Probabilistic Approach to Scheduling Divisible Load on Network of Processors

Manar Arafat1, Sameer Bataineh2* and Issa Khalil1

1Department of Computer Science, An-Najah National University, Nablus, Palestine
2Faculty of Computer and Information Technology, Jordan University of Science and Technology, Jordan
3Qatar Foundation, Qatar

Abstract

Divisible Load Theory (DLT) is a very efficient tool to schedule arbitrarily divisible load on a set of network processors. Most of previous work using DLT assumes that the processors’ speeds and links’ speeds are time-invariant. Closed form solution was derived for the system under the assumption that the processors’ speed s and the links’ speeds stay the same during the task execution time. This assumption is not practical as most of distributed systems used today have an autonomous control. In this paper we consider a distributed system (Grid) where the availability of the processors varies and follows a certain distribution function. A closed form solution for the finish time is derived. The solution considers all system parameters such as links’ speed, number of processors, number of resources (sites), and availability of the processors and how much of power they can contribute. The result is shown and it measures the variation of execution time against the availability of processors.

Keywords: Divisible load theory; Scheduling; Availability; Distributed systems

Definitions and Notations

\[ N: \text{The number of available data sources (sites)} \]
\[ M: \text{The number of available nodes to process the grid tasks} \]
\[ S_i: \text{Data source (site). } i=1\ldots N \]
\[ P_i: \text{Available nodes to process the grid tasks. } i=1\ldots M \]
\[ L: \text{Total load at the originating load node} \]
\[ c_i: \text{Computational speed of node } i \text{ in the network} \]
\[ w_i: \text{A constant that is inversely proportional to the computational speed of node } i\text{.} \]
\[ z_{ij}: \text{A constant that is inversely proportional to the link speed between site } i \text{ and node } j. \]
\[ f_i: \text{The fraction of the load } L \text{ assigned to data source } i=1\ldots N \]
\[ L_i: \text{The load assigned to data source } i=1\ldots N \]
\[ a_{ij}: \text{The fraction of load assigned from site } i \text{ to node } j. \]
\[ T_{ip}: \text{The time it takes the } i^{th} \text{ node to process the entire load when } w_i=1. \]
\[ T_i: \text{The time can be transmitted over the } i^{th} \text{ link in time } z_{iT_{cm}}. \]
\[ T_{cm}: \text{The total communication power in the system.} \]
\[ C_i: \text{The equivalent communication power for site } i, (i=1,2,\ldots N). \]
\[ C_j: \text{The total communication power in the system.} \]

Introduction

Geographically distributed heterogeneous systems become very powerful and popular in the past decade. Good examples of such systems are Clusters, Grids and Clouds. Grid can be viewed as a distributed large-scale cluster computing. From another perspective, it constitutes the major part of Cloud Computing Systems in addition to thin clients and utility computing [1-4]. Hence, Grid computing has attracted many researchers [5]. The interest in Grid computing has gone beyond the paradigm of traditional Grid computing to a Wireless Grid computing [6].

There are many attempts to find an analytical solution for scheduling load on the nodes (processors) on those systems. Queuing theory is a very famous tool participated to find analytical solution on such system [7,8]. Divisible Load Theory (DLT), deterministic in nature, was also used and proved that it is very much the same as Markov Chain Modeling [9]. However, DLT has shown that it is an excellent tool to schedule independent jobs on a Grid originated from multiple resources [7,10-14].

As Scheduling plays an important role in determining the performance of Grids, there are many algorithms in literature that discuss the scheduling on Grids [15-23]. It was shown that finding an
analytical solution for general scheduling problem in a Grid is a very difficult task [24].

There are several attempts to use the DLT to model scheduling arbitrarily divisible load on the Grid [25-27]. However, communication time is rarely considered [11]. In [7], communication time is studied but not in dividing the load, so the transfer input time of the load was not part of the model. Though all parameters of the system were considered in [10], the paper did not provide a closed form analytical solution for the finish time.

In our previous work in [28], we managed to come up with closed form solution for the minimum finish time of executing an arbitrarily divisible application on the Grid taking into consideration the communication time and the computation time simultaneously. In [28], we assumed that the nodes are always available to execute the grid task. In distributed environments, this assumption is unrealistic. A node may not be willing to contribute its whole computing power during the time span of a grid task execution.

The objective of this paper is to develop an analytical model to distribute the grid load from multiple sites to all nodes in the Grid such that the load is executed in a minimum time. The model deals with the dynamic availability of each node to serve the Grid tasks. We will derive closed form solutions for the load fraction of each node, and the minimum expected finish time of the total load. The solution considers all system parameters such as the links’ speed, number of processors, number of resources (sites), and availability of the processors and how much of their power they can contribute.

The rest of the paper is organized as follows: In section 5 we present the model of the system, in section 3 we present the notations used throughout the paper. The system equations are in section 6. The results and discussion are in section 7. Finally the concluding remarks are in section 8.

System Equations

Using the Equivalent Processor and Communication Link Concept, which was first introduced in [29], we can replace the set of M nodes in Figure 1 with a single node, which has an equivalent computing power. In other words, if the M nodes are all removed and replaced by the equivalent node, the performance of the system will be exactly the same. We can also replace the M links from each data source by one link which has an equivalent communication power $C_{ij}$ ($i=1,2, N$) as depicted in Figure 2. The value of $C_i$ is given by:

$$c_i = \frac{N}{\sum_{j=1}^{M} \frac{1}{Z_{ij}}} \quad i=1 \ldots N$$

(1)

The total communication power in the system is

$$c_T = \frac{1}{\sum_{j=1}^{M} \frac{1}{Z_{ij}}} = \frac{1}{\sum_{j=1}^{M} \frac{1}{Z_{ij}}} = \sum_{i=1}^{N} C_i$$

(2)

The distribution of the total load $L$ from the originating node to the N sites depends on the links speed to the equivalent node. Each site $S_i$ will be assigned a fraction that depends on the equivalent communication power $C_{ij}$ ($i=1,2, N$) and the total communication power in the system $C_T$. Consequently, the fraction of the load to be assigned to site $i$ is $f_i = \frac{C_{ij}}{c_T}$ for $i=1 \ldots N$, such that $\sum_{i=1}^{N} f_i = 1$. It follows that the share of the load $L_i$ to be assigned to each site $i=1 \ldots N$ is given by

$$L_i = f_i L$$

(3)

System Model

The system has three types of nodes: originating load node, N data sources (or sites) $S_i$, $i=1, \ldots, N$, and M available nodes $P_j$, $j=1, \ldots, M$ to process the grid tasks. Nodes have different speeds $w_j$, $j=1,2, \ldots, M$. The links that connect the data sources with the nodes have also different speeds $z_i$, $i=1,2, \ldots, N$, $j=1,2, \ldots, M$.

The system works as shown in Figure 1. The originating node receives the total load $L$ and distributes it to $N$ available sites. The fraction of the load to be assigned to each site is $f_i$ for $i=1 \ldots N$. It follows that the share of the load $L_i$ to be assigned to data source $i$ is given by $L_i = f_i L$, $i=1 \ldots N$. Then, each data source will distribute its load $L_i$ to the M available nodes for processing. The fraction of the load to be assigned to each node $j$ from site $i$ is $a_{ij}$, $i=1,2, \ldots, N$) $\phi$, $j=1,2, \ldots, M$).

In section 6, first we obtain the fraction of load that has to be assigned to each site from the load originating node. Each site will be assigned a fraction that depends on the speed of the links in the network. Second, we obtain the fraction of load that has to be assigned from each site to each of the available nodes in the network. The fractions depend on the speed of the links, the speed of the nodes and the dynamic availability of each node in the grid.

Our analytical model guarantees that the total load is executed in a minimum time. We derive closed form solution for the minimum expected finish time of the total load.
Next, we obtain the fraction of load that has to be assigned from each site to each of the available nodes in the network. The assumption that computational resources at nodes are dedicated to Grid tasks is impractical. Each of the M nodes may not be willing to contribute its whole computing power during the time span of a Grid task execution. Each node \( P_j (j=1\ldots M) \) will be assigned a load fraction that depends on the speed of the links, the speed of the nodes and the dynamic availability of each node in the grid. So it is not possible to take advantage of the equivalent power concept at this stage.

In Figure 3, each data source has the illusion as if it is the only data source that is distributing its load to the M available nodes. In other words, we can view the system as N multiples of a single data source as shown in Figure 4.

In Figure 4, each site will distribute its load to M available nodes such that the total load is executed in a minimum time. The solution is based on the optimality principle [30]. Optimality solution assumes that all processors finish at the same time. It is analytically proved that a minimal solution time is achieved when the computation by each node finishes at the same time [31]. Intuitively, this is because otherwise, the processing time could be reduced by transferring some fractions of load from busy nodes to idle nodes [32-36].

We now derive closed form solution for the load fraction of each node, and the minimum expected finish time of the total load. The solution is based on the optimality principle [37]. Optimality solution assumes that all processors finish at the same time.

In general, the time to complete execution of \( \alpha_{ij} \) on node \( j \) consists of the communication time \( c_{ij}T \) and processing time \( p_{ij}T \) divided by the probability of finding node \( j \) available during \( T\).

\[
T_j = T_0 + \frac{T_j}{P_j(t)} \quad \text{where } i=1,2,\ldots, N; j=1,2,\ldots,M
\]

(4)

\[
T_j = \alpha_j Z_j T_m + \frac{w_j T_j}{P_j(t)}
\]

(5)

Let \( i=1 \). Then the system equations are:

\[
T_j = \alpha_1 (Z_1 T_m) + \frac{w_1 T_j}{P_j(t)} \quad j=1,2,\ldots,M
\]

(6)

Applying the optimality criterion that all processors should stop computing at the same time [31,32]:

Where \( T_{(j)}, T_{(j+1)} \) where \( j=1,2,\ldots,M-1 \)

(7)

From equation (7), we obtain equations (8), (9), and (10) respectively.

\[
\alpha_{12} = \alpha_{11} \frac{P_1(t)}{P_j(t)} \frac{(Z_1 T_m P_1(t) + w_1 T_p)}{(Z_1 T_m P_1(t) + w_1 T_p)}
\]

(8)

\[
\alpha_{1j} = \alpha_{11} \frac{P_1(t)}{P_j(t)} \frac{(Z_1 T_m P_1(t) + w_1 T_p)}{(Z_1 T_m P_1(t) + w_1 T_p)}
\]

(9)

\[
\alpha_{(M-1)j} = \alpha_{11} \frac{P_1(t)}{P_j(t)} \frac{(Z_1 T_m P_1(t) + w_1 T_p)}{(Z_1 T_m P_1(t) + w_1 T_p)}
\]

(10)

Now we can write all the above equations as a function of \( \alpha_j \) and the parameters of the Grid as follows:

\[
\alpha_{12} = \alpha_{11} \frac{P_1(t)}{P_j(t)} \frac{(Z_1 T_m P_1(t) + w_1 T_p)}{(Z_1 T_m P_1(t) + w_1 T_p)}
\]

(11)

\[
\alpha_{13} = \alpha_{11} \frac{P_1(t)}{P_j(t)} \frac{(Z_1 T_m P_1(t) + w_1 T_p)}{(Z_1 T_m P_1(t) + w_1 T_p)}
\]

(12)

In general,

\[
\alpha_{1j} = \alpha_{11} \frac{P_1(t)}{P_j(t)} \frac{(Z_1 T_m P_1(t) + w_1 T_p)}{(Z_1 T_m P_1(t) + w_1 T_p)} \quad \text{where } j=1,2,\ldots,M
\]

(13)

Obviously, the summation of all fractions of the load must equal 1.
In the following we use the equations derived in the paper to study the effect of different parameters on the system performance. The results are compared with the deterministic case. Deterministic results are those obtained with (t)=1.

In Figures 5 and 6, we assume that the M available nodes in the network will have the same probability of availability to serve the grid loads (0<probability estimate ≤ 1). Figure 5 relates the expected finish time to the estimate of probability, for M=25, 100, 150 and 200, assuming that \( z_{j}=1 \) and \( w_{j}=1 \) (for \( i=1...N \) and \( j=1...M \)), and \( T_{cp}=T_{cm}=1 \).

Figure 5 shows that as the number of available nodes in the grid increases, the expected finish time will decrease. For our parameters, there will be no significant improvement after \( M=150 \). The minimum expected finish time occurs when all the nodes are available to serve the grid loads. Comparing the results with the deterministic case, when the nodes devoted all power to one job, in other words when the probability estimate =1, we notice that the expected finish time could be 5 times slower when the system has a large number of processors.

Figure 6 relates the expected finish time to the estimate of probability for \( M=100 \) and \( z_{j}=0.1, 0.5, 1, 1.5 \) (for \( i=1...N \) and \( j=1...M \)), assuming that \( w_{j}=1 \) (for \( j=1...M \)), and \( T_{cp}=T_{cm}=1 \).

Figure 6 shows that as each link speed increases, the expected finish time will decrease.

Figure 7 relates the estimate of probability for node \( k \), (\( k \): may be any node \( 1...M \)), to its load fraction \( a_{ik} \) assigned from load source \( i \), for \( M=25, 50, 100 \) and 150. Assuming that the estimate of probability of each other available node is equal to one and that \( z_{j}=1 \) and \( w_{j}=1 \) (for \( i=1...N \) and \( j=1...M \)), and \( T_{cp}=T_{cm}=1 \).

From Figure 7, nodes that are available for serving the grid loads
most of the time will be assigned larger load fractions. This is exactly what is expected because we assume that all processors must stop at the same time to abide with the optimality principle.

### Conclusion

Unlike all previous work, which using DLT, in this paper we propose a model where the processors speeds is a function of number of jobs that a node in the distributed system is in charge at time ‘t’. We assumed that the availability of a node is based on certain probability and it varies with time. Of course, the probabilistic function should be derived from a certain realistic distribution.

The load is found in N sites and has to be distributed and assigned to M nodes such that total load is executed in a minimum time. We derived closed form solution for the load fraction of each node, and the minimum expected finish time of the total load. The result is shown and it measures the variation of execution time against the availability of processors for different system parameters. Our next step is to find a distribution function that can apt the system behavior. This will generate an accurate estimate for the probabilities to precisely reflect the availability of the processors in the system.

### References

1. Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, et al. (2009) Above the Clouds: A Berkeley View of Cloud Computing. Technical Report No. U CB/EECS-2009-28.
2. Nandaagankar SV, Raut AB (2014) A Comprehensive Study on Cloud Computing. International Journal of Computer Science and Mobile Computing 3: 733-738.
3. Kathrine GJW, Ilaghi MU (2012) Job Scheduling Algorithms in Grid Computing - Survey. International Journal of Engineering Research & Technology 1: 7.
4. Qureshi MB, Dehnavi MM, Allah NM, Qureshi MS, Hussain H, et al. (2014) Survey on Grid Resource Allocation Mechanisms. Journal of Grid Computing 12: 399-415.
5. Toma I, Iqbal K, Roman D, Strang T, Fensel D, et al. (2007) Discovery in grid and web services environments: A survey and evaluation. Multagent and Grid Systems 3: 341-352.
6. Manvi SS, Birje MN (2009) Wireless Grid Computing: A Survey. IETE Journal of Education 50: 119-131.
7. Byuna EJ, Choja SJ, Baikb MS (2007) MJSA Markov job scheduler based on availability in desktop grid. Future Generation Computer Systems 23: 616-622.
8. Tian G, Xiao C, Xu X, Gao1 CQ, Nuslati, et al. (2010) Grid Workflow Scheduling Based on Time Prediction of Queuing Theory. Proceedings of the 2010 IEEE, International Conference on Information and Automation pp: 36-39.
9. Moges MA, Robertazzi TG (2009) Grid Scheduling Divisible Loads from Two Sources. Computers and Mathematics with Applications 58: 1081-1092.
10. Abdullah M, Othman M, Ibrahim H, Subramaniam S (2009) Closed form Solution for Scheduling Arbitrarily Divisible Load Model in Data Grid Applications: Multiple Sources. American Journal of Applied Sciences 6: 626-630.
11. Moges M, Robertazzi TG (2003) Optimal Divisible Load Scheduling and Markov Chain Models. Proceedings of the 2003 Conference on Information Sciences and Systems, The Johns Hopkins University, Baltimore, MD, USA.
12. Jia J, Veeravalli B (2010) Scheduling Multisource Divisible Loads on Arbitrary Networks. IEEE Transactions On Parallel And Distributed Systems 21: 520-531.
13. Li X, Veeravalli B (2010) PPDD: scheduling multi-site divisible loads in single-level tree networks. Cluster Computing 13: 31-46.
14. Robertazzi TG, Yu D (2006) Multi-source grid scheduling for divisible loads. In: Proceedings of the 40th Annual Conference on Information Sciences and Systems (CISS’06) pp: 188-191.
15. Moise D, Moise I, Pop F, Cristea V (2008) Resource Co-Allocation for Scheduling Tasks with Dependencies, in Grid. The Second International Workshop on High Performance in Grid Middleware HiPerGRID.