NC State University

Department of Electrical and Computer Engineering

ECE 463/563: Fall 2019 (Dr. Huiyang Zhou)

Project #2: Branch Prediction

By

Pratik Suresh Raut
200322512

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

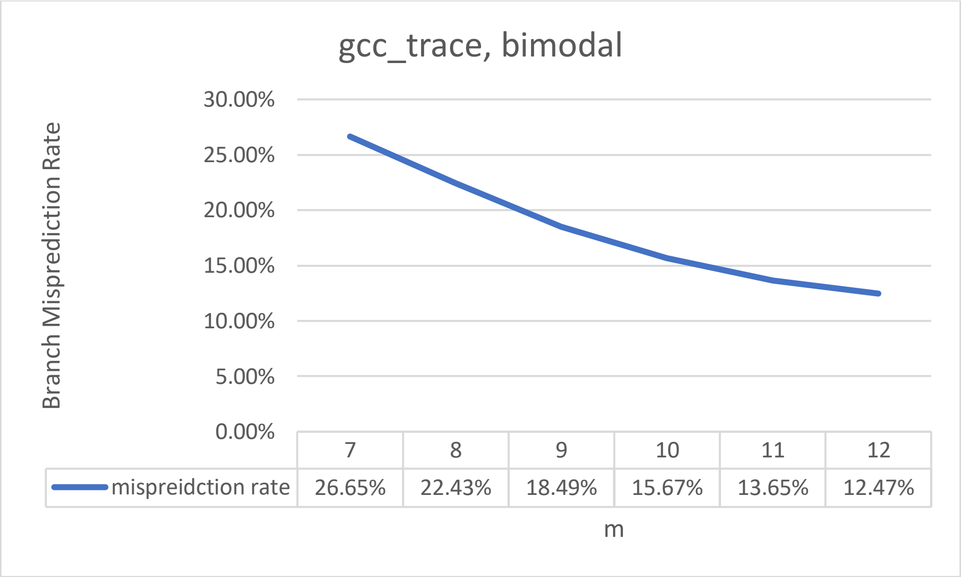
Student’s Electronic Signature: Pratik Suresh Raut

Course Number: 563

1. Bimodal Branch Predictor

Benchmark: *gcc_trace*

- **Graph**



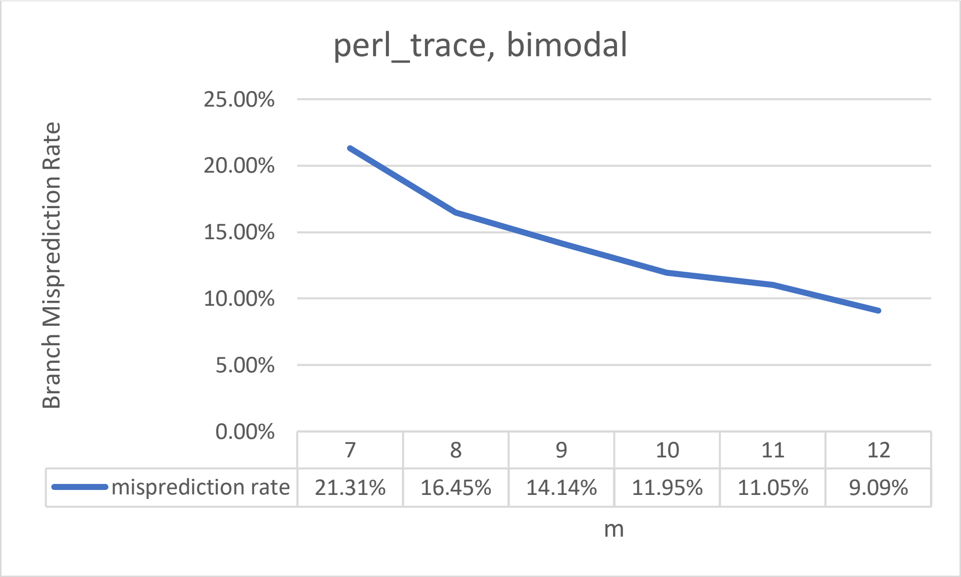
- **Design:**

m	misprediction rate
7	26.65%
8	22.43%
9	18.49%
10	15.67%
11	13.65%
12	12.47%
13	11.72%
14	11.37%
15	11.30%
16	11.21%

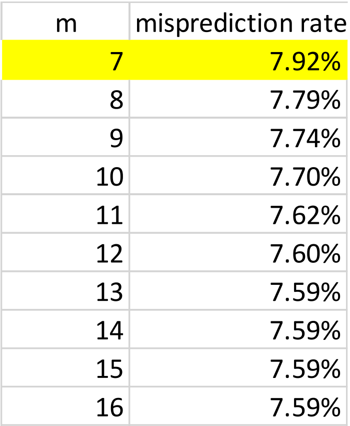
From the above table, we can observe that, as the value of m, i.e. number of pc bits used to index the bimodal prediction table increases from 7 to 12, we can see a substantial decrease in misprediction rate. When m is increased beyond 12, there is a negligible change in misprediction rate. Hence, for *gcc_trace*, bimodal branch predictor with m = 12 gives the best design with a considerably less value of misprediction rate.

Benchmark: *jpeg_trace*

- Graph



- **Design:**

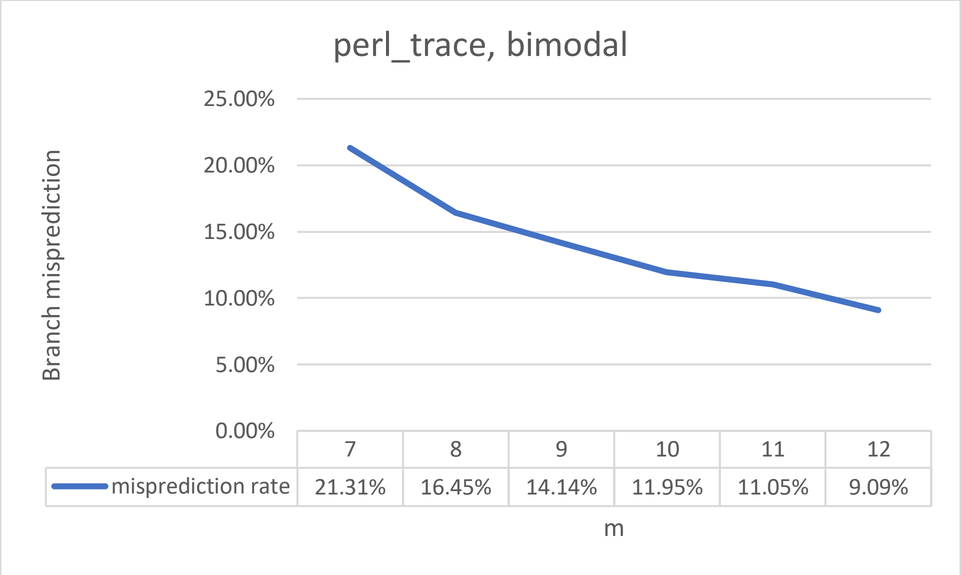


m	misprediction rate
7	7.92%
8	7.79%
9	7.74%
10	7.70%
11	7.62%
12	7.60%
13	7.59%
14	7.59%
15	7.59%
16	7.59%

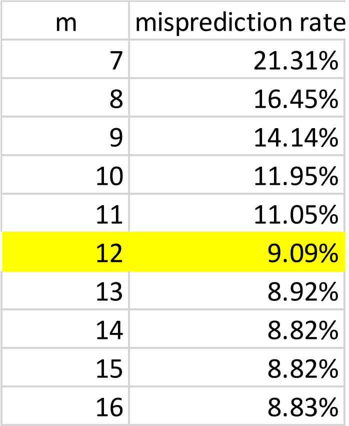
In case of *jpeg_trace*, the minimum value for the trade-off is attained for a value of m = 7. The lowest value of ‘m’ gives the best trade-off in this case since the value of the misprediction rate for the *jpeg_trac*e does not reduce much with increase in the value of m. The reason behind this might be because of the lack of unique branch addresses or high looping structure of the *jpeg_trace*. Reoccurrence of similar pc addresses leads to a low misprediction rate which allows the bimodal predictor counter to be trained well. Thus, m=7 results in an optimized design for this benchmark.

Benchmark: *perl_trace*

- **Graph**



- **Design:**



m	misprediction rate
7	21.31%
8	16.45%
9	14.14%
10	11.95%
11	11.05%
12	9.09%
13	8.92%
14	8.82%
15	8.82%
16	8.83%

The *perl_trace* follows a similar trend to the *gcc_trace*. Initially, increasing the value of ‘m’ gives a good trade-off between the predictor size and the misprediction rate. For m greater than 12, the miss prediction rate saturates around ~9%. Thus, m=12 gives a good design for *perl_trace.*

- **Analysis of Bimodal Predictor Performance:**

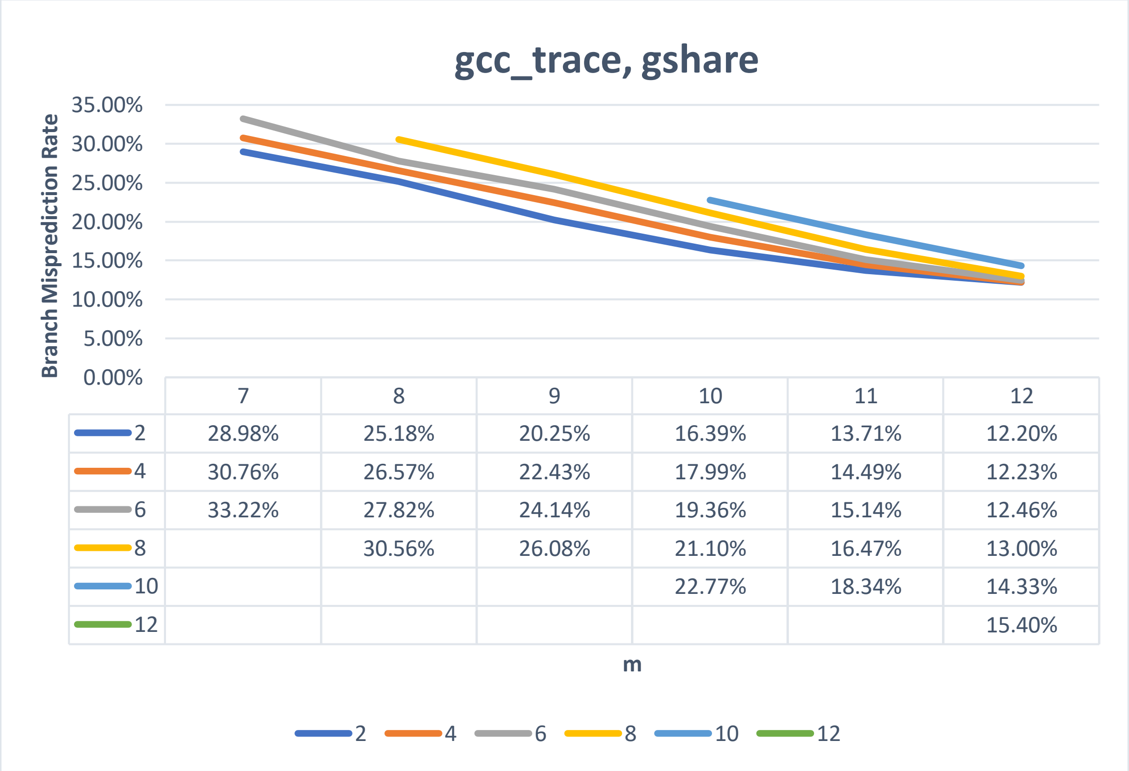
  From the above graphs, we can observe that the misprediction rate for all three address traces reduces as the number of PC bits used to index the branch prediction is increased. Using a greater number of bits to index the prediction table means there will be a greater number of counter values in the branch history table. As a result, addresses will map to different rows in the branch history table, thus leading to better prediction. Also, the misprediction rate for the jpeg_ trace is lower than either gcc_trace or perl-trace.

  From the comparison graph, we can observe that for a size of m=12, the misprediction rate for all the three traces stabilizes, and there is not much improvement on account of increasing the value of ‘m’. Thus, m=12 gives the best configuration for these three address traces.

2. Gshare Branch Predictor

Benchmark: *gcc_trace*

- **Graph**



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

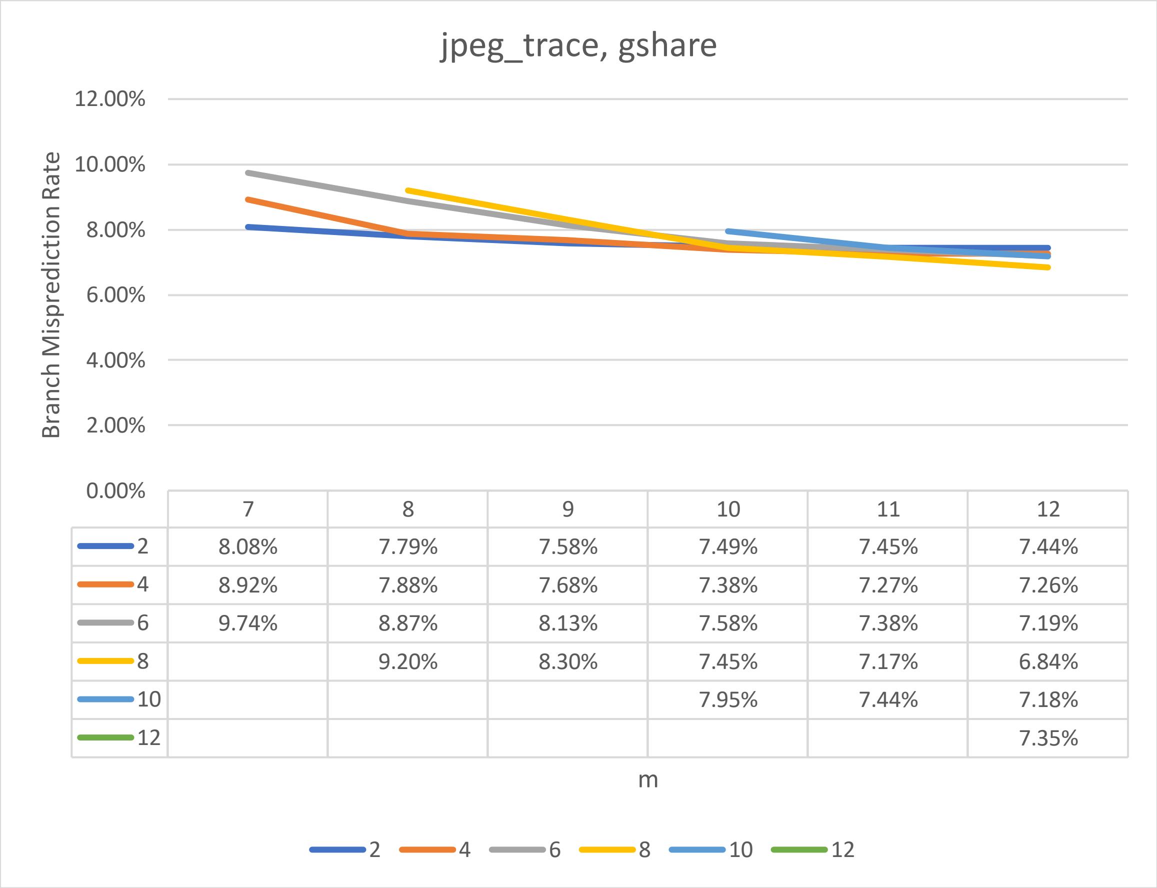
- **Design:**

m↓n →	2	4	6	8	10	12	14	16
7	28.98%	30.76%	33.22%					
8	25.18%	26.57%	27.82%	30.56%				
9	20.25%	22.43%	24.14%	26.08%				
10	16.39%	17.99%	19.36%	21.10%	22.77%			
11	13.71%	14.49%	15.14%	16.47%	18.34%			
12	12.20%	12.23%	12.46%	13.00%	14.33%	15.40%		
13	11.11%	10.57%	10.59%	11.00%	11.68%	12.68%		
14	10.42%	9.69%	9.08%	9.34%	9.83%	10.48%	11.13%	
15	10.13%	9.13%	8.30%	8.22%	8.46%	9.01%	9.48%	
16	9.93%	8.77%	7.89%	7.57%	7.61%	7.86%	8.34%	8.75%

At m=16 and n=8, we can achieve the least miss prediction rate after observing the graphs. In this case, the number of bits used is more, hence it leads to greater table size. In order to achieve both optimum number of bits used to index the table and a minimum misprediction rate, m = 13 and n = 4 can be used. Higher m and lower n work for this benchmark.

Benchmark: *jpeg_trace*

- **Graph**



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

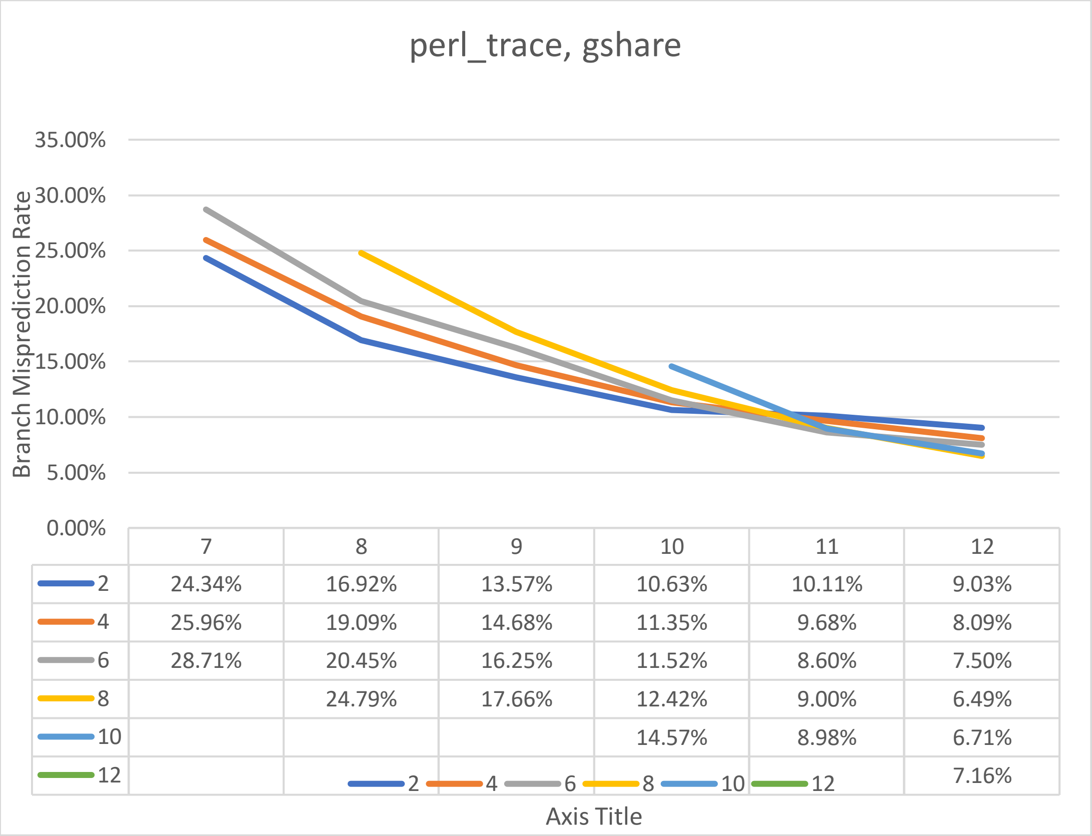
- **Design:**

m↓n →	2	4	6	8	10	12	16
7	8.08%	8.92%	9.74%				
8	7.79%	7.88%	8.87%	9.20%			
9	7.58%	7.68%	8.13%	8.30%			
10	7.49%	7.38%	7.58%	7.45%	7.95%		
11	7.45%	7.27%	7.38%	7.17%	7.44%		
12	7.44%	7.26%	7.19%	6.84%	7.18%	7.35%	
13	7.33%	7.24%	7.16%	6.83%	7.02%	7.17%	
14	7.32%	7.17%	7.14%	6.69%	6.84%	6.84%	6.93%
15	7.31%	7.13%	7.09%	6.69%	6.72%	6.70%	6.67%
16	7.31%	7.13%	7.08%	6.65%	6.70%	6.66%	6.57%

At m=10 and n=4, we can deduce that this configuration result into relatively low miss prediction rate. This gives a optimum configuration for the design of gshare predictor for *jpeg_trace*.

Benchmark: *perl_trace*

- **Graph**



- **Design:**

m↓n →	2	4	6	8	10	12	14	16
7	24.34%	25.96%	28.71%					
8	16.92%	19.09%	20.45%	24.79%				
9	13.57%	14.68%	16.25%	17.66%				
10	10.63%	11.35%	11.52%	12.42%	14.57%			
11	10.11%	9.68%	8.60%	9.00%	8.98%			
12	9.03%	8.09%	7.50%	6.49%	6.71%	7.16%		
13	9.23%	7.27%	6.09%	5.26%	4.92%	5.09%		
14	8.07%	7.35%	5.43%	4.51%	3.80%	4.30%	3.75%	
15	8.02%	7.28%	5.71%	4.13%	3.58%	3.35%	3.58%	
16	8.04%	6.54%	5.07%	4.12%	3.84%	3.53%	3.01%	2.91%

The *perl_trace* follows a similar trend to the *gcc_trace*. Analyzing the trade-off between the predictor size and misprediction rates, we find that the configuration n=16, m=16 provides the best possible value. However, the size of the predictor for a configuration of n=16, m=16 exceeds the upper bound of 16KB area budget for the predictor. Hence, at m=13 and n=10, we can observe that the miss prediction rate is the lowest, i.e., optimized configuration under the given constraints.

- **Analysis of Gshare Predictor Performance:**
  Looking at the misprediction rates for the three benchmarks, we observe that the *gcc_trace* benchmark has a higher misprediction rate compared to *jpeg_trace* and *perl_trace*. This observation is similar to the Bimodal predictor. This is quite expected since we have just changed the indexing strategy in gshare, and not the predictor counter size (2 bits) or the algorithm to set the counter. In gshare, the minimum achieved misprediction rate is lower than the minimum achieved in bimodal. This is due to the better indexing strategy employed in gshare.