CSE 222A Project Proposal
Coflow Scheduling

Group 6

Ruby Pai Amit Borase Sreejith Unnikrishnan
Stas Mushits Ritvik Jaiswal

October 18, 2015

1 Introduction

Data parallel cluster applications running on fast CPUs are limited by relatively slower network connections. The coflow abstraction for such applications was recently proposed in [1], which argued that identifying the individual parallel data flows associated with an application as a "coflow" and applying application-aware scheduling would improve application performance. Subsequently, Varys, a coflow scheduler was published [3], followed by Aalo, which improves on Varys by BRIEF DESCRIPTION HERE [2]. Varys is based on heuristics that draw on insights about characteristics of an optimal coflow scheduler that would allow scheduling for minimal coflow completion time (CCT) or to meet a deadline.

The Varys scheduler was shown to improve CCTs by up to 3.16 times on average, and to increase the number of coflows meeting deadlines by a factor of 2 [3]. This is promising not only for application performance, but has implications for issues such as data center energy efficiency, as most of the power wasted in data centers is due to CPUs idling while waiting for the network. However, a significant barrier to usage of coflow schedulers like Varys or Aalo is that they require coflows to be explicitly identified by the programmer to the scheduler via an API. This poses problems in terms of legacy code, programmer burden, and finally, in cases in which the coflows that exist in an application may not be obvious or known.

Therefore, in this project, we propose to examine coflow scheduling in two parts. First, we will replicate the results of [3] using the same trace, which is available on [4] and extend the experiment to traces for different data parallel applications. Second, taking Varys as the established state-of-the-art in coflow scheduling (as the code for Aalo does not appear to be available yet) we will examine the impact of incomplete coflow identification on the scheduler. Examining the robustness of the current state-of-the-art scheduler to incomplete coflow labeling will inform whether automated algorithms for identifying coflows, using for example machine learning techniques, which cannot be expected to be 100% accurate, are viable with current coflow scheduling techniques.

2 Approach

As mentioned Section 1, the proposed project consists of two parts. The first part is to validate results from published literature, extending the results to new workloads. The second part consists of taking the first step in considering how the published result (in this case, the Varys scheduler) could work with automated coflow identification, which we argue is important for the reasons stated in 1.

3 Related Work

Besides the seminal work of Chowdhury et al on coflows and coflow scheduling [1, 3, 2],

NEED TO FINISH THIS SECTION Three types of papers to look for 1. on the concept of coflows in general 2. on coflow scheduling –Varys paper alludes to non-preemptive coflow schedulers 3. on coflow identification –office hours: "intervention" experimental approach (stop a flow and see which application suffers)

4 Plan

–How? Applications Used: Describe simulation environment Metrics for evaluation –When Done?

5 Schedule

Milestones 11/5 (2 weeks)

Milestone 11/24 (2.5 weeks)

Deliverables 12/3 Poster Draft Deliverable 12/8 8:00 am Poster Presentation 12/8 Midnight Final Report Due

References

[1] Mosharaf Chowdhury and Ion Stoica. Coflow: A networking abstraction for cluster applications. In *ACM Hotnets*, 2012.

[2] Mosharaf Chowdhury and Ion Stoica. Efficient coflow scheduling without prior knowledge. In *ACM SIGCOMM*, 2015.

[3] Mosharaf Chowdhury, Yuan Zhong, and Ion Stoica. Efficient coflow scheduling with varys. In *ACM SIGCOMM*, 2014.

[4] `https://github.com/coflow`. Accessed: 2015-10-18.