CS752 Project Proposal: Load Value Prediction

Prajyot Gupta, Satvik Maurya, Robert Viramontes

Topic Description

In our project, we intend to explore the concept of value locality and load value prediction, as proposed in [1]. We will characterize the performance of a load value prediction scheme on modern workloads and enhance the prediction scheme by using a combination of predictors (possibly including a perceptron-based predictor).

Phase 1

In Phase 1, we will be running SPEC CPU 2006 Benchmark, based on around 6-7 checkpoints, to get an estimate of simulation runtimes and computational power required for project implementation. We will also be reviewing the O3 CPU code to understand where our load value prediction objects will need to wire in to modify the CPU behavior.

Phase 2

In Phase 2, we will be re-implementing the load value prediction design proposed in [1] and comparing the results on modern SPECCPU workloads. We will implement the primary functional units of the design within the existing gem5 CPU model, including the load value prediction table (LVPT) and the Load classification table (LCT), Constant Verification Unit (CVU).

- LVPT – Takes in the instruction and predicts the data. Contains history of last 'n' unique values from given instruction.
- LCT – This is a 2-bit saturation confidence counter which introduces hysteresis to the prediction.
- CVU – This unit verifies the speculated data with the actual data fetched from memory. It also generates the control signals for flushing instructions from the pipeline in case of a misprediction.

We will also modify the existing pipeline structures as necessary to interface with these new functional units. In particular, the LVPT and LCT would be in the frontend of the pipeline and the CVU will be implemented in the backend (before an instruction retires). The implementation of the CVU will need a recovery mechanism to be implemented as well (in case of a misprediction). We will conclude Phase 2 by evaluating the design against a baseline CPU model without load value prediction to determine the impact of this modification.

Phase 3

In Phase 3, we plan to investigate improvements to the load value prediction algorithm and compare results of our evaluation metrics with the implementation from [1]. Our modifications to the value prediction algorithm will be based on our observations from Phase 1, but we are currently considering modifications informed by a perceptron-based branch predictor.

In short, our contributions will be 1) *evaluating the load value predictor design of* [1] *on modern workloads* and 2) *modifying the prediction algorithm using approaches similar to* [2] & [3] *to improve the efficacy of load value prediction in modern workloads*.

Proposed Timeline

Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8
Proposal			Progress Report		Status email	Status email	Presentation
Phase 1	Phase 1						Buffer
	Phase 2	Phase 2	Phase 2	Phase 2			Buffer
				Phase 3	Phase 3	Phase 3	Buffer

Motivation

As we have seen in lecture, in recent decades compute performance has increased at a faster rate than memory performance. This trend is commonly referred to as the memory wall. We are interested in investigating techniques that help hide the gap between compute and memory performance. Of the techniques we have studied and found in the literature, we are particularly interested in exploring the concept of value locality, as discussed in [1] and [4]. With value locality, the goal is to alleviate dataflow limits that result from true dependencies and expose more opportunity for instruction-level parallelism in serial programs. [1] was published in 1996 based on models of the PowerPC 620 and Alpha AXP 21164. We believe there is value in studying how this technique performs today in a recent gem5 model of an x86 processor with more modern workloads. When the speculative values are correctly predicted, we can hide the memory latency that would typically be incurred by true data dependencies.

Evaluation Methods

Tool – Gem5 simulator with x86 O3 CPU. The simulator will be run on the CS Department Linux machines and, if appropriate, batch jobs for benchmark runs may be submitted via Condor.

Workloads – We initially plan to use the SPEC CPU 2006 workload to benchmark the performance as we have access to gem5 checkpoints that will simplify initial simulation efforts. Time permitting, we would like to extend our workloads to include recent benchmarks like SPEC CPU 2017.

Reference Implementation – We will compare our results with those reported by Lipasti *et al* [1].

Metrics – Our primary metric will be the speedup of the benchmarks. We are also be interested in collecting metrics that profile the prediction performance, such as percent % of LVPT predictions correct.

References

[1] M. H. Lipasti, C. B. Wilkerson and J. P. Shen, "Value locality and load value prediction," *ACM SIGPLAN Notices,* vol. 31, p. 138–147, 9 1996.

[2] A. Perais and A. Seznec, "Practical data value speculation for future high-end processors," in *2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA)*, Orlando, 2014.

[3] R. Sheikh and D. Hower, "Efficient Load Value Prediction Using Multiple Predictors and Filters," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Washington, 2019.

[4] M. H. Lipasti and J. P. Shen, "Exceeding the dataflow limit via value prediction," in *Proceedings of the 29th Annual IEEE/ACM International Symposium on Microarchitecture. MICRO 29*, Paris, 1996.