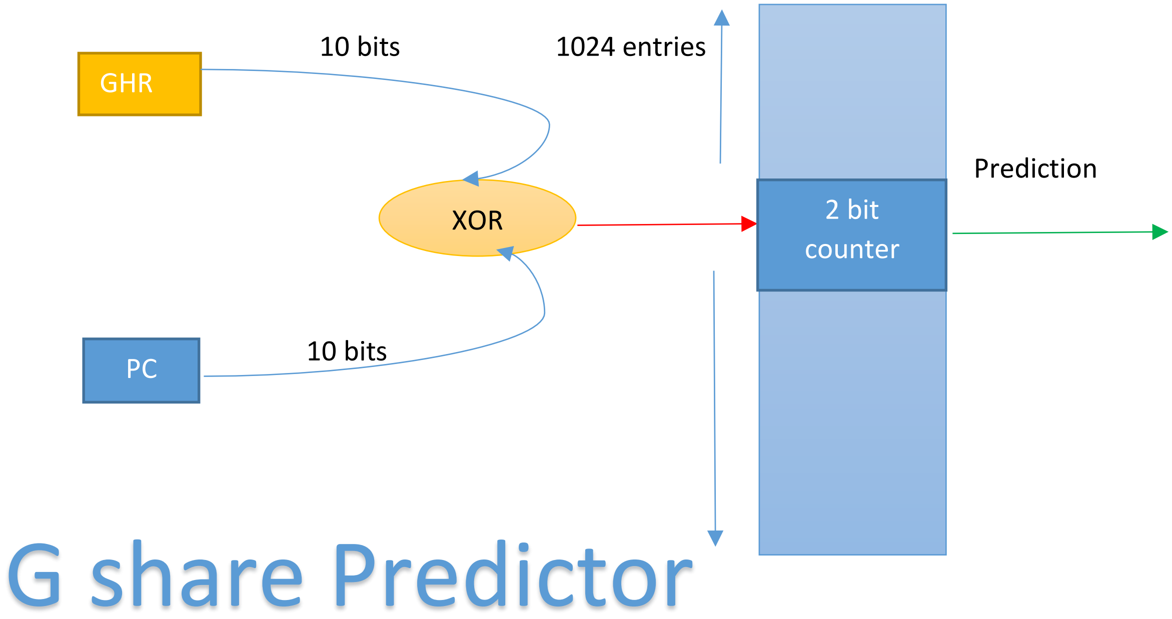
Ritik Jain
2021JCS2260

October 11, 2021

1 Implement a Branch Predictor

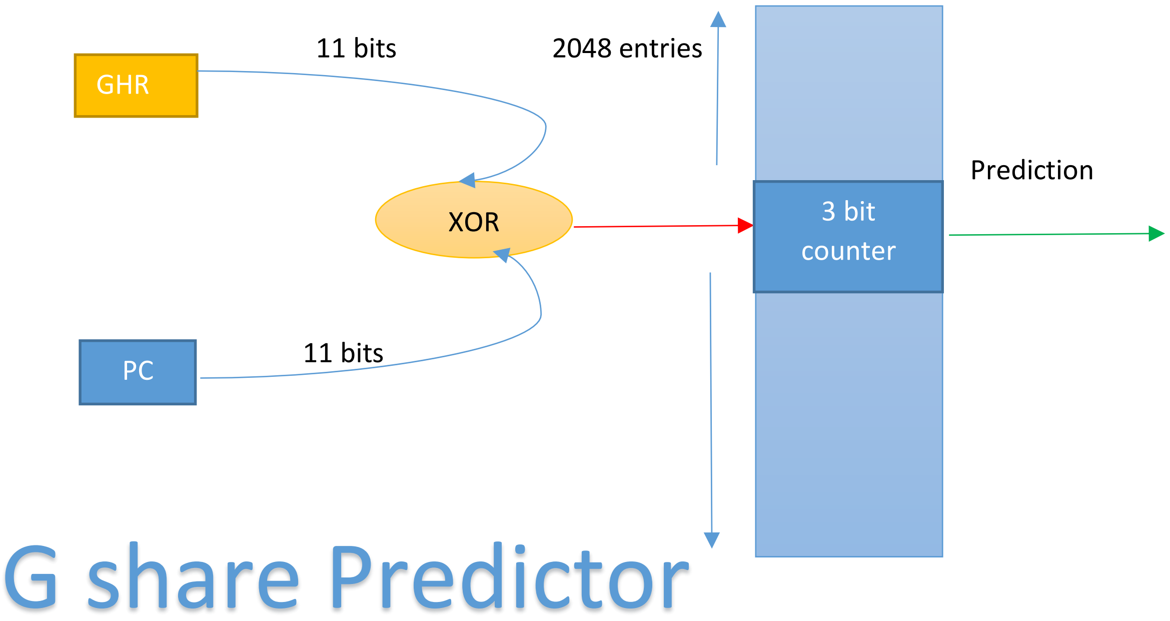
1.1 Part 1

1.1.1 2400 bits Predictor



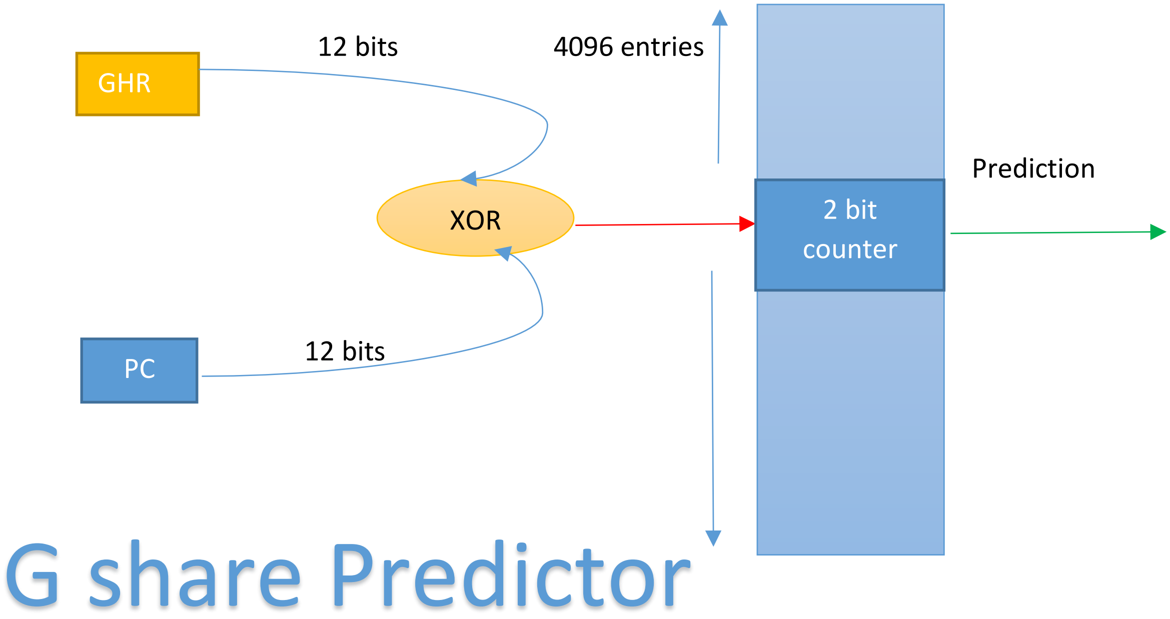
- Predictor used- GShare Predictor
- Saturating counter bits used- 2
- PC bits used-10
- GHR bits used-10
- Table formed-1024 entries
- Table size=1024*2= 2048 bits
- Total bits used = 2058 bits
- Expected Accuracy=94.84%
- Achieved Accuracy=96.38%

1.1.2 6400 bits Predictor



- Predictor used- GShare Predictor
- Saturating counter bits used- 3
- PC bits used-11
- GHR bits used-11
- Table formed-2048 entries
- Table size=2048*3= 6144 bits
- Total bits used = 6155 bits
- Expected Accuracy=95.13%
- Achieved Accuracy=97.14%

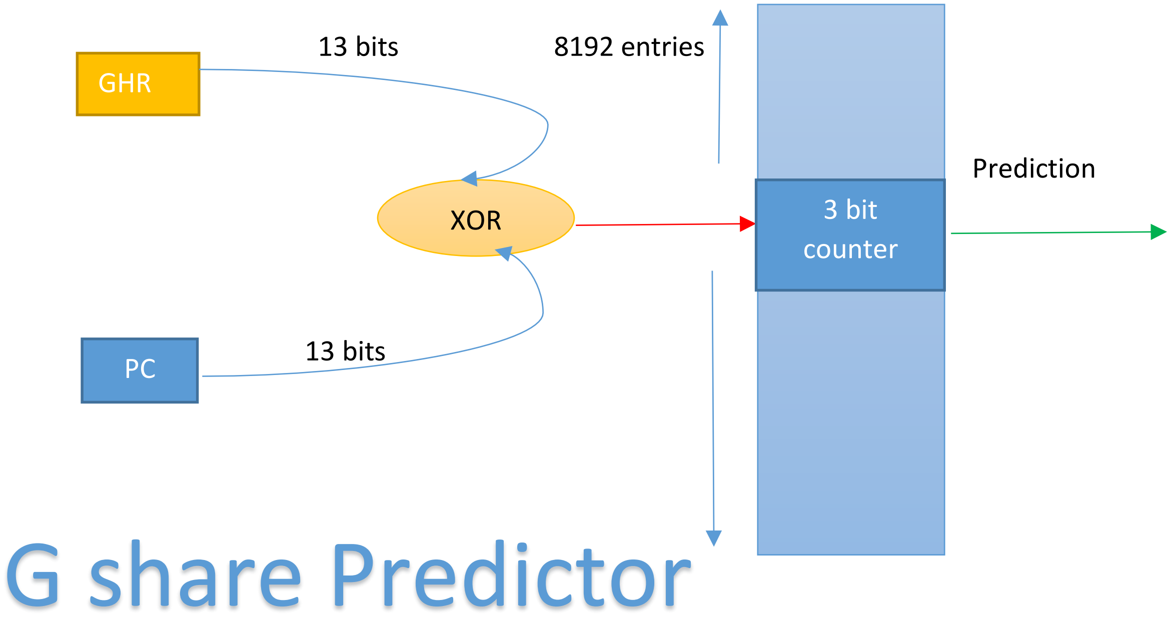
1.1.3 9999 bits Predictor



G share Predictor

- Predictor used- GShare Predictor
- Saturating counter bits used- 2
- PC bits used-12
- GHR bits used-12
- Table formed-4096 entries
- Table size=4096*2= 8192 bits
- Total bits used = 8204 bits
- Expected Accuracy=95.31%
- Achieved Accuracy=97.36%

1.1.4 32000 bits Predictor



G share Predictor

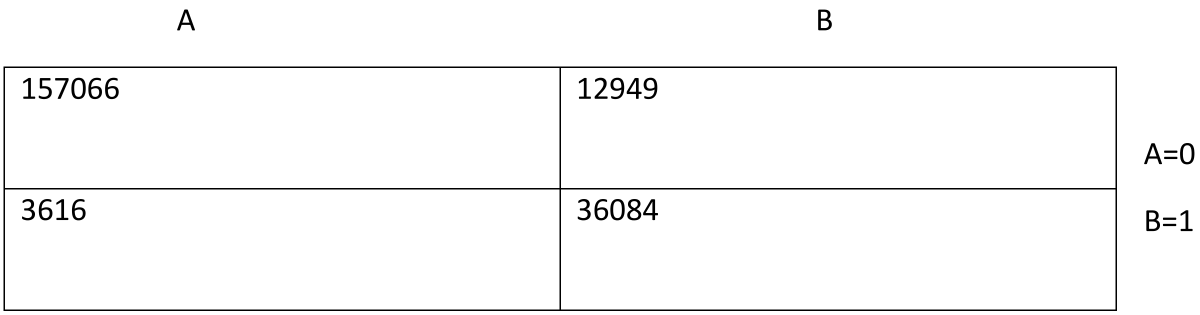
- Predictor used- GShare Predictor
- Saturating counter bits used- 3
- PC bits used-13
- GHR bits used-13
- Table formed-8192 entries
- Table size=8192*3= 24576 bits
- Total bits used = 24589 bits
- Expected Accuracy=95.12%
- Achieved Accuracy=97.85%

1.2 Comparing with a Machine Learning Algorithm

- I have used WEKA tool and first converted trace files to csv and then arff extension.
- Then for preprocessing I convert numeric to nominal data.

1.2.1 Trace 1

- ALgorithm Used: Decision Tree
- Instances: 1048575
- Testing mode: Percentage split 80% Training 20% Testing
- Correctly Classified Instances :193150
- Incorrectly Classified Instances:16565
- Accuracy: 92.10%
- Confusion Matrix:



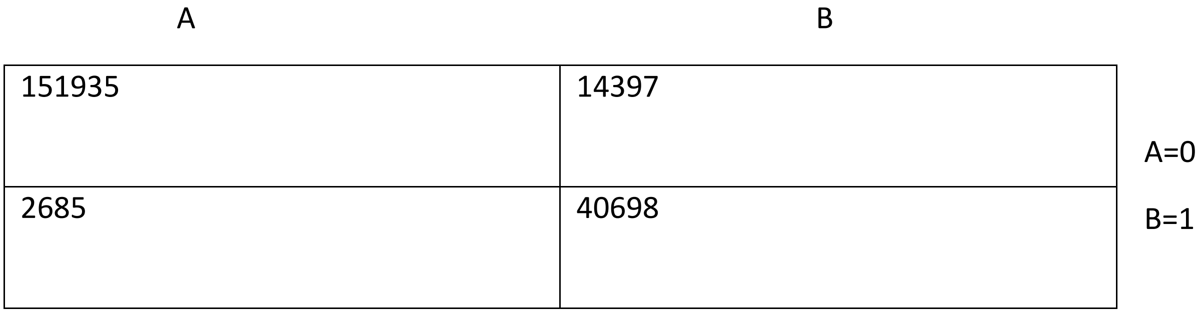
A	B	
157066	12949	A=0
3616	36084	B=1

Conclusions:

- Our algorithm works average in this case because there are few Type-1 and Type-2 errors.
- Majority of times it predicts branch is not taken(A=0).

1.2.2 Trace 2

- ALgorithm Used: Decision Tree
- Instances: 1048575
- Testing mode: Percentage split 80% Training 20% Testing
- Correctly Classified Instances :192633
- Incorrectly Classified Instances:17082
- Accuracy: 91.85%
- Confusion Matrix:



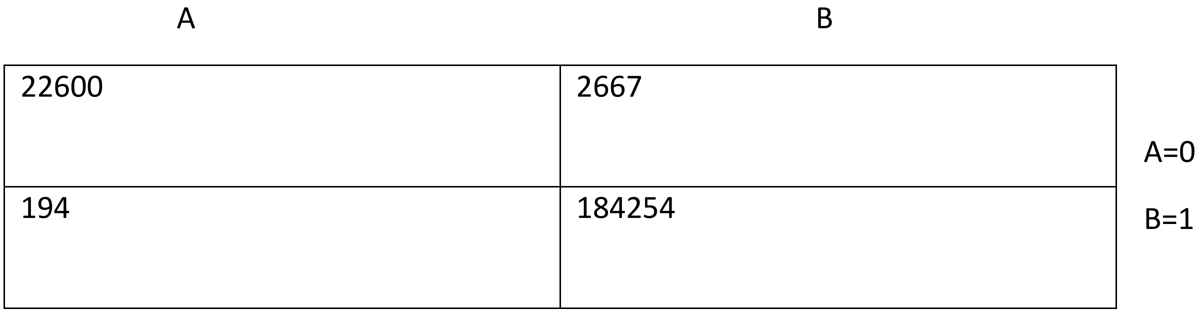
A	B	
151935	14397	A=0
2685	40698	B=1

Conclusions:

- Our algorithm works average in this case because there are few Type-1 and Type-2 errors.
- Majority of times it predicts branch is not taken(A=0).

1.2.3 Trace 3

- ALgorithm Used: Decision Tree
- Instances: 209715
- Testing mode: Percentage split 80% Training 20% Testing
- Correctly Classified Instances :206854
- Incorrectly Classified Instances:2861
- Accuracy: 98.63%
- Confusion Matrix:



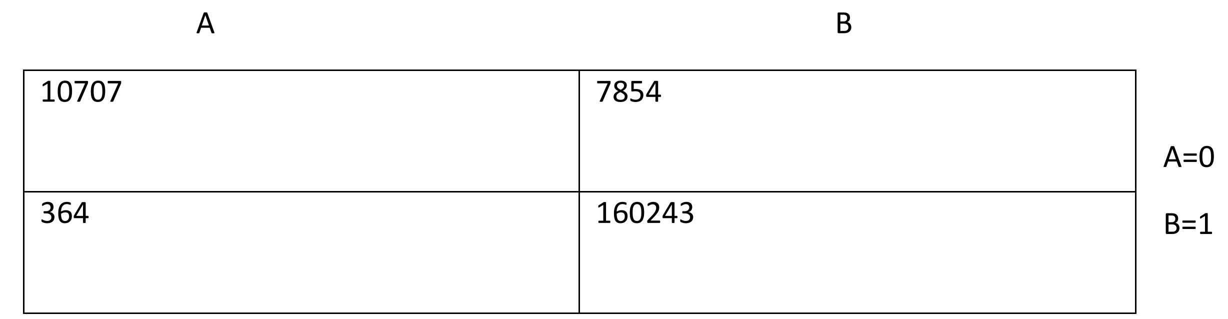
A	B	
22600	2667	A=0
194	184254	B=1

Conclusions:

- Our algorithm works good in this case because majority of them are True Positive and True Negative and achieves good accuracy.
- Majority of times it predicts branch is taken(B=1).

1.2.4 Trace 4

- ALgorithm Used: Decision Tree
- Instances: 179168
- Testing mode: Percentage split 80% Training 20% Testing
- Correctly Classified Instances :170950
- Incorrectly Classified Instances:8218
- Accuracy: 95.41%
- Confusion Matrix:



A	B
10707	7854
364	160243

A=0

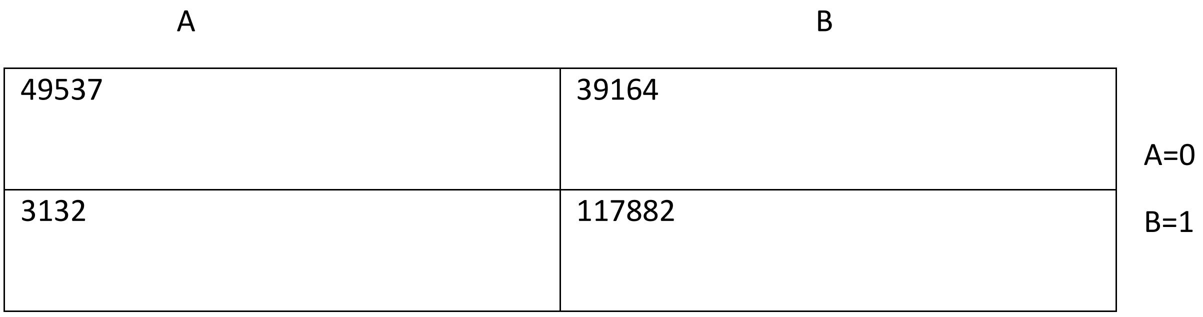
B=1

Conclusions:

- Our algorithm works fairly good in this case because there are few type-1 and type-2 errors.
- Majority of times it predicts branch is taken(B=1).

1.2.5 Trace 5

- ALgorithm Used: Decision Tree
- Instances: 209715
- Testing mode: Percentage split 80% Training 20% Testing
- Correctly Classified Instances :167419
- Incorrectly Classified Instances:42296
- Accuracy: 79.83%
- Confusion Matrix:



A	B	
49537	39164	A=0
3132	117882	B=1

Conclusions:

- Our algorithm works poorly in this case because there are lots of Type-1 and Type-2 error.
- Majority of times it predicts branch is taken(B=1).

2 Installed the Tejas Simulator and Understand its operation

- I have Installed Tejas 1.3 on Ubuntu
- I have tested on config.xml and change processor from outoforder and inorder interchangeably .
  Following are the observations taken on these Predictor:
- Predictor Type=TAGE
- PC Bits=32
- BHR size=13
- Saturating Bits=3

2.1 Frequency 6400 MHz

2.1.1 Inorder Results:

- Total Cycles Taken:495119 cycles
- Total IPC=0.2189 in terms of micro-ops
- Total IPC=0.1757 in terms of CISC instructions
- number of brances=19046
- number of mispredicted branches=1758
- branch Predictor Accuracy=90.7697%
- Time Taken: 77.3623 μs

2.1.2 OutofOrder Results:

- Total Cycles Taken:327876 cycles
- Total IPC=0.3310 in terms of micro-ops
- Total IPC=0.2652 in terms of CISC instructions
- number of brances=19039
- number of mispredicted branches=1664
- branch Predictor Accuracy=91.2600%
- Time Taken: 51.2306 μs

2.2 Frequency 4800 MHz

2.2.1 Inorder Results:

- Total Cycles Taken:444397 cycles
- Total IPC=0.2453 in terms of micro-ops
- Total IPC=0.1967 in terms of CISC instructions
- number of brances=19182
- number of mispredicted branches=1768
- branch Predictor Accuracy=90.7830%
- Time Taken: 92.5827 μs

2.2.2 OutofOrder Results:

- Total Cycles Taken:285162 cycles
- Total IPC=0.3822 in terms of micro-ops
- Total IPC=0.3066 in terms of CISC instructions
- number of brances=19177
- number of mispredicted branches=1675
- branch Predictor Accuracy=91.2656%
- Time Taken: 59.4088 μs

2.3 Conclusions

- OutofOrder Processor executes better than Inorder Processor in terms of IPC.
- OutofOrder takes less time than Inorder to execute.
- OutOforder takes less cycles than Inorder to execute.

3 References Used

- www.cs.waikato.ac.nz/ml/weka/
- https://www.cse.iitd.ac.in/ srsarangi/advbook/index.html
- http://www.cse.iitd.ac.in/tejas/overview.html