Journal of Physics: Conference Series **513** (2014) 032038 doi:10.1088/1742-6596/513/3/032038

Job scheduling in Grid batch farms

Andreas Gellrich

DESY, Notkestr. 85, 22607 Hamburg, Germany

E-mail: Andreas.Gellrich@desy.de

Abstract. We present here a study for a scheduler which cooperates with the queueing system TORQUE and is tailored to the needs of a HEP-dominated large Grid site with around 10000 jobs slots. Triggered by severe scaling problems of MAUI, a scheduler, referred to as MYSCHED, was developed and put into operation. We discuss conceptional aspects as well as experiences after almost two years of running.

1. Introduction

The EGI [1] Grid site DESY-HH at DESY in Hamburg, Germany, operates a federated multi-VO Grid infrastructure for 20 VOs and serves as a Tier-2 centre for ATLAS, CMS, and LHCb within the world-wide LHC Computing Grid (WLCG) [2]. Computing resources are used according to agreements and contracts, such as the WLCG Tier-2 Memorandum of Understanding (MoU), in which certain pledges per VO are defined. Additional and temporarily unused resources are distributed opportunistically among all VOs. The resources are provided by means of a batch system in a farm of *worker nodes* (WN), controlled by a queueing system and a scheduler. DESY-HH, as one of the world-wide largest WLCG Tier-2 centres for CMS, provides as of January 2014 a total of 10000 job slots.

One of the main goals of a Grid site is stable operations while optimally utilizing the computing resources. At large multi-VO sites computing tasks, called jobs, have a variety of different characteristics with respect to resource requirements. They range from CPU-dominated simulation to I/O-intensive analysis jobs, with large differences in CPU and wall-clock time demands from minutes to days as well as local disk space usage. It is therefore essential to intelligently distribute the jobs to the worker nodes to avoid bottlenecks of the node resources as discussed in [3]. This can be achieved by maximizing the number of jobs with different characteristics per worker node. Since it is not possible to reveal the job characteristics directly, the number of jobs of different users is maximized instead. We define the job *diversity* as the ratio of the number of different users and jobs per worker node.

In this paper we describe conceptional aspects of a scheduler for a batch system and its requirements in the environment of a multi-VO Grid site. We then present a feasibility study of a home-grown scheduler, referred to as MYSCHED, which is able to overcome severe scaling issues of the standard EMI middleware [6] installation, deploying TORQUE [4] and the MAUI [5]. The scheduler MYSCHED may even allow to include further Grid-specific information in the scheduling process such as the status of the mass storage system or the network. This feature is not covered by the schedulers usually deployed by Grid sites.

2. The Computing Element

In the Grid a job is submitted to the so-called *computing element* (CE), which maps the users X509-proxy to a POSIX users/groups (UID/GID) and submits it to the batch system, which uses the VO, group, and user of the jobs.

2.1. The batch farm

The queue set-up at DESY-HH is shown in table 1. A configuration with five queues was chosen, which reflects the mass storage system structure, to allow for easy operations and maintenance. It contains individual queues for the LHC VOs ATLAS, CMS, and LHCb, a queue for all other VOs, and a queue for monitoring purposes. The queueing system enforces the adjusted limits by rejecting excessive jobs directly before they are queued.

Table 1. Queue set-up at DESY-HH.

Queue	VO	CPU time	wall-clock time	max running	max queueable
atlas	atlas, ops	60 h	90 h	4000	10000
cms	cms, ops	60 h	90 h	4000	15000
desy	others	60 h	90 h	4000	10000
lhcb	lhcb, ops	60 h	90 h	4000	10000
operations	ops	15 min	1 h	50	1000

Initially one of the standard set-ups of EMI with the queueing system TORQUE and the scheduler MAUI was deployed for the batch farm. This set-up achieved an occupancy of above 90% of the job slots. It turned out though that the complexity of the scheduling algorithm of MAUI as well as its manifold configuration options were hard to control. Features to precisely control the distribution of jobs to the nodes were missing. At DESY-HH the performance of MAUI did not scale beyond 4800 running plus 10000 queued jobs. Blocking queues and unused job slots and hence low occupancy were observed. In view of the increasing demand for computing resources by the LHC experiments, which leads to a significant increase of the number of job slots, this appeared to be a severe problem.

3. The MYSCHED study

The operational experiences with the batch system at DESY-HH, in particular the problems with MAUI, let to the considerations described here. The actual development of the scheduler was finally triggered by the scaling problems of MAUI. Although there is experience with other batch systems, such as SGE and LSF, at DESY, abandoning TORQUE was considered as too risky, whereas MYSCHED could be developed and tested while MAUI was still in operation. Coding and testing of the scheduler MYSCHED started in February 2012. MYSCHED is in production from summer 2012 on.

3.1. Concept

The following assumptions were made:

- Parallelization is done on the job level. Therefore jobs are independent and self-contained and can be treated individually.
- So far one job slot per core is assigned. A dynamical assignment of multi- and single-core jobs is not supported. It is possible though to operate queues with multiple cores per job slot.

- Jobs contain VO, group, and user names.
- There is a variety of users and groups per VO which have potentially different characteristics with respect to resource requirements. If all jobs of a VO were submitted through one pilot, there would be no way to intelligently distribute jobs to the worker nodes.
- The computing is data centric because jobs massively access data files which are stored on so-called *storage elements* (SE). The access methods are versatile, ranging from copying of complete files via GridFTP to direct access via NFS4.1.
- Status and load information of the SEs is available and can be utilized.
- The queueing process is performed by TORQUE.
- TORQUE itself does not submit jobs. Job submission is handled by the scheduler.

3.2. Requirements

The following requirements were made. The scheduler should

- achieve close to hundred percent occupancy,
- be light-weighted (e.g. short processing times, little resource usage),
- guarantee at any time the number of running per VO given by the share,
- opportunistically distribute jobs to free slots by considering the wall-clock time consumption in the past (typically two days),
- optimize resource utilization by trying to maximize diversity,
- allow to adjust the scheduler's behaviour by way of a configuration file,
- allow to interface to external information (e.g. the load of the SE).

3.3. Algorithm

The scheduler is running as a separate program which is periodically executed. After reading in the configuration, the scheduling process is carried out in two steps.
Firstly, the list of queued jobs is re-ordered with the goal to have as many jobs per VO running as the share of the VO guarantees. If free job slots remain, more jobs are submitted by prioritizing VOs according to their wall-clock time consumption in the past; typically 2 days.
Secondly, in order to optimally distribute the jobs to the worker nodes, the re-ordered job list is processed job by job. For each job all nodes are checked. Those which have already reached the maximal number of jobs per VO or group are rejected. From the remaining nodes the one with the smallest number of jobs of the user and the highest diversity is selected.

3.4. Realization

TORQUE provides a C-API library to obtain information about queues, jobs, and nodes as well as to execute commands such as submitting jobs. The accounting data are available in text files. The scheduler MYSCHED was implemented in C++ and contains roughly 8000 lines of program code. It uses *Libconfig*, a library for processing structured configuration files [7]. It is compiled and linked for Scientific Linux 6. Besides the MYSCHED program a number of utility commands is available to obtain status information about nodes, queues, jobs, and accounting data. Limits on the maximal number of jobs per VO or group in total and per node can be assigned in a configuration file. In addition, types of worker nodes can be defined, e.g. to dedicate nodes with write access to the VO software area to certain groups. Nodes can be explicitly assigned to queues. Moreover, lists of queues, VOs, groups, and users (marked *hot*) to by-pass the ordering algorithm can be defined. For a complete list of parameters and some working example refer to the appendix of the proceedings. Information from the storage systems is not yet evaluated. This feature is under construction. It would allow to block VOs or groups if the associated SE is overloaded.

3.5. Operations

The MYSCHED program runs on the batch server host. At DESY-HH this is a virtual machine (Scientific Linux 6) with 2 cores and 4 GB of memory. The program is regularly executed in a loop; hence does not run as a daemon. MYSCHED logs comprehensive information in files which can be used for monitoring purposes, e.g. to fill RRDTOOL [8] databases for visualization as the figures shown in figures 2 and 3.

3.6. Experiences

Since the first production run in July 2012, the number of job slots was increased from 4800 to 10000. The comparison of MAUI and MYSCHED with respect to the job slot occupancy is shown in figure 1. The drops on the number of slots (NSlots) was caused by nodes which were temporarily taken offline for maintenance. The figures 2 and 3 show running (R) and queueing (Q) jobs of a two weeks period in fall 2013 with many active VOs. The occupancy is close to 100% as long as there are queued jobs to fill free slots. Regularly as well as sporadically submitting VOs obtain their shares as configured. Since direct information of the load of the storage elements is not (yet) available for MYSCHED, a limit on the maximal number of jobs of the VOs saves currently against overload. This limit blocks 'atlas' and 'cms' at peak times. The figures 4 and 5 show the time one scheduler execution takes (decision-time) and the job submissions of a 24 h period in fall 2013. The MYSCHED process typically consumes 150 MB of resident and 130 MB of virtual memory. 1-CPU and takes around 100 s for 1000 running plus 5000 queueing jobs in the system. Typically 10 to 100 jobs are submitted per MYSCHED run.

4. Conclusions

MYSCHED was developed as an alternative to MAUI, targeting on a solution which is tailored to the needs of a Grid site such as DESY-HH with mainly HEP-VOs. The scheduler was put into operation almost two years ago as a replacement for MAUI at DESY-HH which had shown severe scaling issues. MYSCHED achieved an occupancy of close to 100% even with an increased number of job slots of 10000 as of January 2014. The intelligent distribution of jobs to the worker nodes by maximizing the number of jobs of different users (diversity) allowed to optimize utilization of the compute resources. The performance in terms of stability, occupancy, and resource utilization of DESY-HH was significantly increased. One of the key ideas to include information from the storage system in the scheduling process has not been realized yet. This feature is investigated and might be added in future versions of the scheduler. MYSCHED can not mix single- and multi-core jobs per node. This would require mechanisms to reserve nodes until the requested number of free cores is available. Even with a back-fill mechanism to use free slots for short jobs, it would be difficult to achieve a high occupancy. The scheduling process relies on the ability to distinguish job characteristics by the group; this ansatz does not work for pilot jobs. In the extreme case of only one pilot job per VO, MYSCHED could only try to achieve a maximal diversity of VOs per node to optimize the resource usage.

References

[1] EGI: `http://www.egi.eu`,
[2] WLCG: `http://wlcg.web.cern.ch`
[3] Gellrich A 2012 J. Phys.: Conf. Series **396** 4 04022
[4] TORQUE: `http://www.adaptivecomputing.com/products/open-source/torque`
[5] MAUI: `http://www.clusterresources.com/products/maui-cluster-scheduler.php`
[6] EMI: `http://www.emi.eu`
[7] libconfig: http://www.hyperrealm.com/libconfig
[8] RRDTOOL: http://oss.oetiker.ch/rrdtool

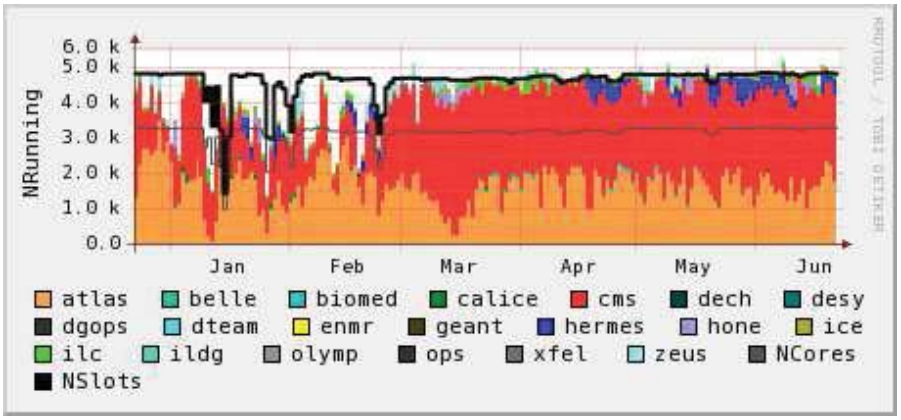

Figure 1. Job slot occupancy with MAUI (until Feb 2012) and MYSCHED (since Mar 2013).

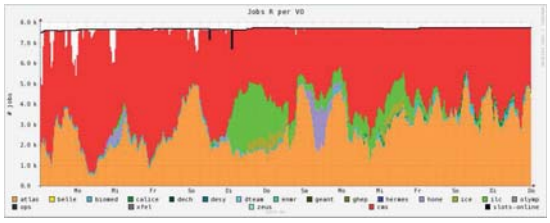

Figure 2. Running jobs per VO.

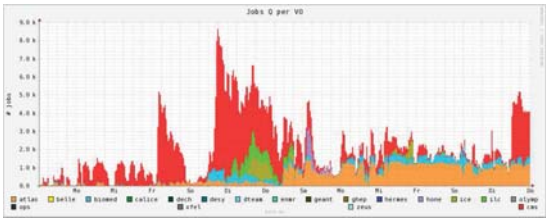

Figure 3. Queueing jobs per VO.

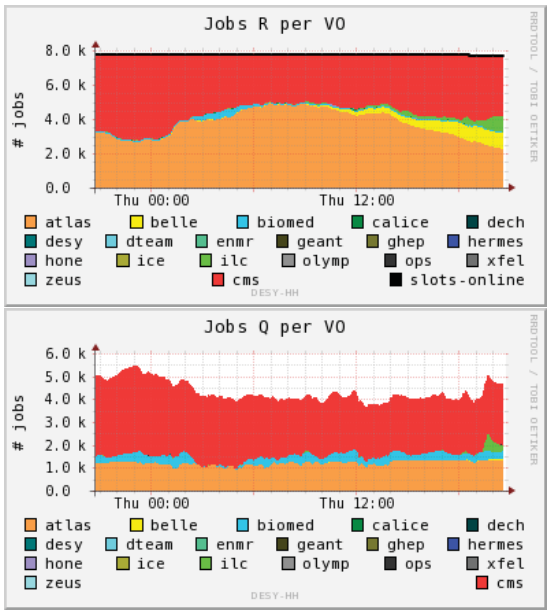

Figure 4. Running and queueing jobs.

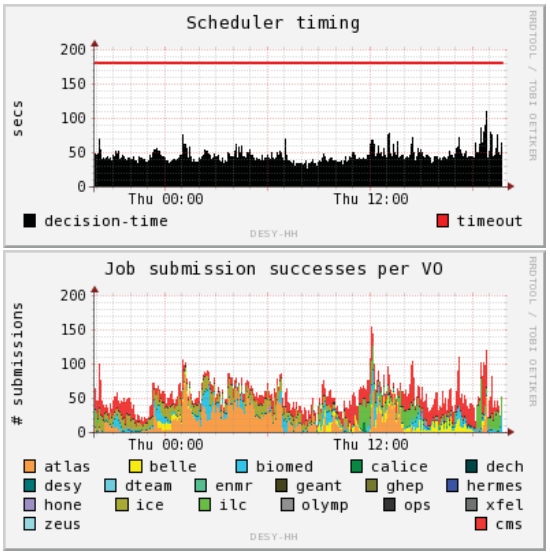

Figure 5. Time consumption and job submissions.