Grid Scheduling Algorithms for Heterogeneous Environment

G.Sumathi,
Sri Venkateswara College of Engineering,
Sirperumbudur - 602 105, India.
sumathiganesan@yahoo.com

N.P.Gopalan
National Institute of Technology,
Tiruchirappalli- 620 015, India.
gopalan@nitt.edu

Abstract

Grids have emerged as paradigms for the next generation parallel and distributed computing. Computational Grid can be defined as large-scale high-performance distributed computing environments that provide access to high-end computational resources. Grid scheduling is the process of scheduling jobs over grid resources. Improving overall system performance with a lower turn around time is an important objective of Grid scheduling. In this paper a new logical framework for the Grid with three different scheduling algorithms Shortest Job Fastest Resource (SJFR), Longest Job Fastest Resource (LJFR) and Priority Based Algorithm are proposed. The SJFR and LJFR algorithms take the computational complexity of the jobs and the speed of the resources into consideration while scheduling the jobs. In the priority based algorithm a new parameter named "priority" has been taken into consideration. This divides the jobs into high, medium and low categories based on priority. Generally, a job, which needs high computational power and exhibits less parallelism is given a high priority. Prioritizing the jobs in this way is found to improve the performance of computational grids. The algorithms are tested and their performance evaluation is studied.

1. Introduction

Computational Grids are emerging as a new computing paradigm for solving grand challenge applications in science, engineering and economics [1]. Computational Grid can be defined as large-scale high-performance distributed computing environments that provide access to high-end computational resources [2]. Each of these resources could be a uni-processor machine, a symmetric multiprocessor cluster, a distributed memory multiprocessor system, or a massively parallel supercomputer. Each resource (node) consists of a number of heterogeneous resources. The resources on the grid are usually accessed via an executing "job".

Grid scheduling is the process of scheduling jobs over grid resources. A grid scheduler is different from local scheduler in that a local scheduler only manages a single site or cluster and usually owns the resource. A grid scheduler is in charge of resource discovery, grid scheduling (resource allocation and job scheduling) and job execution management over multiple administrative domains.

In heterogeneous grid environment with its multitude of resources, a proper scheduling and efficient load balancing across the grid can lead to improved overall system performance and a lower turn-around time for individual jobs. First Come First Serve (FCFS) algorithm neither considers any of the job parameters nor the resource parameters. Shortest Job Fastest Resource (SJFR) and Longest Job Fastest Resource (LJFR) are the proposed algorithms that consider computational complexity of jobs for scheduling and ignore the priority of a job. A scheduling algorithm based on priority of the jobs is proposed and tested. Prioritizing the jobs based on their nature helps in improving the real time performance of computational grids.

The grid framework is discussed in the next section. The scheduling algorithms and their performance

analysis are discussed in Section 3 & 4. Section 5 gives the conclusion.

2. Grid Framework

Fig.1 shows the framework of the grid. The Global and Local Grid Resource Brokers (**GGRB** & **LGRB)** and Grid Information Server (**GIS)** are the three main components of the grid. Each of these components has its own independent functionalities that help in grid management and job scheduling and thus serve the purpose of a grid.

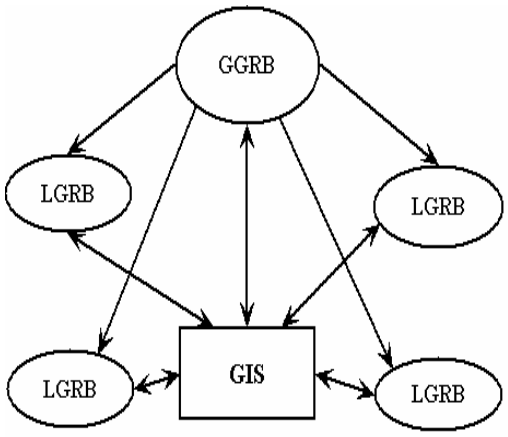

Fig. 1. Frame Work of a Grid

2.1 Local Grid Resource Broker (LGRB)

The local grid resource broker is a synonym for a grid resource. Each grid resource has been categorized based on it's processing speed as follows:

a) Type 1
b) Type 2
c) Type 3

This categorization adds to the heterogeneous nature of a grid. Each LGRB in the grid can be any one of the above three resources. There can be many LGRBs possible in the grid. With the addition of every LGRB, the number of resources and consequently the number of processing elements (**PE**s) are increased. A job submitted to the grid may be migrated to any of the LGRBs in the grid for execution. Once a job has been migrated to a particular LGRB, the LGRB ensures execution of the job on the specified number of processors. Since computational grids have been taken into consideration, the number of processing elements in an LGRB is the actual resource of the grid.

2.2 Global Grid Resource Broker (GGRB)

All the jobs are submitted to the GGRB. A single GGRB takes care of scheduling jobs in the grid based on the resources available as per the scheduling algorithm. Once a task has been scheduled to a particular LGRB, the GGRB migrates the job to that LGRB for execution.

2.3 Grid Information Server (GIS)

The Grid Information Server is the database bank of the grid. It keeps track of the resources available in the grid. Any new LGRB should register itself with the GIS. The GIS provides information regarding free resources to the GGRB based on which the GGRB schedules the jobs.

2.4 Working of the Grid

Registration

Any new LGRB should register itself with the GIS by sending a request. The GIS responds with an acknowledgement, which means that it is ready to accept a new resource as a grid member. Now, it's the LGRBs turn to send the details regarding itself, its type, and number of processing elements and speed of each processing element.

Job Scheduling

The GGRB stores the incoming jobs in a queue. When scheduling is to be done the GGRB requests the GIS with a query for the suitable resources. As soon as the GIS receives a request from the GGRB it sends the IP address of the suitable resource to GGRB, if available. Jobs submitted to the GGRB are migrated to the LGRBs based on a scheduling algorithm for execution.

3. Scheduling Algorithms

A proper scheduling algorithm can lead to an improved overall system performance and a lower turn around time. Since a Grid has heterogeneous resources it is often complex to design an efficient scheduling algorithm. There are different types of scheduling algorithms proposed.

3.1 First Come First Serve (FCFS)

As the name suggests this scheduling algorithm schedules the jobs on a " First come First serve" basis [5]. As soon a job is submitted to the GGRB, the

scheduler searches the Resource Database for an appropriate resource linearly. The job is migrated to that resource for execution. This algorithm neither considers any of the job parameters nor the resource parameters.

3.2 Shortest Job Fastest Resource(SJFR)

Shortest Job Fastest Resource is a scheduling algorithm, which tries to reduce the overall turn around time of the jobs [5]. The shortest job (based on computational complexity) is scheduled to the fastest resource (based on speed of each system available in that node). SJFR algorithm does not consider the priority of the job submitted.

3.3 Longest Job Fastest Resource (LJFR)

Longest Job Fastest Resource is a scheduling algorithm, which tries to reduce the overall execution time of the jobs [5]. The longest job is scheduled to the fastest resource. Since the longest job is submitted to the fastest resource the execution time of the longest job is drastically reduced when compared to its execution time on any other resource in the grid. LJFR does not consider the priority of a job. As far as execution time is considered it gives the best results.

3.4 Priority Based Algorithm

In the priority based algorithm a new parameter named "priority" has been taken into consideration. This divides the jobs into high, medium and low categories based on priority. *Generally a job which needs high computational power and which exhibits less parallelism is given a high priority*. A job, which exhibits high parallelism and needs less computational power, is given a low priority. A job, which exhibits a medium level of parallelism and needs normal computational power, is given a medium priority.
The algorithm is given below:

```
Procedure Schedule (GridletList G_List)
SortGridlets_ByPriority (G_List)
If (G_List = NULL)
        Return
End If
While (G_List ≠ NULL)
    Send a request to GIS for the resource requirements
    having PEs_required and priority as the search
    parameters.

    Resource = SearchForResource (PEs_required,
    Priority)
    If Resource not available
        Add the job to G_List
    Else
        Send the job to the specified Resource
    End If
End While
End Schedule


Procedure SortGridlets_ByPriority (G_List)
While (G_List ≠ NULL)
If (G_List->Gridlet_Object.Priority = 1)
        Add the gridlet to Sorted_List1
Else If (G_List->Gridlet_Object.Priority = 2)
        Add the gridlet to Sorted_List2
Else If (G_List->Gridlet_Object.Priority = 3)
        Add the gridlet to Sorted_List3
End If
End While
Join Sorted_List1, Sorted_List2 and Sorted_List3
End SortGridlets_ByPriority

// PROCEDURE TO BE IMPLEMENTED IN GIS
Procedure SearchForResource (PEs_required, Priority)
If (Priority = 1)
   Query the database for a Resource with
   Resource_Type = "Type 1" and having the required
   PEs in the descending order of speed
   Select the top most tuple

   If a resource is found Return resource Else
      Query the database for a Resource with
      Resource_Type = "Type 2" and having the
      required PEs in the descending order of speed
      Select the top most tuple
      If a resource is found
         Return resource
      Else
         Query the database for a Resource with
         Resource_Type = "Type 3" and having the
         required PEs in the descending order of speed
          Select the topmost tuple
          If a resource is found
             Return resource Else return empty resource
          End If
      End If
   End If
End If
If (Priority = 2)
   Query the database for a Resource with
   Resource_Type = "Type 2" and having the required
   PEs in the descending order of speed
   Select the top most tuple
   If a resource is found
      Return resource
```

```
    Else
        Query the database for a Resource with
        Resource_Type = "Type 3" and having the
        required PEs in the descending order of speed
        Select the top most tuple
        If a resource is found
                Return resource
        Else
            Query the database for a Resource with
            Resource_Type = "Type 1" and having the
            required PEs in the ascending order of speed
            Select the top most tuple
             If a resource is found
                Return resource
            Else
                return empty resource
             End If
        End If
    End If
End If


If (Priority = 3)
    Query the database for a Resource with
    Resource_Type = "Type 3" and having the
    required PEs in the descending order of speed
    Select the top most tuple
     If a resource is found
        Return resource
     Else
    Query the database for a Resource with
Resource_Type = "Type 2" and having the required
PEs in
    the ascending order of speed
    Select the top most tuple
            If a resource is found
                Return resource
            Else
                Query the database for a Resource with
                Resource_Type = "Type 1" and having
                the required PEs in the ascending order of
                speed
                Select the top most tuple
                If a resource is found
                    Return resource
                else
                    return empty resource
                End If
        End If
     End If
End If
Send the Resource IP address to GGRB
End SearchForResource
```

SJFR	LJFR	FCFS	PRIORITY

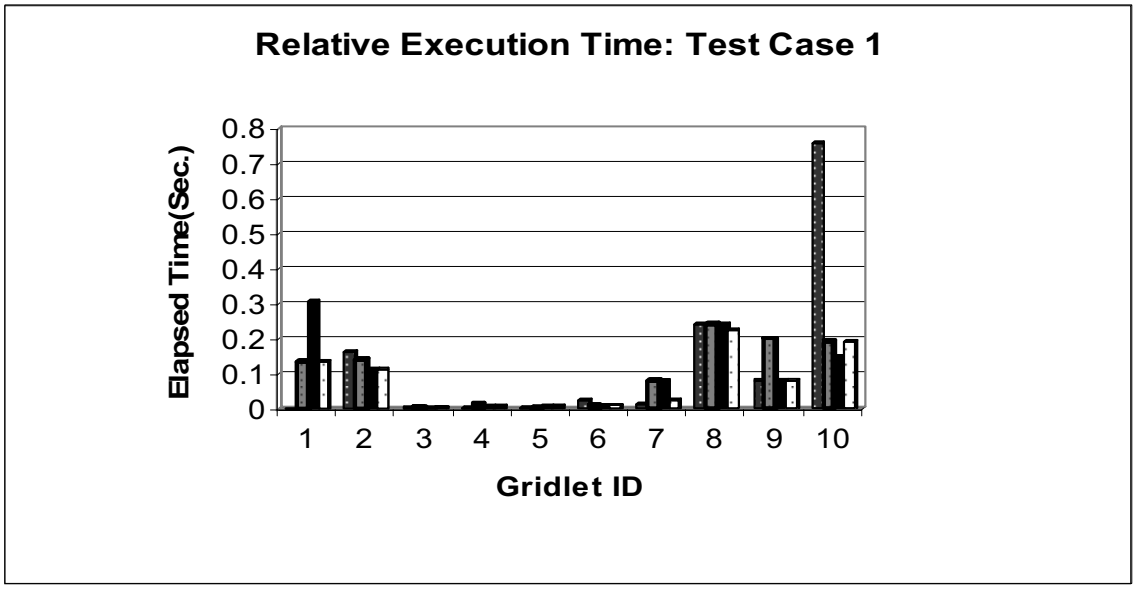

Fig. 2

SJFR	LJFR	FCFS	PRIORITY

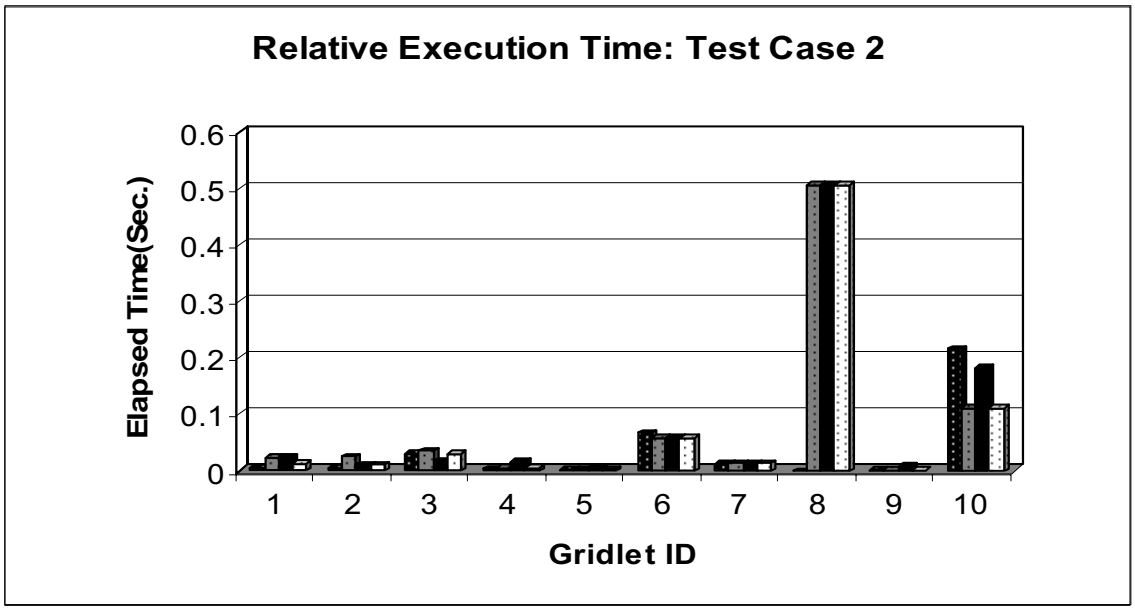

Fig. 3

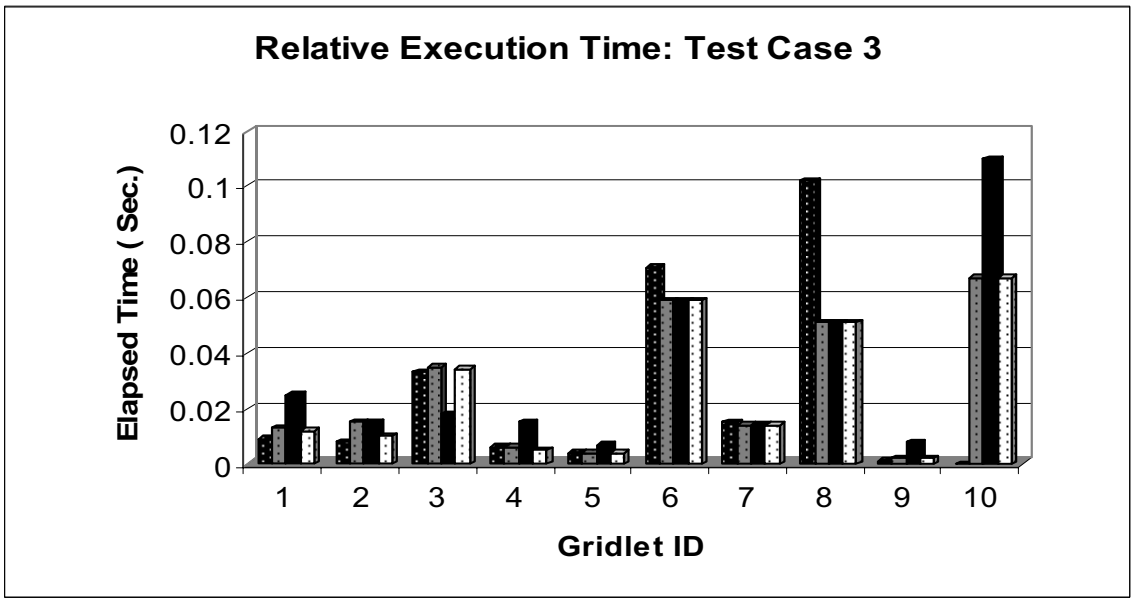

Fig. 4

4. Performance Analysis of Different Scheduling Algorithms

Performance evaluation is done based on execution time. Execution time for each job is calculated by calculating the elapsed time between submission time and the completion time of the job. Three test cases have been taken to have a relative comparison of FCFS, SJFR, LJFR and Priority based Scheduling algorithms. Graphs are drawn with the GridletID (JobID) on X-axis and Execution Time in seconds on Y-axis.

All the three algorithms outperforms FCFS. LJFR and Priority based algorithms outperforms SJFR. Sometimes LJFR and Priority based algorithms are close and sometimes Priority based algorithm outperforms LJFR.

4. Conclusion

Design of a proper scheduling algorithm with an aim to improve the performance of a grid has indeed been a complex job with a lot of parameters to be taken into consideration. The SJFR and LJFR algorithms take the computational complexity of the jobs and the speed of the resources into consideration while scheduling the jobs. In the Priority-based algorithm a new parameter named "priority" has been used in the analysis and consequently, the jobs are divided into high, medium and low categories. Generally, a job, which needs high computational power, exhibiting less parallelism is given a high priority. This is found to improve the performance of computational grids. The algorithms are tested and their performances are analyzed.

References

[1] Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann (1998)

[2] Foster, I., Kesselman, C.: The Globus Project: a Status Report In Proc. IPPS/SPDP'98 Workshop on Heterogeneous Computing, pp. 4-18, 1998.

[3] Manish Arora and Sajal K. Das, Rupak Biswas, "A De-centralized Scheduling and Load Balancing Algorithm for Heterogeneous Grid Environments", Proceedings of the International Conference on Parallel Processing Workshops (ICPPW'02).

[4] Spooner, D.P., Jarvis, S.A., Cao, J., Saini, S. and Nudd, S.R.: " Local grid scheduling techniques using performance prediction", IEE Proc.-Comput. Digit. Tech., Vol. 150, No. 2, March 2003.

[5] Abraham. A., Buyya. R., and Nath. B.: "Nature's heuristics for scheduling jobs on computational grids", Proc. 8th IEEE Int. Conf. on Advanced computing and communications, Cochin, India, 2000.

[6] Rajkumar Buyya, David Abramson and Srikumar Venugopal, "The Grid Economy", Proceedings of the IEEE, Vol. 93, No. 3, Match 2005

[7] Rajkumar Buyya, David Abramson and Jonathan Giddy, "Economy Driven Resource Management Architecture for Computational Power Grids", International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2000), Las Vegas, USA, 2000.

[8] Francine Berman, Richard Wolski, Henri Casanova and Walfredo Cirne, "Adaptive Computing on the Grid Using AppLes", In IEEE Transactions On Parallel and Distributed Systems, volume 14, pages 369-382, April 2003.

[9] Tsan Sheng Hsu, Joseph C. Lee, Dian Rae Lopez and William A. Royce, "Task Allocation on a Network of Processors", In IEEE Transactions On Computers, volume 49, pages 1339-1353, December 2000.

[10] Rajkumar Buyya, "GridSim: A Toolkit for Modeling and Simulation of Grid Resource Management and Scheduling", Chapter 5, John Wiley & Sons, Ltd