Harmony: Coordinating Network, Compute, and Storage in Software-Defined Clouds

Robert Grandl, Yizheng Chen, Junaid Khalid, Suli Yang
Ashok Anand, Theophilus Benson and Aditya Akella

September 15, 2013

1 Introduction

The progress of a big data job is often a function of storage, networking and processing. Hence, for efficient job execution, it is important to collectively optimize all three components. Prior proposals [1], in contrast, have focused on mainly on one or two of the three components. This narrow focus constraints the extent to which these proposals can support efficient operation of big data applications.

We argue for treating all three components as equal entities and facilitating tighter coordination among them. We argue that this helps substantially improve the performance of big data applications. This becomes apparent when we consider how coordination helps to deal with unexpected situations, e.g., straggler or "slow" tasks, or failed machines: in such cases, e.g., the cloud controller can invoke multiple replica tasks at well-informed locations (based on input about data locations and network conditions), while at the same time the network assigns higher bandwidth to the corresponding flows for faster execution. Equally importantly, coordination also improves overall resource utilization. For example, if it is known that a job may not meet its deadline [2], we can revoke its network resources (stop running flows), storage resources (clear memory for the data), and processing resources (empty out slots) collectively.

SoCC'13, 1–3 Oct. 2013, Santa Clara, California, USA.
ACM 978-1-4503-2428-1. http://dx.doi.org/10.1145/2523616.2525961

2 Architecture

Given the shortcomings of earlier approaches, we envision an architecture that enables interaction and coordination among three main components – Cloud controller (CC), Storage controller (SC) and Network controller (NC) – providing a platform for efficient execution of big-data applications. The architecture should have the following properties:

- **Participation:** The architecture should allow each of the components to participate in making decisions for any map/reduce operation in any stage.
- **Authoritative Decision:** The architecture should delegate the final decision for any map/reduce operation to the component which has the best knowledge of the domain to which the operation belongs.
- **Dynamic control:** The architecture should enable components to dynamically change the configuration for any big-data application. For example, it should be possible to revoke certain tasks, and their flows.

3 Challenges

What makes coordination and collective optimization challenging, and delegating key actions to the network sub-optimal, is that there is a high degree of dependency between the three components. For instance, the performance of storage (e.g., retrieval of data for a map task) is dependent on the underlying network conditions, which in turn is dependent on the transfers between different map and reduce tasks. The availability of processing slots determines where tasks can be placed, and in turn plays a crucial role in defining network utilization.

Achieving coordination requires disentangling and streamlining these dependencies. In other words, we must identify which of the three components – processing, storage or network – must make the authoritative decision for a certain action, what information it needs, and how it obtains the information.

References

[1] M. Chowdhury and I. Stoica. Coflow: a networking abstraction for cluster applications. In *HotNets*, 2012.

[2] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron. Better never than late: meeting deadlines in datacenter networks. In *SIGCOMM*, 2011.