MAX PLANCK
COMPUTING & DATA FACILITY
MPCDF

Bits & Bytes

No.207, August 2021

Computer Bulletin of the Max Planck Computing and Data Facility (MPCDF)*
https://docs.mpcdf.mpg.de/bnb

High-performance Computing

HPC system *Raven* fully operational

Figure 1: HPC system Raven

The deployment of the new HPC system of the Max Planck Society, *Raven*, has been completed by Lenovo and MPCDF in June 2021. The machine now comprises 1592 CPU compute nodes with the new Intel Xeon IceLake-SP processor (Platinum 8360Y with 72 cores per node). In addition, *Raven* provides 192 GPU-accelerated compute nodes, each with 4 Nvidia A100 GPUs (4 x 40 GB HBM2 memory per node and NVLink 3). *Raven* entered the most recent June 2021 edition of the Top500 list[1] of fastest supercomputers with a measured HPL (high-performance linpack) benchmark performance of 8620 TFlop/s (rank 47) for the GPU-accelerated part, and 5416 TFlop/s (rank 79) for the CPU-only part. On the HPL benchmark, the GPU-accelerated part of *Raven* with 768 Nvidia A100 GPUs achieves a power efficiency of 22.9 GFlop/s/W which puts it on rank 12 of the Green500 ranking[2] of the most energy-efficient supercomputers. Together with Cobra (rank 77, 5613 TFlop/s) and its (unranked) GPU-partition, researchers of the Max Planck Society have an aggregate HPL performance of ca. 22 PFlop/s at their disposal, which is roughly the equivalent of a Top-15 supercomputer.

More details about *Raven* can be found below, on the MPCDF webpage[3] and in a previous Bits & Bytes article[4]. *Raven* users are referred to the technical documentation and to specific trainings offered by the MPCDF (see below).

Hermann Lederer, Markus Rampp

GPU Computing on *Raven*

Overview The GPU-accelerated part of *Raven* comprises 192 nodes, each with 4 Nvidia A100 GPUs ("Ampere" architecture) which are mutually interlinked with the high-speed interconnect NVLink 3 (100 GB/s per direction for every pair out of the 4 GPUs in a node) and are connected to the host CPUs (which are of the same type as the CPU-only part of *Raven*, Intel Xeon Platinum 8360Y) via the PCIe-4 bus at a speed of 32 GB/s per direction for every GPU. The GPU-accelerated nodes in *Raven* are connected to the Infiniband network at 200 Gbit/s which is twice the bandwidth available in the CPU-only part. A subset of 32 nodes is connected with 400 Gbit/s (at the expense of a reduced internal PCIe bandwidth). These nodes can be requested by a special slurm option (`--constraint="gpu-bw"`). The MPCDF documentation provides example scripts for the slurm batch system[5]. Users may request 1, 2, or 4 GPUs and a corresponding fraction of cores of the host CPUs (18, 36, or 72 cores). If multiple nodes are requested, all nodes including all cores and GPUs are allocated exclusively for

*Editors: Dr. Renate Dohmen & Dr. Markus Rampp, MPCDF
[1] https://www.top500.org/lists/top500/list/2021/06/
[2] https://www.top500.org/lists/green500/list/2021/06/
[3] https://www.mpcdf.mpg.de/services/supercomputing/raven
[4] https://docs.mpcdf.mpg.de/bnb/206.html#hpc-system-raven-deployment-of-the-final-system
[5] https://docs.mpcdf.mpg.de/doc/computing/raven-user-guide.html#batch-jobs-using-gpus

the job. Computing time on GPU-accelerated nodes is accounted using a weighting factor of 4 relative to CPU-only jobs. Users are advised to check the performance reports of their jobs in order to monitor adequate utilization of the resources.

In addition to the standard software stack, the MPCDF module system provides CUDA and related GPU libraries like cuFFT, cuBLAS, . . . , the Nvidia HPC SDK (nvhpcsdk, formerly known as PGI) with OpenACC and OpenMP-5 capable C, C++ and Fortran compilers, as well as further GPU-enabled HPC libraries and applications like MAGMA, ELPA, GROMACS, NAMD, OCTOPUS, and also GPU-optimized machine-learning software like TensorFlow. In addition to the standard Intel-MPI (impi) library an optimized installation of OpenMPI is provided for enabling faster GPU-to-GPU transfers (see article below). Further software of common interest can be installed on request. Dedicated training material and upcoming events can be found on the MPCDF webpage under Training & Education (see also below).

CUDA-aware MPI on *Raven* GPU nodes An easy way to leverage the fast NVLink interconnect in an HPC code using MPI (Message-Passing Interface) communication is to employ CUDA-aware MPI which allows to transfer data between the GPUs of a node via NVLink rather than transferring via PCIe and through the host CPUs. CUDA-aware MPI is a feature of certain MPI libraries such as OpenMPI and MVAPICH and it allows to transfer data directly between GPUs. This is achieved by handling pointers to GPU buffers transparently in MPI calls: if the MPI library detects that a pointer to a buffer in an MPI call points to GPU memory, it will initiate a transfer directly from or to that GPU. This also makes the code easier: the user does not have to transfer the data to the CPU before calling MPI routines. Currently, peer-to-peer calls (i.e., send and receive) have the best support and will lead to direct GPU-GPU transfers. If the communicating ranks are placed on different nodes, using CUDA-aware MPI has the advantage that no additional buffer on the CPU memory is needed. Instead, the transfer can be initiated from the GPU memory over the network. Collectives are also supported, but for most of them a transfer to the CPU will happen internally. In the future, the MPI libraries will probably be able to leverage NCCL (Nvidia collective communication library), which provides also collectives that use GPU buffers and also launch kernels (e.g., for reductions). On our *Raven* system, we support a CUDA-aware version of OpenMPI that can be loaded using

```
module load gcc/10 cuda/11.2 openmpi_gpu/4
```

It also includes support for the low-level drivers “gdrcopy” and “nvpeermem” via UCX.

Moreover, we recommend to profile your code using “Nsight systems” (`module load nsight_systems`) which will yield useful information on kernel launches and data transfers, including transfer size and speed, and whether it is a CPU-GPU or GPU-GPU transfer. A profiling run can be started using

```
nsys profile -t cuda,nvtx,mpi srun ./app
```

in a batch script. This will create a report file that can later be opened using the GUI (`nsight-sys`) either remotely or locally. The GUI will show a timeline of the execution including all kernel launches and data transfers. This can be used to check which transfers already employ the NVLink interconnect.

Sebastian Ohlmann, Markus Rampp

HPC system *Cobra* - Module system to be aligned with *Raven*

Currently on Cobra, a default set of Intel environment modules is loaded automatically during login and during the startup of a batch job. Moreover, default versions are currently configured for the environment modules of the Intel Compiler and Intel MPI. Please note that with an upcoming Cobra maintenance in September this will change. After that maintenance no defaults will be defined for the Intel compiler and MPI modules and no modules will be loaded automatically at login. This change aligns the configuration with *Raven* where users already have to specify module versions, and no default modules are loaded. The exact date will be announced in due time, but users are encouraged to take notice and adapt their scripts already now.

What kind of adaptations of user scripts are necessary? Please load a specific set of environment modules with explicit versions consistently when compiling and running your codes, e.g. use

```
module purge
module load intel/19.1.3 impi/2019.9 mkl/2020.4
```

in your job scripts as well as in interactive shell sessions. Note that you must specify a specific version for the ‘intel’ and the ‘impi’ modules (no defaults), otherwise the command will fail. Please note that for your convenience, pre-compiled applications provided as modules like vasp or gromacs will continue to load the necessary intel and impi modules automatically, i.e. no changes of the batch scripts are required for these applications.

The Intel versions that are currently recommended are specified in the documentation, but other versions provided by the MPCDF work as well. User codes compiled prior to the maintenance will continue to work provided that the user loads the correct environment modules in the job script.

Klaus Reuter, Sebastian Ohlmann

Decommissioning of *Draco*

On July 15th, 2021 after more than 5 years of operation, the main part of the HPC extension system *Draco* has been decommissioned. The dedicated nodes which were added to *Draco* by individual institutes (NOMAD laboratory at the FHI, Max Planck Institute for the Structure and Dynamics of Matter, Max Planck Institute for Animal Behaviour) at a later time continue to operate exclusively for their owners. For the other users, resources on the HPC systems *Cobra* and *Raven* are offered as a replacement. Users of the *Draco* GPU nodes will find comparable resources in the RTX5000 GPUs of *Cobra* (partitions gpu_rtx5000 and gpu1_rtx5000), and moreover GPUs of type V100 on *Cobra* and of type A100 on *Raven*.

Access to data in /draco/u, /draco/ptmp and /draco/projects remains possible via the login nodes at least until the end of 2021. Note, however, that no external file systems are mounted on *Draco* any more. Data transfer from *Draco* to *Cobra* or *Raven* has to be initiated from *Cobra* or *Raven*, respectively.

Renate Dohmen, Mykola Petrov

HPC Cloud

In collaboration with the main funders, the Fritz Haber Institute, the Max-Planck-Institut für Eisenforschung, and the Max Planck Institute for Human Cognitive and Brain Sciences, MPCDF has designed and built a hybrid cloud to complement the HPC system *Raven*. The HPC Cloud system, based on OpenStack, Ceph, and IBM Spectrum Scale, comprises the following hardware:

Figure 2: HPC Cloud

- 60 Intel IceLake-based (Xeon Platinum 8360Y @ 2.4 GHz) compute nodes totaling 4320 cores and 44 TB of RAM. Included are eight nodes each with 2 TB of RAM to support TB-scale "huge" virtual machines and four nodes each with 3 Nvidia A30 GPUs.
- 460 TB of block and object storage including 80 TB of network-attached SSDs and 80 TB of host-attached SSDs.
- 3.5 PB of file storage accessible from both *Raven* and cloud servers.
- Dual 25 Gb/s Ethernet uplinks from all compute nodes to a fully-redundant 100 Gb/s backbone.

Conceptually, the HPC Cloud enables scientists to combine batch and cloud-based computing within the same pipeline, taking advantage of both massive HPC cluster resources and highly flexible software environments. This enables projects to realize novel hybrid solutions while also benefiting from simple scaling within the cloud and the ability to rapidly prototype new solutions. Practically, the system offers standard cloud computing "building blocks", including virtual machines based on common Linux operating systems, software-defined networks, routers, firewalls, and load balancers, as well as integrated block and S3-compatible object storage services. All resources can be provisioned and managed via a web browser or industry-standard RESTful APIs.

Each project will receive a quota for each resource type, within which institute-based admins have the freedom to allocate their resources as necessary to realize their projects. The MPCDF Cloud Team is available to provide consulting and advice during both the project planning and realization phases.

Moreover, to ensure the seamless flow of data between the HPC Cloud and *Raven*, an IBM Spectrum Scale filesystem has been deployed. Each project may request space within the filesystem which can then be mounted on both systems. This provides one data repository for the users and enables simple data flows between, e.g., high-performance simulations and post processing or data analytics and machine learning on cloud servers.

The current activities are focused primarily on hardware commissioning and realizing solutions for the initial partners and main funders. In parallel, best practices are being developed for new projects, for which the MPCDF-funded parts of the HPC Cloud will be open in late 2021.

Brian Standley, John Alan Kennedy

Poetry: Packaging and Dependency Management for Python

Introduction

Poetry is a tool for *"Python packaging and dependency management made easy"* (https://python-poetry.org/). It supports Python developers during the process of code writing in several ways:

- **Dependency Management**: Poetry is tracking and managing all package dependencies of a Python project.
- **Virtual Environment**: By creating a virtual environment, Poetry takes care of an appropriate development and runtime system.
- **Publishing**: Poetry is tightly integrated into the *Python Package Index (PyPI)*. Poetry projects can be easily published to PyPI.

Poetry can be installed into a user's home directory, so there is no need for a system-wide installation. More information on installing Poetry can be found here[6].

A Poetry project and initial configuration

After successful installation, a new Poetry project can be created:

```
poetry new my-poetry-project
```

This command creates a new folder, which contains already the basic structure of a Poetry project:

- **README.rst**: the README file for the project, will be displayed as an overview page of GitLab or GitHub projects
- **pyproject.toml**: configuration file
- **my__poetry__project** (folder): the place for your code, already as Python Package declared via an empty file ___init___.py
- **tests** (folder): Python unit tests are going here

The configuration file *pyproject.toml* after the initialization of a new project consists of several sections:

```
[tool.poetry]
name = "my-poetry-project"
version = "0.1.0"
description = ""
authors = ["Tom Zastrow"]

[tool.poetry.dependencies]
python = "^3.8"

[tool.poetry.dev-dependencies]
pytest = "^5.2"

[build-system]
requires = ["poetry-core>=1.0.0"]
build-backend = "poetry.core.masonry.api"
```

[6]https://python-poetry.org/docs/#installation

Dependency Management

The section *tool.poetry* takes some generic metadata. The sections *tool.poetry.dependencies* and *tool.poetry.dev-dependencies* are declaring the dependencies of the project during development and deployment. After adding external packages, they need to be installed (Poetry is fetching the packages from the *Python Package Index*):

```
poetry install
```

In the background, Poetry has now created a virtual environment for the current project and installed the dependent packages.

Besides manually editing the dependencies, it is also possible to let Poetry do the work. For example, the following command installs the *Pandas* library and all its dependencies into the current project:

```
poetry add pandas
```

Running your code

An individual Python script can be executed in the current virtual environment:

```
poetry run python my-script.py
```

It is also possible to permanently activate the virtual environment as Poetry shell:

```
poetry shell
```

Package creation and publishing

After you are done with programming, Poetry can create a package out of the current project:

```
poetry build
```

You will find the result in the folder *dist*. As a last step, Poetry helps you publishing your project on the *Python Package Index (PyPI)*:

```
poetry publish
```

This command will ask you for your PyPI credentials and upload your package. After a short time, the package will appear in the PyPI catalogue.

Conclusion

Poetry is a tool which simplifies some common steps during the development of a Python application. Besides the commands demonstrated in this article, Poetry's full list of commands can be displayed:

```
poetry --help
```

Thomas Zastrow

More functionality for the SelfService - Call to action for 2FA users

Migration of MyMPCDF functionality

Following the switch to the new website in March the MPCDF SelfService platform has received a range of additional features that had previously been located in the "MyMPCDF" section of the MPCDF website. The following services were migrated from MyMPCDF to the SelfService:

- Account creation
- Password change
- Mailing list administration
- Administration of supervised users
- TSM backup form (release later this month)
- Accounting view (release planned for next month)

There is also a new link to the vacation notice e-mail setting located under "My Account". The corresponding forms now have a more modern feel and some extra functionality has been added. Please, send suggestions on how to further improve the SelfService to support@mpcdf.mpg.de[7].

Updated password policy

The migration of the password change feature to the SelfService entailed a stricter validation of new passwords. Passwords now have to pass a check against the cracklib library which rejects passwords that are based on dictionary words or are too simplistic. Long passphrases are still accepted. In a later iteration passwords will also get checked against the haveibeenpwned database. This database contains passwords that have been leaked in attacks against other websites and must be considered insecure.

Our users interact with a plethora of different client software to connect to our services and not all software accepts all characters in passwords. While we try to allow as many characters as possible we were forced to limit the set of allowed characters to increase compatibility.

Two-Factor Authentication (2FA) - Call to action

It is important that our users enroll at least two different tokens to prevent getting locked out if their token gets lost. A common scenario is the switch to a new smartphone since the existing app token does not automatically transfer to the new phone. When the user realizes this the old phone is often already reset to factory settings. This is why the SelfService now enforces enrolling at least one secondary token after the enrollment of a primary token (app or hardware). The available secondary token types are:

- **SMS token** (protects against loss of app or hardware token)
- **E-mail token** (protects against loss of phone or hardware token)
- **TAN list** (protects against loss of phone or hardware token; needs to be printed and stored at a secure location)

If you don't have a secondary token as a backup mechanism yet, please enroll one to avoid losing access to our services if you lose your primary token. You can learn how to enroll a backup token in our FAQs[8].

Amazigh Zerzour, Andreas Schott

[7]mailto:support@mpcdf.mpg.de
[8]https://docs.mpcdf.mpg.de/faq/2fa.html#how-do-i-enroll-and-use-a-secondary-backup-token

News & Events

Brochure “High-Performance Computing and Data Science in the MPG”

The MPCDF has issued a brochure with examples of science that is currently being supported by the MPCDF services. The by-no-means comprehensive selection includes 28 articles from astrophysics, brain research, materials and bio science, high-energy physics, plasma physics and fusion research, turbulence research, demographics, contributed by various Max Planck Institutes and is available on our webpage[9].

Erwin Laure, Markus Rampp

GPU bootcamp

For the first time, Nvidia and MPCDF organize a so-called GPU Bootcamp from October 19th to 20th. In two half-days the basics of GPU programming (architecture, libraries, OpenACC) will be introduced and extensive hands-on sessions with OpenACC example codes shall help the participants become familiar with GPU programming. Registration and further details can be found at the event website[10].

Introductory course for new users of MPCDF

Another edition of our biannual introductory course for new users of the MPCDF will be given as online course on October 13th, 14:00 to 16:30 (CEST). Registration is open until October 1st, via the MPCDF website[11].

Advanced HPC workshop

MPCDF will again organize an advanced HPC workshop for users of the MPG from Monday, November 22nd until Wednesday, November 24th, 2021 with an optional day with hands-on on Thursday, November 25th. The workshop will be given online. The main topics of the lectures are

- Software engineering for HPC codes (git, gitlab, CI, testing)
- Debugging and profiling of CPU and GPU codes
- Porting codes to GPU-accelerated systems

As a prerequisite, we require participants to have an account for the HPC machines of MPCDF and are already familiar with accessing, building and running their codes.

If you are interested in bringing in your own code to work with the experts and to apply the techniques and tools taught in the lectures, please apply by adding a short description of your code and specific goals in the registration form. The entire Thursday, November 25th is dedicated to working on the selected code projects.

The workshop will be given by members of the application group of the MPCDF together with experts from Intel and Nvidia. The registration will open soon and can be accessed via the MPCDF training website[12]. The deadline for registration for the lectures is November 12th. Please note the earlier deadline October 15th for the additional hands-on day. The applicants for the hands-on day are asked to test the building of the code on *Raven* and to prepare a representative test case for the problem that they want to inspect (ideally the test can be run on a single *Raven* node, either GPU or CPU) in advance of the workshop. Assistance by MPCDF is provided on request.

Python for HPC

From October 5th to October 7th, 2021, MPCDF offers another iteration of the course on *Python for HPC* to participants of the MPG. The course will be given online via Zoom with lectures in the morning and exercises in the later afternoon.

The course teaches approaches to use Python efficiently and appropriately in an HPC environment, covering performance-related topics such as NumPy, Cython, Numba, compiled C and Fortran extensions, profiling of Python and compiled code, parallelism using multiprocessing and mpi4py, and efficient I/O with HDF5. In addition, topics related to software engineering will be addressed, such as packaging, publishing, testing, and the semi-automated generation of documentation.

The lectures will be given based on Jupyter notebooks and will include many reusable code examples. For each topic, hands-on exercises will be provided and discussed in separate sessions. On the last day, there will be time for a general question & answer session. The registration is open and can be accessed via the MPCDF training website[13].

Tilman Dannert, Sebastian Ohlmann, Klaus Reuter

[9]https://www.mpcdf.mpg.de/MPCDF_Brochure_2021
[10]https://gpuhackathons.org/event/mpcdf-n-ways-gpu-programming-bootcamp
[11]https://www.mpcdf.mpg.de/services/training
[12]https://www.mpcdf.mpg.de/services/training
[13]https://www.mpcdf.mpg.de/services/training