INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

A SURVEY ON JOB RESCHEDULING AND META-SCHEDULING APPROACHES IN GRID ENVIRONMENT

Nancy Priyadharshini U [1], Karthikeyan P [2]

[1] Post-Graduate Student, Department of Computer Science and Engineering, Karunya University, India
[2] Assistant professor, Department of Computer Science and Engineering, Karunya University, India

ABSTRACT*:* Grid computing is a emerging technology which has combination of heterogeneous and homogeneous resources. The Grid computing provides infrastructure to access and use distributed resources in a efficient way. To execute the application and schedule jobs in a grid computing environment has become a crucial task and leads to poor utilization of resources. To overcome this problem several Rescheduling algorithms and Meta-scheduling techniques have been proposed. In this paper, a comparative study has been done about some Rescheduling and Meta-scheduling algorithms that showing their efficiency of improving the poor resource utilization. This paper shows that Preventive and Reactive Rescheduling Approach is efficiently to tackle the poor resource utilization.

.

KEYWORDS: grid computing, meta-scheduling, rescheduling**.**

I. INTRODUCTION

Grid computing consists of geographically distributed heterogeneous resources. Grid system is a dynamic system in which resources may join and leave the system at any time. This dynamic and heterogeneity nature of Grid makes the system difficult to schedule resources which results in poor resource utilization and management. The scheduling in Grid computing is the process of allocating jobs to available resources. In order to perform proper scheduling of jobs to available resources several techniques has been developed but still it is impossible to achieve proper resource utilization. The poor resource utilization will become a challenge in provisioning the Quality of Service (QoS) to the Grid users. When resources are allocated for jobs, resources with good performance may get assigned to jobs thus overloading the resource and reducing the proper utilization of resources of the system. This can lead to reduced overall system performance. The solution to enhancing the provision of QoS is to perform Meta-scheduling and Rescheduling Jobs. To avoid problem in scheduling the jobs already scheduled are rescheduled using several methods of Rescheduling.

The Meta-scheduling is used to support scheduling of jobs by providing the information about available jobs and resources using Prediction mechanism. The Meta-scheduler provides services to control and monitor resources on remote Grids. This paper provides a study on the different algorithms that was proposed as a solution to the issue of poor resource utilization while scheduling of jobs to the available resources and the survey result shows that the Preventive and Reactive Rescheduling improves the resource utilization in a efficient way.

This paper is framed as follows. Section 2 includes a discussion on key concepts in this paper; section 3 gives a comparative study and section 4 gives a discussion on Preventive and Reactive rescheduling Approaches.

I. KEY CONCEPTS

A. Grid Computing

Grid is a system that coordinates distributed resources and uses standard, open, general purpose protocols and interfaces in order to deliver non-trivial qualities of service. Grid Computing is an emerging computing model that provides ability to perform higher throughput computing with the help of many networked computers which forms a model of virtual computer architecture that is able to distribute process execution across a parallel infrastructure. Grid system has several characteristics like dynamicity, heterogeneity, large scale, resource sharing etc. A grid system is said to be *dynamic* because of the rapid changes in resources and tasks. The grid is an set of distributed resources and tasks of wide variety connected by heterogeneous multi-level networks, which makes it *heterogeneous.* The grid system must provide the requirements of users in *large scale*, i.e. it should be capable enough to gathers together large number of resources and tasks and makes them accessible in a secure manner to users and applications.

B. Meta-Scheduling

Meta-Scheduling is the method of provisioning of QoS in Grids by Scheduling of Jobs when no suitable computing resource exists locally based on the information provided by the Predictor. The Meta-Scheduler supports resource discovery and matches it for applications based upon user defined requirements. There are several kinds of Meta-Scheduling which mainly consists of Inter-Domain Meta-Scheduling and Intra-Domain Meta-Scheduling

In *Intra-Domain Meta-Scheduling*, when a new Job is submitted by the user the Meta-Scheduler will check the available resources within the domain and allocates the Job to the resource which has the free slot to run that submitted Job. In *Inter-Domain Meta-Scheduling* [2], When Meta-Scheduler finds insufficient resource in its domain, and starts to communicate with one of its neighboring domain in order to gain access of resources from its neighboring domain. The examples of Meta-Scheduler are Grid way, Grid Service Broker, Grid Lab Resource Management System, and Community Scheduler Framework.

C. Rescheduling

Rescheduling is the method of Rescheduling of Jobs which are already scheduled in a certain time interval by using their start and end time information [4]. The Rescheduling process must determine whether the Rescheduling is profitable or not based on the information received from the Database. This process should also estimate the remaining work in the application and the effort required to moving the Job into the new resource.These are deployed for improved performance.

I. COMPARATIVE STUDY

This section includes a study on some of the Rescheduling and Meta-Scheduling, and its efficiency in improving the system performance

A. Rescheduling Methods in GrADS

The Rescheduling methods in Grid Application Development software (GrADS) [3], project has developed to provide programming tools and an execution environment to improve program development for the Grid. When the available resource which is already scheduled is insufficient for the new Job the Rescheduler will be initiated. The GrADS Rescheduler initially must determine whether the Rescheduling is beneficial based on the Sensor data and estimates the remaining work of the application in order to improve the system performance. If the Rescheduling is beneficial the Rescheduler computes a new schedule by communicating with Rescheduling Actuators located in each processor. The Actuators are responsible for initiating Job migration and load balancing. The GrADS project consists of Rescheduling by *Stop and Restart* method and Rescheduling by *Process Swapping* method. The first method to migrate Job is based on *Stop and Restart Rescheduling*. In this method the application will be suspended and migrated only when the better resource set is identified. When the Rescheduler of the *stop and restart* method identifies any application needs migration then the running application will be terminated. Before terminating the application it will processes to checkpoint data and terminate. The Rescheduled execution will then initiated by running the application on the new set of resources which then read the checkpoint data and continue its execution. The second method Process Swapping Rescheduler is initiated with application has more number of machines that are useful for computation. Some of these machines will be in *active set* i.e., machines become part of computation while some in *inactive set* i.e., some do nothing initially. During execution the monitoring process will be done periodically to check the performance of the machines and swaps slower machines in *active set* with faster machines in *inactive set*. So by doing this the system performance will be improved. A drawback recognized in *Stop and Restart* Rescheduling is very expensive because of large data transfer for each migration. The *Process swapping* Rescheduling is that it needs application modification. The Rescheduling in GrADS require much tuning and debugging to achieve acceptable resource utilization and application performance.

B. Dynamic Adaptation of Checkpoints and Rescheduling

The Dynamic Adaptation of Checkpoints and Rescheduling [1], ensures good performance fault tolerance in terms of Resource failure. The method implemented to achieve Fault Tolerance is periodically check pointing. That is the state of the Job is saved periodically. But an inappropriate Check pointing leads to delay in the Job execution and reduces the system performance. The Dynamic Adaptation of Checkpoints is based on the information maintained by information server about the current status and history of failure information of the resources. In case of resource failure while scheduling the Fault Index Based Rescheduling (FIBR) is initiated which is responsible for rescheduling the Jobs from the failed resources to some other resources and continues its execution from the last saved checkpoint. When the user submits the Job along with its start time and deadline the Job will be allocated to resource. The Resource Broker will then fix the response time within a time interval for the submitted Job. If the resource failed to get the result of its execution within the fixed time interval then it results in occurrence of fault. If the fault has occurred the fault index value of that resource will be incremented by 1, or decrements by 1 on successful completion and this value will be updated in the information server. When there is a resource failure the Rescheduler will initiates its Rescheduling process by communicating the information server. Based on the fault index value of the available resources in the information system the Job will be Rescheduled to some other available resources and continue its execution form the last saved checkpoint. The fault index value provides the rate of resource failure. Thus the FIBR increases the percentage of Job execution and resulting in improved resource utilization. A drawback of FIBR is it the fault index value has to be updated every time which may result in high processing time.

C. Low-Cost Rescheduling Policy

The Low-Cost Rescheduling Policy [5], performs Rescheduling of Jobs at a few selected points during execution. The workflow is modelled by means of Directed Acyclic Graph (DAG). In order to address the mapping problem in scheduling the Jobs are Rescheduled which may introduce re-evaluation cost as well as cost of transferring tasks across machines overhead to the scheduling and execution process. This mapping problem is reduced by Low-Cost

Rescheduling policy which improves initial static schedule of DAG by Rescheduling only selective tasks. The Selective Rescheduling Policy evaluates the start time of each tasks before its execution as start time in the static schedule and the slack in order to make decision for Rescheduling. The Rescheduler takes DAG, with its associated values and a static schedule as a input for Rescheduling. When the tasks of DAG are executed the Rescheduler maintains two schedules which contains static schedule constructed using estimated values as S_1 and information about the tasks that have finished its execution as S_2. Before each task can start execution, its start time will be considered as known. Comparing the start time estimated in S_1 and the slack the decision for Rescheduling is taken. If any delay occurs between the real and expected start time of the task is greater than the slack then the Rescheduling process will be initiated. The unexecuted tasks and the tasks whose executions has been completed are put into the scheduling algorithm to build a new schedule which is stored in S_1. The slack value for each tasks will be subsequently recomputed from S_1. The algorithm reduces the rescheduling overhead by performing Rescheduling for selective tasks. The disadvantage that it does not deal with dynamic scheduling and does not support management of run time information.

D. **Peer-to-Peer Meta-Scheduling**

Peer-to-Peer Meta-Scheduling [2] intended to manage QoS in Grids by performing Meta-Scheduling of Jobs to computing resources through interactions between administrative domains. This technique is implemented in an entity called Grid Network Broker (GNB). When the Meta-Scheduler receives a job to be scheduled and there is no suitable resource exists locally the Meta-Scheduler chooses one of the neighbour domains and forwards the request to the domain which has resource with the requested requirements. To implement this technique each domain must be capable of providing the advertised resource, the resource monitor should provide exactly same measurement of resources in all the domains and also authentication for security purpose. The Peer-to-Peer Meta-Scheduling achieves better rate of succeeded Jobs and better latencies. But the system has some disadvantages as well as it does not provide efficient communication in powerful Network with highly loaded computing resources and it does not be efficient for Query Forwarding.

E. **Network-Aware Meta-Scheduling in Advance**

The Meta-Scheduling in Advance [7], estimates the completion time of jobs based on the previous executions in order to predict the dynamic behaviour of the resources in the future. The estimation of completion time of the jobs are done by the Grid way Meta-Scheduler. There is an intermediate layer called Scheduler in Advance Layer (SA-Layer) which collects the information from Gap Management and Predictor. The Scheduler in Advance management communicates with the Predictor in order to get the previous application executions and the resource status. The information about the previous application executions are stored in Database Execution and the status of the Resources are stored in Database Resource to estimate the completion time. Both the DB Execution and DB Resource are maintained by the Predictor. The intermediator then communicates with the Gap Management in order to get details about the available free slots in the resources. Based on the information gathered from Gap management and Predictor the characteristics of the jobs and the power of the CPU of the resources and network future status are predicted by the Meta-Scheduler. So the allocation of jobs by the Meta-Scheduler will be done efficiently. It accomplishes good resource utilization. A drawback in this method is Time Complexity is high.

F. **Meta-Scheduling in Advance with Autonomous Self-Tuning System**

Meta-Scheduling in Advance with Autonomous Self-Tuning System [6] inherits the features of Network-Aware Meta-Scheduling in advance to provide QoS. This technique is used to estimate the completion time of the jobs and autonomous behaviour is obtained by computing a trust for each resource and Rescheduling of failed jobs. The Resource Trust means accurate previous estimation made for the jobs executed in each resource. The Meta-Scheduler gathers information regarding free slots, resource status, previous execution time from Gap Management and Predictor and then calculates the completion time of the jobs[7]. This system presents an autonomic behaviour which allows the system to adapt itself to changes in the Grid system. The autonomous behaviour of the system is

based on Resource Trust and Rescheduling functionalities. The Resource Trust is calculated by comparing the estimated completion time and real completion time. This estimation reveals the errors made in the prediction for each resource. When the resource fails or shut down the jobs scheduled on it will be rescheduled to other hosts by the Job Rescheduler. The Rescheduling is done by monitoring the currently active resource and checking its time slots. When the resource become unavailable the Rescheduler selects the jobs already scheduled and releases its reserved time slots by deleting the previous scheduled decisions and performs rescheduling of jobs to other resources. Based on the Resource Trust and Job Rescheduling the system tunes itself to the changes made in the Grid system. The system tackle variable availability of resources by means of Rescheduling of failed jobs thus improves the autonomic behavior of the system. A setback in this technique is poor estimation of transfer times.

I. PREVENTIVE AND REACTIVE RESCHEDULING APPROACHES

In Preventive and Reactive Rescheduling Approaches [4], have been developed to provision QoS by reducing the resource fragmentation which leads to poor resource utilization. The Rescheduling is done with the help of middleware that supports Meta-Scheduling in Advance. The Meta-Scheduler collects information from the gap management and predictor in order to perform Rescheduling. The Job Rescheduler gets the free slots, resource status, previous execution information from the Meta-Scheduler and performs Rescheduling of jobs. In Preventive Technique the jobs are rescheduled by sorting the tasks already scheduled based on their start time in a certain time interval. The Reactive Technique will be initiated whenever the job allocation done by Preventive Technique fails and it reallocates already scheduled jobs and uses its released time slots for allocating new incoming jobs. The Reactive Technique performs Rescheduling by checking which job has more probability of being reallocated and which job has assigned time slots which could be useful to allocate the new incoming job. The Preventive and Reactive Rescheduling approaches increase both the scheduled job rate and resource usage thus improves the resource utilization

PARAMETERS/PAPERS	CENTRALIZED/ DISTRIBUTED	DYNAMICITY	ALLOCATION EFFICIENCY	RESOURCE UTILIZATION
Rescheduling Methods in GrADS	Distributed	Medium	Medium	Low
Dynamic Adaptation of Checkpoints and Rescheduling	Distributed	High	Low	Medium
Low-Cost Rescheduling Policy	Centralized	Low	Medium	Medium
Peer-to-Peer Meta-Scheduling	Distributed	High	Low	Low
Network-Aware Meta-Scheduling in Advance	Distributed	Medium	Medium	Low
Meta-Scheduling in Advance with Autonomous Self-Tuning System	Distributed	High	Medium	Medium

Preventive and Reactive Rescheduling Approaches	Distributed	High	High	High

CONCLUSION

In this paper some of the Rescheduling and Meta-Scheduling Techniques that accomplish proper resource utilization have been discussed. A comparative study has been made regarding the merits and demerits of all the techniques. The comparisons prove that Preventive and Reactive Rescheduling approach can overcome many of the flaws in the techniques discussed. It is an efficient model for resource management in Grid Environment.

REFERENCES

[1] Antony Lidya Therasa.S, Sumathi.G, Antony Dalya.S (May 2010) "Dynamic Adaptation of Checkpoints and Rescheduling in Grid Computing" , International Journal of Computer Applications (0975 – 8887) Volume 2 – No.3

[2] Caminero, A., Rana, O., Caminero, B., Carrion, C. (2011) "Network-aware heuristics for inter-domain metascheduling in Grids" ,J. Comput. Syst. Sci. **77**(2), 262– 281.

[3] F. Berman, H. Casanova, A Chien, K. Cooper, H. Dail, A. Dasgupta, W. Deng, J. Dongarra, Johnsson, K. Kennedy, C. Koelbel, B. Liu, X. Liu, A. Mandal, G. Marin, M. Mazina, J. Mellor-Crummey, C. Mendes, A. Olugbile, M. Patel, D. Reed, Z. Shi, O. Sievert, H. Xia, and A.YarKhan (June 2005) "New Grid Scheduling and Rescheduling Methods in the GrADS Project" ,International Journal of Parallel Programming, Vol. 33, Nos. 2/3, DOI: 10.1007/s10766-005-3584-4.

[4] Luis Tomas · Blanca Caminero , Carmen Carrion, Agustín C, Caminero (2012) "On the Improvement of Grid Resource Utilization: Preventive and Reactive Rescheduling Approaches" , J Grid Computing 10:475–499 DOI 10.1007/s10723-012-9226-3.

[5] Rizos Sakellariou and Henan Zhao (2005) "A Low-Cost Rescheduling Policy for Efficient Mapping of Workflows on Grid Systems" School of Computer Science, University of Manchester Oxford Road, Manchester M13 9PL, UK.

[6] Tomas, L., Caminero, A.C., Caminero, B (2011) "Network-aware meta-scheduling in advance with autonomous self-tuning system" ,Future Gener. Comput. Syst. **27**(5), 486–497.

[7] Tomas, L., Caminero, A., Caminero, B., C.(2010) "Using network information to perform metascheduling in advance in Grids" ,In: Proc. of the Sixteenth International Conference on Parallel Computing (Euro-Par). Ischia, Italy.