OPS

John Doe[1] and Jane Monroe[2]
[1]*Dept. Cloud, Cloud University* [2]*Cloud National Labs*

Submission Type: Research

Abstract

In distributed computing frameworks like MapReduce and Spark, shuffle phases are barriers between the adjacent computing phases. Since the transmission of the massive data, shuffle phases account for more than 33% of job completion time. In this paper, we optimize shuffle phases in two ways: (1) we decouple the shuffle from the computing phases to increase the resource utilization, and (2) we optimize the network scheduling to decrease the communication time in the shuffle phases. We present *OPS*, a system that enables the common computing frameworks to use scheduling algorithms to optimize the whole data shuffle progress. We show that *OPS* improves the end-to-end job completion time by up to ??. We also show ...

1 Introduction

In recent years, multiple computing frameworks, such as Hadoop MapReduce [], Spark [] and Dryad [], are widely used to analyze big data. Many existing studies are focused on improving the performance of the frameworks. For example, ...

Most of the computing frameworks use the directed acyclic graphs (DAGs) to define the execution logic of jobs. The communication between stages of the most computing frameworks follows the bulk-synchronous parallel (BSP) model, such as the shuffle phases in the Hadoop MapReduce and Spark. According to the BSP model, the shuffle phases are burdens between the adjacent computing phases. Since the communication relies on plenty of disk and network I/O, the shuffle phases usually have a heavily affect on the end-to-end application performance. In the production workload [], the shuffle phases occupy 33% of the job completion time on average., and up to 70% in shuffle-heavy jobs.

Several efforts are made to optimize the shuffle phases, both on the application-level [] and the network-level [].

In this paper,

However, such early-start has several deficiencies. First, the early-start always introduces an extra early allocation of the slot leading to a slow execution of the current stage. Second, the early-start can not fully overlap the shuffle phases and the descendent stage. Due to the shuffle phases are coupled with the reduce phases and the size of slots is limited, only parts of reduce tasks can be early-start. Last but not least, it is hard to find the optimal time to begin the early-start. If too early, the descendent stages are not ready to output enough intermediate data for the shuffle. If too late, the idle network resource is wasted.

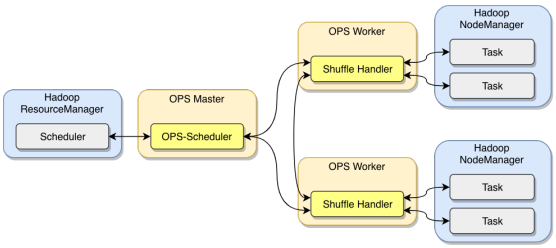

Figure 1: OPS Architecture

2 Motivation

Incast [1]

3 Design

hello

4 OPS Overview

4.1 Architecture

5 Evaluation

hello

6 Conclusion

hello

References

[1] V. Vasudevan, A. Phanishayee, H. Shah, E. Krevat, D. G. Andersen, G. R. Ganger, G. A. Gibson, and B. Mueller. Safe and effective fine-grained tcp retransmissions for datacenter communication. In *ACM SIGCOMM computer communication review*, volume 39, pages 303–314. ACM, 2009.