NC State University

Department of Electrical and Computer Engineering

ECE 521: Spring 2014

Project #2: Branch Prediction

by

<< ARAVIND SANKAR >>

NCSU Honor Pledge: "I have neither given nor received unauthorized aid on this test or assignment."

Student's electronic signature: Aravind Sankar

Course number: 521

.

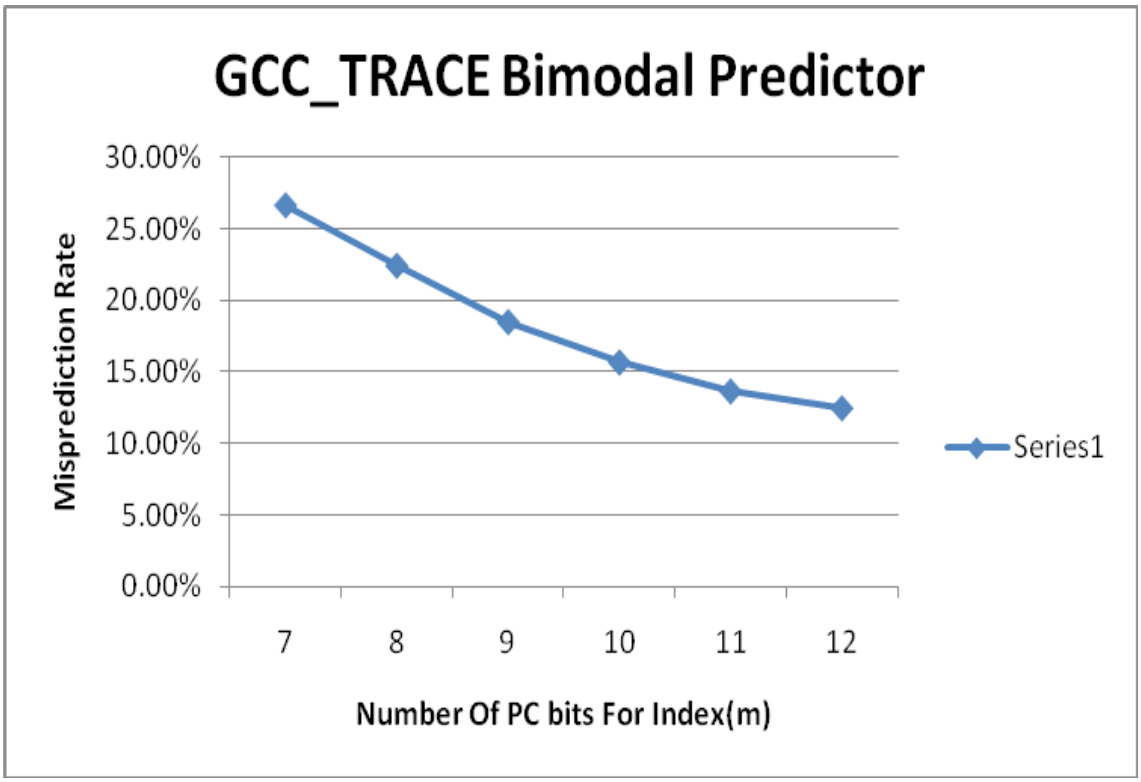
Bimodal Predictor

gcc_trace

The bimodal predictor was given the following different inputs.

Number Of PC bits(m)	Misprediction Rate
7	26.65%
8	22.43%
9	18.49%
10	15.67%
11	13.65%
12	12.47%

Below graph shows the behaviour of bimodal branch predictor for the above input parameters. The graph shows the misprediction rate for different m values.



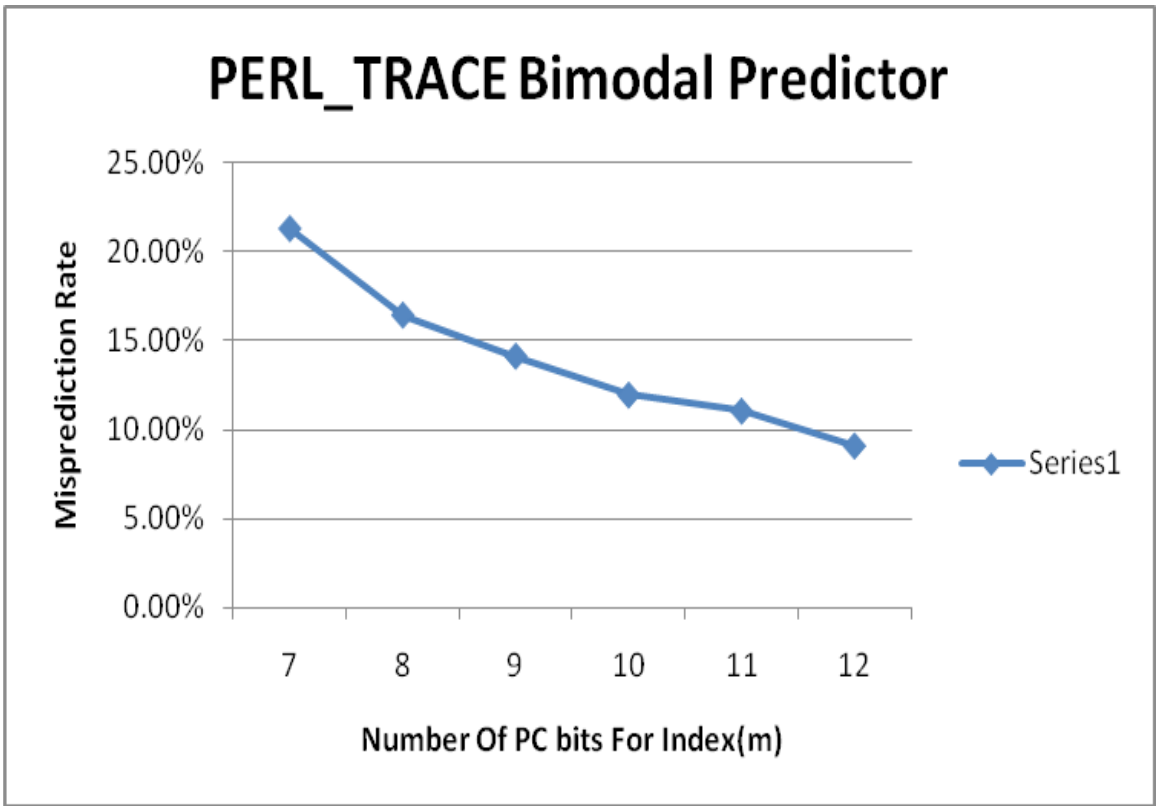
.

perl_trace

The bimodal predictor was given the following different inputs.

Number Of PC bits(m)	Misprediction Rate
7	21.31%
8	16.45%
9	14.14%
10	11.95%
11	11.05%
12	9.09%

Below graph shows the behaviour of bimodal branch predictor for the above input parameters. The graph shows the misprediction rate for different m values.



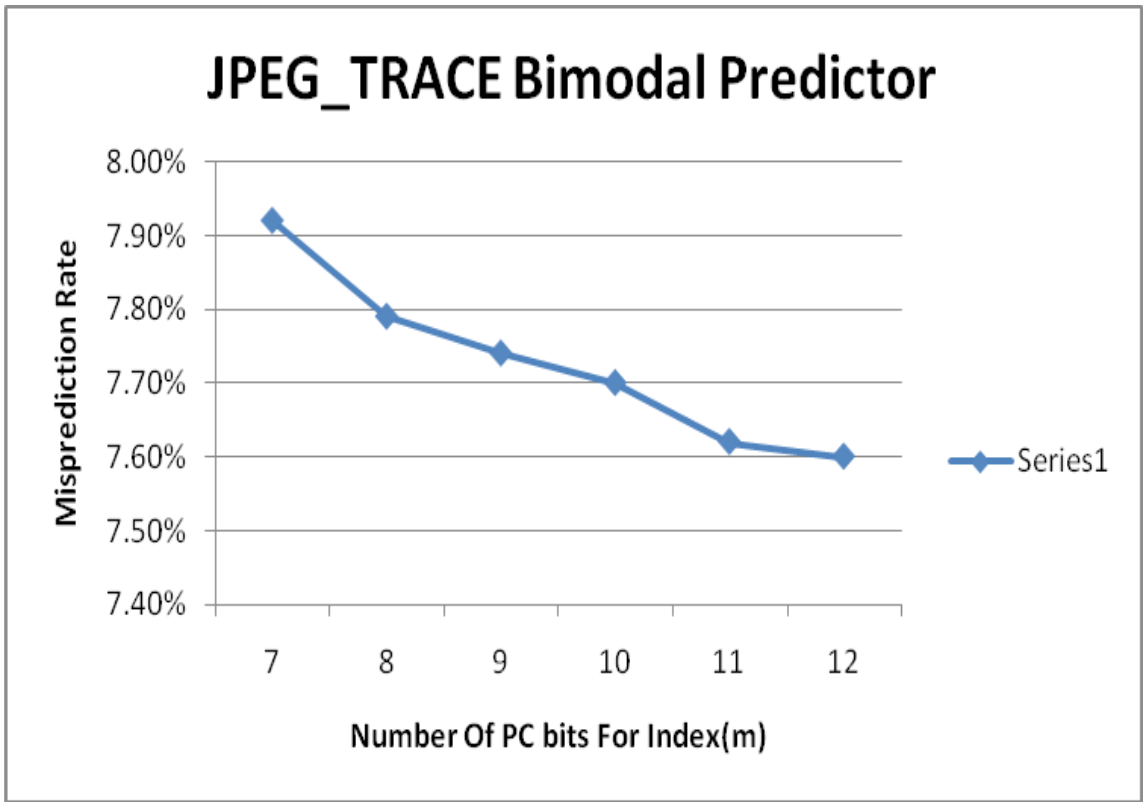
.

jpeg_trace

The bimodal predictor was given the following different inputs.

Number Of PC bits(m)	Misprediction Rate
7	7.92%
8	7.79%
9	7.74%
10	7.70%
11	7.62%
12	7.60%

Below graph shows the behaviour of bimodal branch predictor for the above input parameters. The graph shows the misprediction rate for different m values.



.

Analysis of Branch Predictor:

Based on the tables and graphs from gcc_trace & perl_trace, we can see that the mispreiction rate comes down as the value of m is increased. However, after a certain point the misprediction rate starts to level. Thus, we can say from this observation that the extra space used for the prediction table does not pay off for high m values.

Interestingly, for jpeg_trace we can see that the misprediction rate does not reduce much for higher bits of PC used to index the prediction table, i.e we can see diminishing returns. Thus, the performance of the predictor is better without increasing the PC bits, as the cost of additional space would easily override the low gain in performance.

Design:

For gcc_trace, the ideal configuration would be m=10 with misprediction rate at 15.67%, after which the misprediction rate does not reduce much and reaches the min 2% decrease. Hence, it is evident from the table and graph that we are safe to use m=10 rather than using higher number of bits and slowing the processing.

For perl_trace the ideal configuration would be m=10 with misprediction rate at 11.95%, after which the misprediction rate does not reduce much and reaches the min 2% decrease. Hence, it is evident from the table and graph that we are safe to use m=10 rather than using higher number of bits and slowing the processing.

For jpeg_trace the ideal configuration would be m=7 with misprediction rate at 7.92%, at which the misprediction starts to level off and there is only a 0.1% decrease overall as bits are added. Hence, it is evident from the table and graph that we are safe to use m=10 rather than using higher number of bits and slowing the processing.

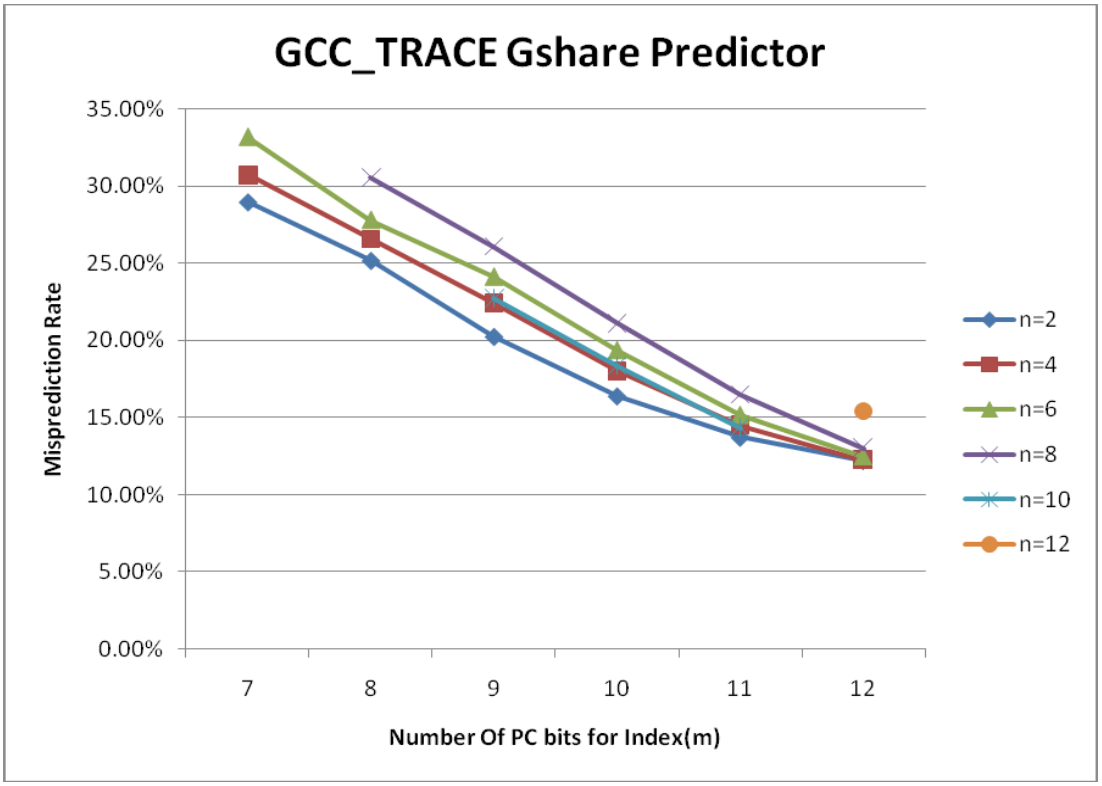
.

Gshare predictor

gcc_trace

The behaviour of gshare branch predictor for this file is shown below.

	M=7	M=8	M=9	M=10	M=11	M=12
N=2	28.98%	25.18%	20.25%	16.39%	13.71%	12.20%
N=4	30.76%	26.57%	22.43%	17.99%	14.49%	12.23%
N=6	33.22%	27.82%	24.14%	19.36%	15.14%	12.46%
N=8		30.56%	26.08%	21.10%	16.47%	13.00%
N=10				22.77%	18.34%	14.33%
N=12						15.40%



.

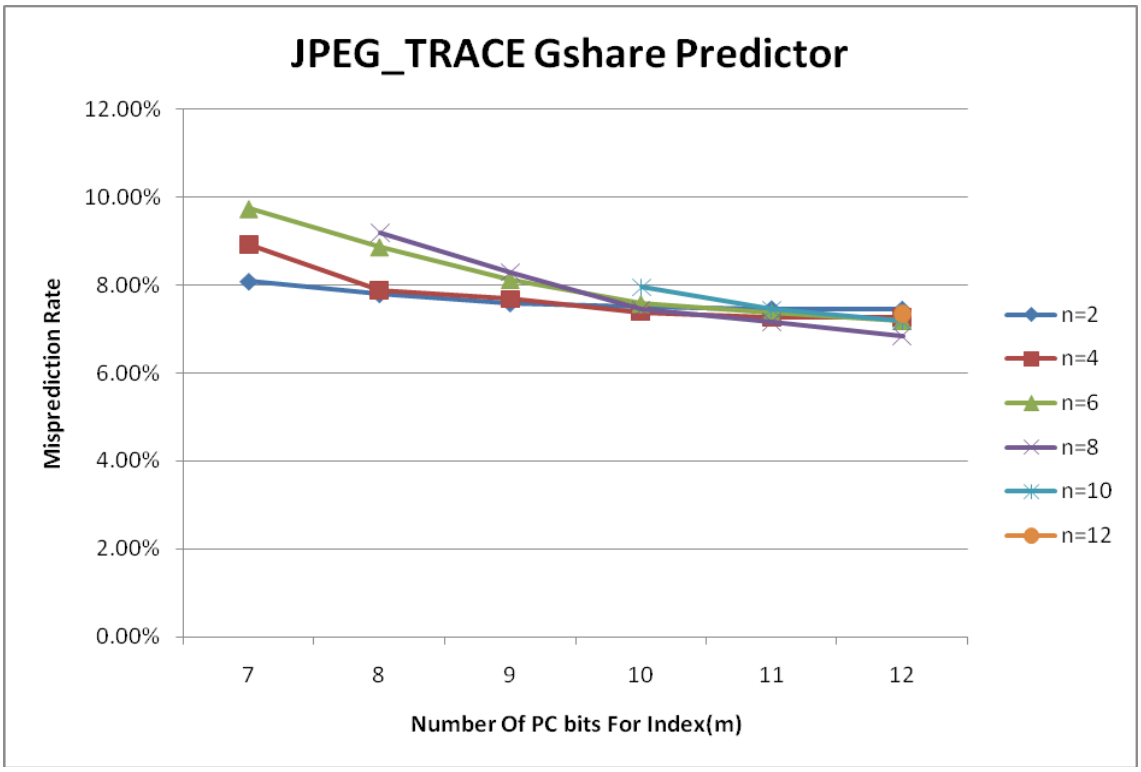
perl_trace

	M=7	M=8	M=9	M=10	M=11	M=12
N=2	24.34%	16.92%	13.57%	10.63%	10.11%	9.03%
N=4	25.96%	19.09%	14.68%	11.35%	9.68%	8.09%
N=6	28.71%	20.45%	16.25%	11.52%	8.60%	7.50%
N=8		24.79%	17.66%	12.42%	9.00%	6.49%
N=10				14.57%	8.98%	6.71%
N=12						7.16%

jpeg_trace

	M=7	M=8	M=9	M=10	M=11	M=12
N=2	8.08%	7.79%	7.58%	7.49%	7.45%	7.44%
N=4	8.92%	7.88%	7.68%	7.38%	7.27%	7.26%
N=6	9.74%	8.87%	8.13%	7.58%	7.38%	7.19%
N=8		9.20%	8.30%	7.45%	7.17%	6.84%
N=10				7.95%	7.44%	7.18%
N=12						7.35%

.



Analysis of Branch Predictor

Based on the tables and graphs of gcc_trace & perl_trace, it is evident that as we increase the number of bits for PC and the number of bhr bits N, the misprediction rate reduces but after a certain point we get diminishing returns. Also to be noted is that when "m" is kept constant and "n" is varied the misprediction rate starts increasing, only a little performance benefit is obtained at the cost of adding the extra bit.

For the jpeg_trace right, there is very little performance benefit as we increase the number of bits of PC and the number of bhr bits. We can see that when the misprediction rate does not improve by more than 2%, it is best to utilize that value of "m", "n" so that the overhead of additional space can be avoided.

Design

For gcc_trace the ideal configuration would be m=11 and n=2 with misprediction rate at 13.71%, after which we get diminishing returns and the misprediction rate reaches 2% decrease. So we are safe to use m=11 and n=2 and avoid further usage of bits and slowing the processing down by adding more bits(this is evident from the table and graph).

For perl_trace the ideal configuration would be m=11 and n=6 with misprediction rate at 8.60%. It is best to use m=11 and n=6 where the space effeciency is more and the also good results of misprediction rate are obtained.

.

For jpeg_trace the ideal configuration would be m=7 and n=2 with misprediction rate at 8.08%. The best configuration to use m=7 & n=2, this avoids further usage of bits and slowing the processing down by adding more bits.

.