Evaluating the Reliability of Large-Scale Heterogeneous Grid Computing Systems in Dynamic Workload Environments

Peng Xiao* and Dongbo Liu

Department of Computer and Communication, Hunan Institute of Engineering
**xpeng4623@yahoo.com.cn; liudongbo1974@gmail.com*

Abstract

With the increasing scale of grid systems, reliability evaluation for both grid systems and applications become more and more challenging, especially when taking the heterogeneity and dynamical workload into consideration. In this work, a workload-aware reliability evaluation model is proposed, in which queuing system is applied to describe the dynamic workload and working of grid resources. To supporting deadline-sensitive applications, a new class of resource fault, namely Deadline-Miss fault, is introduced to evaluate the reliability of these applications. The validity of the proposed model and its approach to calculate deadline-sensitive job's reliability are presented theoretically. Extensive experiments are conducted to verify its performance, and the results show that the proposed model can significantly improve the accuracy of reliability evaluation in presence of dynamic workload. Also, scheduling algorithm based on this model can reduce mean response time and the deadline-miss rate for deadline-sensitive grid applications.

Keywords: *Grid Computing; Quality of Service; Resource Fault; Queue System; Probability Analysis*

1. Introduction

Grid systems have emerged as an important network-based computing platforms, distinguished from conventional distributed systems by its focus on large-scale resource sharing, innovative applications, and high-performance orientation [1,2]. To address issues such as large-scale resource sharing, wide-area communication, and multi-institutional collaboration, resource reliability plays a critical role when deploying and executing grid applications. From the perspective of grid applications, the application reliability can be defined as the probability of successful execution under a given scheduling scheme, which specifies the resource assignments of all the sub-tasks. From the perspective of grid platforms, the reliability of the system can be defined as the probability for all of the applications to be executed successfully under the current resource configurations.

On the other side, due to its open architecture, grid resources from different virtual organizations are inherently heterogeneous and dynamic, which lead to resource failures occur frequently and hard to be detected [3]. Furthermore, wide-area resource sharing over uncertain network increases the probability of resource failures since data transferring are often intensive in most of grid applications [4]. Therefore, reliability analysis and evaluation of grid systems becomes a challenging issue and attracts more and more attentions of researchers.

Existing studies show that the difficulties of reliability evaluation in grid environments are as following:

- Heterogeneity of resources results in various types of resource fault, which can hardly be taken into consideration in a single model [3, 19, 20].
- Failure model is greatly influenced by resource's workload, which often fluctuates dramatically and unpredictable in runtime [2, 3, 21, 22].
- Due to the large-scale and the complexity of grid systems, reliability are difficult to model, analyze, and evaluate [2, 3, 4, 5].

Among the existing grid reliability evaluation models, the *Tree-structured Grid Reliability Model* [2] (TGRM) proposed by Y.S. Dai *et al*. is considered as the most effective and efficient. However, the shortcoming of TGRM is that it does not take into account the dynamic workload when calculating reliability, which makes the TGRM only suitable for static reliability evaluation. With the development of grid computing technologies, more and more applications are requiring the grid system providing negotiable QoS mechanism, in which runtime resource reliability is one the most mentioned parameters. Therefore, static reliability evaluation is no longer sufficient in current grid systems. Motivated by this observation, in this paper we propose a dynamical reliability evaluation model, which is based on the TGRM and enhanced with the workload-aware mechanism.

The rest of this paper is organized as follows: Section 2 presents the related work. In Section 3, we analyze the shortcomings of TGRM, and present the workload-aware TGRM. In Section 4, extensive simulations are conducted to verify the performance of the proposed model. Finally, Section 5 concludes the paper with a brief discussion of future work

2. Related Work

In [6], *Distributed Program Reliability* (DPR) and *Distributed System Reliability* (DSR) are two metrics firstly introduced to the reliability evaluation of distributed systems by Raghavendra. Also, Raghavendra propose an algorithm based on graph-traversal to calculate DSR and DPR. However, the complexity of the algorithm is exponential, which means it is unsuitable to evaluation the reliability of large-scale systems. So, Chen, *et al.*, [7] improves Raghavendra's algorithm by using the graph-cutting approach. The common shortcoming of both two algorithms is that they all assume that the failure rates of resources and links are constant, which is unsuitable to the geographically distributed and dynamic grid environments.

So, in [5] Y.S. Dai, *et al.*, addresses the issue of reliability evaluation in grid environments. Dai's approach is based on *Minimal-Task-Spanning-Tree* (MTST) just like Raghavendra's, but he uses random failure model instead of assuming the failure rate being constant. In [2], Y.S. Dai, *et al.*, further their study and propose a *Tree-structured Grid Reliability Model* (TGRM) for analyzing the performance and the reliability of grid services in the presence of common failures caused by sharing communication links. TGRM is the first grid reliability model and has been proven to be effective and efficient.

However, TGRM dose not take into account the effects of workload on the reliability evaluation. In [3], Czajkowski has pointed out that "The reliability of grid system is greatly affected by many factors, and the dynamic changing workload on resources is one of the most significant one". Many other studies also prove this conclusion. For example, in [8] Kermarrec, *et al.*, gives the formal expression to describe the relationship between performance and workload in large-scale distributed systems. In

[9-10], Bucur and Epema conduct extensive experiments in grid testbed DAS-2 [11] to evaluate the performance of various co-allocation policies under different workload. Their experimental results show that workload-aware co-allocation policies are effective to reduce the probabilities of executing fault, since those policies are effective to avoid the negative effects brought by dynamic changing of workload on resources. In addition, the studies in [5, 7, 10, 22, 23] confirm that the conception of workload-aware is of significant importance not only for the performance of scheduler but also for the whole performance of grid systems.

In this paper, we apply queuing system [15] to describe the workload of grid resources. Recently, queuing system has been widely applied by many researchers to model the working of grid resources. For example, Sun Xian-He [12] has used queuing theory to predicate the availability of grid resources; Wu Ming [13] has applied M/G/1 queuing system to describe working model of single resource to analyze the effects of advance reservation on local job's scheduling; Berten [14] uses M/M/C queuing system to model the service of a single cluster to study the capability-based allocation policy in multi-cluster grid. Their studies indict that queuing system is capable of precisely describing the working and workload model of grid resources.

3. Reliability Model

3.1 TGRM Analysis

In TGRM, RMS (Resource Management System) is the root node of the tree model, which is responsible for task admission and scheduling. The resources are represented by leaf nodes, between which the edges represent the network links. TGRM assumes that the occurrence of resource and link faults follow exponential distribution. In order to simplify the evaluation of reliability, TGRM make three assumptions as following:

Assumption 1. The RMS is completely reliable, which means there will be no faults in RMS.

Assumption 2. The RMS receives tasks, assigns them to available resource, and integrates the received results for user's tasks. All the intermediate data are assembled on RMS, and then being sent back to users.

Assumption 3. Each available resource can only provide service for a single user's task.

Based on the above assumptions, TGRM gives the expressions to calculating the reliabilities of the resources and the links as following:

$$RF_j^k = \exp\left[-\varpi_k \cdot \left(\frac{c_j}{x_k} + \frac{a_j}{s_k}\right)\right] \tag{1}$$

$$LF_j^k = \exp\left[-\pi_k \cdot \left(\frac{c_j}{x_k} + \frac{a_j}{s_k}\right)\right] \tag{2}$$

where RF_j^k is the probability that sub-task j is successful completed on resource k, LF_j^k is the probability that there is no occurrence of link fault during the execution of sub-task j, ϖ_k and π_k are the mean fault rates of the resource k and the links, c_j is the size of sub-task j, a_j is the amount of data transmitted between resource k and RMS when sub-task j is running, x_k is the computing speed of resource k, s_k the minimal

bandwidth of all the network links between resource k and RMS. Therefore, the reliability of a task is that

$$\text{Reliability} = \max_{1 \le j \le m}[\max_{k \in w_j}(RF_j^k \cdot LF_j^k)] \quad (3)$$

where m is the number of sub-tasks, and w_j is the set of selected resources for executing the task.

3.2 Workload-aware TGRM

As mentioned in Section 2, queuing system has been proven to be able to precisely describe the working and workload model of grid resources. So, we also use queuing system to construct the working and workload model of grid resources. By this way, we incorporate workload-aware mechanism into the TGRM.

Among the assumptions of TGRM, it is obviously that the third assumption is too conservative and unsuitable. For instance, multi-cluster systems are a typical class of grid system that generally used in scientific fields. In multi-cluster systems, a high-performance cluster exposes its service interface to the out-side users in form of *single-image*. So, a high-performance cluster should be considered as a single resource with paralleling computing capability. This is also true for those *Massive Parallel Processor* (MPP) systems. To be compatible with TGRM, we use M/M/1 queuing system to model those resources that can only provide service for a single task at a time, and M/M/C queuing system for those resources with parallel capability.

Meanwhile, as shown in (1), (2) and (3), TGRM ignores the waiting time of jobs when calculating the reliability. It may be feasible when resources are abundant or workload is in low-level, but not suitable to high-level workload which is common in grid environments. Another problem of TGRM is that it does not take into consideration the deadline constraint, which is often required by real-time tasks. For those real-time tasks, if it cannot be completed before the deadline the results of the tasks are of less or no value for users. So, deadline-miss should be considered a sort of resource failure.

To overcoming the above shortcomings of TGRM, we relax the third assumption of TGRM to provide supports for parallel serving resources. Also, we introduce a new type of fault, called *Deadline-Miss Fault* (DMF), into TGRM to support the reliability evaluation of real-time tasks. Therefore, equation (1), (2) and (3) listed previously is modified as following:

$$RF_j^k = \exp\left[-\varpi_k \cdot (\frac{c_j}{x_k} + \frac{a_j}{s_k} + d_j)\right] \quad (4)$$

$$LF_j^k = \exp\left[-\pi_k \cdot (\frac{c_j}{x_k} + \frac{a_j}{s_k} + d_j)\right] \quad (5)$$

$$\text{Reliability} = \max_{1 \le j \le m}[\max_{k \in w_j}(RF_j^k \cdot LF_j^k \cdot DMF_j^k)] \quad (6)$$

where d_j is the deadline requirement of sub-task j, DMF_j^k is the probability that there is no deadline-miss when sub-task j is scheduled on resource k.

As shown in (6), the key issue of workload-aware TGRM is the approach to calculate the DMF_j^k, which is greatly affected by the workload on resources. So, in the next section, we will focus on the calculation of DMF_j^k by using queuing system.

3.3 Calculation of Deadline Miss Failure

As mentioned above, workload-aware TGRM uses M/M/1 queuing system to model those resources that can only provide service for a single task at a time, and M/M/C queuing system for those resources with parallel capability (i.e. cluster or MPP). Therefore, the approaches to calculate the DMF_j^k for these two types of resources are different.

Theorem 1. If resource R_k is modeled as M/M/ C_k queuing system, then the probability that there is no deadline-miss when sub-task j is scheduled on R_k is

$$DMF_j^k = \sum_{n=0}^{C_k} \delta \cdot \frac{(\rho_k \cdot C_k)^n}{n!} + \sum_{n=1}^{C_k \cdot \mu_k \cdot d_j - 1} \delta \cdot \frac{\rho_i^{n+C_k} \cdot C_k^{C_k}}{C_k!}$$

where $\delta = \left[\sum_{n=1}^{C_k} \frac{(\rho_k \cdot C_k)^n}{n!} + \frac{(\rho_k \cdot C_k)^{C_k}}{C_k} \frac{1}{1-\rho_k} \right]^{-1}$, $\rho_k = \lambda_k / (C_k \cdot \mu_k)$, d_j is the deadline of sub-task j, λ_k is the mean interval of sub-tasks on R_k, μ_k the mean service time of R_k, and C_k is the paralleling processing capability of R_k.

Proof. Let ψ be the random variable representing the number of waiting tasks in R_k. According to queuing theory [15], the probability that there are m waiting jobs in R_k is

$$\Pr\{\psi = m\} = \begin{cases} \delta \cdot \frac{\rho_k^{m+C_k} \cdot C_k^{C_k}}{C_k!}, & m > 0 \\ \sum_{n=0}^{C_k} \delta \cdot \frac{(\rho_k \cdot C_k)^n}{n!}, & m = 0 \end{cases} \tag{7}$$

$$\text{where } \delta = \left[\sum_{n=1}^{c_i} \frac{(\rho_i \cdot c_i)^n}{n!} + \frac{(\rho_i \cdot c_i)^{c_i}}{c_i} \frac{1}{1-\rho_i} \right]^{-1} \tag{8}$$

For M/M/ C_k queuing system, the service rate is $C_k \cdot \mu_k$, which means the system can complete $C_k \cdot \mu_k$ tasks in a unit time. So, the amount of tasks that R_k can complete in period d_j is $C_k \cdot \mu_k \cdot d_j$. Therefore, the probability that R_k can guarantee a tasks's deadline d_j is equal to the probability that the waiting tasks in R_k is not more than $C_k \cdot \mu_k \cdot d_j - 1$. That is

$$DMF_j^k = \Pr\{\psi \le c_i \cdot \mu_i \cdot d_j - 1\} \tag{9}$$

Combing formulas (7)(8)(9) and the above analysis, we can get that

$$\begin{aligned} DMF_j^k &= \Pr\{\psi \le C_k \cdot \mu_k \cdot d_j - 1\} = \sum_{m=0}^{C_k \cdot \mu_k \cdot d_j - 1} \Pr\{\psi = m\} \\ &= \sum_{m=0}^{C_k} \delta \cdot \frac{(\rho_k \cdot C_k)^m}{m!} + \sum_{m=1}^{C_k \cdot \mu_k \cdot d_j - 1} \delta \cdot \frac{\rho_k^{m+C_k} \cdot C_k^{C_k}}{C_k!} \end{aligned} \tag{10}$$

■

Theorem 2. If resource R_k is modeled as M/M/1 queueing system, then the probability that there is no deadline-miss when task j is scheduled on R_k is

$$DMF_j^k = \sum_{m=0}^{\mu_k \cdot d_j - 1} (1-\rho_k) \cdot \rho_k^m$$

where $\rho_k = \lambda_k / \mu_k$, d_j is the deadline of task j, λ_k is the mean interval of sub-tasks on R_k, μ_k the mean service time of R_k.

Proof. Let ψ be the random variable representing the number of waiting tasks in R_k. According to queuing theory [15], the probability that there are m waiting jobs in R_k is

$$\Pr\{\psi = m\} = (1-\rho_k) \cdot \rho_k^m, \quad m \geq 0 \tag{11}$$

For M/M/C_k queuing system, the service rate is μ_k, which means the system can complete μ_k tasks in a unit time. Similar to the analysis in Theorem 1, we can obtain that

$$\begin{aligned} DMF_j^k &= \Pr\{\psi \leq \mu_i \cdot d_j - 1\} = \sum_{m=0}^{\mu_k \cdot d_j - 1} \Pr\{\psi = m\} \\ &= \sum_{m=0}^{\mu_k \cdot d_j - 1} (1-\rho_k) \cdot \rho_k^m \end{aligned} \tag{12}$$

Based on Theorem 1 and Theorem 2, we can obtain the workload-aware TGRM by substitute the (10) or (12) into in (4), (5) and (6). As to the generation of *Minimum Spanning Tree*, we can directly use the approach that proposed in [2].

4. Experiments and Performance Comparison

4.1. Experimental settings

In experiments, we use GridSim [16], a distributed resource management and scheduling simulator, to construct a multi-cluster grid model. The grid model follows the TGRM's tree-structured, and its topology and setting of individual resources are deprived from the grid test-bed DAS-2 [11]. As shown in Figure 1, the grid model consists of twelve *Computational Elements* (CE 1 ~ CE 12), each representing a high-performance cluster. The clusters are grouped into five groups by their geographical positions. Within each group, the clusters are connected by LAN (Link 1 ~ Link 12). Then, they are connected by WAN (Link 13 ~ Link 17) between groups. The failure of links follows exponential distribution with various parameters. According to the statistics in [17, 19, 21], the failures of LAN link are different from the WAN's. So, we set that the LAN's fault rate limited in $[0.02, 0.04]$, and the WAN's fault rate limited in $[0.04, 0.12]$.

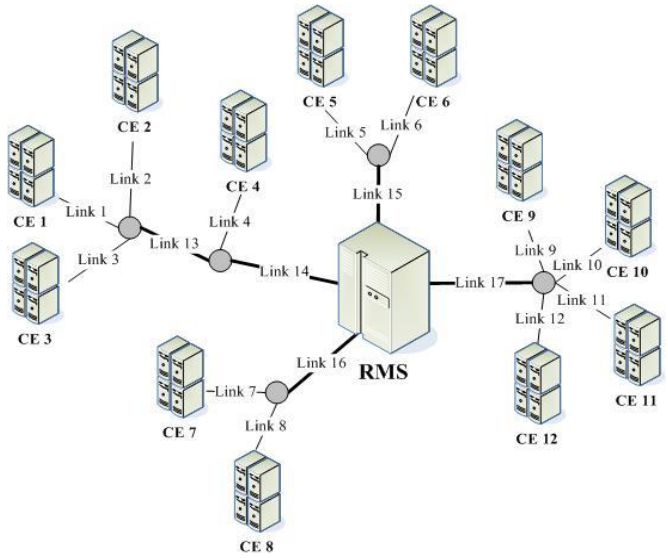

Figure 1. Grid Model in Simulations

The detailed configurations of each CE are presented in Table 1, which is also deprived from grid test-bed DAS-2.

Table 1. Setting of Simulative Grid Model

ID	Processor number	MIPS / processor
CE 1	64	377
CE 2	128	410
CE 3	256	380
CE 4	128	285
CE 5	128	285
CE_6	64	515
CE 7	128	215
CE 8	64	285
CE 9	256	380
CE 10	512	215
CE 11	256	333
CE 12	128	410

In simulations, the basic workload (tasks stream) is generated by using Lublin-Feitelson model [18], which is derived from the logs of real supercomputers. It consists of 10000 tasks, each is characterized by its arrival time T_a, resource demands R, execution time T_e. However, this basic workload can not meet the requirements of our simulation, because it lacks of deadline and time of data transmission. So, we modify the basic workload by append each task with deadline d and data transmission time T_{data}, which is obtained as following.

$$d = T_a + g \cdot T_e \tag{13}$$

$$T_{data} = h \cdot T_e \tag{14}$$

where g is a random variable that uniformly distributed in $[5.5, 10.5]$, and h is a uniformly distributed in $[0.5,1.5]$. Therefore, each task in the modified workload is characterized by five-tuple: $< T_a, T_e, T_{data}, R, d >$.

5.2 Comparison of Accuracy

Firstly, we mainly focus on the accuracy of *Workload-aware TGRM* (WA_TGRM) comparing with TGRM. Also, we investigate WA_TGRM's performance in term of *Mean Response Time* (MRT). As mentioned in Section 3.2, the differences between WA_TGRM and TGRM are the reliability calculating formulas as shown in (4), (5) and (6). So, the algorithms of reliability calculation and task scheduling in WA_TGRM are as same as that in TGRM, which are specified in the appendix of [2]. In this experiment, we first calculate the reliabilities of tasks by using TGRM and WA_TGRM respectively, then schedule the tasks onto resources. As to each task, there are three type of executing results: success, abort (resource fault or link fault), deadline-miss fault. The experimental results are shown in Figure 2, in which we present the reliabilities calculated by TGRM and WA_TGRM. In Figure 2, the practical reliability is calculated as ratio of number of successful tasks to the total tasks.

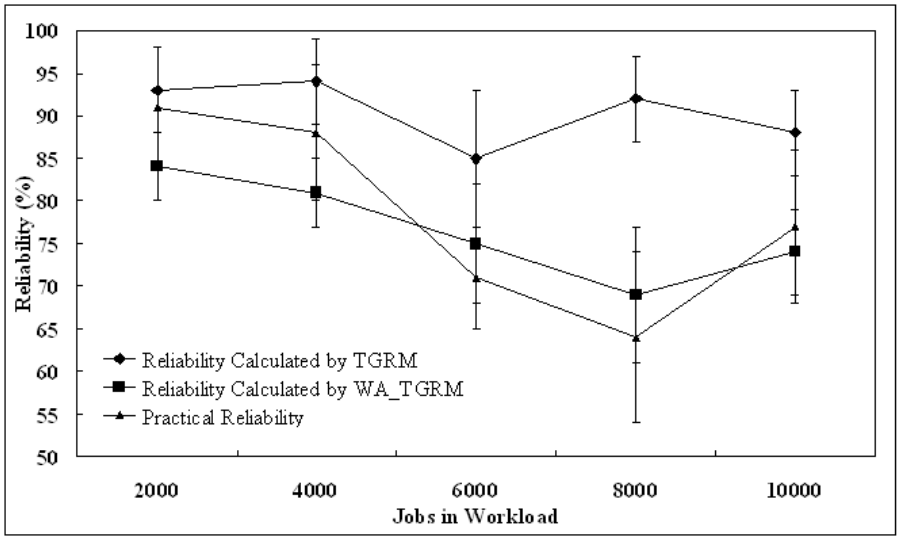

Figure 2. Accuracy of TGRM and WA_TGRM

We define the ratio of theoretic reliability to real reliability as the accuracy of TGRM or WA_TGRM. As shown in Figure 2, the accuracy of TGRM is about 97% for the first 2000 tasks, and 93% for the second 2000 tasks. As to WA_TGRM, they are about 92% and 91% correspondingly. However, the high accuracy of TGRM does not be maintained with the continuing of the simulation. In fact, TGRM's accuracy decreases dramatically for the third and the fourth 2000 tasks, which are only about 84% and 69%. On the contrary, WA_TGRM's accuracy is always kept above 90%. In order to examine such results, we record all the causes of unsuccessful execution and their percentages,

which are shown in Figure 3. As we can see, the percentages of resource fault and link fault are very high for the first 4000 tasks. However, percentage of deadline-miss fault keeps increasing with the continuing of the experiment, which becomes the highest (about 34%) when executing the 6000^{th} - 8000^{th} tasks.

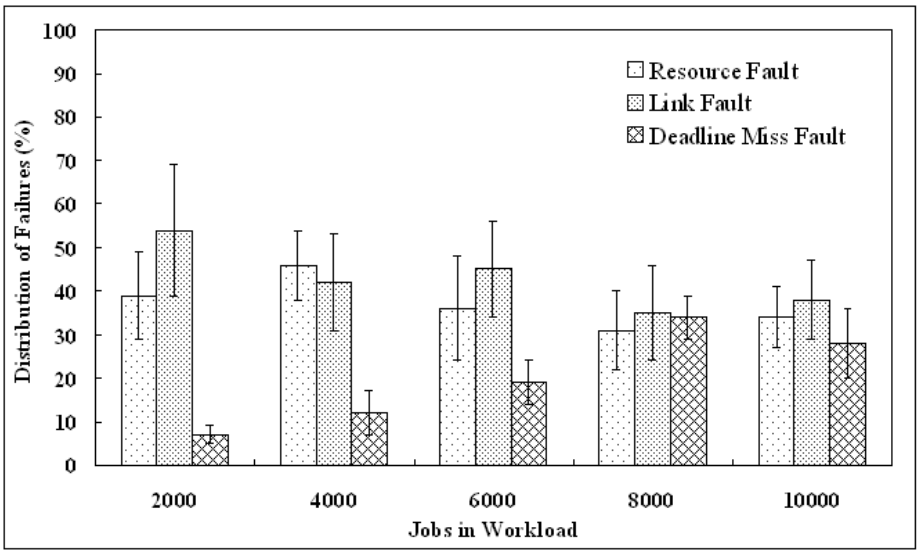

Figure 3. Distribution of Execution Failure Causes

The detailed observation of the simulation shows that most of the tasks are scheduled onto low-load resources at the beginning. With the increasing of new arriving tasks, the workload of resources gradually becomes heavier and heavier. As a result, the waiting time of tasks increased consequently, which causes higher deadline-miss fault. As TGRM does not take the waiting time into account, nor dose it consider deadline-miss fault as a type of fault, its accuracy inevitably decreases in presence of high deadline-miss fault just as shown in Figure 2. On the contrary, WA_TGRM overcomes those TGRM's shortcomings. As a result, TGRM's accuracy can always be kept in high-level in various cases, which indicates that WA_TGRM is more adaptive than TGRM in practice grid systems.

5.3 Comparison of Scheduling Performance

Finally, we want to investigate the scheduling performance of TGRM and WA_TGRM in term of *Mean Response Time* (MRT). Although the scheduling algorithm of both TGRM and WA_TGRM are the same, their scheduling schemes (*Minimal-Task-Spanning-Tree, MTST)* for individual tasks are different, because their reliability evaluation formulas are different. The experimental results are show in Figure 4.

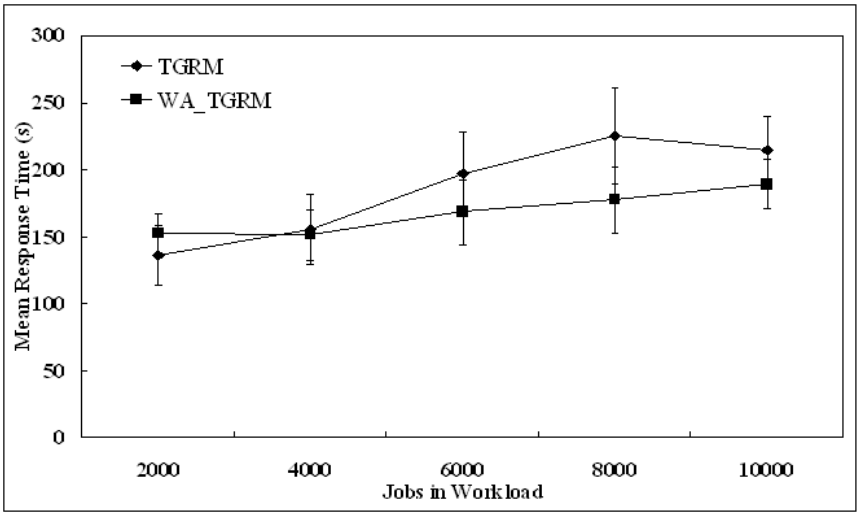

Figure 4. Mean Response Time of Jobs

As shown in Figure 4, The results show that the mean response time of WA_TGRM is about 20% lower than TGRM since after executing the first 4000 tasks. The detailed observation of the simulation shows that the MTSTs generated by WA_TGRM often obtains many low-load resources. As shown in (6), DMF_j^k is an important factor when calculating reliability in WA_TGRM. The effects of DMF_j^k on the results of reliability evaluation are influenced by the resource's workload, which can be clearly seen from the conclusions in Theorem 1 and Theorem 2. More specifically, when workload is low the effects of DMF_j^k is also lower, and vice versa. So, when the system is in face of the heaviest workload (during the 6000^{th} – 8000^{th} tasks) WA_TGRM is more effective to be aware such heavy workload and generate better scheduling scheme. The experimental results further confirm the Bucur's conclusion that "workload-aware scheduling algorithm is more effective to reduce mean response time and load-balance" in [9].

5. Conclusion

In this paper, we propose a novel Workload-aware Reliability evaluation model, which is deprived from TGRM. In the proposed model, queuing system is applied to describe the workload grid resources. In order to evaluate the reliability of jobs with constrain to deadline, a new type of resource fault (Deadline-Miss fault) is introduced to the proposed model. In the simulations, we compare the performance of WA_TGRM with TGRM in terms of accuracy. The experimental results show that WA_TGRM can provide more accurate reliability evaluation when grid system in presence of high-level workload. This indicates that WA_TGRM is more adaptive than TGRM in practice grid systems. Also, simulations are conducted to examine the performance of reliability-based job scheduling. Experimental results show that WA_TGRM is able to reduce the jobs' mean response time about 15% because its ability to generate workload-aware scheduling schemes. At present, our WA_TGRM concentrates on the reliability

evaluation when allocating resources. In practice grid systems, advance reservation is an effective technique that generally used to improve the reliability of co-allocation from multi-institutions. So, our future work is to take advance reservation into account when calculating resources' reliability. More specific, we plan to introduce another type of fault, namely Reservation-miss Fault, into current WA_TGRM design.

Acknowledgements

This work is supported by the Provincial Science & Technology plan project of Hunan (No.2012GK3075).

References

[1] M. Rahman, R. Ranjan, R. Buyya and B. Benatallah, "A Taxonomy and Survey on Autonomic Management of Applications in Grid Computing Environments", Concurrency and Computation-Practice & Experience, vol. 16, no. 23, **(2011)**, pp. 1990-2019.
[2] Y. S. Dai and G. Levitin, "Reliability and Performance of Tree-Structured Grid Services", IEEE Transactions on Reliability, vol. 2, no. 55, **(2006)** pp. 337-349.
[3] V. Thierion, P. -A. Ayral, G. Jacob, S. L. Sophie and P. Olivier, "Grid Technology Reliability for Flash Flood Forecasting: End-user Assessment", Journal of Grid Computing, vol. 3, no. 9, **(2011)** pp. 405-422.
[4] X. Tang, K. Li, M. Qiu and E. H. M Sha, "A Hierarchical Reliability-driven Scheduling Algorithm in Grid Systems", Journal of Parallel and Distributed Computing, vol. 4, no. 72, **(2012)**, pp. 525-535.
[5] Y. S. Dai, M. Xie and K. L. Poh, "Reliability Analysis of Grid Computing Systems", Proceedings of Pacific Rim International Symposium on Dependable Computing, **(2002)**, pp. 97-104.
[6] C. S. Raghavendra, V. K. P. Kumar and S. Hariri, "Reliability Analysis in Distributed Systems", IEEE Transactions on Computers, vol. 3, no. 37, **(1988)**, pp. 352-358.
[7] D. J. Chen and T. H. Huang, "Reliability Analysis of Distributed Systems Based on a Fast Reliability Algorithm", IEEE Transactions on Parallel and Distributed Systems, vol. 2, no. 3, **(1992)**, pp. 139-154.
[8] A. M. Kermarrec, L. Massoulie and A. J. Ganesh, "Probabilistic Reliable Dissemination in Large-Scale Systems", IEEE Transactions on Parallel and Distributed System, vol. 3, no. 14, **(2003)**, pp. 248-258.
[9] A. I. D. Bucur and D. H. J. Epema, "Scheduling Policies for Processor Coallocation in Multicluster System", IEEE Transactions on Parallel and Distributed Systems, vol. 7, no. 18, **(2007)**, pp. 958-962.
[10] A. I. D. Bucur and D. H. J. Epema, "The Performance of Processor Co-Allocation in Multicluster Systems", Proceedings of IEEE International Symposium on Cluster Computing and the Grid, **(2003)**, pp. 302-309.
[11] H. Bal, R. R. Bhoedjang and R. Hofman, "The Distributed ASCI Supercomputer Project", ACM Operating Systems Review, vol. 4, no. 34, **(2000)**, pp. 76-96.
[12] X. H. Sun and M. Wu, "Grid Harvest Service: A System for Long-Term, Application-Level Task Scheduling", Proceedings of International Symposium on Parallel and Distributed Processing, **(2003)**, pp. 1-9.
[13] M. Wu, X. H. Sun and Y. Chen, "QoS Oriented Resource Reservation in Shared Environments", Proceedings of International Symposium on Cluster Computing and the Grid, **(2006)**, pp. 601-608.
[14] V. Berten, J. Goossens and E. Jeannot, "On the Distribution of Sequential Jobs in Random Brokering for Heterogeneous Computational Grids", IEEE Transactions on Parallel and Distributed Systems, vol. 2, no. 17, **(2006)**, pp. 113-124.
[15] D. Gross and C. M. Harris, "Fundamentals of Queuing Theory", USA: John Wiley and Sons, **(1998)**.
[16] A. Sulistio, U. Cibej, S. Venugopal, B. Robic and R. Buyya, "A Toolkit for Modelling and Simulating Data Grids: an Extension to GridSim", Concurrency and Computation-Practice & Experience, vol. 13, no. 20, **(2008)**, pp. 1591-1609.
[17] H. Li, "Realistic Workload Modeling and Its Performance Impacts in Large-Scale eScience Grids", IEEE Transactions on Parallel and Distributed Systems, vol. 4, no. 21, **(2010)**, pp. 480-493.
[18] U. Lublin and D. G. Feitelson, "The Workload on Parallel Supercomputers: Modeling the Characteristics of Rigid Jobs", Journal of Parallel and Distributed Computing, vol. 11, no. 63, **(2003)**, pp. 1105-1122.
[19] Y. Y. Chen and T. J. Wu, "A Decentralized Coordination Method for Optimal Load Redistribution in Heterogeneous Service Grids", Concurrency and Computation-Practice & Experience, vol. 6, no. 23, **(2011)**, pp. 633-645.
[20] A. Iosup and D. Epema, "Grid Computing Workloads", IEEE Internet Computing, vol. 2, no. 15, **(2011)**, pp. 19-26.

[21] M. Waldburger, M. Göhner, H. Reiser, G. D. Rodosek and B. Stiller, "Evaluation of an Accounting Model for Dynamic Virtual Organizations", Journal of Grid Computing, vol. 2, no. 7, **(2009)**, pp. 181–204.
[22] H. Cao, H. Jin and X. Wu, "DAGMap: Efficient and Dependable Scheduling of DAG Workflow Job in Grid", Journal of Supercomputing, vol. 2, no. 51, **(2010)**, pp. 201-223.
[23] W. C. Yeh and S. C. Wei, "Economic-based Resource Allocation for Reliable Grid-computing Service based on Grid Bank", Future Generation Computer Systems, vol. 7, no. 28, **(2012)**, pp. 989-1002.

Authors

Peng Xiao received the Ph.D degree in computer science from the Central South University in 2009. Currently, he is an associate professor in the Hunan Institute of Engineering. Also, he is the advanced network engineer in HP High-performance Network Centre in Hunan. His research interests include grid computing, distributed resource management. He is a member of ACM, IEEE, and IEEE Computer Society.

Dongbo Liu received his master degree in Hunan University in 2004. Now he works in Hunan Institute of Engineering and is a Ph.D candidate in Hunan University. His research interests include distributed intelligence, multi-agent systems, high-performance application. He is now a student member of CCF in China, and worked as Senior Engineer in HP High-performance Lab.