Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 3, Issue, 7, pp.181-185, July, 2011

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

AN ECONOMIC MODEL FOR RESOURCE ALLOCATION AND SCHEDULING FOR COMPUTATIONAL GRIDS

G. Murugesan* and Dr. C. Chellappan

Department of Computer Science and Engineering, Anna University, Chennai

ARTICLE INFO

Article History:
Received 2nd April, 2011
Received in revised form
7th May, 2011
Accepted 19th June, 2011
Published online 16th July 2011

Key words:
Computational Grids,
Economy Model,
Resource Allocation,
Grid Scheduling.

ABSTRACT

Computational grids are distributed systems composed of heterogeneous computing resources which are distributed geographically and administratively. These highly scalable systems are designed to meet the large computational demands of many users from scientific and business orientations. The allocation of distributed resources to user application is one of the most challenging tasks in Grid Computing paradigm. The problem of allocating resources in scheduling requires the coordinated access to resources managed by autonomous entities. The resource owners of these resources have different usage or access policies and cost models, and varying loads as well as availability of resources. In order to address complex resource management issues, we have proposed a computational economy framework for resource allocation for optimally assigning jobs to resources to achieve a business objective. This framework provides mechanism for optimizing resource provider and consumer objective functions through linear programming principles. In the Grid computing arena, however, such scheduling has mostly been done from the perspective of maximizing the utilization of hardware resources. In this paper, we propose a mathematical approach as an economical model for Grid resource allocation with the objective of minimize the cost of grid resource users' under the availability of resources and budget as constraints.

INTRODUCTION

The basic unit in a computational grid is the processor, which is typically housed in a single or multi-processor worker node. These nodes are commonly joined at institutional level to form clusters, often running a variant of the Unix/Linux or windows operating systems. The clusters are then joined at the national or international level using middleware, a software suite of protocols and tools to form a computational grid. Computational grids are alternatively referred as compute grids, multi-clusters or utility grids which are commercial variants. Computational grids consist of large number of services and they are classified in to four categories. They are job management, data management, information systems and security. The job management services are the most visible service provided by a computational grid. It consists of two interfaces: grid interface and cluster interface. The major component in a grid interface is grid resource broker. The resource broker is responsible for grid resource management such as identifying the resources with respect to the user application; match the task to resources etc. The function of cluster interface is to relay jobs between the grid interface and the clusters. The data management is responsible for managing the data used by the computational grid. Managing data is in the form of secure, reliable and fast transmission of data as well as indexing data locations and metadata in catalogs. The information system service is responsible for update the resource status, availability of resources, capacity and functionality of the resources. Also this service is responsible for accounting, auditing and functionality testing purposes.

One of the most critical components of the interface in a computational grid is the resource broker. The broker is responsible for the allocation of grid resources to the resource requester in the form of jobs or tasks. Each task specifies its budget, deadline and others. The process of selecting a set of tasks to the available resources relies on the availability of the information about the task and the resources. The information required to schedule the tasks are requirements, submission time, and priority, estimated running time, and so on. The required information about the scheduler becomes availability, capacity, queue length, estimated wait time, architecture, performance, operating system and so on. In this paper we are focusing the role of grid broker mainly on resource allocation. We have developed a mathematical model to identifying the best resource for a set of jobs or tasks.

Corresponding author: murugesh02@gmail.com

The main goal of this paper is to analysis the different aspect of resource allocation model in the form of divisible load scheduling. The considered problems will be presented in the form of mathematical model. There are four aspect of the resource allocation problem will be taken into consideration. They are

- Single non-divisible load: It is assumed in this case that there is only one non-divisible load which is sent to a processor at most one message. This is not come under the category of grid computing arena. So we are not discussing this category in detail.
- Single divisible load: In this problem it is assumed that the load is sent to the processors in many portions of loads instead of big one. When an entire large amount of load is distributed the time spent waiting for the load is longer. Thus the load distribution scheme with multiple portions of loads has been an advantage in activating processor earlier.
- Multiple non-divisible loads: In this problem it is assumed that the multiple divisible loads are processed on the same computing facility. The loads are scheduling entities independent of each other. The loads from requester are submitted in to a appropriate processor with respect to the requirements of the requester. This means that a task send at most one message to a processor.
- Multiple divisible loads: In this problem it is assumed that there is more than one grid user and all of their loads are divisible loads and which are processed in the form of multiple portions in different processor. This means that a task sends at most one message to each of the processor those who are participating in the schedule. The difficulty of this problem rests on the coordination of computations and communications of many divisible loads.

There is no problem with single non-divisible load, because only one grid user and the workload are also not divisible. This paper is mainly focusing on the second case of Single divisible load. The rest of the paper is organized as follows: the first section focusing on the problem formulation for the divisible load scheduling in the heterogeneous environment. The next section proposes the mathematical model to divide the total workload in to smaller portions with respect to the resource availability and the budget allotted for completion of the workload process. The next section deals with the Computational Experiments of the solution received by solving the mathematical model with some arbitrary values. The next section focuses the related proposed work with different methodologies.

Problem formulation

We assume that a star topology as single level tree network with a set of p processing elements. The center element of the topology called central processor (pc) or the originator which gathers workloads from the grid users and distribute the loads to the processing elements $p_1, p_2, \ldots, p_m$. We will use the word processor to denote the processing elements with cpu, memory, and communication links. The names processor, computer and processing element will denote the same thing. Initially the workloads are held by the central processor pc. There is no direct communication between the processors $p_1, p_2, \ldots, p_m$. All the communications are through the central processor pc. To simplify the mathematical model we assume that the result returning time is negligible. This simplification is not limiting generality of our considerations because result gathering can be included in the model. The computation will be starts whenever a processor receives its entire loads. We assume that all the processors have independent communication hardware which allows for simultaneous communication and computations on the previously received loads. Here we consider for scheduling multiple divisible loads. Moreover each load of the application is a same type of parallel application. The set of loads is $L=\{L_1, L_2, \ldots, L_n\}$. Each load L_j is represented by the volume of load W_j that must be processed. To achieve good performance of computation the load L may be reordered by the central processor. The central processor splits the loads into parts and sends then to the processors $p_1, p_2, \ldots, p_m$ for remote processing. To process the load L_j only some subset $P_j \subseteq P$ of processors may be used. The size of the part of load L_j is defined as α_{ij} which is sent to the processor p_i. The part of load α_{ij} can be expressed in load units (eg. in bytes). The sum of load parts α_{ij} become a total workload of L_j. ie. $\sum \alpha_{ij}=W_j$. The role of central processor is not only to select a processor, also it assign a load W_j into chunks α_{ij}. The goals of this paper are twofold: to propose an algorithms to schedule divisible load computations such that schedule with minimum processing cost with the budget and the availability is bounded, and to study the influences of system parameters on the quality of the scheduling problem solutions. An alternate formulation of the problem could be find the minimum completion time such that the processing cost and the availability is bound.

Single divisible load

In this section we analyze the complexity of scheduling a single divisible load, divided in to multiple portions. The problem we consider is a bi-criteria optimization problem. This bi-criteria problem can be relaxed to two simple problems: (i) minimization of schedule length with respects to the budget available, (ii) minimization of cost on condition that the budget and the processor availability. The criteria are schedule length and the processor usage cost. With respect to the schedule length lot of research has been made and also it is mainly focusing the optimization of system performance. Here we are focusing the second case of processor usage cost. Our aim is to allocate the resource for the loads to minimize the grid usage cost as well as to get the considerable amount profit for the grid resource providers. Both problems can be solved in polynomial time by use of linear programming, provided that the set of processors used and the sequence of their activation is known. For the mathematical simplicity we are not included the processor start-up time, and assuming that all the resources are dedicated one.

Notations used

c_{p_j} – Amount required to process a unit portion of load

t_{p_j} – Time required to process a unit portion of load

t_{z_i} – Time required to transmit a unit portion of load

α_i – Portion of workload allocated to the i^{th} processor

B – Budget available to complete the toatl workload

A_i – Available time duration of the i^{th} processor

D – Finish time of a workload

W_t – Total available workload

M_t – Total available memory

t_{s_i} Starting time og i^{th} processor

t_{a_j} – Available time of the j^{th} processor

t_{u_j} – Unavailable time of the j^{th} processor

t_{r_i} – Ready time of the i^{th} processor

x_{ij} – Binary variable, values become 1; if load is allotted; else 0

Minimize

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{p_i} \alpha_i x_{ij} \quad (1.1)$$

Subject to

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{p_i} \alpha_i x_{ij} \le B \quad (1.2)$$

$$\sum_{i=1}^{m} t_{s_i} \alpha_i x_{ij} + \sum_{i=1}^{m} t_{p_i} \alpha_i x_{ij} \le A_i \quad ; j = 1 \text{ to } n \quad (1.3)$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} t_{s_i} x_i x_i + \sum_{i=1}^{m}\sum_{j=1}^{n} t_{p_i} \alpha_i x_{ij} \le D \quad (1.4)$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_i x_{ij} \le W_t \quad (1.5)$$

$$\alpha_i x_{ij} \le M_t \quad ; i = 1 \text{ to } m \;, j = 1 \text{ to } n \quad (1.6)$$

$x_{ij} = (0,1)$; $i=1$ to m , $j=1$ to n (1.7)

$$\alpha_i \ge 0 \quad ; i = 1 \text{ to } m \quad (1.8)$$

Equation 1.1 is the objective function to minimize the cost of the grid usage, from equation 1.2 to 1.8 is the constraint set. The equation 1.2 is the capacity constraint, 1.3 is to match the availability of the processor with the process execution time. The equation 1.4 shows that the deadline constraint, the time allotted to complete the workload process should satisfy the process time by each of the processor. The equation 1.5 satisfies that the sum of portion of the entire sub task should match with the total workload. Equation 1.6 is the processor available time with the processing time of the workload portion allotted to that processor. The equation 1.7 and 1.8 is the binary value constraint and the non-negativity constraints respectively.

Computational Experiments

The objective is to investigate rate at which the resource users and the resource providers are matched by the system's dynamic allocation mechanism as they emerge. The Table.1 shows the properties of the resources available in the grid system. From the table we are assuming that there are five resources which is in the first column of the table. The second column specifies the amount of time required to execute/process one unit of workload per unit time with respect to the resources. The third column data specifies the cost required to execute/process a unit of workload in the various resources. And the data in the last column shows the resource available time in the schedule. We assumed that all the resources are dedicated resources and the resources which are joined in the scheduling process could be available in the entire scheduling process. Also that all the resources are always ready to receive the workload portions and there is no communication delay to transfer the portion of workload from the scheduler to the resources, and also sending the result back to the scheduler. The resource will process its workload whenever it receives its complete portion of workload allotted at a time. One resource can process more than one portion of workload of a source when some resource is fails to process its assigned job. Here we considered a single source ie. only one grid user can utilize the grid system.

Table 1. Resource availability

Resource	Time/Unit Load	Cost/Unit Load	Available Time
R1	2	6	25
R2	3	4	30
R3	2	6	25
R4	2	6	25
R5	3	4	30

Table 2. Workload allocation

Resource	Work Load Allotted	Time taken to Process
R1	11	22
R2	10	30
R3	12	24
R4	7	14
R5	10	30

To perform the experiment we have consider the total workload as 50 workload units with the dead line of 120 time unit and the budget allotted to complete the workload is 260 cost units with five resources are in the grid system. We first perform the operation which focuses on cost minimization and later on time minimization. From Table.2 the total workload of 50 workload unit is divided into five portions such as 11, 10, 12, 7 and 10 and which is allotted into the resources R1 to R5 respectively. The grid system completes the job with the budget of 260 cost unit and it takes 120 time units to complete the entire workload. Figure 1 shows the resource utilization chart with the resource available time and resource utilization time are the two bars in the chart. Figure.2 shows the resource utilization chart with different basic solutions. For all the basic solution the resource 2 and the resource 5 is fully utilized because its cost is low compared with the other resources. The experiment can be extended with very large number of resources. IBM offers On Demand Business Service similar to Sun Grid, where the customers are well versed in pay-as-you-go service so that the cost incurred towards the excess hardware and software investment and maintenance become minimized. The actual pricing model has not been disclosed as in the case with Sun but is rather on per-customer basis. Beyond the computational grids, research grids such as

TeraGrid and SURAgrid exist where the cost is not directly tied to currency but rather to allocation time. Regardless of how one approaches the grid, there is a cost associated with the available services.

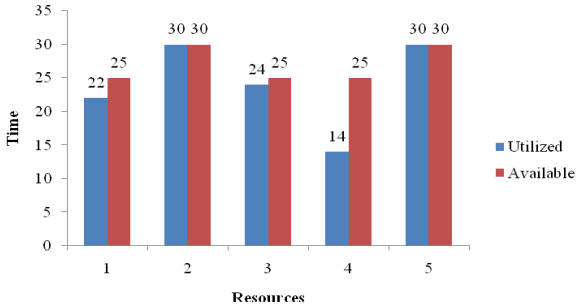

Fig. 1. Resource Utilization Chart

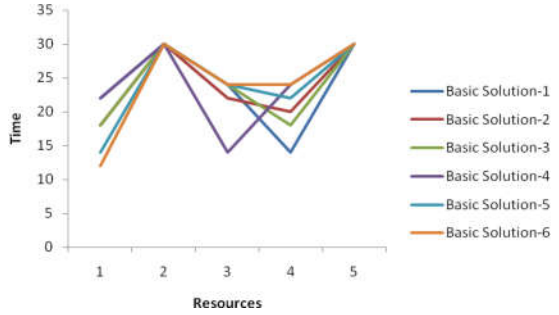

Fig. 2. Resource Utilization Chart for various Starting Solutions

Table 3 shows the workload allocation for time minimization problem. Here we set the objective is to minimize the time taken to execute the entire workload. The allotment is different when compared with the previous cost minimization problem. The total time taken to complete the entire workload is 120 time unit and the total cost consumed is 260 cost units.

Table 3. Workload allocation

Resource	Work Load Allotted	Time taken to Process
R1	6	12
R2	10	30
R3	12	24
R4	12	24
R5	10	30

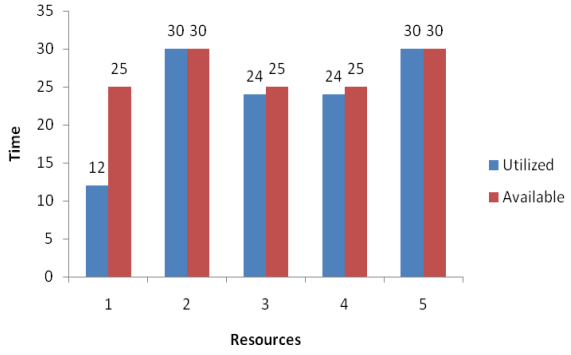

Fig. 3. Resource Utilization chart

Figure 3 and 4 shows the resource utilization chart and the resource utilization chart with different basic solutions respectively for time minimizing as the objective function. Here to the second and the fifth resources are fully utilized. The total time conserved is 120 time units.

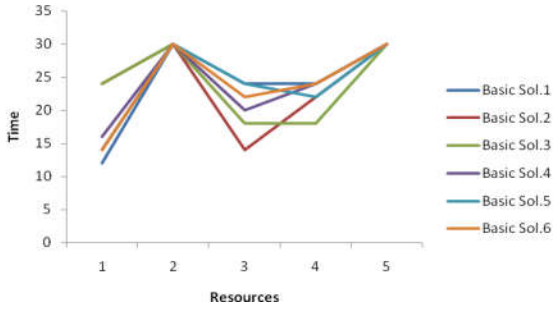

Fig 4. Resource utilization chart with different basic solutions

Conclusion and Future work

The problems we analyzed in this paper consist of determining optimum destinations for the load chunks and adjusting their sizes to the speeds of processors, communication links and possibly memory sizes. A non linear programming formulation has been proposed for a fixed processor activation sequence. Also we studied the problem of scheduling single divisible load on star network for the schedule length and the schedule cost criteria. To the successful execution of the model we should know the resource availability and the sequence of execution. Also we assumed that all the resources are the dedicated resources. In future it can be extended into non-dedicated resources and also include the communication delay and the execution startup time. Our future work is to implement the other two method of work distribution such as multiple loads with single distribution and multiple loads with multiple distributions.

Related works

As economic models are introduced into Grid computing, new research opportunities arise. Because the economic cost and profit are considered by Grid users and resource providers respectively, new objective functions and scheduling algorithms optimizing them are proposed. The economic problem exists only when the resources and the participants, namely consumers and producers, maximize their utility by deciding among needs that cannot concurrently satisfied and lead to different utility levels (Nakai, 2002). There are various economic models for Grid scheduling has been proposed with different techniques and objectives such as Commodity market model, Bargaining model, Tender/Contract-net model, Auction model, Community model, Monopoly/Oligopoly and so on proposed in (Buyya *et al.,* 2002). In (Takefusa et al. 2001 and , Viswanathan *et al.,* 2007), a deadline scheduling algorithm that supports load correction and fallback mechanisms to improve the algorithm's performance in an environment that is based on client-server model is proposed. It submits each job to a resource that can finish it in time less than or equal than the time specified for completing the job. The aim of this work is to decrease the number of jobs that don't meet their deadlines. The resources are priced according to their performance. It uses bricks simulator which is written

in Java after extending it to assess the performance of the implemented deadline algorithm. In (Buyya *et al.,* 2000), several algorithms called deadline and budget constrained (DBC) scheduling algorithms are presented which consider the cost and makespan of a job simultaneously. These algorithms implement different strategies. For example, guarantee the deadline and minimize the cost or guarantee the budget and minimize the completion time. The difficulties to optimize these two parameters in an algorithm lie in the fact that the units for cost and time are different, and these two goals usually have conflicts (for example, high performance resources are usually expensive). We can also find such examples in (Venugopal and Buyya, 2005) and (Yu *et al.,* 2005). The most straight forward method of an early grid and matching price model is the Sun Grid. The Sun Grid offers a pool of pre-installed applications as well as options for users to deploy their own applications on Sun's resources. The pricing model for this service is straight forward: $1/CPU-hour. The effective workload allocation model with single source has been proposed (Ger and De, 2008) for data grid system (Abdullah *et al.,* 2009). The time and cost trade-off has been proposed (Garg *et al.,* 2009) with two meta-scheduling heuristics algorithms, that minimize and manage the execution cost and time of user applications. Also they have presented a cost metric to manage the trade-off between the execution cost and time.

REFERENCES

Abdullah M,.Othman M, Ibrahim H and Subramaniam S, 2009. Load Allocation Model for Scheduling Divisible Data Grid Applications. *Journal of Computer Science*, 5 (10): 760-763.

Buyya R, Giddy J, and Abramson D, 2000. An Evaluation of Economy-based Resource Trading and Scheduling on Computational Power Grids for Parameter Sweep Applications, in Proc. of the 2nd International Workshop on Active Middleware Services (AMS 2000), pp. 221-230, , Pittsburgh, USA.

Buyya R, Abramson D, Giddy J, and Stockinger H, 2002. Economic models for resource management and scheduling in grid computing. Journal of Concurrency and Communication: *Practice and Experience*, pages 1507-1542.

Garg S.K, Buyya R and Siegel H J, 2009. Scheduling Parallel Applications on Utility Grids: Time and Cost Trade-off Management, Proceedings of the Thirty-Second Australasian Computer Science Conference (ACSC2009), Wellington, Australia, Conferences in Research and Practice in Information Technology (CRPIT), Vol. 91

Ger K, De B, 2008. Resource Allocation in Grid Computing. *Journal of Scheduling,* 11:163-173.

Nakai J, 2002, Reading between the lines and beyond. Technical Report NAS-01-010, NASA Ames Research Center.

Takefusa A, Matsuoka S, Casanova H, and Berman F, 2001, A study of deadline scheduling for client-server systems on the computational grid. In HPDC'01: Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10'01), pages 406-415. IEEE Computer Society.

Venugopal S and Buyya R, 2005, A Deadline and Budget Constrained Scheduling Algorithm for eScience Applications on Data Grids, in Proc. of 6th International Conference on Algorithms and Architectures for Parallel Processing(ICA3PP-2005),pp.60-72,Melbourne,Australia, Oct. 2-5.

Viswanathan S, Veeravalli B and Robertazzi T G, 2007, Resource-aware distributed scheduling strategies for large-scale computational cluster/grid systems. IEEE Transactions on Parallel and Distributed Systems,18: 1450-1461.

Yu J, Buyya R, and Tham C K, 2005. QoS-based Scheduling of Workflow Applicationson Service Grids, in Proc. of the 1st IEEE International Conference on e-Science and Grid Computing (e-Science'05), Melbourne, Australia. "Sun Grid" http://www.sun.com/service/sungrid/.

IBM–On Demand Business, http://www.ibm.com/e-business/ondemand/us/index.html."TeraGrid",http://www.teragrid.org"SURAgrid", http://www.sura.org

****