Shallow Online Neural Branch Prediction

Pulkit Gupta, Jack Lafiandra, Tom Conte
School of Computer Science, Georgia Tech

The State of Branch Prediction

Branch mispredictions remain a substantial limiting factor in improving the performance and energy efficiency of modern processors. Recent predictors attempt to resolve “Hard-to-Predict” branches by augmenting the state of the art TAgged GEometric (TAGE) predictor with correction mechanisms or convolutional neural mechanisms.
The convolutional mechanism introduced by BranchNet is very costly to implement and requires offline training, disqualifying it from inclusion in new processors. Instead, the Shallow Online Neural (SON) branch predictor uses existing state information historically used for branch predictors (Branch Address, Local History, and Global History) to train a small neural network “online” and attempts to predict with higher accuracy than TAGE.

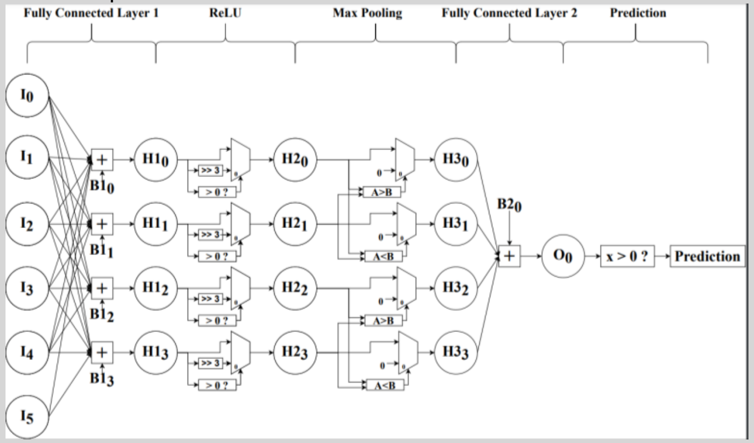
Brief History of Neural Predictors

Neural networks are fundamentally capable of learning almost any relationship given enough layers and computational complexity. Because of this, they have been explored from time to time as advancements in density and manufacturing nodes have enabled more and more complex designs. The Hashed Perceptron predictor was proposed in 2001 and uses a table of perceptrons to extract branch pattern and path information. In 2014, an augment to TAGE used a small perceptron table to generate confidence and statistical corrections for the predictions made by the base predictor.

Now, as node sizes enable higher density chips, we explore a predictor that uses more complex neural structures with hidden layers and activation functions as the main prediction mechanism and attempts to use a small tournament mechanism to decide between the neural predictor and another predictor.

Design Summary of the SON predictor

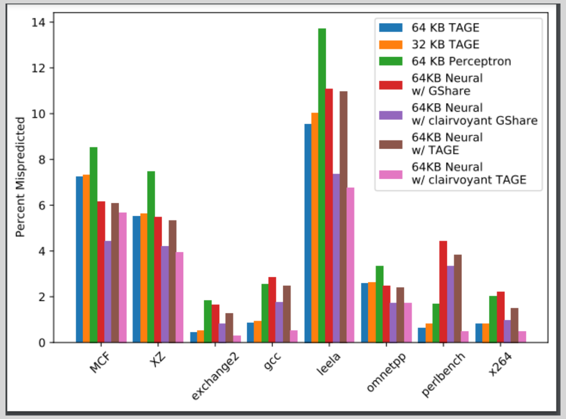
The SON predictor takes, as its inputs the Global History and Local History, transforming them into “bipolar-binary” values. A table of network weights is indexed using the PC to distribute the “learning load” on each network. After the first fully connected layer, we apply an activation layer in the form of LeakyReLU which keeps positive values and scales negative values by a factor of 0.125. After which we perform pair-wise max pooling and another fully connected layer. The sign bit of the resultant value is used as the prediction value.
As part of the training mechanism we first apply another activation function in the form of sigmoid. The training algorithm uses a Binary Cross Entropy loss function and a Learning Rate of 1/64. We then use a small tournament comprising of 2-bit smith counters to decide between the predicted value and the output of a small GShare or TAGE predictor



Physical Realizability of Neural Predictors

The default SON predictor uses 32-bit floating point weights, making it impossible to implement. As such we quantize the weights using a Q6.18 fixed point representation, allowing for utilization of integer arithmetic hardware to perform computation. Additionally, with 64 input nodes and 8 hidden layer nodes, the total area of the computation component of the predictor is equivalent in area to a 16KB SRAM block.

Performance Characteristics



Comparison is done against Gshare and TAGE, and with a clairvoyant and a 2KB tournament.
The 64 KB SON predictor performs admirably for some of the SPEC benchmarks. We see that the clairvoyant tournaments show the theoretical bounds of the SON predictor.

Future Work

As seen in the performance characteristics, Improving the accuracy of the tournament mechanism is key to improving the accuracy of the predictor.

Future work will investigate stronger tournament mechanisms and include substantial trace analysis to determine the important factors and available data/state that should be used by the tournament mechanism.