International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 14 (2015) pp 34479-34484


# Job-Profile based selection of Scheduling Algorithms in Grid Environment

**Bimal.V.O[#], G.Raju[##]**
[#] Research Scholar, Department of Computer Science, Karpagam University, Coimbatore, Tamil Nadu, India, bimalvo@gmail.com
[##]Department of IT, Kannur University, Kannur, Kerala, India, kurupgraju@rediffmail.com

**Abstract-** We have many efficient scheduling algorithms in Grid computing. The performance of these algorithms may vary based on the profile of the job. So the selection of the appropriate scheduling algorithm for jobs will definitely improve the overall waiting time of the grid. There are various ongoing research focussed on the grid scheduling algorithms. The main focus of such algorithms is to reduce the overall execution time and the waiting time of jobs. This paper gives an introduction to the job-profile based selection of scheduling algorithms including TimeLine algorithm, which is developed by us for the computer grid. We have selected some important parameters of job and created a profile for the job. The job profile plays an important role for the selection of appropriate scheduling algorithm. Experiments with Alea simulation tool demonstrate that this job profile based selection of scheduling scheme will improve the performance of the grid.

**Keywords-** grid computing, scheduling algorithms, job profile, timeline algorithm, FCFS, EDF, alea

## 1. INTRODUCTION

Grid computing is used by various sciences to solve CPU intensive problems, creating possible benefits to the entire society. With the advance development of grid technology, it is very likely that the industry, educational and research institutions, corporations, and other public sector organizations will deed grids to enhance their computing infrastructure. In recent years there has been a large increase in research on grid technologies, which has produced some reference grid implementations[2]. A computational grid is a parallel and distributed system with a collection of computers that enables the dynamic sharing, selection, aggregation and management of resources for collaborative problem solving[1]. There are various conventional job scheduling algorithms employed in the grid environment. Yet scheduling remains a serious topic of research as there is no single algorithm suitable for the ever increasing demand for distributed computing. The advanced development in the area of grid computing improved the performance of scheduling algorithm design and implementation[2].

Task scheduling is an important area of parallel and distributed computing. Lot of advanced research has been done in this area and many results have been widely accepted by the grid community. With the emergence of the computational grid, new scheduling algorithms are in demand for addressing new issues arising in the grid. In this environment, the scheduling problem is to plan a stream of applications from different users to a set of computing resources to maximize system utilization. This scheduling involves matching of jobs needs with resource availability. In this context, various algorithms are introduced like the FCFS, backfilling EDF etc. The efficiency of these algorithms varies based on the computing environment and computational requirement. In this paper, a new scheduling scheme, based on the job profile is proposed. The objective of this scheme is not to design a new scheduling algorithm but the effective selection of existing algorithms making use of the job profile. Alea, a popular simulation tool in job scheduling is employed for conducting experimental studies[3]. Alea has a capacity to deal with common problems related to job scheduling like heterogeneity of jobs, resources and dynamically changing runtime jobs. The simulation environment allows us an easy implementation of various experiments using different scheduling algorithms.

It is difficult for an accurate prediction of execution performance in any grid environments due to its dynamic nature. The utilization and availability of resources varies over time and a better resource can join at any time. Constructing a schedule for entire execution before the real submission of job may result in a pitiable schedule. If a resource is allocated to each task at the beginning of execution, the execution environment may be entirely different at the time of task execution. A `best' resource may become a `worst' resource. Therefore, the scheduler must be able to adapt the resource dynamics and update the schedule using up-to-date system information[6]. Several approaches have been proposed to address these problems. In this paper, we focus on the approaches which can apply the algorithms in dynamic environments.

## 2. GRID SIMULATORS

There are many grid simulators, For example SimGrid, Simbatch, Microgrid. etc which are widely employed in distributed computing environment. Any new algorithm or algorithm selection scheme should be tested thoroughly before applying to any real application. Hence many simulators have been designed, which overcome the testing cost of the algorithms. A properly designed simulation software is very helpful to simulate various real scenarios and thereby evaluating the performance of scheduling algorithms.

The Alea has been developed since 2007. Over the time many core improvements have been done concerning the design, the scalability and the functionality. Several new scheduling algorithms and objective functions were included along with the support for additional job and machine characteristics. It is a flexible toolkit with very good documentation. It simulates basic grid environment and behaviour. It provides simple implementation of common entities such as computational resources, users and jobs. New

parameters and new functions can be added to get desired functionality for the simulator. Each Grid resource is responsible for the job execution. We can design new scheduling algorithm and design scheduling strategies and selection of scheduling algorithm using Alea. The resource is selected by the scheduler and job is then submitted by the job submission system[3].

## 3. SCHEDULING ALGORITHMS

In this section, a review of a set of prominent scheduling algorithms is given.

**FCFS ( First come first serve )**

FCFS algorithm is a simple and old algorithm where the jobs will be processed based on the arrival ie the first come job will be processed irrespective of any other parameters[4]. CFS scheduling is commonly applied in batch systems. It is non-pre emptive. The order the tasks arrive is very important for the Turnaround-Time: So the negative side of this approach is Convoy Effect and the order of task arrival is very important for average Turnaround time.

**EDF (Earliest Deadline First )**

EDF is an optimal scheduling algorithm on pre-emptive uniprocessors, in the following sense: Given a collection of independent jobs, each characterised by an arrival time, an execution requirement and a deadline. The requirement is that they must be scheduled such a way that all jobs complete by their deadline. The EDF scheduling policy try to achieve this to the extend possible.

However, when the system is overloaded, the set of jobs that will miss deadlines is largely unpredictable. This is a considerable disadvantage to a real time systems like a grid computing. There is a practical difficulty to implement this algorithm because there is a tricky issue of representing deadlines in different ranges (deadlines must be rounded to finite amounts, typically a few bytes at most). If a modular arithmetic is used to calculate future deadlines relative to now, the field storing a future relative deadline must accommodate at least the value of the (("duration" {of the longest expected time to completion} * 2) + "now").

In this approach all jobs will be sorted based on the deadline. The emergency job will execute first irrespective of any other parameter in the scheduling criteria. So the average turnaround time in EDF is better than compared to FCFS algorithm.

**TimeLine Algorithm**

This algorithm is the contribution by us. It is based on the timestamp and the threshold value of the job. Threshold value is the minimum load value of a job for executing locally. It can be static or dynamic. This value can be set by the grid administrator. Time stamp is the local time of the job. In other algorithms the local time factor of the job and resources is not considered. When we look at the real grid environment, the value of a computing cycle mainly depends on the demand and the availability. The owner of any grid resources sells his computation cycle at a higher cost when the demand is high and on the other hand the resources will be available at a cheaper rate when the demand is low. This is the strategy in any marketing industry. In a real grid environment, the local load of any resources will be in a saturated level during day time and comparatively low in late night hours. Hence the probability of the availability of the computation cycles and the price are inversely proportional. So the location of the resource and the TimeLine of that region/country plays an important role. These interesting parameters combined effectively in this algorithm. The timeline algorithm is based on the following observation. During the late night hours the local load of the machines will be negligible and this idle time of a machine in any country is wasted. So based on certain parameters, the entire job can be migrated to a remote system where the timestamp gap is more. This approach leads a considerable improvement in the waiting time of jobs in the job pool[7].

Time Line algorithms or simply TL performs job scheduling based on the load, threshold value of job, geographic time of resource and job.

A simulated study is carried out with the clusters from from USA, Mexico, India, China and Nepal. The jobs are allocated based on the availability of resources in this regions. The algorithm checks the threshold value of the needed resource. We have carried out experiment with the static threshold value 1 for processing element ( the unit of processing power used) . If the resource needed for a particular job is less than the threshold value, the job will be submitted in the local region. Otherwise the job will be migrated to the resource in a country with the maximum timestamp difference. If this region is short of resources, the algorithm will check the next country in the timeline based on the list of resource pool. Ultimately the short job will execute locally and the heavy jobs will be migrated. So this algorithm ensures the maximum elimination of waiting time of any job.

## 4. PROPOSED FRAMEWORK FOR JOB PROFILING AND SELECTION OF ALGORITHM

As discussed in the section III, we have many effective scheduling algorithms developed by the researches in the area of grid computing. The main disadvantage of such algorithm is that they are not considering the job profile. The profile of a job consist of the average processing time of N batch jobs, time stamp of job arrival and number of jobs in the queue. Figure-1 shows the parameters inside the Job profile. Average processing time is calculated using equation -1

$$Tav = \frac{\sum_{i=1}^{N} PTi}{N} \qquad \text{Eq-1}$$

where N is the total number of jobs and PT is the processing time

Jobs are submitted as batch jobs. The time stamp is the current local time of the batch job. There will be several number of jobs in a batch. Each job has its own processing time and job number. The approximate processing time required is calculated from the total resource capacity of particular grid.

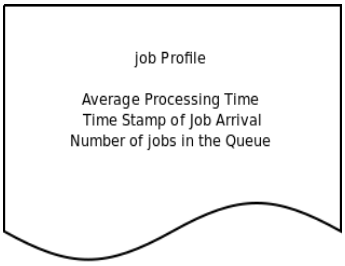


Fig - 1

Table – 1

**Algorithm – Job Profile based selection**
1: start
2: compute average processing time ,avp
3: Set threshold average processing time=tavp
4. Set threshold number of jobs=tj
5: Local time stamp = t , number of jobs=n
6: if timestamp is grater than 8.00 and less than 23.00
select TimeLine() policy
go to step 9
end if
7: if avp<tavp and j<tj
select FCFS() policy
go to step 9
end if
8: select EDF() policy
9: stop

The Fig-2 shows the selection and routing of the algorithm. If a job is submitted during day hours, the TimeLine algorithm will be called for the execution. It is evident from the analysis of different algorithms that the waiting time of the job is much reduced in timeline algorithm. So it is the best suited algorithm for jobs which are submitted during day hours.[7]

If the average processing time and number of jobs are under a threshold value, the proposed grid scheduler will select conventional FCFS algorithm. Table -1 shows the algorithm for Profile based selection of Scheduling algorithm.

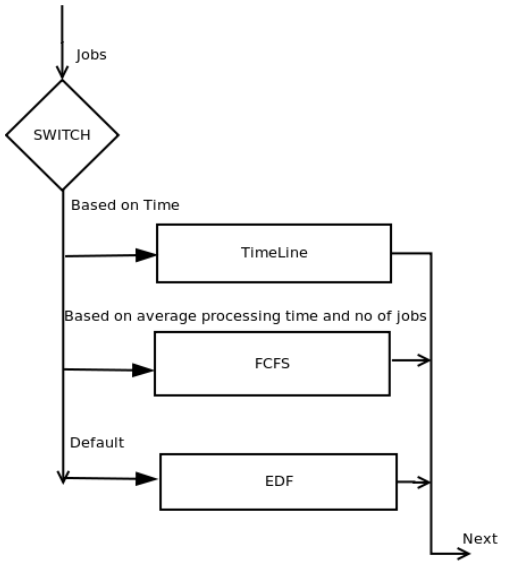


Fig -2

The threshold value for processing time and number of jobs can be set by the grid administrator. Since FCFS is a simple algorithm which is best suited for the above scenario. The default algorithm - EDF will be applied to all other scenarios.

## 5. EXPERIMENTAL SETUP

To setup an experiment, we need some real grid datas. We have collected this data from CERIT-SC workload log. This log was graciously provided by the CERIT-SC and the Czech National Grid Infrastructure.

Table-2 shows the cluster name, total number of CPUs in each cluster and the number of machines in each cluster. These clusters/ CPUs are spread across the globe. 2.93E8 MIPS is the total speed of this grid infrastructure. Total 802 fault free CPUs are available for the experiment. These CPUs are grouped together and formed different clusters. These clusters are named based on the country to which they belong. We conducted experiment with 14 clusters in the 5 regions viz USA, Mexico, India, China and Nepal.

Table – 2

| Name | CPUs | Machines |
|---|---|---|
| Cluster_11_USA | 152 | 19 |
| Cluster_6_USA | 144 | 35 |
| Cluster_10_USA | 92 | 23 |
| Cluster_3_MEX | 80 | 5 |
| Cluster_7_MEX | 70 | 35 |
| Cluster_12_MEX | 64 | 8 |
| Cluster_5_IND | 64 | 32 |
| Cluster_13_IND | 44 | 11 |
| Cluster_4_IND | 32 | 16 |
| Cluster_8_CHN | 20 | 10 |
| Cluster_1_CHN | 16 | 1 |
| Cluster_2_CHN | 10 | 10 |
| Cluster_0_NEP | 8 | 1 |
| Cluster_9_NEP | 6 | 3 |

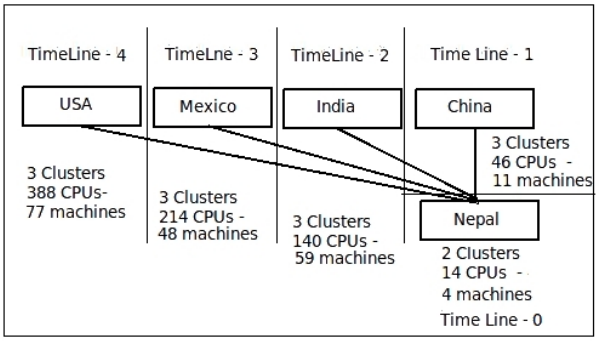


Fig – 3 (Distribution of Timeline clusters)

## 6. Experimental Result And Analysis

In this section, we analyse the job profile based selection of scheduling algorithm. The grid scheduler will schedule the job for TimeLine when the average processing time and number of jobs are above a threshold value. This threshold

value can be fixed by the grid administrator manually. Timeline algorithm will work only during day time. If the Timestamp gives the night Time, this algorithm never execute,. If the number of jobs and average processing time are under a threshold value the scheduler will select FCFS. If the conditions of FCFS and TimeLine are not met, the scheduler will select EDF algorithm.

The performance of the algorithms were analysed by varying the different parameters such as total number of jobs, average processing time and the timestamp of the job.

The figures-4, 5 and 6 shows the waiting time for FCFS algorithm, EDF algorithm and TimeLine algorithm respectively. In the first experiment we set total number of jobs equal to 500 and the time stamp of the job to night.

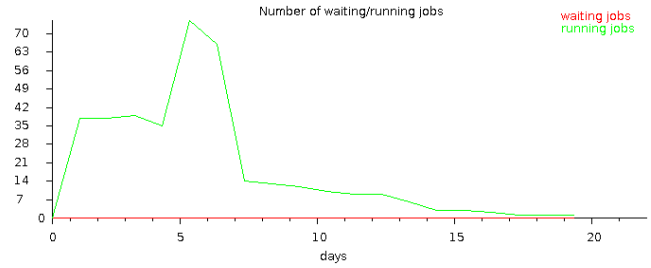


Fig - 4 (FCFS 500 jobs- Selected by Scheduler)

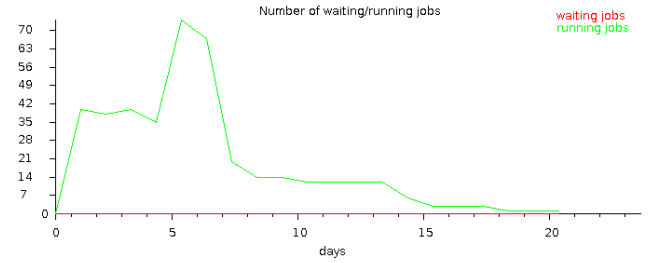


Fig - 5 (EDF, Submitted 500 jobs)

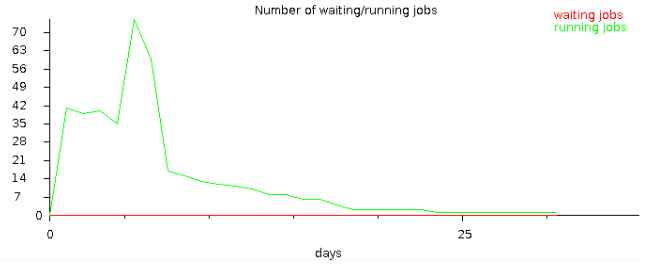


Fig - 6 (TimeLine, Submitted 500 jobs)

By analysing these graphs it is seen that FCFS algorithm is better when we submit less number of jobs, say 500. There is no waiting time for other algorithms also but when we consider the total execution time FCFS has advantage. In the above experiment with 500 jobs, FCFS finished the execution in 17 days whereas EDF took 22 days and TimeLine tooks 32 days. So in this particular scenario the best suitable algorithm is FCFS. Let us consider the scenario when total number of jobs are increased to 5000. Figure-7, 8 and 9 shows the graphs of FCFS, EDF and TimeLine respectively . The time stamp of the job is set to 'day'.

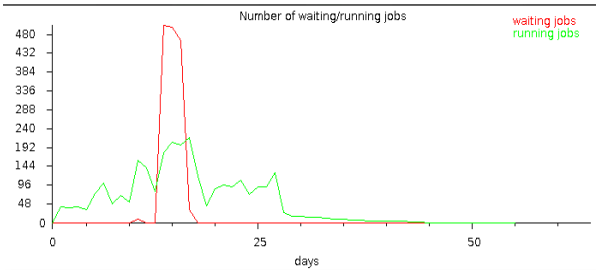


Fig - 7 (FCFS – Submitted jobs 5000)

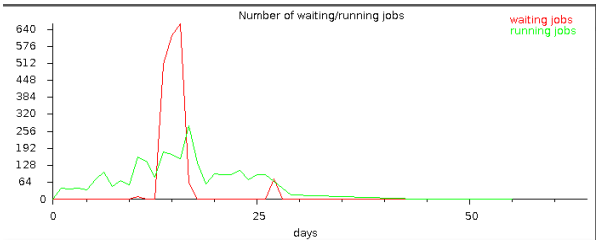


Fig - 8 (EDF, Submitted 5000 jobs)

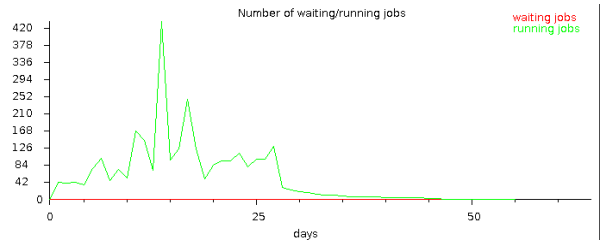


Fig - 9 (TimeLine - 5000 jobs- Selected by the scheduler)

From the above three figures, it is seen that there is no change in the total execution time. In FCFS the number of requested CPUs went above 480 whereas in EDF (figure-8) it is 640 . Form the graph, It is evident that FCFS and EDF scheduling algorithm put some of the jobs into waiting state whereas in timeline algorithm none of the jobs are in waiting state. This is one of the advantages of the timeline algorithm when we execute the jobs at day time. On the other hand the same algorithm gives waiting time during night time. The following graph(Fig-10) shows the execution of TimeLine during Night.


Fig - 10 (Timeline during Night, 5000 jobs)

During night, the waiting time of TimeLine algorithm is slightly less compared to FCFS and EDF but when we consider the network errors and migration cost of jobs, TimeLine algorithm is not a suitable algorithm. We have more free resources available locally during night hours. So exporting jobs to a remote location is not a good idea.

In the next experiment, we set the total number of jobs to 15000, In Fig -11, 12, 13 presents the performance analysis of FCFS, EDF and TimeLine respectively.

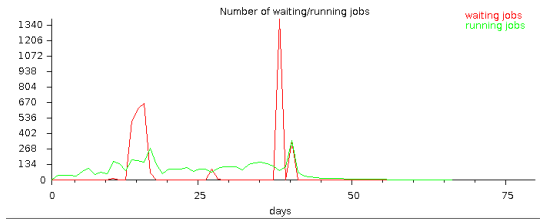


Fig - 11 (FCFS, Submitted jobs 15000)


Fig - 12(EDF, Submitted jobs 15000)


Fig - 13 (TimeLine, submitted jobs 15000)

When we submit large number of jobs(For Eg-15000) there is an increase in the waiting time. The result shows that the FCFS algorithm requested 1340 CPUs whereas EDF (Fig-11,12,13) requested 1330 CPUs and TimeLine is requested around 790 CPUs. It is seen that TimeLine algorithm reduces the waiting time effectively beyond 15000 jobs. It is also noted that when the total no of jobs submitted is equal to 15000, all jobs are completed around 67 days. So there is no increase in the total execution time for TimeLine strategy when compared to other existing algorithms but waiting time of jobs are reduced much.

The below graph shows (Fig - 14 ) the performance of TimeLine during Night when we submit 15000 jobs . It is very clear from the graph that the waiting time is very much increased. So this algorithm is not suitable during night. Job profile based selection of scheduling algorithm avoids TL algorithm during night.


Fig - 14 (TimeLine during night, 15000 jobs)

In FCFS many jobs are executed first come first serve basis, so many instance of time, the clusters are not free and it creates a huge waiting time, On the other end in the case of EDF ,It is noted that the high cost local resources are busy and the low cost resources are free. The earliest deadline jobs get priority. So other jobs have to wait for long time. Hence the resource demanding jobs are intentionally put in waiting state. In the cluster usage of TimeLine algorithm the job distribution is more scattered. The low threshold jobs are submitted in the local area while the lengthy jobs are submitted in the remote area based on three factors.( as mentioned in section IV) These considerations will impact dramatically in the improvement of cost per cycle and waiting time of individual jobs. So TimeLine algorithm effectively and economically use the computing power. [7] In this circumstances a job profile based selection of scheduling algorithm will dynamically select the suitable algorithm .

In Fig-15 the job waiting graph of FCFS, EDF, TimeLine during day and TimeLine during Night is shown. It will help to analyse the waiting time of different scheduling algorithm in the same experiment setup. It is obvious from the graph that the waiting time of the TimeLine algorithm is less compared to others . The profile based selection of scheduling selection will select the TimeLine algorithm

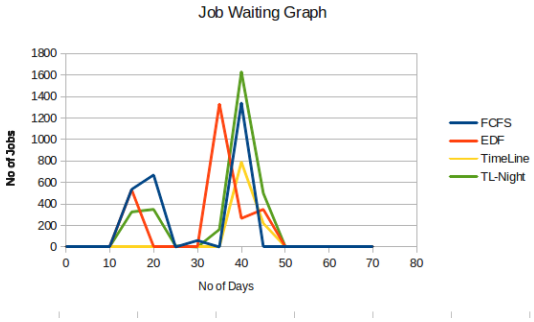


Fig – 15 (Job waiting graph)

## 7. Conclusion

Grid Computing is a promising technology to solve high demanding computational applications . The main goal of the grid environment is to achieve high performance computing by optimal usage of geographically distributed and heterogeneous resources. But the performance remains a challenge in dynamic grid environment. There are a number of factors, which can affect the grid application performance like load balancing, heterogeneity of resources and resource sharing in the Grid environment. This paper, describe a novel approach in the selection of suitable job scheduling algorithm based on the job profile. We found that it is necessary to adapt a particular job scheduling algorithm based on the profile. It avoids static nature of the selection of the scheduling algorithm. Through simulation study the efficiency of proposed job profile based scheduling algorithm in terms of the waiting time and cost of resource is established through comparative analysis.

## References

[1] Foster, C. Kesselman, J. Nick, and S. Tuecke, "The Physiology of the Grid: An Open Grid Services

Architecture for Distributed Systems Integration", 2002, Open Grid Service
[2] Infrastructure WG, Global Grid Forum.
[3] D. Abramson, R. Buyya, M. Murshed, and Venugopal. Scheduling parameter sweep applications on global Grids: A deadline and budget constrained cost-time optimisation algorithm. International Journal of Software: Practice and
[4] Dalibor Klusáček and Hana Rudová. Alea 2 - Job Scheduling Simulator. In proceedings of the 3rd International
[5] ICST Conference on Simulation Tools and Techniques (SIMUTools 2010), ICST, 2010
[6] A. Takefusa, S. Matsuoka, K. Aida, H. Nakada, and U. Nagashima. Overview of a performance evaluation system for global computing scheduling algorithms. In HPDC '99: Proceedings of the 8th IEEE International Symposium on High Performance Distributed Computing, pages 97 – 104, USA, 1999. IEEE.
[7] Efficient Scheduling Algorithms based on Computational Grid. International Journal of Computer Science and Communication Engineering IJCSCE Special issue on "Emerging Trends in Engineering" ICETIE 2012
[8] J. Blythe et al., Task Scheduling Strategies for Work∞ow-based Applications in Grids, IEEE International Symposium on Cluster Computing and the Grid(CCGrid 2005)
[9] Bimal VO, G.Raju. Performance Analysis of TimeLine Algorithm in Grid Environment using Alea. International Journal of Computer Science System Engineering and Information Technology" (July-Dec 2014) ISSN : 0974-8385