ISSN No. 0976-5697

Volume 2, No. 6, Nov-Dec 2011

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Hybrid Fault Tolerance Scheme in Grid Computing

Deepak Srivastava*, Ritu Garg
Department of Computer Engineering National Institute of Technology
Kurukshetra, Haryana, India – 136119
*deepak05cs23@gmail.com
ritu.59@gmail.com

***Abstract*—** An important goal in Grid environment is to construct foolproof fault tolerant system. Fault tolerance service is a mechanism that recovers the system following the occurrence of faults. In this paper, we suggest an improved hybrid checkpoint fault tolerance technique. In such scheme two types of checkpoints, either Cooperative checkpointing with Store-checkpoint (SCPs) or Cooperative checkpointing with Compare-checkpoint (CCPs) are inserted between store-and-compare-checkpoint (CSCPs). SCP and CCP are used to provide reduction in execution time by dynamically adjusting checkpoint interval and in Cooperative checkpoint, less likely checkpoints can be skipped and hence by combining these approaches overall performance is improved. Finally, this paper presents a simulation-based experimental result showing comparisons of performance of other technique too.

***Keywords*—**Grid computing, Checkpoint technique, Cooperative checkpoint, Simulation.

I. INTRODUCTION

In Grid environment, usage of checkpoints reduces the time to restart a task from the beginning in the presence of any random failure. In Checkpoint technique, a task is divided in optimum equal time intervals and on each interval checkpoint is assigned. The main checkpoint purpose is to detect random failure and to avoid large rollback time. At each checkpoint, we save the state of the task. When any fault occur between two checkpoint, we rollback to the last saved checkpoint, through which we avoid the rollback from start and save the rollback time.

CSCP is used to compare state of processor and store their state, Store-checkpoint (SCPs) in checkpoint schemes are used for storing the state and Compare-checkpoint (CCPs) are comparing state [2]. Each checkpoint in grid environment is having 2 purposes: the first is to save the state and introducing correct state to avoid rollback from start. And second is to find out random fault, occurring in the processor state.[5] Cooperative checkpoint provide a better performance in which checkpoint requests may be skipped , which depend on the programmer , compiler , and the run time system. All decision of accepting request or skipping checkpoint is taken by them. At run time, dynamically application of accessing checkpoint or denying checkpoint is define by gatekeeper, the whole mechanism of decision of accepting or denying of checkpoint request is performed by gatekeeper [1] .Without co-operative checkpoint, there may be some checkpoint which are useless, and just create overhead and effect overall performance of grid system.

II. TERMS AND DEFINITION

A. SCPs and CCPs:

Suppose T, is native execution time of completion of task, which does not include any fault occurrence and hence no retry and checkpoint schemes are used. If fault arrive with a constant failure rate λ ($\lambda>0$). Then checkpoint are placed with constant interval of $k= \sqrt{(2C/\lambda)}$ [5], where C is overhead of a checkpoint, time to store state and compare state. It is having different value for different checkpoint characteristics. We have taken number of expected random faults as n. Total time to store-compare state and then rollback to their last saved checkpoint, is define as z. When CCPs or SCPs are inserted between CSCPs, then we get reduction in execution time of grid system. Using SCPs, when CSCPs detects any error, then task rollback to most recent SCP instead of CSCP to save rollback time. In CCPs, when CCPs detects any error, then task rollback to most recent CSCP, it provide a quick detection of failure .Each CSCP interval of checkpoint is divided into P equal parts, where length of equal parts P1 is equal to (P1=k/p). All parameters value has defined later through table-{1, 2} in this paper. It plays a key role in reducing task execution time in terms of task level fault tolerance. Using such a scheme in which we place SCPs or CCPs between consecutive CSCPs, respectively in Grid workflow systems. We achieve a substantial decrement in the execution time by applying this checkpointing scheme. Checkpointing intervals are adjusted based on the frequency of fault occurrences and the amount of time remaining before the task deadline. When checkpoints are associated by using SCPs-CCPs applied to any operation, then it takes less time and hence can be used more frequently than the checkpoints associated with operation that takes more time.

CONFERENCE PAPER
International Conference on Issues & Challenges in Networking, Intelligence & Computing Technologies
Organized by Krishna Institute of Engineering and Technology (KIET) Ghaziabad, India

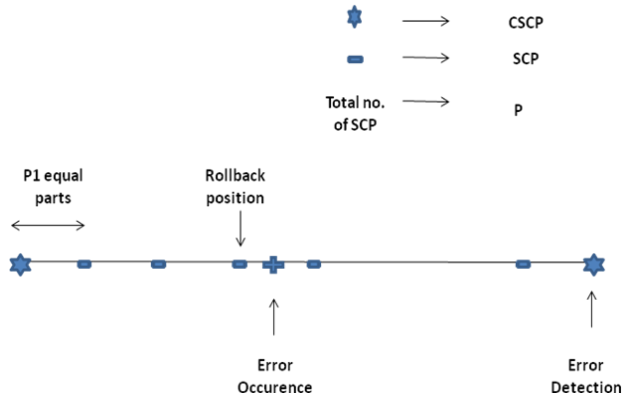

Figure.1: SCPs are placed between CSCPs. When any two CSCP state are not agree at any time, then we find out last SCP with identical state and rollback to it, instead of rollback to last CSCP. Hence provide better performance in execution time of task.

B. Cooperative Checkpoint:

When any task is executing on any processor, programmer, compiler and system jointly decide checkpoint position. At run time checkpoint request is given and either granted or denied depend on system [1]. Application programmer placed checkpoint request according to constant interval of k. The system receives the checkpoint request and this may be granted or denied, based on I/O traffic, critical event predictions, and user requirement. All this mechanism of accessing of granted or denied of request is handled by gatekeeper.

As shown in the below figure:2, task is performing at any resource and checkpoint is placed as typical periodic behavior. At each checkpoint, task is storing their state .When error occur in task then it rollback to their last saved checkpoint, and then task perform their execution. But if no more error occurrence left, then there is wastage of time to store state on checkpoint, so 2C wasted work. While using cooperative checkpointing ,when their request is granted then at run time checkpoint is placed. When last random error occur as per the failure rate , then gatekeeper will denied all the request of checkpoint, and hence there is saving of time of storing state on checkpoint, which provide a better performance in grid application.[1].

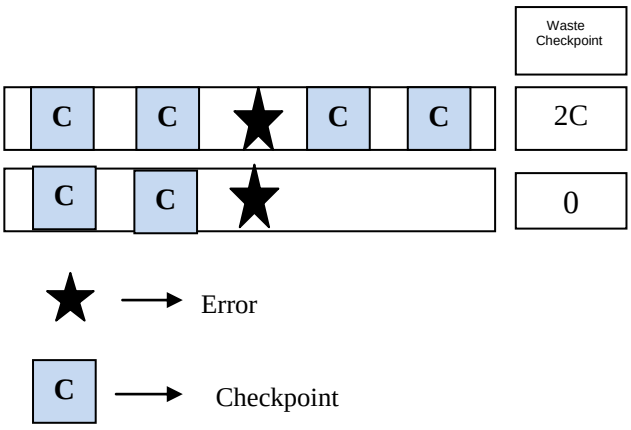

Figure.2: Task is running on resource, after occurrence of failure, other 2 checkpoints are wasted and no need of saving state on these checkpoint. After occurrence of failure using co-operative checkpoint, request of checkpoint is denied and hence time of storing state is saved.

III. PROPOSED HYBRID CHECKPOINTING SCHEME

Cooperative checkpoint is useful in providing better mean execution time by denying unlikely checkpoint request at runtime , through which state storage time on useless checkpoint is saved. Checkpoint request at run time is accepted as an upper bound on the remaining number of faults is not equal to zero. In CSPs-SCPs, we save rollback time, as rollback to last most identical SCP state instead of last saved CSCP. Using either adaptive SCPs scheme or adaptive CCPs scheme in checkpointing, we receive near-by same result. In this paper, we simulate cooperative checkpointing with adaptive SCPs scheme and believe same result for CCPs scheme. In our proposed scheme, we suggest a hybrid technique, combining both the techniques for a single task and perform simulation on different resources. By combining both techniques (Cooperative checkpoint and inserting either SCPs between CSCPs), we achieve better execution time. In this scheme , all checkpoint is placed dynamically at run time and when all random failure occurrence in task execution is over, then other next checkpoints request are denied , by which time of storing state on these checkpoint is saved. And by using SCPs, we save rollback time also, as by rollback to most identical SCP state instead of last saved CSCPs. Here, additional SCPs scheme in Grid workflow systems which overhead time is mainly determined by the time to compare processors' states is used. Hence by combining both techniques, overall performance is improved in terms of task level fault tolerance. Fault is injected into resource by Poisson process of constant arrival rate λ and Mean Time to Failure (MTTF) is taken as (1/λ)[4]. Number of random failure occurrence can be find out by failure rate of the Poisson process. Here we have taken time to store state on checkpoint as c=0.5 and rollback time as r= 0.5[4] and t_{csp}=0.45 and t_{scp}=0.5[2].

IV. EVALUATION

In this section, a set of experiment is done with many resources. Due to random nature of fault occurrence, the experiments are repeated many times for the same task and average value of result is taken on different failure arrival rate as governed by Poisson distribution [9]. Execution time without failure is T. Number of expected random failure occurrence due to failure rate of resource is n=λ*T. The experimental values of simulation are given below:-

Table 1: Comparison of execution time of various techniques fault tolerance

Sn	Failure Rate(λ)	Checkpoint Interval k= √(2c/λ)	Checkpoint technique	Cooperative Checkpoint technique	CSPs-SCPs technique	Cooperative Checkpoint with CSPs-SCPs
1	0.06	4.0	124.0	122.3	122.5	120.7
2	0.05	4.4	123.6	121.2	121.0	118.5
3	0.04	5.0	120.7	117.8	119.3	116.5
4	0.03	5.7	117.7	115.7	115.9	113.8

Table.2: Saved work in sec. as comparison to Checkpoint fault tolerance technique.

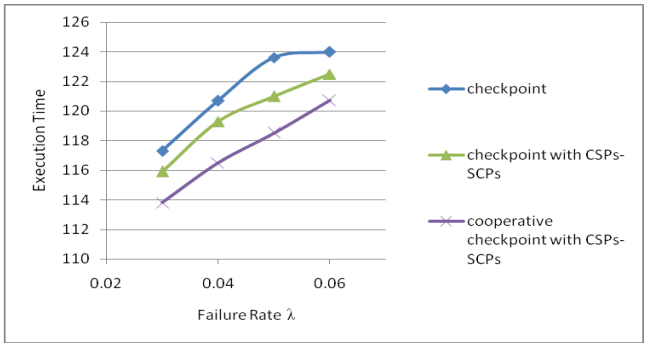

Figure.3: Comparison of execution time as the function of failure rate of all techniques.

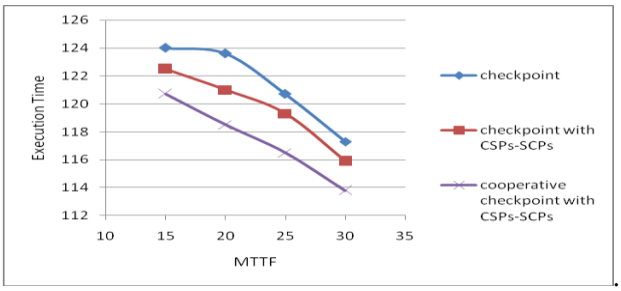

Figure.4: Comparison of execution time as the function of mean time to failure (MTTF) with all techniques.

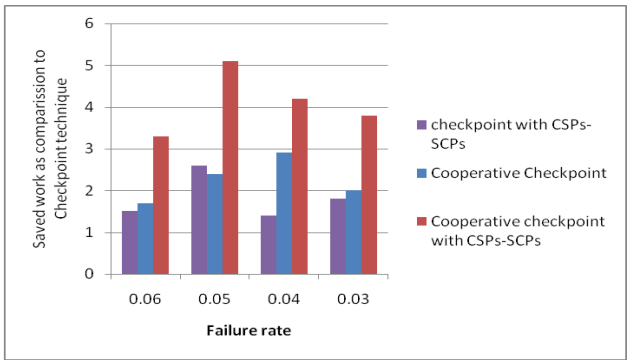

Figure.5: Saved work of all techniques as comparision to checkpoint technique.

From the Table-{1,2} and Figure-{3,4,5} , it is clear that by using Cooperative checkpoint with either SCPs-CCPs, execution time of task is improved for given parameters as compared to other existing techniques.

V. CONCLUSION

In this paper we proposed a hybrid Fault tolerance technique for grid environment. We have evaluated the performance of proposed strategy and compared with other existing techniques through simulation using GridSim Toolkit-4.0. The proposed technique is done by combining Cooperative checkpoint and inserting SCPs or CCPs between CSCPs.We evaluated the performance of our proposed technique and compare with different techniques using different parameters such as failure rate, mean time to failure(MTTF) and saved work in second. The experimental results clearly show that proposed technique yields better results in all given conditions than other techniques.

Sn	Failure Rate(λ)	CSPs-SCPs technique	Cooperative Checkpoint technique	Cooperative Checkpoint with CSPs-SCPs
1	0.06	1.5	1.7	3.3
2	0.05	2.6	2.5	5.1
3	0.04	1.4	2.9	4.2
4	0.03	1.8	2.0	3.8

VI. REFERENCES

[1]. Adam J. Oliner, Larry Rudolph and Ramendra K. Sahoo. "Cooperative Checkpointing: A Robust Approach to Large-Scale System Reliability". In ACM June 28-30 (2006).

[2]. Yang Xiang and Zhongwen Li, Hong Chen. "Optimizing Adaptive Checkpoint Schemes for Grid Workflow Systems". In Fifth International Conference on Grid and Coperative Computing Workshop (GCCW'06) IEEE (2006).

[3]. Kalim Quereshi, Fiaz Gul Khan, Paul Manuel, Babar Nazir. "A hybrid fault tolerance in grid computing system".Published online: 19 January (2010).

[4]. Soonwook Hwang and Carl Kesselman. "A Flexible Framework For Fault Tolerance in the Grid".in Journal of Grid Computing (2003).

[5]. John W.Young. "A First Order Approximation to the Optimum Checkpoint Interval". Communication of ACM September (1974).

[6]. Fiaz Khul Khan, Kalim Qureshi and Babar Nazir. "Performa--nce evaluation of fault tolerance techniques in grid computing system." Published at Elsevier Journal (2010) .

[7]. E.N.Elnozahy,D.B.Johnson, and W.Zwaenepoel. "The performance of consistent checkpoint", In 11th Symposium on Reliable Dist Distributed System, Houston, TX,Oct. (1992).

[8]. Fault-tolerant System: http://en:Wikipedia.org/wiki/Faulttolerant system

[9]. A. Duda: The Effects of Checkpointing on Program Execution Time, Information Processing Letters, Vol16, pp.221-229, (1983)."