Proceedings of

, , ,

IMPROVING SERVER THROUGHPUT FOR BIG DATA ANALYTICS USING SMART OFFLOADING

Emmanouil Anagnostakis
Graduate Research Assistant
Computer Architecture and VLSI Systems Laboratory, ICS-FORTH
Heraklion, Greece
manosanag@ics.forth.gr

ABSTRACT

With the increasing size and complexity of big data, there is a growing need for efficient and scalable data processing frameworks such as Apache Spark. However, the high computational and memory requirements of big data analytics workloads pose significant challenges for cluster computing systems. In this paper, we propose an offloading technique to improve job throughput for big data analytics workloads on Spark clusters.

Our offloading technique leverages the capabilities of servers that run instances of the Java Virtual Machine, the core of Big Data Analytics like Apache Spark, to move large parts of the Java Heap from the main memory to a memory mapped secondary heap over a fast storage device called TeraHeap. TeraHeap 1) eliminates Serialiazation/Deserialization overheads imposed by these kind of frameworks when moving data offheap to/from fast storages devices 2) eliminates GC over the secondary heap therefore significantly minimizing overall GC overhead. By offloading computation-intensive tasks, we aim to reduce the workload on Spark workers, thereby improving their performance and job throughput. We also explore the tradeoffs between the cost of offloading and the performance gains achieved.

We evaluate our offloading technique using various big data analytics workloads on a Spark cluster. Our results show that our offloading technique can significantly improve job throughput for big data analytics workloads, while maintaining a reasonable cost of offloading.

Overall, our offloading technique offers a promising approach to improving job throughput for big data analytics workloads on Spark clusters, particularly for computation-intensive tasks. The proposed technique can be easily integrated into existing Spark clusters and offers a scalable solution for processing increasingly large and complex big data workloads.

INTRODUCTION

With the exponential growth of data in various fields such as finance, healthcare, social media, and e-commerce, there is a significant need for scalable and efficient big data processing frameworks. Apache Spark is one such framework that has gained popularity due to its ability to handle large-scale data processing and analytics. Spark provides a distributed computing platform that can process data in parallel across multiple nodes in a cluster. However, with the increasing size and complexity of big data workloads, Spark clusters are facing significant challenges in meeting the performance and throughput requirements.

One of the main challenges in Spark clusters is the high computational and memory requirements of big data analytics workloads. These requirements can result in excessive CPU and memory usage on Spark workers, leading to performance bottlenecks and slow job completion times. To address these challenges, researchers have proposed various techniques to optimize the performance of Spark clusters, including data partitioning, caching, and resource allocation.

In this paper, we propose a new technique for improving the performance and job throughput of Spark clusters by offloading computation-intensive tasks to fast storage devices such as NVMe. Our approach leverages the capabilities of the executing machine to offload computation-intensive tasks from the Spark workers, thereby reducing the workload on the workers and im-

proving their performance and job throughput.

The main contribution of this paper is a comprehensive evaluation of the performance and cost trade-offs of offloading computation-intensive tasks to fast storage devices in Spark clusters. We demonstrate the effectiveness of our approach using various big data analytics workloads on a Spark cluster. We also compare our approach with the native Spark distribution and show that our approach can be used instead of this distribution to improve performance.

The rest of the paper is organized as follows. In section 2, we discuss related work on Spark optimization techniques and offloading techniques. In section 3, we describe our proposed offloading technique and its implementation in Spark clusters. In section 4, we present our experimental results and evaluate the performance and cost trade-offs of our approach. Finally, we conclude the paper in section 5 and discuss future research directions.

BACKGROUND

DESIGN

EVALUATION

RESULTS

RELATED WORK

FUTURE WORK

CONCLUSION

In this paper, we proposed a new technique for improving the performance and job throughput of Spark clusters by offloading computation-intensive tasks to fast storage devices such as NVMe. Our approach leverages the capabilities of the underlying running machine to offload computation-intensive tasks from the Spark workers, thereby reducing the workload on the workers and improving their performance and job throughput.

Our experimental results demonstrate the effectiveness of our approach using various big data analytics workloads on a Spark cluster. We also compared our approach with the native Spark distribution and showed that our approach can be used instead of this distribution to further improve performance.

Our work contributes to the growing body of research on improving the performance and scalability of Spark clusters for big data analytics workloads. Our approach offers a scalable solution for processing increasingly large and complex big data workloads and can be easily integrated into existing Spark clusters.

Future research directions include investigating the use of other storage mediums such as the hybrid NVM and developing techniques for dynamically adjusting the offloading decisions based on workload characteristics and resource availability.

Overall, our offloading technique offers a promising approach to improving job throughput for big data analytics workloads on Spark clusters, particularly for computation-intensive tasks. With the increasing demand for efficient and scalable big data processing frameworks, our approach provides a valuable contribution to the field of big data analytics.

ACKNOWLEDGMENT

REFERENCES

[1] Kolokasis, I. G., Papagiannis, A., Pratikakis, P., Bilas, A., Zakkak, F., Evdorou, G., Akram, S., and Kozanitis, C., 2023. "Teraheap: Reducing memory pressure in managed big data frameworks". In ASPLOS '23, March 25–29, 2023, Vancouver, BC, Canada, Association for Computing Machinery.

[2] Marco, V. S., Taylor, B., Porter, B., and Wang, Z., 2017. "Improving spark application throughput via memory aware task co-location: A mixture of experts approach". In Proceedings of Middleware '17, Las Vegas, NV, USA, Association for Computing Machinery.

[3] Kirisame, M., Shenoy, P., and Panchekha, P., 2022. "Optimal heap limits for reducing browser memory use". In OOPSLA, Association for Computing Machinery.

[4] Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., and Stoica, I., 2016. "Apache spark: A unified engine for big data processing". In Communications of the ACM, Association for Computing Machinery.

[5] Amaro, E., Branner-Augmon, C., Luo, Z., Ousterhout, A., Aguilera, M. K., Panda, A., Ratnasamy, S., and Shenker, S., 2020. "Can far memory improve job throughput?". In EuroSys '20, April 27–30, 2020, Heraklion, Greece, Association for Computing Machinery.

[6] Weiner, J., Agarwal, N., Schatzberg, D., Yang, L., Wang, H., Sanouillet, B., Sharma, B., Heo, T., Jain, M., Tang, C., and Skarlatos, D., 2022. "Tmo: Transparent memory offloading in datacenters". In ASPLOS '22, February 28 – March 4, 2022, Lausanne, Switzerland, Association for Computing Machinery.

[7] Sharma, P., Kulkarni, P., and Shenoy, P. "Per-vm page cache

partitioning for cloud computing platforms". Association for Computing Machinery.