Journal of Fundamental and Applied Sciences

ISSN 1112-9867

Available online at http://www.jfas.info

Research Article

Special Issue

COST-AWARE REAL-TIME DIVISIBLE LOADS SCHEDULING IN CLOUD COMPUTING

M. L. A. Majid, S. Chuprat*

Advance Informatics School (AIS), Universiti Teknologi Malaysia

Published online: 01 February 2018

ABSTRACT

Cloud computing has become an important alternative for solving large scale resource-intensive problems in science, engineering, and analytics. Resource management play an important role in improving the quality of service (QoS). This paper is concerned with the investigation of scheduling strategies for divisible loads with deadlines constraints upon heterogeneous processors in a cloud computing environment. The workload allocation approach presents in this paper is using Divisible Load Theory (DLT). It is based on the fact that the computation can be partitioned into some arbitrary sizes and each partition can be processed independently of each other. Through series of simulation against the baseline strategies, it can be found that the worker selection order in the service pool and the amount of fraction load assigned to each of them have significant effects on the total computation cost.

Keywords: Cloud computing, Divisible Load Theory (DLT), Cost, Quality-of-service (QoS).

1. INTRODUCTION

Cloud computing has recently emerged as a compelling paradigm for solving large- scale data-intensive computing. Amongst the areas that are likely to benefit greatly from it are the real-time applications such as in bioinformatics, signal processing and astronomy.

Author Correspondence, e-mail: suriayati.kl@utm.my

doi: http://dx.doi.org/10.4314/jfas.v10i2s.1

Since these applications often demand massive computation requirements and due to the highly elastic environment in cloud computing, providers must be able to dynamically manage the available resources so that they can be optimized.

In divisible load theory (DLT) paradigm in the case of clouds environment, an arbitrarily independent divisible load can be processed in parallel on multiple computing nodes. Kang et al. [1] have studied the case of a global surveillance system where a large number of geographically distributed sensors involve in system monitoring. In their paper, they adopt DLT for minimizing the total execution time but the cost factor was not considered. There are some significant works of scheduling divisible loads in cloud computing discussed about cost [2-5]. However, none of these works deals with deadline constraint workloads and considers the heterogeneity of computing nodes.

Thus, in this research, the Real-time Divisible Load Theory (RT-DLT) proposed in [6, 7] is extended to heterogeneous processors in the cloud computing platform. This paper also puts forward the idea of including worker selection strategy in the scheduling framework for improving the computation cost. The aim is to design optimal workload allocation strategies that meet the deadline while minimizing the cost. This in return will help cloud provider to gain maximize benefit while guaranteeing their QoS with the users?

The rest of this paper is structured as follows. Section 2 introduces an overview of the RT-DLT concepts and the scheduling framework; proposed RTDLT-cost model used in this paper is discussed in Section 3. Section 4 described simulation setup and result analysis. Finally, this paper ends with the findings and conclusions in Section 5.

2. RT-DLT FOUNDATIONS

Job and system models of RT-DLT used in this research are described in this section.

Job Model The job model in RT-DLT allows for the parallel execution of a job upon multiple processors. Each divisible job J_i consists of a single invocation characterized by ordered pair (σ, D), where $\sigma > 0$ is the total load size of the job and $D > 0$ is its relative deadline, indicating that it must complete execution within D time unit of arrival.

System Model Proposed architecture for scheduling divisible load on cloud environment is described in Figure 1 below. The centralized scheduler (CS) denoted as head node P_0 is responsible for scheduling loads on computing nodes, has all the information about loads and capacities of computing nodes. The scheduler needs to assign loads to compute nodes

(CN) denoted P_1, P_2, P_3… P_N. The head node does not involve in the computation – its role is to accept or reject incoming jobs, execute the scheduling algorithm, divide the workload and distribute data chunks to the computing nodes.

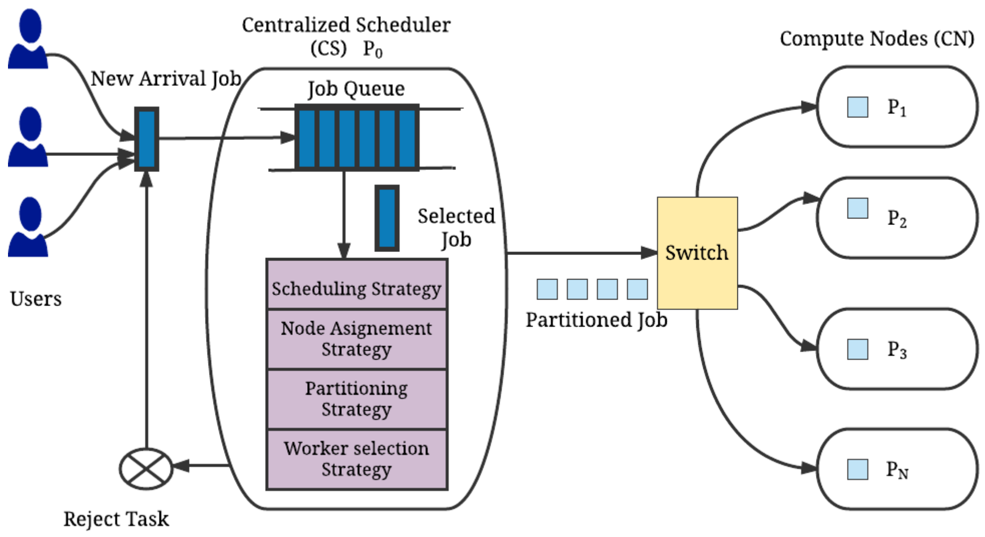

Fig.1. Scheduling framework

Scheduling Framework Figure 1 above illustrates the scheduling framework upon heterogeneous resources in cloud computing. Most authors in the literature consider three strategies, which is *Scheduling Strategy, Node Assignment Strategy and Partitioning Strategy* [2-4, 8, 9]. In this paper, the computing nodes that are heterogeneous in term of speed and cost characteristics is studied. Hence, another one strategy that is, *Worker Selection Strategy* is adopted to investigate the impact of worker order selection towards reducing computation cost.

As shown in the diagram, each arrival jobs will be arranged in the Queue. The ***Scheduling Strategy*** to determine the job order; the Node ***Assignment Strategy to*** decide how many computing node are required for the job execution**.** The third decision is ***Partitioning Strategy*** to determine a strategy to partition and compute the chunk size of load to be processed and transmit them to each compute nodes via a switch. Two different partitioning strategies are investigated: the *Optimal Partitioning Rule* (OPR), and the *Equal Partitioning Rule* (EPR). The OPR is a superior partitioning strategy based on divisible load theory (DLT) which has been introduced in [8, 10]. It states that the optimal execution time is obtained when all nodes allocated to a task complete their computation at the same time. For comparison, a baseline approach to partition a workload N equal-sized subtasks is adopted. This approach is namely as the *Equal Partitioning Rule* (EPR). Each partitioning methods has different advantages, computation time and cost would be expected to be vary with different strategy.

The last decision is ***Worker Selection Strategy*** to determine the worker sequence order. Selection of workers in cloud computing currently is done using a simple round- robin scheme and the importance of worker selection has been highlighted in the literature [11-13]. To observe the impact of this additional strategy, three sequence order of workers were examined. The sequence is most expensive to the cheapest worker, cheapest to the most expensive worker and random. These three sequence orders are using OPR partitioning strategy and compared against baseline approach EPR. If no schedule can be found to satisfy job's timing requirement although enough virtual machines have been added, the job will be rejected. The result of the simulation is thoroughly analyzed in Section 4.

3. PROPOSED CARTDLT MODEL

This section presents the proposed Cost-Aware Real-time Divisible Load Theory scheduling algorithm named CARTDLT for cloud environments. Here, the calculation to get the total computation cost is discussed.

Fig. 2 below illustrates an example of task execution time diagram. For a given divisible workload j=(σ, D) and a given number of compute nodes N, let α_i denote the fraction of load

assigned to the i^{th} machine, $0 < \alpha \leq 1$, $\sum_{i=1}^{N} \alpha = 1$, C_m is the cost to transmit a unit workload and C_{pi} is the cost to process a unit workload.

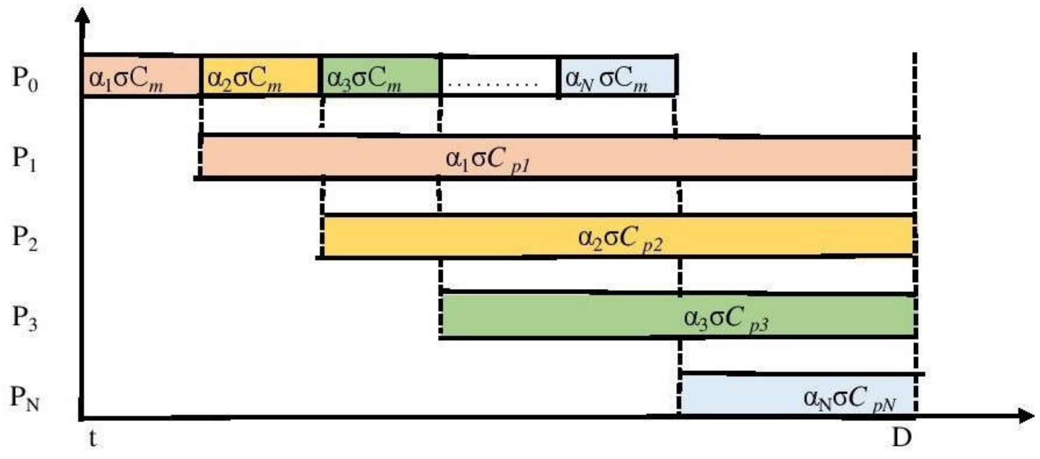

Fig.2. Task execution time diagram

Let t denote the time instant at which this occurs, Pi receives data over an interval $[\,t, t+\sigma\alpha_i C_m\,]$, and processes this data over the interval $[\,t+\sigma\alpha_i C_m\,,\,t+\sigma\alpha_i C_m+\sigma\alpha_i C_{pi}\,]$.
Since it must complete by the deadline D, we therefore have

$$t+\sigma\alpha_i C_m+\sigma\alpha_i C_{pi}\leq D \tag{1}$$

By the optimality principle, P_i and Pi+1 complete execution at the same time instant. The fraction of load to be allocated to each compute node is**:**

$$\alpha_i\leq\frac{(D-t)}{\sigma\left(C_m+C_{pi}\right)} \tag{2}$$

Total cost use is the energy cost C_i consumed by the CPU to process an entire load. Assuming a linear system, without loss of generality, this total cost is simply the sum of all individual link processor's cost discussed above.

The cost for executing j job for the i^{th} machine:

$$\alpha_i\times\sigma\times C_{pi}\times C_i \tag{3}$$

Thus, the total cost to execute j$^{\text{th}}$ job can be calculated by:

$$\zeta=\sum\left(\alpha_i\times\sigma\times C_{pi}\times C_i\right) \tag{4}$$

The pseudo-code of the algorithm is as listed below in Fig. 3.

```
Input: Job J
Output: $\alpha_i$
1:  Sort computing node according to strategy  /*worker selection strategy*/
2: bal ← 1 /*balance of load fraction to be allocated
3: $i$ ←1 /* determine of fraction load for $i$'th compute node*/
4: $t$←0
5:  **while** bal > 0 do
6:      **if** ($i$ > N) **then**
7:          declare failure /*all workers have been use up before entire load been assigned*/
8:      **end if**
9:      $\alpha_i \leftarrow \min \{bal, \frac{(D-t)}{\sigma(C_m + C_{pi})}\}$
10: bal ←bal - $\alpha_i$
11: t←t + $\sigma\alpha_i C_m$
12: $i$ ←$i$ + 1
13: **end while**
```

Fig.3. CARTDLT scheduling algorithm

4. EXPERIMENTAL RESULTS

This section describes the simulation setup used in this research.

4.1. Simulation Setup

In the simulation, a cloud system with computing nodes N = 16 and 32 is considered. For all divisible loads arriving at P_0, their loadsizes(σ) in the range [100,400] are uniformly generated, Deadline(D) is 1000 and C_m =1. The detailed price for each processing node is as in Table 1 below. MIPS stands for how many million machine instructions that a computer can execute in one second.

Table 1. Prices and processing speeds

Service No	Processing speed C_{pi} (MIPS)	Cost C_i ($/unit time)
1	10	3
2	20	6
4	30	12
5	40	24

4.2. Analysis of Results

To observe the impact of *worker selection strategy* and *chunk sizes* assigned for each of the workers, three sequence order strategy were examined. The total cost for each job with the varying loadsizes for the three workers' sequence order strategy are measured as below:

4.2.1. Most expensive to cheapest worker

The sequence of the worker is indexed in non-increasing order from fastest processors which having most expensive cost to slowest processors which having cheapest cost. Without loss of generality, let us assume that the processors are indexed according to non-increasing so that C_{pi} is $C_{pi} \geq Cp(i+1)$ for all i, $1 \leq i \leq N$. The experimental result is in Figure 4 and Figure 5.

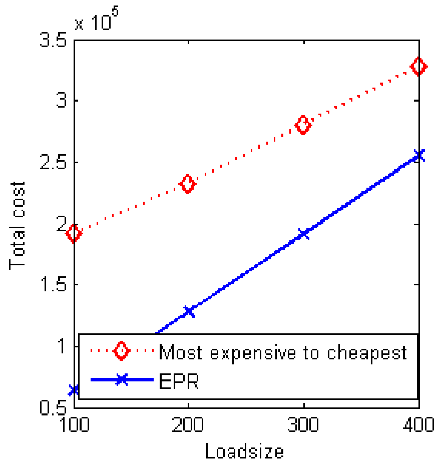

Fig.4. N =16

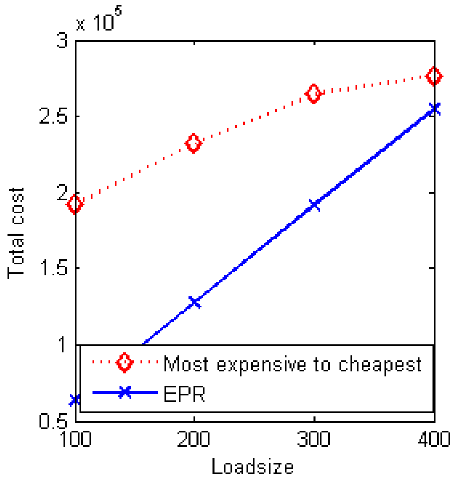

Fig.5. N =32

4.2.2. Cheapest to most expensive worker

The sequence of the worker is indexed in non-decreasing order from the worker with the lowest speed which is having cheaper cost, to the worker with the highest speed which is having most expensive cost so that C_{pi} is $C_{pi} \leq Cp(i+1)$ for all i, $1 \leq i \leq N$ The experimental result is in Figure 6 and Figure 7.

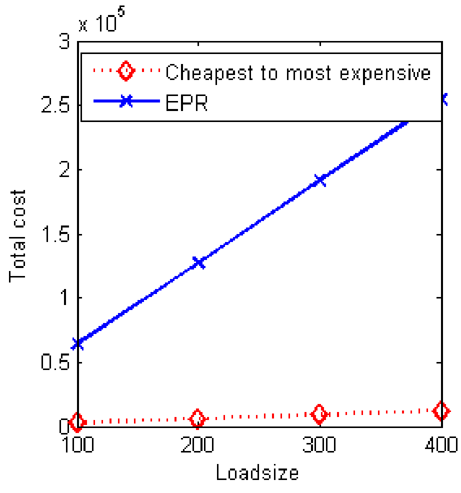

Fig.6. N =16

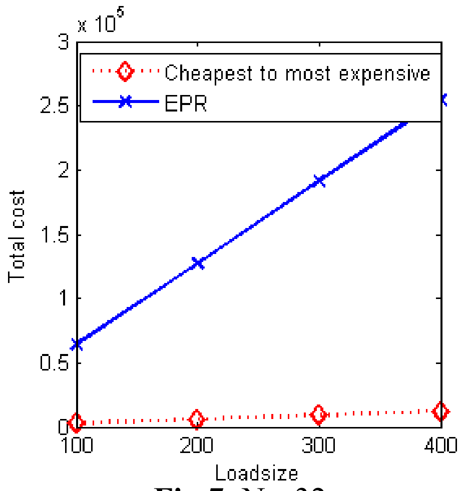

Fig.7. N =32

4.2.3. Random

The sequence of the worker is selected randomly. This is to study the effect if the worker selection strategy is not implemented. The experimental result is in Figure 8 and Figure 9.

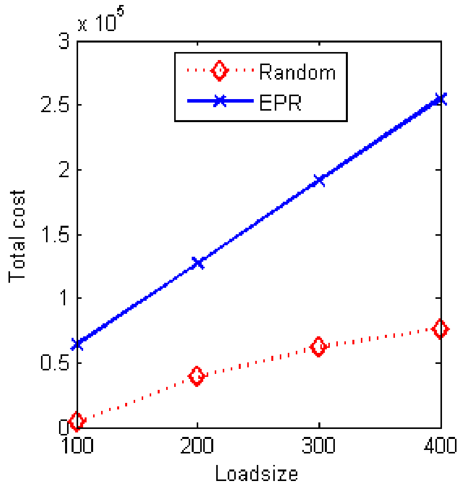

Fig.8. N=16

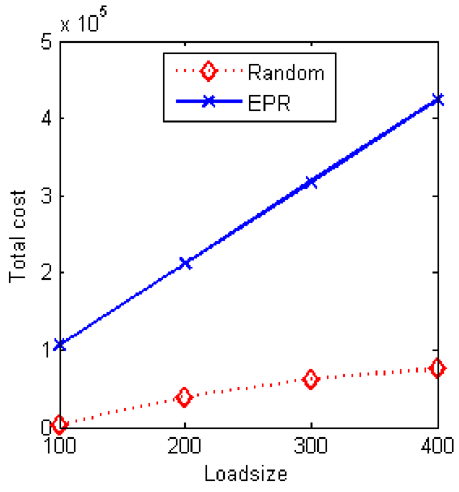

Fig.9. N =32

The work conducted in this research evaluated the merits of implementing worker selection strategy towards reducing total computation cost. It can be seen from the simulation results that sorting worker sequence in non-decreasing order, from lowest speed which is having cheapest cost to the worker with the highest speed which is having most expensive cost, has yield the lowest cost as compared with other strategies. The proposed CARTDLT scheduling algorithm allows a cloud provider to make the optimal decision in the amount of fraction load assigned to each of the workers, so as to maximize profit.

5. CONCLUSIONS

In this paper, the dual problem of scheduling to meet deadlines while minimizing cost using divisible load theory (DLT) for the provision of performance guarantees has been successfully addressed. The experimental results demonstrate the usefulness of the implementation of worker selection strategy towards reducing total computation cost.

6. ACKNOWLEDGMENT

This research was partially funded by the Research University Grant (GUP-12J85) of Universiti Teknologi Malaysia.

7. REFERENCES

1. Kang, S., B. Veeravalli, and K.M.M. Aung. Scheduling Multiple Divisible Loads in a Multi- cloud System. in 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing. 2014.

2. Abdullah, M. and M. Othman, Cost-based Multi-QoS Job Scheduling Using Divisible Load Theory in Cloud Computing. Procedia Computer Science, 2013. 18: p. 928-935.

3. Abdullah, M. and M. Othman, Scheduling Divisible Jobs to Optimize the Computation and Energy Costs. International Journal of Engineering Science Invention, 2015. 4(2).

4. Hu, M., J. Luo, and B. Veeravalli. Optimal provisioning for scheduling divisible loads with reserved cloud resources. in 2012 18th IEEE International Conference on Networks (ICON). 2012.

5. Kim, S.H., et al., A Science Gateway Cloud With Cost-Adaptive VM Management for Computational Science and Applications. IEEE Systems Journal, 2017. 11(1): p. 173-185.

6. Chuprat, S. Divisible load scheduling of real-time task on heterogeneous clusters. in 2010 International Symposium on Information Technology. 2010.

7. Chuprat, S. and S. Baruah. Real-Time Divisible Load Theory: Incorporating Computation Costs. in 2011 IEEE 17th International Conference on Embedded and Real-Time Computing Systems and Applications. 2011.

8. X. Lin, Y.L., J. Deogun, and S. Goddard. Real-time divisible load scheduling for

cluster computing. in Proceedings of the IEEE Real-Time Technology and Applications Symposium (RTAS). 2007. Bellevue, Washington.

9. Hu, M. and B. Veeravalli, Requirement-Aware Strategies with Arbitrary Processor Release Times for Scheduling Multiple Divisible Loads. IEEE Transactions on Parallel and Distributed Systems, 2011. 22(10): p. 1697-1704.

10. Bharadwaj, V., D. Ghose, and T.G. Robertazzi, Divisible Load Theory: A New Paradigm for Load Scheduling in Distributed Systems. Cluster Computing, 2003. 6(1): p. 7-17.

11. Pokahr, A. and L. Braubach, Elastic component-based applications in PaaS clouds. Concurrency and Computation: Practice and Experience, 2016. 28(4): p. 1368-1384.

12. Shakhlevich, N.V., Scheduling Divisible Loads to Optimize the Computation Time and Cost, in Economics of Grids, Clouds, Systems, and Services: 10th International Conference, GECON 2013, Zaragoza, Spain, September 18-20, 2013. Proceedings, J. Altmann, K. Vanmechelen, and O.F. Rana, Editors. 2013, Springer International Publishing: Cham. p. 138- 148.

13. Yuan, H., et al., CAWSAC: Cost-Aware Workload Scheduling and Admission Control for Distributed Cloud Data Centers. IEEE Transactions on Automation Science and Engineering, 2016. 13(2): p. 976-985.

How to cite this article:
Majid M L A, Chuprat S. Cost-aware real-time divisible loads scheduling in cloud computing. J. Fundam. Appl. Sci., 2018, *10(2S), 1-10.*