DGSS: A Dependability Guided Job Scheduling System for Grid Environment^{*}

Yongcai Tao, Hai Jin, and Xuanhua Shi

Services Computing Technology and System Lab

Cluster and Grid Computing Lab

School of Computer Science and Technology

Huazhong University of Science and Technology, Wuhan, 430074, China

`hjin@hust.edu.cn`

Abstract. Due to the diverse failures and error conditions in grid environments, node unavailability is increasingly becoming severe and poses great challenges to reliable job scheduling in grid environment. Current job management systems mainly exploit fault recovery mechanism to guarantee the completion of jobs, but sacrificing system efficiency. To address the challenges, in this paper, a node TTF (*Time To Failure*) prediction model and job completion prediction model are designed. Based on these models, the paper proposes a dependability guided job scheduling system, called DGSS, which provides failure avoidance job scheduling. The experimental results validate the improvement in the dependability of job execution and system resources utilization.

1 Introduction

In grid environment, resources can enter and depart without any prior notice. In addition, failures can occur at any moment by various reasons, e.g. the change of resource local policy and the breakdown of resources and network fabric [1]. Hence, jobs fail frequently and QoS can not be guaranteed. Node unavailability poses great challenges to grid job scheduling. To address these issues, existing job management systems mostly resort to fault recovery mechanism, such as checkpoint, retry and replication. Although relieving the challenges to some extent, this mechanism sacrifices system resources. For example, checkpoint usually requires extra disk space and network bandwidth to record the job running information, e.g. intermediate results and data for continuing job without starting from scratch. Replication policy runs the job at multiple available resources. The fault recovery mechanism belongs to compensating methodology and can not prevent the job failures in advance. To prevent the job failures proactively, the accurate information of temporal and spatial distribution of grid node availability in the future should be predicted. Thus, jobs can be scheduled onto nodes with long uptime instead of upcoming failure nodes.

So far, researches mainly give attention to model and predict the characteristics of cluster nodes' availability. As the scale is expanding rapidly, grid integrates not only

^{*} This paper is supported by Nation Science Foundation of China under grant 90412010 and 60603058.

cluster resources, but more wide-area personal resources which are mostly non-dedicated. Therefore, the characteristics of node unavailability in grid environment distinguish greatly from that in cluster environment. In the paper, the terms node unavailability and node failure are used interchangeably to refer to the event that nodes are inaccessible for nodes' leaving, hardware and software breakdown, and network connection malfunction, etc.

In this paper, a node TTF (*Time To Failure*) prediction model and job completion prediction model are designed. Based on these models, the paper proposes a dependability guided job scheduling system, called DGSS. Rational of DGSS is that if the completion time of job is less than the TTF of node, the job can expect to finish successfully; otherwise, the job is likely to fail and has to be restarted on the same node or somewhere else. Experiments are conducted and demonstrate that DGSS can enhance the dependability of job execution and resources utilization.

The rest of the paper is organized as follows. Section 2 reviews the related work. We introduce DGSS in section 3. The experimental evaluation is presented in section 4. Finally, we conclude this paper in section 5.

2 Related Work

There are a vast amount of related works on grid job scheduling. Existing strategies of grid job scheduling can be categorized into performance-driven, market-driven and trust-driven [2].

Papers [3, 4] adopt performance-driven strategy to submit jobs onto resources and achieve optimal performance for users and system (e.g. minimum job execution time and high throughput). The work in [5] utilizes market-driven strategy during job assignment. Recently, some research works have strived to address the scheduling problems with reliability optimization. In [6], trust-driven scheduling strategy is exploited to map jobs onto appropriate resources according to their trust levels. This strategy avoids selecting malicious nodes and non-reputable resources. However, it does not take the job completion prediction time into consideration. In [7], reliable scheduling adopts "Reliability Cost" as the indicator of how reliable a given system is when a group of tasks are assigned to it, but the failure rate of grid nodes and the communication links between them adopted in the reliability cost model is set experimentally.

To support reliable job scheduling, much attention is paid to model the characteristics of node failure to measure node reliability, which mainly focuses on homogeneous computing environments. Papers [8, 9] investigate the failure logs of cluster systems and conclude that the time between reboots of nodes is best modeled by a Weibull distribution with shape parameters of less than 1, implying that a node becomes more reliable the longer it has been operated. Based on the observed characteristic of node failure, researchers also design resources allocation strategy to improve service availability.

Due to the diverse failures and error conditions in grid environment, grid node failures are more stochastic than clusters. Hence, current models can not adapt well to grid environment.

3 Dependability Guided Job Scheduling System - DGSS

In this section, two critical components of DGSS, the node TTF prediction model and job completion time prediction model, are introduced. Then, the corresponding dependable scheduling algorithm is described in detail.

3.1 MC Based Node TTF Prediction Model

In many application fields, Markov model can be extended to model stochastic processes. *Discrete-time Markov chain* (DTMC) is defined at the basis of a set of finite state $M(S_1, S_2, \dots, S_m)$ and $M \times M$ state transition matrix P . In matrix P , P_{ij} denotes the transition probability from state S_i to state S_j .

DTMC is mainly used to predict the state occurrence probability in the future. Suppose at time t_k , system state is S_i ($1 \leq i \leq M$) and the distribution of state S_i is $P_k(S_i) = e_i$, where e_i is $1 \times M$ row vector, the value at location i is 1, and others are 0. Thus, we can predict the distribution of S_i at next time: $P_{k+1}(S_i) = P_k(S_i)P = e_i P$. At time $k+2$, the distribution of S_i is: $P_{k+2}(S_i) = P_{k+1}(S_i)P = e_i P^2$. At time $k+n$, the distribution of S_i is: $P_{k+n}(S_i) = P_{k+n-1}(S_i)P = e_i P^n$. So, with DTMC, we can obtain the distribution of state S_i at this time and next time, and therefore we can get the occurrence probability of each state at each time.

In grid environment, TTF of nodes at each time is stochastic. Markov model can be used to model the node TTF stochastic process. In MC based node TTF prediction model, TTF can be seen as system state. In the Markov model described above, the M and P are invariable. The dynamic nature of grid requires large storage space for M and P , which make the model complex and unpractical. In order to address this issue, we present an adaptive MC based node TTF prediction model which can dynamically amend M and P .

When a node becomes inaccessible, new TTF is produced (called TTF_{new}). Then the M would be traversed. If there exists S_i whose absolute difference value and TTF_{new} is less than the specified value, S_i and TTF_{new} would be merged. New S_i is the average of S_i and TTF_{new} and the number of state transition would be added 1. Conversely if there does not exist this state, new state S_{m+1} would be created. At the same time, P would change correspondingly according to the following formula:

$$P_{ij} = n_{ij} / \sum_K n_{ik} \quad (1)$$

where n_{ij} is the transition number from state i to state j with K failures. $\sum_K n_{ik}$ is the all state transition number of K failures.

3.2 Job Completion Time Prediction Model

The job completion time prediction model is also a critical component and has great impact on the efficiency of scheduling system. To predict the completion time of jobs, we adopt the following model similar to the one proposed in [10].

$$CT_{ij} = OPT \times (WT_{ij} + ReT_i + PT_{ij} + WrT_i) \quad (2)$$

Some symbols used in the model are listed in Table 1.

Table 1. Parameters Definition

Symbol	Meaning
CT_{ij}	Completion time of job i at node j
BW_{ij}	Bandwidth between node i and j
$Parent(S_i)$	Parent set of node i
$Child(S_i)$	Child set of node i
$DVol(S_i)$	Data volume produced by job i
OPT	Correction parameter
Q_i	Number of jobs in waiting queue of node i
Q	Number of jobs submitted by user
P_i	Computation capability of node i
ReT_i	Time to read the input data for executing job i
WrT_i	Time to write the output data for executing job i
PT_{ij}	Processing time of job i at node j
WT_{ij}	Waiting time of job i at node j
S_j	The node on which the job is executed
TTF_i	Time to failure of node i
STT_i	Startup time of node i
$CURT$	Current time of system
$RTTF_i$	Remaining time to failure of node i
QSR	Array used to store qualified set of resources
$QDIFF$	Array used to keep the difference between job CT and qualified resource's $RTTF$
$UQSR$	Array used to store unqualified set of resources
$UQDIFF$	Array used to keep the difference between job CT and unqualified resource's $RTTF$
$SelectedR$	Array used to store selected resources

The model is composed of four parts. First, when the job is assigned to a specific resource, it must wait until all jobs in waiting queue are finished. WT_{ij} denotes the waiting time of job i at node j . Second, the node fetches the data needed during the job execution. ReT_i represents the time to read the input data. Third, PT_{ij} stands for the processing time of job i at node j . Finally, the node outputs the results of job execution and the relative data. WrT_i denotes the time to output the data. Therefore, the completion time of job can be obtained as following:

$$CT_{ij} = OPT \times \left(\sum_{p \in \text{Parent}(S_j)} \frac{DVol(p)}{BW_{pj}} + \sum_{p \in \text{Child}(S_j)} \frac{DVol(p)}{BW_{pj}} + \frac{Q_j}{P_j} + \frac{Q}{P_j} \right) \quad (3)$$

where OPT is a parameter which can be adjusted to correct the predicted result, BW_{ij} can be obtained through the *Network Weather Service* (NWS).

3.3 Dependable Scheduling Algorithm

The 4-tuple $\langle R, M, J, Q \rangle$ is defined as the job scheduling including the resource performances metric and job QoS requirement description. Given a grid system with m resources: $R = \{R_1, R_2, \dots, R_M\}$. $M = \{M_1, M_2, \dots, M_M\}$ is the set of performance metric of corresponding resources such as reliability, charges. $J = \{J_1, J_2, \dots, J_N\}$ is used to denote a set of job and $Q = \{Q_1, Q_2, \dots, Q_N\}$ is the set of QoS requirements of each job, which can be time constraint, cost constraint and so on. The dependability guided scheduling algorithm is described in Algorithm 1.

Algorithm 1: Dependability Guided Scheduling

Input: a set of job requests

Output: a set of selected resources

```

while  $\exists$  job  $i$  not assigned do
   $m = 1$ ;  $n = 1$ ;
  foreach available resource  $j$  do
    calculate  $CT_{ij}$ ;  $RTTF_j = (TTF_j - (CURT - STT_j))$ ;
    if  $CT_{ij} < RTTF_j$  then
       $QSR[i][m++] = j$ ;  $QDIFF[i][m++] = RTTF_i - CT_{ij}$ ;
    else
       $UQSR[i][n++] = j$ ;  $UQDIFF[i][m++] = CT_{ij} - RTTF_i$ ;
    endif
  endforeach
  if  $QSR$  is not null then
     $SelectedR[i] = f_1(QSR[i][m++], QDIFF[i][m++])$ ;
  else
     $SelectedR[i] = f_1(UQSR[i][n++], UQDIFF[i][m++])$ 
    or report fault;
  endif
endwhile

```

In the above algorithm, the function f_1 is defined to adopt different heuristics. For example, for each job, it can select the node whose difference between CT and $RTTF$ is largest to maximize the dependability of job execution, or the node whose difference is smallest to improve the overall system performance while guarantee the job execution. The function f_1 can also exploit users' QoS requirements as its secondary heuristics criteria such as the minimum completion time or the lowest cost.

For each job, at the beginning, the algorithm predicts the job completion time on each node and calculates the remained time to the failure of nodes. Then, by comparing the completion time of job with the $RTTF$ of the node, system obtains the qualified resources and unqualified resources from the available resources pool. If QSR is not empty, the optimal resource will be chosen according to specific heuristic adopted

by function f_j . If QSR is empty, the objective resource will be chosen from the UQSR or relative fault is reported. In this case, job may fail because it takes the risk of being executed on a node with its $RTTF$ lower than the completion time of job. This process continues until all jobs are assigned appropriate resources.

4 Performance Evaluation

We compare DGSS with two common scheduling algorithms in terms of the amount of assigned jobs of nodes, success rate of jobs and system throughput. The two scheduling algorithms are as follows:

- RDS: Random Scheduling. In this scheme, jobs are submitted to specific resources which are randomly picked from the available resources.
- HCS: Heuristic Cost Scheduling. In this policy, jobs are assigned to the resources according to the ranking value of resources' cost while satisfying user's QoS requirements.

We perform the experiments in a real grid environment consisting of five grid nodes: three nodes at *Cluster and Grid Computing Lab* (CGCL), and two nodes at *National Hydro Electric Energy Simulation Laboratory* (NHEESL). The grid platforms are deployed with ChinaGrid Support Platform grid middleware (CGSP) [11]. We collect the running log of these five grid platforms for three months and use them to create MC based node availability prediction model to predict nodes' availability in the future. We test three types of applications. One is gene sequence matching application, FASTA, which deals with large size of database. The second is image processing application, which needs to process lots of images. The third is video conversion, which is both computing-intensive and data-intensive. We run 300 FASTA jobs, 300 image processing jobs and 300 video conversion jobs at different time interval.

Figure 1A, 1B, and 1C show the amount of three kinds of jobs assigned to different nodes. We can conclude that due to high uptime, nodes 2 and 3 are assigned more jobs than other nodes in DGSS. Meanwhile, nodes 4 and 5 are assigned more jobs because of lower charge in HCS mode, but the job failure ratio may be high for nodes' poor reliability. In addition, as FASTA requires longer execution time, the difference in amount of assigned jobs in Figure 1A is larger than that of Figure 1B and Figure 1C.

Figure 1D shows the change of system throughput of three different scheduling algorithms while the number of jobs increases. DGSS considers the node availability and job completion time while scheduling, so node loads are balanced and job execution is likely to success. Correspondingly the system throughput is high. Finally, we summarize the experimental data. The success rate of RDS is 51.65%, HCS is 68.57% and DGSS is 79.47%. Obviously, DGSS improves the success rate of job execution greatly. However, since there are still errors in the prediction models of job completion prediction and TTF of nodes, job failures may still occur. It can be seen that the accuracy of prediction models plays an important role on the efficiency of DGSS.

Fig. 1. (A) Amount of assigned FASTA jobs of nodes; (B) Amount of assigned image processing jobs of nodes; (C) Amount of assigned video conversion jobs of nodes; (D) Throughput comparison between three scheduling algorithms

5 Conclusions

In this paper, a dependability guided job scheduling system (DGSS) is proposed, which mainly consists of a node TTF prediction model, a job completion prediction model and corresponding dependable job scheduling algorithm. DGSS makes job scheduling decision by comparing the TTF of nodes with the job prediction completion time to avoid job failures. Experiment results evaluate DGSS against other scheduling algorithms and prove that DGSS can enhance the success ratio of job execution and resources utilization, and improve system performance.

References

1. Foster, I., Kesselman, C. eds.: The Grid: Blueprint for a New Computing Infrastructure. 2nd edition, Morgan Kaufmann, November (2003).
2. Krauter, K., Buyya, R., Maheswaran, M.: A Taxonomy and Survey of Grid Resource Management Systems for Distributed Computing. *Software Practice and Experience*, 32(2):135-164, February (2002).
3. Cooper, K., Dasgupta, A., Kennedy, K.: New Grid Scheduling and Rescheduling Methods in the GrADS Project. In *Proceedings of NSF Next Generation Software Workshop at International Parallel and Distributed Processing Symposium*, Santa Fe, IEEE CS Press, Los Alamitos, CA, USA, April (2004).

4. Cao, J., Jarvis, S. A., Saini, S., Nudd, G. R.: GridFlow: Workflow Management for Grid Computing. In Proceedings of 3rd International Symposium on Cluster Computing and the Grid (CCGrid'03), Tokyo, Japan, IEEE Computer Society Press, Los Alamitos, May 12-15, (2003).
5. Venugopal, S., Buyya, R., Winton, L.: A Grid Service Broker for Scheduling Distributed Data-Oriented Applications on Global Grids. In Proceedings of 2nd International Workshop on Middleware for Grid Computing (Middleware'04), Toronto, Ontario, Canada, ACM Press, New York, NY, USA, October 18, (2004).
6. Song, S. S., Kwok, Y. K., Hwang, K.: Trusted Job Scheduling in Open Computational Grids: Security-Driven Heuristics and A Fast Genetic Algorithm. In Proceedings of 19th IEEE International Parallel & Distributed Processing Symposium (IPDPS'05), Denver, CO, USA., IEEE Computer Society Press, Los Alamitos, CA, USA., April 4-8, (2005).
7. He, Y., Shao, Z., Xiao, B., Zhuge, Q., Sha, E.: Reliability Driven Task Scheduling for Heterogeneous Systems. In Proceedings of the Fifteenth IASTED International Conference on Parallel and Distributed Computing and Systems, pp.465-470, (2003).
8. Sahoo, R., Sivasubramaniam, A., Squillante, M., Zhang, Y.: Failure Data Analysis of a Large-scale Heterogeneous Server Environment. In Proceedings of the International Conference on Dependable Systems and Networks (DSN'04), Florence, Italy(2004).
9. Heath, T., Martin, R., Nguyen, T. D.: Improving Cluster Availability using Workstation Validation. In Proceedings of ACM SIGMETRICS 2002, Marina Del Rey, CA, (2002).
10. Jin, H., Shi, X., Qiang, W., Zou, D.: An Adaptive Meta-Scheduler for Data-Intensive Applications. International Journal of Grid and Utility Computing, Inderscience Enterprises Ltd., Vol.1, No.1, pp.32-37(2005),.