Cooperative Job Scheduling and Data Allocation in Busy Data-Intensive Parallel Computing Clusters

Guoxin Liu
Clemson University
guoxinl@clemson.edu

Haiying Shen
University of Virginia
shenh@virginia.edu

Background and Motivation

Data-intensive parallel computing clusters have become more important than ever in meeting the needs of big data processing. Such a computing cluster is usually shared by multiple users who submit data and jobs to the cluster.

It is important to provide deadline-guaranteed service to jobs while minimizing the resource usage (e.g., network bandwidth and energy) in the cluster in order to reduce the cluster capital investment and operation cost [1]. To reduce network load, job schedulers need to achieve data locality, in which a task is assigned to the server closest to its requested data. The schedulers also need to reduce energy consumption by minimizing the number of running servers.

Current solutions

A **Locality-aware job scheduler** improves data locality in order to improve throughput. A **Deadline-aware job scheduler** focuses on meeting job deadline requirements. A **Data allocation** focuses on data availability to handle machine failures in computing clusters.

Our approach firstly uses job-scheduling-first computing framework to enable job scheduler and data allocation scheduler to cooperatively achieve high data locality, deadline guarantee and high energy savings simultaneously**.**

Our approach

CSA: Cooperative job Scheduling and data Allocation method.

CSA proposes a requester-consolidation deadline aware pre-scheduling to reduce deadline violations. Further, based on the determined task schedule, it proposes a cooperative data allocation to allocate a data block as close as possible to the server that hosts most of this data's requester tasks in the system in order to maximally achieve data locality. Finally a recursive refinement recursively adjusts the task schedule and the data allocation schedule to achieve the tradeoff between data locality and energy savings with specified weights.

Design Details

CSA novelly changes data-allocation-first to **job-scheduling-first**. Job-scheduling-first enables CSA to proactively consolidates tasks with more common requested data to the same server when conducting deadline-aware scheduling, and also consolidate the tasks to as few servers as possible to maximize energy savings.

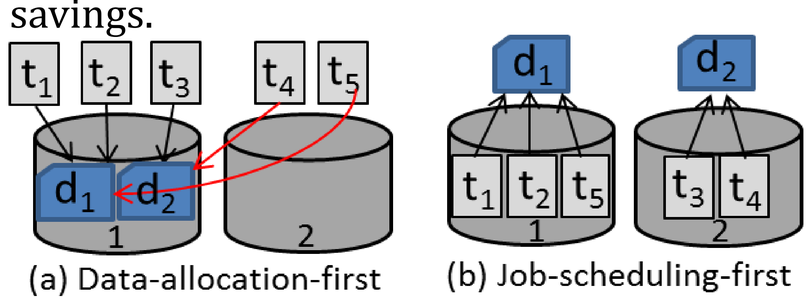

Figure 1: Data-allocation-first vs. job-scheduling-first.

Scheme 1: Locality-aware job scheduler

- Common-data Requester Consolidation: CSA proactively consolidates tasks with more common requested data to the same server.
- Deadline-aware FIFO Pre-scheduling: CSA schedule the submitted jobs according to their deadline urgency.

Scheme 2: Cooperative Data Allocation

- Giving a higher priority to a data block with a larger utilization to be allocated first to minimize its network load can reduce the total network load more.
- In order to maintain the effectiveness of energy savings achieved by the pre-scheduling algorithm, CSA allocates the data to active servers first.
- Allocate a data block into a server with more tasks requesting it

Scheme 3: Recursive Refinement

- As shown in Figure 2, this recursive refinement process includes a task schedule refinement algorithm and the cooperative data allocation algorithm introduced above.
- Each iteration improves the data locality, but increases the energy cost while maintaining the same number of tasks without deadline violations, which terminates till no cost improvement than its last iteration.

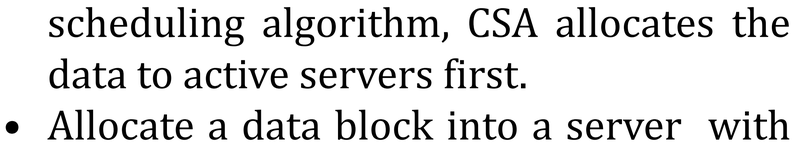

Figure 2 Data allocation recursive process.

Experimental Results

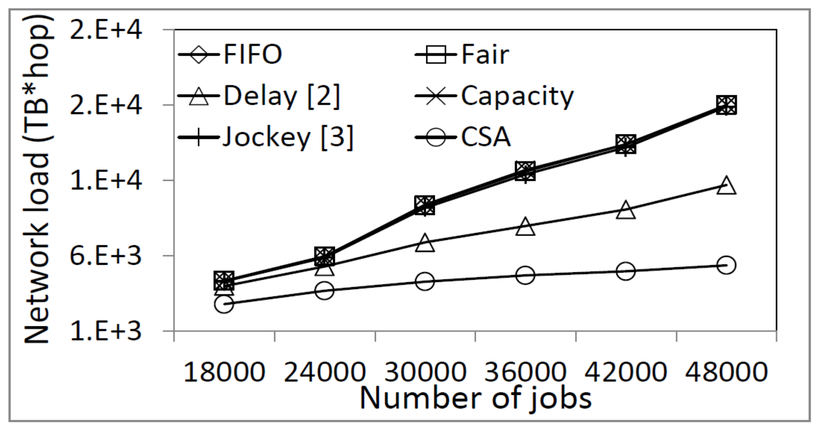

Figure 3 Network load.

Result: Highest data locality.

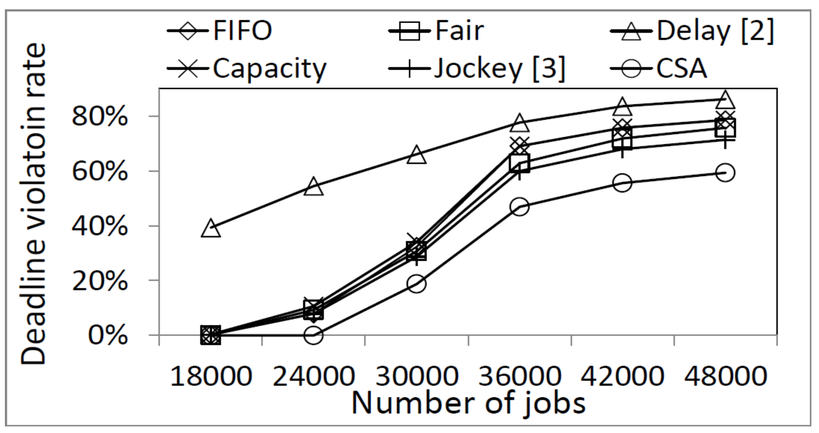

Figure 4 Deadline guarantee.

Result: Lowest deadline violation rate.

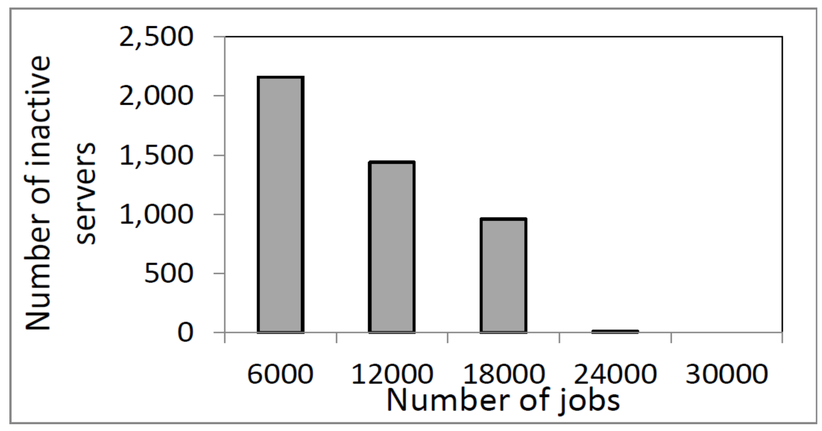

Figure 5 Energy savings.

Result: Enable energy saving.

References:

[1] M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, and J. Wilkes. Omega: Flexible, Scalable Schedulers for Large Compute Clusters. In Proc. of EuroSys, 2013.

[2] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica. Delay Scheduling: A Simple Technique for Achieving Locality and Fairness in Cluster Scheduling. In Proc. of EuroSys, 2010.

[3] A. D. Ferguson, P. Bodik, S. Kandula, E. Boutin, and R. Fonseca. Jockey: Guaranteed Job Latency in Data Parallel Clusters. In Proc. of EuroSys'13.

Future Work

In the future, we will extend the scheduling algorithm for jobs without planned submission.

Acknowledgments

U.S. NSF grants NSF-1404981, IIS-1354123, CNS-1254006, IBM Faculty Award 5501145 and Microsoft Research Faculty Fellowship 8300751.