Proceedings of the First EELA-2 Conference
R. Mayo et al. (Eds.)
CIEMAT 2009

On the Co-existence of Service and Opportunistic Grids

Francisco Brasileiro, Alexandre Duarte, Rafael Silva, Matheus Gaudêncio

Universidade Federal de Campina Grande
Departamento de Sistemas e Computação
{fubica, alex, rafael, matheusgr}@lsd.ufcg.edu.br

Abstract

Most computational grids currently in production are either service grids or opportunistic grids. While a service grid provides high levels of quality of service, an opportunistic grid provides computing power only on a best-effort basis. Nevertheless, since opportunistic grids do not require resources to be fully dedicated to the grid, they tend to assemble larger numbers of resources. Moreover, these grids cater very well for the execution of the so-called embarrassingly parallel applications, a type of application that is frequently found in practice. In this paper we present an approach for supporting the co-existence of a service grid and an opportunistic grid on the same infrastructure. The advantages of this hybrid infrastructure are twofold: firstly, the co-existence allows idle resources belonging to the service grid to be used in an opportunistic way; secondly, the provision of an opportunistic grid allows shared resources to be added to the infrastructure, a feature that turns out to be very important for consortia in which many of the member institutions cannot afford the provision of dedicated resources. The proposed approach is currently being implemented in the context of the EELA-2 project.

1. Introduction

Throughout this decade, grid computing has established itself as a key technology for the support of an important portion of the scientific activity developed worldwide, a trend that has been dubbed e-Science. As the technology matured, several different types of grid infrastructures have been built and are currently in operation. These, in turn, may be broadly divided in two classes, namely: service grids and opportunistic grids. Service grids normally assemble high performance, dedicated computing resources, such as clusters, supercomputers, and large data storage systems that are spread over a relatively small number of administrative domains. For instance, the largest service grid currently in operation is the EGEE grid 00 which encompasses more than 79; 000 processing cores distributed among approximately 260 resource centres in 53 countries. Service grids provide high and well defined levels of quality of service (QoS). To achieve such levels of QoS, a lot of effort is employed in the monitoring and management of the infrastructure. Moreover, these infrastructures are set up to run

complex distributed applications that, in many cases, require important grid core services to be in operation.

On the other hand, opportunistic grids are somewhat more "lightweight" grid infrastructures that scavenge idle computing cycles from non-dedicated resources. Several types of opportunistic grids have been proposed, implemented and deployed. Among the most important representatives of this class of grids one can list: desktop grids, first proposed by the Condor project [13]0; voluntary computing platforms such as the pioneer SETI@home system 00 and its successor BOINC 0 and, more recently, peer-to-peer grids such as those supported by the OurGrid middleware 0.

In all cases, the functioning of an opportunistic grid is very similar and simple. The owners of the computing resources install a grid agent that is in charge of monitoring the utilization of the resource and detect when it is idle. Upon detecting that the resource is idle, the agent passes this information to a scheduler that is in charge of dispatching tasks to be executed on the idle resource. Whenever a resource is required to run applications spawned by its owner, immediately, any grid-related task that may be running is either suspended or interrupted.

Since the availability of the resources is dependent on the behavioural pattern of their owners, it is normally not effective to ensure any QoS in this type of grid. Thus, they work basically on a best-effort basis. Nevertheless, this relatively unsophisticated strategy has proved to be very effective in assembling enormous amounts of computing cycles for the execution of scientific applications 0. Moreover, *embarrassingly parallel applications*, i.e. those parallel applications that can be divided in a large number of independent tasks that do not communicate with each other, can be very easily scheduled and successfully executed in this kind of infrastructure. This is very reassuring, once embarrassingly parallel applications, also known as *bag-of-tasks* (BoT) applications, are the prevalent type of scientific applications that currently execute on the main grid infrastructures 0.

In the past three years, the Federal University of Campina Grande in Brazil has been deeply involved in an initiative to deploy and operate a service grid infrastructure (or e-infrastructure, for short) gathering resources from research centres in several Latin American and European countries. This effort was initiated in the context of the project "E-infrastructure shared between Europe and Latin America" (EELA) 00. The EELA project was a 2-year project run by 21 institutions from Europe and Latin America under the 6th Framework Programme for Research, Technological Development and Demonstration (FP6) of the European Commission (EC). The objective of the EELA project was to bring the e-infrastructures of Latin American countries to the level of those already available in Europe. For this purpose it built an e-infrastructure for scientific applications benefiting from the mature state of the EC funded project "America Latina Interconectada Con Europa" (ALICE) [1] and of its product, the RedCLARA network - with the ultimate goal of promoting a sustainable framework for e-Science in Latin America.

The EELA approach has been to create a collaboration network that deployed a large portfolio of grid-enabled applications on a robust and well managed pilot e-infrastructure. Although the EELA project has been very successful, during its development it was possible to identify that many institutions, especially in Latin America, were unable to commit dedicated resources to the e-infrastructure. Also, it was possible to identify many partners that could provide a relatively large amount of non-dedicated resources to the e-infrastructure. This motivated us to seek alternatives for allowing the co-existence of two different grid systems over the same e-infrastructure: one that would operate as a service grid and that encompassed only dedicated resources and another that could assemble non-dedicated resources in a best-effort opportunistic grid.

In this paper we present our approach to enable the coexistence of a service grid and an opportunistic peer-to-peer grid on the same e-infrastructure. The research reported in this paper is currently being developed in the context of the project "E-science grid facility for Europe and Latin America" (EELA-2) 0, which aims at transforming the current EELA e-infrastructure into a high capacity, production-quality, scalable e-infrastructure, providing round-the-clock worldwide access to distributed computing, storage and network resources for a wide spectrum of applications from European and Latin American scientific communities. The EELA-2 project is also co-funded by the EC under the 7th Framework Programme (FP7) and involves more than 50 institutions from 9 countries in Latin America and 5 in Europe.

The rest of this paper is organised as follows. In Section 2 we detail the context for the work, explaining the main components of a gLite-based service grid and an OurGrid-based opportunistic grid. Then, in Section 3 we discuss our approach for supporting a hybrid e-infrastructure that allows the co-existence of these two types of grids. Section 4 brings results from simulations that assess the impact of the proposed approach in the amount of time that jobs stay in the queues waiting for the availability of processing nodes. Finally, our concluding remarks and directions for future work are presented in Section 5.

2. Background

2.1. A gLite-based Service Grid

The gLite 0 middleware was born from the collaborative efforts of more than 80 people in 12 different academic and industrial research centres as part of the EGEE Project [11, 4]. A gLite-based service grid is composed by a set of resource centres running services to provide remote access to local dedicated computational resources and central services that form the backbone of the service grid. The gLite grid services can be thematically grouped into 4 groups: Access and Security Services, Information and Monitoring Services, Job Management Services and Data Services. The prime aim of the Access and Security Services is to identify users, allowing or denying access to services, on the basis of agreed policies. It provides credentials having a universal value that works for many purposes across several infrastructures, communities, Virtual Organisations (VOs), and projects. To carry out this task, gLite uses the Public Key Infrastructure (PKI) X.509 technology with Certification Authorities as trusted third parties.

The Information and Monitoring Services provide information about the gLite resources and their status. The published information is used to locate resources and for monitoring and accounting purposes. Much of the data published in the Information Service conforms to a schema that defines a common conceptual data model to be used for resource monitoring and discovery.

The Job Management Services are responsible for dealing with all aspects of the execution of a job on the grid. The Computing Element (CE) service represents a set of computing resources localised at a resource centre (i.e., a cluster, a computing farm, a SMP machine, etc.) and is responsible for the local job management: (submission, control, etc.). A CE provides a generic interface to the cluster where the cluster itself is represented by a collection of Worker Nodes (WN). Another important service in this group is the Workload Management System (WMS). The WMS is a Resource Broker responsible for the distribution and management of jobs across different resource centres. The purpose of the WMS is to accept user jobs, to schedule them to the most appropriate CE according to user's requirements, to record their status, and to retrieve their output. There are other services in

this groups used to collect information about the resource usage done by users or groups of users (VOs).

The Data Services are responsible to manage all aspects related to the location and movement of data among the resource centres of the grid. A service called Storage Element (SE) is a gLite component that provides a common interface to the storage back-end present in the resource centre. It is not uncommon to use an SE to provide remote access to large robots controlling hundreds of terabytes of tape storage. Another data service is the Large File Catalogue (LFC) that keeps metadata information, mapping a common namespace into the location of each file in the SEs present in the grid. Last but not least, there is a File Transfer Service (FTS) that is used to establish optimised transfer channels among two SEs in the Grid.

All the gLite Services can be accessed and used through the User Interface (UI) service. The UI provides a set of command-line tools that allow users to authenticate themselves in the grid, submit jobs and retrieve their output and transfer files to remote grid resource centres.

The grid services described above can also be grouped by their scope. Services like Computing Elements and Storage Elements are services with local scope. They must be deployed by each resource centre to provide remote access to their local resources. Service like the Workload Management System, Large File Catalogue and File Transfer Service are global services that are deployed at some core resource centres of the grid and shared by all users. There are also services like the Access and Security Services and the Information and Monitoring Services that are both local and global services. They need to be deployed at each resource centre and need some global instances to co-ordinate the interaction and propagation of information among local instances.

The lifecycle of a typical gLite job can be summarised by the following steps: i) the job is submitted via the User Interface; ii) the WMS queues the job and starts searching for a suitable Computing Element to execute it; iii) the job is forwarded to the chosen CE and is executed there; iv) after completion, the user can retrieve the job output; v) all the time the job is tracked by logging services, which provide the user with the view of the job state and further details of job processing; vi) after the user retrieves the job output, the middleware data on the job is passed to a job provenance service and purged from their original locations

2.2. An OurGrid-based Peer-to-Peer Opportunistic Grid

An opportunistic peer-to-peer grid supported by the OurGrid middleware has three main components, namely the OurGrid Worker, the OurGrid Resource Manager Peer (or simply the OurGrid Peer) and the OurGrid Broker. Figure 1 depicts the OurGrid architecture.

The OurGrid Worker is an agent that executes on the grid resources (referred as grid machines) and is responsible for implementing a secure environment for the execution of the tasks of grid applications. A policy defined by the resource owner drives the opportunistic behaviour of the OurGrid Worker. It runs an idleness detection algorithm that triggers the availability of the grid machine when the resource is idle (as defined by the local policy), and interrupts any grid task that is executing when, considering the local policy, the resource is deemed not to be idle.

The OurGrid Peer is the component that manages, at each administrative domain (or site), the set of grid machines that are made available to the grid by the corresponding site. In general, one peer per site is installed. A peer joins the grid by notifying a peer discovery service about its existence and it is immediately informed about the presence of other peers on the grid.

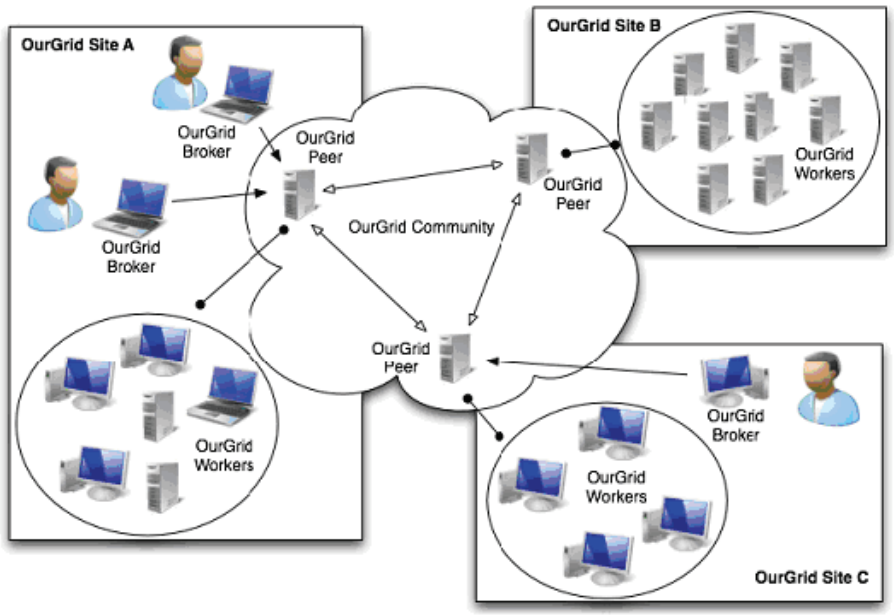

Figure 1. OurGrid's architecture

The OurGrid Broker is responsible for providing users with an interface to submit their applications. It is also responsible for performing the scheduling of the applications on the grid machines and for managing the execution of the scheduled applications. A user wishing to run an application must use the OurGrid Broker to connect to a known peer (usually her own site's peer, named its local peer), becoming a local user of that peer.

Parallel applications run on OurGrid are structured as a set of independent tasks (a BoT). There are several ways to interact with the OurGrid Broker to submit applications. However, no matter how the user interacts with the broker, when an application is submitted for execution, the broker contacts its local peer asking for the number of grid machines required to run the application (this request may carry specific attributes for the machines, such as a particular operating system, or a minimal amount of memory, etc). When the peer receives such a request it tries to find enough machines that are suitable to execute the application's tasks. If the peer cannot satisfy the request with its own local machines, it tries to obtain machines from other community peers, by forwarding the request to remote peers that may have suitable machines. It then waits for these peers to deliver remote machines. Whenever new machines (local or remote) are made available, the local peer delivers these machines to the requesting broker. As soon as one or more machines are delivered to the broker, it schedules tasks and starts the management of their execution.

OurGrid has been built to be fast, simple, scalable and secure. It is fast as it allows users to improve the turn-around time of their applications. This capability is provided by features of the OurGrid Broker, such as scheduling with task replication and file transfer optimizations. The system's simplicity is mainly due to its capacity of hiding the grid heterogeneity behind the OurGrid middleware. In order to achieve scalability, a peer-to-peer approach has been employed and the Network of Favours, a technology that promotes cooperation among peers, has been designed and implemented.

On OurGrid, security is delivered to users by means of the sandboxing mechanism implemented by the OurGrid Worker. It isolates a potentially malicious application inside a virtual machine, thus protecting the machine and the network from attacks. A more detailed discussion on the OurGrid approach to each one of these issues can be found in [9].

3. The Hybrid Grid Infrastructure

In this section we present our approach for supporting the co-existence of a service grid and an opportunistic grid on the same infrastructure. The main objectives are to allow the usage of the non-dedicated resources provided by an OurGrid-based opportunistic grid to execute jobs submitted to the gLite-based service grid and to allow the usage of idle computing cycles from the gLite-based service grid to execute jobs submitted to the OurGrid-based opportunistic grid.

These objectives are orthogonal to each other and can be achieved independently. We start by describing how the idle resources in the service grid are exposed to the opportunistic grid and jobs submitted to the opportunistic grid are able to scavenge idle computing cycles from the service grid. Later we describe how OurGrid peers are integrated into the service grid and how jobs submitted to the service grid are able to exploit the shared resources in an opportunistic way.

3.1. Exposing Dedicated Resources to the Opportunistic Grid

Exposing the service grid dedicated resources managed by a particular gLite CE to the opportunistic grid requires simply the installation of an OurGrid Peer and OurGrid Workers in every node managed by the CE, as showed in Figure 2. In this way the same set of resources would be visible both by the gLite-based service grid and by the OurGrid-based opportunistic grid. However, it is important to define how to prioritise the execution of jobs coming from the service grid. The OurGrid Workers should be allowed to run in the cluster only in the nodes that are not being used to execute jobs from the service grid (gray circles in Figure 2).

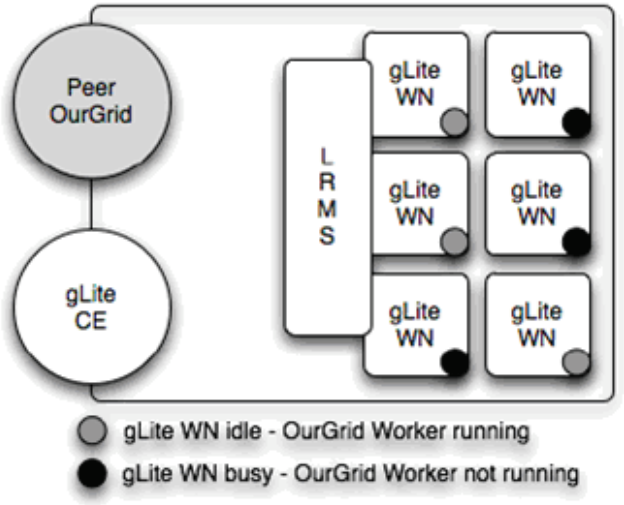

Figure 2. Scavenging Idle Cycles from the Service Grid

In gLite-based service grids the CE is the service responsible for exposing local clusters in the grid. However, the CE has no power to control how the resources in the cluster are actually used. Instead, the CE delegates this power to the Local Resource Manager System (LRMS) and directs the remote jobs received trough the grid to the LRMS queues. So, in order to implement this prioritisation policy we need to modify the way the LRMS manages the cluster resources. The solution is based on the utilisation of the hooks provided by most LRMSs to execute some commands before and after the execution of the actual job. Prior to

the execution of a job in the cluster the LRMS executes a command to stop the OurGrid Worker running in the selected node, then it executes the job on the node and after the execution is concluded and the node is idle again the LRMS starts an OurGrid Worker again in the node. So, the execution of OurGrid Workers does not tag the cluster nodes as busy and whenever a job is scheduled to run in a given node the OurGrid Worker running there is interrupted to be restarted only after the cluster jobs are finished.

With this scheme, whenever a node in the cluster is idle it will be running an OurGrid Worker and will be ready to run jobs coming from the opportunistic grid. On the other hand, whenever a node in the cluster is needed to execute a job coming from the service grid it will be readily available as well.

3.2. Integrating OurGrid Peers into a Service Grid

As described in Section 2.2, the OurGrid middleware is composed by three main components: the OurGrid Worker, the OurGrid Peer and the OurGrid Broker. These components have analogous counterparts on the services provided by the gLite middleware, i.e. the gLite Worker Node, the gLite CE and the gLite Workload Management System. As these two middleware were developed targeting different types of resources, shared on the case of OurGrid and dedicated in the case of gLite, they deal with resources in different ways. The OurGrid Peer has been designed to manage a number of shared resources and provide access to each of them individually through the OurGrid Workers.

The gLite CE was designed to provide a remote access point to clusters of resources, directing remote jobs coming from the grid into the cluster's LRMS queues. Thus, a first step to allow the integration of OurGrid sites into the service grid is to provide an equivalent functionality using the OurGrid middleware. We decided to develop a new kind of OurGrid Worker called Cluster Worker that will direct the jobs that are submitted to it to a cluster's LRMS. This new component is shown in Figure 3.

The second step is to allow the use of resources from an OurGrid-based grid to execute jobs submitted to the gLite-based service grid. As the OurGrid Peer is now able to deal with the same kind of resources managed by the gLite CEs, the easiest way to integrate OurGrid peers into a gLite-based service grid is to expose them as if they were normal gLite CEs. We have created a new OurGrid component called OurGrid CE, as showed in Figure 3. This new service provides an additional CE-like interface to the OurGrid Peer and allows for the execution of jobs coming both from the opportunistic grid and from the service grid in the same set of resources managed by the OurGrid Peer. These resources can be either a dedicated cluster using the Cluster Worker or a set of desktops in a department/laboratory.

The OurGrid CE publishes its status information in the gLite Information Service to allow service grid users to discover the new resources available in the e-infrastructure. By interacting with the LRMS we defined two priority levels to jobs according to the way they arrive in the resources. Jobs submitted to the OurGrid Worker without entering the LRMS queues are using the resources in an opportunistic way and thus can be interrupted at any time (gray circles in Figure 3). Jobs submitted to the Cluster Worker (CW) and scheduled by the LRMS are not pre-emptable and behave like jobs submitted locally to the cluster, like in a service grid (black circles in Figure 3). These jobs can even interrupt jobs running in an opportunistic way to use more resources.

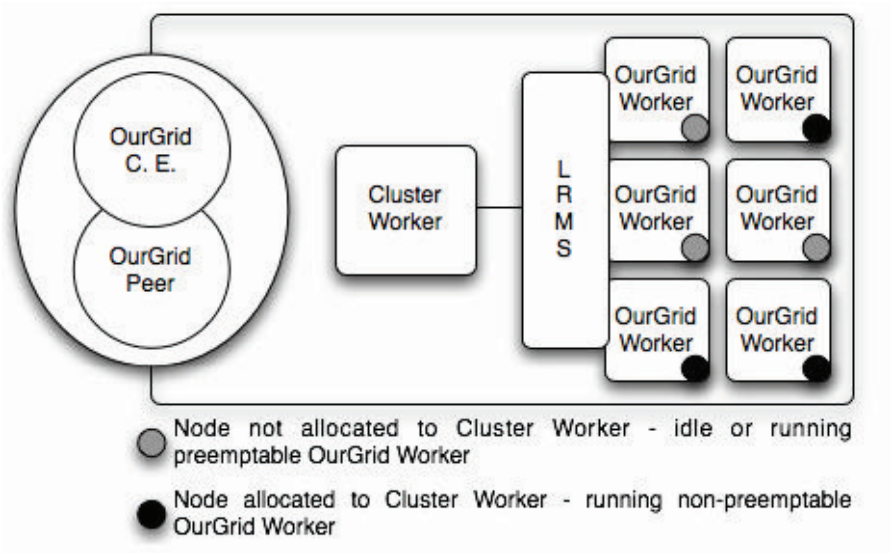

Figure 3. Cluster Worker

4. Simulations

In order to assess the impact of having BoT jobs being executed in the opportunistic grid and tightly-coupled applications in the service grid, we have developed a discrete-event simulator and have fed it with data from real service grid infrastructures 0. Our experiments used the DAS-2 workload from the Grid Workload Archives as input 0. We have simulated the scheduling of the jobs using four well-known algorithms: FCFS (First Come First Served), CBF (Conservative Backfilling), PRE, which is a meta-scheduler that is aware of the hybrid grid and schedule BoT jobs in the opportunistic grid and tightly-coupled applications in the service grid (using CBF), and PRECH, which is similar to PRE, only that the e-infrastructure checkpoints the BoT tasks that are pre-empted in the service grid (remember that the desktop grid is opportunistic). We used the grid size and the number of desktops in the opportunistic grid as variables of the simulations. We evaluated the average time the jobs remain in queue (*WaitTime*) for the two categories of jobs: those that are BoT (*NProc* = 1) and those who are tightly-coupled applications (*NProc* > 1).

The DAS2 workload consists of jobs that require at most 128 processors. We evaluated the *WaitTime* for one site of 128 nodes and one site of 256 nodes as shown in Figure 4.

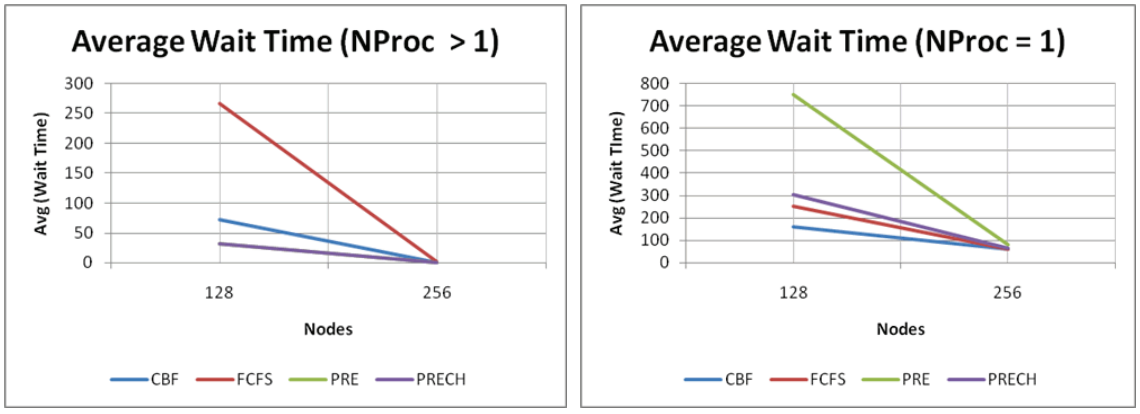

Figure 4. Average Wait Time for an e-infrastructure with the resources of a service grid

As can be seen, for *NProc* > 1 and a grid with 128 nodes, both PRECH and PRE[1] yield waiting times that are smaller than CBF and FCFS. This comes at the expense of a substantial increase on the wait time of BoT jobs. On the other hand, a grid with 256 nodes can be considered over provisioned for our workload: the wait time of all four algorithms converge to similar values in both job categories.

Next we evaluate a hybrid grid. We defined that our hybrid grid is composed of one service grid with one site of 128 interconnected nodes, and an opportunistic grid with a variable number of desktops. We chose values that represents 0, 1/16, 1/8, 1/4, 1/2 and 1 time the service grid size (in terms of number of processors), i.e. scenarios with 0, 8, 16, 32, 64 and 128 desktops in our opportunistic grid.

Grid schedulers currently used in service grids do not know how to take advantage of opportunistic grids. So, the existence of the opportunistic grid does not affect the average *WaitTime* any further from the value presented in Figure 4, where there is a cluster with 128 nodes and no desktops. On the other hand, both PRE and PRECH can take advantage of the desktops of the opportunistic grid by scheduling BoT jobs (*NProc* = 1) on these machines. Figure 5 shows the behaviour of the schedulers in such scenario.

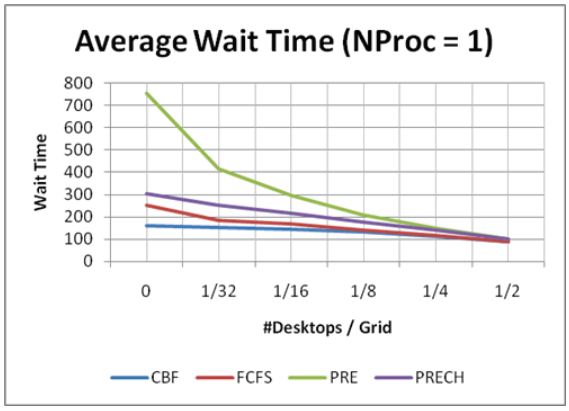

Figure 5. Average wait time for a hybrid grid

Overall, the addition of desktops machines to perform the work decreases the wait time of jobs that require a single computing node to execute. With this approach, the PRE algorithm substantially increases its performance. By aggregating an opportunistic grid whose processing power is equivalent to 25% of the power of the service grid, it is possible to decrease the wait time by a factor of five. From this point on, the wait time value tend to converge; henceforth, the addition of more desktop machines will not decrease significantly the wait time.

5. Concluding Remarks

In this paper we presented an approach for supporting the co-existence of a service grid and an opportunistic grid on the same e-infrastructure. The advantages of this hybrid infrastructure are twofold: firstly, the co-existence allows idle resources belonging to the service grid to be used in an opportunistic way; secondly, the provision of an opportunistic

[1] In this case, the curve for PRE and PRECH are the same.

grid allows shared resources to be added to the infrastructure, a feature that turns out to be very important for consortia in which many of the member institutions can not afford the provision of dedicated resources. The proposed approach is currently being implemented in the context of the EELA-2 project (http://www.eu-eela.eu/), an initiative funded by the European Commission within its Seventh Framework Programme and involving more than 50 institutions both in Latin America and Europe.

The first version of the services proposed in this paper will be available by early 2009 and the final version, with corrections from possible problems detected during evaluation and performance improvements, will be available by January 2010. See http://www.ourgrid.org/ for an up-to-date status of this work.

Acknowledgments

Authors would like to thank the European Commission for the co-funding of both the EELA and the EELA-2 projects. Francisco Brasileiro would like to thank the financial support from CNPq/Brazil (grant 300.646/96).

References

[1] America Latina Interconectada con Europa. http://alice.dante.net/.

[2] E-infrastructure shared between Europe and Latin America. http://www.eu-eela.org/first-phase.php.

[3] E-science grid facility for Europe and Latin America. http://www.eu-eela.eu.

[4] Enabling Grids for e-Science web site. http://www.euegee.org.

[5] gLite: Lightweight middleware for grid computing. http://cern.ch/glite.

[6] SETI@home Statistics Page. http://setiathome.ssl.berkeley.edu/totals.html.

[7] SETI@home web site. http://setiathome.ssl.berkeley.edu/.

[8] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer. SETI@home: an experiment in public resource computing. Comm. ACM, 45(11):56–61, Nov. 2002.

[9] W. Cirne, F. Brasileiro, N. Andrade, L. Costa, A. Andrade, R. Novaes, and M. Mowbray. Labs of the world, unite!!! Journal of Grid Computing, 4(3):225–246, 2006.

[10] D. H. J. Epema, M. Livny, R. van Dantzig, X. Evers, and J. Pruyne. A worldwide flock of condors: load sharing among workstation clusters. Future Gener. Comput. Syst., 12(1):53–65, 1996.

[11] F. Gagliardi, B. Jones, F. Grey, M.-E. Begin, , and M. Heikkurinen. Building an infrastructure for scientific grid computing: status and goals of the EGEE project. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 363(1833):1729–1742, 2005.

[12] A. Iosup, H. Li, M. Jan, S. Anoep, C. Dumitrescu, L.Wolters, and D. H. J. Epema. The grid workloads archive. Future Gener. Comput. Syst., 24(7):672–686, 2008.

[13] M. Litzkow, M. Livny, and M. Mutka. Condor - a hunter of idle workstations. In Proc. 8th Int'l Conf. Distributed Computing Systems, 1988.

[14] B. Marechal, P. H. R. Bello, and D. Carvalho. Building a grid in Latin America: The EELA project e-infrastructure. In CCGRID '07: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid, pages 835–839, Washington, DC, USA, 2007. IEEE Computer Society.

[15] University of California. Berkeley Open Infrastructure for Network Computing. http://boinc.berkeley.edu/, 2004.