Simulating Perceptron based Predictors

Vicki Anand, 12793

I. Introduction

More accurate prediction of conditional branches is one of the central problems in improving the performance of computer architectures. Some of the good modern branch predictors are based on the idea of neural networks.
In this project, I have simulated the two such branch predictors based on perceptron (Basic Perceptron [1] and Piecewise Linear Perceptron [2]).

II. Simulation Kit

The simulation infrastructure provided by **Championship Branch Prediction 2014 (CBP-4)** [3] has been used for simulating the predictor and collecting the results. Also the the results provided by them for gshare predictors has been used for comparing with the simulated perceptron based predictor.
As asked the in the championship, all the predictors were built for fixed storage budgets. Both the perceptron predictor and the piecewise linear predictor was simulated for three different storage budgets - 4KB, 8KB, and 32KB.

III. Implementation

For the predictors, the algorithm has been taken as is from the respective papers. The calculations for different parameters of the algorithms have have shown alongside the initialization of the parameter in respective program files. For the piecewise linear predictor the value of all the parameters for different budgets have been given as a table in the paper [2]. Both the papers have reported the the tuned best values for their parameter.

IV. Results

Both the perceptron predictor and the piecewise linear predictors were simulated for three different storage budgets - 4KB, 8KB, and 32KB. Following are the tables for 4 KB1, 8 KB2, 32 Kb3 budgets, showing the mispredictions per 1000 instructions for all 40 different traces given with CBP-4 kit. The arithmatic mean table 4 sums up the the results by taking an arithmatic mean over the results of all the 80 traces.

Table 1: For 4 KB storage budget

Trace name	Mispredictions per 1000 instructions(MPKI)		
	gshare	**perceptron**	**Piece. linear**
LONG-SPEC2K6-00	5.876	4.244	4.554
LONG-SPEC2K6-01	8.619	7.640	7.423
LONG-SPEC2K6-02	6.170	5.498	4.997
LONG-SPEC2K6-03	6.092	6.456	6.775
LONG-SPEC2K6-04	10.968	10.210	10.325
LONG-SPEC2K6-05	6.122	5.671	5.380
LONG-SPEC2K6-06	4.805	2.547	2.764
LONG-SPEC2K6-07	22.910	20.417	20.255
LONG-SPEC2K6-08	2.210	1.476	1.665
LONG-SPEC2K6-09	5.536	5.013	5.218
LONG-SPEC2K6-10	5.700	4.482	3.926
LONG-SPEC2K6-11	3.891	3.648	3.912
LONG-SPEC2K6-12	13.172	11.595	11.623
LONG-SPEC2K6-13	15.675	14.258	12.276
LONG-SPEC2K6-14	7.634	4.092	6.829
LONG-SPEC2K6-15	3.374	2.571	2.722
LONG-SPEC2K6-16	5.392	3.963	4.307
LONG-SPEC2K6-17	7.202	6.336	5.912
LONG-SPEC2K6-18	1.538	1.201	1.474
LONG-SPEC2K6-19	2.693	2.244	2.363
SHORT-FP-1	4.137	2.466	2.775
SHORT-FP-2	1.146	1.119	1.136
SHORT-FP-3	0.441	0.434	0.436
SHORT-FP-4	0.299	0.207	0.270
SHORT-FP-5	0.957	0.791	0.946
SHORT-INT-1	8.678	7.461	6.795
SHORT-INT-2	9.698	10.323	9.133
SHORT-INT-3	15.612	10.484	9.728
SHORT-INT-4	2.833	2.741	3.062
SHORT-INT-5	0.532	0.384	0.410
SHORT-MM-1	9.523	7.694	7.681
SHORT-MM-2	11.448	10.006	9.553
SHORT-MM-3	5.518	2.449	3.120
SHORT-MM-4	1.876	1.539	1.879
SHORT-MM-5	7.891	7.364	7.260
SHORT-SERV-1	7.874	7.780	7.353
SHORT-SERV-2	8.137	8.320	7.626
SHORT-SERV-3	8.358	9.159	9.010
SHORT-SERV-4	8.562	8.461	7.621
SHORT-SERV-5	8.595	8.363	7.150
Arithmatic Mean	**6.692**	**5.778**	**5.691**

Table 2: For 8 KB storage budget

Trace name	**Mispredictions per 1000 instructions(MPKI)**		
	gshare	**perceptron**	**Piece. linear**
LONG-SPEC2K6-00	5.354	3.767	3.910
LONG-SPEC2K6-01	8.523	7.617	7.324
LONG-SPEC2K6-02	5.731	4.883	4.462
LONG-SPEC2K6-03	5.867	6.137	6.335
LONG-SPEC2K6-04	10.895	10.115	10.125
LONG-SPEC2K6-05	5.973	5.688	5.183
LONG-SPEC2K6-06	4.544	1.630	2.755
LONG-SPEC2K6-07	19.359	17.069	15.744
LONG-SPEC2K6-08	1.932	1.317	1.639
LONG-SPEC2K6-09	5.520	4.990	5.205
LONG-SPEC2K6-10	4.861	3.488	3.411
LONG-SPEC2K6-11	3.913	2.171	3.888
LONG-SPEC2K6-12	13.023	11.592	11.756
LONG-SPEC2K6-13	13.066	11.994	10.115
LONG-SPEC2K6-14	5.886	2.866	5.795
LONG-SPEC2K6-15	2.986	1.970	2.332
LONG-SPEC2K6-16	4.927	3.761	4.153
LONG-SPEC2K6-17	6.203	6.188	4.764
LONG-SPEC2K6-18	1.525	1.122	1.462
LONG-SPEC2K6-19	2.640	2.238	2.328
SHORT-FP-1	3.843	2.236	2.763
SHORT-FP-2	1.124	1.096	1.079
SHORT-FP-3	0.442	0.432	0.435
SHORT-FP-4	0.299	0.188	0.267
SHORT-FP-5	0.805	0.791	0.793
SHORT-INT-1	8.057	6.913	5.984
SHORT-INT-2	8.670	8.938	7.448
SHORT-INT-3	14.362	9.438	9.139
SHORT-INT-4	2.569	2.961	2.819
SHORT-INT-5	0.475	0.414	0.392
SHORT-MM-1	9.299	7.608	7.498
SHORT-MM-2	11.011	9.984	9.202
SHORT-MM-3	4.471	1.917	3.011
SHORT-MM-4	1.843	1.504	1.817
SHORT-MM-5	6.959	6.334	6.054
SHORT-SERV-1	5.772	4.909	4.385
SHORT-SERV-2	5.865	5.023	4.603
SHORT-SERV-3	7.136	7.469	7.262
SHORT-SERV-4	6.817	6.200	5.706
SHORT-SERV-5	6.815	5.800	4.909
Arithmatic Mean	**5.984**	**5.019**	**4.956**

Table 3: For 32 KB storage budget

Trace name	**Mispredictions per 1000 instructions(MPKI)**		
	gshare	**perceptron**	**Piece. linear**
LONG-SPEC2K6-00	3.974	2.897	3.042
LONG-SPEC2K6-01	8.406	7.499	6.962
LONG-SPEC2K6-02	5.176	4.522	3.590
LONG-SPEC2K6-03	5.658	5.711	6.020
LONG-SPEC2K6-04	10.739	9.911	9.688
LONG-SPEC2K6-05	5.780	5.659	5.043
LONG-SPEC2K6-06	4.160	1.179	2.524
LONG-SPEC2K6-07	14.062	13.105	10.231
LONG-SPEC2K6-08	1.911	1.127	1.518
LONG-SPEC2K6-09	5.502	4.991	4.943
LONG-SPEC2K6-10	3.710	2.358	2.439
LONG-SPEC2K6-11	3.929	1.412	1.396
LONG-SPEC2K6-12	12.844	11.546	11.526
LONG-SPEC2K6-13	9.880	9.455	7.103
LONG-SPEC2K6-14	4.687	1.431	3.555
LONG-SPEC2K6-15	2.648	1.605	1.805
LONG-SPEC2K6-16	4.435	3.534	3.558
LONG-SPEC2K6-17	5.464	5.996	4.222
LONG-SPEC2K6-18	1.525	0.927	1.461
LONG-SPEC2K6-19	2.599	1.364	2.206
SHORT-FP-1	3.479	1.723	2.366
SHORT-FP-2	1.061	1.094	1.015
SHORT-FP-3	0.443	0.435	0.432
SHORT-FP-4	0.258	0.186	0.214
SHORT-FP-5	0.788	0.640	0.789
SHORT-INT-1	7.347	4.535	4.568
SHORT-INT-2	7.669	7.841	5.581
SHORT-INT-3	11.784	8.926	7.574
SHORT-INT-4	2.250	2.637	1.780
SHORT-INT-5	0.438	0.365	0.355
SHORT-MM-1	9.172	7.520	6.985
SHORT-MM-2	10.601	9.864	8.755
SHORT-MM-3	4.267	0.988	2.448
SHORT-MM-4	1.811	1.489	1.479
SHORT-MM-5	5.651	5.263	4.446
SHORT-SERV-1	3.646	2.264	1.997
SHORT-SERV-2	3.665	2.273	1.948
SHORT-SERV-3	5.870	5.566	4.846
SHORT-SERV-4	5.324	3.962	3.529
SHORT-SERV-5	5.208	3.297	3.057
Arithmatic Mean	**5.196**	**4.177**	**3.925**

Table 4: Arithmatic means

4 KB			**8 KB**			**32 KB**		
gshare	**perceptron**	**piece. lin.**	**gshare**	**perceptron**	**piece. lin.**	**gshare**	**perceptron**	**piece. lin.**
6.692	**5.778**	**5.691**	**5.984**	**5.019**	**4.956**	**5.196**	**4.177**	**3.925**

V. Conclusions

The overall summed up results as given in the arithmatic mean table 4 is as expected.
For all the budget categories, both the perceptron based predictors have performed significantly better than the gshare predictor.
However for 4 KB and 8 KB budget the improvement in misprediction rate for the piecewise linear predictor over the simple perceptron predictor is 1.5% and 1.2% respectively. Which is much lesser as compared to this difference for 32 KB budget, which is 6.0%. This can be explained as below :
In piecewise linear algorithm, for 4 KB and 8 KB budget, the value of parameter n was set to 1 and 2 respectively, as per the tuned result table given in paper. This means that all the branch PC would map to same entry for $n = 1$. As given in paper, it can be noticed that for $n = 1$, the algorithm reduces to a path-based neural predictor. Also there is a figure (Figure 7) in paper which shows the misprediction values by varying the values of parameters m and n, while keeping the value $m * n$ constant. At both the ends ($n = 1$ and $m = 1$), performance is comparable.
And for $m = 1$, this algorithm reduces to the basic perceptron algorithm. So, for 4 KB and 8 KB budget, the piecewise linear algorithm cannot not perform so much better as 32 KB case, when the value of n is 8, due to budget constraints.

Furthmore, considering the individual trace results, it can clearly noticed that piece wise linear algorithm very consistently beats the gshare algorithm, which is as expected. But there are few traces(like SHORT-INT-2, SHORT-SERV-3, ..) in all three budget categories, for which gshare performs better than the simple perceptron predictor. This can be attributed to the inability of simple perceptron based model to learn linearly inseparable functions. This inability has been discussed in the paper [1].

References

[1] Daniel A. Jiménez, Calvin Lin. **Dynamic Branch Prediction with Perceptrons**. High-Performance Computer Architecture, 2001. HPCA. `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=903263&tag=1`

[2] Daniel A. Jiménez. **Piecewise linear branch prediction**. Computer Architecture, 2005. ISCA '05. `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1431572`

[3] **Championship Branch Prediction (CBP-4)** `http://www.jilp.org/cbp2014/`