THESIS TITLE

M.Tech Project Stage 1 Report

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology

by
Anubhav Jais
(Roll No. 203079012)

Under the guidance of
Prof. Virendra Singh

Department of Electrical Engineering
Indian Institute of Technology Bombay
October 22

Acknowledgement

I express my gratitude to my guide Prof. Virendra Singh for providing me the opportunity to work on this topic.

Anubhav Jais
Electrical Engineering
IIT Bombay

Abstract

XX.

Contents

List of Figures

Chapter 1

Introduction

Graph processing is an important class of computations with valuable applications in network analysis, path-planning, machine learning, and COVID-19 therapeutics discovery. Sparse irregular algorithms are widely deployed in several application domains including social networks, online navigation systems, machine learning, and genomics. In the past, large graphs were primarily processed on distributed systems. Today, increased main memory capacity and core counts allow processing graphs with billions of edges more efficiently on a single machine rather than a distributed system.

Despite their prevalence, current hardware-software implementations on the CPUs offer sub-optimal performance that can be further improved. This is due to the irregular nature of their memory access patterns over large data sets, which are too big to fit in the on-chip caches, leading to several costly DRAM accesses.

The large size of input graphs make high performance single-machine graph processing a challenge. Processing a typical input graph leads to a working set size much larger than the available on-chip cache capacity, leading to many DRAM accesses. The latency of these DRAM accesses often dominate execution time because memory accesses in graph processing are irregular and depend on graph structure. Prior work estimates that graph kernels spend up to 80% of total time simply waiting for DRAM. Techniques to improve cache locality and eliminate DRAM accesses are a major opportunity for improving end-to-end graph application performance. A cache's replacement policy is a key determinant of locality. Decades of work have produced high-performance replacement policies for various workloads. However, we find that state-of-the-art replacement policies are ineffective for graph processing. Graph data reuse is dynamically variable and graph-structure-dependent, two properties not captured well by existing replacement policies.

Therefore, traditional techniques to improve memory latency—out-of-order processing, on-chip caching, and spatial/address-correlating data prefetching are inadequate.

1.1 Work done so far

1.1.1 P-OPT: Practical Optimal Cache Replacement for Graph Analytics

The main insight of this work is that for graph applications, the transpose of a graph succinctly represents the next references of all vertices in a graph execution; enabling an efficient emulation of Belady's MIN replacement policy. In this work, P-OPT is proposed which is an architecture solution that uses a specialized compressed representation of a transpose's next reference information to enable a practical implementation of Belady's MIN replacement policy.

The paper shows that state-of-the-art cache replacement policies are ineffective for graph processing. It shows that guiding replacement based on graph structure in T-OPT allows emulating Belady's MIN policy without oracular knowledge of all future accesses. P-OPT's quantized Rereference Matrix data structure and architectural mechanisms that allow low-cost access to the graph transpose for optimal replacement. Evaluate P-OPT across a range of graphs and applications comparing to many state-of-the-art systems, showing P-OPT's consistent performance benefits.

- Studied the implementation for POPT and TOPT codes in SniperSim simulator.
- Studied how Push and Pull based executions happen in POPT for pagerank-delta benchmark.
- Trying to implement LIGRA applications, Single-Source-Shortest-Path (sssp) to make them work in SniperSim.

1.1.2 Prodigy: Improving the Memory Latency of Data-Indirect Irregular Workloads Using Hardware-Software Co-Design

Prodigy is a low-cost hardware-software codesign solution for intelligent prefetching to improve the memory latency of several important irregular workloads. Prodigy targets irregular workloads including graph analytics, sparse linear algebra, and fluid mechanics that exhibit two specific types of datadependent memory access patterns. Prodigy adopts a “best of both worlds” approach by using static program information from software, and dynamic run-time information from hardware.

The core of the system is the Data Indirection Graph (DIG)—a proposed compact representation used to express program semantics such as the layout and memory access patterns of key data structures. The DIG representation is agnostic to a particular data structure format and is demonstrated to work with several sparse formats including CSR and CSC. Program semantics are automatically captured with a compiler pass, encoded as a DIG, and inserted into the application binary. The DIG is then used to program a low-cost hardware prefetcher to fetch data according to an irregular algorithm’s data structure traversal pattern. We equip the prefetcher with a flexible prefetching algorithm that maintains timeliness by dynamically adapting its prefetch distance to an application’s execution pace.

The paper evaluates the performance, energy consumption, and transistor cost of Prodigy using a variety of algorithms from the GAP, HPCG, and NAS benchmark suites.

- Studied the Prodigy paper and tried to find what improvements can be made to better the performance.
- Studying the code currently and understanding how Prodigy is implemented in SniperSim simulator.

1.2 Final Goal of the work

The final aim of this work is to reduce the memory latency in graph applications thereby increasing the performance of the system.

Chapter 2

Future Work

- Future work consists of finding techniques through which we can get more hits in LLC and L2 cache.
- To check for nodes with higher degree, whether a performance improvement can be obtained there.