Smart Page Size Allocation of Parallel Graph Applications in a Tiered-memory System

Kaiwen Xue
kaiwenx@andrew.cmu.edu

1 Justification for a One-person Team

The course staff permitted me to work on this project individually. The project is an extension of my ongoing research work done in the CAOS group at CMU. Sum Yin Kwok and Kaiyang Zhao, while not enrolled in our class, also contributed to the original project. Their specific effort will be credited in the final report.

2 Links to Project Website

This project is hosted on http://kevinrsx.github.io/memory-gapbs. The GitHub repository for this project is available at https://github.com/KevinRSX/memory-gapbs.

3 Summary

In this project, we shall analyze the performance of parallel graph applications in a tiered-memory system with multiple page sizes and explore how smartly allocating pages of different sizes inter-plays with such an environment. Both studies will involve various aspects of the parallel applications, including end-to-end performance, TLB pressure, cache performance, work imbalance, etc.

4 Background and Challenges

Graph applications are often memory intensive. Techniques to boost the performance of those applications include: 1) using huge pages to reduce the TLB miss rate to improve address translation overhead, 2) leveraging a tiered-memory system, where the traditional main memory is divided into heterogeneous devices with different latency, bandwidth, and cost, to balance the cost and performance, and 3) utilizing shared-memory-based parallel programming frameworks, such as OpenMPI, to distribute the task to multiple processors.

Currently, studies focusing on huge pages [3–5] typically assume a homogeneous main memory, whereas those centered on tiered-memory systems, such as CXL [1], mainly address the migration of standard 4KB pages across the different tiers. For a system with homogeneous main memory extremely optimized for minimizing TLB miss rate using huge pages, it might fail to offload proper cold pages to lower tiers because they are promoted to a huge page and cannot be directly offloaded. On the other hand, if the hotness of smaller regions is imbalanced within a huge page, candidate pages selected for offloading to lower tiers using existing algorithms developed by tiered-memory researchers could be sub-optimal.

Parallelism complicates these issues even more for various reasons. For example, additional TLB shootdown is required whenever a page is migrated, or additional contention can happen since two processors are accessing the same page in the lower tier memory (CXL) together. It would therefore be challenging to answer the questions such as:

- what pages should be promoted to large pages?
- what pages should be offloaded to the CXL (lower tier) memory?
- how do different processors behave in a tiered-memory system with multiple page sizes?

5 Resources

We will use the Entropy server provided by the Parallel Data Lab to conduct the experiment. We will use modified versions of of QEMU and The Structural Simulation Toolkit to simulate the hardware environment we need. We will use the GAP Benchmark Suite [2] to run the parallel graph applications with OpenMPI.

6 Goals and Deliverables

This project is composed of two parts. The first part is an analysis of parallel graph applications running on tiered-memory

systems with different page sizes. This part will motivate a need for a set of smart memory management policies in this environment. In the second part, we will propose an algorithm for smartly managing page sizes based on the page access temperature gathered from a novel technique called *hotness walk*. In both parts, we will compare work balance and memory access pattern across different processors, and describe how the environment and our algorithm inter-plays with the parallel workloads.

Specifically, by the deadline of the project final report, we plan to have completed the following:

- a hardware simulation environment for CXL and multiple page sizes running on multiple cores
- a study that motivates a need for a set of smart memory management policies in a tiered-memory system with different page sizes
- a characterization of the memory access behavior as well the related overhead in the data cache and TLB on different cores

We hope to complete the following if the previous part shows solid results and if the time permits:

- development of an algorithm that dynamically allocates pages of different sizes based on historical memory access data and an analysis of its effect on parallel graph applications

7 Schedule

Date	Progress Anticipated
Wed Nov. 23	Hardware simulation environment
Wed Nov. 30	Motivation study
Wed Dec. 7	Multicore characterization
Wed Dec. 14 (Deadline)	Algorithm development or report writing

References

[1] Compute express link. https://www.computeexpresslink.org/.

[2] Scott Beamer, Krste Asanović, and David Patterson. The gap benchmark suite, 2017.

[3] Youngjin Kwon, Hangchen Yu, Simon Peter, Christopher J. Rossbach, and Emmett Witchel. Coordinated and efficient huge page management with ingens. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, page 705–721, USA, 2016. USENIX Association.

[4] Aninda Manocha, Zi Yan, Esin Tureci, Juan Luis Aragón, David Nellans, and Margaret Martonosi. Architectural support for optimizing huge page selection within the os. In *Proceedings of the 56th International Symposium on Microarchitecture (MICRO)*. IEEE, 2023.

[5] Ashish Panwar, Sorav Bansal, and K. Gopinath. Hawkeye: Efficient fine-grained os support for huge pages. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19, page 347–360, New York, NY, USA, 2019. Association for Computing Machinery.