ISBN 978-952-5726-11-4
Proceedings of the Third International Symposium on Electronic Commerce and Security Workshops (ISECS '10)
Guangzhou, P. R. China, 29-31, July 2010, pp. 156-159

Replication Based Job Scheduling in Grids with Security Assurance

Congfeng Jiang , Xianghua Xu, and Jian Wan
Grid and Service Computing Technology Lab, Hangzhou Dianzi University, Hangzhou, 310037, China
Email:cjiang@hdu.edu.cn
Grid and Service Computing Technology Lab, Hangzhou Dianzi University, Hangzhou, 310037, China
Email:{xhxu,wanjian}@hdu.edu.cn

***Abstract*—Security assurance is a critical requirement for QoS or SLA satisfactions in risky grid environments because jobs may be scheduled to multiple machines across different distributed administrative domains. Unlike conventional methods using fixed-number job replications, in this paper, we propose a security-aware parallel and independent job scheduling algorithm based on adaptive job replications to make sure the job scheduling decision secure, reliable and fault tolerant. In risky and failure-prone grids, the replication number is changed according to the current security conditions and the end-user settings. Simulation results show that it is robust due to its adaptive job replications and rescheduling mechanism. The performance results also show that it is better than non-security-aware scheduling algorithms based on fixed-number job replications.**

***Index Terms*— adaptive replication; job scheduling; grid security; fault tolerance.**

I. INTRODUCTION

In a large-scale grid [1], Job executions are usually carried out between many virtual organizations in business applications or scientific applications for faster execution or remote interaction. However, grid security is a main hurdle to make the job scheduling secure, reliable and fault tolerant. If a host in the grid is under attack or malicious usage, its resources may not be accessible from remote sites. Thus, the jobs scheduled to that host may be delayed or failed because of the system infections or crashes.

Many algorithms have been developed for scheduling jobs in grids [2-8]. Unfortunately, most of the existing proposed scheduling algorithms had ignored the grid security while scheduling jobs onto geographically distributed grid sites with a handful of exceptions [3, 9, 10]. In a real life scenario, security threats always exist and the jobs are subject to failures or delays caused by infected hardware, software vulnerability, and distrusted security policy [3].

Furthermore, the jobs scheduled to the grid sites are subject to failure easier than in a centrally controlled or locally controlled environment because of the dynamicity of grids [11]. The failure rate grows higher when the scale of the grid becomes larger. The whole application will fail due to some key tasks or sites failures. This is not acceptable for some small granularity, large scale and long running grid applications. Job replications are commonly used to provide fault tolerant scheduling in grids. However, existing job replication algorithms use a fixed-number replication [3, 4].

In this paper, we have tackled the secure job scheduling in grids by developing security-driven and fault tolerant strategies and offered a job scheduling algorithm for use under failure-prone and risky conditions. We design a security-driven and fault tolerant scheduling strategy based on adaptive job replication. We then compare the proposed algorithm with existing heuristics based on fixed-number job replications and other trust-driven algorithms. Simulation results show that security assurance performance could be achieved, if we integrate the security-driven and fault tolerant features into job scheduling algorithms in grids.

The rest of the paper is organized as follows: Section 2 presents a brief review of related work. In Section 3, we present a trust model and specify the scheduling strategies based on adaptive job replication. We present extensive simulation results and compare the relative performance and scalability of the proposed scheduling algorithm in Section 4. Finally, we summarize the contributions in Section 5.

II. RELATED WORK

In security-driven job scheduling, the scheduling process becomes much more challenging [11,12]. Humphrey and Thompson [13] provided usage models for security-driven grid computing, such as authentication, authorization, integrity, and confidentiality. However, the implementation details of a security-driven and fault tolerant scheduler were not specified. In this paper, we focus on how to establish a security-driven and fault tolerant algorithm based on adaptive job replications and how the security-aware algorithm affects the overall performance of the user jobs in grids.

Song et al [14] developed a security-binding scheme through site reputation assessment and trust integration across grid sites. The binding is achieved by periodic exchange of site security information and matchmaking to satisfy user job demands. In this paper, we try to quantify the trustworthiness of resource sites, based on previous security performance data, such as prior success rates of job execution, cumulative site utilization, and intrinsic security attributes of resource sites.

Hwang [15] presented a generic failure detection mechanism and a flexible failure handling framework as a fault tolerant mechanism in grids. They use notification mechanism to transfer failure messages between grid sites. In this paper, our work is partially based on the above notification mechanism. We assume that there is a server to transfer messages between grid sites when jobs failed or successfully completed.

We derive security-aware parallel and independent job scheduling algorithm based on adaptive job replication for risky and failure-prone grid environment. The security-driven algorithm hereby developed can be applied to modify many other existing heuristics or sufferage algorithms.

III. Trust Model and Adaptive Job Replication

In this paper, we assume that the estimates of expected task execution times on each machine in the grid sites are known. Approaches for doing this estimation include intrinsic and extrinsic factors method [16], analytic benchmarking [17], etc. We also assume that the system components may fail and can be eventually recovered from failures. Both hardware and software failures obey the fail-stop [18] failure mode.

Let M denote hosts set, $M = \{m_j \mid j = 1,2,3,...,m\}$,T denote tasks set, T={ t_i | i=1,2,3,…,n}.We define the following parameters :

(1) SD_i : The security demand of task t_i . SD_i is specified when the task is submitted, and SD_i is a real fraction in the range [0, 1] with 0 representing the lowest and 1 the highest security requirement. In some grid environments, setting SD_i equal to 1 is unnecessary although it seems to be risk-free. If SD_i is always equal to 1, maybe there are no sites could satisfy the security demand. For example, in a volunteer grid environment such as SETI@Home[19],the security demand of jobs may be lower than a computation-intensive and real-time scientific grid, or e-commerce environment, in order to earn volunteer computing power as more as possible.

(2) TL_j : The trust level of host m_j . TL_j is in the same range [0, 1] with 0 for the most risky resource site and 1 for a risk-free or fully trusted site. TL_j is the aggregation of historical security performance data of the grid sites, such as prior success rates of job execution, cumulative site utilization, and intrinsic security attributes [3].

(3) $\overline{TL}$: The trust level of the grid environment. $\overline{TL}$ is in the range [0, 1] with 0 for the most risky and 1 for a risk-free or fully trusted grid environment. TL_j is updated periodically with the site operations. This can be achieved by using some network services like NWS (Network Weather Service) [20] and MDS (Monitoring and Discovery System) [21] when scheduling.

TL_j is computed as following:

$$TL_j = \alpha \times e^{-TB_j} + \beta \times rs_j + \gamma \times ru_j \qquad (1)$$

Where:

TB_j : denotes the time interval from the last successful task execution on host m_j , $TB_j > 0$;

rs_j : denotes the accumulative success rate of job executions on host m_j , $0 < rs_j < 1$;

ru_j : denotes the accumulative utilization rate of host m_j , $0 < ru_j < 1$;

α , β , γ are weighted factors of time interval, success rate of job executions and utilization rate respectively . α , β , γ are in the same range(0,1) with $\alpha + \beta + \gamma = 1$. α , β , and γ change dynamically according to the historical performance data of the sites.

$\overline{TL}$ is computed as following:

$$\overline{TL} = \frac{\sum_{j=1}^{m} TL_j \times p_j}{\sum_{j=1}^{m} p_j} \qquad (2)$$

We assume that there is a server which collects both SD_i and TL_j . Then $\overline{TL}$ can be computed by the scheduler through equation 1 and equation 2.In a real grid, $\overline{TL}$ can also be maintained by services like MDS [21].

IV. Simulation Results and Performance Analysis

We use simulations to study the performance of the security-driven and fault tolerant scheduling algorithm based on adaptive job replication. To evaluate the self-adaptive job replication scheduling algorithm, we use the following metrics:

• Makespan: the total running time of all jobs;

• Scheduling success rate: the percentage of jobs successfully completed in the system;

• Grid utilization: defined by the percentage of processing power allocated to user jobs out of total processing power available over all grid sites;

• Average waiting time: the average waiting time spent by a job in the grid.

A. *Simulation Results*

We first test the performance of our algorithm, named SDFTS, based on SimGrid [23] package.

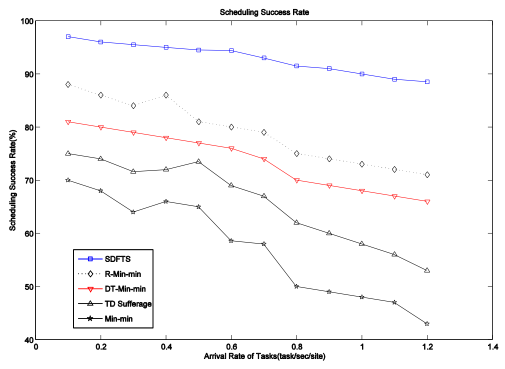
We compare the relative performance such as makespan, scheduling success rate, grid utilization and average waiting time. The simulation results are shown in Fig.1 and Fig.2. All the data in the figures are mean values of 15 simulation results, the waiting factor in DT-Min-min heuristics is 0.2, and the number of replication in R-Min-min is 2 for all simulations.

B. *Performance Comparison*

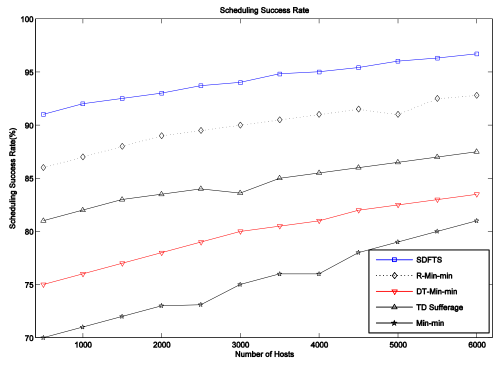
In Fig.1(a),SDFTS has the highest scheduling success rate and scales well when the number of tasks increases. It is due to the adaptive replication execution in a failure-prone grid environment.

In Fig.1(b), scheduling success rate increases when the number of hosts increases and scales well for all five algorithms. The reason is that there are more available hosts when the number of hosts increases. SDFTS also has the highest scheduling success rate because of its security-driven feature and adaptive replication execution.

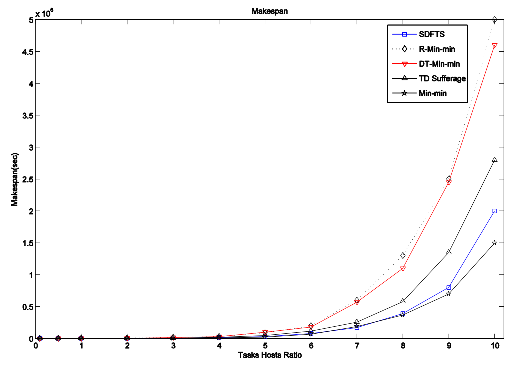
In Fig.1(c), the makespan increases when the task-host ratio increases for all algorithms. This is because that there becomes hosts-racing problem when the task to host ratio increases. The hosts-racing problem makes the tasks be executed later. However, the makespan of



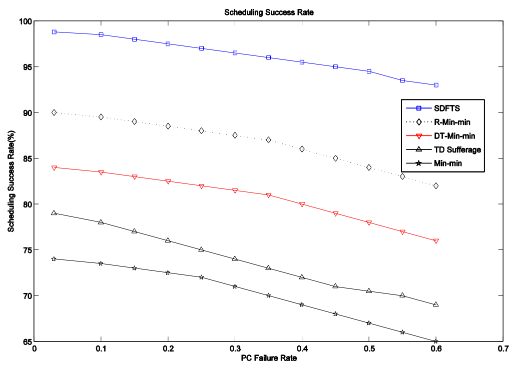
1(a)



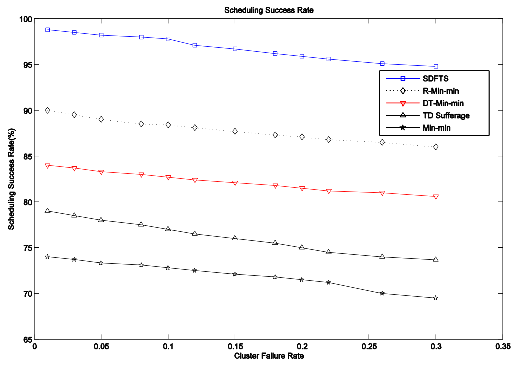
1(b)



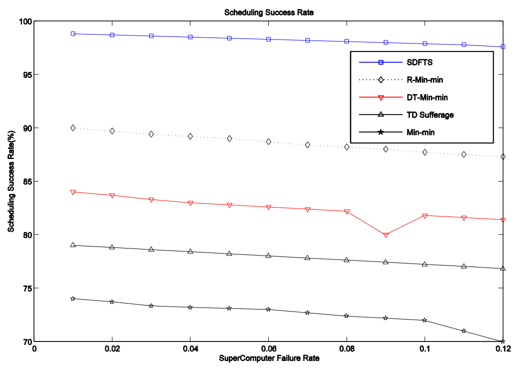
1(c)
Figure. 1 comparison of scalability



2(a)



2(b)



2(c)
Figure.2 comparison of fault tolerance

SDFTS doesn't increase dramatically when task-host ratio increases.

The results in Fig.1 show that the SDFTS algorithm has relative higher scalability than others.

The results in Fig.2 show that the scheduling success rate decreases slightly when hosts failure rate increases and SDFTS has the highest scheduling success rate than others. Consequently, SDFTS has the best fault tolerance performance. We also observe from Fig.2 that the scheduling success rate decreases more slightly when supercomputer failure rate increases than that of PCs and clusters. This is because that in our simulations the occupation frequency of supercomputer is rather small. Thus, the curve in Fig.2(c) is milder than the other two in Fig.2.

In summary, SDFTS changes the number of replications dynamically according to the dynamicity of the grid security level. Thus SDFTS is applicable to the grid where the trust level changes frequently and can reduce the number of total job replications. SDFTS reschedules the failed tasks in a failure-prone grid environment which makes the scheduling robust and higher success rate on cost of slightly longer makespan. Moreover, SDFTS scales well in a large grid and it is applicable to security-driven and fault-tolerant scheduling.

V. CONCLUSIONS

We have studied scheduling strategies that are of use when minimizing job failures and improving reliability and fault tolerance in grid environments. In our simulations, the average successful job scheduling rate is 97%, and average grid utilization is 74%. It is clear that profound changes in system performance, robustness, reliability and scalability can be achieved by security-driven and fault tolerant scheduling algorithm based on adaptive job replications. In addition, the current grid is risky and failure-prone and very few scheduling algorithms are being constructed from a security and fault tolerant perspective.

ACKNOWLEDGMENT

The funding support of this work by Natural Science Fund of China (NSFC, Grant No. 60873023 and 60973029), Science and Technology Research and Development Program of Zhejiang Province, China (Grant No. 2009C31033),and Hangzhou Dianzi University Startup Fund (Grant No. KYS 055608058 and KYS 055608033,) are greatly appreciated.

REFERENCES

[1] I. Foster, C. Kesselman, & S. Tuecke. The anatomy of the grid: enabling scalable virtual organizations, International Journal of High Performance Computing Applications, 15(3), 2001, 200-222.

[2] A. Dogana, & F. Ozguner, Scheduling of a meta-task with QoS requirements in heterogeneous computing systems, Journal of Parallel and Distributed Computing, 66(2), 2006, 181-196.

[3] S. Song, K. Hwang, & Y. Kwok, Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling, IEEE Transactions on Computers, 55(6), 2006, 703-719.

[4] J.H. Abawajy, Fault-tolerant scheduling policy for grid computing systems, Proc. IEEE 18th International Parallel and Distributed Processing Symposium (IPDPS04), Santa Fe, NM, 2004, 238-244.

[5] J. Kim, S. Shivle, H. J. Siegel, A.A. Maciejewski, T.D. Braun, M. Schneider, S. Tideman, R. Chitta, R.B. Dilmaghani, R. Joshi, A. Kaul, A. Sharma, S. Sripada, P. Vangari, S.S. Yellampalli, Dynamically mapping tasks with priorities and multiple deadlines in a heterogeneous environment, Journal of Parallel and Distributed Computing, 67(2), 2007,154-169.

[6] K. Kaya, & C. Aykanat, Iterative-improvement-based heuristics for adaptive scheduling of tasks sharing files on heterogeneous master-slave platforms, IEEE Transactions on Parallel and Distributed Systems ,17 (8) ,2006,883-896.

[7] S. Baskiyar, & C. Dickinson, Scheduling directed a-cyclic task graphs on a bounded set of heterogeneous processors using task duplication, Journal of Parallel and Distributed Computing, 65(8) ,2005, 911-921.

[8] X. Qin, & H. Jiang, A novel fault-tolerant scheduling algorithm for precedence constrained tasks in real-time heterogeneous systems, Parallel Computing, 32(5-6), 2006, 331-356.

[9] C. Jiang, C. Wang, X. Liu, & Y. Zhao, Adaptive Replication Based Security Aware and Fault Tolerant Job Scheduling for Grid Computing, accepted to appear in Proc. 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD07), Qingdao, 2007.

[10] C. Jiang, C. Wang, X. Liu, & Y. Zhao, A Fuzzy Logic Approach for Secure and Fault Tolerant Grid Job Scheduling, accepted to appear in Proc. 4th International Conference on Autonomic and Trusted Computing (ATC-2007), Hongkong, 2007.

[11] C. Jiang, C. Wang, X. Liu, & Y. Zhao, A Survey of job scheduling in grids, Lecture Notes in Computer Science Vol.4505,2007,419-427 (in press)

[12] F. Azzedin, & M. Maheswaran, Integrating trust into grid resource management systems, Proc. 31st International Conference on Parallel Processing, Vancouver, BC, Canada, 2002, 47-54

[13] M. Humphrey, & M.R. Thompson, Security implications of typical grid computing usage scenarios, Proc. 10th IEEE International Symposium on High Performance Distributed Computing, San Francisco, CA,2001, 95-103.

[14] S. Song, K. Hwang, & Y. Kwok, Trusted grid computing with security binding and trust integration, Journal of Grid Computing, 3(1), 2005,53-73.

[15] S. Hwang, Grid workflow: a flexible framework for fault tolerance in the grid, doctoral diss., University of Southern California, Los Angeles, CA, 2003.

[16] K. Jong, & K.G. Shin, Execution time analysis of communicating tasks in distributed systems, IEEE Transactions on Computers, 45(5), 1996, 572-529.

[17] M.A. Iverson, F. Ozguner, & L. Potter, Statistical prediction of task execution times through analytic benchmarking for scheduling in a heterogeneous environment, IEEE Transactions on Computers,48(12), 1999,1374-1379.

[18] F. B. Schneider, Byzantine generals in action: Implementing failstop processors, ACM Transactions on Computer Systems, 2(2), 1984, 145-154.

[19] SETI@home Project, http://setiathome.ssl.berkeley.edu.

[20] R. Wolski, N.T. Spring, & J. Hayes, Network weather service: a distributed resource performance forecasting service for metacomputing, Future Generation Computer Systems, 15(5), 1999, 757-768.

[21] J.M. Schopf, L. Pearlman, N. Miller, C. Kesselman, I. Foster, M. D'Arcy, & A. Chervenak, Monitoring the grid with the Globus toolkit MDS4, Journal of Physics: Conference Series, 46(1), 2006, 521-525.

[22] W. Zhang, X. Liu, X. Yun, H.Zhang, M. Hu, & K. Liu ,Trust-driven job scheduling heuristics for computing grid. Journal of Communication, 27(2), 2006, 73-79(in Chinese)

[23] A. Legrand, L. Marchal, & H. Casanova, Scheduling distributed applications: the Simgrid simulation framework, Proc. 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid03), Tokyo, Japan, 2003, 138-145.