Part 2

Due: April 26, 2018

Architectural simulation is widely used in computer architecture studies because it allows us to estimate the performance impact of new designs. In this part of the project, you are asked to add a new branch predictor implementation to SimpleScalar. It has implemented 2-bit branch history table (called “bimod” in the simulator), correlating predictor (called “2lev”) and several others. The 2-bit branch history table uses the 2-bit saturating counter that we have discussed in the class.

In this project, you are asked to implement another 2-bit branch history table that uses the 2-bit counter as shown in the textbook, Figure C.18. You need to make changes to bpred.h and bpred.c of SimpleScalar source code to implement the new 2-bit counter and also a small change to sim-outorder.c to add a new choice of branch predictor types.

Next, you can run the simulation and report the observed results (including the screenshots of timestamps and statistics and the changes to bpred.c in your report). What is the impact on both the overall performance and branch prediction accuracy when using the two different implementations of 2-bit counters? Compared with the perfect branch predictor that is implemented by SimpleScalar, how much is the performance loss due to branch mis-predictions for these two 2-bit branch predictors?

Similar to the experiments in Project Part 1, please use the option “-fastfwd 500000000” to fast-forward the first 500 million instructions and collect statistics on the next 200 million instructions (using the option “-max:inst 200000000) for the program equake (equake.ss < equake.in).