jamk.fi
IT-instituutti

JAMK High Performance Computing (HPC)

Juha Jokinen, Mika Rantonen, Jesse Raitapuro, Antti Mäkelä, Joni Korpihalkola, Janne Alatalo, Eppu Heilimo

KESKI-SUOMEN LIITTO
Regional Council of Central Finland
PIRKANMAA
Vipuvoimaa EU:lta 2014–2020
Euroopan unioni
Euroopan aluekehitysrahasto

TABLE OF CONTENTS

1 INTRODUCTION

With the fast development of technology, the terms Digitalization and Big Data have gained strong publicity. Due to digitalization, data is produced at an accelerated pace almost everywhere where people's actions and various, mainly electronic systems meet. Often data is compared to crude oil, which acts as raw material and through the refining process produces new products for industry. In order to get benefits from data, Data-analytics (DA) and Artificial Intelligence (AI) have appeared in publications titles in recent years and have become very hot topics.

High-performance computing (HPC) is defined as the ability to process a massive amount of data and perform complex calculations at high speeds. Technologies such as DA and AI need the big amount of data and computing resources. Especially the deep learning (DL) and neural network development need lots of computational power which cannot be achieved using traditional CPU (Central Processing Unit) based computing. DL is a field with intense computational requirements, and the choice of GPU (Graphics Processing Unit) will fundamentally determine the deep learning experience.

Cloud computing is the on-demand availability of computer system resources such as data storage, computing power and memory. The term is generally used to describe data centres available to anyone over the Internet. Cloud providers offer computing capability; however, it is very difficult to estimate the costs due to the fact that the amount of data can be substantial depending on the application. The most commonly used environments are cloud-based, where scalable computing time, storage, and memory can be purchased. While cloud computing capacity, storage and memory volumes are compared and pricing is very clear, it is difficult to determine cost in advance for DA and AI projects. For example, when training a neural network, the use of computing capacity may come as a surprise. Another issue that has emerged in business collaboration is data security, security and legal issues for cloud-based data analytics and artificial intelligence environments. The most common issue regarding the disclosure of corporate data. Companies want to be sure that data is stored properly, and they may not want to place it in the cloud. An alternative to these is to build one's own HPC environment. The costs of one's own computing environment setup may be high; however, after that the costs consist mainly of maintenance and power consumption.

The purpose of this document is to give the reader a technical overview of the HPC system used by JAMK University of Applied Sciences (JAMK). The document is by no means a user guide nor a system administrator's guide as this information is for internal use only. Instead, this document gives the reader a general understanding of the capabilities of the system. Some example use cases are also included.

2 JAMK UNIVERSITY OF APPLIED SCIENCES

JAMK is an international higher education institute that hosts more than 8,500 students from over 70 countries working and studying in eight different fields of study. The Institute of Information Technology (later JAMK IT) is part of the School of Technology at JAMK that educates people and provides services for companies and public sector. JAMK

IT focuses on training engineers and providing R&D services in information and communication technologies, cyber security, and software development.

In spring 2017, JAMK IT started to develop its competence in DA and AI. Accordingly, R&D in DA and AI in project form has started to grow as the expertise increases. Project preparation and application have been made possible by in-depth knowledge of DA and AI, both in theory and in practice, which enables us to apply funding from various funders. R&D at JAMK can be called *applied R&D*, which differs significantly from academic research in the university world. Applied R&D uses fully open source products and tools, which means applying existing ready methods to business problems. The more academic and basic research in university research goes much deeper in algorithms or neural network architecture as is the case in the applied R&D. The emphasis has been placed on the practical application of open source AI and DA products, either based on available open data or on a real business problem. The application of AI has been facilitated by the introduction of AI libraries by various commercial operators, perhaps best known as Google's Tensorflow. In DA, the Python programming language is selected due to its well-known capability in conducting scientific calculation.

3 PROJECTS IN JAMK

JAMK IT has an active research group in DA and AI with large external and internal project portfolios. One of the investors is European Regional Development Fund (ERDF) that supports two projects: *New Business Innovations from Data analytics* and *Information Secure R&D-environment for Data Analytics*.

The goal of the projects is to enable and improve data analytics know-how as well as develop and pilot concrete solutions of dark data using data analytics methods, which help companies to create new business opportunities or increase the growth in their business and add to the amount of vacant jobs. With the development of data analytics environments and tools, utilizing data and combining various data-/information sources has become a new opportunity for companies to develop business.

The focus of the Information Secure R&D-environment for Data Analytics project is to implement the needed investments in order to plan, build and deploy an integrated, information secure data analytics development, i.e. HPC environment. The data produced by companies may include very critical information as far as data protection is concerned, and the storing, filtering and data security need to be taken into account. HPC environments based on cloud services face challenges in data security, data protection and juridical questions. An alternative to cloud services is to build a company's own HPC environment. Besides, the EU General Data Protection Regulation (GDPR) defines sanctioned obligations to companies concerning processing and securing data. Therefore, the latter project focuses on building its own information secure data analytics/HPC development environment.

4 JAMK HPC ENVIRONMENT

4.1 Requirements

The requirements of environment come from the specifications and needs of data science/analysis and machine learning projects. The above mentioned projects differ radically from the traditional Business Intelligence projects. Many data analysis/science or machine leaning projects are more exploratory in nature. For this reason, it is critical to understand the data and to ensure that the participants are thorough and rigorous in their approach. The Data Analytics Lifecycle is designed specifically for Big Data problems and data science projects. Due to the variety, volume and velocity of the data, the environment must be planned to be capable to analyze the data in the rest as well as the real time streamed data. The security aspect of HPC environment is also discussed in the next sections.

- Data analytics using Python (data in the rest)
- Data analytics using Apache Spark (real time streamed data)
- Machine learning using Python
- Neural network computing using Tensorflow, Keras etc.
- Secure environment, i.e. authentication and file storage accesses
- Workload management

4.2 HPC Hardware

JAMK IT department uses Cisco Unified Computing System (UCS) based servers for scalability and centralized management of compute, storage and network resources. The HPC server itself is a single Cisco UCS C480 M5 rack-mounted server located on-premises. The server utilizes performance class technology as follows:

- Four 16-core Intel Xeon Gold 6130 processors, 64 cores total
- 768GB DDR4 memory, error-correcting
- Four nVidia Tesla V100 32GB graphics cards in computing mode
- Two 240GB SSD drives for the operating system, RAID1 mirrored redundancy
- Two 1.6TB HGST SN200 NVMe drives for fast local data storage
- 2x10Gb redundant uplinks for networking. Possible to upgrade to 2x40Gb if necessary
- Four 1600W power supplies

The server also uses storage from the IT department's all-SSD SAN array for long-term storage.

Other notable hardware-related aspects:

- All BIOS configuration and system settings are centrally managed from the Cisco UCS manager. This way the setup is easily replicable to another server of the same kind.

- Hyper-threading is turned off as it usually does not help computing loads and can even be detrimental to them. This is also in line with Cisco HPC best practices.
- Cisco UCS C480 M5 does not support NVLink interconnect for Tesla V100 cards. Newer server chassis such as C480 ML M5 should be used for new installations.

5 JAMK HPC ENVIRONMENT SOFTWARE

5.1 CentOS Linux

JAMK IT department uses mainly CentOS Linux based on Red Hat Enterprise Linux (RHEL). RHEL/CentOS is widely used for mission-critical applications and is tested to be a solid base for any production environment. The ideology behind RHEL is to use certain versions of applications in the long term providing only security updates and minor fixes, which improves system stability and manageability. This can be a problem in development environments; however, application versions in package repositories are much older than current releases. This was deemed as not an issue when evaluating different distributions as the base of the server, as almost all of the software used will be compiled from sources.

The current installation is based on CentOS 7. Plans are to migrate to CentOS 8 after its release to provide longer support and more modern base packages.

5.2 Performance options

No larger performance tuning has been done but several basic steps have been taken:

- *tuned* and *numad* services are turned on and *tuned*-service is set to use *throughput-performance* profile as per Red Hat's recommendations
- *vm.swappiness* is lowered from default value to 10 in order to minimize premature swapping to disk
- *fstrim* service is enabled to handle TRIM on SSD/NVMe devices

The system remains monitored in several ways to further analyze and mitigate performance bottlenecks.

Performance measurements will be done in the future. Performance results and optimization will also be reported.

5.3 CUDA

CUDA is a proprietary language developed by NVIDIA. CUDA is used by many deep learning frameworks such as TensorFlow and PyTorch to speed up the training of neural networks. The key component of CUDA is a C compiler that grants access to parallel computing in the GPU. The GPU utilizes arithmetic logic units to compute kernel operations such as matrix multiplication in parallel.

TensorFlow and other deep learning frameworks have built libraries utilizing the C compiler to perform matrix operations on the GPU. Usage of CUDA libraries in TensorFlow

requires installation of NVIDIA drivers, CUDA development and runtime libraries. Additionally, NVIDIA's "cuDNN" library is used to provide optimized functions used in building deep neural networks such as activation functions or convolution layers.

5.4 TensorFlow

TensorFlow is a permissively licensed open-source computation framework developed and maintained by Google. While TensorFlow has multiple applications, its primary use case is ML and neural networks. Due to the popularity of TensorFlow, a vast quantity of DL programs is available online ready for use. TensorFlow scales to different compute devices, ranging from single CPUs and GPUs to full-fledged computation clusters. Production ready applications can be served with a multitude of methods: over a network with *TensorFlow Serving*, directly on the client's web browsers with *TensorFlow.js* or by simply embedding the neural network model to the application.

Users of TensorFlow can program computational graphs utilizing a large library of operations included in the framework. The users can also build their own custom computations from the fundamental operations. TensorFlow includes an automatic differentiation system for computing gradients over a graph of computations, which is useful when training neural networks with back-propagating optimization methods.

5.5 Apache Spark

Apache Spark is a computation framework optimized for cluster computing. In addition to one-off computations with data at rest, Spark supports real time data stream processing with Spark Streaming. Apache Spark is available as open-source with a permissive Apache License 2.0.

Spark achieves a high performance by keeping the data in memory instead of storing it in mass storage devices. Spark also has high fault tolerance due to the deterministic nature of Spark computations: if a computation node goes down, the results of the lost computation can be replicated by simply retracing the steps of the computation.

5.6 Python

Python is an interpreted programming language, and in the past it was mostly used for fast prototyping in software development because of its simple syntax. Nowadays Python has grown into one of the most used programming languages in the field of DA and ML.

Because Python is an interpreted language, it has some performance disadvantages over compiled languages such as C or C++. The key for the performance problem is solved by using optimized (and parallel execute unlocking) libraries in Python such as Numpy, Multiprocessing or Tensorflow, which are written in lower level programming languages. GPUs can also be utilized (normally CPUs execute code), when using correct libraries for machine learning such as Tensorflow. GPUs allow specific task(s) to be processed rapidly compared to CPUs. Optimized libraries and their methods/attributes can be glued together with Python, and for that reason, Python can compete with more efficient programming languages.

5.7 User environment

Authentication is based on IT LDAP directory for centralized account management. Users will receive a home folder, temporary work folder and a long-term storage folder appointed to them with correct permissions and paths during login. File system quotas are in place to limit storage usage.

5.8 Libraries and modules

As HPC workloads utilize many differing libraries and modules, it is impossible to install these as system-wide defaults for all users. Versioning becomes a problem as does management since compiling different libraries and modules takes time and effort. Using module management system solves these problems. JAMK IT HPC server uses two applications to manage modules, **Lmod** and **EasyBuild**.

Lmod is used to provide the users the ability to select which modules they wish to use when running/compiling their own workloads. Modules can be loaded to provide different versions of compilers, MPI libraries, Python interpreters and such. Modular design also makes it easy to bundle different modules as saved workspaces and automatically load dependent modules when needed. An example is shown below in Figure 1, where the user loads a version of Python 3 interpreter compiled with the latest CUDA10 support.

```
[jojuh@laskutikku ~]$ module load Python/3.6.6-fosscuda-2018b10
[jojuh@laskutikku ~]$ module list

Currently Loaded Modules:
  1) GCCcore/7.3.0                10) libpciaccess/0.14-GCCcore-7.3.0              19) ncurses/6.1-GCCcore-7.3.0
  2) binutils/2.30-GCCcore-7.3.0  11) hwloc/1.11.10-GCCcore-7.3.0                  20) libreadline/7.0-GCCcore-7.3.0
  3) GCC/7.3.0-2.30               12) OpenMPI/3.1.1-gcccuda-2018b10                21) Tcl/8.6.8-GCCcore-7.3.0
  4) CUDA/10.0.130-GCC-7.3.0-2.30 13) OpenBLAS/0.3.1-GCC-7.3.0-2.30                22) SQLite/3.24.0-GCCcore-7.3.0
  5) gcccuda/2018b10              14) gompic/2018b10                               23) GMP/6.1.2-GCCcore-7.3.0
  6) zlib/1.2.11-GCCcore-7.3.0    15) FFTW/3.3.8-gompic-2018b10                    24) libffi/3.2.1-GCCcore-7.3.0
  7) numactl/2.0.11-GCCcore-7.3.0 16) ScaLAPACK/2.0.2-gompic-2018b10-OpenBLAS-0.3.1 25) Python/3.6.6-fosscuda-2018b10
  8) XZ/5.2.4-GCCcore-7.3.0       17) fosscuda/2018b10
  9) libxml2/2.9.8-GCCcore-7.3.0  18) bzip2/1.0.6-GCCcore-7.3.0


[jojuh@laskutikku ~]$ python -VV
Python 3.6.6 (default, Feb  6 2019, 15:34:51)
[GCC 7.3.0]
[jojuh@laskutikku ~]$
```

Figure 1. Loading a CUDA-enabled Python environment and dependencies

All the module paths (binary, library and include locations) are handled automatically by the Lmod.

Compiling all these modules by hand with necessary options and toolchains to make modular loading work is, while not impossible, very tasking. For automating module compilation **EasyBuild** is used. EasyBuild is based on its own configurable ebfiles, which inform the EasyBuild process on how to build the software in a modular way. EasyBuild integrates with Lmod, automatically providing the correct modulefiles after the software is compiled and handling compile-time dependencies. Figure 2 shows an example of building a specific version of Apache Spark with EasyBuild.

```
[jojuh@laskutikku ~]$ module load EasyBuild
[jojuh@laskutikku ~]$ eb -S Spark
CFGS1=/opt/ext/easybuild/software/EasyBuild/3.8.1/lib/python2.7/site-packages/easybuild_easyconfigs-3.8.1-py2.7.egg/easybuild/easyco
nfigs/s/Spark
 * $CFGS1/Spark-1.3.0.eb
 * $CFGS1/Spark-1.4.1.eb
 * $CFGS1/Spark-1.5.0.eb
 * $CFGS1/Spark-1.6.0.eb
 * $CFGS1/Spark-1.6.1.eb
 * $CFGS1/Spark-2.0.0.eb
 * $CFGS1/Spark-2.0.2.eb
 * $CFGS1/Spark-2.2.0-Hadoop-2.6-Java-1.8.0_144.eb
 * $CFGS1/Spark-2.2.0-Hadoop-2.6-Java-1.8.0_152.eb
 * $CFGS1/Spark-2.2.0-intel-2017b-Hadoop-2.6-Java-1.8.0_152-Python-3.6.3.eb
 * $CFGS1/Spark-2.3.0-Hadoop-2.7-Java-1.8.0_162.eb
 * $CFGS1/Spark-2.4.0-Hadoop-2.7-Java-1.8.eb
[jojuh@laskutikku ~]$ eb Spark-2.3.0-Hadoop-2.7-Java-1.8.0_162.eb -Dr
== temporary log file in case of crash /tmp/jojuh/eb-jp11I1/easybuild-M3YEVm.log
Dry run: printing build status of easyconfigs and dependencies
CFGS=/opt/ext/easybuild/software/EasyBuild/3.8.1/lib/python2.7/site-packages/easybuild_easyconfigs-3.8.1-py2.7.egg/easybuild/easycon
figs
 * [x] $CFGS/j/Java/Java-1.8.0_162.eb (module: Java/1.8.0_162)
 * [ ] $CFGS/s/Spark/Spark-2.3.0-Hadoop-2.7-Java-1.8.0_162.eb (module: Spark/2.3.0-Hadoop-2.7-Java-1.8.0_162)
== Temporary log file(s) /tmp/jojuh/eb-jp11I1/easybuild-M3YEVm.log* have been removed.
== Temporary directory /tmp/jojuh/eb-jp11I1 has been removed.
[jojuh@laskutikku ~]$
```

Figure 2. Dry-run of building specific version of Apache Spark with EasyBuild

5.9 Workload management

In order to provide ample resources for all users without overcommitting the system, a workload management system is required to queue computing workloads between users. JAMK IT HPC Server uses **Slurm** for scheduling workloads. The system differs from many HPC platforms in such as it has no separate login node in place as there are currently only a handful of users. Users are able to use the HPC server freely for computing as they wish; however, they are instructed to submit workloads to Slurm queues using commands such as *srun* and *sbatch*. Slurm will then queue the workload for compute time. No mechanism prevents the users from going around these limitations except for peer feedback but plans are to implement login node(s) next time when upgrading the operating system.

5.10 Monitoring

Monitoring HPC resource usage is an important part of the system. It provides information about system utilization for the system administrators, and it can also give feedback for the users about their program performance. Proper visualization of recourse usage can help to find bottlenecks in user's code and helps to tune the program so it can utilize all the available recourses. The monitoring system is implemented using open-source tools Telegraf, influxDB and Grafana. InfluxDB is the time series database where the metrics are stored, Telegraf is a daemon program that sends the metrics to the influxDB database, and Grafana is a simple web-based visualization software that can be used to visualize data from different data sources. Telegraf default installation includes many plugins for collecting different metrics from the server. In our setup, the daemon is configured to send CPU, RAM and GPU metrics to the database. The data is then visualized in custom Grafana dashboards. Grafana supports LDAP user authentication, so users can log in to Grafana with the same account details that are used for SSH connections. Figure 3 shows Grafana dashboard visualizing server metrics.

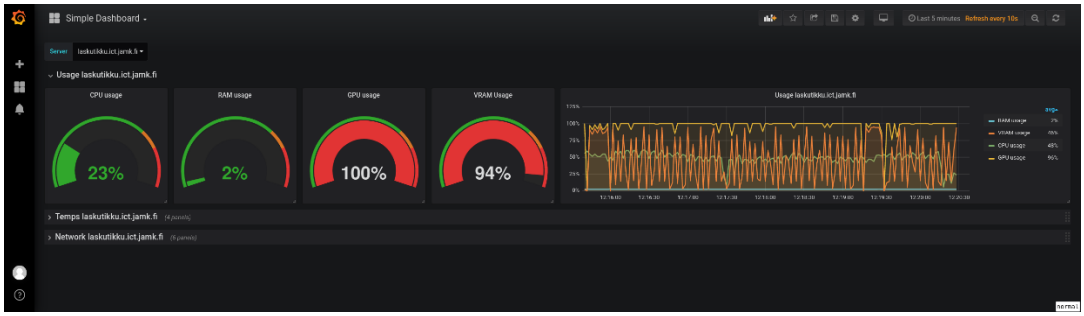

Figure 3. Screenshot of Grafana web application visualizing metrics from the server

6 CONCLUSION

JAMK's computing cluster has been in the daily use of 10 experts and researchers since the beginning of 2019. The HPC environment has operated without performance problems and downtime issues and it will provide more accurate performance measurements in the near future. Performance measurements will be used to map potential bottlenecks, and the performance will be optimized to be as good as possible.