AGARM: An Adaptive Grid Application and Resource Monitor Framework^{*}

Wenju Zhang, Shudong Chen, Liang Zhang, Shui Yu, and Fanyuan Ma

Shanghai Jiaotong University, Shanghai, P.R.China, 200030
{zwj03, chenshudong, zhang_liang, merlin, fyma}@sjtu.edu.cn

Abstract. Most people assume that grid computing is the mainstream high performance and throughput computing platform in the future. To monitor these grid environment and provide grid information service become extremely important for using grid computing resource efficiently. In this paper we propose an adaptive grid application and resource monitor framework named AGARM. The AGARM framework can be used not only to monitor resources but also to monitor applications in the grid environment. Moreover, we show how the monitor activity is optimized by the adaptive pulling techniques, which is based on the moving average estimation algorithm.

1 Introduction

A grid [3] system is an environment in which users, such as scientists, can access resources in a transparent and secure manner. Many grid middleware have been developed to facilitate people unitize those distributed resources including grid security, grid resource monitor and management, meta-scheduler, user interface, etc. It is extremely important to monitor these grid computing resources and provide grid information service for using grid computing resource efficiently. There are some grid middleware have the capability to do that to some extent [2, 4, 8, 9, 10, 11]. But, we need a monitoring system that is scalable, flexible, adaptable, modular, open standard, and OGSA-compatible.

However, due to the complexity and diversity of the applications and resources in the grid computing environment, existing grid monitoring architecture can not monitor all of the resource belonging to the grid. When the size of computing facility grows, existing monitoring strategy will significantly increase system overhead. The dynamic characteristics of grid resources allows the grid computing resource participate and withdraw from the resource pool constantly. Only a few existing monitoring system address this characteristics partially.

This paper proposes an adaptive grid application and resource monitor framework named AGARM to solve grid application and resource monitor problems.

^{*} This research work is supported in part by the the National High Technology Research and Development Program of China (863 Program), under Grant No. 2002AA104270.

It leverages grid services and adaptive pulling techniques which based on estimation algorithm to monitor the grid applications and resources. The primarily evaluation result confirms that the AGARM framework is scalable, effective and flexible.

The rest of the paper is organized as follows. Section 2 describes overview of the AGARM. Section 3 presents the architecture of the AGARM. Section 4 introduces the prototype implementation and evaluation result; Section 5 discusses the relevant grid application and resource monitor systems. We conclude in Section 6.

2 AGARM Overview

2.1 Service and Layered Model

The number one element of SOA is the flexibility. Essentially, an SOA is a set of loosely coupled services. Each service is relatively inexpensive to build and replace if necessary. Loose coupling further allows the architecture to adapt to changes unlike the traditional tightly coupled architectures that are more brittle. In an SOA you can replace one service with another without having to worry about the underlying technology. The interface is what matters, and it is defined in the universal standards of OGSA/OGSI, Web services and XML. That is flexibility through interoperability.

The grid system usually spans different continents and administrative domains. So we keep the layer architecture in mind in the design and implementation of the AGARM.

2.2 Pull and Push Event Model

Grid application and resource status information can be divided into two categories: static and dynamic. Static status information keeps almost the same in the lifetime of the grid computing environment, such as the number of CPU of a grid computing node. Dynamic status information fluctuates with the time, such as the CPU utilization of a grid computing node. We monitor this two categories status information by means of two different event models: *push* event model for static status information and *pull* event model for dynamic status information. The evaluation result of AGARM shows that this model has the minimum overhead. The essential problem of traces of dynamic status information is how to deal with the stale information. We introduce the EWMA (Exponential Weighted Moving Average) algorithm [10, 12] to estimate the time of the next significant update. The EWMA algorithm is a very simple but effective time series estimation instrument.

3 Architecture

The AGARM framework is illustrated in Figure 1. In virtual organization A, there are many resources (compute, storage, instruments, etc.) to be monitored.

We collect and present those resources status information through application and resource monitor service components. Those grid status information of virtual organization are stored in XML database and can be utilized by other management applications such as scheduling and performance tuning services. The virtual organization is the base unit we monitor. In order to monitor different virtual organization, we just only point to them through virtual organization switcher.

Fig. 1. AGARM architecture

3.1 Components

The AGARM framework is composed of five key components: UI and Data Visualization, Virtual Organization Switcher, Application and Resource Information Collect and Present Service (ARICPS), Application and Resource Monitor Service (ARMS) and XML Database.

UI and Data Visualization: This is a web portal or a standalone application function as the user interface and data visualization. Usually, this component is integrated with resource scheduling components.

Virtual Organization Switcher: Virtual Organization (VO) is “abstract entity grouping Users, Institutions and Resources (if any) in the same administrative domain” [3]. VO switcher acts as a bridge between ARICPS in different VO and UI component.

Application and Resource Information Collect and Present Service (ARICPS): This is the collect layer in the AGARM framework. The ARICPS provides the functionality within which Service Data Elements can be collected,

aggregated, and queried; data feeds can be monitored; and Service Data Elements can be created dynamically on demand.

Application and Resource Monitor Service (ARMS): This is a grid service component function as the sensor in some resource monitoring systems. The ARMS provides data to the ARICPS either in *active* or *passive* mode.

XML Database: All the data we collect are stored in XML databases. It is very convenient to exchange data in XML format. Our prototype implementation is based on open source Berkeley DB XML.

3.2 Adaptive Pull and Estimation Mechanism

It is very essential to keep the status information updated with the minimize overhead for grid monitoring systems. The Moving Average estimation algorithm is a very effective but simple time series estimation instrument [10, 12]. The service maintains a window over the timestamps of significant measurements from the monitor service. The Moving Average estimation algorithm does an average of the time interval between two consecutive updates. At any time instance t , the estimation of the time to next pull Z_t is given by

$$Z_t = \frac{1}{N} \sum_{i=1}^N x_i, \quad (1)$$

where x_i is the i^{th} update interval value. The effectiveness of this scheme depends on the window size that is used. Larger window sizes are more suitable for usage patterns that have a gradual or long-term fluctuation patterns.

The Exponential Weighted Moving Average (EWMA) operates on the principle of assigning exponentially lower weights to older values of the time series. There are different types of EWMA estimation algorithm. The Single EWMA algorithm is effective in estimating traces which do not exhibit well defined trends or patterns. At any time instance t , the estimation of the time to next pull Z_t can be reduced by formula (1) as follow

$$Z_t = \lambda \bar{x}_t + (1 - \lambda)Z_{t-1}, \quad 0 < \lambda \leq 1 \quad (2)$$

In this definition $\bar{x}_t$ is the sample mean from time period t , λ is the weight assigned to the current observation. Smaller the value of λ is, the more the weight is given to the older values in the series. Larger values of λ result in more weight being given to more recent observations.

It is essential to choose the right value for λ . That requires some amount of knowledge of the usage patterns on the series. This implies that the value of λ should be chosen dynamically using the observed time series characteristics.

4 Prototype Implementation and Evaluation

We have implemented a prototype of the AGARM framework on top of Globus Toolkit version 3. Our evaluation preliminary proved the effectiveness of we

Fig. 2. CPU utilization of our sample application server

proposed AGARM framework and the estimation algorithm. The experiment data was collected over a three week period with a periodicity of 60 seconds on the sample application server with in the four nodes Linux cluster. As shown in Figure 2, the CPU utilization exhibits a flat behavior with only occasional changes in a small period and a large amount of fluctuation with sharp changes in a large period.

4.1 Application and Resource Monitor Techniques

In traditional systems, after a user initiates the jobs, she or he has the ability to monitor the jobs and the corresponding resource consumption as they execute. Queue systems have been used to manage job scheduling on a cluster of nodes. Most queues operate in the “batch” mode. In other words, when a user submits a job, the submission program immediately returns the user to the prompt and provides her with a ticket or job ID or token, which can be used to monitor the job at any later time.

However, there are a number of disadvantages with respect to application and resource monitoring in traditional and queue systems. For example, the monitoring requires the user have an account on each machine, requires knowing on which nodes the jobs are executing, only monitor limited status of application and resource in queue systems, etc. Those tools are well designed but do not meet the needs of grid systems.

Typically, in a grid system, when a user submits a job, the job runs with the permissions of an ordinary user. Grid systems typically span multiple organizations and administrative domains. Often grid systems run on machines that are controlled by queuing systems. Currently, there are no standard methods for monitoring the progress of jobs executed by grid systems. Different grid systems use different techniques for monitoring applications and resources.

In the AGARM framework, our Application and Resource Monitor Service (ARMS) component solves this problem practically. The ARMS component

leverages the grid services and traditional operating system provided tools to monitor the applications and resources of grid systems.

4.2 The Performance of the Estimation Algorithm

Single EWMA estimation algorithm is characterized by its parameter λ . A lower value of λ results in larger weight given to the oldest historical information, a higher value of λ implies greater weight for the more recent observations. As shown in Figure 3, the out of sync period varies along with λ . What is the optimization λ for CPU utilization usage pattern of our sample application server? We conclude that λ_{opt} ($\lambda = 0.8$) is the optimization λ from Figure 3. Does this conclusion hold for all usage patterns? This paper does not cover this problem. We will study the optimization λ of different usage pattern. This is our future work.

Fig. 3. Out of sync period at different λ

The performance of monitoring is also influenced by the window size. The window size determines the significance attached to older observations in the current estimation. Figure 4 illustrates the effect of varying window sizes on the number of pulls and out of sync period. Those data were collected at optimization λ_{opt} ($\lambda = 0.8$) for CPU utilization usage pattern of our sample application server. Intuitively, one might expect that nodes which exhibit a larger non-periodic rate of change will require a smaller window size for better response times, while nodes which exhibit a smooth fluctuation or infrequent changes will require a larger window size in order to capture this behavior.

5 Related Work

GMA (Grid Monitoring Architecture) [6] is the grid monitor architecture proposed by Global Grid Forum (GGF). It consists of three components: consumers,

Fig. 4. Effect of window size at different number of pulls

producers and a directory service. R-GMA (Relational Grid Monitoring Architecture) [1] is based on the GMA from GGF. Its strong comes from relational model.

Ganglia (<http://ganglia.sourceforge.net>) [2] is an open-source project that grew out of the University of California, Berkeley Millennium Project. It is based on a hierarchical design targeted at federations of clusters. It relies on a multicast-based listen/announce protocol to monitor state within clusters and uses a tree of point-to-point connections amongst representative cluster nodes to federate clusters and aggregate their state.

Globus MDS [4] aims to provide a standard mechanism for publishing and discovering resource status and configuration information. It have been designed to cater for the OGSA/OGSI service architecture. It is the most resemble architecture to our AGARM framework hitherto.

NWS (Network Weather Services) [7] provides accurate forecasts of dynamically changing performance characteristics from a distributed set of grid computing resources. It can produce short-term performance forecast based on historical performance measurement.

The above systems solve the grid monitor problem to some extent. However, none of these systems provides any mechanism for monitoring the grid applications execution. None of these systems is OGSA/OGSI compatible with the exception of Globus MDS3. Some of these systems, such as Globus MDS and NWS, impose heavy overhead on the being monitored resources.

6 Conclusions

This paper has discussed how the services provided by the AGARM framework help constructing monitor services for grid computing environments. A novel monitor service for grid applications and resources has been presented. We have

identified the main components of their design, and evaluated the effect of each components on the overall performance of the models. Furthermore, the single EWMA estimation algorithm and the adaptive pulling mechanism have been introduced. Therefore, the AGARM framework has its practical significance for the grid applications and resources monitor.

References

1. Andy Cooke, et al. "The Relational Grid Monitoring Architecture: Mediating Information about the Grid". Kluwer Academic Publishers. April 2004.
2. Ganglia: <http://ganglia.sourceforge.net>
3. Foster, I., and C. Kesselman and S. Tuecke, "The Anatomy of the Grid", International Journal of High performance Computing Applications, 15, 3 2001.
4. Globus MDS3: <http://www.globus.org/mds>
5. X. Zhang, J. Freschl, and J. Schopf. "A Performance Study of Monitoring and Information Services for Distributed Systems". Proceedings of HPDC, August 2003.
6. Tierney, B., R. Aydt, D. Gunter, W. Smith, V. Taylor, R. Wolski, and M. Swany, "A Grid Monitoring Architecture". The Global Grid Forum GWD-GP-16-2, January 2002.
7. NWS: <http://nws.cs.ucsb.edu>
8. Anand Natrajan, Michael P. Walker, "Monitoring Remote Jobs in a Grid System", <http://www.cs.virginia.edu/~mpw7t/papers/NatrajanWalker01.pdf>
9. Plale, B., P. Dinda, and G. Laszewski, "Key Concepts and Services of a Grid Information Service", ISCA 15th International Parallel and Distributed Computing Systems (PDCS), 2002.
10. R. Sundaresan, M. Lauria, T. Kurc, S. Parthasarathy, and Joel Saltz, "Adaptive Polling of Grid Resource Monitors using a Slacker Coherence Model", Proceedings of HPDC-12, Seattle, June 2003.
11. Ken'ichiro Shirose, Satoshi Matsuoka, Hidemoto Nakada, and Hirotaka Ogawa, "Autonomous Configuration of Grid Monitoring Systems", In Proceedings of the 2004 International Symposium on Applications and the Internet (SAINT 2004 Workshops) pp. 651-657, 2004.
12. Hunter, J. S., "The Exponential Weighted Moving Average", Journal of Quality Technology, 18, 203C210, 1986.