International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 7, July - 2013

Performance Analysis Of Scheduling Algorithms In Grid Computing

Navjot Kaur
Punjabi University Regional Center of Information Technology and Manegement, Mohali, India

Nivit Gill
Assistant Professor,Department of of Computer Science

1. Abstract

The aim of grid computing is to aggregate the power of widely distributed resources, and provide non-trivial services to users. To achieve this goal, an efficient grid scheduling system is an essential part of the grid. Scheduling in grid poses a number of new challenges. Grid scheduling algorithms can be classified in number of ways, such as static vs. dynamic policies, or on the basis of objective functions, or based on QoS constraints [3]. In this paper, static algorithms are compared to that of dynamic scheduling algorithms. Under static category, two algorithms are implemented namely: max-min and min-min, while under dynamic category, an algorithm implemented is: dynamic time quantum scheduling algorithm. The efficiency of these algorithms is analyzed based on two performance parameters namely average waiting time and average turnaround time.

2. Introduction

Grid is a type of parallel and distributed computing, where a collection of interconnected standalone computers work together as a single, integrated computing resource to solve problems. The goal is to create the illusion of a simple yet large and powerful self managing virtual computer out of a large collection of connected heterogeneous systems sharing various combinations of resources [6].

To make effective use of the tremendous capabilities of the grids, efficient task scheduling algorithms are required. The arrangement of a number of related operations in time is called scheduling. Grid scheduling is defined as the process of making scheduling decisions involving resources over multiple administrative domains. This process can include searching multiple administrative domains to use a single machine or scheduling a single job to use multiple resources at a single site or multiple sites. Grid Scheduling is a software framework with which the scheduler collects resource state information, selects appropriate resources, predicts the potential performance for each schedule, and determines the best schedule for the applications to be executed on a Grid System subject to QoS goals [4]. Grid scheduling algorithms can be classified in number of ways, such as static vs. dynamic policies, or on the basis of objective functions, or based on QoS constraints [3]. The efficiency of these algorithms is analyzed based on various performance parameters like throughput, CPU utilization, average waiting time and average turnaround time, etc. Hierarchical taxonomy for scheduling algorithms in distributed computing is shown in Figure 2.1.

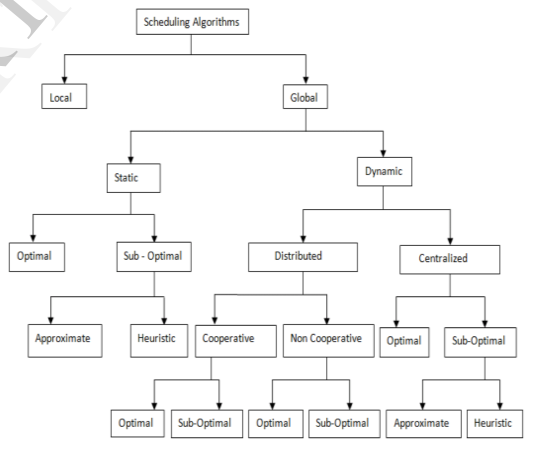

Figure 2.1: A Hierarchical taxonomy for scheduling algorithms [3]

Static and dynamic algorithms that lie under global category of scheduling algorithms are considered. Static algorithms are compared to that of dynamic scheduling algorithms. Under static category, two algorithms are implemented namely: max-min and min-min, while under dynamic category, an algorithm implemented is: dynamic time quantum scheduling algorithm. The efficiency of these algorithms is analyzed based on two performance parameters, namely average waiting time and average turnaround time.

3. Static Vs Dynamic Algorithms

The broad category of algorithms based on which the comparison of algorithms is performed are static and dynamic algorithms.

3.1 Static Algorithms:

In static algorithms, information regarding all the resources as well as the jobs in the grid is assumed to be available by the time that application is scheduled. Once the jobs are scheduled, the scheduling decision can not be changed. Two static algorithms that are implemented are:

3.1.1 Min-min

The min-min algorithm starts with the set of all unmapped tasks. From that, the set of minimum completion time for each task is selected; and the task with the overall minimum completion time from is chosen and is assigned to the corresponding resource. Last, the newly mapped task is removed from the set of tasks, and the process repeats until all tasks are mapped.

```
for all tasks Ti
   for all resources Rj
      Cij=Eij+rj
   do until all tasks are mapped
      for each task find the earliest completion time and the resource that obtains it
        find the task Tk with the minimum earliest completion time
        assign task Tk to the resource Rl that gives the earliest completion time
        delete task Tk from set
        update rl
        update Cil for all I
   end do
```

Figure 3.1: Pseudo code of Min-min scheduling Algorithm

3.1.2 Max-min

The max-min algorithm is almost same as that of min-min algorithm. It also begins with the set of all unmapped tasks. Then, the set of minimum completion time is found. Then, the task with the overall maximum time is selected and assigned to the corresponding resource. Last, the newly mapped task is removed from the set of tasks, and the process repeats until all tasks are mapped.

```
for all tasks Ti
   for all resources Rj
      Cij=Eij+rj
   do until all tasks are mapped
      for each task find the earliest completion time and the resource that obtains it
        find the task Tk with the maximum earliest completion time
        assign task Tk to the resource Rl that gives the earliest completion time
        delete task Tk from set
        update rl
        update Cil for all I
   end do
```

Figure 3.2: Pseudo code of Max-min scheduling Algorithm

3.2 Dynamic Algorithms:

In case of dynamic, jobs are allocated at the time of execution of the application. After allocating the jobs to the resources once, the schedule can be changed afterwards. One dynamic algorithm has been chosen for implementation:

3.2.1 Dynamic Time Quantum

In dynamic time quantum scheduling, same time of processor is allocated to all the processes. After each execution, a new time is allocated to the remaining processes dynamically according to their requirements. In dynamic time quantum, equal priority is assigned to each process but according to the earliest completion time, another time quantum is assigned to the remaining processes. Due to this, each process get the processor time which in turn affects the performance of the application.

```
for all tasks Ti
  for all resources Rj
  Cij=Eij+rj
  Timer=earliest completion time
  do until all tasks are mapped
     for each task find the earliest completion time and the resource that obtains it
     find the task Tk with the maximum earliest completion time
     assign task Tk to the resource Rl that gives the earliest completion time
    delete task Tk from the set of tasks
    update rl
   update Cil for all i
   update timer
end do
```

Figure 3.1: Pseudo code of Dynamic Time Quantum scheduling Algorithm

3.3 Performance Analysis Parameters

These scheduling algorithms are evaluated on the basis of following performance parameters.

3.3.1 Average Waiting time

Waiting time can be defined as the amount of time that a process spends waiting to get a resource. Or we can say that waiting time is the sum of the periods spent waiting in the ready queue. Average waiting time is therefore, the sum of all the waiting times calculated for all the processes.

3.3.2 Average Turnaround time

Turnaround time is defined as the interval of time from the submission of a process to the time of its completion. Hence, average turnaround time is the sum of the turnaround times calculated for all the processes.

4. Simulation

The algorithms have been implemented using GridSim Toolkit 5.2 and NetBeans IDE 6.9.

The experimentation includes nine gridlets on three machines and three users. Each machine has two resources but different number of gridlets. The description of these three machines is given below:

Machine 1: 4 gridlets, 2 resources

Machine 2: 3 gridlets, 2 resources

Machine 3: 2 gridlets, 2 resources

4.1 Machine 1

Table 4.1 shows the description of completion time of gridlets each both the resources, i.e. R1 and R2.

Table 4.1 Description of completion time of gridlets with R1 and R2 for each simulation at Machine 1

Gridlets	Simulation 1		Simulation 2		Simulation 3		Simulation 4		Simulation 5	
	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2
G0	5	13	10	2	4	14	3	7	20	25
G1	7	16	15	4	5	18	7	14	20	29
G2	9	21	18	15	7	19	19	28	7	18
G3	18	24	26	24	14	27	19	29	18	21

4.2 Machine 2

Table 4.2 gives the description about three gridlets and their completion time with resources R1 and R2 respectively, for all the simulations at Machine 2.

Table 4.2 Description of completion time of gridlets with R1 and R2 for each simulation at Machine 2

Gridlets	Simulation 1		Simulation 2		Simulation 3		Simulation 4		Simulation 5	
	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2
G0	5	8	1	1	17	1	6	5	8	3
G1	22	9	22	15	25	12	17	11	13	8
G2	28	28	26	29	28	25	23	22	2	20

4.3 Machine 3

Table 4.3 gives the description about two gridlets and their completion time with resources R1 and R2 respectively, for all the simulations at Machine 3.

Table 4.3 Description of completion time of gridlets with R1 and R2 for each simulation at Machine 3

Gridlets	Simulation 1		Simulation 2		Simulation 3		Simulation 4		Simulation 5	
	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2
G0	24	10	10	15	26	8	17	4	22	27
G1	28	24	13	25	28	21	17	12	5	19

5. Results and Discussions

The results of the simulations of the implemented algorithms are presented and evaluated in this section. The comparison is done on the basis of two performance parameters, i.e. average waiting time and average turnaround time and both the parameters are calculated at each machine separately for each algorithm.

5.1 Machine 1: Results

Table 5.1 shows the comparison of average waiting times for the three algorithms at Machine 1. Table 5.2 shows the comparison of average turnaround time for the algorithms.

Table 5.1 Average waiting time at Machine 1

Algorithms	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
Dynamic Time Quantum	6	6	4	9	16
Min-Min	9	7	7	10	19
Max-Min	19	26	15	25	26

Table 5.2 Average turnaround time at Machine 1

Algorithms	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
Dynamic Time Quantum	23	20	18	25	40
Min-Min	19	18	14	22	35
Max-Min	29	37	22	37	43

Figure 5.1 and figure 5.2 represents the comparison of average waiting time and average turnaround time, respectively, of dynamic time quantum scheduling algorithm, max-min scheduling algorithm and min-min scheduling algorithm at Machine 1 graphically.

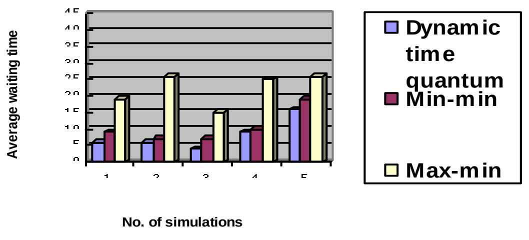

Figure 5.1 Comparison of average waiting time at Machine 1

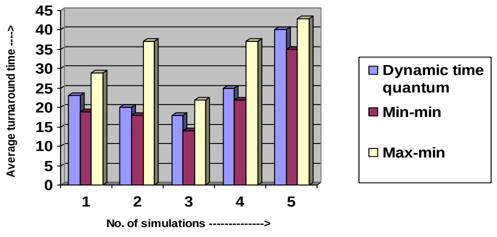

Figure 5.2 Comparison of average turnaround time at Machine 1

5.2 Machine 2: Results

Table 5.3 shows the comparison of average waiting times for the three algorithms at Machine 2 and table 5.4 shows the comparison of average turnaround time for the algorithms at Machine 2.

Table 5.3 Average waiting time at Machine 2

Algorithms	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
Dynamic Time Quantum	3	5	4	3	6
Min-Min	8	6	5	7	4
Max-Min	21	24	20	18	16

Table 5.4 Comparison of Average waiting time at Machine 2

Algorithms	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
Dynamic Time Quantum	30	21	17	21	23
Min-Min	23	20	16	19	15
Max-Min	36	39	33	31	36

Figure 5.3 shows the comparison of average waiting time and figure 5.4 shows the comparison of average turnaround time for dynamic time quantum scheduling algorithm, max-min scheduling algorithm and min-min scheduling algorithm at Machine 2 graphically.

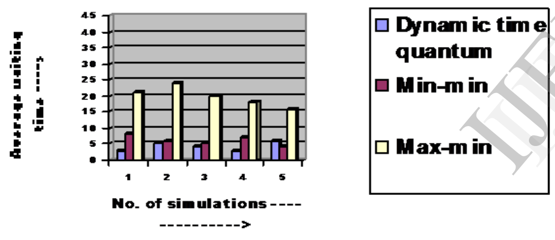

Figure 5.3 Comparison of average waiting time at Machine 2

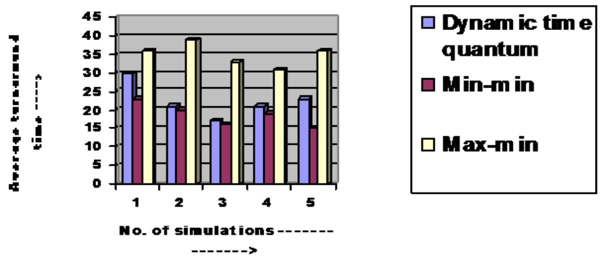

Figure 5.4 Comparison of average turnaround time at Machine 2

5.3 Machine 3: Results

Table 5.5 and table 5.6 shows the comparison of average waiting times and average turnaround time, respectively, for the three algorithms at Machine 3.

Table 5.5 Average waiting time at Machine 3

Algorithms	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
Dynamic Time Quantum	5	5	4	4	4
Min-Min	5	7	4	2	9
Max-Min	12	12	10	6	13

Table 5.6 Average waiting time at Machine 3

Algorithms	Simulation 1	Simulation 2	Simulation 3	Simulation 4	Simulation 5
Dynamic Time Quantum	23	27	18	12	34
Min-Min	22	16	17	10	32
Max-Min	29	32	25	14	36

Figure 5.5 and figure 5.6 shows the comparison of average waiting time and average turnaround time, respectively, for dynamic time quantum scheduling algorithm, max-min scheduling algorithm and min-min scheduling algorithm at Machine 3 graphically.

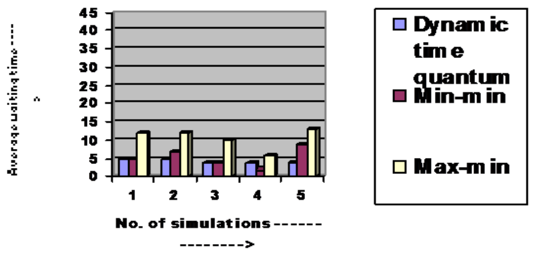

Figure 5.5 Comparison of average waiting time at Machine 3

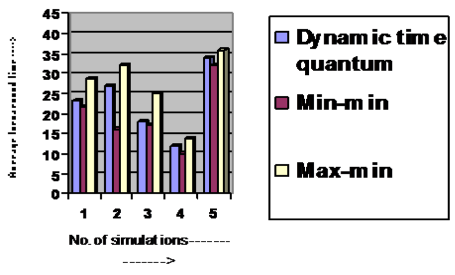

Figure 5.6 Comparison of average turnaround time at Machine 3

5.4 Discussions

Simulation results clearly show that the average waiting time for dynamic time quantum scheduling algorithm is less than both the static algorithms. This difference is due to the reason that in case of min-min and max-min scheduling algorithm, once a job is scheduled to a resource, the resource gets free only after the job is completed and other jobs remain waiting till that time. In case of dynamic time quantum scheduling algorithm, the timer gets changed after each round trip; and hence each job gets the resource for a particular time span and the waiting time in case of dynamic algorithm is less. So, dynamic time quantum scheduling algorithm is better than both the min-min and max-min scheduling algorithms.

On the basis of average turnaround time, min-min scheduling algorithm outperforms dynamic time quantum scheduling algorithm and max-min scheduling algorithm. Since, in dynamic time quantum scheduling algorithm, the jobs need to pre-empt the resources again and again after the completion of their allotted time, the time span between the submission and the completion of the jobs increases and hence, the turnaround time of jobs also increases.

Further, under static category, min-min scheduling algorithm is better than max-min scheduling algorithm as the average waiting time for min-min scheduling algorithm is lesser than max-min scheduling algorithm. This is due to the reason that according to the max-min scheduling algorithm, the longest jobs are scheduled first, followed by the jobs with lesser completion time. Hence, the jobs having highest completion time gets the resource first; and hence the other jobs need to wait for longer time to get the resource.

The performance of max-min scheduling algorithm is the worst out of three algorithms under study, both in terms of average turnaround time and average waiting time. Dynamic time quantum scheduling is the best in terms of average waiting time and min-min scheduling algorithm is best in terms of average turnaround time.

6. Conclusion

Grid computing has progressed a lot but still the areas like resource management and resource scheduling have many challenges that need to be addressed. The research focuses primarily on comparison of static and dynamic grid scheduling algorithms. These algorithms are implemented and successfully simulated on top of open source GridSim toolkit version 5.2.

According to the simulation results, dynamic time quantum scheduling algorithm results in lowest average waiting time than those of max-min and min-min scheduling algorithms. Average turnaround time for min-min scheduling algorithm is the lowest followed by dynamic time quantum and max-min scheduling algorithms. Min-min scheduling algorithm is better than max-min scheduling, both in terms of average waiting time and average turnaround time.

Dynamic time quantum scheduling algorithm is more optimal and feasible in terms of average waiting time of jobs while min-min scheduling algorithm is better in terms of average turnaround time.

7. References

[1] C. Kesselman and I. Foster, "The Grid: Blueprint for a New Computing Infrastructure", *Morgan Kaufmann, San Francisco, CA*, pp.677, 1999.

[2] Dr. K Vivekanandan et al, "A Study on Scheduling in Grid Environment", *International Journal on Computer Science and Engineering*, pp. 940-950, 2 Feb, 2011.

[3] Fangpeng Dong and Selim G. Akl , "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems", *Queen's University, Kingston, Ontario, School of Computing*, Technical Report No. 2006-504, January 2006.

[4] L. Ferreira, et al, "Introduction to Grid Computing with Globus", Redbook IBM Corporation, *http://www.redbooks.ibm.com/redbooks/pdfs/sg246895.pdf*

[5] Saeed Parsa and Reza Entezari-Maleki, "RASA: A New Task Scheduling Algorithm in Grid Environment", *World Applied Sciences Journal 7 Special Issue of Computer & IT,* 152-160, 2009.

[6] Tanu Gupta, Dr. Inderveer Chana, "Efficient Scheduling Algorithm based on Dynamic Time Quantum for Grid Scheduler", *International Conference on Technology and Information Systems*, 2010.

[7] V. Somasundaram and S Radhakrishnan, "Node Allocation In Grid Computing Using Optimal Resouce Constraint (ORC) Scheduling", *IJCSNS International Journal of Computer Science and Network Security,* VOL.8 No.6, June 2008.