Using Realistic Simulation to Identify I/O Bottlenecks in MapReduce Setups

Guanying Wang (student), Ali R. Butt, Prashant Pandey, and Karan Gupta
Virginia Tech *IBM Almaden Research*
{*wanggy,butta*}*@cs.vt.edu* {*ppandey,guptaka*}*@us.ibm.com*

1 Introduction

The exponentially growing data demands of modern enterprise and scientific applications poses critical challenges in sustaining the applications at scale. The MapReduce [1] programming model has served as the key enabler for executing resource-intensive applications over huge datasets. However, its configuration design-space has not been studied in detail. This is a complex problem as a typical MapReduce configuration can encompass hundreds of parameters, e.g., node configuration (number of disks and compute capacity), network topology (inter and intra-rack), choice of file system, data partitioning and layout, types of schedulers, etc – all of which affect application performance. While empirical insights for certain specific configurations, e.g., Google's MapReduce infrastructure [1], do exist, they cannot be simply extended to other setups. Moreover, no tool or model is available to the community for studying MapReduce application performance.

In this work, we explore how choices about cluster design, run-time parameters, multi-tenancy and application design, affect I/O patterns, network communication and performance of MapReduce applications. Since the scale of the system precludes using actual machines for this exploration, we are developing an accurate MapReduce simulator, *Dumbo*, to facilitate performance analysis.

The insights gained through Dumbo will be useful in comprehending the factors that affect MapReduce application performance. We expect Dumbo to be used by researchers and practitioners to understand how their MapReduce applications will behave on a particular configuration, and how they can improve the applications and platforms to optimize performance. *Dumbo, used as a planning tool, will make MapReduce deployment far easier by reducing the number of parameters that currently have to be hand-tuned using trial-and-error and rules of thumb.*

1.1 Challenges in Simulator Design

Dumbo is designed to simulate MapReduce applications and environments, and provide information about I/O patterns, network communications and application performance. To this end, we have used Hadoop [2], a publicly available implementation of MapReduce, as the reference for our simulator. Dumbo aims to answer questions being asked by the community about MapReduce setups: Can a particular cluster setup yield a desired I/O throughput? Can a MapReduce application provide linear speed-ups as number of machines increases? Can a particular class of applications benefit from MapReduce? In addition, Dumbo can be used to understand the sensitivity of application performance to platform parameters, network topology, node resources and failure rates.

A key challenge that we faced was determining the right level of component abstraction. If every component is simulated thoroughly, it may take prohibitively long to produce results. Conversely, if important components are abstracted out, the results may not be accurate. We opted for a balanced approach and used ns-2's [3] packet-level simulation; Computation and I/O accesses are simulated via process queues, and applications via their computation and I/O characteristics.

The performance of a MapReduce application depends on the data layout within and across racks and the associated job scheduling decisions. Dumbo is layout-aware and capable of modeling different scheduling policies. Furthermore, in some MapReduce applications, the reduce phase is dependent on the distribution of intermediate results and require special consideration for correct simulations. For now, we have assumed a uniform distribution model to address this issue, but intend to explore a variety of distribution schemes as we proceed.

Another critical issue is to correctly model failures, as failures are common in large scale commodity clusters and directly affect performance. For this purpose, we are extending Dumbo to utilize failure-traces and accurately model expected application performance under specified failure conditions.

2 Implementation of the Simulator

We have implemented a prototype which provides fine-grained simulation at sub-phase level and models network communications and activities inside a single node, such as the processor time consumed by a job, and disk I/O time for reading inputs and writing results. Dumbo takes as input node specification, cluster topology, data layout, and job description. The output is a detailed trace, which provides the execution time, the amount of data transferred, and the time-line of each task. The output trace can also be visualized for analysis.

3 Verification of Simulation Results

We have verified Dumbo using a small-sized (8-node, 32-core) cluster, and a medium-sized (40-node, 320-core) cluster to test Dumbo under different network topologies, per-node resources, and application behaviors.

We have already seen good fidelity between simulated and actual timings for certain aspects of application performance (e.g. during the Map phase). However, in our test runs, we found that some of the results predicted by Dumbo were not in line with what was observed for the real applications. We discovered that in our test configuration, network resources were under-utilized when data was being copied between the Map and Reduce phases. With this insight, we changed the Hadoop code to better utilize resources and were able to improve Hadoop's performance to the level predicted by Dumbo.

References

[1] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Commun. of the ACM*, 51(1):107–113, 2008.
[2] Hadoop, 2008. http://hadoop.apache.org/core/.
[3] ns-2, 2008. http://nsnam.isi.edu/nsnam/index.php/Main_Page.