A Survey of Machine Learning Branch Prediction

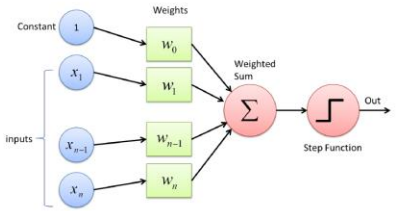
Armin Hedzic (ahedzic)

Harsha Puttapaka Leela (hputtapa)

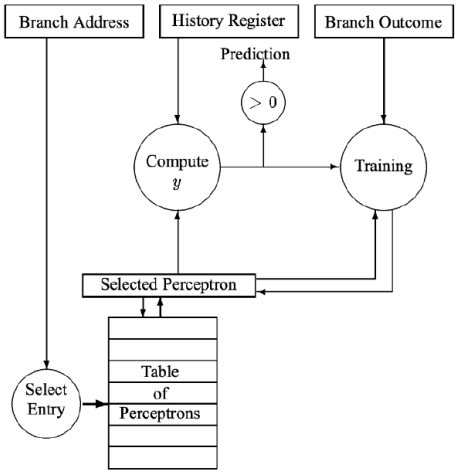
Sreeharsha Raveendra (sraveend)

Concept

- Control hazards due to branches are a major performance consideration in pipelined CPUs.
- Branch prediction is a speculative way to overcome control hazards but has the price of flushing the pipeline on mispredictions.
- With recent advancements in machine learning, we wanted to try to see the effect of a simple perceptron neural network (figures on the right) and reinforcement learning (QTable) on branch prediction accuracy.
- We chose these simpler machine learning models with the consideration of the cost of implementing more complex models in the hardware.
- We understand that a significant accuracy increase would be needed to warrant the cost of such an implementation in hardware.



A perceptron [1]



Using perceptrons in a branch predictor [1]

Project Description

- We are comparing the accuracy of 6 traditional branch prediction algorithms:
  - Static not-taken
  - Static taken
  - GShare
  - GSelect
  - Tournament
- With the accuracy of machine learning based prediction algorithms:
  - Perceptron
  - Reinforcement learning QTable
- The comparison is done by scaling global bit history over 4 workloads

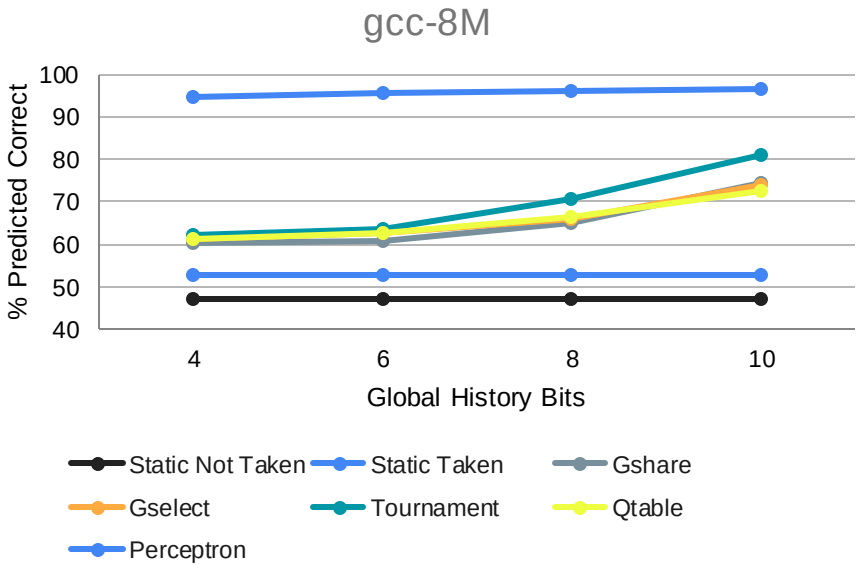
- We are utilizing 4 workload traces:
  - A mobile processor[2] (~29 million)
  - A webserver[2] (~149 million instr.)
  - 2 collected from runs of gcc (~10k and ~8.5 million instr.)
- Expected result: Increase in accuracy for perceptron model. Roughly same accuracy in QTable model.

Project Description

- Perceptron
  - A perceptron is a single layer neural network that consists of a processor which takes multiple inputs with weights and biases to generate single output.
  - It works by multiplying inputs with weights and then the weighted sum of all multiplied values are taken and applied to an activation function
  - It uses only the global branch history as input, but its interpretation of global branch history is continuously changing as it is "online" learning.
- QTable
  - Concatenates instruction address with global history just like GSelect
  - Difference is instead of saturating counters it keeps track of a probability of choosing taken and not taken.
  - Probability is affected by corrected of previous predictions
  - Uses exploration (epsilon) to occasionally make a random prediction.

Results

- The accuracy of perceptron-based algorithm is significantly higher than the traditional algorithms.
- While the higher perceptron accuracy was expected, the magnitude of increase was surprising.
- The QTable predictor does on par with GShare, GSelect and Tournament predictors.
- The QTable result was as expected due to it simply determining a statistic value for each address instead of a saturating counter.
- These results are similar across all workloads and can be found in the results.xslx file.



Conclusion

- We tested machine learning based branch predictors, specifically a perceptron and reinforcement learning QTable.
- Perceptron's are attractive because they can use long history lengths without requiring exponential resources.
- Despite the fact that they increase computational complexity, they can be implemented efficiently in terms of both area and delay.
- The perceptron model significantly outperforms all other models in the four datasets we tested, which is the desired outcome.
- The QTable performed on par with traditional approaches, but carries the weight of more complexity.
- Testing more complex machine learning models may yield higher accuracy, but given accuracy in the mid 90%s with a simple perceptron, higher complexity models may not be worth the trade-off.

References

- [1] A Survey of Deep Learning Techniques for Dynamic Branch Prediction – Rinu Joseph
- [2] Server and Mobile workloads - https://jilp.org/cbp2016/