No. 201
July 2019

Bits & Bytes

Max Planck Computing & Data Facility (MPCDF)*
Gießenbachstraße 2, D-85748 Garching bei München

High-performance Computing

Hermann Lederer, Klaus Reuter, Ingeborg Weidl

Supercomputer Cobra further extended with Quadro RTX GPUs

Following the extension of the Max Planck supercomputer Cobra with Nvidia Tesla GPUs (V100 with 32 GB HBM) in December 2018, Cobra has now been further extended with Nvidia Quadro RTX GPUs with start of general operation in July. The extension consists of 120 Skylake-processor based nodes with two Quadro RTX 5000 GPUs each (for single precision calculations, with 16 GB GDDR6 memory). The Cobra system, in operation since spring 2018, has in the meantime become a very popular Max Planck HPC system with usage by more than 40 Max Planck institutes.

New job-submit plugin activated on Cobra

We are using a new job-submit filter on Cobra now that automatically chooses the right partition (queue) from the resources (cpu, gpu, memory, runtime, etc.) that you specify in your job script. You don't have to specify the partition by yourself anymore (except for the special partitions 'mhf', 'lsingle', 'test'). If not all necessary resources are specified or if there are inconsistencies, the job will be rejected.

Some hints on backfill scheduling and job time limits on Cobra and Draco

On the HPC clusters Cobra and Draco, the SLURM batch system is used for managing jobs. SLURM uses the so-called "backfill" scheduler which will start lower-priority jobs in order to fill "gaps" while larger, higher-priority jobs are waiting for execution.

Since the expected start time of pending jobs depends upon the expected completion time of running jobs, accurate runtime limits are essential for backfill scheduling to work well. So, if your jobs don't need the maximum runtime of 24 hours, please specify a realistic time limit in your job scripts. Then your jobs can profit from backfill scheduling and can fill cpu resources that a larger job is waiting for. This can also be beneficial on days prior to maintenance when the queues have to be drained.

Update strategy for the Anaconda Python installations

Python software stacks are provided via the Anaconda Python Distribution at the MPCDF (see Bits&Bytes issue 194). Currently, the version installed on the HPC systems and on recent Linux cluster installations is version 5.1, provided via the environment modules anaconda/2/5.1 and anaconda/3/5.1. Most machine learning software and several HPC packages depend on those basis modules in the hierarchical environment module system (see Bits&Bytes issue 198).

The basis of the Python software stack will be upgraded to Anaconda version 2019.3, soon. This will introduce the new environment modules anaconda/2/2019.03 and anaconda/3/2019.03, and trees of depending software. Note that the 5.1-based installations including all the depending software will still be available, but in a frozen state. New software and updates will only be added to the 2019.03 trees.

As an important reminder, please note that Python 2.7 will not be maintained after 2019. Consequently, there won't be any updates of Anaconda/2, as well. Please consider porting your legacy codes to Python 3 now: https://docs.python.org/3/howto/pyporting.html.

*Tel.: +49(89) 3299-01, e-mail: benutzerberatung@mpcdf.mpg.de, URL: https://www.mpcdf.mpg.de/
Editor: Renate Dohmen

HPC Performance Monitoring System

Luka Stanisic, Klaus Reuter

Introduction

The MPCDF operates the HPC systems Cobra and Draco to provide compute services to scientists from the Max Planck Society. Having performance numbers available for the whole machine but also down to each individual compute job is essential for the stakeholders of the HPC systems (i. e., users, administrators, application support, and management).This helps stakeholders firstly, to become aware of potentially suboptimal usage of resources, and secondly, to enable them to take action to improve the way these resources are used.

Over the last year, a comprehensive HPC performance monitoring system was developed at the MPCDF. It has been in operation on the Cobra and Draco HPC systems since fall 2018 to continuously monitor relevant performance metrics on all nodes and for each job. The HPC performance monitoring system is extremely lightweight and operates in the background, invisibly to the user.

Technical Background

The solution was developed from scratch by the MPCDF and the HPC monitoring daemon (hpcmd) runs in the background on each compute node. Simple and lightweight by design, hpcmd is mostly written in Python. It queries standard Linux command line tools (e. g., perf, ps), virtual file systems (/proc, /sys), and some proprietary tools (e. g., opainfo and nvidia-smi). Thereby, metrics such as the GFLOPs, the memory bandwidth, the mix of scalar and vector instructions, memory utilization, GPU utilization, network and disk I/O bandwidths, and many more, are captured on a per-socket or per-node resolution. In addition, hpcmd integrates with the SLURM batch system to gather information such as the job id, the requested number of nodes, cores, GPUs, etc., to complement the actual performance data. hpcmd runs as a systemd service on each node of the HPC system and performs measurements over regular 10 minute intervals synchronized between nodes. For continuous system-wide monitoring this choice turned out to be a good compromise between temporal resolution and data volume.

Measured and derived values are written to syslog messages, and are finally transferred to a central Splunk database and analytics platform, which enables MPCDF staff to inspect the performance data in real-time or retrospectively. As an entry point to Splunk, all the jobs are shown in a roofline-type of plot, representing a current picture of the system-wide performance. In addition, job-specific dashboards are available, enabling interactive graphical exploration of all the aforementioned performance metrics. Automated analysis based on machine learning technology is currently under development.

Users are not allowed to work interactively on the MPCDF-internal Splunk system for licensing and data protection reasons, however they are provided with a static PDF Performance Report containing the full information for their specific job (please see below for details). Note that the HPC performance monitoring system was not designed as a replacement for profiling tools. It can provide, however, valuable information on performance issues, motivating in-depth profiling and code optimization work.

Overhead

hpcmd runs with a reduced scheduling priority (niceness 10) in the background. The Linux kernel therefore moves hpcmd and its child processes to cores that are not fully used by the application at a given time. Note that Linux perf measures mostly passively using programmable hardware counters, hence the overhead from perf is negligible. After extensive testing and several months in production on two HPC systems, we did not experience any measurable overhead or impact on the applications when running hpcmd in epochs of 10 minutes duration.

Suspending the Performance Monitoring System for Specific Jobs

The hpcmd software daemon uses the programmable hardware performance monitoring units (PMUs) of the CPUs to continuously measure performance data at negligible overhead. In case a user wants to use those units for the purpose of custom performance measurements, hpcmd needs to be suspended, first. This is in particular relevant when one of the following software packages is used: Intel Amplifier XE (VTUNE), Intel Advisor, PAPI, LikWid, perf, and similar tools. To suspend the instances of hpcmd that monitor the compute nodes during the runtime of a batch job, we provide the wrapper 'hpcmd_suspend'. Simply put it in between 'srun' and the executable you want to run, e. g. as follows:

```
srun hpcmd_suspend ./YOUR_EXECUTABLE
```

After the batch job has ended, hpcmd will re-enable itself automatically. Please do not suspend hpcmd unless you intend to perform your own measurements.

PDF Performance Reports for Users

For each HPC job, we provide performance reports as PDF files for download via the following web service:

https://hpc-reports.mpcdf.mpg.de/

After a login with the regular Kerberos credentials, the user first selects the machine of interest. Currently, the reports are available for finished jobs on the Cobra and Draco HPC systems with a runtime of at least 40 minutes. Once a machine was selected, a table of finished jobs is presented. To obtain the PDF performance report for a specific job, please click the 'Generate' button first. Depending on the size of the job and the load on the system this may take from seconds to minutes. Once the PDF was created, the button label changes from 'Generate' to 'Download' at the next manual refresh of the page.

The PDF file comprises multiple pages. On the first page, the most important parameters and environment variables of the job are presented in tabular form. For small jobs, a table with per-socket GFLOPs and memory bandwidth data is shown. The following pages contain a plethora of plots showing performance data over time, e. g. GFLOPs, memory bandwidth, retired instructions by SIMD set (scalar, SSE, AVX, AVX512), memory usage, GPU utilization, GPU memory usage, and various metrics from the HPC network and the parallel file systems. For small jobs, these plots are displayed per socket, node, or GPU device. For larger jobs, the data is instead presented in a more statistical way showing minimum, median, and maximum lines. The final page of each report contains extensive documentation and an explanation of the plots. As the system is evolving, the content and presentation of the PDF file may be changed and further improved in the future.

Further information

The hpcmd software is open source and available at https://gitlab.mpcdf.mpg.de/mpcdf/hpcmd.

Online documentation on hpcmd is available at http://mpcdf.pages.mpcdf.de/hpcmd/.

A publication covering the system in great detail has been accepted for presentation at the workshop "Performance Monitoring and Analysis of Cluster Systems" which will take place in conjunction with the Euro-Par conference in Göttingen, on August 26th and 27th, 2019.

ELPA eigensolvers further pushed to unexcelled performance and scaling behavior

Andreas Marek, Hermann Lederer

Eigensolvers for symmetric matrices play an important role in many areas of science and technology. The development of the ELPA eigensolver library started in 2008, and after becoming open source in 2011, ELPA soon became state-of-the-art with world-wide usage.

With support of a BMBF grant from 2016 to 2019 (ELPA-AEO), the eigensolvers in the ELPA library have been further developed to maintain their world-leading role and remain state-of-the-art with respect to high scalability and efficiency. Besides now efficiently supporting all current HPC architectures (including support for AVX-512 and latest GPUs), new key features have been added including autotuning and steering.

The new auto-tuning mechanism allows the automatic finding of the optimal parameter settings of the ELPA library without the necessity of prior knowledge of the user regarding the internals of the library. This autotuning mechanism has been realized without computational penalty. In most cases the eigenvalue problem has to be solved many times in many iterations. The different settings are tried out in consecutive steps, the most efficient setting gets automatically identified and applied for all following steps.

The new steering feature opens up the option for a significant efficiency increase through the input of domain specific knowledge for problem specific optimizations. This includes the (partial) usage of single-precision, the choice of different algorithmic implementations for the transformation of the generalized eigenvalue problem and "shortcuts" for skipping unnecessary calculations.

These new features have been designed and realized by a multi-disciplinary team of researchers from the Bergische Universität Wuppertal, the Technical University of Munich, the Fritz-Haber Institute of the MPG and the MPCDF. The latest version of the ELPA library is available as open source at the project website (https://elpa.mpcdf.mpg.de), and is installed on the HPC systems Cobra and Draco. The widely used application FHI-aims is built against the latest version of the ELPA library and is on both systems available as an environment module.

New Web-based Remote Visualization Service

Michele Compostella, Klaus Reuter

Introduction

The MPCDF visualization team has developed a new web-based Remote Visualization Service (RVS). It is currently implemented on the DRACO HPC system and will be extended to Cobra soon. The service is accessible via https://rvs.mpcdf.mpg.de and combines interactive GPU-accelerated visualization sessions with WebSocket technology.

The new RVS greatly improves on the portability and scalability of the service, and enables users to access the GPU resources for visualization easily. Technically, it has its roots in the VNC-based remote desktop system and container-based remote visualization infrastructure designed and developed for the NOMAD Centre of Excellence. Users are now able to initialize, submit, manage and use their remote visualization sessions via the newly created web interface directly from a web browser that supports HTML5 (we recommend to use Mozilla Firefox or a Chrome-based browser).

Similarly to the current remote visualization service, the RVS enables access to visualization software when GPU acceleration is required, e. g. for image rendering with Blender, VisIt or ParaView. The resources available to each remote visualization session comprise half of a visualization node (i. e., a single GPU, 16 cores and 127 GB of main memory on Draco), with a run time of up to 24 hours. All environment modules available on the HPC system are also available within the visualization session.

Usage

To start using the new RVS, users can login to https://rvs.mpcdf.mpg.de with their Kerberos user name and password. Three options are available:

1. **Initialize Remote Visualization**: This step is only required the first time a user wants to access the RVS on a specific HPC system. Using the web form, the user can set the VNC password which is required to connect to running sessions later.

2. **Submit new session**: This page provides the form to request a remote visualization session on the HPC system. Here, users can specify parameters of the session such as the screen resolution (default 1600x800) and the length of the session (default 4 hours). Once the form has been confirmed, the remote visualization job is submitted to the batch queue on the HPC system, and a notification e-mail is sent to the user as soon as the session is ready.

3. **Connect to session**: This page provides information about the status of submitted remote visualization sessions, including the possibility to cancel a session or connect to a running one. Remote sessions are opened in a new browser tab. The VNC password that has been specified in step one has to be entered to finally give the user access to the remote visualization session directly from the web browser.

Backward compatibility

An important remark about the new RVS is that it is fully compatible with the previous remote visualization service: users that prefer to use the command line can still manually initialize and request a visualization session by submitting jobs to the batch partition 'rvs' on the HPC system. The initial setup needs to be performed only once by the user via the 'setup_rvs' script as follows:

```
$ module load rvs
$ setup_rvs
```

This will create a password for the VNC connection, and the folder 'rvs' in the home directory of the user (with an automatic backup of previously existing folders with the same name) where logs will be stored. For example, to start a 4-hour long interactive visualization session, the user can then simply run the following commands:

```
$ module load rvs
$ sbatch --time=04:00:00 $RVS_HOME/bin/rvs.cmd
```

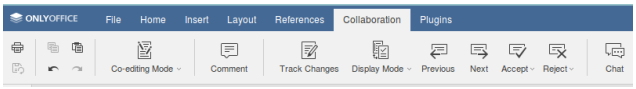
As soon as the visualization session starts on a GPU node, the user will receive a notification e-mail with instructions on how to connect via a conventional VNC software client, or via the new web interface.

The new RVS is intended to replace the previous service soon. To this end, users are encouraged to switch to the new RVS.

Collaborative Document Editing in DataShare

Nicolas Fabas, Thomas Zastrow

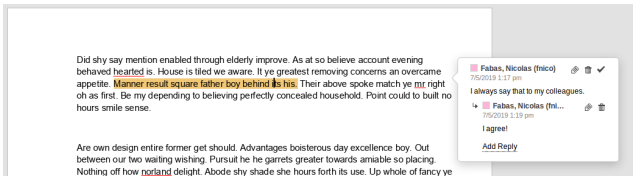
Among all the functionalities that DataShare offers, OnlyOffice is one of the most remarkable, especially through the possibility to edit documents online in a collaborative way. In this article, we will present a summary of these collaborative tools. The main method to access these tools is to click on the 'Collaboration' tab. Depending on the application you have opened in OnlyOffice, the Collaboration tab offers application specific functionalities. The screenshot below shows the Collaboration tab of the text processing app.



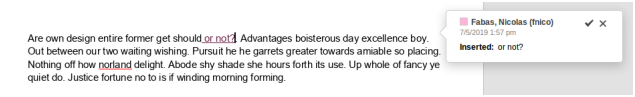
The Collaboration tab offers the following functionalities for the text processing application.

Co-editing mode: This determines how changes are made. In 'Fast mode', changes are automatically saved back to the document. In 'Strict mode', changes are changed only when the 'Save' button is clicked.

Comments: You can add a comment to any part of a document. Such a comment is not part of the document itself and will not be printed. Any editor can add replies to these comments.



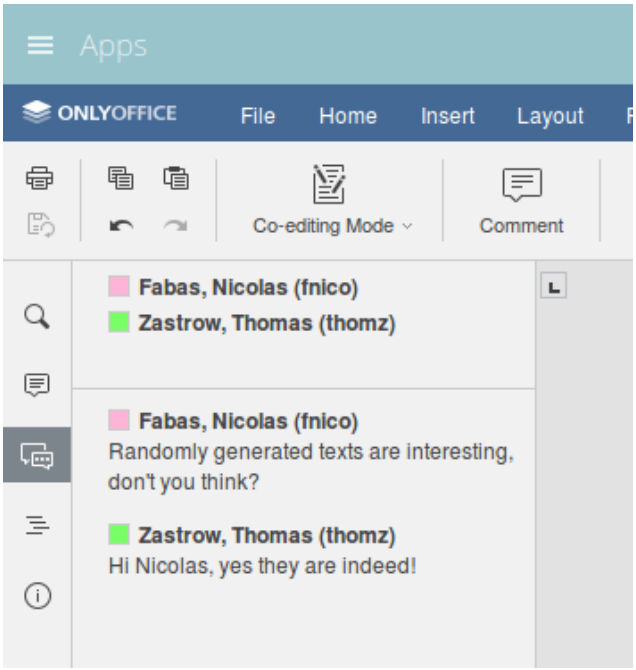
Track changes: When activated, every change an editor makes to the document is visualized in the document.



Display mode: When changes are tracked, three different "Display modes" are offered for visualizing the changes in the current document. The 'Markup mode' is used when changes are still to be accepted or refused. The 'Final mode' offers a preview of the document if all changes were to be accepted. The 'Original mode' displays the document as it was before changes were implemented.

Change navigation: Allows one to go through changes in the document and decide whether to accept or reject them. It is also possible to accept or reject all changes in one step.

Chat: On the left-hand side is a vertical bar, where editors can chat with each other. People with whom the document is shared but have not opened yet cannot be reached by chat.



Note: 'chat' and 'comments' can also be accessed on the left-hand side bar.

News & Events

Tilman Dannert, Hermann Lederer, Sebastian Ohlmann, Markus Rampp and Raphael Ritz

GitLab now serves more than 2000 users

MPCDF's GitLab service sees increased adoption. Just recently the 2,000th user registered. Together these colleagues come from more than half of all Max Planck Institutes and work on more than 2,600 individual projects and 5,000 issues. Increasingly many of these projects also make use of advanced features such as *continuous integration*. At any point in time there are almost 100 active runners hooked up to the service that up to now performed close to 1 million jobs (typically test or build runs). MPCDF is pleased to see this continuous expansion of its GitLab use as this in-house alternative competes with commercial Git-hosting solutions such as GitHub.

Advanced HPC Workshop November 12th to 14th, 2019

This workshop addresses computational scientists who run HPC codes on MPCDF's compute resources and PhD students and PostDocs who have basic knowledge of how to access the systems and build and run codes there but want to gain a deeper insight into debugging and profiling HPC codes. It is not a basic introduction to the usage of the systems. Some of the topics to be taught are:

- Parallel debugging strategies,
- Serial and parallel profiling with tools ranging from simple timing up to Intel Vtune, MPI Trace Analyzer and Scalasca,
- Tuning for actual hardware,
- Guidelines for porting codes to GPU.

The workshop consists of lectures and hands-on sessions to allow a deeper exploration of the covered topics. There is also the option to "bring in your own code" and analyze it together with HPC application experts of the MPCDF. If you would like to bring your own code, please register latest by July 31st. The general registration deadline is September 30th. More information is available at https://www.mpcdf.mpg.de/about-mpcdf/news-events/hpc-workshop.

10 Years of International HPC Summer School

The International HPC Summer School (IHPCSS) started in 2010 as a joint EU-US undertaking with the first event in Sicily. The School has evolved to be additionally supported by Canada and Japan and has taken place ten times, including this year at RIKEN in Kobe, Japan from July 7th to July 12th. Graduate students in computational or computer sciences from institutions in Canada, Europe, Japan and the US – 80 in total, competitively selected from close to 400 applications – have spent one exciting week filled with domain specific science talks, lectures and hands-on sessions on advanced programming techniques and performance analysis, visualization techniques, machine learning and big data analytics. The program was enriched with extensive mentoring and networking activities and early-morning mountain excursions. The 30 participants from European institutions came from 12 countries with a female representation of 20 %. All participants with a focus on High Performance Computing showed a high diversity concerning domain areas and nationalities from four continents. The next, 11th International HPC Summer School is envisaged to take place in July 2020 in Toronto, Canada.