Presented by:
Xtra Computing Group@NTU/NUS
http://pdcc.ntu.edu.sg/xtra/
Shuang Chen*, Shunning Jiang* (SJTU, NTU),
Bingsheng He (NUS), Xueyan Tang (NTU)
* Currently at Cornell University

A Study of Sorting Algorithms on Approximate Memory

Introduction

Recently, approximate storage emerges in the area of computer architecture. It trades off precision for better performance and/or energy. Previous studies have demonstrated the benefits for applications that are tolerant to imprecision such as image processing. However, it is still an open question whether and how approximate storage can be used for applications that do not expose such intrinsic tolerance.

In this paper, we study one of the most basic operations in database, sorting, on a hybrid storage system with both precise storage and approximate storage.

Contributions:

1. The first to leverage precise computing on approximate storage.
2. A novel approx-refine mechanism to guarantee precise sorting on hybrid-memory machines.
3. Experimental results show that approximate storage can improve performance of sorting by up to 11%.

Background

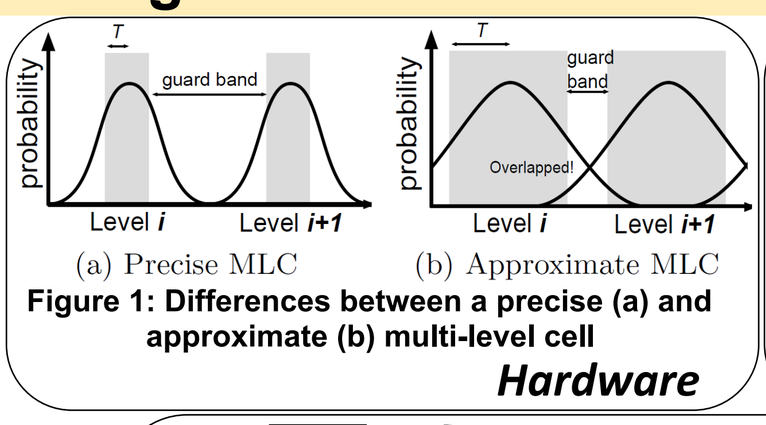

Figure 1: Differences between a precise (a) and approximate (b) multi-level cell

Hardware

We study:
- Mergesort
- Quicksort
- LSD Radixsort
- MSD Radixsort

Algorithm

Architecture

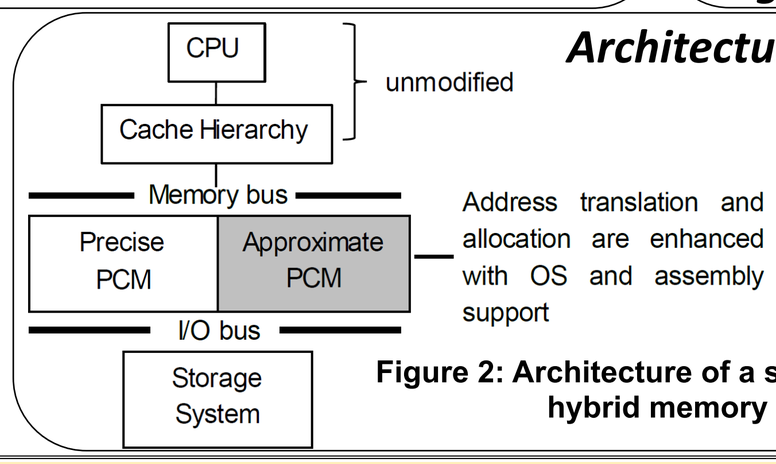

Figure 2: Architecture of a system with hybrid memory

Sorting on Approximate Memory

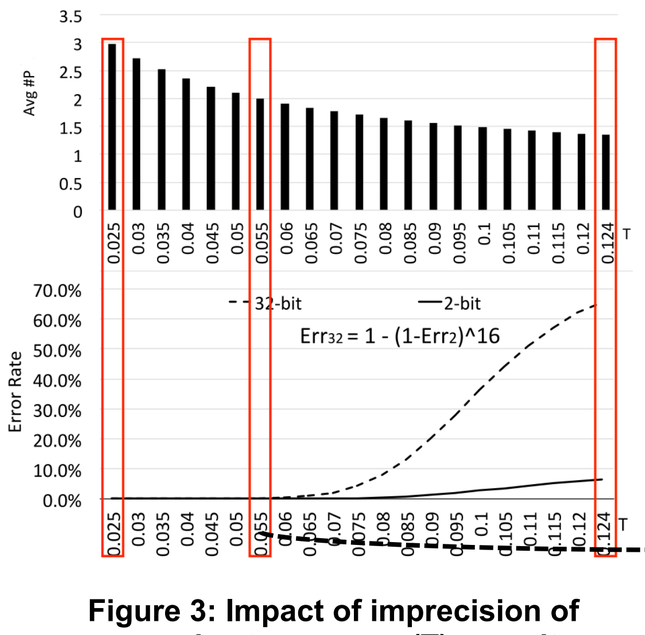

Figure 3: Impact of imprecision of approximate memory (T) on write performance and accuracy for a 2-bit MLC cell (from Monte-Carlo simulations).

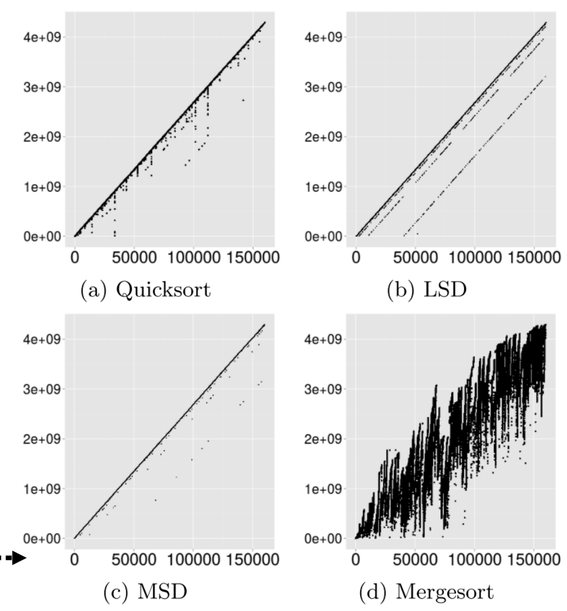

Figure 4: An 160K-integer sequence after sorting in a 1.5X faster approximate memory

- **Characterize unsortedness:** Rem(X) = n − max{ k | X has an ascending subsequence of length k }
- **Takeaways:**
  - Different sorting algorithms have very different behaviors on approximate memory.
  - When memory is not over-approximate, some algorithms can achieve marginal unsortedness.

Approx-Refine Mechanism

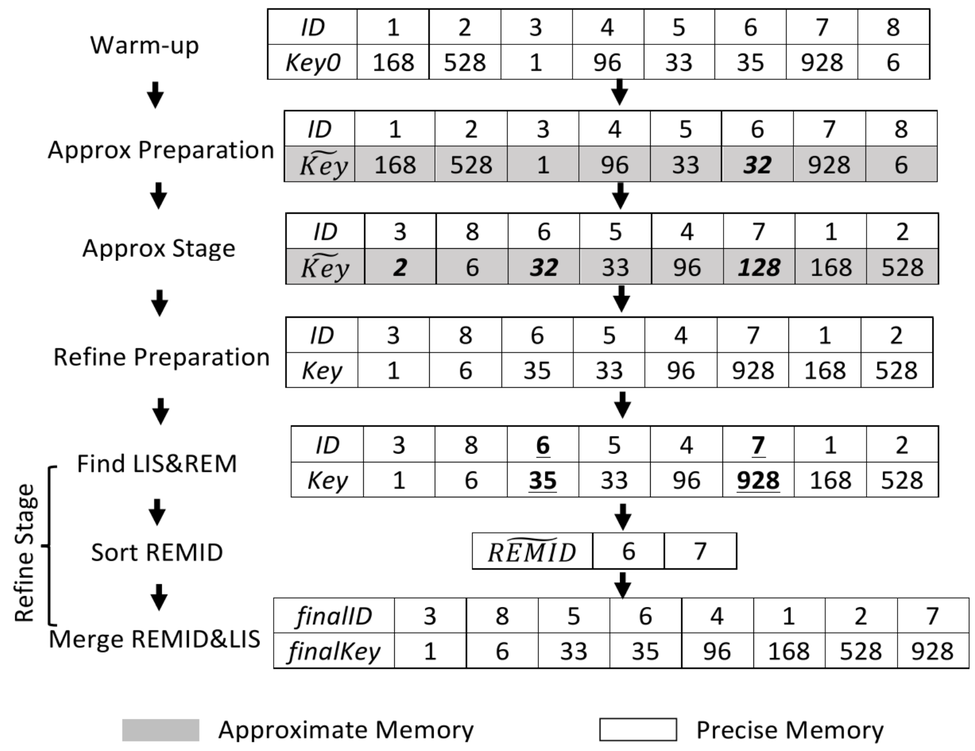

Figure 5: An example of our approx-refine mechanism for sorting. Imprecise elements are marked in bold and italic. Disorders after the approx stage are marked in bold and underlined

- Sorting algorithms are accelerated on approximate memory in the appox stage. Most algorithms get almost sorted sequences after the approx stage.
- Nearly sorted sequences are refined to be strictly sorted on precise memory in the refine stage.

Challenges of the refine stage:
- overhead of copying data between approximate and precise memory is not negligible
- must introduce as few memory writes as possible

Solutions of a lightweight refine algorithm:
- fully make use of presortedness of the sequence *Key*
- use heuristics to find an approximate LIS in O(n)
- only introduce *2n+Rem(Key)* memory writes in total

Evaluation

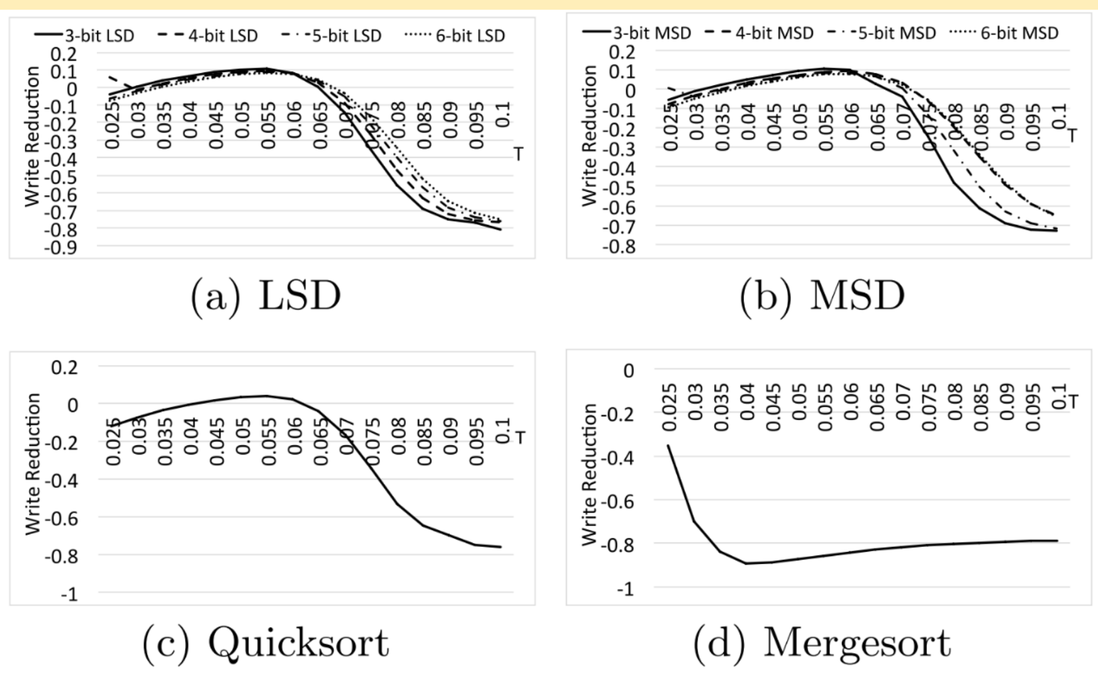

Figure 6: Relationship between imprecision (T) and performance (write reduction) of different algorithms. Please refer to our paper for more detailed evaluations on approximate memory models.

Conclusion

1. We showcase that approximate storage can be not only be used for approximate computing, but also improving performance and/or energy efficiency of precise computing.
2. We evaluate four sorting algorithms on hybrid storage systems. We demonstrate great potential of approximate storage to provide speedup with handful inaccuracies.
3. It is a nontrivial task to refine imperfect results. Refine algorithms should make fully use of the marginal unsortedness and introduce negligible overhead.