Modelling Pilot-Job Applications on Production Grids

Tristan Glatard and Sorina Camarasu-Pop

University of Lyon, CNRS, INSERM, CREATIS, 69621 Villeurbanne, France

Abstract. Pilot-job systems have emerged as a computation paradigm to cope with heterogeneity of production grids, greatly improving fault ratios and latency. Tools like DIANE, WISDOM-II, ToPoS and Condor glideIns are now being widely adopted to conduct large-scale experiments on such platforms. However, a model of pilot-job applications is still lacking, making it difficult to determine submission parameters such as the number of pilots to submit to achieve a given performance level. The variability of production conditions and the heterogeneity of the underlying middleware and infrastructure further complicates this issue. This paper presents a performance model for pilot-job applications running on production grids. Based on a probabilistic modelling, we derive statistics about the number of available pilots along time and the makespan of the application given the number of submitted pilots. Results obtained on a radiotherapy application running on the EGEE production grid show that the model is accurate enough to correctly describe the behavior of the application, setting the basis for further optimization strategies.

1 Introduction

Large-scale production grids such as EGEE[1] are now used by a variety of applications benefiting from computing power and storage space provided by federations of computing centers. Shared by thousands of users, those infrastructures have become complex systems, providing heterogeneity, high variability, low reliability and high latencies as a downside of the huge amount of resources. On EGEE, dozen of minutes of latency with similar standard-deviations and fault ratios of 20% are a common toll to access some of the 80,000 provided CPUs.

Applications relying on such grids had to adopt pragmatic solutions to handle heterogeneity. Among these, pilot jobs provide a submission scheme where tasks are no longer pushed through the grid scheduler but are put in a master pool and pulled by pilots running on computing nodes. Although pilots are still submitted through the regular grid middleware, such a pull model nicely adapts to heterogeneity (pilots running on faster resources pull more tasks than the others), reduces faults and improves latency.

Several pilot-job frameworks have been developed. In particular, systems interfaced with EGEE include DIANE [11], WISDOM-II [1,7], ToPoS[2], BOINC

[1] `http://eu-egee.org/`

[2] `https://wiki.nbic.nl/index.php/ToPoS`

H.X. Lin et al. (Eds): Euro-Par 2009 Workshops, LNCS 6043, pp. 140–149, 2010.

tasks [9] and gPTM3D in radiology [3]. Condor[3] also has its pilot-job submission framework, coined glideIns [15]. However, models are lacking to describe, analyze and optimize the performance of applications relying on pilot-job systems on production grids [8]. This paper proposes such a model and evaluates it in real conditions. Following previous works [5,6], we adopt a probabilistic approach relying on the latency distribution to derive statistics about metrics of interest. The model is detailed in section 2 and evaluated using a radiotherapy application in section 3. Results and capabilities of the model are discussed in section 4.

2 Definitions and Modelling

2.1 Pilot Jobs

Production grids such as EGEE are operated as super-batch systems in which jobs are submitted to a grid scheduler which determines to which computing center they have to be sent. Computing centers then implement their own batch queue to schedule jobs on the computing nodes. By default, applications split the workload into jobs before submitting (i.e. pushing) them to the grid scheduler. Although quite simple, this has disadvantages in terms of fault tolerance and job turn-over. Failed jobs have to be resubmitted to the grid scheduler and go through all middleware components before reaching again a computing node. Besides, heterogeneity can hardly be coped with since the workload is split before the actual computing nodes are known to the application.

In the pilot-job model, the workload is divided in *tasks* by the application and submitted to a *master* managing a task pool on the application host. In parallel, generic grid *jobs* are submitted to the grid scheduler. Once they reach a computing node, those pilots (also called agents or workers) keep on fetching tasks from the master until they all successfully complete. With such a late task-to-node binding, heterogeneity is naturally handled since pilots running on fast resources fetch more tasks than the others. Moreover, failures have a limited impact because failed tasks can directly be executed by other pilots (faulty pilots are removed). Finally, latency is reduced since tasks are directly scheduled by the master on the pilots without going through the whole grid middleware.

2.2 Notations and Assumptions

Given a number of submitted pilots, the modelling aims at estimating (i) the evolution of the number of available pilots along time, i.e., the number of pilots that have reached a computing node and (ii) the makespan of the application, i.e., the duration between the submission of the first pilot and the completion of the last task.

In the following, L denotes the latency (i.e. the total duration between job submission and the beginning of its execution), N the number of available pilots, n the total number of pilots submitted at $t = 0$ and w_0 the total amount of work

[3] `http://www.cs.wisc.edu/condor/`

to be performed by the application. w_0 is expressed as a work time and includes both computing and data transfers. Probabilistic density functions (*pdf*) are denoted with f and cumulative ones (*cdf*) with F. Capitals are random variables and lowercases are fixed values.

The following assumptions are made. First, pilots are assumed to be submitted at $t = 0$, which assumes that they belong to a specific user and that he/she only executes a single application at a time. Moreover, pilots are also assumed generic, i.e., a given pilot can execute any task of the application.

Besides, failed jobs are considered to have an infinite latency, which is adapted to model pilots that are submitted to the grid but never connect back to the master. Those faults occur with ratio ρ. To improve readability, ρ will be omitted in the following of this section: equations taking faults into account can be derived by replacing F_L by $(1 - \rho)F_L$ everywhere. Although this simple model probably does not hold on platforms such as the ones mentioned in [14,2], it is here supported by evidence found in [10] (Fig. 9), based on an analysis of the trace of 33 millions of EGEE jobs submitted between 2005 and 2007. Results show that using such a $(1 - \rho)F_L$ fault model instead of the actual fault distribution only has a small impact on the considered parameter estimation.

The modelling also assumes that pilots face independent and identically distributed (*iid*) latencies. In particular, no saturation of the grid scheduler or queues during the application run is considered. This is realistic as long as we consider applications of reasonable size with respect to the grid infrastructure (i.e. the application itself does not cause system saturation, which makes sense on production grids given their large scale) and that no significant burst period occurs during the run. Handling bursts is a problem currently under study and exploiting works such as [12] may further improve our modelling. Another bottleneck potentially breaking this *iid* assumption is the job submission process. Though jobs can usually be submitted at once (e.g., using bag-of-tasks or DAG submission facilities), applications often still use sequential submission for scalability reasons. How the modelling handles this particular case is described in section 2.5. Another potential limit of the *iid* assumption is the evolution of grid conditions while the application is running. In particular, time, day of week and month may have an impact on the latency. To cope with that, experimental data such as the one reported in [4] could be exploited to adapt the model along time.

2.3 Number of Available Pilots

At instant t, the probability to have k pilots available out of n submitted is:

$$f_N(k,t) = \binom{n}{k} F_L(t)^k (1 - F_L(t))^{(n-k)}$$

where $\binom{n}{k} = \frac{n!}{k!(n-k)!}$. Indeed, any k pilots among n may be available at instant t (i.e. $L \leq t$ for k pilots, which occurs with probability $F_L(t)^k$), while the other $(n - k)$ are still being scheduled or queued ($L \geq t$ for $n - k$ pilots, probability $(1 - F_L(t))^{n-k}$). We can notice that at a given instant N follows a binomial distribution with parameter $F_L(t)$. Thus its expectation and variance are:

$$E_N(t) = nF_L(t) \tag{1}$$
$$\sigma_N(t)^2 = n(1 - F_L(t))F_L(t) \tag{2}$$

From equation 1 we have $\lim_{t\to\infty} E_N(t) = n(1-\rho)$, i.e., after enough time, the user will manage to control all resources he/she submits to, up to the failure ratio. This remains realistic as long as the number of submitted pilots is lower than the number of available computing resources, which is reasonable for large-scale grids such as EGEE that gathers some 80,000 CPUs.

2.4 Makespan

We assume here that the total amount of work (CPU time and data transfers) of an application can be split in any number of tasks, and that linear speed-up w.r.t the number of available pilots is achieved. The latter considers that an application running on a given set of pilots would behave as if all the pilots were achieving the average performance. Although asymptotically realistic, this may lead to some inaccuracies in transient phases. To cope with that, the model could be enhanced by including distributions of host performance. However, to our knowledge, no such model is available on EGEE and building it may be challenging since it is very likely to depend on the application.

Let us denote $W(t)$ the remaining work time at time t ($W(0) = w_0$). Since linear speed-up w.r.t the number of pilots is assumed, we have:

$$dW = -N(t)dt$$

so that the remaining amount of work at a given instant is:

$$W(t) = \max(w_0 - \int_0^t N(u)du, 0)$$

and given equation 1:

$$E_W(t) = \max(w_0 - n\int_0^t F_L(u)du, 0)$$

The expectation $E_M(n)$ of the makespan $M(n)$ of an application is then obtained by solving the following in t:

$$w_0 - n\int_0^t F_L(u)du = 0, \quad \text{i.e. :}$$

$$\int_a^{E_M(n)} F_L(u)du = \frac{w_0}{n} \tag{3}$$

where a is the lower bound of the support of F_L (a is the largest value for which $F_L(a) = 0$). In case L is a fixed value (L=a), then $F_L(t) = 1$ for every $t > a$ so that $E_M(n) = a + \frac{w_0}{n}$. We also have the following limit:

$$\lim_{n\to+\infty} E_M(n) = a \tag{4}$$

2.5 Determination of F_L for Sequential Job Submission

In case jobs have to be sequentially submitted to the grid scheduler, the assumption of *iid* latencies obviously does not hold. This is for instance the case when using EGEE's Resource Broker, as done in the experiment section of this paper. In such a case, the submission of a job is delayed by the sum of the submission times of its predecessors, so that the latency faced by the i^{th} submitted job of an application is:

$$R_i = T_i + G$$

where T_i is the submission time of job i (it depends on the number of previously submitted jobs) and G is the rest of the grid latency. On the other hand, grid latency measures (e.g. based on probe monitoring) capture the following:

$$M = S + G$$

where S is the submission time of a single job.

We can consider that T_i are *iid* and the job rank i is uniformly distributed between 1 and n. Thus, $T_i = T$ and T has the following *pdf*:

$$f_T(t) = \frac{1}{n}\left(\sum_{k=1}^{n} f_{kS}(t)\right) = \frac{1}{n}\left(\sum_{k=1}^{n} f_S^{*k}(t)\right)$$

where f_{kS} denotes the *pdf* of random variable $\sum_{j=1}^{k} S$ and f_S^{*k} is f_S convolved k times with itself. Then, $L = T + G$ and we have:

$$F_L(t) = \int_0^{\infty} \frac{1}{n}\left(\sum_{k=1}^{n} f_S^{*k}(u)\right) F_G(t-u)du \tag{5}$$

Equations 1, 2 and 3 still hold and F_L can be computed from the latency distribution using equation 5.

3 Experiment Set-Up and Results

The modelling was evaluated on a radiotherapy simulation application running on the EGEE grid with the DIANE pilot-job framework [13]. The experiments were conducted on the `biomed` Virtual Organisation (VO) of the EGEE grid, gathering approximately 190 computing sites and more than 40 grid schedulers (Resource Brokers) at the time of the experiment.

Nineteen (19) runs of the application (6h45min each on a 2.4 GHz Intel Core Duo PC) were performed, varying the number of submitted pilots (n) from 25 to 150. Pilots were submitted to a single grid scheduler.

The distribution of the grid latency (F_L) was measured from probe round-trip times, which can be problematic, e.g., when schedulers are configured to use different priorities depending on job length, user or recent resource usage.

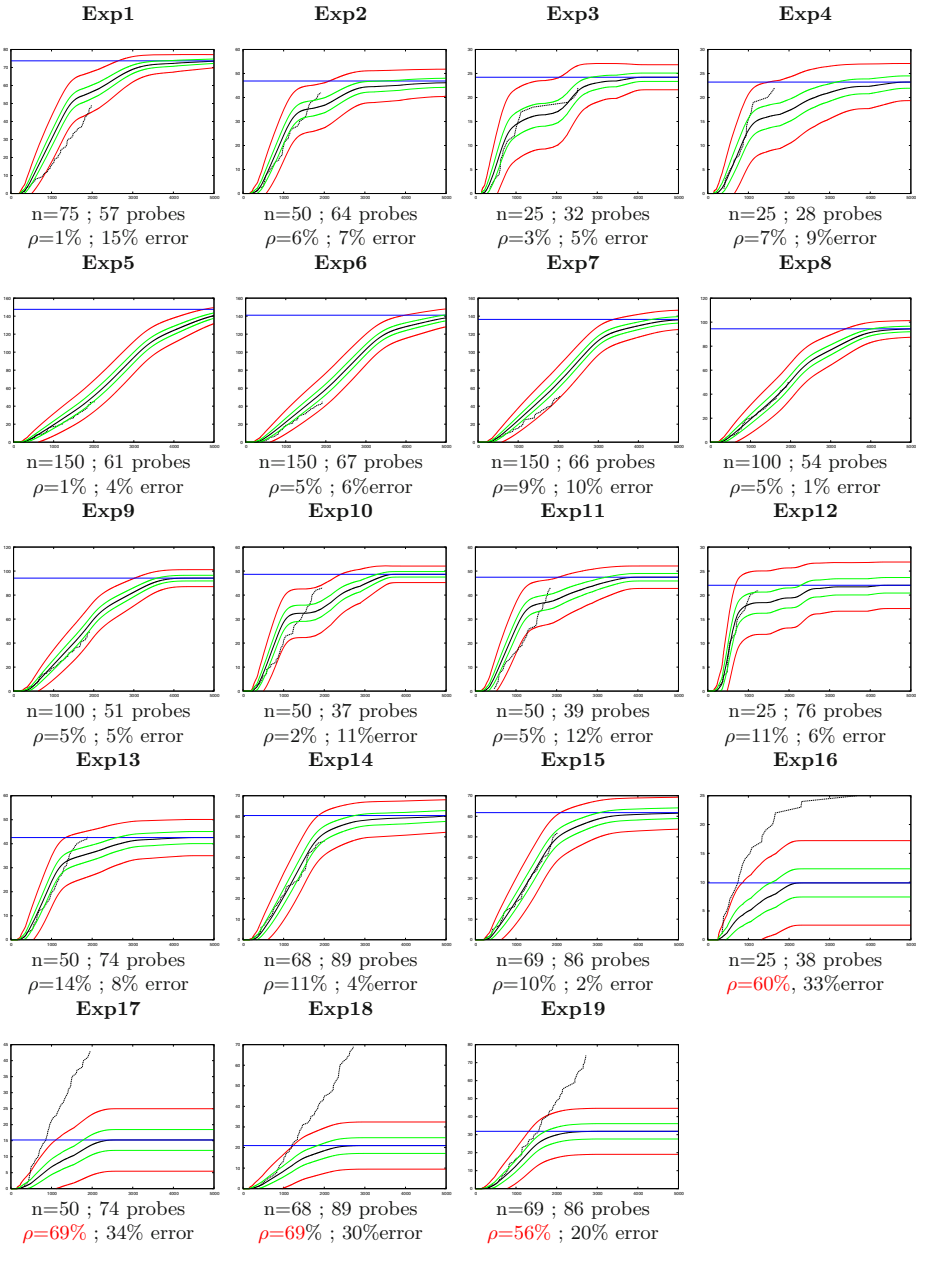

Fig. 1. Number of available pilots along time: the model is figured with plain lines ($E_N(t)$ is the central black line, $\pm\sigma_N(t)$ is in green and $\pm 3\sigma_N(t)$ is in red) and the measure with dashed lines. The x-axis represents time in seconds and the y-axis denotes the number of agents. When monitoring probes are reliable enough (Exp 1 to 15) the model fits the data with a 7% error w.r.t the number of submitted pilots.

In such cases, historical data from the considered user and application could be used to estimate the latency distribution. This could be obtained either by application-level monitoring or by querying middleware services.

To avoid biases, probes and application jobs were submitted from machines located in different cities, using different user credentials and different submission codes. To evaluate the relevance of our modelling of sequential job submission (section 2.5), we used an EGEE Resource Broker with sequential submission.

A fixed number of probes was permanently maintained inside the system. To ensure that enough probes were used to build the latency *cdf*, probes submitted up to 1 hour before the experiment were considered in addition to the ones submitted during the experiment. For each experiment those probes were used to build F_L using equation 5. A total number of 1093 probes has been used for the 19 experiments, among which a global 6.8% fault ratio has been measured. Their mean latency was 767 seconds and the standard-deviation was 766 seconds.

The computation of $E_N(t)$ and $\sigma_N(t)$ was done straight from equations 1 and 2. The total work time w_0 was estimated as the value leading to the minimal mean-square error between modelled and experimental makespans.

3.1 Results

Figure 1 compares the theoretical and measured number of available pilots throughout the 19 experiments. For each graph, the central black plain line corresponds to $E_N(t)$ as computed from equation 1. The interval delimited by plain green lines covers $E_N(t) \pm \sigma_N(t)$ and the red covers $E_N(t) \pm 3\sigma_N(t)$, $\sigma_N(t)$ being computed from equation 2. The blue horizontal line corresponds to $\lim_{n\to\infty} E_N(t)$, i.e. to $n(1-\rho)$ where ρ is the fault ratio among monitoring probes. Experimental data is figured with dashed black lines. For each experiment, the number of submitted pilots n, the number of probes used to parametrize the model, the probes fault ratio ρ and the model error are shown. Model error is computed as $\frac{\frac{1}{k}\sum_{i<k}|m_i-e_i|}{n}$ with k the total number of measure samples, m_i the value of $E_N(t_i)$ obtained from equation 1, e_i the corresponding experimental value (measure) and n the number of submitted pilots for the experiment.

Figure 2 plots the experimental and modelled makespans. The w_0 value minimizing the mean square error w.r.t the data is 39,600s (11 hours). More than 80 grid sites were used for those experiments.

4 Discussion

4.1 Number of Available Pilots

The model is clearly off on 4 of the 19 experiments presented on figure 1, with mean errors greater than 20% (**Exp 16** to **19**). For those experiments, a high number of faults ($> 55\%$) is observed among the monitoring probes while the application is not impacted. In these cases, the asymptotic line of the model

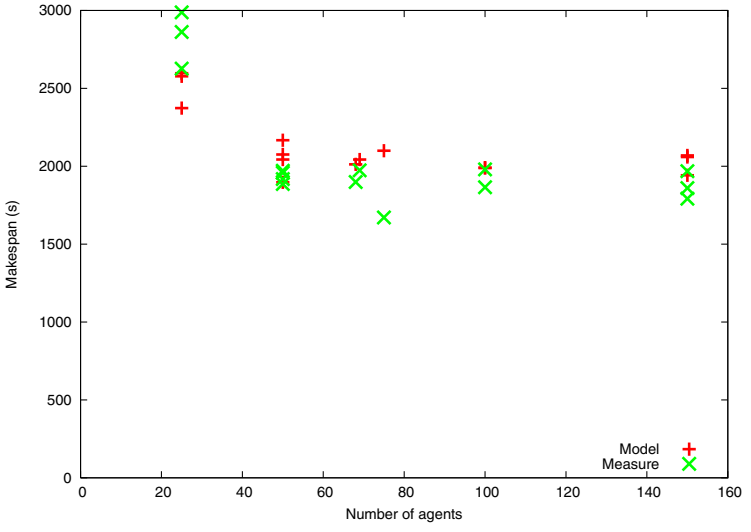

Fig. 2. Measured VS modelled makespans after mean square minimization of the model w.r.t w_0. Mean error is 5min27s; w_0 is 11 hours.

(blue line with y-value $n(1-\rho)$) is very low while the experimental curve grows to regular values. A detailed inspection of the data revealed that such a fault ratio was indeed not faced by pilots submitted during the experiment. Accurately measuring fault ratios is challenging because some kinds of faults occur during a very localized time period, for instance when a service becomes unavailable for a couple of minutes while being restarted. Since the model is parametrized with probes acquired up to 1 hour before the experiment, such localized faults completely puzzle the monitoring system while the experiment may not be impacted at all. In such cases, the grid behavior is too dynamic to be properly captured. In the following discussion we only consider **Exp 1** to **Exp 15** for which fault ratios remain under 15%, a common value on EGEE.

Excluding those 4 experiments, the average model error is 7% with respect to n. This can be considered quite low given the grid variability and the independence of the monitoring probes from the experiments. All the measures but some of **Exp 1** stay in the red interval $[E_N(t)-3\sigma_N(t),E_N(t)+3\sigma_N(t)]$ and a significant proportion is in the green $[E_N(t)-\sigma_N(t),E_N(t)+\sigma_N(t)]$. From a qualitative point of view, the shape of the experimental data is properly captured by the model: for instance, **Exp 3** shows a strong bi-modal behavior nicely described by the model. The model generally tends to overestimate the performance of the application, in particular when the number of submitted pilots is important (see in particular **Exp 1**,**5**, **6**, **7**, **8** and **9** for which $n \geq 75$). This may be due to the fact that in some cases, the makespan is lower than the total submission time, thus reducing the actual number of submitted pilots.

4.2 Makespan Estimation

The experimental makespan (green crosses on figure 2) shows significant variability for a given value of n, which is explained by the variability of grid execution conditions and the heterogeneity of the infrastructure. Still, one can notice a clear decreasing trend between $n = 25$ and $n = 50$ followed by a stable phase around 2000s where the makespan seems to reach a limit, as forecast by the theoretical study in section 2. However, this limit is clearly superior to the value given by the theory, supposed to be a, the lower bound of the support of the *grid* latency *cdf* (150s for the 1093 monitoring probes considered here). The reason for that is obviously the *application* latency, mainly composed of data transfers and the initialization step.

The model (red crosses on figure 2) is able to properly capture the behavior of the experimental makespan. After mean-square error minimization, the mean error between the model and the experimental data is 327 seconds (5 min 27 s). Given the standard-deviation measured on the latency of the monitoring probes (766s), this error can be considered as very small. The dynamicity of the grid is captured thanks to the monitoring system while the evolution with respect to n is correctly described by the probabilistic model.

Finally, the w_0 value given by the minimization (11 hours) approximates the mean measured w_0 (11 hours 13min) with an error of only 2%, which is extremely accurate for an application running in production conditions. We thus conclude that (i) the model is able to explain the observed makespan and (ii) the model is able to estimate the total work time w_0 of the application.

5 Conclusion

We presented a model for pilot-job applications, an execution scheme becoming increasingly adopted by applications to cope with heterogeneity of production grids. Based on the distribution of the latency, the mean and standard-deviation of the number of available pilots along time as well as the makespan of the application are described. An estimation of the total work time of the application is also provided. Sequentially submitted jobs can also be handled. Faults are taken into account by applying a basic transformation on the latency distribution.

Nineteen (19) experiments have been carried-out on a radiotherapy simulation application running on the EGEE production grid using DIANE pilot jobs. Relying on a parametrization of the model using independent monitoring probe jobs, results show that (i) the model is able to correctly describe the evolution of the number of available pilots along time, provided that the fault ratio of the monitoring probes remains reasonable, (ii) the makespan of the application is accurately described and (iii) the total work time of the application is estimated with a 2% error.

The proposed model is thus a suitable tool for analyzing the behavior of pilot-job applications. Though the experiments were carried-out using DIANE, the modelling does not include any implementation-specific assumption, which

makes the results applicable to other pilot jobs systems deployed on the same type of production grid infrastructures.

Acknowledgement

We warmly thank David Sarrut for the expertise provided with the radiotherapy application, Jakub Mosciki for his assistance with the DIANE framework and Thomas Grenier for fruitful discussions. We are also grateful to the EGEE European project for providing the grid infrastructure and related user support.

References

1. Ahn, S., Namgyu, K., Seehoon, L., Soonwook, H., Dukyun, N., Koblitz, B., Breton, V., Sangyong, H.: Improvement of Task Retrieval Performance Using AMGA in a Large-Scale Virtual Screening. In: NCM'08, pp. 456–463 (September 2008)
2. Fu, S., Xu, C.-Z.: Exploring event correlation for failure prediction in coalition of clusters. In: Supercomputing (2007)
3. Germain, C., Loomis, C., Mosciki, J.T., Texier, R.: Scheduling for Responsive Grids. JGC 6(1), 15–27 (2008)
4. Glatard, T., Lingrand, D., Montagnat, J., Riveill, M.: Impact of the execution context on Grid job performances. In: WCAMG'07 (CCGrid'07), pp. 713–718 (May 2007)
5. Glatard, T., Montagnat, J., Pennec, X.: Probabilistic and dynamic optimization of job partitioning on a grid infrastructure. In: PDP'06, pp. 231–238 (February 2006)
6. Glatard, T., Montagnat, J., Pennec, X.: Optimizing jobs timeouts on clusters and production grids. In: CCGrid'07, pp. 100–107 (May 2007)
7. Jacq, N., Salzeman, J., Jacq, F., Legre, Y., Medernach, E., Montagnat, J., Maass, J., Reichstadt, M., Schwichtenberg, H., Sridhar, M., Kasam, V., Zimmermann, M., Hofmann, M., Breton, V.: Grid-enabled Virtual Screening against malaria. JGC 6, 29–43 (2008)
8. Juve, G., Deelman, E.: Resource Provisioning Options for Large-Scale Scientific Workflows. In: eScience'08, pp. 608–613 (December 2008)
9. Kacsuk, P., Farkas, Z., Fedak, G.: Towards making BOINC and EGEE interoperable. In: eScience'08, pp. 478–484 (December 2008)
10. Lingrand, D., Montagnat, J., Martyniak, J., Colling, D.: Analyzing the EGEE production grid workload: Application to jobs submission optimization. In: Frachtenberg, E., Schwiegelshohn, U. (eds.) JSSPP 2009. LNCS, vol. 5798, pp. 37–58. Springer, Heidelberg (2009)
11. Mosciki, J.T.: Distributed analysis environment for HEP and interdisciplinary applications. Nuclear Instruments and Methods in Physics Research A 502, 426–429 (2003)
12. Ngoc Minh, T., Wolters, L.: Modeling Job Arrival Process with Long Range Dependence and Burstiness Characteristics. In: CCGrid'09, pp. 324–330 (May 2009)
13. Sarrut, D., Guigues, L.: Region-oriented ct image representation for reducing computing time of monte carlo simulations. Med. Phys. 35(4) (2008)
14. Schroeder, B., Gibson, G.A.: A large-scale study of failures in high-performance computing systems. In: DSN, pp. 249–258 (2006)
15. Sfiligoi, I.: glideInWMS: a generic pilot-based workload management system. Journal of Physics: Conference Series 119(6) (2008)