Proposal for Project 1: Big Data Demystified

Xin Zhou
Department of ECE
Duke University
xin.zhou@duke.edu

Di Jin
Department of ECE
Duke University
di.jin@duke.edu

Huayang Cui
Department of CS
Duke University
hycui@cs.duke.edu

ABSTRACT

The infrastructure for supporting big data contains many aspects(e.g., the scheduling disciplines, the network architecture, the mechanisms for effectively sharing the network, the mechanism for enforcing low latency). In order to achieve the best performance, different combination of these techniques are applied under different circumstances.

This paper presents the analysis of the best configuration for different environment based on simulation of MapReduce. The simulator applied is based on MyPerf and MR-Sim. In order to explore the configuration under different environment, traces from Facebook and Cloudera are utilized.

Keywords

simulation, MapReduce, performance modeling, environment

1. MOTIVATION

In order to obtain better performance for processing big data, various methods have been studied. Over the last few years, various methods are given to improve different components in isolation. However, since these methods are developed based on models with different properties(e.g., scale, network), a given workload cannot apply all of these improvements. Therefore, determining the best configuration of different components for a given workload becomes critical: It provides enterprises a practical way to achieve the best performance.

On the other hand, in the process of determining the best combination of components, it is difficult to apply various methods on the same workload in a real data center. Alternatively, simulation presents a faster approach because of its features: convenience, ease-to-replace. With proper simulation method, the result obtained can be applied to real data center, thus saves time and cost.

In order to obtain the best simulation results, traces from facebook and Cloudera are applied.

2. RELATED WORK

Cardona et al.[1] implemented a simple simulator for MapReduce workloads to evaluate scheduling algorithms; Guanying Wang[2] implemented MR Perf, a simulator for MapReduce; Google Project[3] implemented MR Sim, an open-source simulator for MapReduce as well.

Roughly, the simulators can be classified into two categories: schedulers evaluation and individual jobs simulation.

2.1 MapReduce Simulators for Evaluating Schedulers

According to Guanying Wang, the simulator from Cardona et al. was the first MapReduce Simulator for evaluating schedulers. Mumak[4] implemented a simulator adopting Hadoop code. This simulator runs within actual world. SimMR[5] is implemented from scratch, which does not run schedulers implemented in Hadoop code. Therefore, simulation run done by these simulators should be fast.

2.2 Limitations of Prior Works

Most of prior simulators are limited in that they are not aware of resource contention, so tasks execution time may not be accurate. In addition, these simulators are not aware of other jobs in a workload, which makes them not applicable unless only one job is on a cluster. These disadvantages were overcame by MR Perf, according to Guanying Wang. However, MR Perf has limitation as well: its network simulation is based

[1]K. Cardona, J.Secretan, M. Georgiopoulos, and G. Anagnostopoulos. A Gird Based System for Data Mining Using MapReduce. Technical Report TR-2007-02, The AMALTHEA REU Program, 2007.
[2]Guanying Wang. Evaluating MapReduce System Performance: A Simulation Approach. PhD degree thesis, 2012.
[3]Google Project. mrsim-Map Reduce Simulator. website: http://code.google.com/p/mrsim/
[4]A.C. Murthy. Mumak: Map-Reduce Simulator. MAPREDUCE-728, Apache JIRA, Also available at http://issues.apche.org/jira/browse/MAPREDUCE-728, 2009.
[5]A. Verma, L. Cherkasova, and R.H. Campbell. Play It Again, SimMR! In *2011 IEEE International Conference on Cluster Comupting*, page 253-261. IEEE, Sept. 2011.

Table 1: Comparison of MapReduce simulators.

	based on	Workload-aware	Resource-contention-aware
Our simulator	ns-3	yes	yes
MRPerf	ns-2	yes	yes
Cardona et al.	GridSim	yes no	
Mumak	Hadoop	yes	yes
SimMR	from scratch	yes	no
HSim	from sc-tatch	no	yes
MRSim	GridSim	no	yes
SimMapReduce	GridSim	no	yes

on ns-2. Compared with ns-3, ns-3 performs better and currently actively developed.

Our simulator combines festures of both MR Perf and MR Sim based on ns-3. Table 1 shows a comparison of advantages and drawbacks of simulators.

3. ROUGH IMPLEMENTATION PROPOSAL

In this project, we plan to adopt a simulation approach to explore the impact of design choices in MapReduce setups. We are concerned with the following components:

- Compute Scheduling: Quincy Scheduler, Delay Scheduling, Default Scheduler
- Network Sharing(Congestion Control): TCP, seawall, Orchester
- Network Topology: Tree, VL2, Fat-tree, Helios/C-through
- Compute Sharing: Mesos, Omega
- Perofrmance Hacks: dynamic number of replicas, Cloning tasks, speculative execution

3.1 Simulator Implementation

3.1.1 Setup Parameters

The simulator that we build aims to comprehensively capture the various design parameters of MapReduce setup. Building a simulator is challenging. The simulator we build aims to explore the impact of factors such as data-locality, network topology, failure on overall performance. These factors can be classified into design choices concerning infrastructure implementation, application management configuration, and framework management techniques.A brief summary of key design parameters modeled in shown in table 2.

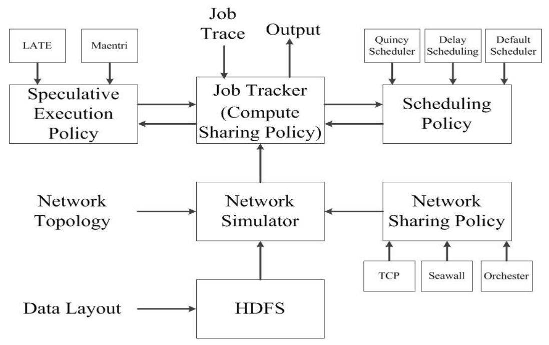
Table 2: MapReduce setup parameters modeled

Category	Example
Cluster parameters	1. Node CPU, RAM, and disk characteristics 2. Node & Rack heterogeneity 3. Network topology(inter & intra-rack) 4. Network sharing
Configuration parameters	1. Data chunk size used by the storage layer 2. Map and reduce task slots per node 3. Number of reduce tasks in a job 4. Compute sharing
Framework parameters	1. Shuffle-phase data movement protocol 2. Task scheduling algorithm 3. Data placement algorithm

3.1.2 Architecture

The rough architecture of our simulator is shown as Figure 1, and we plan to build it based on Java. Also, the network simulator is ns-3.

Figure 1: Simulator Architecture



3.1.3 Jobtracker

Input: The trace

Resource allocation strategy:

The resource allocation strategy includes Mesos and Omega which have full knowledge of the topology of all nodes and the number of slots in each node.

The input is the jobID, the output of the scheduling module that decides which map/reduce task will be executed next. The output is the placement of the task to certain slots in Node network.

Output:

The output includes three levels of information. For each job, it includes the arrival and departure time, the amount of data transferred in the network. For each task, it includes the timestamp for ready, start and end state. For each network operation, it includes the start and end time and the full path from the source to the destination.

3.1.4 Scheduling

Given the information from the trace, the scheduling module maintains a queue, denoted as jobQ, which includes the arrival and departure time of jobs, the start and end time of each map/reduce time and a signal indicating the end of the map task. There are mainly two functions to decide the next map task and reduce task from jobQ, both of which will output the jobID to be executed and related information, e.g., the length of the task, the amount of data that will transfer.

3.1.5 Storage Design

Input:

1. The number of replica: there is a default value for this. Users can also customize the value

2. Input for background traffic, including the traffic size(in Bytes), frequency and number of simultaneous transmit

Inner algorithm:

1. replication algorithm to define the number of replica

2. replication placement algorithm to decide the places for replica

3.1.6 Node

Including

1. The number of slots in each node
2. The vacancy of each slot
3. How many data transfer from and to this node

3.2 TimeTable

By 10/31 finish the main part of the simulator and write the API for all components By 11/15 finish all components By 11/30 write the algorithm to find the optimal combination for certain trace