Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG

D1.2 and D1.3 Annual Reports

Kai Himstedt, Julian Kunkel

Work Package:	WP1
Responsible Institution:	DKRZ, RRZ
Date of Submission:	December 2019

Abstract

The second and third annual report are summarized as a single report in this document. The report contains information about the project progress, status, and sustainability and is published on the project webpage. Similar to the first annual report (covering the period March 1, 2017 to February 28, 2018), a brief overview of the major project goals, as well as the the partners and scientific institutions involved in the project is given for the sake of completeness. The relevant information about the general organization of the project and required adjustments is also included.

The results achieved in the six work packages are listed in more detail in the technical section of the report. The emphasis is on performance and software engineering concepts, tuning parallel programs without modifying the source code, an automatic tuning approach using a Black Box Optimizer tool (based on genetic algorithms), and success stories based on best practices to support scientists in their daily work. We have presented the results at various conferences and workshops and hosted a workshop on HPC Training, Education, and Documentation in Summer 2019 in Hamburg. The related documents (presentation slides, project posters, deliverables, software artifacts, ...) are available for download on the project webpage.

Contents

Chapter 1

Introduction

The objectives of the Performance Conscious HPC (PeCoH) project are, firstly, to raise awareness and knowledge of users for performance engineering, i.e., to assist in identification and quantification of potential efficiency improvements in scientific codes and code usage. Secondly, the goal is to increase the coordination of performance engineering activities in Hamburg and to establish novel services to foster performance engineering that are then evaluated and implemented on participating data centers.

The PeCoH project started on March 1, 2017 and was planned for a duration of three years. It was originally planned for one PhD candidate and one postdoctoral full time position. The funding ended on 31 of December 2019.

The first annual report [HK18] covered the period from March 1, 2017 to February 28, 2018. For the sake of efficiency, we have decided to produce a single report covering the period from March 1, 2018 to December 31, 2019 (i.e., M13 – M34)[1]. Hereinafter, this period is referred to as final reporting period. This report is published on the project's webpage [PeC18]. For results achieved in the first reporting period, refer to the first annual report [HK18].

1.1 Partners

The project consortium consist of three research groups from the Universität Hamburg; three local computing centers are affiliated.

1.1.1 Computing Centers

German Climate Computing Center / Deutsches Klimarechenzentrum (DKRZ)

The German Climate Computing Center (DKRZ) is a national service facility and a major partner for climate research. With the hosted high-performance computers Mistral, long-term data storage and services, it provides the central research infrastructure for simulation-based climate science in Germany. One department of the DKRZ is dedicated to user consultancy. It supports the users of the DKRZ in the effective use of its systems by providing general personal advice on the use of the systems, helping users to port applications to the HPC systems, and by offering conceptual guidance on parallelization

[1] In particular, by the end of the second annual reporting period most working packages were still in progress and their results would not have been available for the report. Furthermore, the third annual report, due to the delayed project start by two months caused by the late hiring, would only cover a period of 10 months.

and optimization strategies for user code, specifically with respect to the provided HPC system.

Regional Computing Center / Regionales Rechenzentrum der Universität Hamburg (RRZ)

The HPC team at RRZ operates a 396 node Linux cluster and more than 2 PByte of disk storage. The team is part of the consulting network of the North German Supercomputing Alliance (Höchstleistungsrechenzentrum Nord (HLRN) in German). HPC activities (locally and for HLRN) include user support, user education (in parallel programming and single-processor optimization) and benchmarking. The RRZ maintains and further develops BQCD (Berlin quantum chromodynamics program) [ABS08] as one of the HLRN application benchmark codes [ABS18].

Computer Center of Hamburg University of Technology / RZ der Technischen Universität Hamburg (TUHH RZ)

The TUHH RZ has HPC consultants to support local users as well as users of the North German Supercomputing Alliance (HLRN). The TUHH RZ operates a 244 nodes Linux cluster. TUHH RZ and RRZ cooperate in sharing specialized parts of their HPC hardware.

1.1.2 Research Groups

Scientific Computing / Wissenschaftliches Rechnen at Universität Hamburg

The Scientific Computing group of Prof. Thomas Ludwig has a long history in parallel file system research but also investigates energy efficiency and cost-efficiency aspects and has developed tools for performance analysis. Prof. Ludwig is the director of German Climate Computing Center (DKRZ). The group is embedded into the German Climate Computing Center (DKRZ) since 2009 and, thus, also addresses important aspects for earth system scientists. With regards to teaching, the group offers interdisciplinary seminars and other courses about software engineering in science since 2012.

Scientific Visualization and Parallel Processing (SVPP) at Universität Hamburg

The Scientific Visualization group of Prof. Stephan Olbrich focuses on the development of methods for parallel data extraction and efficient rendering for volume and flow visualization of high-resolution, unsteady phenomena. Prof. Olbrich is the director of Regional Computing Center (RRZ). The group integrated their software into HPC applications, e.g., for climate research. Their activities are part of the cluster of excellence Integrated Climate System Analysis and Prediction (CliSAP) that focuses on post-processing data sets as well as on parallel data extraction at the runtime as part of the simulation.

Software Engineering and Construction Methods / Softwareentwicklungs- und -kontruktionsmethoden at Universität Hamburg

The Software Construction Methods group of Prof. Matthias Riebisch and his prior group at the Ilmenau University of Technology have experience in the adaptation and optimization of software architectures and software development processes. This includes, for example, measuring software quality properties, the forecast of properties after changes

using impact analysis techniques, and methods for partitioning software applications for optimized execution on parallel computing platforms.

1.2 General Project Schedule

The project start was originally scheduled for January 1st, 2017 and we applied for funding of two full time positions for the duration of three years: A PhD candidate is responsible to conduct the mentioned research and helps to transfer them into production via a corresponding service infrastructure. A postdoctoral researcher is responsible to establish performance engineering and to implement the corresponding services.

The good labor market situation for computer scientists and IT experts made it difficult to fill the vacancies as planned and the project started with a two-month delay on on March 1, 2017, when the postdoc position was filled.

The PhD candidate position has been divided among two persons. Both candidates were hired during the first reporting period: The first candidate started on July 1, 2017 (part time position – 24 months), and the second candidate started on January 1, 2018 (full time position – 12 months). Both PhD candidates left the project as planned during the final reporting period.

Based on the situation caused by the late hiring, the split of positions, and an effective project duration shortened by two months, we made several minor adjustments to the project plan during the final reporting period, as we had already done in the first reporting period.

Chapter 2

Technical Report

The PeCoH project plan contains six work packages (WPs) and associated tasks in each WP. This section reports on the activities and the progress made in each WP.

2.1 WP1 Management

As in the first reporting period, the project's progress was regularly verified and compared to the expected status also during the final reporting period. In regular face-to-face meetings, conference calls, and video conferences the project status was determined and discussed with the partners in order to avoid risks like delay of deliverables and the overall project. Furthermore, a series of working-level meetings were held for processing the work packages.

Apart from the minor adjustments to the project plan for organizational reasons as mentioned in Section 1.2, some changes in the requirements of the work packages became necessary in the final reporting period. For example, in WP2 there seemed to be no urgent need to investigate compiler-assisted development techniques and in WP5, after gaining experience with manually tuning parallel applications in a time-consuming way, the focus was realigned, and experiments were performed with a Black Box Optimizer tool, which is based on genetic algorithms, in order to automatically tune parallel applications.

Furthermore, our HPC certification program (also see Deliverable 4.1 [HHK+18]) was positively accepted by the community so that we explored means to sustain this effort. The HPC Certification Forum[1] (HPC-CF) was kick-started in the first reporting period received a good response and therefore we focused on this topic investing more time than anticipated (e.g., to present the status of the HPC certification program in international conferences). Contributing institutions to the HPC Certification Forum are, for example, University of Reading [Rea19], Universität Hamburg [UHH19], EPCC [epc19], DDN Storage [Sto19], University of Melbourne [Mel19], and Johannes Gutenberg Universität Mainz [JGU19]. Recently the ACM SIGHPC Education chapter [ACM19] has expressed its interest in joining the HPC Certification Forum and joint meetings were held.

2.1.1 Project Management (Task 1.1)

The Git repository we set up as a collaborative workspace in the first reporting period at Universität Hamburg was also used in the final reporting period to host all new artifacts created and related to the project (deliverables, source code, meeting notes, etc.). All

[1] `https://hpc-certification.org`

deliverables are published on the project's webpage [PeC18] and are also available now via the HHCC website [HHC18b]. Selected artifacts are also available on the GitHub of PeCoH[2] or are integrated in the GitHuB of the HPC Certification Forum[3].

2.1.2 Coordination between Data-Centers (Task 1.2)

The coordination is supported by the Hamburg regional HPC Competence Center (HHCC) that we bootstrapped as part of PeCoH in the first reporting period. HHCC acts as a virtual organization and is represented by a corresponding website [HHC18a]. The website, established in M1 and available online since M2, was updated constantly in the final reporting period. The idea behind HHCC is to bundle the know-how and to combine the strengths of the three compute centers in order to give their users a better support, particularly by education and by giving feedback. HHCC is also the basis to disseminate existing and new performance engineering concepts. For this purpose, the HHCC supports a list of answers to the most frequently asked questions (FAQ), a list of abbreviations and acronyms, a list of recommended readings, and an area for downloads. The content of the lists and the download area has been extended accordingly in the final reporting period.

Like during the first reporting period, we attended various meetings in the data centers that covered aspects of performance engineering and provided a basic level of coordination. We believe there is more potential behind this idea than actually exploited during the project.

2.2 WP2 Performance Engineering Concepts

2.2.1 Identification of Suitable Concepts (Task 2.1)

A first concept and classification matrix that aims to assess the different concepts has been created in the first reporting period. This was additionally supported by identifying essential performance engineering concepts during the classification of HPC skills for establishing the HPC certification program.

Based on the resulting skeleton of Deliverable 2.1 that was created in the first reporting period, further ideas on performance and software engineering concepts have been documented to complete the deliverable in the final reporting period. The particular challenge is that many HPC users are often not trained in performance and software engineering but only interested in publishing the results of the parallel program as early as possible. They are therefore afraid of an alleged extra effort when using current software engineering techniques. With the collection of suitable concepts we aim to support scientists in improving the performance of their scientific software and increasing the productivity of the software development. The concepts are evaluated against selected criteria in order to show their benefits when applied in scientific programming. The advantages and disadvantages are discussed and the benefits are qualitatively assessed. The collection of performance and software engineering concepts is not considered to be complete. Nevertheless, the deliverable represents a good starting point for a further refinement and identification of performance engineering and software engineering concepts. Applying a selected set of concepts successfully on a real-world project is addressed below in Section 2.2.5.

[2] `https://github.com/pecoh`
[3] `https://github.com/HPC-certification-forum`

For a detailed description of the concepts refer to deliverable 2.1 [HS19].

2.2.2 Benefit of Big Data Analytics (Task 2.2)

Goal of this task was to explore how big data tools can benefit HPC applications and perform a co-development with users. As part of Deliverable 2.1, a high-level analysis of the potential has been described. During the course of the project, it showed that conversions of existing workflows to Big Data solutions were not directly beneficial – at least for the investigated cases. We measured performance on a 5-Node Spark and Hadoop cluster. Native Big Data solutions are often inefficient in terms of processing floating point operations as they are optimized for low-cost servers in which storage performance is the limiting factor but not the available CPU. E.g., in PySpark[4], a startup time of several seconds is required, then a floating point operation had been measured to take 700 cycles, compared to a nearly native operation using NumPy (4 cycles, limited by memory throughput). There are (commercial) adjusted versions that are adapted for HPC environments that promise better performance in those environments, however, we could not explore them further.

Regardless, to estimate gains, a performance modelling tool was developed that can predict performance for different scenarios. The tool is a Python script that reads JSON files that are a performance model description. Using this tool, a specific user problem was analyzed to explore potential design choices including bulk-parallel models from typical Big Data tools such as Hadoop and Spark.

During the project, we were approached by a user that encountered performance issues when reading GRIB (GRidded Binary or General Regularly-distributed Information in Binary form) data using the Climate Data Interface (CDI) [MPI19a]. They experienced a slowdown when reading compressed data. For large files, a runtime of about 30 minutes could be observed. Instrumentation was used to measure the performance of the involved libraries.

The performance measurements were feed into the modelling tool to explore various design choices. Parallelizing the decoding process promised to complete execution below 400 seconds. Bulk-synchronous execution using Big Data tools didn't improve performance further but would have meant substantial changes to the current execution paradigm. This runtime was acceptable for the scientists, so we parallelized the decoding in the existing tool chain. The actual benchmark execution show that the parallelization using 8 workers achieved a speedup of about 5, which was comparable to the predicted time according to the model.

2.2.3 Benefit of in-situ Visualization (Task 2.3)

During the project period, DKRZ implemented a prototypical integration of the ICON [DWD19] model with ParaView. Since there was no urgent need for the evaluation of the Distributed Simulation and Virtual Reality Environment (DSVR) [RRZ15] in-situ visualization framework in an example application, e.g., by complementing the ICON model and running scalability tests, we decided – in agreement with the partner who initially brought this task to the project proposal – not to process it in favour to focus on other activities like automatically tuning parallel programs.

[4] `https://spark.apache.org/`

2.2.4 Compiler-assisted Development (Task 2.4)

Similar to Task 2.3 it became apparent in this task, that the investigation of tools for static code analysis that will assist in predicting performance and improve productivity of scientists, would be of minor importance in comparison to other tasks.

As with Task 2.3 we decided to skip this task.

2.2.5 Code Co-Development (Task 2.5)

This task intended to improve code relevant for users following a co-design approach with the users. Thus, computational challenges brought to by users and identified as potential candidates for improvement were the main focus. In the first reporting period, we started our activities in this task by speeding up an R program using the rlassoEffects-function [SCH18] in the context of regression analysis as a real-world example and an R program for analyzing satellite night images by exploiting the parallelization potential of sequential loops without data dependencies. The associated performance engineering know-how was successfully transferred to end users.

In the final reporting period a collection of suitable performance and software engineering concepts were documented (also see 2.2.1) and a subset chosen that was considered suitable and useful for scientists during their programming tasks. This laid the base for performing a representative code co-development experiment.

Since the code co-development process basically creates a success story, it seems obvious to design and apply a generic template for the outline of a success story. The template is divided into four sections: 1) The selection of the concepts that have been selected for the code co-development, 2) the concrete steps and methods that have been performed for code co-development, 3) the results obtained by code co-development, and 4) supplementary material that has been used in the code co-development process.

To keep the code co-development as simple as possible for HPC developers but with a good transferability of results, we chose the use of an Integrated Development Environment (IDE), refactoring, consistent coding style, documentation, debugging, and unit testing as particularly relevant concepts, given the experience that many HPC users are not aware of the benefits of applying these concepts in practice.

We discussed the selected concepts and methods with the participants. As supplementary material a tutorial has been designed for the participants for each selected concept. To receive the user feedback the participants were asked to fill out a survey after each tutorial part. The results are based on this feedback and collected by talking to the participants: Indentation and naming conventions helped to enhance and keep the understandability of the code and introducing naming convention does not require high effort. Refactoring to divide the code into functions of shorter length keeps the structure of the code and improves understandability. Documentation in form of code comments improved understandability of code.

However, during the code co-development process, we found that it is challenging to introduce software development methods into the development process of scientists in the context of HPC. Especially the documentation of the code was considered time consuming. Scientists fear that following software engineering practices might slow down the entire research process. One plausible explanation for this is that they mostly write code without having sustainability in mind and without considering that the code will often be shared with other scientists. But even if the code will not be shared with other scientists it can be assumed a break-even will be reached also in the field of HPC when scientists use modern software development methods.

Nevertheless further studies should be carried out to broaden the use of software engineering techniques in the field of HPC in order to increase the performance of parallel programs.

For a detailed description of the experience report on the code co-development process refer to Deliverable 2.2 [Sch19].

2.3 WP3 Performance Awareness, HPC Cost Model

Most work of this WP could already be achieved during the first reporting period. The results were provided by Deliverable 3.1 and D3.3 [Hü18b]. The source code was provided as Deliverable 3.2 [Hü18a].

2.3.1 Modeling Costs of Running Scientific Applications (Task 3.1)

This task was completed during the first reporting period.

2.3.2 Reporting Costs of User Jobs (Task 3.2)

This task was completed during the first reporting period.

2.3.3 Deploying Feedback Tools for User Jobs (Task 3.3)

In the first reporting period two tools were developed to apply the four cost models to SLURM jobs: The first tool is supposed to be run from the job-epilogue script and reports the cost of a single job. The second tool is supposed to be run from the command line and calculates costs and statistics for a set of selected jobs based on the accounting records of SLURM.

The tools are capable to be run in a production environment and had been tested. We explored to deploy the tool at DKRZ, however, it requires the agreement of User Group and management. These discussions were started in the first reporting period but lead to complex questions and we did not produce a decision on production roll-out. In the final reporting period, these discussions with the RRZ and DKRZ are still going on.

2.3.4 Analyzing Data and Giving Feedback to Users (Task 3.4)

Originally, it was planned for the final reporting period to use the reporting tools on a monthly basis to reveal the most likely performance issues. For this purpose, we investigated practicable options to give feedback to users: For example, to report about compute time usage the SLURM epilogue can be used, a daily/monthly reporting about online storage usage is available directly and without any special effort, and an instrumentation of archiving commands can be used to report about archive usage.

However, as with deploying feedback tools for user jobs (see Section 2.3.3), which is anyway a necessary prerequisite, we discussed this approach with the RRZ and the DKRZ user-group but could not resolve them, yet.

2.4 WP4 HPC Certification Program

2.4.1 Classification of Competences (Task 4.1)

This task was completed during the first reporting period by the implementation of a meaningful tree of HPC skills for the classification of the HPC competences. In the final reporting period we were still active in some ways in this task in connection with the HPC Certification Forum (HPC-CF).

We initially started with four major branches (“HPC Knowledge”, “Use of the HPC Environment”, “Performance Engineering”, and “Software Engineering for HPC”) and have recently added two branches for “Administration” and “Big Data Analytics” to the skill tree, which is hosted on the HPC-CF website.

Additionally, in connection with the preparation of content for basic level HPC skills, the structure of some skills in the tree was minimally refactored. For a detailed description of the classification of HPC competences regarding the PeCoH project refer to Deliverable 4.1 [HHK+18] and the latest version of our corresponding concept paper [HHKS18].

2.4.2 Development of the Certification Program (Task 4.2)

This task was completed during the first reporting period based on the idea to separate the certificate definition from the providing of content, similar to the concept of a high school graduation exam (Zentralabitur in German).

Our HPC certification program (also refer to Deliverable 4.1 [HHK+18] and the HPC Certification Forum received a good response and, therefore, we were active in this task in the final reporting period. In the first reporting period, we submitted a proposal for a Birds-of-a-Feather (BoF) session to the ISC-HPC conference to start the discussion with contributors. The proposal was not accepted, yet we incorporated the comments of the referees and submitted a revised version to the next year’s ISC-HPC conference [KFGH19]. This proposal was accepted and the BoF stimulated lively and constructive discussions. We received constructive feedback especially for the classification of HPC topics based on the skill tree.

2.4.3 Workshop Material (Task 4.3)

In the first reporting period, the creation of training material was a bit delayed, because we were in discussion with other performance engineering teams and international stakeholders to coordinate the creation of material as part of the HPC certification program. In particular, our goal was to complement existing material and to contribute, and not to recreate similar material. To enhance the discussion further, we hosted a workshop on “HPC-Training, Education, and Documentation” at Universität Hamburg in July 2019 [ZiH19b] and also presented the PeCoH project and in particular the HPC skill tree and our content production workflow [HHS+19b].

The workshop participants came from other projects within the DFG-Call Performance Engineering for Scientific Software [DFG15], and from other institutions. In contrast to the HPC Certification Forum, where HPC learning material is explicitly not in the focus, in the PeCoH project, content has was provided for basic HPC skills. Content production is of central interest for many of the other projects as well. The workshop was characterized by a fruitful cooperation: Immediately after the presentations, the exchange of ideas took place. This also involved the question of how projects can share

content, e.g., using an appropriate Creative Commons (CC) [CC16b] license model. It became clear that license type Attribution-ShareAlike (CC BY-SA) [CC16a] is most suitable.

The HPC-Wiki [HW19], for example, which was developed in connection with the Process-Oriented Performance Engineering (ProPE) project [Pro19], still uses a CC BY-SA license and this is also planned for the HPC learning material that was produced in the PeCoH project.

Our main goal, is to augment the existing material and ongoing efforts to produce valuable teaching material that is complementary and hence most beneficial to users.

However, since some questions regarding the licensing model are not answered, we produced the workshop material for the basic HPC skills on a provisional basis independently from other scientific institutions. Deliverable 4.2 contains the workshop material [HHK+19].

2.4.4 Online Tutorial (Task 4.4)

In the first reporting period, we derived a workflow and tools based on the results from Task 4.1 and Task 4.2 to produce online tutorials. As one major step of the workflow, the sets of skills can easily be bundled beforehand to generate new subtrees. This is similar to defining a new target in a Makefile and thus the skill-tree can be filtered and rearranged in a very flexible way. For the online tutorial, we bundled an appropriate set of skills at the basic level. The new skill was named “Getting Started with HPC Clusters”. At the top level it covers the topics “System Architectures”, “Hardware Architectures”, “I/O Architectures”, “Performance Modeling”, “Parallelization Overheads”, “Domain Decomposition”, “Job Scheduling”, “Use of the Cluster Operating System”, “Use of a Workload Manager”, and “Benchmarking”. Some of these top level skills are divided into sub-skills. As a result of the collaboration of DKRZ, RRZ, and TUHH RZ consolidated information has been applied to the content, e.g., for the creation of tables presenting SLURM commands of various categories (job submission and cancellation, monitoring job and system information, or retrieving accounting information) or a command comparison between PBS/TORQUE and SLURM. The content of a single skill is finally based on a list of Markdown files.

To convert the content files in markdown format – based on the automated build process – to the target formats required, HTML in this case, we use the Pandoc-Tool [Pan18]. For more details on the content production workflow refer to Section 2.4.3 in the first annual report [HK18].

The Content Management System (CMS) Fiona [Fio18], on which the implementation of the HHCC website is based, supports per single user interaction only the upload of individual files via the graphical user interface (GUI). Since the online tutorial consists of a variety of content and helper files (like indices) in the HTML format, connected by hyperlinks for navigating the entire content, an automatic file upload functionality was implemented using Tcl-scripts and using the Fiona system via an appropriate interface [Fio19].

In general, the online tutorial does not depend on the Fiona CMS and, therefore, it would be potentially possible to navigate the tutorial, for example, offline with a standard web browser. We do this for debug purposes. The content files could also be uploaded to other CMS after minor adjustments.

Deliverable 4.3 consists of the tutorial which is available online via the HHCC website [HHC18a].

2.4.5 Online Examinations (Task 4.5)

A fully functional prototype was implemented to carry out multiple choice tests for the online examinations. Once the test is completed and the test results manually approved, the system will assess the results and create a PDF with the certificate. A pool of questions to test basic level HPC skills is in an advanced stage of development, but does not yet cover all topics of basic level HPC skills.

Originally it was planned to link the online tutorial and the online examinations directly and to perform the examinations via the HHCC website. With the development of the HPC Certification Program (HPC-CF) and the separation of the certificate definition from the providing of content it became apparent that the HHCC website is not the proper place to host the examinations. We are confident that the pool of questions to test basic level HPC skills will be completed within a short time and that issuing certificates will then be continuously – also in terms of sustainability – managed via the HPC-CF.

For a detailed description of the prototypical process of the online examination refer to Deliverable D4.4 [Hü19].

2.5 WP5 Tuning of Software Configurations

2.5.1 Help Desk (Task 5.1)

The help desk was set up as a ticketing system in the first reporting period. It received only a few emails during the final reporting period. Users tend to favor the help channels they already know. We are confident that the ticketing system will be better accepted when the visibility of the HHCC website has further increased.

2.5.2 Determination of Tuning Possibilities (Task 5.2)

In the first reporting period, we approached the users and gave priority to the statistics package R. We studied three use cases in order to describe the specific tuning information relevant for the R software package. For two of the three use cases we had studied in the first reporting period we used a co-development approach to exploit the parallelization potential of sequential loops without data dependencies. A typical high level tuning was performed for the third use case, for which no source code was modified, by selecting particularly efficient libraries and by using a suitable combination of compiler and OpenMP/MPI environment. (For more details on the use cases refer to Section 3.5 in the first annual report [HK18].)

All further experiments performed in the final reporting period are likewise based on tuning without or nearly without the source code being modified. This includes experiments for finding good settings for using the standard software packages Gaussian [Gau19b] and MATLAB [Mat19] in an HPC environment. For MATLAB, experiments we additionally parallelized sequential loops using the `parfor` paradigm.

The manual tuning approach used for the experiments was suited to produce good tuning results, but nevertheless traditional manual tuning has turned out to be much more time consuming than anticipated. For this reason, we realigned the focus in the final reporting period to perform experiments with a Black Box Optimizer tool, which is based on genetic algorithms, in order to automatically find the parameter combination for a parallel application that gives the best benchmark result. In initial experiments the Black Box Optimizer was used to automatically tune two small test programs and afterwards, based on the promising tuning results, we determined to use it for tuning

two real applications: BQCD (Berlin Quantum Chromodynamics program) [ABS18], a parallel program for simulating lattice QCD with dynamical Wilson fermions, and Fesom2 (Finite-Element/volumE Sea ice-Ocean Model) [Fes19], a parallel program for simulating the circulation of the global ocean with regional focus.

2.5.3 Setup of Realistic Use Cases (Task 5.3)

Using Gaussian in an HPC Environment

For the Gaussian software parameters for parallel execution were studied and documented together with other parameters that affect performance. For detailed information refer to Deliverable 5.1 [Him19b].

Using MATLAB in an HPC Environment

For MATLAB parallel execution with `parfor` and the setup of a parallel enviroment was documented. For detailed information refer to Deliverable 5.1 [Him19b].

Automatic Tuning Using a Black Box Optimizer Tool

For our automatic tuning approach, the combination of the build- and subsequent benchmark step can be regarded as a simulation model or black box, respectively. The input parameters of this black box control the effective build step (compiler-, version- and MPI selection, optimization level, ...) and the further setting of suitable runtime options (e.g., to tune the MPI configuration). The build step may be obsolete in the case of an optimization from the outside to find the best setting for application specific options, for example, for the domain decomposition by an appropriate partitioning and tiling.

The first approach to solve the problem would be a parameter variation algorithm that finds all possible combinations and thus determines the optimum by a brute-force approach. But such a brute-force approach is not suitable for practical use, because the number of combinations grows exponentially with the number of input parameters. For real problems more efficient search strategies are used, and genetic algorithms have proved especially successful in the past.

Such a Black Box Optimizer tool, based on genetic algorithms, was already successfully used to find best input parameter values automatically with regard to a certain optimal behavior of a coupled system simulating the different processes at an airport [HKMW14]. In the final reporting period, we came up with the idea of using it to tune a wide range of optimization parameters to speed up the runtime of parallel applications. The underlying optimization framework is based on Web Services [W319] and the use of XML standards for the exchange of structured data. Several simulation instances (and in particular the corresponding benchmark steps) may also run in parallel – controlled by a load balancer component – in order to speed up the optimization process itself. The load balancer component can also be used in the interest of fairness to other cluster users to limit the number of simultaneously active jobs in the job queue in order to avoid using too many cluster nodes for the optimization process at once.

Initially, we successfully tuned a small MPI test program to calculate π (contained in the download package of the MPICH implementation of MPI [MPI19b]) and another small MPI test program to solve boolean satisfiability problems (SAT) (see [Bur19] with reference to [Qui03]). The tuning results of the first two test programs were so promising that we decided to switch to the automatic tuning of real applications quite early.

For our first real-world use case we selected the BQCD (Berlin Quantum Chromodynamics) program. BQCD is a Hybrid Monte Carlo program for simulating lattice QCD with dynamical Wilson fermions [HNS19]. The development of BQCD was started in 1998 by Stüben for the two flavour case and the original Wilson action [ABS18]. The sources are available for download [ABS18]. For the second real-world use case we selected the Fesom2 (Finite-Element/volumE Sea ice-Ocean Model), a multi-resolution ocean general circulation model that solves the equations of motion describing the ocean and sea ice using finite-element and finite-volume methods on unstructured computational grids [Fes19]. The sources are available for download via GitHub [Fes19].

2.5.4 Benchmarking (Task 5.4)

In the first reporting period several real benchmark experiments were performed manually for three R uses cases in order to verify that good tuning settings were found. Benchmarking was also of major importance in the final reporting period.

Parallel excution of Gaussian was tested with examples that come with the software. In MATLAB parallelization with `parfor` loops was tested with a simple loop that is given in the MATLAB online documentation.

In the final reporting period, all further benchmarks were performed automatically by the Black Box Optimizer. The genetic optimizer was used to automatically determine the best compiler, best compiler generation, and so on.

First, two test programs (solvers for π and the boolean satisfiability problems (SAT)) were tested. Then, the two scientific programs BQCD and Fesom2 were tested. The automatically tuned version of BQCD was about 10–15% faster than results achieved with parameter settings based on educated guesses by the author of the BQCD program. The automatically tuned version of Fesom2 was as fast as the manually tuned version, whereby the performance of manual tuning being previously based on expert knowledge and time-consuming profiling. For detailed information refer to Deliverable 5.1 [Him19b].

2.5.5 Documentation (Task 5.5)

Deliverable 5.1 contains the documentation of recommendations [Him19b] resulting from the interpretations of the various benchmarks in the first and final reporting periods. The shift of focus from manually tuning parallel applications in the first reporting period to automatically tuning them with the Black Box Optimzer tool in the final reporting period is reflected accordingly in the deliverable.

2.6 WP6 Dissemination

2.6.1 Web Presence of the Hamburg HPC Competence Center (Task 6.1)

The web presence of the HHCC was created during the first reporting period. During the final reporting period, we continuously updated its information, especially with regard to the news section, the recommended readings section, the list of (HPC) abbreviations and acronyms, the download area, and hosting the online tutorial as a bundled set of appropriate HPC skills at the basic level named “Getting Started with HPC Clusters”.

For more information about the website refer to Section 2.6.1 in the first annual report [HK18].

2.6.2 Collecting Success Stories (Task 6.2)

As part of the PeCoH project, standard software, as well as individual software, were enhanced and tuned also in the final reporting period. This task was started in the first reporting period by collecting results with performance improvements for examples using the language R, along with OpenMP and MPI.

In the final reporting period, several representative examples have been examined to derive best practices and strategies supporting the scientists in their daily work, additionally some results from the co-development approach were generated. The results are documented as success stories.

The success stories which are based on code co-development contain five elements: 1) *problem description* to characterize the initial situation 2) *procedure* that outlines the concrete steps and the methods used 3) *results* possibly emphasizing the advantages and disadvantages of the applied concepts 4) *software engineering concepts* that have been selected for the code co-development 5) *material* that have been used in the code co-development process.

For example, in the final reporting period, we have encouraged HPC users who have only used normal editors and the command line interface for the program development in the past to also use integrated development environments (IDEs) like Eclipse [Ecl19] and Visual Studio Code (VS Code) [Mic18] for this purpose. Deliverable 2.2 describes in more detail how HPC users carried out a tutorial to become familiar with a selection of important software engineering concepts like refactoring, using a consistent coding style, documentation, debugging, and unit testing in order to apply them in their everyday work.

In a further success story, it was demonstrated quite spectacularly how an insidious bug in connection with porting BQCD (Berlin Quantum Chromodynamics program), i.e., a large Fortran program, to use SIMD instructions for arithmetic with complex numbers was efficiently found using the GNU project debugger `gdb` [GNU19] via VS Code. As an essential advantage of using a debugger for bug fixing in the present case compared to using print statements for that purpose, as is still popular especially in the field of HPC, it has been shown that particular knowledge of the source code was not required. The debugging process was essentially performed according to a pattern. There exist many debugger extensions for VS Code for comfortably debugging other languages including PHP, Go, Python, C++, and even Bash-scripts.

Another success story dealt with reading GRIB (GRidded Binary or General Regularly-distributed Information in Binary form) data using the Climate Data Interface (CDI) [MPI19a]. It experienced a severe slowdown when the data had been stored using compression. Instrumentation was used to measure the performance of CDI in detail and libraries used by CDI were identified to be responsible for the slowdown. Because one of the two libraries could not be changed with reasonable effort, we decided to parallelize the decoding process with several workers to achieve a speedup. This parallelization was implemented in a way that makes it transparent to the user code. Benchmarks show that the parallelization using 8 workers achieved a speedup of about 5 when reading compressed data.

Increasing the performance of parallel programs using the Black Box Optimizer tool (also see Section 2.5.3) to automatically tune their build and runtime parameters can also be regarded as a success story.

Deliverable 6.2 contains the description of the success stories [Him19c].

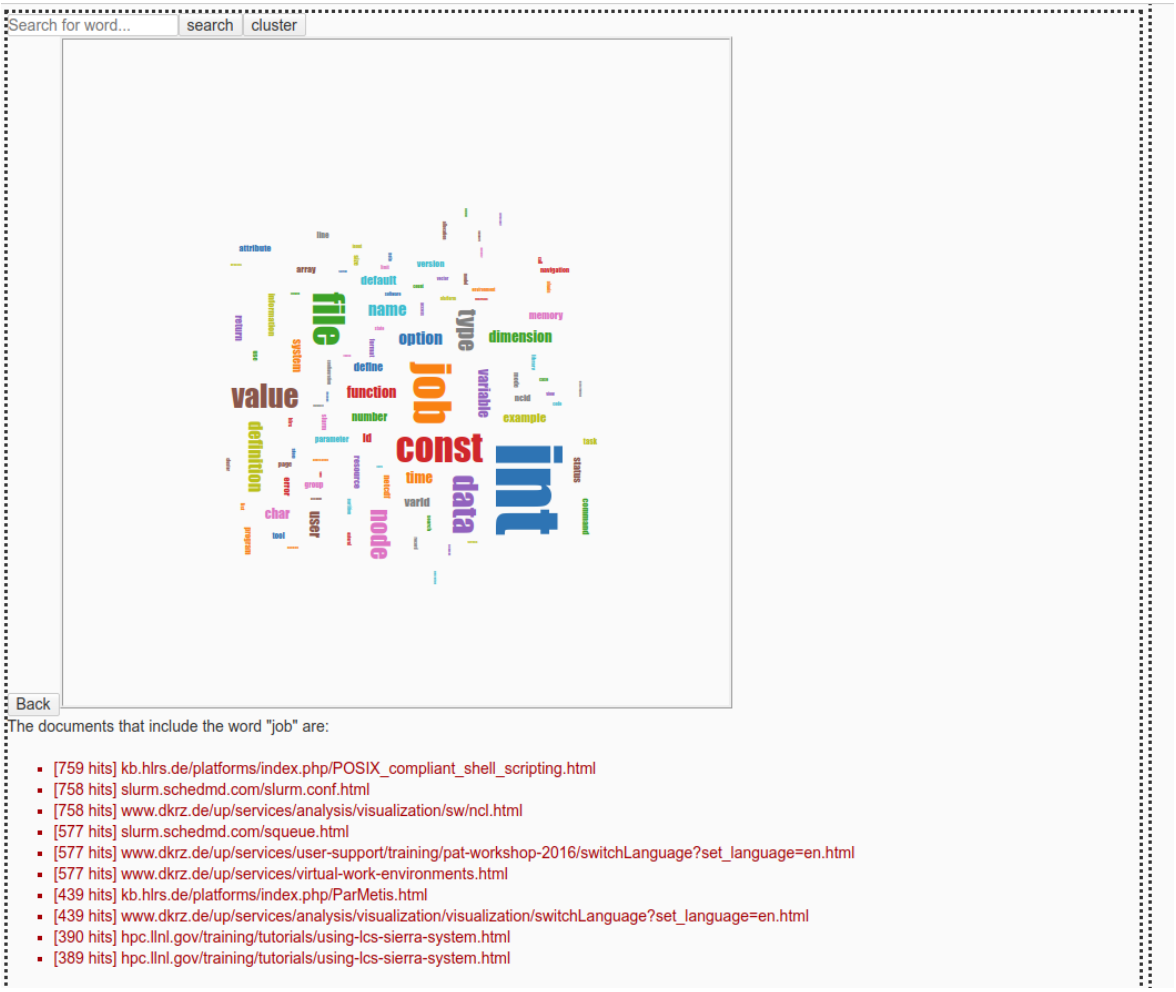

Figure 2.1: Screenshot of the search tool

2.6.3 Knowledge Base (Task 6.3)

The challenge in this task was to increase the searchability of existing performance engineering material. Initially, the annotated bibliography in the recommended readings section of the HHCC website provides a basis for the knowledge base.

In addition, a prototypical web crawler was implemented which uses hyperlinks as they are listed in the recommended readings section to compile a database with HPC specific content. The idea is to enable a targeted search in this HPC related database in order to find the right contents or solutions, respectively.

We explored several variants of the tool, a prototype is shown in Figure 2.1. For this prototype, we crawled the webpages of several trustworthy datacenters and external pages such as NetCDF. In contrast to Google, the viewer shows a Wordcloud of possibly relevant search terms. When searching for example for "Error", the user can see that often this terms occurs together with the term "job", then allowing users to realize, indeed I was searching for a "job error". This allows to narrow down the search. We also implemented clustering on search results, e.g., when clustering a search query for "error", one would obtain clusters with MPI errors compared to Job scheduler errors. At least, that was the idea. While could build a first prototype as part of PeCoH, we could not complete this work into a production system.

2.7 Deliverables

The deliverables that were completed during the final reporting period are listed below. For the sake of completeness the deliverables that were already completed in the first reporting period are merged into the list.

D1.1: “Annual Report” [HK18]
- completed in the first reporting period

D1.2 & D1.3 : “Annual Reports” [Him19a]

D2.1 “Performance Engineering Concepts and Software Engineering Concepts for HPC” [HS19]

D2.2 “Code Co-Development” [Sch19]

D3.1 & D3.3 “Costs for Running Applications” [Hü18b]
- completed in the first reporting period

D3.2 “Integration of Cost-Efficiency in SLURM” (source code) [Hü18a]
- completed in the first reporting period

D4.1 “HPC Competences and Certification Program” [HHK+18]
- completed in the first reporting period

D4.2 “Workshop Material” [HHK+19]

D4.3 “Online Tutorial” available via the HHCC website [HHC19]

D4.4 “Online Examination” [Hü19]

D5.1 “Documentation of Recommendations” [Him19b]

D6.1 “Web presence of the Hamburg HPC Competence Center” [HHC18a]
- completed in the first reporting period

D6.2 “Collection of Success Stories” [Him19c]

All deliverables of the PeCoH project were completed.

2.8 Dissemination Activities

The first version of the HHCC Website is online since April 2017 [HHC18a] and the content was continuously extended during the final reporting period.

In June 2018, we presented our project poster at the ISC 2018 [HHS+18]. We received positive feedback in several meetings and discussions.

In October 2018, we presented the PeCoH project [KHH+18] at the HPC-Status-Conference of the Gauß-Allianz [Gau18]. The presentation focuses on the status of the project and on our HPC Certification Program.

In October, we also presented the HPC Certification Program on a SIG HPC Education webinar[5].

At the SC18 [SC18], in collaboration with other researchers, we presented an extended abstract titled “Towards an HPC Certification Program” [KHH+19] in a lightning talk in the SC18 workshop: Best Practices for HPC Training and Education. We enjoyed the lively exchange with other researchers in the community.

In March 2019, we presented the PeCoH project at the Performance Engineering Workshop [ZiH19a] in Dresden, Germany. We received very positive feedback for our presentation.

In February 2019, a follow up meeting with SIG HPC Education was held regarding the HPC Certification Program[6].

In June 2019, we presented the current project poster at the ISC 2019 [HHS+19a] showing our efforts to establish an “International HPC Certification Program”. Additionally, we presented the skill tree and the PeCoH project [PeC19] in connection with the Birds-of-a-Feather (BoF) titled “International HPC Certification Program” [KFGH19]. The BoF stimulated lively and constructive discussions. We received feedback especially for the classification of HPC topics based on the skill tree.

In July 2019, we hosted a workshop on “HPC-Training, Education, and Documentation” at Universität Hamburg [ZiH19b] and also presented the PeCoH project and in particular the HPC skill tree and our content production workflow [HHS+19b]. Also involved were ideas how content might be shared between the different projects that are also in the DFG-Call Performance Engineering for Scientific Software [DFG15]. The interest of the other workshop participants in a license model like Creative Commons (CC) [CC16b] type Attribution-ShareAlike (CC BY-SA) [CC16a] could be aroused.

In October 2019, we presented the PeCoH project [SHH+19] at the HPC-Status-Conference of the Gauß-Allianz [Gau19a]. The presentation focused on the status of the project, our content production workflow, and using a Black Box Optimizer tool, which is based on genetic algorithms, for an automatic tuning of parallel programs (runtime options, compiler vendor selection, MPI selection, ...).

At Supercomputing 2019, a talk about “One Year HPC Certification Forum in Retrospective“ was given at the “Workshop on HPC Education and Training for Emerging Technologies”.

During the whole reporting period, there was a lively exchange with other HPC projects, in particular with the ProfiT-HPC (Profiling Toolkit for HPC Applications) [Pro18] and the ProPE (Process-Oriented Performance Engineering) [Pro19] projects.

For the sharing of experiences between these projects, we have met six times at the DKRZ and via video conferences.

[5] Slides are available here: `https://sighpceducation.acm.org/events/HPC_certpreso.pdf`

[6] Slides are available here `https://hpc-sig.org.uk/wp-content/uploads/2019/04/HPC_Certification_Kunkel.pdf`

Chapter 3

Sustainability

The HHCC website hosted by the Regional Computing Center at Universität Hamburg (RRZ) has already proven its worth – in particular also for the project workers – as a reliable source of information about the project and its related HPC topics.

When the project has ended a permanent employee at the RRZ will update for the time being the HHCC website regularly, e.g., to extend the recommended readings section or to publish announcements regarding HPC courses or workshops in the news section. At the end of the project, the motivation to produce content was so high that learning material partly exceeding the basic level of a skill was produced for HPC topics. It appears that permanent employees at the RRZ will use the Content Production Workflow as it was implemented in the PeCoH project to continuously publish newly produced learning material in the future.

Often it turns out to be difficult to consider sustainability aspects for projects with a limited duration. In PeCoH we are fortunate to successfully establish the HPC Certifcation Forum (HPC-CF) as a platform, for which is planned to remain active far beyond the end of the PeCoH project. The HPC-CF thus offers an appropriate context, for example, to extend the HPC skill tree, as was recently done by adding the two major branches “Administration” and “Big Data Analytics”, and to finish the development of the multiple choice questionnaire to validate the basic HPC skills. It is planned in particular to organize standardized online examinations for participants and to long-term host the issuing of corresponding certificates.

The HPC-CF is free to join for everyone. However, contributions to the overall program are expected from full members (with voting rights). The development is available under open source license on GitHub [HC19]. The HPC-CF welcomes pull requests and contributions to the skill tree and tools.

The researcher who filled the postdoc position in the project will start a new job on January 1, 2020 at the DKRZ GmbH (which was already involved in the PeCoH project) and plans to remain faithful to the project via the HPC-CF, e.g., by holding the curriculum chair as an active member in the steering board.

Chapter 4

Summary and Conclusions

The project start was originally scheduled for January 1, 2017 and involved three computing centers and three scientific organizations as project partners. The planned duration of the project was three years for two full-time positions (one PhD candidate and a postdoc position). The good labor market situation for computer scientists and IT experts made it difficult to fill the vacancies as planned, so the project started with a delay of two month, when the postdoc position was filled in the first reporting period on March 1, 2017. The PhD candidate position was divided between two persons who were both hired during the first reporting period and left the project as planned during the final reporting period. These deviations from the planning in the first reporting period caused some minor adjustments to the project plan during the final reporting period as well. Basically, the processing order of the tasks in the various work packages was affected. Additionally some changes in the requirements of the work packages emerged in the final reporting period but we could realign the focus at one point or another (e.g., by shifting it in WP5 from manually tuning to automatically tuning parallel programs), so this was not a major issue for the project.

The PeCoH project plan contains six work packages (WPs).

In WP1 (Management) the collaborative workspace (Website, Git repository, ...) that was established at Universität Hamburg during the first reporting period was also used in the final reporting period to foster the cooperation between the project members and the coordination between the computing centers.

In WP2 (Identification of Suitable Concepts) further ideas on performance and software engineering concepts have been documented to complete Deliverable 2.1, which were based on essential concepts identified in the first reporting period. For the code co-development process we chose a subset of these concepts considered suitable and useful for scientists during their programming tasks as follows: use of an Integrated Development Environment (IDE), refactoring, consistent coding style, documentation, debugging, and unit testing. These concepts are particularly relevant, given the experience that many HPC users are not aware of the benefits of applying them in practice. Scientists are asked to apply them in their everyday work. Furthermore, we designed a tutorial to teach them the most important principles of the software engineering practices. For some of the concepts we received positive feedback directly, but the documentation of the code, for example, was considered time consuming. However, we found that it is challenging to introduce software development methods into the development process of scientists in the context of HPC. Scientists fear, in our observation, that following software engineering practices might slow down the entire research process. It seems to be necessary to convince them that a break-even is generally reached rapidly

and that software engineering practices can greatly support them in their programming tasks.

In WP3 (Performance Awareness, HPC Cost Model) most of the work could be completed during the first reporting period by developing two tools to apply four cost models to SLURM jobs: The first tool is supposed to be run from the job-epilogue script and reports the costs of a single job. The second tool is supposed to be run from the command line and calculates costs and statistics for a set of selected jobs based on the accounting records of SLURM. The agreement of User Group and management at DKRZ is required, however, to run these tools in a production environment. Discussions on the topic at RRZ and DKRZ were started in the first reporting period and are still going on. It was also planned to use the reporting tools on a monthly basis to analyze the data and reveal likely performance issues. However, discussions on this topic are also going on as well.

In WP4 (HPC Certification Program) the implementation of a meaningful tree of HPC skills for the classification of the HPC competences was already completed during the first reporting period. Even though the structure of some skills in the tree has to be minimally refactored in connection with the preparation of content for basic level HPC skills, the structure proved to be rather stable. We kick-started the HPC Certification Forum (HPC-CF) in the first reporting period, and in this context we submitted a Birds-of-a-Feather (BoF) proposal titled ”International HPC Certification Program” at ISC-HPC 2019. The BoF stimulates lively and constructive discussions also very fruitful for the HPC-CF. In the first reporting period, the creation of workshop material was slightly delayed. For the efficient creation of the material, we pursued the idea to share the content with other projects also in the DFG Call Performance Engineering for Scientific Software. We discussed this especially during a workshop on “HPC-Training, Education, and Documentation” we hosted at Universität Hamburg in July 2019 and it became clear that the most suitable licensing model corresponds to a Creative Commons license of the type Attribution-ShareAlike (CC BY-SA). Since some questions regarding the preferred licensing model are still open, we produced workshop material for the basic HPC skills temporary independent from other scientific institutions. Nevertheless we were able to make up for the delay in the first reporting period. For the online tutorial we bundled an appropriate set of skills at the basic level and released it via the HHCC website. Fiona is used as Content Management System (CMS) for the HHCC website and we enhanced our content production workflow, which automatically produces HTML files from the Markdown content files, by an automatic file upload functionality that was implemented as Tcl-scripts. For the content production workflow, there is essentially no dependence on Fiona and after appropriate minor adjustments the content files could also be uploaded to other content management systems. For the online examinations, a fully functional prototype was implemented to carry out multiple choice tests. We developed a pool of questions too but it does not yet cover all topics of basic level HPC skills. With the development of the HPC Certification Program (HPC-CF) and the separation of the certificate definition from the providing of content we shifted the online examinations to the HPC-CF website. The HPC-CF will manage the tests and the issuing of certificates as soon as the pool of questions to test basic level HPC skills is completed, which will be in the near future.

In WP5 (Tuning of Software Configurations), experiments were performed that are based on tuning without or nearly without the source code being modified. For the standard software packages Gaussian and MATLAB it was determined how parallelism can be used on a cluster system to use its resources efficiently, e.g., by setting appropriate environment variables or runtime parameters. For MATLAB the `parfor` paradigm was

additionally used to parallelize sequential loops. The manual tuning approach used in the first reporting period turned out to be much more time consuming than anticipated. In the course of the project realization it was therefore replaced by a Black Box Optimizer tool, based on genetic algorithms, that automatically finds the build and runtime parameters for a parallel application that give the best benchmark result. Initially a small MPI test program to calculate π and a small MPI test program to solve boolean satisfiability problems (SAT) were successfully tuned. The tuning results of the first two test programs were so promising that we decided to switch to the automatic tuning of BQCD and Fesom2 as real applications quite early. The experiments showed that the Black Box Optimizer automatically finds good solutions, either comparable to the ones found by time-consuming manual tuning (Fesom2) or actually better (BQCD).

In WP6 (Dissemination), the content of the web presence of the HHCC was continuously updated. All project results are available via the HHCC website. All related documents (e.g., annual reports, information on performance engineering concepts and software engineering concepts for HPC, information on HPC competences and the certification program, workshop material) are available for download on the corresponding HHCC webpage.

All deliverables were completed within the duration of the PeCoH project.

Acknowledgement

The PeCoH project has received funding from the German Research Foundation (DFG) under grants LU 1353/12-1, OL 241/2-1, and RI 1068/7-1.

Bibliography

[ABS08] M. Allalen, M. Brehm, and H. Stüben. Performance of quantum chromodynamics (qcd) simulations on the SGI Altix. *Computational Methods in Science and Technology*, 14(2):69–75, 2008.

[ABS18] M. Allalen, M. Brehm, and H. Stüben. Berlin quantum chromodynamics program (BQCD) — source code download. https://www.rrz.uni-hamburg.de/services/hpc/bqcd, 2018.

[ACM19] ACM. ACM SIGHPC Education: Promotion of interest in and knowledge of applications of high performance computing (HPC) — home page. https://sighpceducation.acm.org/, 2019.

[Bur19] John Burkardt. Circuit satisfiability using MPI — home page. https://people.sc.fsu.edu/~jburkardt/c_src/satisfy_mpi/satisfy_mpi.html, 2019.

[CC16a] CC. Creative Commons — Attribution-ShareAlike 4.0 international (cc by-sa 4.0). https://creativecommons.org/licenses/by-sa/4.0/deed.en, 2016.

[CC16b] CC. Creative Commons — sharing and reuse of creativity and knowledge through the provision of free legal tools. https://creativecommons.org/, 2016.

[DFG15] DFG. Performance engineering für wissenschaftliche software. https://www.dfg.de/foerderung/info_wissenschaft/2015/info_wissenschaft_15_75/index.html, 2015.

[DWD19] DWD. ICON (Icosahedral Nonhydrostatic) – General Circulation Model. https://www.dwd.de/EN/research/weatherforecasting/num_modelling/01_num_weather_prediction_modells/icon_description.html, 2019.

[Ecl19] Eclipse. The platform for open innovation and collaboration – home page. https://www.eclipse.org/„ 2019.

[epc19] epcc. Edinburgh parallel computing centre — home page. https://www.epcc.ed.ac.uk/, 2019.

[Fes19] Fesom2. Finite-element/volume sea ice-ocean model — home page. https://fesom.de/models/fesom20/, 2019.

[Fio18] Fiona. Fiona CMS — home page. https://infopark.com/de/plattform/cms-fiona, 2018.

[Fio19] Fiona. Tcl interface reference. https://kb.infopark.com/tcl-interface-referenz-64b7c175222d9202?locale=en, 2019.

[Gau18] Gauss. 8^{th} HPC-status-conference of the Gauß Allianz (October 8 – 9, 2018 in Erlangen, Germany). https://gauss-allianz.de/en/hpc-status-konferenz-2018?name=value, 2018.

[Gau19a] Gauss. 9^{th} HPC-status-conference of the Gauß Allianz (October 17 – 18, 2019 in Paderborn, Germany). https://gauss-allianz.de/en/hpc-status-konferenz-2019?name=value, 2019.

[Gau19b] Gaussian. Expanding the limits of computational chemistry — home page. https://gaussian.com/, 2019.

[GNU19] GNU. GDB: The GNU project debugger – home page. https://www.gnu.org/software/gdb„ 2019.

[Hü18a] Nathanael Hübbe. Integration of cost-efficiency in slurm — PeCoH deliverable D3.2 (source code). Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2018.

[Hü18b] Nathanael Hübbe. Modelling HPC usage costs — PeCoH deliverable D3.1 & D3.3. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2018.

[Hü19] Nathanael Hübbe. Online examinations — PeCoH deliverable D4.4 (source code). Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[HC19] HPC-CF. HPC Certification Forum on github. https://github.com/HPC-certification-forum, 2019.

[HHC18a] HHCC. Hamburg HPC Competence Center — home page. https://www.hhcc.uni-hamburg.de, 2018.

[HHC18b] HHCC. Hamburg HPC Competence Center — PeCoH project deliverables. https://www.hhcc.uni-hamburg.de/pecoh/deliverables.html, 2018.

[HHC19] HHCC. Online tutorial: Getting started with HPC clusters. https://www.hhcc.uni-hamburg.de/hpc-certification-program/getting-started-with-hpc-clusters-b.html, 2019.

[HHK+18] Kai Himstedt, Nathanael Hübbe, Julian Kunkel, Sandra Schröder, and Hinnerk Stüben. D4.1 HPC competences and certification program. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2018.

[HHK+19] Kai Himstedt, Nathanael Hübbe, Julian Kunkel, Sandra Schröder, and Hinnerk Stüben. D4.2 workshop material. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[HHKS18] Kai Himstedt, Nathanael Hübbe, Julian Kunkel, and Hinnerk Stüben. An HPC certification program proposal meeting HPC users' varied backgrounds. Concept paper https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, DKRZ and RRZ, 2018.

[HHS+18] Kai Himstedt, Nathanael Hübbe, Sandra Schröder, Hendryk Bockelmann, Michael Kuhn, Julian Kunkel, Thomas Ludwig, Stephan Olbrich, Matthias Riebisch, Markus Stammberger, and Hinnerk Stüben. Performance Conscious HPC (PeCoH) – 2018 — project poster. In *ISC High Performance 2018 (June 24 — 28, 2018). Frankfurt, Germany*, 2018.

[HHS+19a] Kai Himstedt, Nathanael Hübbe, Sandra Schröder, Hendryk Bockelmann, Michael Kuhn, Julian Kunkel, Thomas Ludwig, Stephan Olbrich, Matthias Riebisch, Markus Stammberger, and Hinnerk Stüben. Performance Conscious HPC (PeCoH) – 2019 — project poster. In *ISC High Performance 2019 (June 16 — 20, 2019). Frankfurt, Germany*, 2019.

[HHS+19b] Kai Himstedt, Nathanael Hübbe, Sandra Schröder, Michael Kuhn, Julian Kunkel, Hinnerk Stüben, Thomas Ludwig, Stephan Olbrich, and Matthias Riebisch. PeCoH: HPC skill tree and content production workflow — [presentation at the PeCoH workshop on HPC-training, education, and documentation [ZiH19b]]. Slides — online available via our HHCC website https://www.hhcc.uni-hamburg.de/en/support/downloads.html, 2019.

[Him19a] Kai Himstedt. D1.2 and D1.3 annual reports. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[Him19b] Kai Himstedt. D5.1 documentation of recommendations. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[Him19c] Kai Himstedt. D6.2 collection of success stories. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[HK18] Kai Himstedt and Julian Kunkel. D1.1 annual report. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2018.

[HKMW14] Kai Himstedt, Steven Köhler, Dietmar P. F. Möller, and Jochen Wittmann. Ein Framework-Ansatz für die simulationsbasierte Optimierung auf High-Performance-Computing-Plattformen. In Jochen Wittmann and Dimitris K. Maretis, editors, *Simulation in Umwelt- und Geowissenschaften. Workshop Osnabrück 2014*, pages 109–122. Shaker Verlag, Aachen, 2014.

[HNS19] T.R. Haar, Y. Nakamura, and H. Stüben. BQCD manual. https://www.rrz.uni-hamburg.de/services/hpc/bqcd, 2019.

[HS19] Nathanael Hübbe and Sandra Schröder. D2.1 performance engineering concepts and software engineering concepts for HPC. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[HW19] HPC-Wiki. The source for site-independent high performance computing information — home page. https://hpc-wiki.info/hpc/HPC_Wiki, 2019.

[JGU19] JGU. Johannes Gutenberg Universität Mainz — home page. http://www.uni-mainz.de/eng/index.php, 2019.

[KFGH19] Julian Kunkel, Weronika Filinger, Anja Gerbes, and Kai Himstedt. Birds-of-a-Feather (BoF) ISC 19: International HPC certification program. https://2019.isc-program.com/presentation/?id=bof120&sess=sess195, 2019.

[KHH+18] Julian Kunkel, Kai Himstedt, Nathanael Hübbe, Sandra Schröder, Michael Kuhn, Hinnerk Stüben, Thomas Ludwig, Stephan Olbrich, and Matthias Riebisch. PeCoH PeCoH – performance conscious HPC: Status [presentation of the work in progress at the HPC-status-conference of the Gauß-Allianz [Gau18]]. Slides — online available via our HHCC website https://www.hhcc.uni-hamburg.de/en/support/downloads.html, 2018.

[KHH+19] Julian Kunkel, Kai Himstedt, Nathanael Hübbe, Hinnerk Stüben, Sandra Schröder, Michael Kuhn, Matthias Riebisch, Stephan Olbrich, Thomas Ludwig, Weronika Filinger, Jean-Thomas Acquaviva, Anja Gerbes, and Lev Lafayette. Towards an HPC Certification Program. *Journal of Computational Science Education*, pages 88–89, 01 2019.

[Mat19] MatLab. Math. graphics. programming. — home page. https://www.mathworks.com/products/matlab, 2019.

[Mel19] Uni Melbourne. University of Melbourne — home page. https://www.unimelb.edu.au/, 2019.

[Mic18] Microsoft. Visual Studio Code: Code editing. redefined – home page. https://code.visualstudio.com, 2018.

[MPI19a] MPI. Max-Planck-Institut für Meteorologie: CDI climate data interface – overview. https://code.mpimet.mpg.de/projects/cdi, 2019.

[MPI19b] MPICH. A high performance and widely portable implementation of the message passing interface (MPI) standard — home page. https://www.mpich.org/, 2019.

[Pan18] Pandoc. Pandoc: a universal document converter — home page. https://pandoc.org/, 2018.

[PeC18] PeCoH. Scientific computing at Universität Hamburg: PeCoH project – home page. https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2018.

[PeC19] PeCoH. Kai Himstedt – on behalf of the HPC-CF board: The HPC skill tree – a brief overview — presented at Birds-of-a-Feather (BoF) ISC 19: International HPC certification program [KFGH19]. https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[Pro18] ProfiT. ProfiT-HPC: Profiling Toolkit for HPC Applications — home page. https://profit-hpc.de/, 2018.

[Pro19] ProPE. ProPE: Process-Oriented Performance Engineering — home page. https://blogs.fau.de/prope/, 2019.

[Qui03] Michael J. Quinn. *Parallel Programming in C with MPI and OpenMP*. McGraw-Hill Education Group, 2003.

[Rea19] Uni Reading. University of reading — home page. http://www.reading.ac.uk/, 2019.

[RRZ15] RRZ. The Distributed Simulation and Virtual Reality Environment (DSVR) — home page. https://www.dsvr-software.de/, 2015.

[SC18] SC. SC18: International conference for high performance computing, networking, storage and analysis (November 11 — 16, 2018 Dallas, Texas). sc18.supercomputing.org/, 2018.

[SCH18] Martin Spindler, Victor Chernozhukov, and Christian Hansen. hdm: High-dimensional metrics. https://cran.r-project.org/web/packages/hdm/index.html, 2018.

[Sch19] Sandra Schröder. D2.2 code co-development. Deliverable — online available via the PeCoH project webpage https://wr.informatik.uni-hamburg.de/research/projects/pecoh/start, 2019.

[SHH+19] Hinnerk Stüben, Kai Himstedt, Nathanael Hübbe, Sandra Schröder, Michael Kuhn, Julian Kunkel, Thomas Ludwig, Stephan Olbrich, and Matthias Riebisch. PeCoH PeCoH – performance conscious HPC: Status [presentation of the work in progress at the HPC-status-conference of the Gauß-Allianz [Gau19a]]. Slides — online available via our HHCC website https://www.hhcc.uni-hamburg.de/en/support/downloads.html, 2019.

[Sto19] DDN Storage. Datadirect networks — home page. https://www.ddn.com/, 2019.

[UHH19] UHH. Universität hamburg — home page. https://www.uni-hamburg.de/, 2019.

[W319] W3. World Wide Web consortium – web services activity. https://www.w3.org/2002/ws, 2019.

[ZiH19a] ZiH. Performance engineering workshop dresden — workshop (March 25 — 26, 2019 in Dresden, Germany. https://https://event.zih.tu-dresden.de/Events/view/39, 2019.

[ZiH19b] ZiH. Workshop on HPC-training, education, and documentation (July 30 – – 31, 2019 in Hamburg, Germany)]. https://www.hhcc.uni-hamburg.de/pecoh/workshop.html, 2019.