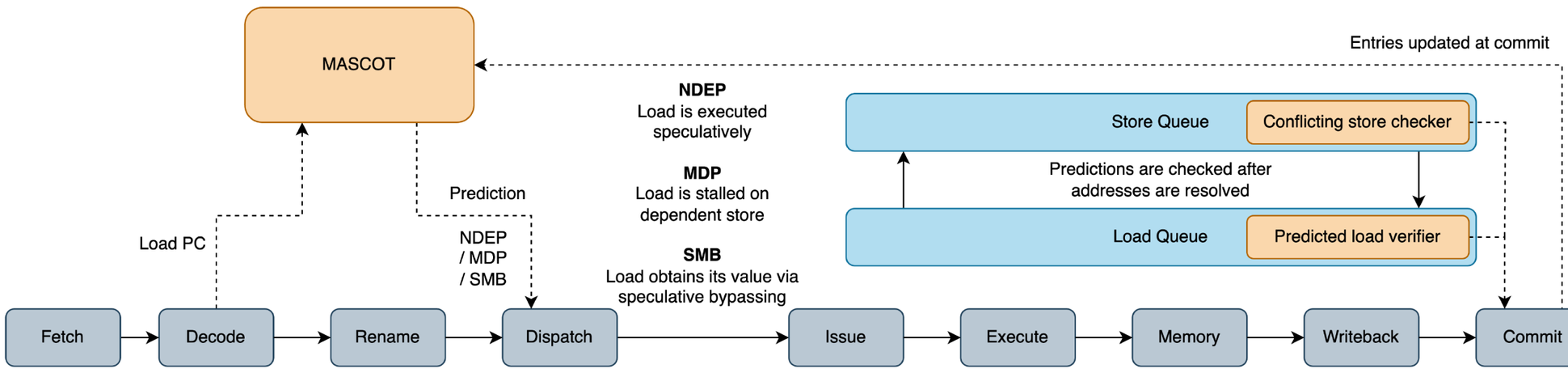
MASCOT: Predicting memory dependencies and opportunities for speculative memory bypassing

UNIVERSITY OF CAMBRIDGE
Department of Computer Science and Technology

Karl H. Mose [1] Sebastian S. Kim [2] Alberto Ros [2]
Timothy Jones [1] Robert Mullins [1]

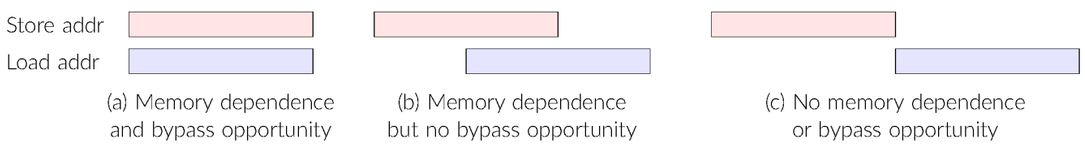
[1]University of Cambridge [2]University of Murcia

UNIVERSIDAD DE MURCIA



Memory Dependence Prediction vs Speculative Memory Bypassing

Increasing Instruction Level Parallelism (ILP) is crucial for modern out-of-order processors, but memory dependencies present both challenges and opportunities. Here we explore two prediction strategies to exploit them.



- **Memory-Dependence Prediction (MDP)**
  - Increases ILP by allowing load instructions to be issued even when addresses in the store queue are unknown. MDPs predict if a load will alias with a specific prior store, and if so delay the issue of the load.
- **Speculative Memory Bypassing (SMB)**
  - Increases ILP by short-circuiting a predicted load-store dependence to forward the value written by a store, without necessarily knowing the value of either address.
- **Why not both?**
  - Both MDP and SMB require predictors with high accuracy, but the benefits are not equal. For MDP, false negatives are more costly to performance than false positives, since a missed dependence can lead to a load being issued too early. For SMB, false positives can cause loads to use incorrect values, requiring squashing. Because of this, prior works have focused on doing just one of the two.

TAGE-based predictors

TAGE (TAgged-GEometric history length predictor), while originally a branch predictor, has been used at the base for everything from indirect-target predictors to value prediction. But how does it work?

- **Increasing lengths of global history** TAGE stores patterns of global history which it uses for prediction. It does this via an array of tables that are indexed by hashing the program PC with geometrically increasing lengths of global history.
- **Predicting** TAGE accesses all tables in parallel, and picks the biggest match it can find.
- **Learning** Upon misprediction in table N, TAGE allocates a new entry in table N+1, which is indexed with a longer global history, thereby allowing it to learn more complex patterns. Easy-to-predict instructions are stored in the tables with shorter lengths of global history, while harder ones use multiple entries in the bigger-history tables.

TAGE has also been adapted for load-store dependency prediction (MDP and SMB). In this case, it's accessed via load instruction PCs, with entries encoding load-store dependencies and using store queue offsets instead of taken/not-taken counters.

The challenge in learning load-store dependencies

TAGE-based predictors have been less successful in MDP/SMB than in other areas. Unlike branch prediction, where a TAGE-based approach can always allocate a new entry after a misprediction, memory-dependence predictors typically store only dependencies. Because of this, a false dependency prediction creates a dilemma.

- If dependent on a different store, a new entry can be allocated as usual.
- If not dependent on any store, the choice is less obvious, as there is no new dependency to learn. Previous TAGE-based memory-dependence and SMB predictors so far have opted to deal with this by decrementing the usefulness of the predicting entry.

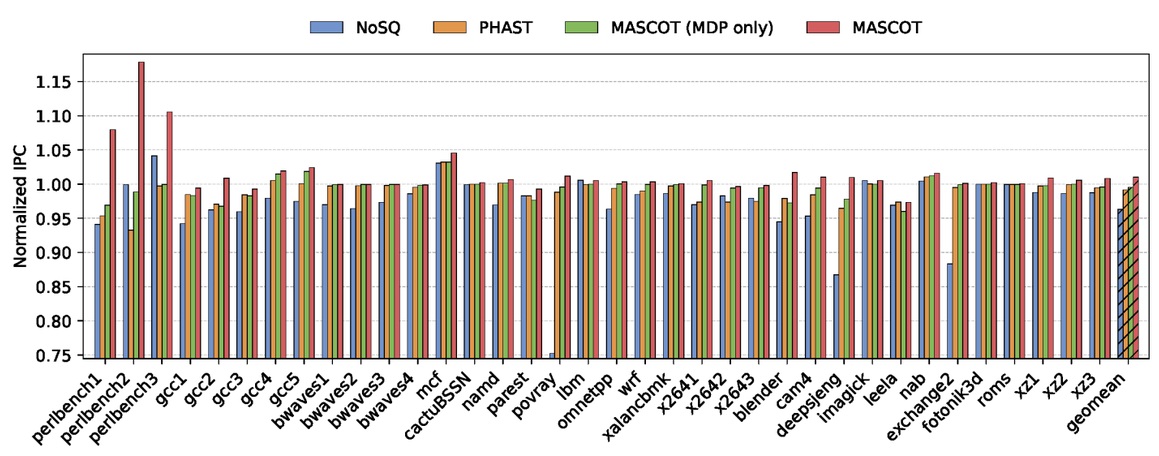
This approach risks multiple mispredictions without learning. Instead, we propose to allocate a new entry encoding a non-dependence.

MASCOT

MASCOT is a TAGE-like predictor that opts to deal with the above problem by tracking both load-store dependencies and non-dependencies. In MASCOT, entries can encode dependencies as well as non-dependencies. When it incorrectly predicts a dependence, it will allocate a non-dependent entry in the next-history table. As a result, MASCOT has high sensitivity to both false-positives and false-negatives, making it uniquely suited for doing both MDP as well as SMB.

Results

When used for MDP only, MASCOT achieves on average a 0.4% performance increase over the previous state-of-the-art in PHAST. When enabling SMB, it increases its lead to 1.9%, with some benchmarks seeing improvements as high as 26%. It reduces false negatives by 39% and false positives by 91% compared to PHAST.



Furthermore, comparing MASCOT to a similar TAGE-based design that does both MDP and SMB but does not track non-dependencies, we found that MASCOT decreases false dependencies by 92%, and increases IPC on average by 1.2%.

Contact: km781@cam.ac.uk