Lever: Towards Low-Latency Batched Stream Processing by Pre-Scheduling
Chen, Fei; Wu, Song; Jin, Hai; Yao, Yin; Liu, Zhiyi; Gu, Lin; Zhou, Yongluan

Published in:
Lever: Towards Low-Latency Batched Stream Processing by Pre-Scheduling

Publication date:
2017

Citation for published version (APA):
Lever: Towards Low-Latency Batched Stream Processing by Pre-Scheduling

Fei Chen¹, Song Wu¹, Hai Jin¹, Yin Yao¹, Zhiyi Liu¹, Lin Gu¹ and Yongluan Zhou²
¹SCTS/CGCL, Huazhong University of Science and Technology, China
²Department of Computer Science, University of Copenhagen
wusong@hust.edu.cn

ABSTRACT

With the vast involvement of streaming big data in many applications (e.g., stock market data, sensor data, social network data, etc.), quickly mining and analyzing such data is becoming more and more important. To provide fault tolerance and efficient stream processing at scale, recent stream processing frameworks have proposed to adapt batch processing systems, such as MapReduce and Spark, to handle streaming data by putting the streams into micro-batches and treating the workloads as a continuous series of small jobs [1].

The fundamental challenge of building a batched stream processing system is to minimize the processing latency of each micro-batch. In this paper, we focus on the straggler problem, where a subset of workers are straggling behind and significantly affecting the job completion time. The straggler problem is a well-known critical problem in parallel processing systems. In comparing to large batch processing, the straggler problems in micro-batch processing are more severe and harder to tackle.

We argue that the problem of using the existing straggler mitigation solutions for micro-batch processing is that they detect (or predict) stragglers and re-schedule stragglers too late in the data handling pipeline. The re-scheduling actions are carried out during the task execution period, hence it would inevitably increase the processing time of the micro-batches. Furthermore, as the data have already been dispatched, re-scheduling would inherently incur expensive data relocation. Such overhead would become significant in micro-batch processing due to the short processing time of each micro-batch. We refer to this type of methods as post-scheduling techniques.

To address the problem, we propose a new pre-scheduling framework, called Lever, which predicts stragglers and makes timely scheduling decisions to minimize the processing latency. As shown in Figure 1, Lever periodically collects and analyzes the historical job profiles of the recurring micro-batch jobs. Based on such information, Lever pre-schedules the data through three main steps, i.e. identify potential stragglers, evaluate node capacity and choose suitable helpers. More importantly, Lever makes the re-scheduling decisions before the batching module dispatches the data. As the scheduling is done while the data are being batched, it would not increase the processing time of the micro-batch.

We implemented Lever in Spark Streaming, which has been contributed to the open source community as an extension of Apache Spark Streaming. To the best of our knowledge, this is the first work specifically addressing the straggler problem in continuous micro-batch processing. We conduct various experiments to validate the effectiveness of Lever. The experimental results demonstrate that Lever reduces job completion time by 30.72% to 42.19% and outperforms traditional techniques significantly.

CCS CONCEPTS
• Organization and Design → Distributed Systems;

KEYWORDS
stream processing; recurring jobs; straggler; scheduling;

ACKNOWLEDGMENTS
This research is supported by National Key Research and Development Program under grant 2016YFB1000501, 863 Hi-Tech Research and Development Program under grant No. 2015AA01A203, and National Science Foundation of China under grants No. 61232008.

REFERENCES