A Hybrid Scheduling Algorithm to Achieve Fault Tolerance in Grids

A. Jainul Fathima, G. Murugaboopathi

Abstract— Grid is a computational infrastructure that provides ability to securely integrate large amount of computing resources to handle workloads that are geographically dispersed. The performance of grid is usually measured based on its complex workflow, criticality and fault tolerance property. Formally, fault tolerant is achieved by checkpointing and replication of task periodically. The major drawback with these technologies is that they produce run time overhead. To overcome the drawback, this paper proposes an algorithm that dynamically implements checkpointing and replication and provides high job throughput in the existence of failure and improves the performance of heterogeneous grids. The Simulation studies are carried out to evaluate the proposed algorithm. The results show that combined dynamic approaches improves the fault-tolerant property in the simulated grid environment. It is also inferred that the system performance depend on workload, and failure frequency.

Keywords— Distributed computing; Fault tolerance; GridSim

I. INTRODUCTION

Grid Computing provides an infrastructure that enables the integrated, collaborative use of databases, network and computer systems. Heterogeneous resources and decentralized management are the major cause of complexity in grids [1]. The availability of grid depends on network failure rates and variation in system load. Delays are not acceptable for time-related jobs, hence fault tolerance should be considered. Fault tolerant in grid while considering the resources and job execution is the major objective to be noted. To achieve it, checkpointing and replication techniques are applied. The major drawback with this approach is that these techniques cannot handle unanticipated failure condition in the grids [2]. Therefore, we proposed an algorithm that dynamically adjusts checkpointing frequency and replication count based on system properties. The proposed algorithm will handle hardware failure in grids.

The hybrid algorithm monitors grid state, characteristics of job and historical information to take decision**.** The simulation studies are carried out using GridSim tool and real world logs from large scale distributed system taken as dataset. The algorithm implemented achieve high throughput in case of any failure.

This paper is structured as follows: Section 2 discusses on grid model and related work.

Revised Manuscript Received on December 16, 2019.

* Correspondence Author

A.Jainul Fathima *, Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education, Krishnankoil, India. Email:ajainulfathima@gmail.com.

G.Murugaboopathi, Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education, Krishnankoil, India.. Email: gmurugaboopathi@gmail.com

Following that in section 3, detailed checkpointing and replication algorithm are explained. Section 4, elaborate on hybrid algorithm combining adaptive checkpointing and replication. Section 5 concludes the paper.

II. GRID MODEL & RELATED WORK

A. Grid Model

A grid is composed of heterogeneous, separately handled subsystems in a distributed computing environment. The Figure1 depicts a grid model that consists of 'n' number of users and they are geographically dispersed and connected over Wide Area Network. Each user can create 'n' number of jobs and these jobs are submitted to the scheduler. In the scheduler the job arrival pattern is monitored. Information service will provide the resource status information. Based on the information provided by the service provider, a set of rules based on checkpointing and replication is executed and job- resource matchmaking will be done and jobs will be submitted to the node where the jobs will be computed and all these details will be periodically stored in the checkpoint server where the data will be made persistent[3].

The grid provides a distributed working environment, while resources in Local Area Networks (LAN) share resources which makes super power working environment. The resources can be memory, CPU power and database. For realizing fault tolerance

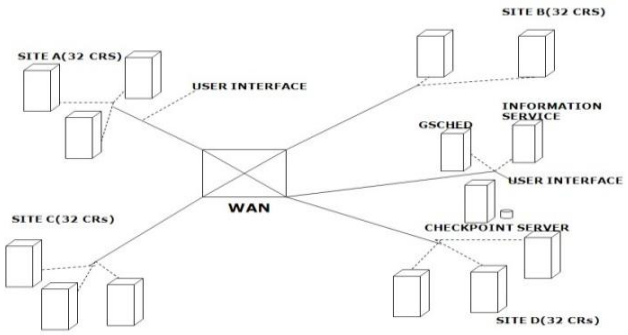

Figure 1 : Grid Model

in grid, it is made assumption that Computational Resources(CR) are unstable, with a resource failure affecting all CPUs within a given CR. Differing to the traditional environment, failures considered in this work are independent and spatially and temporarily coerrlated.

B. Related Work

Realizing a fault tolerance in grid become an major research area in grid computing environment. Formally fault tolerance is achieved by focusing on checkpointing and replication of active jobs in grid. Consideration on various fault tolerance mechanism in grid computing were done by many researchers. Many

research focus on implementing checkpointing and replication techniques in grid environment. In this paper [4] the authors developed a fault tolerant grid model and also gives information about various methods to provide fault tolerance in software grid application. Here, the author implemented an approach that combines replication with dynamic scheduling of resources. Also this paper suggested to implement working fault tolerant model. In another research [5] the author proposed a technique that combines job scheduling with replication to execute jobs efficiently. In another research [6] fault recovery mechanism is implemented in grid by task scheduling optimization model. And the research mainly focuses on software failures. Fault detection methods are designed and implemented in many research and that focuses on fault prediction and recovery mechanism [7]. The main focus of this research is to improve the performance of the system in case of failure and also to handle overhead time. The strength of the algorithm contingent upon adjusting checkpointing interval and replication of jobs. Finding best possible values require understating the application and grid environment.

Many research focus on techniques to handle checkpointing overhead. One such method is incremental checkpointing, briefly implemented in [8]. This concept decrease the data stored by changing the block of memory from the time the last checkpoint. In another research [9] checkpointing overhead is brought down during execution and allows recovery with the help of logging messages which is applicable for homogenous working environment. Another research [10] is carried out to find the most favorable checkpointing frequency. They provide analytical solution to particular system hypothesis. Also, to find the suboptimal checkpointing sequence min-max checkpointing concept is implemented [11]. Similarly finding the optimal job replication is also a issue and many research focus on it. In the paper [12] the author implemented dynamic replication method. The drawback of this approach is it cannot be implemented for heavily loaded job. Another interesting research is group based dynamic replication approach which will consider the reliability of the system [13]. Combined checkpointing and replication algorithms are reported in many research [14], [15] that focuses on transient faults. With this knowledge on checkpointing and replication concept, our objective is to develop a hybrid algorithm that has enhanced resource utilization and job execution time.

III. HYBRID SCHEDULING ALGORITHM

This algorithm is a combination of checkpointing and Replication method which dynamically switches between the two mode. Considering the following scenario, When the grid environment has 2 resources represented as R1, R2. Each resources can run two jobs at same time. Hence the system is initialized as Min Rep=1, Max Rep=2 and Free Active CPU (ACPU) =2. The switching of both checkpointing and replication happens in following condition. Checkpoint mode is executed when the number of ACPU is less than active free CPU. When processing the next jobs, following condition can occur.

Step 1: Active Job Replica [(AJR) > 0]: Checkpoint the active replica and stop other replicas

Step 2: [AJR=0]: In this case, if it has more active CPU. Hence execute the job in least loaded site

Step 3: [AJR = 0]: In the other situation, When there is no active CPU randomly select a replicated job and checkpoint it.

Step 4: When the system load decreases the algorithm works in replication mode and handles failure. During replication mode, all the active jobs with Min Replicas are submitted to active resources

Step 5: Finally, the job is assigned to fastest resource and job is executed in checkpoint mode and stops execution. Thus dynamically switching from replication to checkpoint mode.

A. Simulation Environment

The algorithm is simulated in GridSim tool kit. GridSim affirms modeling and simulation of heterogeneous resources. Scheduling algorithm can be modeled and simulated for distributed applications. Usually in the distributed environment, resources are geographically distributed managed by multiple domains. The Schedulers in grid environment is responsible to enhance the system performance. The tool kit also provides capability to model and simulate network connectivity with different domains. The installation parameters are well explained in [16].

As explained in the introduction part, the grid model has four sites modeled to handle failure and restore behavior. Failure frequency is modeled with Weibull distribution with decreasing hazard rate. Mean repair time, in turn, varies across the sites from less than a minute to more than a hour and is modeled by a logarithmic distribution. The considered grid system has a total availability of 90%. Time between failure is not modeled well by an exponential distribution, which accord with earlier findings for other types of systems. The time between failure at individual nodes, as well as at an entire system, is fit well by a gamma or Weibull distribution with decreasing hazard rate. The Resource availability is computed using the formula

$$Ar = \left(\left(1 - \left(\sum_{n=1}^{N} (tf_{r,n} - tr_{r,n}) / T_{sim} \right) \right) * 100 \right)$$

where N is the Number of resource failure, tf r,n is the time stamp of the resource failure, tr r,n is the timestamp of restore and Tsim - total simulation time. So the total grid availability (A grid)

$$Agrid = \left(\left(\sum_{r=1}^{R} Ar \right) / (T_{sim} * R) \right) * 100$$

where R is the number of resource in the grid

B. Simulation Results

The user makes the job and submits to the scheduler. The scheduler will schedule it to the resource which has processing elements to process thejob. To find the fault-tolerant of the grid environment some machines should be failed randomly so to fail a machine in the grid environment. The GIS global information server uses Weibull failure models for doing the following that includes three tasks, first model sample indicate when the resource fail andsecond model sample how long the resource fail should be failed status and the third model indicate how many number of machines to failure in the resource.

The GIS takes each one sample from these three Weibull model at the same time and select a resource which is in working condition for failure. And the GIS will send the failure signal, FAILURE-MACHINE to the selected resource.

On receiving the failure signal, the resource change the machine status into failed. The failed machine partly worked jobs are return to scheduler with failure status. In the simulation, we have used four rate of failure by reducing sampling interval of GIS for failure signaling. The four type of failure rate of machines made by changing sampling interval creates gird system failure environments and used for replication, checkpointing and combined approach. The table 1 shown below displays the recorded values for these four types of failure models

Table 1 – Recorded values for the four types of failure model

Failure Model	Number of Machine Failed	MTBF in hours	Failure rate in failure/hours	Availability in percentage
Low	1075.0	44.65	0.0223	88.05
Normal	1346.0	35.66	0.028	82.12
Medium	1830.0	26.23	0.038	68.15
High	2720.0	17.64	0.0566	27.792

In the simulated grid model four types of failure rates are considered that includes low, normal, medium and high failure rates. Model 1 is designed with low failure rate, and model 2 with normal failure rate, model 3 with medium failure rate and model 4 with high failure rate. To compare the performance of proposed hybrid approach several CheckPointing(CP) approach like last failure CP, Mean failure CP and Periodic CP are considered**.** It is assumed that jobs running on single node follows Lublin job generation model for the submitted workload. The best result can be provided for short checkpointing interval and when the failures occur in particular period of time and the algorithm can easily predict the failure. The same concept will be applied for Last Failure CP. In case of dynamic scheduling , Mean failure CP is efficient. Random checkpointing frequency starts and results in executed jobs and average job run-time and the outcome obtained by periodic CP with best performing CP interval. The small decline in the count of checkpoints taken by Mean Failure CP get best checkpointing values. This decrease can be explained by a shorter calibration period required to achieve the "optimal" value

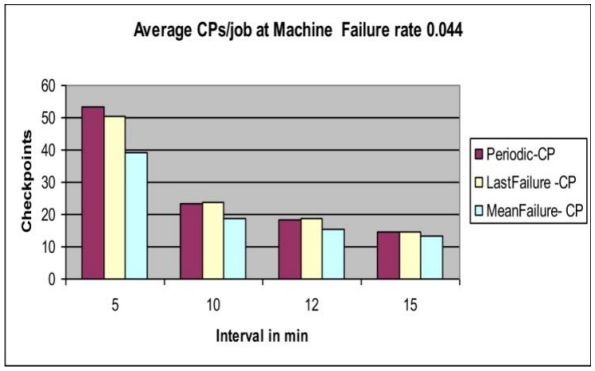

Fig 2 : Average CPs job at machine failure rate 0.044

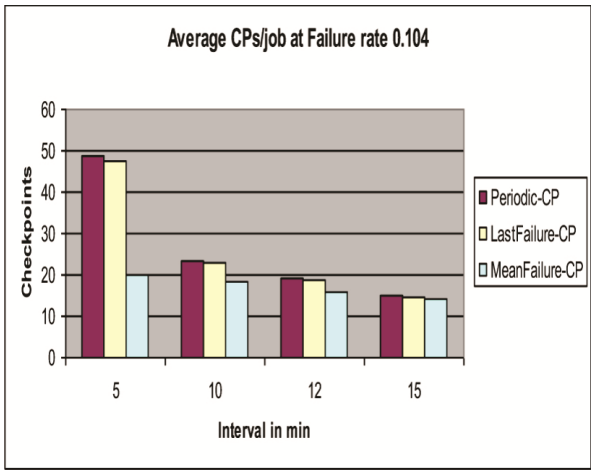

Fig 2.1 : Average CPs job at Failure rate 0.104

IV. RESULTS AND DISCUSSION

In this section the implementation of replication based and hybrid approach is compared with best checkpointing heuristic. The simulation is performed within grid systems with unstable load and availability. Here we considered both replication algorithms are considered namely Load dependent replication and failure dependent replication. The minimum replica and maximum replica of job replicas set to 1 and 2, and the free CPU limit initialized to 40. The combined approach is initialized with the same replication parameters as Failure DependentRep and switches in the checkpointing mode to the MeanFailureCP approach. The chosen parameter values for the replication-based heuristics are not necessarily optimal but they are believed to be reasonable for the case at hand. Finally, the throughput and average job execution times generated by combined for both types of system load are located, as can be observed in graph, between respectively the throughputs and average job execution times of FailureDependentRep and MeanFailureCP. This the logical consequence of the fact that job submissions are clustered in time and that the heuristic performs some calibrations, after each variation in the system load, before achieving its "optimal" state. Regarding the other observed performance parameters, combined is almost fully fault-tolerant and results in one of the best average job lengths among the considered algorithms.

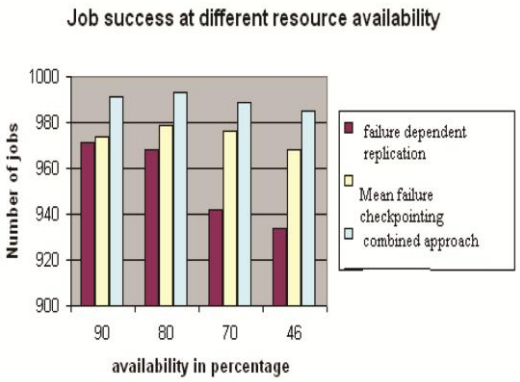

Fig 3: Job Success at different resource availability

Fig.3 shows the jobs success at different resource availability. The graph is plotted with resource availability on X-axis and number of jobs in the y-axis. And thenGraph also compares the best checkpointing heuristics (MeanFailureCP), best replication based heuristics(failureDependent Rep) with the combined approach. When the resource availability decreases the performance of combined approach is better than the checkpointing alone and replication alone heuristics.

V. CONCLUSION AND FUTURE WORK

Observing Fault tolerance in grids is the major research objective in distributed computing environment. It is a major challenges, when the resources are heterogeneous and is a research objective, detailed study on checkpointing and replication is carried out. As a result this paper proposed an hybrid algorithm and the heuristics are evaluated in GridSim simulator. The run-time overhead feature to periodic checkpointing can significantly abridged when the checkpointing frequency is dynamically modified in function of resource stability and lasting job execution time. Finally, the proposed hybrid approach is best suited for distributed systems. It is observed that checkpointing can be used when there is high fault rate.Replication can be implemented in case of low fault rate and low downtime. When size of checkpoint is small the response time is less and the downtime increases. And hence the fault-tolerance in the grid was realized. Here in this paper the checkpointing interval has been fixed by heuristic method which is not suitable for all the environments and also the fixed interval does not give best results for the other environment. As a future part of work, a novel scheduling and intelligent methods can be proposed that suits to dynamically varying execution time of job.

REFERENCES

[1] Chtepen, M., Claeys, F., Dhoedt, B., Turck, F. D., Demeester, P., & Vanrolleghem, P. (2009). Adaptive Task Checkpointing and Replication: Toward Efficient Fault-Tolerant Grids. *IEEE Transactions on Parallel and Distributed Systems, 20*(2), 180-190. doi:10.1109/tpds.2008.93.

[2] Derbal, Y. (2006). A new fault-tolerance framework for grid computing. *Multiagent and Grid Systems, 2*(2), 115-133. doi:10.3233/mgs-2006-2203.

[3] Li, H., & Buyya, R. (2007). Model-Driven Simulation of Grid Scheduling Strategies. *Third IEEE International Conference on E-Science and Grid Computing (e-Science 2007)*. doi:10.1109/e-science.2007.51

[4] Camargo, R. Y., Goldchleger, A., Kon, F., & Goldman, A. (2004). Checkpointing-based rollback recovery for parallel applications on the InteGrade grid middleware. *Proceedings of the 2nd Workshop on Middleware for Grid Computing* -. doi:10.1145/1028493.1028499.

[5] Hwang, S., & Kesselman, C. (2003). A Flexible Framework for Fault Tolerance in the Grid. *Journal of Grid Computing, 1*(3), 251-272. doi:10.1023/b:grid.0000035187.54694.75.

[6] Masubuchi, Y., Hoshina, S., Shimada, T., Hirayama, B., & Kato, N. (n.d.). Fault recovery mechanism for multiprocessor servers. *Proceedings of IEEE 27th International Symposium on Fault Tolerant Computing*. doi:10.1109/ftcs.1997.614091.

[7] Kohila, R. (2014). Efficient Resource Management Mechanism with Fault Tolerant Model for Computational Grids. *International Journal of Computer Applications Technology and Research, 3*(12), 774-777. doi:10.7753/ijcatr0312.1004.

[8] 8 Agarwal, S., Garg, R., Gupta, M. S., & Moreira, J. E. (2004). Adaptive incremental checkpointing for massively parallel systems. *Proceedings of the 18th Annual International Conference on Supercomputing - ICS 04*. doi:10.1145/1006209.1006248 .

[9] Young, J. W. (1974). A first order approximation to the optimum checkpoint interval. *Communications of the ACM, 17*(9), 530-531. doi:10.1145/361147.361115.

[10] Tantawi, A. N., & Ruschitzka, M. (1984). Performance analysis of checkpointing strategies. *ACM Transactions on Computer Systems, 2*(2), 123-144. doi:10.1145/190.357398.

[11] Lan, Z., Li, Y., Zheng, Z., & Gujrati, P. (2008). Enhancing application robustness through adaptive fault tolerance. *2008 IEEE International Symposium on Parallel and Distributed Processing*. doi:10.1109/ipdps.2008.4536383.

[12] Hou, C., & Shin, K. (n.d.). Replication and allocation of task modules in distributed real-time systems. *Proceedings of IEEE 24th International Symposium on Fault- Tolerant Computing*. doi:10.1109/ftcs.1994.315660.

[13] 14. Choi, S. J., Baik, M., Gil, J., Park, C., Jung, S., & Hwang, C. (2006). Group-based dynamic computational replication mechanism in peer-to-peer grid computing. *Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID06)*. doi:10.1109/ccgrid.2006.1630902.

[14] Li, Z., Xiang, Y., & Chen, H. (2006). Performance Optimization of Checkpointing Schemes with Task Duplication. *First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS06)*. doi:10.1109/imsccs.2006.250.

[15] 16. R. Buyya & M. Murshed,(2002). GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing," *J. Concurrency and Computation: Practice and Experience, vol. 14, pp. 13-15.*

AUTHORS PROFILE

A.JainulFathima received her B.Tech. degree in Information Technology from Anna University- Chennai in 2007and M.Tech degree in Computer Science and Engineering from Anna University – Tirunelveli in 2009 . She has 3 years of teaching experience. She is currently pursuing Ph.D. degree in Kalasalingam Academy of Research and Education , Krishnankoil. Her Research area includes Big data analytics, Computational Drug discovery, andBioinformatics.She is a Life Member of the Indian Society for Technical Education (ISTE).

Dr.G.Murugaboopathi received the Undergraduate Degree in Computer Science and Engineering from Madurai Kamaraj University in 2000, the Post Graduate degree in Digital Communication and Network from Madurai Kamaraj University in 2002 and Ph.D in Computer Science and Engineering at Bharath University, Chennai. He has more than 45 publications in National, International Conference and International Journal proceedings. He has more than 15 years of teaching experience. His areas of interest include Wireless Sensor Networks, Bioinformatics.Mobile Communication, Mobile Adhoc Networks, Mobile Computing, Cloud Computing, Network Security, Network and Data Security, Cryptography and Network security. He is currently working as an Associate Professor in the Department of Computer Science and Engineering at Kalasalingam Academy of Research and Education, Tamil Nadu,India