Memory Dependence Prediction

Rhushikesh Prabhune
rprabhu@ncsu.edu

Ajinkya Tundurwar
aatundur@ncsu.edu

March 20, 2020

1 Key Outcomes:

- Implement store sets paper [1] to improve the performance of memory prediction.
- Add 2-bit counters to further try to improve the performance of memory prediction.
- Analyze variation of the memory dependence performance with different times and conditions of decrementing and/or resetting the counters.
- Compare performance with no speculation, naive speculation, blind speculation, sticky bits, store sets and oracle memory disambiguation.

2 Major Tasks, and Assignment of Tasks to Students:

- Design an algorithm to generate Store Set Identifiers (SSID) and map the PC of loads/stores into the Store Set Identity Table (SSIT). The SSID's present in SSIT will be used to map into the Last Fetched Store Table (LFST) to get memory dependencies.
- Design an algorithm to add a 2-bit counter to every entry in the SSIT, increment the counter for a correct memory dependence prediction, decrement the counter for an incorrect memory dependence prediction.
- Find alternate conditions for decrementing and/or resetting the counters so as to boost the memory dependence performance.
- **Distribution of Tasks**:
  - Ajinkya Tundurwar: Design and implement the algorithm to generate SSID and mapping the entries in SSIT (SSID) to index into the LFST.
  - Rhushikesh Prabhune: Design and implement the algorithm for inclusion of 2-bit counters in the store sets predictor and control logic for incrementing, decrementing/resetting the counters.
  - Both the students will work on finding alternate conditions to decrement and/or reset the counters so as to boost the memory dependence performance.

3 Experimental Plan

- Analysis of simulator performance based on different benchmarks to get a peek into the distribution of memory access instructions and memory dependencies in the benchmarks.
- Sticky Bit Memory Dependence Predictor:
  - Perform sensitivity analysis on IPC due to different sizes and cyclic clear interval of the table.
  - Compare the difference in performance with respect to oracle memory disambiguation due to factors such as aliasing, false dependency etc.
- Store Sets Memory Dependence Predictor:
  - Compare with oracle memory disambiguation to study difference in performance.
  - Compare performance of store sets memory dependence predictor with and without store set merging.
  - Sensitivity analysis on IPC due to SSIT table sizing, LFST table sizing and the cyclic clear interval.
  - Compare the performance with and without augmenting 2-bits counters to the SSIT.

4 Contingency Plan

- In the event of inadequate time to develop the store set predictor and incorporate the 2-bit counter in the predictor, we would like to implement the store vector memory dependency predictor [2] as an alternative to store sets.
- Alternate Tasks: Create a store vector class containing the store vector table (SVT), load scheduling matrix (LSM) and implementation of the algorithm in [2].
- Alternate Experimental Plan
  - Performance comparison with oracle memory disambiguation.
  - Sensitivity analysis of IPC on store vector length.
  - Performance comparison with other memory dependence predictors based on overall performance (IPC).

References

[1] G. Z. Chrysos and J. S. Emer, ”Memory dependence prediction using store sets,” Proceedings. 25th Annual International Symposium on Computer Architecture (Cat. No.98CB36235), Barcelona, Spain, 1998, pp. 142-153.

[2] Samantika Subramaniam and G. H. Loh, ”Store vectors for scalable memory dependence prediction and scheduling,” The Twelfth International Symposium on High-Performance Computer Architecture, 2006., Austin, TX, 2006, pp. 65-76.

[3] A. Moshovos, ”Memory Dependence Prediction,” Ph.D dissertation, Computer Sciences Department, University of Wisconsin - Madison, USA, 1998.