Abstract
In this paper we have presented WQDR-FT, information based on line fault-tolerant scheduler with dynamic replication based on WQR-FT in which we are considering a replication threshold for making the replication dynamic and also using check pointing. WQDR-FT is able to do better resource utilization than other scheduling strategies.

1. Introduction
The enormous reputation of Internet has made another much expansive scale open door for Grid computing i.e., many desktop PCs, whose idle cycles can be changed to run Grid applications, are joined with wide-zone systems both in the business enterprises and in the home. These new stages for high throughput applications are called Desktop Grids [1,2,3]. Due to fault occurrence at any time in Desktop Grid, there is a need to formulate scheduling strategies that will increase the performance even in the presence of faults.

As per our study it has been observed that information free schedulers are using much more resources than necessary so cannot exploit the full potential of Desktop Grids. For instance in WQR-FT various replicas of the same task are created & scheduled on several resources. If a task has scheduled on very sluggish resource then also it has not been rescheduled on faster resources because of the presence of extra replicas of the same task scheduled on faster resources. If we are having the information about resources & tasks in advance then better scheduling decisions may be done. For example if we know about the availability of resource & the execution time of task then scheduler can decide that which resource is able to complete the execution of task.

The rest of the paper is structured as follows. Section 2, analyze some related works. In Section 3 we discuss about Information free scheduling. In Section 4 proposed scheduler WQDR-FT is presented. Section 5 presents the results. Finally, Section 6 concludes the paper.

2. Related Work
Various schedulers have been proposed in the scientific literature for desktop grid for allotting a set of tasks to a set of resources. Basically Scheduling on Grids is generally called as meta-scheduling [4]. Meta-scheduling means to select a resource to execute a job. Usually the term mapping is being used because the meta-scheduler will allocate tasks onto resources. Once the resource is selected, batch scheduler will schedule tasks on
processors, as a result we are using a two-level scheduling [5]. As jobs arrive continuously, heuristics used for mapping are online ones [6]. Using online algorithms [7], independent jobs are being executed by selecting parallel resources. Various online schedulers proposed in scientific literature are: Random, Round Robin(RR), OLB, MET & MCT.

M. Macheswaran et al. in [8] investigated that on line scheduling heuristic is suitable only when the arrival rate is low. When the arrival rate is high then batch mode scheduling is suitable to use because there will be a sufficient number of tasks are there collected in a batch to keep hosts busy. Various batch mode scheduling heuristics proposed are Work Queue with Replication (WQR), WQR-FT, Min-Min, Max-Min, LJFR-SJFR and Suffrage scheduling strategies.

From our study, we can conclude fault may occur at any time in desktop grid. As a result of this application execution time increases multifold. So there is a need to formulate scheduling strategies that attempt to increase performance of application even if fault occurs. For this reason we propose an information-based on line fault tolerant scheduler based on the WQR-FT able to achieve performance better than other scheduling strategies.

3. Information free Scheduling

Scheduling applications on a Grid is not a simple task. The set of Grid resources may greatly vary over time, the performance a resource delivers also varies from an application to another. The availability of correct information about both the resources as well as task is required to achieve performance in these situations for appropriate scheduling, such kind of schedulers are called information-based schedulers. Unfortunately, due to heterogeneity of Grid resources makes obtaining this information very difficult. The information-free schedulers are predominantly appealing as they do not rely on the information which is either concerned with the status of resources or the characteristics of applications for scheduling their decisions.

3.1. The WorkQueue with Replication(WQR) Scheduler

WorkQueue with Replication(WQR)[9] which enhances the classical WorkQueue(WQ) scheduler in which tasks are randomly chosen & are scheduled to the resources as they become available. In WQR tasks are being replicated on several resources till replication threshold. As multiple replicas are there to execute a task so there are chances that one of the instances may be executed on faster resource.

3.2. The WorkQueue with Replication–Fault Tolerant Scheduler(WQR-FT)

If a task fails then WQR is not able to complete the execution of that task. For successful completion of all tasks in a bag WQR-FT adds automatic restart to WQR. However if every time in case of task failure, execution begins from start then the work already done by it has wasted. So checkpointing[10] is also added to WQR-FT.

4. The WorkQueue with Dynamic Replication Fault Tolerant Scheduler

As per our study we have observed that information free schedulers are using much more resources than necessary so cannot exploit the full potential of Desktop Grids. For instance in WQR-FT various replicas of the same task are created & scheduled on several resources

Proposed scheduler, WQDR-FT: A Fault-Tolerant Scheduler with dynamic replication for BoT Applications
adds dynamic replication to WQR-FT which is beneficial in determining when & by how much the jobs need to be replicated.

By considering the average delay time for resource selection & dynamic threshold using replication for making the replication dynamic, WQDR-FT not only has the ability of efficient resource selection, but is also able to achieve improved performance as compared to the alternative scheduling strategies.

Following algorithm details the behavior of WQDR-FT.

1: Q {is the set of tasks}
2: M {is the set of resources indexed according to average delay time.}
3: DTHRPL {is the maximum number of available replicas calculated for each task}
4: NRRPL(t) {returns the number running replicas of task t}
5: getRscAvail(M) {returns the availability of resource}
6: deleteReplicas(t) {deletes all running replicas of task t}
7: allocate(m,t) {allocate task t to resource m}
8: allocateChk(m,t) {allocate t to resource m execution of task t will start from the checkpoint}
9: Chk(t,r) {true if task t & resource has checkpoint compatible}
10: getresourcelist()
11: while Q is null
12: getresourcelist();
13: maintain the list according to Resource history.
14: if resource is not in resource history keep them at last of list.
15: if (event == "RscAvail") then
16: m=getRscAvail(M); {m is a available resource}
17: t=popfront(Q); {extracts the first task t of the queue}
18: if (NRRPL(t) < DTHRPL) AND chk(t; r) then
19: allocateChk(r,t); {allocate t to resource r}
20: else if (NRRPL(t) < DTHRPL) AND NOT chk(t; r) then
21: allocate(r,t); {allocate task t to resource r}
22: end if
23: push(Q,t); {adds task t to the end of the queue}
24: else {event=="TaskComplete"}
25: deleteReplicas(t); {deletes all running replicas of task t}
26: end if
27: end while

Our algorithm attempts to locate most suitable resource for a task by calculating the average delay time of all the resources using the method CalAvgDelayTime(M,t) & will maintain a history of all the average delays using method UpdateHistory(R,D). Based on the history our scheduler assign the task to the most suitable resource having minimum average delay time & also calculates the Replication Threshold for each task by considering the average delay time of each resource maintained in the history file.

5. Results
The proposed heuristic is compared with WQR-FT for performance analysis by using Simulation environment known as GridSim Toolkit[11] and the experimental results shown in Figure (3.1),(3.2) & (3.3).
The factors used for simulation are shown in table 3.1

Table 3.1: Factors for simulation

<table>
<thead>
<tr>
<th>Factors</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>No of tasks</td>
<td>5</td>
</tr>
<tr>
<td>No of resources</td>
<td>4</td>
</tr>
<tr>
<td>Resource requirement (MIPS)</td>
<td>377</td>
</tr>
<tr>
<td>Task Workload (MI)</td>
<td>300,000–500,000</td>
</tr>
</tbody>
</table>

Figure 3.1: Comparison of WQR-FT & WQDR-FT w.r.t. Total CPU Time

Figure 3.2: Comparison of WQR-FT & WQDR-FT w.r.t. Total Waiting Time.
6. Conclusion

In this paper we have presented WQDR-FT, information based on line fault-tolerant scheduler with dynamic replication based on WQR-FT. By considering the average delay time for resource selection & dynamic threshold for replication for making the replication dynamic, WQDR-FT is also able to get performance better than other scheduling strategies.

References


