Dynamic provisioning of local and remote compute resources with OpenStack

M Giffels, T Hauth, F Polgart, G Quast

Karlsruhe Institute of Technology, Institut für Experimentelle Kernphysik, Wolfgang-Gaede-Str. 1, 76131 Karlsruhe, Germany

E-mail: `thomas.hauth@kit.edu`

Abstract. Modern high-energy physics experiments rely on the extensive usage of computing resources, both for the reconstruction of measured events as well as for Monte-Carlo simulation. The Institut für Experimentelle Kernphysik (EKP) at KIT is participating in both the CMS and Belle experiments with computing and storage resources. In the upcoming years, these requirements are expected to increase due to growing amount of recorded data and the rise in complexity of the simulated events. It is therefore essential to increase the available computing capabilities by tapping into all resource pools.

At the EKP institute, powerful desktop machines are available to users. Due to the multi-core nature of modern CPUs, vast amounts of CPU time are not utilized by common desktop usage patterns. Other important providers of compute capabilities are classical HPC data centers at universities or national research centers. Due to the shared nature of these installations, the standardized software stack required by HEP applications cannot be installed. A viable way to overcome this constraint and offer a standardized software environment in a transparent manner is the usage of virtualization technologies. The OpenStack project has become a widely adopted solution to virtualize hardware and offer additional services like storage and virtual machine management.

This contribution will report on the incorporation of the institute's desktop machines into a private OpenStack Cloud. The additional compute resources provisioned via the virtual machines have been used for Monte-Carlo simulation and data analysis. Furthermore, a concept to integrate shared, remote HPC centers into regular HEP job workflows will be presented. In this approach, local and remote resources are merged to form a uniform, virtual compute cluster with a single point-of-entry for the user. Evaluations of the performance and stability of this setup and operational experiences will be discussed.

1. Introduction

Grid computing in general and the Worldwide LHC Computing Grid (WLCG) specifically are an extremely successful implementation of the concept of distributed computing and storage and have enabled spectacular discoveries in recent years. The most prominent is arguably the confirmation of the Higgs Boson by the CMS and ATLAS Collaborations at the LHC in the year 2012. The LHC is expected to resume its operation at an increased center-of-mass-energy of $\sqrt{s} = 13\,\text{TeV}$ in the summer of 2015. Other large high energy physics (HEP) experiments are also scheduled to begin operation in the coming years, like the Belle II detector at the SuperKEKB high-intensity collider in the year 2018.

One thing these experiments have in common is that enormous amounts of computation and storage requirements need to be fulfilled to achieve the best possible science performance. Data recorded by these experiments needs to be reconstructed, stored and analyzed. Furthermore, Monte-Carlo simulations need to be performed in order to compare the recorded data to model predictions. Depending on the experiment, at least a comparable amount of simulated events needs to be available in order to achieve a sufficient statistical accuracy. In practice, many experiments need two or three times more simulated events than recorded ones in order to test for different models. With the increased data rates of the next-generation HEP experiments, the computing requirements will also grow in the coming years.

Classically, the Grid infrastructure is installed on dedicated hardware resources which are located in data centers and tailored to serve as Grid computing sites. This allows for a high specialization of the installed hardware, software and trained personnel. The downside is, that these data centers can only be used by Grid computing workloads. Often universities and funding agencies encourage research groups to share a common cluster installation and profit from the economies of scale effect. These cluster installations are mostly tailored to suit high performance computing (HPC) workloads, which require a tight interconnect between individual worker nodes in the cluster.

Most shared HPC-centers have to provide an operating system which can be used by all involved research groups. With the Scientific Linux [1] distribution, the HEP community maintains a custom operating system to achieve reproducibility and standardization among the many participating Grid sites. Most experiment-specific software in use today requires an underlying Scientific Linux installation to fulfill the requirements set by the experiment's collaborations.

The opportunities for HEP-research to profit from such shared installations exist and in light of the aforementioned computing challenges, every effort should be taken in order to have access to these resources. One possibility to benefit from a shared cluster setup and still be able to support specialized HEP workloads has been implemented by the EKP with the ViBatch system. Here, a classical batch system on an HPC cluster is extended to start a virtualized operating system for a specific class of HEP workloads [2].

Since the ViBatch installation, Cloud computing and virtualization have become a well-established technology in recent years and provide ideal means to make use of common data centers which have not been tailored to satisfy HEP needs.

2. Cloud Computing and Virtualization

The term Cloud computing covers a wide area of applications and services. For this document, the more well defined Infrastructure-As-A-Service (IaaS) paradigm is used synonymously with Cloud computing. IaaS describes the concept to provide an abstract set of resources, like compute, storage and network, transparent of the underlying hardware and configurable and scalable by the requesting user. One of the enabling technologies is virtualization, foremost of CPUs, to allow users to boot customized operating systems and application software stacks. The virtualization of network services like interconnect, routing and firewall isolates the network traffic of individual users and therefore increases the operational security of a multi-tenant cluster.

Today, many commercial providers offer IaaS products, mostly on a pay-per-use basis. Among the biggest are the Amazon EC2 service and Rackspace. These commercial services have been successfully used to execute Grid workloads [3] and integrating EC2 worker nodes into an institute's local batch system was successfully demonstrated at EKP with the ROCED cloud scheduler [2].

Additionally, more and more research institutions provide easy access to Cloud services for their users. The CERN OpenStack Private Cloud [4] offers users the possibility to upload virtual machine (VM) images and to start and stop them at the user's discretion. Furthermore, Cloud computing has been successfully used on the High-Level Trigger farms of the CMS and ATLAS experiments as Grid sites in times where no data from the detectors needs to be processed. Here, Cloud computing is not primarily employed to boot custom VM images but to allow for a fast turn-around between the HLT and Grid operations modes [3] [5].

3. Technology Choices

In the last couple of years, a significant growth in available Cloud computing technologies could be observed. With the introduction of hardware-accelerated virtualization by Intel and AMD in the mid-2000s, projects like the kernel-based Virtual Machine (KVM) became attractive, as the massive performance loss, which was previously associated with virtualization, was mitigated. In the late 2000s, more complete IaaS software suites, like Eucalyptus and OpenNebula emerged and augmented the bare virtualization with image and VM lifetime management and cluster scheduling.

The technologies, which have been chosen to integrate remote HPC resources into HEP workflows described in this document, will be introduced. Many similar or complimentary products exist on the market today and the selection made is based on various factors which will be listed with each product. For other use cases, one might arrive at a different selection, which might be better suited.

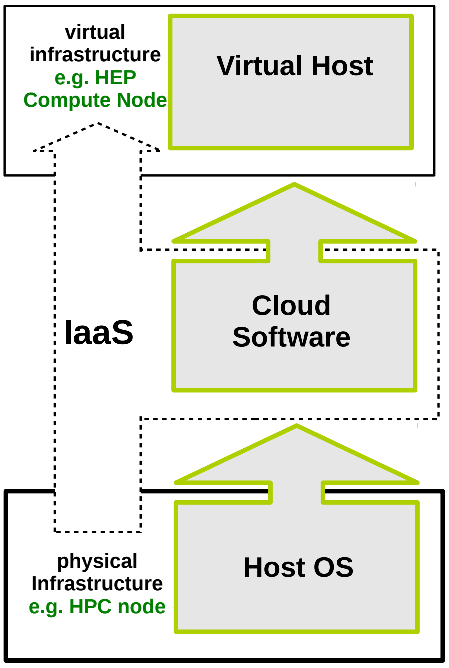

Figure 1. The Infrastructure-As-A-Service paradigm can be employed to virtualize a physical infrastructure.

3.1. OpenStack

Today, the open-source OpenStack application [6] has emerged as one of the most widely used and supported IaaS implementations. The functionality of OpenStack is implemented as a range of so-called services which are foreseen to run in a distributed and fault-tolerant manner across multiple physical hosts. Not all available OpenStack services have been installed in the presented setup and therefore only a subset of the OpenStack application suite will be described. The interested reader can find information on all OpenStack services in the corresponding documentation [7].

The Keystone service is the central component to manage user credentials and the associated privileges. The Nova service is split into a control component, which runs on a central controller node and a nova-compute component, which is installed on each worker node hosting virtual machines. Worker nodes hosting virtual machines are also named *Hypervisors*. The Nova service is responsible for scheduling VM requests to free cluster resources and to boot and terminate the VM on the hypervisors. The Neutron service takes care of creating a virtual network for the started VMs and to route traffic among VMs and to outside networks. Custom virtual machine images can be uploaded and managed via the Glance service and the Horizon service offers a HTML-based frontend to most of OpenStack's features to the user. Some auxiliary services, like MySQL, also need to be installed to enable the internal communication and persistency of OpenStack.

The OpenStack system has already been adopted by some HEP-related data centers, like the CERN computing center, and CERN personnel play an active role in the OpenStack foundation board. This gives us confidence that OpenStack will see an even wider adoption in the HEP-computing field and more HEP software suites and tools will have built-in OpenStack support.

3.2. HTCondor

The open-source HTCondor project provides a workload management system which is highly configurable and modular [8]. Batch processing workflows can be submitted and are then forwarded by HTCondor to idle resources. HTCondor maintains a resource pool, which worker nodes in a local or remote cluster can join. Once HTCondor has verified the authenticity and features of the newly joined machines, computing jobs are automatically transferred. Special features to connect from within isolated network zones, for example via a NAT-Portal, to the central HTCondor pool are available. The Connection Brokering (CCB) service is especially valuable to connect virtual machines to the central pool [8]. These features and the well-known ability of HTCondor to scale to O(100k) of parallel batch jobs lets us decide to use HTCondor as a workload management system.

3.3. OZ image Provisioning Toolkit

In order to generate custom virtual machines images, which can be uploaded and started at remote HPC sites, we choose to employ a fully-automated procedure. Here, the OZ toolkit [9] has been selected. All requirements and configuration options of an image can be specified via an XML file, called template. The partitioning and installation process of the operating system is fully automated, as OZ will use the remote-control capabilities of KVM. After the installation of the operating system, additional libraries and configuration files can be installed. Once the image has been built, it is automatically compressed and uploaded to the OpenStack Glance service.

Using this technique allows to build images in a reproducible fashion, as all templated files are version controlled using git. Furthermore, existing template files are easy to adapt to new sites and experiment configurations.

4. Utilizing local idle resources with OpenStack

The EKP provides individual work stations to each researcher and student. While individual workstations are important to support web browsing and communication tasks, we observe that most users prefer to perform their computation-heavy tasks, like compiling software and running analysis code, on dedicated portal servers and batch farms. Idle resources of these workstations can be exploited for the execution of CPU-intensive tasks, like Monte Carlo simulations, alongside the regular desktop applications.

One important factor has to be considered for this system to succeed on a social level: the daily work tasks of the desktop users must not be impeded in any way by running the virtual machines. Therefore, we decided to only integrate machines into our institute's OpenStack cloud which have been bought with virtualization's requirements in mind.

Over the last one and a half year, our institute procured new desktop machines, needed anyway to replace older ones, with 4-core CPUs and 16 GB RAM. This configuration ensures that sufficient CPU and memory resources are available for both the users daily tasks and to host a virtual machine.

At the writing of this paper, 15 desktop workstations have been integrated into the local OpenStack system and provide in total 60 cores usable via virtual machines. The central OpenStack services, like Keystone and Nova, have been installed on one physical machine, which is equipped with a 6-core CPU and 32 GB RAM and is sufficient for an OpenStack installation of our size.

The booted virtual machine images are almost identical to the ones used on remote cloud sites. As one additional features, the local image mounts the NFS-based storage of the institute's file servers on startup so users are able to read and write to their local data directory directly.

5. Integration of Remote HPC Centers

In the following, the installation and operation of worker nodes located at a remote HPC site will be presented. Although this setup is described using the three endpoint locations involved, the used concept is generic and can be adapted to different locations and network configurations.

The shared compute cluster bwForCluster will be installed at the University of Freiburg, which is located around 150 km south of Karlsruhe. The state funding has been secured and the full installation with an expected number of 8000 cores will be installed in fall 2015. The cluster is shared among three diverse groups of users: Particle Physics, Neuroscience, Microsystems technology. Early in the conception phase, it became evident that virtualization is a key technology to allow for an efficient sharing among the user groups. Especially for the specific needs of HEP-Users, like custom operating system and software library versions, having the flexibility to start custom VM images is essential.

A dedicated 10 Gbit/s link between the Freiburg data center and the WLCG Tier-1 GridKa data center in Karlsruhe allows for an efficient transfer of input and output data to the Freiburg site.

For the concept and benchmarks described here, a test system for the final bwForCluster installation was used. It consists of 800 CPU cores and a fully functional OpenStack system.

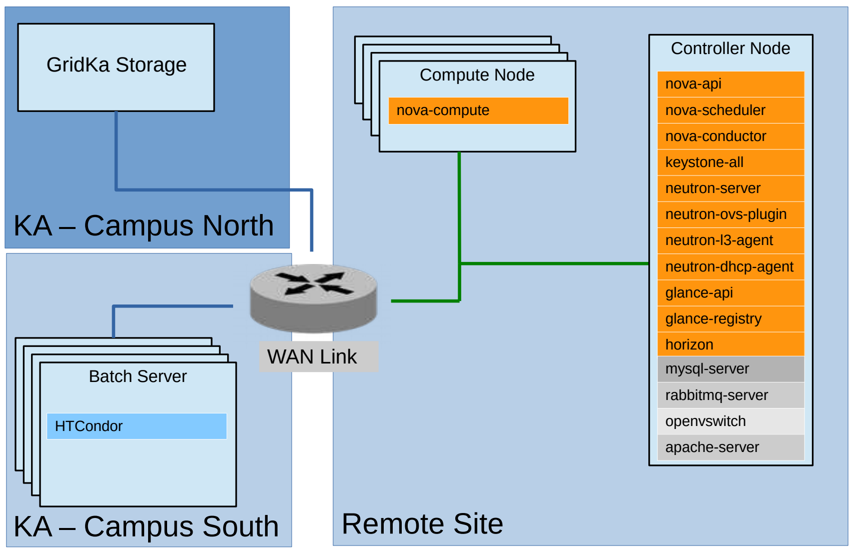

Figure 2. Structure and location of the local and remote components.

Figure 2 visualizes the structure of the three physical sites involved. The OpenStack control node and compute nodes are located at the remote site, in the here described case at the Freiburg data center. The worker nodes use the private WAN link to connect to the central HTCondor batch server at the institute's local data center, which is located at the KIT Campus South. Input and output data is transferred to the GridKa Storage Element located at the KIT Campus North.

6. Generating Virtual Machine Images with OZ

The virtual machines booted at the remote Cloud center were created using the OZ toolkit and the CERN OpenStack SL6.5 image templates [10] were used as a starting point. A lot of modifications were added to the bare Scientific Linux installation. The HTCondor daemon, which integrates the VM at boot-time into the resource pool, was installed. A shared secret key is used to authenticate the remote machines before they are added to the pool. Furthermore, CVMFS [11] was installed to serve the Grid UI and CMS software. Finally, some superficial files were removed to decrease the size of the VM image.

7. Job Submission and Workflow Management

One important goal of this system is to allow the Institute's user group seamless and hassle-free access to both the Institute's local and the remote Cloud resources. Therefore, users can submit their jobs using the regular condor submit command. The EKP institute also operates a local OpenStack setup, which is utilizing spare resources of desktop machines.

The recommended workflow is to use the job submission tool grid-control [12] which automatically splits workflows into a list of jobs and tracks the job's status. By selecting a specific HTCondor requirement when submitting the job, users can either submit only to local Cloud worker-nodes, with direct fileserver access, or submit only to remote Cloud worker nodes. If the user jobs are flexible enough, jobs can also be sent to both, the local and remote resources, at the same time and thereby maximize the processing throughput.

8. Challenges of a Distributed Computing Infrastructure

Once the remote HPC resources are provisioned via OpenStack and workloads can be send via HTCondor, some important questions need to be answered. The stability of the OpenStack hypervisor and virtual machines needs to be guaranteed so the jobs will not randomly fail. Furthermore, the stability of the HTCondor integration is an important requirement.

Depending on the processed workload, large amounts of input or output data need to be transferred to and from the virtualized worker nodes. Therefore, the network connections and routing must be reliable, as random network disconnects or slowdowns are not tolerable. The following sections will describe some of the studies which have been done to answer these questions for the previously described setup.

9. Data-transfer Rates and Reliability

To simulate a typical heavy-IO workload, 400 virtual machines were running at the remote site. To test the maximal output transfer rates (Stage-Out), an artificial benchmark running in each virtual machine generated a 1 GB large file with random content and transfer this file to the Tier-1 storage element using the well-known GridFTP tools. This ensures the benchmark's behavior is replicating a typical HEP job, but enough bandwidth is used to saturate the available link with 400 running jobs.

To test the maximum input rates (Stage-In), each job copies one of ten files with a size between 250 MB to 1 GB from the GridKa Tier-1 storage element into the virtual machine.

The advantage of this benchmark method, above other more direct benchmark suites like iperf, is its ability to probe all components necessary for the data transfer of jobs. This includes the routing between local and remote site, firewalls, the network virtualization of OpenStack, but also the GridKa storage elements and the GridFTP client software.

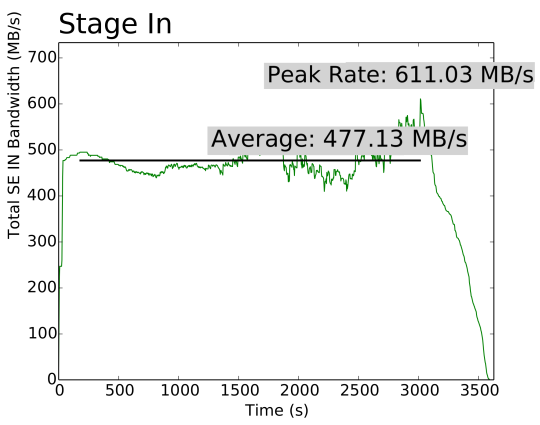

Figure 3. Total input bandwidth.

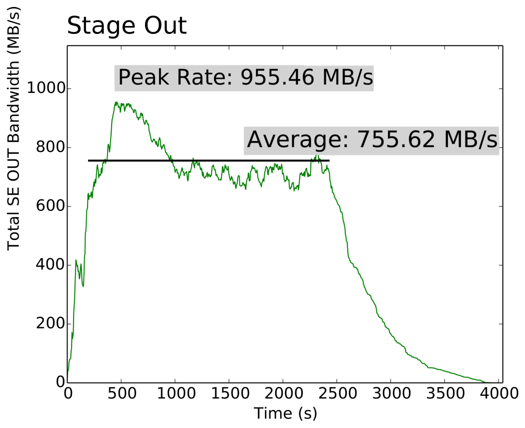

Figure 4. Total output bandwidth.

As visible in Figure 4, plotting the output bandwidth over the test period, the 10 Gbit/s is saturated with an average data throughput of 755 MB/s while a maximum rate of up to 955 MB/s can be reached. Figure 3 shows the input bandwidth when reading files from the GridKa Tier-1 storage pools. The available bandwidth is not fully saturated here, but files can also be transferred with an excellent rate of 477 MB/s. During all network transfer tests, no canceled or stalled file transfers were observed, which indicates a good overall network stability.

10. Long-term Example: QCD Computations

As an additional benchmark, a long-term QCD workflow was performed using the remote worker nodes. Here, batch jobs are submitted by a researcher to the local HTCondor and then automatically transferred to the remote worker nodes. Input and Output data of the jobs is transferred from a remote Grid Storage Element. The FastNLO [13] package is used to compute QCD interpolation tables. This kind of workload has moderate IO requirements but a high demand in CPU.

Over the course of two month more than 20000 jobs haven been processed on the Cloud worker nodes without problems. Figure 5 displays the accumulated cpu time during the benchmark period. Figure 6 shows the accumulated number of processed jobs over the same time period. During the days 25 to 52, no new jobs were submitted to the system due to the Christmas break. Apart from this break, the system was mostly processing jobs without any downtime.

No problems with either the virtualization or the batch node integration of HTCondor were observed during the benchmark period of two month. This underlines the excellent reliability which the used software suites, OpenStack and HTCondor, already have and the overall stability of the integration of both, which is part of the work presented here.

For the computation campaign presented here, the required virtual machines have been manually requested via OpenStack's dashboard user interface. Since the time of the benchmark, the ROCED cloud scheduler has been installed which requests and terminates virtual machines based on the amount of jobs available at the local HTCondor system.

11. Conclusion

The ability to harness virtualization technologies is essential to allow access for the HEP community to shared HPC clusters, commercial Cloud service provider and other common

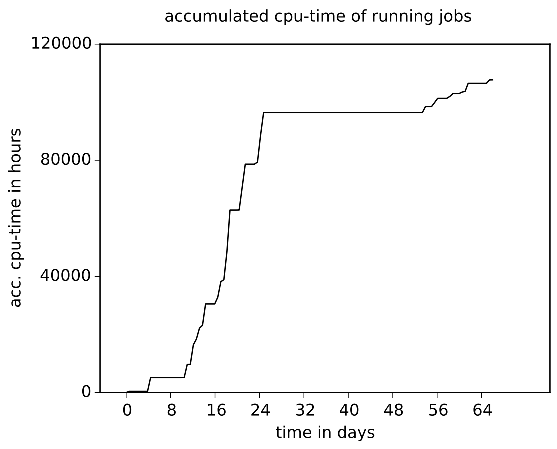

Figure 5. Accumulated cpu time.

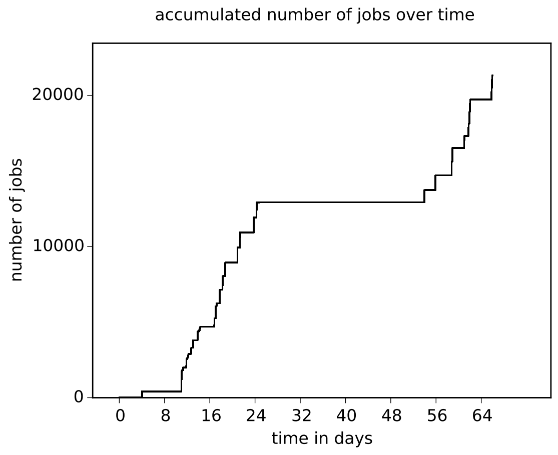

Figure 6. Accumulated number of jobs.

computing installations. As the computing requirements of the next-generation HEP experiments increase, theses avenues to procure computing time can no longer be ignored.

Mature software for virtualization (KVM, OpenStack) and flexible workload management (HTCondor) exists today. As presented, these products can be combined to create a dynamic and reliable way to bring traditional HEP workloads to remote HPC resources: remote computing resources can be requested at peak times and shut down if not needed. Furthermore, the same concept can also be easily adapted to tap into commercial Cloud providers.

The use-case of the bwForCluster project has been presented. It uses virtualization to solve the diverse requirements of its tenants. First evaluations of the bwForCluster test system were successful and we are convinced the shared cluster model can work on a technical level and bring benefits to our HEP groups. The further procurement of the final bwForCluster system will be accompanied and the final installation will provide a significant addition to the compute resources available to the institute's researchers.

[1] ScientificLinux website `https://www.scientificlinux.org/` (10.5.2015)
[2] Oberst O, Hauth T, Kernert D, Riedel S and Quast G 2012 *Journal of Physics: Conference Series* **396** 032081 URL `http://stacks.iop.org/1742-6596/396/i=3/a=032081 (10.5.2015)`
[3] Panitkin S, Megino F B, Bejar J C, Benjamin D, Girolamo A D, Gable I, Hendrix V, Hover J, Kucharczyk K, Llamas R M, Love P, Ohman H, Paterson M, Sobie R, Taylor R, Walker R, Zaytsev A and Collaboration t A 2014 *Journal of Physics: Conference Series* **513** 062037 URL `http://stacks.iop.org/1742-6596/513/i=6/a=062037 (10.5.2015)`
[4] CERN OpenStack Private Cloud Guide `http://clouddocs.web.cern.ch/clouddocs/` (10.5.2015)
[5] Colling D, Huffman A, McCrae A, Lahiff A, Grandi C, Cinquilli M, Gowdy S, Coarasa J A, Tiradani A, Ozga W, Chaze O, Sgaravatto M and Bauer D 2014 *Journal of Physics: Conference Series* **513** 032019 URL `http://stacks.iop.org/1742-6596/513/i=3/a=032019 (10.5.2015)`
[6] OpenStack website `https://www.openstack.org` (10.5.2015)
[7] OpenStack documentation website `http://docs.openstack.org/` (10.5.2015)
[8] HTCondor website `http://research.cs.wisc.edu/htcondor` (10.5.2015)
[9] OZ website `https://github.com/clalancette/oz` (10.5.2015)
[10] CERN OZ templates website `https://github.com/cernops/openstack-image-tools` (10.5.2015)
[11] CVMFS website `http://cernvm.cern.ch/portal/filesystem` (10.5.2015)
[12] Grid-control website `https://ekptrac.physik.uni-karlsruhe.de/trac/grid-control` (10.5.2015)
[13] FastNLO website `http://fastnlo.hepforge.org` (10.5.2015)