Adaptive Divisible Load Model for Scheduling Data-Intensive Grid Applications

M. Othman*, M. Abdullah, H. Ibrahim, and S. Subramaniam

Department of Communication Technology and Network,
Faculty of Computer Science and Information Technology
University Putra Malaysia, 43400 UPM Serdang, Selangor D.E., Malaysia
mothman@fsktm.upm.edu.my, monabdullah@hotmail.com

Abstract. In many data grid applications, data can be decomposed into multiple independent sub datasets and schedule for parallel execution and analysis. Divisible Load Theory (DLT) is a powerful tool for modelling data-intensive grid problems where both communication and computation load is partitionable. This paper presents an Adaptive DLT (ADLT) model for scheduling data-intensive grid applications. This model reduces the expected processing time approximately 80% for communication intensive applications and 60% for computation intensive applications compared to the previous DLT model. Experimental results show that this model can balance the loads efficiently.

Keywords: Divisible Load Theory, Data grid, Scheduling, Load Balancing.

1 Introduction

Grid systems are interconnected collections of heterogeneous and geographically distributed resources. In data grid environments, many large-scale scientific experiments and simulations generate very large amounts of data in the distributed storages, spanning thousands of files and data sets [8].

Scheduling an application in such environments is significantly complicated and challenging because of the heterogeneous nature of a Grid system. Grid scheduling is defined as the process of making scheduling decisions involving allocating jobs to resources over multiple administrative domains [5].

Recently, DLT model has emerged as a powerful tool for modelling data-intensive grid problems [2]. DLT exploits the parallelism of a divisible application which is continuously divisible into parts of arbitrary size, by scheduling the loads in a single source onto multiple computing resources [3].

The load scheduling in Data Grid is addressed using DLT model with additional constraint that is each worker node receives the same load fraction from each data source [3]. However, this does not take into account the communication time and assumes that the communication time is faster than the computation time. Whereas, if we want to achieve high performance, we must consider both communication time and computation time [6,7].

* Institute of Mathematical Science Research (INSPEM), University Putra Malaysia.

In [4], the Constrained DLT (CDLT) is used for scheduling decomposable data-intensive applications. It is compared with results of Genetic Algorithm (GA). The same constraint was tested, which was suggested in [3] and each worker node receives the same load fraction from each data source. They considered the communication time but not in dividing the load. Firstly, they divided the load using DLT model then added the communication time to the expected processing time.

In this paper, an adaptive divisible load model which takes into account both computation time and communication time is proposed. The main objective of our model is to distribute loads over sites to achieve the desired performance level for large jobs - common in Data Grid applications such as Compact Muon Solenoid (CMS) experiment, see [1].

2 Scheduling Model

In [5], the target data intensive applications model can be decomposed into multiple independent subtasks and executed in parallel across multiple sites without any interaction among sub tasks. Lets consider job decomposition by decomposing input data objects into multiple smaller data objects of arbitrary size and processing them on multiple virtual sites. For example in theory, the High Energy Physic (HEP) jobs are arbitrarily divisible at event granularity and intermediate data product processing granularity [1]. Assume that a job requires a very large logical input data set (D) consists of N physical datasets and each physical dataset (of size L_k) resides at a data source (DS_k , for all $k = 1, 2, \dots, N$) of a particular site. Fig 1 shows how the logical input data (D) is decomposed onto networks and their computing resources.

The scheduling problem is to decompose D into datasets (D_i for all $i = 1, 2, \dots, M$) across M virtual sites in a Virtual Organization (VO) given its initial physical decomposition. Again, we assume that the decomposed data can be analyzed on any site.

2.1 Notations an Definitions

M	The total number of nodes in the system
N	The total number of data files in the system
L_i	The loads in data file i
L_{ij}	The loads that node j will receive from data file i
L	The sum of loads in the system, where $L = \sum_{i=1}^N L_i$
α_{ij}	The amount of load that node j will receive from data file i
α_j	The fraction of L that node j will receive from all data file
w_j	The inverse of the computing speed of node j
Z_{ij}	The link between node i and data source j
T_{cp}	The computing intensity constant.
$T(i)$	The processing time in node i .

Fig. 1. Data decomposition and their processing

2.2 Cost Model

The execution time cost (T_i) of a subtask allocated to the site i and the turn around time ($T_{Turn_Around_Time}$) of a job J can be expressed as follows

$$T_i = T_{input_cm}(i) + T_{cp}(i) + T_{output_cm}(i, d)$$

and

$$T_{Turn_Around_Time} = \max_{i=1}^M \{T_i\},$$

respectively. The input data transfer ($T_{input_cm}(i)$), computation ($T_{cp}(i)$), and output data transfer to the client at the destination site d ($T_{output_cm}(i, d)$) are presented as a $\max_{k=1}^N \{l_{ki} \cdot \frac{1}{Z_{ki}}\}$, $d_i \cdot w_i \cdot ccRatio$ and $f(d_i) \cdot Z_{id}$, respectively. The function $f(d_i)$ is an output data size and $ccRatio$ is the non-zero ratio of computation and communication. The turn around time of an application is the maximum among all the execution times of the sub tasks.

The problem of scheduling a divisible job onto M sites can be stated as deciding the portion of original workload (D) to be allocated to each site, that is, finding a distribution of l_{ki} which minimizes the turn around time of a job. The proposed model uses this cost model when evaluating solutions at each generation.

3 Adaptive Scheduling Model

In all the literature related to the divisible load scheduling domain so far, an optimality criterion [7] is used to derive an optimal solution is as follows.

It states that in order to obtain an optimal processing time, it is necessary and sufficient that all the sites that participate in the computation must stop at the same time. Otherwise, load could be redistributed to improve the processing time. This optimality principle in the design of our load distribution strategy is used. The new proposed model is an improvement of DLT model, see [3]. The communication time fraction is added into the new model.

3.1 Computation Time Fraction

In [3], the authors have proposed the computation time fraction successfully. The optimal schedule is all nodes finish computation at the same moment of time. Based on that, we will have

$$\sum_{i=1}^N \alpha_{i,j} w_j T_{cp} = \sum_{i=1}^N \alpha_{i,j+1} w_{j+1} T_{cp}, \quad j = 1, 2, \dots, M-1. \quad (1)$$

As our objective is to determine the above optimal fractions $\alpha_{i,j}$, we impose the following condition in our strategy. Let $\alpha_{i,j} = \alpha_j L_i$, for all $i = 1, 2, \dots, N$ and $j = 1, 2, \dots, M$. This condition essentially assumes that each node receives a load that is proportional to the size of the load from the source. Moreover, each node receives the same load fraction (percentage of total load) from each source. Without this condition, the system of equation is under constrained, and additional constraints need to be added for a unique solution. With this condition, Eq. (1) can be derived as,

$$\sum_{i=1}^N \alpha_j L_i w_j = \sum_{i=1}^N \alpha_{j+1} L_i w_{j+1}, \quad j = 1, 2, \dots, M-1$$

$$\alpha_j w_j = \alpha_{j+1} w_{j+1}, \quad j = 1, 2, \dots, M-1 \quad (2)$$

$$\alpha_x w_x = \alpha_j w_j, \quad x = j+1, \quad j = 1, 2, \dots, M-1 \quad (3)$$

$$\sum_{x=1}^M \alpha_x = \sum_{x=1}^M \frac{\alpha_j w_j}{w_x}, \quad j = 1, 2, \dots, M \quad (4)$$

Using Eq. (4) together with $\sum_{i=1}^M \alpha_i = 1$, thus will leads to,

$$\sum_{x=1}^M \frac{\alpha_j w_j}{w_x} = 1, \quad j = 1, 2, \dots, M \quad (5)$$

$$\alpha_j = \frac{1}{w_j (\sum_{x=1}^M \frac{1}{w_x})} \quad (6)$$

Hence, the fraction of load that should be given by data source i to node j is

$$\alpha_{i,j} = \frac{1}{w_j (\sum_{x=1}^M \frac{1}{w_x})} L_i. \quad (7)$$

3.2 Communication Time Fraction

Scheduling has to consider the bandwidth availability of transfer between computational nodes to which a job is going to be submitted and the storage resource(s) from which the data required is to be retrieved.

In DLT, it is assumed that computation and communication loads can be partitioned arbitrarily among a number of processors and links, respectively. By considering communication time (Z) instead of computation time (w) in Eq. (3), we have

$$\alpha_x Z_{x,y} = \alpha_j Z_{i,j}, \quad x = j = 1, 2, \dots, M. \quad (8)$$

As mentioned in section 3.1, all links must stop transmitting the fraction of load α_i at the same time. Eq. (8) shows that, it is true for all links and load fractions, in order to obtain the optimality. In order to derive the $\alpha_{i,j}$ from communication time, we have to compute the summation of all links at first as,

$$\sum_{i=1}^N \sum_{j=1}^M \frac{1}{Z_{i,j}} \quad (9)$$

From Eqs. (6) and (9), thus leads to

$$\alpha_j = \frac{1}{Z_{i,j} \sum_{x=1}^N \sum_{y=1}^M \frac{1}{Z_{x,y}}}. \quad (10)$$

3.3 The Final Form

After getting the computation time fraction and communication time fraction in (6) and (10) respectively, the final fraction will be given as

$$CM_{i,j} = \frac{1}{w_j (\sum_{x=1}^M \frac{1}{w_x})} + \frac{1}{Z_{i,j} \sum_{x=1}^N \sum_{y=1}^M \frac{1}{Z_{x,y}}} \quad (11)$$

$$\alpha_j = \frac{CM_{i,j}}{\sum_{i=1}^N \sum_{j=1}^M CM_{i,j}} \quad (12)$$

$$\alpha_{i,j} = \frac{CM_{i,j}}{\sum_{i=1}^N \sum_{j=1}^M CM_{i,j}} L_i. \quad (13)$$

4 Numerical Experiments

To measure the performance of the proposed ADLT model against the previous model, randomly generated experimental configurations were used, see [4]. The network bandwidth between sites is uniformly distributed between 1Mbps and 10Mbps. The location of n data sources (DS_k) is randomly selected and each physical dataset size (L_k) is randomly selected with a uniform distribution in the range

of 1GB to 1TB. We assume that the computing time spent in a site i to process a unit dataset of size 1MB is uniformly distributed in the range $1/r_{cb}$ to $10/r_{cb}$ seconds where r_{cb} is the ratio of computation speed to communication speed.

Example, consider a system with one data source and two worker nodes with parameters $w_0 = 0.1$ and $w_1 = 0.2$. Let the size of data source is 100MB. The bandwidth among links $Z_{0,0}=3\text{Mbps}$ and $Z_{0,1}=5\text{Mbps}$. By using the model in [5] and proposed model (see Eq. (13)), we have the expected processing time as shown in Table 1.

Table 1. Expected Processing Time for the CDLT and ADLT Models

	CDLT	ADLT
$T(0)$	28.86	22.2
$T(1)$	13.32	19.2
$Max\{T(1), T(0)\}$	28.86	22.2

Based on CDLT model, the expected processing time $T(0)$ (28.86 secs) is more than $T(1)$ (13.32 secs). In other words, the load is not divided equally. With the proposed ADLT model, it has better load balancing. Both node 0 and node 1 have nearly same load and the expected processing time is 22.2 secs. Thus, indicated that the proposed ADLT model performs better in load balancing.

Table 2. Expected Processing Time for the CDLT and ADLT Models

$ccRatio$	$CDLT$	$ADLT$
0.001	29013	5385
0.01	29014	5385
0.1	29018	5687
1	29059	5394
10	29468	5470
100	33563	6646
1000	74507	22091
10000	483954	198496

Table 2 shows the experimental results for the two models. We examined the overall performance of each model by running them under 100 randomly generated Grid configurations. We varied the parameters: $ccRatio$ (0.001 to 10000), M (3 to 20), N (2 to 20) and r_{cb} (10 to 500). From the above values, the expected processing time of the proposed model seems to be the better one. It is clear that, our model balances the loads among the nodes more efficiently. It should be noted that as the $ccRatio$ is less than 10, the differences of the expected processing time of the two models are near to be same due to the effect of $ccRatio$ in cost model. This model reduced the expected processing time by 80% for communication intensive applications and by 60% for computation intensive applications on an average compared to previous DLT model, as shown in Fig 2.

Fig. 2. Expected processing time for the CDLT and ADLT Models

Fig. 3. The impact of output data size to input data size (a) $oiRatio = 0$: No output or small size of output (b) $oiRatio > 0.5$

The impact of the ratio of output data size to input data size is shown in Fig 3. ADLT model performs well for communication intensive applications that generate small output data compared to input data size (low $oiRatio$). For computation intensive applications, the ratio of output data size to input data size does not affect the performance of the algorithms much unless when $ccRatio$ is 10000.

5 Conclusion

The problem of scheduling data-intensive loads on grid platforms is addressed. We used the divisible load paradigm to derive closed-form solutions for

processing time considering communication time. In the proposed model, the optimality principle was utilized to ensure an optimal solution. The experiment results of the proposed model ADLT show better performance as compared to the CDLT model in terms of expected processing time and load balancing.

References

1. Holtman K., *et al.*: CMS Requirements for the Grid, In proceeding of the International Conference on Computing in High Energy and Nuclear Physics, Science Press, Beijing China, (2001).
2. Robertazzi T.G.: Ten Reasons to Use Divisible Load Theory, IEEE Computer, **36**(5) (2003) 63-68.
3. Wong H. M., Veeravalli B., Dantong Y., and Robertazzi T. G.: Data Intensive Grid Scheduling: Multiple Sources with Capacity Constraints, In proceeding of the IASTED Conference on Parallel and Distributed Computing and Systems, Marina del Rey, USA, (2003).
4. Kim S., Weissman J. B.: A Genetic Algorithm Based Approach for Scheduling Decomposable Data Grid Applications, In proceeding of the International Conference on Parallel Processing, IEEE Computer Society Press, Washington DC USA, (2004).
5. Venugopal S., Buyya R., Ramamohanarao, K.: A Taxonomy of Data Grids for Distributed Data Sharing, Management and Processing, ACM Computing Surveys, **38**(1) (2006) 1-53.
6. Mequanint, M.: Modeling and Performance Analysis of Arbitrarily Divisible Loads for Sensor and Grid Networks, PhD Thesis, Dept. Electrical and Computer Engineering, Stony Brook University, NY 11794, USA, (2005).
7. Bharadwaj, V., Ghose D., Robertazzi T. G.: Divisible Load Theory: A New Paradigm for Load Scheduling in Distributed Systems, Cluster Computing, **6**, (2003) 7-17.
8. Jaechun N., Hyoungwoo, P.: GEDAS: A Data Management System for Data Grid Environments. In Sunderam *et al.* (Eds.): Computational Science. Lecture Notes in Computer Science, Vol. 3514. Springer-Verlag, Berlin Heidelberg New York (2005) 485-492.