High Priority Requests in Grid Environment

Tariq Alwada'n[1]
School of Computing, Engineering and Digital Technologies
Teesside University
Middlesbrough, UK

Salah Alghyaline[2]
Department of Computer Science
The World Islamic Sciences and Education University
Amman, Jordan

Azmi Alazzam[3]
Computer Information System Department
Higher Colleges of Technology
Al Ain, UAE

***Abstract*—Grid computing is an enhanced technology consisting of a pool of connected machines that belong to multiple organizations in different sites to form a distributed system. This system can be used to deal with complex scientific or business problems. It is developed to help share distributed resources that may be diverted in nature and solve many computing problems. The typical decentralized grid computing model faces many challenges, such as; different systems and software architectures, quickly handling the enormous amount of grid requesters, and finding the appropriate resources for the grid users. Some of the grid requests might need to get significant attention and fast response to the other requests. Usually, the Grid Broker (GB) works as a third party or mediator between grid service providers and grid service requesters. This paper introduced a new automated system that can help exploit the grid power, improve its functionality, and enhance its performance. This research presents a new architecture for the Grid Broker that can assist with high priority requests and be processed first. This system is also used to monitor and provision the grid providers' work during the job running. It uses a multi-agent system to facilitates its work and accomplish its tasks. The proposed approach is evaluated using the Jade simulator. The results show that using the proposed approach can enhance the way of dealing with high priority requests coming to the grid.**

Keywords—Grid computing; multi-agents system; grid broker; static priority mechanism; distributed resources

I. Introduction

A. Grid Computing

A grid is a system that has the capability to organize and manage resources and services that are distributed among different control domains, utilize protocols and interfaces, and provide a high quality of services [1, 2]. The primary goal of grid computing is to increase reliability, reduce computing costs, and increase flexibility by moving computers from an entity that we buy and run by ourselves to an entity run by a third party [3, 4].

Grid computing comes into sight due to combining multi-network computer systems to create a wide-scale and heterogeneous system used to resolve scientific or industrial problems [1]. Therefore, grid computing faces many challenges, such as reducing the number of rejected jobs, finding appropriate resources, and prioritizing some requests from grid users [5]. “Fig. 1” shows an example of a Grid Model.

Grid depends on improved software that ensures seamless communication between grid nodes [6]. The Grid architecture comprises the following elements [7]: A- Grid Portal (GP): it is also called a grid interface, a virtual computing interface used by the grid users to reach the grid resources. A portal contains many characteristics, such as hiding the grid's complexity from users using a simple interface. In this way, it facilitates the classification of grid job requirements. B- The Grid Broker (GB): this element is considered the heart of grid computing. It performs significant functions in building an effective grid environment by organizing user jobs to suitable grid resources to accomplish specific targets, such as; raising resource utilization, reducing communication delays, and effectively delivering jobs among resources. It is also responsible for monitoring and provisioning jobs and sending back the outputs to the grid users.

To find appropriate resources for the grid users, the resource broker receives the GP's job requirements and searches for the right resources to meet these requirements. To do that, first, the Broker asks for all information about the free resources from the Information Service (IS) and the data information stored in the Replica Catalogue (RC) components. Then it selects the resources that can meet the job requirements. The (IS) Information is a critical element in grid computing. It is considered a directory service that keeps data about all the grid resources and their running jobs. This information can be static or dynamic. The static data can be used for specifying the hardware and the operating system requirements, while the dynamic information one is related to the job presently running, the resources available time, type of application software, policies, and disk space. All the grid resources have to register their information in the IS so that the GB can quickly locate them. The (RC) is another essential element for the GB. It offers information to help in accessing the stored data in the grid. It defines the places of data in the grid, maps logical file names to the real physical locations on grid resources, and updates data resources. The GB asks the RC for information about the data place and the access control mechanism required to use this data to utilize the grid data.

B. Multi-Agent System (MAS)

An intelligent agent is a pre-programmed component intended to support other parts to accomplish a task or fulfill a pre-defined job. MAS is a group of agents that cooperate and work together with the outer system. It can work with a pre-defined level of self-government to do the requested jobs or complete particular tasks. MAS is incorporated to operate independently in a changing environment [8]. It has many features such as [9], the ability to cooperate, and connect with other agents to acquire commands, exchange data, and supply responses. The MAS can also work dynamically on behalf of the programs or clients, enhance the performance regularly while working with the external environment, and negotiate where the MAS can run channels for communication between agents to achieve a specific level of cooperation.

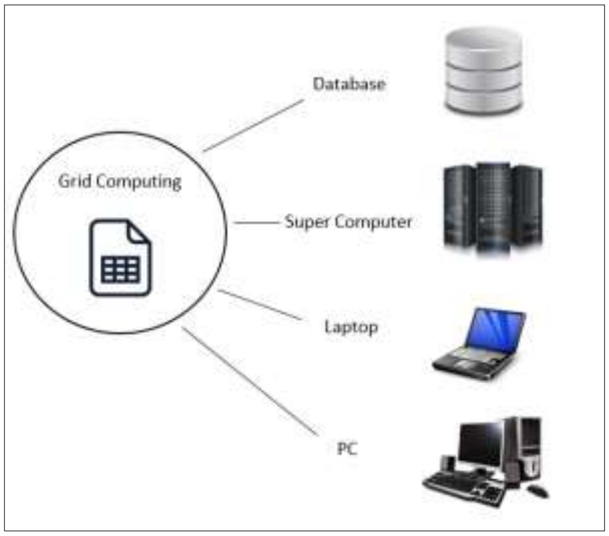

Fig. 1. Grid Computing Model.

Several methods were introduced for the development of a system that includes many agents. For example, knowledge-based methodologies, such as CommonKADS [10] or Multi-agent System Engineering (MaSE) [11]. The proposed system in this research is based on MaSE, where the GB is employing the MAS to negotiate the job requirements and conditions with the grid users. The GB also uses the MAS to monitor and exchange information with the grid providers and provision the running jobs.

C. Congestion Management Algorithms

Due to the rising requests on the grid services, applications, and grid service requesters request to reduce the running time for their tasks while minimizing the related cost. Also, both ask for the minimum QoS through the execution of their jobs. Most of the applications now are somewhat sensitive to the delays faced when transmitting and running their jobs to the grid. So, it is required to support different types of traffic with various QoS [12]. One crucial question to answer is distributing the users' jobs with a high priority value to the available resources on the grid.

This research introduces an advance redirecting policy that can be used to pick high priority requests from the grid users (as a paid service) and pass them by the GB to the appropriate grid resources. To do that, there is a demand for queuing the low priority requests and keep them in the GB memory until the high priority requests get processed by the Broker.

Our proposed technique is adopted from the cisco mechanism for queuing on router interfaces, where the queuing on cisco consists of hardware and software modules [13]. The proposed technique is used to facilitate the management of the congestion that might occur on the GB side. The chosen mechanism is the Static Priority Scheduler mechanism. This mechanism classifies packets automatically, with every flow being situated into a separate queue according to its priority [13]. In the proposed system, if there are any jobs with high priority flags coming to the grid, the GB directly processes these jobs and put them at the beginning of the queue to forward them to the appropriate grid resources first. In this case, the GB guarantees high-speed processing time for this type of traffic.

This paper is organized as follows. Section 2 presents a literature review. Section 3 offers the proposed system model. The experimental results are discussed in Section 4. Finally, the conclusion of this paper is presented in Section 5.

II. Related Works

G. Garimella, in [14], applies a reservation server that runs in cooperation with the Dynamic Soft Real-Time system [15]. The Garimella system aims to preserve the resource of the CPU in advance. In Advance Reservations Server, the client requires to state some QoS factors, such as that CPU percentage needed, the start time, and the duration. After the reservation request is initiated, the booked resources shall be accessible for the user following the beginning time for the time at the predetermined percentage. Yet, in practice, the majority of applications have QoS conditions that are negotiable. Since the reservation server does not keep up re-negotiations, this produces many reservation requests that have been rejected.

Another suggested model is the Resource Broker (RB) indicated in [16] combined with the reservation server introduced in [14]. The RB enhances the reservation server to give constant and fast reply time and several negotiation possibilities for the clients. The amount of re-negotiation inserts additional extensive overhead to the system in case of an unexpected shortage of resources. Thus, to preserve a resource to several contending applications, there is a need for a mechanism by the admission control to estimate and differentiate between different applications to refuse fewer priority applications first to assure the full set of clients obtain most of the benefit.

W. Smith and others in [17] introduced and evaluated many algorithms for maintaining enhanced reservations method in scheduling systems for supercomputers. These algorithms enhance regular organizing algorithms by uniting routine scheduling jobs from queues with the reservation demand. These enhanced reservations allow users to ask for multiple resources concurrently from scheduling systems at certain times. Nevertheless, [17] allocates the "time slots" entirely. In other words, the resources are not kept up in a shared way by many users for the same period. The applications are presumed to run on a best-effort basis, and the booking requests are then assumed to have diverse priority between the applications. These priorities are taken into account as the applications and the system schedules the reservations.

In [18], R. Min and M. Maheswaran suggested a Functions called (RSPB). The RSPB stands for Reservation Scheduler with Priorities and Benefit, which allows for algorithm program reservations while allowing for the relation priorities of the several reservation requests. In RSPB, every reservation call has a related profit function that measures the client's profit by saving the resource at the needed stage. Once the user is prepared to negotiate for short service stages, it could specify this by offering a profit function that displays a cut but significant profit for short resource stages. This capability provided by the functions eliminates the necessity for negotiations as there is a resource shortage.

III. System Model

In this section, the details of our proposed system are presented. The system is a multi-agent system that considers several aspects when making the final decisions, such as the number and shortage of grid provider (resources), the hardware and software specifications, and the customers' priority.

“Fig. 2” shows the Architecture of our automated system, which can assist in exploiting the grid power, improving its functionality, and enhancing its performance. The system is designed to allow the GB to deal with high priority requests and be processed first. The system is divided into three parts; The Grid Portal part, the Grid Broker part, and the Grid Service Provider part.

A. *The Grid Portal Part*

A grid portal or grid interface is an interface for authorized grid users to reach the grid. A portal is essential as it is hiding the grid's complexity from users by a friendly interface, which can simplify the organization of grid job requirements. Our system has used the multi-agent system by creating a local agent attached to each grid user. The local agent is responsible for classifying the priority of the grid jobs. As a paid services, the grid users can flag their careers with a high priority in order to notify the GB about this job.

B. *Grid Broker Part*

This part is divided into three main elements. The main one is the monitor and the scheduler. This element is responsible for receiving the grid users’ jobs and determining the appropriate free resources. The scheduler is responsible for checking the users’ jobs for a priority flag. If there is no flag, that means the job is treated as a normal priority (low priority), and it will be served according to the First-In-First-Out mechanism. If the job flag is high, it will be moved to the next one in the queue. The secular asks for all information about the free resources from the Information Service (IS) element and the data information stored in the Replica Catalogue (RC) elements. The monitor is responsible for monitoring and provisioning the running jobs in the grid providers.

C. *Grid Provider Part*

After the Monitor and Scheduler sends the jobs to the appropriate resource(s) for operating, it gets messages periodically from the agents at the providers' side to check the status of the running jobs. The local agent at the providers is also responsible for informing the Broker when the job is done. The provider can contain many physical and virtual machines. The Hypervisor in each machine is responsible for creating and organizing the virtual machines on top of each physical machine.

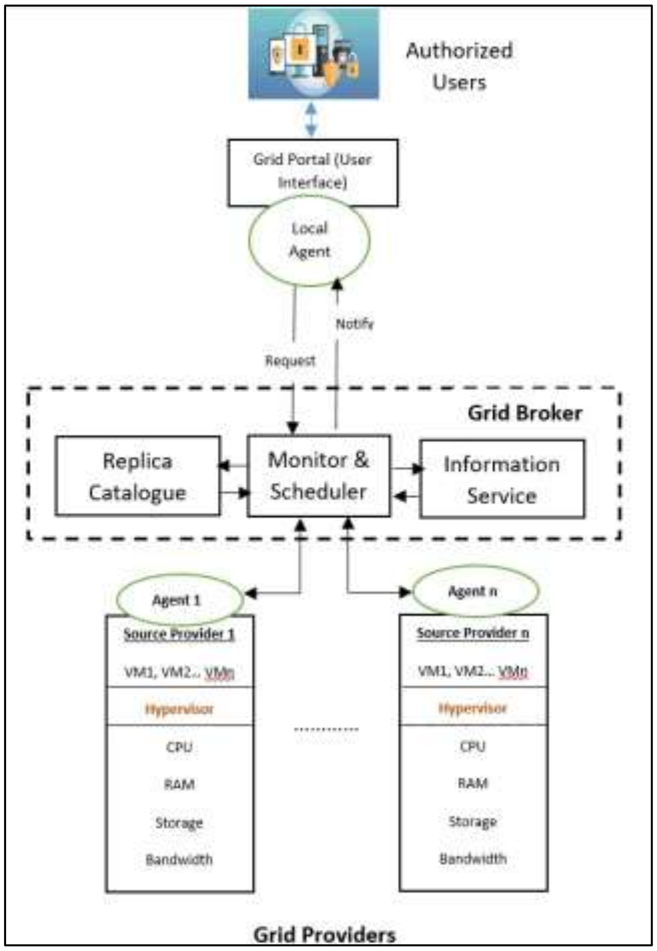

Fig. 2. System Architecture.

IV. Experiments and Results

In order to evaluate the performance of our proposed system, a Jade tool was employed. Jade is designed to support the agent software development as specified by the FIPA qualities for applied smart multi-agent systems.

In our experiment, two tests were created. Both tests have the same settings. These settings include 225 jobs from 3 grid users (75 each), with one physical machine and 4 virtual machines as grid providers. The Grid Broker is programmed to accept up to 125 jobs and reject all the jobs above that. The simulation time is 300 seconds. “Tables I, II and III” show the properties of the physical machine, the VMs, and the jobs, respectively.

Table I shows the number of virtual machines and their types. Also, it shows the Microprocessor without Interlocked Pipelined Stages (MIPS) for the CPU, Ram capacity, the network bandwidth for each VM, and the storage capacity.

Table II shows the characteristics of the physical machine used in the experiment. This includes the number of hosts and types of VMs and MIPS of the CPU, RAM, storage capacity, and network bandwidth.

Table III shows the characteristics of each job include job length and size.

TABLE I. VMS CHARACTERISTICS

Number of VMs	4
VMs types	4
MIPS of CPU	1000
Number of PEs	1
RAM Capacity	1048
Bandwidth	100 Mbit/s
Storage Capacity	10 GB

TABLE II. PHYSICAL MACHINES CHARACTERISTICS

Number of Hosts	1
VMs types	4
MIPS of CPU	1000
Number of PEs	1
RAM Capacity	2048
Bandwidth	1 Gbit/s
Storage Capacity	10 GB

TABLE III. JOBS CHARACTERISTICS

Number of jobs	225
Job Length	1200
File Size	300
Output Size	300

In the first part of the experiment, no paid services for high priority jobs were done. In this case, the local agents for each of the three users will consider all the jobs as a low priority (priority is equal for the three users). In the second part of the experiment, the local agent for one of the users labeled the jobs as a high priority while the other two grid users with low priority.

Both tests planned to send all the jobs to the grid broker at the same time to check the time required to process all the jobs together. It is also intended to check the number of rejected jobs because of the large number of requests.

The experiment's two tests were compared and assessed on several facets, such as the time needed to complete the jobs and the number of jobs rejected for each user.

First Part: 225 jobs were sent to the grid broker from 3 users (75 jobs from each). No priority in this part. The Broker got the jobs and processed them as FIFO jobs. "Fig. 4" shows that the first 75 jobs were coming from the first user treated without any rejection. For the second user, only 50 jobs were accepted, and 25 rejected. But the Broker declined all jobs from the third user as the Broker reached the bounders of the accepted jobs (as in its policy). The last user had to resend the jobs later for processing. "Fig. 5" shows the processing time for the first part of the experiment. The needed time to complete the first user's jobs is less than the second and third users. The processing time for the third user is the longest one, as the user has to resend the jobs again to the grid and wait until the services at the grid become available.

Second Part: 225 jobs were sent again to the Broker. This time the local agent labeled the jobs for user number 3 as high priority (paid service). The other users stay as a low priority (normal priority). "Fig. 4" and "Fig. 5" present the rejected jobs and the processing time correspondingly for the second part of the experiment. "Fig. 3" shows the part of the simulation for this experiment.

"Fig. 4" shows that user 3 has made use of the paid service priority service, as all of user 3's jobs were fulfilled. While "Fig. 5" shows that the time needed for the processing of user 3's jobs was minimal compare to the first part. Furthermore, there are no rejected jobs for user 3. The other results show that user 1 has 25 rejected jobs, while user 2 has 75 rejected jobs, and that was because the Broker processed their jobs according to the FIFO mechanism.

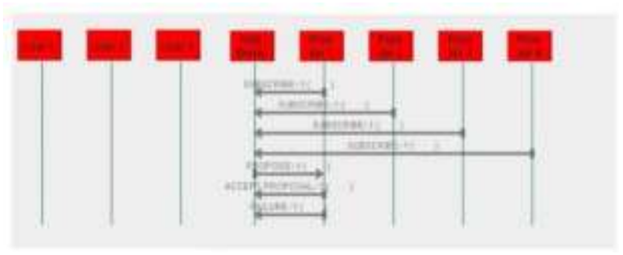

Fig. 3. Simulation for both Parts.

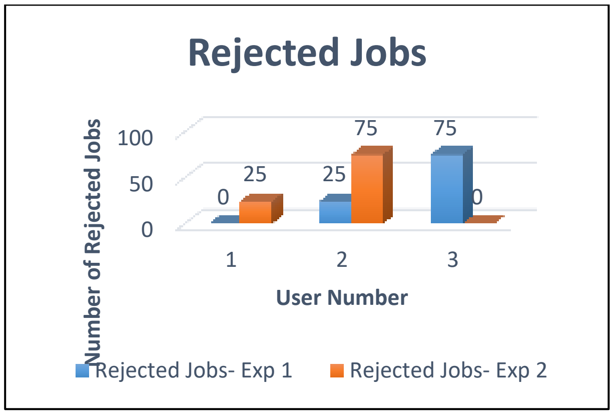

Fig. 4. Rejected Jobs for both Parts.

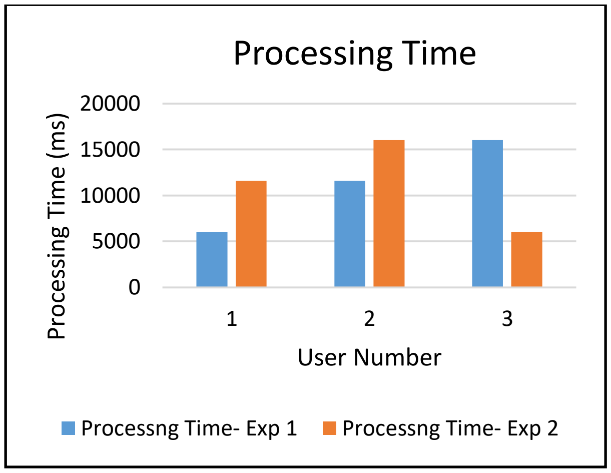

Fig. 5. Processing Time for both Parts.

V. Conclusions and Future Works

Grid computing comes into sight due to combining multi-network computer systems to create a wide-scale and heterogeneous system used to resolve scientific or industrial problems. This paper presented a dynamic management system that can handle the high priority client requests as paid service to get a fast response for their jobs. Also, this system is used to provision and monitor the grid providers' work during the job running. The proposed system is evaluated using the Jade tool. The results show that using our system enhances the degree of customers' (QoS) requirements during the resource utilization, helps exploit the grid power, improves its functionality, and enhances its performance. Our future work includes improving our proposed system's functionally when dealing with multiple grid brokers instead of one Broker.

References

[1] Ian Foster and Carl Kesselman, editors. The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[2] T. Alwada'n, H. Aldabbas, H. Janicke, T. Khdour and O. Aldabba. Dynamic policy management in approach grid environments. International Journal of Computer Networks & Capproachcations (IJCNC), 4(2), 35–51. 2012.

[3] I. Foster's. weblog. http://ianfoster.typepad.com/blog/2008/01/theres-grid-in.html. 2008.

[4] H. Alhakami, H. Aldabbas, and T. Alwadan. Comparison between cloud and grid computing: Review paper. International Journal on Cloud Computing: Services and Architecture (IJCCSA), 2, 08 2012.

[5] T. Alwadan, H. Janicke, O. Aldabbas, and M. Alfawair. New framework for policy support for mobile grid services. The 6th International Conference on Risks and Security of Internet and Systems (CRISIS2011), Romania 2011.

[6] T. Alwadan, H. Janicke, O. Aldabbas, and H. Aldabbas. New framework for dynamic policy management in grid environments. In Recent Trends in Wireless and Mobile Networks, Third International Conferences, WiMo 2011 and CoNeCo 2011,, volume 162, pages 297–304. Springer, 2011.

[7] T. Alwadan, H. Janicke, A. Alarabeyyat, A. Alkharabsh, and T. Khdour. Policy-based support for mobile grid services. International Journal of Computer Science Issues (IJCSI), 10, 01 2013.

[8] T. Alwada'n, A. Alarabeyyat, T. Khdour and A. Rodan, "Utilizing Multi-Agent Systems in Grid Environments," 2019 Sixth HCT Information Technology Trends (ITT), Ras Al Khaimah, United Arab Emirates, pp. 138-143, doi: 10.1109/ITT48889.2019.9075099, 2019.

[9] A. S. Grimshaw, M. A. Humphrey, and A. Natrajan. A philosophical and technical comparison of legion and globus. IBM J. Res. Dev., 48:233– 254, March 2004.

[10] Rajkumar Buyya, David Abramson, and Jonathan Giddy. A case for economy grid architecture for service-oriented grid computing. In Proceedings of the 15th International Parallel & Distributed Processing Symposium, IPDPS '01, pages 83–, Washington, DC, USA. IEEE Computer Society. 2001.

[11] Zsolt N. Ne´meth and Vaidy Sunderam. A formal framework for defining grid systems. In Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID '02, pages 202–, Washington, DC, USA. IEEE Computer Society. 2002.

[12] Szilágyi, Szabolcs and Béla, Almási. (2012). A Review of Congestion Management Algorithms on Cisco Routers. Journal of Computer Science and Control Systems. 5. 103-107.

[13] QOS, "Implementing Cisco Quality of Service", Student Guide, Volume 2, Version 2.2, © 2006 Cisco Systems Inc.

[14] G. Garimella, "Advance CPU Reservations with the Dynamic Soft Real-Time Scheduler", Master's Thesis, University of Illinois at Urbana-Champaign, 1999.

[15] H. Chu and K. Nahrstedt, "A Soft Real Time Scheduling Server in UNIX Operating System," European Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS '97), Sep. 1997.

[16] K. Kim and K. Nahrstedt, "A Resource Broker Model with Integrated Reservation Scheme," IEEE International Conference on Multimedia and Expo 2000 (ICME '00), Aug. 2000.

[17] W. Smith, I. Foster, and V. Taylor, "Scheduling with Advanced Reservations," International Parallel and Distributed Processing Symposium (IPDPS '00), May 2000.

[18] Rui Min and Muthucumara Maheswaran. Scheduling Co-Reservations with Priorities in Grid Computing Systems. 13th International Conference on Parallel and Distributed Systems, Anaheim, 2002. pages 266–268. 2002.