 ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

An Observation on Performance Analysis of Grid Scheduler

[1]Ashish Chandak, [2]Bibhudatta Sahoo, [3]Ashok Kumar Turuk
[1,2,3]Dept. of CSE, National Institute of Technology, Rourkela, India

Abstract
Grid scheduler allocates resources to task. In this paper, we proposed architecture of grid scheduler for task scheduling in grid. The goal of the scheduler in this paper is minimize makespan and maximize success execution rate of task. A number of experiments were carried out to check performance of grid scheduler and we evaluated performance of grid scheduler using Max-Min, Min-Min and FCFS heuristics.

Keywords
Task Scheduling, Grid Scheduler, Task States.

I. Introduction

Grid computing is considered to be a wide area distributed computing [1, 2] which provides sharing, selection and aggregation of distributed resources that spans not only locations but also various organizations, machine architectures and software boundaries to provide unlimited power, collaboration and information access to everyone connected to a grid and makes them use for their computational purpose. One of the important aspects of a grid is task scheduling. Scheduling is the creation of a schedule: a (partially) ordered list specifying how contending accesses to one or more sequentially reusable resources will be granted. Such resources may be hardware such as processors, communication paths, storage devices or they may be software, such as locks and data objects. Task scheduling is the assignment of a set of tasks to some certain resources by means of starting and ending time of tasks, subject to certain constraints [3]. Task scheduling is integrated part of grid computing. The purpose of task scheduling is to allocate resources for executing task. Task scheduling guides resource allocation. As there are many nodes where a task can be executed, the first question to be answered is how to assign the tasks to them. This assignment is known as task allocation, or global scheduling. Once tasks have been allocated, becomes one of defining a feasible local schedule for each node. Scheduler is responsible for task dispatching as well as assigning each task to resources. A grid scheduler act as interface between the user and resources. It hides the complexity of grid from the grid users. A scheduling algorithm can be classified into clairvoyant or non clairvoyant, with regard to knowledge about characteristics of tasks. A clairvoyant scheduling algorithm may use information of tasks characteristics such as service demand, whereas a non clairvoyant algorithm assumes nothing about the characteristics of the jobs [4]. In this paper, we assume that tasks service demands are known to scheduler. Depending upon task requirement, task can be executed on single site or multiple sites.

A. Single Site Scheduling

In single site scheduling, tasks are not shared among sites. It is executed only on single site.

B. Multi Site Scheduling

Sometimes task scheduling is not possible at single site so in this case tasks are distributed to multiple sites. Grid scheduler dispatches task to multiple sites. In this case, task is divided into subtask and it is distributed over multiple administrative domains. This allows the execution of large task requiring more nodes than available on a single site. In this paper, we presented grid scheduler architecture and performance of grid scheduler is evaluated using Max-Min, Min-Min and FCFS heuristics. We presented various possible task states in grid and system model that explains system behavior.

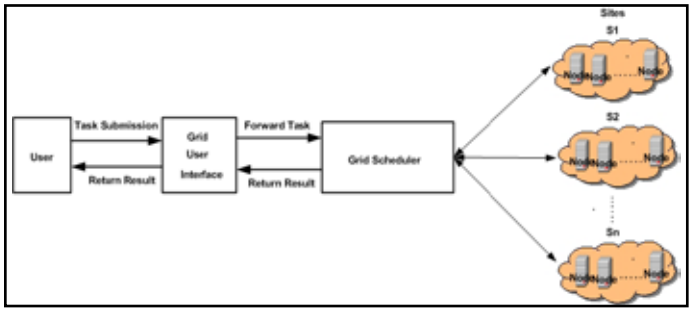

Fig. 1: Grid System

The rest of the paper is organized as follows. We presented grid system description in Section II. Grid scheduler architecture is presented in Section III. System Model is presented in section IV. Various possible states of task in grid are discussed in section V. Performance evaluation of grid scheduler is discussed in Section VI. Finally, some conclusions are drawn in Section VII.

II. Grid System Description

Grid is a system having a number of independent sites as shown in fig. 1 and task execution in grid system is shown in fig. 2. A site may have either a single computing node or a number of computing nodes connected in a distributed manner. Resources in a site are not exclusively dedicated for grid usage. Sites can freely participate in grid computing by offering resources. We represent a grid as two tuple G = <S, TM> where S is the set of sites and TM is the set of tasks. We further represent the set S as S = { $S_1^{N_1}$, $S_2^{N_2}$,..... $S_i^{N_i}$,....., $S_n^{N_n}$ } where $S_i^{N_i}$ is the ith site have Ni number of resources, TM = { Tj | j ε 1,2 .... ,M}, the set of tasks to be executed in the grid. The resources at site Si can be of data, computational or I/O type. Each site S_i is associated with few attribute. They are status St_i of the site (whether working or not working) and maximum capacity Cap_i of the site. A site S_i can be represented in three tuple S_i = <R_i, St_i, Cap_i>. The resource R_i at site S_i can be represented in three tuple R_i = <I/O_i, C_i, D_i> where,
I/O_i = Set of resources of I/O type,
C_i = Set of resources of computational type and
D_i = Set of resources of data type.
The QoS for I/O resources is characterized by speed and latency. The QoS for computational resources is characterized by computational speed and load. The QoS for data resources is characterized by space and disk bandwidth. So, the total no of resources available at all sites at a point of time tj for a task is :
R={C_1, C_2, C_3....., C_n, I/O_1, I/O_2, I/O_3....., I/O_n, D_1, D_2, D_3....., D_n}

III. Grid Scheduler Architecture

In this section, we presented an architecture and flow of task in grid scheduler as shown in fig. 3 and fig. 4 respectively. Functions of different components are explained below:-

A. Site Information System (SIS)

It gathers the following information about each site S_i.
1. Identity of site S_i.
2. Status of the site S_i.
3. The total load at the site S_i.
4. A Site S_j to which the transaction can be copied in case of any failure, at site S_i.

B. Site Information Database (SID)

The information gathered by SIS is stored in SID.

C. Global Task Database (GTD)

This stores information about each task. The following attribute about a task is maintained at GTD:-

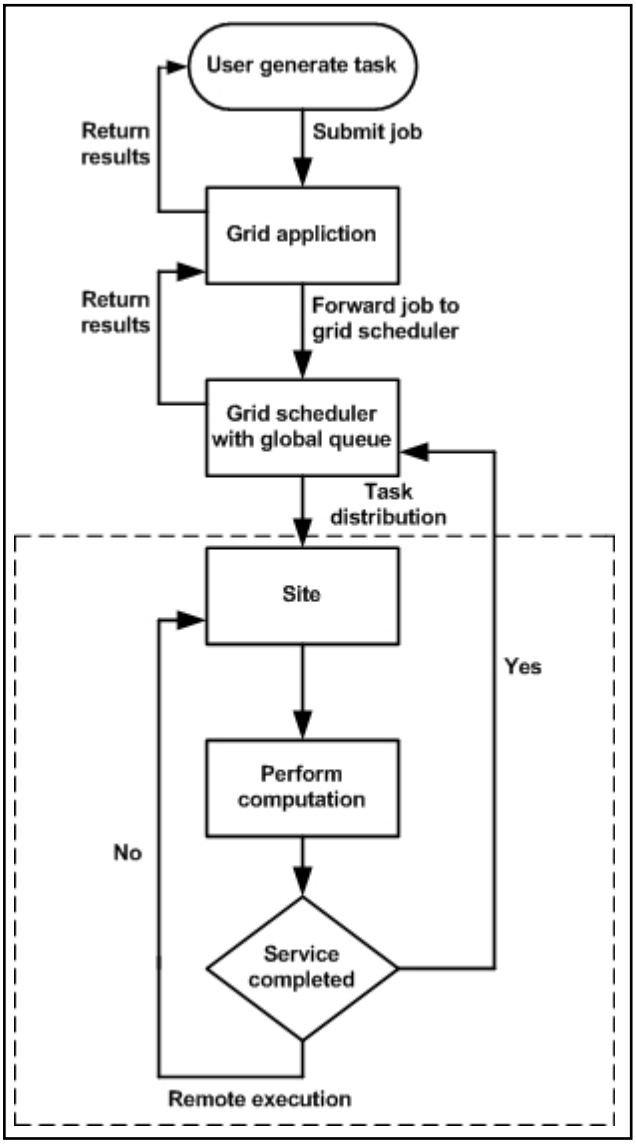

Fig. 2: Task Execution in Grid System

1. User Request Identity (Owner of the task).
2. Grid task Identity (Assigned by grid task scheduler).
3. Arrival time of the task at the grid scheduler.
4. Expected execution time required by the task.

D. Task Determination

This determines the type of resource viz. I/O, data and/or computational required by the task.

E. Task Distribution Manager (TDM)

It dispatches task to remote sites. It is also responsible for management of tasks over multiple administrative domains.

F. Task Matchmaker (TMM)

Matching of resources to a particular task request is done by TMM.

G. Result Management System (RMS)

For collection and returning results.

IV. System Model

We represent the grid system as a M/M/s: N/FCFS queuing model as shown in fig. 5 where: M –

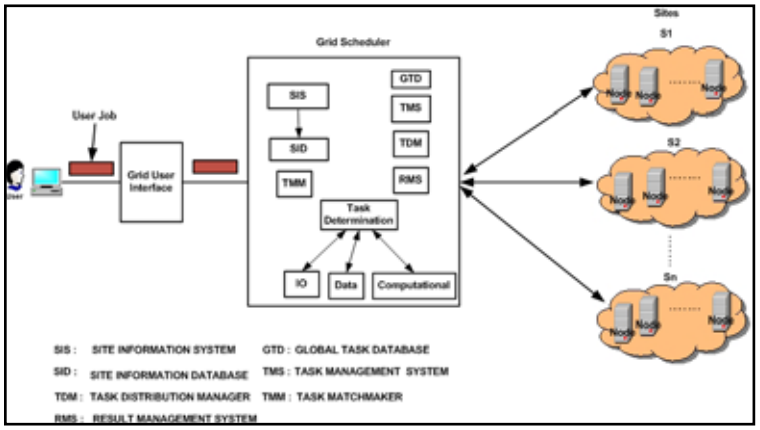

Fig. 3: Centralized Scheduler Architecture

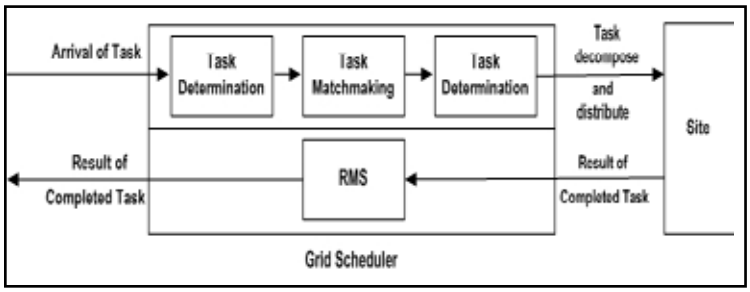

Fig. 4: Flow of Task in Proposed Grid Scheduler

represents exponential inter arrival times between tasks, M- represents exponential execution time of tasks, s - represents number of computing sites, N - represents capacity of system i.e maximum task allowed in the system (this includes executing task plus waiting task), FCFS – represents First Come First Serve queue discipline.

Let λ_i be the rate of arrival of task from each grid user i at the grid scheduler. Assuming that there are j numbers of grid user, the total rate λ at which task arrive at the grid scheduler $\lambda = \sum_{i=1}^{j} \lambda_i$

Let μ_i be the rate at which a task is served at each site i. We assume that the service rate is independent and identically distributed. The combined service rate of all sites in a grid is. $\mu = \sum_{i=1}^{n} \mu_i$ Our queuing model is characterized by following parameters:-

n - Number of tasks in the system.
λ- Arrival rate of tasks.
μ- Service rate of tasks.

ρ- The expected fraction of time the sites are busy i.e. Utilization factor is given by $\lambda/s\mu$.
Then, steady state probability of having n tasks in the system as given by [5]

$P_n = (s\rho)^n P_0 /n!$ for $0 \le n \le s$
$= (s\rho)^n P_0 /s!$ for $s \le n \le N$
$= 0$ for $n > N$

Where P_0 is written as

$P_0 = [\sum_{n=0}^{s}(s\rho)^n/n! + \sum_{n=s}^{N}(s\rho)^n/s!\ (s^{n-s})]^{-1}$

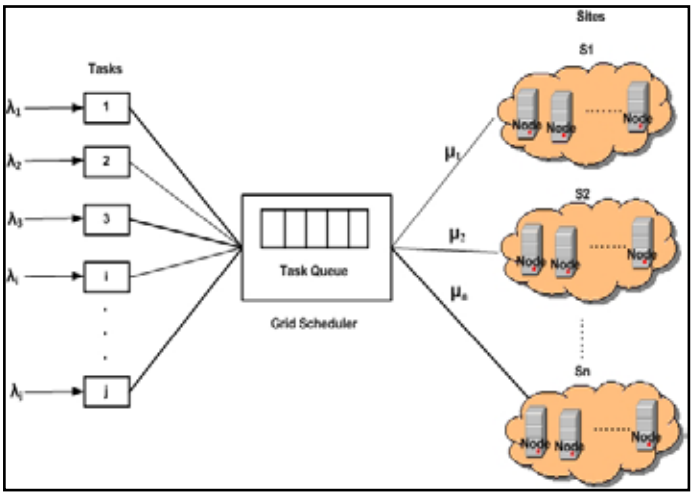

Fig. 5: System Model

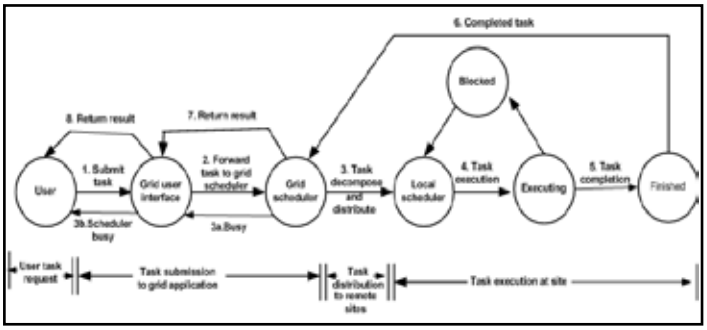

Fig. 6: Task States in Grid Computing

Expected task queue length i.e. expected number of tasks in the task queue given by [5]

$Lq = \sum_{n=s}^{N} (n-s)\ Pn$

$= (s\rho)\ s\ P_0\ \rho / (s!\ (1-\rho)^2)\ [(1-\rho^{N-s+1}) - (1-\rho)\quad (N-s+1)\ \rho^{N-s}]$

V. Task States in Grid Computing

Task in grid application can be any one of the states as shown in fig. 6. Basic fundamental states are:-

A. User

Tasks are generated by the user.

B. Grid user interface

User submit task through grid user interface.

C. Grid scheduler

Task arriving at a grid scheduler is placed in a global task queue and is served on the order of their arrival i.e. in first come first served (FCFS) manner. Grid scheduler checks workload and resource availability of each site and forward task to site that satisfy task requirement. Thus, it will decompose and distribute tasks to remote sites. If scheduler is busy or no sites satisfy task requirement then task is terminated and appropriate message is sent to user.

D. Local scheduler

It contains task which are assigned by grid. scheduler and are waiting to get share of resources. It is responsibility of local scheduler to schedule the task for execution which is assigned by grid scheduler.

E. Executing

Task which is actually using CPU cycles belongs to this state.

F. Blocked

Task for which required resource is currently not available is put in the blocked state.

G. Finished

Task after complete execution goes into this state. Result of the completed task returned to user through grid scheduler.

VI. Performance Evaluation

A. Performance Metrics in Grid Computing

Grid scheduler performance has been measured by various performance metrics. The most commonly used performance metrics are

1. Makespan
2. Resource Utilization
3. Workload Balancing
4. Service Reliability
5. Flowtime
6. Fairness Deviation
7. Throughput
8. Success Task Execution. Table I shows description of various performance metrics. Uses of Performance Metrics.

(i). Performance metrics helps in measuring significant work.
(ii). Unmeasured work can be minimized or eliminated.
(iii). Desired outcomes are necessary for work evaluation.
(iv). It helps in timely corrective action.
(v). Work that is not measured or assessed cannot be managed because there is no objective information to determine its value.

B. Simulation Model

We developed a simulation application in matlab to carry out the experiments. Each simulation experiment ends when 500 tasks executions are completed. Fig. 7, shows the simulation model which consist of nine nodes each having different computing capability. The arrival of tasks is modeled as Poisson random process. To evaluate performance we have considered following three types of tasks: a) I/O intensive tasks b) Data intensive tasks c) Computational intensive tasks. We evaluate performance of grid scheduler and tasks are schedule using simple heuristic viz. Max- Min, Min-Min and FCFS heuristics.

C. Makespan Result

Table 2, shows the comparison of makespan with Max Min, Min Min and FCFS heuristics. From comparison, scheduling task using Max-Min heuristic gives better results as compared Min-Min and FCFS heuristics.

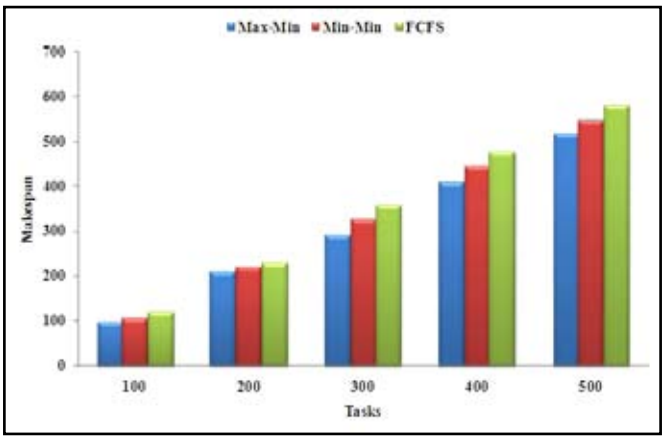

Fig. 7: Makespan Comparison

D. Success Execution Result

Success execution rate is shown in fig. 8. We can see from the results that Max-Min task scheduling gives better success execution rate as compared to Min-Min and FCFS heuristics.

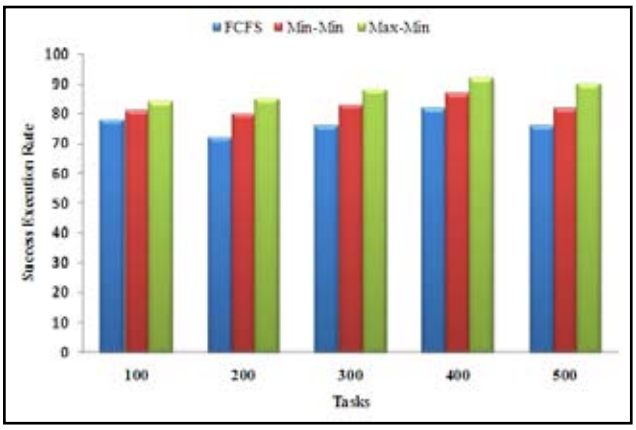

Fig. 8: Success Execution Task Rate Comparison

Table 1 Performance Metrics of Grid Computing

Sl.	Parameter	Reference	Description
1.	Makespan	[6]	Makespan is maximum of completion time. makespan=max(CT_i)
2.	Resource Utilization	[7]	The resource utilization is defined as amount of resource is busy in executing tasks.
3.	Service Reliability	[8]	It is defined as a) Accessibility: - Service is available when desired i.e. when consumer want service. b) Continuity - Consumer has uninterrupted service over desired duration. c) Performance - Meets the consumer expectation
4.	Fairness Deviation	[7]	The fairness of the market means that each resource owner has an equal opportunity to offer its resource and it can obtain a fair profit according to its capability.
5.	Success Task Execution	[7]	Success Task Execution means task is executed within its deadline.

Table 2: Makespan Comparison

Tasks	Makespan in Seconds		
	Max-Min	Min-Min	FCFS
100	96	97	98
200	210	219	228
300	290	326	358
400	409	443	476
500	516	547	580

VII. Conclusions

In this paper, we evaluated the performance of grid scheduler using Max-Min, Min-Min and FCFS heuristics. The experimental results clearly revealed that Max-Min gives better results for minimizing makepan and maximizing success execution rate of task. These results indicate that using max-min heuristic for task scheduling is a suitable selection.

References

[1] Ian Foster, Carl Kesselman, Steven Tuecke,"The anatomy of the grid, Enabling scalable virtual organizations", International Journal High Performane Computing Application 15, pp. 200-222, 2001.

[2] Carl Kesselman Ian Foster, "The Grid 2, Blueprint for a New Computing Infrastructure", ELSEVIER, Second edition.

[3] C. Stein D. Karger, J. Wein, "Scheduling algorithm", in Algorithms and Theory of Computation Handbook. CRC Press, 1999.

[4] C.E. Leiserson Y. He, W.J. Hsu, "Provably efficient online nonclairvoyant adaptive scheduling", IEEE Trans. Parallel Distrib. Syst., 19, pp. 1263-1279, 2008.

[5] S. D. Sharma, "Operation Research", Kedar Nath Ram Nath and Co, Fourteenth edition, 2001.

[6] M. Naghibzadeh Kobra Etminani, "A min-min max-min selective algorihtm for grid task scheduling", In ICI 2007. 3rd IEEE/IFIP International Conference in Central Asia, 2007.

[7] S Hesam Izakian, Ajith Abraham, Behrouz Tork Ladani, "An auction method for resource allocation in computational grids", Future Generation Computer Systems, 26(2), pp. 228 -235, 2010.

[8] Yuan-Shun Dai, Xiao-Long Wang, "Optimal resource allocation on grid systems for maximizing service reliability using a genetic algorithm", Reliability Engineering and System Safety, 91(9), pp. 1071-1082, 2006.

Ashish Chandak received B.E. from Amravati University. Currently, he is research scholar at National Institute of Technology Rourkela India. His research interest includes distributed systems and web technologies.

Bibhudatta Sahoo received the M.Sc. Engineering in Computer Science from National Institute of Technology Rourkela, INDIA, in 1999. He is currently an assistant professor in the Department of Computer Sc. & Engineering, NIT Rourkela, India. His interest include Parallel & Distributed Systems, Networking, Computational Machines, System Software, High performance Computing, VLSI algorithms He is a member of the IEEE Computer Society & ACM.

Ashok Kumar Turuk is a associate professor of computer science and engineering at National Institute of Technology (NIT), Rourkela. He received his B.E. degree in computer science and engineering from NIT, Rourkela, India in 1992, and M.E. degree in computer science from NIT, Rourkela, India in 2000, and Ph.D degree from Indian Institute of Technology (IIT), Kharagpur, India in 2005. His research interests include grid computing, photonic networks, ad hoc networks.