J Grid Computing (2008) 6:77–101
DOI 10.1007/s10723-007-9089-1

Statistical Analysis and Modeling of Jobs in a Grid Environment

Konstantinos Christodoulopoulos · Vasileios Gkamas · Emmanouel A. Varvarigos

Received: 31 January 2007 / Accepted: 7 September 2007 / Published online: 29 November 2007
© Springer Science + Business Media B.V. 2007

Abstract The existence of good probabilistic models for the job arrival process and the delay components introduced at different stages of job processing in a Grid environment is important for the improved understanding of the Grid computing concept. In this study, we present a thorough analysis of the job arrival process in the EGEE infrastructure and of the time durations a job spends at different states in the EGEE environment. We define four delay components of the total job delay and model each component separately. We observe that the job inter-arrival times at the Grid level can be adequately modelled by a rounded exponential distribution, while the total job delay (from the time it is generated until the time it completes execution) is dominated by the computing element's register and queuing times and the worker node's execution times. Further, we evaluate the efficiency of the EGEE environment by comparing the job total delay performance with that of a hypothetical ideal super-cluster and conclude that we would obtain similar performance if we submitted the same workload to a super-cluster of size equal to 34% of the total average number of CPUs participating in the EGEE infrastructure. We also analyze the job inter-arrival times, the CE's queuing times, the WN's execution times, and the data sizes exchanged at the *kallisto.hellasgrid.gr* cluster, which is node in the EGEE infrastructure. In contrast to the Grid level, we find that at the cluster level the job arrival process exhibits self-similarity/long-range dependence. Finally, we propose simple and intuitive models for the job arrival process and the execution times at the cluster level.

Keywords Grid computing · Job profiling · Delay components · Probabilistic modeling

Abbreviations

LHC	Large Hadron Collider
LCG	LHC Computing Grid
EGEE	Enabling Grids for E-sciencE
RB	Resource Broker
CE	Computing Element
WN	Worker Node
SE	Storage Element
UI	User Interface
VO	Virtual Organizations
WMS	Workload Management System
RTM	Real Time Monitor

K. Christodoulopoulos (✉) · V. Gkamas · E. A. Varvarigos
Computer Engineering and Informatics Department and
Research Academic Computer Technology Institute,
University of Patras,
26500 Rio, Patras, Greece
e-mail: kchristodou@ceid.upatras.gr

V. Gkamas
e-mail: gkamas@ceid.upatras.gr

E. A. Varvarigos
e-mail: manos@ceid.upatras.gr

LRMS	Local Resource Management System
BDII	Berkeley Database Information Index
MMPP	Markov Modulated Poisson Processes
NHPP	Non-Homogeneous Poisson Process
HE	Hyper-Exponential Distribution

1 Introduction

Grids introduce new ways to share computing and storage resources across geographically separated sites by establishing a global resource management architecture [1]. The job inter-arrival times, the job execution times, and the times jobs spent at different phases of processing in Grids are unknown and are better modeled probabilistically. Finding good probabilistic models for the job submission process, the job delay components, and the job characteristics gives us important insight into the operation of Grid systems and can be used by the developers and the research community in several ways. By observing the system behavior under different values of the parameters involved the developers can evaluate the performance, study the way it depends on the choice of different parameters, and possible identify problems and propose new methods to improve and optimize the employed middleware. Given the high cost involved in setting up actual hardware implementations, simulations are a viable alternative. A necessary prerequisite to obtain useful results is an adequate model of the traffic (i.e. job arrival process) and the times the job spend at different states. Probabilistic models can be used to facilitate the dimensioning of Grid systems and the prediction of their performance under different scenarios. Moreover, simulations can be used to evaluate new quality of service policies and scheduling algorithms both at the Grid (meta-scheduling) and at the cluster level (metacomputing).

Even though a large number of works on job characterization and modeling [2] for single parallel supercomputers [3, 4] and clusters [5] have appeared in the literature, the corresponding attempts in the area of Grid computing are quite limited [6]. In [6], Medernach analyzed and modeled the workload of a LCG/EGEE cluster. More specifically, a two-dimensional Markov chain, which is equivalent to a two-phase hyper-exponential process, was proposed for modeling user behavior in a Grid environment. The user shifts between login and logout states and submits jobs when being in the login state. The results indicate that the two-phase hyper-exponential process can satisfactorily model the submission behavior of a single user.

Taking a different approach, Li et al. [7] used the LCG real time monitor tool [8] to collect data from resource brokers (RBs) located at CERN, Germany, and the UK, and proposed traffic models for the job arrival processes at three different levels: Grids, virtual organizations and regions. By comparing a set of m-state Markov modulated Poisson processes (MMPP) with Poisson and hyper-exponential processes, they conclude that MMPP models with a certain number of states are capable of modeling the submitted job traffic at the three examined levels. However, the proposed models are not intuitive enough, and they do not provide an easy, adaptable or extensible way for profiling arrival processes in Grid environments. Focusing on certain VOs that present strong pseudo-periodic components in their job interarrival times at the Grid level, a methodology to analyse and synthesize pseudo-periodic job arrival processes is examined in [9]. D. Nurmi et al. [10] used accurate predictions of both the execution time of the task and the time the task spends waiting in the queue of a cluster in order to propose enhancements for a workflow scheduler. Experiments in five HPC showed that incorporating these enhancements improve the workflow execution time when batch queues impose significant delays on these workflows.

The measurements that are presented in this study correspond to two different levels of observation: (1) the Grid level, meaning that we have considered the overall LCG/EGEE infrastructure as a single entity and observed the general properties of job submission and execution in this real and highly utilized Grid environment and (2) the cluster level; in particular we present measurement for our local LCG/EGEE cluster named *kallisto.hellasgrid.gr.*

Based on the LCG/EGEE job flow diagram we distinguish four delay components of the job processing, each corresponding to time spent at different states in the LCG/EGEE environment, from the submission of a job until the retrieval of the corresponding output data. Considering the Grid level of observation we analyze and model each delay component separately. At this level, we also model the job arrival process and examine the efficiency of the LCG/EGEE environment and the currently employed super-scheduling algorithm.

We also analyze the inter-arrival times and the workload at our local LCG/EGEE cluster (*kallisto.hellasgrid.gr*), and propose models for the job arrival process and execution times at a Grid node. Our models are simple and use a small number of modeling parameters, so as to remain comprehensive and intuitive.

Our results indicate that if we consider the LCG/EGEE Grid as the level of our observation, jobs are submitted continuously without any specific weekly or daily patterns. The job inter-arrival times are found to match very well with a rounded exponential distribution with mean 1.6077 s. We then define and model the four delay components that comprise the overall job processing in the LCG/EGEE environment. More specifically, the first delay component (D_1) corresponds to the time a job spends in the pending, submitted and waiting states and can be adequately modeled as a deterministic (constant) parameter. The second delay component (D_2) corresponds to the time a job stays in the ready state and can be modeled very well by a two-phase lognormal distribution. We observe that the total time a job stays in the LCG/EGEE environment is dominated by the computing element's register and queuing delay and the worker node's execution time that correspond to the third (D_3) and fourth (D_4) delay components, respectively. The register and queuing times (D_3) can be modeled with the same distribution (but different parameters) as D_2. For the WN execution time D_4, we find that a hyper-exponential model with three states is sufficient for modeling the stepwise patterns observed in the empirical distribution obtained from our measurements.

We evaluate the efficiency of the LCG/EGEE environment and (indirectly) of the currently employed super-scheduling algorithm by comparing the total delay experienced by a job in the LCG/EGEE environment with that of a hypothetical ideal super-cluster, and conclude that we would have similar performance if we submitted the same workload to a super-cluster having 34% of the total average number of CPUs participating in the LCG/EGEE project. This is an indication that the Grid computing concept can meet to a satisfactory degree its main promise, which is to provide to users the ability to treat distributed computing and storage resources, as if belonging to a single computer.

Turning our attention to the cluster level we again observe that it is difficult to find patterns for the weekly and daily cycle of the arrival process. By computing the Hurst parameter of the inter-arrival times we found that the job arrival process exhibits self-similarity/long-range dependence. We investigated four models for the job arrival process: a non-homogeneous Poisson process model, a hyper-exponential model, a Markov modulated Poisson process model and a custom model, firstly introduced here, the Pareto-exponential model. We found that, despite its simplicity, the proposed Pareto-exponential model appears to adequately describe the job arrival process at the cluster level, and is more accurate than the other models proposed in the literature. Similar to the Grid-level, we found that a hyper-exponential process with three states is sufficient to model the stepwise patterns observed in the distribution of the jobs' Worker Node execution time. By looking at the CE queuing times we found that a high percentage of jobs are served almost immediately, while there are also jobs that remain for a long period in the corresponding queues.

We have to mention that the statistical analysis, and the modeling of the delay components presented in this paper correspond to the LCG/EGEE production Grid. Since the LCG/EGEE infrastructure consists of a large number of heterogeneous and distributed sites that is used by a wide diversity of users and applications we believe that it is representative of large Grid infrastructures. Moreover, the LCG g-lite middleware follows a general architecture (http://www.globus.org/) and thus the definitions of the delay components are applicable to a large variety of systems.

The rest of this work is organized as follows. The LCG/EGEE environment is presented in Section 2. Section 3 describes the job flow in the LCG/EGEE environment and the various metrics used for the analysis of the job arrival process and the job delay components. In Section 4 we introduce the candidate traffic models considered. Section 5 presents the statistical results obtained at the Grid level. The modeling of the inter-arrival times and the job processing delay components is presented in Section 6. The evaluation of the overall efficiency of the LCG/EGEE environment is presented in Section 7. The *kallisto.hellasgrid.gr* Grid node is presented in

Section 8. Section 9 presents the statistical analysis of the kallisto's computing element (CE) and storage element (SE), while in Section 10 we propose and validate models for the corresponding job arrival process and the job execution times. Section 11 concludes the study.

2 LCG/EGEE Projects and Infrastructure

The EGEE project [11] aims at providing to the researchers access to a geographically distributed Grid infrastructure, available 24 h a day. It focuses on maintaining the gLite middleware [12] and on operating the infrastructure for the benefit of a large and diverse research community.

The worldwide LHC computing Grid project (LCG; http://lcg.web.cern.ch/LCG/) was created to prepare the computing infrastructure for the simulation, processing and analysis of the data of the large hadron collider (LHC) experiments. The LCG and the EGEE projects share a large part of their infrastructure and operate it in conjunction. For this reason, we will refer to it as the LCG/EGEE infrastructure. Currently, 207 clusters (sites) from 48 different countries participate in the LCG/EGEE infrastructure. In the observation period of our study, there were totally 39,697 CPUs and about 5 Petabytes of storage in the LCG/EGEE infrastructure, while the total average number of available CPUs was 31,228 (http://goc.grid.sinica.edu.tw/gstat/index.html).

In the LCG/EGEE environment, users are organized in virtual organizations (VOs), which are dynamic collections of individuals and institutions sharing resources in a flexible, secure and coordinated manner. A user has to belong to a VO to be able to use the LCG/EGEE infrastructure. A list of existing VOs in the EGEE is available at (https://lcg-registrar.cern.ch/virtual_organization.html).

3 Job Flow in the LCG/EGEE Environment and Metrics Used

Generally, a user has to login to a user interface (UI) through which he can submit a job directly to a cluster or submit it via a resource broker (RB). The second case is the most commonly used and can be monitored, and for this reason we have concentrated on it. The description of the job is written in a specific format (JDL – job description language [13]). This is forwarded to the corresponding resource broker (RB) where the matching process is performed [12]. An RB runs the workload management system (WMS) service that intercommunicates with the information system (IS, providing information about the Grid resources and their status). The RB uses the job description, the related VO and available global load information to decide about whether or not and where to forward the job. Users, when submitting a job, give a rough estimate of its maximum running time, but this value is usually overestimated and is considerably larger than the actual job execution time.

When a job is submitted to the LCG/EGEE environment it passes through several states till the user gets back the desired output data. These states insert corresponding delay components to the total job processing time. The job flow from its submission from a UI, till the retrieval of the job output is shown in Fig. 1. Figure 2 presents the various states in which a job can be in the LCG/EGGE environment. These states come from the gLite 3 user's guide [12] enhanced with a new state (pending state) and specific time instances (epochs) useful for the analysis of the inter-arrival times and the delay components that comprise the job life cycle in the LCG/EGEE environment.

The time instances (epochs) of specific events of our interest for the purposes of modeling are the following:

- V_1 = *userinterface_regjob_Epoch:* The time instance the user submits a job from the UI to a resource broker
- V_2 = *networkserver_accepted_Epoch:* The time instance the job is accepted by the network server of the resource broker
- V_3 = *workloadmanager_match_Epoch:* The time instance the WMS starts looking for the best available CE to execute the job
- V_4 = *jobcontroller_transfer_Epoch:* The time instance the job controller of the RB starts sending the job request to the appropriate CE
- V_5 = *logmonitor_accepted_Epoch:* The time instance the CE receives the request
- V_6 = *lrms_running_Epoch:* The time instance the local resource management system (LRMS) assigns the job for execution to an available worker node from the local farm

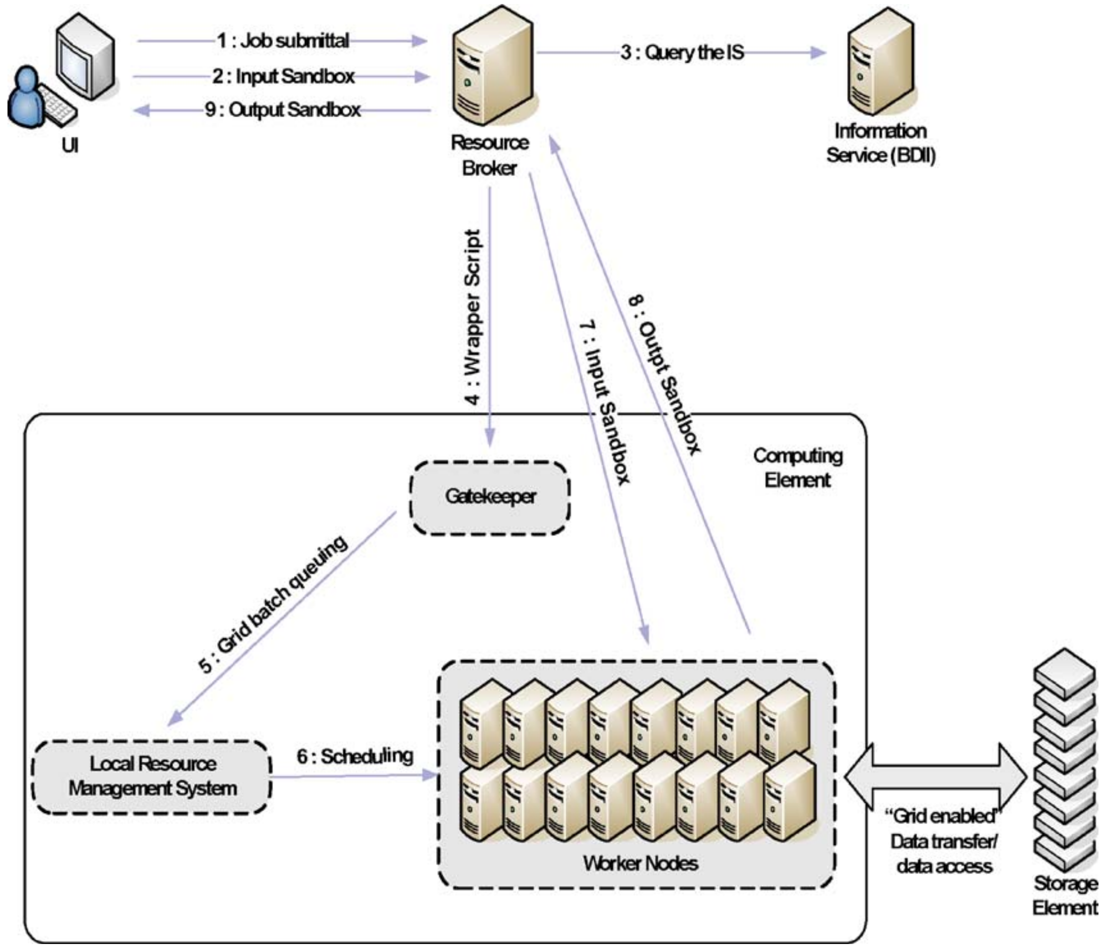


Fig. 1 Job flow in the LCG/EGEE environment

- V_7 = *logmonitor_running_Epoch:* The time instance the user files have completed transferring from the RB to the WN where the job will be executed
- V_8 = *lrms_done_Epoch:* The time instance the CE starts transferring the output back to the RB node
- V_9 = *logmonitor_done_Epoch:* The time instance after which the user can retrieve the job output to the UI.

Based on the aforementioned time epochs, we can define the various states (Fig. 2) at which a job can be at any given time in the LCG/EGGE environment as follows:

- The status of the job becomes *pending* at time instance V_1 (userinterface_regjob_Epoch) at which the job (more specifically, the job JDL file) is submitted from the UI to the RB.
- The RB receives the JDL file, which may specify one or more files to be copied from the UI to the worker node. This set of files is referred to as the input sandbox. The status of the job becomes *submitted* at time instance V_2 (networkserver_accepted_Epoch) at which the network server of the RB accepts the job.

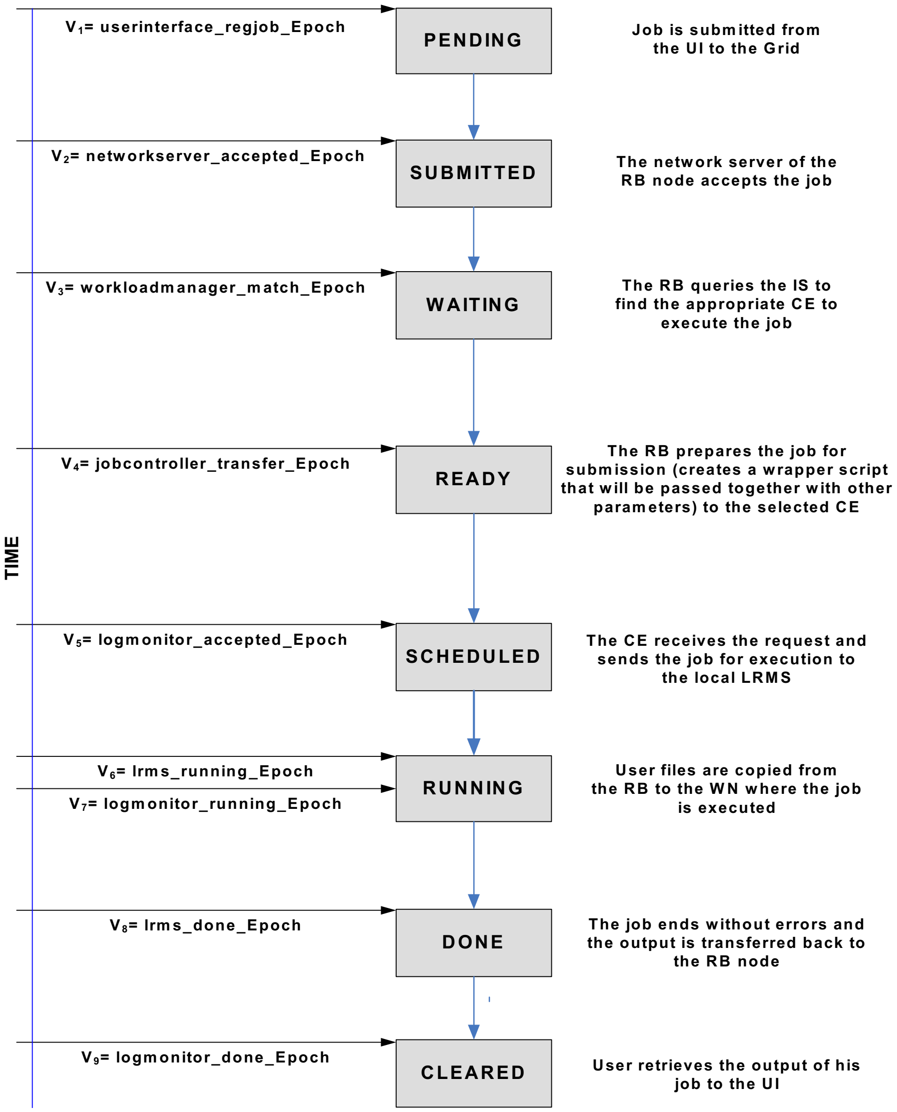

V₁= userinterface_regjob_Epoch
PENDING
Job is submitted from the UI to the Grid
V₂= networkserver_accepted_Epoch
SUBMITTED
The network server of the RB node accepts the job
V₃= workloadmanager_match_Epoch
WAITING
The RB queries the IS to find the appropriate CE to execute the job
V₄= jobcontroller_transfer_Epoch
READY
The RB prepares the job for submission (creates a wrapper script that will be passed together with other parameters) to the selected CE
TIME
V₅= logmonitor_accepted_Epoch
SCHEDULED
The CE receives the request and sends the job for execution to the local LRMS
V₆= lrms_running_Epoch
V₇= logmonitor_running_Epoch
RUNNING
User files are copied from the RB to the WN where the job is executed
V₈= lrms_done_Epoch
DONE
The job ends without errors and the output is transferred back to the RB node
V₉= logmonitor_done_Epoch
CLEARED
User retrieves the output of his job to the UI

Fig. 2 The states of a job in the LCG/EGEE environment and the corresponding time instances (epochs)

- The RB node runs the WMS service whose role is to find the best available CE to execute the job based on the requirements the user has specified in the JDL file and the status and utilization of every site. The WMS service starts to execute at time V_3 (workloadmanager_match_Epoch) at which point the status of the job becomes *waiting*.
- The RB creates a wrapper script to be passed, together with other parameters, to the selected CE. The status of job becomes *ready* at time instance V_4 (jobcontroller_transfer_Epoch) at which the RB job controller sends the job to the appropriate CE.
- The CE receives the request at time instance V_5 (logmonitor_accepted_Epoch) and the gatekeeper of the CE that controls access to local resources maps the job certificate to a local UID. Various mechanisms are in place to allow or disallow jobs based on the accompanying certificate. Following a successful mapping to UID, the gatekeeper creates a jobmanager process to manage the local submission and execution of the job on the local batch farm. The status of the job then becomes *scheduled*.
- The local resource management system (LRMS) is the service running at the CE and is responsible for the handling of the job execution on the local

farm of worker nodes. A job remains in the LRMS queue until the time instance V_6 (lrms_running_Epoch) at which time the LRMS assigns the job to a WN, and the status of the job becomes *running*. The user files complete transferring from the RB to the WN at time V_7 (logmonitor_running_Epoch).

- If the job completes without errors, the output of the job (called output sandbox), starts transferring back to the RB node at time instance V_8 (lrms_done_Epoch), at which point the status of the job becomes *done*.
- At time instance V_9 (logmonitor_done_Epoch) the output sandbox has been transferred to the RB and the user can retrieve the output of his job back to the UI. The status of the job then becomes and remains *cleared*.

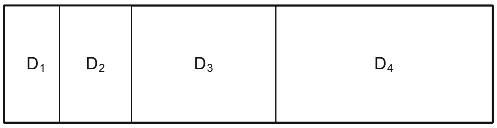
Using the previous epochs we can calculate the metrics shown in Table 1. These metrics will be used for the analysis of the various delay components that comprise the job execution in the LCG/EGEE environment. The first column of Table 1 indicates the name of the variable, the second column the corresponding state or states at which the variable refers, and finally the third column shows how every variable is calculated. For example the V_{12} variable (getting ready to transfer to CE time) describes the time spent at pending, submitted and waiting states and is computed by subtracting V_1 from V_4 time instance. We have to mention that V_{16} variable (WN execution time logmonitor) and V_{17} variable (WN execution time-lrms) correspond to the execution times of the jobs. The first one is logged by the RB while the second one by the CE and their difference is the time the job spends at the Done state (see Table 1). Also the variable V_{19} (efficiency) is defined as the WN execution time (logmonitor) divided by the total time (V_{19}=V_{17}/V_{18}). Thus, if V_{19} approaches 1.00 this means that the delay introduced at other states is negligible compared to the execution time of the job.

Based on these metrics we define the four main delay components that comprise the job processing in the LCG/EGEE environment (Fig. 3); the total time of a job (V_{18}) is the sum of these four delay components.

- D_1=V_{12} = *getting ready to transfer to CE time* describes the time the job stays at the pending, submitted and waiting states. This delay component consists of the time a job requires to register with the RB, and the time the RB takes to run the match making service and create the wrapper scripts to transfer the job to the chosen CE.
- D_2=V_{13} = *transfer time* describes the time the job stays at ready state. This time consists of the time required to transfer the job wrapper scripts from the RB to the chosen CE.
- D_3=V_{15} = *CE register and queuing time* describes the time the job stays at Scheduled state. This time corresponds to the time required by the CE gatekeeper to accept and match the job to a local UID and the time the job stays at the CE queue before it starts to execute at a WN. It also includes the time that is required to transfer the input user files – input sandbox – from the RB to the WN.
- D_4=V_{16} = *WN execution time* (logmonitor) describes the time the job stays at running and done states. This time consists of the time required to execute the job and to transfer the output files –

Table 1 Metrics used for analysis of the various states of the job in the Lcg/Egee environment

Variables	Corresponding states	
V_{10} = registration_Time	Pending	(V_2−V_1)
V_{11} = match_Time	Submitted	(V_3−V_2)
V_{12} = getting_ready_to_transfer_to_CE_Time	Pending + submitted + waiting	(V_4−V_1)
V_{13} = transfer_Time	Ready	(V_5−V_4)
V_{14} = logmonitor_CE_total_Time	Scheduled + running + done	(V_9−V_5)
V_{15} = logmonitor_CE_register_queueing_Time	Scheduled	(V_6−V_5)
V_{16} = logmonitor_WN_Time	Running + done	(V_9−V_6)
V_{17} = lrms_wn_Time	Running	(V_8−V_6)
V_{18} = total_Time	Submitted + waiting + ready + running + done	(V_9−V_1)
V_{19} = efficiency	(Running + done) /(submitted + waiting + ready + running + done)	(V_{17}/V_{18})



D_1 : Getting Ready to Transfer to CE time (Pending+Submitted+Waiting)
D_2 : Tranfer time (Ready)
D_3 : CE Register and Queuing time (Scheduled)
D_4 : CE Execution time (Running + Done)

Fig. 3 Delay component of a job in LCG/EGEE environment

output sandbox – to the corresponding RB from which the user can retrieve them. It is worth noting that after the output files have been transferred to the RB the job state becomes and remains cleared (until the user retrieves the output files or the system discards them). In the definition of the delay components previously presented, we have not considered the time the job stays in the cleared state since it mainly depends on the user and does not correspond to a quantifiable characteristic of the Grid.

4 Analytical Models

It is possible to use directly log traces for the job arrivals as an input to a static simulation, but it is usually more convenient to define and use analytic models for the job arrival process. Analytic models are more flexible, since they allow the generation of traces using different values of the parameters involved, helping better understand the way these parameters affect system performance.

In this section we present analytical models that are used in the modeling sections (Sections 6 and 10) of this study.

The classical Poisson process, in which the inter-arrival times are exponentially distributed, forms the basis for some of the more advanced models.

4.1 Non-Homogeneous Poisson Process (NHPP)

A non-homogeneous Poisson process (NHPP) is a Poisson process whose arrival rate λ at time t is a function of time $\lambda(t)$. More specifically, the number of arrivals $N(t)$ in the interval (0,t) follows the distribution:

$$\Pr(N(t)=n)=e^{-m(t)}\frac{(m(t))^n}{n!},n\geq 0 \; and \; m(t)=\int_0^t \lambda(s)ds$$

4.2 Phase Type – Hyper Exponential model

A random variable that follows the m-phase-type distribution (PT) can be defined as the transition time until absorption of a continuous-time Markov chain (CTMC) with m transient states and one absorbing state. Generally, any inter-arrival process can be approximated by a phase-type distribution provided that a sufficient number of states are used.

From this general class we chose to consider only the hyper-exponential subclass, which is the one most often used in the literature.

The probability density function (pdf) of an m-phase hyper-exponential random variable (r.v.) X is given by:

$$f_x(x)=\sum_{i=1}^{m}p_i\cdot f_{Ei}(e)=p_1\cdot f_{E1}(e)+p_2\cdot f_{E2}(e)$$
$$+\ldots+p_m\cdot f_{Em}(e)$$

where E_i is an exponential r.v. with mean $1/\lambda_i$, and p_i is the probability that X takes on the form of E_i (thus, $\sum_{i=1}^{m}p_i=1$)

4.3 Phase Lognormal Model

A r.v. L_i is said to follow the lognormal distribution if the r.v. $ln(L_i)$ is normally distributed.

Similarly to the hyper-exponential model, the pdf of an m-phase lognormal r.v. X is given by:

$$f_x(x)=\sum_{i=1}^{m}p_i\cdot f_{Li}(l)=p_1\cdot f_{L1}(l)+p_2\cdot f_{L2}(l)$$
$$+\ldots+p_m\cdot f_{Lm}(l)$$

where L_i is a lognormal r.v. with average a_i and standard deviation d_i, and p_i is the probability that X will take on the form of L_i ($\sum_{i=1}^{m}p_i=1$).

4.4 Markov Modulated Poisson Process (MMPP) Model

An m-state MMPP is a doubly stochastic Poisson process [14]. Assuming an m-state continuous-time Markov chain (CTMC), arrivals occur according to a

Poisson process of rate λ_i when the chain is in state i. An MMPP can be fully described by the parameters:

$$Q = \begin{bmatrix} -\sigma_1 & \sigma_{12} & \dots & \sigma_{1m} \\ \sigma_{21} & -\sigma_2 & \dots & \sigma_{2m} \\ \dots & \dots & \dots & \dots \\ \sigma_{m1} & \sigma_{m2} & \dots & -\sigma_m \end{bmatrix}, \sigma_\iota$$

$$= \sum_{j=1, j \neq i}^{m} \sigma_{ij} \textit{ and } \Lambda = [\lambda_1, \lambda_2, \dots \lambda_m]$$

where Q is the generator of the CTMC, and the entries of Λ correspond to the Poisson arrival rates at each state.

4.5 Pareto-Exponential Model

This model, to be referred to as the Pareto-exponential model, will be used for modeling the job arrival process at the cluster level (Section 10.1). Under this model, the VOs submit jobs that have exponential inter-arrival times (with rate λ jobs per second) during busy periods, each of which has an exponential duration (with mean 1/μs). The times between the beginnings of the VO busy periods are distributed according to a truncated Pareto distribution with Pareto shape parameter a, minimum value parameter $X_{\min}$ and maximum value parameter $X_{\max}$. The Pareto-exponential model is depicted in Fig. 4.

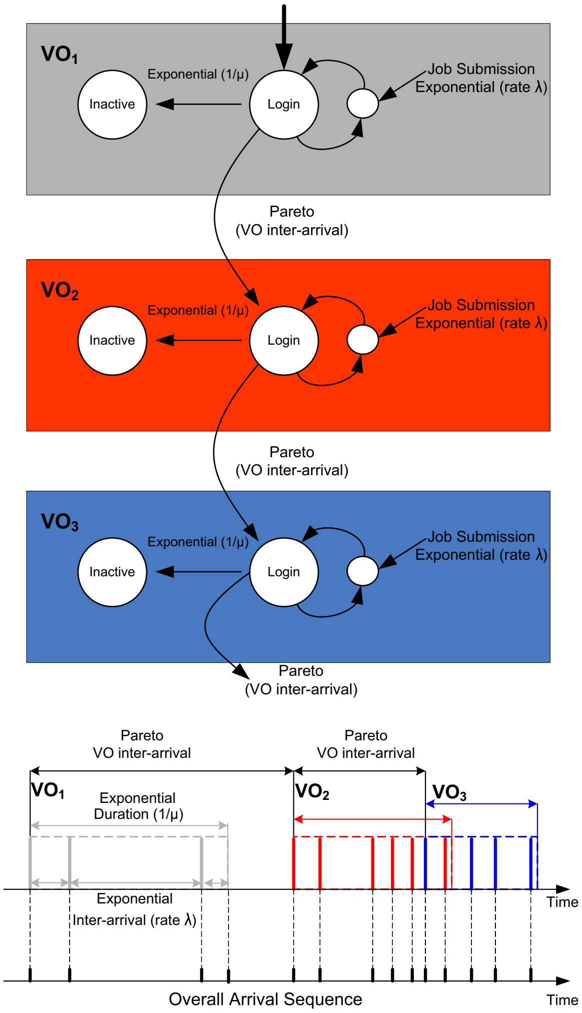


Fig. 4 Proposed Pareto-exponential model for the job arrival process

5 Statistical Results On The LCG/EGEE Usage

Using the daily reports in ASCII format supplied by the real time monitor tool we acquired information on the traffic submitted to the LCG/EGEE infrastructure and the time durations the jobs spent in each processing state before completing execution. The real time monitor (RTM) [8] is a Java applet that monitors the LCG in real time. It shows the times at which user jobs are submitted to the resource brokers all over the world, the way they are distributed to the sites, and finally, depending on their successful or not execution, the times at which the jobs complete the different states of their processing. It also presents the times of delivery of the execution outcome to the corresponding user. The real time monitor (RTM) tool uses the Berkeley Database Information Index (BDII) to automatically discover and plot new sites that join the Grid. The users of the RTM can view the LCG/EGEE traffic in real time.

We concatenated the daily ASCII report files and obtained a file that included the desired information in a form suitable for processing with statistical analysis tools. The time period of the observation was 1 month (from the 1st of October 2006 until the 31st of October 2006). The total number of jobs submitted during this period was 2.228.838.

From the real time monitor tool we were able to retrieve general information regarding the job processing and also the time epochs that correspond to specific events in the LCG/EGEE environment. By manipulating these epochs we were able to calculate the metrics presented in Table 1 and thus analyze the

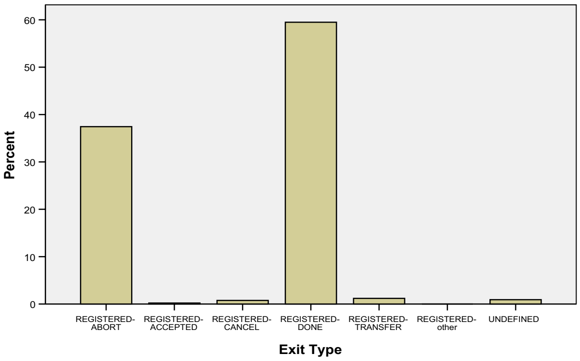


Fig. 5 Exit type of the jobs

times the job spent at different states of its processing and the corresponding delay components. The results obtained are described in the following sections.

5.1 General Statistics

In this section we present general statistics obtained for the LCG/EGEE infrastructure using the daily report files supplied by the real time monitor tool.

5.1.1 Exit Type

According to Real Time Monitor report files, there are various exit types that describe the final-exit status of a job. After manipulating the different exit types and grouping them together we obtained the results of Fig. 5. We observe that a high percentage of jobs (~59.5%) were successfully completed, while there is also a considerable percent of jobs (~37.4%) that were aborted due to middleware or hardware errors. The remaining seven exit types presented in Fig. 5 were observed with frequency less than 1%. We note that the CANCEL exit type corresponds to the case a job is canceled by the user while it was being executed.

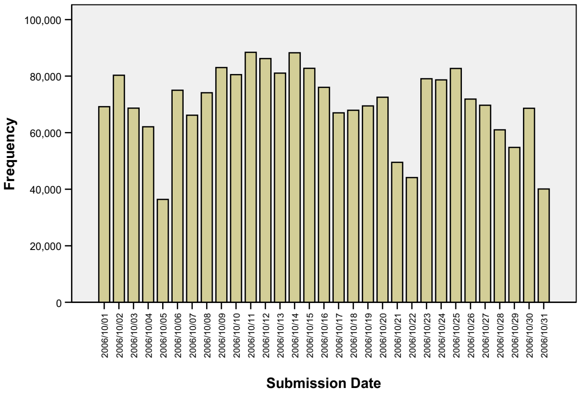


Fig. 6 Number of jobs per day

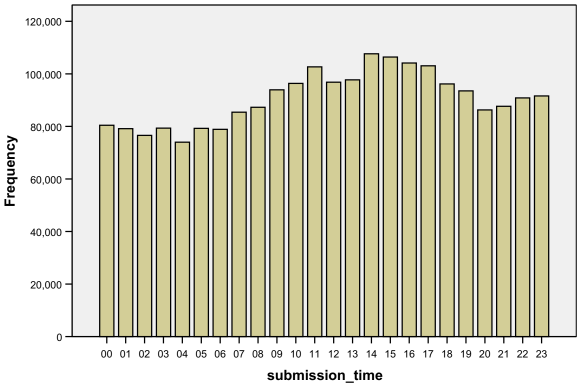


Fig. 7 Hour distribution of jobs during October 2006

5.1.2 Daily and Hour Cycles

Figure 6 shows the number of submitted jobs at all the resource brokers with respect to the submission date (in October 2006), while Fig. 7 shows the number of jobs submitted at different hours within a day (for each hour we summed up the jobs submitted during that hour in October). We observe that it is difficult to identify any pattern with respect to the date of the submission process. Jobs are submitted to the resource brokers during all days of October but not with the same frequency. There are few days that the usage is low, near 40.000 jobs per day (4 days), and some days that the usage is high, more that 80.000 jobs per day.

Regarding the daily cycle of the submission process, we observe that the value varies between 80.000 and 100.000 jobs per hour (summed for all days of October). The peak in utilization is observed during midday hours (13:00–17:00), while the lowest utilization is observed during night hours (00:00–06:00). Times refer to the GMT time zone.

5.1.3 Virtual Organisations

Regarding the VOs, there are 75 VOs participating in EGEE. The LCG/EGEE resources are not utilized to the same degree by all VOs. Figure 8, shows the percentage contribution of every VO to the total number of submitted jobs. The VOs whose percent-

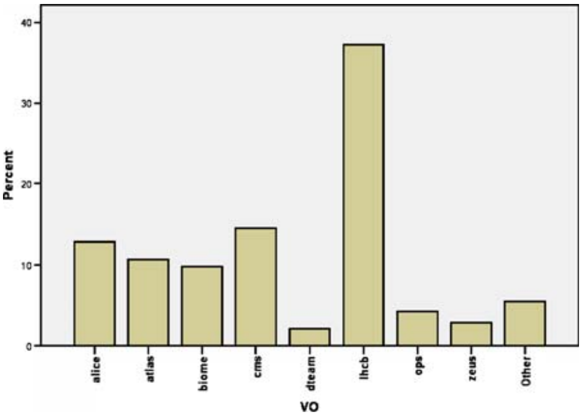


Fig. 8 Percentage of submitted jobs per VO (most active VOs, ≥1%)

age of contribution is less than 1% are categorized as «Other». The five most active VOs are:

- Lhcb VO, contributing 37.16% of the total number of jobs,
- Cms VO, contributing14.63% of the jobs,
- Alice VO, contributing 12.84% of the jobs,
- Atlas VO, contributing 10.72% of the jobs, and
- Biomed VO, contributing 9.82% of the jobs.

These top five VOs contribute 85.17% of the total traffic, while 89% of all the VOs contribute less than 0.1% of the total traffic each.

5.1.4 Resource Brokers

There are 59 RBs in total that serve the jobs in the LCG/EGEE environment and forward them to the appropriate CE. Figure 9 shows the percentage of the total traffic that is served by each RB. The RBs whose percentage of use is less than 1% are categorized as «Other». The five most active RBs are:

- rb107.cern.ch RB, handling 14.01% of the jobs,
- gridit-rb-01.cnaf.infn.it RB, handling 6.83% of the jobs,
- rb108.cern.ch RB, handling 5.16% of the jobs, and
- rb01.pic.es RB, handling 5.05% of the jobs.
- mu3.matrix.sara.nl, handling 5.01% of the jobs.

These top five RBs serve 36.06% of the total traffic, while 52.54% of the RBs serve less than 0.1% of the total traffic each.

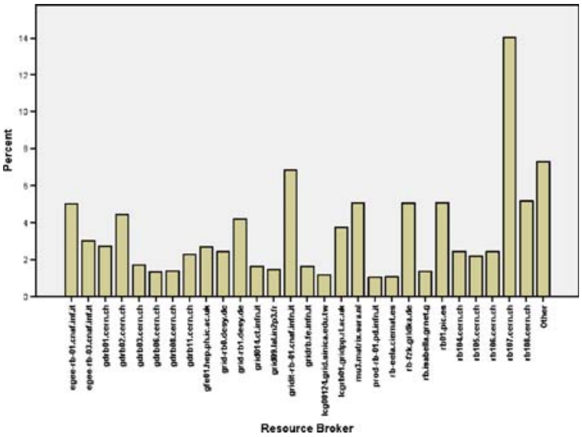


Fig. 9 Percentage of served jobs per resource broker (most active RBs, ≥1%)

5.1.5 Computing Elements

There are totally 343 CEs at which jobs can be executed in EGEE. In order to visualize how the workload is distributed among the various CEs of the EGEE infrastructure Fig. 10 presents the percentage of jobs served by CEs that served more than 1% of the total load. The remaining CEs are grouped together as "other". Also, there is a percentage of CEs that is categorized as "unknown" (we did nït have the related information). Figure 11 shows the exit types of the jobs that were categorized as "unknown". We can observe that a high percentage of these jobs fall in the exit type category REGISTERED–ABORT and also a smaller number in the categories UNDEFINED–ABORT and UNDEFINED–na. Therefore, CE "unknown" corresponds to jobs that were aborted or did not register correctly with their RB.

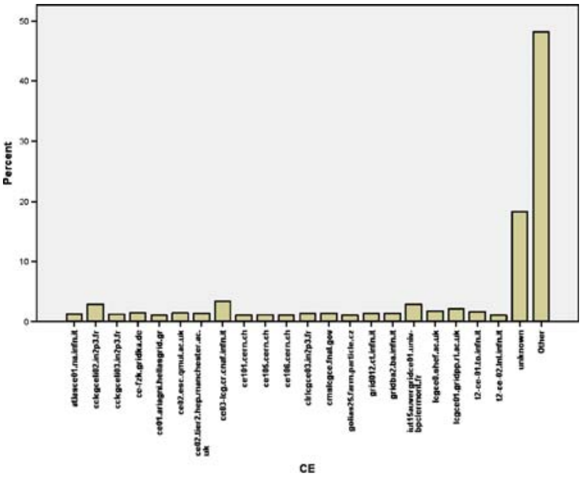


Fig. 10 Percentage of executed jobs per cluster-CE (most active Ces, ≥1%)

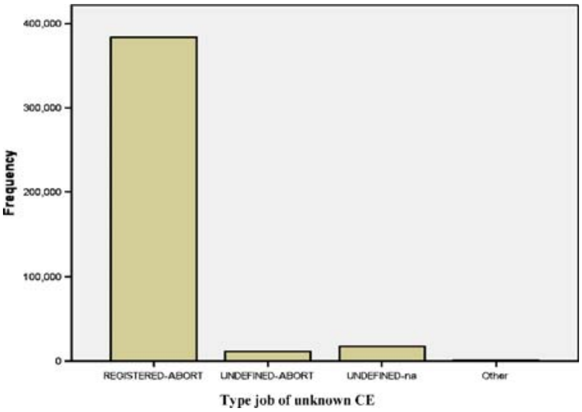


Fig. 11 Distribution of the job exit type for the 'unknown' CE

In order to identify whether there are any hot spots in the EGEE infrastructure we present the top five CEs:

- ce03-lcg.cr.cnaf.infn.it CE, with percentage of use 3.42%,
- cclcgceli02.in2p3.fr CE, with percentage of use 2.9%,
- iut15auvergridce01.univ-bpclermont.fr. CE, with percentage of use 2.88%.
- lcgce01.gridpp.rl.ac.uk, with percentage of use, 2.16% and
- lcgce0.shef.ac.uk, with percentage of use 1.72%

The top five CEs handle 13.08% of the total traffic, while 93.29% of the CEs serve less than 0.1% of the jobs each. Moreover, "other" appears with 48.18%, while the "unknown" appears with 18.32%.

5.2 Analysis of the Inter-Arrival Times and the Times of the Job at the Different States in the LCG/EGEE Environment

Table 2 shows the values of the minimum, the maximum, the mean, and the standard deviation of the job inter-arrival times, and the metrics (V_{10} to V_{19}) representing the time durations spent by a job at different states in the LCG/EGEE environment. In the first column of Table 2 we have the number of available entries-jobs (N) from which the statistics were computed. Due to the fact that jobs were discarded in different states we did not have the same N for all the used metrics. For example, for $V_{12}(D_1)$ we had N= 1.784.806 while for V_{18} (D_1+D_2+D_3+D_4) we had only N= 1.025.887. It seems that the jobs that remain in the system (and especially these that successfully finish D_3 and D_4) are the jobs that have smaller delay component values. For this reason, the addition of the mean values of metrics V_{12}, V_{13}, V_{15} and V_{16} (D_1+D_2+D_3+D_4) is larger than the mean value of the metric V_{18} (total time).

5.2.1 Job Inter-Arrival Times

Figure 12 illustrates the cumulative distribution function (cdf) of the inter-arrival times of the jobs submitted to the LCG/EGEE infrastructure. It must be noted that the Real Time Monitor tool, from which we obtained the measurements, records the corresponding time instances in seconds, which means that the real time values are rounded to the closest integer second. This determines the accuracy of our observations. We observe that with

Table 2 Statistical results for the metrics used

	N	Min	Max	Mean	SD
Inter-arrival time	980,581	0	60	1.25	1.52
V_{10} = registration_Time	2,166,574	1	14,679	14.90	79.58
V_{11} = match_Time	1,824,822	1	65,794	96.76	841.78
V_{12} = D_1 = ready _to_transfer_to_CE_Time	1,784,806	1	65,808	141.44	894.82
V_{13} = D_2 = transfer_Time	1,767,897	1	999,822	12,411.7	72,363.76
V_{14} = D_3 + D_4 = logmonitor_CE_total_Time	1,365,789	2	1,099,682	39,757.3	88809.95
V_{15} = D_3 = logmonitor_CE_queue_Time	1,170,688	2	1,099,673	16,899.1	61,007.08
V_{16} = D_4 = logmonitor_wn_Time	1,170,804	1	1,201,163	14,454.5	38,012.27
V_{17} = lrms_wn_Time	1,039,674	1	1,752,808	14,248.7	36,403.98
V_{18}=D_1+D_2+D_3+D_4 = total_Time	1,025,887	17	1,099,957	49,286.7	113,684.69
V_{19} = efficiency	1,042,871	0.01	1.00	0.519	0.33

N is the number of jobs from which the results were computed. Minimum, maximum, mean and std deviation values are measured in seconds.

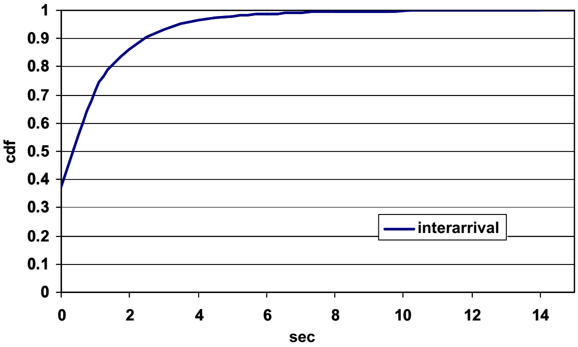


Fig. 12 Empirical cdf of the inter-arrival times

high probability (around 0.4) the inter-arrival time between two jobs is close to 0 s (the inter-arrival times represented as 0 s include the inter-arrival times up to 0.5 s). The maximum observed value was 60 s, and the probability of observing an inter-arrival time greater than 7 s was negligible. Since the inter-arrival times' standard deviation is quite small and close to its mean (Table 2) we can conclude that the inter-arrival process is quite close to a Poisson process.

5.2.2 Registration Times, Match-Making Times, Getting Ready to Transfer to CE Times and Transfer Times

In this section, we present results regarding the: registration (V_{10}), match-making (V_{11}), getting ready to transfer to CE (V_{12}=D_1), and transfer (V_{13}=D_2) times.

From Fig. 13 we observe that the match making times and getting ready to transfer to CE times exhibit similar behaviors, with the majority of the observed values lying in the range of a few seconds to a few tens of seconds, as can be deduced from the steep step-like form of the cumulative distribution function (cdf) in that region. Registration time has a small probability (~0.06) to be less than 5 s and a high probability (~0.9) to be between 6 and 50 s. Match making time has a small probability (~0.07) to be less than 7 s and a high probability (~0.85) to be between 8 and 66 s. Getting ready to transfer to CE time includes the registration time (pending state), the match making time (submitted state) and an additional delay in which the RB creates a wrapper script and prepares the job for submission to the chosen CE (waiting state). Since the match making time dominates the two other delay components, getting ready to transfer to CE times cdf is similar to the cdf of the match making times shifted by a few seconds (10 to 100). This observation can also be verified by comparing the mean and standard deviation of the getting ready to transfer to CE times with those of the match making times – Table 2 (their mean values differ by 50 s while the values of their standard deviation are almost equal).

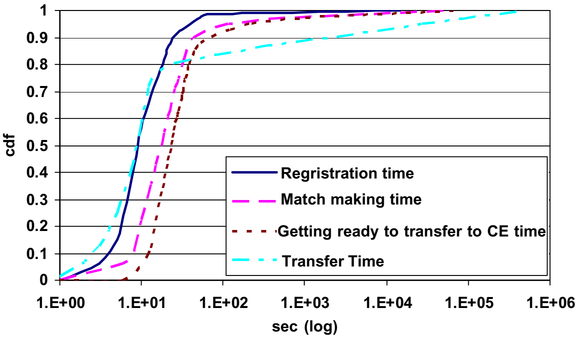


Fig. 13 Empirical cdfs of the registration times (V_{10}), match-making times (V_{11}), getting ready to transfer to CE times (V_{12}=D_1) and transfer times (V_{13}=D_2)

From Fig. 13 we see that the probability of observing a value for the transfer time smaller than 3 s is small (~0,06), while the probability of observing a value less than 80 s is high (~0.84). However, from the transfer times cdf we can see that this variable exhibits a heavy tail, and there is a considerable probability (~0.16) of observing values in the range of hundreds to millions of seconds. The difference of the transfer times (namely its heavy tail) with the variables analyzed in the previous paragraph can be also verified by the large value of the transfer times' standard deviation (Table 2). The large deviation in the transfer times is due to the large deviation in the sizes of the input data and the deviation of the propagation delays (due to geographic distribution of sites). Moreover, problems with the RBs, the clusters, and the middleware can affect this metric.

5.2.3 CE Register and Queuing Times, WN Execution Times, and Total CE Times

In this section, we present results regarding the delay introduced at the computing element (CE) of an LCG/EGEE cluster. More specifically, we present results for the CE register and queuing (V_{15}=D_3), the logmonitor WN execution (V_{16}=D_4), the lrms WN execution (V_{17}), and the CE total (V_{14}=V_{15}+V_{16}=D_3+D_4) times.

Comparing Fig. 14 and Fig. 13 we observe that the cdf of the variables presented in this section increase

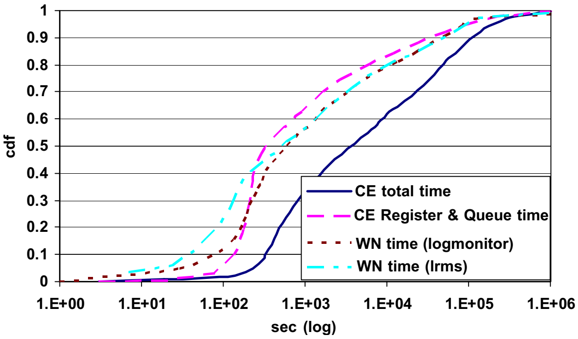


Fig. 14 Empirical cdfs of the total CE times (V_{14}), and the components that comprise it. We plot the cdfs of the CE queuing times (V_{15}=D_3) and the WN execution times according to logmonitor (V_{16}=D_4) and lrms (V_{17})

less rapidly than the cdf of the variables presented in the previous section. The results of Fig. 14 indicate that a job register and queuing time starts from 100 s and have a high probability to be less than 200 s. However, CE register and queuing times can also take large values and even reach 10^6 s.

The logmonitor WN times and the lrms WN times differ only slightly for values less than 1,000 s (specifically, lrms WN times have a higher probability to take smaller values) and converge for large values. They appear with equal probability (~0.56) to be less than 1,000 s, and can reach values of 10^6 s. Note that the difference between these two variables (logmonitor WN – lrms WN) corresponds to the time a job spends in the done state, which is the time required to transfer the output sandbox from the CE to the RB, indicating that the output sandbox requires only a small amount of time to be transferred.

CE total time includes the CE register and queuing and logmonitor WN time. There is a medium probability (~0.35) to observe a CE total time less than 1,000 s, while this variable can reach values of the order of 10^6 s. The mean value of the CE total times was measured to be equal to 38.7510^3 s and its standard deviation was 88.810^3 s.

5.2.4 Total Times and Efficiency

The results in Fig. 15 indicate that the job total times (V_{18}=D_1+D_2+D_3+D_4) exhibit almost similar behavior with the CE total times (CE register and queuing+ WN execution times=D_3+D_4). CE total times dominate the total delay, while getting ready to transfer to CE times (D_1) and transfer times (D_2) contribute negligibly to overall delay. The job total times are between 200 and 10^5 s with probability ~0.91, and can also take large values (10^7 s).

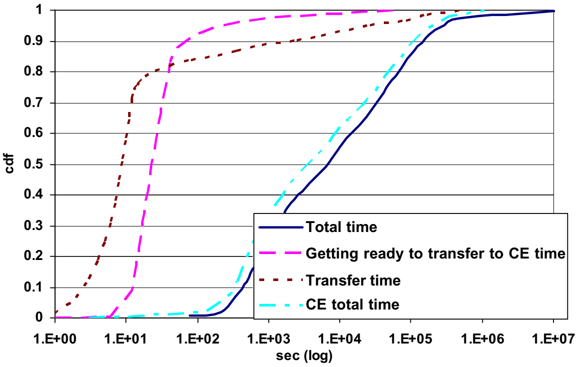


Fig. 15 Empirical cdfs of the total job times (V_{18}), and the constituent delays that comprise it. We plot the cdfs of the Getting ready to transfer to CE times (V_{12}), the transfer times (V_{13}) and the CE total times (V_{14})

Figure 16 illustrates the cumulative distribution function of the efficiency of the executed jobs, defined as the ratio of the WN execution time over the total time. We can observe that the cdf of the efficiency approaches a linear function. Therefore, a job submitted to the Grid has roughly equal probability to exhibit efficiency between 0 and 1.

6 Modeling of the Inter-Arrival Times and the Delay Components of a Job in the Lcg/Egee Environment

In this section we are interested in modeling the job arrival process and the delay components incurred by a job in the LCG/EGEE environment. As delay components we consider the four delay components introduced in Section 3 (Fig. 3).

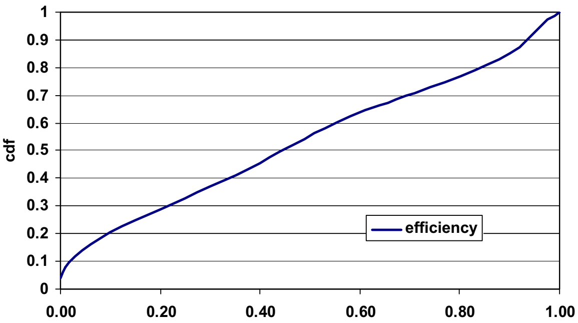


Fig. 16 Empirical cdf of the efficiency coefficient (V_{19})

6.1 Modeling the Job Arrival Process

Based on the descriptive statistics (Table 2) and the cumulative distribution function of the inter-arrival times (Fig. 12) we want to characterize the overall job arrival process in EGEE/LCG. Since the standard deviation of the inter-arrival times is quite close to their mean and the corresponding cdf does not seem to exhibit a heavy tail, a Poisson process is quite likely to accurately model the arrival process behavior. We have experimented with exponential distributions and parameters close to $^1/_{observedmean}$. Figure 17 shows the cdf of the inter-arrival times and the cdf of an exponential distribution with mean 1.6077 s. It is worth noting that the observed values were integer (our observations were rounded to the closest second). Therefore, in order to fairly compare the two distributions we have rounded to the closest integers the values produced by the proposed exponential distribution (referred to as rounded exponential model). After this adjustment, the exponential distribution with mean 1.6077 s resulted in a distribution with mean 1.15 s and standard deviation 1.57.

Figure 18 shows the probability–probability (P–P) graph of the rounded exponential model versus the actual data. Given the two CDFs, a P–P plot is constructed by pairing percentiles that correspond to the same value. A "good" fit corresponds to a P–P plot that is nearly linear. From this graph we can observe that a rounded exponential distribution with mean 1.6077 s can adequately model the job arrival process in the EGEE/LCG environment.

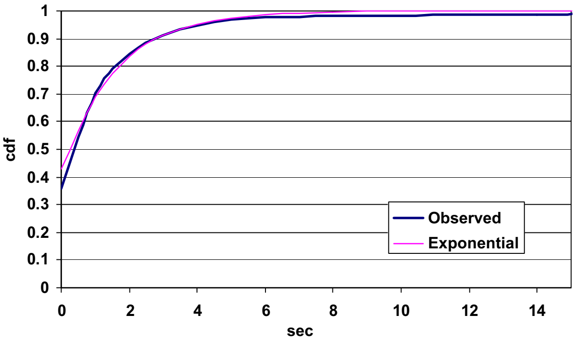


Fig. 17 Cdfs of the interarrival times of the actual observations and the examined exponential model

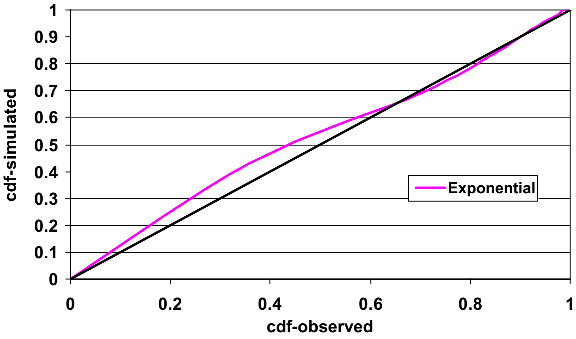


Fig. 18 Exponential vs observed inter-arrival times P–P plot

6.2 Getting Ready to Transfer to CE Times (D1) Modeling

From Table 2 we observe that delay component D_1 exhibits the smallest standard deviation among the delay components defined in Section 3. D_1 corresponds to the time a job stays at the pending, submitted and waiting states and thus is the time that the jobs spends in the UI and the RB before being transferred to a cluster. From Fig. 13 we observe that D_1 takes with high probability values close to the value 25 s (rises sharply between 15 to 30 s). Moreover, from Fig. 15 we see that the job total delay is dominated by CE times (D_3 and D_4). Therefore, we regard that the modeling of the getting ready to transfer to CE delay component (D_1) as a constant equal to 25 s is an acceptable approximation, since in any case it contributes the smallest delay to the total delay.

6.3 Transfer Times (D2) Modeling

The transfer times (V_{13}=D_2) presented in Fig. 13, as well as the CE register and queuing times (V_{15}=D_3) presented in Fig. 14, exhibit linear behavior at different stages (with different slopes) in the logarithmic scale. We investigate how a hyper-exponential process (in the general category of phase type distributions) and a phase lognormal distribution can fit the behavior of these delay components. We examined these two alternatives since the hyper-exponential distribution is widely used for modeling, while the phase lognormal distribution seems appropriate to model the linear behavior observed at different stages in the logarithmic scale.

Regarding the modeling of the transfer times (V_{13}= D_2), we considered three alternatives: (1) a three-phase hyper-exponential model (H3), (2) the sum of a deterministic and a lognormal r.v. and (3) a two-phase lognormal distribution.

We chose to use two phases for the lognormal model driven by the observation that the empirical cdf of Fig. 13 exhibits linear behavior in two different periods in the logarithmic scale. For the hyper-exponential model we used three phases driven by the observation that Fig. 13 exhibits one noticeable step and also has a heavy tail (assuming that we need one phase to model the step and at least two phases to model the heavy tail). For the hyper-exponential model we used the EMpht utility [15] to obtain the corresponding parameters.

The parameters that provide the best fits of the Transfer times (D_2) under the three models examined were found to be:

- Case (1) p_1=0.8635, p_2=0.0711, λ_1=9.377× 10^{-2} s^{-1}, λ_2=2.959×10^{-3} s^{-1}, and λ_3=1.4× 10^{-5} s^{-1},
- Case (2) p_1=0.83, constant=9, lognormal average= 8.8126 s, standard deviation=3.1227 s, and
- Case (3) p_1=0.83, a_1=2.027 s, d_1=0.7380 s, a_2= 8.8126 s, d_2=3.1227 s

Figure 19 shows the empirical cdf of the job transfer time, as presented in Section 5, and the cdfs we obtained for the proposed models, while Fig. 20 shows the corresponding P–P plots. From Fig. 20 we can observe that the two-phase lognormal distribution is the more accurate model while the hyper-exponential and the sum of a deterministic and a lognormal distribution converge to the observed data only for large values (in the heavy tail region). Since, in general, the heavy tail dominates the performance of this delay component, these two alternatives can also be considered as acceptable approximations.

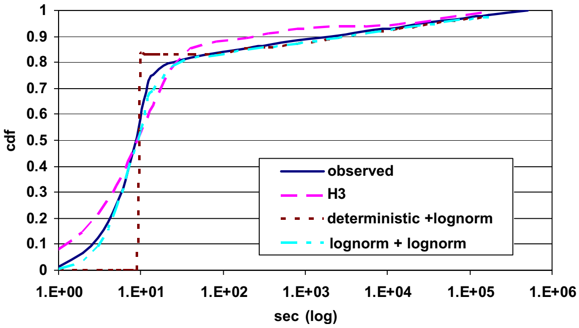


Fig. 19 Cdfs of the transfer times of the actual observations and the examined models

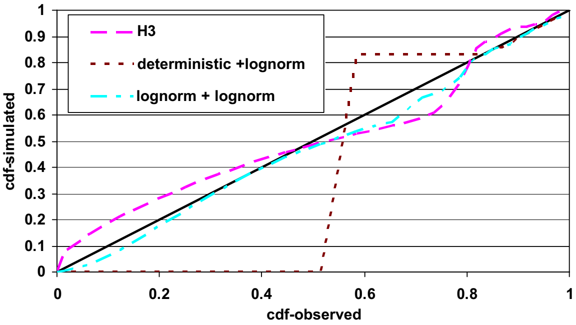


Fig. 20 P–P plots of the examined models vs the observed Transfer times

6.4 CE Register and Queuing Times (D3) Modeling

We considered again three alternatives for modeling the CE register and queuing times (V_{15}=D_3): (1) a three-phase hyper-exponential model (H3), (2) the sum of a deterministic and a lognormal r.v. and (3) a 2-phase lognormal distribution. The corresponding parameters that provide the best modeling accuracy with the observed data were found to be:

- Case (1) p_1=0.619, p_2=0.2408, λ_1=1.536× 10^{-3} s^{-1}, λ_2=2.71×10^{-4} s^{-1}, and λ_3=1.2×10^{-5} s^{-1},
- Case (2) p_1=0.32, constant=210 s, lognormal average=7.1093 s, standard deviation=2.85 s, and
- Case (3) p_1=0.34, a_1=5.13 s, d_1=0.211 s, a_2= 7.1093 s, d_2=2.85 s

Figure 21 shows the empirical cdf of the job CE register and queuing time as presented in Section 5 and the cdfs we obtained by the proposed models, while Fig. 22 shows the corresponding P–P plots. Similar to D_2, the two-phase lognormal distribution seems to be the best model for the CE register and queuing delay component, D_3, while the other two models are also good approximations since they accurately simulate the observed data for large values.

6.5 WN Execution Times (D4) Modeling

The WN execution times (V_{16}=D_4), as presented in Section 5 (Fig. 14), exhibit peaks at certain periods. We investigated how well a hyper-exponential random variable can fit this behavior. We used only this type of process since it is widely used in the literature to model execution times. More specifically, we considered two cases: (1) a three-phase (H3), and

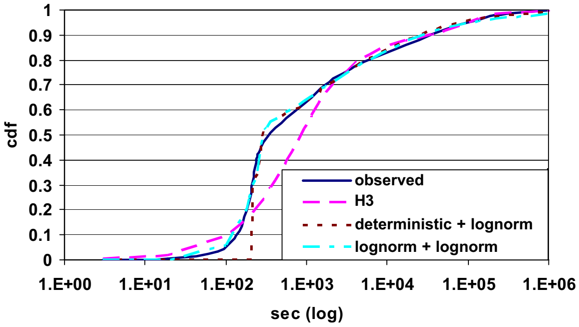


Fig. 21 Cdfs of the CE register and queuing times of the actual observations and the examined models

(2) a four-phase (H4) hyper-exponential distribution. We chose to use these values for the number of phases driven by the observation that Fig. 14 exhibits three to four steps. We used again the EMpht utility [15] to obtain the corresponding parameters:

- Case (1) p_1=0.3888, p_2=0.3635, λ_1=8.0314× 10^{-3} s^{-1}, λ_2=5.47×10^{-4} s^{-1}, and λ_3=1.46× 10^{-5} s^{-1}, and
- Case (2) p_1=0.3776, p_2=0.3614, p_3=0.1199, λ_1= 9.021×10^{-3} s^{-1}, λ_2=5.52×10^{-4} s^{-1}, λ_3=1.359× 10^{-5} s^{-1}, λ_4=1.559×10^{-5} s^{-1}.

Figure 23 shows the empirical cdf of the job WN execution time as presented in Section 5 and the cdfs obtained for the two hyper-exponential models, while Fig. 24 shows the corresponding P–P plots. Since the modeling accuracies obtained by the three- and four-phase models are similar, we conclude that a three-phase hyper-exponential model, which is the simpler of the two, is sufficient for modeling the WN execution times.

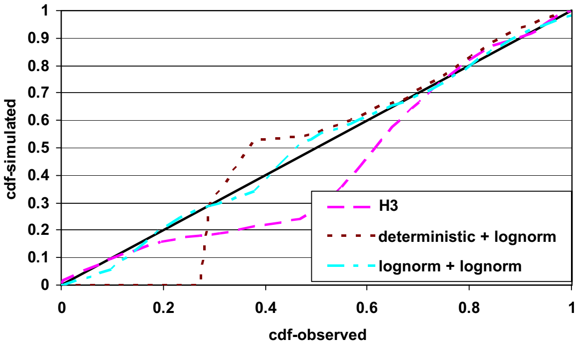


Fig. 22 P–P plots of the examined models vs the observed CE register and queuing times

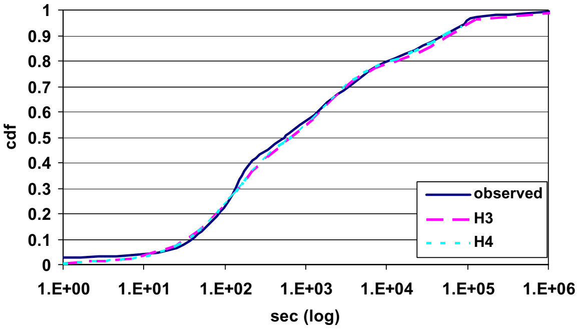


Fig. 23 Cdfs of the WN execution times of the original observations and the examined three- and four-phase hyper-exponential model

7 Efficiency of the Egee Environment and the Employed Super-Scheduling (RB) Algorithm

In Grids it is common to use a decentralized super-scheduling architecture to assign the jobs to the CEs. The current scheduling system in EGEE/LCG is a distributed version of the centralized resource broker (RB) originated from the EU DataGrid project, with multiple RB instances distributed at different regions/ countries. A Grid tries to create a virtual computing architecture for the execution of processes across geographically distributed resources. We want to evaluate the degree to which this objective is achieved and thus measure indirectly the efficiency of the EGEE environment and of the super-scheduling algorithm used. In order to do so, we compared the job total time delay metric (V_{19}) obtained from the actual measurements to that obtained in a hypothetical ideal super-cluster consisting of N CPUs. We wanted to find the number N of CPUs of a single super-cluster to which

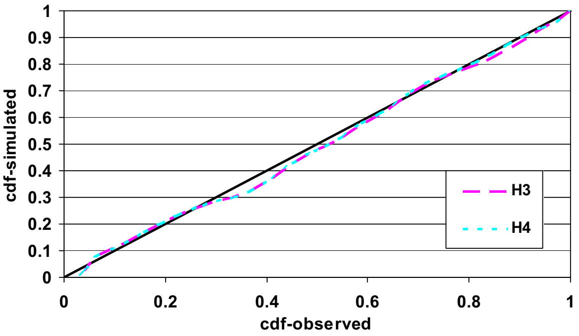


Fig. 24 3H and 4H vs observed WN execution times P–P plots

if we submitted the same workload we would obtain the same total delay performance with that obtained in the LCG/EGEE infrastructure (Fig. 25).

In the case of EGEE, the total time metric corresponds to the overall delay experienced by a job (the time spent at all job preparation or processing states), until completion. In the case of a single ideal super-cluster of *N* CPUs there is no preparation or communication overhead, but there is a delay introduced at the input queue (time to find a free CPU). Therefore, by comparing the performance of these two architectures we can evaluate the efficiency of the EGEE environment.

We used a static simulation with input the inter-arrival times and the WN execution times measured in the LCG/EGEE environment for the 1 month period of this study. We assumed a single super-cluster of *N* CPUs that uses a simple FCFS queuing strategy to store and process the arriving jobs. In the single super-cluster case we did not use any kind of access policy, but we just scheduled the jobs on a FCFS basis in order to obtain the bare performance of an ideal model. We estimated the number *N* of CPUs of the ideal super-cluster required to obtain the same total delay performance to be around 10,650 CPUs, while the maximum number of jobs that were queued in the single-cluster FIFO queue was around 32,000. Considering that for the period of our observation the *average* number of CPUs in the LCG/EGEE infrastructure was 31,228 (the number of CPUs varied dynamically with time) we estimate the efficiency of the EGEE environment to be equal to 10,650/31,228= 0.34. We believe this is a rather satisfactory figure, and it shows that the LCG/EGEE Grid infrastructure is used efficiently. It also indicates that the Grid computing concept can deliver to a satisfactory degree to its main promise, which is that of providing users with the ability to treat distributed computing, storage and other resources, as if belonging to a single computer.

8 Local Grid Infrastructure

The *kallisto.hellasgrid.gr* node is part of the HellasGrid [19] – EGEE infrastructure and has been a production

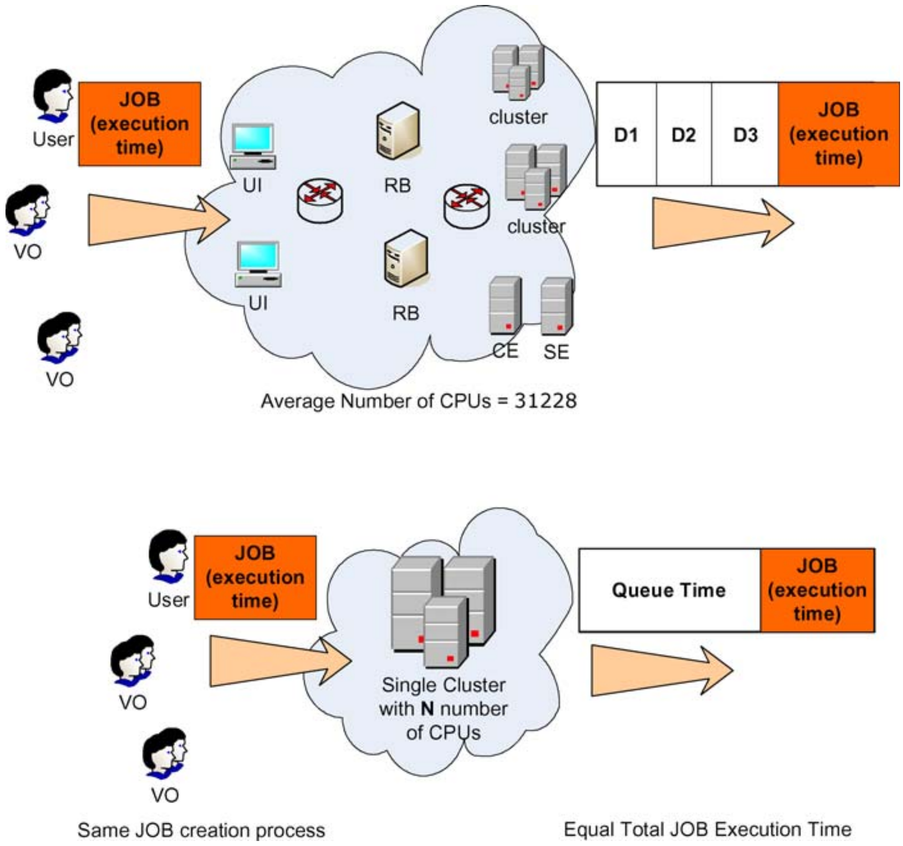


Fig. 25 Efficiency evaluation of the EGEE environment and the employed super-scheduling algorithm

site since February 1, 2006. The node's hardware consists of two HP racks with 64 servers with Intel Xeon CPUs at 3.4 GHz. There are 4 HP servers, each with two 80 GB SCSI hard disks running RAID1, 2 GB RAM and two processors that comprise the core elements of the EGEE site (CE, SE, Monitoring Box and Quattor server). The remaining 60 machines are the Working nodes, each of which has 80 GB SATA hard disk, 1 GB RAM and one processor. The racks also include a SAN that controls the 14 SCSI disks (300 GB each) of the main storage and an optical switch to connect the servers to the storage. The total capacity of the Storage Element is 4.2 TB. All servers are running Scientific Linux v.3 (SL3) and the deployed middleware is gLite middleware.

The *kallisto* node serves the following VOs: Dteam (development team), See (South Eastern Europe), Lhcb (large hadron collider beauty), Esr (earth science research), Atlas (A Toroidal LHC apparatus), Cms (compact Muon solenoid), Biomed (biomedical – drug discovery), Magic (MAGIC telescope), Compchem (computational chemistry) and Hgdemo (Hellas Grid demo). These VOs determine the queues in the MAUI configuration of the CE. MAUI [16] is a local scheduling engine that is used together with the PBS batch system [17]. The MAUI configuration of our node, reserves one slot for Dteam so that site functional tests can run without waiting. Previous LCG versions used queues that were based on the estimation of the job execution times, and thus our site configuration and the presented results differ from those reported in [6] in this respect.

The workload of the LCG/EGEE is solely composed of work-pile tasks termed *bags*. A *bag* is a collection of serial independent jobs that perform *no* communication and are not required to execute simultaneously or to be assigned to the same cluster/site. Jobs communicate, by writing output files to Grid Storage Elements or to the user's machine enabling other jobs to read and work on the generated data (forming "pipelines" of jobs). Each job requests a single processor and thus the degree of parallelism is one (trivial parallel tasks). A higher level scheduler fragments each *bag* into individual jobs and places them on (possibly) different sites. Therefore, observing the jobs executed or queued at a site we get a set of independent processes and thus we cannot see if there are additional jobs belonging to the same *bag* running on the same or other remote machines.

Using the log files of the CE (located under the directory /var/spool/pbs/server_priv/accounting/) we acquired information that was locally maintained in the *kallisto* node. The time period of the observation was three months (from February 1, 2006 until April 30, 2006), and the total number of jobs submitted during this period was 25,737. We parsed the log files and obtained the desired information in a form suitable for processing using statistical analysis tools. This was achieved by enhancing the Perl scripts (http://www.cs.huji.ac.il/labs/parallel/workload/swf.html) in order to match our metrics.

9 Statistical Analysis of the Kallisto Cluster

In order to obtain good models for the job submission process and the job characteristics at the cluster level, we performed a thorough statistical analysis of the logs that were stored in the *kallisto* cluster. Apart from examining the weekly and daily cycles of the workload we studied the job inter-arrival times, the job running times (worker node execution times), the CE queuing times of the jobs and the data transfers involved.

9.1 General Statistics

9.1.1 Submission Date and Time

Among the first things we looked at is whether the cluster is in use for all days of the week and for 24 h per day, or its utilization decreases during specific days (e.g., weekends, holidays) or specific daily periods (e.g., at nights). Figure 26 shows the number of submitted jobs during different days in a week, while Fig. 27 shows the number of jobs during different submission

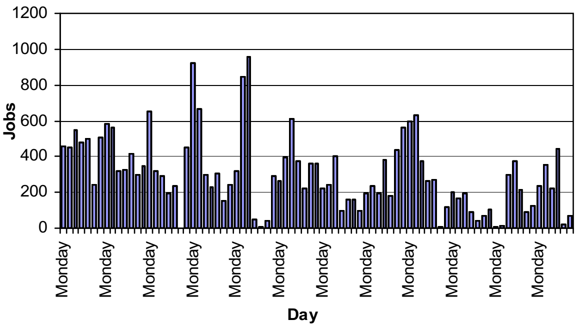


Fig. 26 Number of jobs per day

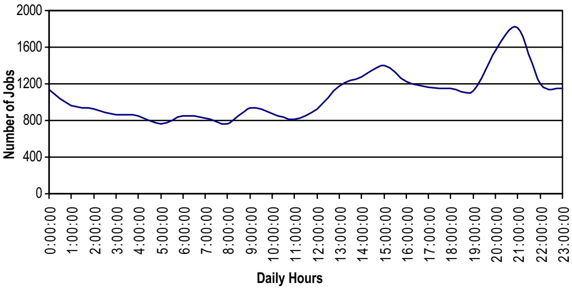


Fig. 27 Daily distribution of jobs

periods within a day. The graphs show that it is difficult to identify any patterns with respect to the date and time of the submission process. Jobs are submitted to the cluster during all days of the week and, contrary to our expectations, the cluster exhibits a gradual increase of its usage at the late hours of the day. These observations can be explained by the fact that users are active across different time zones, and they often schedule their jobs for later times, resulting in a rather even distribution of jobs across all weekly/daily cycles. In interpreting these results we also have to take into account the geographical position of Greece relative to that of the other EGEE users.

9.1.2 Job Execution Times

The node's resources are not utilized to the same degree by all VOs. The five most active VOs are listed in Table 3, while the other VOs had a relatively small number of jobs (~3% maximum). The Atlas VO contributed approximately 50% of the jobs submitted to our cluster during the duration of our observations.

Tables 4 and 5 show the mean and standard deviation of the CPU execution time and the worker node execution time which is the total running time (CPU + I/O), for all jobs and for each VO separately. Comparing these tables we observe that the standard deviations for the whole set of jobs and for each VO separately were almost equal. The difference between the averages of Table 4 and Table 5 correspond to the duration of the I/O operations and, since it is relatively small, we can deduce that the jobs sent to our cluster were CPU and not I/O intensive.

Table 3 Number and percentage of jobs per VO

VO	Atlas	Biomed	Dteam	Lhcb	Magic
Number of jobs	12548	3126	1315	4395	1929
Percentage	49	12	5	17	7

Table 4 Mean and SD of the CPU execution time (in seconds)

VO	Total	Atlas	Biomed	Dteam	Lhcb	Magic
Mean	15,321	16,139	24,656	13	8,511	2,736
SD	29,801	30,146	25,964	25	21,236	546

9.2 Analysis of the Inter-Arrival Times, CE Waiting and WN Execution Times at the Cluster Level

9.2.1 Job Inter-Arrival Times

In this section we present results on the job arrival process at our local node cluster. Figure 28 illustrates the cumulative distribution function (cdf) of the inter-arrival times for the jobs belonging to all the VOs and for the jobs belonging to the VO Atlas, which is the one that contributed the majority of jobs to our node. It is worth noting that site functional tests from the Dteam VO are performed every 3 h (10,800 s) [11], posing an upper limit on the inter-arrival times.

To study the way job arrivals are distributed with respect to the time of day, we divided the 24 h of a day into three 8-h periods, and present the corresponding graphs in Fig. 29. We observe that the cdfs have the same shape for the different time periods, while jobs that arrive between 4:00 P.M. and 12:00 P.M. have a slightly higher frequency when compared to the other two investigated periods (these results are in agreement with the results presented in Fig. 27).

9.2.2 Self-Similarity

Self-similarity deals with burstiness, and is a measure of the degree to which a process includes periods of increased activity and periods of little or no activity. Self-similarity implies correlation across different

Table 5 Mean and SD of the WN execution time (in seconds)

VO	Total	Atlas	Biomed	Dteam	Lhcb	Magic
Mean	15,400	16,150	24,682	17	8,532	2,749
SD	29,850	30,163	25,978	27	21,258	567

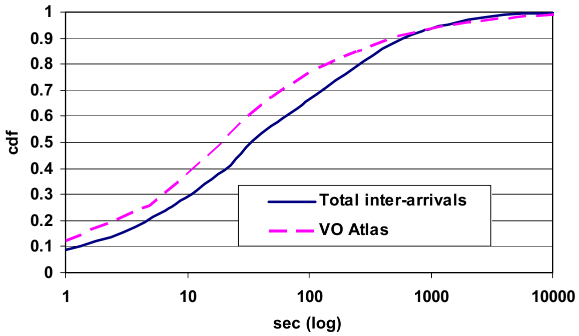


Fig. 28 Empirical cdfs of the inter-arrival times for the jobs belonging to all the VOs and for the VO Atlas

time scales, in the sense that what happens at the present time is correlated to what happened in the recent and also in the more distant past.

One way for checking if a process is self-similar is the rescaled range method (or R/S) originally used by Hurst. It produces a log-log plot of the R/S statistic versus the number of points of the aggregated series. This plot should be a straight line with the slope being an estimation of the Hurst exponent. We computed the Hurst parameter (H) of the inter-arrival times using a variety of methods (aggregate variance, R/S, periodogram, absolute moments, variance of residuals, Abry–Veitch estimator, Whittle estimator; [18]). For the above methods we also obtained the correlation coefficient, which gives us a reliability factor for the H estimate (values higher than 0.9 should be sufficient). The higher correlation coefficient (99.31%) was computed using the R/S method, indicating that this was in our case the most reliable method for estimating the Hurst parameter. Using that method, the Hurst parameter of the job arrival process at our local cluster was found to be H=0.684 (Fig. 30). The Poisson process, which is not self-similar as indicated by its memoryless property, has H=0.5. When $0.5\leq H\leq 1$, as is true in our case, the process has positively correlated consecutive steps. Thus, we conclude that the job arrival process in our local cluster exhibits self-similarity/long-range dependence.

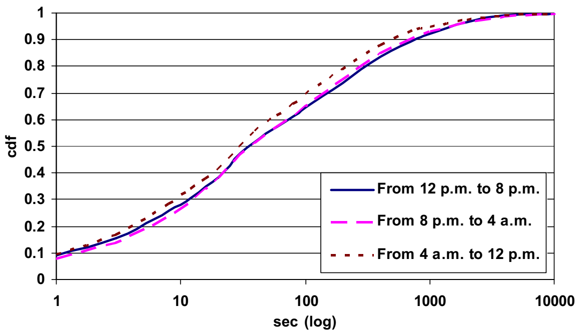


Fig. 29 Empirical cdfs of the inter-arrivals per periods of day

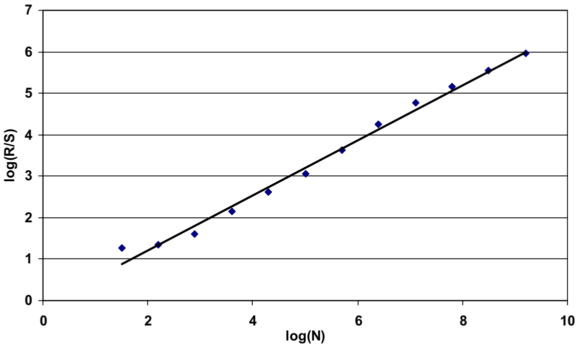


Fig. 30 Hurst parameter estimation using the R/S method

9.2.3 CE Queuing Times

We present results regarding the CE queuing times of the jobs, defined as the time between the acceptance of the job by the local resource management system (LRMS) and the time it starts execution on a WN. When a job is accepted by the CE gatekeeper, it is forwarded to a local scheduler (LRMS) that ensures the low queuing times of the accepted jobs. Our system in particular uses a MAUI-PBS LRMS whose configuration employs a separate queue for each VO and reserves one time slot for the Dteam.

Figure 31 shows the empirical cdfs of the *kallisto* CE queuing times and the EGEE CE register and queuing times (presented in Section 5.2.3). The

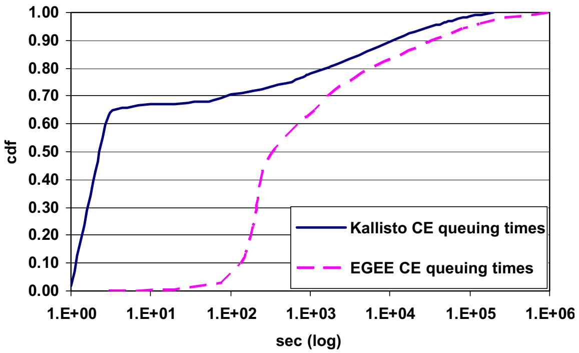


Fig. 31 Empirical cdfs of the *Kallisto* CE queuing times and the EGEE CE register and queuing times

Table 6 Mean and SD of the kallisto CE queuing times (in seconds)

VO	Total	Atlas	Biomed	Dteam	Lhcb	Magic
Mean	5,503	3,412	9,731	236	2,450	867
SD	19,851	13,809	19,774	19,851	11,223	4,625

results of Fig. 31 indicate that a job stays in a *kallisto* CE queue for less than 2 s with large probability (~0.7). There are also, however, a few jobs that stay in their queue for a long time period due to congestion, general or specific problems of our system. The mean and the standard deviation of the waiting time for all the VOs together and separately for each VO are shown in Table 6. We can observe that Dteam experiences the lower average delay, while Biomed the highest. This is because of the local queues priority policies and the fact that Dteam's jobs require the smallest CPU times (Table 4), while Biomed's jobs are CPU-intensive and thus exhibit the highest delays.

The difference between *kallisto* CE queuing and EGEE CE register and queuing time corresponds to the time required by the CE gatekeeper to accept and match the job to a local UID and forward it to the LRMS. This period is included in the empirical cdf reported in previous sections that reported measurements for the overall EGEE infrastructure (i.e., at the Grid level), while in the *kallisto* measurements (i.e., at the cluster level) we start measuring the queuing time at the time the job enters the LRMS.

9.2.4 Job Worker Node Execution Time

The job WN execution time is the actual execution time of a job including the I/O time. When users submit their jobs they also provide an estimate of the job run time, but this is usually a very loose overestimate of the job run time. In Fig. 32 we give the cdf of the "actual" *kallisto* WN execution times and the EGEE WN execution times (presented in Section 5.2.3). The difference between the EGEE and *kallisto* WN execution times can be explained by the different number of VOs that these measurements correspond to (the results presented in Section 5.2.3 correspond to the whole EGEE infrastructure that served 75 VOs, while the *kallisto* cluster served only 11 VOs during the period of our observations). Moreover, the most active VO in the case of *kallisto* was VO Atlas, which was the third most active in the case of the whole EGEE measurements.

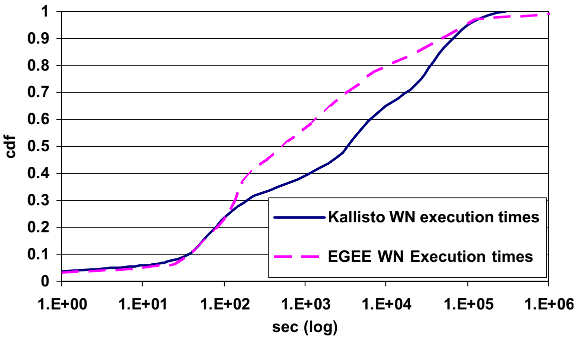


Fig. 32 Empirical cdf of the *Kallisto* WN execution times and the EGEE WN execution times

10 Modeling Jobs at the Cluster Level (Kallisto)

10.1 Modeling the Job Arrival Process

We considered and evaluated four different models for the job arrival process at the cluster level:

(a) *Non-Homogeneous Poisson Process (NHPP) model*

Taking into account the variations of the job arrival rate with respect to the days of a week (Fig. 26) and the hours of day (Fig. 27) we initially investigated if the job arrival process can be modeled as a non-homogeneous Poisson process (NHPP). Using the results of Fig. 27, we defined a stepwise function for $\lambda(t)$, obtained by averaging over all days in our observation period the number of job arrivals observed during each 1 hour interval of a day.

(b) *Hyper Exponential model*

We considered two cases: (1) a two-phase (H2) and (2) a three-phase hyper-exponential distribution (H3). To find suitable parameters [three parameters in case (1) and five parameters in case (2)] we used the EMpht program [15] to obtain the following parameters:

- Case (1) p_1=0.37, λ_1=1.37×10^{-3} s^{-1}, λ_2=4.65×10^{-2} s^{-1}, and λ_3=1.46×10^{-5} s^{-1}, and
- Case (2) p_1=0.444, p_2=0.457, λ_1=5.38×10^{-2} s^{-1}, λ_2=9.07×10^{-2} s^{-1}, λ_3=5.12×1^{0-3} s^{-1}

(c) *Markov Modulated Poisson Process (MMPP) model*

We investigated two MMPP models: (1) a three-state MMPP (3MMPP) and (2) a four-state MMPP (4MMPP). To find suitable parameters [four parameters in case (1) and nine parameters in case (2)] we used the program found in (http://www.liacs.nl/~hli/gwm/index.htm) to obtain the following MMPP parameters that best fit our measurements.

- Case (1) 3MMPP: $\sigma_{12}=6\times10^{-3}$ s^{-1}, $\lambda_1=98\times10^{-3}$ s^{-1}, $\sigma_{21}=0.45\times10^{-3}$ s^{-1}, $\lambda_2=4.1\times10^{-3}$ s^{-1}
- Case (2) 4MMPP: $\sigma_{12}=3.2\times10^{-3}$ s^{-1}, $\sigma_{13}=4.3\times10^{-3}$ s^{-1}, $\lambda_1=139\times10^{-3}$ s^{-1}, $\sigma_{21}=0.1\times10^{-3}$ s^{-1}, $\sigma_{23}=0.2\times10^{-3}$ s^{-1}, $\lambda_2=0.9\times10^{-3}$ s^{-1}, $\sigma_{31}=0.45\times10^{-3}$ s^{-1}, $\sigma_{32}=0.55\times10^{-3}$ s^{-1} and $\lambda_3=11.9\times10^{-3}$ s^{-1}.

(d) *Pareto-Exponential model*

We have chosen to also examine a truncated Pareto distribution model with $X_{\max}=10{,}800$ s since we know that the job inter-arrival times are upper-bounded by 3 h (the times of the Dteam periodic submissions of site functional tests). For the other parameters we conducted a number of trials and concluded in the following values for our case: mean $\lambda=18$ arrivals per second for busy periods, mean duration $1/\mu=22.5$ s of the busy periods, $a=0.48$ and $X_{\min}=32$ s.

Figure 33 shows the cdf of the inter-arrival times as presented in Section 9.2.1 and the cdfs we obtained from the traces of the four proposed models. Figure 34 shows the probability–probability (P–P) graphs of the better performing H3, 3MMPP and Pareto-exponential models versus the actual measurements.

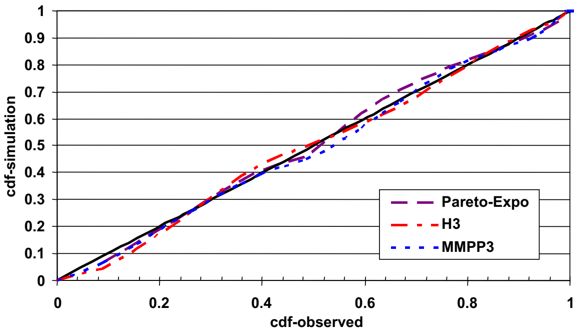


Fig. 34 P–P functions of the proposed models

From the above graphs we can conclude that the proposed Pareto-exponential model generates traces that are very close, according to the P–P plot, to those observed in our cluster. H3 and 3MMPP models simulate also satisfactorily the job arrival process. However, the Pareto-exponential model is simpler, more concise and more intuitive than the other proposed models, since it is based on a smaller number of parameters, and seems to correspond to actual VO behavior.

As expected, by increasing the number of phases in the hyper-exponential process the accuracy of that model also improves. This is, however, only due to fact that by adding complexity (more states) to the hyper-exponential model, we can approximate any process. Similarly, by increasing the states in the MMPP process we obtain better accuracy. However, this is a “mechanical” and not an intuitive way to model the inter-arrival process.

We have also computed the Hurst parameter for the four models. Only the Pareto-exponential and the MMPP models experience long-range dependence ($H=0.58$ for the Pareto-exponential, $H=0.62$ for 2MMPP and $H=0.64$ for 3MMPP with confidence

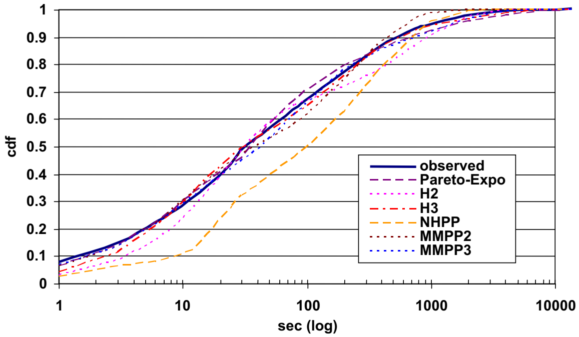


Fig. 33 Empirical cdf and the cdfs of the proposed models for kallisto inter-arrival times

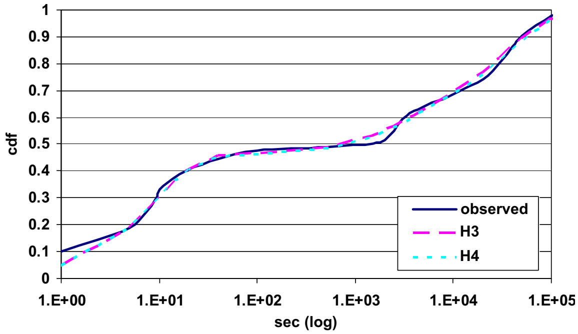


Fig. 35 Empirical cdf and the cdfs of the proposed models for the *kallisto* WN execution times

levels higher than 99%), while the models (a) and (b) have a Hurst parameter of 0.5. Given that the MMPP model requires a large number of parameters, the Pareto-exponential model seems to be more appropriate for modeling the job arrival process at a Grid node, since it also fits very well the real traffic in our observations and exhibits long-range dependence as indicated by the calculated Hurst parameter.

10.2 Modeling the Job WN Execution Times

The worker node execution times, presented in Section 9.2.3 (Fig. 32), exhibit peaks at certain values. Execution times differ in their nature from the inter-arrival times since they do not depend on the human factor, and thus it is difficult to find a physical explanation for their behavior. Therefore, our criteria for modeling WN execution times are more relaxed. We investigated how a hyper-exponential process can fit the observed behavior. More specifically, we considered two cases: (1) a three-phase (H3) and (2) a four-phase (H4) hyper-exponential distribution. We chose to use these values for the number of phases driven by the observation that Fig. 32 is of a stepwise form with three noticeable steps. We used again the EMpht utility to obtain the corresponding parameters:

- Case (1) p_1=0.3290, p_2=0.2805, λ_1=1.0731×10^{-2} s^{-1}, λ_2=2.65×10^{-4} s^{-1}, and λ_3=2.1×10^{-5} s^{-1}, and
- Case (1) p_1=0.3270, p_2=0.2805, p_3=0.14, λ_1=1.0531×10^{-2} s^{-1}, λ_2=2.65×10^{-4} s^{-1}, λ_3=2.4×10^{-5} s^{-1}, and λ_4=1.8×10^{-5} s^{-1}

Figure 35 shows the empirical cdf of the job WN execution time as presented in Section 9.2.3 and the cdfs we obtained from the traces of the two hyper-exponential processes.

By comparing the results presented in this section and the corresponding fitting accuracy at the EGEE Grid level (Section 6.5) we can conclude that a three-phase hyper-exponential distribution is in both cases adequate to model the WN execution times, while the increase from three to four phases improves slightly the modeling accuracy. As stated above, the difference between the EGEE and *kallisto* cases is the result of the different number and content of VOs that these measurements correspond to.

11 Conclusions

A thorough analysis of the job arrival process and the time durations jobs spend at different states in the EGEE/LCG environment was presented. The job inter-arrival times at the Grid level were found to match very well with a rounded exponential distribution. We defined four delay components of the total job delay, and proposed and validated probabilistic models for each component separately. We also evaluated the efficiency of the Grid environment and calculated that we would obtain similar performance if we submitted the same workload to a super-cluster having 34% of the total average number of CPUs participating in the EGEE/LCG infrastructure.

We also presented a comprehensive and thorough traffic analysis of our local Grid cluster. At the cluster level the job arrival process exhibits long-range dependence as indicated by the Hurst parameter calculated. We proposed several models for the job arrival process at the cluster level. The custom "Pareto-exponential" model is simple, intuitive, and matches well with the actual measurements. This model incorporates exponential job inter-arrival times during busy periods of exponential duration (corresponding to a single VO's job submissions). The times between VO busy periods are distributed according to a truncated Pareto distribution. Finally, a three-state hyper-exponential process was found to be sufficient for modeling the job execution times.

Acknowledgment This work has been supported by the EC through the IP Phosphorus IST project (http://www.ist-phosphorus.eu/). K. Christodoulopoulos has been supported by the Greek General Secretariat for Research and Technology –GSRT– via the PENED project. The authors would like to thanks Gidon Moont from Imperial College London for his help with the logs of the Real Time Monitor tool. The *kallisto.hellasgrid.gr* Grid infrastructure is partially funded by the EGEE project [11].

References

1. Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure, 2nd edn. (Morgan Kaufman, San Francisco, 2003)
2. Feitelson, D.: Workload modeling for computer systems performance evaluation", http://www.cs.huji.ac.il/~feit/wlmod
3. Cirne, W., Berman, F.: A Comprehensive Model of the Supercomputer Workload. Proceedings of the 4th IEEE Annual Workshop on Workload Characterization (2001)

4. Song, B., Ernemann, C., Yahyapour, R.: Parallel Computer Workload Modeling with Markov Chains. Proceedings of the 10th JSSPP (2004)
5. Denneulin, Y., Romagnoli, E., Trystram, D.: A Synthetic Workload Generator for Cluster Computing. Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS) (2004)
6. Medernach, E.: Workload Analysis of a Cluster in a Grid Environment. Proceedings of the 11th JSSPP (2005)
7. Li, H., Muskulus, M., Wolters, L.: Modeling Job Arrivals in a Data-Intensive Grid. Proceedings of the 12th JSSPP (2006)
8. Real Time Monitor: http://gridportal.hep.ph.ic.ac.uk/rtm/
9. Li, H., Heusdens, R., Muskulus, M., Wolters L.: Analysis and Synthesis of Pseudo-Periodic Job arrivals in Grids: A matching Pursuit Approach. Proceedings of CCGrid07 (2007)
10. Nurmi, D., Mandal, A., Brevik, J., Koelbel, C., Wolski, R., Kennedy, K.: Grid Scheduling and Protocols – Evaluation of a Workflow Scheduler Using Integrated Performance Modelling and Batch Queue Wait Time Prediction. Proceedings of Supercomputing (2006)
11. The EGEE project homepage: http://public.eu-egee.org/
12. gLite-3 user's guide: https://edms.cern.ch/file/722398//gLite-3-UserGuide.pdf
13. Job description language: How To. Publicly available at http://www.infn.it/workload-grid/docs/DataGrid-01-TEN-0102-0_2-Document.pdf
14. Fischer, W., Meier-Hellstern, K.: The Markov-modulated Poisson process (MMPP) cookbook. Perform. Eval. **18**(2), 149–171 (1993)
15. The EMpht program: publicly available at http://home.imf.au.dk/asmus/pspapers.html
16. Maui Scheduler: http://supercluster.org/maui
17. Open PBS: http://www.openpbs.org/
18. Karagiannis, T., Faloutsos, M., Molle, M.: A User-Friendly Self-Similarity Analysis Tool. ACM SIGCOMM Computer Communication Review (2003)
19. HellasGrid task force: http://www.hellasgrid.gr/