High precision calculations of particle physics at the NEMO cluster in Freiburg

C. Heidecker, M. Giffels, G. Quast, K. Rabbertz, M. Schnepf
Institut für Experimentelle Teilchenphysik (ETP)
Karlsruher Institut für Technologie (KIT)
Karlsruhe, Germany

***Abstract*— At the Large Hadron Collider of CERN in Geneva particles such as protons are smashed onto each other at the highest man-made energies ever. Despite being gigantic in size, custom-designed detectors are capable of recording billions of such collisions with high accuracy. To accompany these measurements, predictions of highest precision are required, for some of which it took the theory community more than 20 years to develop the necessary methods and techniques. We explain how these CPU intensive calculations have been adapted to run efficiently on the computing resources available within the bwHPC project and present the required tools and infrastructure we developed for this purpose.**

Keywords—particle physics; software deployment; resource allocation

I. INTRODUCTION

To fully profit from precise measurements of high-energy particle collisions by modern particle detectors, these measurements must be matched with at least equally precise theoretical predictions. For the example of proton-proton collisions at the Large Hadron Collider (LHC) of CERN in Geneva, the theory of choice is quantum chromodynamics (QCD) [1], which is assumed to describe all phenomena involving the strong interaction or, in more common terms, nuclear interactions. Although solutions to the equations of QCD are not known in general, particle collisions at high energy, or equivalently very small distance, can be described by a perturbative approach to QCD (pQCD). Leading-order (LO) results in pQCD have been derived already in the seventies but provide only an order-of-magnitude estimate for the production rates of particular event types. More specifically, the example events used in this article contain at least one high-energetic "particle jet", which is the name given to a bundle of strongly interacting particles leaving a collision in roughly the same direction. Such jets are considered to be the visible manifestation of otherwise not freely observable proton constituents like quarks [2]. Next-to-leading order (NLO) predictions for jet production improve the relative precision to a level of about 5-10% and have become available in the nineties [3,4]. The computing time (CPU time) required for one such calculation is of the order of 1,000 hours. The pQCD calculations at next-to-next-to-leading order (NNLO) needed nowadays for an ultimate precision of 1-2% have recently been completed [5]. They require more than 100,000 hours of CPU time per desired observable.

In the following we describe the solutions we developed for the deployment of task-specific software to the worker nodes and for an efficient usage of the shared resources of the NEMO high-performance computing (HPC) cluster in Freiburg [6]. Their successful application is exemplified by means of a typical NNLO computation involving jet production.

I. SOFTWARE ENVIRONMENT

Large experimental collaborations in particle or high energy physics (HEP) develop and utilise their software within an environment adapted to their workflows. This implies the use of a particular operating system as well as numerous specific software packages. Computing resources like the NEMO HPC cluster that serve multiple user communities in the state-wide bwHPC concept cannot provide dedicated worker nodes without compromising their usability by all users. Therefore, a solution is sought to provide the software specific to the HEP community to the worker nodes of the NEMO cluster without perturbing other applications.

A. Static provisioning of software

One part of the solution is based on the virtualisation option of the HPC centre [7], which makes it possible to provide community-specific environments. Such a preconfigured environment can comprise i.a. a dedicated operating system with pre-installed HEP specific software packages.

B. Dynamic provisioning of software

Additionally, workflow- and user-specific software and tools that change frequently need to be provided. Here, only a more dynamical solution than virtualisation is feasible.

For purpose B we employ a remote file system called CERN Virtual Machine File System (CVMFS) [8] that is commonly used in HEP. This file system was developed to deliver software packages to all world-wide distributed LHC computing centres.

On the server side, files are hosted on web servers, which are organised in repositories. The utilisation of the HTTP protocol simplifies the world-wide distribution of files as well as the usage of caching proxies to reduce incoming traffic. In their standard configuration such proxies [9] cache all files with short life-cycles that are transferred via the HTTP protocol. Hence, software packages can either be provided directly via CVMFS or via other web services.

On the client side, CVMFS provides a common POSIX read-only file system in user space, which allows the user transparent access to files inside the repositories. To further reduce traffic, each CVMFS client uses an additional cache.

For usage within the NEMO HPC cluster, we installed two caching proxies, which serve as central cache and provide CVMFS to all worker nodes. Each one has a cache size of 240 GBytes and is directly connected to the internal worker node network. For even more flexibility, we have set up our own CVMFS server at KIT in addition to the CVMFS repositories provided by CERN.

This concept of static provisioning of the basic software environment via virtual machine images and the dynamic loading of workflow- and user-specific software packages via CVMFS has been operated successfully at the NEMO HPC cluster and is now in standard use for large-scale computing campaigns within our physics workflows. As a consequence, we were also able to adapt and test it for the deployment of our specific software environment to other shared resources like commercial cloud providers. Furthermore, other scientific communities with similar requirements could equally profit from the developed concept.

II. RESOURCE PROVISIONING

A. *Usability requirements*

For an efficient usage of the shared resources at the NEMO cluster, a dynamic setup is required that matches resource allocation to demand of computing power. Furthermore, a simplified and standardised system to submit workflows is necessary to make virtualised NEMO resources transparently accessible to end-users at the KIT.

Since the HEP community at the KIT manages computing resources via the batch system HTCondor [10], we have chosen to integrate the NEMO resources into our HTCondor pool. This necessitates an interface between the HTCondor system, which manages our workflow processing, and the Moab/Torque batch system [11] that is responsible for the resource management at NEMO. The life-cycle management of the virtual machines is performed by the OpenStack instance [12] at NEMO. For a successful operation, all three components must interact with each other.

B. *Virtualisation management at NEMO*

At the NEMO cluster, communication between the OpenStack instance and the Moab batch system is required to match virtual machines to allocated resources. Here, the assignment is handled by an automated system developed by the NEMO HPC team. For the integration into our HTCondor pool, the demand of resources has to be translated into requests to the NEMO cluster. Furthermore, newly allocated resources must be added, while unused resources are deallocated and removed from our HTCondor pool.

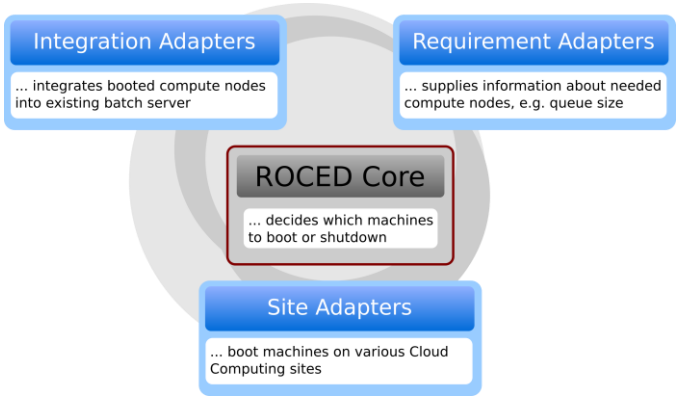

Fig. 1. Modular structure of ROCED based on three independent adapters connected via the ROCED core.

C. *Resource scheduler ROCED*

As a solution, we developed the resource scheduler responsive on-demand cloud-enabled deployment (ROCED) [13], which is designed to interact with various batch systems and resource providers. ROCED monitors the demand for resources and the current state of the resource pool. If the demand is higher than the amount of free resources, ROCED sends a resource request to a provider such as an HPC centre or a cloud provider.

For that purpose, a modular structure of independent adapters was built around the ROCED core. These include requirement adapters, site adapters, and integration adapters as shown in Fig. 1. The requirement adapter monitors the HTCondor pool at the KIT and collects information about pending workflows waiting for free resources. Simultaneously, the site adapter manages the requested resources at the NEMO cluster. After detecting a resource demand, the ROCED core decides to request additional resources via the corresponding site adapter that communicates with the resource provider. In our setup, this is performed by sending a batch job with a request for booting an OpenStack virtual machine to the Moab batch system at NEMO as shown in Fig. 2. Subsequently, the integration adapter adds the new resources into the HTCondor pool at KIT. The fact that HTCondor worker nodes automatically connect to the configured pool at startup simplifies this step.

In our current setup, a maximum runtime of one day is configured for the virtual machines. Up to 8,000 CPU cores, the maximum that can be allocated to one user, can be used. Each of these virtual machines only accepts user workflows with a suitable runtime and automatically shuts down either when reaching its runtime limit or when being idle for some time. For testing reasons, the maximum was raised to over 11,000 CPU cores so we could demonstrate the capability of

our setup to dynamically request, boot up, and shut down such an amount of virtual machines within a short duration. This demonstrated flexibility is also observed during normal daily operation. Fig. 3 shows our usage of the NEMO cluster as a function of time for one month. The blue area represents the number of CPU cores running user workflows, while the red area corresponds to the number of idle CPU cores. For the latter, matching workflows could not be found because of memory, disk space, or runtime constraints resulting in a small share of allocated but unused resources. Red and blue numbers together represent the total number of CPU cores allocated at the NEMO cluster. The excess of idle CPUs that can be seen versus the end of the one-day runtime of the virtual machines is caused by the fact that jobs with a suitable runtime cannot be found anymore and new virtual machines must be started.

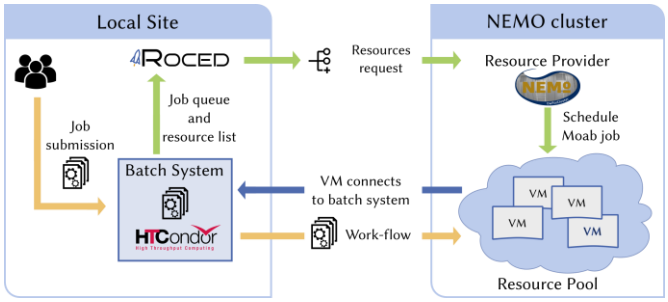

Fig. 2. Management cycle of ROCED for interaction between our local user group at KIT and the NEMO cluster.

The demand for computing resources varies over time. Hence, a highly dynamical setup like ours facilitates the efficient usage of shared resources, which are allocated only during processing of workflows and are given back afterwards. Also, the resources of large (commercial) computing providers can be accessed in a similar way opening up additional possibilities for peak demands of scientific projects.

III. Application Workflow

A typical pQCD calculation at NNLO performed by us on the NEMO cluster proceeds in three steps. First, the setup for the numerical integration of many complex multidimensional integrals must be optimised. This is performed through direct submission with Moab of one multicore job per job type of the NNLOJET program [5] and needs about 2,000 hours of total CPU time. Secondly, interpolation grids for later fast reevaluations for varying input parameters such as assumptions on the proton structure are prepared by exploring the phase space in terms of proton momentum fractions and the energy scale accessible in the examined process. Two software packages are available for this technique, fastNLO and APPLGRID [13,14], which can save enormous amounts of computing time by avoiding to rerun the full NNLO computation in many cases. We apply the latest version of the fastNLO framework [15]. In the third step thousands of single-core jobs are submitted in parallel, each for about 20h of CPU time, in order to evaluate the prepared integrals by generating

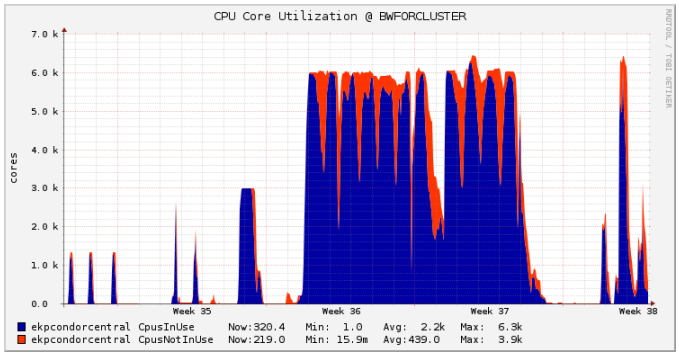

Fig. 3. Allocation and deallocation of virtual machines during one month.

pseudo-events and filling the interpolation grids. Table I gives an overview of the resources typically required in this last step. In a test campaign on the NEMO cluster we were able to finish one such computation in only two days with at maximum 7,800 cores working in parallel.

Table 1. Production Campaign of a typical pQCD calculation at NNLO accuracy. The job type corresponds to different parts of such a computation, which are evaluated separately.

Job type	# Jobs	Events / job / millions	CPU time / job [h]	# Events / billions	Total output [GBytes]	Total CPU time [h]
LO	10	140	20.6	1.4	0.024	206
NLO-R	200	6	19.0	1.2	1.3	3,800
NLO-V	200	5	21.2	1.0	1.2	4,240
NNLO-RRa	5,000	0.06	22.5	0.3	26	112,500
NNLO-RRb	5,000	0.04	20.3	0.2	27	101,500
NNLO-RV	1,000	0.2	19.8	0.2	6.4	19,800
NNLO-VV	300	4	20.5	1.2	2.0	6,150
Total	11,710	–	–	5.5	64	248,196

IV. Summary

To cope with frequent changes in community- and user-specific software environments we use the virtualisation option of the NEMO cluster to provide a static virtual machine image and dynamically load workflow specific software via CVMFS. For an efficient and user-friendly utilisation, we integrate virtualised resources of the NEMO cluster into our HTCondor batch system. To manage these resources dynamically, we employ the resource scheduler ROCED developed at KIT.

Precision measurements at the forefront of physics require equally precise theoretical predictions leading to the most CPU intensive workflow we have successfully exercised with

the described setup and the large resources available to us thanks to the NEMO cluster in Freiburg. As demonstrated, the presented components allow us to run workflows with about 100,000 CPU hours in just days. Moreover, our presented concepts can be adapted to similar needs of other scientific communities and can facilitate the access to resources of commercial computing providers for example in times of peak demands.

ACKNOWLEDGMENT

The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant no INST 39/963-1 FUGG.

REFERENCES

[1] R.K. Ellis, W.J. Stirling, B.R. Webber, "QCD and Collider Physics", Cambridge University Press, 1996.

[2] K. Rabbertz, "Jet Physics at the LHC", Springer, Berlin, 2016.

[3] S.D. Ellis, Z. Kunszt, D.E. Soper, "One-jet inclusive cross section at order alpha_s^3: Quarks and Gluons", Phys. Rev. Lett. 64 (1990), 2121.

[4] W.T. Giele, E.W.N. Glover, D.A. Kosower, "Higher order corrections to jet cross-sections in hadron colliders", Nucl. Phys. B 403 (1993) 633.

[5] J. Currie, E.W.N. Glover, J. Pires, "Next-to-Next-to Leading Order QCD Predictions for Single Jet Inclusive Production at the LHC ", Phys. Rev. Lett. 118 (2017) 072002.

[6] NEMO HPC Cluster https://www.hpc.uni-freiburg.de/nemo accessed 01.03.2018

[7] K. Meier et al., "Dynamic provisioning of a HEP computing infrastructure on a shared hybrid HPC system", J. of Phys., Conf. Ser. Vol. 762 (2016) 012012.

[8] P. Buncic et al., "CernVM – a virtual software appliance for LHC applications", *J. of Phys., Conf. Ser. Vol. 219 Part 04 (2010) 042003.*

[9] Squid proxy http://www.squid-cache.org/ accessed 10.12.2017

[10] HTCondor https://research.cs.wisc.edu/htcondor/ accessed 10.12.2017

[11] Moab HPC Suite http://www.adaptivecomputing.com/products/hpc-products/moab-hpc-basic-edition/ accessed 10.12.2017

[12] OpenStack https://www.openstack.org/ accessed 10.12.2017

[13] ROCED https://github.com/roced-scheduler/ROCED accessed 10.12.2017

[14] T. Kluge, K. Rabbertz, M. Wobisch, "fastNLO: Fast pQCD calculations for PDF fits", in Proc. of 14th International Workshop on Deep Inelastic Scattering (DIS 2006), doi:10.1142/9789812706706_0110.

[15] T. Carli et al., "A posteriori inclusion of parton density functions in NLO QCD final-state calculations at hadron colliders: The APPLGRID Project", Eur. Phys. J. C 66 (2010) 503.

[16] D. Britzger, K. Rabbertz, F. Stober, M. Wobisch, "New features in version 2 of the fastNLO project", in Proc. of 20th International Workshop on Deep Inelastic Scattering (DIS 2012), doi:10.3204/DESY-PROC-2012-02/165.