E-Infrastructures
H2020-EINFRA-2016-2017

EINFRA-11-2016: Support to the next implementation phase of Pan-European High-Performance Computing Infrastructure and Services

PRACE-5IP

PRACE Fifth Implementation Phase Project

Grant Agreement Number: EINFRA-730913

D7.3
Evaluation of Tools and Techniques for Future Exascale Systems
Final

Version: 1.0
Author(s): Venkatesh Kannan, ICHEC
Nawar Akhras, ICHEC
Date: 29.03.2019

Project and Deliverable Information Sheet

PRACE Project	Project Ref. №: EINFRA-730913	
	Project Title: PRACE Fifth Implementation Phase Project	
	Project Web Site: http://www.prace-project.eu	
	Deliverable ID: D7.3	
	Deliverable Nature: Report	
	Dissemination Level: PU	Contractual Date of Delivery: 29 / 03 / 2019
		Actual Date of Delivery: 29 / 03 / 2019
	EC Project Officer: Leonardo Flores Añover	

* The dissemination level are indicated as follows: **PU** – Public, **CO** – Confidential, only for members of the consortium (incl. the Commission Services) **CL** – Classified, as in Commission Decision 2991/844/EC.

Document Control Sheet

Document	Title: Evaluation of Tools and Techniques for Future Exascale Systems	
	ID: D7.3	
	Version: 1.0	Status: *Final*
	Available at: http://www.prace-project.eu	
	Software Tool: Microsoft Word version 16.22	
	File(s): D7.3.docx	
Authorship	Written by:	Venkatesh Kannan, ICHEC Nawar Akhras, ICHEC
	Contributors:	Oguz Selvitopi, Bilkent University M. Ozan Karsavuran, Bilkent University Cevdet Aykanat, Bilkent University Georg Zitzlsberger, IT4I Jan Christian Meyer, NTNU Marian Gall, CC SAS Michal Pitoňák, CC SAS Adrián Rodríguez-Bazaga, University of Cambridge Valeria Bartsch, Fraunhofer - ITWM Alan O'Cais, E-CAM CoE David Swenson, E-CAM CoE Mariusz Uchroński, WCNS Adam Włodarczyk, WCNS Jacob Finkenrath, CaSToRC Giannis Koutsou, CaSToRC Swen Metzger, CaSToRC Hendrik Elbern, Juelich Jonas Berndt, Juelich Myles Doyle, ICHEC Guillaume Houzeaux, BSC Ricard Borrell, BSC
	Reviewed by:	Anton Kozhevnikov, ETH Zürich Thomas Eickermann, Juelich
	Approved by:	MB/TB

Document Status Sheet

Version	Date	Status	Comments
0.1	07/03/2019	Draft	First Draft for PMO review
1.0	29/03/2019	Final version	PMO Review comments addressed

Document Keywords

Keywords:	PRACE, HPC, Research Infrastructure, Exascale, Tools, Techniques, Centres of Excellence, FET-HPC projects

Table of Contents

List of Figures

List of Tables

References and Applicable Documents

[1] B. B. Gursoy et al., PRACE-4IP D7.4 Evaluation of Tools and Techniques for Future Exascale Systems, pdf: http://www.prace-ri.eu/IMG/pdf/D7.4_4ip.pdf.

[2] M. Lysaght et al., PRACE-3IP D7.2.1 A Report on the Survey of HPC Tools and Techniques, pdf: http://www.prace-ri.eu/IMG/pdf/d7.2.1.pdf.

[3] B. B. Gursoy et al., PRACE-4IP White Paper Mini-Workshop on Preparing for PRACE Exascale Systems, pdf: http://www.prace-ri.eu/IMG/pdf/WP261.pdf.

[4] To appear at http://www.prace-ri.eu/white-papers/.

[5] Nicola Mc Donnell, PRACE-4IP D7.3 Inventory of Tools and Techniques, pdf: http://www.prace-ri.eu/IMG/pdf/D7.3_4ip.pdf.

[6] David Horák, Marek Pecha, et al. http://permon.vsb.cz/team.htm.

[7] Dirk Pleiter, Markus Richter, Mark Parsons, PRACE-3IP D8.1.1 Technical Specifications for the PCP and for Phase 1, pdf: http://www.prace-ri.eu/IMG/pdf/d8.1.1-resubmit.pdf

[8] Sebastian Von Alfthan, CSC; Stephen Booth, EPCC, PRACE-3IP D8.3.4 Technical Lesson Learnt from the Implementation of the Joint PCP for PRACE-3IP, pdf: http://www.prace-ri.eu/prace-3ip/

List of Acronyms and Abbreviations

ACES IV	Advanced Concepts in Electronic Structure Theory; a quantum chemistry programming package
AISBL	Association International Sans But Lucratif (legal form of the PRACE-RI)
ANTAREX	Auto-tuning and Adaptivity approach for Energy-efficient Exascale HPC Systems
BioExcel	Centre of Excellence for Biomolecular Research
CFD	Computational Fluid Dynamics
CFOUR	Coupled-Cluster techniques for Computational Chemistry; a quantum chemistry programming package
CG	Conjugate Gradient
CINECA	Italian non-profit consortium of 70 universities, 4 national research centres, and the Ministry of Universities and Research (MIUR)
CoE	Center of Excellence
CoeGSS	Center of Excellence for Global Systems Science
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture (NVIDIA)
DDD	Drug Design and Discovery
DEEP-EST	Dynamic Exascale Entry Platform – Extreme Scale Technologies
DMMT	Dense Matrix Transpose Multiply
DMU	Dense Matrix Update
EC	European Commission
E-CAM	An e-Infrastructure for software training and consultancy in simulation and modelling
EoCoE	Energy oriented Centre of Excellence for computer applications
ESCAPE	Energy-efficient SCalable Algorithms for weather Prediction at Exascale
ESiWACE	Excellence in Simulation of Weather and Climate in Europe
ExaHyPe	Exascale Hyperbolic PDE Engine
ExCAPE	Exascae Compond Activity Prediction Engines
FET-HPC	The Future and Emerging Technologies (FET) Proactive call for High Performance Computing
Fraunhofer – ITWM	Fraunhofer Institute for Industrial Mathematics
FS/BS	Forwards and Backward Substitution
GASPI	Global Address Space Programming Interface
GHz	Giga (= 10^9) Hertz, frequency =10^9 periods or clock cycles per second
GPI-2	Second version of GPI that implements GASPI
GPU	Graphics Processing Unit
GROMACS	GROningen MAchine for Chemical Simulations, a molecular dynamics package
H2020	Horizon 2020 framework programme
HPC	High Performance Computing; Computing at a high-performance level at any given time; often used synonym with Supercomputing
HPC2N	High Performance Computing Center North, Sweden
HTC	High-Throughput Computing
I/O	Input/Output
INTERTWine	Programming Model Interoperability Towards Exascale
JURECA	Jülich Research on Exascale Cluster Architectures
JVM	Java Virtual Machine

KNL	Knights Landing architecture from Intel®
LBM	Lattice Boltzmann Methods
LRZ	Leibniz Rechenzentrum at Munich
MANGO	Exploring Many-Core Architectures for Next-Generation HPC Systems
MaX	Materials Design@eXascale
MKL	Math Kernel Library from Intel®
MB	Management Board (highest decision-making body of the project)
MPI	Message Passing Interface
NetCDF	Network Common Data Form
NextGenIO	Next Generation I/O for Exascale
NLAFET	Parallel Numerical Linear Algebra for Extreme Scale Systems
NWChem	An open-source high-performance computational chemistry package
OpenACC	OpenACC is a programming standard for parallel computing developed by Cray, CAPS, Nvidia and PGI
OpenCL	Open Computing Language
OpenMM	A molecular dynamics simulation toolkit
Open MP	Open Multi-Processing
OPS	OpenPathSampling
OVA	One Versus All; a binary classification technique
OVO	One Versus One; a binary classification technique
PCP	Pre-commercial Procurement
PDGEMM	Parallel Double precision General Matrix Multiply
PERMON	Parallel, Efficient, Robust, Modular, Object-oriented, Numerical
PermonSVM	PERMON version implemented with Support Vector Machines
PETSc	Portable, Extensible Toolkit for Scientific Computation
PMO	Project Management Office
POP	Performance Optimisation and Productivity
PRACE	Partnership for Advanced Computing in Europe; Project Acronym
RAM	Random Access Memory
RDMA	Remote Direct Memory Access
READEX	Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing
SAGE	Percipient Storage for Exascale Data Centric Computing
SMP	Symmetric Multiprocessing
SPH	Smoothed Particle Hydrodynamics
SPMD	Single Process Multiple Data
SpMM	Sparse Matrix Dense Matrix Multiply
SVM	Support Vector Machine
TB	Technical Board (group of Work Package leaders)
Tier-0	Denotes the apex of a conceptual pyramid of HPC systems. In this context the Supercomputing Research Infrastructure would host the Tier-0 systems; national or topical HPC centres would constitute Tier-1
VI-HPS	Virtual Institute – High Productivity Supercomputing
WRF	Weather Research and Forecast model
WRF-Solar	Weather Research and Forecast model for Solar energy

List of Project Partner Acronyms

Bilkent	Bilkent University, Turkey
BSC	Barcelona Supercomputing Centre, Spain
CaSToRC	Computation-based Science and Technology Research Center (CaSToRC) of The Cyprus Institute, Cyprus
CC SAS	Computing Center, Centre of Operations of the Slovak Academy of Sciences – CC SAS, Solvakia
ICHEC	Irish Centre for High-End Computing, Ireland
IT4I	IT4Innovations, Czech Republic
NTNU	The Norwegian University of Science and Technology, Norway
WCNS	Wroclaw Centre of Networking and Supercomputing, Poland

Executive Summary

PRACE-5IP Work Package 7 (WP7) 'Applications Enabling and Support' enables support for High Performance Computing (HPC) applications codes to ensure that they effectively exploit multi-petaflop systems and future PRACE Exascale systems. Within WP7, Task 7.2 'Preparing for PRACE Exascale Systems' looks ahead to future European exascale systems and investigates the tools and techniques that are developed to improve the performance and efficiency of applications that will run on these systems.

This deliverable reports on the exploitation of state-of-the-art HPC tools and techniques developed in different European FET-HPC projects and applies them for developing, analysing and optimising the application codes. The application codes were selected with a focus on the European scientific and engineering research communities by working with European Centres of Excellence (CoEs). Much of the work reported in this deliverable is an extension to the work carried out in PRACE-4IP Task 7.2a as reported in D7.4 'Evaluation of Tools and Techniques for Future Exascale Systems' [1] and in PRACE-3IP WP7 as reported in D7.2.1 'A Report on the Survey of HPC Tools and Techniques' [2].

PRACE-5IP Task 7.2 was implemented through four lines of action, which are a follow up to the four main topics that were addressed in PRACE-4IP D7.4 [1], and are summarised below:

1. **Novel Algorithms**
   Several of the application codes within CoEs are concerned with solving large sparse and dense linear systems, where the performance of such solvers greatly influences the overall scalability and performance of the applications. In this line of action, we have focussed on improving the scalability of new iterative solvers proposed in the FET-HPC project NLAFET. We find that encapsulation of computational load balance during sparse and dense matrix computations simultaneously leads to better scalability, and that 2D-partitioning models perform better in general for smaller block sizes and a large number of processes. Another work in this line of action is on analysing and improving the implementation of PermonSVM, which is a Support Vector Machine (SVM) version of the PERMON application suite [v]. The findings contributed to improving PermonSVM through reduced storage and memory requirements, and reduced data conversion overhead when loading data objects for training with the SVM.
2. **Novel Programming Models**
   We noted from the survey of CoE requirements carried during PRACE-4IP T7.2a, that many of the CoEs have not yet shown evidence of exploring emerging or novel programming models, beyond the now well-established models, such as MPI, OpenMP, CUDA and OpenCL. As a result, we have focused mainly on exploiting these well-known models on emerging many-core systems and, where appropriate have focused on investigating upcoming programming models such as GASPI. In this line of action, we have focussed on the use of task-based programming models to build scalable parallel libraries for applications related to computational fluid dynamics to be able to target scalability on Exascale systems. Additionally, we have also leveraged the results from the FET-HPC project INTERTWine, in which one of the main research subjects was the interoperability between GASPI and MPI (and other parallel frameworks). This is in line with some of the recommendations from PRACE-4IP in deliverable D7.4 [1]. We see the lack of awareness of emerging and novel programming models, as well as a possible aversion to risk in utilizing these, as a worthwhile challenge for WP7 to confront more aggressively beyond PRACE-5IP. With a view to future exploitation of programming models (e.g. in PRACE-6IP), we feel that there is an important role for similar tasks to

carry out exploratory work on more novel programming models and tools, as well as to increase awareness of such programming models within the CoEs.

3. **Novel Coding and I/O Techniques**
   Parallel execution on heterogenous platforms and I/O performance plays a key role in many high-performance computing (HPC) applications and future Exascale systems. Furthermore, simulation applications that are of particular interest to the E-CAM CoE require execution many thousands of times with varying inputs, and their computational characteristics often fit different parallel hardware. To manage such computations on future heterogeneous Exascale systems, a flexible, scalable, efficient and adaptive task scheduling library has been developed. It is capable of simultaneously utilising CPUs, Intel KNL and GPUs, and dynamically adjusts the scheduling based on the workload. Additionally, weather forecasting models such as WRF and WRFSolar are globally used community codes and of particular interest to EoCoE. To address this, we have investigated the scalability and readiness of the WRF and WRFSolar models on PRACE Tier-0 systems with different architetures. This includes identification of functions with high workloads and, an analysis of I/O and data management to understand the potential bottlenecks on Exascale systems.
4. **Energy Efficiency**
   Over the lifetime of PRACE, the energy consumption of large-scale European HPC systems has continued to increase. It poses a significant challenge in meeting the power constraints of future European Exascale systems to be commissioned by the early 2020s time-frame. It is now increasingly appreciated that a more holistic approach, which includes all layers of the HPC stack including the application layer, is required. In this context, several CoEs are keenly focused on improving the energy efficiency of their applications, several of which are widely used in the European scientific community and find their place in several benchmark suites. With this interest, we investigate the application of a tool suite, known as READEX, that was developed by the H2020 FET-HPC project (also named READEX) to address the optimisation of parameters on all layers of the HPC stack. We find that the READEX tool suite is capable of identifying dynamism in highly scalable European applications that regularly use large-scale PRACE resources and leverages it to improve their energy efficiency on such HPC systems. We apply the READEX tool suite on a widely-used multi-physics simulation application called Alya that is a part of the EoCoE.

In particular, we see the need for an increased focus on energy efficient computing within WP7 in PRACE-5IP in order to take full advantage of the solutions being delivered as part of the final phase of the PRACE Pre-commercial Procurement (PCP) [7], [8] and beyond.

While some of the tools and techiques we have exploited are well established and standardised, such as MPI and OpenMP, we have also driven effort into exploring newer, pathbreaking tools and techniques, such as the GASPI programming model, a novel SVM-based implementation of numerical solvers, weather research and forecasting models, and a runtime tuning tool for increased energy efficiency on petascale and Exascale systems. We feel strongly that the engagement between PRACE, CoEs and the FET-HPC projects (as well as other Exascale research and development projects) reflected by such WP7 projects should be further expanded and strengthened during PRACE-6IP.

Below, we provide a flavour of some of the investigative work carried out as part of Task 7.2. For a more detailed description of each of the projects summarised in this document, we refer the reader to the PRACE-5IP whitepapers associated with the eight projects [4].

1. Introduction

1.1. Purpose of the document

The objective of the PRACE-5IP Work Package 7 (WP7) "Applications Enabling and Support" is to provide enabling support for High Performance Computing (HPC) applications codes, to ensure that these applications can effectively exploit multi-petaflop systems and future PRACE Exascale systems. The Task 7.2, within WP7, "Preparing for PRACE Exascale Systems" aims to look further ahead to future European exascale systems. This is done by investigating the various tools and techniques that are developed to target performance and efficiency on exascale systems by applying them to important application codes.

In this deliverable, we report on the exploitation of state-of-the-art HPC tools and techniques developed in European FET-HPC projects by applying them in the analysis, development and optimisation of application codes that are of interest to scientific and engineering research community. The application codes were selected with a focus on the European community by working with European Centres of Excellence (CoEs). In this sense, this task and report are informed from PRACE-4IP deliverable D7.4, 'Evaluation of Tools and Techniques for Future Exascale Systems', which represented the secod phase of activity in Task 7.2a [1]. Much of the exploitation work reported here is an extension to the work carried out in PRACE 4IP Task 7.2a as reported in D7.4 [1] and in PRACE 3IP WP7 as reported in D7.2.1 [2].

In this task, we focus on four lines of action that we have identified as being important to enable European applications on the road to exascale, and which mirror four of the topics that were addressed in D7.4 [1]. The fours lines of action reported here are as follows:

1) Novel Algorithms
2) Novel Programming Models
3) Novel Coding and I/O Techniques
4) Energy Efficiency

1.2. Structure of the Document

This document presents four sections (Sections 2, 3, 4 and 5) one for each of the four lines of action listed above. In each section, a short introduction is provided to introduce the contents of the individual section, which is then followed by a collection of reports summarising the work and findings of each of the Task 7.2 projects that were executed during the exploitation phase. Section 6 summarises the document by highlighting the work carried out in Task 7.2 and the inferences from projects that were executed in the exploitation phase.

1.3. Organization of Work

The task PRACE-5IP Task 7.2 of WP7 "Preparing for PRACE Exascale Systems" started at the PRACE-5IP Kick-Off meeting in Athens, Greece on February 1, 2017. The task was executed between January 2017 and March 2019, and its implementation was split into two phases: (1) survey phase and (2) exploitation phase.

1.3.1. Survey Phase

The survey phase was carried out between January 2017 and January 2018 with an objective to survey the tools/techniques from FET-HPC projects, applications from CoEs relevant for exascale systems, and identify the best methodology to investigate the application of the

tools/techniques on the applications in the context of European exascale systems. Through task level all-hands calls in Q1 & Q2 2017, it was agreed that within the task the PRACE partners will lead mini-projects that are created to apply FET-HPC tools/techniques for the analysis, development or optimisation of HPC applications. To facilitate the specification of these mini-projects, a 1-day workshop was conducted in June 2017 within the PRACE-5IP WP7 face-to-face meeting at the Forschungszentrum Jülich (FZJ) Germany in June 2017. The objective of the workshop was to bring together representatives from the CoEs, FET-HPC projects and PRACE applications experts.

The following CoEs were represented at the workshop:

- E-CAM, An e-infrastructure for software, training and consultancy in simulation and modelling(https://www.e-cam2020.eu)
- ESiWACE, Excellence in Simulation of Weather and Climate in Europe (https://www.esiwace.eu)
- MaX, Materials design at the eXascale (http://www.max-centre.eu)
- POP, Performance Optimisation and Productivity (https://pop-coe.eu)
- CoeGSS, Center of Excellence for Global Systems Science (http://coegss.eu)
- EoCoE, Energy oriented Centre of Excellence for computer applications (http://www.eocoe.eu)
- BioEXCEL, Centre of Excellence for Biomolecular Research (http://bioexcel.eu)

The following FET-HPC projects were represented at the workshop:

- MANGO, Exploring Manycore Architectures for Next-Generation HPC systems (http://www.mango-project.eu)
- MontBlanc, European approach towards scalable and power efficient HPC platform (http://montblanc-project.eu)
- NextGenIO, Next Generation I/O for Exascale (http://www.nextgenio.eu)
- SAGE, Percipient StorAGe for Exascale Data Centric Computing (http://www.sagestorage.eu)
- ExaNoDe, European Exascale Processor Memory Node Design (http://exanode.eu)
- ExaHyPe, An Exascale Hyperbolic PDE Engine (http://exahype.eu)
- ESCAPE, Energy-efficient Scalable Algorithms for Weather Prediction at Exascale (http://www.hpc-escape.eu)
- ExCAPE, Exascale Compound Activity Prediction Engine (http://excape-h2020.eu)
- ANTAREX, AutoTuning and Adaptivity appRoach for Energy efficient eXascale HPC systems(http://www.antarex-project.eu)
- ALLScale, An Exascale Programming, Multi-objective Optimisation and Resilience Management Environment Based on Nested Recursive Parallelism (http://www.allscale.eu/home)
- READEX, Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing (http://www.readex.eu)
- INTERTWinE, Programming Model INTERoperability ToWards Exascale (http://www.intertwine-project.eu)

The PERMON (Parallel, Efficient, Robust, Modular, Object-oriented, Numerical) toolbox was also presented at the workshop as a solution for extreme scale applications.

Based on presentations from each CoE and FET-HPC project at the workshop, discussions to identify potential mini-projects were conducted and initial proposals documented. The proceedings at the workshop were detailed in the white paper WP261 [3]. As a follow up to the workshop, the task participants create proposals for nine mini-projects that would be executed during the exploitation phase of the task. To obtain a formal approval, an intent to

review the mini-project proposals and execute them was submitted to the PRACE management board (MB) with the following as the main guidelines for each mini-project:

- A link to a real application, preferably to a CoE-application;
- A link to a state-of-the-art tool/technique, preferably developed by FET-HPC projects;
- A link to the Exascale and future PRACE extreme-scale systems;
- Collaboration with a CoE or FET-HPC.

In collaboration with the PRACE-5IP PMO, a review panel was created to evaluate the nine mini-project proposals that constituted of the following members:

- George Beckett (INTERTWinE),
- Jesus Labarta (MontBlanc-3),
- Paul Gibbon (EoCoE),
- Anton Kozhevnikov (MaX),
- Florian Berberich (PRACE PMO),
- Walter Lioen (PRACE MB)
- Debora Testi (PRACE AISBL).

Each mini-project proposal was anonymously reviewed by all members in the panel. The proposals were either unconditionally accepted, accepted with recommendations or rejected. In the first round of reviews, six proposals were accepted unconditionally or with recommendations, while three proposals had rejection votes by one reviewer each. For the conditionally accepted proposals, the recommendations were implemented by the project owners and reviewed by the task leader. For the proposals with reject votes, the review panel agreed to review them again if changes were implemented. Following this, the project owners updated the proposals to address the recommendations and the updated proposals were passed anonymously to the reviewers that gave reject votes who then accepted the proposals. Consequently, nine mini-projects were created for implementation during the exploitation phase of Task 7.2 as summarised in Table 1.

Lines of Action	Projects	PRACE Centres	Collaborations
Novel Algorithms	Scaling Block Conjugate Gradient Variants Orthomin and Orthodir	Bilkent	NLAFET
	Development and Optimization of Multitask SVM for Chemogenomics	IT4I	ExCAPE, POP
	Multi-code Coupling on Heterogeneous Architectures	HPC2N, BSC	DEEP-EST, EoCoE
Novel Programming Models	LBM and SPH Scalability Using Task-based Programming	NTNU	INTERTWinE
	Architectural Scalability of Neural Network Inference Using Task-based Programming	NTNU	INTERTWinE
	Optimization of Computationally and I/O Intense Patterns in Electronic Structure and Machine Learning Algorithms	CC SAS	INTERTWinE, NextGenIO
Novel Coding and I/O Techniques	Approaching Exascale with the Weather Research and Forecasting Solar Model	CaSToRC	EoCoE
	Task Scheduling Library for Optimising Time-Scalar Molecular Dynamics Simulation	WCNS	E-CAM
Energy Efficiency	Investigating Application Dynamism in Alya with READEX for Energy Efficiency	ICHEC	READEX, EoCoE

Table 1. Summary of mini-projects created in survey phase

Towards the end of the survey phase the mini-projects that were created for execution during the exploitation phase were presented at the PRACE-5IP WP7 face-to-face meeting held at CaSToRC in November 2017.

1.3.2. Exploitation Phase

The exploitation phase of Task 7.2 was carried out between January 2018 and March 2019 with an objective to execute the mini-projects. To accomplish this, a workplan was created within each mini-project by the collaborators led by the PRACE partner. The workplan included a set of activities to be carried out for each quarter between January 2018 and March 2019 along with milestones to be achieved for each. This workplan was used to monitor the progress and issues within each mini-project during task-level monthly conference calls.

In February 2018, it was observed that the mini-project headed by HPC2N and BSC in collaboration with DEEP-EST and EoCoE ("Multi-code Coupling on Heterogeneous Architectures") could not be carried out as planned due to unexpected delays in the availability of the DEEP-EST prototype. Hence, with the approval of WP7 leader, this project was discontinued, and efforts of the partners were moved to other tasks within PRACE-5IP.

During the exploitation phase, the remaining eight mini-projects shared their work and status during the task-level monthly calls and at the PRACE-5IP WP7 face-to-face meeting held at Edinburgh in October 2018. The work completed in the eight mini-project with their results have been documented in the form of white papers (WP275, WP276, WP277, WP278, WP279, WP280, WP281 and WP282) [4]. Each white paper was reviewed by two reviewers (one task-internal peer reviewer and one from the PMO). The summaries of these white papers are presented in Sections 2, 3, 4 and 5 in this deliverable.

1.4. Intended Audience

Our objective in preparing this report is to exploit the most promising HPC tools and techniques that may aid preparing important applications for near-term European multi-petascale and future Exascale systems. Targeted primarily at the European HPC community, including the European CoEs, this report provides an overview of how a selection of state-of-the-art HPC tools and techniques fared when enabling real applications targeting petascale systems/future exascale systems. We also hope that the report here will be of interest to the European scientific computing community more generally.

2. Novel Algorithms

In this section, we report on two projects that have each focused on exploiting state-of-the- art algorithmic techniques and tools, in order to enable applications for multi-petaflop/future exascale systems, as summarised in Table 2. Each subsection provides a summary of each project along with a reference to the associated PRACE-5IP whitepaper. We recommend that the reader also refers to the associated whitepapers, which provides a more detailed report than is provided here.

FETHPC Projects/Tools, CoEs	Application
NLAFET: Parallel Numerical Linear Algebra for Future Extreme Scale Systems	Block CG variants Orthomin and Orthodir developed by NLAFET
ExCAPE, POP	PermonSVM with ExCAPE data sets

Table 2: Tools/Techniques exploited along with corresponding applications with in the Novel Algorithms

2.1. Scaling Block Conjugate Gradient Variants Orthomin and Orthodir

WP275: Scaling Block Conjugate Gradient Variants Orthomin and Orthodir

Authors: Oguz Selvitopi (Bilkent University), M. Ozan Karsavuran (Bilkent University), Cevdet Aykanat (Bilkent University)

FETHPC Project/Tool: NLAFET: Parallel Numerical Linear Algebra for Future Extreme Scale Systems

Applications: Block CG variants Orthomin and Orthodir developed by NLAFET

Summary:

Iterative solvers based on Krylov subspace methods are widely used for the solution of the problems that appear in large-scale parallel scientific simulations. These solvers, when parallelized, often suffer from global synchronization overheads due to the collective communication operations as discussed in several works.

The FET-HPC project NLAFET (Parallel Numerical Linear Algebra for Future Extreme Scale Systems) proposed new iterative methods based on Enlarged Krylov subspaces and that are variants of the Conjugate Gradient (CG) method. Block CG variants have the advantage of reduced communication overhead at the expense of increased computation per iteration. These methods are based on Orthomin and Orthodir variants and they dynamically reduce the number of search directions compared to existing block CG methods. These two variants are reported to converge faster and reduce the computation and memory overheads compared to the existing block CG variants for solving linear systems with a single right hand side.

The aim of this project was the scalable parallelization of Orthomin and Orthodir methods proposed by FET-HPC project NLAFET to enable the use of these methods on future exascale systems. Both methods contain the following kernel operations: sparse matrix dense matrix multiply (SpMM), dense matrix update (DMU), dense matrix matrix transpose multiply (DMMT), and forward and backward substitution (FS/BS) on triangular factors of block matrices. Intelligent partitioning of the sparse coefficient matrix is crucial for the efficient parallelization of these kernel operations. The input and output dense matrices of the SpMM operation should be partitioned conformably in order to avoid redundant communication during DMU and DMMT operations. In this setting, SpMM incurs irregular point-to-point communications, DMU incurs no communication, DMMT incurs a regular collective communication, and FS/BS incurs no communication based on sequential factorization of a block matrix.

We investigated and implemented one-dimensional (1D) and two-dimensional (2D) partitioning models with single-constraint (sc) and multi-constraint (mc) variants. In 1D partitioning, we implemented row-wise partitioning of the sparse coefficient matrix utilizing the standard graph model that aims to minimize the total inter-processor communication volume while maintaining computational load balance. In 2D partitioning, we adopted a two-

phase approach in which we utilized the described 1D partitioning in the first phase and we utilized 2D cyclic matrix distribution in the second phase. The 2D model has the nice property of providing $O(\sqrt{K})$ bound on the number of messages sent/received by a processor, where K denotes the number of processors. Note that the bound for the same metric in 1D partitioning is O(K). In both 1D- and 2D-partitioning, using a single constraint aims at maintaining balance on the computational loads of processors during the parallel SpMM operations, whereas using two constraints aims at maintaining balance on the computational loads of processors during both parallel SpMM and DMU-DMMT-FS/BS operations.

Parallel Orthomin and Orthodir codes were developed for both 1D- and 2D-partitioned sparse coefficient matrices. The row-parallel and the row-column-parallel SpMM algorithms were utilized for the former and latter parallelization schemes. The parallel codes follow the steps of the sequential MATLAB code that is provided by NLAFET. The relative performance of the above-mentioned four different partitioning techniques were evaluated by speedup values obtained through running the parallel Orthomin and Orthodir codes on two distributed memory systems, Sariyer and Juwels, with up to 1024 processors for 26 matrices from the SparseSuite Matrix Collection.

Table 3 displays the average speedup values obtained by the parallel Orthomin algorithm, on Sariyer. In the table, columns 3, 4, 5, and 6 display the actual speedup values obtained by 1D-sc, 1D-mc, 2D-sc, and 2D-mc, respectively, whereas columns 7, 8, and 9 display the speedup values of 1D-mc, 2D-sc, and 2D-mc, respectively, normalized with respect to those of 1D-sc.

		actual values				normalized values w.r.t. 1D-sc		
K	t	1D-sc	1D-mc	2D-sc	2D-mc	1D-mc	2D-sc	2D-mc
64	4	39.63	39.83	40.29	40.71	1.01	1.02	1.03
	8	39.74	40.13	40.93	42.01	1.01	1.03	1.06
	16	39.91	40.55	41.58	42.72	1.02	1.04	1.07
	32	40.69	42.51	41.34	42.99	1.04	1.02	1.06
256	4	51.19	56.16	64.97	63.08	1.10	1.27	1.23
	8	60.91	65.00	68.33	71.92	1.07	1.12	1.18
	16	74.65	76.78	84.21	85.62	1.03	1.13	1.15
	32	89.45	94.88	96.42	101.98	1.06	1.08	1.14
1024	4	49.85	53.27	74.61	69.62	1.07	1.50	1.40
	8	68.73	66.61	89.65	85.37	0.97	1.30	1.24
	16	99.83	104.91	113.53	118.39	1.05	1.14	1.19
	32	132.02	143.43	150.22	150.00	1.09	1.14	1.14
average values for different K values								
$K = 64$		39.99	40.75	41.03	42.11	1.02	1.03	1.05
$K = 256$		69.05	73.21	78.48	80.65	1.06	1.14	1.17
$K = 1024$		87.61	92.06	107.00	105.84	1.05	1.22	1.21
average values for different t values								
$t = 4$		46.89	49.75	59.96	57.80	1.06	1.28	1.23
$t = 8$		56.46	57.25	66.30	66.43	1.01	1.17	1.18
$t = 16$		71.46	74.08	79.77	82.24	1.04	1.12	1.15
$t = 32$		87.39	93.61	95.99	98.32	1.07	1.10	1.13

Table 3: Averege speedup values obtained by the parallel Orthomin algorithm.

As seen in Table 3, for parallel Orthomin algorithm, 1D-sc, 1D-mc, 2D-sc, and 2D-sc achieve average speedup values of 132, 143, 150, and 150 for the blocksize t=32 on 1024 processors,

respectively, averaged over all matrices. Note that, compared to 1D-sc, 1D-mc improves the parallel performance by 9% on average, whereas 2D-sc and 2D-mc improve it by 14%, on average. Note that the improvement obtained by the multi-constraint partitioning is more visible for 1D partitionings compared to 2D partitionings.
As a concluding remark, these experimental findings show that encapsulating computational load balance during sparse matrix and dense matrix computations simultaneously via two-constraint partitioning formulation leads to better scalability. 2D-partitioning models are found to perform better in general for smaller block sizes and larger number of processors.

2.2. Development and Optimization of Multitask SVM for Chemogenomics

WP279: Development and Optimization of Multitask SVM for Chemogenomics

Author: Georg Zitzlsberger (IT4I)

FETHPC Project/Tool: ExCAPE

Application: PermonSVM with ExCAPE date sets

CoEs: POP

Summary:

Supervised modelling is widely used in the pharmaceutical industry especially during the early stage of drug design and development (DDD). Computational chemogenomics is a part of the DDD and aims to predict all possible protein targets and biological activities of the chemical compound by reusing all available information to support current and prospective needs. On one hand, this significantly helps to reduce the cost, time and animal usage during the DDD process. On the other hand, the amount of possible combinations grows exponentially with adding new targets or compounds and scales beyond petascale levels for practical databases.

Chemogenomics aims to model all possible biological activities (protein, toxicity, metabolism) of the chemical compound: this can be achieved via multitask supervised methods like matrix factorization and deep learning. Both of those methods are computationally very expensive and require domain knowledge expertise. Single-task models have inherited limitations as they only capture binary context, and do not benefit from transfer learning available in multitask methods. Yet, single-task methods can be turned into multitask methods via binarization (binary classification) techniques such as one versus one (OVO), or one versus all (OVA) – also synonymously referred to as one versus rest. Using a Support Vector Machine (SVM) for such binary classifications is one possible solution.

In the underlying white paper we use the SVM implementation from the PERMON team [v] which is called PermonSVM. Its implementation is, at the time of writing, in an early stage and under heavy development. We will be using a first prototype implementation of an OVA implementation kindly provided by the PERMON team. The PermonSVM source code is avilable on Github. Training data is made avilable by courtesy of the Excape project. We use their baseline *chem2vec* dataset, which was used for evaluation of different machine learning methods. The dataset is applied in a multi-label setup where the features of chemical compounds are related to chemogenomic targets (labels). The result is a multi-label model that allows to relate features of various chemical compounds to a specific target. For performance analysis of PermonSVM, different VI-HPS tools, such as Score-P, Cube and Vampir, are applied. We document the setup of Score-P for the PermonSVM software stack to retrieve profiling and tracing information and visualize them in Cube and Vampir, respectively.

We used Score-P to collect profiling information in two ways. First, we analyzed the entire software stack, which is the easiest solution. Second, we narrowed down analysis to PermonSVM using the library wrappers which allows selective profiling. The setup of the library wrappers for PermonQP and PETSc has been described.

Visualization of the original PermonSVM implementation's profiling data with Cube unveiled a hot spot in the data conversion when loading the training data. We have applied an improvement to reduce this overhead, both for runtime and storage, for a better time to solution.

Finally, both PermonSVM versions were traced with Score-P and visualized with Vampir. Given the size of trace files we documented our approach to reduce storage and memory requirements by applying a filter. The filter was balanced to retain the most important trace information for PermonSVM, its parallelization and (MPI) synchronization. Figure 1 shows the final results by comparing both original and new PermonSVM implementation.

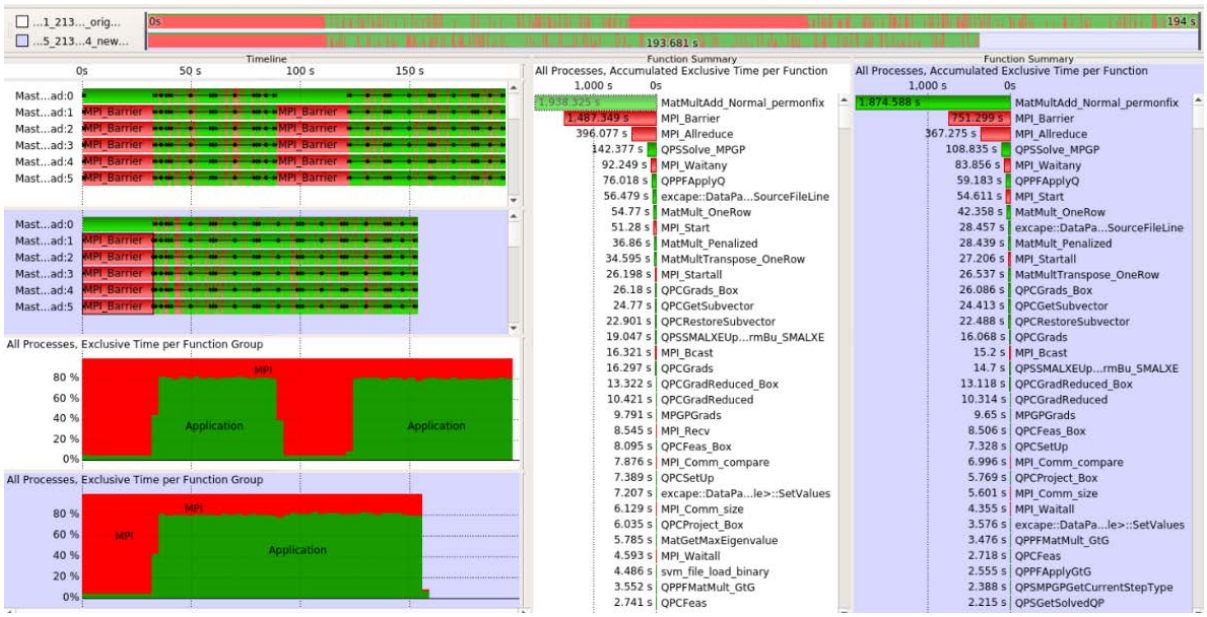

Figure 1: Visualization of Score-P traces with Vampir. It shows the original and improved (blue background) PermonSVM implementation with less overhead and better time to solution.

3. Novel Programming Models

In this section, we report on three projects that have each focused on exploiting the widely used and popular upcoming programming models, in order to enable applications for European multi-petaflop/future exascale systems, as listed in Table 4. Each subsection provides a summary of the project along with a reference to the associated PRACE-5IP whitepaper. We recommend that the reader also refers to the associated whitepapers.

FETHPC Projects/Tools	Application
INTERTWinE	LBM and SPH performance proxy applications
INTERTWinE	Kinetic Ising Model performance proxy application
INTERTWinE	K-means algorithm, TeraSort, Tensor Contractions, parallel matrix multiplication

Table 4: Programming Interfaces and Standards exploited along with corresponding applications

3.1. Architectural Scalability of Neural Network Interface using Task-based Programming Models

WP280: LBM and SPH Scalability Using Task-based Programming

Author: Jan Christian Meyer (NTNU)

FETHPC Project/Tool: INTERTWinE

Application: LBM and SPH performance proxy applications

Summary:

CFD problems admit a great variety of both numerical and technical solutions, each suited to different problem classes, parallelization strategies and performance characteristics of the computing platform. Lattice-Boltzmann Methods (LBM) and Smoothed Particle Hydrodynamics (SPH) both present numerical solutions that admit highly parallel solutions, but the complexity of their programmatic implementation details create challenges with adapting them to emerging computer platforms, as design decisions embedded in a complete application program can result in unanticipated performance constraints on emerging platforms.

The objective of this project is to explore the performance parameters of both LBM and an SPH proxy applications, which are simultaneously developed to make it simple to experiment with programming alternatives, while retaining sufficient detail to simulate known physical effects in a CFD context. We focus on the efficiency of adapting the data structures of the proxy applications so that task-based programming constructs can be applied to them, and investigate the parallel scalability of the resulting solutions. Task-based programming models aim to exploit the potential for hiding memory latency by exposing a greater number of independent, parallel tasks than the number of physical processing cores, thus trading the performance overhead of creating an execution schedule for the possible savings of dynamically adapting it to run-time conditions.

The restrictions of the proxy applications are that they simulate a limited number of known physical phenomena, for arbitrary input sizes. Specifically, the LBM application simulates fluid flow through three particular problem geometries featuring variable amounts of solid matter obstructing the flow, and the SPH application simulates the d*am break* problem, which is a widely used CFD benchmark. Practical applications will dictate problem parameters which may not be as flexible in terms of size and workload, but we consider the obtained performance figures as indicative of attainable scalability under idealized conditions.

By comparing the performance of straightforward implementations with that of versions which have been altered to admit task-based parallelism, this project has identified cases where task-based programming can support the scalability of the applications, in anticipation of future Exascale systems.

The LBM application benefits from being adapted for task-based programming because it enables a partitioning of its problem geometry which can avoid redundant computations in regions of the flow that contain solid bodies. This is visible in Figure 2: Comparison of LBM compute rates for conventional (left) and task-based (right) problem decompositions, the attainable computation rate for the *Moffatt* input problem is substantially greater with the task-based approach, due to the large amount of solid matter obstructing the fluid flow, while a conventional, Cartesian decomposition is favorable for the other two problems, where the problem case contains less solid matter. We conclude that the LBM application can obtain improved scalability by an analysis of the input case at startup, to determine which decomposition is better to apply.

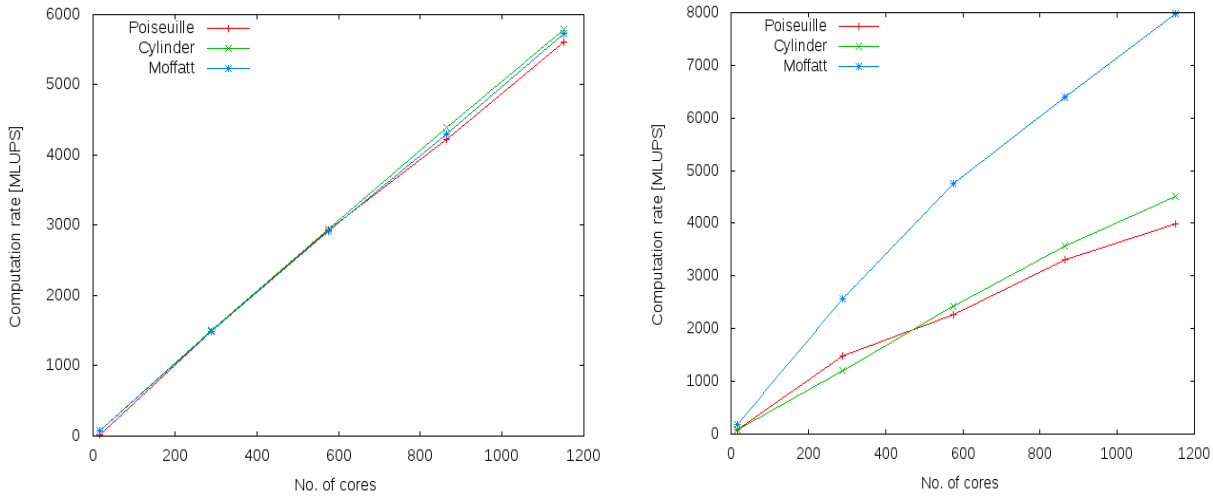

Figure 2: Comparison of LBM compute rates for conventional (left) and task-based (right) problem decompositions

The SPH application benefits from being adapted for task-based programming because an added level of problem partitioning reduces the number of fluid particles that must be compared to each other before calculating their interactions. This results in improved scalability irrespective of the input problem, but as shown in Figure 3, performance improvements remain strongly connected to a node architecture that gives high performance for classical SMP programs, and further algorithmic improvements will be necessary in order to adapt this application for massively parallel systems with lightweight cores, such as the anticipated first generation of Exascale systems.

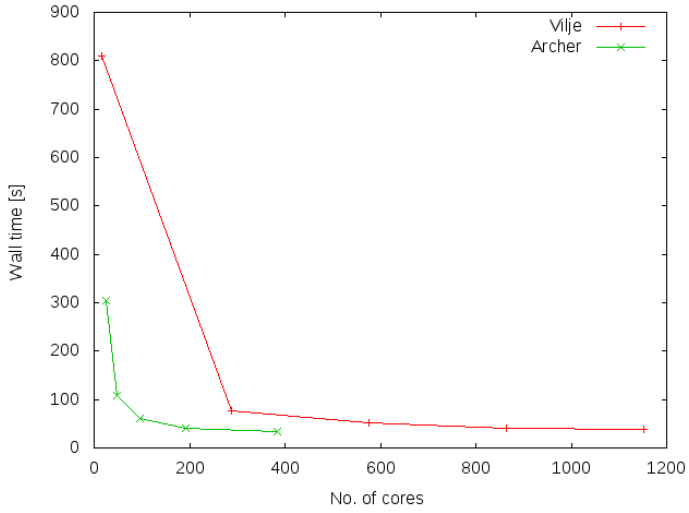

Figure 3: Comparison of SPH execution times on Altix ICE X (Vilje) and Cray XC30 (Archer)

3.2. Comparison of LBM and SPH Scalability using Task-based Programming Models

WP281: Architectural Scalability of Neural Network Inference Using Task-based Programming

Author: Jan Christian Meyer (NTNU)

FETHPC Project/Tool: INTERTWinE

Application: Kinetic Ising Model performance proxy application

Summary:

The kinetic Ising model from statistical physics presents a highly parallel method to analyze non-equilibrium systems, which is applicable to the problem of estimating the internal structure of hidden neural networks, based on limited experimental data records of its external behavior. Its utility is related to the size of network that can be represented and the time required to infer its structure, both of which invite highly parallel solutions. Previous studies have shown that successive generations of accelerator hardware provide improved performance with the evolution of their memory subsystems, but they have so far been inconclusive with respect to consistently providing better absolute performance than shared memory parallelizations on conventional architectures. This suggests that performance is linked to non-trival interactions with the memory hierarchy.

The objective of this project is to investigate the impact of combining task-based programming models with MPI communication, and applying them to a proxy application which implements this method, and identify combinations with favorable scalability properties. Task-based programming models aim to exploit the potential for hiding memory latency by exposing a far greater number of independent, parallel tasks than the number of physical processing cores, thus trading the performance overhead of creating an execution schedule for the possible savings of dynamically adapting it to run-time conditions.

The restrictions of the proxy application are that it generates the computational load by inferring the structure of an artificially constructed network of arbitrary size, for benchmarking purposes, and it completes execution before its analysis converges, so as to emulate the performance characteristics of extended runs without requiring large allocations of computation time. As the application quickly reaches a sustainable, steady computation rate, we consider these measurements to give realistic expectations of the performance that can be obtained by making similar modifications to a complete implementation.

Experimental results indicate that the application makes a good candidate for Exascale computations, exhibiting nearly linear speedup up to 1152 processing cores already at a moderate input problem size, shown in Figure 4. The curves represent different implementations of the proxy application that utilize thread parallelism differently: the MPI version has no thread parallelism, the MPI+Workshare version uses conventional OpenMP directives, and the MPI+Task version uses OpenMP task constructs.

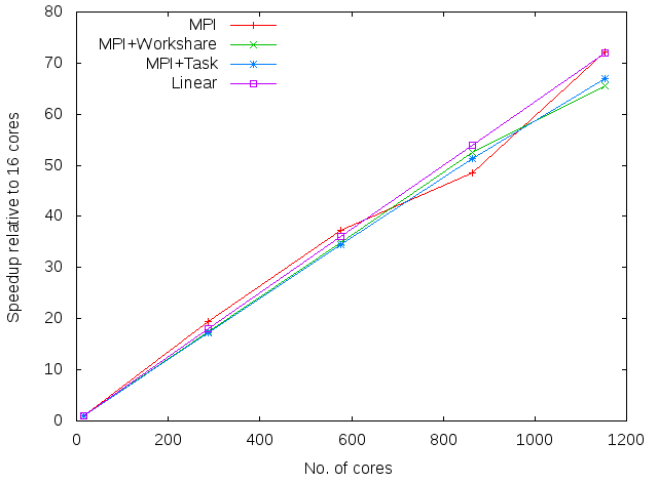

Figure 4: Proxy application scalability with MPI, MPI+Workshare and MPI+Task implementations

While these various parallelizations improve performance comparably with increasing system sizes, however, thread level parallelism incurs an overhead that leads to far longer total execution times, and this is exacerbated by the additional overhead of task scheduling, as shown in Figure 5.

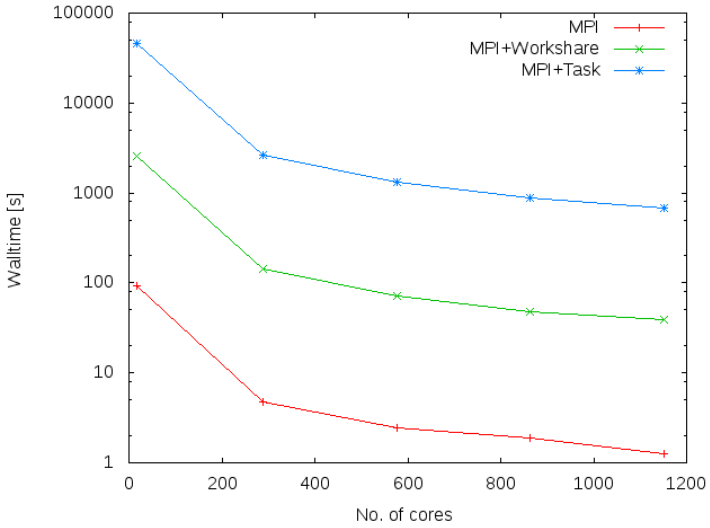

Figure 5: Comparison of absolute execution times with MPI, MPI+Workshare and MPI+Task implementations

The conclusion is that the application is a viable candidate for Exascale computations, but does not benefit from the latency-masking potential of task parallelism on present systems. This is likely to present the greatest architectural challenge to an efficient implementation on a massively parallel array of lightweight compute cores, which is anticipated for the first generation of Exascale systems.

3.3. Optimization of Computationally and I/O Intense Patterns in Electronic Structure and Machine Learning Algorithms

WP277: Optimization of Computationally and I/O Intense Patterns in Electronic Structure and Machine Learning Algorithms

Authors: Marian Gall, Michal Pitoňák (CC SAS), Adrián Rodríguez-Bazaga (University of Cambridge), Valeria Bartsch (Fraunhofer - ITWM)

FETHPC Project/Tool: INTERTWinE

Applications: K-means algorithm, TeraSort, Tensor Contractions, parallel matrix multiplication

Summary:

The HPC application software development for supercomputers with exaflop performance certainly requires several specific aspects to be addressed. Though we don't know yet, what hardware such system will be built with, we can assume by extrapolating current trends it will be a heterogeneous cluster of lots of nodes connected with a high-speed network, each node containing one or more accelerators. Due to anticipated convergence of supercomputers' utilization on HPC and Big Data tasks, we can expect the current I/O systems to change significantly. Local storage / drives on HPC compute nodes are rarely seen nowadays, and I/O is typically provided via a parallel file system with a certain number of I/O nodes connected to external disk array(s). This approach is often sufficient, particularly in real-life supercomputer operations, where only a minority of users running in parallel have I/O-demanding applications. If opposite was true, and a majority of running jobs would require high I/O loads, like in a typical Big data cluster, this approach would break down. Current HPC I/O systems simply cannot compete with the aggregate I/O performance of (once popular) locale drives setup. The applications for exascale architecture are thus expected to scale efficiently up to (tens of) thousands of nodes, be resilient to failure of (few) parallel processes, utilize accelerators, and take advantage of next generation I/O systems (such as non-volatile RAMs, etc.). More than that, not only some of the aforementioned aspects should to be addressed, but all of them must be perfected in order to utilize "theoretical" performance of the future exaflop machine.

Within this mini-project we restrict our attention to two of these aspects: efficient parallel scaling and run-time resilience. GASPI (Global Address Space Programming Interface), a Partitioned Global Address Space (PGAS) API, enables us to switch from traditional synchronous, two-sided MPI communication to one-sided, asynchronous and fault-tolerant execution utilizing RDMA (Remote Direct Memory Access). GASPI specifications are implemented in the open source C/C++/Fortran GPI-2 library, developed at Fraunhofer ITWM. GASPI interoperability with MPI (and other parallel frameworks) is one of the research subjects of our partner FET-HPC project, INTERTWinE.

Two classes of problems were selected for GASPI implementation. First, it is the tensor contraction, the most time-consuming part of the majority of quantum chemistry algorithms (perturbation theory, coupled-clusters, etc.), second, popular machine-learning algorithms, such as K-means and TeraSort.

Tensor contraction engine is the hearth of most modern, massively parallel scaling quantum chemistry program packages, such as NWChem, ACES IV, CFOUR, or available as a separate library, such as Libtensor. Our goal is to improve tensor contraction performance via overlap of computation and communication, and eventually use it to improve parallel scaling of coupled-cluster modules in (not exclusively) the OpenMolcas program package, readily available to CoEs.

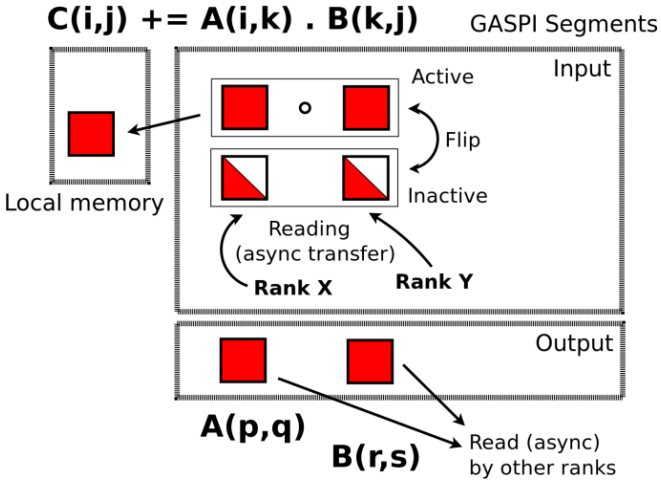

Figure 6: Schema of the GASPI matrix-matrix multiplication on an arbitrary rank (parallel process).

In Figure 6, a simplified schema of our C += A * B matrix multiplication implementation, or a tensor contraction, if indices are aligned properly, is depicted. Two sets of GASPI segments (*i.e.* continuous memory segment accessible by other ranks without active participation of the host rank) are used: "inactive", for asynchronous receiving of (A and B) matrix blocks from other ranks, and "active", for local matrix multiplication of received blocks. These two segments change their roles whenever possible, allowing thus for efficient computation and communication overlap. Each rank stores one (or more) block(s) of matrices A, B and C. Measured parallel performance of our GASPI approach is slightly better than that of MKL PDGEMM, moreover, without special requirements on memory partitioning of the matrices.

Our goal in the machine learning part of the mini-project is to demonstrate, that we can outperform popular JVM- (Java Virtual Machine) based big data frameworks (*e.g.* Apache Spark) by using HPC tools, while preserving their outstanding features, such as run-time resilience and use of distributed parallel file systems. To this end we chose two, rather different algorithms: K-means and TeraSort.

K-means is a popular, iterative, clustering algorithm, and is known to run efficiently on JVM frameworks, at least for a reasonable number of cluster centers, "centroids". It is particularly suitable to demonstrate fault tolerant implementations, which we managed with great help from GPI CP, GASPI checkpointing library. GASPI has (timeout-based) mechanism to detect rank failure, and one of ways to react is to start, for example a non-shrinking recovery process from data saved in a checkpoint and a spare node(s). GPI CP automates the process of distribution of checkpoint data to mirrors via asynchronous GASPI communication, thus introduces negligible overhead to standard implementations.

The amount of data that has to be exchanged between nodes is typically small (coordinates of centroids), unlike in the case of the TeraSort algorithm. In the TeraSort algorithm, out-of-core parallel sorting of terabytes of data, nodes partition the input data and send the resulting chunks to destination ranks for sorting. Large amounts of data are transferred through the network, but unlike in the K-means algorithm, only in a single step. In our GASPI K-means and TeraSort implementations we aimed for improved efficiency, even beyond that of MPI, while avoiding the use of blocking and collective operations. Presence of blocking calls in the parallel code eventually limits its scaling, and certainly is not a recommended design pattern when targeting Exascale.

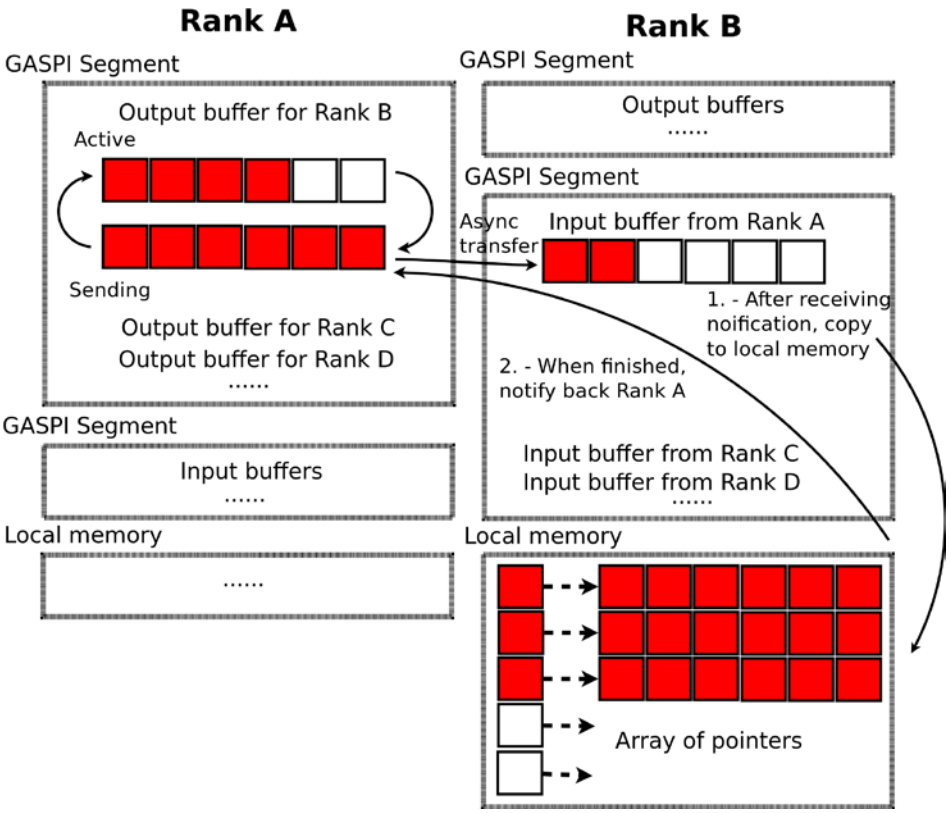

Figure 7: Asynchronous communication between two ranks in the GASPI TeraSort implementation.

The way we managed to overlap computation with communication in TeraSort algorithm is depicted in Figure 7. Rank "B" has a pair of GASPI segments dedicated to (asynchronous) communication with rank "A". Anytime the rank "A" fills up its local buffer with data addressed to rank "B", it can immediately initiate the data transfer without active participation of rank "B". Our GASPI implementation outperformed MPI by about 25% in weak scaling tests up to 128 ranks and 240 GB of sorted file size.

4. Novel Coding and I/O Techniques

In this section, we report on two projects that have each focused on exploiting state-of-the- art coding and I/O techniques to enable scalability and performance of applications for European future Exascale systems, as listed in Table 5. Each subsection provides a summary of the project along with a reference to the associated PRACE-5IP whitepaper. We refer the reader to the associated whitepaper for each project, which provides a more detailed report on the projects than is provided here.

FETHPC Projects/Tools, CoEs	Applications
E-CAM CoE	OpenPathSampling (OPS), OpenMM and/or GROMACS, Dask.jobqueue Python library
Energy Oriented Centre of Excellence (EoCoE)	WRF, WRFsolar

Table 5: Novel Coding and I/O Techniques exploited along with corresponding applications

4.1. Task Scheduling Library for Optimising Time-Scalar Molecular Dynamics Simulation

WP282: Task Scheduling Library for Optimising Time-Scalar Molecular Dynamics Simulation

Authors: Alan O'Cais (E-CAM CoE), David Swenson (E-CAM CoE), Mariusz Uchroński (WCNS), Adam Włodarczyk (WCNS)

Application: OpenPathSampling (OPS), OpenMM and/or GROMACS, Dask.jobqueue Python library

CoEs: E-CAM

Summary:

In the particular use case for the mini-project described here, E-CAM is interested in the challenge of bridging timescales. To study molecular dynamics with atomistic detail, timesteps must be used on the order of a femto-second. Many problems in biological chemistry, materials science, and other fields involve events that only spontaneously occur after a millisecond or longer. That means that around 10^12 time steps would be needed to see a single millisecond-scale event. This is the problem of "rare events" in theoretical and computational chemistry. To fully characterize a transition with proper statistics, many examples are needed. In order to obtain many examples the same application must be run many thousands of times with varying inputs. To manage this kind of computation task scheduling library is needed. The main elements of mentioned scheduling library are: task definition, task scheduling (handled in Python) and task execution (facilitated by the MPI layer). While traditionally an HTC workload is looked down upon in the HPC space, the scientific use case for extreme-scale resources exists and algorithms that require a coordinated approach make efficient libraries that implement this approach increasingly important in the HPC space. The 5 Petaflop booster technology of JURECA is an interesting concept with respect to this approach since the offloading approach of heavy computation marries perfectly to the concept outlined here.

The provided library is written in Python programming language as an additional feature for *Dask.jobqueue* which in turn utilizes the *Dask* library. The former is targeted for deploying the latter on several job queuing systems, such as Slurm or PBS with the use of a Python programming interface. Another feature provided by subject library is a set of Python decorators for the library. Such an approach separates the inner implementation of batch scripts creation and communication with the Slurm system from the user side code declaration. Main decorators are *on_cluster* and *task*. In many scenarios inserting those two decorators in user's code is enough for the transition from plain Python code into a distributed computation. The library also supports MPI-based computation to distribute tasks over nodes of a cluster. This is provided by the *mpi_task* decorator.

During computational experiments we directly measure the overhead of the library and used data from the resource manager to account for the system overhead. We visualized results as the total overhead of the framework in seconds, in Figure 8. There are 2 nodes per worker (68 cores on a KNL node, 4 GPUs on a GPU node and 24 physical cores on a CPU node), with 10 workers each (so, at peak utilisation, a total of 480 CPUs, 1360 KNL cores and 80 GPUs). The total overhead is relatively static over time until we get to larger task counts. The sudden increase at 2000 tasks for the KNL data is mostly due to the fact that the simulation took long enough that our workers began to exceed their allowed walltimes, resulting in the resilience mechanism being utilised and additonal workers being started. The overall message is that overhead is very small. At higher task counts we are seeing almost 90% throughput effciency

for trivial tasks. If the tasks executed for any reasonable length of time, then this throughout would be much higher.

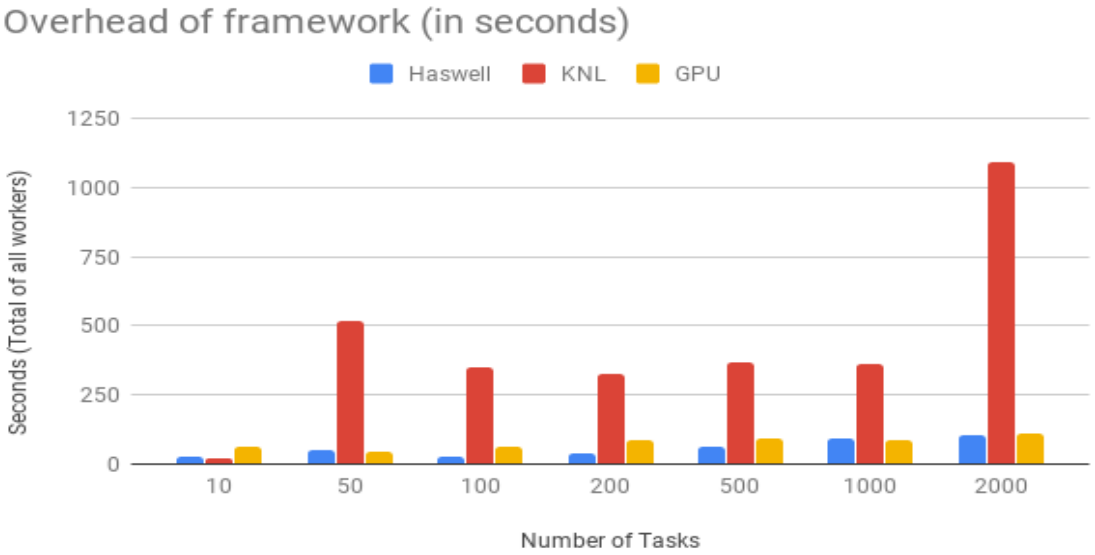

Figure 8. Total overhead of the framework in seconds for various numbers of tasks and on various architectures.

Time savings (in seconds) of running tasks through the library rather than through the resource manager can be seen in Table 6. A final remark is that these savings scale according to the resources used by the task (i.e, the savings scale linearly with respect to the resources used by the task).

Tasks	Haswell	KNL	GPU
10	33.19	-93.61	-48.8
50	357.99	441.55	297.58
100	871.84	1759.37	657.45
500	4836.35	11321.85	3827.79
1000	9796.77	23327.39	7824.52
2000	19784.67	46548.92	15803.22

Table 6. Time savings (in seconds) of running tasks through the library rather than through the resource manager

The resulting library is flexible, scalable, efficient and adaptive. It is capable of simultaneously utilizing CPUs, KNL and GPUs and dynamically adjusting its use of these resources based on the resource requirements of the scheduled task workload. The ultimate scalability and hardware capabilities of the solution is dictated by the scalability characteristics of the tasks themselves (if there are 100 nodes per task with GPUs, then the library can carry out N of these at once depending on resource availability).

4.2. Approaching Exascale with the Weather Research and Forecasting Solar Model

WP276: Approaching Exascale with the Weather Research and Forecasting Solar Model

Authors: Jacob Finkenrath (CaSToRC), Giannis Koutsou (CaSToRC), Swen Metzger (CaSToRC), Hendrik Elbern (Juelich), Jonas Berndt (Juelich)

Application: WRF, WRFsolar

CoEs: Energy Oriented Centre of Excellence (EoCoE)

Summary:

This project analysed the readiness of the Weather Research and Forecast model (WRF) for exascale computing together with collaborators from the Energy Oriented Centre of Excellence (EoCoE). The objective was to investigate the scalability of WRF (solar) on PRACE Tier-0 systems with different architectures. This includes an identification of functions with high work-loads and analysis of I/O and data management in order to understand bottlenecks of applications of WRF on exascale machine. Moreover, the miniproject compared different HPC-platforms and analyze different parallelization strategies.

The target was to identify bottlenecks in WRF employing PRACE Tier-0 machines, including SuperMUC at LRZ, Marenostrum 4 at BSC and Marconi Phase 2 with KNLs at CINECA. We used tools like Scalasca/Score-P or extrae, to measure the work-load of the WRF functions, to analyze I/O to storage and to test performance over thousands of cores. Moreover, the different architectures available to us allowed for comparisons. We identified possible bottlenecks in the application by comparing the performance of WRF for different architectures, like KNL-based and CPU based systems of different generations. This enabled us to make predictions on the capability of WRF for possible future Exascale systems. We investigated one model, refered to as model 2, which is based on the solar branch of WRF. WRFsolar is an augmentation of WRF in terms of improved irradiance forecasts for solar power predictions. Moreover, we defined other test-cases, like a case used for standard weather forecast (model 1) and investigated some additional WRF like WRF-fire (model 3).

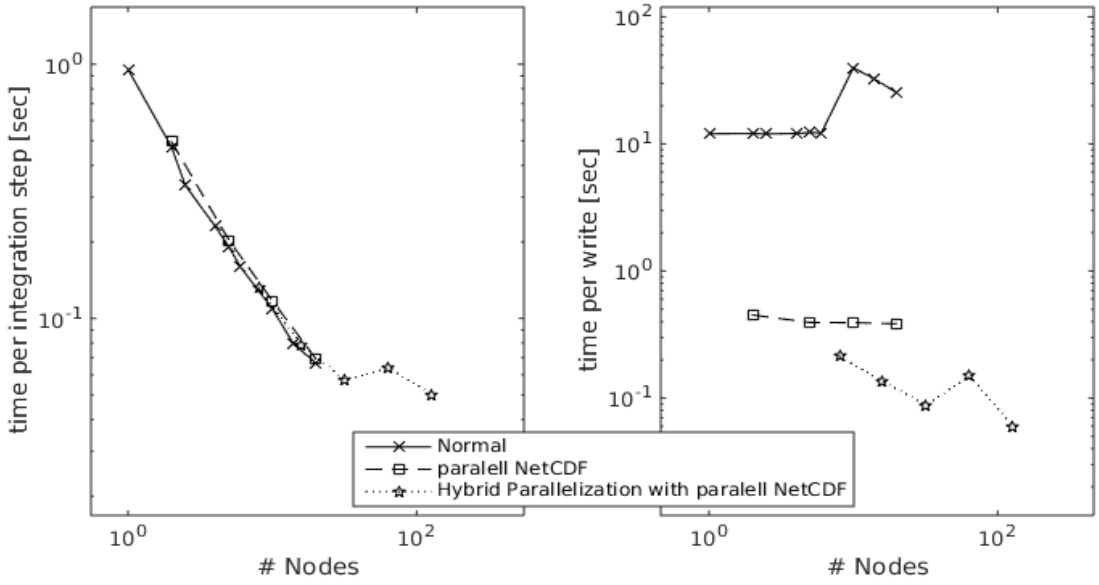

Figure 9: Strong scaling of WRF4.0 on MareNostrum

Analysis of the I/O: By using strong scaling we measured the precentage of I/O on the total run time. The timing for I/O becomes a significant part of the total run time of upt to 80% for parallelization larger than 500 MPI task. In the default setup of WRF 4 the I/O was done by using NetCDF by the MPI task with rank 0. The I/O procedure use *MPI_Gather* which hinders the scalability and increases the run time in case of larger parallelization significantly.

A solution to this problem may be to use parallel I/O. In WRF 4.0, this can be enabled by using parallel NetCDF. We studied the behavior on Marenostrum using strong scaling employing model 1 with a volume of V=384x384x50 for native MPI and hybrid parallelization. The scaling is shown in Figure 9. The left panel of Figure 9 shows a comparison of the time to integrate in three scenarios: i) using NetCDF with MPI, ii) using parallel NetCDF with MPI, and iii) using parallel NetCDF with hybrid parallelization. Using parallel NetCDF reduces the time for I/O by an order of magnitude. However, in this case, a reduction of the time by increasing the number of MPI tasks is not observed. However, we found that the time of I/O can be further decreased by using a hybrid parallelization. As shown in the right panel of Figure 9, the time for I/O reduces for increasing number of nodes. This shows that approaching massive parallelization, WRF with parallel NetCDF is the most promising setup. To conclude, if for model 1 a local size of around 23,000 grid points per node is required, WRF shows good scaling on large partitions that would be available on pre-exascale systems, provided the global problem size is sufficiently large.

Outlook and conclusion: WRF and exascale: In investigating WRF, we found that the computation kernel scales well down to local domain sizes of as small as 700 grid points for the cases of models 1 and 2, and down to 9,600 grid points for the case of model 3. This means that for different models, the scalability of WRF can alter significantly such that a careful choice of the parallelization is mandatory in order to run effectively a given model using large partitions. We found that the time for I/O scales poorly compared to the computational kernels and constitutes the major bottleneck when scaling on PRACE Tier-0 systems. Moreover, when weak scaling, the time for I/O increases proportionally to the global grid size, while the computational kernels scales almost perfectly.

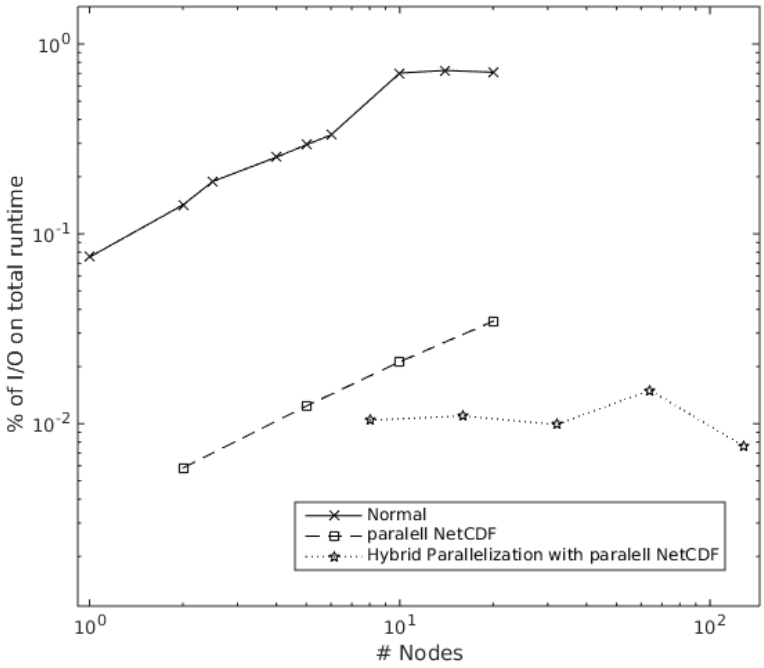

Figure 10: I/O time from using parallel NetCDF

However, using parallel NetCDF can mitigate the time for I/O as shown in Figure 10. In the employed test cases we found that the time can be decreased by almost one magnitude. Furthermore, we found that on nodes equipped with Skylake chips by using a mixed parallelization approach the time for the I/O can be further reduced by not effecting the time to solution of the computational kernel. We investigated the node performance of the computational kernel by running similar test cases on the Haswell partition of SuperMUC and the Skylake partition of Marenostrum 4. The comparison shows that model 1 performs better on Skylake CPUs by around a factor of 1.7 and is optimized for AVX512 instructions. However, for the case of model 2, the comparison between Haswell and Skylake yields similar time-to-solution. In general, this means that the performance of WRF depends strongly on the use cases and the specific computational kernel.

This analysis could be used to propose some strategies in order to efficiently use pre-exascale systems. In general, the efficiency of the computational kernel of WRF is maintained as long as the local volume is not smaller than a threshold size. The threshold depends on the computational kernel, e.g. for models 1 and 2 this is found to be V~700 while for model 3 this is V~9,600. These thresholds should be taken into account, as well as the global physical box size, the resolution, and whether ensemble simulations, which can run in parallel independently, are required.

Another conclusion is that on Skylake architectures, hybrid parallelization should be preferred, i.e.~on Marenostrum 4 the run with 8 MPI tasks with 3 OpenMP tasks per MPI task was found to be more efficient compared to using only MPI. Hybrid parallelization enables better I/O performance and scalability on a larger number of nodes. Linking to a parallel I/O library is also mandatory when using a PRACE Tier-0 system. As pointed out, this reduces the total time for I/O by more than an order of magnitude.

5. Energy Efficiency

In this section, we report on one project that has focused on exploiting a state-of-the-art HPC auto-tuning tool in order to enable energy-efficiency of HPC applications on European multi-petaflop/future Exascale systems, as summarised in Table 7. A summary of the project is provided along with a reference to the associated PRACE-5IP whitepaper. We refer the reader to the associated whitepaper, which provides a more detailed report on the projects than is provided here.

FETHPC Projects/Tools, CoEs	Application
READEX, EoCoE	Alya

Table 7: Scalable Libraries and Algorithms exploited along with corresponding applications

5.1. Investivating Application Dynamism in Alya with READEX for Energy Efficiency

WP278: Investigating Application Dynamism in Alya with READEX for Energy Efficiency

Authors: Venkatesh Kannan (ICHEC), Myles Doyle (ICHEC), Guillaume Houzeaux (BSC), Ricard Borrell (BSC)

FETHPC Project/Tool: READEX

Application: Alya

CoE: EoCoE

Summary:

High performance computing (HPC) is a major driving force for European research and innovation in many scientific and industrial domains. The applications in these areas are highly complex, and demand high performance and efficient execution. As a result of this growing need for computational performance, the energy consumption of the HPC systems has continued to increase. This is a cause for concern because the energy requirements of near-future European exascale systems will be a major factor of the total cost of owning the HPC system. Therefore, it is crucial to improve the energy-efficiency of the applications that run on these systems. A significant source of improvement for applications is that they commonly exhibit dynamic resource requirements. This may stem from different regions in the application that are executed or changes in the workload at runtime. Consequently, such dynamism in an application presents opportunity to tailor the utilisation of resources in the HPC system based on the requirements of the application at runtime.

The objective of this project is to use the READEX tool suite (created in the H2020 FETHPC READEX project) to investigate and exploit the dynamism exhibited by the Alya application (created by the EoCoE at BSC).

The READEX (Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing) project leverages this approach to deliver improved performance and energy-efficiency when executing applications on current and future extreme-scale systems. The goal of the READEX project is to create a tool suite that

- Identifies existing dynamism in an application to determine its tuning potential;
- Determines the configurations for different hardware-, system software- and application-level tuning parameters that suit different scenarios that may arise during the application's execution;
- Applies the best configuration for the tuning parameters during the application's runtime.

Alya is a high performance computational mechanics application that is present in the Unified European Application Benchmark Suite and the PRACE Accelerator Benchmark Suite. It is hybrid parallelized with MPI, OpenMP, OpenACC and CUDA, with dynamic workload balancing.

In this project, the application dynamism potentially present in Alya is investigated and exploited by READEX using the Taurus cluster at TU Dresden as the evaluation platform.

We have identified the existing dynamism in Alya and performed experiments, aided by the READEX tool suite, to identify optimal values for application parameters in addition to the settings for processor core/uncore frequencies and number of OpenMP threads. Our evaluations show that energy consumption and execution time savings are achievable between 5-25% on upto 40 node (960 core) runs. These results are indicative of a promising line of action to better understand the reasons for the dynamisms in Alya and pursue further investigations to tune application paramters for different scales of Alya's production runs. These results are illustrated in Figure 11 and Figure 12.

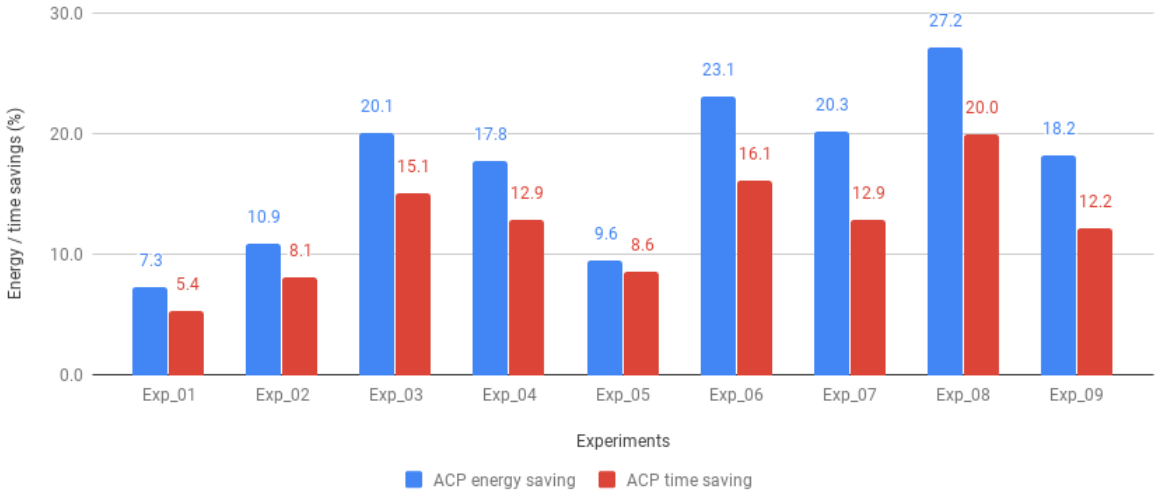

Figure 11: Energy and time savings for Alya with READEX for tuning application parameters

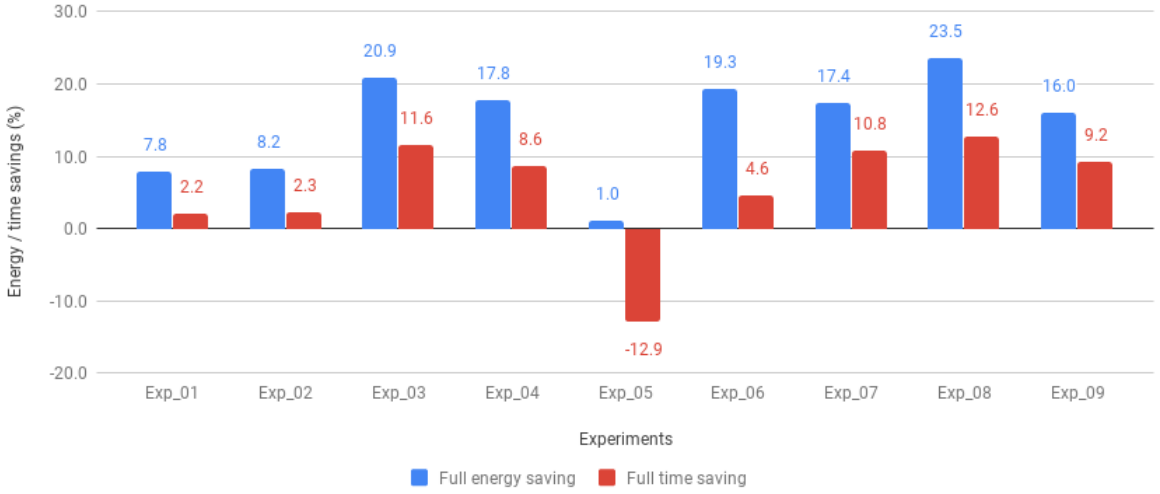

Figure 12: Energy and time savings for Alya with READEX for tuning processor frequencies, OpenMP thread count and application parameters

6. Summary

This deliverable reports on the eight projects undertaken as part of PRACE-5IP Task 7.2 “Preparing for PRACE Exascale Systems”, where the projects are aligned with four lines of actions that we consider relevant to enable applications on current multi-petascale and future exascale systems, and where inspiration has been taken from the comprehensive survey of CoE HPC requirements and state-of-the-art HPC tools and techniques developed in FET-HPC projects that were investigated in this task as well as PRACE 4IP Task 7.2a as reported in D7.4 [1] and in PRACE 3IP WP7 as reported in D7.2.1 [2].

In this section, we summarise our findings for each line of action. In-depth conclusions for each of the projects reported on in this deliverable can be found in the associated whitepapers and so here we list only what we think are the most salient findings made during the exploitation phase.

1. **Novel Algorithms**
   We have reported on how the scalability of new iterative solvers proposed in the FET-HPC project NLAFET has been improved. In particular, we find that the encapsulation of computational load balance during sparse and dense matrix computations simultaneously leads to better scalability, and that 2D-partitioning models perform better in general for smaller block sizes and large number of processes. In another work carried out under this line of action, we have analysed the implementation of PermonSVM, which is a Support Vector Machine (SVM) implementation of the PERMON application suite. This has resulted in improving PermonSVM through reduced storage and memory requirements, and reduction on data conversion overhead when loading data objects for training with the SVM.
2. **Novel Programming Models**
   As highlighted in the survey of CoE HPC requirements in PRACE-4IP D7.3 [5], and from the CoE-FETHPC workshop during the survey phase of this task described in Section 1.3, one programming model still dominates CoE applications more than any other, namely, Single Program Multiple Data (SPMD) message passing using MPI for internode communication. However, we feel that the clear lack of exploitation of MPI 3.0 and other programming models such as GASPI (an implementation of the PGAS API) within the CoEs. To address this, we have explored GASPI, which enables to switch from traditional synchronous, two-sided MPI communication to one-sided, asynchronous and fault-tolerant execution using RDMA (Remote Data Memory Access). This is achieved by leveraging the results of the FET-HPC project INTERTWine, in which one of the main research subjects was the interoperability between GASPI and MPI (and other parallel frameworks). This is in line with some of the recommendations in PRACE-4IP D7.4 [1]. Additionally, we have also focussed on the use of task-based programming models to build scalable parallel libraries for applications related to computational fluid dynamics to be able to target scalability on Exascale systems.
3. **Novel Coding and I/O Techniques**
   We reported on the development of a flexible, scalable, efficient and adaptive task scheduling library to enable parallel execution on heterogenous platforms in future Exascale systems. The capabilities of the scheduling library were tested and demonstrated by optimising time-scalar molecular dynamics simulation code that is of particular interest to E-CAM CoE. The library is capable of simultaneously utilising CPUs, Intel KNL and GPUs, and dynamically adjusts the scheduling based on the resource requirements of the scheduled task workload. Futhermore, we have investigated the scalability and readiness of the WRF and WRFSolar models on PRACE Tier-0 systems with different architetures. This includes identification of functions with high workloads and, an analysis of I/O and data management to understand the potential bottlenecks on Exascale systems. These results could be used to propose strategies in order to effectively use current pre-exascale and future Exascale systems.
4. **Energy Efficiency**
   We have reported the exploitation of a tool suite, known as READEX, that was developed by the H2020 FET-HPC project (also named READEX). The work applies the tool suite on a widely-used multi-physics simulation application called Alya, which is a part of the EoCoE, to identify existing dynamism in the application. This is then used to optimise parameters in all layers of the HPC stack to improve its energy efficiency during its execution on HPC systems. Several other CoEs are also keenly focused on improving the energy efficiency of their applications running on extreme-scale systems, including MaX, ESiWACE and POP, and we expect our investigations to also be of interest to the wider community, particularly in the context of the PRACE PCP and future European extreme-scale systems.

In line with the recommendations from PRACE-4IP and PRACE-5IP, we have made the utmost effort to focus on tools, techniques and applications that are of direct relevance to the CoEs by leveraging FET-HPC project results. We feel that all of the tools and techniques we have investigated during the exploitation phase should be of interest to the vast majority of the CoEs and the European HPC community more widely. Furthermore, the requirements of the CoEs' are expected to change over their lifetime and as such communication between PRACE, CoEs and FET-HPC projects should be strengthened, as we face the Exascale frontier together. Finally, we should re-emphasise that, as we look towards PRACE-6IP and beyond, similar tasks will continue to be informed and inspired by the ongoing research across the various European and US exascale projects.