034115

PHOSPHORUS

Lambda User Controlled Infrastructure for European Research

Integrated Project

Strategic objective:
Research Networking Testbeds

Deliverable reference number <D.5.1>

Job Demand Models

Due date of deliverable: 2006-12-31
Actual submission date: 2006-12-31
Document code: <Phosphorus-WP5-D.5.1 >

Start date of project:
October 1, 2006

Duration:
30 Months

Organisation name of lead contractor for this deliverable:
IBBT

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	PU
PP	Restricted to other programme participants (including the Commission	
RE	Restricted to a group specified by the consortium (including the	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract

Grid computing is an emerging computing paradigm that exploits networked computers to create a virtual architecture for the distributed, resource-transparent execution of computational tasks. Grids use job scheduling and resource management to establish a global architecture for sharing computing and storage resources across geographically separated sites. Next generation networks will bring this grid functionality to the optical layer, allowing a more tightly integrated optical grid environment to exist, in which higher performances can be achieved.

Due to the high equipment cost involved in the research of these optical grids if actual hardware were to be used, *simulation techniques* are often put forward as a viable alternative. In order to obtain accurate and useful results from these simulations, it is important that a realistic grid job load is used as input for the simulation. To this end, *realistic models of the job submission process* are to be established. This document will detail several of these **models**, and the **process to extract the model parameters from actual grid log traces**. This approach guarantees a very flexible, analytical job submission model, which does not require data from log traces to be fed to the simulator at the time of simulation, yet providing a very **realistic approximation of the real life grid job submission pattern.**

Note that although the actual traces used in the studies presented here, it is to be expected that at least some of the applications to be deployed in the Phosphorus test bed will similar characteristics. Moreover, the modelling techniques and fitting methodologies will be applicable to Phosphorus traces as well.

Table of Contents

List of Figures

List of Tables

0 Executive Summary

Grid computing is becoming an increasingly important tool for handling job execution. Grids offer a transparent interface to geographically scattered computational and storage resources. Job scheduling and resource management – including network resource management – form the basis of any grid architecture. The network technology of choice to supply the necessary bandwidths to enable data-intensive applications obviously is the optical (Dense) Wavelength Division Multiplexing (DWDM) networks. The integration of the optical network awareness in the Grid control and scheduling mechanisms is a major research challenge addressed within Phosphorus project and esp. WP5.

To allow detailed studies of algorithms and protocols governing optical Grids, simulation and analysis are the tools of choice (given the high costs associated with actually deploying an optical Grid). To allow such simulation/analysis studies, a mandatory prerequisite is the availability of realistic models of the jobs to be handled by the Grid. Thus, we need quantitative data on the **job submission process, both on the inter-arrival times (IATs) and the execution times** of the jobs. In this deliverable, such data **measured on real-life Grids** is discussed and used to verify which **analytical models** reflect the same characteristics and thus are suitable candidates to use in simulation/analysis studies. We hereby also discuss the **fitting methodology**, showing how to fit the models to actual measurement data. In addition, sample series are generated by **software implementing the models** and validated by comparison to the measurement data.

Measurement data was gathered on two different levels of aggregation: we collected traces of jobs arriving to a complete European Grid, and also collected job arrival data at individual Grid sites (ie. local clusters). The models considered comprised:

- Poisson process (the classical exponential inter-arrival time distribution with mean rate λ)
- Non-homogeneous Poisson Process (NHPP; a Poisson process with time-varying rate λ(t))
- Hyper-exponential Process (HP; i.e. a phase type process, where the IAT is the sum of multiple Poisson phases)
- Markov-Modulated Poisson Process (MMPP; process probabilistically moving between states, each of them being a Poisson process)
- Pareto-Exponential Model (Busy periods with exponential duration and exponential job IATs during them. The times between the busy periods are distributed according to a truncated Pareto distribution)

From our fittings we conclude that:

- Job **inter-arrival times** on the observed Grid level can be successfully modelled by a Poisson process, but on the Grid site level (eg. Kallisto traces) the long range dependency needs to be taken into account and HP, MMPP or Pareto-Exponential models need to be used.

- For the **job execution times**, we achieved the most satisfactory results with a (3 phase) hyper-exponential process.

1 Introduction

1.1 Motivation

Grid computing is an emerging computing paradigm that exploits networked computers to create a virtual computer architecture for the distributed execution of computational tasks. Using job scheduling and resource management, a global architecture is established for sharing computing and storage resources across geographically separated sites.

To study Grids without having to actually deploy them, adequate models for simulation and analysis are needed. The exact job arrival times, execution times, and data sizes in Grids are to a large extent unknown and thus better modelled probabilistically. The existence of good probabilistic models for the job arrival process and the job characteristics is important for the improved understanding of grid systems. Such models would facilitate the design and dimensioning of grid systems, the prediction of their performance, the evaluation of new scheduling strategies, and the design of a QoS framework for Grid users.

To obtain accurate and useful results, realistic job submission processes and job resource requirements need to be realistic. To this end, two different approaches can be applied:

a) data from actual grid log traces can be used as input for the simulator, or
b) an analytical model can be formulated and embedded in the simulator.

The former approach, while guaranteeing realistic submission patterns are being used for simulation, has the intrinsic downside of being very static. Using an analytical model to generate job loads adds flexibility: the influence of the different parameters in the model can be studied, and possibly some results (e.g. performance parameters) can be calculated using analytical methods.

This deliverable aims at describing the models and how to obtain model parameters from real world traces to produce models reflecting the observed job submission behaviour of existing applications.

1.2 Related Work

A great deal of work has appeared in the literature on job characterization and modelling [DF-B] for single parallel supercomputers [**CB01**], [**SEY04**], but the corresponding work in the area of Grid computing is quite limited [EM05], [LMW06]. Medernach [EM05] analyzed the workload of a LCG/EGEE cluster, proposing a 2-dimensional Markov chain for modelling user behaviour in a Grid environment. The user shifts between login and logout states and submits jobs when in the login state. The results indicate that this model can satisfactorily approximate the submission behaviour of a single user.

Taking a different approach, Li et al [LMW06] used the LCG Real Time Monitor [RTM] to collect data from Resource Brokers (RBs) participating in the EGEE project [EGEE], and propose models for the job arrival process 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 sufficient number of states are capable of simulating the job traffic at the three examined levels. However, the proposed models are not intuitive enough, and they do not provide an easily adaptable or extensible way for profiling arrival processes in a general Grid environment.

2 Statistical Analysis

2.1 Motivation

The aim of this section is to provide an overview of the available data, and to describe the general behaviour of the different grid processes. These insights will then be used to propose adequate models for our data traces. This section first describes the relevant infrastructure details, such as middleware processes and deployed hardware. This is followed by a discussion of the actual job traces, where statistics at multiple levels in the grid (e.g. virtual organization, cluster level or grid level) are provided.

2.2 Infrastructure

A detailed statistical analysis of data cannot be complete without a thorough understanding of how this data was captured. We therefore provide some insight in the way submitted jobs are treated throughout their lifetime in the grid infrastructure. The first two subsections describe the relevant middleware processes, detailing the job flows and the various timings which are available for our statistical analysis. In the final subsections, we provide some details on the hardware infrastructure where the job traces were collected.

2.2.1 LCG/EGEE

The EGEE project [EGEE] aims at providing researchers with access to a geographically distributed Grid computing infrastructure, available 24 hours a day. It focuses on maintaining the gLite middleware [GLITE3] and on operating a large computing infrastructure for the benefit of a large and diverse research community.

The World wide LHC Computing Grid Project (LCG) [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 LHC, which is being constructed at the European Laboratory for Particle Physics (CERN), will be the world’s most powerful particle accelerator. 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 this study, there were totally 39697 CPUs and about 5 Petabytes of storage, while the total average number of available CPUs was 31228 [GOC].

In 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. The users have to belong to a VO in order to be able to use the LCG/EGEE infrastructure. A list of existing VOs in the LCG/EGEE is available at [LCGVO].

While the measurements are not performed on the actual Phosphorus test bed (which is not ready yet), we do believe the data may be representative of at least some of the applications that will be deployed within Phosphorus. Also the job submission and scheduling/execution process as discussed below will be similar.

2.2.2 LCG/EGEE environment and job flow

Generally, a user cannot submit a job directly to a cluster (site); instead, the user has to login to a local User Interface (UI) and submit a job. The description of the job is written in a specific format (JDL – job description language). This is forwarded to the corresponding Resource Broker (RB) where the matching process is performed [EGEE]. RB runs the services of the Workload Management System (WMS) that intercommunicates with the Information System (IS - provides information about the Grid resources and their status). The RB takes into account the job description, the related VO and the available global traffic load information and decides whether or not and where to forward the job. Users, when submitting a job, give a rough estimation of the maximum running time of their job, but this value is usually overestimated and is often 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 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 retrieve of the output of the job is shown in Figure 1 [EM05]. Figure 2 presents the various states of a job in the LCG/EGGE environment. These states come from the gLite 3 user's guide [GLITE3] 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 execution in the LCG/EGEE environment.

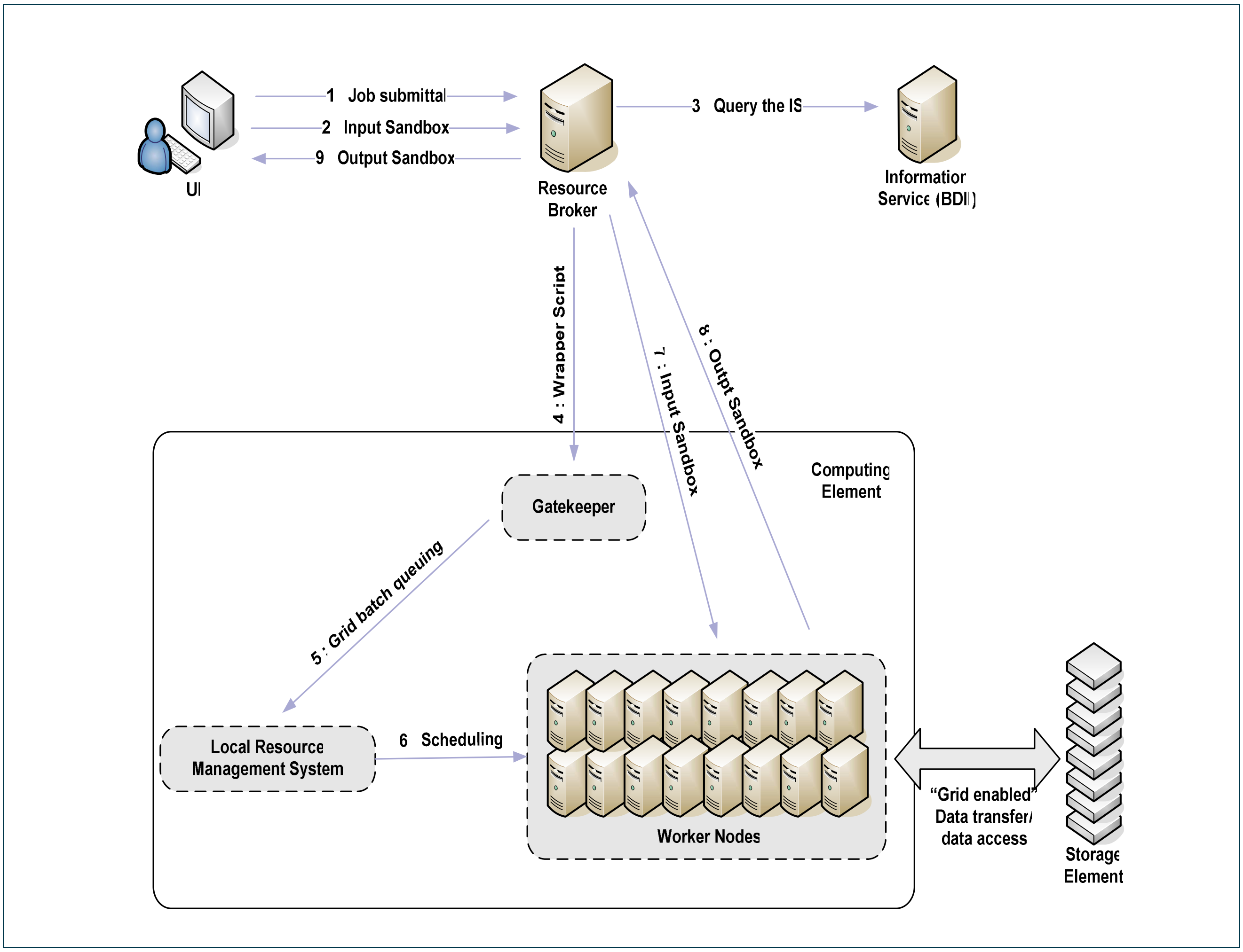


Figure 1: Job flow in LCG/EGEE environment

The time instances (Epochs) of specific events of interest to us for the purposes of modelling 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 Network Server of the Resource Broker accepts the job.
- **V_3 = workloadmanager_match_Epoch**: The time instance the WMS starts looking 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 for execution 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 LRMS assigns the job for execution to an available Worker Node from the local farm.

- **V_7 = logmonitor_running_Epoch**: The time instance the user files have been copied from the RB to the WN where the job is 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 the user can retrieve the output of his job to the UI.

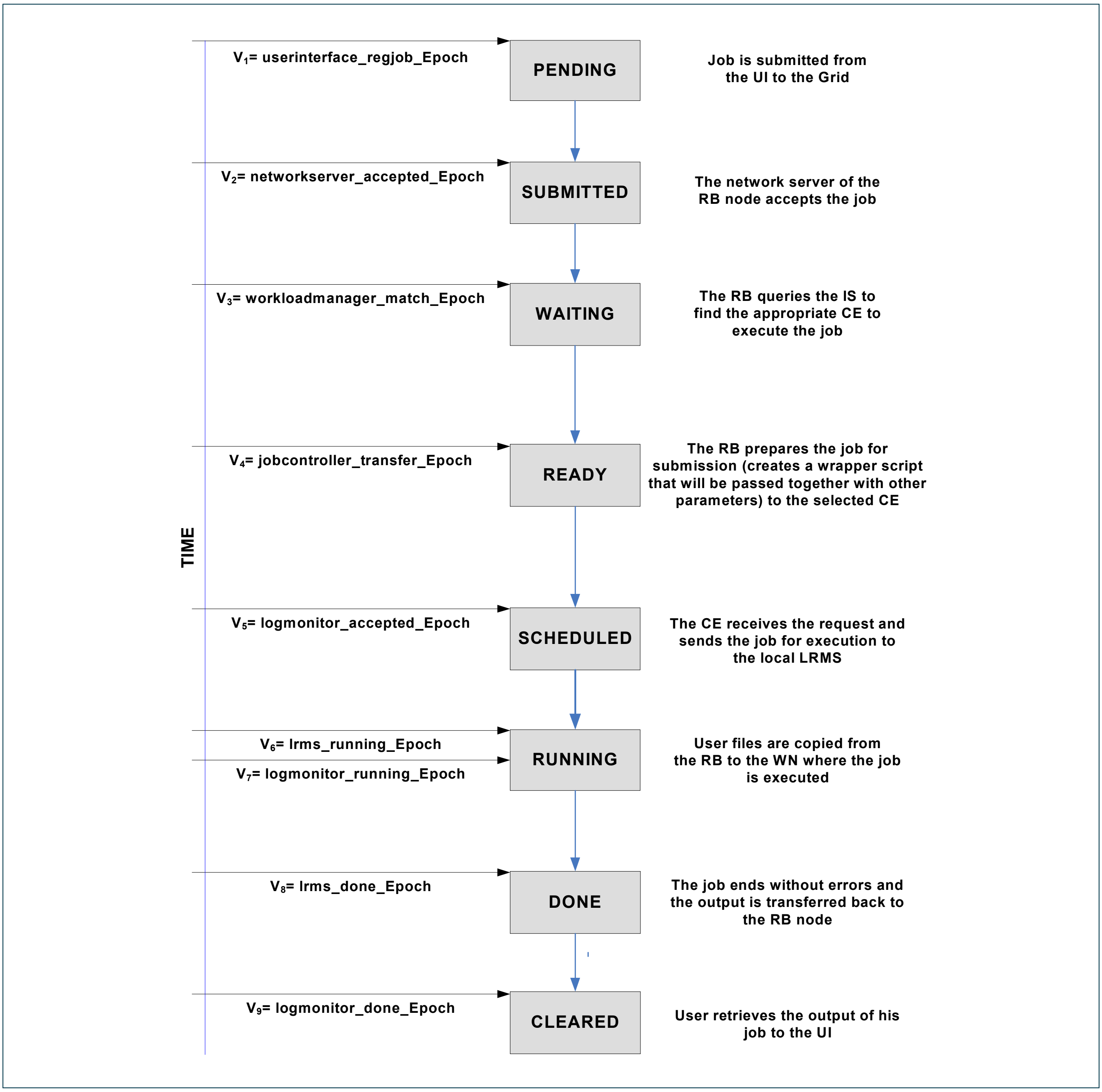

TIME
V₁= userinterface_regjob_Epoch
PENDING
Job is submitted from the UI to the Grid
V₂= networkserver_accepted_Epoch
SUBMITTED
The network server of the RB node accepts the job
V₃= workloadmanager_match_Epoch
WAITING
The RB queries the IS to find the appropriate CE to execute the job
V₄= jobcontroller_transfer_Epoch
READY
The RB prepares the job for submission (creates a wrapper script that will be passed together with other parameters) to the selected CE
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

Figure 2: The states of a job in the LCG/EGEE environment and the corresponding time instances (Epochs)

With respect to the aforementioned Epochs we describe the job flow (Figure 1) and the various states (Figure 2) of a job in the LCG/EGGE environment:

- The time instance at which the job (more specifically the job JDL file) is submitted from the UI to the RB is denoted by V_1 (userinterface_regjob_Epoch), at which point the status of the job becomes **Pending**.
- The RB receives the JDL file. The JDL file can 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 time instance that the Network Server of the RB accepts the job is V_2 (networkserver_accepted_Epoch) at which point the status of the job becomes **Submitted**.
- The RB node runs the WMS service whose role is to find the best available CE to execute the job according to the various requirements the user has specified in the JDL file and the state of every site. The WNS 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 that will be passed, together with other parameters to the selected CE. At the time instance V_4 (jobcontroller_transfer_Epoch) the job controller of the RB sends the job for execution to the appropriate CE and the status of job becomes **Ready**.
- The CE receives the request at the time instance V_5 (logmonitor_accepted_Epoch) and the Gatekeeper of the CE sends the job for execution to the LRMS. The status of the job becomes **Scheduled**.
- The 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 the LRMS assigns the job to a WN and the status of the job becomes **Running.** The user files are copied from the RB to the WN at the time instance V_7 (logmonitor_running_Epoch).
- If the job completes without errors, the output of the job, which is called Output Sandbox, starts to be transferred back to the RB node at the time instance V_8 (lrms_done_Epoch) at which point the status of the job becomes **Done**.
- At time instance V_7 (logmonitor_done_Epoch) the Output Sandbox has been completed transferred and the user can retrieve the output of his job to the UI. The status of the job becomes and remains **Cleared**.

We must note that a user has to obtain a digital certificate from a trusted Certification Authority, register in a VO and obtain an account on a User Interface in order to successfully submit a job in the LCG/EGEE. Then he can log in to the UI and create a proxy certificate to authenticate him in the previous described secure interactions. All the events are logged by the Logging and Bookkeeping service (LB) [GLITE3] which tracks jobs managed by the WMS.

Using the previous Epochs we 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.

Variables	**Corresponding States**	
V10= registration_Time	Pending	(V_2-V_1)
V11= match_Time	Submitted	(V_3-V_2)
V12= ready_tp_transfer_to_CE_Time	Pending+Submitted+Waiting	(V_4-V_1)
V13= transfer_Time	Ready	(V_5-V_4)

V14= logmonitor_CE_total_Time	Scheduled+Running+Done	(V_9-V_5)
V15= logmonitor_CE_queue_Time	Scheduled	(V_6-V_5)
V16= logmonitor_wn_Time	Running+Done	(V_9-V_6)
V17= lrms_wn_Time	Running	(V_8-V_6)
V18= total_Time	Submitted+Waiting+Ready+ Running+Done	(V_9-V_1)
V19= efficiency	(Running+Done) / (Submitted+Waiting+Ready+ Running+Done)	(V_{17}/V_{18})

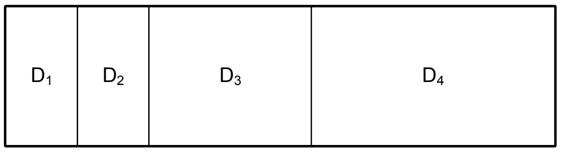
Table 1: Metrics used for analysis of the various states of the job in the LCG/EGEE environment

With respect to Table 1, the definitions of the used metrics are:

- **V_{10}= registration time** of a job, defined as the duration between the submission of a job to the LCG/EGEE environment and the time the job is accepted by the network server of the RB node,
- **V_{11}= match making time** of a job, defined as the duration between the acceptance of a job from the network server of the RB node and the time the WMS service of the RB node finds the appropriate CE for executing the job,
- **V_{12}= ready to transfer to CE time** of a job defined as the duration between the submission of a job to the LCG/EGEE environment and the time the job reaches the appropriate CE and is forwarded to the Gatekeeper of the CE,
- **V_{13}= transfer time** of a jobs, defined as the duration between the time the job controller of the RB node sends the job for execution to the appropriate CE and the time the job reaches the appropriate CE and is forwarded to the Gatekeeper of the CE.
- **V_{14}= Total CE time** of a job, defined as the duration between the time the CE receives the request and the time the output of the job has been transferred back to the RB node. This time duration corresponds to the time that a job spends at the CE.
- **V_{15}= CE queuing time** of a job, defined as the duration between the reception of request by the CE and the time the user files have been copied from the RB to the WN where the job will be executed.
- **V_{16}= WN execution time (logmonitor)** of a job, defined as the duration between the time the user files have been copied from the RB to the WN where the job will be executed and the time the user can retrieve the output of his job to the UI.
- **V_{17}= WN execution time (lrms)** of a job, defined as the duration between the time the LRMS handles the job execution on the available local farm of worker nodes and the Epoch the output of the job has been transferred back to the RB node.
- **V_{18}= Total time** of the jobs, defined as the duration between the submission of a job to the LCG/EGEE environment and the time the user can retrieve the output of his job to the UI,
- **V_{19}= Efficiency**, defined as the WN execution time (logmonitor) divided by the total time. Thus, if this value approaches 1.00 this means waiting times are negligible compared to execution times.

Based on these metrics we define four delay components that comprise the job processing in the LCG/EGEE environment (Figure 3). By definition the Total time of the job (V_{18}) is the sum of these four delay components.

- **D_1 = V_{12} = ready to transfer to CE time** describes the time the job stays at the Pending, Submitted and Waiting states. This delay component consist of the time that a job requires to register with the RB, the time the RB requires to run the match making service and to create the wrapper scripts that would 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 queuing time** describes the time the job stays at Scheduled state. This time corresponds to the time the job stays at the CE queue before it starts to execute at a WN (including the time that is required to transfer the input user files –input sandbox- from the RB to the that 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 –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.



D_1 : Ready to transfer to CE time (Pending+Submitted+Waiting)
D_2 : Tranfer time (Ready)
D_3 : CE queuing time (Scheduled)
D_4 : CE execution time (Running + Done)

Figure 3: Delay component of a job in the LCG/EGEE environment

2.2.3 HellasGrid – Kallisto cluster

Hellasgrid [HGR] main objective is the development of a National Strategy on Grid technologies and the coordination of the related communities' activities & actions, so as to provide a seamless electronic infrastructure environment throughout Greece and facilitate the communities' participation in pan-European and international efforts. The HellasGrid infrastructure of GRNET consists of 6 computer clusters of a total capacity of 768 CPUs (384 dual) and 90 TB of storage (30 TB disks and 60 TBs of tape libraries) around Greece. The cities that the clusters are located are: Athens (3 clusters), Thessaloniki (1 cluster), Patras (1 cluster) and Heraclion-Crete (1 cluster). At the Isabella cluster located in Athens, there is the *rb.isabella.grnet.gr* Resource Broker participating in LCG/EGEE infrastructure. In addition, 4 Access Grid nodes are used as a Virtual Collaboration environment for advanced communications and video conferencing.

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

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 [MAUI] is a local scheduling engine that is used together with the PBS batch system [OPBS]. 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 *Kallisto* site configuration and the corresponding results differ from those in [EM05] in this respect.

2.2.4 BEgrid

BEgrid is the computing/data grid infrastructure that results from the Belnet Grid Initiative, which was started at the beginning of 2003. The infrastructure currently has more than 400 CPUs available and 3 terabytes of storage capacity. This computing and storage capacity is distributed across various project participants, with the main contributors being University of Leuven (34 CPUs), University of Ghent (138 CPUs), University of Antwerp (77 CPUs) and University of Brussels (132 CPUs) .

The site at Ghent was installed in two distinct phases. In the first phase, 41 dual Opteron (1.6 GHz) worker nodes, equipped with 4 GB RAM memory, were installed. In the second phase, an additional 15 dual dualcore Opteron (2 GHz) with 4 GB RAM were added to the infrastructure. The first phase was completed in March 2003 while the second phase was completed at the end of May 2006. To efficiently coordinate all grid services (broker, user interface,), 5 service nodes are available. All nodes are running Scientific Linux 3, and use gLite middleware for efficient coordination of all grid services.

2.3 Discussion of traces

2.3.1 Statistics in the Grid level - LCG/EGEE

Using the daily reports in ASCII format supplied by the Real Time Monitor tool we acquired information on the traffic (jobs) submitted to the LCG/EGEE infrastructure and the time durations the jobs spent in each of the processing states before completing execution. The Real Time Monitor (RTM) [RTM] 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, the times at which the jobs complete the different states of their processing, and finally depending on the successful or not execution it also presents the times of delivery of the execution outcome to the corresponding user. Real Time Monitor tool uses the Berkeley Database Information Index (BDII) [GLITE3] 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 and choose among three different forms: 3D Globe Visualization, Google Earth and 2D Map Visualisation. The view can be requested to show just the activity of a single Virtual Organization, and has an option to show just the sites of regional infrastructures connected to the EGEE Grid, such as EELA (Extending the EGEE infrastructure to Latin America) or SEE-GRID. Finally, in the Real Monitor Tool site there are available Daily Reports, each corresponding to the activity of the EGEE over a 24 hour period.

The time period of the observation was one month (starting from 1st of October 2006 until 31st of October 2006). The total number of jobs that were submitted during this period was 2228838.

We concatenated the daily ASCII report files and obtained a file that included the desired information in a form that was suitable for processing using statistical analysis tools. The structure of the merged file consists of a table whose fields are presented in Appendix A.

From the Real Time monitor Tool we were able to retrieve general information regarding the job execution and also the time epochs that correspond to specific events in the LCG/EGEE environment (as presented in chapter 2). By manipulating these epochs we were able to calculate the metrics that were presented in Table 1 and thus analyze the times the job spent at different states of its processing and thus the corresponding delay components.

2.3.1.1 *Exit Type*

As presented in Appendix A, the **Exit Type** describes the final-exit status of a job. A detailed presentation of the different Exit Types that are reported in the Real Time monitor tool is presented in Appendix B. After manipulating the different Exit Types and grouping them together we managed to obtain a form that is more simple and comprehensive (Figure 4). From Figure 4 we can observe that a high percentage of jobs (~59.5%) were successfully completed while there is also a considerable percent (~37.4) of jobs that were aborted (ABORT) due to middleware or hardware errors. The rest of the seven presented **Exit Types** were observed with a frequency less than 1%. It is worth noting that the **Exit Type** CANCEL corresponds to the case a job is canceled by the user while it was being executed.

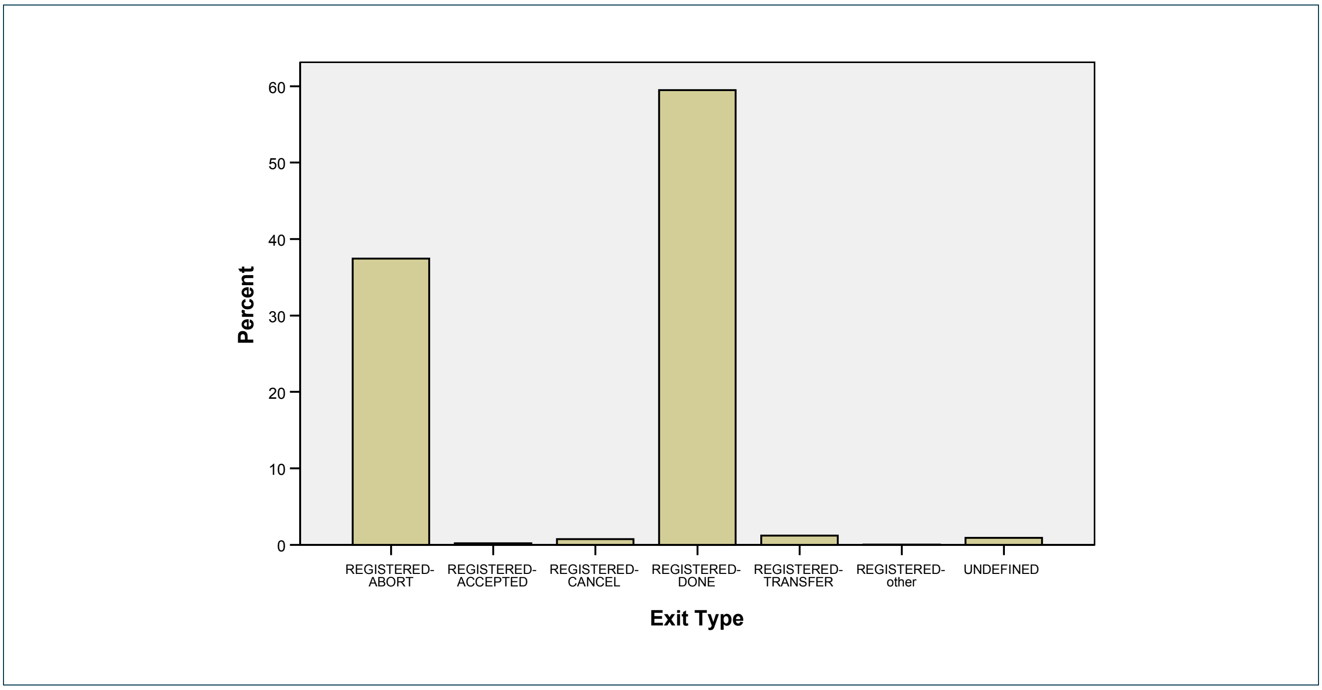


Figure 4: Exit Type of the jobs

2.3.1.2 *Daily and hour Cycles*

Figure 5 shows the number of submitted jobs to all the Resource Brokers with respect to the submission date (October 2006), while Figure 6 shows the number of jobs submitted at different hours within a day (for each hour we summed all the jobs that were submitted during that hour in the October). From Figure 5 we can 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 (4 days in total), and some days that the usage is high, more that 80.000 jobs. Regarding the daily cycle of the submission process, we can observe that the value varies between 80.000 and 100.000 jobs per hour (summed during October). The **peak** of utilization is observed during **midday** hours (13:00-17:00), while the **minimal** is observed during the **night** hours (00:00-06:00). It is worth noting that the observations correspond to the GMT time zone.

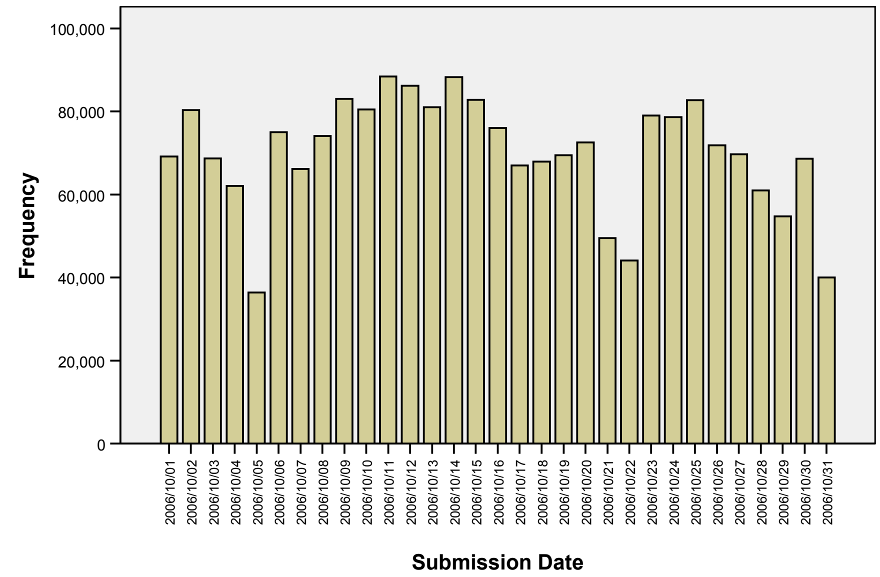


Figure 5: Number of jobs per day

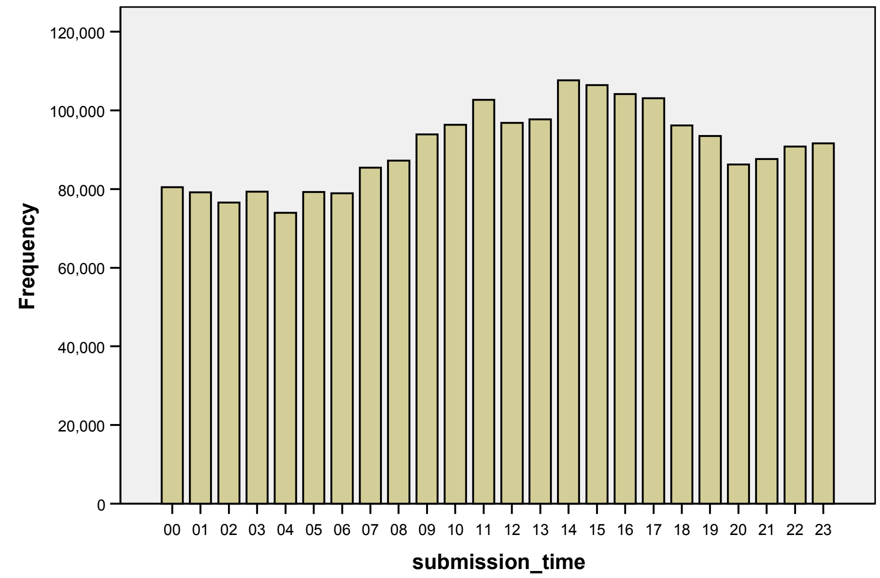


Figure 6: Hour distribution of jobs during October 2006

2.3.1.3 *Virtual Organisations*

Regarding the VOs, there are 75 VOs participating in the EGEE. The LCG/EGEE resources are not utilized to the same degree by all VOs. Figure 7, shows the contribution of every VO. The VOs whose percentage 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.

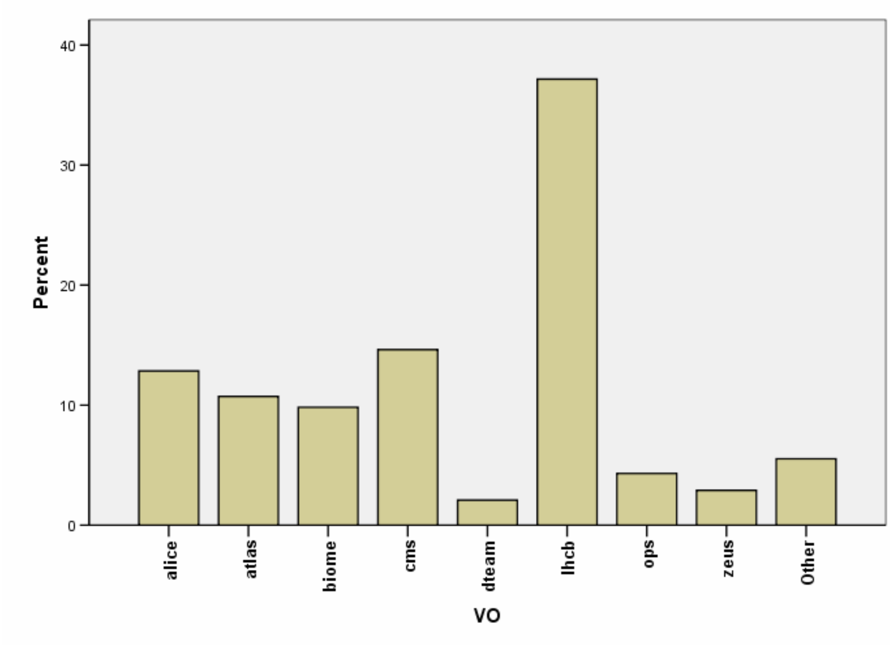


Figure 7: Percentage of submitted jobs per VO (most active VOs: ≥1%)

2.3.1.4 *Resource Brokers*

Regarding the RBs, there are totally 59 RBs that serve the jobs in the LCG/EGEE environment and forward them to the appropriate CE. Figure 8 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,
- other RB, handling 7,28% 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.

These **top five RBs serve 38,33% of the total traffic**, **while 52,54% of the RBs serve less than 0,1%** of the total traffic each.

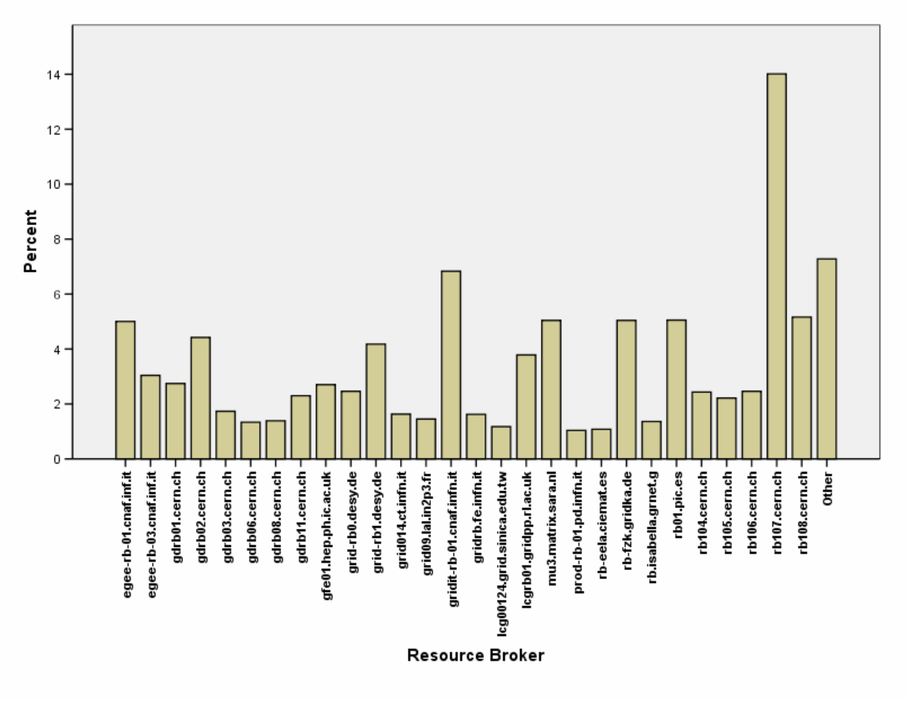


Figure 8: Percentage of served jobs per Resource Broker (most active RBs: ≥1%)

2.3.1.5 *Computing Elements*

There are totally 343 CEs in EGEE at which jobs can be executed. Figure 9 records the percentage of jobs served by each CE. The CEs that served less than 1% each of the total number of jobs are grouped together as “other”. Also, there is a percentage of CE that is categorized as «unknown» (we did not have the related information).

Figure 10 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 didn’t register correctly with their RB.

The top five CEs are:

- 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%.

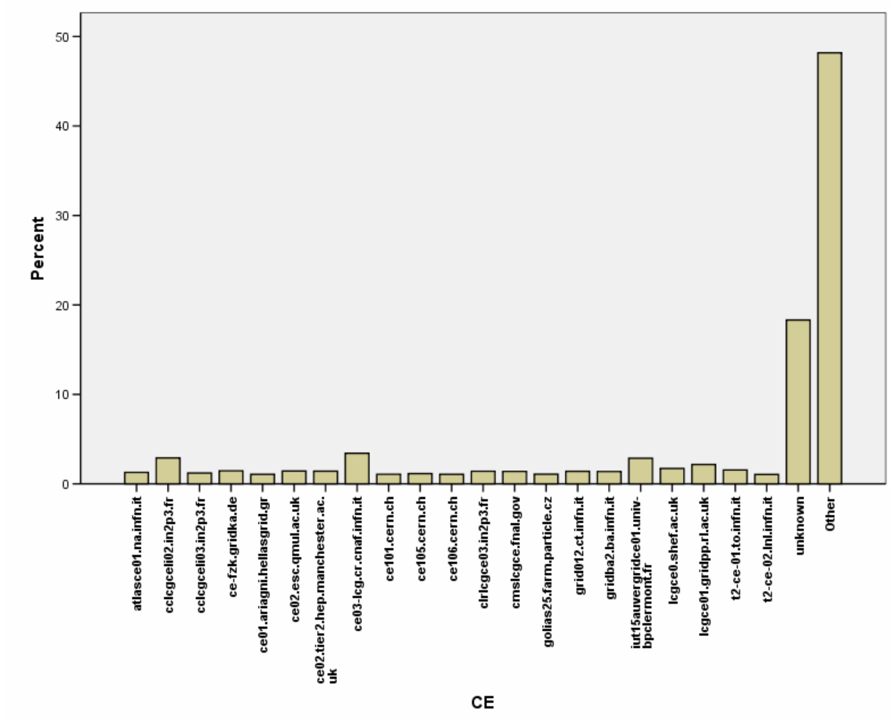


Figure 9: Percentage of executed jobs per cluster-CE (most active CEs: ≥1%)

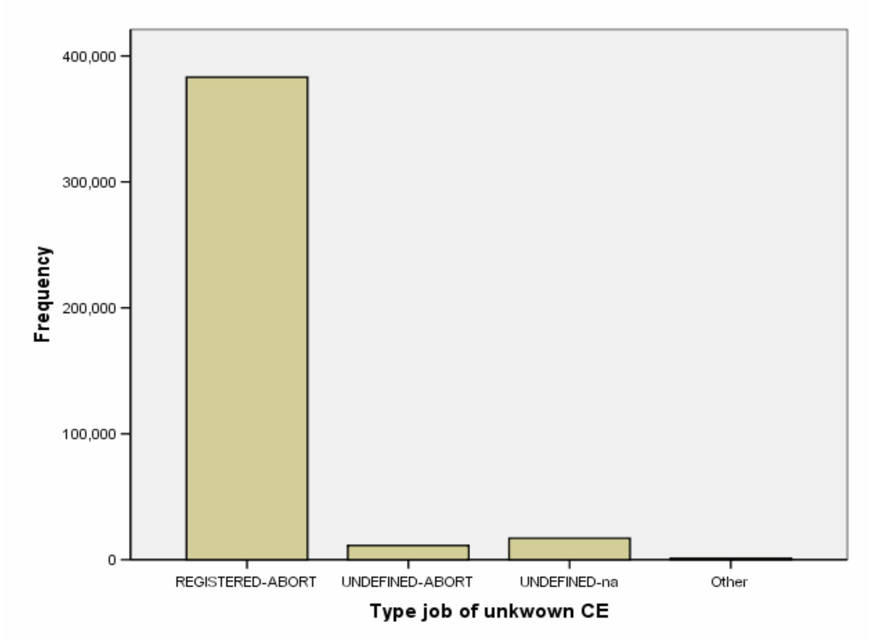


Figure 10: Distribution of the job Exit Type for the 'unknown' CE

2.3.2 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 minimum, the maximum, the mean and the standard deviation values of the job inter-arrival times and the metrics (V_{10} to V_{19}) used to measure and analyse the time durations spent by each job at the different processing states in the LCG/EGEE environment. By analyzing these metrics we are able to understand better the delay introduced by different states of the job processing and thus to define the four job delay components (D_1 to D_4) presented in section 3.

	N	Minimum	Maximum	Mean	Std. Deviation
Inter-arrival time	980581	0	60	1.25	1.52
V_{10}= registration_Time	2166574	1	14679	14.90	79.579
V_{11}= match_Time	1824822	1	65794	96.76	841.783
V_{12}= D_1= ready _to_transfer_to_CE_Time	1784806	1	65808	141.44	894.825
V_{13}= D_2= transfer_Time	1767897	1	999822	12411.69	72363.756
V_{14}= D_3+D_4 = logmonitor_CE_total_Time	1365789	2	1099682	39757.34	88809.945
V_{15}= D_3= logmonitor_CE_queue_Time	1170688	2	1099673	16899.05	61007.083
V_{16}= D_4= logmonitor_wn_Time	1170804	1	1201163	14454.54	38012.270
V_{17}= lrms_wn_Time	1039674	1	1752808	14248.76	36403.976

	N	Minimum	Maximum	Mean	Std. Deviation
V_{18}= D_1+D_2+D_3+D_4 = total_Time	1355887	17	1099957	49286.74	113684.688
V_{19}= efficiency	1042871	.01	1.00	.519	.33

Table 2: Descriptive statistics of the used metrics. N is the number of jobs from which the statistics were computed, while minimum, maximum and mean values are measured in seconds

2.3.2.1 *Inter-arrival times*

Figure 11 illustrates the cumulative distribution function (cdf) of the inter-arrival times of the jobs submitted to the LCG/EGEE infrastructure. It is worth noting that the Real Time Monitor tool, from which we obtained the measurements, stores the corresponding time instances in seconds, which means that the real (continuous) time values are rounded to the closest second integer. Thus the accuracy of our observations is the second scale. We can observe that with high probability (around 0.4) the inter-arrival time between two jobs is close to 0 sec (the inter-arrival times that are represented as 0 sec include the inter-arrival times up to 0.5 sec). The maximum observed value was 60 sec. However, as Figure 11 indicates the probability of observing an inter-arrival time greater than 7 sec is negligible. Since the inter-arrival times' **standard deviation is quite small** and close to its mean (1.25 s) we can assume that the **inter-arrival process is quite close to a Poisson process**.

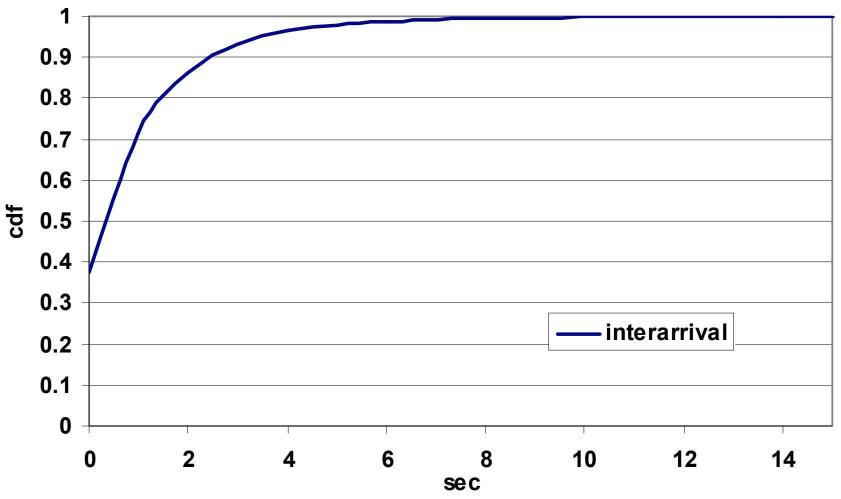


Figure 11: Empirical cdf of the inter-arrival times.

2.3.2.2 *Registration, Match-making, Ready to transfer to CEr and Transfer times*

In this section, we present results regarding the: **Registration** (V_{10}), **Match-making** (V_{11}), **Ready to transfer to CE** (V_{12} = D_1) and **Transfer times** (V_{13} = D_2).

From Figure 12 we can observe that the **match making times and ready to transfer to CE times** of the jobs exhibit similar behaviours and the majority of the observed values lies in the range of a few to a few tens of seconds, as can be deduced from the steep step-like form of the cdf in that region. Registration time has a small probability (~ 0.06) to be less than 5 sec and a high probability (~0.9) to be between 6 and 50 sec. Match making time has a small probability (~0.07) to be less than 7 sec and a high probability (~0.85) to be between 8 and 66 sec. 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 times dominates the two other delay components, 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 values of the ready to transfer to CE times with the match making times - Table 2 (their mean values differ by 50 sec while the values of their standard deviation are almost equal).

From Figure 12 we can observe that the probability of observing a **transfer time** smaller than 3 sec is small (~0,06), while the probability of observing a value less than 80 sec is high (~0,84). However, from the transfer times cdf we can observe that this variable seems to **exhibit a heavy tail** and thus there is also a considerable probability (~0.16) of observing values in the range of hundreds to millions of sec. The difference of the transfer times (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).

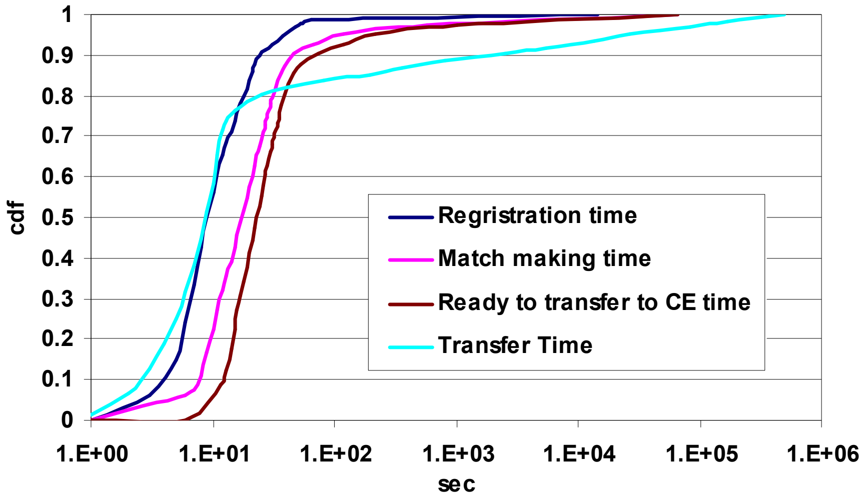


Figure 12: Empirical cdf's of the Registration times (V_{10}), Match making times (V_{11}), Ready to transfer to CE times ($V_{12} = D_1$) and Transfer times ($V_{13} = D_2$).

2.3.2.3 *CE Queuing, WN Execution 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 Queuing (V_{15} = D_3)** the **logmonitor WN Execution (V_{16} = D_4)**, the **Irms WN Execution (V_{17})** and the **CE Total times (V_{14} = V_{15}+V_{16} = D_3+D_4)**.

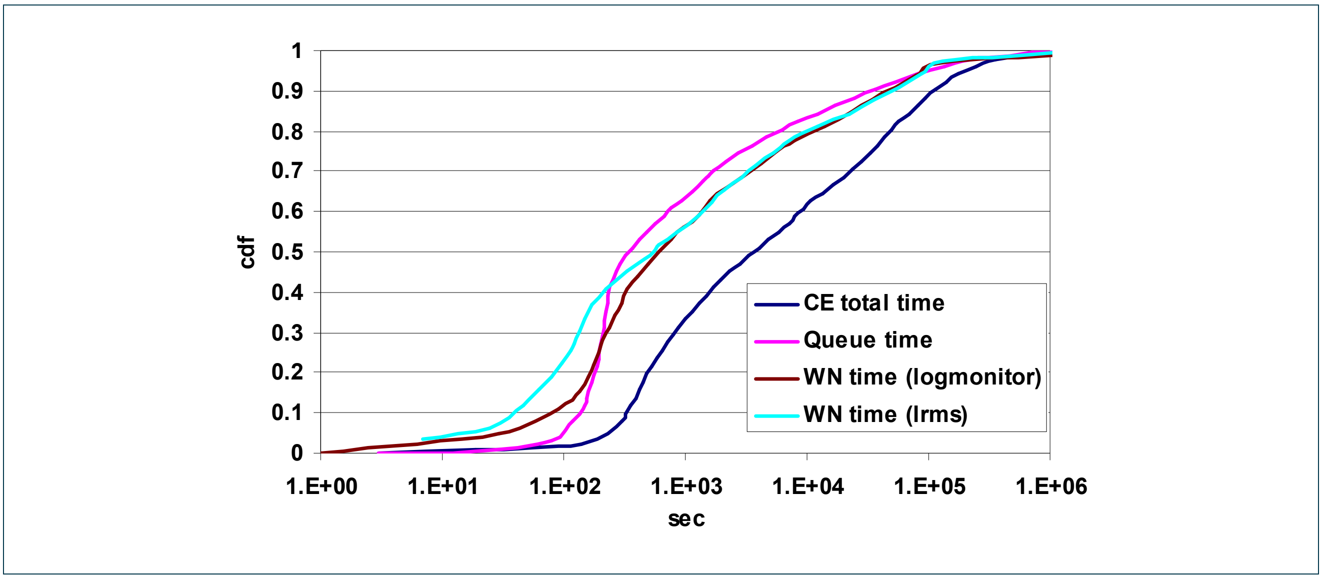


Figure 13: Empirical cdf's of the Total CE times (V_{14}), and the subsets that comprise it. More specifically we plot the cdf's of the CE Queuing times (V_{15}= D_3) and the WN Execution times according to logmonitor (V_{16} = D_4) and Irms (V_{17})

By comparing Figure 13 with Figure 12 we can observe that the cdf's of the variables presented in this section increase less rapidly than the cdf's of the variables presented in the previous paragraph.

The results of Figure 13 indicate that a job queuing time starts from 100 sec and have a high probability to be less than 200 sec. However, we can observe that **queuing times can take large values and even reach 10^6 sec**.

The logmonitor WN times and the Irms WN times have a small difference for values less than 1000 sec (specifically Irms 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 1000 seconds, and can reach values of 10^6 sec.

CE total time includes the queuing and logmonitor WN time. There is a medium probability (~0,35) to observe a CE total time less than 1000 seconds while this variable can reach values of the order of 10^6 sec. The mean value of the CE total times was measured to be equal to $38.75 \cdot 10^3$ sec and its standard deviation was $88.8 \cdot 10^3$.

2.3.2.4 *Total times and Efficiency*

The results in Figure 14 indicate that the total times (V_{18} = D_1+D_2+D_3+D_4) of the jobs exhibit almost similar behaviour with the CE total times (CE queuing + WN execution times = D_3+D_4). **CE total times dominate the total times**, while ready to transfer to CE times (D_1) and transfer times (D_2) contribute a negligible delay to the overall delay. The job total times ranges between 200 and 10^5 sec with probability ~0.91, and can also take large values (10^7 sec).

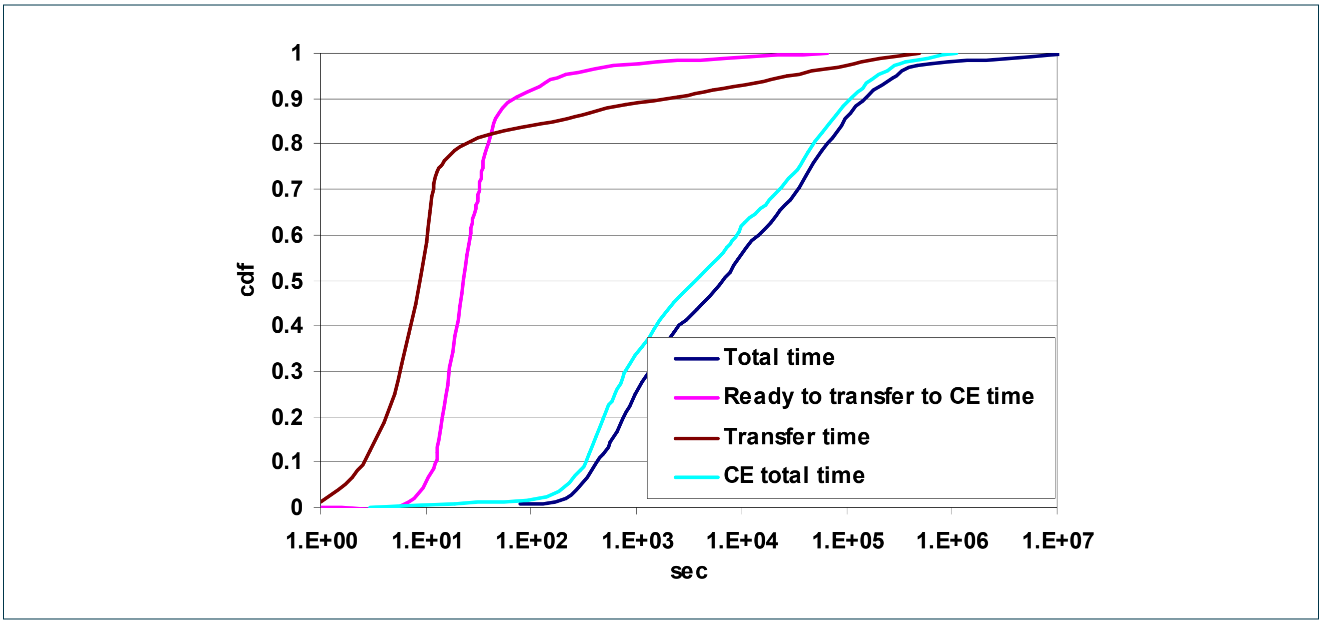


Figure 14: Empirical cdf's of the Total job times (V_{18}), and the constituent delays that comprise it. More specifically we plot the cdf's of the Ready to transfer to CE times (V_{12}), the Transfer times (V_{13}) and CE Total times (V_{14})

Finally, Figure 15 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 that is submitted to the grid has almost equal probability to exhibit efficiency between 0 and 1.

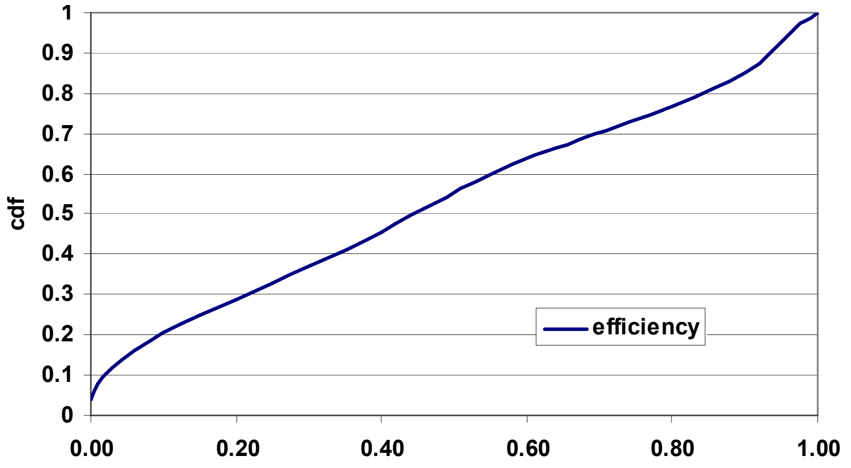


Figure 15: Empirical cdf of the efficiency (V_{19})

2.3.3 Statistics in the cluster level - Kallisto

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). The total number of jobs submitted during this period was 25737.

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 [WOSCR] in order to match our metrics. The file we obtained after parsing the log files consist of a table with the following entries:

- A consecutive number (id) for each job.
- The job's exact submission date and time.
- The job's relative submission time.
- The time interval each job waited in its queue.
- The job Worker Node execution time (sum of I/O and CPU time).
- The job CPU time.
- The amount of memory each job utilized.
- The estimate of the job CPU time, assigned by the user who submitted the job. (This number has some default values – in almost all the observed jobs its value was 259200 seconds = 3 days).
- The estimate of the amount of memory required by a job, assigned by the user who submitted the job. (This number has some default values–in almost all the observed jobs its value was 512 MB).
- The Job's status (whether the job finished successfully, was canceled, or failed to complete).
- The id of the user who submitted the job.
- The id of each queue.

Apart from the workload analysis we also examined the data transfers between our cluster and the remaining EGEE infrastructure. More specifically, we used the log files of the Storage Element (located under the directory /var/log) to acquire relevant information. The period of the observation was the same as with the CE's and the total number of grid-ftp connections during this period was 10587.

In order to obtain good models for the job submission process and the job characteristics, we performed a thorough statistical analysis of the measurements presented in the previous section. Apart from examining the weekly and daily cycles of the workload we studied the job inter-arrival times, the (CPU) job execution times, the waiting times of the jobs and the data transfers involved.

2.3.3.1 *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 hours per day, or its utilization decreases during specific days (e.g., weekends, holidays) or specific daily periods (e.g., at nights). Figure 16 shows the number of submitted jobs during different days in a week, while Figure 17 shows the number of jobs during different submission 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. Moreover, the majority of the jobs processed at Kallisto takes execution times in the range of multiple hours (see Table 5, overall mean of over 4 hours). This leads to users letting their jobs run at night while interpreting the results during the following day, submitting new jobs mainly in the afternoon or evening.

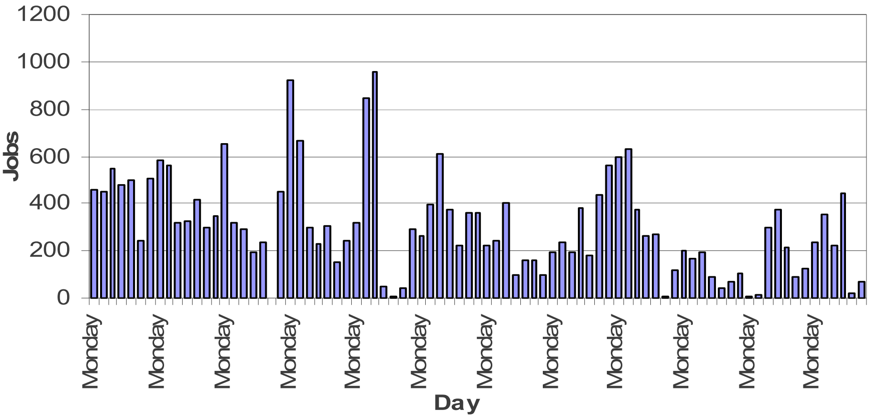


Figure 16: Number of jobs per day

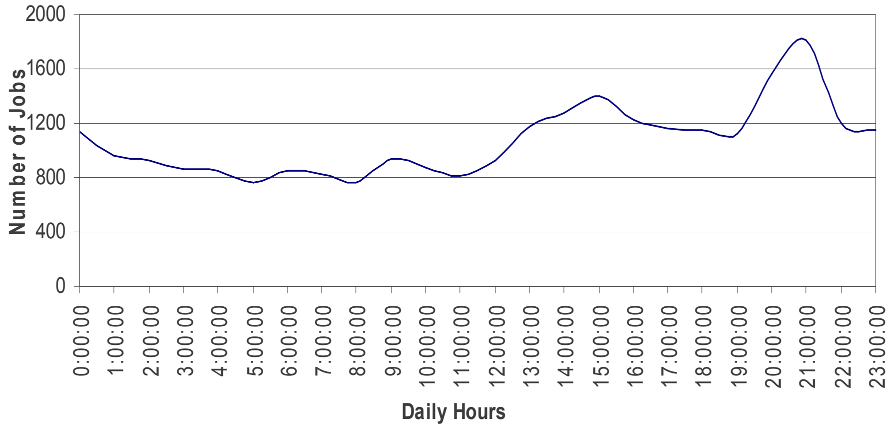


Figure 17: Daily distribution of jobs

2.3.3.2 *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.

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

Table 3: Number and percentage of jobs per VO

Table 4 and Table 5 show the mean and standard deviation of the CPU execution time and 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.

VO	Total	Atlas	Biomed	Dteam	Lhcb	Magic
Mean	15321	16139	24656	13	8511	2736
Standard deviation	29801	30146	25964	25	21236	546

Table 4: Mean and standard deviation of the CPU time (in seconds)

VO	Total	Atlas	Biomed	Dteam	Lhcb	Magic
Mean	15400	16150	24682	17	8532	2749
Standard deviation	29850	30163	25978	27	21258	567

Table 5: Mean and standard deviation of the running time (in seconds)

2.3.3.3 *Job inter-arrival times*

In this section we present results on the job arrival process at our local node. Figure 18 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 hours (10800 sec) [GOC], posing an upper limit on the inter-arrival times.

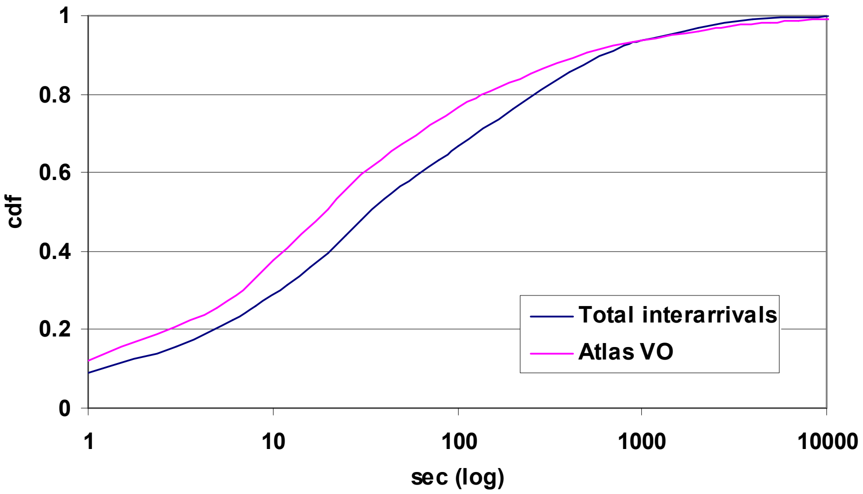


Figure 18: Empirical cdf's of the inter-arrival times for the jobs belonging to all the VOs and for the VO Atlas

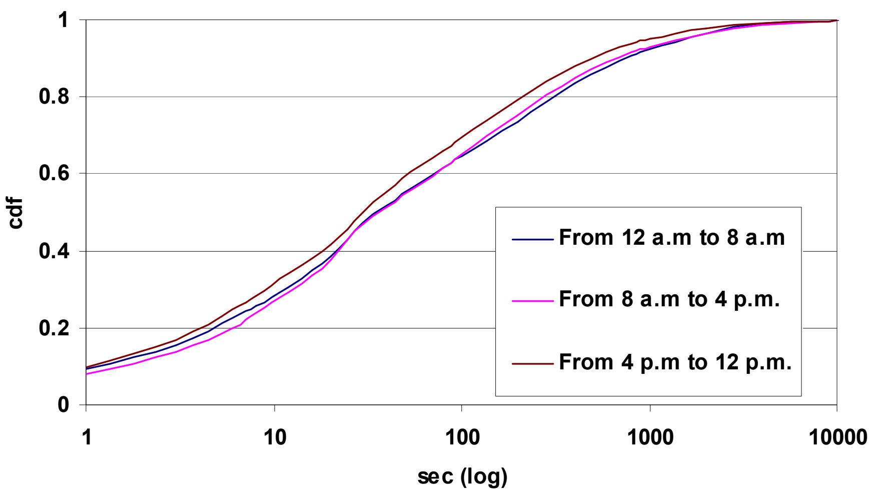


Figure 19: Empirical cdf's of the inter-arrivals per periods of day

To study the way job arrivals are distributed with respect to the time of day, we divided the 24 hours of a day into three 8-hour periods, and present the corresponding graphs in Figure 19. We observe that the cdfs have the same shape for the different time periods, while jobs that arrive between 4p.m. and 12p.m have a higher frequency when compared to the other two investigated periods (these results are in agreement with the results presented in Figure 17).

2.3.3.4 *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 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.

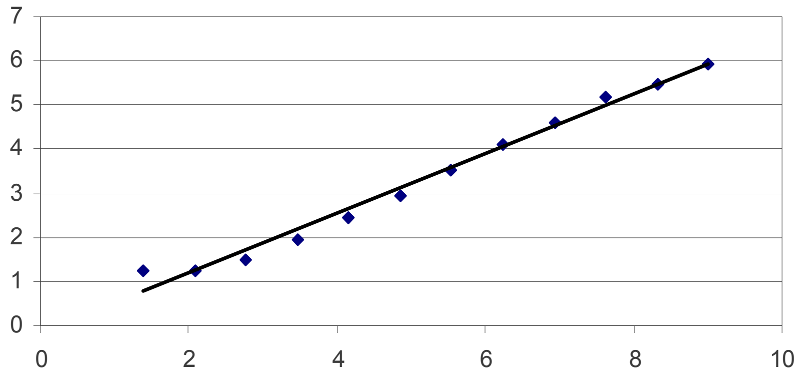


Figure 20: Hurst parameter estimation using the R/S method

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) [KFM03]). 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** (Figure 20). The Poisson process, which is not self-similar as indicated by its memoryless property, has H = 0.5. When 0.5 ≤ H ≤ 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.

2.3.3.5 *Job waiting times*

We present results regarding the waiting times of the jobs, defined as the time between the arrival of a job at the node and the time it starts execution. When a job arrives at our system, it enters a queue until a CPU becomes available to serve it. Modern grid systems, such as the one that operates in our node, employ local schedulers to ensure low waiting times for the queued jobs. Our system in particular uses a MAUI-PBS configuration that employs a separate queue for each VO and reserves one time slot for the Dteam.

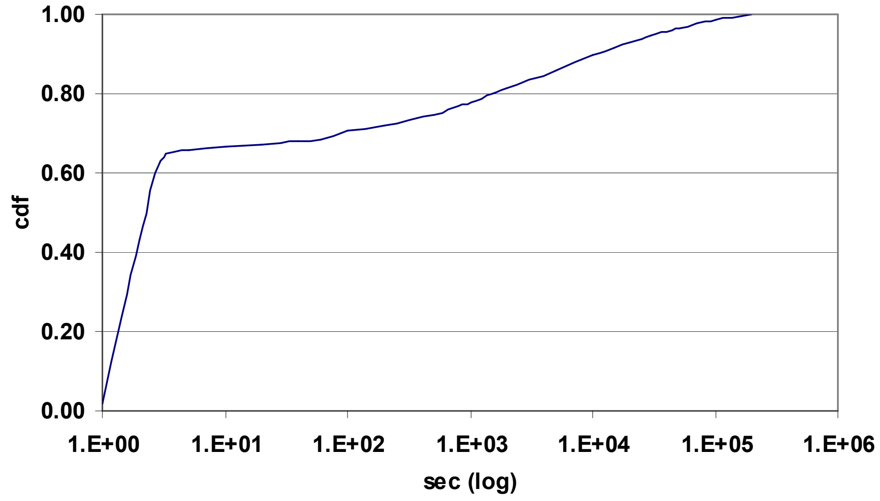


Figure 21: Empirical cdf of the job waiting times

The results of Figure 21 indicate that a job stays in a queue for less than 2 seconds with large probability (~0.7). However, the distribution still has a **fairly long tail**, the cdf showing only gradual increase towards 1E6: a few jobs 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.

VO	Total	Atlas	Biomed	Dteam	Lhcb	Magic
Mean	5503	3412	9731	236	2450	867
Standard deviation	19851	13809	19774	19851	11223	4625

Table 6: Mean and standard deviation of the waiting time

2.3.3.6 *Job Worker Node execution time*

The job WN execution time is the actual execution time of a job including the I/O time that corresponds to the time that the job stays in the Running state (Figure 2). 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 Figure 22, we give the cdf of the actual job WN execution times for the Kallisto cluster and also the corresponding cdf of the lrms WN Execution times (V_{17}) for the whole EGEE infrastructure (as presented in subsection 2.3.2.3). The observed difference between Kallisto and the whole EGEE is probably due to the small number of VOs (eleven VOs) that Kallisto serves. Additionally, VO Atlas contributes approximately 50% of the jobs submitted to

Kallisto, while it is only the third most active VO in the case of the whole EGEE.. From Figure 22 we can observe that in the Kallisto case the **job WN execution times exhibit stepwise characteristics**:

- With small probability (~0.15) a job completes its execution within a few seconds (less than 60 sec). Usually such jobs are site functional tests, ldap queries, etc.

- With medium probability (~0.25) a job completes its execution within several minutes after entering the WN (less than 30 minutes) – small jobs.

- With large probability (~0.6) a job completes its execution several hours after entering the WN. These jobs usually correspond to large experiments.

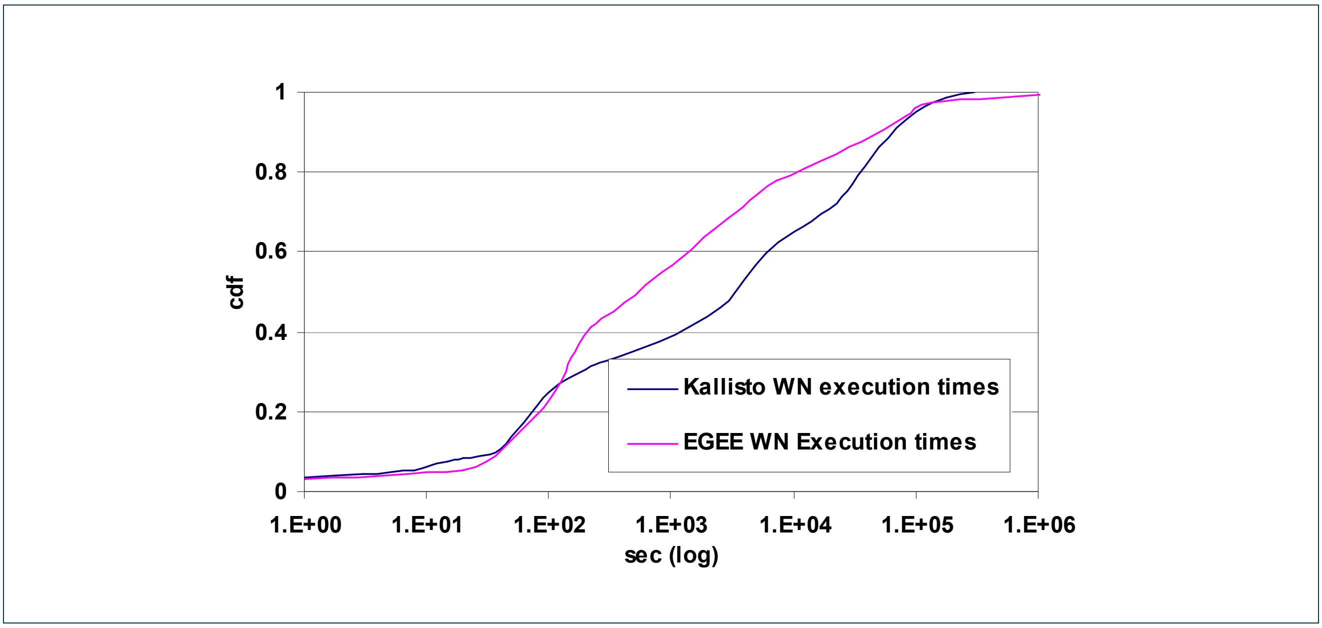


Figure 22: Empirical cdf of the job WN execution times for the Kallisto cluster and the whole EGEE infrastructure

2.3.3.7 *Storage measurements*

We have also analyzed the **grid-ftp** traffic at our local node. Figure 23 shows the cdfs of the retrieved, stored and total number of bytes exchanged between our Storage Element (SE) and the remaining EGEE infrastructure. During the observation period, the total number of grid-ftp connections to our SE was 10587 (3753 store and 6834 retrieve requests). We observe that the cdf graphs have a step-wise constant form. This is because the **majority of the data exchanges are related to the Dteam site functional tests** (SFTs). More specifically there are two sets of SFTs: (i) of size 240B that are sent periodically every 1 or 3 hours (~5000 connections), and (ii) of size 41KB that are sent at irregular time intervals (~3450 connections). The Atlas VO exchanged a large number of 103 MB chunks of data (~1800 connections), while the other VOs had rather low activity with respect to data transfers.

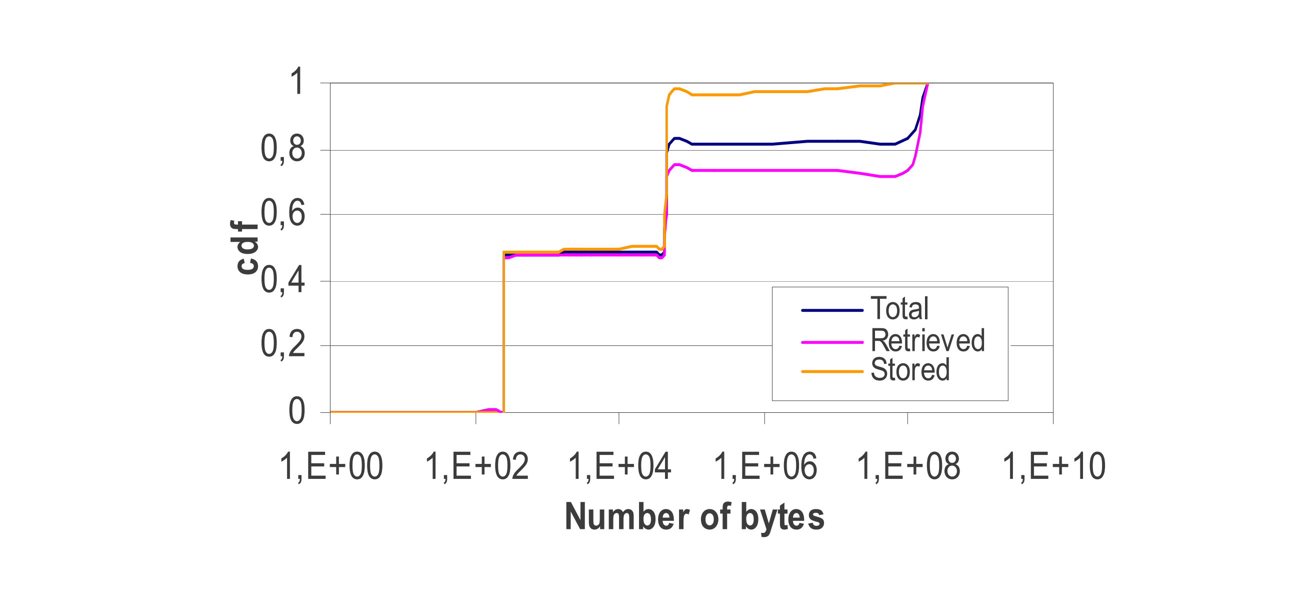


Figure 23: Empirical cdf's of the grid-ftp transfer sizes

2.3.4 Statistics in the cluster level - BEgrid

The job traces, captured at the UGent site of BEgrid, were made between 22nd of March 2006 until 7th of November 2006, amounting to data for 230 days of grid operation. The total number of jobs submitted during this period is 54055 jobs. Among others, the following information is available for further analysis:

- Job ID: a unique identifier
- Job status, e.g., finished, canceled or failed
- User who submitted the job and VO to which the user belongs
- Job timings: submission time, time when execution started, time when job was finished, actual CPU time (excluding blocking for I/O)

In the following subsections, we present a thorough statistical analysis of the job traces, examining average number of jobs submitted, job interarrival times, job CPU times and job waiting times.

2.3.4.1 *Submission date and time*

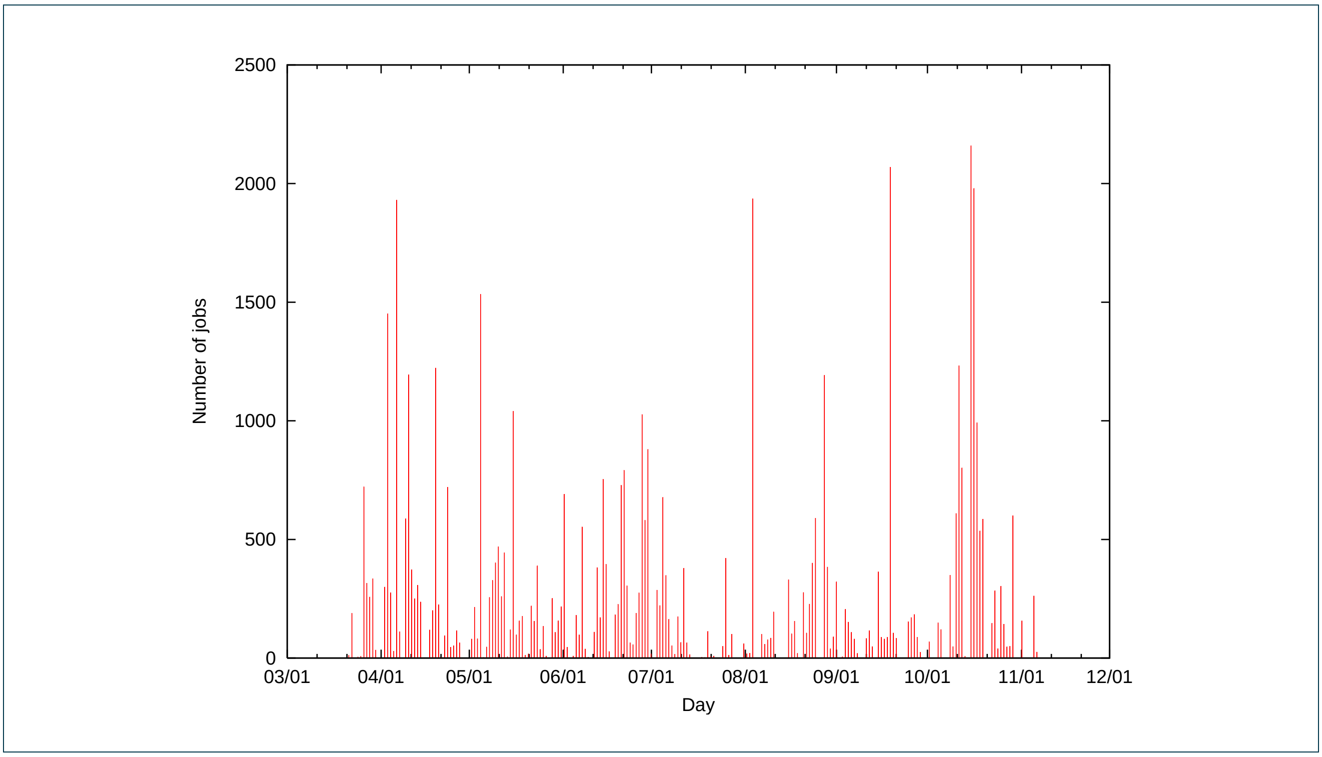


Figure 24: Number of submitted jobs per day (BEgrid).

Figure 24 shows the number of jobs that are submitted each day at the cluster. Clearly, it is difficult to identify trends or patterns in this graph, although there is a slight decrease in job submissions during the summer months (July and August), which can obviously be related to the holiday seasons.

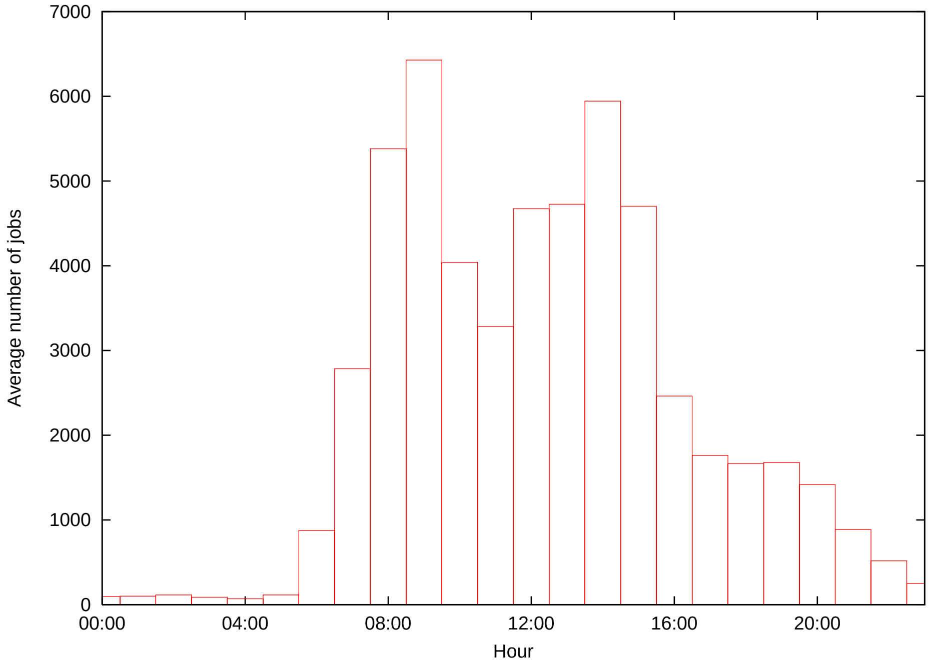


Figure 25: Average number of jobs per hour submitted (BEgrid).

In contrast to Figure 24, an obvious pattern emerges from the average number of submitted jobs per hour (Figure 25). Clearly, most jobs are submitted during office hours, while almost no jobs arrive at the cluster during the night. Additionally, two peaks can be identified, which correspond to the start of both the morning and afternoon.

The reason for this fundamentally different daily pattern compared to the Kallisto node can be found in the job execution times. As shown further on (Figure 29), a large portion of the jobs has execution times below 1 hour (~60%). This leads to different user behaviour: since they can expect to be able to results within a relatively short time frame, they can work with the results the same working day and do not feel the need to submit jobs to be executed during the night.

2.3.4.2 *Job inter-arrival times*

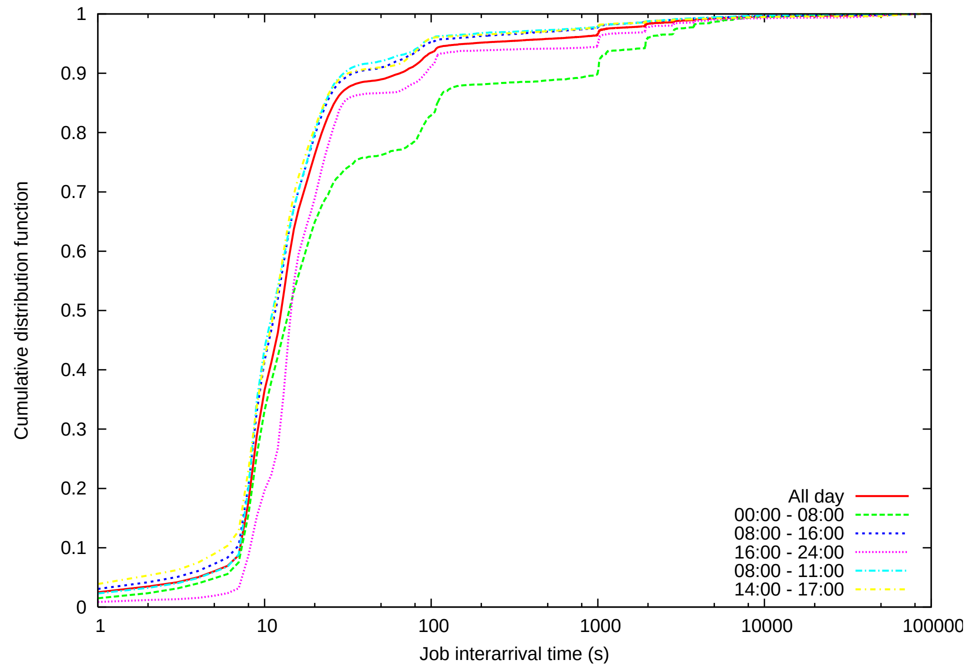


Figure 26: Cumulative density function for job inter-arrival times for different periods of the day.

Figure 26 shows the cumulative density function (cdf) of the job inter-arrival times, for different periods of the day. In general, the shape of the arrival process does not change fundamentally during different periods of the day. As noted in Section 2.3.4.1, the increase in job submissions during office hours can also be observed in this graph, since higher inter-arrival times occur much more frequently in the time interval between 00:00 and 08:00 when compared to other periods. Note also that the much steeper, almost step-like shape of the cdf of these BEGrid measurements (compared to Kallisto) is most likely a result of the typically shorter jobs (see Figure 29) leading to higher responsiveness of users.

2.3.4.3 *Self similarity*

As discussed previously, self similarity deals with burstiness in the arrival process, and is a measure for the correlation across different time scales (see Section 2.3.3.4 for more details). We computed the value of the Hurst parameter using several methods, and obtained the highest correlation coefficient (99.80%) using the aggregate variance method. The Hurst parameter was estimated to be 0.610, and as such we can conclude that the **job arrival process exhibits self-similarity** (since the Hurst parameter is situated in the interval [0.5, 1]). This is in line with the observations on the Kallisto node.

The results of the Hurst parameter estimation and their respective correlation coefficient, are summarized in Table 7.

	Aggregate variance	R/S	Periodogram	Absolute moments	Variance of residuals	Abry-Veitch estimator	Whittle estimator
Hurst parameter	0.610	0.529	0.592	0.396	1.462	0.580	0.539
Correlation coefficient	99.80	98.02	14.19	64.99	96.23	n/a	n/a

Table 7: Hurst parameter estimation for the job arrival process at a BEGrid node

A different method to observe self-similarity is to study the autocorrelation function (ACF), as illustrated in Figure 27. A strongly self-similar process has a much longer memory than a weakly self-similar process, in which case the ACF will quickly converge to zero as the lag increases. It is clear that, even for high lag values (> 50s), there are still non-negligible correlation values.

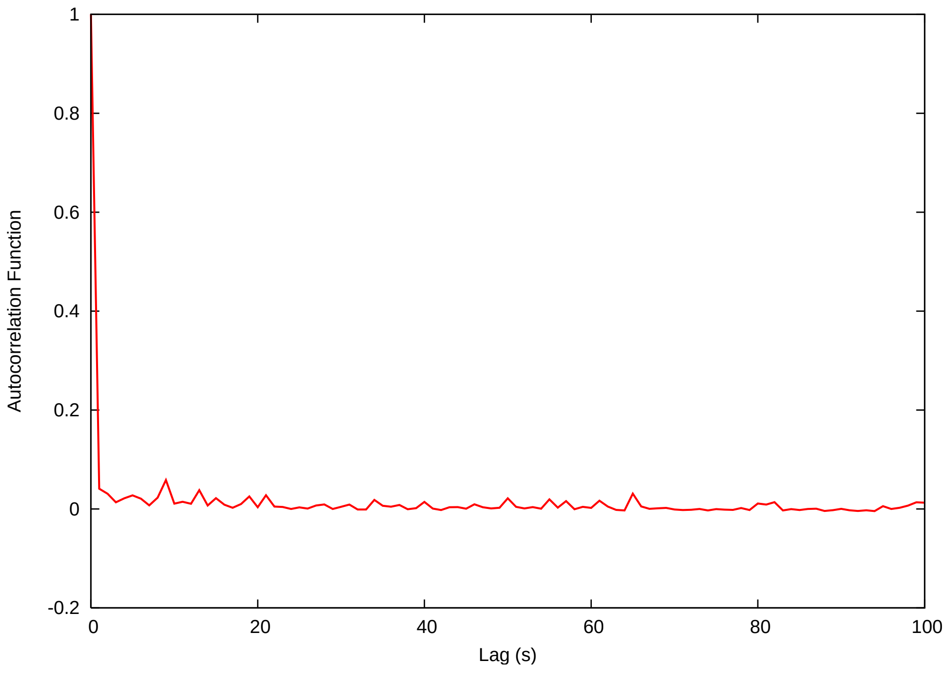


Figure 27: Empirical autocorrelation function (BEgrid)

2.3.4.4 *Job waiting times*

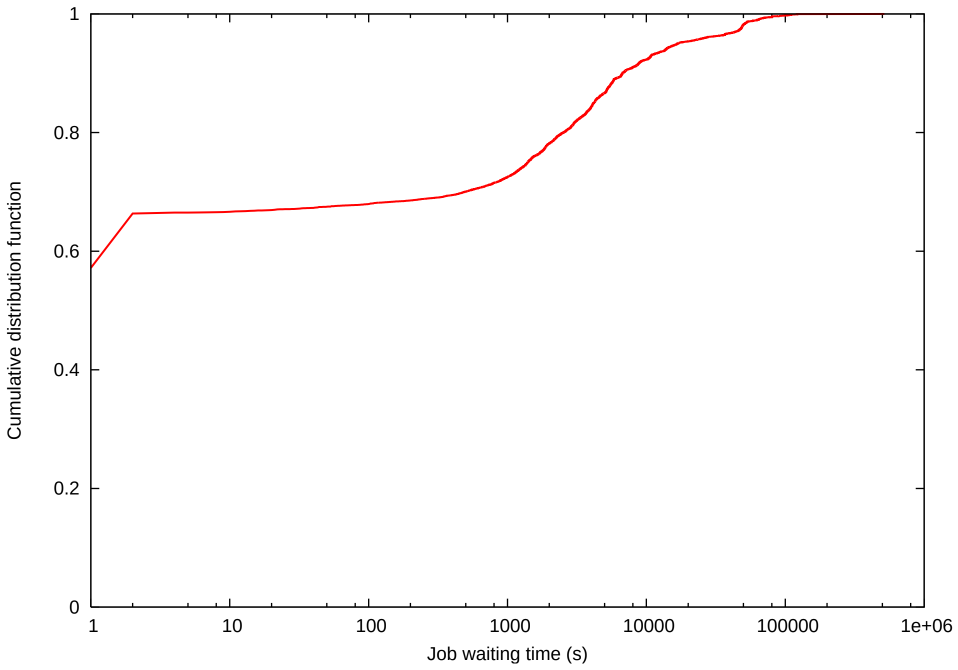


Figure 28: Empirical cdf of the job waiting times

The job waiting time is defined as the difference between the time of arrival of the job and the starting time of the actual execution. As shown in Figure 28, there are three distinct regions for the job waiting times:

- A large portion of jobs (~ 65%) enters the execution phase nearly instantaneously. The small waiting time (on the order of a couple of seconds) is due to the time needed for the scheduling algorithm and the time to transfer the job to its processing node.

- A small fraction of jobs (~ 5%) undergoes a highly variable waiting time: between 5 and 1000 seconds. This can be attributed to moderate congestion of the cluster on a temporary basis.

- The remainder of jobs (~ 30%) experiences very long waiting times (from 1000 seconds up to several hours), due to severe congestion of the cluster environment over long time periods.

2.3.4.5 *Job CPU time*

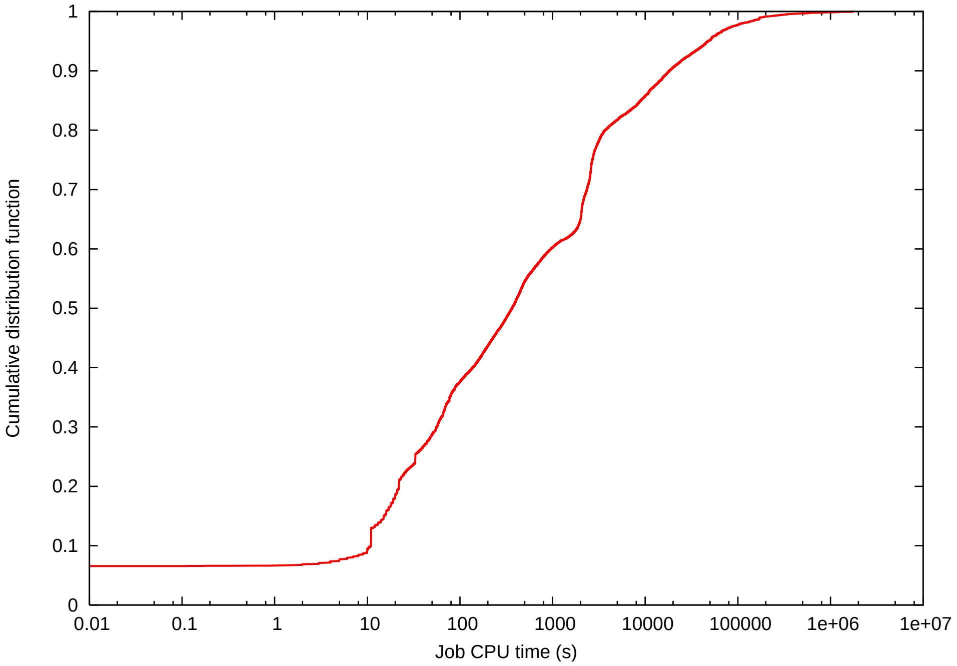


Figure 29: Cumulative distribution function of the job CPU execution times

The job CPU time is the actual execution time of a job, excluding periods of blocking due to I/O operations. Figure 29 shows the cdf of the job CPU times, where three distinct regions clearly emerge:

- A small fraction of jobs (< 10%) completes its execution within a few seconds (max 10 sec) after the start of the processing phase.
- **A large portion of the jobs (~ 70%) completes its execution within several minutes** (between 10 and 10000 seconds).
- The remainder of jobs (< 20%) exhibits very long running times, in the order of several hours (> 10000 seconds).

3 Job Demand Models

When simulating a grid environment, accurate and useful results can only be obtained by using realistic job submission processes and job resource requirements. To this end, two different approaches can be applied: (a) data from actual grid log traces can be used as input for the simulator, or (b) an analytical model can be formulated and embedded in the simulator. The former approach, while guaranteeing realistic submission patterns are being used for simulation, has the intrinsic downside of being very static. Using an analytical model to generate job loads adds flexibility: the influence of the different parameters in the model can be studied.

This section will introduce various distributions as realistic models for grid job submission processes and provide a more in-depth treatment of several types of these distributions. EM (Expectation-Maximization) algorithms are used to map actual measured grid log data onto these models, in order to obtain an analytical model capable of realistically simulating grid job creation processes. The measurement data used for these fits are those presented in the previous sections.

3.1 Models

As shown in Section 2, the generation of grid jobs does not happen in a constant fashion over time. The amount of jobs and the job characteristics vary significantly as hours and days pass. On national and European scales, day-night differences are often distinguishable. On a weekly scale, there is an obvious downfall of jobs during the weekends. Other effects can be observed when further expanding the time scale.

It is obvious that this behaviour can be modelled by a distribution with different states. The generation of parameters such as job inter-arrival time or processing and storage requirements can be partitioned into states. During day time job inter-arrivals will be closer by than during the night. This indicates the existence of two states with different job generation parameters (in this case, the inter-arrival time).

A state type process can be seen as a system which contains one or more states. These states are interrelated and can each be characterized by their state properties. The interaction between these states—the sequence in which they occur—can by governed by a deterministic or stochastic process. A state type distribution is a probabilistic system, characterized by a distribution function which is modulated by a state type process. As such, the system consists of several states, each characterised by its own distribution. These distributions can either be of the same type – a common choice is Poisson processes in which case every state has a different arrival rate (Markov Modulated Poisson Process) – or they can be different functions altogether.

3.1.1 Non-homogenous Poisson process (NHPP)

Taking into account the variations of the job arrival rate with respect to the days of a week and the hours of day, we initially investigated if the job arrival process can be modeled as a non-homogeneous Poisson process (NHPP). A NHPP is a Poisson process in which the arrival rate *λ* is a function of time t: *λ(t)*.

More specifically, the number of arrivals *N(t)* in the interval [0,t) follows the distribution:

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

Considering *λ(t)* to be a stepwise function so that in each time interval *δt λ(δt)* is constant, we considered a NHPP as a state process. State transitions are simple: after a designated amount of time the system evolves from one state to the next, where the distribution is characterized by a new λ.

3.1.2 Phase type process (PT)

An m-Phase-Type distribution (PT) represents a random variable whose values are the transition times until absorption of a continuous-time Markov chain with m transient states and one absorbing state. In general, any inter-arrival process can be approximated by a phase-type distribution if a sufficient number of states are used.

The following probability distributions are all considered special cases of a Phase-type distribution

- *Exponential distribution* - 1 phase.
- *Erlang distribution* - 2 or more identical phases in sequence.
- *Deterministic distribution* (or constant) - The limiting case of an Erlang distribution, as the number of phases become infinite, while the time in each state becomes zero.
- *Coxian distribution* - 2 or more (not necessarily identical) phases in sequence, with a probability of transitioning to the terminating/absorbing state after each phase.
- *Hyperexponential* - 2 or more non-identical phases, that each has a probability of occurring in a mutually exclusive, or parallel, manner. (Note: The exponential distribution is the degenerate situation when all the parallel phases are identical.)

Assume that for $m \geq 1$ the matrix **P** is the transition matrix of a m + 1-state discrete Markov Chain with one absorbing state (size: $(m+1)\times(m+1)$). Hence, arranging the matrix to have the (m + 1)th state as the absorbing one, we will have,

$$\mathrm{P}=\begin{bmatrix} \mathrm{T} & \vec{\mathrm{T}}^{0} \\ \vec{0} & 1 \end{bmatrix},\ \mathrm{T}=\begin{bmatrix} -\sigma_1 & \lambda_{12} & \cdots & \lambda_{1m} \\ \lambda_{21} & -\sigma_2 & \cdots & \lambda_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{m1} & \lambda_{m2} & \cdots & -\sigma_m \end{bmatrix}, \sigma_i = \sum_{j=1, i\neq j}^{m} \lambda_{ij} \text{ and } \vec{\mathrm{T}}^{0} = \begin{bmatrix} \lambda_{10} \\ \lambda_{20} \\ \vdots \\ \lambda_{m0} \end{bmatrix}$$

From the above: **T** is a $m \times m$ matrix and $\vec{\mathrm{T}}^{0}$ is a $m \times 1$ vector, while the last line of **P** corresponds to the absorbing state *0*.

The absorbing state should not be unreachable from the other states (in which case the time till absorption would be infinite), and thus at least one row of the matrix **T** should sum to less than one. Moreover, a more important constrain is that **T** cannot have any right eigenvalue with absolute value of more than one. This basically means that the matrix **T**$^{\mathbf{k}}$ tends to zero as *k* goes to infinity.

The time till absorption of the Markov Chain represented by the matrix **P** given above is denoted as $PH(\vec{a}, \mathrm{T})$. Here, $\vec{a}$ is the initialization probability and is a 1×m vector of all non-negative entries summing up to one ($\sum_{i=1}^{m} a_i = 1$). The probability density function of the Phase Type random variable equals to:

$$f(x) = \vec{a}\ \exp(\mathrm{P}x)\ \vec{\mathrm{P}}, \text{ where } \vec{\mathrm{P}} = -\mathrm{P}\ \vec{1}$$

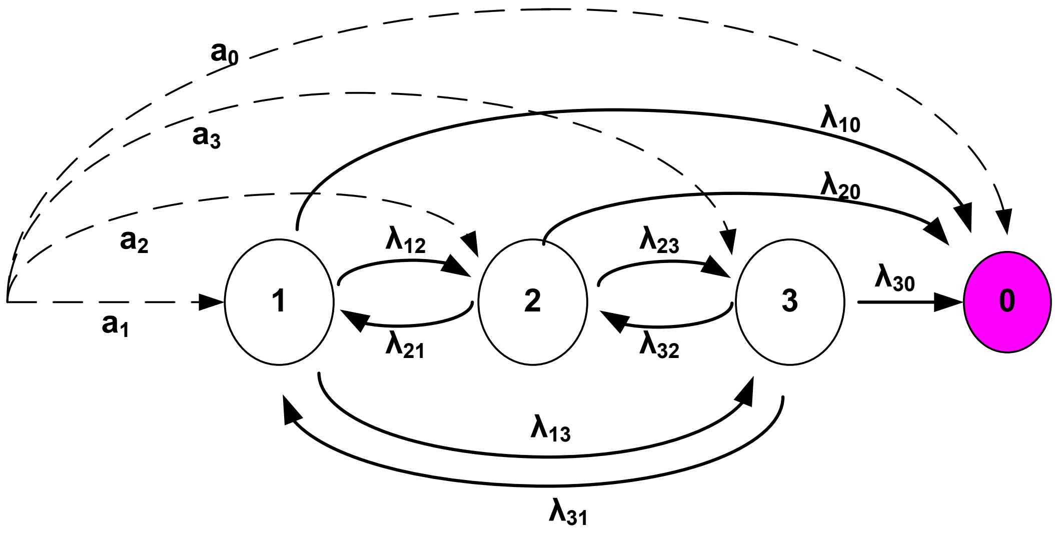


Figure 30: A three-phase PH distribution

From this general class we chose to consider only the hyper exponential subclass, which is the one most often used in the literature. A Hyper-exponential process is a Phase Type process in which the transmission matrix **T** is diagonal.

The probability density function of an *m*-phase hyper exponential random variable X is given by:

$$f_X(x) = \overset{r}{a}\ \exp(\mathrm{P}x)\ \overset{r}{\mathrm{P}} \Rightarrow f_X(x) = \sum_{i=1}^{m} a_i \mathrm{g}\exp(-\sigma_i\, x)\mathrm{g}(\sigma_i) \Rightarrow$$

$$f_X(x) = \sum_{i=1}^{m} a_i \mathrm{g} f_{Yi}(y) = a_1 \mathrm{g} f_{Y_1}(y) + a_2 \mathrm{g} f_{Y_2}(y) + ... + a_m \mathrm{g} f_{Ym}(y)$$

where Y_i is an exponentially distributed random variable with rate parameter $\sigma_i = \lambda_{i0}$ (the rate from phase *i* to absorbing state *0*), and a_i is the probability that *X* will take on the form of $\mathrm{Y_i}$ (thus, $\sum_{i=1}^{m} a_i = 1$).

3.1.3 Pareto-exponential model (PE)

Under the Pareto-exponential (PE) model, the VOs submit jobs that have exponential inter-arrival times (with rate *λ* jobs per sec) during busy periods, each of which has an exponential duration (with mean 1/*μ* sec). The times between the beginnings of the VO busy periods are distributed following a truncated Pareto distribution with Pareto shape parameter *a*, minimum value parameter X_{min} and maximum value parameter X_{max}

The proposed model is depicted in the following figure

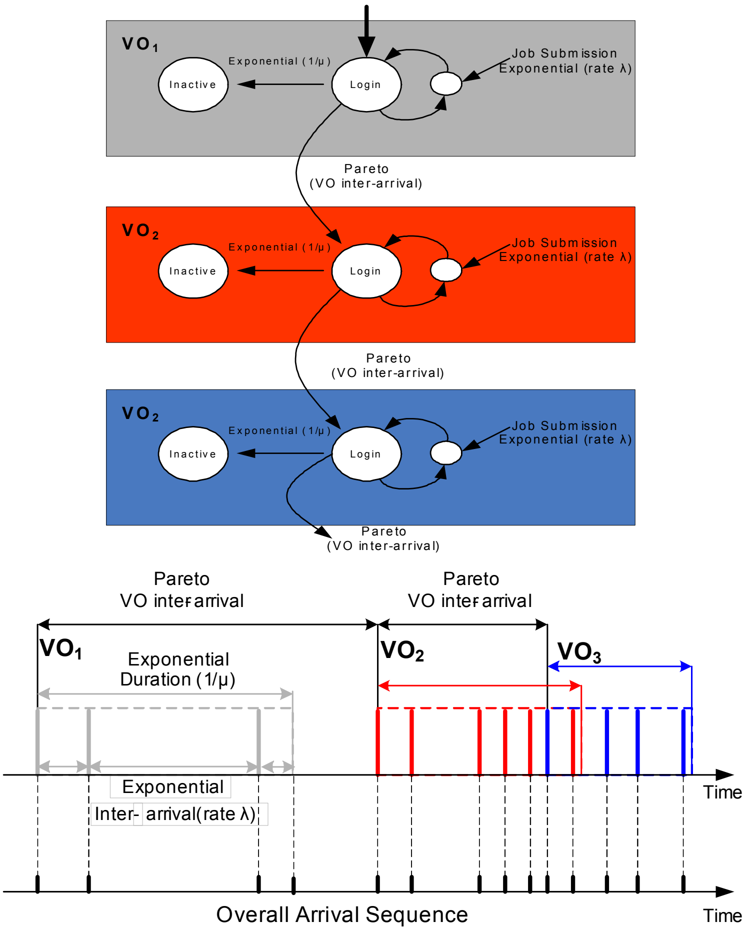


Figure 31: Proposed Pareto-Exponential model for the job arrival process

3.1.4 Markov-modulated Poisson process (MMPP)

The Markov-modulated Possion process (MMPP) is a doubly stochastic Poisson process [FH93]. MMPP uses a continuous time Markov chain (CTMC) with *m* states, each of which is characterized by a Poisson process with arrival rate λ_i ($i = 1, \ldots, m$). State transitions are modelled using a state-transition matrix Q. The MMPP model can thus be regarded as a Poisson process whose arrival rate is governed by a (finite-state) Markov chain. An MMPP can be fully described by

$$Q=\begin{bmatrix} -\sigma_1 & \sigma_{1,2} & \Lambda & \sigma_{1,m-1} & \sigma_{1,m} \\ \sigma_{2,1} & -\sigma_2 & \Lambda & \sigma_{2,m-1} & \sigma_{2,m} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} & \mathrm{M} \\ \sigma_{m-1,1} & \sigma_{m-1,2} & \Lambda & -\sigma_{m-1} & \sigma_{m-1,m} \\ \sigma_{m,1} & \sigma_{m,2} & \Lambda & \sigma_{m,m-1} & -\sigma_m \end{bmatrix} \text{ where } \sigma_i = \sum_{j=1,i\neq j}^{m} \sigma_{i,j} \text{ and } \Lambda=[\lambda_1,\lambda_2,\ldots\lambda_m]$$

The MMPP model is often used in traffic modelling because of its ability to adequately characterize traffic, including correlation between inter-arrival times, while still maintaining a relatively low complexity.

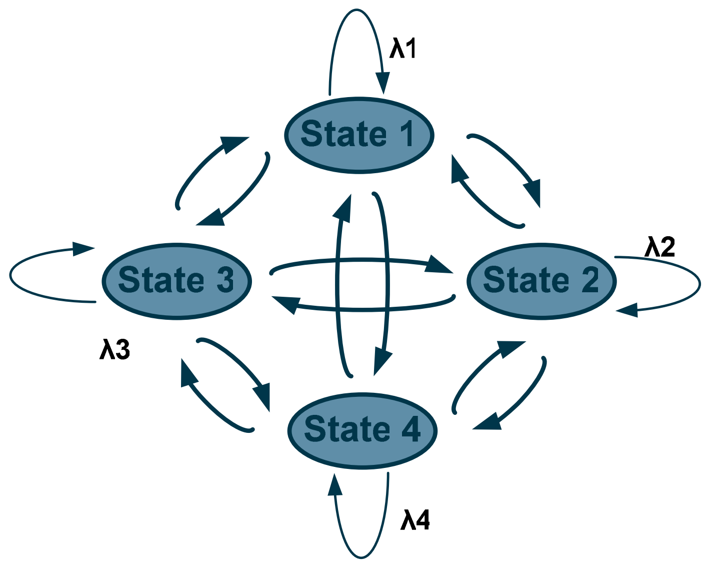


Figure 32: Markov-modulated Poisson process: Markov state interaction diagram for m = 4.

Note that MMPP is a generalized process. More specific cases can often be modeled with an MMPP by fixing some of the transition parameters σij as 0. Specifically for NHPP (see Section 3.2.1), σ_{ij} = 0 for all j ≠ i + 1.

A more generalized class of these Markov process driven systems exists. In these, each state is characterized by its own stochastic process, which is not necessarily a Poisson process (as is the case for MMPP). Fitting data to this model requires more complex fitting routines, and requires a model for the individual processes to be proposed before the actual fitting. We shall not further consider this generalized class of Markov modulated processes.

3.2 Algorithms for data-to-model fitting

In order to obtain realistic parameters for each job demand model presented in Section 3.2, experimental data (in the form of e.g. measured inter-arrival times) has to be mapped onto the parameters of those models, i.e. we need to *estimate* parameters from the data. There are many ways to estimate a particular parameter for a given distribution, but we only used one type, *maximum-likelihood estimation* (MLE). This can be done by using iterative algorithms such as Expectation-Maximization (EM).

3.2.1 Expectation – Maximization algorithms

Expectation-Maximization (EM) algorithms are algorithms used for finding maximum likelihood estimates of parameters in probabilistic models, where the model depends on unobserved latent variables.

The EM algorithm alternates an Expectation step (E) with a maximization step (M). In the former, an expectation of the likelihood is calculated, by including the latent variables in the model as though they were observed. The latter step computes the maximum likelihood estimates of those parameters by maximizing the expected likelihood found during the E step. Those new parameters found during the M step are then used to begin another E step, repeating the process until the likelihood function converges.

Details of the EM technique are outlined below, but we start with an introduction on the basic goal of the technique.

Let $p(x|\theta)$ be a density function governed by the set of parameters θ. We also have a data set of size N, which is drawn from this distribution: $X = \{x_1, x_2, ..., x_N\}$. These vectors are thus independent and identically distributed with distribution p. Their resulting density function $p(X|\theta) = \Gamma(\theta|X)$ is called the likelihood function of the parameters θ, given the data X and can be seen as a function of θ for fixed X. Maximizing this function $\Gamma(\theta|X)$ in θ yields the parameter set we wish to estimate (in our case, the parameters to be inserted in the model for the job submission process).

To clarify this problem, suppose we are trying to estimate the parameters of a Gaussian distribution, $\theta = (\mu, \sigma^2)$. Maximizing the likelihood function can then easily be done by setting the derivates to μ and σ^2 to 0 and solving. In our case, maximizing the likelihood function is not that easy, which is where the EM technique steps in.

EM generalises this problem by obtaining maximum likelihood estimates of parameters in probabilistic models, *where the model depends on unobserved latent variables*. Specifically, this means that the complete data set $Z=(X,Y)$ consists of a set of observed values X (the incomplete data) and a set of unobserved values Y. While this appears to complicate the problem, it will actually allow simplification of the likelihood function by assuming the existence of these hidden/missing values.

The joint density function of the complete dataset (which at the same time provides the likelihood of the complete dataset) is then given by:

$$p(Z|\theta) = p(X,Y|\theta) = p(Y|X,\theta)\cdot p(X|\theta) = \Gamma(\theta|X,Y) = \Gamma(\theta|Z)$$

Starting with a set of estimated parameters θ_0, EM will then first find the expected value of the (log) likelihood $\log p(X,Y|\theta)$ with respect to the unknown data Y and given the current parameter estimate θ_0. This expectation is given by:

$$Q(\theta,\tilde{\theta}) = E\left[\log p(X,Y|\theta)\middle|X,\tilde{\theta}\right],$$

where $\tilde{\theta}$ is the current parameter estimate (used to evaluate the expectation) and θ is the (new) set of parameters which optimize Q. This evaluation of the expectation is called the E-step of the EM algorithm. The second step is the actual maximisation of the found expectation function:

$$\theta_i = \arg\max_{\theta} Q(\theta,\theta_{i-1})$$

These two steps are then repeated as necessary, each time using the newly found current estimates of the parameter set θ as input for the expectation step, determining the new expectation function; and then optimizing that function, finding a new, and better, set of parameters θ. It can be shown that the EM algorithm converges to a (local) maximum of the likelihood function.

For situations where maximizing the parameters during the M step is too complex, incremental versions of the EM algorithms exist. These algorithms, instead of requiring maximization, accept a mere improvement of the likelihood function.

EM is a group of related algorithms, rather than a single algorithm. EM can be regarded as a recipe for algorithms that attempt to solve likelihood maximization problems. As such, several algorithms exist for different problems, such as the Baum-Welch and the Ryden algorithms, both for MMPP processes.

3.3 Fitting results

3.3.1 Modelling in the Grid level - LCG/EGEE

3.3.1.1 *Inter-arrival times*

Based on the descriptive statistics (Table 2) and the cumulative distribution function of the inter-arrival times (Figure 11) we want to verify if the job arrival process can be modelled by a Poisson process. Since the standard deviation of the inter-arrival times is quite close to its mean and from the corresponding cdf this variable doesn't seem to exhibit a heavy tail, a Poisson process is quite likely to model the arrival process behaviour. We have experimented with exponential distributions and parameters close to $^{1}/_{\text{observed mean}}$. Figure 33 shows the cdf of the inter-arrival times (presented in Section 4.2.1) and the cdf of an exponential distribution with mean 1.6077 sec. It is worth noting that the observed values were integers (our observations were based on the second scale). 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 manipulation the exponential distribution with mean 1.6077 sec resulted in a distribution with mean 1.15 sec and standard deviation 1.57.

Figure 34 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 sec can adequately model the job arrival process in the Grid environment**.

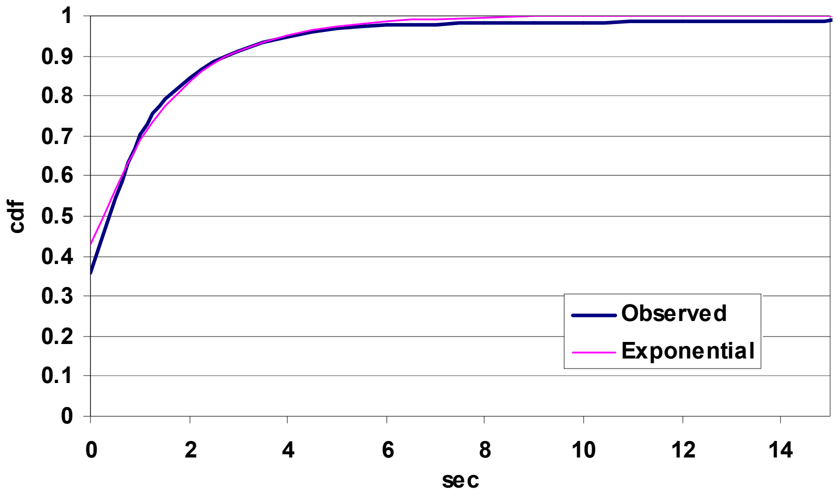


Figure 33: Cdfs of the inter-arrival times of the actual observations and the examined exponential model

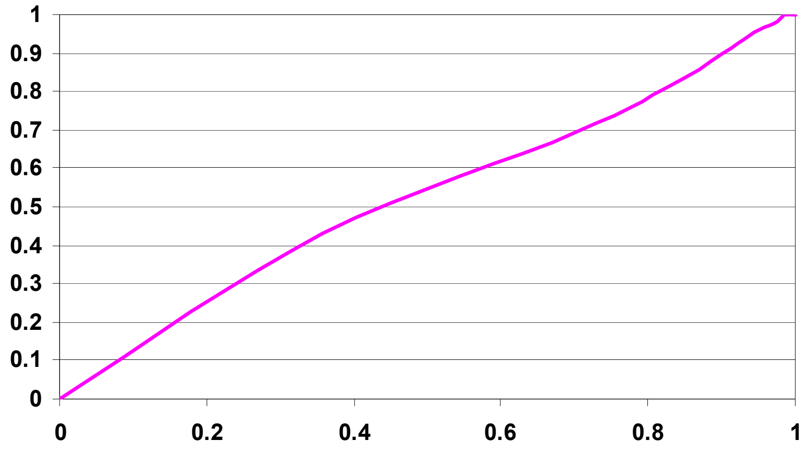


Figure 34: Exponential vs observed inter-arrival times P-P plot

3.3.1.2 *Worker Node Execution times*

The lrms WN execution times (V_{17}), as presented in Section 2.3.2.3 (Figure 13), exhibit peaks at certain periods. WN times differ in their nature from inter-arrival times since they do not depend on the human factor, and thus it is difficult to find a related physical explanation. Therefore, our criteria for modelling this process are more relaxed. We investigated how a hyper exponential process can fit this behaviour. More specifically, we considered two cases: (i) a 3-phase (H3) and (ii) a 4-phase (H4) hyper exponential distribution. We chose to use these values for the number of phases driven by the observation that Figure 13 exhibits 3-4 steps. We used again the EMpht utility [EMPHT] to obtain the corresponding parameters:

i) H3 Case: p_1=0.3888, p_2=0.3635, λ_1= $8.021 \cdot 10^{-3}$, λ_2= $5.47 \cdot 10^{-4}$ and λ_3= $1.5 \cdot 10^{-5}$,

ii) H4 Case: p_1=0.3888, p_2= 0.3635, p_3=0.1832, λ_1= $8.021 \cdot 10^{-3}$, λ_2= $5.47 \cdot 10^{-4}$, λ_3= $2.5 \cdot 10^{-5}$ and λ_4= $5 \cdot 10^{-6}$.

In order to evaluate and validate the proposed models we have simulated them in C++ and generated trace files. Figure 35 shows the empirical cdf of the job WN execution time as presented in Section 4.2.3 and the cdfs we obtained from the traces of the two hyper exponential processes, while Figure 36 shows the corresponding P-P plots. Since the modelling accuracies obtained by the 3- and 4-phase processes are almost similar, we can conclude that a **3 phase hyper exponential process is sufficient for modelling the WN execution times**.

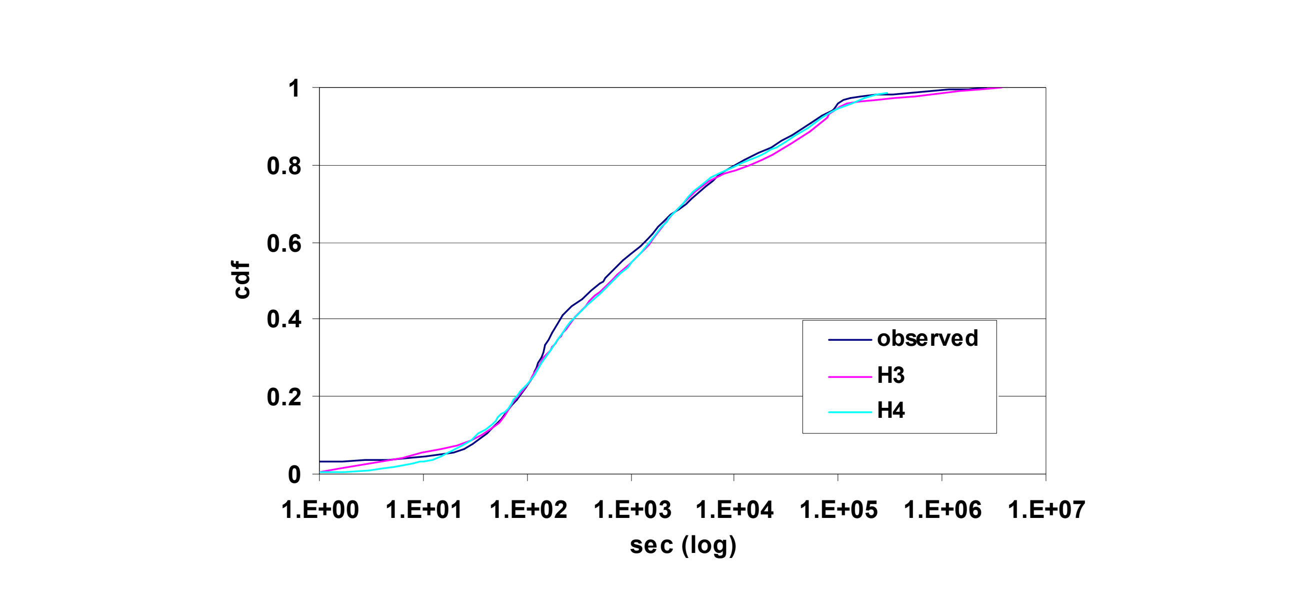


Figure 35: Cdfs of the WN execution times of the original observations and the examined 3- and 4- phase Hyper-exponential model.

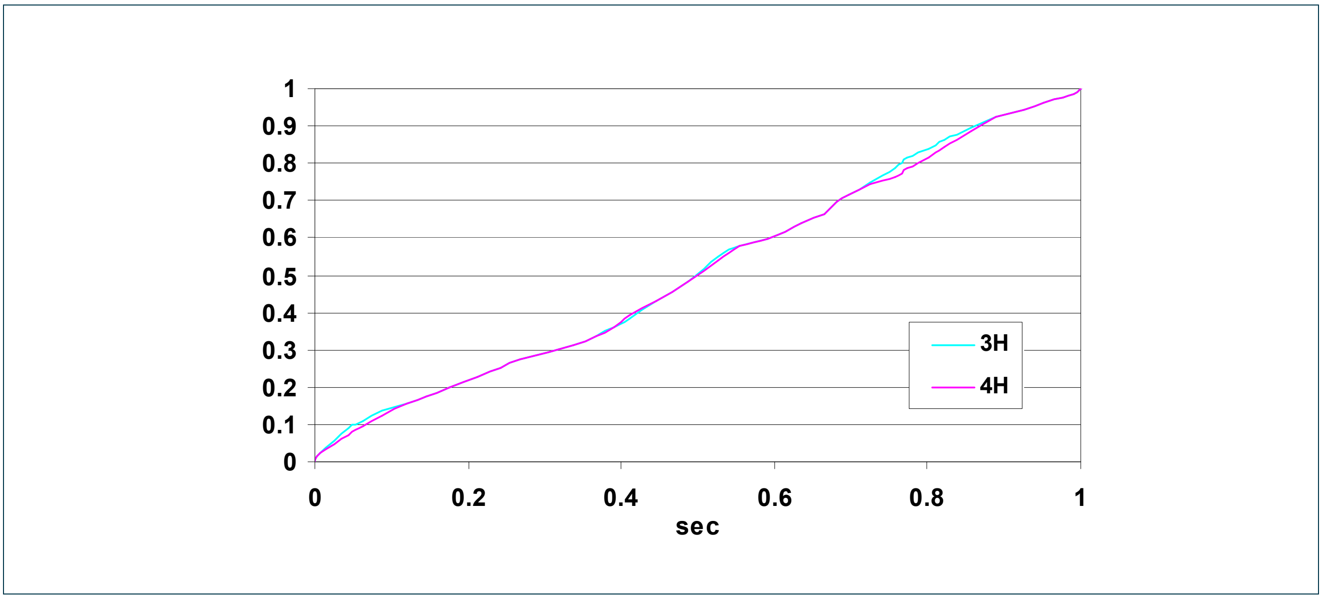


Figure 36: 3H and 4H vs observed WN execution times P-P plots.

3.3.2 Modelling in the cluster level - Kallisto

In this section we model the job arrival process and the execution time of the jobs in the *Kallisto* cluster. We decided to model the job arrival process and the execution times for the traffic generated by all the VOs together, and not separately for every VO, in order to look for general properties in the workload that the local cluster has to tackle.

3.3.2.1 *Modelling the job arrival process*

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

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

Using the results of Figure 17, we defined a stepwise function for λ(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: (i) a 2-phase (H2) and (ii) a 3-phase hyper exponential distribution (H3). To find suitable parameters (3 parameters in case (i) and 5 parameters in case (ii)) we used the EMpht program [EMPTH], which employs an Expectation Maximization (EM) algorithm [ANO96], to obtain the following parameters:

i) 2H Case: p_1=0.37, λ_1=0.00137 and λ_2=0.0465 and

ii) 3H Case: p_1=0.444, p_2=0.458, λ_1=0.0538, λ_2=0.0907 and λ_3=0.0050.

(c) Markov Modulated Poisson Process (MMPP) model

We investigated two MMPP models: (i) a 3-state MMPP (3MMPP) and (ii) a 4-state MMPP (4MMPP). To find suitable parameters (4 parameters in case (i) and 9 parameters in case (ii)) we used the program found in [MMPPF], that employs an Expectation Maximization algorithm [RED06], to obtain the following MMPP parameters that best fit our measurements:

iii) 3MMPP Case: $\sigma_{12}=6\cdot 10^{-3}$, $\lambda_1=98\cdot 10^{-3}$, $\sigma_{21}=0.45\cdot 10^{-3}$, $\lambda_2=4.1\cdot 10^{-3}$

iv) 4MMPP Case: $\sigma_{12}=3.2\cdot 10^{-3}$, $\sigma_{13}=4.3\cdot 10^{-3}$, $\lambda_1=139\cdot 10^{-3}$, $\sigma_{21}=0.1\cdot 10^{-3}$, $\sigma_{23}=0.2\cdot 10^{-3}$, $\lambda_2=0.9\cdot 10^{-3}$, $\sigma_{32}=0.45\cdot 10^{-3}$, $\sigma_{32}=0.55\cdot 10^{-3}$ and $\lambda_3=11.9\cdot 10^{-3}$.

(d) Pareto-Exponential model

We have chosen to use a truncated Pareto distribution with Xmax=10800 sec since we know that the job inter-arrival times are upper-bounded by 3 hours (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 λ=18 arrivals per sec for busy periods, mean duration 1/μ=22.5 sec of the busy periods, a=0.48 and Xmin=32 sec.

Simulation and validation of the job arrival process model.

In order to evaluate and compare the proposed models we have simulated them in C++ and generated trace files. Figure 37 shows the cdf of the inter-arrival times as presented in Section 2.3.2.3 and the cdfs we obtained from the traces of the four proposed models. Figure 38 shows the Probability-Probability (P-P) graphs of the better performing H3, 3MMPP and Pareto-Exponential models versus the actual measurements. Given 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.

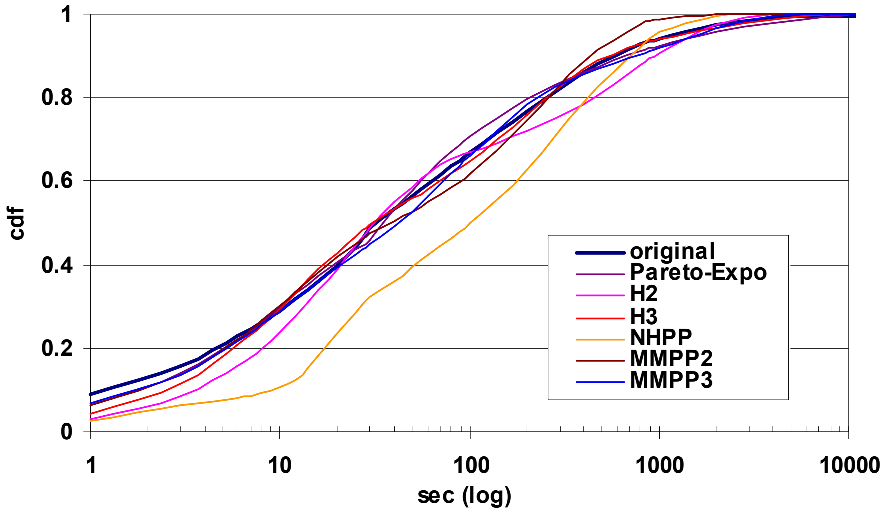


Figure 37: Cdf's of the inter-arrival times of the original observations and the proposed models

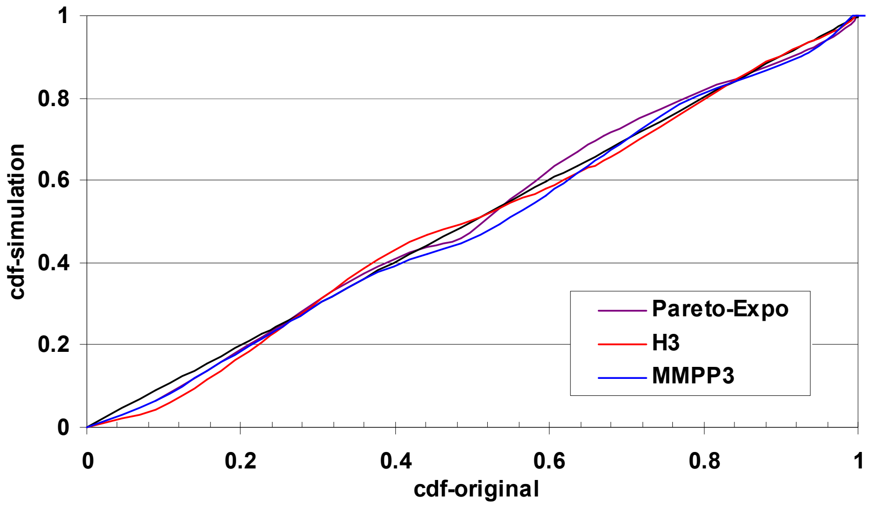


Figure 38: P-P functions of the proposed models

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 also simulate 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 behaviour.

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" 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 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 modelling 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.

3.3.2.2 *Modelling the job WN execution times*

The WN execution times, presented in Section 2.3.2.2 (Figure 22), exhibit peaks at certain values. Similar to subsection 3.3.1.2, we considered two cases: (i) a 3-phase (H3) and (ii) a 4-phase (H4) hyper exponential distribution. We chose to use these values for the number of phases driven by the observation that Figure 22 is of a stepwise form with 3-4 steps. We used again the EMpht utility to obtain the corresponding parameters:

i) 3H Case: p_1=0.3290, p_2=0.2805, λ_1=1.07 ⋅ 10^{-2}, λ_2=2.65 ⋅ 10^{-4} and λ_3= 2 ⋅ 10^{-5}

ii) 4H Case: p_1=0.3290, p_2=0.2805, p_3= 0.1410, λ_1=1.07 ⋅ 10^{-2}, λ_2=2.65 ⋅ 10^{-4}, λ_3= 2,5 ⋅ 10^{-5} and λ_4= 1 ⋅ 10^{-5}.

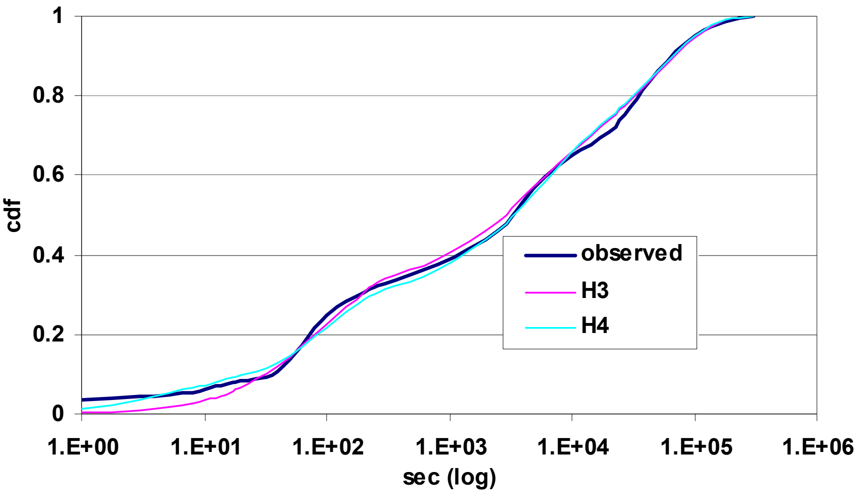


Figure 39: Empirical cdf and cdf's of the proposed models for the WN execution times.

By comparing the results presented in this subsection and the corresponding fitting in the Grid level (subsection 3.3.1.2) we can conclude that a 3 phase hyper exponential distribution is in both cases adequate to model the WN execution times, while the increase from 3 to 4 phases improves slightly the fitting. The difference in the fitting parameters for the EGEE and Kallisto cases may result from the different number of VOs that these measurements correspond (the results presented in subsection 3.3.1.2 correspond to the whole EGEE infrastructure serving 75 VOs, while Kallisto cluster served only 11 VOs during the period of observations). Moreover, the most active VO in the case of Kallisto was VO Atlas, which was the third more active in the case of the whole EGEE.

Simulation and validation of job execution time model.

Similar to previous section, we simulated the proposed models in C++, produced traces and compared them with those obtained in actual measurements. Figure 39 shows the empirical cdf of the job CPU execution time as presented in section 2.3.2.2 and the cdfs we obtained from the traces of the two hyper exponential processes. Since the modelling accuracies obtained by the 3- and 4-phase processes are almost similar, we can conclude that a 3 phase hyper exponential process is sufficient for modelling the CPU execution times.

3.3.3 Modelling in the cluster level - BEgrid

3.3.3.1 *Job inter-arrival times*

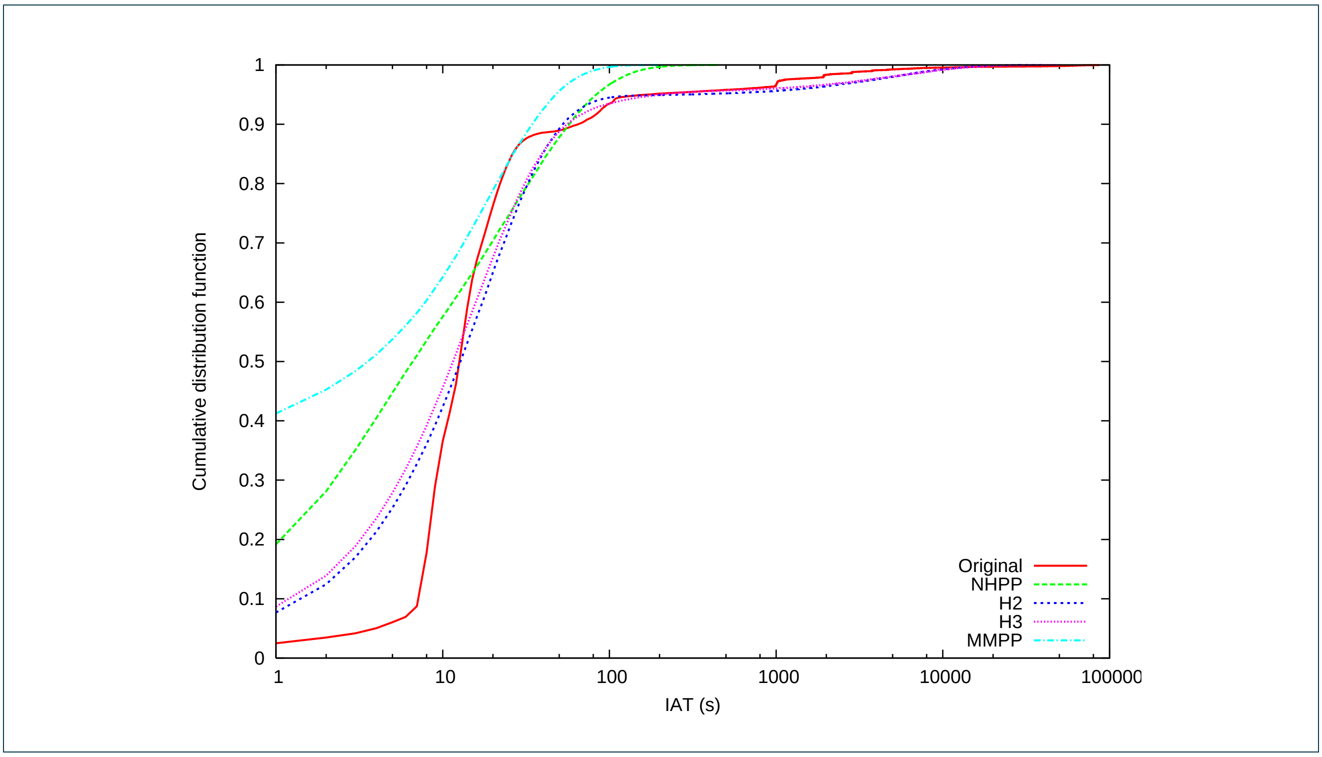


Figure 40: Fitting of cumulative density function (cdf) for job interarrival times.

In contrast to the Kallisto fitting results, we can see in Figure 40 that it is much harder to accurately fit the cdf of the interarrival times for the BEgrid cluster. The main reason lies in the specific shape of the original (emprical) cdf; it shows an extremely steep increase of the cdf for IAT values between 8 and 15 seconds. It is very difficult for an exponential distribution to capture this quasi step-wise behaviour. To accommodate for this quick increase in cdf, all models start at a higher initial value than the original cdf. We can conclude that the hyperexponential distributions (H2 and H3) show the best fitting results, although a larger number of phases could lead to further improvements (albeit at an increased complexity).

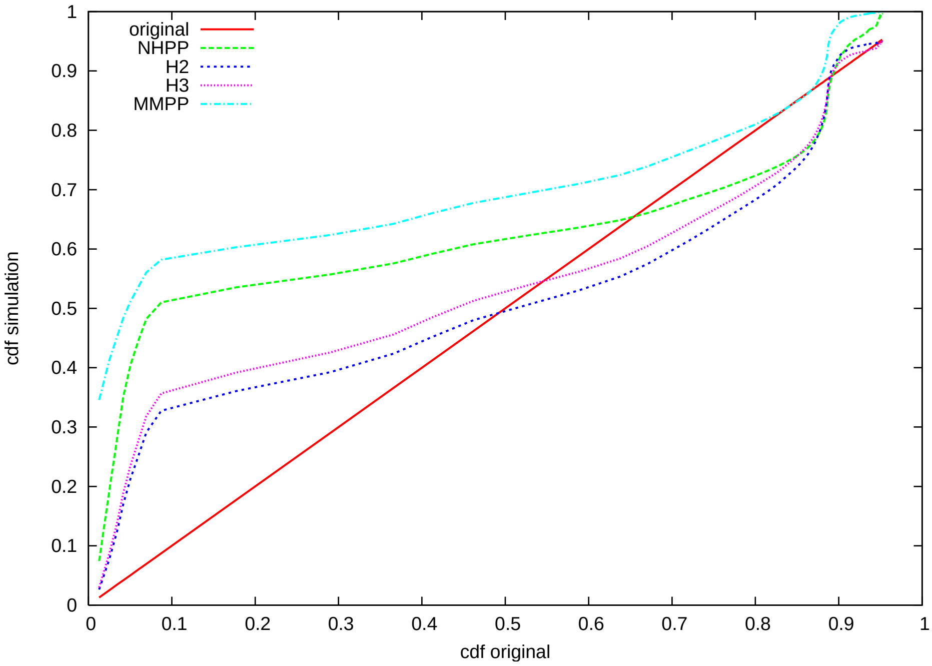


Figure 41: P-P functions of the proposed models

The P-P plot in Figure 41 confirms the discussion of Figure 40. The worst results are obtained for lower values of the cdf (between 0 and 0.3). As discussed previously, this is due to the inability of the exponential models to capture the very steep increase of the cdf. As such, the models try to compensate this by initially overestimating the cdf, so good correspondance with the original cdf is still reached for higher cdf values (> 0.4).

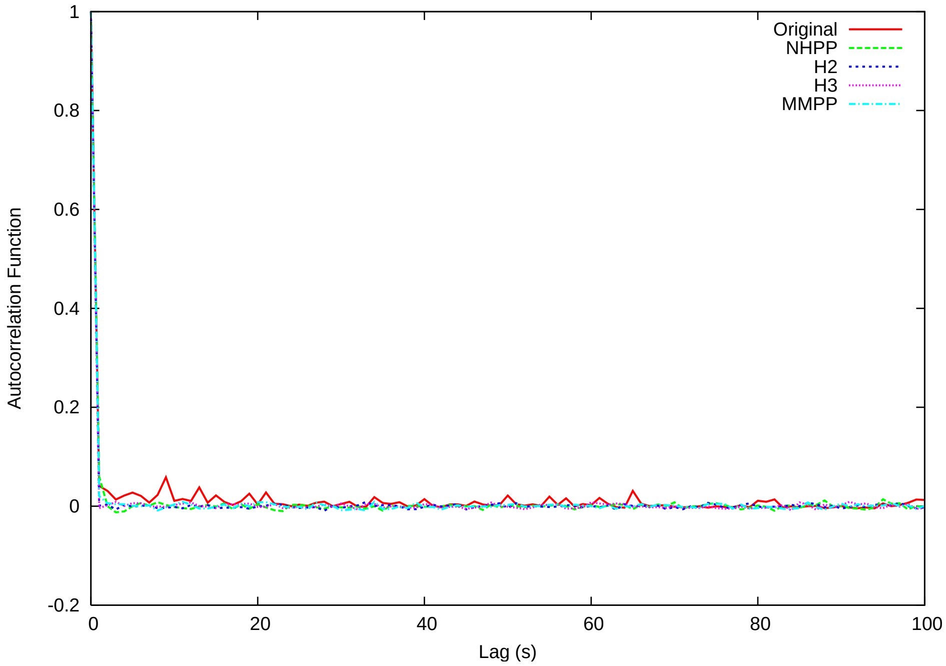


Figure 42: Fitting of autocorrelation function (acf) (BEgrid).

Figure 42 shows the autocorrelation function for the different models. All models show good correspondance to the original acf, although the correlation for larger lag values are less pronounced in comparsion to the original. Further insight can be gained from the estimation of the Hurst parameters, which is depicted in Figure 43. This shows the average (and standard deviation errorbars) value of the Hurst parameter, calculated with the following methods: aggregate variance, R/S, periodogram, Abry-Veitch estimator and Whittle estimator. It follows from the graph that all proposed models have a Hurst parameter in the interval [0.5; 1] and thus generate traffic with the desired long-range dependency (bursty traffic). Additionally, the average values obtained by the different models are all in good correspondance to the original Hurst parameter.

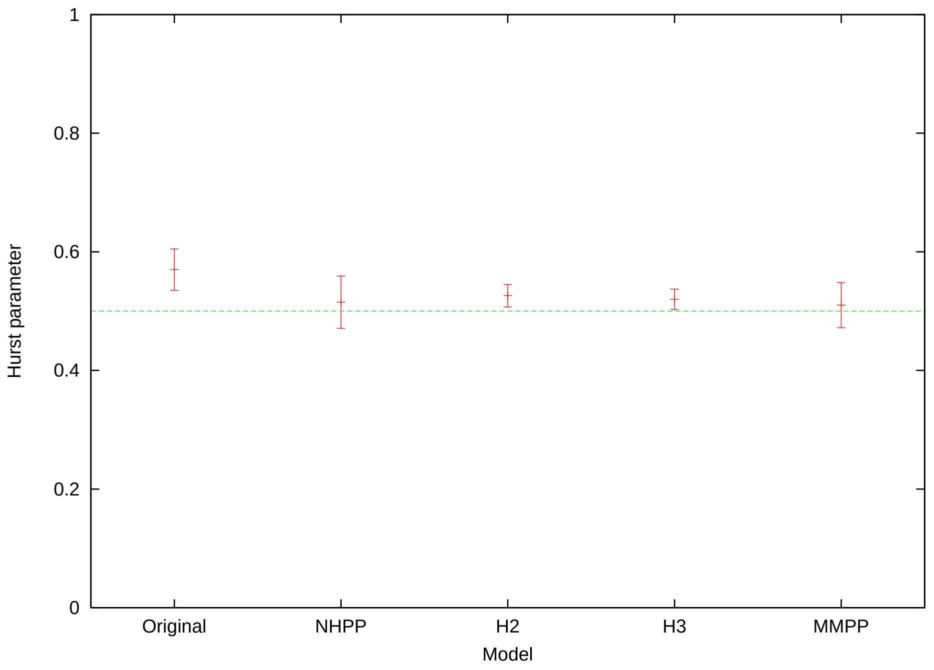


Figure 43: Hurst parameters for different models (BEgrid).

3.3.3.2 *Job CPU times*

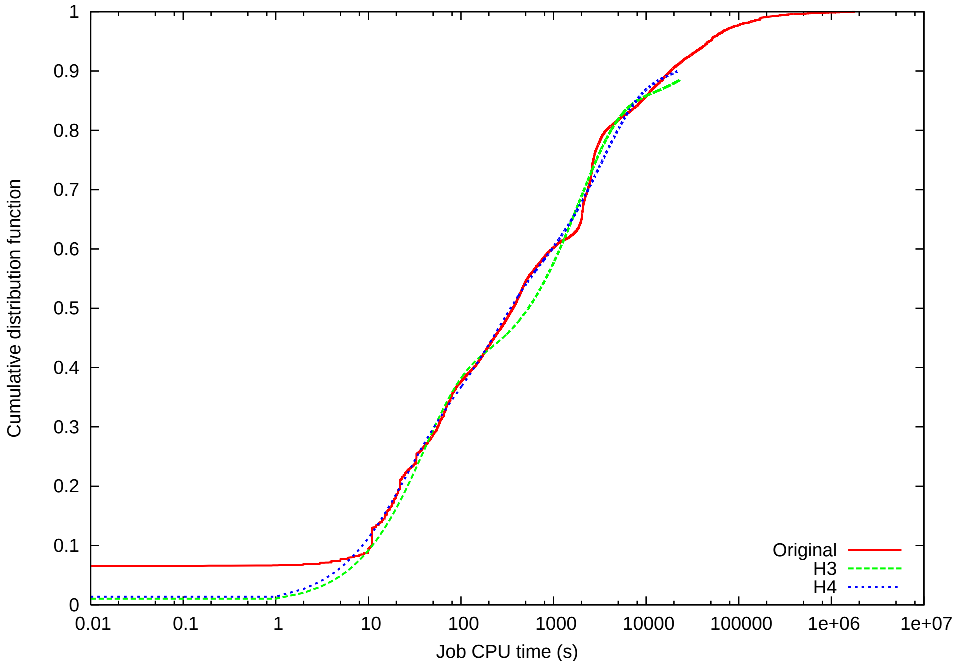


Figure 44: Job CPU times for different models (BEgrid).

The job CPU times are shown for different fitting models in Figure 44. We can observe a very good correspondance between both hyperexponential models and the original data. The increase from 3 to 4 phases improves the fitting only slightly, indicating that sufficient accuracy is obtained with 3 phases. Finally, these results confirm the conclusions from the Kallisto cluster; i.e. using a hyperexponential process to model job CPU times.

4 Conclusions

The purpose of this deliverable was to come up with realistic models describing typical Grid job arrival patterns and execution times. In a first part we **successfully gathered data at different aggregation stages** (Grid level vs single cluster sites) from existing, real life Grid infrastructure. [Note: Since the Phosphorus test bed is not operational yet, these were the only available measurements to date. However, since the architecture (at least from a grid user, job, perspective) will not be fundamentally different and some of the applications running on the measured infrastructure will also run on Phosphorus we deem it safe to be confident that the obtained results will be representative of what will be observed in the Phosphorus test bed]

The second portion of this document focused on proposing **candidate models for synthetic job generation** (ie. job inter-arrival times and execution times). We subsequently judged their usefulness by verifying how well the models could be fit to produce traces similar to the measurements discussed before. The models considered comprised:

- Poisson process (the classical exponential inter-arrival time distribution with mean rate λ)

- Non-homogeneous Poisson Process (NHPP; a Poisson process with time-varying λ(t))

- Hyper-exponential Process (HP; i.e. a phase type process, where the IAT is the sum of multiple Poisson phases)

- Markov-Modulated Poisson Process (MMPP; process probabilistically moving between states, each of them being a Poisson process)

- Pareto-Exponential Model (Busy periods with exponential duration and exponential job IATs during them. The times between the busy periods are distributed according to a truncated Pareto distribution)

From our fittings we conclude that:

- Job **inter-arrival times** on the observed Grid level can be successfully modelled by a Poisson process, but on the Grid site level (eg. Kallisto traces) the long range dependency needs to be taken into account and HP, MMPP or Pareto-Exponential models need to be used.

- For the **job execution times**, we achieved the most satisfactory results with a (3 phase) hyper-exponential process.

To achieve these conclusions we have also developed the necessary know-how and tool kit to analyse the traces and fits. In addition, we have implemented the models in software, allowing the generation of synthetic traces (which we validated against the measured data to judge their suitability).

5 References

[DF-B] D. Feitelson, “Workload modelling for computer systems performance evaluation”, www.cs.huji.ac.il/~feit/wlmod

[CB01] W. Cirne and F. Berman, “A comprehensive model of the supercomputer workload”, Proc. 4th IEEE annual workshop on workload characterization, 2001.

[SEY04] B. Song, C. Ernemann and R. Yahyapour, “Parallel Computer Workload Modelling with Markov Chains”, Proc. 10th JSSPP workshop, 2004.

[EM05] E. Medernach, “Workload analysis of a cluster in a Grid environment”, Proc. 11th JSSPP workshop, 2005.

[LMW06] H. Li, M. Muskulus and L. Wolters, “Modelling Job Arrivals in a Data-Intensive Grid”, Proc. 12th JSSPP, 2006.

[RTM] Real Time Monitor: http://gridportal.hep.ph.ic.ac.uk/rtm

[EGEE] The EGEE project homepage: http://public.eu-egee.org/

[GOC] http://goc.grid.sinica.edu.tw/gstat/index.html

[LCG] http://lcg.web.cern.ch/LCG/

[HGR] http://www.hellasgrid.gr/

[MAUI] Maui Scheduler: http://supercluster.org/maui

[OPBS] Open PBS: http://www.openpbs.org/

[WOSCR] http://www.cs.huji.ac.il/labs/parallel/workload/swf.html

[KFM03] T. Karagiannis, M. Faloutsos and M. Molle, “A User-Friendly Self-Similarity Analysis Tool”, ACM SIGCOMM Computer Communication Review, 2003.

[EMPHT] The EMpht program: publicly available at http://home.imf.au.dk/asmus/pspapers.html

[ANO96] S. Asmussen, O. Nerman and M. Olsson, “Fitting phase-type distribution via the EM algorithm”, Scnd. J. Statist. 23:419-441, 1996.

[FH93] W. Fischer. and K. Meier-Hellstern. “The Markov-modulated Poisson process (MMPP) cookbook”. Performance Evaluation, 18(2):149–171, 1993.

[MMPPF] http://www.liacs.nl/~hli/gwm/index.htm

[RED06] W. J. J. Roberts, Y. Ephraim, and E. Dieguez. “On Ryden’s EM algorithm for estimating MMPP’s”, IEEE Sig. Proc. Let, pp 373- 376, 13(6), June 2006.

[GLITE3] GLITE-3 user’s guide, https://edms.cern.ch/file/722398//gLite-3-UserGuide.pdf

[LCG-VO] https://lcg-registrar.cern.ch/virtual_organization.html

6 Acronyms

(D)WDM	(Dense) Wavelength Division Multiplexing
CDF	Cumulative Distribution Function
CPU	Central Processing Unit
IAT	Inter-Arrival Time
MMPP	Markov-Modulated Poisson Process
NHPP	Non-homogenous Poisson Process
PT	Phase Type Process
HP	Hyper Exponential Process
QoS	Quality Of Service
RB	Resource Broker
CE	Computing Element
SE	Storage Element
WN	Worker Node
UI	User Interface
VO	Virtual Orginiazation
ACF	Autocorrelation Function
EGEE	Enabling Grids for E-sciencE
LCG	LHC Computing Grid Project

Appendix A Fields of the Table constructed by the daily reports of the Real Time Monitor tool

- **EXIT TYPE**: The final-exit status of the job. There are various types, for example «REGISTERED-DONE-RAN», «REGISTERED-ABORT-CLEARED», «REGISTERED-ACCEPTED», «REGISTERED-ENQUEUED», etc.
- **FINAL REASON**: The reason for the final/exit status of a job. The job may be executed successfully, or may be aborted for a variety of reasons. There are various entries of such type. For example «7 an authentication operation failed», «Job terminated successfully», «Cannot plan: BrokerHelper: no compatible resources», «Job proxy is expired», etc.
- **FINAL EXIT CODE**: An id whose value corresponds to FINAL REASON field.
- **RB**: The name of the Resource Broker which served the specific job.
- **UI**: The User Interface from which the user has submitted the job.
- **CE**: The Computing Element to which the specific job was sent for execution.
- **VO**: The VO to which the user belongs.
- **RegisteredTimeString**: The time the user was registered in string format
- **userinterface_regjob_Epoch**: The time instance the user submitted a job from a UI.
- **networkserver_accepted_Epoch**: The time instance the network server of the RB node accepted the job.
- **workloadmanager_match_Epoch**: The time instance the Workload Management System (WMS) of the RB node start looking for the appropriate CE for executing the job according the requirements the user specified at its job description language (JDL) file.
- **jobcontroller_transfer_Epoch**: The time instance the job controller of the RB node starts sending the job for execution to the appropriate CE.
- **logmonitor_accepted_Epoch**: The time instance the CE receives the job for execution.
- **lrms_running_Epoch**: The time instance the Local Resource Management System of the Computing Element assigns the job execution to an available Worker Node.
- **logmonitor_running_Epoch**: The time instance the user files have been copied from the RB to the WNs where the job is executed.
- **lrms_done_Epoch**: The time instance the CE starts transferring the output back to the RB node.
- **logmonitor_done_Epoch**: The time instance the user can retrieve the output of his job to the UI.
- **Retrials: The number of times that the job was resubmitted to the LCG/EGEE**

Appendix B Exit Types of the jobs in the Real Time Monitor tool

The Exit Type in the Real Time Monitor tool consists of 2 hyphenated strings with the addition of 2 optional strings:

String1-String2-[String3-String4]

String1 which is related to the registration of the job with a RB, can take 2 values: REGISTERED and UNREGISTERED. A job is considered REGISTERED if the time instances (Epochs) of the registration with the user interface and the network server have been reported successfully (as the following code snippet shows):

```
if( ( userinterface_regjob_Epoch > 0 ) && (networkserver_accepted_Epoch > 0 ))
    {
     // test for a positive value of registration_Time is no good due to clock errors
     type = "REGISTERED" ;
    }
```

String2 corresponds to the final state of the job. It can take the following values: TRANSFER, ACCEPT, ENQUEUED, DEQUEUED, ABORT, CANCEL, DONE. The difference between ABORT and CANCEL is that in the former there was a middleware or hardware error while in the later the user stopped the process while it was executed.

String3 takes the value «RAN» if the time instance (Epoch) that it has finished execution is successfully reported back to the RB (as the following code snippet shows).

```
if( logmonitor_running_Epoch > 0 )
    {
     type = type + "-RAN" ;
    }
```

String4 takes the value «CLEAR» if the job was cleared (removed) by the user within 2 hours after its execution.

Table 8 presents statistics with respect to the Exit Types of the jobs for the observed period.

	Frequency	Percent	Cumulative Percent
REGISTERED-ABORT-CLEAR	43	.0	.0
REGISTERED-ABORT-RAN	8534	.4	.4
REGISTERED-ABORT	825883	37.1	37.4

REGISTERED-ACCEPTED	4680	.2	37.6
REGISTERED-CANCEL-RAN	147	.0	37.7
REGISTERED-CANCEL	16586	.7	38.4
REGISTERED-DEQUEUED	82	.0	38.4
REGISTERED-DONE-RAN	1162124	52.1	90.5
REGISTERED-DONE	163401	7.3	97.9
REGISTERED-ENQUEUED	66	.0	97.9
REGISTERED-MATCH	22	.0	97.9
REGISTERED-na	6	.0	97.9
REGISTERED-PENDING	76	.0	97.9
REGISTERED-RESUBMIS	3	.0	97.9
REGISTERED-RUNNING	2	.0	97.9
REGISTERED-TRANSFER	26715	1.2	99.1
UNDEFINED-ABORT	11337	.5	99.6
UNDEFINED-na	8424	.4	100.0
UNDEFINED-REFUSED	706	.0	100.0
Total	2228838	100.0	

Table 8: Exit Type of the jobs

In order to obtain a table that is easier to comprehend, we manipulated Table 1 and produced Table 9. To obtain Table 9 we removed the optional strings in the Exit Code (String3 and String4) and also grouped together the states that were observed with less than 0.1% percentage.

	Frequency	Percent	Cumulative Percent
REGISTERED-ABORT	834460	37.4	37.4
REGISTERED-ACCEPTED	4680	.2	37.6
REGISTERED-CANCEL	16733	.8	38.4
REGISTERED-DONE	1325525	59.5	97.9
REGISTERED-TRANSFER	26715	1.2	99.1
REGISTERED-other	258	.0	99.1
UNDEFINED	20467	.9	100.0
Total	2228838	100.0	

Table 9: Exit Type of the jobs - after manipulation