Presented at Workshop of Big Data and HPC: Ecosystem and Convergence, Russian Supercomputing Days, September 2017

Iran's National Grid Initiative: Objectives, Challenges, and Opportunities

Majid Dashtbani[1], Pejman Lotfi-Kamran[1,2] Dara Rahmati[2], Saeid Gorgin[3,2,1], and Hamid Sarbazi-Azad[4,2]

[1] Grid Computing Group, Institute for Research in Fundamental Sciences (IPM)
[2] School of Computer Science, Institute for Research in Fundamental Sciences (IPM)
[3] Department of Electrical Engineering and Information Technology, Iranian Research Organization for Science and Technology (IROST)
[4] Department of Computer Engineering, Sharif University of Technology

Abstract. Nowadays, almost every organization has an IT department to offer services essential for the core business of the organization. These services include Web serving, data storage, business analytics, etc. Data centers are the backbone of such services. As demand on the data centers varies over time, organizations need to significantly overprovision the capacity of their data centers to handle worst-case scenarios. Grid computing is a several decades old idea to connect computing resources of various organizations and enable them to provision the resources based on the average usage instead of worst case.

In this paper, we introduce Iran's national grid initiative to connect computing resources of universities and offer a unified access point for all computing resources of several universities. The students can use the whole computing resources regardless of where they study. The power of computing resources varies vastly across different universities: while some universities have more computational power than they actually need, other universities do not even have the necessary equipment to serve their students with the computational power needed to do their homework. Moreover, even if universities have the necessary hardware, they usually lack the software stack that enables them to efficiently manage the hardware and offer a seamless service to the students. In this paper, we talk about a comprehensive effort to unify the computational power of several universities across the country along with an easy-to-use interface for the students to have the computation they want without the need to know where the computation is being carried out. We go over the challenges of having such a system and the solutions that we come up with to address the challenges.

1 Introduction

In the age of information technology, almost all organizations have IT departments, which are responsible to offer services that are essential to the core business of the organizations. Web servers, data storage, business analytics, machine

learning are examples of such services. Date centers [1,2] are the computing platforms that run and offer such services. Data centers, or warehouse-scale computers, are large-scale compute platforms that consist of thousands of servers. As data centers are important to organizations, there have been many efforts to optimize data centers [3–8] or their components [9–15].

Due to load variation (i.e., over time) on data centers [3, 16], organizations need to significantly overprovision the capacity of their data centers to handle worst-case scenarios [10, 11]. The overprovisioning of the capacity of the data centers not only increases the initial capital outlay to build a data center, but also increases the cost of operating a data center due to increase in power and maintenance expenditures.

One way to reduce the overprovisioning problem in data centers is grid computing. Grid computing is an old idea to connect computing resources of various organizations and offer a unified and seamless service. As an organization can use the computing resources of other organizations through the grid system, the organization does not need to overprovision the computing resources for the worst case. The resources can be provisioned for the average case. When the required resources exceed what are available, the needed extra resources can be acquired through the grid system from the computing resources of other organizations. As many organizations are connected to the grid system, it is unlikely that all of them utilize all of their resources at the same time.

In this paper, we introduce Iran's national grid initiative. The goal of this initiative is to connect computing resources of several universities together to form a grid system and offer a unified access point for the whole computing resources to the students regardless of where they study. The power of computing resources varies vastly across different universities. Some universities have more computational power than they need, but many universities do not have the necessary computational power to serve their students properly. The grid system can distribute computational resources fairly to the students: the students receive the necessary computational power regardless of where they study.

Moreover, even if universities have the necessary computational hardware, they usually lack the software stack, which enables them to efficiently manage the hardware and offer a seamless service to the students. In this paper, we talk about a comprehensive effort to unify the computational power of several universities across the country through a grid system. Not only the grid system unifies the computational resources, but also offers an easy-to-use interface for the students to have the computation they want without the need to know where the computation is being carried out. We go over the challenges of having such a system and the solutions that we come up with to address the challenges.

2 Motivation

University students are one of the major consumer of computational power in Iran. Iran has the fastest growing rates of the number of publications in the past decade [17–19]. As many publications require computer simulations/evaluations

to verify the claim, there is a growing need for computational power in Iranian universities. Not only students need computational power for publication, but also, they need it to do their homework.

Unfortunately, computational resources are not distributed to universities based on their needs (e.g., proportional to the number of students in the university). Every university, independently, attempted to acquire the computational resources that their students may need. Depending on many parameters, including the wealth and status of the university, some universities succeeded and have lots of computational resources, sometimes beyond what they currently need, while others failed to get the necessary computational power. While the unbalanced distribution of computation resources exists among the universities even in a city, the problem exacerbates as the universities become further away from the capital.

While there are lots of computational resources across the country, many students who need computational resources do not have an easy way to access what they need. This is mostly because computational resources are located in many geographically-distant and administratively-independent locations. To address this problem, we envision a system that connects these independent islands of computational resources together and form a grid of computational resources.

With the grid system, the independent islands of computational resources should appear as a single system to the end user. A user should submit its job through a unified interface. Upon receiving the job, the system itself, without any user intervention, should determine where to execute the job. Upon completion of the job, the results will be sent to the user through the unified interface. In this paper, we talk about the challenges of implementing such a system. We also illustrate some of the solutions that we come up with to address these challenges.

3 The Grid System

To goal is to connect computing resources of universities (and other organization) to form a grid computational system. The grid system should offer a unified interface for end users (mostly university students) to submit their jobs to the grid. The users should not need to be worried about where the computation is carried out. When the computation ends, the results along with an invoice will be sent to the user.

We assume that universities and other organizations have a cluster of compute nodes. Moreover, they also need to have a head node through which one can submit jobs to the compute cluster. The grid system should connect several universities and organizations with the mentioned compute cluster. The grid, as a whole, should appear as a single system. The end user should not be concerned about the structure of the grid when they submit jobs or collect the results.

When a user wants to submit a job to the grid, they go to the unified interface. The interface should get the job along with the number of cores that the user wants the job to run on. Based on the type of the job and the number of requested

cores, the grid system should pick a suitable computational resource to execute the job on. While the job is running, the user needs to be able to see the partial results of their job. When the job finishes, the grid sends the whole result back to the end user.

4 Challenges of Designing a Grid System

There are several challenges associated with building a grid system. As universities are autonomous organizations, they are not willing to share their computing resources through the grid system, even if the utilization of their resources is low.

To give incentive to universities to join the national grid system, we decided to charge users based on how much time they used the computing resources that are connected to the national grid. A significant fraction of the collected finance will be distributed to universities based on how much time their computational resources were busy running jobs submitted by the grid.

Another challenge of building such a system is unwillingness of universities to give the grid full control to the computational resources. Not only universities are unwilling to give full control of their computational resources to the grid, but also many of them do not allow the grid to run even a single control program on the compute cluster, mainly due to management issues.

While the grid requires a mechanism to accurately measure usage of the computational resources to charge end users accordingly, universities are unwilling to let the grid run control programs on the compute nodes. We need to come up with a plan with minimal access requirement to perform all grids functionalities.

We need to build a grid system on top of autonomous computer resources of various universities (and other organizations) with minimal access requirement. Organizations, due to various reasons, might not give the grid access to their entire computational cluster. The minimum requirement is administrative control to a single compute node and the ability to submit jobs to the whole compute cluster with user access control. Organizations does not need to give the grid full access to their compute cluster, and the grid should be able to function properly with the minimal access rights.

Not only with the minimal access rights, the grid needs to perform all of its functionalities, i.e., getting the job from a user, dispatching the job to free computational resources without user intervention, billing, resource-usage monitoring, etc.), but also, the grid needs to make sure that organizations offer to the users what they promised to offer. As users submit their jobs through the unified interface offered by the grid and do not know where the job is actually running, any dishonesty by one of the organizations connected to the grid is seen by end users as dishonesty by the grid.

It is the responsibility of the grid to make sure that when a user asks for a core, they actually receive a physical core and not a virtual core that runs with many other virtual cores on a single physical core. As we do not have direct access to the whole compute clusters of universities and organizations,

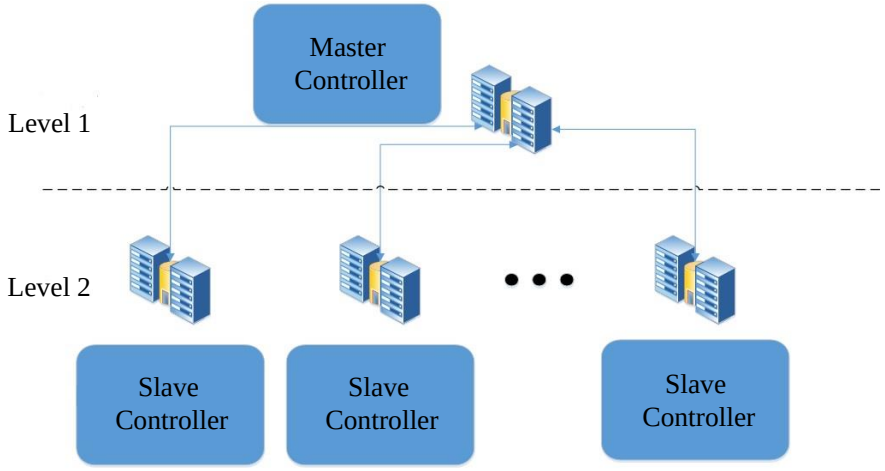

Fig. 1. The logical structure of Iran's national grid.

guaranteeing the service-level agreement is tricky. In this paper, we discuss what we plan to do to perform this job.

5 Iran's National Grid

In this section, we introduce the design of Iran's national grid and the rationale behind the choices made in this design. We also talk about the ways that we address the challenges introduced in the prior section.

Iran's national grid is a tree with two levels as shown in Figure 1. The first level (or the root) is a single logical node responsible for receiving, distributing, and load-balancing the user requests. Moreover, the first level is responsible for billing and monitoring of the requests. There are many nodes in the second level. Each node represents the compute cluster of a university or an organization. There is no restriction on the number of nodes in the second level.

As there are many nodes on the second level, at this level, the design by itself is resilient against failure of any single node. If for any reason, computational resources of an organization become inaccessible (e.g., network problem, power outage, etc.), the grid can use the compute resources of other organizations to run users' requests. However, when it comes to the first level, as there is a single root node, the design by itself does not guarantee resiliency against failure. We note that the single-node root is a logical design. To add resiliency to the grid at the root level, the physical design benefits from redundancy at the compute level and replication for the data at geographically-distant sites.

Unlike the second level, the root node belongs to the grid (and not to one of the organizations connected to the grid). Therefore, the grid has full control over

the root node. We can run any piece of code with any privilege on the root node. However, as discussed, universities and organizations may impose restrictions on what we can run on their cluster. To minimize the requirement at the leaf level (i.e., second level), we just require a single dedicated machine, which we refer to as the machine in the rest of the text, at each organization plus access to the head nodes of their cluster to submit jobs and read the log of their cluster manager (i.e., the piece of software responsible to get a job and distribute it to the cluster). In the head nodes of the cluster, we do not need root access and user access suffices.

The brain of the grid is running on the root node. What is running on the root node can broadly be classified as portal and core. Portal is the user interface between the end user and the grid system. It consists of frontend and backend. Frontend is part of the portal that users can see and interact with. Backend is part of the portal that is not directly accessible by end users, and at the same time, is necessary for the correct functionality of the portal (e.g., database of registered users, authentication code, etc.).

Core is another piece of code running on the root node. Core is responsible for all the functionality of the grid. Core is responsible to get and distribute users' requests, redistribute users' request across leaf nodes for load balancing, accounting and monitoring, etc. As the load on the core increases with the increase in the number of users, to make the core scalable, we decided to push as much of the responsibility of the core as possible to the leaf nodes.

As per our design, there is a single compute node in every leaf node with which the grid has full control. We try to push a significant fraction of the functionality of the core to the leaf node to make the grid scalable. We refer to part of the core that runs on the root node as master controller and part that runs on the leaf nodes as slave controller, as shown in Figure 1.

When the master controller receives a request, its checks the status of the leaf nodes that are capable to run the request. The master controller chooses a leaf node which is not heavily loaded. The master controller sends a request to slave controller of the chosen leaf node to notify it about the request. Upon receiving the request, the slave controller is responsible to copy the data related to the user request from the root node to the leaf node. Then the slave controller submits the request to the head node of the cluster to be executed. The slave controller checks the status of all running jobs in the cluster. Upon completion of a job, the slave controller notifies the master controller. The master controller, consequently, notifies the end user.

The slave controller is not only responsible to copy the request, dispatch it, and send the response back to the master controller, but also is responsible to make sure that the organization is offering what is supposed to give to the users. The grid expects that organizations and universities give physical cores (non-virtualized cores) to the users. Moreover, users specify the number of cores when they submit their job to the grid. The grid expects that the cores be dedicated to them (not to be shared with several users). The slave controller is also responsible to make sure that such criteria are met.

The grid does not have full control over the organizations' clusters. As such, we cannot continuously run a monitoring program on the cluster to control the resource allocation. Therefore, we decided to run the monitoring programs similar to user jobs. The slave controller, which runs on the leaf node's machine, occasionally sends a short monitoring job to the head node of the cluster to be executed on one of the compute nodes. While the monitoring program is running, it makes sure that the cores are physical cores (as opposed to virtualized cores) and are dedicated to the grid's jobs.

Due to the fact that we have to pay the organizations for using their compute resources, we particularly designed the monitoring program to be executed in a short period of time. As we run monitoring program occasionally (as opposed to continuously) and for a short period of time, the monitoring cannot always detect the violation of service-level agreement. Nevertheless, the statistical detection of the violation of the service-level agreement combined with the penalty of such a violation prevent organizations from deliberately violating the agreed-upon service-level agreement.

6 Related Work

Computational grids employ the distributed computing resources dynamically. These resources include processing clusters, storage systems, supercomputers, etc. Buyya provides a comprehensive definition for the grid concept [20]. This definition clearly describes a wide range of applicability for the idea of grid computing: "Grid is a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed 'autonomous' resources dynamically at runtime depending on their availability, capability, performance, cost, and users' quality-of-service requirements" [20].

There are many examples of national-level grid networks worldwide, which initiated basically to meet challenges or exhibit new ideas for computing purposes. Some of them are the World-Wide Grid (WWG) [21], UK eScience Grid [22], Tera Grid [23], and the World-wide LHC Computing Grid (WLCG) [24]. WWG is initiated by the Global Data-Intensive Grid Collaboration, which exhibits the use of many data-intensive grid applications. The UK eScience Grid is a grid network to be used by the UK academies. Tera Grid is a project for American researchers, which started in 2001 and performed from 2004 through 2011. The WLCG project is a global teamwork performed in 42 countries aggregating worldwide grid structures in more than 170 computing centers.

There are, also, a large variety of corporate grid activities or projects, which basically provide different core services suitable to set up computing grid infrastructures. Examples include Aneka [25], which enables a .Net based grid and cloud computing platform. The Cosm P2P Toolkit [26] is a library of open source applications or protocols to enable the worldwide computers to collaborate for distributed computing. This includes GRID, sensor-net, or Cloud applications. The Globus [27, 28] project is a well-known data management service to be used for research services. This project recently announced the availability of its ser-

vice for Google Drive. This mechanism enables the use of a concrete logical interface for manipulating, sharing or data transfer in all accessible storage systems. The Legion project [29] demonstrates a worldwide virtual computer. It is basically an object-based software environment that ties millions of processing cores and users connected by high-speed links. A user of the system has the illusion of using just a single computer, while in reality, a variety of physical resources and different kinds of data are used. These include video stream, cameras, processing accelerators, simulators, etc. The ProActive parallel suite [30] is an open source project used for dynamic management of heterogeneous grids and clouds. The main focus of the project is on application parallelization tightly coupled with seamless enterprise scheduling and arrangement.

The grid economy project [31] provides an economic paradigm for grid computing for the purpose of enabling a service-oriented computing grid. There is also a large body of work and projects not covered in this paper (e.g., [32–35]). We refer interested readers to the more comprehensive surveys of grid computing [36–38].

In this paper, we try to provide a feasible and customized implementation of the Iranian national grid initiative, considering what have been done before, combined with the realistic assumptions and computation demands of the grid customers including the researchers in Iran.

7 Conclusion

In this paper, we introduced Iran's national grid initiative. The goal of this initiative is to form a computational grid out of computing resources of universities and offer a unified access point for all computing resources to the students and researchers across the country. We talked about a comprehensive effort to unify the computational power of several universities across the country for the purpose of providing an easy-to-use interface for the students to have the computation they want without the need to know where the computation is being carried out. We went over the challenges of having such a system and the solutions that we came up with to address the challenges.

References

1. Barroso, L.A., Hlzle, U.: The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines. 1st edn. Morgan and Claypool Publishers (2009)
2. Barroso, L.A., Clidaras, J., Hlzle, U.: The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines. 2nd edn. Morgan and Claypool Publishers (2013)
3. Meisner, D., Sadler, C.M., Barroso, L.A., Weber, W.D., Wenisch, T.F.: Power Management of Online Data-intensive Services. In: Proceedings of the 38th Annual International Symposium on Computer Architecture (ISCA). (June 2011) 319–330

4. Shah, M., Ranganathan, P., Chang, J., Tolia, N., Roberts, D., Mudge, T.: Data Dwarfs: Motivating a Coverage Set for Future Large Data Center Workloads. In: Workshop on Architectural Concerns in Large Datacenters. (June 2010)
5. at a Time, G.B.S.E.O.D.C. http://googleblog.blogspot.com/2008/10/saving-electricity-one-data-center-at.html
6. Karidis, J., Moreira, J.E., Moreno, J.: True Value: Assessing and Optimizing the Cost of Computing at the Data Center Level. In: Proceedings of the 6th ACM Conference on Computing Frontiers. (May 2009) 185–192
7. Soundararajan, V., Anderson, J.M.: The Impact of Management Operations on the Virtualized Datacenter. In: Proceedings of the 37th Annual International Symposium on Computer Architecture (ISCA). (June 2010) 326–337
8. Lo, D., Cheng, L., Govindaraju, R., Barroso, L.A., Kozyrakis, C.: Towards Energy Proportionality for Large-scale Latency-critical Workloads. In: Proceeding of the 41st Annual International Symposium on Computer Architecuture (ISCA). (June 2014) 301–312
9. Cavium Networks: Cavium Announces Availability of ThunderX^{TM}: Industry's First 48 Core Family of ARMv8 Workload Optimized Processors for Next Generation Data Center & Cloud Infrastructure. http://www.cavium.com/newsevents-Cavium-Announces-Availability-of-ThunderX.html (December 2014)
10. Grot, B., Hardy, D., Lotfi-Kamran, P., Falsafi, B., Nicopoulos, C., Sazeides, Y.: Optimizing Data-Center TCO with Scale-Out Processors. IEEE Micro **32**(5) (September 2012) 52–63
11. Bilal, K., Khan, S.U., Zomaya, A.Y.: Green Data Center Networks: Challenges and Opportunities. In: Proceedings of the 11th International Conference on Frontiers of Information Technology (FIT). (December 2013) 229–234
12. Ozer, E., Flautner, K., Idgunji, S., Saidi, A., Sazeides, Y., Ahsan, B., Ladas, N., Nicopoulos, C., Sideris, I., Falsafi, B., Adileh, A., Ferdman, M., Lotfi-Kamran, P., Kuulusa, M., Marchal, P., Minas, N.: EuroCloud: Energy-Conscious 3D Server-on-Chip for Green Cloud Services. In: Proceedings of the Workshop on Architectural Concerns in Large Datacenters in conjunction with ISCA. (June 2010)
13. Byan, S., Lentini, J., Madan, A., Pabn, L.: Mercury: Host-side Flash Caching for the Data Center. In: Proceedings of the 28th IEEE Symposium on Mass Storage Systems and Technologies (MSST). (April 2012) 1–12
14. Lotfi-Kamran, P., Grot, B., Ferdman, M., Volos, S., Kocberber, O., Picorel, J., Adileh, A., Jevdjic, D., Idgunji, S., Ozer, E., Falsafi, B.: Scale-Out Processors. In: Proceedings of the 39th Annual International Symposium on Computer Architecture (ISCA). (June 2012) 500–511
15. Malladi, K.T., Lee, B.C., Nothaft, F.A., Kozyrakis, C., Periyathambi, K., Horowitz, M.: Towards Energy-proportional Datacenter Memory with Mobile DRAM. In: Proceedings of the 39th Annual International Symposium on Computer Architecture (ISCA). (June 2012) 37–48
16. Kanev, S., Darago, J.P., Hazelwood, K., Ranganathan, P., Moseley, T., Wei, G.Y., Brooks, D.: Profiling a Warehouse-scale Computer. In: Proceedings of the 42nd Annual International Symposium on Computer Architecture (ISCA). (June 2015) 158–169
17. Tribune, F.: Growth Rate in Science and Technology 16%. https://financialtribune.com/articles/people/51469/growth-rate-in-science-and-technology-16 (October 2016)

18. Akhondzadeh, S.: Iranian science shows world's fastest growth: ranks 17th in science production in 2012. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3732862 (July 2013)
19. MacKenzie, D.: Iran showing fastest scientific growth of any country. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3732862 (February 2010)
20. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I.: Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing As the 5th Utility. Future Generation Computer Systems **25**(6) (June 2009) 599–616
21. Kacsuk, P., Kiss, T.: Towards a scientific workflow-oriented computational World Wide Grid. In: Technical Report TR-0115, CoreGRID, Institute on Grid Systems, Tools and Environments. (December 2007)
22. Hey, T., Trefethen, A.E.: UK e-Science Programme: Next Generation Grid Applications. The International Journal of High-Performance Computing Applications **18**(3) (August 2004) 285–291
23. Simms, S.C., Stewart, C.A., McCaulay, S.D.: Cyberinfrastructure Resources for U.S. Scholarship: The TeraGrid. In: Proceedings of the 36th Annual ACM SIGUCCS Fall Conference: Moving Mountains, Blazing Trails. (October 2008) 341–344
24. Shiers, J.: The Worldwide LHC Computing Grid (Worldwide LCG). Computer Physics Communications **177**(1-2) (July 2007) 219–223
25. Chu, X., Nadiminti, K., Jin, C., Venugopal, S., Buyya, R.: Aneka: Next-Generation Enterprise Grid Platform for e-Science and e-Business Applications. In: Proceedings of the 3rd IEEE International Conference on e-Science and Grid Computing (e-Science 2007). (Dec 2007) 151–159
26. McHugh, J.: Build Your Own Supercomputer. Forbes Magazine (November 1999)
27. Foster, I.: Globus Online: Accelerating and Democratizing Science through Cloud-Based Services. IEEE Internet Computing **15**(3) (May 2011) 70–73
28. Allen, B., Bresnahan, J., Childers, L., Foster, I., Kandaswamy, G., Kettimuthu, R., Kordas, J., Link, M., Martin, S., Pickett, K., Tuecke, S.: Software As a Service for Data Scientists. Communications of the ACM **55**(2) (February 2012) 81–88
29. White, B.S., Walker, M., Humphrey, M., Grimshaw, A.S.: LegionFS: A Secure and Scalable File System Supporting Cross-domain High-performance Applications. In: Proceedings of the ACM/IEEE Conference on Supercomputing (SC). (November 2001) 59–59
30. Caromel, D.: ProActive Parallel Suite: Multi-cores to Clouds to autonomicity. In: Proceedings of the 5th IEEE International Conference on Intelligent Computer Communication and Processing. (August 2009) xi–xii
31. Buyya, R.: The Grid Economy Project. http://www.buyya.com/ecogrid/
32. Buyya, R.: Grid Computing Info Centre (GRID Infoware). http://www.gridcomputing.com/
33. e.V., U.F.: A European Federation Software Suite. https://www.unicore.eu/
34. GOST: Global Operating Systems Technology Group. http://gost.isi.edu/
35. Team, C.: Cactus Code. http://cactuscode.org/
36. Liu, Y., Rong, Z., Jun, C., Ping, C.Y.: Survey of Grid and Grid Computing. In: Proceedings of the International Conference on Internet Technology and Applications. (August 2011) 1–4
37. Ahuja, S.P., Myers, J.R.: A Survey on Wireless Grid Computing. The Journal of Supercomputing **37**(1) (July 2006) 3–21
38. Haque, A., Alhashmi, S.M., Parthiban, R.: A Survey of Economic Models in Grid Computing. Future Generation Computer Systems **27**(8) (October 2011) 1056–1069