SCHEDULING STRATEGIES IN FEDERATED GRIDS

Katia Leal
Universidad Rey Juan Carlos
Dep. de Sistemas Telemáticos y Computación
Escuela Sup. de Ciencias Experimentales y Tec.
Tulipán SN, Mósteles, Madrid, Spain
Email: katia.leal@urjc.es

Eduardo Huedo, Rubén S. Montero, Ignacio M. Llorente
Univ. Complutense de Madrid
Dep. Arquitec. de Comp. y Automática
Facultad de Informática 28040, Spain
Email: contact@dsa-research.org

KEYWORDS

Federated Grids, Scheduling

ABSTRACT

The GridWay Metascheduler enables efficient sharing of computing resources managed by different LRM (Local Resource Management) systems, not only within a single organization, but also across several administrative domains. The possibility to access to the services available in different Globus based grids allows the union of grids to create a federation. This scenario has particular characteristics, possibly the most important one is that it has different types of users: internal users, external users, and direct users. Basically, all these users compete for the resources of the federated grid to achieve their own particular goals. However, GridWay is not providing the best scheduling strategy under this scenario, because its current scheduling policy does not take into account resource ownership. In this paper we introduce a variation of GridWay's current scheduling strategy suitable to this new scenario. This variation is based on the parameters provided by a previously proposed performance model. In addition, the results obtained by simulation lead us to conclude that there is a real necessity to enhance scheduling policies in federated grids.

INTRODUCTION

A Federated Grid can be formed of several grid infrastructures. However, the participants in the Federated Grid do not collaborate to achieve the same goal, like the participants of a Global Grid. Here the idea is that each participant shares resources with the rest, but always having in mind that the main user of those resources is the participant itself. GridWay (Huedo et al. (2004)) provides the technology to build Federated Grids, both directly and through *GridGateWays*. A GridGateWay is a WS-GRAM (Web Services Grid Resource Allocation and Management) service hosting a GridWay workload manager that enables remote access to GridWay's metascheduling capabilities through a WSRF (Web Services Resource Framework) interface. However, GridWay applies scheduling strategies that are better fitted to Partner and Enterprise Grids. There is a huge ongoing research effort on grid scheduling (Dong and Akl (2006); Andrieux et al. (2003)), but it is mainly centered on Partner Grids. With this paper we want to drive attention to the particular characteristics of Federated Grids, and in the necessity of new scheduling policies to support them. Thus, we propose an alternative to GridWay's current scheduling policy based on a performance model (Montero et al. (2006)) that allows to parametrize and compare different Grids.

The rest of the paper is structured as follows: we first present and compare other scheduling approaches with our solution. Then, we explain the mapping strategy used by our scheduling proposal to maximize the throughput. Next, we present the design and implementation of the scheduling model. We also show some experimental results. Finally, we explain the conclusions and future work.

RELATED WORK

It is well known that the general scheduling problem is NP-complete (Ullman (1975)). A large number of algorithms have been applied to schedule jobs in computational grids. However, none of them seems suitable to federated grids. Here we enumerate some scheduling algorithms and the drawbacks they present under this scenario.

The Opportunistic Load Balancing (OLB), the Minimum Completion Time (MCT), the Min-min, and the Max-min are similar algorithms and also have very similar drawbacks under a federated grid. The first is a problem of scalability: the time to calculate the selected node increases with the number of nodes. The second problem is that in a federated grid nodes are different, so we should not simply assign a job to the next available node.

The main disadvantage of the Weighted metascheduling (Song et al. (2005)), and of the QoS guided Min-Min (He et al. (2003)) algorithms is that the former is specific of data intensive applications, and the latter focuses in long-term applications.

More close to our problem is the work of (Wiriyaprasit and Muangsin (2004)) that analyzes the impact of local policies on the performance of grid scheduling on a computational grid. However, their simulated scenario has

Proceedings of the 2008 High Performance
Computing & Simulation Conference ©ECMS
Waleed W. Smari (Ed.)
ISBN: 978-0-9553018-7-2 / ISBN: 978-0-9553018-6-5 (CD)

certain drawbacks: it is not based in a real testbed, it can only be applied to computational grids, and it doesn't reflect the effects of including the algorithm in a real scheduler, like GridWay. In contrast, we show experimental results obtained from a simulated federated grid based on a real testbed, and using GridWay as the scheduler.

A NEW SCHEDULING MODEL TO MAXIMIZE THE THROUGHPUT

In this section we first analyze the environment conditions of Federated Grids, and then we introduce our proposal of a new scheduling model.

Federated Grid

In a Federated Grid the different participants collaborate by sharing their resources with the whole Grid. However, they do not try to achieve the same goals as they have to satisfy their own users demands. In doing so, each participant can use his own internal resources, but also the grid resources that the rest of participant are willing to share. Each participant decides which resources to contribute with, who can use those resources, and the access policy to them. Of course, all these restrictions can change dynamically, and brokers should be prepared for that. A possible Federated Grid schema could be like the one shown in the Figure 1. As it can be seen, there are different types of resources and users. We have identified two types of resources:

❶ *Internal Resources*: these are the resources directly accessible by the broker through the corresponding local workload manager. That is, the resources owned by the particular research center, laboratory or company.

❷ *External Resources*: we can classify Enterprise Grids, Partner Grids, and Utility Grids provided by third part companies in this category.

Also, we have identified internal, external, and direct users. They differ in the way, and in the rights they have to access resources:

❶ *Internal Users*: the jobs submitted by these users through GridWay can be executed in both the internal and the external resources. Depending on different parameters, such as the local load, GridWay will decide to which resource submit the job.

❷ *External Users*: all the jobs received by GridWay through the GRAM interface will be from external users. GridWay will apply different policies to decide whether to accept or not the jobs received.

❸ *Direct Users*: GridWay cannot control the jobs submitted by this type of internal users. However, they are important since they have an influence in the load of the resources.

As a result, each type of user introduces its own requirements that will affect GridWay scheduling policies.

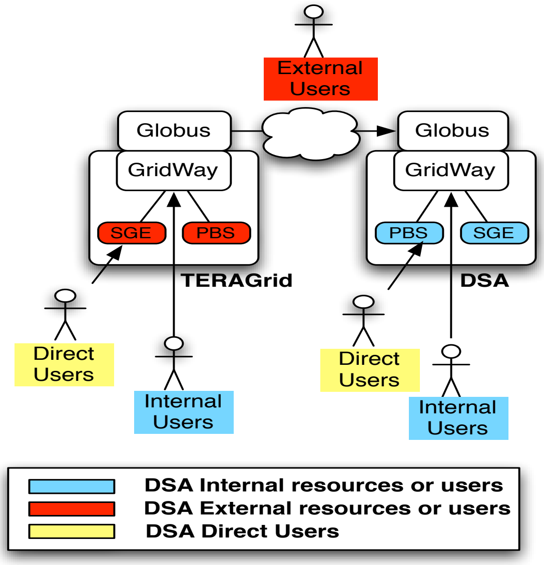

Figure 1: Example of a Federated Grid.

The mapping strategy

GridWay receives jobs directly through the command line interface from internal users, and through the GRAM interface from external users. In this way, GridWay can differentiate the jobs submitted by internal users from those submitted from external users. However, GridWay currently operates in the same way, and applies the same policies for both internal and external jobs.

Next, we explain the modifications already included in our simulated GridWay to support the restrictions introduced by the different type of users on a federated grid. Our simulated GridWay will work in different ways depending on the type of job received.

In scheduling an internal job

Our simulated GridWay almost includes all the configurable restrictions of a real GridWay: i.e. the maximum number of jobs that will be dispatched at each scheduling action and the period (in seconds) between two scheduling actions. In scheduling an internal job, the normal version of the simulated GridWay firstly checks if there are free nodes available in internal resources. If there are free internal nodes, GridWay schedules the job to an internal resource. In contrast, if there are no free internal nodes but free external ones, GridWay schedules the job to an external resource. However, we want to improve the normal scheduling policy to maximize the number of jobs that can be executed while maintaining makespan value. Thus, the scheduling policy should take into account not only which is the next available node. To maximize the throughput we need to obtain the number of jobs that should be submitted to internal resources and to external resources. We have used the equation that represents the best characterization of the Grid to obtain these numbers. The characterization can be obtained if we take the line that represents the average behavior of the system, as proposed by Hockney, and Jesshope (Hockney and Jesshope (1988)):

$$n(t) = r_{\infty}t - n_{1/2} \qquad (1)$$

In the Equation 1 *n* represents the number of completed tasks as a function of time *t*. The other parameters are:

- **Asymptotic performance** r_{∞}: is the maximum rate of performance in tasks executed per second. In the case of an homogeneous array of *N* processors with an execution time per task *T*, we have r_{∞}= *N/T*.
- **Half-performance length** $n_{1/2}$: is the number of tasks required to obtain the half of the asymptotic performance. This parameter is also a measure of the amount of parallelism in the system as seen by the application.

The linear relation represented by Equation 1 can be used to define the performance of the system (tasks completed per second). We explain later how we can also use this linear equation to obtain the number of jobs that should be submitted to internal resources and to external resources to maximize the throughput.

In scheduling an external job

As soon as GridWay receives an external job, it has to decide whether to accept it or not. By default, the simulated GridWay accepts all the external jobs received from external users.

When GridWay has to schedule an external job, it employs a different strategy than when scheduling an internal job. In the case of an external job, GridWay applies it's current scheduling policies. That is, it schedules the external job to the next internal resource with free nodes. In this way we avoid the situations on which a participant of the Federated Grid can receive from another one a job previously submitted to it.

DESIGN AND IMPLEMENTATION

We have previously (Vázquez et al. (2008, 2007)) set up a simple, but real infrastructure, where a client runs an instance of the GridWay Metascheduler interfacing internal resources in an enterprise grid, the DSA (Distributed System Architecture) enterprise grid at Complutense of Madrid University, based on Globus Toolkit 4 (GT4) WS interfaces, and a GridGateWay that gives access to resources from a partner grid (*fusion* VO of the EGEE), based on GT pre-WS interfaces found on gLite 3.0. However, prior to run our enhanced scheduling algorithm on a real production infrastructure, we have first implemented the modified algorithm on a simulated environment. The deployment on a real environment will require involvement of a large number of active users and resources, which is very hard to coordinate and build. Thus, the simulation appears to be the easiest way to analyze the modified scheduling policy. Based on the simulation results, we can later encourage or discourage the deployment on a real production environment.

We have used the well known GridSim toolkit (`http://www.gridbus.org/gridsim/`) to simulate our test scenario.

Test Scenario

Since the idea is to finally deploy the new GridWay on a real infrastructure, the simulation results have to be as realistic as possible. Thus, the simulated scenario has to be close enough to reality. We will start with the simple, but more or less realistic scenario depicted in Figure 2.

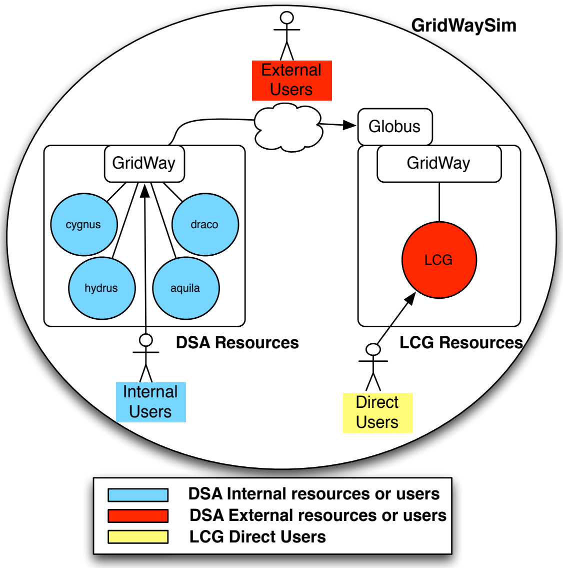

Figure 2: A simple test scenario.

As it can be seen, in this test scenario there are only two grid resources: the DSA (Distributed System Architecture) and the LCG (LHC Computing Grid). The DSA testbed represents the resources of the Distributed System Architecture research group at the Complutense of Madrid University. In the same way, the LCG testbed represents the Large Hadron Collider (LHC) Computing Grid. From the point of view of a *DSA internal user*, the DSA GridWay is her broker, the DSA resources are internal resources, and the LCG resources are external resources. In the same way, all the jobs received by the LCG GridWay through the Globus GRAM interface are from external users. While the GridWay on the DSA site has to apply a policy to submit jobs to internal and/or external resources, the LCG GridWay has to decide whether or not to accept the jobs from external users.

Table 1 shows the number of computing elements, aka PEs (Processing Elements), and MIPS (Millions Instructions Per Second) of each machine in the DSA infrastructure. The Table 2 shows the same values for the machines in the LCG resource. We have calculated the MIPS value based on machine's model, and number of MHz. We use all these characteristics to simulate resources in order to obtain as realistic as possible results.

Machine	PEs	MIPS/PE
hydrus	4	9787
aquila	5	9787
orion	1	9787
cygnus	2	6536
draco	1	6536

Table 1: Characteristics of the machines in the DSA research testbed.

Machine	PEs	MIPS/PE
machine0	800	9787
machine1	640	6536
machine2	560	4902

Table 2: Characteristics of the machines in the LCG research testbed.

GRIDWAYSIM ENTITIES

We have called *GridWaySim* to our simulation of the scenario shown in Figure 2. We explain the different participating entities of GridWaySim in the next sections.

GridWaySim

This entity represents the whole simulation, and is responsible of the creation of the main simulated entities: GridWay brokers, users, DSA and LCG resources, and workload (or LCG direct users).

GridWay

The *GridWay* entity represents a generic GridWay metascheduler. Since we need to interconnect the DSA, and LCG grids to form a federation, we have to instantiate two GridWay brokers: one for DSA, and the other for the LCG. Thus, from the point of view of the DSA GridWay, DSA resources are internal resources, and LCG is an external resource. On the other hand, for the LCG GridWay, DSA is an external resource, and LCG is an internal resource. For this first test scenario, the flow of jobs is only from DSA GridWay to LCG GridWay. However, communication can be done in both directions. Also, the DSA GridWay only receives experiments (a collection of jobs) from her internal users, and the LCG GridWay only receives jobs from DSA GridWay. Finally, to simulate a real environment, we have also introduced direct users in the LCG resource by means of the *Workload* entity.

Testbed: DSATestbed, LCGTestbed

A *Testbed* represents a generic set of grid resources. The resources of the DSA research group are represented y the *DSATestbed* entity, and the LCG ones by the *LCGTestbed* entity. Each follows the configurations depicted in Tables 1 and 2, respectively. The main difference between these two entities is that the LCG resources are represented by an unique resource, while DSA ones are represented as they really are, that is, by five resources. We have instantiated all the resources to use the spaced-shared policy as the internal management. As defined in the GridSim API, this policy uses the First Come First Serve (FCFS) algorithm.

User

The *User* models an user that submits experiments to a GridWay broker. We use this entity to represent internal as well as external users. The functionality of each user includes the submission of experiments to the correspondent broker, and waiting for it completion.

Experiment

An *Experiment* is a collection of jobs. We use this entity to recover important information about the experiment (such as the start and end times), and of all its jobs.

Job

The *Job* entity represents a generic job submitted to the grid. This entity provides specific information about each job: start time, end time, and CPU time among others. We can represent jobs of different computation times, and with different input and output file sizes.

Workload: The LCG Grid log

Since the main purpose of our simulation is to create a realistic environment, we have used the *Workload* entity in our tests. The Workload entity submits jobs by reading resource traces from a file. Thus, our jobs are competing with the jobs submitted by the Workload entity. For this reason, the LCG grid resources might not be available at certain times. The file follows the standard workload format as specified in `http://www.cs.huji.ac.il/labs/parallel/workload/`. As trace file, we have used the LCG Grid Log that contains 11 days of real activity from multiple nodes that make up the LCG (Large Hadron Collider Computing Grid, `http://lcg.web.cern.ch/LCG/`). Next, we enumerate some details about this testbed:

❒ **Number of jobs submitted**: 188,041. The log specifies the submit time and the run time of each job.

❒ **Start time**: Sun Nov 20 00:00:05 GMT 2005.

❒ **End time**: Mon Dec 05 10:30:24 GMT 2005.

❒ **Maximum number of machines**: 170.

- **Maximum number of computing elements**: 24,515.

Although the number of PEs of the real LCG testbed is 24,515, we do not know the real number of PEs involved in this experiment. So, after running some simulations, we decided to reduce the number of PEs in our simulated LCG testbed to those in the Table 2. We have reduced the number of PEs in order to force LCG saturation scenarios.

EXPERIMENTS

We have implemented two versions of GridWaySim that only differ in the scheduling policy they implement: the normal, and the enhanced scheduling policy. As a result, the scheduling policy is the only factor that can cause throughput variations between the different GridWaySim versions. Apart form that, all versions rely on the same configuration, with the same number of users that submit at the same time the same experiment with the same number of jobs (each with the same length, and input and output files size) to the same broker. Also, the number of brokers and resources is the same across the different GridWaySim versions.

Next we describe the exact configuration of the simulation:

- **Entities**: when we start the simulation GridWaySim creates 11 Users, 2 GridWay brokers, 1 DSATestbed, 1 LCGTestbed, and 1 Workload. Each of which is an independent thread attending petitions in their `body()` method.

- **Experiment**: every Experiment is a collection of 300 equal Jobs.

- **Job**: the main parameters of each Job are the length or size (in Million of Instructions, MI) of the Job to be executed, the input files (in bytes), and the output files (also in bytes) to be submitted to the corresponding resource. All Jobs have the same values for the three parameters: the size is 6,000,000 MI, the input file size is 1,000,000 bytes, and the output file size is 2,000,000 bytes.

- **User**: when we create the User, we have to indicate a submit time for her Experiment. Each user only submits one Experiment. In this simulation, each User submits her Experiment 24 hours after the previous one. The first User submits her Experiment at 12:00 of the first day of the simulation. Thus, each User submits her experiment to the DSA GridWay at 12:00 of the corresponding day of simulation.

- **Workload**: the Workload entity submits 188,041 jobs to the LCGTestbed at the time specified in the trace file.

- **DSATestbed**: simulates the resources described in Table 1. All the DSA resources uses the spaced-shared policy as the internal management policy (as defined in the GridSim API, this policy uses the First Come First Serve (FCFS) algorithm).

- **LCGTestbed**: simulates the resource with the machines described in Table 2. The LCG resource also uses the spaced-shared policy as the internal management policy.

As we mentioned before, to maximize the throughput we need to obtain the number of jobs that the DSA GridWay should submit to internal resources and to partner resources. We have used the equation that represents the best characterization of the Grid to obtain these numbers (Montero et al. (2006)). Thus, we need to run GridWaySim to obtain the linear equations of each infrastructure. Figure 3 shows throughput achieved by using the normal scheduling policy in DSA, LCG, and Federated Grid infrastructures for User-0 and User-3. It can be also seen the linear equations of both, DSA and LCG infrastructures (as function of time). We can represent the tasks executed by DSA, and LCG as follows:

$$t^{DSA}(x) = m^{DSA}x + b^{DSA} \tag{2}$$

$$t^{LCG}(N-x) = m^{LCG}(N-x) + b^{LCG} \tag{3}$$

The *minimum* number of tasks that should execute DSA infrastructure is the point of intersection of these two lines. To determine this point we have to equal the linear Equations 2 and 3, which are functions of the completed tasks, and work out the values of m and b from Equation 1, which is function of time,

$$min = \frac{r_\infty^{DSA} n_{1/2}^{LCG} - n_{1/2}^{DSA} r_\infty^{LCG}}{r_\infty^{DSA} + r_\infty^{LCG}} + \frac{r_\infty^{DSA}}{r_\infty^{DSA} + r_\infty^{LCG}} N \tag{4}$$

Being N the total number of jobs (300), in the Equation 4 the min represents the maximum number of tasks that should be executed in the DSA infrastructure without increasing the makespan. Consequently, $N - x$ is the number of tasks that should be executed in the LCG infrastructure. Since there are only 2 participants in our proposed test scenario, the *minimum* method is enough to calculate the number of tasks to be executed in each infrastructure. However, in case of having 2 or more participants, we can determine the number of tasks to be executed in each participant by using the *aggregation* or *federation* model proposed in Vázquez et al. (2008).

Table 3 summarizes the number of executed and estimated tasks of User-0 and User-3. As mentioned before, the simulation creates 11 Users each one submitting 1 Experiment with 300 Jobs to the DSA GridWay. Instead

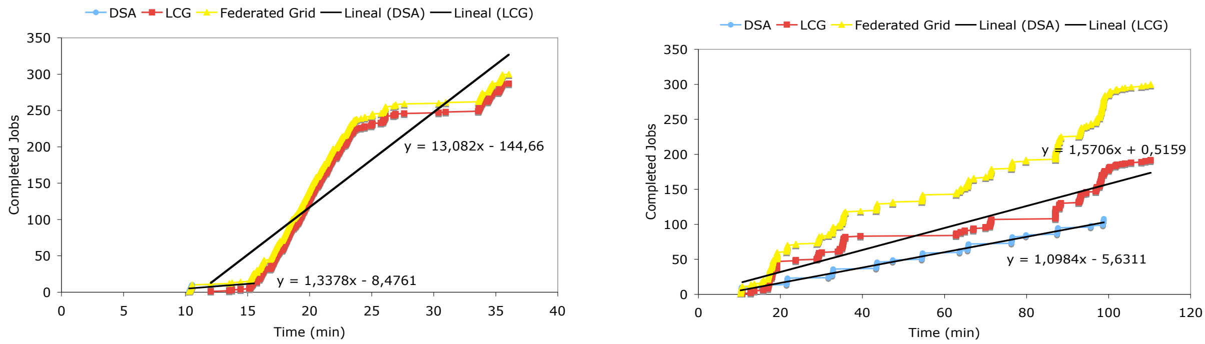

Figure 3: Throughput achieved by using the normal scheduling policy in DSA, LCG, and Federated Grid infrastructures for User-0 (left) and User-3 (right).

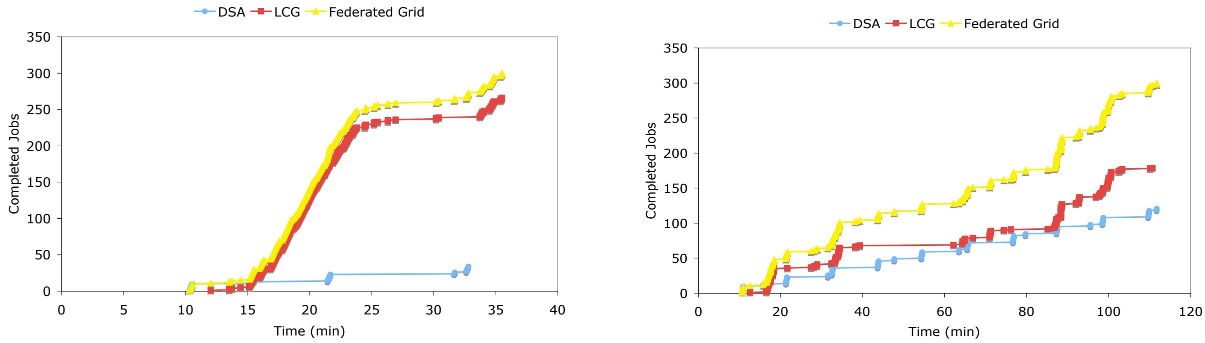

Figure 4: Throughput achieved by using the enhanced scheduling policy in DSA, LCG, and Federated Grid infrastructures for User-0 (left) and User-3 (right).

of providing the results of every User, we concentrate in two of them that represent different LCG saturation scenarios. Thus, simulation results show that User-0 represents an ideal scenario in which the LCG infrastructure always presents free PEs: *low saturation scenario*. As it can be seen in the column *Normal - DSA* of Table 3 the normal scheduling policy submits only 13 of 300 jobs to the DSA infrastructure. The *medium saturation scenario* is the one suffered by User-3, in this case the LCG resource has less free PEs, therefore 108 of 300 jobs are executed in the DSA infrastructure.

	Normal DSA – LCG	Estimated DSA – LCG
User-0	13 – 287	34 – 266
User-3	108 – 192	121 – 179

Table 3: Summary of the number of executed and of estimated jobs in both resources.

Column *Estimated* of Table 3 summarized the number of tasks that should be submitted to both infrastructures to increase the throughput, as depicted in Figure 4. Since we have changed the scheduling goal of GridWay, but not how GridWay achieves it, the completion time of the Experiments of every User obtained in the normal as well as in the enhanced scheduling simulation were the same, as you can see in Table 4. Thus, the estimation enhances GridWay normal scheduling policy: it maximizes DSA throughput while maintaining the makespan. Moreover, the enhanced algorithm not only maximizes the throughput of the DSA infrastructure under the same conditions, it also provides a fairness distribution of jobs between both resources compared with the normal policy: instead of abusing of the external grids, the DSA GridWay submits more jobs to its internal resources.

	Normal Makespan (min.)	Enhanced Makespan (min.)
User-0	36.05	35.52
User-3	110.27	111.75

Table 4: Experiment completion time for User-0 and User-3 in the normal and enhanced scheduling simulation.

CONCLUSIONS AND FUTURE WORK

In this paper we have presented two variations of GridWay's current scheduling policy that adapt to Federated Grids. Thus, the new scheduling policies has been built having in mind restrictions, such as the different types of users, and resources. We have also included the GridWaySim simulated environment to demonstrate that our enhanced scheduling strategy maximizes the throughput of internal resoruces, but without increasing the computational time, and provides a fairness distribution of the jobs by means of the r_∞, and $r_{1/2}$ parameters. Finally, the simulation results provided by GridWaySim show that the enhanced scheduling policy proposed improves GridWay's normal one.

Our current work focuses on the implementation of a scheduling policy that dynamically works out the values r_∞ and $n_{1/2}$. Also, we are adding more entities to our testbed to simulate more complex scenarios.

ACKNOWLEDGE

This research was supported by Consejería de Educación de la Comunidad de Madrid, Fondo Europeo de Desarrollo Regional (FEDER) and Fondo Social Europeo (FSE), through BIOGRIDNET Research Program S-0505/TIC/000101, and by Ministerio de Educación y Ciencia and through the research grant TIN2006-02806.

Research work also supported by the Madrid Regional Research Council under project TIC-000285-0505.

REFERENCES

Andrieux, A., Berry, D., Garibaldi, J., Jarvis, S., MacLaren, J., Ouelhadj, D., and Snelling, D. (2003). "Open Issues in Grid Scheduling". Technical Report ISSN 1751-5971, UK e-Science Institute.

Dong, F. and Akl, S. G. (2006). "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems". Technical Report 2006-504, Ontario Queens University.

He, X., Sun, X., and von Laszewski, G. (2003). "QoS guided min-min heuristic for grid task scheduling". *Journal of Computer Science and Technology*, 18(4):442–451.

Hockney, R. and Jesshope, C. (1988). *"Parallel Computers 2: Architecture, Programming, and Algorithms"*. Adam Hilger Ltd.

Huedo, E., Montero, R. S., and Llorente, I. M. (2004). "A Framework for Adaptive Execution on Grids". *Software – Practice and Experience*, 34(7):631–651.

Montero, R. S., Huedo, E., and Llorente, I. M. (2006). "Benchmarking of High Throughput Computing Applications on Grids". *Parallel Computing*, 32(4):267–269.

Song, J., Koh, C.-K., See, S., and Leng, G. K. (2005). "Performance Investigation of Weighted Meta-scheduling Algorithm for Scientific Grid". In *Proceedings of the 4th International Conference on Grid and Cooperative Computing(GCC 2005)*, volume 3795, pages 1021–1030. LNCS.

Ullman, J. D. (1975). "NP-Complete Scheduling Problems". *Journal of Computer and System Sciences*, 10(3):384–393.

Vázquez, C., Fontán, J., Huedo, E., Montero, R. S., and Llorente, I. M. (2008). "A Performance Model for Federated Grid Infrastructures". In *Proceedings of the 16th Euromicro International Conference on Parallel, Distributed and network-based Processing (PDP 2008)*, pages 188–192.

Vázquez, C., Huedo, E., Montero, R. S., and Llorente, I. M. (2007). "Evaluation of A Utility Computing Mode based on Federation of Grid Infrastructures". In *13th International Euro-Par Conference (Euro-Par 2007)*. Lecture Notes in Computer Science (LNCS).

Wiriyaprasit, S. and Muangsin, V. (2004). "The Impact of Local Priority Policies on Grid Scheduling Performance and an Adaptive Policy-based Grid Scheduling Algorithm". In *Proceedings of the Seventh International Conference on High Performance Computing and Grid in Asia Pacific Region (HPCAsia04)*, volume 0-7695-2138-X/04. IEEE.