Journal of Theoretical and Applied Information Technology
31st May 2014. Vol. 63 No.3
© 2005 - 2014 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

GROUPING BASED SCHEDULING WITH RESOURCE FAILURE HANDLING IN COMPUTATIONAL GRID

[1]P. LATCHOUMY, [2]P. SHEIK ABDUL KHADER

[1]Research Scholar, Department of Information Technology, BSA University, Chennai, India

[2]Professor & Head, Department of Computer Applications, BSA University, Chennai, India

E-mail: [1] lak75dhana@gmail.com, [2]hodca@bsauniv.ac.in

ABSTRACT

Grid computing provides a virtual framework that integrates resources and services distributed across multiple controlled domains. Grouping based job scheduling is an important issue in grid computing in aspects of minimizing the processing time and maximizing the resource utilization. In particular, reliable scheduling is very important for efficient job execution. In general the failure of resources affects job execution during runtime fatally due to dynamic nature of grid environment. Our proposed system, grouping based scheduling with resource failure handling handles the resource failure before and after allocating the grouped fine-grained jobs with the suitable resources in computational grid. In this system, the reliability of sites is monitored and the historical data is used for predicting resource failures. This information is taken into account when jobs groups are created and dispatched to resources. Hence, it minimizes the probability of job failure during execution. And our scheduler also can recover the failed job group by an appropriate recovery method with last checkpoint using reduced check pointing time strategy within the specified deadline. Due to resource failure, even high processing power resource may not be completed the job successfully and hence it increases the number of rescheduling and minimizes the resource utilization. Our proposed failure handling method is to provide reliable job execution with minimum processing time and maximum resource utilization. Through simulation we have evaluated the performance of the system. In this method, job grouping based scheduling with resource failure handling algorithm is developed for the grid environment to provide guaranteed service to the grid user within the specified deadline.

Keywords: *Capability Value (RCV), Resource Failure Prediction, Group Performance Time (GPT), Group Checkpointing Time (GCPT), Resource Utilization.*

1. INTRODUCTION

The term grid computing is a way to make the computation power of idle work stations available to remote grid users for the execution of their computation hungry jobs [1]. In grid environment, the resources are heterogeneous, dynamic, geographically scattered and may be connected over high latency networks leads significant communication cost and can't be ignored. Hence, the communication cost has become an important factor of performance measurement and must be taken into consideration while scheduling jobs into the grid.

Grouping based job scheduling is responsible for selecting suitable resources for achieving maximum resource utilization with minimum processing time [2]. An application with large number of fine-grained jobs when submitted individually to the grid resources over the networks incur a communication overhead that is more than the total computation time of each job at the resource. Moreover, these also lead to poor utilization of communication network and uneven utilization of the resources. Hence we concentrate on reduction in communication time by transmitting jobs as a group of jobs instead of a single job.

Therefore, jobs can be grouped at the scheduling level according to the processing capabilities of the available resources, and proceed with the job scheduling and deployment activities [3]. Hence, our approach consists of sending complete groups of jobs to grid sites for execution instead of independent jobs. Our system deals with job scheduling in grids by considering failure of nodes.

There may be a wide range of failures due to distributed and dynamic nature of grids. Running applications in such an environment is susceptible to wide range of failures as revealed, with real users on fault treatments in the grid [4]. The failure occurrence of resources in the grid computing is higher than in a traditional parallel computing. In general the failure of resources affects job execution and hence it decreases the success rate of the job. Fault tolerance service is essential in computational grids. Also the emergence of grid computing will further increase the importance of fault tolerance. Grid computing will impose a number of unique new conceptual and technical challenges to fault-tolerance researchers. Thus, the incorporation of fault tolerance related features in a grid job scheduling strategy should not be an optional feature, but a necessity. The different issues in fault tolerance have been discussed [5].

In general, the work on Grid fault tolerance can be divided into proactive and reactive mechanisms. In proactive mechanisms, the failure consideration for the Grid is made before the scheduling of a job, and dispatched with hopes that the job does not fail. A reactive mechanism handles the job failures after it has occurred. From reviews of literature, many works are primarily reactive in nature and deal with failures through Grid monitoring [6]. Proactive or reactive method be alone is unlikely sufficient to provide a reliable solution for efficient job execution.

The motivation of this paper is to develop Job grouping based scheduler and resource failure handling algorithm that must be efficient and effective in reducing failure probability and total processing time.

Hence, this work considers the resource failure before and after the job submission in grouping based job scheduling. The objective of our proposed strategy is to develop a model that selects suitable resources by considering failure information and current status of the resource before the submission of job group submission and successfully complete the job group in the presence of resource failure within user specified deadline.

To summarize, this paper's contributions include (a) Design of failure aware resource selection, (b) Allocation of resources such that minimize the failure probability, (c) grouping/ungrouping of jobs, (d) Providing checkpoint service, and (f) Selection of appropriate recovery method to provide reliable service to the grid user within the specified deadline.

The rest of the paper is organized as follows: A brief review of related work is given in Section 2 to motivate our work. In section 3, proposed system is described with job grouping based scheduling with resource failure handling algorithm design. Section 4 elaborates the experimental results. Finally, we summarize and lay out our future work in Section 5.

2. RELATED WORK

In [3], author presents dynamic job grouping strategy which concentrates on maximizing the utilization of grid resource processing capabilities and reducing the overhead time and cost taken to execute the jobs through a batch mode dynamic scheduling. This strategy is not considering the reliability of the resources while submitting the grouped job. But our system is considering the resource failure before and after submission of job group.

Authors developed an algorithm in which grouping of independent jobs with small processing requirements into suitable jobs with large processing requirements and considered network bandwidth into account for scheduling to improve the performance of job scheduling [7]. This system maximizes the resource utilization and reduces the network latency. But it is not consider the dynamic characteristics of resources while submitting the grouped job. Our system considers the current status of the grid resources (availability and speed).

It is observed that an application demand aware performs better when user satisfaction is taken into account [8], but dynamic characteristics of the resources are not considered. Our system considers the reliability of the resource using history from the grid information system (GIS) and the current status of the grid resources (availability and speed) before submitting the job group.

An agent based scheduling with grouping strategy is discussed [9]. The main focus on this paper is to maximize the resource utilization and minimize the processing time and consider only the submission failure. If the selected resource is not suitable, then simply schedule the group with the next suitable resource. It does not consider the failure information before and after scheduling the group.

But our system concentrate on reliability of the resource before submitting the group, hence it guarantees the successful completion of the group. If group is failed due to resource failure, then our system recovers the failed group by selecting suitable recovery method (recover with the same or backup resource).

Author developed a prioritized user demand algorithm which considered user deadline for allocating jobs from different users to different heterogeneous resources from different administrative domains [10]. In this algorithm better

Journal of Theoretical and Applied Information Technology
31st May 2014. Vol. 63 No.3
© 2005 - 2014 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

make span and more user satisfaction is achieved but data requirement is not considered. They consider user deadline, execution time and communication time for scheduling the tasks to resources. But dynamic characteristics of resources and processing overhead are not considered in this algorithm. This system is not concentrate on reliability of resources while selecting the resource for job group submission.

Time-minimization algorithm for job grouping based scheduling is developed [11]. In this algorithm the overhead time is considered to minimize the overall processing time of grouped job. But it is not concentrated on user satisfaction. Out system update the status of the job completion with respect to the completion of the job group (within deadline or with overhead).

Authors proposed task granularity policies for deploying bag-of-tasks application on global grids [12]. The policies are optimizes the task granularity (the number of tasks that should be grouped in a batch) for each resource at runtime leads an efficient grid utilization. But this system is not considering the reliability while setting policies for deploying the group of jobs.

From the literature survey, the above mentioned grouping based algorithms gave little bit attention to the resource failure. Hence, our system is considering the resource failure before and after scheduling the grouped job with the suitable resources for reliable job execution. In general, job grouping is based on either MIPS or memory or speed (BW) of resource or combination of all the three. In our proposed system, the reliability of sites is monitored and the historical data is used for predicting resource failures. This information is taken into account when jobs groups are created and dispatched to resources. And handle the resource failure by selecting appropriate recovery method. This combined failure handling method is to provide reliable job execution with minimum processing time and maximum resource utilization. In this method, job grouping based scheduling algorithm is developed for the grid environment to provide guaranteed service to the grid user within the specified deadline. This study focuses and evaluates and extension to grouping based scheduling with resource failure handling, which aims to reduce both failure probability and processing time of the job groups.

3. PROPOSED FRAMEWORK

3.1 Grouping based Scheduling with Resource Failure Handling (GS-RFH)

Scheduling a set of fine-grained jobs onto suitable resources aims to minimizing communication and computation time, and to maximizing the resource utilization. Due to resource failure, even the high processing power resource may not be completed the job successfully and hence it increases the number of rescheduling and minimizes the resource utilization. Therefore, we need to consider the failure information about the resources before and after scheduling of grouped job with the suitable resources.

Hence, our proposed system, grouping based scheduling with resource failure handling (GS-RFH) in computational grid consider the resource failure before and after allocating the grouped job with the suitable resource to provide guaranteed service to the grid user with minimum completion time of job group.

3.2 System Model: GS-RFH

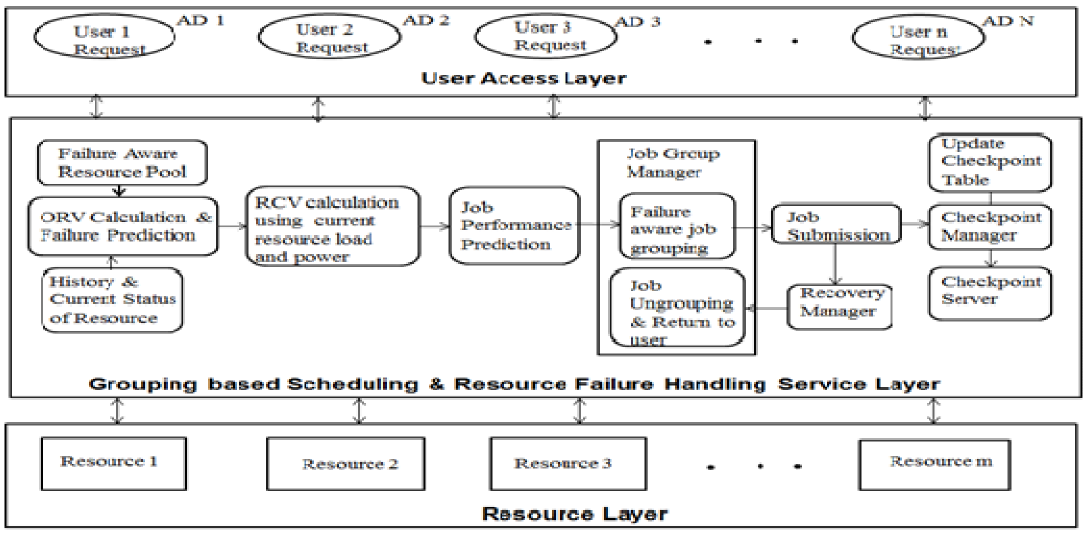
The motivation of this paper is to develop a system named as grouping based scheduler using resource failure handling that must be efficient and effective in reducing the number of failure during execution of job group and hence it increase the success rate of group with minimum processing time and hence it maximize the resource utilization. Grid users submit their application jobs with job requirements such as size of each job with MI (Millions of Instructions), required number of Processing Elements (PEs), budget and deadline. The details of the available grid resources are obtained from Grid Information System (GIS) entity that keeps track of the resources available in the grid environment.

Whenever a scheduler has group of job to execute, it collects the resource details from the GIS for mapping of the grouped job with the suitable resource.

Each grid resources is described in terms of their various characteristics, such as resource ID, name, total number of machines in each resource, total processing elements (PEs) in each machine, computational power of PE in millions of instructions per second (MIPS), and bandwidth speed.

Our grouping based scheduling with resource failure handling (GS-RFH) model is designed as three layer architecture (Figure 1). The lower layer consists of the collection of heterogeneous resources which are registered by different resource providers. The middle layer provides reliable job execution by selecting suitable resources and allocating user job groups with the selected resources to complete successfully within the user given deadline. The higher layer is called as user access layer where different users submit their jobs request from different Administrative Domains (ADs).

GS-RFH model handles resource failure before and after allocating grouped job within Virtual Organization (VO). First our system gets the user applications requirements and the resource specifications.


User 1 Request
AD 1
User 2 Request
AD 2
User 3 Request
AD 3
User n Request
AD N
User Access Layer
Failure Aware Resource Pool
ORV Calculation & Failure Prediction
History & Current Status of Resource
RCV calculation using current resource load and power
Job Performance Prediction
Job Group Manager
Failure aware job grouping
Job Ungrouping & Return to user
Job Submission
Recovery Manager
Update Checkpoint Table
Checkpoint Manager
Checkpoint Server
Grouping based Scheduling & Resource Failure Handling Service Layer
Resource 1
Resource 2
Resource 3
Resource m
Resource Layer
Figure 1: Grouping based Scheduling with Resource Failure Handling

Then it calculates the overall resource value (ORV) using history and current grid information. It minimizes the probability of the job failure during execution; hence it reduces the number of rescheduling. And, it calculate the resource capability value (RCV) for getting more suitable resource by considering ORV, computational power (MIPS) and the current load (1-Load$_j$) of the resource before assigning the job group by the scheduler. By considering the computational power and the current load, our system selects more available computation power resource for an application and thus should be considered first. It sorts the resources in descending order using ORV for selecting suitable resources for reliable execution. It reduces the group processing time of the grouped job and hence it increases the throughput. Jobs are grouped based RCV and group performance time (GPT). Then our system assigns the unique id to the created group and dispatches the grouped job with the reliable resources.

Then the system calls the check pointing manager to find the minimum number of check points for periodically saving the state of the job. The state of the job group is updated in the checkpoint table (Group id, % of group check pointed and status of the job group). If successful group execution, it returns the result to the group manager for ungrouping and then return to the user. Else if failure occurred, the system calls the recovery manager to select an appropriate recovery method to minimize the recovery time with last checkpoint using reduced check pointing time strategy. At the end of the process, the status of the resources (completed within deadline or with overhead) is gets updated with

resource status table (Table 1) for every transaction. Hence, our system reduces the group processing time of the grouped job and hence it increases the throughput.

3.3. Module Descriptions

3.3.1. Resource Failure Prediction

The reliability of sites is monitored and the historical data is used for predicting resource failures. This information is taken into account when jobs groups are created and dispatched to resources.

3.3.1.1. Overall Resource Value Estimation

The overall resource value (ORV) is calculated based on the reliability of the resource and the current resource status (CRS).

$$ORV_j = Reliability_j * w1 + CRS_j * w2 \qquad (1)$$

Where, ORV=0-1 & w1 and w2 value is 0.5, since we have equal priority over reliability and the current status of the resource.

Table 1: Job completion status & Job completion status value

Job Completion Status	Job_Completion _Status _Value	Successful/ Un Successful
within deadline	*1*	*Successful Completion*
with 5% overhead	*0.8*	*Successful Completion with Overhead*
With 10% overhead	*0.6*	
With 15% overhead	*0.4*	
With 20% overhead (Maximum allowed overhead)	*0.2*	
>20% overhead	*0*	*Un Successful*

The reliability of the resource is based on the job completion time (JCT) and the job completion status value (JCSV). JCSV is based on the job completion status by the previous transactions with respect to the selected resource (Table 1).

$$Reliability_j = JCT_{i,j} * JCSV_{i,j} \qquad (2)$$

The current resource status (CRS) is based on the availability of the resource and the network speed (NS) between the scheduler and the resources.

$$CRS_j = Availability_j * w3 + NS_j * w4 \qquad (3)$$

Where, ORV=0-1 & w1 and w2 value is 0.5, since we have equal priority over avilability of resource and the n/w speed to connect the selected resource.

Availability-The ratio between the number of times a resource being available to the grid and the number of attempts made to access the resource.

$$Availability_j = \frac{No.ofTimesAvailable_j}{TotalNo.ofTimesAccessed_j}$$

Network Speed (BW)- Required network bandwidth is used to connect the selected resource is disccused in our previous work [13].

3.3.1.2. Resource Capability Value Estimation

The resource capability value (RCV) is calculated based on the ORV, MIPS and the current load in the resource.

$$RCV_j = ORV_j * MIPS_j * (1 - Load_j) \qquad (4)$$

MIPS are the machine utilization and (1-Load$_j$) means how much percent of the CPU is available for a remote job. MIPSs are the machine's computing power. So the product stands for the amount of computing power available in the reliable resource to the job. A higher value product indicates that the resource has more available computing power with higher reliability for an application and thus should be considered first.

And sort the resources using RCV in descending order and selects the first 'r' number of resources from the selected list. Where, r is number of jobs in the job queue divided by 2. From the r resources we select the minimum completion time with respect to the job in the queue and map/group, related to the minimum completion time of job. Continue the same process until all jobs are grouped by job group manager. After grouping, an unique ID is assigned to the group and the scheduler dispatches the group with the hope that the group does not fail during execution. By considering the reliability factor of the resources our system minimizes the probability of failure during the execution of the grouped job.

3.3.2. Group Performance Time

The intuition behind this algorithm is to assign jobs to resources one by one while keeping the differece among the job completion time on each resource as little as possible.

Group performance time (GPT) is calculated based on the execution time of the grouped job(GET), waiting time of the queued grouped job in the resource queue (GWT) and transfer time of the grouped job between the scheduler and the selected resources (GTT).

$$GPT_{k,j} = GET_{k,j} + GWT_{k,j} + GTT_{k,j} \qquad (5)$$

GET is calculated based on the group size and job execution time (JET).

$$GET_k = \sum_{i=1}^{n} JET_{i,k}$$

$$JET_{i,j} = \frac{Workload_i}{(MIPS_j * (1 - Load_j))}$$

$$Load_j = \frac{\sum_{i=0}^{n} JobSize_i}{\sum_{j=0}^{m} ResourcePower_j}$$

The current resource load is calculated using job size and the resource power. Load j is the ratio of aggregated length of jobs submitted to the grid to the aggregated job length that the grid is capable of performing. Where n is the number of jobs submitted to the grid, m is the number of resources, Job Size i denotes the length of job i (MI) and Resource Power j is the Sum of computational power of processing elements (PEs) in MIPS. By considering the group workload while calculating the GPT, our algorithm selects the least load resource while submitting the grouped job leads to minimize the GPT and maximize the resource utilization.

GWT is calculated by summing up the waiting time of the each group in the resource queue.

$$GWT_k = \sum_{i=1}^{n} QWT_{i,j}$$

GTT is calculated based on the group size and the speed of the network between the scheduler and the selected resources.

$$GTT_{k,j} = \frac{GroupSize_k}{NetBW_j}$$

3.3.3. Check Pointing Process

Due to larger check pointing interval, the overhead is occurred. Hence, our system take care of reduction in check pointing time (P. Latchoumy and P. Sheik Abdul Khader, 2013). The scheduler calls the checkpoint manager (CM) to calculate minimum number of checkpoints based on based on runtime condition of the job and failure information of the resources. CM calls checkpoint server to save the states of the execution and it calculates group checkpointing time, GCPT $_{k,j}$ using minimum number of checkpoints.

Group CheckPointing Time(GCPT) Calculation

GCPT is the sum of jobs checkpointing time in the group. Single jobs checkpoint is the product of number of checkpoints and single checkpoint time.

$$GCPT_{k,j} = \sum_{i=1}^{n} NCP * SCT_{i,j}$$

Where

$$SCT_{p,j} = \frac{CheckpointSize_p}{DiskBW_j}$$

3.3.4. Recovery Method Selection

Scheduler waits for some Timeinterval. If there is successful completion within the timeinterval of the group then the scheduler ungrouped it and will send to the user.

$$TimeInterval_{k,j} = GPT_{k,j} + GCPT_{k,j}.$$

If resource failure occurs, our system check whether there is a node crash or QoS failure. If node crash then the failed resource is get deleted from the selected list. Else if QoS failure (due to resource unavailability), we can get the partial output using checkpointing table and continue with the same resource if $RCV_j > MAX(RCV)_j/2$, else with the backup resource using the last saved check point. And the system calculates GTotalTime $_{k,j}$=TimeInterval $_{k,j}$ +RO $_{k,j}$ for continuing with the same resource and GTotalTime $_{k,j}$=TimeInterval$_{k,j}$ +RO$_{k,j}$ +BTT $_{k,b}$ for continuing with the backup resource by the failed group. Our recovery selection method is take care of providing the flexibility over the fault tolerant method for recovering the failed group. By selecting an appropriate fault tolerant method (with check pointing & without backup, and with check pointing & with backup), our system maximizes the resource utilization.

4. GROUPING BASED SCHEDULING WITH RESOURCE FAILURE HANDLING ALGORITHM

Input: *User Jobs with Requirements & Resource History and Current Information*

Output: *Successful completion of grouped jobs with Minimum Processing Time, Failure Probabilityand Checkpoinitng Time & Maximum Resource Utilization*

GroupingbasedSchedulingwithResourceFailure Handling ()

```
{
  (1) Get user applications requirements
  (2) Get resources specifications
  (3) Calculate ORV using reliability & current
      resource status for predicting resource failures
  (4) Sort the failure aware resources using Overall
      Resource Value (ORV) in descending order
  (5) Calculate resource capability value (RCV) using
      ORV, MIPS and current load of resource (1-
      $Load_j$) & sort in descending order and select first
      (number of jobs/2) resources for group
      submission; minimum group size should be at
      least 2.
  (6) //map jobs with the minimum completion time
      //resources
      While (Job Queue!=NULL)
      {
      For each job Ji from a queue
      { For each selected resource Rj from the selected
  list
```

```
    { $JET_{i,j} = \frac{Workload_i}{(MIPS_j * (1 - Load_j))}$
        //where, JETi,j is the expected exectuion
        //time of job Ji on resource Rj
      j++;        // End for    }
    //Find a resourcej where GWTi,j+JETi,j is minimal
    // GWT,j is the current estimated completion time of
    //the assigned jobs on res. Rj && Find map of Ji with
    // Rjwhere GWTi,j+JETi,j is minimal
(7) Groupk=GroupkU{Jobi} && GWTj=GWtj+JET;
(8) Assign unique ID to newly created Group $_k$
(9) Calculates Group Performance Time (GPT) $_{k,j}$
        $GPT_{k,j} = GET_{k,j} + GWT_{k,j} + GTT_{k,j}$
        $GET_k = \sum_{i=1}^{n} JET_{i,k}$
        $GWT_k = \sum_{i=1}^{n} QWT_{i,j}$
        $GTT_{k,j} = \frac{GroupSize_k}{NetBW_j}$      }
(10) Dispatch Group $_k$ to SelectedResource (Rbest$_j$),
     so that Group$_k$,Rbest$_j$=MIN(GPT $_{k,j}$) (Execution Starts)
(11) Calculate Group Checkpointing Time (GCPT)
     {
        $GCPT_{k,j} = \sum_{i=1}^{n} NCP * SCT_{i,j}$
     Where SCT$_{i,j}$=CheckpointSize$_i$/DiskBW$_j$ &
     Update Checkpoint Table in the Checkpoint Manager}
(12) Scheduler waits a period of Timeinterval
(13) If there is no response within Timeinterval
     // where, TimeInterval=GPT $_{k,j}$+GCPT $_{k,j}$
        If (Node Crash) //Check Failure Type
        { Delete the resource from the selected resource
          list and transfer the states of the failed grouped job with
          the other resource using last checkpoint state & go to step 16}
(14) Else if (QoS Failure),
        Call Recovery_Selection_Request ()
(15) Update the Resource Status Table using
     GTotalTime GTotalTime $_{k,j}$ with Successful
     Completion and ugrouped the grouped jobs by
     job group manager and return it to the user. (OR)
     Unsuccessful Completion and get the partial
     execution from Checkpoint Manager and
     reschedule the failed group to the next reliable
     resource in the selected list with last check point.
(16) i++;} // End for
(17) } // End while
```

Recovery_Selection_Request ()

```
{ (14). a. If ((RCV$_j$>MAX(RCV)$_j$/2)
        { Recover the failed job with Same Resource
          using the last checkpoint &
          Return GTotalTime $_{k,j}$=TimeInterval $_{k,j}$ +RO $_{k,j}$
        }
   b. Else
        {Recover the failed job with Backup Resource
         using the last checkpoint &
         Return GTotalTime $_{k,j}$=TimeInterval$_{k,j}$+RO$_{k,j}$+BTT $_{k,b}$}
}
```

[***Note:*** GTotalTime $_{k,j}$ is the total time is required for execution of the complete group. That is difference between the completion time of the last job in the group and submission time of the first job in the group on the resource Where, GTotalTime $_{k,j}$ <= GDeadline $_{k,j}$ And GDeadline $_{k,j}$= SUM(JobsDeadlines).]

5. RESULTS AND DISCUSSION

We have implemented our proposed model using Grid Simulation toolkit GridSim 5.2 [14]. A simulation is conducted in heterogeneous environment where each resource has machines with different characteristics such as MIPS, bandwidth & memory size, reliability, availability and load.

The inputs to the simulations are user job requirements (number of jobs with MI (Millions of Instructions), deadline and cost), resource specifications (number of resources, processing power of resources in MIPS and failure information & current status of reosurce) and required network bandwidth to connect the selected resources with the reliable scheduler. The different parameters such as group processing time, failure probability, resource utilization and number of checkpoints are analyzed and verify the improvement of the proposed job grouping based scheduling with resource failure handling over job grouping based scheduling approach. The simulation parameters and its values are given in Table 2.

Table 2: Simulation Setup

Parameters	Value of Parameter
No. of Jobs	500,400,300,200
Job Size	1500 MI, 1000 MI, 500MI
No. of Resources	100,75,50,25
Deadline	1000 (ms)
Budget	300$
Cost of each PE	20$/Hour
Computational Power of each PE	2000 MIPS
Network Speed	10 Mbps

5.1 Group Performance Time

Simulation results illustrate that our proposed model minimizes the group performance time (GPT) because it can select resources by considering reliability, availability & processing capability before sucheduling the job group. And also our system group the job based on the minimum completion time of the resources is further reduce the group performance time. Hence, our grouping based scheduling with resource failure handling (GS-RFH) approach complete the group with miminum completion time (Figure 2) over dynamic job grouping based scheduling algorithm [3].

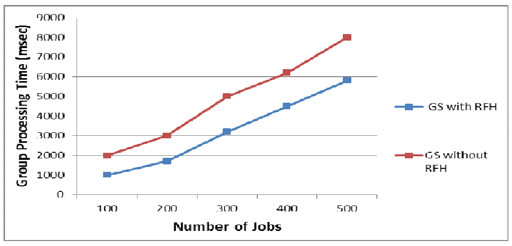


Figure 2: Group Performance Time Vs Number of Job

5.2 Failure Probability

Our system reduces the probability percentage of failure in the selective resources for executing user grouped jobs. Suitable resources are selected using its past history and current status. By considering the reliability factor in grouping based scheduling should increases the success rate of the group execution. Resources with more fault history were not selected for group submission. Our system (GS-RFH) minimizes probability of failure during grouped job execution and hence reduces the number of rescheduling (Figure 3).

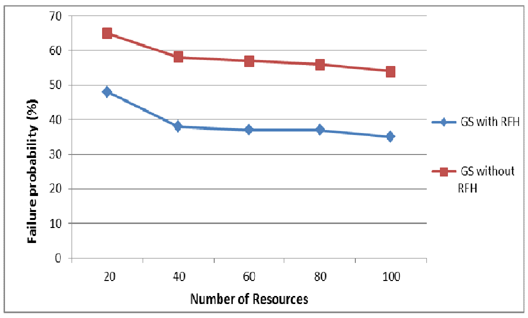


Figure 3: Failure Probability Vs Number of Resources

5.3 Resource Utilization

It can be observed from Figure 4 that our approach outperforms in terms of resource utilization with different resources. The resources are effectively utilized by selecting most suitable resource using reliability and performance of the resources for group of jobs submission. Our model improves the resource utilization further by selecting appropriate recovery method after the failure has occured. The failed group can be recovered with the same resource if $ORV_j > MAX(ORV_j)/2$, else with the backup resource. It reduces the group migration process. Hence we can attain the maximum resource utilization. This approach has 15% improvement on the average over resource utilization using grouping based scheduling with resource failure handling (GS with RFH).

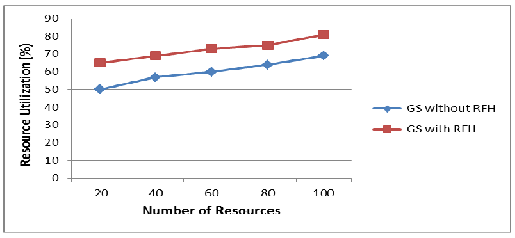


Figure 4:Resource Utilization Vs Number of Resources

5.4 Number of Checkpoints

An inappropriate check pointing interval to cause delay in the job execution, and hence it reduces the throughput. Our proposed work, finds the minimum number of checkpoints based on runtime condition of the job and failure information of the resources using reduced checkpointing time (RCT) strategy. If the size of checkpointing interval is static, then the system takes more time for checkpointing and hence it takes more time to complete the job. In our system, we adjust the checkpointing interval dynamically and it shows 30%-38% improvement on the average over the group processing time of the grouped job (Figure 5).

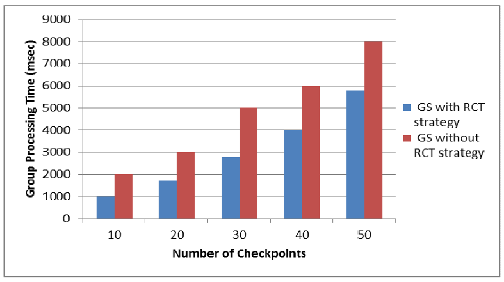


Figure 5: Goup Performance Time Vs No.of Checkpoints

5.5 Recovery Satus of Failed Group

Our proposed model to reduce the recovery time by selecting an appropriate recovery method. After failure occurred, the most of the failed group is recovered with the same reosurce (within deadline) and some group is recovered with backup resource (with some overhead). Hence, it minimizes the number of rescheduling /migration and and the total group completion time
(Figure 6).

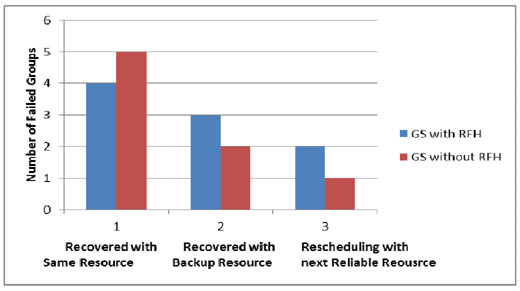


Figure 6: Number of Failed Groups Vs. Recovery Status

6. CONCLUSION

Reliable job execution in grouping based scheduling is very important issue in failure prone grid environment. Our proposed strategy grouping based scheduling with resource failure handling provides reliable job execution by considering some quality factors such as reliability, availability, load and processing capability of the resources before submitting the grouped job with the suitable resources. It minimizes the probability of job failure during execution and hence it reduces the number of rescheduling. And our scheduler also recovered the failed job group after the failure has occurred by selecting an appropriate recovery method with last checkpoint using reduced check pointing time strategy. This system provides the flexibility over the selection of the fault tolerant (recovery) method to minimize the group completion time and hence it increases the system throughput.

The performance of our system can be enhanced by finding the granularity using the failure rate of the resources for setting the size of the group, developing an appropriate algorithm in selecting the backup resource, and finding other reasons for resource failure and recovery methods as future work. Also we incorporate our strategy in cloud computing environment in future.

7. ACKNOWLEDGEMENT

The authors would like to thanks the reviewers for their reviews and comments that helped improve this paper.

REFERENCES:

[1]Foster, C. Kesselman, and S. Tueke, " The anatomy of the grid: Enabling scalable virtual organizations", Supercomputing Applications, 2001.

[2] Quan Liu and Yeqing Liao, "Grouping-based Fine-grained Job Scheduling in Grid Computing", First International Workshop on Education Technology and Computer Science, 78-0-7695-3557-9/09, IEEE, DOI 10.1109/ETCS.2009.132, 2009.

[3]Nithiaplidary Muthuvelu, Junyang Liu Soe, Srikumar Venugopal, Anthony Sulistio and Rajkumar Buya, " A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on global Grids", conferences in research and practice in Information Technology, Vol. 44, 2005.

[4] Medeiros R, Cirne W, Brasileiro F, Sauv'e J, "Faults in grids: why are they so bad and what can be done about it?", In: Grid Computing, proceedings. Fourth international workshop, ISBN 1-59593-414-6, 2003.

[5] P. Latchoumy and P. Sheik Abdul Khader, "Survey on Fault Tolerance in Grid Computing", International Journal of Computer Science & Engineering Survey (IJCSES), Vol. 2, No. 4, 2011.

[6] Huda MT, Schmidt HW, Peake ID, "An agent oriented proactive fault tolerant framework for grid
Computing", in: First international conference on e-science and grid computing (e-science'05), 2005.

[7] Ng Wai Keat, Ang Tan Fong, Ling Teck Chaw and Liew Chee Sun, "Scheduling framework for bandwidth-aware job grouping-based scheduling in Grid Computing", Malasian Journal of Computer Science, Vol 19(2), 2006.

[8] Jie Lin, Bin Gong, Hui Liu, Chaoying Yang, Yuhui Tian, "An Application Demand aware Scheduling Algorithm in Heterogeneous Environment", IEEE Proceedings of the 11th International Conference on Computer Supported Cooperative Work inDesign, IEEE Xplore press, Melbourne, Vic, pp509-604,DOI:10.1109/ CSCWD.2007.4281504, 2007.

[9] Raksha Sharma,Vishnu Kant Soni, Manoj Kumar mishra, Prachet Bhuyan and Utpal Chandra Dey, "An agent based dynamic resource scheduling model with FCFS-job grouping strategy in grid computing", World academy of science, engineering and Technology, 64, 2010.

[10]Suresh.P, Balasubramanie.P and Keerthika.P, "Prioritized User Demand Approach for Scheduling Meta Tasks on Heterogeneous Grid Environment", International Journal of Computer Applications (0975 – 8887), Volume 23– No.1, DOI:10.5120/2856-3670, 2011.

[11] Manoj Kumar Mishra, Prithviraj Mohanty and G.B. Mund, "A Time-minimization Dynamic Job grouping-based Scheduling in Grid Computing", International Journal of Computer Applications (0975-8887), Volume 40-No. 16, 2012.

[12] Nithiaplidary Muthuvelu, Christian Vecchiola, Ian Chai, Eswaran Chikkannan and Rajkumar Buya, "Task granularity policies for deploying bag-of-task applications on global grids", Future generation computer systems 29, 170-181, 2012.

[13]. P. Latchoumy and P. Sheik Abdul Khader, "A Combined Approach: Proactive and Reactive Failure Handling for Efficient Job Execution in Computational Grid', in Advances in Intelligent Systems and Computing, Vol. 243 [ISBN: 978-81- 322-1664-3], pp. 713-725, Springer, 2014.

[14] www.buyya.com/gridsim/