

Comparative Study a Performance and Capability Scheduling Techniques in Grid Computing

Mohd Kamir Yusof

Fakulti Informatik
Universiti Darul Iman Malaysia (UDM)
21300 Kuala Terengganu, Malaysia
mohdkamir@udm.edu.my

Muhamad Azhar Stapa

Fakulti Sains Komputer dan Sistem Maklumat
Universiti Teknologi Malaysia (UTM)
81310 Johor, Malaysia
stapaazhar@yahoo.com

Abstract— This paper shows the comparative study about existing scheduling technique in grid computing. This paper also shows the performance and capability for each scheduling technique. Four existing scheduling technique are AppLeS, Condor-G, Nimrod/G and GrADS have been studied and analyzed. Several experiments have been done for each technique by using simulation and tested in real grid. The results shows a performance and capability for each scheduling technique and was analyzed in summary table.

Keywords- AppLeS, Condor-G, GrADS, Grid Computing, Nomrod/G, Scheduling.

I. INTRODUCTION

Grid computing is a growing network application that can be expected to become an important and required component of the new global knowledge of the 21st century. The term "Grid" was ideal to explain this environment as an analogy to the electric power grid that is a major pervasive, readily available resource that empowers multiple different devices, systems and environments at distributed site.

The term "Grid Computing" originate in the early on 1990s as a mark for making computer power as easy to access as an electric power grid in Ian foster and Carl Kesselmanns seminar work, "the Grid: Blueprint for a new computing infrastructure". Computer scientist Ian Foster has developed the software to take shared computing to a global level. Just as the internet is a tool for mass communication, Grids are a tool for amplify computer power and storage space [1].

Grid will help to promote the internet to a true computing platform, combining the qualities of service of enterprise computing with the ability to share heterogeneous distributed resources-everything from applications, data, storage and servers [2].

Another definition, from The Globus Alliance (a research and development initiative focused on enabling the application of Grid concepts to scientific and engineering computing), is as follows [3]:

"The Grid refers to an infrastructure that enables the integrated, collaborative use of high-end computers, networks, database, and scientific instruments owned and managed by multiple organizations. Grid applications often involve large amounts of data or computing and often require secure resource sharing

across organizational boundaries, and are thus not easily handled by today's Internet and Web infrastructures".

II. GRID WORKS

Many of the big ideas behind the Grid have been around long before the name Grid appeared, but there are five big areas where Grid developers are spending their time and efforts. Sharing is very essential for the Grid. Resource sharing is the root of the Grid philosophy [4]. The Grid is about putting mechanisms in place so that everyone involved benefits from the efficiencies and advantages of sharing. The Grid will give access to extra computing power and can compute things using the Grid that cannot compute using just one computer or one computer centre.

There must be a high level of trust between resource providers and users, who often don't know each other. Sharing resources is fundamentally in conflict with the conservative security policies being applied at individual computer centers and on individual PCs. The second big idea behind the Grid is secure access; this is a direct consequence of the first big idea. Secure access to shared resources is one of the most challenging issues for Grid development. To ensure secure access, Grid developers and users need to manage three important things are access policy, authentication and authorization [5].

The Grid needs efficiently to track all of this information, which may change from day to day. This means the Grid needs to be extremely flexible, and has a reliable accounting mechanism. Ultimately, such accounting will be used to decide pricing policy for the Grid.

III. GRID SCHEDULING SYSTEMS

Overviews of some current Grid scheduling that have been used in Grid environments. This scheduling has been implementing in real Grid environment.

A. Condor-G

Gondor-G [6] represents the marriage of technologies from the Condor and Globus projects. The Globus [7] comes the use of protocols for secure inter-domain communications and standardized access to a variety of remote batch systems. The Condor, it comes the user concern of job submission, job allocation, error recovery, and creation of a friendly execution environment. The result is a tool that binds resources spread

across many systems into a personal high-throughput computing system.

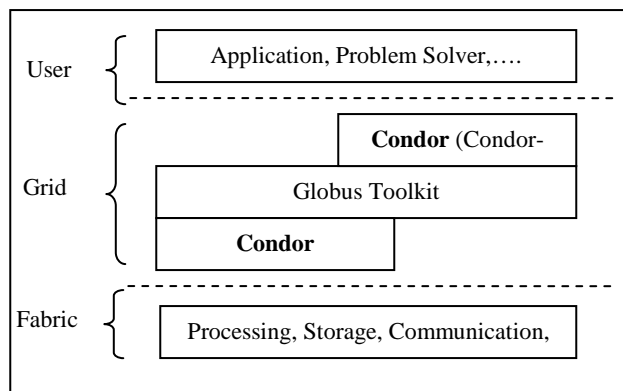


Fig. 1 Condor in the Grid [8]

Condor technology can exist at both the front and back ends of a Grid, as depicted in figure 1. Condor-G can be used as the reliable submission and job management service for one or more sites, the Condor High Throughput Computing system can be used as the fabric management service (“a grid generator”) for one or more sites and the Globus Toolkit can be used as the bridge between them.

The goal of the Condor project [9] is to develop, implement, deploy and evaluate mechanisms and policies that support high-throughput computing (HTC) by using distributed environments that can deliver large amounts of processing capacity over long periods of time.

The key to HTC is effective management and exploitation of all available computing resources. Recent trends in the cost/performance ratio of computer resources in the hands of individuals and small groups. These distributed owners will be willing to include their resources in an HTC environment only after they are convinced that their needs will be addressed and their rights will be protected. It is the owner of each workstation in the collection who defines the conditions under which Condor can allocate the workstation to an external user.

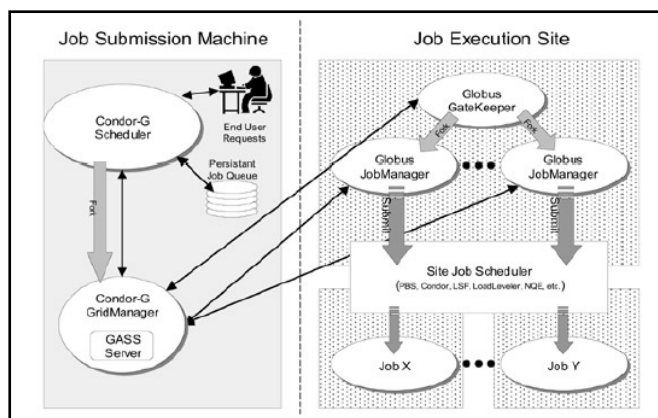


Fig. 2 Condor/Condor-G Scheduling system

When a user submits a job to Condor, the system finds an available machine on the network and begins running the job

on the machine as shown in figure 2. If Condor detects that a machine running a Condor job is no longer available, Condor can checkpoint the job, move (migrate) it to a different machine, otherwise that would be idle and continue the job on new machine from precisely, where that job left off.

Condor provides powerful resource management by matchmaking resource owners with consumers. Condor implements ClassAds [10,11], a way to describe jobs and resources in a fashion similar to a newspaper’s classified ads. All machines in the Condor pool use a resource offer ad to advertise their resource properties, both static and dynamic, such as available RAM memory, CPU type, CPU speed, virtual memory size, physical location, and current load average. A user specifies a resource request ad when submitting a job. The request defines both the required and desired set of properties of the resource to run the job. Condor acts as a broker by matching and making resource Grid offer ads with resource request ads, making certain that all requirements in both ads are satisfied.

B. AppLeS

AppLeS (Application-Level Scheduling) [12] project at the University of California, San Diego primarily focuses on developing scheduling agents for individual applications on production computational Grids. It uses the services of Network Weather Service (NWS) to monitor changes in performance of resources dynamically. AppLeS agents use static and dynamic application and system information while selecting a viable set of resources and resource configurations. It interacts with other resource management systems such as Globus, Legion, and NetSolve to implement application tasks. The applications have embedded AppLeS has been applied to many application areas including Magnetohydrodynamics [13], Gene Sequence Comparison, and Tomography [14]. Another effort within AppLeS project framework is the development of AppLeS templates. It is similar to Nimrod-G framework and resource broker, but it does not support quality of services-driven scheduling since it does not take Grid economy into consideration.

As a focus of AppLeS project is on scheduling, it follows the resource management model supported by the underlying Grid middleware systems. An AppLeS scheduler is central to the application that performs mapping of jobs to resources, but the local resource schedulers perform the actual execution of application units similar to Nimrod-G, AppLeS schedulers do not offer QoS support and build on a resource model supported by an underlying system. AppLeS can be considered to have a predictive heuristic state estimation model with online rescheduling and application oriented scheduling policies.

The AppLeS project has developed several templates specific to application types. One of these is the AppLeS Parameter Sweep Template [15]. Parameter sweep applications (PSAs) are typically structured as sets of “experiments”, each of which is executed with a distinct set of parameters. Although parameters sweep applications are independent (i.e. they do not communicate), many PSAs are structured so that distinct experiments share large input files,

and produce large output files. In order to achieve efficiency for large-scale runs, shared data files must be co-located with experiments and the PSA must be scheduled to adapt to the dynamically fluctuating delays and qualities of the service of the shared resources. The end user or application developer provides its AppLeS agent with application-specific information about current implementation(s) (via the Heterogeneous Application Template) as well as user preferences. This information is combined with dynamic system information (provided by the Network Weather Service) by the AppLeS Coordinator to determine a potentially performance-efficient application schedule. The coordinator then works with the appropriate resource management systems to implement the schedule on the relevant resources.

C. Nimrod/G

Nimrod is a specialized parametric modeling system that enables a user to run distributed parametric modeling experiments [16]. Nimrod uses a simple declarative parametric modeling language to express a parametric experiment. Its functionality includes running and monitoring multiple experiments. Results from those experiments are also gathered by Nimrod and placed in a single location. Nimrod has been applied to a range of application areas, including Bioinformatics, Operations Research, Network Simulation, Ecological Modeling and other areas.

Over the years, it has been in development, Nimrod was evolving and today there are a number of different Nimrod flavors. Originally Nimrod was developed so it could be run on department clusters and utilized these department resources. Nimrod/G is a research prototype and different from original Nimrod in that it operates on a computational grid (this site is all about Nimrod/G). Nimrod/O has been designed for performing automatic design optimization.

As the title states, Nimrod/G is a "tool for distributed parametric modeling". Usually such applications need to be run multiple times, varying key parameters in order to explore as much of the space so that the results are more accurate and complete. With applications that have small number of parameters, or rather small number of possible parameter combinations, it is not that difficult to start that application many times. On the other hand, if we imagine an experiment where there are hundreds or even thousands of possible parameter combinations it seems very unlikely that someone will be willing to create all those possible combinations and run the application manually. Moreover, if the application is a CPU intensive task that takes long time to complete, it gets more frustrating when waiting for the results that you might want very quickly. One can easily see all the advantages of a system that would allow them a quick parameter sweep generation as well as job distribution to multiple resources to simultaneously run multiple parameter combinations.

This is where Nimrod/G can help greatly as it is designed to run experiments with small or large number of parameter combinations. Nimrod/G uses a simple declarative language which user can use to tell Nimrod/G what the experiment attributes are (parameter types and possible values) and what

Nimrod/G needs to do in order to complete a single parameter combination execution. Once this is done, Nimrod/G will have all it needs to run the complete parameter sweep of that experiment [16].

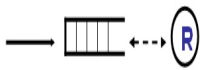
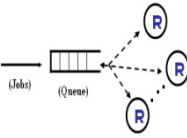
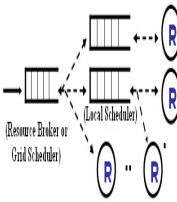
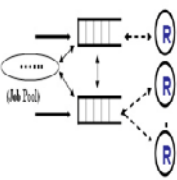
D. GrADS

The Grid Application Development Software (GrADS) project [17] aims to provide programming tools and execution environments for ordinary scientific users to develop, execute, and tune applications on the Grid. GrADS is a collaboration between several American Universities. GrADS supports application development either by assembling domain-specific components from a high-level toolkit or by creating a module by relatively low-level (e.g., MPI) code [18].

GrADS provide application-level scheduling to map workflow application tasks to a set of resources. New Grid scheduling and rescheduling methods are introduced in GrADS. These scheduling methods are guided by an objective function to minimize the overall job completion time (make span) of the workflow application. The workflow scheduler ranks each qualified resource for each application component. A rank value is calculated by using "a weighted sum of the expected execution time on the resource and the expected cost of data movement for the component." After ranking, a performance matrix is constructed and used by the scheduling heuristics to obtain a mapping of components onto resources. Three heuristics have been applied in GrADS; those are Min-Min, Max-Min, and Sufferage heuristics [19].

GrADS have built up an architecture-independent model of the workflow component from individual component models. It employs analytical models that are constructed semi-automatically from empirical models (historical data/sample execution data), in order to estimate the performance of a workflow component on a single Grid node. It uses hardware performance counters to collect operation counts from several executions of the workflow components with different, small-size input problems, and then it performs a least-squares fit to the data to construct computational models. In addition, GrADS reuses distance data on small inputs to predict the fraction of cache hits and misses on the given data and cache configuration by its memory-hierarchy performance models, as mentioned in Table 1.

Table 1: Scheduler model and architecture

Schedular Model	Scheduler Architecture	Example System
Centralized (single resource)		Unix, Linux, Window OS
Centralized (multiple resources and single/multiple domains)		Cluster system: PBS, LSF, SGE, Condor
Decentralized (hierarchical and multiple domains)		Multi-clusters/Grid System, AppLeS, SGE-E Nimrod-G, Condor-G, Gridbus Broker
Decentralized (self coordinated or job pool & multiple domains)		Cooperative Clusters (exchange workload) P2P systems (no exchange of jobs)

IV. SUMMARY OF SCHEDULING SYSTEM

Table 2 shown comparative study of scheduling system base on five aspects is below:

Table 2: Comparative of scheduling system

AppLeS	
Organization	Hierarchical
Scheduling	Decentralized scheduler, predictive state estimation, online rescheduling, fixed application oriented policy (system-centric)
Function/Apply	Application specific scheduler, weather service, tomography
Application	Diverse
Resources	Single-domain Grid, non-dedicated, time shared
Condor-G	
Organization	Cooperative/Centralized Scheduler
Scheduling	Flat
Function/Apply	A wide area job processing system
Application	Single-job
Resources	Multi-domain Grid, non-dedicated, non time-shared
Nimrod/G	
Organization	Hierarchical Cells
Scheduling	Decentralized scheduler, predictive pricing models, event driven rescheduling, fixed application oriented scheduling privacy
Function/Apply	Economic-base Grid resource broker for parameter sweep/task framing application, bioinformatics, ecological modeling
Application	Soft real-time parametric study
Resources	Multi-domain Grid, non-dedicated, time-shared
GrADS	
Organization	Hierarchical
Scheduling	Execution on heterogeneous
Function/Apply	Scientific
Application	Diverse
Resources	Multi-domain Grid, non-dedicated, time-shared

V. CONCLUSION

Many scheduling technique has been discussed and implement on this area. In table 2, list of some technique have been studied for scheduling system and how they implement the technique in simulation or real Grid tasted. Many of research have been done by using simulations over than test in real grid.

REFERENCES

- [1] Foster. I, and Kesselman. C. (eds). The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, 1999.
- [2] L. Zhang, J. Chung, and Q. Zhou, "Developing Grid Computing Applications, <http://www.106.ibm.com/developerworks/grid/library/gr-grid1/>, Oct. 2002 (Discover Grid computing, developerWorks Journal, (February 2003), 14.19) "
- [3] Globus Alliance, Press Releases, c/o Carl Kesselman, USC/Information Sciences Institute, 4676 Admiralty Way, Suite 1001, <http://www.globus.org>
- [4] I. Foster, What is the Grid? A Three Point Checklist, GRID Today, 1(2002). <http://www.gridtoday.com/02/0722/100136.html>
- [5] A. L. Pereira, V. Muppavarapu, and S. M. Chung, "Role-Based Access Control for Grid Database Services Using the Community Authorization Service," IEEE Trans. on Dependable and Secure Computing, Vol. 3, No. 2, 2006, pp. 156-166.
- [6] Frey J., Tannenbaum T., I. Foster, M. Livny, and S. Tuecke. "Condor-G: A computation management agent for multi-institutional grids." In Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC), pages 7{9, San Francisco, California, August 2001.
- [7] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. International Journal of Supercomputer Applications, 11(2):115{128, 1997.
- [8] R. Buyya, D. Abramson, and J. Giddy, A Case for Economy Grid Architecture for Service-Oriented Grid Computing, Proceedings of the International Parallel and Distributed Processing Symposium: 10th IEEE International Heterogeneous Computing Workshop (HCW 2001), April 23, 2001, San Francisco, California, USA, IEEE CS Press, USA, 2001.
- [9] Condor, <http://www.cs.wisc.edu/condor/publications.html>. 2008.
- [10] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. International Journal of Supercomputer Applications, 11(2):115{128, 1997.
- [11] Condor, <http://www.cs.wisc.edu/condor/publications.html>. 2008.
- [12] F. Berman and R. Wolski, The AppLeS Project: A Status Report, Proceedings of the 8th NEC Research Symposium, Berlin, Germany, May 1997.
- [13] H. Dail, G. Obertelli, F. Berman, R. Wolski, and Andrew Grimshaw, Application-Aware Scheduling of a Magnetohydrodynamics Application in the Legion Metasystem, Proceedings of the 9th Heterogeneous Computing Workshop, May 2000.
- [14] S. Smallen, W. Cirne, J. Frey, F. Berman, R. Wolski, M. Su, C. Kesselman, S. Young, and M. Ellisman, Combining Workstations and Supercomputers to Support Grid Applications: The Parallel Tomography Experience, Proceedings of the 9th Heterogeneous Computing Workshop, May 2000.
- [15] H. Casanova, A. Legrand, D. Zagorodnov, and F. Berman, "Heuristics for Scheduling Parameter Sweep applications in Grid Environments", Proceedings of the Heterogeneous Computing Workshop, May 2000.
- [16] Website <http://www.csse.monash.edu.au/~nimrod/nimrod/ng.html>
- [17] F. Berman, A. Chien, K. Cooper, J. Dongarra, I. Foster, D. Gannon, L. Johnsson, K. Kennedy, C. Kesselman, J. Mellor-Crummey, D. Reed, L. Torczon, and R. Wolski. The GrADS Project: Software Support for High-Level Grid Application Development. International Journal of High Performance Computing Applications (JHPCA),15(4):327-344, SAGE Publications Inc., London, UK, Winter 2001.
- [18] K. Cooper, A. Dasgupta, Kennedy, C. Koelbel, A. Mandal, G. Marin, M. Mazina, J. Mellor-Crummey, F. Berman, H. Casanova, A. Chien, H. Dail, X. Liu, A. Olugbile, O. Sievert, H. Xia, L. Johnsson, B. Liu, M. Patel, D. Reed, W. Deng, C. Mendes, Z. Shi, A. YarKhan, J. Dongarra. New Grid Scheduling and Rescheduling Methods in the GrADS Project. Proceedings of the NSF Next Generation Software Workshop, International Parallel and Distributed Processing Symposium, Santa Fe, IEEE CS Press, Los Alamitos, CA, USA, April 2004.
- [19] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen, and R. Freund. Dynamic Matching and Scheduling of a Class of Independent Tasks onto Heterogeneous Computing Systems. Proceedings of the 8th Heterogeneous Computing Workshop (HCW'99), Juan, Puerto Rico, IEEE Computer Society, Los Alamitos, April 12, 1999.



Mohd Kamir Yusof obtained her Master of Computer Science from Faculty of Computer Science and Information System, Universiti Teknologi Malaysia in 2008. Currently, he is a Lecturer at Department of Computer Science, Faculty of Infomatics, Universiti Darul Iman Malaysia (UDM), Terengganu, Malaysia.



Muhamad Azahar Stapa obtained her Master of Computer Science from Faculty of Computer Science and Information System, Universiti Teknologi Malaysia in 2008. Currently, he is a teacher a SMK Teknik Kangar, Perlis, Malaysia.