STATE OF THE ART: TASK SCHEDULING ALGORITHMS IN A HETEROGENEOUS GRID COMPUTING ENVIRONMENT

M.Hemamalini¹, M.V.Srinath²

¹Department of Computer Science, A.V.C College (Autonomous), India.
E-mail: maliniavce@gmail.com

² Research Supervisor, Bharathidhasan University, Trichy, India
E-mail: sri_induja@rediffmail.com

Abstract

Grid Computing is the distributed form of parallel computing which is used to solve large scale and complicated problems. Grid Task scheduling is a NP complete problem. It is used to schedule a task on an appropriate grid node. One of the main idea to improve the performance of grid is how tasks are mapped on to suitable resources. A schedule is said to be finest if it provides a solution to make use of the available resources efficiently. Several scheduling algorithms improves the makespan but it causes a severe load imbalance. This paper presents an assessment of task scheduling algorithm based on various factors such as communication cost, execution time, memory requirements, node failure, task duplications and network behavior.

Keywords:
Grid computing, load balancing, fault tolerance, distributed computing.

1. INTRODUCTION

As the scientific problem grows up in the modern computing technology, an organization needs higher computational resources such as more computing power and storage space. Grid computing is a form of distributed computing that links the computers, databases and users in a flawless web.

Advancement in technology has led to the possibility of using resources to solve the increasing need for scientific, engineering and research problems in a distributed environment [7]. The overall computation time will be reduced if the network and resource status are in sufficient level. The Grid architectures provide a middleware technology for various reasons such as resource allocation; task scheduling, authorization, security and data management [11]. This paper is organized as follows: section 1 presents the preliminaries, section 2 elaborates the factors influencing the task scheduling algorithm, Section 3 describes the workflow task scheduling algorithm for grid computing, section 4 illustrates the related work, section 5 shows the comparison of Grid Task scheduling algorithms and we conclude this study in section 6.

2. SCHEDULING IN GRID

The main goal of a grid is to provide efficient access to remote and geographically distributed resources. A scheduling service is needed to coordinate the access to the different resources. Due to its dynamicity and heterogeneity, scheduling becomes difficult. The scheduler dispatches the tasks to the available processors.

2.1 TYPES OF GRID

Different types of grids are developed based on available resources for grid usage, structure of the organization usage, services provided by grid. There are many grid such as departmental grid like Folding@Home, which is used to solve problems for a particular group, Global Grid are accessible over the internet like. White Rose Grid, Compute Grids like smart grid are exclusively used for providing access to computational resources , Utility Grid, which provide access to resources, Extraprise Grids like Amazon.com are established between companies, customers, etc[4].

2.2 TASK SCHEDULING

Grid architecture enforces complex features due to high heterogeneity of the resources, nature of applications and their high demand, distance covered by the network, large volumes of data that needs to be shipped around as required [5].

As many parallel methods emerge an important issue, load balanced scheduling take place as shown in figure.1. In general, load balancing techniques requires knowledge of available resources and their operating condition.

As grid size increases from ten to thousands of hosts, fault tolerance becomes the main issue. To solve this problem, grid functionalities should be decentralized to avoid bottlenecks and ensure scalability. A way to guarantee grid scalability is to implement P2P models and techniques to implement decentralized grid services [20].

Many application domains are resource intensive and require multiple resources to perform the required task efficiently. In a non-dedicated heterogeneous environment, it is essential to have prior knowledge of resource and current operational condition of environment [5]. In a homogeneous environment, tasks can be divided into several subtasks and scheduled on the available resources. If the resources are shared among multiple applications, some factor may change during execution such as memory size, bandwidth and processor utilization.

2.2.1. Load balancing:

It is a method to achieve good performance between service providers in a distributed environment. The load of an entity can be its access rate, the number of executions of some important steps for each access, the number of bits transfered for each request etc... Execution time and memory requirements are the two most common parameters used for load balancing [17]. We categorize load balancing methods as either static or dynamic.
2.2.2 Static methods:
These methods usually acquire knowledge about the environment before execution and thus distributes the task accordingly. This is applicable only for dedicated heterogeneous environment. For example, linear equation is solved to decide on the size of each subtask based on the systems specifications [13].

Another example illustrated in [14] using maximum flow algorithm to reduce the task assigned to the heavily loaded resource or the slowest resource.

2.2.3 Dynamic Load Balancing:
Load distribution may involve several methods. Master/slave model is one of the easiest models as it divides the task into multiple partitions and distributes those to slaves through several steps [8].

2.2.4 Central versus Distributed methods:
It is the other categorization for load balancing. Centralized load balancing is dominant as many applications requiring also uses a centralized model.

2.2.5 Semi distributed load balancing algorithm:
In [22], the computing environment is clustered in a mesh structure such that each node informs the others if it was idle or if one of its neighbors is idle. The communication overhead is reduced in decentralized load balancing algorithm in grid environment. It maintains the updated state information during run time [1]. Generally, load balancing technique requires prior knowledge of the resources and operating conditions of environment during runtime.

2.3 CATEGORIZATION OF OBJECT, DIMENSION AND SEGMENTS OF THE GRID SYSTEM
Table 1 illustrates the factors that influence the task scheduling in grids and their various dimensions. They are generally said to be, dependency among tasks, communication cost, execution time, error factors such as node failure, network failure, data storage and task duplications. The scheduling can performed based on Quality of Service, execution cost, heterogenic network behavior, fault tolerance, performance, resource recovery, resource allocation and management and security [4].

<table>
<thead>
<tr>
<th>Object</th>
<th>Dimension</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Availability</td>
<td>Static</td>
<td>Load Balancing</td>
</tr>
<tr>
<td></td>
<td>Dynamic</td>
<td>Task Duplication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Job Migration</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rescheduling</td>
</tr>
<tr>
<td>Nature of the resource</td>
<td>Homogeneous</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Heterogeneous</td>
<td></td>
</tr>
<tr>
<td>Task</td>
<td>Dependent</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Independent</td>
<td></td>
</tr>
<tr>
<td>Problem</td>
<td>Resource Oriented</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Application Oriented</td>
<td></td>
</tr>
<tr>
<td>Mapping the Task</td>
<td>Priority Based</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random Allocation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Algorithm based</td>
<td></td>
</tr>
<tr>
<td></td>
<td>QOS Constraint based</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Memory Requirement, Communication Cost Network</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bandwidth</td>
<td></td>
</tr>
<tr>
<td>Task Processing</td>
<td>Readily Available</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task Duplication Availability of memory</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task Replication Availability of memory</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Access from the source Based on shortest path</td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>Resource Failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task Migration</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rescheduling</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Idle Resource</td>
<td></td>
</tr>
</tbody>
</table>

3. WORKFLOW SCHEDULING ALGORITHMS FOR GRID COMPUTING
There are two kinds of workflows such as abstract workflows and concrete workflows. Abstract workflows are composed of tasks and shows the data dependencies between tasks. Concrete workflows are the mappings of abstract workflows to grid resources. There are many users competing for the resources, so scheduling must be made in the shortest possible time. Workflow scheduling is essential for non-dedicated and heterogeneous environments which need to deal with the issue of data transmission across various communication links. The input of workflow scheduling algorithms is generally an abstract workflow model which describes the tasks without stating location of resources on which the tasks are scheduled [7]. Deterministic type of abstract workflow model is represented using Directed Acyclic Graph (DAG).
3.1 DIRECTED ACYCLIC GRAPH

A grid application consists of many dependent tasks, which are represented by a Directed Acyclic Graph (DAG). Let G = (V,E), where V = Vi for i = 1..n. The vi represents the node i. E= (vi,vj) i,j = 1..n is the set of directed edges and T is a set of t nodes. A sample DAG is shown in figure 2.

![Sample DAG Diagram]

Each node in the DAG represents a task which is a set of instructions. This must be executed successively in the same resource. The computation cost of a node ti is represented by w(ti). Each edge in the DAG is denoted as (ti,tj), ¥ i,j € n, where i<j. The communication cost or the weight of an edge is represented by c(ti,tj).

The source node of an edge is called the parent node and the sink node is called the child node. A node with no parent is called an entry node and a node with no child is called an exit node.

4. RELATED WORK

Nadar Mohammed et al., proposed delay tolerant load balancing algorithm, in which the client is the central control of the process, instructing the server to start their task and providing them with enough information on how and where to start [5]. The client is also responsible for collecting and aggregating the incoming results and informing the server when it is time to stop their work. The technique Dual Direction Operation permits the application to make use of the resources efficiently. The tasks are handled in the opposite direction, continues the process until workers meet at a point which shows all tasks are done. This is occurred by dividing the work among the server which is grouped in to pair. Each pair works on its part from opposite direction and the server continues its work without consulting the client. The new delay tolerant dynamic load balancing technique can be used to reduce the execution time of the tasks while minimizing the control overhead.

Savio S.H Tse, defined, three online algorithms for balancing the load based on two independent criteria with object reallocation [9]. Tse, defined online Bounds on Balancing scheduling algorithm based on two independent criteria with Replication and reallocation. This algorithm gives three online results for balancing the loads and storage spaces among homogeneous servers [17]. The load refers to a parameter while load balancing is referred to a classical problem.

The server load can be the volume of the documents stored inside the server and the storage space of a server can be the size of the documents stored inside, and the size of the documents can be the memory space needed to process the document. In this algorithm they have considered only loads and storage spaces. They have not considered CPU time and memory requirements.

Kassian Plankensteiner, et. al., proposed a new term called Resubmission Impact for fault tolerance in workflow executions in highly distributed environments which is based on task replication and task resubmission. This system reduces the resource waste compared to task replication and resubmission techniques in the Austrian Grid environment. This algorithm uses this metric to describe the workflow task and the number of replications generated without considering the resource failure characteristics.

Reassignment or Document Reallocation is a technique to attain better outcome. Object reallocation is a method used to balance the load. It can be applied to areas such as distributed database systems, online processor scheduling and distributed memory management [3].

Jasma Balasangameshwara et al. proposed an algorithm performance driven desirability aware load balancing with primary back-up approach based on grid architecture, resource heterogeneity, resource unpredictability and communication delay. This algorithm develops a distributed fault tolerant and load balanced scheduling algorithm that reduces the communication and replication cost of independent jobs.

Weifeng sun et al., proposed a Priority based task scheduling Algorithm in a distributed environment which schedules the task based on priority. In this algorithm, task priority (TP) is calculated by using formula.

\[
TP(i) = \frac{1}{\text{Depth}(i)} \times \frac{1}{\text{DLC}(i)}
\]

Depth (i) – Depth of task Vi
ECT – Estimate Complete Time

Initially the tasks are shortlisted by using priority. The maximum data transmission time with its immediate predecessor is called up link cost and the data transmission time with its immediate successor is called down link cost. The resource having a minimum completion time will be preferred.
for scheduling. P-TSA produces better results than the Min-Min and Max-Min Grid Task Scheduling Algorithm. The Scheduling length is the only factor to be considered in solving the dependent task.

Savio S.H.Tse. Proposed an algorithm “Online Bounds on Balancing Two independent Criteria with replication and reallocation” gives online results for balancing the loads and storage spaces among homogeneous servers. The reallocation cost depends on the nature of parameter such as CPU time and memory requirements.

Dharamendra Chouhan, et al., proposed a Multilevel Feedback Queue Scheduling Technique which allocates the resources for jobs or gridlets. The user jobs are assigned to Processing Elements (PEs), and remaining service time of job is shifted between queues of the multilevel feedback queue scheduler. In this queue, the architecture is partitioned into many prioritized queues. This approach is used for jobs which are in the lower priority queue for long time to obtain resources. This decreases the response time of the starved jobs and overall turnaround time of the process decreases. This technique is simulated using Alea GridSim toolkit [10].

T. Kokilavani et al., proposed Load Balanced Min-min (LBMM) Meta task scheduling algorithm schedules the task first by executing Min-min algorithm and then by rescheduling to balance the load [11].

T. Kokilavani et al., proposed an algorithm “An Ant Colony Optimization Based Load Sharing Technique” which distributes the load among available resources based on the behavior of argentine ants. The resources should be chosen based on RAM requirement and resource availability. The ants choose the path based on the probability value. The Probability value Pj can be calculated using the formula.

\[
P_j = \frac{(R_i + K)h}{\sum_{i=1}^{n} (R_i + K)h}
\]

(2)

Allotment Percentage \(A_j = P_j \times TR_i\)

Where \(A_j\) is the amount of task allotted in resource \(R_j\) and \(TR_i\) is the memory requirement of task \(i\). Based on the value of the coefficient \(K\) and \(h\) the probability of choosing the resource will change. This algorithm shares the load among the resources and reduces the overall response time and increases the resource Utilization [16].

Saule et al. defined bicriteria load balancing algorithm [15]. They proposed two P-time algorithms. The first one is for computing an \((1+\Delta+\epsilon, 1=1/\Delta+\epsilon)\) approximated schedule, for any \(\Delta, \epsilon > 0\), for independent tasks. The dependent task can be represented by using Directed Acyclic Graph.

Sanjay Kumar Panda, et al., Proposed an algorithm “a Semi-Interquartile Min-Min Max-Min (SIM^2) Approach for grid Task scheduling” which is used to minimize the completion time of the tasks and improves the makespan and resource utilization[12].

Masahiro Tanaka, et al., proposed workflow scheduling used to reduce data movement using Multi-Constraint Graph partitioning. This technique relates the dimension of weight vectors to the rank of a task. The method reduces the file size accessed from remote nodes and also reduces the elapsed time [6].

Etminani.K proposed A Min-Min Max-Min selective Approach for Grid Task scheduling Algorithm to improve the resource Utilization. It is a combination of Min-Min and Max-Min grid task scheduling algorithm. This algorithm used to calculate the Standard Deviation Using Expected Completion Time. Based on the Standard Deviation, it selects any one of the algorithm. Time Complexity of this algorithm is \(O(t^2)\) [2].

Chao-Chin Wu et al [19] proposed a Genetic Algorithm based integrated job scheduling strategy that support various fault tolerance mechanisms. They are

i. Job Retry Mechanism(JRT). The JRT method is used to re-execute the failed task on the same computational node.

ii. Job Migration without check pointing (JMG) mechanism. This mechanism is used to move the failed task to another computational node and re-execute the job from the beginning.

iii. Job Migration with Check Pointing (JCP) Mechanism. This mechanism is used to move the failed task to another computational node and resume the execution from the last check point and record the status of the job periodically at run time.

iv. Job Replication Mechanism(JRP). This method is used to replicate a job to multiple computational nodes so that the task has high success rate. If any one of the replicas is completed, then all other replicas should end their execution to save the computing power.

In favor to JRP, the scheduler assigns many computational sites to execute the task in parallel. The scheduler can execute a certain task by using fault tolerance mechanisms. They proposed a new chromosome encoding approach. A chromosome is a list of variable length integer sequences. Each integer sequence corresponds to a gene, whose length depends on the type of fault tolerance mechanism. The integer sequence represents the job identity (job id) and the node identity (node id).

Whenever a task fails the job is moved to the next node for re-execution based on the node ids. The Roulette Wheel Selection Method [18] is used to select chromosomes based on the fitness value; the highest fitness value will be the selected chromosome for the next generation. They employed cut and splice operator to pick two points in two chromosomes and exchange chromosomes after the points. Since the length of the chromosome is variable in this proposed approach. The mutation operations arbitrarily choose a gene in a chromosome and then mutate its value.
5. COMPARISON OF GRID TASK SCHEDULING ALGORITHMS:

Table 2. Comparison of Grid Task Scheduling Algorithms

<table>
<thead>
<tr>
<th>Scheduling Algorithm</th>
<th>Scheduling Methods</th>
<th>Scheduling Factors</th>
<th>Features</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min-Min Max-Min Selective</td>
<td>Independent</td>
<td>Standard Deviation Using Expected Completion Time</td>
<td>Combination of Min-Min and Max-Min. Based on SD it selects any one of the Algorithm</td>
<td>Improved Resource Utilization</td>
</tr>
<tr>
<td>priority</td>
<td>Dependency</td>
<td>Task Priority</td>
<td>Decision based on a set of parallel task with highest priority value</td>
<td>Resource Utilization increased by 11.6% compared with Max-Min Min-Min.</td>
</tr>
<tr>
<td>Online Bounds on Balancing</td>
<td>Independent</td>
<td>Load and Size</td>
<td>Scheduling Based on Replication and Reallocation</td>
<td>Gives online results for balancing the load and storage space</td>
</tr>
<tr>
<td>Performance –Driven Load Balancing</td>
<td>Independent</td>
<td>Transfer delay between sites and jobs individual response time</td>
<td>Neighbor resources are sort by Transfer delay with Primary Back up Approach</td>
<td>Fault Tolerant scheduling with Low communication cost and Replication cost</td>
</tr>
<tr>
<td>Delay-Tolerant Dynamic Load Balancing</td>
<td>Independent</td>
<td>Well defined boundaries (Known star and end points)</td>
<td>Each server concurrently process the assigned blocks from opposite directions</td>
<td>Load balancing with minimum communication overhead</td>
</tr>
<tr>
<td>An Ant Colony Optimization Based Load Sharing Technique</td>
<td>Independent</td>
<td>Memory Requirements - QOS</td>
<td>Load sharing based on behavior of argentine ants</td>
<td>Reduces the response time and waiting time of the tasks</td>
</tr>
<tr>
<td>Multilevel Feedback Queue Scheduling</td>
<td>Independent</td>
<td>Priority</td>
<td>Architecture is divided into multiple prioritized queues.</td>
<td>Response time of the starved jobs decreases and turnaround time decreases.</td>
</tr>
<tr>
<td>Workflow Scheduling to Minimize Data Movement Using Multi-Constraint Graph Partitioning</td>
<td>Dependent</td>
<td>Weight Vector in DAG</td>
<td>relates the dimension of weight vectors to the rank of a task phase</td>
<td>File size accessed from remote nodes and the elapsed time is reduced</td>
</tr>
<tr>
<td>A Genetic Algorithm based integrated job scheduling</td>
<td>Dependent</td>
<td>chromosome encoding approach</td>
<td>The task will be moved to the next node for re-execution based on the node ids.</td>
<td>Integrated job scheduling strategy that support various fault tolerance mechanisms</td>
</tr>
</tbody>
</table>

6. CONCLUSION

Task scheduling is a significant issue in the current Grid computing scenario. Efficient task scheduling algorithm is required to utilize the resource effectively and minimizes the overall completion time. Task scheduling algorithm can make use of processing capacity of the grid system, thus improving the application performance and producing optimized throughput. The grid scheduling problem involves optimization of several objectives including completion time, job priority, resource utilization, QoS (Quality of Service) metrics, costs, reliability factors, resource requirements of the task etc. Many tasks scheduling algorithm does not consider the failure of any task or resource during scheduling. Some task scheduling algorithm improves the makespan but it causes a load imbalance. An efficient scheduling algorithm can be designed based on Qos factor to balance the load as well as improve the schedule rate in spite of grid node failures. This method is very
useful in grid environment because there is a possibility of any node to fail due to various factors.

7. REFERENCES


AUTHOR

Mrs.M. Hemamalini working as Assistant Professor in Department of computer science, A.V.C College, Mannampandal, Mayiladuthurai. She has obtained Masters Degree in Computer Science from Bharathidasan University, Tiruchirapalli. Currently doing research in the field of Grid Computing. She has presented papers in National conferences and International conferences. She has also published papers in International journals. Her research interests include Parallel Computing, Grid Computing, Mobile Computing ,Cloud Computing and Data Mining.
Dr. M.V. Srinath, Currently Research Supervisor Bharathidhasan University Trichy, India. He has published more than 100 Journals in National and International level and he also presented more than 60 papers in the National and International conferences. He has delivered more than 100 key note addresses and invited talks to different Universities and Colleges. He is a member of different Professional bodies like ISTE, CSI, ISTD, Member - World Council for Curriculum and Instruction.