Journal of Computer Science 10 (5): 774-782, 2014
ISSN: 1549-3636
© 2014 Science Publications
doi:10.3844/jcssp.2014.774.782 Published Online 10 (5) 2014 (http://www.thescipub.com/jcs.toc)

QUANTITATIVE EVALUATION OF JOB AND RESOURCES FOR BETTER SELECTION TO IMPROVE MAKESPAN IN GRID SCHEDULING

[1]Krishnamoorthy Natarajan, [2]Asokan Ramasamy and [3]Sangeetha Subramanian

[1]Department of Computer Science and Engineering,
Kongu Engineering College, Perundurai, Erode, Tamil Nadu, India
[2]Kongunadu College of Engineering and Technology, Thottiyam, Tamil Nadu, India
[3]Department of Computer Applications, Kongu Arts and Science College, Perundurai, Erode, Tamil Nadu, India

Received 2013-12-06; Revised 2013-12-17; Accepted 2014-01-04

ABSTRACT

This study presents the Priority based ranking of jobs and resources to improve the Makespan in the grid scheduling problem. Grid environment's effectiveness largely depends on scheduler's effectiveness/efficiency as they act as local resource brokers. The scheduler is responsible to select resources/scheduling jobs so that users/application requirements are met regarding overall execution time (throughput) and the resources use cost. The scheduler selects resources that suit user imposed constraints/conditions like CPU usage, RAM available/disk storage. Resource/Jobs are selected using WPR algorithm which improves in performance like Makespan. Results are compared with Round Robin/Weighted Round Robin algorithms where the proposed method has better performance.

Keywords: Priority, Selection, Ranking, Scheduling, Weighted Round Robin

1. INTRODUCTION

The Grid is a new paradigm to solve problems in engineering, science, industry/commerce. Applications use grid infrastructure to meet computational, storage/other needs. A single site no longer meets all resource needs of today's demanding applications and using distributed resources brings benefits for application users. Deployment of grid systems involves managing heterogeneous, geographically distributed/dynamically available resources.

Grid objective is a coordinated heterogeneous resource used to maximize combined resources performance and increasing cost-effectiveness. Due to resources diverse nature, a grid is a Heterogeneous Computing (HC) system where not all machines suit every task Li and Baker (2005). Some tasks have specific machine requirements, i.e., the need for specific instruction set. Hence to lower overall task execution time and increase system throughput, correct resources should be assigned to each task.

The task of grids scheduling is complicated as many machines, each with a different local policy, are involved. A Meta computer scheduler/grid Meta scheduler is implemented over local job schedulers. It is the Meta scheduler's responsibility to schedule jobs to local schedulers which then schedule jobs based on local scheduling policy.

A grid scheduling system is divided into three parts: A scheduling policy, an objective function and a scheduling algorithm Magoules *et al*. (2009)

The scheduling policy is defined by the owner/administrator of the machine/organisations owning

Corresponding Author: Krishnamoorthy Natarajan, Department of Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode, Tamil Nadu, India

the machine. It includes a collection of rules to define resource allocation for jobs submitted to a machine, i.e., a scheduling policy, in an organization, may provide jobs from department A more priority, than jobs from department B. So if jobs from both departments are submitted simultaneously, job from department A is scheduled ahead of that from department B.

Objective function provides a numerical value to schedule and selects a schedule from more than one possibilities. Usually, an objective function has more than one parameter, which the scheduling system aims to maximize/minimize.

Scheduling algorithms are the heart of scheduling systems. A good scheduling algorithm produces a near optimal schedule regarding the chosen objective function and does not require too much resource/time for execution.

Job/resources are two main components in scheduling; in grid scheduling job/resource represents as follows.

1.1. Job

A computational activity made up of various resource requirements (CPU, software libraries, nodes number and memory) and constraints, expressed in job description. Also, in a simple case, a job will have one task/numerous tasks requiring varied processing capabilities.

1.2. Resource

It is a computational entity (computational device/service) where jobs, tasks/applications are scheduled, allocated/processed. Resources comprise their own characteristics like CPU characteristics, memory and software. Some parameters are resource linked, between them processing speed/workload, which transforms with time. Resources can also belong to different administrative domains, involving different policies on access/usage.

In addition to scheduling algorithms type used, applications nature also affects scheduling results and must be considered during scheduling. Generally, applications are divided into two basic classes, data-intensive and computation-intensive. Data-intensive applications dedicate most operation time to access data Wong *et al*. (2004) while computation-intensive applications devote most operation time to compute/process data Xhafa and Abraham (2010) almost no application belongs to either of the two groups specifically; nevertheless it needs data/computational resources proportionally for execution. Each application is both data/computation-intensive but the ratio between both differs with applications.

Grid is a large-scale, heterogeneous, dynamic independent systems collection, geographically distributed/interconnected with high speed networks. Resource allocation in grids, allocates user jobs to CPUs. Jobs are divided into tasks allocated to various computers on grid for execution. Resource allocation is a critical grid technology feature. It was found that resource heterogeneity impacts resource allocation quite significantly regarding performance, reliability, robustness/scalability.

A heterogeneous grid infrastructure is a dynamic environment where elements location, type/performance constantly changes, i.e., a component resource can be put into/pulled out from a grid any time. Resources may not be totally dedicated to such environments and hence a system’s computational capabilities varies over time Abba *et al*. (2012).

Each site in a Grid has own scheduling policy. Certain jobs have higher priority on specific resources. For example, local jobs will be given higher priority so that local jobs are better served on local resources.

Resource management/scheduling systems for Grid computing manage resources/application execution based on resource consumers’/owners’ requirements and need to continuously adapt to changes in resources availability requiring introducing many challenging issues needing addressing like heterogeneous substrate, site autonomy, online control, policy extensibility, resource allocation or co-allocation, resource trading and QoS-based scheduling. Grid resource manager provides functionality to discover/publish resources and job scheduling/submission/monitoring. But, computing resources are geographically distributed under varied ownership each with own access policy, cost/constraints.

Most scheduling algorithms concentrate on resource centric/job centric. To overcome this new algorithm is proposed as Weighted Priority Based Ranking (WRP). In the new algorithm resources are given weightage by considering CPU/RAM and total weight assigned to CPU/RAM is 10 and for Job both user priority/system priority are considered and Job location sum is calculated with this information. Then a High priority Job is allocated to high weighted resource.

The rest of paper is organized as follows. The section 1.1 describes the related work of solving the grid scheduling problem. The method to solve problem is presented in section 2. In the section 3, problem solutions are presented and the section 4 concludes and describes some future work.

1.3. Related Work

A priority based multiple queue scheduling algorithm for grid was proposed by Singh and Kaur (2008). Priority based multiple queue approach solves issues in choosing the best job sequence combination. This increases scheduler performance and in turn Grid environment. Priority based multiple queue scheduling algorithm uses first come first serve, shortest job first, round robin scheduling to locate a best combination for job sequence. Priority based multiple queues, has 3 queues, each with its own algorithm for job arrangement in respective queue. First Come First Serve in first queue, Shortest Job First in second and Round Robin in last queue (FCFS -> SJF -> RR).

Kayande and Shrawankar (2011) proposed priority based pre-emptive task scheduling which involves interrupting low priority tasks when high priority tasks are in queue. This scheduling is used for mobile operating system as CPU utilization is medium, turnaround time/response time is high. SMS categorization is achieved by redirecting them to Priority Inbox.

Azmi and Bakar (2011) stated that Priority rules also referred as Queue-based. Instead of guaranteeing optimal solution, such techniques find solutions in a short time. Though it is a suboptimal algorithm, it is still frequently used to solve scheduling problem in real world due to ease of implementation and low time complexity. This study used six priority rules algorithms Earliest Deadline First (EDF), First Come First Serve (FCFS), Shortest Job First (SJF), Earliest Release Date (ERD), Longest Job First (LJF) and Minimum Time to Due Date (MTTD).

Soni (2010) proposed Grouping-Based Job Scheduling Model in Grid Computing where it is a Memory based Grouping Job Scheduling strategy. Jobs are grouped according to resource capability. This maximizes Grid resources use, reduces jobs processing time/network delay to schedule/execute grid jobs.

Selvarani and Sadhasivam (2010) suggested that scheduling approach tasks be grouped/allocated non-uniformly. Resource processing capability percentage on total processing capability of all resources is calculated. Using the percentage, resource’s processing capability based on total length of tasks to be scheduled is calculated. This approach, due to job grouping optimizes computation/communication ratio and increases resource use. For effective resource use and to distribute jobs to available resources, resource processing capability on total processing capability of all resources is calculated.

A dynamic priority scheduler for advanced reservation in grid computing was proposed by Ahuja *et al.* (2009). The concepts consist of two components DPSAR/Advance Resource Reservation (ARM). DPSAR does job scheduling by resolving job priorities dynamically while ARM handles job reservation scheduled by DPSAR.

A Guest-Aware Priority-Based Virtual Machine Scheduling for Highly Consolidated Server was proposed by Kim *et al.* (2008). The suggested scheduling scheme selects next task to be scheduled based on task priorities and I/O usage status of virtual machines.

Al-Khateeb *et al*. (2012) stated that primary meta-scheduling problem was selecting best resources (sites) to execute underlying jobs while achieving objectives: Reducing mean job turnaround time, ensuring site load balance and considering job priorities. A user’s priority is considered to indicate user’s requirements. Job scheduling submits jobs with higher priorities before those with lower priorities. High priority jobs have potential to access more powerful resources in this policy.

A new model that assigns priority of each user level jobs was proposed by Datta and Banerjee (2012). Jobs are submitted to Grid Broker (GLO) which lists them based on priority and sends them to Local Broker Manager (L) for allotment of job resources.

Kirubanand and Palaniammal (2011) study mainly focuses on M/M (a,b)/1 markovian model with adaboost algorithm and user selection algorithms to find performance on wired and wireless technologies in terms of service rate, arrival rate, Expected waiting time and Busy period.

Lee *et al*. (2011) dealt with scoring of computing resources among clusters. An adaptive scoring method is used to schedule jobs in grid environment in the suggested system. ASJS selects fittest resource for job execution according to resource status. High computing power cluster selected among various clusters and appropriate resources in selected cluster are identified to submit jobs using average transmission power.

Nojabaei *et al*. (2012) prposed a method allows data (time stamp, time action, priority) of jobs on different scales to be compared by bringing them to a common scale. Secondly, the jobs should be arranged based on three criteria which are priority, time action and time stamp. This sorting algorithm is programmed via MATLAB Distributed Computing Server (DCS) software.

2. MATERIALS AND METHODS

The above works are either job centric of Resource centric which does not improve the overall completion time of the application. So a new approach that takes account on both a side is proposed. **Figure 1** shows the grid scheduling architecture.

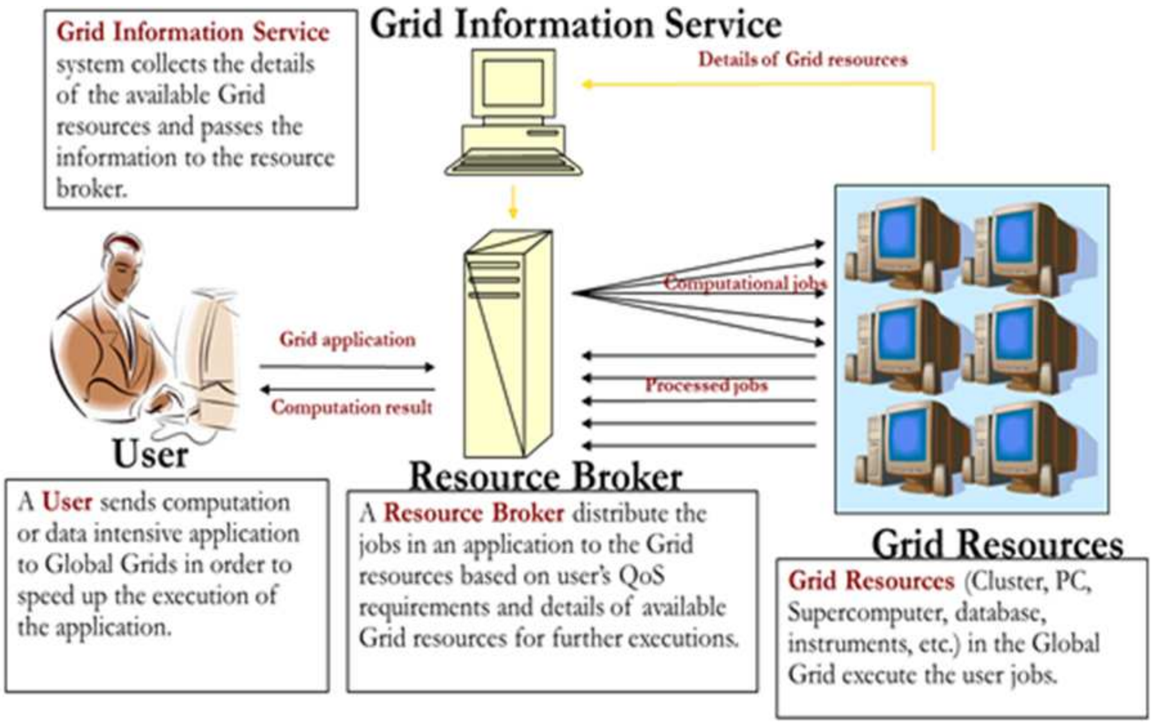
Grid Information Service
Grid Information Service system collects the details of the available Grid resources and passes the information to the resource broker.
Details of Grid resources
Computational jobs
Grid application
Computation result
Processed jobs
User
Resource Broker
Grid Resources
A User sends computation or data intensive application to Global Grids in order to speed up the execution of the application.
A Resource Broker distribute the jobs in an application to the Grid resources based on user's QoS requirements and details of available Grid resources for further executions.
Grid Resources (Cluster, PC, Supercomputer, database, instruments, etc.) in the Global Grid execute the user jobs.

Fig. 1. Grid scheduling architecture

2.1. Weighted Priority Based Ranking Algorithm

There many scheduler algorithms in use which decides the order of execution when there are many processes in a queue. The schedulers are either based on preemptive or non-preemptive technique. In preemptive methods, once the jobs are given to the CPU, the scheduler can interrupt it whereas in non-preemptive the jobs cannot be interrupted. Various well known CPU scheduling algorithms are First Come First Serve (FCFS), Shortest Job First (SJF) and Priority scheduling (Li and Baker, 2005) all of which are non-pre emptive and unsuitable for time sharing systems. In FCFS, jobs are executed in the arrival order. In SJF, the job with least expected completion time is executed first. Shortest Remaining Time First (SRTF) and Round Robin (RR) are pre-emptive in nature with RR being highly suitable for time sharing systems.

Round Robin (RR) algorithm overcomes this by assigning time intervals called quantum to jobs when they are run. If a job is incomplete during a quantum it reverts back to the queue awaiting the next round Soni (2010). The only challenge with this algorithm is finding a suitable quantum length. Round Robin Algorithm drawbacks are that it gives equal time to all processes (processes are scheduled in a first come first serve manner) as Round Robin Algorithm drawbacks ensure it is inefficient for processes with smaller CPU bursts leading to increased waiting and response times thereby lowering system throughput.

The grid scheduling architecture has differnet components like Broker, Information Serviceand scheduler. Scheduler schedules jobs and resources based on the information provided by the broker as shown in **Fig. 1**. The proposed algorithm eliminates drawbacks of round robin algorithm implementation by scheduling processes through weight assignment. The architecture of the proposed system is shown in **Fig. 3**. The proposed Weighted Round Robin algorithm depends on:

- Number of hops from task allocating server to job performing cluster
- Average bandwidth between allocation server and cluster

Weighted round robin algorithm's performance is compared to simple RR for specific resource cluster number and varying tasks number. A new method that evaluates both Job and Resources is proposed. The Scheduling architecture and activity is represented in **Fig. 2**.

First Jobs and priority are received from the user and information about the Resources from the grid information service; with this information perform a Quantitative analysis of job and resources for better paring to improve the overall turnaround time. The Priority by the user and System are considered, the system priority is calculated by the Shortest Job First and First come First Served with that Common Location sum is calculated. With the common Location Sum value is ranked. Similarly the CPU and RAM weightage Resources are ranked.

2.2. Resource

For a given Job resource Selection process is to choose the best Resources from the R Selected List. A good algorithm is needed to choose the best resource, Random Selection may work but it is not an ideal resource selection policy.

The algorithm should take it into the current state of the resource and choose the best one on the basis of Quantitative evaluation. The Algorithm only takes CPU and RAM in to the account. The total weight of the algorithm is 10 Where the CPU Weight is 6 and the RAM weight is 4. The minimum CPU speed is 1 GHz and Minimum RAM size is 256 MB (**Table 1**).

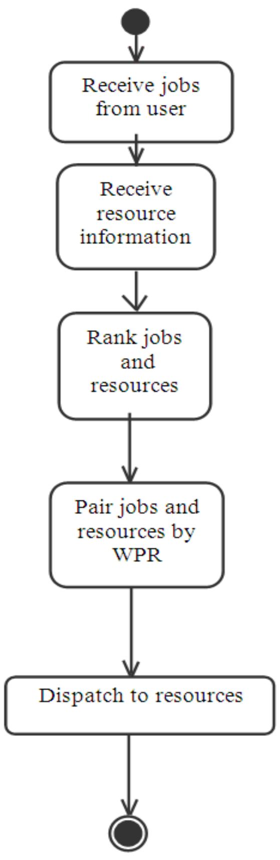

Fig. 2. Scheduling activity

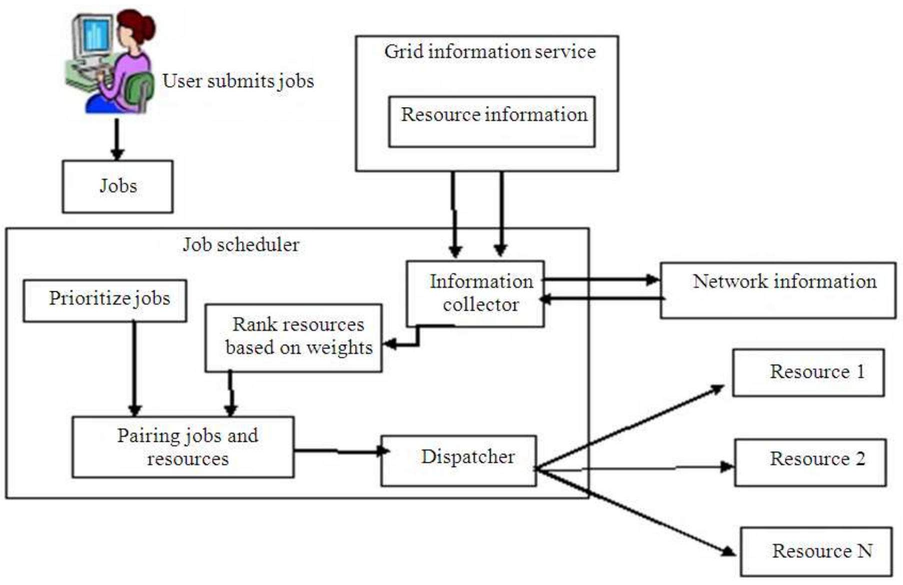

Fig. 3. Proposed architecture

Table 1. The resource information matrix giving resource capabilities

Resource	CPU speed (GHz)	CPU load (%)	RAM size (MB)	RAM usage (%)
Resource1	3.6	60	1024	60
Resource2	5.2	80	256	70
Resource3	2.4	50	512	40
Resource4	1.5	40	256	50
Resource5	2.2	70	512	80
Resource6	1.0	50	1024	50
Resource7	1.8	50	256	50
Resource8	2.6	70	512	60
Resource9	1.2	40	512	30
Resource10	1.0	50	256	50

Table 2. Quantitative evaluation of Resource and its ranking.

Resources	Evaluation values	Ranking based on weights evaluated
R1	1.504	1
R2	0.744	6
R3	1.200	2
R4	0.740	7
R5	0.556	9
R6	1.100	3
R7	0.740	8
R8	0.788	5
R9	0.992	4
R10	0.440	10

Table 2 shows the quantitative evaluation and its ranking. Resources are evaluated on the total weight 10 and it is ranked on the weights obtained Equation (1 to 3):

$$\text{Evaluation}_{\text{Resource}} = \frac{\text{Evaluation}_{\text{CPU}} + \text{Evaluation}_{\text{RAM}}}{\text{W}_{\text{CPU}} + \text{W}_{\text{RAM}}} \quad (1)$$

$$\text{EvaluationCPU} = \text{W}^{*}_{\text{CPU}}\left(1 - \text{CPU}_{\text{load}}\right) * \frac{\text{CPU}_{\text{Speed}}}{\text{CPU}_{\text{min}}} \quad (2)$$

$$\text{EvaluationRAM} = \text{W}^{*}_{\text{RAM}}\left(1 - \text{RAM}_{\text{usage}}\right) * \frac{\text{RAM}_{\text{size}}}{\text{RAM}_{\text{min}}} \quad (3)$$

Where:

W_{CPU}	=	The weight allocated to CPU speed
CPU_{load}	=	The current CPU load
CPU_{speed}	=	Real CPU speed
CPU_{min}	=	Minimum CPU speed
W_{RAM}	=	The weight allocated to RAM
RAM_{usage}	=	The current RAM usage
RAM_{size}	=	Original RAM size and
RAM_{min}	=	Minimum RAM size Equation (4)

$$\text{Evaluation}_{\text{Resource1}} = \frac{8.64 + 6.4}{10} = 1.504 \quad (4)$$

2.3. Job

For the job side, both the user and system priority is taken into account. By using those priorities, the common location sum is calculated for the jobs. Then high priority jobs can be assigned to the first ranked site to achieve the minimum turnaround time for the completion of the job with the available resources. The following **Table 3** shows the Priority wise Ranking of Jobs.

In the system, consider the user priorities and the system priorities at the same time to obtain the benefits of the queuing criteria. Each queuing criterion is sorted in ascending order; the highest priority job is the first location priority. Therefore, according to the user priority criterion, J2 has the highest priority and is assigned location 1. According to SJF, J5 has the highest priority. According to FIFO, J9 has the highest priority. The job location sum is the total sum of the job location scores for each criterion. For example, J2 has location 1 according to the user priority criterion and it has location 3 by the SJF criterion and location 5 by the FIFO criterion. All of the jobs are treated in the same way. The last column in **Table 3** is the job priority, which sorted in descending order according to the job location sum, thus assigning the highest job priority to the job with the lowest job location sum.

Table 4 shows a sample for 10 jobs and 10 resources to assign weights and priority Ranking. This helps in better pairing of Jobs and Resources.

2.4. Experimental Setup

Simulations were carried out in Simgrid framework. The Number of node clusters taken is 5, No of jobs used in the simulation are 100, 200, 300, 400, 500, Jobs are uniform size, job failure probability -ρ is 0.2 scheduling schemes used are Round Robin, Weighted Round Robin and Weighted Priority Based Ranking. To determine performance quality of scheduling the execution time metric is considered. Execution time is the time required to run a job on a resource. The scheduler aims to choose a resource leading to least execution time. **Table 5** tabulates the simulation results for time taken to execute Varied Number of Tasks.

Table 3. Priority wise ranking of jobs

	The queuing criteria				
Job Location	User Priority	SJF	FIFO	Job Name	Job location Sum
1	J2	J5	J9	J1	6+ 9 + 9 = 24
2	J5	J10	J3	J2	1+ 3 + 5 = 9
3	J7	J2	J6	J3	4+ 8 + 2 = 14
4	J3	J6	J4	J4	7+ 6 + 4 = 17
5	J8	J9	J2	J5	2+ 1 + 7 = 10
6	J1	J4	J7	J6	10+ 4 + 3 = 17
7	J4	J8	J5	J7	3+ 10 + 6 = 19
8	J9	J3	J10	J8	5+ 7 + 10 = 22
9	J10	J1	J1	J9	8 + 5 + 1 = 14
10	J6	J7	J8	J10	9+ 2 + 8 = 19

Table 4. Pairing of jobs and resource based on the ranks and weights

Resource based on the weights	Jobs based on the priority based ranks
R1	J2
R6	J5
R2	J9
R7	J3
R9	J4
R3	J6
R8	J7
R5	J10
R4	J8
R10	J1

Table 5. Execution time achieved

Number of tasks	Round Robin (RR)	Weighted Round Robin (WRR)	Weighted Priority Ranking (WPR)
10	2.41724	2.41724	1.82318
50	19.79920	17.45030	16.22010
100	41.11520	37.59150	34.24070
150	64.78020	54.68290	50.34150
200	87.00250	74.82480	62.41240
250	106.70900	91.52230	83.21260
300	128.02500	112.05800	103.42800
350	151.69000	129.14900	115.28500
400	173.91200	148.90000	132.56000
500	214.93500	185.35000	157.61000

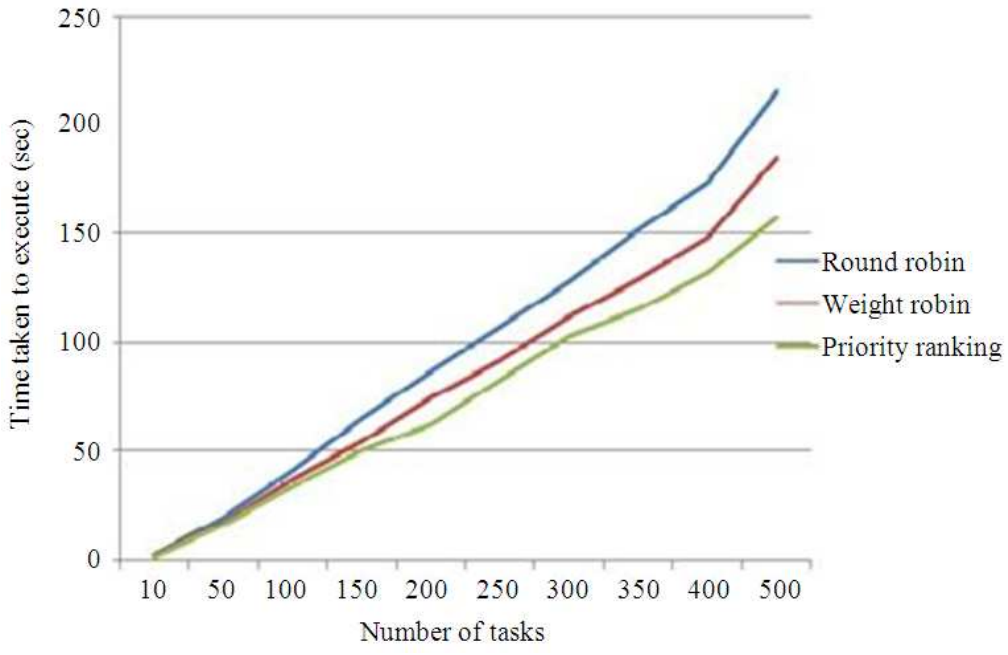

Fig. 4. Execution time

3. RESULTS AND DISCUSSION

The Graph depicted in **Fig. 4** demonstrates the proposed weighted Priority based Ranking algorithm has the less execution time while comparing with the Round Robin and Weighted Round Robin scheduling algorithm.

It is observed from **Fig. 4** that the proposed weighted Priority based ranking algorithm achieves 16.72% to 26.67% decrease in execution time when compared with Round robin scheduling. When compared with weighted round robin, the proposed method decreases execution time by 7.05% to 24.58% for varying number of tasks.

4. DISCUSSION

A Grid environment is a potential complex globally distributed system involving large sets of diverse, geographically distributed components for many applications. Grid system scheduling decisions are based on mapping best resources with jobs. Grid performance can be improved by ensuring all available Grid resources are used optimally through a good scheduling algorithm. Simulation results show that the proposed weighted Priority based Ranking algorithm achieves 16.72% to 26.67% decrease in execution time when compared with Round robin scheduling.

Future work can be concentrated, after pairing resources/jobs by WPR. Status of every individual resource should be considered for updated ranking of Jobs and updated weights for resources from local/global updates in Information Service.

5. REFERENCES

Abba, H.A., N.B. Zakaria and N. Haron, 2012. Grid resource allocation: A review research. Res. J. Inform. Technol., 4: 38-55.

Ahuja, R., G. Gabrani and A. De, 2009. A dynamic priority scheduler for advance reservation in grid computing. Int. J. Soft Comput., 4: 60-67.

Al-Khateeb, A., N.A. Rashid and R. Abdullah, 2012. An enhanced meta-scheduling system for grid computing that considers the job type and priority. Computing, 94: 389-410. DOI 10.1007/s00607-011-0168-6

Azmi, Z.R.M., K.A. Bakar, A.H. Abdullah, M.S. Shamsir and W.N.W. Manan, 2011. Performance comparison of priority rule scheduling algorithms using different inter arrival time jobs in grid environment. Int. J. Grid Distrib. Comput., 4: 61-70.

Datta, A. and I. Banerjee, 2012. Priority based mathematical modelling for grid computing environment. Undergraduate Acad. Res. J., 1: 112-117.

Kayande, D. and U. Shrawankar, 2011. Priority based pre-emptive task scheduling for android operating system. Int. J. Comput. Sci. Telecommun., 2: 17-21.

Kim, D., H. Kim and M. Jeon, 2008. Guest-aware priority-based virtual machine scheduling for highly consolidated server. Proceedings of the 14th International Euro-Par Conference on Parallel Processing, Aug. 26-29, Springer Berlin Heidelberg, Spain, pp: 285-294. DOI: 10.1007/978-3-540-85451-7_31

Kirubanand, V.B. and S. Palaniammal, 2011. Study of performance analysis in wired and wireless network. Am. J. Applied Sci., 8: 826-832. DOI: 10.3844/ajassp.2011.826.832

Lee, Y.H., S. Leu and R.S. Chang, 2011. Improving job scheduling algorithms in a grid environment. Future Generat. Comput. Syst., 27: 991-998. DOI: 10.1016/j.future.2011.05.014

Li, M. and M. Baker, 2005. The Grid: Core Technologies. 1st Edn., John Wiley and Sons, ISBN-10: 0470094176, pp: 452.

Magoules, F., J. Pan, K.A. Tan and A. Kumar, 2009. Introduction to Grid Computing: Chapman and Hall CRC Numerical Analysis and Scientific Computing. 1st Edn., CRC Press, ISBN-10: 1420074075, pp: 334.

Nojabaei, S., Z. Leman, S.H. Tang and S. Sulaiman, 2012. Development of priority oriented scheduling method to increase the reliability of manufacturing systems. Am. J Applied Sci., 9: 1435-1442. 10.3844/ajassp.2012.1435.1442

Pinedo, M. L., 2012. Scheduling: Theory, Algorithms and Systems. 1st Edn., Springer, New York, ISBN-10: 1461423619, pp: 693.

Selvarani, S. and G. Sudha, 2010. Sadhasivam improved job-grouping based PSO algorithm for task scheduling in grid computing. Int. J. Eng. Sci. Technol., 2: 4687-4695.

Singh, S.K. and D. Kaur, 2008. Priority based multiple queue scheduling algorithm for grid. Proceedings of the 2nd IEEE Asia-Pacific, Service Computing Conference, Dec. 9-12, Jiaosi, Yilan Taiwan.

Soni, V.K., 2010. Grouping-based job scheduling model in grid computing. World Acad. Sci. Eng. Technol., 41: 781-784.

Wong, H.M., V. Bharadwaj, Y. Dantong, T.G. Robertazzi, 2004. Data intensive grid scheduling: Multiple sources with capacity constraints. Proceedings of the 15th International Conference on Parallel and Distributed Computing Systems, (PDCS’ 04), IEEE Xplore Press, Cambridge, MA, USA., pp: 163-170.

Xhafa, F. and A. Abraham, 2010. Computational models and heuristic methods for grid scheduling problems. Future Generat. Comput. Syst., 26: 608-621. DOI: 10.1016/j.future.2009.11.005