Resource Allocation Scheme for Hadoop in Campus Networks

Tomohiro Matsuno, Bijoy Chand Chatterjee, Satoru Okamoto, and Eiji Oki
Dept. of Information and Communications
University of Electro-Communications, Japan
Emails: {t.matsuno, bijoycc, satoru.okamoto, eiji.oki}@uec.ac.jp

Malathi Veeraraghavan
Dept. of Electrical and Computer Engineering
University of Virginia, USA
Email: mv5g@virginia.edu

Naoaki Yamanaka
Dept. of Information and Computer Science
Keio University, Japan
Email: yamanaka@ics.keio.ac.jp

***Abstract*—On most campus networks, outdated computational resources are not used. This paper proposes a Hadoop-based resource allocation scheme for leveraging these outdated resources. The proposed scheme splits a computational job into tasks using appropriate splitting ratios based on computer processing performance and network resource availability. We compare our resource allocation scheme with a conventional scheme, which divides a job equally without consideration of computer processing performance and network resource availability. Both experiments and simulations are used to compare our proposed scheme and the conventional scheme. Simulation results show that the proposed scheme outperforms the conventional scheme in terms of job execution time.**

I. Introduction

In campus environments, including universities and institutes, different types of computational resources are available. In most cases, outdated computational resources are not used, which leads to an inefficient utilization of available computational resources. Different types of available computational resources can be utilized in the Hadoop framework [1], [2], [3].

Hadoop adopts a scale-out approach, which increases the capacity of the overall system by increasing the number of servers in order to improve system performance. In the current Hadoop framework [1], [4], a job is divided equally into multiple tasks, which are they are assigned to slave servers. If the processing capacities of these servers differ significantly, the total computation time may not decrease even when the number of servers is increased. In the other words, with heterogeneous servers (i.e., servers with different processing capabilities), the advantages of Hadoop may not be realized. Thus, outdated resources on campuses are often not included in jobs run in the Hadoop framework.

This paper proposes a resource allocation scheme that utilizes outdated campus servers to decrease job execution time in the Hadoop framework. The proposed scheme divides a job into unequal tasks such that tasks are matched with servers based on the task computational requirements and the processing capabilities of the servers. Also, network delays required to send messages to the slave servers from the master server are considered while making the task allocations to slave servers.

The resource allocation scheme was evaluated using a combination of experiments and simulations. First, experiments were executed on five machines in a campus network to demonstrate the problem with the current Hadoop approach of dividing a job into equal tasks with no consideration of server processing capability or network delays. Simulations were used to compare the performance of the proposed resource scheme and the current Hadoop scheme. Results show that the proposed scheme is effective in meeting the stated objective of task execution time reduction while using outdated servers.

II. Proposed Scheme

Figure 1 illustrates the concept of the proposed scheme. A Job allocation and network resource manager decides how to divide a job into tasks based on the processing capabilities of servers and network delays. Figure 2 illustrates how tasks are allocated to computational resources in the proposed scheme.

First, the processing capability of each server c_i, $1 \leq i \leq M$, is determined. Next, network delays, d_i, $1 \leq i \leq M$, are measured between the master server and each slave server. An optimization problem is solved to determine the ideal splitting ratio x_i for the task assigned to server i, $1 \leq i \leq M$ in order to minimize the longest task completion time across all slave servers. Task completion includes the transfer of results from slave servers to the master server.

Our objective is to minimize the largest processing time of slave servers. The objective function is defined as below.

$$\min \quad r \tag{1a}$$

$s.t.$

$$f(x_i, c_i, d_i) \leq r \qquad \forall i \in I \tag{1b}$$

$$\sum_{i \in I} x_i = 1 \tag{1c}$$

Eq. (1b) indicates that processing time of task i does not exceed r. To suppress processing time $f(x_i, c_i, d_i)$, of task i $\forall i \in I$, we split the job with appropriate splitting ratios according to both computer processing performance and network resource availability. Eq. (1c) shows that the total proportion of divided tasks is equal to one.

III. Performance Evaluation

This section first describes an experiment that evaluates the performance of the Hadoop framework on a network of

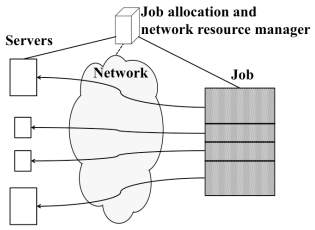

Fig. 1. Overview of proposed scheme.

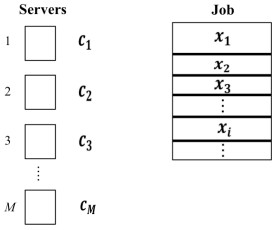

Fig. 2. Selection of computational resources and job allocation.

heterogeneous servers. Next, results of a simulation study that compares our proposed solution with the conventional Hadoop solution are presented.

A. Experimental Setup

The testbed consists of five physical machines, each executing the VMware Player software. One of the machines is used as a master server, while the remaining four machines are set up as slave servers. The job selected for this experiment was a computation of the first 100 million digits of π (=3.141$\cdots$). To create an environment with heterogeneous servers, the same π computation job was executed to add load to two of the four slave servers.

B. Analysis of the experimental results

Figures 3 and 4 plot job execution time in minutes as a function of the number of slave servers used in the computation. Fig. 3 shows that when the number of slave servers increases, the processing time decreases when the servers were homogeneous (i.e., all servers had equal processing capacity). On the other hand, Fig. 4 shows job execution time in a heterogeneous environment. The execution time increases with increases when the number of slave servers was increased. This is because of the additional computational load placed on the third and fourth slave servers. This experiment demonstrates that the expected scaling advantages of Hadoop may not be realized in an network with heterogeneous servers.

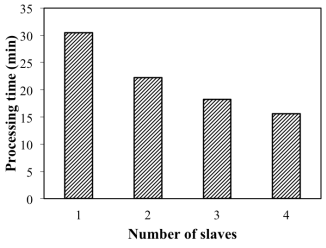

Fig. 3. Instance of efficient scale-out.

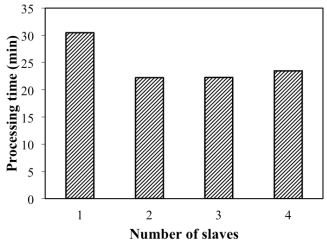

Fig. 4. Instance of inefficient scale-out.

C. Simulation Study

The goal of this study was to compare the job execution time under the conventional scheme and our proposed scheme. In the simulation, 18 computers were set to operate as slave servers. Of these, nine computers were modern high-performance servers, while the remaining nine computers were outdated low-performance servers.

Figure 5 plots the normalized job execution time, r, as a function the number of used slave servers. The normalization is with respect to the execution time of the job when using a single server. The normalized execution time using the proposed scheme decreases as the number of slave servers used is increased. This is because of the appropriate matching of tasks with server processing capabilities. On the other hand, Figure 5 shows that the normalized execution time in the conventional scheme increases, when the number of slave servers used is greater than nine. We thus conclude that the conventional scheme does not scale well when the low-performance servers are included as slaves. The simulation results for the conventional scheme are consistent with the experimental results observed in Fig. 4.

The results show that the proposed scheme can leverage outdated low-performance servers to reduce overall job execution time when compared to the conventional scheme.

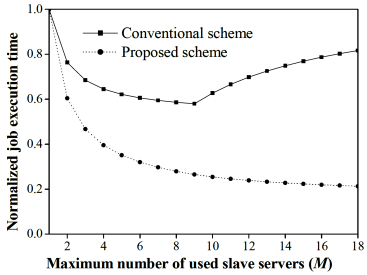

Fig. 5. Job execution time of proposed and conventional schemes.

IV. Conclusions

This paper proposed a Hadoop resource-allocation scheme for effectively utilizing outdated campus servers to reduce job execution time. The proposed scheme splits a job into unequal tasks and matches tasks based on their computational requirements with the processing capabilities of the servers to which the tasks are assigned. Additionally network delays between the master and slave servers is considered in the task sizing and assignments. We compared our resource allocation scheme with a conventional scheme, which divides a job equally without consideration of server processing capabilities and network delays. Experiments were used to demonstrate the problem with equal splitting, and simulations were used to compare our proposed scheme with the conventional scheme. Simulation results showed that the proposed scheme outperforms the conventional scheme in terms of job execution time.

Acknowledgement

This work was supported in part by the National Institute of Information and Communications Technology (NICT), Japan and the NSF grant (CNS-1405171), USA.

References

[1] "Apache Hadoop," http://hadoop.apache.org/, last Accessed: 2015-04-24.

[2] T. White, *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 2012.

[3] "Cloudera Impala Project," http://impala.io/, last Accessed: 2015-04-24.

[4] Y. Yao, J. Wang, B. Sheng, J. Lin, and N. Mi, "HaSTE: Hadoop YARN Scheduling Based on Task-Dependency and Resource-Demand," in *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*. IEEE, 2014, pp. 184–191.