DOI: 10.14529/jsfi160302

Easy Access to HPC Resources through the Application GUI

Matthijs van Waveren[1], *Ahmed Seif El Nawasany*[1], *Nasr Hassanein*[1], *David Moon*[1], *Niall O'Byrnes*[1], *Alain Clò*[1], *Karthikeyan Murugan*[1], *Antonio Arena*[1]

The computing environment at the King Abdullah University of Science and Technology (KAUST) is growing in size and complexity. KAUST hosts the tenth fastest supercomputer in the world (Shaheen II) and several HPC clusters. Researchers can be inhibited by the complexity, as they need to learn new languages and execute many tasks in order to access the HPC clusters and the supercomputers. In order to simplify the access, we have developed an interface between the applications and the clusters, that automates the transfer of input data and job submission to the clusters and also the retrieval of results to the researchers local workstation. The innovation is that the user now submits his jobs to the cluster from within the application GUI on his workstation, and does not have to directly log into the cluster anymore. This article details the solution and its benefits to the researchers.

Keywords: KAUST, Remote Job Submission, Middleware, MATLAB, VASP, MedeA, ADF.

Introduction

The computing landscape of the King Abdullah University of Science and Technology (KAUST) is increasing in complexity. Researchers have access to the tenth fastest supercomputer in the world (Shaheen II [14]) and several HPC clusters. They work on local Windows, Mac, or Linux workstations. In order to facilitate the access of the HPC systems, we have developed interfaces for several research applications that automate input data transfer, job submission and retrieval of results. The user now submits his jobs to the cluster from within the application GUI on his workstation, and does not have to physically go onto the cluster anymore. This leads to reduced time-to-solution, easier access to the HPC systems, and less time spent on mastering unfamiliar Linux commands.

Remote job submission mechanisms were initially developed in the context of Grid Computing. We cite several frameworks that support remote job submission. A framework covers the user workstations where the jobs are created and the remote resources used to execute the jobs. The Globus Toolkit [4] was one of the first frameworks to support the development of service-oriented distributed computing applications and infrastructures with a remote job submission mechanism. The Uniform Interface to Computing Resources (UNICORE) [1] is a framework originating from Europe for the development of improved and more uniform interfaces to High Performance Computing and data resources. The HPC Gateway, originally called SynfiniWay [5], and before that TAO-W, is a virtualized IT framework developed by Fujitsu that provides a uniform and global view of resources within a department, a company, or a company with its suppliers. It also includes the linking of jobs in a workflow. GRIDBLAST [7] is a framework used to execute Life Science Basic Local Alignment Search Tool (BLAST) searches on a grid consisting of a server and remote worker nodes. eQUEUE [13] is a web-based job submission to run jobs on a cluster. Prajapati and Shah [10] made an experimental study of remote

[1]IT Research Computing, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia; email: Matthijs.vanWaveren, Ahmed.SeifElNawasany, Nasr.Hassanein, David.Moon, Niall.OByrnes, Alain.Clo, Karthikeyan.Murugan, Antonio.Arena@kaust.edu.sa

job submission and execution on local resource managers through grid computing mechanisms. Bright Manager [3], software for deployment, monitoring and managing of HPC clusters, and HPCSpot [9], a solution for HPC on demand, both are related to remote job submission.

More recently, application software vendors have been implementing remote job submission into the application software products themselves. The novelty of the present work is that we configure remote job submission by using the vendor-supplied implementation in the applications. This allows us to configure the access to HPC resources in such a way, that the user can stay in the application GUI.

1. KAUST Computing Ecosystem

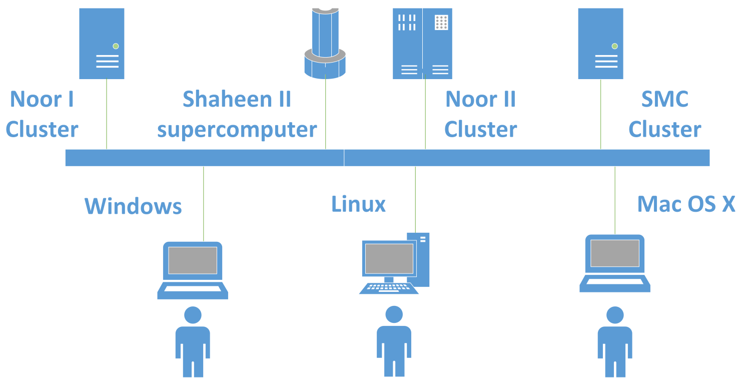

Figure 1. KAUST Computing Ecosystem

Researchers typically start out running their computational jobs on either a Windows, Linux or Mac OS X workstation.

If they need more performance, they have at their disposal the supercomputer Shaheen II and several HPC clusters, called Noor I, Noor II, and SMC, as shown in Fig. 1.

Shaheen II is a 36-cabinet Cray XC40 supercomputer with DataWarp technology for accelerating I/O. It has 192568 processors cores, 5.536 Pflops of sustained LINPACK performance, and 7.2 Pflops of theoretical peak performance. It is number 10 in the Top 500 list of June 2016 [14].

Noor II is a Linux cluster based on the quad-socket AMD Opteron 6376 processor. It provides a total capacity of 160 compute nodes or 10256 cores for a total of 105 TFlops of peak floating point performance. Each node can run 64 threads. It has four sockets of 8-core CPUs, with 2 threads per core, equipped with a high-performance, low latency Infiniband interconnect.

SMC[2] is a Linux cluster based on the Intel Xeon (Sandy Bridge) processor. It provides a capacity of 309 nodes, or 5940 cores for a total of 246 TFlops of peak floating point performance. 108 nodes have two sockets of 8 cores, 177 nodes have two sockets of 10 cores and 24 nodes have two sockets of 14 codes. They are equipped with a high-performance, low latency Infiniband interconnect.

Noor I is a Linux cluster based on the Intel Xeon (Nehalem) processor. Each node has two sockets of 4 cores equipped with a high-performance, low latency Infiniband interconnect.

The workload of all these systems is managed by the Simple Linux Utility for Resource Management (SLURM [2]), which is an open source cluster management and job scheduling system.

[2]SMC = Shared Mini Cluster

Researchers submit jobs to SLURM, and SLURM then allocates resources of the corresponding system to the jobs of the researchers.

2. User Barriers

There are barriers for researchers at KAUST to use the available HPC clusters and a supercomputer. In the course of migration from workstation to cluster or supercomputer, the researchers need to learn a whole new language: Linux commands, SLURM scheduler commands, and data transfer commands. This is especially the case if they come from a Windows world. It slows them down, and may even inhibit them from effectively utilizing the available HPC resources. Researchers need to login to the cluster, transfer input data to the cluster, submit a batch job, wait for it to finish, then transfer results back to the workstation. If they divide their computational work over different clusters, they need to do this sequence of tasks for each cluster separately, and combine the results from each cluster on their workstation, as shown in Fig. 2. This work is tedious, adds no value, and wastes valuable time of the researchers. With this work, we want to give the researchers a way to reduce the time spent on migrating from workstation to HPC resources.

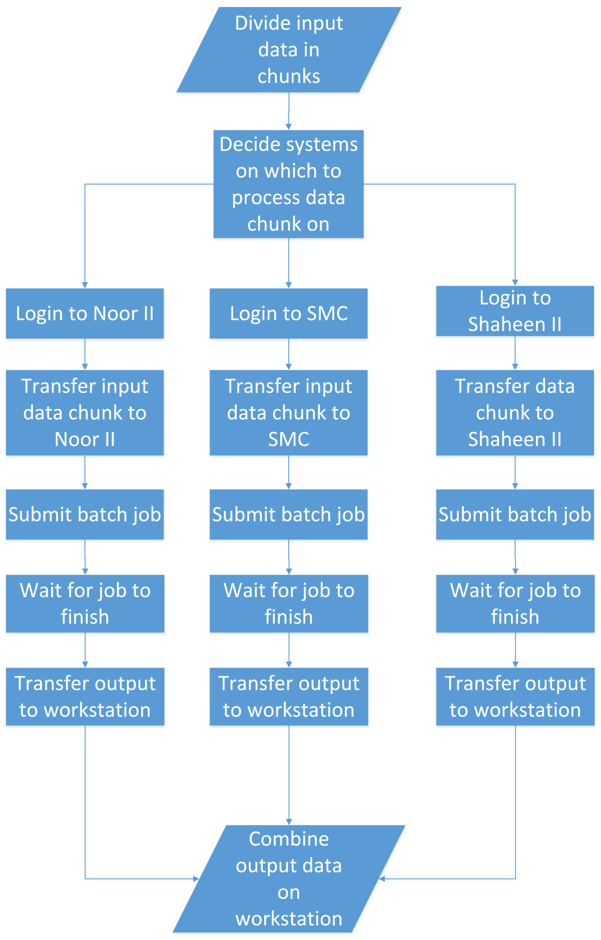
Divide input data in chunks
Decide systems on which to process data chunk on
Login to Noor II
Login to SMC
Login to Shaheen II
Transfer input data chunk to Noor II
Transfer input data chunk to SMC
Transfer data chunk to Shaheen II
Submit batch job
Submit batch job
Submit batch job
Wait for job to finish
Wait for job to finish
Wait for job to finish
Transfer output to workstation
Transfer output to workstation
Transfer output to workstation
Combine output data on workstation

Figure 2. Initial user workflow

3. Solution

We automated the sequences of tasks shown in Fig. 2 in order to reduce the time spent by researchers on executing these tasks. This automation is attained by implementing an interface between the research application running on the workstation (Linux, Windows or Mac) and the clusters. This interface automatically executes the opening of a session on the cluster, the transfer of input data, the job submission, and the retrieval of results. We call these interfaces HPC Add-ons, as they add an HPC capability to the research application running on the workstation. We have implemented these HPC Add-ons for the following research applications - MATLAB, ADF, and VASP. The innovation is that the user now submits his jobs to the cluster from within the application GUI on his workstation, and does not have to directly log into the cluster anymore. The resulting optimized workflow is shown in Fig. 3. The method of implementation of the HPC Add-ons is not standardized and differs for each application.

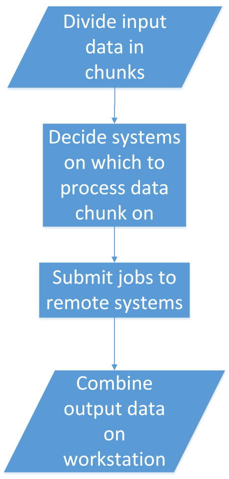
Divide input data in chunks
Decide systems on which to process data chunk on
Submit jobs to remote systems
Combine output data on workstation

Figure 3. Optimized user workflow

3.1. MATLAB

MATLAB [6] is a high-level language for scientific and engineering computing. The MATLAB HPC Add-on architecture is illustrated in Fig. 4. The HPC Add-on interface is implemented in MATLAB code, and is installed on the client side. This interface code takes care of opening a session to the remote resource, creating and sending a job submission script to the SLURM scheduler [2] on the remote resource, and monitoring the status of the job. The job is executed using the MATLAB Distributed Computing Server, which lets users run MATLAB jobs in parallel on HPC resources. The output directory is mirrored on the local workstation, so the researcher always has access to the latest results.

One of the KAUST researchers developed an algorithm for measuring single-molecule diffusion using MATLAB [12]. With a single run requiring about 200,000 Gaussian fittings, he soon found out that running on a single processor took too long to be practical. To shorten processing times he used the MATLAB Parallel Computing Toolbox to perform the computations on a workstation with multiple cores. Using four cores experiments took about three hours, and with 16 cores, just 45 to 50 minutes.

He often needs to run many simulations and experiments to obtain valid statistical results. To further accelerate the process he began running his jobs on 512 cores at a time on the HPC

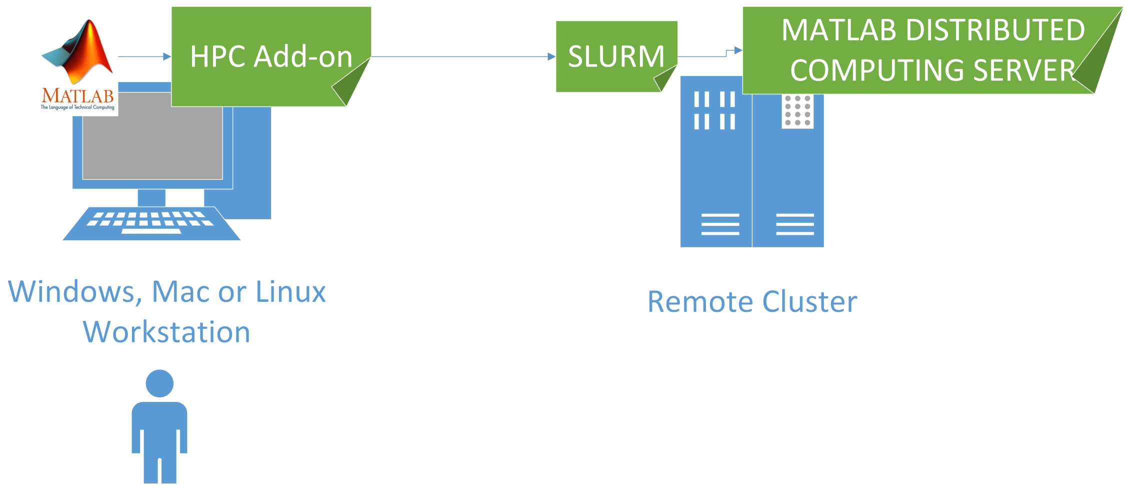
MATLAB
HPC Add-on
SLURM
MATLAB DISTRIBUTED COMPUTING SERVER
Windows, Mac or Linux Workstation
Remote Cluster

Figure 4. MATLAB HPC Add-on Architecture

clusters with the HPC Add-on and MATLAB Distributed Computing Server. Using this setup he can complete a set of experiments that took 24 hours on a multicore machine in just 15 minutes.

3.2. VASP

VASP [8] is a complex package for performing ab-initio quantum-mechanical Molecular Dynamics simulations using pseudopotentials or using the projector-augmented wave method and a plane wave basis set. The VASP HPC Add-on architecture is illustrated in Fig. 5. The interface between VASP and the remote resources is implemented by a software product called MedeA [11].

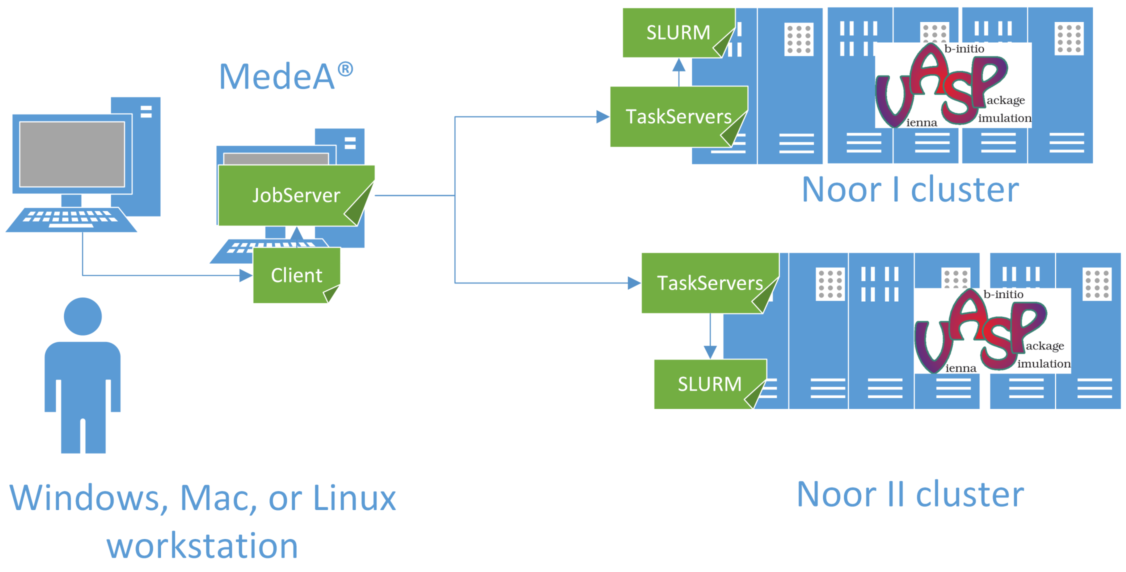

Figure 5. VASP HPC Add-on Architecture

MedeA has a tiered architecture, consisting of the MedeA graphical user interface (main tier), the JobServer (middle tier) and the TaskServer (end tier). The JobServer is the central hub for computational job control, job preprocessing, post processing. The TaskServer takes care of the communication with the SLURM scheduler on the remote cluster. There are several clusters at KAUST, and we only show the Noor I and the Noor II clusters in the figure.

3.3. ADF

ADF [15] is a molecular density-functional theory code used in many areas of chemistry and materials science. ADF is particularly strong in molecular properties and inorganic chemistry. The ADF HPC Add-on architecture is illustrated in Fig. 6.

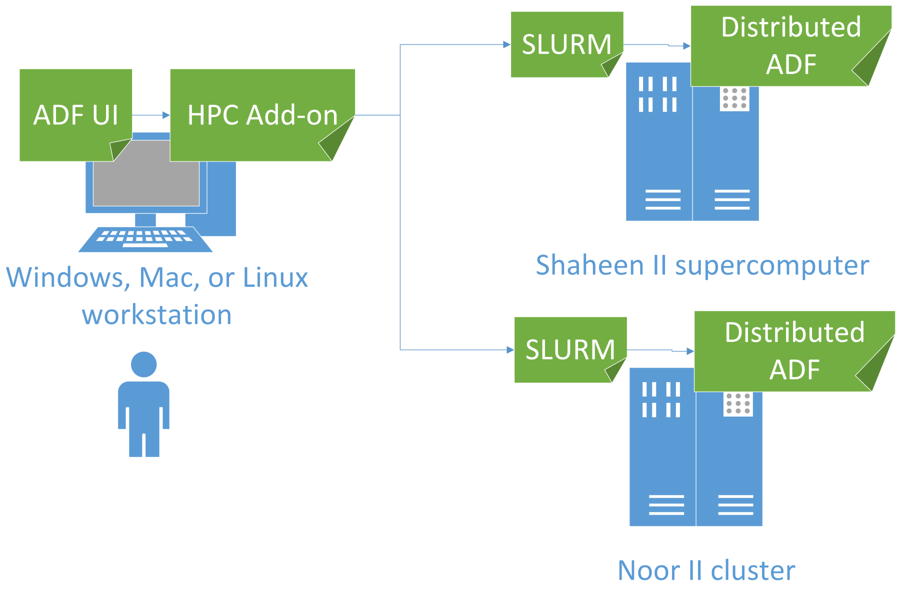

Figure 6. ADF HPC Add-on Architecture

The GUI is written in Tcl/Tk, and it reads a configuration file with details of the remote system. The GUI then opens a session via ssh to the remote system, and submits a job to the SLURM scheduler on the remote system. It can check the status of jobs on the remote system, and can kill the job. Transfers of files to and from the remote system is handled by the GUI automatically (via ssh).

Conclusions

We show that we can simplify the workflow of a user of HPC resources considerably by automating the workflow sequence: logging into the HPC system, job submission and transfer of input and output data. This has been implemented in the user interface of the applications directly, allowing the user to stay in this user interface. The applications we have discussed are MATLAB, VASP and ADF.

Benefits for the researchers include reduced time-to-solution, easier access to the HPC systems, and less time spent on mastering unfamiliar Linux commands. One of the MATLAB researchers used to need 24h wall clock time to execute runs to measure single-molecule diffusion on his workstation, but now has his results in 15 minutes by using the HPC Add-on.

Our future work includes making the code of our HPC Add-ons available as open source, and implementing HPC Add-ons for other applications that have built-in support for remote job submission, for example MaterialsStudio of Dassault Systmes. We plan also to implement an

HPC Gateway for automating the workflow sequence for applications that do not have built-in support for remote job submission.

We acknowledge support given by the KAUST Supercomputer Laboratory (KSL), The MathWorks Inc, Scientific Computing and Modelling NV and Materials Design Inc. We acknowledge the thorough review of Dr Samar Aseeri of KSL.

References

1. Jim Almond and Dave Snelling. UNICORE: uniform access to supercomputing as an element of electronic commerce. *Future Generation Computer Systems*, 15(5):539–548, 1999. http://0-www.sciencedirect.com.libsys.kaust.edu.sa/science/article/pii/S0167739X99000072.
2. D. Auble, B. Christiansen, M. Jette, A. Sanchez, and T. Wickberg. Slurm workload manager. http://slurm.schedmd.com/slurm.html.
3. Martijn de Vries. Bright Cluster Manager for HPC: advanced cluster management made easy. http://www.brightcomputing.com/product-offerings/bright-cluster-manager-for-hpc.
4. Ian Foster. Globus Toolkit Version 4: Software for Service-Oriented Systems. *Journal of Computer Science and Technology*, 21(4):513–520, 2006.
5. Roger Henri, Pierre Lagier, and Didier Plaindoux. FSE grid middleware: Collaborative grid environment for distributed computing. *Fujitsu Scientific and Technical Journal*, 40(2):269–281, 2004.
6. Yuliya V. Karpievitch and Jonas S. Almeida. mGrid: A load-balanced distributed computing environment for the remote execution of the user-defined MATLAB code. *BMC Bioinformatics*, 7(1):139–139, 2006.
7. Fumikazu Konishi and Akihiko Konagaya. The Architectural Design of High-Throughput BLAST Services on OBIGrid. In Akihiko Konagaya and Kenji Satou, editors, *Grid Computing in Life Science: First International Workshop on Life Science Grid, LS-GRID 2004, Kanazawa, Japan, May 31-June 1, 2004, Revised Selected and Invited Papers*, pages 32–42, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. http://www.springer.com/us/book/9783540252085.
8. G. Kresse and J. Furthmüller. Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set. *PHYSICAL REVIEW B*, 54(16):11169–11186, 1996. DOI: 10.1103/PhysRevB.54.11169.
9. Alexandre Morel. HPCSpot for HPC on demand. http://www.oxalya.com/?p=2173.
10. Harshadkumar B. Prajapati and Vipul A. Shah. Experimental study of remote job submission and execution on LRM through grid computing mechanisms. In IEEE, editor, *Fourth International Conference on Advanced Computing & Communication Technologies*, 2014.
11. X. Rozanska, P. Ungerer, B. Leblanc, P. Saxe, and E. Wimmer. Automatic and systematic atomistic simulations in the MedeA (r) software en-

vironment: Application to EU-REACH. *OIL & GAS SCIENCE AND TECHNOLOGY-REVUE IFP ENERGIES NOUVELLES*, 70(3):405–417, 2015;2014;. http://ogst.ifpenergiesnouvelles.fr/articles/ogst/abs/2015/03/ogst140090/ogst140090.html. DOI: 10.1103/PhysRevB.54.11169

12. Maged F. Serag. Accelerating Development of a New Single-Molecule Localization and Tracking Technique. *MathWorks Newsletter*, 2016. http://www.mathworks.com/company/newsletters/articles/accelerating-development-of-a-new-single-molecule-localization-and-tracking-technique.html.

13. Wade Sisson. eQUEUE Job Submission Tool. http://y2u.be/0IcTq4BMYwQ.

14. E. Strohmaier, J. Dongarra, H. Simon, and M. Meuer. Top 10 sites for June 2016. https://www.top500.org/lists/2016/06/.

15. G. te Velde, F. M. Bickelhaupt, E. J. Baerends, C. Fonseca Guerra, S. J. A van Gisbergen, J. G. Snijders, and T. Ziegler. Chemistry with ADF. *Journal of Computational Chemistry*, 22(9):931–967, 2001.