Modeling and Performance Analysis of the VEGA Grid System

Haijun Yang, Zhiwei Xu, Yuzhong Sun and Qinghua Zheng
Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080
navy@ict.ac.cn

Abstract

In this paper, we propose four general queueing models based on input and server distributions, to analyze a special Grid system, VEGA grid system version 1.1 (VEGA1.1). The mean queue lengths and mean waiting times of these models are deduced. The two classic applications, the computing-oriented application (Blast computing) and online transaction processing application (air booking service) are employed to analyze and evaluate the grid system and the models. Meanwhile, we validate VEGA's four prominent characteristics, Versatile Services, Enabling Intelligence, Global Uniformity and Autonomous Control. Finally, we analyze the experiment results, predict the behavior of VEGA1.1 in equilibrium and point out the bottlenecks of the system.

1. Introduction

Like many significant concepts and technologies, the grid as a novel one has been inspired by practical requirements. The emergence of grid is due to needs of large scale resource sharing and coordinating. It was named in the mid-1990 to distinguish conventional distributed computing environments[1]. As a novel computer technology, the grid has made a rapid progress in recent years. At the same time, the grid has changed greatly in its architecture. From the layered grid architecture[1,2] to open grid services architecture (OGSA)[1,3,4], grid at present employs a new architecture called Web Services Resource Framework(WSRF)[5]. The Globus Toolkit Version2 (GT2) adopted the layered grid architecture, used as the de facto standard for grid computing. GT2 is consists of fabric layer, connectivity layer, resource layer, collective layer and application layer. The Globus Toolkit Version3(GT3) is a service-oriented grid which takes on the characteristics of OGSA. The Globus Toolkit Version4 (GT4) is compatible with WSRF and OGSA[6]. All of them apply protocols as main tools to build grid. There is another approach to establish grid, which takes the grid as a computer system [7].

The VEGA Grid consists of multiple grid nodes that could span wide-area locations, interconnected through the Internet. Within each node, there are one or more computers (servers) that host computing, storage, data, and program resources. All resources are encapsulated as services. Currently, the VEGA Grid architecture supports web services and OGSI-compliant grid services. Two new architectural components are the grid resource router and switch. They differ from a network router/switch in that they do not just route messages, but provide a link to other grid nodes and to the users. The main function of the grid router/switch is to connect grid resources together. A grid switch connects resources within a grid node, which is often managed by one institute. A grid router connects resources among multiple grid nodes, which could be owned by multiple organizations. These grid router/switch features are already implemented on a Linux server in software. The VEGA grid system has some traits that are called as versatile services, enabling intelligence, global uniformity and autonomous control[7-9].

- Versatile Services: The VEGA grid, a system platform, is compliant with current computing technologies and provides diversified services. This platform supports composition of services, services sharing and services cooperation, which provides following three basic grid operations:
  (1)File transmission,
  (2)Remote deployment & execution of programs,
  (3)Query & interaction of information.
- Enabling Intelligence: The VEGA grid possesses preliminary intelligence. The evaluation standards are described as follows:
  (1)Easy to enroll,
  (2)Easy to deploy,
  (3)Easy to organize,
  (4)Easy to program,
  (5)Automatic or semiautomatic management.

Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05)
0-7695-2448-6/05 $20.00 © 2005 IEEE

- Global Uniformity: A grid is a global uniformity from the view of end-users. That is to say, information islands do not exist in such a grid.
- Autonomous Control: The administrators, operators and resource providers are able to control their own resources. Without changing the encapsulated services, resource providers can update the hardware and the software.

At the same time, some grid teams use middleware technology to share and coordinate resources, such as e-Science, Condor, NetSolve, GrADS and so on. With the rapid progress of the grid technologies, the performance analysis of the grid becomes a critical problem. The results of the performance analysis can help us to understand the grid more correctly. Moreover, an exact performance model can predict the behavior of the grid, especially in equilibrium. A grid is a dynamical system, so it is very important to forecast its dynamical behavior.

We have found that both scientific and industrial communities can take advantages from grid technologies. Not only in the research field but also in the industry field, the performance analysis and evaluation of grid systems becomes a more increasingly interesting problem [9-15].

The goal of this paper is modeling the grid system like VEGA, testing its performance under typical applications, predict the behavior of VEGA1.1 and point out its system bottleneck. This analysis methodology can be used to other grid system.

The remainder of this paper is organized as follows. In section 2, we address the related works of grid systems. We give performance models of the VEGA grid system in section3. Then, we take two typical scenarios to validate the VEGA1.1, present some experiment results and their analysis in section 4. A discussion is given in section 5. Section 6 summarizes this paper and points out the future work.

2. Related works

In order to help people to share computing power, databases and other software, the Globus alliance has released GT2 and GT3. Its latest version is GT4. Although they did not take performance analysis as a special area in their research project, improving performance of the grid system is always one of their important work[1,6]. If we want to promote the performance, we must analyze it firstly.

European Union DataGrid(EU DataGrid) project endeavors to find a novel computer technology which can solve large-scale databases sharing between different locations. At the same time, U.K e-Science grid program began in 2001, which established the e-Science grid to solve some challenging scientific computing problems, and integrated computing, data and visualization resources on a global scale. Their goal is to share the computing and information resources at many universities and research institutes. Furthermore, they have developed some applications with workflow on the e-Science, which have been applied by some famous pharmacy corporations. In 2003, G. Fox and D. Walker interviewed with the members of e-Science community, issuing a technical paper which pointed out some faults of e-Science. In paper[14], they analyzed the e-Science grid and gave some advices to improve it. In fact, their performance analysis methodology is difficult to ensure the objectivity of the analysis. Furthermore, Zsolt et al. studied the performance evaluation of grid and pointed out that computing grid and conventional distributed computing systems are different in performance evaluation. Parameters of performance evaluation and experiment environments must reflect the characteristics of the underlying grid systems. They use the Mercury monitor to support grid system performance analysis [15]. Considering characteristics of grid, such as dynamism, complexity, and large-scale, they thought that monitor was a strong tool to analyze performance of grid systems.

Though they have different philosophical bases and development histories, Condor[16], GrADS[17,18], and NetSolve[19] all can be considered as grid middleware. GrADS supports different formats by which the earth science data is accessed, manipulated and visualized. X. Liu et al. designed a grid simulator called the MicroGrid at UCSD[19]. The MicroGrid can be used to simulate a virtual grid environment. They employed applications to validate the implementation of it. They thought that the MicroGrid could run practical applications and middleware which was helpful to understand behavior of it. Furthermore, some experiment data were shown in details. Above all of them pay attention to grid protocols. The performance analysis is related with the protocols. But not all grid systems focus on protocols, for example, the VEGA grid. The above mentioned grid systems, are all in short of mathematics models to analyze their performance. So, we will model the VEGA grid system by queueing system in the next section.

3. Modeling the VEGA Grid System

In this section, we model the VEGA1.1 with various queueing system models. In this paper, we assume that each input of the VEGA grid is a Poisson

Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05)
0-7695-2448-6/05 $20.00 © 2005 IEEE

process. From the view of input, there are two modes, the one is called single mode, another is called bulk mode. In the single mode, each arrival has only one task. In the bulk mode, each arrival has multiple tasks. On the server side, there are two different service modes, the one is single server mode, another is multiple servers one. So, we obtain four models to describe the VEGA 1.1 in fig.1.

If the service times are exponential distribution, the four models can be denoted as following queueing system models.

(a) $M/M/1$, Single input and single server.
(b) $M^{\xi}/M/1$, Bulk input and single server.
(c) $M/M/m$, Single input and multiple servers.
(d) $M^{\xi}/M/m$, Bulk input and multiple servers.

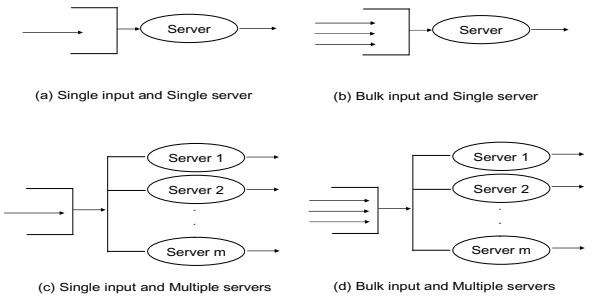

Fig.1. Queueing system models of the VEGA grid system version 1.1

The bulk input is a random variable denoted as ξ, which can take on any positive integral value but less than $+\infty$ .The expectation of variable ξ is denoted as $k = E(\xi)$, the deviation of ξ is σ_{ξ}^2. The expected value of service-time is μ and its variance is denoted as $1/\mu^2$. We use $E(Q)$ denote the mean queue length and $E(W)$ denote the mean waiting time. On the basis of existent results [20,21], we derive the following formulae straightly.

(a) $$E(Q) = \frac{\lambda/\mu}{1-\lambda/\mu} = \frac{\lambda}{\mu-\lambda} \quad (1)$$

$$E(W) = \frac{1}{\mu-\lambda} \quad (2)$$

(b) $$E(Q) = \frac{k\lambda}{\mu} + \frac{1}{2k(\mu-k\lambda)} \bullet (\mu k^2 - \mu k + k^3\lambda^2/\mu + \mu\sigma_{\xi}^2 + \mu\lambda^2 k^3\sigma^2)$$

$$= \frac{k\lambda}{\mu} + \frac{\mu k^2 + 2\lambda^2 k^3/\mu - \mu k + \mu\sigma_{\xi}^2}{2k(\mu-k\lambda)} \quad (3)$$

$$E(W) = E(Q)/k\lambda = \frac{1}{\mu} + \frac{\lambda k}{\mu(\mu-k\lambda)} + \frac{\mu(k^2-k+\sigma_{\xi}^2)}{2k^2\lambda(\mu-k\lambda)} \quad (4)$$

(c) $$E(Q) = \frac{(\lambda/\mu)^{m+1}}{m\cdot m!(1-\lambda/m\mu)^2} \bullet P_0 \quad (5)$$

Where $$P_0 = \left(\sum_{i=0}^{m-1} \frac{\lambda^i}{\mu^i \cdot i!} + \frac{\lambda^m}{m!\cdot(\mu^m - \lambda\mu^{m-1})} \right)^{-1}$$

$$E(W) = E(Q)/\lambda = \frac{\lambda^m}{m\cdot m!(1-\lambda/m\mu)^2 \cdot \mu^{m+1}} \bullet P_0 \quad (6)$$

(d) $$E(Q) = \left(\sum_{i=1}^{m-1} \frac{(k\lambda/\mu)^i}{(i-1)!} + \frac{(k\lambda/\mu)^m (k\lambda/m\mu + (m-k\lambda/\mu))}{m!(1-k\lambda/m\mu)^2}\right) P_0 \quad (7)$$

Where $$P_0 = \left(1 + \sum_{i=0}^{m-1} \frac{(k\lambda/\mu)^i}{i!} + \sum_{i=m}^{\infty} \frac{(k\lambda/\mu)^i}{m!} \frac{1}{m^{i-m}}\right)^{-1}$$

$$E(W) = E(Q)/k\lambda = \left(\sum_{i=1}^{m-1} \frac{(k\lambda/\mu)^{i-1}}{\mu\cdot(i-1)!} + \frac{(k\lambda/\mu)^{m-1}(k\lambda/m\mu + (m-k\lambda/\mu))}{\mu\cdot m!(1-k\lambda/m\mu)^2} \right) P_0 \quad (8)$$

For different application conditions, we can employ above formulae (1)-(8) to forecast the mean queue length and the mean waiting time of the VEGA grid. The two measurement values are very important to reflect the system performance. Furthermore， these theoretical results can be used in other grid systems.

4. Experiments Design and Results

There are many applications can be used in the grid system. These applications each can be placed into one of three categories: file transmission, remote deployment & execution of programs, and query & interaction of information. In this section, we will apply two typical applications to reflect the three operations. One is Blast(basic local alignment search tool) computing problem, another is air booking service problem. The first application presents two basic operations, file transmission and remote deployment & execution of programs. These programs execute batch jobs which are computing problems. The second application shows query & interaction of information, which is considered as the information service problem.

4.1. Blast Testing

In this subsection, we show the testing results respectively. There are two input modes in Blast computing. One is a concurrent arrival input, and another is a single Poisson process arrival input. In the Blast testing, three kinds of jobs were performed. Blast application batch job is called 10s job which costs about 10 seconds on a single server. So, 01m job and

Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05)
0-7695-2448-6/05 $20.00 © 2005 IEEE

10mjob are similarly named. Deploying, execution and ending time will be recorded in the experiment. Total time is sum of the three times.

4.1.1 Blast Testing with Concurrent Inputs. In this testing, we applied one node, two nodes and four nodes to perform three kinds of jobs, respectively. The experiment results are illustrated as follows.

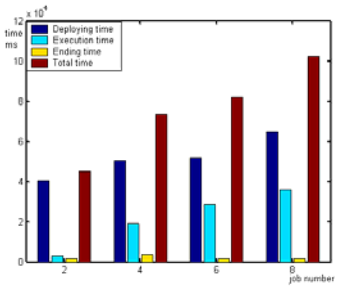

Fig.2. 10s job mean running time with one node

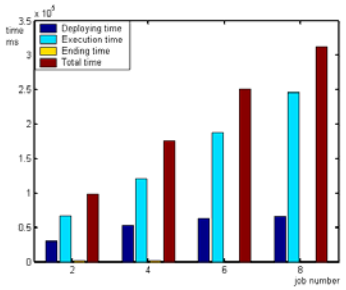

Fig.3. 01m job mean running time with one node

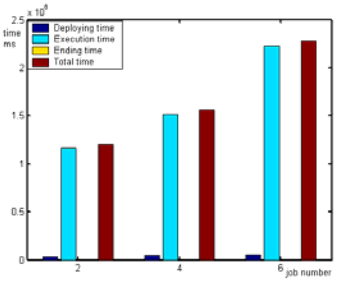

Fig.4. 10m job mean running time with one node

Form fig.2, 3 and 4, we can find that the deploying time and ending time show very slow increases when concurrent jobs increase from two to eight. For example, the deploying time increases 16.5% when concurrent jobs change from four to eight. But, the execution time presents very quick increase (about 186%), especially for short time jobs. That is to say, the VEGA1.1 has the stable mechanism to deploy batch jobs in single node. This is a basic function for a grid system. The performance on multiple nodes will be given as follows.

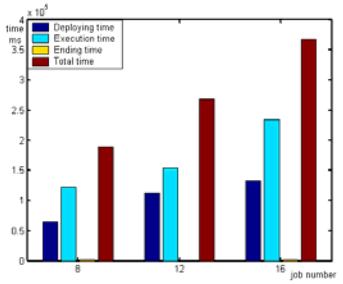

Fig.5. 01m job mean running time with two nodes

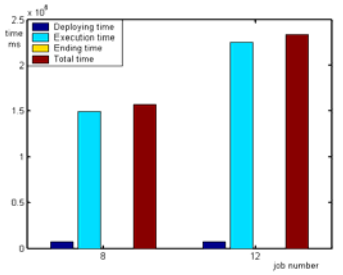

Fig.6. 10m job mean running time with two nodes

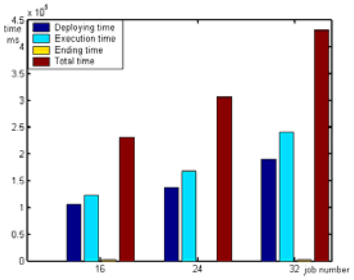

Fig.7. 01m job mean running time with four nodes

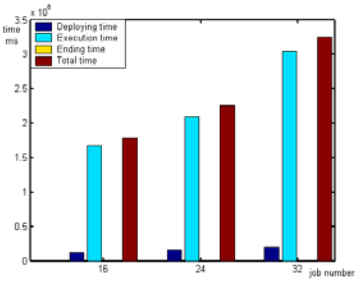

Fig.8. 10m job mean running time with four nodes

Above four figures illustrate the mean time of the VEGA1.1 by using multiple nodes. With the increase of concurrent jobs, the deploying time, execution time and ending time all increase. These increases are all very mild. For example, the total time increases 94.7% when concurrent jobs increase 100% for 01m job with two nodes. Similarly, the total time increases 82.9% when concurrent jobs increase 100% for 10m job with four nodes. The results show that the VEGA1.1 has a good performance when the jobs are doubled. These figures just address the direction of performance when

Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05)
0-7695-2448-6/05 $20.00 © 2005 IEEE

the numbers of concurrent jobs and nodes increase. Another input mode will be tested in next subsection.

4.1.2 Blast Testing with Poisson Inputs. A single Poisson process input is a very common arrival mode that has been validated in different scenarios. So we will adopt this arrival mode to test and analyze the VEGA1.1. We employ different jobs, nodes and arrival rates to complete this experiment.

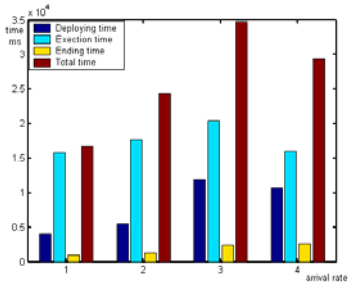

Fig.9. 10s job mean running time with Poisson arrival and one node

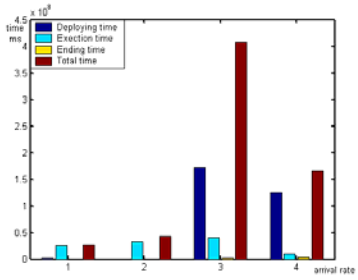

Fig.10. 10s job deviation of running time with Poisson arrival and one node

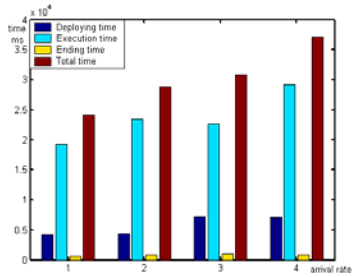

Fig.11. 10s job mean running time with Poisson arrival and two nodes

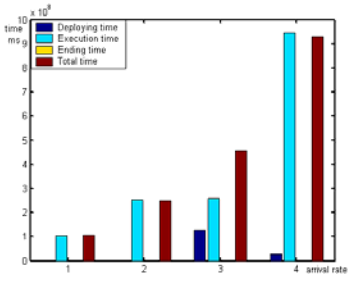

Fig.12. 10s job deviation of running time with Poisson arrival and two nodes

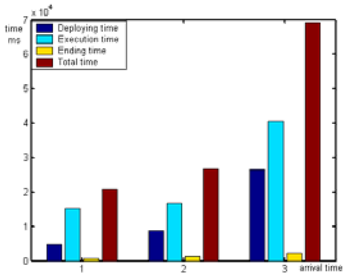

Fig.13. 10s job mean running time with Poisson arrival and four nodes

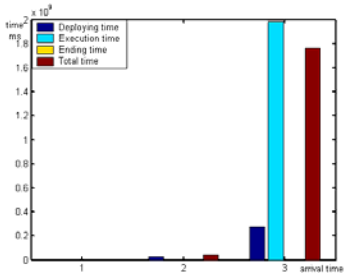

Fig.14. 10s job deviation of running time with Poisson arrival and four nodes

We first address the meaning of abscissa in above six figures. The abscissa values of 1, 2, 3 and 4 denote arrival rate with 0.2, 0.5, 0.8 and 1.2 in figure 9, 10, 11 and 12. The abscissa values of 1, 2 and 3 denote arrival rate with 0.5, 0.8 and 1.2 in figure 13 and 14.

In this experiment with single node and 10s job, we employed total 50 jobs by a Poisson process input. Fig.12 shows that mean(deploying, execution, ending and total) time increase when arrival rates increase from 0.2 to 0.8. But, an abnormal phenomenon appears when arrival rate changed to 1.2. The same situation can be found in fig.10 which is more evident than fig.9. Why does this phenomenon happen? We find that only 26%jobs have been finished at arrival rate being 1.2, which brings out the abnormal phenomenon. This condition can be predicted by formulae (3) and (4) which describe model (b) in figure 1.

Fig.12 and fig.14 show that the VEGA1.1 presented a bad stability with arrival rate being 1.2. This situation can be foreseen by results of queueing system. If maintaining this arrival rate, the VEGA1.1 will dump. These results validate our theorem models. These limit results can be obtained by queueing models. Some figures will illustrate the equilibrium state as following. At the same time, the experiment results have validated the theoretical analysis.

For 01m and 10m jobs, we will give the total time in following table. Mean time and deviate of the time will be given in the table. These data will educe some important properties of the VEGA1.1.

Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05)
0-7695-2448-6/05 $20.00 © 2005 IEEE

Table 1 The results of total time

Jobs \ Arrival rate		0.2	0.5	0.8	1.2
01m node1	mean	74927.5	87576	89091	97204.8
	deviation	92045375	190285154.9	212276896	477762865.5
01m node2	mean	75638.8	77454.1	91961.2	128986.5
	deviation	109734708	138856324.5	297377342.7	47789762414
01m node4	mean	104348.3	118391.9	104658.95	85732.9
	deviation	4.402E+09	5907102808	4741022278	150067974.8
10m node1	mean		760873.4	772944.1	775020.2
	deviation		15220576039	28074752912	17629576397
10m node2	mean		794841.3	816677.9	848325.5
	deviation		16190766696	19847736320	36153321597
10m node4	mean		742638.9	885655.4	1027309.6
	deviation		17099886886	95794427101	1.14E+16

From above data, we find that the mean time decreased at same arrival rate when the nodes changed from one to four. And the mean time increased when arrival rate increased. These trends are consistent with the theorem results in section 3. The deviation of time told us some results. With the arrival rate increasing, the stability of VEGA1.1 decreased. With the number of nodes increasing, the system stability increased. That is a good property which supports the system to enlarge its scale.

On the basis of the Blast experiment results, we applied the formulae in section 3 to predict the equilibrium state of VEGA1.1. The prediction results are illustrated as follows.

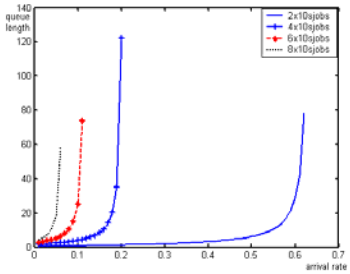

Fig. 15.Prediction of queue length with bulk input and one node

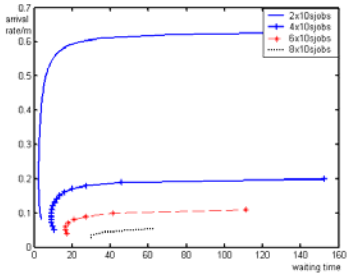

Fig. 16.Prediction of waiting time with bulk input and one node

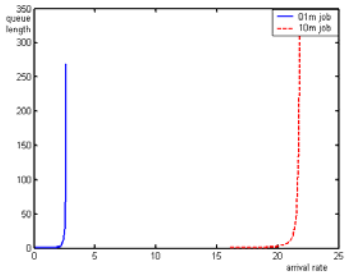

Fig. 17.Prediction of queue length with single input and four nodes

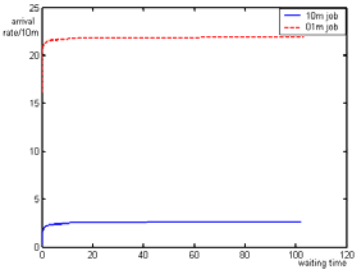

Fig. 18.Prediction of waiting time with single input and four nodes

4.2. Air Booking Service Testing

Air booking service is another typical application which can be applied in the grid system. In order to make this experiment more credible, we adopt practical data obtained from Air China Corporation instead of simulated data. MYSQL is taken as database. In this experiment, the existing air booking service has been deployed on the VEGA1.1, firstly. Secondly, the browser submits concurrent transactions to the portal container. Thirdly, the portal container transmits them to the air booking service though VEGA1.1. Finally, the air booking service sends the results to the browser. We employ threads to simulate concurrent transactions in client and change the number of concurrent transactions. We use data1, data2 and data3 to denote 10, 20 and 30 concurrent transactions in following figures. In order to avoid the influence of occasional factors, each condition has been repeated ten times.

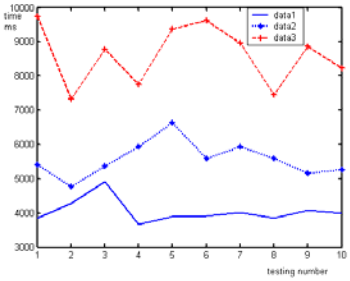

Fig. 19.The maximum running time of VEGA1.1

Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05)
0-7695-2448-6/05 $20.00 © 2005 IEEE

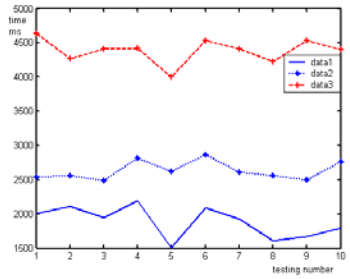

Fig. 20.The mean running time of VEGA1.1

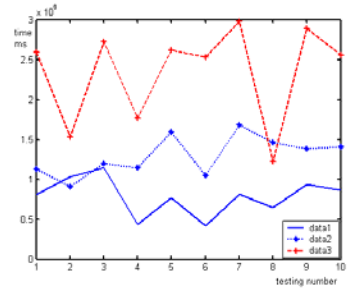

Fig. 21.The deviation of VEGA1.1 running time

Fig.19 shows that the maximum running time of VEGA1.1 increase along with the increase of concurrent transactions at the same arrivals. Specially, the time increase is rapid from data2 to data3. This phenomenon implicates that the respond time decreases very evidently. This is a critical point to evaluate the VEGA1.1. We can also validate this result by above queueing reults. Fig.20 illustrates the mean running time of VEGA1.1 at three different conditions. These curves become smoother than fig.19. But the changes among the three curves are identified with fig.19. We use deviation of running time to depict stability of the VEGA1.1 by fig.21. From this figure, we can learn that the data3 has a worst stability than the two others. This result is related with the results of fig.19, i.e., the tested system is not stability enough under current workload. This result is inevitable when the system goes to equilibrium. When the workloads beyond the equilibrium point($\lambda > \mu$), the VEGA1.1 will go to dump. This result is coincident with the above queueing analysis. Furthermore, we can use formula (1) and (2) to forecast the mean queue length and the mean waiting time, which are important to end-users and service providers.

From above experiment results, we find that too many APIs invoking operations congest the service queues. These APIs become system bottlenecks when concurrent transactions increase. There are two approaches to solve this critical problem. One is to improve the efficiency of APIs. The amelioration work will be performed as soon as possible. Another is to avoid to using overmuch APIs in grid. Virtual machine can be used to achieve the goal.

5. Discussion

Performance analysis and evaluation is a fundamental challenge to any grid systems. To model the systems exactly, choosing an appropriate mathematical tool is an important problem. In VEGA grid, the service providers provide services to end-users by the system. Each service provider can be considered as a service system. And the VEGA is a huge service system which consists of many different small service systems. The grid system is autonomous and large-scale which decide it is stochastic. So, grid systems can be considered as stochastic service system. In this paper, we obtain four news models $M/G/1$, $M^{\xi}/G/1$, $M/G/m$ and $M^{\xi}/G/m$. When $m \to \infty$, models $M/G/m$ and $M^{\xi}/G/m$ can be written as $M/G/\infty$ and $M^{\xi}/G/\infty$. The mean queue lengths are $E(Q) = \lambda/\mu$ and $E(Q) = k\lambda/\mu$, then mean waiting time are same as $E(W) = 1/\mu$. In models $M/M/1$ and $M^{\xi}/M/1$, if $k\lambda/\mu \geq 1$, the system will never get its equilibrium and go to dump. Similarly, in models $M/M/m$ and $M^{\xi}/M/m$, if $k\lambda/m\mu \geq 1$, the system will never get its equilibrium and go to dump also. For a service provider using the VEGA grid system, he can use our formulae(1)-(8) to adjust his setting. For example, he can choose an appropriate number of servers to satisfy the end-users and decrease its costs. Moreover, he also can use these results to predict the behaviors of the VEGA grid system. The critical point can be obtained immediately. Furthermore, these discussion results can be used for other grid systems.

VEGA1.1 supports remote deployment and execution of programs by Blast computing testing. Air booking service shows that VEGA1.1 can provide the function of query & interaction of information. On the other hand, the VEGA1.1 includes a file system named VEGAhotfile which supports file transmission function. Thus, the VEGA grid system supports versatile services. Furthermore, the Blast computing and the air booking service can be enrolled in five minutes after it has been encapsulated by the service providers. Hence, it can be deployed on the VEGA1.1 easily. It is convenience to maintain and manage the service for the service providers. And VEGA1.1 can provide optimization function by its grid router. So it has preliminary intelligence. Every internet user is a potential grid user. If he accedes to the VEGA grid system, he becomes a user of it. Thus, the VEGA grid system is a global uniformity. The service providers can deploy and destroy their services all by themselves. This function shows that the VEGA1.1 is an

Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05)
0-7695-2448-6/05 $20.00 © 2005 IEEE

autonomous control system. In one word, these applications embody the four characteristics of the VEGA grid system.

6. Conclusion and Future work

We propose four general queueing models which can be used to forecast the behavior of VEGA1.1. Moreover, the results can be applied to other grid systems. The VEGA grid supports computing oriented and online transaction processing applications. Its four prominent characters have been validated. Meanwhile, we point out that more APIs invoking will become a bottleneck of the grid system. So, it is useful to improve efficiency of VEGA's APIs.

In this paper, we assume that the distribution of service time is an exponential distribution. We will prove it by more experiments under different application environment. Modeling and analyzing the system deeply is our next focused work.

Acknowledgements

Supported in part by National "863" Program(No. 2002AA104310) and Theory Foundation of ICT(No. 20056130).

References

[1]I. Foster, and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure, Second Edition, Morgan Kaufmann, Amsterdam, 2001.

[2]I. Foster, C. Kesselman and S. Tuecke. "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", International Journal of Supercomputer Applications, 15(3), 2001, pp. 200-222.

[3]I. Foster, C. Kesselman , J. M. Nick, and S. Tuecke, "Grid Services for Distributed System Integration", Computer, 35(6), 2002, pp. 37-46.

[4]D. Talia, "The Open Grid Services Architecture: where the grid meets the Web", IEEE Internet Computing, 6(6), 2002, pp. 67-71.

[5]M. Baker and Ian Foster, "On recent changes in the grid community", IEEE Distributed Systems Online, 5(2), 2004, pp. 4/1 - 4/10.

[6]The Globus Project Team Open Grid Services Architecture, available at http://www.globus.org/ ogsa/.

[7]Zhiwei Xu, et al., "VEGA: A computer systems approach to grid computing Technology", The Journal of Grid Computing, Vol2. No.2 2004, pp. 109-120.

[8]Zhiwei Xu, Et al. Architectural Guidelines of VEGA GOS 2.0 Institute of Computing Technology, Chinese Academy of Science, Tech Rep:VGD-8, 2004.

[9]T. Zhang, S. Subramanyam, and A. Sucharitakul, "Optimizing Web Services Performance for Sun Java System Application Server", Sun Microsystems, Tech Rep.

[10]H. Dail, F. Bern, and H. Casanova, "A decoupled scheduling approach for Grid application development environments", International Journal Parallel & Contributed System, l63(5), 2003, pp. 505-524.

[11]D. Kondo, M. Taufer, C. Brooks, H. Casanova, and A. Chien, "Characterizing and Evaluating Desktop Grids: An Empirical Study", In Proceeding of the 18th International Parallel and Distributed Processing Symposium, Sante Fe, New Mexico, 2004, pp. 26b.

[12]M. Migliardi and R. Podest, "Performance Improvement in Web Services Invocation Framework", In Proceeding of the 18th international Parallel and Distributed Processing symposium, workshop1, 2004, pp. 113a.

[13]G. Chun, H. Dail, H. Casanova, and A. Snavely, "Benchmark Probes for Grid Assessment", In Proceeding of the 18th international Parallel and Distributed Processing symposium, Grid Performance workshop, 2004, pp. 276a.

[14]G. Fox and D. Walker, e-Science Gap Analysis, available on-line at http://www.nesc.ac.uk/ technical_papers/UKeS-2003-01/index.html.

[15]Z. Németh, G. Gombás, and Z. Balaton, "Performance Evaluation on Grids: Directions, Issues, and Open Problems", In Proceedings of the Euromicro PDP 2004, A Coruna, Spain, IEEE Computer Society Press. 2004, pp. 290-297.

[16]D. Thain, T. Tannenbaum, and M. Livny, "Condor and the Grid Douglas", in Grid Computing: Making the Global Infrastructure a Reality, 2003, pp. 299-335.

[17]F. Berman, A. Chien, et al., "The GrADS Project: Software Support for High-Level Grid Application Development", International Journal of High Performance Computing Applications, 2001, pp. 327-344.

[18] Http://Grads.iges.org/grads/grads.html

[19]Xin Liu, Huaxia Xia, Andrew A.Chien, "Validating and Scaling the MicroGrid: A Scientific Instrument for Grid Dynamics" The Journal of Grid Computing, to appear.

[20]Kleinrock, L. Queueing Systems, Vol. 1: Theory, Wiley, New York, 1975.

[21]Donald, G. and Carl, H.H, Fundamentals of Queueing Theory, Third edition. John Wiley & Sons, Inc, New York, 1998.