Greedy Grid Scheduling Algorithm in Static Environment

Manish Kumar[1], Ashish Avasthi[2], Rocky Kumar[3], Neeraj Kushwaha[4], Dipti Kumari[5]

[1]Professor, Department of Computer Science, Poornima University, Jaipur, Rajasthan,
Email: manish.kumar1@poornima.edu.in
[2]Professor, Department of Computer Science, Poornima University, Jaipur, Rajasthan,
Email: ashish.avasthi@poornima.edu.in
[3]Teaching Assistant, Department of Computer Science, Poornima University, Jaipur, Rajasthan,
Email: rockykumar18031@gmail.com
[4]Teaching Assistant, Department of Computer Science, Poornima University, Jaipur, Rajasthan,
Email: neerajkushwaha06719@gmail.com
[5]Assistant Professor, Department of Computer Science, Poornima University, Jaipur, Rajasthan,
Email: dipti.kumari@poornima.edu.in

Received: 11.04.2024 Revised : 18.05.2024 Accepted: 28.05.2024

ABSTRACT
Grid scheduling is a technique by which the user demands are met and the resources are efficiently utilized. The scheduling algorithms are used to minimize the jobs waiting time and completion time. Most of the minimization algorithms are implemented in homogeneous resource environment. In this paper the presented algorithm minimize average turnaround time in heterogeneous resource environment. This algorithm is based on greedy approach which is used in static job submission environment where all the jobs are submitted at same time. Taken all jobs independent the turn around time of each job is minimized to minimize the average turnaround time of all submitted jobs.

Keywords: Greedy, Grid, Heterogeneous, High Performance Computing, Scheduling.

INTRODUCTION

Using the distributed resources to solve the applications involving large volume of data is known as grid computing [1], [2]. There exists many tools to submit jobs on the resources which have different computational power and are connected via Local Area Network (LAN) or Virtual Private Network (VPN). The main challenge in grid computing is efficient resource utilization and minimization of turnaround time. The existing system model consists of the web basedgrid network platform with different management policies, forming a heterogeneous system where the computing costand computing performance become significant at each node [3], [4].
In gridcomputingenvironment, applications are submittedfor use of grid resources by users from their terminals. The resources include computing power, communication power and storage. An application consists of number of jobs; users want to execute these jobs in an efficient manner [5]. Thereare two possibilities of submission of jobs/data on resources; in one of them, job is submitted on the resources where the input data isavailable and in the other, on thebasis of specific criteria, resource is selected on which both job and input data are transferred. This paper uses second approach, wherein the job is submitted on a scheduler and data on a resource identified by the scheduler. A resource in existing algorithms is selected randomly, sequentially or according to its processing power [2], [6], [7]. In this paper the proposed algorithm chooses a resource on the basis of processingpower, job requirement and time to start at that resource.
The next section describes details about system model.Section 3 describes the proposed scheduling algorithm instatic job submission environment. In section 4 the experimental details and the results of experiments are presented with comparison of some existing algorithms. In section 5 conclusions and suggestions for futureimprovements are proposed.

System Model

A grid is considered as the combination of multiple layers. In our model the whole system is composed of three layers (Fig.1). The first layer is the user application layer in which the user authentication is done and jobs are submitted to the scheduler by the user. The second layer contains schedulerand GIS. The scheduler schedules jobs among various resources after taking resource status information from GIS. The second layer is connected through a VPN to user. VPN provides additional security and only authorized users can access services. All the resources reside in third layer where user's jobs are executed which are also connected through VPN.

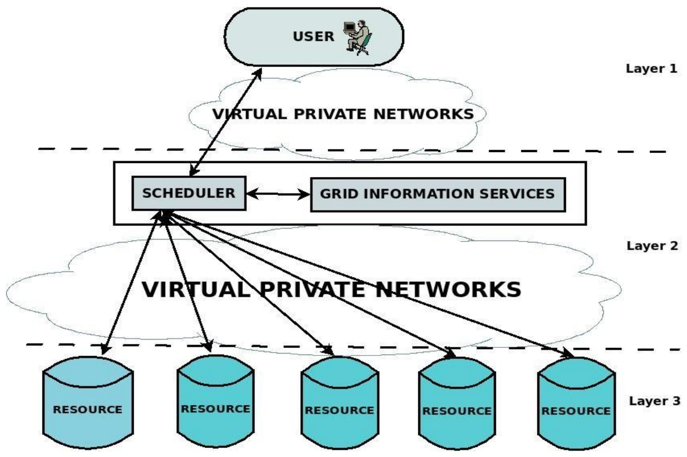

Fig.1: Layered architecture.

Proposed algorithm

The existing grid scheduling algorithms are based on thespeed of resources [6], [7]. Each resource of layer 3 (Figure.1) has different processing power and all the resources oflayer 3 are connected via homogeneous communication environment in which the communication delay between scheduler and resources is assumed constant, also the jobs are assumed to submitted on layer 1 having different job requirement.

An algorithm is proposed in this paper which is suitable for static job submission inheterogeneous resource environment connected to the scheduler through homogeneous communication environment. Greedy approach is used tosolve the job scheduling problem. According to the greedy approach "A greedy algorithm always makes the choice that looks best at that moment. That is, it makes a locally optimal choice in the hope that this choice will lead to a globally optimalsolution"[8]. Theproposed algorithmusesthesimilar approach; it takes every job as independent of each other and each of them is scheduled on a resource to give minimum turn around time for that job.T he over all turn around time of all

The job sis thus minimized. The parameters used in this algorithm are as follows:

A set of resources, R ={R_1, R_2, R_3,.........., R_n}.

J_i=The submitted ith job.

Arr_time$_i$=Arrival time of job J_i.

Proc_power$_j$=Processing power of resource R_j.

Strt_time$_j$= Estimated time at which a job starts execution at resource R_j.

Job_req$_i$=Length of job J_i.

Schd_value$_{ij}$=Expected turn around time of ith job at jth resource.

Min = The minimum of Schd_value$_{ij}$ among all resources.

Res_id=Current selected resource id having optimum turn around time.

The algorithm used to schedule a job is given as follows:

```
GREEDY_SCHEDULE
/*Theuserssubmittheirjobsonthescheduler.*/
ForallresourceRj
/*Initializethestarttimeatresources.*/
    Strt_timej=0.0 End For
/*Thejobsarestored inaqueueQ.*/
InsertallthejobsJiinQWhile Q is not emptydo
Delete the job JiSUBMIT_NEW_JOB UPDATE_STATUS
    AdvancetheQpointer End While
End GREEDY_SCHEDULE
```

The scheduler uses SUBMIT_NEW_JOB algorithm to find the best suited resource that minimizes the turn around time. The turn around time is calculated on the basis of expected completion time of a job. The detailed SUBMIT_NEW_JOB algorithm is as follows:

```
SUBMIT_JOB
    Min= ∞
    Foreveryresource $R_j$
/* Calculating the expected turnaround time*/
Schd_value$_{ij}$=Strt_time$_j$+(Job_req$_i$/Proc_power$_j$) If
Min is greater than Schd_value$_{ij}$
    Then
    Min=Schd_value$_{ij}$Res_id = $R_j$
        End If
        End For
    SubmitthejobJ$_i$toRes_id resource
           SubmittheinputdataofJ$_{ij}$jobtoRes_idresourceEnd
           SUBMIT_NEW_JOB
```

Once the scheduler submits a job to a resource, the resource will remains for some time in processing of that job. The UPDATE_STATUS algorithm is used to find out when the resource will be available to process a new job.The UPDATE_STATUS algorithm is given below:

```
UPDATE_STATUS
/*Res_idistheresourceonwhichthejobJ$_i$issubmitted.j is the index of resource on which the job J$_i$is submitted and $R_j$= Res_id*/
  Strt_time$_j$=Schd_value$_{ij}$End
  UPDATE_STATUS
```

The above presented algorithm has the time complexity of O(n) for each job, where n is the number of resources. The above algorithm required additional space to store the resorces current status for availability.

Experimental Results

The GridSim simulator [6] is used to simulate the algorithms. The GridSim toolkit is used to simulate heterogeneous resource environment and the communication environment. The experiments are performed with three algorithms. The algorithms are Random Resource Selection and Equal Job Distribution and Proposed Algorithm. The input data is taken to be the same for all the three algorithms. The simulation is conducted with three resources which are shown in Table 1.

Table 1. Resources with their architecture and processing power.

Resource	R0	R1	R2
Architecture	SunUltra	SunUltra	SunUltra
OS	Unix	AIX	Unix
Proc_power(inMIPS)	48000	43000	54000

The scheduler submits these jobs on resources according to these algorithms. The algorithms are presented one by one with their simulation results.

Random Resource Selection

In this algorithm the scheduler contacts GIS to obtain the resource information and then it chooses a resource randomly [7]. The job is submitted on this chosen resource. This algorithm is very simple to implement and has less overhead on the scheduler. The bar chart (Fig. 2) shows the turnaround time of different jobs. The completion time is a time at which the result of a job is available. After simulation the average turnaround time is found to be 20105.65 seconds and all the jobs are completed at the 64420.25th second.

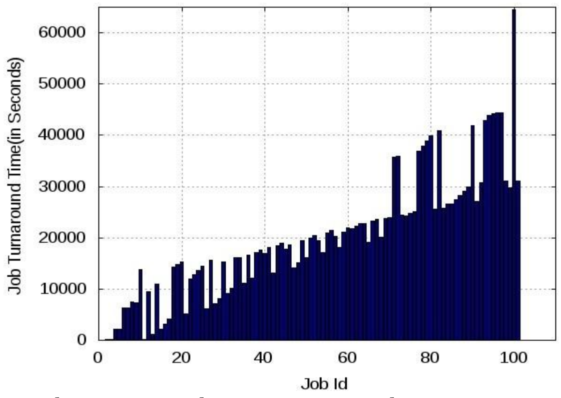

Fig.2: Jobs and turn around time using Random Resource Selection.

Equal Job Distribution

In Equal Job Distribution we firstly calculate the total length of all the jobs and then distribute these lengths equally on every resource. The main notations which are used in the formula are as follows:

L= Total length of all the jobs taken together.

Procpoweri=Processing power of resource Rj.

tProcpower=Total processing power of all resources.

Loadi=Load assigned on resource Rj.

The formula used to calculate the job distribution is given bellow:

Proposed Algorithm

In Proposed Algorithm, the scheduler finds the resource information with the help of GIS and calculates the approximate completion time of this job on every resource. Using these values the scheduler chooses are source which has the minimum of completion time and submits that job on this resource. The turn around time of each job is shown in bar chart in Fig. 4. Through this algorithm the average turn around time of these jobs is 17208.77 seconds and all the jobs are completed at 41840.88th second. The Proposed Algorithm further reduces the average turnaround time by **4.22%** as compared with Equal Job Distribution. The completion time of all jobs takes some more time than Equal Job Distribution algorithm.

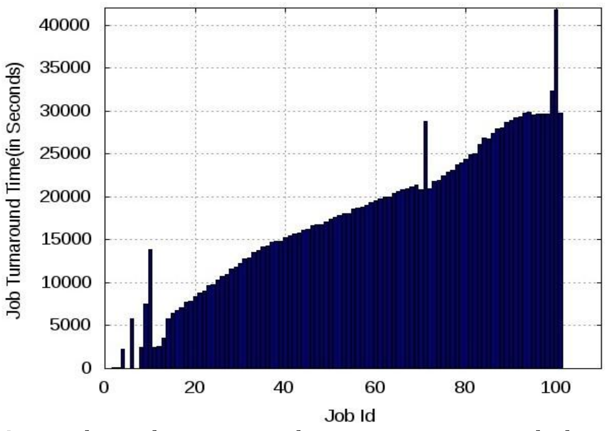

Fig.4: Jobs and turn around time using Proposed Algorithm.

Loadi=L*(Proc poweri/tProcpower)

The turnaround time of each job is shown by the bar chart in Fig. 3. Experimental results show that the average turnaround time is 17968.55 seconds and the last result is outputed at 39000.22thsecond. Equal Job Distribution reduces the average turn around time by **10.62%** and it takes less time in comparison to the Random Resource Selection to give all the results.

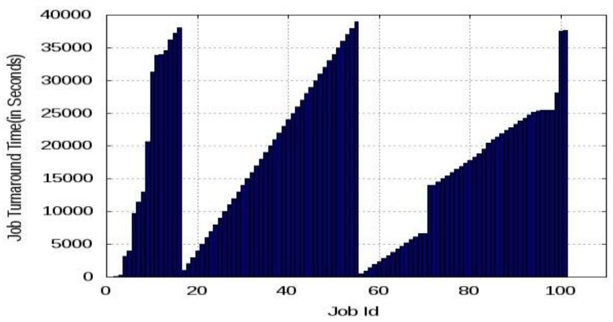

Fig 3: Jobs and turn around time using Equal Job Distribution.

Conclusion and futurework

The proposed scheduling algorithm reduces the average turn around time of all submitted jobs. The considered environment executed the jobs on different resources which are geographically distributed. It is observed that the Proposed Algorithm reduces the average turnaround by 4.22% with Equal Job Distribution (as shown in Table 2). The algorithm uses meta-scheduler where resource failure is not considered.

Table 2: Algorithms with their average turn around time and completion time.

Algorithms	**Average Turn around Time (InSeconds)**	**Completion Time (InSeconds)**
Random Resource Selection	20105.65	64420.25
Equal Job Distribution	17968.55	39000.22
Proposed Algorithm	17208.77	41840.88

REFERENCES

[1] Ammar H. Alhusaini, Viktor K. Prasanna, C.S. Raghavendra, “Unified Resource Scheduling Framework for Heterogeneous Computing Environments", in Proceedings of the Eighth Heterogeneous Computing Workshop,SanJuan, Puerto Rico, pp. 156-165, 1999.

[2] N. Muthuvelu, J. Liu, N. L. Soe, S.r Venugopal, A. Sulistio and R. Buyya, “A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids", Proceedings of the 3rd Australasian Workshop on Grid Computing and e-Research (AusGrid 2005), Newcastle, Australia, 41-48, January30-February4, 2005.

[3] I. Foster, C Kesselman, “The Grid: Blueprint for a new computing infrastructure", Morgan Kaufmann Publishers, San Francisco, USA, 1999.

[4] R. Buyya, D. Abramson, J.Giddy, “Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computation Grid", International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2000), Beijing, China. IEEE Computer Society Press, USA, 2000.

[5] Cong Liu, Sanjeev Baskiyar and Shuang Li, “A General Distributed Scalable Peer to Peer for Mixed Tasks in Grids", High Performance Computing â€“HiPC2007,ISBN:978-3- 540-77219-4, 320-330, 2007.

[6] Rajkumar Buyya, ManzurMurshed, “GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing", Technical Report, Monash University, Nov. 2001. To appear in the Journal of Concurrency and Computation: Practice and Experience (CCPE), pp. 1-32, Wiley Press, May 2002.

[7] Volker Hamscher, Uwe Schewiegelshohn, Achim Streit, RaminYahyapour, “Evaluation of Job-Scheduling Strategies for Grid Computing", in 1st IEEE/ACM International Workshop on Grid Computing (Grid 2000), Berlin, Lecture Notes in Computer Science (LNCS), Springer, Berlin, Heidelberg, New York, pp. 191-202, 2000.[8]Cormen TH, Leiserson CE, Rivest RL, “Introduction to algorithms 2nd edition", MIT and McGraw-Hill Book Company, Boston Massachusetts, cp. 16, 370-403, 2001.

[8] M Kumar, “Implementing Grid Computing in Using the Fractal Technology”, International Conference on Intelligence and Information (ICIIT), IEEE 978-78-1-4244-813 9-2, 2010 (Scopus Indexed).

[9] Yizhun Wang, “Review of greedy algorithm” Theoritical and Natural Science 14(1) 233-239 2023.

[10] Annu Malik, Anju Sharma, Mr. Vinod Saroha, “Greedy Algorithm” International Journal of Scientific and Research Publications, Volume 3, Issue 8, August 2013.