NC State University

Department of Electrical and Computer Engineering

ECE 463/563: Fall 2018 (Rotenberg)

Project #2: Branch Prediction

by

<< Mihir Manoj Joshi >>

Student Id : 200253307

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

Student's electronic signature: ____Mihir Manoj Joshi_____
(sign by typing your name)

Course number: _____563_____
(463 or 563 ?)

Student Id: 2002 53307

Branch Predicton

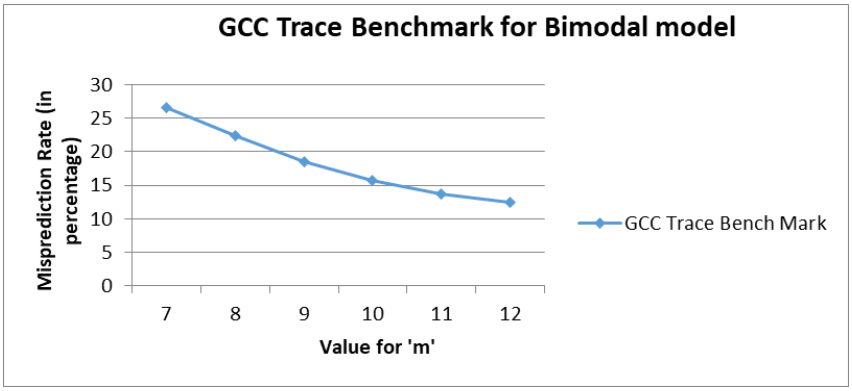
Branch Prediction is a general technique in which a digital circuit tries to guess the outcome of the branch before it is known definitely. It is basically used to improve the flow of the instruction pipeline. It plays a critical role in improving the performance of the pipelined processors.

Bimodal Branch Predictors:

Bimodal Branch Predictor basically uses a 2 bit counter where the prediction takes place from 0 to 1 if the actual outcome is 'taken' or goes from 1 to 0 if the actual outcome is ' not taken.' A graph is plotted w.r.t value of log2(m) [X-axis] and misprediction rate [Y-axis].

- Graphs & Best Design For Bimodal

1. GCC Trace

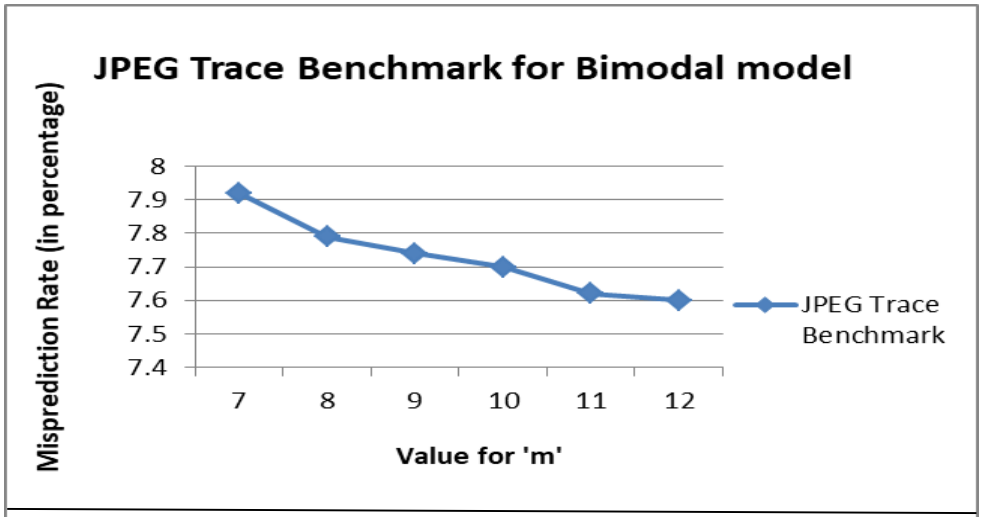


The table below shows the misprediction rate for each value of 'm'. In order to minimize the value of 'm' as well as the value of misprediction rate we consider their product.

Value of m (X-axis)	Value of misprediction rate (Y-axis)	Product
7	26.65	186.55
8	22.43	179.44
9	18.49	166.41
10	15.67	156.7
11	13.65	150.15
12	12.47	149.64
13	11.72	152.36
14	11.37	159.18
15	11.3	169.5
16	11.21	179.36

Conclusion For Best Design: With the value of **'m' = 12** whose misprediction rate is 12.47% is considered optimal for our design.

2. Jpeg Trace



The table below shows the misprediction rate for each value of 'm'. In order to minimize the value of 'm' as well as the value of misprediction rate we consider their product

Value of m (X-axis)	Value of misprediction rate (Y-axis)	Product
7	7.92	55.44
8	7.79	62.32
9	7.74	69.66
10	7.7	77
11	7.62	83.82
12	7.6	91.2
13	7.59	98.67
14	7.59	106.26
15	7.59	113.85
16	7.59	121.44

Conclusion For Best Design: With the value of **'m' = 7** whose misprediction rate is 55.44% is considered optimal for our design.

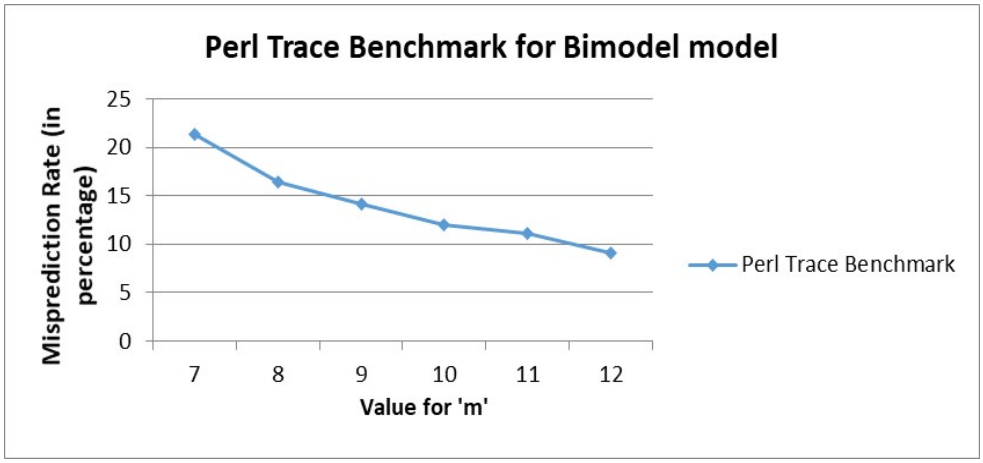
Observation For Jpeg

The lowest value of 'm' is a trade of since there is no possible increase in the rate of decrement in misprediction rate with increase in size of 'm'. Possible reasons are mentioned as follows:

i. The trace file doesn't has unique addresses
ii. There might be a looping structure or a pattern is developed among the addresses.

Because of these reasons the branch predictor is trained well and there is no further decrease in misprediction rate. Thus, there are diminishing returns with increasing the value of 'm' from the initial value of m=7 itself.

3. Perl Trace



The table below shows the misprediction rate for each value of ‘m’. In order to minimize the value of ‘m’ as well as the value of misprediction rate we consider their product.

Value of m (X-axis)	Value of misprediction rate (Y-axis)	Product
7	21.31	149.17
8	16.45	131.6
9	14.14	127.26
10	11.95	119.5
11	11.05	121.55
12	9.09	109.08
13	8.92	115.96
14	8.82	123.48
15	8.82	132.3
16	8.83	141.28

Conclusion For Best Design: With the value of **‘m’ = 12** whose misprediction rate is 9.09% is considered optimal for our design.

- Analysis And Conclusion:

From the graph we make three follwing observations

1. The rate of misprediction decresesas as the size of bimodal counter increases. As the size of bimodal counter increases there will be more chances of unique address to be generated and more blocks for the addresses to be allocated which is the reason for decrease in mispredicition rate. Large size of counter eventually leads to addresses to be mapped to different rows in branch history table leading to a better prediction chance.
2. Some kinds of unique adresses can be easily captured by Bimdoal branch prediction where as for some it requires an alternative branch predictor to do so.
3. At m=12, the graphs for all the three bench marks come to a saturation point beyond which a slight increase in hardware would produce diminishing result in performance (or decrease in value of mispredicitон rate).

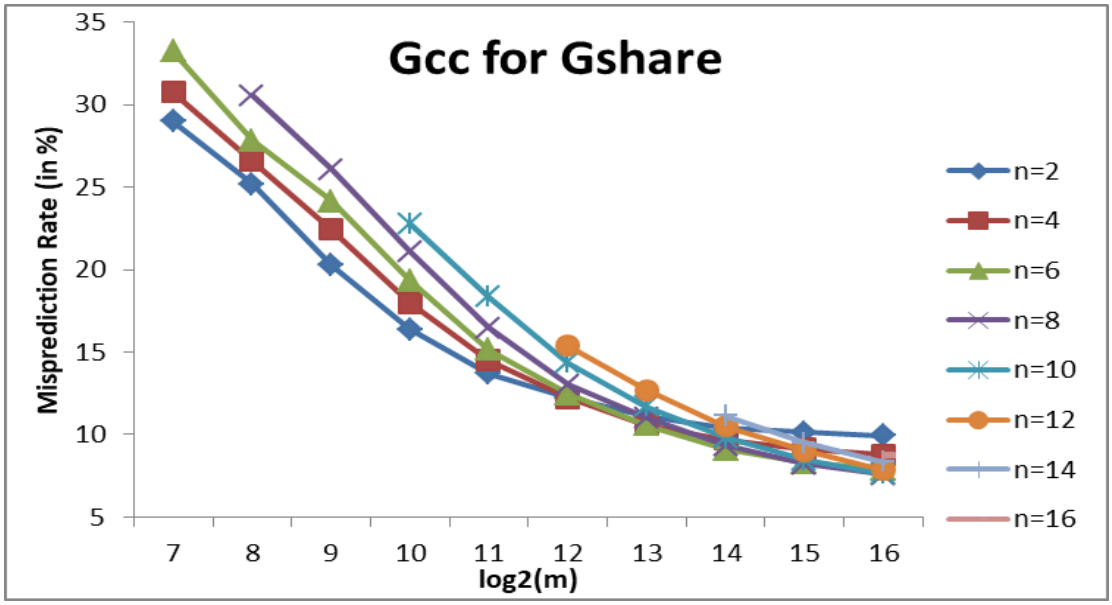
2. G_Share Branch Predictors

The G_Share predictor follows modulus 2 operator. The 'n' bits from the PC address along with the global branch history register in order to generate the index for the branch history table. In the case of a G_Share predictor, the total storage overhead on account of the branch prediction circuitry is n + 2*2^(m).

Assuming a fact that the value of 'n' is negligible compared to the exponential, we can assume the storage overhead to be 2*2^(m).

- Graphs & Best Design For Bimodal

1. Gcc Trace File

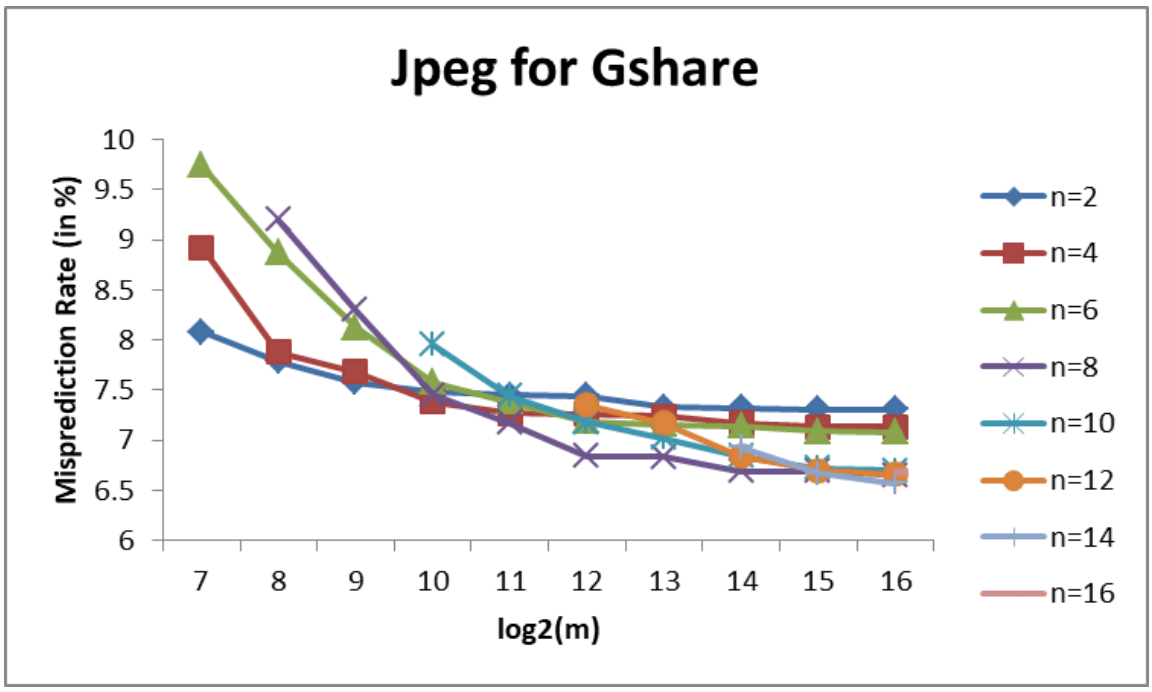


The table below shows the misprediction rate for each value of 'm' and 'n'. In order to minimize the value of 'm' as well as the value of misprediction rate we consider their product

Gcc Trace Benchmark										
	7	8	9	10	11	12	13	14	15	16
n=2	28.98	25.18	20.25	16.39	13.71	12.2	11.11	10.42	10.13	9.93
n=4	30.76	26.57	22.43	17.99	14.49	12.23	10.57	9.69	9.13	8.77
n=6	33.22	27.82	24.14	19.36	15.14	12.46	10.59	9.08	8.3	7.89
n=8		30.56	26.08	21.1	16.47	13	11	9.34	8.22	7.57
n=10				22.77	18.34	14.33	11.68	9.83	8.46	7.61
n=12						15.4	12.68	10.48	9.01	7.86
n=14								11.13	9.48	8.34
n=16										8.75
Best Design				n=8	m=16	7.57				

Conclusion for Best Design: With the value of **'m' = 16 and 'n' = 8** whose misprediction rate is 7.57 % is considered optimal for our design

2. Jpeg Trace

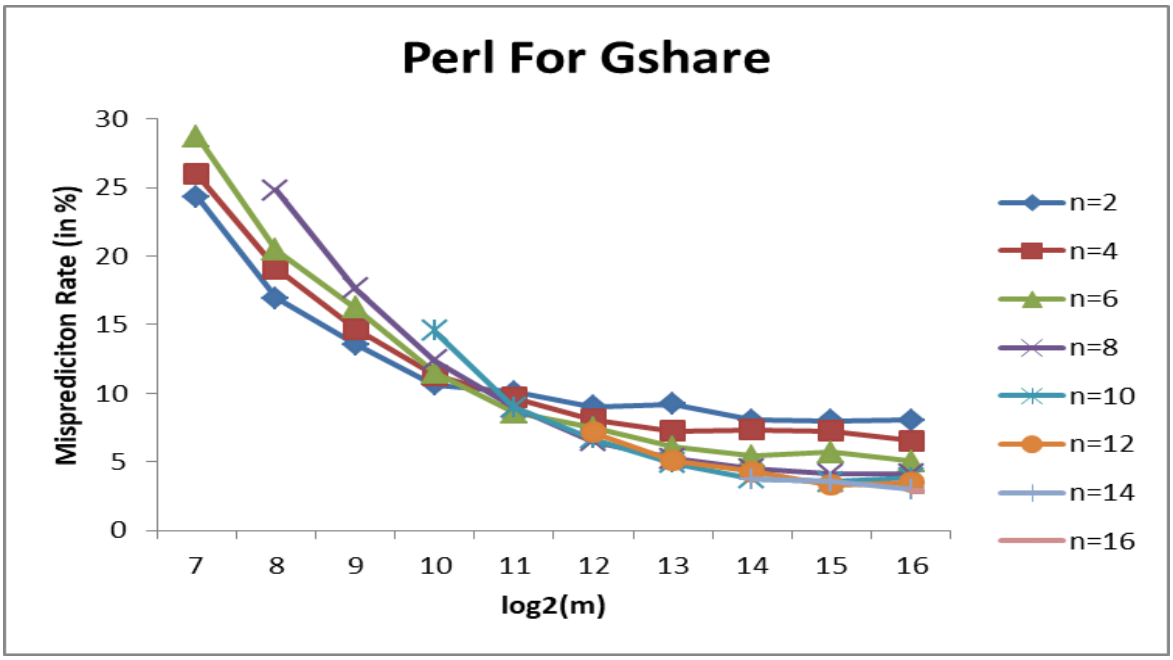


The table below shows the misprediction rate for each value of 'm' and 'n'. In order to minimize the value of 'm' as well as the value of misprediction rate we consider their product.

Jpeg Trace Benchmark										
	7	8	9	10	11	12	13	14	15	16
n=2	8.08	7.79	7.58	7.49	7.45	7.44	7.33	7.32	7.31	7.31
n=4	8.92	7.88	7.68	7.38	7.27	7.26	7.24	7.17	7.13	7.13
n=6	9.74	8.87	8.13	7.58	7.38	7.19	7.16	7.14	7.09	7.08
n=8		9.2	8.3	7.45	7.17	6.84	6.83	6.69	6.69	6.65
n=10				7.95	7.44	7.18	7.02	6.84	6.72	6.7
n=12						7.35	7.17	6.84	6.7	6.66
n=14								6.93	6.67	6.57
n=16										6.68
Best Design				n=14	m=16	6.57				

Conclusion for Best Design: With the value of **'m' = 7 and 'n' = 2** whose misprediction rate is 8.08% is considered optimal for our design.

3. Perl Trace



The table below shows the misprediction rate for each value of 'm' and 'n'. In order to minimize the value of 'm' as well as the value of misprediction rate we consider their product.

Perl Trace Benchmark										
	7	8	9	10	11	12	13	14	15	16
n=2	24.34	16.92	13.57	10.63	10.11	9.03	9.23	8.07	8.02	8.04
n=4	25.96	19.09	14.68	11.35	9.68	8.09	7.27	7.35	7.28	6.54
n=6	28.71	20.45	16.25	11.52	8.6	7.5	6.09	5.43	5.71	5.07
n=8		24.79	17.66	12.42	9	6.49	5.26	4.51	4.13	4.12
n=10				14.57	8.98	6.71	4.92	3.8	3.58	3.84
n=12						7.16	5.09	4.3	3.35	3.53
n=14								3.75	3.58	3.01
n=16										2.91
Best Design				n=16	m=16	2.91				

Conclusion for Best Design: With the value of **'m' = 16 and 'n' = 16** whose misprediction rate is 2.91 % is considered optimal for our design.

Analysis of GShare Predictor Performance:
Looking at the misprediction rates for the three benchmarks, we make following obserations:

1. The Gcc benchmark has a little bit higher misprediction rate compared to Jpeg and Perl.
2. Due to a different indexing strategy employed in G_Share is responsible for the above expected result mentioned in point 1.Comparing values of mispredicttion of G_Share and Bimodal it is seen that the minimum achieved misprediction rate is lower than the minimum achieved in Bimodal. We can conclude at this it is due to the better indexing strategy or technique which is employed in GShare.
3. Looking at the overall trend across all the three address traces, the best trade-off between predictor size and the misprediction rate is attained at a value of **m=16 and n=8 (**though can be ignored).

Conclusion For Project:

1. For Bimodal, best design was at m=12.
   Size and Design Analysis of Bimodal Branch Predictor = [2 * 2^(12)]/[1 Byte]
   = [2^13]/[2^3]
   = [2^10]
   = 1024 Bytes
2. For G_Share, best design was at m=16 and n=8.
   Size and Design Analysis of Bimodal Branch Predictor = [8+2 * 2^(16)]/[1 Byte]
   ~= [2^17]/[2^3]
   ~= [2^14]
   ~= (1024*16) Bytes
   ~= 16384 Bytes

Summary

Branch Predictor	Best Configration	Size(in Bytes)	Size (in Kilobytes)
G_Share	n=16 and m=8	16384	16
Bimodal	m=12	1024	1

Rule	Spec. Section	Did you follow this rule? (Yes/No)
The output from your simulator includes the branch predictor configuration, the final contents of the branch predictor, and all required measurements	6.1	
The output from your simulator matches EXACTLY both the formatting and numerical values of the posted validation runs. The only exception is that you are permitted a variable amount of whitespace (tabs and spaces), although newlines must match (*i.e.*, no extra lines).	6.1	
You explicitly confirmed that your outputs match EXACTLY (except for tabs and spaces) the posted validation runs, by running the command "diff –iw …" as explained in the spec.	6.1	
Simulator compiles and runs on Eos linux machines: *remote.eos.ncsu.edu* and *grendel.ece.ncsu.edu*	6.2	
You can just type "make" to build your simulator (via your Makefile)	6.2	
The simulator built by the Makefile actually runs correctly (some students introduce the Makefile at the last minute and don't actually test the simulator version built by it – don't make this mistake)	TA experience	Needs To Be Checked
The compiled simulator is called "sim"	6.2	

Your simulator "sim" takes in arguments as specified in the spec:
`sim bimodal <M2> <tracefile>`
`sim gshare <M1> <N> <tracefile>`
`sim hybrid <K> <M1> <N> <M2> <tracefile>`
	6.2	

Your simulator prints output to the console (*i.e.*, screen) only	6.2	
You have been vigilant about taking certain precautions: keeping backups, not touching files after the submission deadline (to keep timestamps valid, just in case we need to revisit the original files), *etc*.	6.3	
Simulator completes a single run in less than 2 minutes	6.4	
Submit only a single zip file	8	
The name of your submitted zip file is "project2.zip"	8	
Include all the necessary files in your project2.zip submission: * Source code files * Makefile * report called "report.pdf"	8	
Your project2.zip file does NOT contain a top-level directory. Using the zip command from the spec (and adapting it for your filenames) creates a flat zip file.	8, TA experience	
You verified that the contents of your zip file is **complete** and that it has a flat structure (no top-level directory), using the command "unzip -l project2.zip" to list its contents.	TA experience	
You verified that your zip file is **complete** AND **contains the desired version** of your source code and report.pdf, by unzipping the zip file in a temporary directory, typing "make", doing some trial runs (and checking them), and examining report.pdf.	TA experience	
Report is named "report.pdf" (PDF)	8	
Report consists of the provided cover page (with your name, signed Honor Pledge, *etc*.) and required content as explained in the spec.	8, 7	
You signed the Honor Pledge, verifying that you did the project individually and did not cheat (did not give or receive unauthorized aid)	8	