CS254 PROJECT PROPOSAL: CACHE OBLIVIOUS ALGORITHMS

Kartik Garg
ee180002027@iiti.ac.in

Tarun Gupta
cse180001059@iiti.ac.in

ABSTRACT

A cache-oblivious algorithm (or cache-transcendent algorithm) is an algorithm designed to take advantage of a CPU cache without having the size of the cache (or the length of the cache lines, etc.) as an explicit parameter.

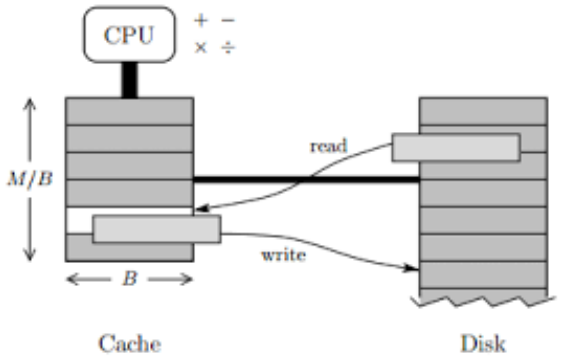
1 INTRODUCTION

An algorithm is defined to be cache-aware if it contains parameters (set at either compile-time or runtime) that can be tuned to optimize the cache complexity for the particular cache size and line length. Otherwise, the algorithm is cache oblivious. Since a lot of computer time can be spent accessing (looking up) data, we should seek to reduce this cost when possible, as faster programs can mean a smaller utility bill in CPU cycles. Big computer systems such as data centers does lots of operations related to accessing data and it makes sense to reduce this costs.

One way to do this is via "caching" with multi-level (hierarchical) "cache" data structures. However, it turns out that the size of this hardware or software cache can have an impact on both hardware cost and the eventual performance. Thus, hardware engineers and software builders (both operating system and application-level) spend time optimizing their cache hierarchies.

However, this means that performance can be sensitive to the size of the cache at each level, making it hard to tweak appropriately. To address this, work has been done in cache-oblivious data structures and algorithms used to work on those constructs.

Resource-oblivious algorithms that nevertheless use resources efficiently offer advantages of simplicity and portability over resource-aware algorithms whose resource usage must be programmed explicitly. In this project, we try to implement and analyse several "cache- oblivious" algorithms that use cache as effectively as "cache-aware" algorithms.



2 PROJECT GOALS

- Understanding of key ideas and concepts involved in design of cache oblivious algorithms.
- Implementation and analysis of the following cache-oblivious algorithms:
  - Matrix multiplication and matrix transposition.
  - Scanning - traversal and aggregates.
  - Fast fourier transform.
  - Funnelsort, mergesort and distribution sort.
  - Static data structures such as Van Emde Boas search tree.
- Pursue to design novel cache-oblivious counterparts for existing cache-aware algorithms.

REFERENCES

Guy E Blelloch, Phillip B Gibbons, and Harsha Vardhan Simhadri. Low depth cache-oblivious algorithms. In *Proceedings of the twenty-second annual ACM symposium on Parallelism in algorithms and architectures*, pp. 189–199, 2010.

Matteo Frigo, Charles E Leiserson, Harald Prokop, and Sridhar Ramachandran. Cache-oblivious algorithms. *ACM Transactions on Algorithms (TALG)*, 8(1):1–22, 2012.