Coflow Scheduling of Multi-stage Jobs with Isolation Gaurantee

Zifan Liu, Haipeng Dai and Wanchun Dou
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, Jiangsu, China
zifanliu@smail.nju.edu.cn, haipengdai,douwh@nju.edu.cn

TABLE I
KEY TERMS AND DESCRIPTIONS

Terms	Description
M	The number of total jobs.
K	The number of machines.
$\mathbf{F}_i = \langle f_i^1, \ldots, f_i^{2K} \rangle$	Demand vector of coflow-i.
$d_i = \langle d_i^1, \ldots, d_i^{2K} \rangle$	Correlation vector of coflow-i.
$a_i = \langle a_i^1, \ldots, a_i^{2K} \rangle$	Bandwidth allocation of coflow-i.
$\overline{f_i} = \max_k f_i^k$	Bottleneck demand of coflow-i.
P_i	Progress of coflow-i.
Γ_m	Progress of job-m.

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—keyword, keyword, keyword**

I. INTRODUCTION

This document is a model and instructions for LaTeX. [1] Please observe the conference page limits.

II. MODEL AND OBJECTIVE

In this section, we describe the model of datacenter networks and coflow.

A. Model

To simplify the discussion, key terms used in our model are summarized in Table 1.

Given the full bisection bandwidth, which has been well developed in modern datacenter [2], we treat the datacenter network as a big no-blocking switch connecting K machines. Each machine has one ingress port and one egress port, thus the whole fabric has $2K$ ports. In this simplified model, the edges are the only place for congestion. Hence we focus sorely on bandwidth of each port. In our analysis, all links are assumed of equal capacity normalized to one.

The coflow abstraction presents the communication demand within two stages of parallel computing model. A coflow is composed of a collection of flows across a group of machines sharing a common performance requirement. The completion time of the latest flow defines the completion time of this coflow. In many data-parallel frameworks like MapReduce/Hadoop, the coflow properties, such as source, destination, amount of data transferred of each flow, are known as a priori [3]–[5].

Specifically, the coflow *demand vector* $\mathbf{F}_i = \langle f_i^1, \ldots, f_i^{2K} \rangle$ captures the data demand of coflow-i, where f_i^k denotes the amount of data transferred on port k. Among all flows in coflow-i, we name the port with largest traffic bottleneck port. Let the data demand on this port be the *bottleneck demand*, defined as $\overline{f_i} = \max_k f_i^k$. To simplify our analysis, the *correlation vector* $d_i = \langle d_i^1, \ldots, d_i^{2K} \rangle$ is engaged to describe the demand correlation across ports, where d_i^k is the normalized data demand on port k by the bottleneck demand, i.e., $d_i^k = f_i^k / \overline{f_i}$. This vector indicates that for every byte coflow-i sends on bottleneck port, at least d_i^k bytes should be transferred on port k.

Coflows have elastic bandwidth demand on multiple ports, comparing with individual flows. Given the bandwidth allocation vector $a_i = \langle a_i^1, \ldots, a_i^{2K} \rangle$, calculated by coflow scheduler given the demand vectors, the coflow progress is restricted by the worst-case port. Formally, *progress* of coflow-k is measured as the mininum demand-normalized allocation across ports, i.e.,

$$P_i = \min_{i:d_i^k>0} \frac{a_i^k}{d_i^k}. \tag{1}$$

Intuitively, progress of coflow-i means the transmission satisfaction ratio on the lowest port, which determines the CCT of coflow-i.

Assume a multi-stage job-m is a collection of N coflows, i.e., $\mathbf{J}_m = \{c_{m,1}, \ldots, c_{m,N}\}$. Given the bottleneck demand and progress of each coflow, i.e., $\{P_{m,1}, \ldots, P_{m,N}\}$, the progress of job-$m$ can be computed as

$$\Gamma_m = \frac{\sum_{n=1}^{N} \overline{f_{m,n}} P_{m,n}}{\sum_{n=1}^{N} d_{m,n}}. \tag{2}$$

Like above, progress of job-m indicates the collectivity transmission satisfaction ratios of all coflows belonging to it, which has significant effect on the JCT of job-m.

B. Objective

Generally,

ACKNOWLEDGMENT

REFERENCES

[1] M. Chowdhury, Z. Liu, A. Ghodsi, and I. Stoica, "HUG: Multi-resource fairness for correlated and elastic demands," in *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. Santa Clara, CA: USENIX Association, Mar. 2016, pp. 407–424.

[2] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano *et al.*, "Jupiter rising: A decade of clos topologies and centralized control in google's datacenter network," in *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4. ACM, 2015, pp. 183–197.

[3] M. Chowdhury, Y. Zhong, and I. Stoica, "Efficient coflow scheduling with varys," in *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4, 2014, pp. 443–454.

[4] M. Chowdhury and I. Stoica, "Efficient coflow scheduling without prior knowledge," in *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4. ACM, 2015, pp. 393–406.

[5] B. Tian, C. Tian, H. Dai, and B. Wang, "Scheduling coflows of multi-stage jobs to minimize the total weighted job completion time."