Analysis and Conclusions

Methodology

We used the ChampSim CPU simulator (in C++) with the SPEC 2006 and IPC1 benchmarks.

In addition to comparing the branch predictors against each other, we tested each branch predictor against two independent variables: number of instructions (duration) and number of table entries (memory budget).

For our main tests, we used four different durations (500K, 1M, 5M, and 10M) on the SPEC 2006 benchmarks. Here "duration" means number of warmup instructions and simulation instructions, so a test with 1M instructions consists of 1M warmup instructions followed by 1M simulation instructions. To get a stronger contrast, we also tested each branch predictor with 10K, 50K, 100K, and 500K instructions on the IPC1 benchmarks. All the aforementioned tests were run with a fixed table size (2^{14} entries).

The memory budget tests were separate. To run them in a reasonable amount of time, we used fewer (500K) instructions on the IPC1 benchmarks. We tested table sizes of 2^{12}, 2^{13}, 2^{14}, and 2^{15}.

Our data and graphs can be found in the document "Branch Predictor Data":
(https://docs.google.com/spreadsheets/d/1wgrtyOM7MFFpiv0K3tMB.FozJDVN7IH2XOj_hdcTBrV0/edit?usp=sharing).

Comparison of Branch Predictors

We mainly focus on the results from when 10M instructions were run, as they are most applicable to the real world. Overall, the TAGE variants (TAGE, TAGE-L, TAGE-SC-L) are the best performing among all of the branch predictors. As for our novel predictors, fourier performs the worst by far, but LSTM performs better than average, comparable to BiMode and gshare. Both bimodal and localHistory also did not perform as well as the other branch predictors, but were still better than fourier. All the other branch predictors (BiMode, gselect, gshare, perceptron) were near the average.

Notable Benchmarks

While the exact contents of the benchmarks are unknown, we have a short description of each test: (https://www.spec.org/cpu2006/publications/CPU2006benchmarks.pdf).

- **astar:** The astar benchmarks really show the power of TAGE; all 3 variants perform much better than any of the other branch predictors on these benchmarks. The geometric history length helps to predict both remote and local branches very well. As seen in the description, the astar benchmarks test the A* algorithm, which searches for the optimal path through a complex space and thus contains many hard-to-predict branches.
- **bzip2, mcf:** These test data compression and a graph algorithm respectively; they are also empirically difficult for branch predictors. The TAGE predictors do less well on these, while the LSTM and especially the perceptron perform comparably better.
- **milc:** Most branch predictors obtain 100% accuracy, which is not too surprising: scientific computing routines like this one tend to have regular control flow. However, the Fourier predictor does much worse, indicating that many branches are in fact conditional; nice fractions among the prediction accuracies suggest a periodic branch pattern.

Memory Budget

Varying the memory budget has a substantial impact on branch prediction. Unsurprisingly, the local history predictor is most affected, since it relies on predicting each branch separately. An anomaly is that changes to the memory budget seem to have negligible or even reverse effects on the TAGE predictors (which still outperform the others at all budgets); this may be a coincidence due to the limited simulation duration.

Number of Instructions

The general pattern is that the more complex the branch predictor, the more instructions it takes to reach its optimal performance. Simpler predictors like BiMode and gselect perform much better than TAGE variants for very few (10K) instructions, but TAGE-based predictors quickly surpass them. More strikingly, the SC and L components of TAGE-SC-L actually seem to worsen it for <10M instructions, but they provide a marginal improvement at 10M; perhaps even longer simulations would show a clearer improvement for SC and L.

Novel Predictors

Fourier

This predictor performed much worse than the others. A partial explanation is that Fourier analysis is intended not for general-purpose use but as part of a larger branch predictor. However, we discovered through white-box testing that our dynamic approach failed to accurately represent even simple periodic patterns.

Examining the amplitudes at each frequency revealed that most of them were near zero (the waves were orthogonal to the branch pattern). As it turns out, this holds mathematically for any frequency that is not an integer multiple of the branch pattern frequency. This demonstrates a major limitation of Fourier branch prediction, especially in a dynamic context: the frequencies must belong to a very narrow set to provide any useful information. Thus, accurate prediction is impractical without a way to identify and select the frequencies in advance.

LSTM

Generally, this predictor performed better than bimodal but worse than the TAGE-based predictors. This makes sense for bimodal because LSTMs can “hold” longer history than bimodal, even with similar space constraints due to the persistence of long-term memory. However, TAGE-SC-L still performs better than LSTMs in many benchmarks. The most plausible reason for this is due to the rate at which LSTMs learn. For example, it takes 200 loops to even learn a basic for loop, whereas the loop predictor in the TAGE-SC-L would be able to pick up on this pattern much quicker.

In terms of weights, we noticed that the long-term memory weight that controls how much memory was utilized typically either rapidly changed over time or remained low in cases of random input, but strengthened or increased when predicting loops. This was done through custom tests in a whitebox, as the SPEC benchmark does not allow us to actually retrieve the code that it runs through traces.