Project 2 Report

The tournament branch predictor implemented uses both a Yeh-Patt and Perceptron predictor. It is dynamically trained to pick the predictor that is performing the best at the given time and program counter. The predictors have some tuning parameters that will scale the following sizes:

- P : log2 of the number of entries in the Yeh-Patt Pattern Table
- L : log2 of the number of entries in the Yeh-Patt History Table
- N : log2 of the number of perceptrons in the Perceptron Table
- G : the width of the global history register used by the perceptrons

and C chooses the initialization of the counters in the Tournament Counter Table according to the following

- C : 0 – 0b0000
- C : 1 – 0b0111
- C : 2 – 0b1000
- C : 3 – 0b1111

In order to choose the optimal sizes, all of the valid configurations can be simulated to pick the best branch prediction accuracy. All valid configurations can first be found using the following method. For each parameter, set all other parameters to the minimum value, then find the maximum value for that parameter while staying in the size budget. This gives a range of all possible parameter values to check. The python script, experiment.py, loops through these parameter combinations, checks whether it is under the size budget of 26 KiB, and then runs the experiment for that configuration. With all of the prediction accuracy results, some simple data analysis can compile the results into meaningful data.

The accuracy of each of the predictor configurations can be averaged over all 4 of the provided traces. Then the maximum value accuracy configuration is most likely to be the best performing predictor. The best predictor had an average accuracy of 0.97126985 with the following configuration.

P – 11, L - 12, N - 9, G - 35, C – 1

To graph the data meaningfully, all of the parameters can be set to this optimum and then the performance of the predictors can be shown while varying a single parameter in the possible range.

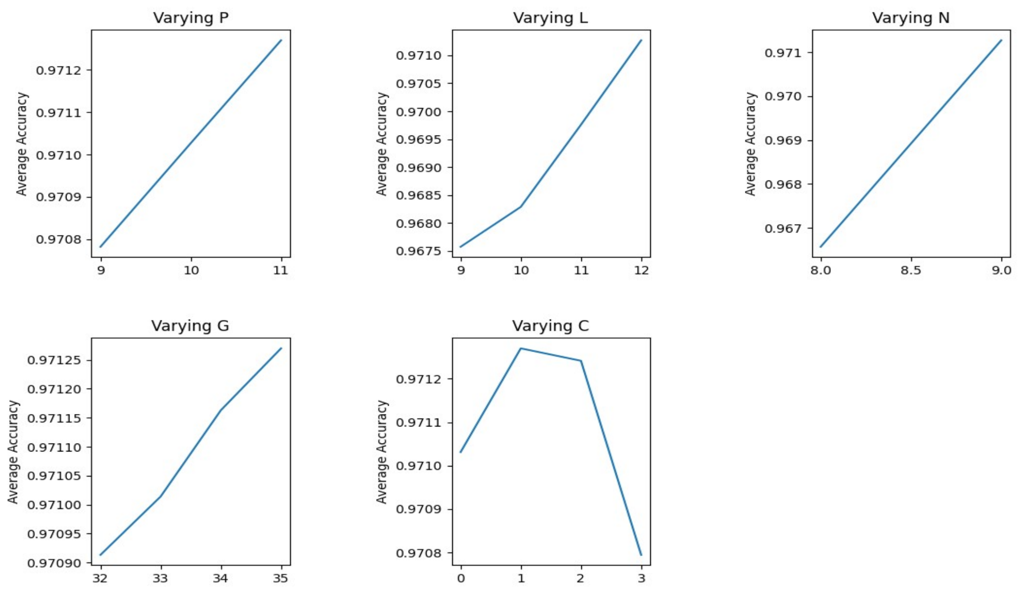

Figure 1. Performance of the predictor while varying each parameter as the others are set to the determined optimum.

Although this method only shows a local subspace of the range of possible configurations, it gives a good insight that for this weight of allocation of resources to the Yeh-Patt and Perceptron combination, more resources will always perform better. Something that is less obvious with this visualization, is that the size of the Perceptron Table only goes up to G = 35 while it's maximum possible value is 64. This means that the experiments proved that it's better to allocate more resources to the Yeh-Patt predictor than the Perceptron predictor. Increasing the Perceptron predictor size returns only marginal benefits when compared to the Yeh-Patt predictor. C is, expectedly, worse when initialized to strongly the Yeh-Patt predictor or strongly the Perceptron predictor, C = 0 and C = 3, respectively.