Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 240 (2024) iii
Procedia Computer Science
www.elsevier.com/locate/procedia

Table of Contents

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 1–2

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Preface

The EuroHPC User Day 2023 was held on 11 December 2023 at the Charlemagne Building in Brussels, Belgium. The EuroHPC User Day was organised by the European High Performance Computing Joint Undertaking (EuroHPC JU) with support from the European Commission.

The EuroHPC JU is an EU body, a legal and funding entity, with the mission to lead the way in European supercomputing. EuroHPC JU allows the European Union and 35 participating countries to coordinate their efforts and pool their resources to make Europe a world leader in supercomputing. This collaborative approach boosts Europe's scientific excellence and industrial strength, support the digital transformation of its economy and strengthen its technological sovereignty. The key priorities in the EuroHPC JU's mission include deploying a world-leading European High Performance Computing (HPC) infrastructure and funding an ambitious research and innovation programme to develop a full European supercomputing supply chain, competence building and usage of EuroHPC systems. This involves widening the usage of HPC to a large number of public and private users across Europe, supporting the development and maintenance of key HPC skills among users, and taking measures to establish a strong HPC user community in Europe.

This event, the first ever EuroHPC User Day, is one such step towards building this EuroHPC user community. The purpose of this annual conference is to provide an opportunity for users to showcase projects that are running on the EuroHPC supercomputers. This event allows project leaders and users to communicate on projects which have been granted computing time on EuroHPC systems, share their methods and lessons learned, and disseminate their scientific results and findings.

Over 140 participants from the European HPC user community had the opportunity to meet and network, as well as provide feedback on the resources and access possibilities available through the EuroHPC JU.
Over 50 speakers presented projects, spanning diverse fields such as computer science, computational physics, universe science, climate modelling, artificial intelligence, engineering, computational chemistry and systems biology. Plenary sessions took place in the morning to present the EuroHPC JU, its access calls and HPC infrastructure and the upcoming quantum infrastructure. The morning sessions were livestreamed, allowing for online attendees to follow along. During the afternoon, parallel tracks were organised, allowing for specialised and technical presentations to take place, in a smaller, offline forum.
The EuroHPC JU is proud to see the range and quality of projects receiving access time on its systems, and the excellent science which has resulted from this access. The manuscripts which follow in this volume are the first of

10.1016/j.procs.2024.07.002

many scientific findings enabled by compute time on EuroHPC machines, and we look forward to publishing many more such results in upcoming editions.
The review selection of the received manuscripts was performed by an editorial board of scientific reviewers. For this first edition of the EuroHPC User Day proceedings, 11 papers have been selected for publication.
Special thanks go to EuroHPC users, as well as to the EuroHPC Hosting Entities, whose collaborative efforts contributed to the success of the first EuroHPC User Day. The EuroHPC JU also wishes to thank the European Commission for the use of its premises and its support in the logistical organisation of the event. This event was the reflection of the vibrant and collaborative users' community which surrounds the EuroHPC JU, and is the start of an exciting journey to build up a close-knit and highly-skilled HPC user community in Europe.

Anders Dam Jensen – Executive Director of the EuroHPC JU
Lilit Axner – Chair of the EuroHPC User Day Programme Committee
& Programme Officer Infrastructure at the EuroHPC JU

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 3–12

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Cross-Facility Federated Learning

Iacopo Colonnelli[a,*], Robert Birke[a], Giulio Malenza[a], Gianluca Mittone[a], Alberto Mulone[a], Jeroen Galjaard[b], Lydia Y. Chen[b], Sanzio Bassini[c], Gabriella Scipione[c], Jan Martinovič[d], Vit Vondrák[d], Marco Aldinucci[a]

[a]*University of Torino, Computer Science Dept., Corso Svizzera 185, 10149, Torino, Italy*
[b]*Delft University of Technology, Computer Science Dept., Van Mourik Broekmanweg 6, 2628 XE Delft, Netherlands*
[c]*CINECA National Supercomputing Center, Casalecchio di Reno, Bologna, Italy*
[d]*IT4Innovations, VSB – Technical University of Ostrava, Ostrava, Czech Republic*

Abstract

In a decade, AI frontier research transitioned from the researcher's workstation to thousands of high-end hardware-accelerated compute nodes. This rapid evolution shows no signs of slowing down in the foreseeable future. While top cloud providers may be able to keep pace with this growth rate, obtaining and efficiently exploiting computing resources at that scale is a daunting challenge for universities and SMEs. This work introduces the Cross-Facility Federated Learning (XFFL) framework to bridge this compute divide, extending the opportunity to efficiently exploit multiple independent data centres for extreme-scale deep learning tasks to data scientists and domain experts. XFFL relies on hybrid workflow abstractions to decouple tasks from environment-specific technicalities, reducing complexity and enhancing reusability. In addition, Federated Learning (FL) algorithms eliminate the need to move large amounts of data between different facilities, reducing time-to-solution and preserving data privacy. The XFFL approach is empirically evaluated by training a full LLaMAv2 7B instance on two facilities of the EuroHPC JU, showing how the increased computing power completely compensates for the additional overhead introduced by two data centres.

Keywords: Federated Learning; High-Performance Computing; Cross-Facility Computing; Hybrid Workflows; StreamFlow.

1. Introduction

The Big Data era is propelling machine learning experiments to unprecedented scales, outpacing even Moore's law [1]. In a decade, AI frontier research has transitioned from the researcher's workstation (AlexNet [2], 2012, trained for six days on two NVIDIA GTX580 3GB) to thousands of high-end specialised chips (PaLM [3], 2022, trained for 50 days on 6144 TPU v4 chips). The advent of foundation models [4] and physics-informed neural networks [5] has

* Corresponding author.
E-mail address: iacopo.colonnelli@unito.it

catapulted Deep Neural Networks (DNNs) to trillions of parameters, requiring an entire exascale data centre to be trained from scratch [6]. This rapid evolution shows no signs of slowing down in the foreseeable future.

While top cloud providers may be able to keep pace with this growth rate [7], for universities and SMEs, the task of obtaining and efficiently exploiting computing resources at that scale is a daunting challenge. Indeed, in the last few years, this dichotomy of industry and academia has caused a reduced representation of academic-only research teams in computing-intensive research topics, especially foundation models [8].

In this scenario, HPC federations such as the one currently promoted by the EuroHPC Joint Undertaking play a crucial role. The massive amount of computing power at their disposal can compete with top-notch private data centres, potentially bridging the gap between industry and academia. However, granting researchers access to public facilities and enabling them to efficiently exploit their computing power efficiently is not trivial. Allocating an entire data centre exclusively to a single experiment for an extended period violates the principle of fair resource usage. On the other hand, efficiently exploiting multiple data centres without high-level techniques and tools to design and orchestrate cross-facility workloads is almost exclusively the prerequisite of senior computer scientists with a deep knowledge of high-performance distributed systems.

This work introduces the Cross-Facility Federated Learning (XFFL) framework to extend the opportunity to efficiently exploit multiple independent data centres for extreme-scale DNN training to data scientists and domain experts. In particular, XFFL combines two different approaches. On the one hand, hybrid workflow abstractions allow users to quickly design and orchestrate cross-facility workloads, decoupling tasks from environment-specific technical details to reduce complexity and increase reusability. On the other hand, Federated Learning (FL) algorithms eliminate the need to move large amounts of data between different facilities, reducing time-to-solution and preserving data privacy. To the best of the authors' knowledge, this work describes the first attempt to run a large-scale XFFL experiment.

The benefits brought by the XFFL approach are evaluated on a realistic scenario, i.e., training the full LLaMAv2 7B Large Language Model (LLM) [9] on top of two facilities of the EuroHPC JU, the Leonardo@CINECA and the Karolina@IT4I clusters, which occupy the 6th and 113th positions in the November 2023 Top500 ranking [10], respectively. In particular, the evaluation aims to assess whether the speedup obtained from the increased computing power is enough to compensate for the additional overhead introduced by using two distinct facilities, e.g., the model transfer time across the Internet or the misaligned execution times due to job queues. The results are promising, paving the way for further experiments with larger models and wider federations of data centres.

This article is organised as follows. Sec 2 introduces the preliminary concepts and analyses the related work. Sec. 3 introduces the proposed XFFL methodology. Sec. 4 describes the experimental environment and discusses the obtained results. Finally, Sec. 5 concludes the article and outlines future research directions.

2. Background and related work

2.1. *Hybrid workflows*

A hybrid workflow allows its steps to span multiple, heterogeneous, and independent computing infrastructures [11]. Each of these aspects has significant implications. Support for multiple infrastructures implies that each step can target a different execution environment. Such environments can be heterogeneous, with different and incompatible protocols for authentication, communication, resource allocation and job execution. They can also be independent of each other without the possibility of establishing direct connections to transfer data. A suitable model for hybrid workflows must then be composite, enclosing a specification of the workflow dependencies, a topology of the involved locations, and a mapping relation between steps and locations.

Hybrid Workflow Management Systems (WMSs) [12, 13, 14, 15, 16] are the software tools in charge of orchestrating hybrid workflow executions on top of complex, large-scale infrastructures, such as cross-facility and cloud-HPC environments. All the complexities of heterogeneous systems coordination, job scheduling, and data transfers are hidden behind a high-level workflow abstraction, which decouples the business code from the coordination layer. Plus, hybrid WMSs provide advanced features out of the box, such as caching, provenance tracking, and fault tolerance. Hybrid WMSs have already proved their superior flexibility and performance in diverse scientific domains: bioinformatics [17, 18], astrophysics [19], environmental science [20], and many more.

This work relies on the StreamFlow WMS [13] to orchestrate a large-scale XFFL workload, building on the initial work described in [21]. StreamFlow is a container-native hybrid WMS based on the Common Workflow Language (CWL) open standard [22]. It has been designed around two primary principles. First, it allows executing tasks requiring complex combinations of software container environments, supporting modern micro-services applications. Second, it relaxes the requirement of a single shared data space, allowing hybrid workflow executions on top of multi-cloud, multi-HPC, and mixed cloud-HPC infrastructures. Since this work requires the orchestration of a fully containerised FL workload on top of two independent HPC facilities, it represents a perfect use case for StreamFlow.

2.2. *Federated Learning*

Federated learning (FL) is a collaborative, distributed approach enabling the solving of a common Machine Learning (ML) problem shared between multiple institutions [23]. The core idea behind FL is sharing locally trained models instead of the local data itself. This simple yet powerful idea allows for overcoming the ML de facto standard data lake approach, in which data is harvested across different sources and accumulated by a single institution. Such a gathering process involves agreement efforts between the data-holding institutions while exposing the data to privacy attacks and misuse. FL offers a natively privacy-preserving approach, allowing multiple parties to train a DNN on their local data collaboratively and merge them into a global, shared DNN, retaining the knowledge extracted from all peers.

FL deployments involve a set of data-holding clients coordinated by a central server, even if other structures are possible [24]. The training process starts with the server communicating an initial set of parameters for the shared DNN model to each client. Each client then trains the received DNN on the local data and returns the obtained parameters to the central server. Once all (or a sufficient number) of local models are accumulated, the central server aggregates them, obtaining a global model. Such a model is then broadcast back to the clients, and the process continues iteratively until convergence. The presented process is just a general design of the FL approach, excluding many details such as which aggregation strategy is used [25, 26], how to make the execution more asynchronous [27, 28], how to speed up the client-server model communication [29, 30, 31], and more [32]. FL has been successfully applied to a variety of models ranging from deep, e.g. CNNs [23] and transformers [33], to classic [34] ML models.

FL can also be seen as a device that further increases the scalability of a distributed ML system. FL allows multiple computing infrastructures to train copies of the same model on different data partitions simultaneously, similar to a distributed data-parallel approach. Nevertheless, differently from other data-parallel strategies [35, 36], data are never shuffled among worker nodes, heavily reducing the communication overhead. Note that FL is not computationally equivalent to a centralised data lake scenario. The federation's number of clients, local and global data distribution, aggregation strategy, and many other hyperparameters heavily influence the final global model performance [37]. FL thus offers a novel trade-off between speeding up ML training and the obtained model performance; however, commercial FL frameworks [38, 39, 40, 41] do not seem to consider this aspect, preferring to focus on its privacy-preserving features rather than supporting complex, large-scale deployments.

FL of billion- and trillion-parameter foundation models is a promising approach to improve time-to-solution [42]. Indeed, FL reduces per-site memory occupancy and computing time and minimises inter-site communication compared to standard distributed training [43]. There are already few attempts to fine-tune foundation models in scientific literature [44, 45]. However, to the best of the authors' knowledge, this work describes the first attempt to train a full LLM on multiple HPC facilities through FL.

3. Cross-facility Federated Learning

By enabling researchers to harness the joint computing power of multiple clusters, we can mitigate the compute divide mentioned in Sec. 1. If a large, complex workload can be split into multiple independent sub-computations, they can run concurrently on different facilities. The partial results obtained on each machine can then be combined, leading to the final outcome of the global computation. In the case of iterative processes, these global results can be returned to the clusters, initiating a new iteration.

However, it is crucial to acknowledge that not all computations are amenable to this approach. Cluster-to-cluster communications are orders of magnitude slower than intra-cluster, node-to-node communications. Moreover, they are subject to varying bandwidths and network speeds typical of the public Internet network, making the process slow and error-prone. For these reasons, problems that necessitate frequent and large data transfers between clusters are not ideal candidates for cross-facility executions.

To ensure more equitable resource usage, lengthy iterative processes can be divided into multiple job submissions, each handling one or more iterations. This strategy also allows for the delegation of cross-facility orchestration to a third party, such as the control plane of a hybrid WMS. This way, a group of HPC centres can be treated as a single, geographically distributed supercluster, with all the technical aspects managed by the WMS runtime. However, it is crucial to consider queue waiting times and their variability whenever a new set of tasks is submitted to different HPC centres. Queue time is generally unpredictable and can range from minutes to hours, depending on the current cluster utilisation and the requested resources. Therefore, the computing time should be sufficiently long to offset the potential delays introduced by waiting times.

For ease of understanding, consider the well-known Bulk Synchronous Programming (BSP) model [46]. A BSP computation is structured into a sequence of supersteps, each consisting of three phases: a concurrent computation phase where each component performs local computations; a communication phase where components exchange data; and a global synchronisation phase where each component waits until all other components have reached the same barrier. When submitting a BSP computation consisting of S supersteps to solve a problem of size n as a single job to an HPC cluster with p nodes, the total time T_1 can be calculated as:

$$T_1 = q + \sum_{s=1}^{S} [w_s(n,p) + g \cdot h_s(n,p)] = q + \sum_{s=1}^{S} T_s(n,p) \tag{1}$$

where q is the waiting time, w is the computing time on p nodes, h is the amount of data to communicate among p nodes, and g is the speed of the interconnection network. Note that we omitted the synchronisation overhead, as it does not change significantly in the different scenarios discussed below. Instead, if we submit each superstep as a separate job to the HPC workload manager, each superstep is subject to queueing time, increasing the total time T_2 to:

$$T_2 = \sum_{s=1}^{S} [q_s + w_s(n,p) + g \cdot h_s(n,p)] = \sum_{s=1}^{S} q_s + \sum_{s=1}^{S} T_s(n,p) > T_1 \tag{2}$$

However, note that if the queue waiting time is much shorter than the execution time, i.e., $q_s \ll T_s(n,p)$, then $T_2 \simeq \sum_{s=1}^{S} T_s(n,p) \simeq T_1$. If now we consider F facilities having the same networking technology (i.e., $g_i = g$, for any $i \in F$) but different sizes (i.e., $p_i \neq p_j$ for any $i \neq j \in F$) totalling $\sum_{i=1}^{F} p_i$ processors, the total time T_3 can be calculated as:

$$\begin{aligned} T_3 &= \sum_{s=1}^{S} \left\{ \max_{f \in F} \left[q_{s,f} + w_{s,f}(n, \sum_{i=1}^{F} p_i) + g \cdot h_{s,f}(n, \sum_{i=1}^{F} p_i) \right] + G \cdot H_s(n, p_1, \ldots, p_F) \right\} \\ &= \sum_{s=1}^{S} \left\{ \max_{f \in F} \left[q_{s,f} + T_{s,f}(n, \sum_{i=1}^{F} p_i) \right] + G \cdot H_s(n, p_1, \ldots, p_F) \right\} \end{aligned} \tag{3}$$

where G is the speed of the cluster-to-cluster network, H is the amount of data to communicate among different facilities, and $\max_{f \in F}$ models that we need to wait for the slowest facility f to finish before the global communication phase. If the problem is scalable enough, then $w_{s,f}(n, \sum_{i=1}^{F} p_i) < w_{s,f}(n, p_f)$ for any $f \in F$. However, G can be orders of magnitude higher than g and queue times can vary significantly from cluster to cluster. Therefore, a good cross-facility approach should try to:

- minimise cross-cluster communications to reduce $H_s(n, p_1, \ldots, p_F)$;

- keep $w_{s,f}(n, \sum_{i=1}^{F} p_i)$ large enough to compensate for the overhead of $q_{s,f}$;
- balance the workload of different clusters e and f such that $T_{s,e}(n, \sum_{i=1}^{F} p_i) \simeq T_{s,f}(n, \sum_{i=1}^{F} p_i)$ for any $e, f \in F$.

If all these conditions hold, Eq. 3 can be rewritten as

$$T_3 \simeq \sum_{s=1}^{S} \left[w_s(n, \sum_{i=1}^{F} p_i) + g \cdot h_s(n, \sum_{i=1}^{F} p_i) \right] = \sum_{s=1}^{S} T_s(n, \sum_{i=1}^{F} p_i) < T_2 \tag{4}$$

The classical data-parallel distributed training of a DNN is much more complex than a pure BSP computation [35]. However, each model update can be approximated with a BSP superstep that locally computes the forward pass and back-propagation on each node, communicates the gradients to all other nodes, and globally synchronises the process to compute the resulting gradients. Unfortunately, this algorithm necessitates substantial communication among all involved facilities, leading to a communication-bound problem [43]. In contrast, the FL approach simplifies this process by only requiring the exchange and aggregation of models, thereby minimising cross-cluster data transfers (each superstep models one FL round). Each cluster trains a model on the local data, which is then exchanged to update the global model before a new FL round starts. By carefully tuning the amount of data available on each cluster, we can ensure a balanced workload among different, potentially heterogeneous facilities. The key question that remains to be answered is whether the training time of foundation models is large enough to offset the queuing times variability and the increased communication overhead in cross-cluster settings, which is the primary focus of Sec. 4.

4. Evaluation

4.1. Test case

We used XFFL to continue training LLaMAv2 [9], a foundation model for natural language processing. LLaMAv2 is a family of large language models ranging up to 70B parameters based on the transformer architecture [47] and pre-trained with up to 2T tokens. The tokens stem from a mix of text corpora selected to provide a fair mix of demographic representations along five axes: religion, gender and sex, nationality, race and ethnicity, and sexual orientation. Nevertheless, the corpora are heavily centred on the English language (89.7%), with the second most represented language (German) lagging far behind (0.17%).

Inspired by this imbalance, we fine-tuned a LLaMAv2 7B model specifically for Italian and Czech using the clean_mC4_it [48] and mC4_cs [49] datasets. The mC4_cs dataset is the plain version of the mC4 corpus Czech split, while the clean_mC4_it dataset is a filtered version of the mC4 corpus Italian split. This filtering involved removing documents containing despicable words, sentences that are too short (<3 words) or too long (>1000 characters), sentences not ending with end-of-sentence punctuation, documents which are either too short (<5 sentences or 500 characters) or too long (>50000 characters), and so on. The result is a cleaned Italian corpus of 103 million documents, comprising 41 billion words.

Due to the size of the entire corpus, which was incompatible with our experiment's time requirements, we opted to use the "tiny" split. During preprocessing, the documents have been converted into fixed-length sequences of 2048 tokens through the standard LLaMAv2 tokeniser. As a result, the Italian dataset used consisted of 10 million documents comprising 4 billion words, converted into a total of 4085342 training samples and 13252 testing samples with 2048 tokens each. We retained the same quantity of data from the mC4_cs dataset to ensure balanced execution times. Each dataset occupies about 102GB of disk space, much more than the 13GB required by the half-precision representation of the LLaMAv2 7B weights. Due to limited resource availability, the total execution time required to train LLaMAv2 7B on the whole clean_mC4_it and mC4_cs datasets had to be estimated. This is possible since the training time is linear in the training dataset size, modulo having fixed hyperparameters. Once the aggregate data processing speed (usually expressed in terms of iterations/second) is stable, estimating the total execution time becomes straightforward, knowing the number of iterations needed for the dataset (1 iteration = 1 data batch = 4 data samples = 8192 tokens in our testbed).

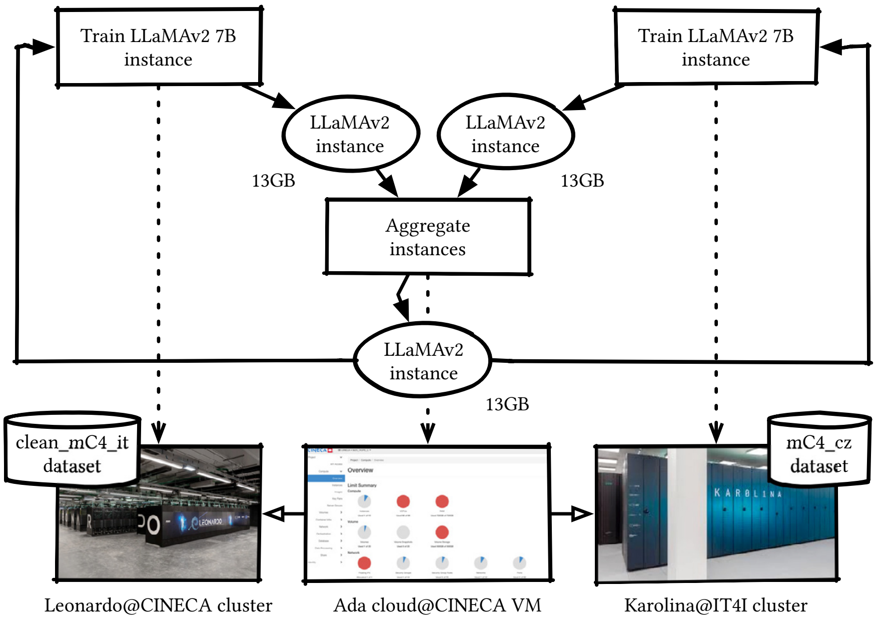


Fig. 1. Pictorial representation of the XFFL hybrid workflow. Steps are represented as rectangles, ports as circles, and dependencies as arrows with black-filled heads. Locations are represented in the lower rectangles, and communication channels between them as arrows with white-filled heads. Finally, mapping relations are expressed as dotted arrows.

4.2. *Execution environment*

The experiment has been performed on two HPC systems, Leonardo@CINECA and Karolina@IT4I. They occupy the 6th and 113th positions in the Top500 ranking [10] (November 2023), respectively. In particular, we trained LLaMAv2 on the reduced clean_mC4_it dataset on Leonardo@CINECA, while we relied on Karolina@IT4I to train the model on the mC4_cs data.

Leonardo is a pre-exascale system with a performance peak of 238.70 PFlop/s (LINPACK R_{max}) hosted by CINECA in Bologna, Italy. It consists of 4992 liquid-cooled compute nodes interconnected through an NVIDIA Mellanox network, with Dragon Fly+, capable of a maximum bandwidth of 200Gbit/s between each pair of nodes. Each of the (Booster) nodes used in our experiment mount a custom BullSequana X2135 "Da Vinci" blade equipped with an Intel Xeon 8358 Ice Lake CPU (32 cores, 2.6GHz), 512GB DDR4 RAM (8 x 64GB, 3200MHz), 4 NVIDIA custom A100 GPUs (64GB, HBM2e), and 2 NVIDIA InfiniBand HDR 2×100Gb/s network cards. Karolina, on the other hand, is a petascale system hosted by the IT4Innovations National Supercomputing Center at the VSB-Technical University of Ostrava, Czech Republic. Each of its GPU nodes mounts an HPE Apollo 6500 Gen10 Plus blade equipped with an AMD EPYC 7763 CPU (64 cores, 2.45GHz), 1TB DDR4 RAM (8 x 128GB, 3200MHz), 8 NVIDIA A100 GPUs (40GB, HBM2), and 4 InfiniBand HDR 200Gb/s network cards. Both facilities rely on SLURM [50] for job submission and workload management.

After each training round, the model aggregation averages the model weights through FedAVG, which has been shown effective also on transformer-type models [51]. This operation is executed on a Virtual Machine (VM) with 96 vCPUs and 720GB RAM, hosted on the OpenStack-based Ada cloud@CINECA. Despite being geographically near the Leonardo@CINECA facility, the Ada cloud@CINECA VMs do not mount any direct interconnection with the HPC computing nodes, requiring all data to be routed on the Internet to be transferred between the two environments.

Table 1. LLaMAv2 7B training performance on Leonardo@CINECA. 1 it = 1 data batch = 4 data samples = 8192 tokens

#Nodes	#GPUs	Loading time (s)	Queuing time (s)	Estimated execution time (hours)	Aggregate data processing speed (it/s)	Speedup	Efficiency
2	8	34	2	774	0.35	2.00	1.00
4	16	34	2	385	0.99	4.02	1.00
8	32	34	2	193	2.83	8.02	1.00
16	64	34	2	98	8.16	15.80	0.99
32	128	38	2	49	23.19	31.59	0.99
64	256	90	222	25	66.32	61.92	0.97
128	512	120	438	14	179.02	110.57	0.86

4.3. Implementation

The training process relies on the PyTorch 2.2.0 framework [52], manually compiled and optimised on both HPC facilities. As a future work, we want to make the experiment more portable and reproducible by relying on software containers (e.g., Singularity [53]), which have also recently shown promising results for performance portability.

First, datasets have been tokenized and manually moved to each HPC centre as described in Sec. 4.1. As assumed in standard FL practice, we assume data to be local to each facility. At each facility, the LLaMAv2 model has been distributed among the available computing nodes through the Fully Sharded Data Parallel (FSDP) methodology [54, 55], using different sharding strategies for each infrastructure to fit the local hardware characteristics better. From a high-level perspective, the LLaMAv2 model is sharded at the intra-node level, following a model parallel approach, and replicated on each node, embracing a data parallel approach. At the end of every round, models are aggregated on the Ada cloud@CINECA VM.

The cross-facility interactions have been modelled as a hybrid CWL workflow [22], ensuring reproducibility and reusability. In detail, the workflow has been specified using CWLv1.2 with the `cwltool:Loop` extension [21], as the current CWL standard does not support iterative workflows. However, this extension is currently under evaluation to be included as a native feature of CWLv1.3. At runtime, the workflow has been orchestrated by a StreamFlow [13] instance deployed on the Ada cloud@CINECA VM. From the VM, StreamFlow can communicate with each HPC centre through SSH connections, interacting with the workload managers to submit and monitor the training steps. StreamFlow also manages all the data transfers, i.e., it collects an instance of the model from each facility at the end of each training round and transfers back a copy of the updated model to each facility after the aggregation step. The aggregation step is performed directly on the VM since it has a low complexity with respect to model training. Note that the datasets are never moved, according to the FL principles. Fig. 1 depicts the whole XFFL workflow and its mapping onto the cross-facility execution environment described in Sec. 4.2. All the code is available as Open Source on GitHub[1].

4.4. Discussion

First, we executed a large-scale training of the LLaMAv2 7B model on the clean_mC4_it dataset on top of the Leonardo@CINECA facility to study the strong scaling of the training process itself. The goal was to check how much we can scale on a single facility and how the requested resources affect the queueing time. The queueing times and an estimation of the execution times ranging from 2 to 128 nodes are reported in Table 1. In our experiments, queueing times were negligible for up to 32 nodes but increased to over 7 minutes for 128 nodes. However, execution times were orders of magnitude higher, ranging from 774 (on 2 nodes) to 14 (on 128 nodes) hours. Therefore, despite

[1] `https://github.com/alpha-unito/xffl`

Table 2. LLaMAv2 7B cross-facility federated training overhead

	Leonardo@CINECA Transfer time (s)	Karolina@IT4I Transfer time (s)	Model aggregation time (s)
Min	138	178	118
Avg	217	242	133
Max	360	344	143

showing a very good strong scaling, training times offset queueing times even for large amounts of computing power, making it perfectly feasible to separate different training rounds into multiple job submissions without a major impact on the overall performance.

Another factor to consider is the loading time, which is the time required for all nodes to perform initial operations, such as loading the model and dataset into RAM and GPU memory and initialising the communication layer between all participating nodes. However, even considering the loading time overhead, which ranged from 34 to 120 seconds, separating iterations into different submissions remains a viable and sustainable option.

Having validated the overall approach as viable on one HPC centre, we ran the full experiment using the FL workflow described in Sec. 4.3 and measured all the additional sources of overhead, i.e., the transfer times between the Ada cloud@CINECA VM and the two HPC facilities and the time needed to aggregate the model on the Ada cloud@CINECA VM. We repeated the experiment five times and collected maximum, minimum, and average values reported in Table 2. Note that these times do not depend on the resources allocated in each HPC facility. Again, empirical measures promote the feasibility of the XFFL approach: none of the measured times exceeds hundreds of seconds, making them negligible compared to the tens of hours needed by each training round.

5. Conclusion and future work

In this work, we propose XFFL as a powerful tool to address the compute divide between industry and academia. By harnessing the capabilities of multiple independent computing centres for extreme-scale DNN training, XFFL offers a promising solution. Hybrid workflows, a key component of XFFL, allow users to abstract complex technical details and orchestration strategies of cross-facility workloads, making the training process more accessible. While this approach introduces some overhead due to additional queueing and loading times, these are insignificant compared to the time required for local training. Additionally, XFFL minimises cross-cluster interconnections, which are significantly slower than intra-cluster communications. Our empirical results demonstrate that model transfer and aggregation times are negligible compared to training round times, further highlighting the efficiency of XFFL.

Looking ahead, full-scale experimentation on a larger federation of EuroHPC facilities is already underway. We also plan to explore a diverse set of models and datasets to test the results' generality better. Another future development of this technology aims to ease workflow design by providing a customisable and parametric CWL workflow template while promoting better integration with StreamFlow and software container technologies, such as SLURM and Singularity. Indeed, other than fostering reproducibility, optimised NVIDIA PyTorch Singularity containers offer a considerable performance boost compared to bare-metal deployments.

Acknowledgements

This article describes work undertaken in the context of the SPACE Center of Excellence, "Scalable Parallel and distributed Astrophysical Codes for Exascale" which has received funding from the EuroHPC JU under grant agreement No. 101093441, and the Spoke "FutureHPC & BigData" of the ICSC – Centro Nazionale di Ricerca in "High Performance Computing, Big Data and Quantum Computing", funded by European Union – NextGenerationEU. We also thankfully acknowledge the CINECA award under the ISCRA initiative, and the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254), for the availability of high-performance computing resources and support.

References

[1] J. Sevilla, L. Heim, A. Ho, T. Besiroglu, M. Hobbhahn, P. Villalobos, Compute trends across three eras of machine learning, in: IEEE IJCNN, 2022, pp. 1–8. doi:10.1109/IJCNN55064.2022.9891914.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90. doi:10.1145/3065386.

[3] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, et al., PaLM: Scaling language modeling with pathways, J. Mach. Learn. Res. 24 (2023) 240:1–240:113.

[4] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: NAACL-HLT, Association for Computational Linguistics, 2019, pp. 4171–4186. doi:10.18653/V1/N19-1423.

[5] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707. doi:10.1016/J.JCP.2018.10.045.

[6] Z. Ma, J. He, J. Qiu, H. Cao, Y. Wang, et al., BaGuaLu: targeting brain scale pretrained models with over 37 million cores, in: ACM PPoPP, 2022, pp. 192–204. doi:10.1145/3503221.3508417.

[7] Meta, Introducing the AI research supercluster — Meta's cutting-edge AI supercomputer for AI research, accessed: 2024-04-10 (2022). URL https://ai.meta.com/blog/ai-rsc/

[8] T. Besiroglu, S. A. Bergerson, A. Michael, L. Heim, X. Luo, N. Thompson, The compute divide in machine learning: A threat to academic contribution and scrutiny?, CoRR abs/2401.02452 (2024). arXiv:2401.02452.

[9] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, et al., Llama 2: Open foundation and fine-tuned chat models, CoRR abs/2307.09288 (2023). arXiv:2307.09288.

[10] TOP500 Project, Top500 supercomputer sites, accessed: 2024-04-07 (2024). URL https://www.top500.org/

[11] I. Colonnelli, Workflow models for heterogeneous distributed systems, in: (ITADATA 2023), Vol. 3606 of CEUR Workshop Proceedings, 2023.

[12] E. Deelman, K. Vahi, M. Rynge, R. Mayani, R. F. da Silva, G. Papadimitriou, M. Livny, The evolution of the Pegasus workflow management software, Comput. Sci. Eng. 21 (4) (2019) 22–36. doi:10.1109/MCSE.2019.2919690.

[13] I. Colonnelli, B. Cantalupo, I. Merelli, M. Aldinucci, StreamFlow: cross-breeding cloud with HPC, IEEE Trans. Emerg. Top. Comput. 9 (4) (2021) 1723–1737. doi:10.1109/TETC.2020.3019202.

[14] D. D. Sánchez-Gallegos, D. D. Luccio, S. Kosta, J. L. G. Compeán, R. Montella, An efficient pattern-based approach for workflow supporting large-scale science: The DagOnStar experience, Future Gener. Comput. Syst. 122 (2021) 187–203. doi:10.1016/J.FUTURE.2021.03.017.

[15] I. Colonnelli, M. Aldinucci, B. Cantalupo, L. Padovani, S. Rabellino, C. Spampinato, R. Morelli, R. D. Carlo, N. Magini, C. Cavazzoni, Distributed workflows with Jupyter, Future Gener. Comput. Syst. 128 (2022) 282–298. doi:10.1016/J.FUTURE.2021.10.007.

[16] R. B. Roy, T. Patel, V. Gadepally, D. Tiwari, Mashup: making serverless computing useful for HPC workflows via hybrid execution, in: ACM PPoPP, 2022, pp. 46–60. doi:10.1145/3503221.3508407.

[17] R. F. da Silva, R. Filgueira, E. Deelman, E. Pairo-Castineira, I. M. Overton, M. P. Atkinson, Using simple PID-inspired controllers for online resilient resource management of distributed scientific workflows, Future Gener. Comput. Syst. 95 (2019) 615–628. doi:10.1016/J.FUTURE.2019.01.015.

[18] A. Mulone, S. Awad, D. Chiarugi, M. Aldinucci, Porting the variant calling pipeline for NGS data in cloud-HPC environment, in: IEEE COMPSAC, 2023, pp. 1858–1863. doi:10.1109/COMPSAC57700.2023.00288.

[19] D. A. Brown, P. R. Brady, A. Dietz, J. Cao, B. Johnson, J. McNabb, A case study on the use of workflow technologies for scientific analysis: Gravitational wave data analysis, in: Workflows for e-Science, Scientific Workflows for Grids, Springer, 2007, pp. 39–59. doi:10.1007/978-1-84628-757-2_4.

[20] J. A. Barron-Lugo, J. L. G. Compeán, J. Carretero, I. López-Arévalo, R. Montella, A novel transversal processing model to build environmental big data services in the cloud, Environ. Model. Softw. 144 (2021) 105173. doi:10.1016/J.ENVSOFT.2021.105173.

[21] I. Colonnelli, B. Casella, G. Mittone, Y. Arfat, B. Cantalupo, R. Esposito, A. R. Martinelli, D. Medić, M. Aldinucci, Federated learning meets HPC and cloud, in: Astrophysics and Space Science Proceedings, Vol. 60, Springer, 2023, p. 193–199. doi:10.1007/978-3-031-34167-0_39.

[22] M. R. Crusoe, S. Abeln, A. Iosup, P. Amstutz, J. Chilton, N. Tijanic, H. Ménager, S. Soiland-Reyes, C. A. Goble, Methods included: Standardizing computational reuse and portability with the Common Workflow Language, Communication of the ACM (2022). doi:10.1145/3486897.

[23] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: A. Singh, X. J. Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, Vol. 54 of Proceedings of Machine Learning Research, PMLR, 2017, pp. 1273–1282.

[24] G. Mittone, N. Tonci, R. Birke, I. Colonnelli, D. Medic, A. Bartolini, R. Esposito, E. Parisi, F. Beneventi, M. Polato, M. Torquati, L. Benini, M. Aldinucci, Experimenting with emerging RISC-V systems for decentralised machine learning, in: ACM CF, 2023, pp. 73–83. doi:10.1145/3587135.3592211.

[25] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, A. T. Suresh, SCAFFOLD: stochastic controlled averaging for federated learning, in: ICML, Vol. 119, PMLR, 2020, pp. 5132–5143.

[26] H. Wang, Z. Kaplan, D. Niu, B. Li, Optimizing federated learning on non-iid data with reinforcement learning, in: IEEE INFOCOM, 2020, pp. 1698–1707.

[27] Y. Chen, Y. Ning, M. Slawski, H. Rangwala, Asynchronous online federated learning for edge devices with non-iid data, in: IEEE BigData, 2020, pp. 15–24.

[28] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. Malek, D. Huba, Federated learning with buffered asynchronous aggregation, in: AISTATS, Vol. 151, PMLR, 2022, pp. 3581–3607.

[29] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, S. Cui, UVeQFed: Universal vector quantization for federated learning, IEEE Trans. Signal Process. 69 (2021) 500–514. doi:10.1109/TSP.2020.3046971.

[30] F. Sattler, S. Wiedemann, K. Müller, W. Samek, Robust and communication-efficient federated learning from non-i.i.d. data, IEEE Trans. Neural Networks Learn. Syst. 31 (9) (2020) 3400–3413. doi:10.1109/TNNLS.2019.2944481.

[31] C. Wu, F. Wu, R. Liu, L. Lyu, Y. Huang, X. Xie, FedKD: Communication efficient federated learning via knowledge distillation, CoRR abs/2108.13323 (2021). arXiv:2108.13323.

[32] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, et al., Advances and open problems in federated learning, Found. Trends Mach. Learn. 14 (1-2) (2021) 1–210. doi:10.1561/2200000083.

[33] B. Y. Lin, C. He, Z. Ze, H. Wang, Y. Hua, C. Dupuy, R. Gupta, M. Soltanolkotabi, X. Ren, S. Avestimehr, FedNLP: Benchmarking federated learning methods for natural language processing tasks, in: NAACL, Association for Computational Linguistics, 2022, pp. 157–175. doi:10.18653/V1/2022.FINDINGS-NAACL.13.

[34] R. Esposito, M. Polato, M. Aldinucci, Boosting methods for federated learning, in: D. Calvanese, C. Diamantini, G. Faggioli, N. Ferro, S. Marchesin, G. Silvello, L. Tanca (Eds.), SEBD, Vol. 3478 of CEUR Workshop Proceedings, CEUR-WS.org, 2023, pp. 439–448.

[35] T. Ben-Nun, T. Hoefler, Demystifying parallel and distributed deep learning: An in-depth concurrency analysis, ACM Comput. Surv. 52 (4) (2019) 65:1–65:43. doi:10.1145/3320060.

[36] P. Viviani, M. Drocco, D. Baccega, I. Colonnelli, M. Aldinucci, Deep learning at scale, in: Proc. of 27th Euromicro Intl. Conference on Parallel Distributed and network-based Processing (PDP), IEEE, Pavia, Italy, 2019, p. 124–131. doi:10.1109/EMPDP.2019.8671552.

[37] G. Mittone, F. Svoboda, M. Aldinucci, N. D. Lane, P. Lió, A federated learning benchmark for drug-target interaction, in: ACM WWW Companion, 2023, pp. 1177–1181. doi:10.1145/3543873.3587687.

[38] P. Foley, M. J. Sheller, B. Edwards, S. Pati, W. Riviera, M. Sharma, P. N. Moorthy, S. han Wang, J. Martin, P. Mirhaji, P. Shah, S. Bakas, OpenFL: the open federated learning library, Physics in Medicine & Biology 67 (21) (2022) 214001. doi:10.1088/1361-6560/ac97d9.

[39] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, N. D. Lane, Flower: A friendly federated learning research framework, CoRR abs/2007.14390 (2020). arXiv:2007.14390.

[40] H. R. Roth, Y. Cheng, Y. Wen, I. Yang, Z. Xu, Y. Hsieh, et al., NVIDIA FLARE: federated learning from simulation to real-world, IEEE Data Eng. Bull. 46 (1) (2023) 170–184.

[41] C. He, S. Li, J. So, M. Zhang, H. Wang, et al., FedML: A research library and benchmark for federated machine learning, CoRR abs/2007.13518 (2020). arXiv:2007.13518.

[42] W. Zhuang, C. Chen, L. Lyu, When foundation model meets federated learning: Motivations, challenges, and future directions, CoRR abs/2306.15546 (2023). arXiv:2306.15546.

[43] B. Yuan, Y. He, J. Davis, T. Zhang, T. Dao, et al., Decentralized training of foundation models in heterogeneous environments, in: NeurIPS, 2022.

[44] M. Xu, D. Cai, Y. Wu, X. Li, S. Wang, FwdLLM: Efficient FedLLM using forward gradient, CoRR abs/2308.13894 (2023). arXiv:2308.138946.

[45] H. Chen, Y. Zhang, D. Krompass, J. Gu, V. Tresp, FedDAT: An approach for foundation model finetuning in multi-modal heterogeneous federated learning, in: AAAI, 2024, pp. 11285–11293. doi:10.1609/AAAI.V38I10.29007.

[46] L. G. Valiant, A bridging model for parallel computation, Commun. ACM 33 (8) (1990) 103–111. doi:10.1145/79173.79181.

[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: NIPS, 2017, pp. 5998–6008.

[48] G. Sarti, M. Nissim, IT5: large-scale text-to-text pretraining for italian language understanding and generation, CoRR abs/2203.03759 (2022). arXiv:2203.03759.

[49] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, J. Mach. Learn. Res. 21 (2020) 140:1–140:67.

[50] M. A. Jette, T. Wickberg, Architecture of the slurm workload manager, in: D. Klusácek, J. Corbalán, G. P. Rodrigo (Eds.), JSSPP, Vol. 14283 of Lecture Notes in Computer Science, Springer, 2023, pp. 3–23. doi:10.1007/978-3-031-43943-8_1.

[51] Y. Tian, Y. Wan, L. Lyu, D. Yao, H. Jin, L. Sun, Fedbert: When federated learning meets pre-training, ACM Trans. Intell. Syst. Technol. 13 (4) (2022) 66:1–66:26. doi:10.1145/3510033.

[52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, et al., PyTorch: An imperative style, high-performance deep learning library, in: NeurIPS, 2019, pp. 8024–8035.

[53] G. M. Kurtzer, V. Sochat, M. W. Bauer, Singularity: Scientific containers for mobility of compute, PLOS ONE 12 (5) (2017) 1–20. doi:10.1371/journal.pone.0177459.

[54] Y. Xu, H. Lee, D. Chen, H. Choi, B. A. Hechtman, S. Wang, Automatic cross-replica sharding of weight update in data-parallel training, CoRR abs/2004.13336 (2020). arXiv:2004.13336.

[55] Y. Zhao, A. Gu, R. Varma, L. Luo, C. Huang, et al., PyTorch FSDP: experiences on scaling fully sharded data parallel, Proc. VLDB Endow. 16 (12) (2023) 3848–3860. doi:10.14778/3611540.3611569.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 13–16

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Training speech recognition models at the National Library of Sweden

Leonora Vesterbacka Olsson[a]

[a]*KBLab, National Library of Sweden, Humlegården 26, 102 41, Stockholm*

Abstract

The abundance of openly available audio data in English enables pretraining of automatic speech recognition (ASR) systems on hundreds of thousands to millions of hours of recorded speech. As a result speech recognition systems are approaching human level robustness in English. Other languages' performance in multilingual speech recognition systems tend to stand in proportion to the amount of data included from the language – or the language family – in question. For low to mid resource languages with fewer speakers, the amount of openly available data may be limited, and as a consequence these languages tend to be underrepresented in large scale efforts to train multilingual speech recognition systems. The National Library of Sweden hosts large collections of audio recordings. These resources allow us to potentially bridge some of the existing speech recognition performance gaps between Swedish and higher resource languages. We believe Swedish speech recognition can be further improved upon by scaling up the amount of training data. We have constructed an inclusive and transparent speech corpus with emphasis on all variations of spoken Swedish that we will use to train speech-to-text models for Swedish.

Keywords: AI ; Automatic speech recognition ; Speech-to-text ; Speech recognition

1. Introduction

The National Library of Sweden is responsible for collecting, preserving and giving access to everything that is published in Sweden. KBLab [3] is a data lab at the Library, recognized as a national research infrastructure, where large scale quantitative research is performed on the library's collections. According to the EU Directive 2019/790 of the European parliament and of the council of 17 April 2019 on copyright and related rights in the Digital Single Market and amending Directives 96/9/EC and 2001/29/ECT, cultural heritage institutions are allowed to use text and

* Corresponding author. ORCID: 0000–0002-6966-5081
E-mail address: leonora.vesterbackaolsson@kb.se

data mining technologies on material they have lawful access to. This directive enables KBLab to train large language models and speech recognition models on the library's collections.

Just like many other cultural heritage institutions and archives across Europe, the National Library of Sweden hosts large collections of audio recordings and has digitized over one million hours of local radio broadcasts covering a broad variety of Swedish dialects. KBLab is using the library's audio data to train speech-to-text models and possibly bridge the existing speech recognition performance gaps between Swedish and higher resource languages. Since much of the library's audio data is not in the public domain, the library is restricted in sharing its data. This necessitates using either our own computing resources or computing resources with sufficiently strong data protection requirements, as is guaranteed by the EuroHPC JU. KBLab has been awarded development access to Leonardo [8] through the EuroHPC Joint Undertaking to preprocess training data and benchmark the training times for speech recognition models, in preparation for a future access where the training will be conducted.

The training of speech recognition models benefits from both unlabeled and labeled audio data. Our project focuses on two established model architectures that use slightly different training objectives. In wav2vec2.0 [2], a training scheme using only unlabeled speech data in the pretraining step followed by a secondary step of finetuning with transcribed speech was introduced. On the other hand, the Whisper model [7] released by OpenAI, has shown that pretraining on large amounts of labeled speech data leads to speech recognition approaching human level robustness in English. Common to both models, the end performance scales with increased amount of speech data, labeled or unlabeled.

The National Library of Sweden has already initiated efforts to pretrain and finetune speech recognition models for Swedish. In 2021, the first wav2vec2.0 model for Swedish was released and was shown to outperform similar multilingual offerings [6]. We believe Swedish speech recognition can be further improved upon by scaling up the amount of training data. The library has over a million hours of digitized audio from radio broadcast, from which only a subset of 10,000 hours was used for the training in [6]. In addition to the radio archives the library also collects all TV broadcasted in Sweden. A large fraction of the TV have transcriptions in the form of subtitles, which serves as a perfect training material for finetuning Whisper. Additionally, KBLab has curated thousands of hours of speech from parliamentary debates and their corresponding transcripts as well as youtube clips with Swedish speech paired with Swedish subtitles. The aim of this project is to pretrain and finetune a wav2vec2.0 model, and finetune a selection of Whisper models of various sizes.

1.1. Speech recognition models

Acoustic models maps a waveform, a signal, to the phonetic units of language. Speech recognition inputs a signal and generates a string of words, whereas speech synthesis inputs a string of words and outputs a synthesized speech as a signal. Our project focuses on two established model architectures for speech recognition that use slightly different training objectives. wav2vec2.0 has a training objective similar to that of a BERT model [5], i.e. masked language modelling where a token (or a part of the signal in the case of speech) is masked and the model learns by guessing what is masked out. This self-supervised learning enables learning without the need for large amounts of labeled training data, which is rare for low resource languages. On the other hand, the Whisper model [7] released by OpenAI is trained in a supervised manner. But the authors [7] could show that lower quality standards of the labeled training data, e.g. subtitles, was enough to still achieve speech recognition approaching human level robustness in English. By relaxing the quality standard of labeled training data vast amounts of training data is unlocked. Common to both models, the end performance scales with increased amount of speech data, labeled or unlabeled. Table 1. shows the fraction of Swedish training data used in the multilingual variants of wav2vec2.0 and Whisper, where it is evident that the representation of Swedish speech is sub optimal.

Table 1. Fraction of Swedish in the original training datasets used to train wav2vec2.0 and Whisper

Model	Developer	Total dataset [h]	Only Swedish [h]
wav2vec2.0 (XLS-R)	Meta	436 000	16 325
Whisper	OpenAI	680 000	2119

1.2. An inclusive speech corpus

With the results from Table 1. in mind, a larger Swedish speech corpus could potentially increase the performance of the models on Swedish speech. In order to train a robust speech recognition model for Swedish, attention has been given to both the size of the corpus, as well as the data sources, to construct a corpus that is inclusive and represents all of the speech spoken in Sweden in terms of gender, age and region of the speaker. The word error rate (wer), a metric used to assess the performance of a model, decreases with the addition of more training data. According to Table 6 in [7, chap. 4.2] the English wer decreases with the addition of training data and reaches a plateau at around 30 000 hours. This dataset size 30 000 hours has been a motivation for KBLab when constructing the speech corpus. The labeled and unlabeled data sources of the inclusive speech corpus will be outlined in the following sections.

1.2.1. TV with subtitles

Everything broadcasted in Sweden is subject to the deposit law. As a result, all TV channels continuously deliver one copy of everything that has been broadcasted to the National Library of Sweden. If the video file contains Swedish subtitles and the subtitles match spoken Swedish, the material is saved in our corpus. The subtitles are deduplicated to remove unwanted duplicates in the training data. Even though subtitles provide a simplified transcription of the speech, often omitting words for simplicity to fit the screen, they are still a suitable training dataset for finetuning Whisper. Depending on the quality criteria of the subtitles, the size of this dataset amounts to approximately 10 000 hours.

1.2.2. Parliament debates

The Riksdag Administration is a government agency responsible for supporting the work of the Swedish Parliament, Riksdagen. Every parliament debate is recorded and transcribed. In this unique collaboration KBLab has been granted access to parliament debates from 1966 until today, of which the total usable material amounts to approximately 20 000 hours.

1.2.3. Dialect recordings

The Institute for Language and Folklore is a government agency which holds large collections of audio recordings of spoken Swedish. The main part of the recordings are not transcribed, but the smaller fraction that is transcribed is a very valuable source of data as it represents a wide variety of Swedish dialects. The transcribed portion of the audio recordings can be used in the finetuning of both Whisper and wav2vec2.0, and amounts to approximately 150 hours, and the unlabeled recordings can be used to pretrain a special dialect version of wav2vec2.0.

1.2.4. Crowd sourced recordings

There exists crowd sourced initiatives that collects audio recordings of volunteers along with a transcription. For some of these sources there are significant contributions in Swedish such as CommonVoice (46 hours) [1] and FLEURS (12 hours) [4] that are included in our training data.

1.2.5. Local radio

The wav2vec2.0 model is pretrained in a self-supervised manner using training data without corresponding transcriptions. The training data used in the pretraining consists of radio broadcasts which the National Library of Sweden obtains as a result of the deposit law, and amounts to 100 000 hours.

1.3. Computing resources

KBLab has been awarded computing resources at the Leonardo supercomputer [8] in a development access project through the EuroHPC Joint Undertaking. The development access consists of a computing budget of 3500 node hours over a one year period. During 2023 KBLab performed a preprocessing of some of the audio sources and a successful benchmarking of the training times of the wav2vec2.0 and Whisper training codes anticipated on the current hardware setup. These benchmarks are used to apply for further computing resources through the EuroHPC Joint Undertaking, to conduct the final training of the two models using the inclusive Swedish speech corpus that has been constructed.

1.4. Future work

The results from the development access is used to apply for the computing resources needed to pretrain and finetune the two models. KBLab plans to produce five finetuned Whisper versions, one for each model size, in order to accommodate for all potential users of the models. In addition to this, KBLab intends to pretrain and finetune two versions of wav2vec2.0, one with a general pretraining on unlabeled radio, as well as a pretraining on an unlabeled dataset with dialect recordings from the Institute for Language and Folklore. KBLab will release all the models trained in this project under an Apache 2.0 license on huggingface[1] for use by the general public, Swedish government agencies, companies and researchers. The performance of the finetuned models will be compared to that of the multilingual versions of Whisper and wav2vec2.0 on an evaluation dataset, comparing the word error rate and BLEU score of the resulting transcriptions. The models performance on Swedish dialects will be evaluated using a dataset with speech annotated with dialect information.

Acknowledgements

The author acknowledges EuroHPC Joint Undertaking for awarding this project development access to Leonardo.

References

[1] Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F.M., Weber, G., 2020. Common voice: A massively-multilingual speech corpus. arXiv:1912.06670.

[2] Baevski, A., Zhou, H., Mohamed, A., Auli, M., 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. arXiv:2006.11477.

[3] Börjeson, L., Haffenden, C., Malmsten, M., Klingwall, F., Rende, E., Kurtz, R., Rekathati, F., Hägglöf, H., Sikora, J., 2023. Transfiguring the library as digital research infrastructure: Making kblab at the national library of sweden. URL: osf.io/preprints/socarxiv/w48rf, doi:10.31235/osf.io/w48rf.

[4] Conneau, A., Ma, M., Khanuja, S., Zhang, Y., Axelrod, V., Dalmia, S., Riesa, J., Rivera, C., Bapna, A., 2022. Fleurs: Few-shot learning evaluation of universal representations of speech. arXiv:2205.12446.

[5] Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805.

[6] Malmsten, M., Haffenden, C., Börjeson, L., 2022. Hearing voices at the national library – a speech corpus and acoustic model for the swedish language. arXiv:2205.03026.

[7] Radford, A., Kim, J.W., Xu, T., Brockman, G., McLeavey, C., Sutskever, I., 2022. Robust speech recognition via large-scale weak supervision. arXiv:2212.04356.

[8] Turisini, M., Amati, G., Cestari, M., 2023. Leonardo: A pan-european pre-exascale supercomputer for hpc and ai applications. arXiv:2307.16885.

[1] https://huggingface.co/KBLab

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 17–30

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

A GPU-ready pseudo-spectral method for direct numerical simulations of multiphase turbulence

Alessio Roccon[a,b,*]

[a] Polytechnic Department of Engineering and Architecture, University of Udine, 33100 Udine, Italy
[b] Institute of Fluid Mechanics and Heat Transfer, TU-Wien, 1060 Vienna, Austria

Abstract

In this work, we detail the GPU-porting of an in-house pseudo-spectral solver tailored towards large-scale simulations of interface-resolved simulation of drop- and bubble-laden turbulent flows. The code relies on direct numerical simulation of the Navier-Stokes equations, used to describe the flow field, coupled with a phase-field method, used to describe the shape, deformation, and topological changes of the interface of the drops or bubbles. The governing equations – Navier-Stokes and Cahn-Hilliard equations – are solved using a pseudo-spectral method that relies on transforming the variables in the wavenumber space. The code targets large-scale simulations of drop- and bubble-laden turbulent flows and relies on a multilevel parallelism. The first level of parallelism relies on the message-passing interface (MPI) and is used on multi-core architectures in CPU-based infrastructures. A second level of parallelism relies on OpenACC directives and cuFFT libraries and is used to accelerate the code execution when GPU-based infrastructures are targeted. The resulting multiphase flow solver can be efficiently executed in heterogeneous computing infrastructures and exhibits a remarkable speed-up when GPUs are employed. Thanks to the modular structure of the code and the use of a directive-based strategy to offload code execution on GPUs, only minor code modifications are required when targeting different computing architectures. This improves code maintenance, version control and the implementation of additional modules or governing equations.

Keywords: multiphase turbulence; phase-field method; GPU-computing

1. Introduction

Turbulent multiphase flows are ubiquitous in nature and our everyday life. These flows play a key-role in various applications, from geophysical phenomena [25, 47], to environmental and industrial processes [45, 58, 59]. With respect to single-phase turbulence, the description and modeling of multiphase turbulence is much more complex, since these type of flows require the description of an ever-moving and deforming interface, its topological modifications,

* Corresponding author. Tel.: +39-0432-558006.
E-mail address: alessio.roccon@uniud.it

and the underlying turbulent flow [15, 51, 54, 63]. Simulations constitute an essential tool to investigate the physics of multiphase turbulence and are becoming increasingly popular in recent years: numerical simulations give access to detailed space- and time-resolved information on the flow field and on the morphology of the two phases.

Obtaining an accurate description of the dynamics of a turbulent multiphase flow on a discretized temporal and spatial grid is however a challenging task because of the large scale separation that characterizes these flows: scales range from the largest flow scale (of the order of the domain size), down to the Kolmogorov scale of turbulence and further down to the molecular scale of the interface. This has direct implications on the description of multiphase turbulence as the spatio-temporal resolution one can reasonably afford is limited by computing capabilities [20]. Specifically, as done for single-phase turbulence [27, 41, 56], it would be beneficial to perform simulations in which all scales are directly resolved, without any model. However, this approach cannot be applied to multiphase flows, since the scale separation between the largest flow scale and the smallest interfacial scale is about eight to nine orders of magnitude, while the most recent high-performance computing (HPC) infrastructures, can handle a maximum scale separation of about three to four orders of magnitude.

In the last 15 years, the use of graphics processor units (GPU) has played a key role in the development of modern HPC systems [38, 42] paving the way for big advances in many different fields [13, 16, 44, 50]. In computational fluid dynamics, a large amount of computing resources are usually required for complex flow problems, especially when direct numerical simulations of multiphase turbulence are involved. Thanks to the integration of GPUs in HPC centers, there is now the possibility to solve large-scale problems [2, 5, 8, 12, 49, 68]. However, with respect to code development for CPUs, programming for GPUs poses new challenges: i) CPUs and GPUs are characterized by different computing architectures; ii) Specific programming languages/models and compilers are required to generate code that can be executed on GPUs; iii) Performance, scaling and code portability across different vendors and infrastructures represent an open issue; iv) CPU and GPU memories are physically separated and efficient strategy for the memory management should be designed [18]. All these aspects render the development and optimization of algorithms on GPU-accelerated infrastructures an open challenge.

In this context, different approaches are available and they are characterized by different degrees of portability, adaptability, and performance. The most commonly used are: i) Compute Unified Device Architecture (CUDA) [40], an extension of Fortran and C/++ programming languages to offload computations on GPUs and to manage memory transfers; ii) Open Computing Language (OpenCL) [65], an open, royalty-free standard for cross-platform, parallel programming of diverse accelerators; iii) OpenMP [10] and OpenACC [21], directives-based models that use directives to define data movement and computation offloading; iv) Library-based solution (e.g. Kokkos [14]), which offers high-level abstract programming concepts for performance portability on different types of high-performance computing infrastructures; v) Standard language parallelism, a parallel programming model that allows for accelerating C++ and Fortran codes without using language extensions or additional libraries. Each of these approaches is characterized by its own advantages and disadvantages from the point of view of the performance, portability, maintenance, short- and long-term programming language compatibility and maintenance, compiler requirements, and future developments that one should carefully evaluate before starting the GPU-porting phase of a code. With the idea of having a single code that can be executed on different architectures, we choose here to follow the directive-based approach offered by OpenACC. Among the available approaches, this approach has the following advantages: i) A single code has to be maintained: OpenACC directives will be ignored when GPU support is not enabled; ii) In the framework of the Nvidia HPC Software Development Kit (SDK) it allows for the use of the managed memory feature, which greatly simplifies data transfer between CPU and GPU memories; iii) Additional features, like the solution of new governing equations, can be easily implemented as the development time required to port to GPUs new code sections is reduced (compared to the other available approaches).

In this work, the OpenACC programming model is applied to a code for direct numerical simulation of multiphase turbulence based on a pseudo-spectral method [23]. In particular, the code relies on a direct solution of the Navier-Stokes equations coupled with a phase-field method to describe the interface shape and its topological changes. The parallelization is based on a multilevel approach (MPI + X). A first level relies on MPI and FFTW libraries [**?**] and it is employed when the code is executed in CPU-based computing infrastructures. A second level of parallelism that employs OpenACC directives and cuFFT libraries [39] is used to accelerate the code execution when GPU-based infrastructures are targeted. Thanks to this combination of OpenACC directives and cuFFT libraries, all the most computationally intensive sections of the code can be offloaded to GPUs. Overall, when GPUs are used, this leads to a remarkable speed-up and thus a reduction of the wall-clock time required for a time step (with respect to when CPUs are used).

The paper is organized as follows. In Section 2, the governing equations are presented; in Section 3 the numerical method is detailed. Then, in Section 4, the GPU-porting and the code performance are detailed and a demo simulation is presented. Finally, conclusions are drawn in Section 5.

2. Methodology

2.1. Governing equations

To describe the dynamics of the system, direct numerical simulation (DNS) of the Navier-Stokes equations, used to describe the flow field, are coupled with a phase-field method (PFM), used to describe interfacial phenomena. These equations are solved in a plane channel geometry: the streamwise and spanwise directions are periodic while along the wall-normal directions no-slip or free-slip, or a combination of them, can be applied.

2.1.1. Phase-field method

The phase-field method (PFM) is an interface-capturing method used for the description of multiphase flows. The method is based on the introduction of a marker function – the phase-field variable ϕ – that is uniform in the bulk of the phases ($\phi = \pm 1$) while it varies smoothly over the thin transition layer that separates the two phases [3, 24, 54]. The time evolution of the phase-field variable is here described by the Cahn-Hilliard (CH) equation which, in dimensionless form, reads as:

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = \frac{1}{Pe} \nabla^2 \mu \,, \tag{1}$$

where $\mathbf{u} = (u, v, w)$ is the velocity vector, μ is the chemical potential. The Péclet number, Pe, represents the ratio between the diffusive timescale, $h^2/\mathcal{M}\beta^2$, and the convective time scale, h/u_τ:

$$Pe = \frac{u_\tau h}{\mathcal{M}\beta} \,, \tag{2}$$

where $u_\tau = \sqrt{\tau_w/\rho}$ is the friction velocity (with τ_w the wall shear-stress and ρ the density), h is the half-channel height, $\mathcal{M}$ is the mobility parameter, and β is a positive constant introduced in the dimensionless procedure.

The chemical potential, μ, is obtained as the variational derivative of the Ginzburg-Landau free-energy functional [4, 24, 60, 66]. To model the case of two immiscible fluids, the free-energy functional, $\mathcal{F}[\phi, \nabla\phi]$, is composed by the sum of two different contributions [4, 24]:

$$\mathcal{F}[\phi, \nabla\phi] = \int_\Omega \left(f_0 + f_{mix} \right) d\Omega = \int_\Omega \underbrace{\frac{1}{4}(\phi^2 - 1)^2}_{f_0} + \underbrace{\frac{Ch^2}{2}|\nabla\phi|^2}_{f_{mix}}, \tag{3}$$

The first term, f_0 (bulk energy), identifies the tendency for the multiphase system to separate into two pure stable phases, while the second contribution, f_{mix} (mixing energy), is a non-local term accounting for the energy stored at the interface (i.e. surface tension). The Cahn number, $Ch = \epsilon/h$, represents the dimensionless thickness of the interfacial layer separating the two phases. By taking the functional derivative of the Ginzburg-Landau free-energy functional, we obtain the expression of the chemical potential:

$$\mu = \frac{\delta \mathcal{F}[\phi, \nabla\phi]}{\delta \phi} = \phi^3 - \phi - Ch^2 \nabla^2 \phi \,. \tag{4}$$

At equilibrium, considering that the chemical potential is constant throughout the domain ($\nabla\mu = \mathbf{0}$), the following (equilibrium) profile is obtained for a planar interface:

$$\phi_{eq} = \tanh\left(\frac{s}{\sqrt{2}Ch} \right) , \tag{5}$$

where s is the coordinate normal to the interface (located at $s = 0$).

2.1.2. Hydrodynamics

To describe the flow-field of the multiphase system, the Cahn-Hilliard equation (1) is coupled with the Navier-Stokes equations [15, 51, 63]. We consider here two incompressible and Newtonian phases and, for the sake of simplicity, we assume that they are characterized by equal density, ρ, and viscosity, μ. With these assumptions, the Navier-Stokes equations can be written as:

$$\nabla \cdot \mathbf{u} = 0\,, \tag{6}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \frac{1}{Re_\tau}\nabla^2 \mathbf{u} + \mathbf{f}_\sigma\,, \tag{7}$$

where p is the pressure-field and $\mathbf{f}_\sigma$ represents the surface tension forces. These forces are here computed using a continuum-surface stress approach [22, 30] :

$$\mathbf{f}_\sigma = \frac{3}{\sqrt{8}}\frac{Ch}{We}\nabla \cdot [\bar{\tau}_c]\,, \tag{8}$$

where $\bar{\tau}_c = |\nabla\phi|^2\mathbf{I} - \nabla\phi \otimes \nabla\phi$ is the Korteweg tensor used to model surface tension forces [28].

The dimensionless numbers that appear in the Navier-Stokes equation are the friction Reynolds number:

$$Re_\tau = \frac{\rho u_\tau h}{\eta}\,, \tag{9}$$

which represents the ratio between the inertial and viscous forces, and the Weber number:

$$We = \frac{\rho u_\tau^2 h}{\sigma}\,, \tag{10}$$

which represents the ratio between inertial and surface tension forces (being σ the surface tension).

2.2. Numerical method

The Navier-Stokes and continuity equations are not solved in primitive variables (velocity and pressure) but are recast in the so-called wall-normal velocity-vorticity formulation. This avoids solving a Poisson equation for pressure. Specifically, Navier-Stokes and continuity equations are replaced by a set of four equations [27, 62, 64]: i) A second-order equation for the wall-normal component of the vorticity; ii) A fourth-order equation for the wall-normal component of the velocity vector; iii) Continuity equation; iv) Definition of wall-normal vorticity. To obtain the set of governing equations for the wall-normal velocity-vorticity formulation, we can first rewrite the Navier-Stokes equations as follows:

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{S} - \nabla p + \frac{1}{Re_\tau}\nabla^2 \mathbf{u}\,, \tag{11}$$

where the term $\mathbf{S}$ contains all the non-linear terms present in the Navier-Stokes equations (e.g. convective terms, surface tension forces, etc.). By taking the curl of the Navier-Stokes equations, the pressure term vanishes thanks to the identity $\nabla \times \nabla p = 0$ and a transport equation for the vorticity vector, ω, is obtained:

$$\frac{\partial \omega}{\partial t} = \nabla \times \mathbf{S} + \frac{1}{Re_\tau}\nabla^2 \omega\,, \tag{12}$$

By taking again the curl of the vorticity transport equation, we obtain the following 4-*th* order equation for the velocity:

$$\frac{\partial \nabla^2 \mathbf{u}}{\partial t} = \nabla^2 \mathbf{S} - \nabla(\nabla \cdot \mathbf{S}) + \frac{1}{Re_\tau}\nabla^4 \mathbf{u}\,, \tag{13}$$

We solve here for the wall-normal components of the vorticity ω_z and velocity w. Hence, the following governing equations are solved:

$$\frac{\partial \omega_z}{\partial t} = \frac{\partial S_y}{\partial x} - \frac{\partial S_x}{\partial y} + \frac{1}{Re_\tau}\nabla^2 \omega_z\,, \tag{14}$$

$$\frac{\partial(\nabla^2 w)}{\partial t} = \nabla^2 S_z - \frac{\partial}{\partial z}\left(\frac{\partial S_x}{\partial x} + \frac{\partial S_y}{\partial y} + \frac{\partial S_z}{\partial z}\right) + \frac{1}{Re_\tau}\nabla^4 w\,, \tag{15}$$

complemented by the continuity equation (6) and the definition of wall-normal vorticity:

$$\omega_z = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\,. \tag{16}$$

The Cahn-Hilliard equation, which is a fourth-order equation, is split into two second-order equations [4]. In particular, the CH equation is first rewritten in the following way:

$$\frac{\partial \phi}{\partial t} = S_\phi + \frac{sCh^2}{Pe_\phi}\nabla^2\phi - \frac{Ch^2}{Pe_\phi}\nabla^4\phi\,, \tag{17}$$

where S_ϕ represents the contribution of the non-linear terms. The operator splitting $\nabla^2\phi = \nabla^2\phi(sCh^2 + 1) - sCh^2\nabla^2\phi$ is then applied [4] where the positive coefficient s has been chosen considering the temporal discretization.

The governing equations (6)-(14)-(15)-(16)-(17) are solved using a pseudo-spectral method: the variables are transformed from the physical into the wavenumber space. Along the periodic directions (x and y), all the quantities are expressed by Fourier series while they are represented by Chebyshev polynomials along the non-homogeneous wall-normal direction. The corresponding $N_x \times N_y \times N_z$ collocation points are equally spaced along the x and y directions while they are stretched along the wall-normal direction where a finer grid resolution is obtained near the two walls. Thanks to the adoption of a pseudo-spectral discretization, the equations are decoupled along the two periodic directions. In this way, a series of 1D Helmholtz independent problems along the wall-normal directions are obtained. All calculations are carried out in the wavenumber space except the non-linear terms, which are first computed in the physical space and then transformed back to spectral space (pseudo-spectral method). This avoids the evaluation of (computationally expensive) convolutions in the wavenumber space [23, 48]. The governing equations are discretized in time using an IMplicit-EXplicit (IMEX) scheme, in which the non-linear terms are integrated with an Adams-Bashfort scheme, while the linear terms are integrated by a Crank-Nicolson (Navier-Stokes) or by an implicit Euler (Cahn-Hilliard) scheme.

To advance the velocity and phase-field variables from time step n to time step $n + 1$, the following intermediate steps are performed (hat notation is used to identify the variable representation in the wavenumber space).

i) The velocity field, $\mathbf{u}^n$ and the phase-field, ϕ^n, are initialized and variables are transformed in the spectral space, $\hat{\mathbf{u}}^n$ and $\hat{\phi}^n$ (where hat notation is used in the following to identify the spectral representation of the variables).
ii) The non-linear convective and surface tension terms, i.e. the term $\mathbf{S}$, are computed. As these terms are non-linear, they are computed in the physical space rather than in the spectral space via convolution operation.
iv) Continuity and Navier-Stokes equations are solved (using the wall-normal velocity-vorticity formulation) to obtain the new velocity field, $\hat{\mathbf{u}}^{n+1}$. This requires the solution of a system of Helmholtz problems along the wall-normal direction for each (x, y) location.
vi) The non-linear terms present in the Cahn-Hilliard equation are computed, i.e. the term S_ϕ is formed. Also in this case, the non-linear terms are computed in the physical space.
v) The Cahn-Hilliard equation is solved and the new value of the phase-field, $\hat{\phi}^{n+1}$, is obtained. Similar to the solution of the Navier-Stokes equations, this requires the solution of two Helmholtz problems along the wall-normal direction for each (x, y) location.

3. Implementation

This numerical scheme has been implemented in a parallel Fortran 2003 MPI in-house proprietary code (FLOW36). The code is written using Fortran 2003 and the main parallelization backbone relies on an MPI paradigm. On top of the MPI backbone, OpenACC directives and cuFFT libraries are used to accelerate the code execution on GPU-based computing infrastructures. The code has been largely used in the past for the investigation of a wide range of complex turbulent flows: particle-laden flows [36], stratified turbulent flows [69], interface-resolved simulations of drop- and bubble-laden flows [61, 34, 54], drag reduced flows [55]. The code is developed using a modular approach: conditional compilation directives are used to enable (or disable) the different governing equations/physics available.

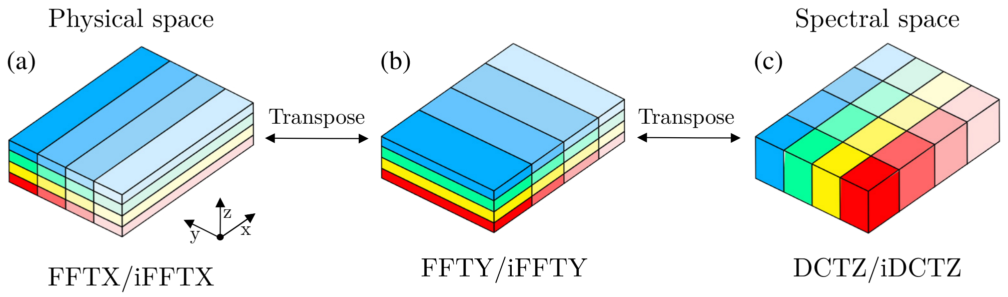


Fig. 1. 2D domain decomposition employed for the first level of parallelization (MPI). Each color corresponds to a different MPI task. In physical space, the domain is divided along the y and z directions (pencils orientated along the x-direction), while in spectral space it is divided along the x and y directions (pencil orientated along the z-direction). Transpositions (i.e. loops of MPI communications) are required to change the pencil orientation and to compute the transform along the various directions: along x in the first configuration shown in panel a, along y in the configuration shown in panel b, and along z in the configuration shown in panel c.

3.1. First level of parallelization: MPI

The parallelization backbone of the code relies on an MPI approach; the overall workload is divided among the different MPI tasks using a 2D domain decomposition (pencil decomposition). Within this strategy, the whole domain is split in so-called pencils: the domain is divided along two out of three directions and each sub-domain is assigned to a different MPI process. In physical space, the domain is divided along the y and z directions (pencils oriented along the x-direction), while in modal space it is divided along the x and y directions (pencils oriented along the z-direction). This change in the pencil orientation is needed when taking the transforms: to compute the Fourier or Chebyshev transforms each process must hold all the points in the transform direction. Thus, when in physical space, at first Fourier transforms are taken in the x direction (Figure 1a), then the parallelization changes in order to have all the points along the y direction. The domain is divided between the x and z directions when taking the Fourier transforms along y (Figure 1b). At this point, each $x - y$ plane holds all the Fourier modes at a certain height. Then the parallelization is again changed, switching to a domain decomposition along x and y directions, thus each MPI process holds all the points in z direction at a certain (x, y) location (parallelization in modal space). Finally, Chebyshev transforms are taken in the z direction (Figure 1c).

The only MPI communications occur when the non-linear terms of the governing equations are computed. These terms are computed in the physical space and thus a change of pencil orientation is required to compute the transforms along the three directions (from spectral to physical and vice-versa). After the calculation of all the non-linear terms, a system of Helmholtz problems along the wall-normal direction (z) is solved at each (x, y) location independently.

3.2. Second level of parallelization: OpenACC directives

On top of the MPI parallelization scheme presented above, cuFFT libraries and OpenACC directives are used to accelerate the code execution. When CPU-based architectures are employed, each MPI task is assigned to a physical core. Differently, when GPU-based architectures are used, each MPI task is assigned to a specific GPU. All the computationally intensive operations are performed on the GPUs. Specifically, the cuFFT libraries are used to perform all the transforms (Fourier and Chebyshev) and the entire solver can be efficiently executed on GPUs thanks to the fine-grain parallelism offered by the numerical scheme (series of 1D independent problems along the wall-normal direction). To obtain a code that can be easily ported between CPU- and GPU-based architectures, we employ the managed memory model present in OpenACC (which exploits the CUDA unified memory feature). As in most systems CPU and GPU memories are physically separated, the use of GPUs usually requires explicit memory transfers and the handling of the copies shared between CPU and GPU memories. Differently, using the managed memory feature, there is no need to explicitly define each memory transfer between the CPU (host) and the GPU (device). In particular, memory can be accessed using a single pointer from CPU or GPU code. All memory transfers are managed by the compiler that makes the decisions about when and how to move data from the host to the device and back. The use of the managed memory model should be specified via the `-gpu=managed` compiling option. Although its use greatly simplifies code optimization, GPU-porting, and implementation of new features, one should bear in mind the three main drawbacks of this approach: i) The actual bandwidth of the memory transfer between CPU and GPU is slightly reduced; this is due to the virtual memory page faulting and migration used by the compiler to migrate data; ii) Data cannot be transferred asynchronously (note that asynchronous compute regions can be created); iii) This feature requires the use of Nvidia compiler and is at the moment non-compatible with GPUs from different vendors.

3.2.1. Fourier and Chebyshev transforms: cuFFT libraries

To accelerate the execution of Fourier and Chebyshev transforms, these operations are performed using the highly optimized and scalable cuFFT libraries. This CUDA-based library is optimized for Nvidia GPUs and offers an interface equivalent to that employed by the FFTW library (for CPUs) [17]. In FLOW36, conditional compilation flags are used to compile the code with the correct library (FFTW when using CPUs and cuFFT when using GPUs). To perform a transform, three main steps are required: i) Creation of a plan; this operation set-ups the library and identifies the best algorithm to be used depending on the available resources; ii) Execution of the transform according to the pre-defined plan; iii) Destruction of the plan. Plans can be also stored so that All transforms are executed using the batched version of the library (specified by the plan): this means that the library is set to perform a certain number of transforms along one direction and the input data is coherent with the advanced data layout structure. For instance, considering the forward transform along the x direction, $N_y \times N_z$ transforms of a signal having length N_x are performed. The transform can be then executed as follows by invoking the cuFFT library via OpenACC. For instance, considering the forward transform along x, this operation can be performed as follows:

```
!$acc data copyin(input) copyout(output)
!$acc host_data use_device(input,output)
gerr=gerr+cufftExecD2Z(plan,input,output)
!$acc end host_data
!$acc end data
```

where data clauses are required to force the data movement between CPU and GPU memories when libraries are used. The suffix `D2Z` defines the type of transform: as here we consider the forward transform along x, the input is an array of double (`D`) while the output is an array of complex numbers (`Z`), i.e. a real-to-complex transform. The above-mentioned strategy can be straightforwardly applied to the Fourier transform along x and y, which correspond to the first and third index of a generic 3D input array.

A different approach is required to perform the Chebyshev transforms along the wall-normal direction z. Two minor issues arise here: i) The cuFFT libraries do not explicitly support real-to-real transforms; ii) We need fo perform a transform along the direction specified by the inner index of a matrix and this does not allow for the use of the batched mode of cuFFT. These two issues can be easily overcome by manipulating the input and output vector so that a Chebyshev transform is performed and by transposing (before and after the transform) the input vector so that it satisfies the advanced data layout. In particular, the Chebyshev transform can be performed using FFT algorithms as it belongs to the group of discrete sine/cosine transforms (specifically, a DCT-I). The Chebyshev transform of an N-point real signal can be obtained by taking the discrete Fourier transform of a 2N-point even extension of the signal [32]. This scheme is applied to both the real and imaginary parts of the input signal and for both forward and backward transforms (the backward transform of a DCT-I transform is a DCT-I [57]). Note indeed that the input is a complex array because the FFTs along the x and y directions have been already performed at this stage, see figure 1c. Although the present strategy involves two transposes (back-and-forth) for each transform, the possibility to perform the Chebyshev transform in batched mode outperforms the other possible approach, i.e. a recursive loop of single 1D transforms, by one order of magnitude in terms of performance. Further optimization of the Chebyshev transform step can be obtained using the strategy proposed by Makhoul [32], which does not require performing a transform of size $2N$ but only of size N by re-ordering the input signal.

3.2.2. Solver execution: OpenACC directives

The computation of the non-linear terms in physical space, which mainly requires matrix multiplications, has been offloaded to GPUs by means of OpenACC directives. In particular, considering the computation of the non-linear terms present in both the Navier-Stokes and Cahn-Hilliard equations, this operation can be straightforwardly parallelized on GPUs as it mainly involves element-wise product (Hadamard product). Indeed, all the flow-field and phase-field variables are collocated and thus defined on the same grid points. For instance, the non-linear term $a = bc$ (where a, b, c are 3D arrays) can be computed as follows:

```
!$acc kernels
do i=1,nx
 do j=1,nyp
  do k=1,nzp
   a(i,k,j)=b(i,k,j)*c(i,k,j)
  end do
 end do
end do
!$acc end kernels
```

where `nx, nyp` and `nzp` are the pencil dimensions in physical space. For this part of the code, the use of the `kernels` construct or `parallel loop` construct exhibit very similar performance. In addition, simple assignment operations (e.g. during time integration) are also performed using the `kernels` construct; this avoids expensive back-and-forth copies from CPU to GPU memories and vice-versa, which are limited by the available bandwidth between host and device.

Finally, the solution of the systems of Helmholtz problems is also offloaded to GPUs. This step requires solving a series of tridiagonal systems using Gauss elimination along the wall-normal direction for each (x, y) position. Thus, the only direction along which dependencies are present is the wall-normal direction while it can be parallelized along the x and y directions of the pencil. Thanks to the adoption of the managed memory feature, the offloading to GPUs of this part of the code is rather straightforward and does not pose particular challenges or require extensive code modifications with respect to the CPU version.

3.2.3. MPI communications

Once all the transforms and solver execution have been ported to the GPUs, as illustrated above, to improve code scalability, MPI communications can be also optimized. Specifically, in traditional MPI installations (Figure 2), considering a message between rank 0 and rank 1, the data computed on GPU 0 has to be staged into the host memory of CPU 0 (green and blue arrows) and then sent to the MPI process 0 via MPI message and, in turn, to the GPU 1 (assigned to rank 1), as shown in Figure 2. These additional passages represent a big overhead and can negatively impact strong and weak scalability results, especially when large-scale simulations are performed (i.e. using more than 100 GPUs) and many non-linear terms have to be computed (which require pencil transpositions and thus loop of MPI communications). A possible solution to this problem is the use of CUDA-aware MPI implementations (or more in general GPU-aware).

To address this problem, we take advantage of the CUDA-aware MPI. CUDA-aware MPI implementations allow for the use of GPUDirect, a group of technologies that provide high-bandwidth, low-latency communications for all kinds of inter-rank communication (intra-node, inter-node, and inter-node) via Remote Direct Memory Access (RDMA) when Nvidia GPUs are used. In particular, when GPUDirect RDMA is available the GPU buffer can be directly moved to the network without passing through the host memory at all. So the data is transferred from the GPU buffer of the MPI Rank 0 to the GPU buffer of MPI Rank 1 (as represented by the long solid arrow in the upper part of the figure). The use of CUDA-aware MPI features requires however the use of pinned buffers. As here data movement is implicitly handled via the use of managed memory, to exploit CUDA-aware features, CUDA Fortran instructions are required to define the buffers used to perform the communications as pinned (thus inhibiting the memory paging performed by the managed memory feature). This can be specified via the `pinned` attribute in the variable declarations and enabling the support for CUDA among the compiling options. Then, via OpenACC directives, the use of CUDA-aware features can be specified as follows:

```
!$acc data copy(bufs,bufr)
!$acc host_data use_device(bufs,bufr)
call mpi_isend(bufs,.....)
call mpi_irecv(bufr,......)
```

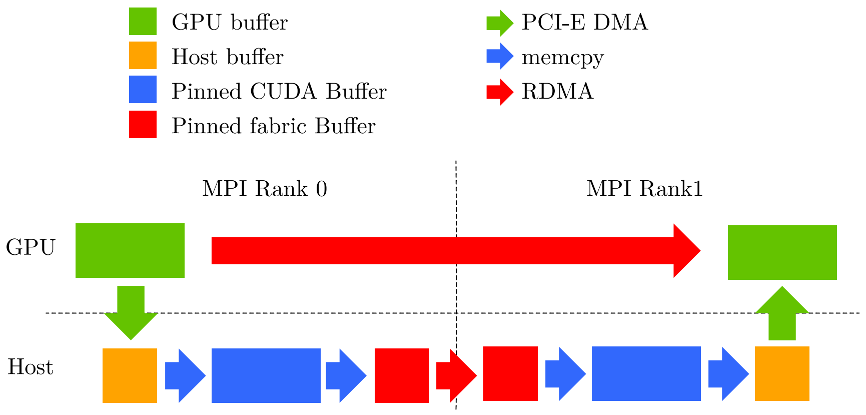


Fig. 2. Schematic showing the step required to perform an MPI communication between data stored in the buffer of GPU 0 and GPU 1. In standard MPI installations, the data computed on GPU 0 has to be moved into host memory of CPU 0 and then sent to the MPI process 0 and, in turn, transferred to the GPU 1. Using a CUDA-aware MPI implementation that exploit GPUDirect technologies, e.g. Remote Direct Memory Access (RDMA), a direct transfer between GPU 0 and GPU 0 can be performed (long red arrow). Reproduced from [29].

```
call mpi_waitall(.....)
!$acc end host_data
!$acc end data
```

where `bufr` and `bufs` are the two GPU pinned buffers used for the MPI communications. It is convenient to allocate these buffers during code start-up as the allocation/deallocation of pinned buffers has a non-negligible overhead. Also, it is worth mentioning that the use of non-blocking MPI communications is advisable as it allows the MPI library to build more efficient pipelines. Finally, as the code relies on the managed memory feature (i.e. CUDA Unified memory), the chosen MPI implementation should support the CUDA unified memory, e.g. OpenMPI and MVAPICH-2 [19, 43]. Indeed, the MPI implementation needs to know where the data is located and which is the optimal strategy to perform the communication [35].

3.3. Optimization, performance and scaling

Thanks to the use of OpenACC directives, the code has been ported to GPU using a step-by-step approach. First, all the Fourier and Chebyshev transforms, which represent the most computationally intensive part of the algorithm and then the solver and the CUDA-aware MPI communications have been ported to GPUs. During each of these steps, the code has been optimized and profiled using the tools provided by the NVIDIA Nsight Systems and using the NVIDIA Tools Extension (NVTX) instructions. Specifically, NVTX instructions have been used to annotate the profiler timeline with events and ranges. This procedure allows to find, and possibly remove, performance bottlenecks as well as to improve specific sections of the code.

To evaluate the performance of the GPU-porting, we first consider a single-node run where all the GPUs present in the node are used. This allows us to obtain a first estimate of the gain that can be obtained by using GPUs, compared to using all the physical cores available on the CPU counterpart. It is also possible to compare the results keeping fixed the number of MPI tasks used; however, this would underestimate CPU performance as only a part of the available cores will be used: the number of GPUs present in a computing node (from 4 to 8) is usually smaller than the number of cores available (from 16 up to 256). For this test case, we consider a single-phase turbulent channel flow solved on a grid having $N_x \times N_y \times N_z = 256 \times 256 \times 257$ grid points. Tests have been performed on four different machines, one local server, two machines present at CINECA (Marconi-100 and Leonardo [67]) and the CPU partition of LUMI (LUMI-C). This latter machine is included as a reference of the performance that can be obtained in a CPU-based computing infrastructure. The technical specifications of the four machines are reported in Table 1. When CPUs are used, the maximum number of cores available is used whereas when GPUs are employed, the number of MPI tasks

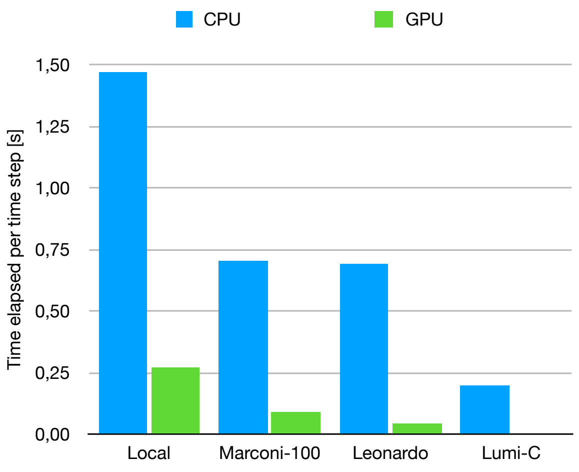


Fig. 3. Time elapsed per time step on different machines using all the physical cores available (blue) and all the GPUs available (green) on a single-node. The results have been obtained considering a single-phase turbulent channel flow and a grid resolution equal to $N_x \times N_y \times N_z = 256 \times 256 \times 257$. For all cases, the code has been compiled using the Nvidia Fortran compiler `nvfortan` with or without the support for GPU-acceleration, depending on the case considered (CPU or GPU). When the `nvfortan` compiler is not available (LUMI-C), the code has been compiled using `gfortran`.

is set equal to the number of GPUs. For all tests, the code is compiled with the Nvidia Fortran compiler `nvfortran` when available and when not with the GNU compiler `gfortran`.

Table 1. Technical details of the computing infrastructures employed in the tests

System	CPU	GPU
Local Server	1 x Intel Xeon 5218 16 cores	2 x Quadro RTX 5000 16 GB
Marconi-100 (CINECA)	2 x IBM POWER9 AC922 16 cores	4 x NVIDIA Volta V100 16 GB
Leonardo (CINECA)	1 x Intel Xeon 8358 32 cores	4 x Nvidia Ampere A100 64GB
LUMI-C (LUMI)	2 x AMD EPYC 7763 64 cores	-

The time elapsed for a single time step is shown in Figure 3 for the different machines and cases considered. Results obtained from the CPU runs are reported in blue while those obtained from GPU runs are in green. In general, it can be observed that when the full node performance is compared, i.e. all physical cores versus all available GPUs, using the GPUs a consistent speed-up is obtained. The specific value of the speed-up obtained depends on the machine considered: generally speaking, this value ranges from 4 (local server) up to 16 (Leonardo). This variation is due to the different performance offered by the CPUs and GPUs present in the computing node. Comparing the results with the CPU runs performed on LUMI-C, it is interesting to observe that the wall-clock time elapsed per time step obtained using GPUs is smaller.

We now move to analyze the strong scaling results. In particular, Figure 4 shows the results obtained employing Marconi-100, please refer to Table 1 for the technical details of the cluster. We consider two different grid resolutions, typical of production runs for direct numerical simulations of multiphase turbulence: $N_x \times N_y \times N_z = 512 \times 512 \times 513$ and $N_x \times N_y \times N_z = 1024 \times 1024 \times 1025$. For the coarser grid resolution (blue dots), tests have been performed starting from 1 node (4 GPUs) up to 32 nodes (128 GPUs) while, for the finer grid resolution (red dots), starting from 8 nodes (32 GPUs) up to 64 nodes (256 GPUs). A different number of nodes has been used for the two problem sizes because of the different memory requirements. We can observe that in general good results are obtained for both the problem sizes considered. For the smaller problem size, strong scaling results start to worsen when more than 16 nodes (64 GPUs) or more are used. This is due to the lower computational load per pencil/slab and the relative cost of MPI communications. Indeed, by increasing the problem size (i.e. considering the finer grid resolution), the computational load per pencil/slab increases and the cost of the MPI communications can be hidden; as a consequence, better results

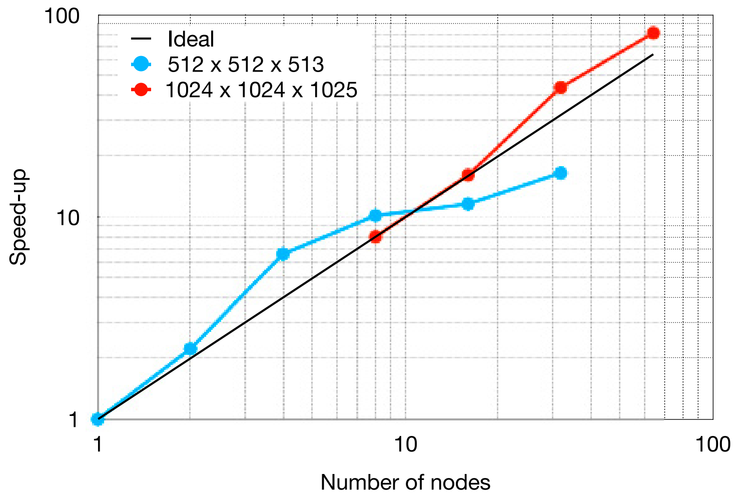


Fig. 4. Strong scaling results for the code FLOW36 obtained on Marconi100. Two different problems sizes have been considered: $N_x \times N_y \times N_z = 512 \times 512 \times 513$ and $N_x \times N_y \times N_z = 1024 \times 1024 \times 1025$. For the smaller problem size (blue dots), tests have been performed starting from 1 node (4 GPUs) up to 32 nodes (128 GPUs) while, for the larger problem size (red dots), starting from 8 nodes (32 GPUs) up to 64 nodes (256 GPUs).

in terms of scaling are obtained. Future developments are required to improve code scalability on a larger number of GPUs as performing large-scale 3D FFT require a significative amount of communications [6, 9, 11, 46, 52].

4. Code capabilities

We provide here an example of the code capabilities. In particular, we consider the injection of a swarm of large and deformable drops released in a turbulent channel flow, a flow configuration similar to the one adopted in previous works [33, 53, 60]. The computational domain is a closed channel with dimensions $L_x \times L_y \times L_z = 4\pi h \times 2\pi h \times 2h$ corresponding to $L_x^+ \times L_y^+ \times L_z^+ = 3770 \times 1885 \times 600$ wall units. Equations are discretized on a grid with $N_x \times N_y \times N_z = 2048 \times 1024 \times 1025$ collocation points. The simulation is performed at a fixed shear Reynolds number of $Re_\tau = 300$ and the Weber number has been set equal to $We = 6$. The simulation starts by releasing 256 spherical droplets in a flow field previously obtained from a single-phase flow simulation of a fully developed turbulent channel flow. After an initial transient, where drops start to break and coalesce following complex dynamics, a new equilibrium situation is reached in which a balance between coalescence and breakage events is attained. Figure 5 shows a qualitative rendering of the steady-state configuration attained by the system at $t^+ = 3000$. The flow moves from left to right (along the streamwise direction x) and a top view of the system is shown (streamwise-horizontal; spanwise-vertical). The interface of the drops is identified as the iso-contour $\phi = 0$ and is rendered using a ray tracing algorithm. We can observe the wide range of scales and shapes that characterize the drops: from very small and almost undeformed spherical droplets to very large drops characterized by a complex three-dimensional shape. It is also worth noticing the presence of elongated filaments/threads that will eventually lead to the formation of small drops.

5. Conclusions and future developments

We detailed the GPU-porting of a pseudo-spectral code tailored towards large-scale simulations of multiphase flow, more specifically interface-resolved direct numerical simulations. The code relies on direct numerical simulation of turbulence coupled with a phase-field method to describe the interface shape, deformation, and topological changes. The development of the code has been designed with the specific goal of simplifying code maintenance, obtaining a portable code that can exploit GPU acceleration, and facilitating the implementation of new modules and new physics (e.g. additional governing equations). To achieve this ambitious goal, we rely on two levels of parallelism: i) A first level that relies on MPI and a 2D domain decomposition to divide the workload among the MPI tasks,; ii) A second

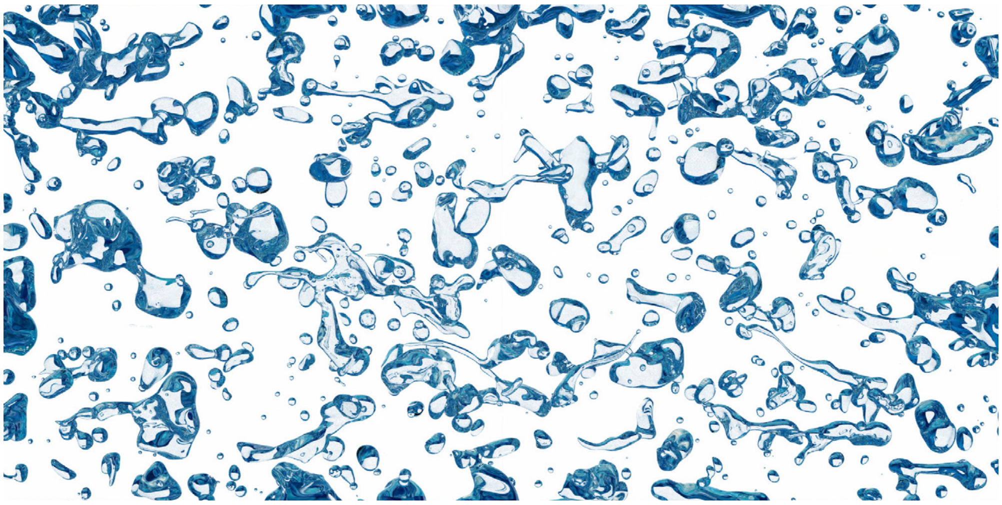


Fig. 5. Top view of a swarm of large and deformable drops released in a turbulent channel flow. The flow moves from left to right (along the streamwise direction) and drops coalesce and break under the action of turbulence fluctuations. The interface of the drops is identified as the iso-contour $\phi = 0$. The grid resolution employed for this demo simulation is $N_x = 2048 \times 1024 \times 1025$. This run has been executed using 64 nodes (256 GPUs) of Leonardo [67].

level that relies on OpenACC directives and CUDA libraries (cuFFT) to accelerate code execution on GPU-based computing infrastructures. The code makes use of the managed feature (CUDA Unified memory) to simplify code maintenance and avoid the explicit definition of data transfers between CPU and GPU memories [31, 37] In addition, pinned memory buffers are used to perform MPI communications so that the GPU Direct technologies can be employed to improve latency and data transfer times. The main limitation of the employed approach is the compatibility with GPUs from other vendors as the managed memory feature is currently available only on the Nvidia HPC-SDK environment. However, a similar feature has been recently released in the context of AMD GPU architectures [7, 26]. Hence, in future developments, we plan to extend the support also to AMD GPUs by exploiting the unified memory feature present in the AMD ecosystem and combining it with openMP directives and rocFFT libraries to accelerate the code execution (solver and transforms) [1].

Acknowledgements

The author would like to thank Giovanni Soligo, who developed the CPU-version of the present code. We acknowledge the CINECA award under the ISCRA initiative, for the availability of high-performance computing resources and support (project HP10CH4PTZ and HP10BUJEO5). We also acknowledge the EuroHPC Joint Undertaking for awarding this project access (PRACE-DEV-2022D01-069) to the EuroHPC supercomputer Marconi-100 and Leonardo, hosted by CINECA (Italy). The authors gratefully acknowledge financial support from the European Union–NextGenerationEU.

References

[1] AMD, . rocFFT library. URL: https://github.com/ROCmSoftwarePlatform/rocFFT.

[2] Ames, J., Puleri, D.F., Balogh, P., Gounley, J., Draeger, E.W., Randles, A., 2020. Multi-gpu immersed boundary method hemodynamics simulations. J. Comput. Sci. 44, 101153.

[3] Anderson, D., McFadden, G., Wheeler, A., 1998. Diffuse interface methods in fluid mechanics. Annu. Rev. Fluid Mech. 30, 139–165.

[4] Badalassi, V., Ceniceros, H., Banerjee, S., 2003. Computation of multiphase systems with phase field models. J. Comput. Phys. 190, 371–397.

[5] Bernardini, M., Modesti, D., Salvadore, F., Sathyanarayana, S., Della Posta, G., Pirozzoli, S., 2023. STREAmS-2.0: Supersonic turbulent accelerated navier-stokes solver version 2.0. Comput. Phys. Commun. 285, 108644.

[6] Chatterjee, A.G., Verma, M.K., Kumar, A., Samtaney, R., Hadri, B., Khurram, R., 2018. Scaling of a fast fourier transform and a pseudo-spectral fluid solver up to 196608 cores. J. Parallel. Distr. Com. 113, 77–91.

[7] Chu, H., 2013. AMD heterogeneous uniform memory access. Proceedings of the APU 13th developer summit , 11–13.

[8] Costa, P., Phillips, E., Brandt, L., Fatica, M., 2021. GPU acceleration of CaNS for massively-parallel direct numerical simulations of canonical fluid flows. Comput. Math. Appl. 81, 502–511.

[9] Czechowski, K., Battaglino, C., McClanahan, C., Iyer, K., Yeung, P.K., Vuduc, R., 2012. On the communication complexity of 3D FFTs and its implications for exascale, in: Proceedings of the 26th ACM international conference on Supercomputing, pp. 205–214.

[10] Dagum, L., Menon, R., 1998. Openmp: an industry standard api for shared-memory programming. Computational Science & Engineering, IEEE 5, 46–55.

[11] Dalcin, L., Mortensen, M., Keyes, D.E., 2019. Fast parallel multidimensional FFT using advanced MPI. J. Parallel. Distr. Com. 128, 137–150.

[12] De Vanna, F., Avanzi, F., Cogo, M., Sandrin, S., Bettencourt, M., Picano, F., Benini, E., 2023. URANOS: A GPU accelerated navier-stokes solver for compressible wall-bounded flows. Comput. Phys. Commun. 287, 108717.

[13] Dematté, L., Prandi, D., 2010. GPU computing for systems biology. Brief. Bioinform. 11, 323–333.

[14] Edwards, H.C., Trott, C., Sunderland, J., 2014. Kokkos: Enabling manycore performance portability through polymorphic memory access patterns. J. Parallel Distrib. Comput. 74, 3202–3216.

[15] Elghobashi, S., 2019. Direct numerical simulation of turbulent flows laden with droplets or bubbles. Annu. Rev. Fluid Mech. 51, 217–244.

[16] Friedrichs, M.S., Eastman, P., Vaidyanathan, V., Houston, M., Legrand, S., Beberg, A.L., Ensign, D.L., Bruns, C.M., Pande, V.S., 2009. Accelerating molecular dynamic simulation on graphics processing units. J. Comput. Chem. 30, 864–872.

[17] Frigo, M., Johnson, S.G., 2005. The design and implementation of FFTW3. Proc. IEEE 93, 216–231.

[18] Fujii, Y., Azumi, T., Nishio, N., Kato, S., Edahiro, M., 2013. Data transfer matters for GPU computing, in: 2013 ICPDS Conference on Parallel and Distributed Systems, IEEE. pp. 275–282.

[19] Gabriel, E., Fagg, G.E., Bosilca, G., Angskun, T., Dongarra, J.J., Squyres, J.M., Sahay, V., Kambadur, P., Barrett, B., Lumsdaine, A., et al., 2004. Open MPI: Goals, concept, and design of a next generation MPI implementation, in: Recent Advances in Parallel Virtual Machine and Message Passing Interface: 11th European PVM/MPI Users' Group Meeting Budapest, Hungary, September 19-22, 2004. Proceedings 11, Springer. pp. 97–104.

[20] Gorokhovski, M., Herrmann, M., 2008. Modeling primary atomization. Annu. Rev. Fluid Mech. 40, 343–366.

[21] working group, O., et al., 2011. The OpenACC application programming interface. Retrieved March 26, 2019.

[22] Gueyffier, D., Li, J., Nadim, A., Scardovelli, R., Zaleski, S., 1999. Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows. J. Comput. Phys. 152, 423–456.

[23] Hussaini, M., Zang, T., 1987. Spectral methods in fluid dynamics. Annu. Rev. Fluid Mech. 19, 339–367.

[24] Jacqmin, D., 1999. Calculation of two-phase Navier–Stokes flows using phase-field modeling. J. Comput. Phys. 155, 96–127.

[25] Jähne, B., Haußecker, H., 1998. Air-water gas exchange. Annu. Rev. Fluid Mech. 30, 443–468.

[26] Jin, Z., Vetter, J.S., 2022. Evaluating unified memory performance in HIP, in: 2022 IEEE International parallel and distributed processing symposium workshops (IPDPSW), IEEE. pp. 562–568.

[27] Kim, J., Moin, P., Moser, R., 1987. Turbulence statistics in fully developed channel flow at low Reynolds number. J. Fluid Mech. 177, 133–166.

[28] Korteweg, D., 1901. Sur la forme que prennent les equations du mouvements des fluides si l'on tient compte des forces capillaires causées par des variations de densité considérables mais continues et sur la théorie de la capillarité dans l'hypothèse d'une variation continue de la densité. Archives Néerlandaises des Sciences Exactes et Naturelles 6, 1–24.

[29] Kraus, J., . An introduction to CUDA-Aware MPI. URL: `https://developer.nvidia.com/blog/introduction-cuda-aware-mpi/`.

[30] Lafaurie, B., Nardone, C., Scardovelli, R., Zaleski, S., Zanetti, G., 1994. Modelling merging and fragmentation in multiphase flows with SURFER. J. Comput. Phys. 113, 134–147.

[31] Lindholm, E., Nickolls, J., Oberman, S., Montrym, J., 2008. Nvidia tesla: A unified graphics and computing architecture. IEEE micro 28, 39–55.

[32] Makhoul, J., 1980. A fast cosine transform in one and two dimensions. IEEE Trans. Acoust. 28, 27–34.

[33] Mangani, F., Roccon, A., Zonta, F., Soldati, A., 2024. Heat transfer in drop-laden turbulence. J. Fluid Mech. 978, A12.

[34] Mangani, F., Soligo, G., Roccon, A., Soldati, A., 2022. Influence of density and viscosity on deformation, breakage, and coalescence of bubbles in turbulence. Phys. Rev. Fluids 7, 053601.

[35] Manian, K.V., Ammar, A., Ruhela, A., Chu, C.H., Subramoni, H., Panda, D.K., 2019. Characterizing CUDA unified memory (um)-aware MPI designs on modern GPU architectures, in: Proceedings of the 12th Workshop on General Purpose Processing Using GPUs, pp. 43–52.

[36] Marchioli, C., Soldati, A., 2002. Mechanisms for particle transfer and segregation in a turbulent boundary layer. J. Fluid Mech. 468, 283–315.

[37] Negrut, D., Serban, R., Li, A., Seidl, A., 2014. Unified memory in CUDA 6.0. a brief overview of related data access and transfer issues. SBEL, Madison, WI, USA, Tech. Rep. TR-2014-09 .

[38] Nickolls, J., Dally, W.J., 2010. The GPU computing era. IEEE micro 30, 56–69.

[39] Nvidia, . cuFFT library. URL: `http://docs.nvidia.com/cuda/cufft`.

[40] NVIDIA, Vingelmann, P., Fitzek, F.H., 2020. CUDA, release: 10.2.89. URL: `https://developer.nvidia.com/cuda-toolkit`.

[41] Orlandi, P., 2000. Fluid flow phenomena: a numerical toolkit. volume 55. Springer Science & Business Media.

[42] Owens, J.D., Houston, M., Luebke, D., Green, S., Stone, J.E., Phillips, J.C., 2008. GPU computing. Proc. IEEE 96, 879–899.

[43] Panda, D.K., Tomko, K., Schulz, K., Majumdar, A., 2013. The MVAPICH project: Evolution and sustainability of an open source production quality MPI library for HPC, in: WSSSPE5.

[44] Pandey, M., Fernandez, M., Gentile, F., Isayev, O., Tropsha, A., Stern, A.C., Cherkasov, A., 2022. The transformational role of gpu computing and deep learning in drug discovery. Nat. Mach. Intell. 4, 211–221.

[45] Paul, E., Atiemo-Obeng, V., Kresta, S., 2004. Handbook of industrial mixing: science and practice. John Wiley & Sons.

[46] Pekurovsky, D., 2012. P3DFFT: A framework for parallel computations of fourier transforms in three dimensions. SIAM J. Comput. 34, C192–C209.
[47] Pereira, R., Ashton, I., Sabbaghzadeh, B., Shutler, J., Upstill-Goddard, R., 2018. Reduced air-sea CO_2 exchange in the Atlantic Ocean due to biological surfactants. Nat. Geosci. 11, 492–496.
[48] Peyret, R., 2002. Spectral methods for incompressible viscous flow. volume 148. Springer Science+Business Media.
[49] Pirozzoli, S., Romero, J., Fatica, M., Verzicco, R., Orlandi, P., 2021. One-point statistics for turbulent pipe flow up to. J. Fluid Mech. 926, A28.
[50] Pratx, G., Xing, L., 2011. GPU computing in medical physics: A review. Medical physics 38, 2685–2697.
[51] Prosperetti, A., Tryggvason, G., 2009. Computational Methods for Multiphase Flow. Cambridge University Press.
[52] Ravikumar, K., Appelhans, D., Yeung, P., 2019. GPU acceleration of extreme scale pseudo-spectral simulations of turbulence using asynchronism, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–22.
[53] Roccon, A., De Paoli, M., Zonta, F., Soldati, A., 2017. Viscosity-modulated breakup and coalescence of large drops in bounded turbulence. Phys. Rev. Fluids 2, 083603.
[54] Roccon, A., Zonta, F., Soldati, A., 2023. Phase-field modeling of complex interface dynamics in drop-laden turbulence. Phys. Rev. Fluids 8, 090501.
[55] Roccon, A., Zonta, F., Soldati, A., 2024. Turbulent drag reduction in water-lubricated channel flow of highly viscous oil. Phys. Rev. Fluids 9, 054611.
[56] Rogallo, R.S., Moin, P., 1984. Numerical simulation of turbulent flows. Annu. Rev. Fluid Mech. 16, 99–137.
[57] Saverin, J., 2023. Sailffish: A lightweight, parallelised fast poisson solver library. arXiv preprint arXiv:2301.01145 .
[58] Schramm, L.L., Stasiuk, E.N., Marangoni, D.G., 2003. Surfactants and their applications. Annu. Rep. Prog. Chem., Sect. C: Phys. Chem. 99, 3–48.
[59] Shaw, R.A., 2003. Particle-turbulence interactions in atmospheric clouds. Annu. Rev. Fluid Mech. 35, 183–227.
[60] Soligo, G., Roccon, A., Soldati, A., 2019a. Breakage, coalescence and size distribution of surfactant-laden droplets in turbulent flow. J. Fluid Mech. 881, 244–282.
[61] Soligo, G., Roccon, A., Soldati, A., 2019b. Breakage, coalescence and size distribution of surfactant-laden droplets in turbulent flow. J. Fluid Mech. 881, 244–282. doi:10.1017/jfm.2019.772.
[62] Soligo, G., Roccon, A., Soldati, A., 2019c. Coalescence of surfactant-laden drops by phase field method. J. Comput. Phys. 376, 1292–1311.
[63] Soligo, G., Roccon, A., Soldati, A., 2021. Turbulent flows with drops and bubbles: what numerical simulations can tell us – freeman scholar lecture. ASME J. Fluids Eng. 143, 080801–1.
[64] Speziale, C., 1987. On the advantages of the vorticity-velocity formulation of the equations of fluid dynamics. J. Comput. Phys. 73, 476 – 480.
[65] Stone, J., Gohara, D., Shi, G., 2010. OpenCL: A parallel programming standard for heterogeneous computing systems. J. Parallel Distrib. Comput. 12, 6–73.
[66] Tóth, G.I., Kvamme, B., 2015. Analysis of Ginzburg-Landau-type models of surfactant-assisted liquid phase separation. Phys. Rev. E 91, 032404.
[67] Turisini, M., Amati, G., Cestari, M., 2023. Leonardo: A pan-european pre-exascale supercomputer for HPC and AI applications. J. Large-scale Res. Fac. 9.
[68] Zhu, X., Phillips, E., Spandan, V., Donners, J., Ruetsch, G., Romero, J., Ostilla-Mónico, R., Yang, Y., Lohse, D., Verzicco, R., et al., 2018. AFiD-GPU: a versatile navier–stokes solver for wall-bounded turbulent flows on GPU clusters. Comput. Phys. Commun. 229, 199–210.
[69] Zonta, F., Onorato, M., Soldati, A., 2012. Turbulence and internal waves in stably-stratified channel flow with temperature-dependent fluid properties. J. Fluid Mech. 697, 175–203. doi:10.1017/jfm.2012.51.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 31–41

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Large eddy simulations (LES) of a three-element high-lift wing: Exploring the active flow control (AFC) capabilities

Ricard Montalà[a,*], Oriol Lehmkuhl[b], Ivette Rodriguez[a]

[a]*TUAREG Universitat Politècnica de Catalunya (UPC), Colom 11, Terrassa, 08222, Spain*
[b]*CASE Barcelona Supercomputing Center (BSC), Eusebi Güell 1-3, Barcelona, 08034, Spain*

Abstract

This study presents an aerodynamic analysis of the 30P30N high-lift three-element wing under varying the angle of attack ($\alpha = 5°$, $9°$, and $23°$) at a constant Reynolds number ($Re_c = 750,000$) by means of large eddy simulations (LES). Baseline results were validated against existing literature, demonstrating good agreement. The increase of the angle of attack entails higher values of lift but also, increased values of drag. At $\alpha = 23°$, a recirculation area appears above the flap, this being the signature of the stall conditions. This is related to the adverse pressure gradient that develops along the main suction side and the separation of the streamlines in its wake. Actuation strategies using synthetic jets were explored to mitigate this flow separation at this angle of attack, and thus, enhance the efficiency of the wing. Three actuation cases with the jets located on the main wing, on the flap, and on both elements at the same time were investigated. While some improvements are observed with the actuation, particularly in the combined actuation case, the overall wing efficiency was only marginally enhanced. Challenges in achieving significant improvements were attributed to complex flow interactions and the dominance of pressure forces, which affect both the lift and the drag coefficients at the same time.

Keywords: Aerodynamics; Turbulence; High-Lift Wing; Active Flow Control

1. Introduction

The deployment of high-lift devices during takeoff and landing operations leads to increased levels of noise, having a bad impact on communities and wildlife near airports. Additionally, the optimization of these devices could yield other benefits such as drag reduction, increased payload capacities, shorter runway lengths, etc., that could have a positive economic and environmental impact on the aeronautical industry. Therefore, understanding the flow complexities in multi-element wings and controlling the flow over its surface is crucial for designing more efficient aviation systems in the future.

* Corresponding author.
E-mail address: ricard.montala@upc.edu

10.1016/j.procs.2024.07.006

Computational Fluid Dynamics (CFD) is identified as a key tool to achieve this and some workshops in the literature have aimed to enhance CFD capabilities for multi-element wings. Notably, the high-lift CFD challenge [1] workshop hosted by NASA and the third workshop on benchmark problems for airframe noise computations (BANC-III) [2] organized by AIAA focused on the benchmark configuration known as 30P30N. As seen in Fig. 1, this is a three-element wing with the slat and flap deflected 30° with respect to the main element. Nevertheless, while the former workshop utilized RANS models, the numerical contributions of the latter were mainly based on hybrid RANS-LES approaches. Moreover, the central point of the latter workshop was the flow dynamics in the slat cove, identified as the main source of noise in high-lift wings and hence, the physical analysis of the flow was rather limited. In the context of these workshops, some experimental [3, 4, 5] and numerical [6, 7] contributions can be found in the literature. Some subsequent works investigating the 30P30N wing configuration have been published over the years [8, 9, 10], despite mainly used the workshop database to test their novel numerical approaches without extending the physical analysis.

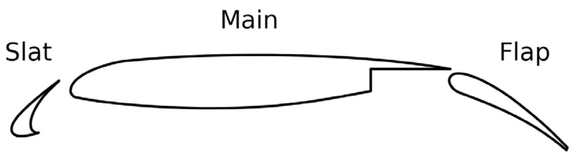


Fig. 1. 30P30N geometry.

Active flow control (AFC) has arisen as a potential technique to control the flow over wings and thus, achieve higher performances. Different AFC techniques can be found in the literature. In this regard, synthetic jets (zero net mass flux) have been the subject of many investigations, showing promising results. Gilarranz et al. [11] achieved controlling the separation of a NACA 0015 airfoil in a wind tunnel, delaying the onset of stall conditions. Similar results were reproduced by You and Moin [12] conducting LES. McCormick [13] experimentally studied the effect of the jet momentum in a single element airfoil, also extending the stall angle to higher values and finding the suite jet momentum range to achieve so, which was in accordance with the experimental findings of Goodfellow et al. [14] in a NACA0025. Based on the set-up employed in the previous studies, Rodriguez et al. [15] also obtained similar results for the flow past an SD7003 airfoil performing LES, increasing the efficiency of the wing at high angles of attack, when a massive separation of the flow is observed. Synthetic jets have also been applied to high-lift wing configurations in both, the experimental [16, 17] and numerical fields [18], and even to complete 3D wing models. This is the case of Lehmkuhl et al. [19], who performed a wall-modeled LES on a full aircraft in stall conditions (the JSM model).

In this study, we investigate the airflow around the 30P30N three-element high-lift wing aiming to gain deeper insights into the flow physics of high-lift wings at different angles of attack, especially considering the limited analysis available in the literature and their constrained computational accuracy, as most of the authors employed hybrid RANS-LES models. To this end, large eddy simulations (LES) are conducted at a moderate Reynolds number of $Re_c = 750,000$, based on the chord length (c), and three different angles of attack: $\alpha = 5°, 9°$, and $23°$. With a clearer understanding of the flow dynamics, this research offers initial findings from an ongoing effort to devise intelligent AFC strategies for drag and noise reduction in wing designs. Consequently, various actuations using synthetic jets are examined based on the observations and outcomes from the baseline case. By doing so, this work seeks to contribute to the development of more effective and environmentally friendly high-lift devices, providing valuable insights and potential applications for the aeronautical industry.

2. Numerical methods

The finite element (FE) code Alya [20] is utilized to solve the incompressible spatially-filtered Navier-Stokes equations in this study. For turbulence modelling, the LES equations are closed using the subgrid-scale eddy-viscosity model proposed by Vreman [21]. In Alya, the convective operator of the equations is approximated using a low-dissipation scheme that conserves energy, momentum, and angular momentum at the discrete level [22]. Time advancement employs an energy-conserving 2nd-order Runge-Kutta explicit method coupled with an eigenvalue-based time-step estimator [23]. A non-incremental fractional-step method is employed to stabilize the pressure.

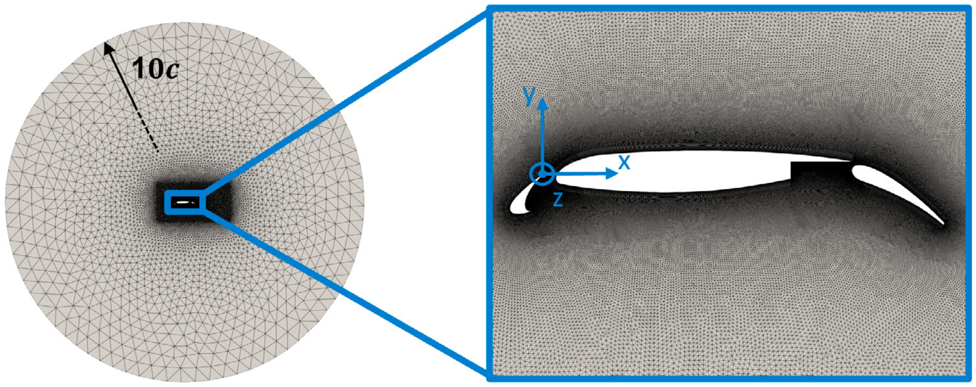


Fig. 2. Computational mesh.

The computational mesh (see Fig. 2) consists of a circular domain with a radius of $r = 10c$ in the x-y plane and extruded a distance of $L_z = 0.1c$ along the z-direction. Inlet flow conditions are applied upstream based on the selected flow conditions, i.e., the Reynolds number (Re_c) and the angle of attack (α). At the outlet, no velocity gradients are enforced, while periodic conditions are applied in the spanwise direction. In the wall, the no-slip condition is considered. To achieve the desired near-wall spacings without substantially increasing the element count, a structured-like mesh is utilized around the airfoil surfaces. Maximum non-dimensional distances of $\Delta y^+ < 1$, $\Delta x^+ < 80$ and $\Delta z^+ < 50$ are enforced near the walls, aligning with the typical values reported in Choi and Moin [24] for wall-resolved LES. The mesh comprises a total of 58 million grid points.

To conduct such scale-resolving simulations, the use of a high-performance computer (HPC) was required. The present simulations were carried out on Vega, an HPC facility located at the Institute of Information Science (IZUM) in Slovenia. The CPU partition of this cluster was employed. Alya has already proven its scalability when working with such supercomputing resources. Some tests were performed in the HPC machine called Irene ROME, which has a similar architecture to the Vega machine. In the weak scalability tests, each CPU was assigned 29,000 unknowns, similar to the present computations, and the efficiency was maintained above 63% for 12,000 CPUs. In the strong scalability tests, Alya kept an efficiency above 72% when using the same amount of cores, and above 92% when 6,144 CPUs were used.

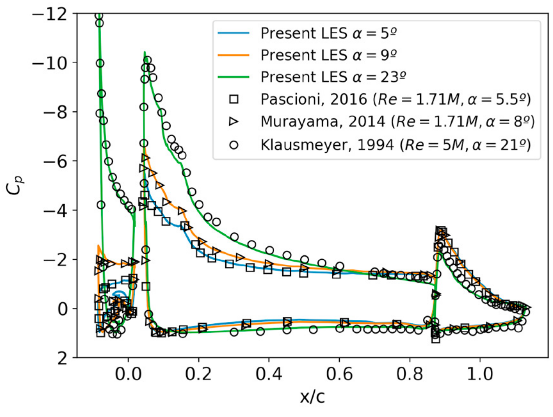
For the current simulations, the computational cost of a single run was approximately 1M CPU-hours, using 2,048 CPUs (8 nodes, each node containing 256 CPUs). Therefore, for all the results presented in this work, which comprises six different cases, a total of about 6M CPU-hours were employed.

3. Results

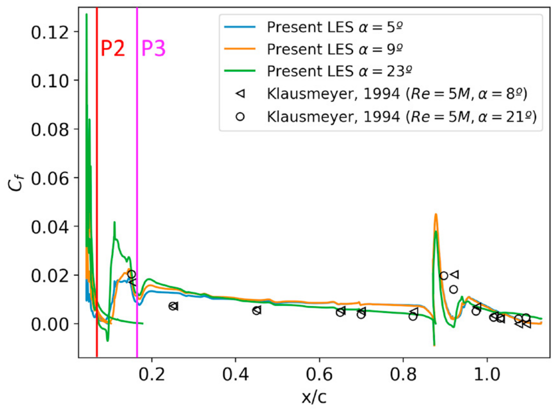
3.1. Baseline

The computed pressure (C_p) and skin friction (C_f) coefficients along the walls of the airfoil are displayed in Fig. 3. Moreover, these distributions are plotted against other results that can be found in the literature for similar flow conditions. Thus, it is validated that the C_p achieves a good level of accuracy compared to those results. The C_f also shows a relatively good agreement. The little deviation that can be detected could be explained by the differences in the Reynolds number. It is expected that this value is more sensible to Re variations.

The integration of these coefficients along the surfaces, i.e., C_p and C_f, yields the pressure and viscous forces acting on each wing element. Further decomposing these forces in the freestream streamwise and normalwise directions, the lift (C_l) and drag (C_d) coefficients. respectively, can be obtained, with their respective pressure and viscous contributions. Fig. 4 illustrates the lift coefficient, and the pressure ($C_{d,press.}$) and viscous ($C_{d,visc.}$) drag coefficients

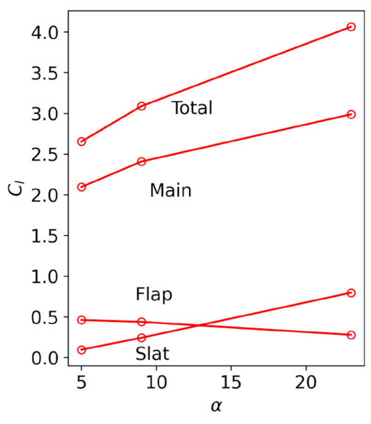


(a)

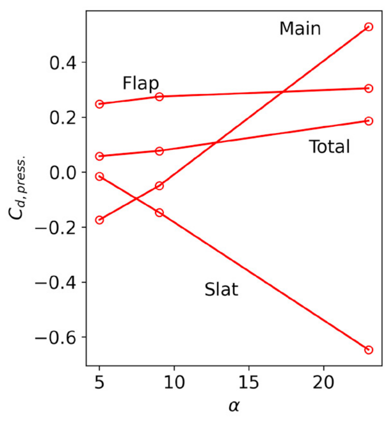


(b)

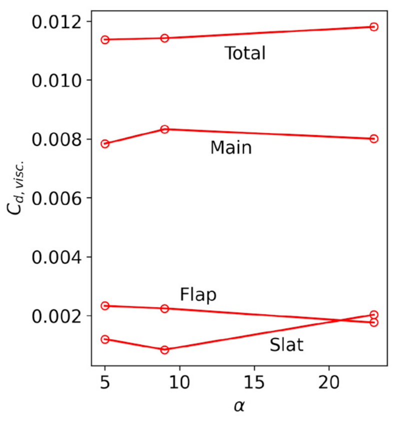
Fig. 3. (a) Pressure coefficient; (b) skin friction coefficient along the main and flap suction sides.



(a)



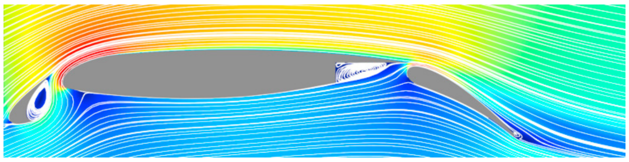
(b)

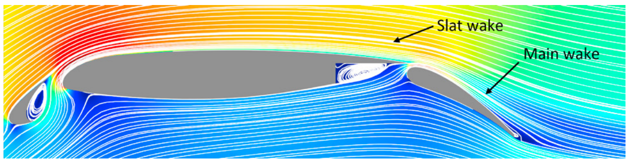


(c)

Fig. 4. (a) Lift coefficient; (b) pressure drag coefficient; (c) viscous drag coefficient.

generated by each element. It is then observed that the main wing is the principal contributor to the lift coefficient, as it exhibits much more prominent suction peaks (see Fig. 3a). Focusing on the drag, the main wing has again the major $C_{d,visc.}$ contribution, which can be mainly attributed to the larger dimensions of the wing. As observed in Fig. 3b, the main and flap elements show similar levels of C_f but, in the case of the flap, these are integrated along a smaller area. It is also detected that C_f shows small variations with the angle of attack, which in turn results in small changes in the $C_{d,visc.}$. In the case of the $C_{d,press.}$, this exhibits visible changes with the angle of attack, linked to the visible changes in the pressure coefficient in Fig. 3a. At $\alpha = 5°$, the main wing has the lowest contribution to the pressure drag but this contribution increases with the angle of attack. At low angles of attack, the main wing is nearly streamlined with the freestream, reducing the pressure component of the drag force. As the angle of attack is increased, besides that the main wing is larger, it also exhibits much higher suction peaks. This is translated to a higher C_l, but also to a higher $C_{d,press.}$ as well. Indeed, the rate of growth of $C_l - \alpha$ decreases with the angle of attack, while the $C_{d,press.} - \alpha$ slope does the opposite. This is a consequence of the airfoil entering the stall conditions, which can be directly visualized in Fig. 5 by the low-velocity recirculation area that is developed at $\alpha = 23°$ above the flap. Regarding the slat, this also develops a prominent suction peak as the angle of attack is increased, resulting from the increased velocities observed in the slat leading edge due to the shift of the stagnation point towards the cusp, as also visualized in Fig. 5. Indeed, a turbulent boundary layer is developed at $\alpha = 23°$ in the slat suction side which was not present in the lower angles of attack.


(a)



(b)

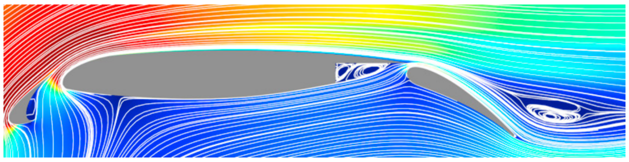
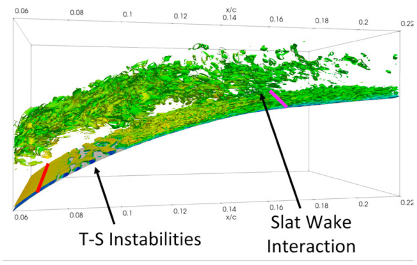
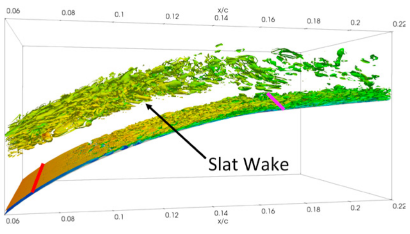

(c)

Fig. 5. Streamlines and velocity magnitude contours at (a) $\alpha = 5°$; (b) $\alpha = 9°$; (c) $\alpha = 23°$.

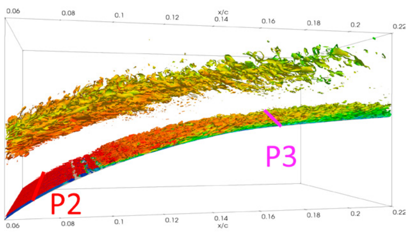
Based on how the recirculation area above the flap is formed, it seems that the main wing is responsible for it, which would be linked to the pressure drag increase of this element. According to the C_f in the flap surface, there is no flow separation along the flap surface. Instead, it is identified in the streamlines that this recirculation comes from the wake of the main wing. The gap between the main element and the flap leading edge feeds the flap suction side with relatively high momentum flow, which is kept attached to the flap's surface. However, it is observed that a separation of the shear layer takes place in the wake of the main wing. This separation would be the signature of the pressure drag increase in the main wing.



(a)



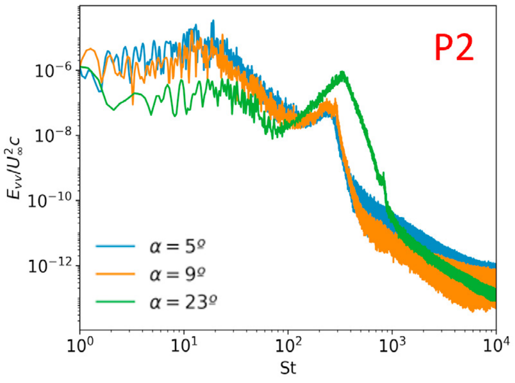
(b)



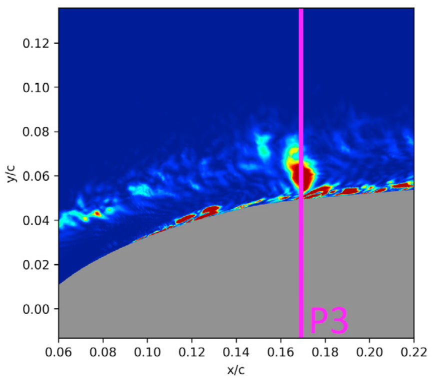
(c)

Fig. 6. Q-criterion iso-contours at the main wing leading edge at (a) $\alpha = 5°$; (b) $\alpha = 9°$; (c) $\alpha = 23°$.

At this point, the suction side of the main element is analysed. In the leading edge of the main wing, a laminar-to-turbulent transition can be detected, as visualized in the Q-criterion iso-contours depicted in Fig. 6, and also evidenced by a sudden reduction of the C_f at around $x/c = 0.068$ (highlighted in red as P2) for the three angles of attack studied (see Fig. 3b). In this region, the emergence of Tollmien-Schlichting (T-S) instabilities triggers the transition just after the peak of the C_p, so that the first instabilities may be produced by the local adverse pressure gradient (APG). The frequencies of the T-S can be observed in Fig. 7a, which shows the spectra of the velocity magnitude along the P2 line. Therefore, the stronger APG at increasing angles of attack leads to higher frequencies of the instabilities, and hence, a more rapid growth of the turbulent fluctuations and a further reduction of the C_f. At $\alpha = 23°$, this parameter even reaches a mean value of $C_f < 0$.



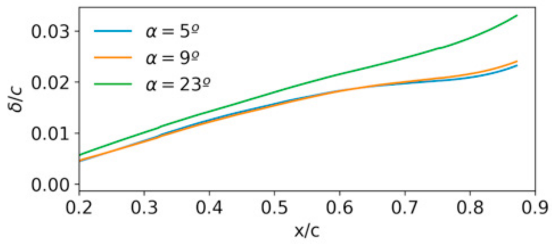
(a)



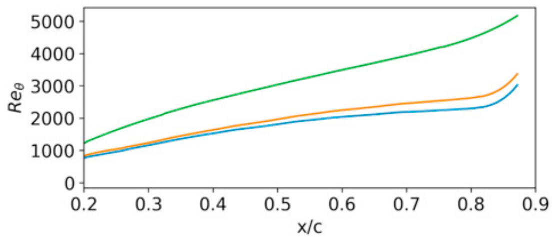
(b)

Fig. 7. (a) Spectra of the velocity magnitude in P2; (b) instantaneous TKE contours in the main leading edge.

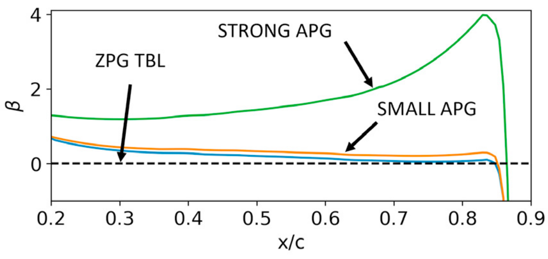
After the flow has reached the fully turbulent state (C_f increases), a second reduction of the C_f coefficient is detected at around $x/c = 0.170$ (highlighted in magenta as P3). This is related to the interaction of the coherent turbulent structures shed from the slat wake reaching the leading edge of the main wing. This phenomenon is represented in Fig. 7b, which describes how one coherent structure approaching from the slat wake, represented by an area of high turbulent kinetic energy (TKE), reaches the main wing surface. As the angle of attack is increased, the slat wake is separated away from the main wing walls, as seen in Fig. 5, which leads to lower impinging events and hence, a lower affection of the C_f as the angle of attack is increased, as detected in Fig. 3b.



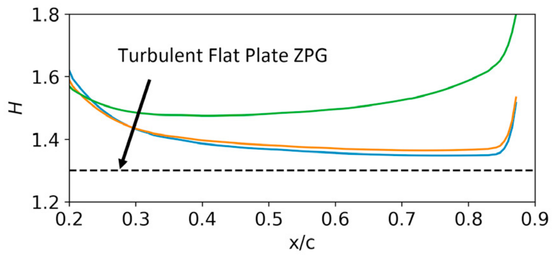
(a)



(b)



(c)



(d)

Fig. 8. Boundary layer development along the main suction side. (a) Boundary layer thickness; (b) momentum thickness Reynolds number; (c) Clauser pressure-gradient parameter; (c) shape factor.

Following the natural transition and the interaction of the slat wake, a turbulent boundary layer is developed along the main suction side ($x/c > 0.2$). Some parameters describing the boundary layer are presented in Fig. 8. These are the boundary layer thickness (δ), the momentum thickness Reynolds number (Re_θ), the shape factor (H) and the Clauser pressure-gradient parameter (β). As shown in these plots, the boundary layer exhibits a similar and moderate growth at $\alpha = 5^\circ$ and 9°. However, at $\alpha = 23^\circ$, this growth is much more pronounced, as observed in δ and Re_θ. Indeed, as approaching the leading edge of the main wing, the boundary layer at the lower angles of attack tends to a fixed value while, at $\alpha = 23^\circ$, the boundary layer keeps growing. This is associated with the evolution of β and H, which are indicators of the pressure gradient. As it can be seen from the figure, β at $\alpha = 5^\circ$ and 9° tends to zero, while H gets close to 1.3 near the trailing edge, which is the expected value in a zero pressure gradient (ZPG) boundary layer in a flat plate [25]. On the other hand, at $\alpha = 23^\circ$, a much stronger APG is detected, with the β and H parameters getting higher as moving downstream. Therefore, the stronger APG found at high angles of attack, i.e., $\alpha = 23^\circ$, associated with the higher suction peaks observed in the C_p, yields a much thicker boundary layer as a result of a higher wall-normal convection in detriment of the streamwise velocity. These conditions favour the emergence of the separation of the shear layer in the wake, which finally evolves into a zone of low velocity and recirculation.

3.2. Actuation

According to the results obtained for the baseline case in the previous section, the aim in this section is to reduce the drag and/or increase the lift of the wing at $\alpha = 23^\circ$ (maintaining the Reynolds number at $Re_c = 750,000$). As discussed above, in this angle of attack, the wing is in stall conditions so that a recirculation area above the flap is detected, this being a signature of the drag increase. Therefore, targeting this structure and reducing it, should cause an increase of the lift and/or a reduction of the drag. As has been commented before, it all points out that this separation of the streamlines is caused by the APG present along the main suction side, which produces a loss of streamwise momentum and an increase of the wall-normal convection. To increase the mixing along the different layers in the wall-normal direction and hence, promote the transport of high momentum flow towards the wall, the idea is to locate the actuators near the trailing edge of the main wing ($x/c = 0.65$), where the boundary layer starts to become especially thick compared to the other cases; and/or at the leading edge of the flap ($x/c = 0.90$), where the flow convected from the main pressure side through the main-flap gap might be transferred into the main wake.

In the present study, synthetic jets (or zero net mass flow rate jets) are considered for the actuated cases. To this end, in the zone where the jet is located, an inlet velocity condition is imposed at the wall of the airfoil along the whole spanwise as:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix}_{act} = U_\infty \sqrt{\frac{c}{h} C_\mu} sin\left(2\pi \frac{F^+ U_\infty}{L_{ref}} t\right) \begin{bmatrix} cos(\phi - \theta_0) \\ sin(\phi - \theta_0) \\ 0 \end{bmatrix} = f(C_\mu, F^+, \phi) \tag{1}$$

where h/c represents the width of the jet (chordwise), C_μ is the momentum coefficient of the jet, F^+ the temporal frequency of the actuation, L_{ref} a characteristic length and ϕ the angle of the jet with respect the local tangential direction of the surface. Therefore, to express the velocity of the jet in xyz global coordinates, θ_0 is the angle between the wall-normal direction and the y-axis. All these parameters are schematized in Fig. 9.

As in Rodriguez et al. [15], L_{ref} is selected to be the distance between the actuator and the trailing edge of the respective wing element. Additionally, similarly to those investigations, the width of the jet is selected to be $h/C = 0.006$ in all the cases. With this being set, the main parameters defining the actuation are the momentum coefficient C_μ, the actuation frequency F^+ and the jet angle ϕ. McCormick [13] achieved to reduce the wake width with $C_\mu = 0.01 - 0.015$, which is in accordance with the values considered by Shmilovich and Yadlin [18] ($C_\mu = 0.015$), and You and Moin [12] ($C_\mu = 0.0123$). For the frequency, the literature has achieved to increase the stall angles towards higher angles of attack considering values close to one: Rodriguez et al. [15] considered $F^+ = 1$, Shmilovich and Yadlin [18] used $F^+ = 1.52$ and You and Moin [12] applied $F^+ = 1.284$, while McCormick [13] found that results were not very

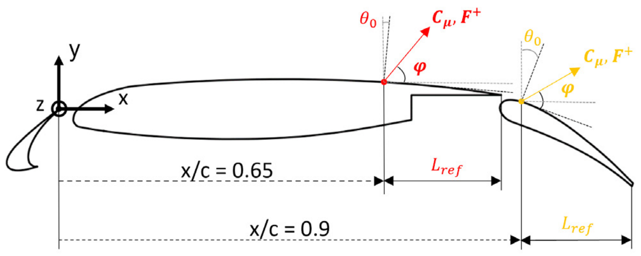


Fig. 9. Schematics of the actuation parameters.

sensible in the range of $F^+ = 0.25 - 3.5$. In the present investigation, the considered AFC cases are summarized in Table 1.

Table 1. Summary of the parameters used in each actuation ($\alpha = 23°$ and $Re_c = 750,000$).

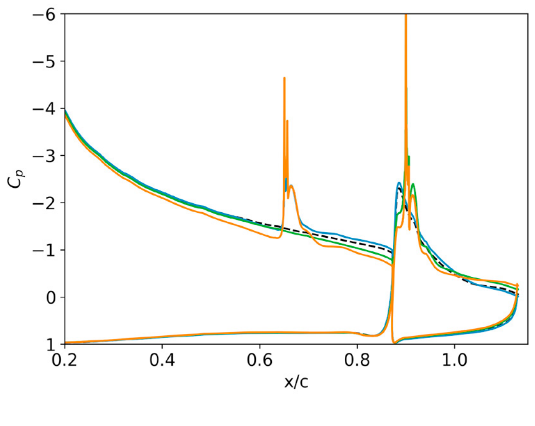
CASE ID	Main - C_μ	Main - F^+	Main - ϕ	Flap - C_μ	Flap - F^+	Flap - ϕ
AFC 1	0.015	1.52	90°	-	-	-
AFC 2	-	-	-	0.025	1.52	90°
AFC 3	0.015	1.52	90°	0.025	1.52	90°

The C_p and C_f distributions obtained for the actuated cases are displayed in Fig. 10, which compares the results with the baseline case. Since the jets are located in the main trailing edge and the flap leading edge, and no visible differences are obtained upstream, results are shown at $x/c > 0.2$. On the other hand, the lift and drag coefficients computed in each case can be consulted in Table 2, which also displays the relative increase of C_l and C_d with respect to the baseline results. Note that the slat coefficients are not reported in this table due to the minor changes that are obtained for this element when the actuation is activated.

According to these results, when the jet is located in the main wing (AFC 1), an increase in the total lift is observed, which is accompanied by an even higher increase in the total drag coefficient. Therefore, for this actuation, the efficiency ($E = C_l/C_d$) of the wing decreases (−9.29%). As it can be seen from the C_p in Fig. 10a, after the actuation ($x/c > 0.6$), the pressure coefficient is lower, which is translated into a higher suction and, in turn, into a higher C_l. However, at this angle of attack, i.e., $\alpha = 23°$, pressure forces are more aligned towards the streamwise direction of the freestream (direction of the drag force) and this entails even higher values of C_d. In this case, a minor reduction of the C_f is detected in Fig. 10b.

When the actuation is located in the flap (AFC 2), a small reduction of the total C_l is observed, while the total C_d increases, hence yielding a worse efficiency of the wing (−3.92%). In this case, the skin friction of the flap is notably affected and reaches smaller values, which means a lower $C_{d,visc.}$. However, as seen before in the baseline results, the total drag is dominated by the pressure contribution rather than by the viscous one. In this manner, higher suction levels are identified along the rear part of the flap suction side, together with a lower suction peak on the leading edge. The net effect is a slight increase of the flap lift coefficient but, due to its inclination and the maintained levels of low C_p near the trailing edge, also an increase of $C_{d,press.}$ and total C_d. Additionally, slightly higher values of C_p are detected along the main suction side and hence, the lower suction here explains why smaller values of C_l are obtained for this element, together with a reduction of the main wing C_d.

Finally, combining both actuations and considering one jet on the main and one on the flap at the same time (AFC 3), this is the only case that achieves to increase the efficiency of the wing (+3.74%). Despite the total lift coefficient is reduced, the drag coefficient is reduced even more. Again, the lower total C_l is explained by the lower suction levels found in the main wing and the flap suction peak (see Fig. 10a). However, as commented before, at high angles of attack pressure forces have a higher impact on the drag rather than on the lift. Hence, a reduction of the lift here implies a further reduction of the drag.



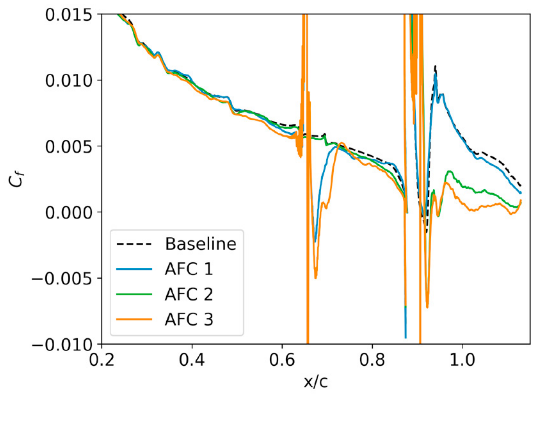


Fig. 10. AFC results: (a) Pressure coefficient; (b) skin friction coefficient

Table 2. Lift and drag coefficients obtained for the AFC cases compared to the baseline results (α = 23° and Re_c = 750, 000).

CASE ID	$C_{l,total}$	$C_{l,main}$	$C_{l,flap}$	$C_{d,total}$	$C_{d,main}$	$C_{d,flap}$
Baseline	4.063	2.988	0.279	0.198	0.536	0.306
AFC 1	4.122 (+1.45%)	3.032 (+1.50%)	0.290 (+4.20%)	0.222 (+11.83%)	0.555 (+3.47%)	0.315 (+2.74%)
AFC 2	4.038 (-0.60%)	2.950 (-1.26%)	0.293 (+4,97%)	0.205 (+3.46%)	0.519 (-3.13%)	0.330 (+7.80%)
AFC 3	3.971 (-2.25%)	2.913 (-2.51%)	0.269 (-3.48%)	0.187 (-5.77%)	0.511 (-4.75%)	0.313 (+2.35%)

Overall, the only case that achieves an increase of the wing efficiency, i.e., AFC 3, also entails a reduction of the lift. As said before, it is then difficult to reduce the drag, highly related to C_p, without altering the lift, which is also closely linked with the pressure coefficient. Then, targeting the viscous forces might seem a good idea, as they mainly act on the drag coefficient. However, this has a low impact on the total drag coefficient compared to the pressure forces.

Although a small increase in the wing efficiency is observed, not very promising results have been obtained in this preliminary study, especially if considering that the amount of improvement might be in the range of accuracy of the CFD predictions themselves. Exploration of alternative parameters or jet distributions might be considered. For instance, Shmilovich and Yadlin [18] significantly increased the efficiency of the wing by considering many more jets distributed along the slat, main and flap surfaces. Also, based on the streamlines depicted in Fig. 5c, placing the jets on the slat surface with the aim to bring the slat wake closer to the main wall could potentially reduce the APG and thus enhance the wing efficiency. It appears that the separation of the slat wake from the main surface contributes to strengthening the APG. However, considering the multitude of parameters involved, addressing this issue through conventional parametric studies seems challenging. Therefore, given the advancements in artificial intelligence (AI), integrating AFC with deep reinforcement learning (DRL) emerges as a compelling approach. The AI agent may be capable of finding much more complex strategies leading to more efficient and effective actuations. An example of this is the work of Rabault et al. [26] and Suárez et al. [27], where both authors successfully controlled the wake of the flow past a cylinder, thereby reducing its drag.

4. Conclusions

In this study, large eddy simulations (LES) on the flow past the 30P30N three-element wing at a fixed Reynolds number Re_c = 750, 000 and three different angles of attack α = 5°, 9°, and 23° are conducted.

The baseline results of both the pressure and skin friction distributions along the wing surfaces demonstrate a good agreement with the existing literature. It is observed that the main wing significantly contributes to the lift coefficient due to its pronounced suction peaks and larger wet surface. Regarding the drag, the pressure and viscous contributions are analysed separately. The viscous drag shows small variations with the angle of attack, while the pressure drag is much more affected by this parameter. It is also identified that pressure drag is the major responsible for the total drag and hence dominates its evolution across the different angles of attack. The pressure drag increases with the angle of attack, especially due to the main wing. At low angles of attack, the main wing pressure drag is low, whereas its contribution considerably increases with the angle of attack and, at $\alpha = 23°$, the main wing exhibits the highest value among the different elements. At this angle of attack, a low-velocity recirculation area emerges above the flap, this being the footprint of stall conditions. It is observed that this is not caused by a flow separation along the flap surface but due to a separation of the streamlines in the main element wake. Thus, this can be understood as a signature of the increased pressure drag that is observed in this element.

The turbulent boundary layer that develops along the main suction side is then analysed. After a laminar-to-turbulent transition in the leading edge and interactions with the turbulent coherent structures shed from the slat wake, the evolution of some boundary layer parameters is studied, i.e., the boundary layer thickness, the momentum thickness Reynolds number, the Clauser pressure-gradient parameter and the shape factor. According to these parameters, while at $\alpha = 5°$ and 9°, the turbulent boundary layer nearly resembles the behaviour of a zero pressure gradient turbulent boundary layer, a much stronger adverse pressure gradient is observed at $\alpha = 23°$. Thus, the growth of the boundary layer is much more pronounced due to the increased wall-normal convection, and the loss of the streamwise momentum might cause the recirculation area that appears in the main wake above the flap.

To mitigate the adverse pressure gradient effects and reduce the flow separation at $\alpha = 23°$, three actuations using synthetic jets have been explored: One actuation with the jet on the main trailing edge, one with the jet on the flap leading edge and one considering both jets simultaneously. However, the only actuation that achieves to increase the wing efficiency is the combined one. Moreover, the enhancement is very small and the reduction of drag is accompanied by also a reduction of lift. Since the pressure forces have a major impact on both lift and drag forces at the same time, it is difficult to target only the drag without altering the lift. Further investigations are needed to develop more effective strategies for drag reduction and lift enhancement, which may imply advanced control strategies that could be found, for instance, through deep reinforcement learning.

Acknowledgements

This research has received financial support from the *Ministerio de Ciencia e Innovación* of Spain (PID2020-116937RB-C21 and PID2020-116937RB-C22). Simulations were conducted with the assistance of the EuroHPC JU, who granted computational resources to the HPC facility of *Vega*, at Institute of Information Science (EHPC-REG-2022R01-030). Ricard Montalà work is funded by a FI-SDUR grant (2022 FISDU 00066) from the *Departament de Recerca i Universitats de la Generalitat de Catalunya*. Oriol Lehmkuhl has been partially supported by a *Ramon y Cajal* postdoctoral contract (RYC2018-025949-I). The authors also acknowledge the support of the *Departament de Recerca i Universitats de la Generalitat de Catalunya* through the research group Large-scale Computational Fluid Dynamics (2021 SGR 00902) and the Turbulence and Aerodynamics Research Group (2021 SGR 01051).

References

[1] Klausmeyer, S. and Lin, J. (1997) "Comparative results from a CFD challenge over a 2D three-element high-lift airfoil." *NASA Technical Memorandum No. 112858*.

[2] Choudhari, M. and Lockard, D. (2015) "Assessment of slat noise predictions for 30P30N high-lift configuration from BANC-III workshop." *21st AIAA/CEAS Aeroacoustics Conference*.

[3] Klausmeyer, S. and Lin, J. (1994) "An experimental investigation of skin friction on a multi-element airfoil." *12th Applied Aerodynamics Conference*.

[4] Murayama, M., Yokokawa, Y., Ura, H., Nakakita, K., Yamamoto, K., Ito, Y., Takaishi, T., Sakai, R., Shimoda, K., Kato, T. and Homma, T. (2014) "Experimental study of slat noise from 30P30N three-element high-lift airfoil in JAXA kevlar-wall low-speed wind tunnel." *AIAA/CEAS Aeroacoustics Conference*.

[5] Pascioni, K. and Cattafesta, L. (2016) "Aeroacoustic measurements of leading-edge slat noise." *22nd AIAA/CEAS Aeroacoustics Conference*.

[6] Bodart, J., Larsson, J. and Moin, P. (2013) "Large eddy simulation of high-lift devices." *21st AIAA Computational Fluid Dynamics Conference*.

[7] Ashton, N., West, A. and Mendon, F. (2016) "Flow dynamics past a 30P30N three-element airfoil using improved delayed detached-eddy simulation." *AIAA Journal* **54** (11): 3657–3667.

[8] Shi, J., Yan, H. and Wang, Z. (2018) ”Towards direct computation of aeroacoustic noise with the high-order FR/CPR method.” *2018 AIAA/CEAS Aeroacoustics Conference*.

[9] Gao, J., Li, X. and Lin, D. (2020) ”Numerical simulation of the 30P30N high-lift airfoil noise with spectral difference method.” *AIAA Journal* **58** (6): 2517–2532.

[10] Jin, Y., Liao, F. and Cai, J. (2020) ”Numerical simulation of 30P30N multi-element airfoil using delayed detached-eddy simulation.” *AIAA Aviation 2020 Forum*.

[11] Gilarranz, J.L., Traub, L.W. and Rediniotis, O.K. (2005) ”A new class of synthetic jet actuators—part II: Application to flow separation control.” *Journal of Fluids Engineering* **127** (2): 377–387.

[12] You, D. and Moin, P. (2008) ”Active control of flow separation over an airfoil using synthetic jets.” *Journal of Fluids and Structures* **24** (8): 1349–1357.

[13] McCormick, D. (2012) ”Boundary layer separation control with directed synthetic jets.” *38th Aerospace Sciences Meeting and Exhibit*.

[14] Goodfellow, S.D., Yarusevych, S. and Sullivan, P.E. (2013) ”Momentum coefficient as a parameter for aerodynamic flow control with synthetic jets.” *AIAA Journal* **51** (3): 623–631.

[15] Rodriguez, I., Lehmkuhl, O. and Borrell, R. (2020) ”Effects of the actuation on the boundary layer of an airfoil at Reynolds number Re = 60000.” *Flow, Turbulence and Combustion* **105**: 607–626.

[16] Melton, L.P., Yao, C.S. and Seifert, A. (2006) ”Active control of separation from the flap of a supercritical airfoil.” *AIAA Journal* **44** (1): 012017.

[17] Khodadoust, A. and Washburn, A. (2007) ”Active control of flow separation on a high-lift system with slotted flap at high Reynolds number.” *25th AIAA Applied Aerodynamics Conference*.

[18] Shmilovich, A. and Yadlin, Y. (2009) ”Active flow control for practical high-lift systems.” *Journal of Aircraft* **46** (4): 1354–1364.

[19] Lehmkuhl, O., Lozano-Durán, A. and Rodriguez, I. (2019) ”Active flow control for external aerodynamics: from micro air vehicles to a full aircraft in stall.” *Journal of Physics: Conference Series* **1522**: 34–41.

[20] Vázquez, M., Houzeaux, G., Koric, S., Artigues, A., Aguado-Sierra, J., Arís, R., Mira, D., Calmet, H., Cucchietti, F., Owen, H., Taha, A., Dering, E., Cela, J.M. and Valero, M. (2016) ”Alya: Multiphysics engineering simulation toward exascale.” *Journal of Computational Science* **14**: 15–27.

[21] Vreman, A.W. (2004) ”An eddy-viscosity subgrid-scale model for turbulent shear flow: Algebraic theory and applications.” *Physics of Fluids* **16** (10): 3670–3681.

[22] Lehmkuhl, O., Houzeaux, G., Owen, H., Chrysokentis, G. and Rodriguez, I. (2019) ”A low-dissipation finite element scheme for scale resolving simulations of turbulent flows.” *Journal of Computational Physics* **390**: 51–65.

[23] Trias, F.X. and Lehmkuhl, O. (2011) ”A self-adaptive strategy for the time integration of Navier-Stokes equations.” *Numerical Heat Transfer, Part B: Fundamentals* **60** (2): 116–134.

[24] Choi, H. and Moin, P. (2012) ”Grid-point requirements for large eddy simulation: Chapman’s estimates revisited.” *Physics of Fluids* **24** (1): 011702.

[25] White, F.M. (2006) ”Flow past immersed bodies”. In: *Fluid Mechanics*, 7th edn. McGraw-Hill, New York: 457–528.

[26] Rabault, J., Kuchta, A., Jensen, U., Réglade, U. and Cerardi, N. (2019) ”Artificial neuronal networks trained through deep reinforcement learning discover control strategies for active flow control.” *Journal of Fluid Mechanics* **865**: 281–302.

[27] Suárez, P., Alcántara-Ávila, F., Miró, A., Rabault, J., Font, B., Lehmkuhl, O. and Vinuesa, R. (2023) ”Active flow control for three-dimensional cylinders through deep reinforcement learning.” *ETMM14 conference proceedings*.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 42–51

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

A Portable Drug Discovery Platform for Urgent Computing

Davide Gadioli[a,*], Gianmarco Accordi[a], Jan Krenek[c], Martin Golasowski[c], Ladislav Foltyn[c], Jan Martinovic[c], Andrea R. Beccari[b], Gianluca Palermo[a]

[a]*Politecnico di Milano - DEIB, Milano, Italy*
[b]*EXSCALATE - Dompé Farmaceutici SpA, Naples, Italy*
[c]*IT4Innovations, VSB – Technical University of Ostrava, Ostrava-Poruba, Czech Republic*

Abstract

Drug discovery is a long and costly process. Recent studies demonstrated how the introduction of an in-silico stage, named virtual screening, that suggests which molecule to test in-vitro, increases the drug discovery success probability. In the context of urgent computing, where it is important to find a therapeutic solution in a short time frame, the number of candidates that we can virtual screen is limited only by the available computation power. In this paper, we focus on LiGen, the virtual screening application of the EXSCALATE platform. In particular, we address two challenges of performing an extreme-scale virtual screening on a modern HPC system. The first one is posed by hardware heterogeneity, where GPUs of different vendors account for a large fraction of their performance. The second challenge concerns the operational difficulties of running the campaign since it requires significant effort and technical skills that are not common among domain experts. We show how hinging on SYCL and the LEXIS Platform, is the solution that the EXSCALATE Platform uses to address these challenges.

Keywords: HPC; Virtual Screening; Performance Portability; Urgent Computing

1. Introduction

Drug discovery aims at finding a molecule with a small molecular weight that yields a therapeutic effect against the target disease. This is a long and expensive process, with a success probability lower than 25% [15]. One challenge is due to the sheer size of the chemical space that we can access to design drug candidates, which is estimated to include 10^{33} molecules [20]. The problem arises when we consider that it is hard to test *in-vitro* more than 10^5 compounds per day [14], forcing pharmaceutical companies to rely only on their expertise to select which molecule to test. Recent studies demonstrated how the introduction of an *in-silico* stage that virtual screens an input chemical library, suggesting which are the most promising candidates, improves the drug discovery success probability [13, 3].

* Corresponding author.
E-mail addresses: davide.gadioli@polimi.it (Davide Gadioli)., gianmarco.accordi@polimi.it (Gianmarco Accordi)., jan.krenek@vsb.cz (Jan Krenek)., martin.golasowski@vsb.cz (Martin Golasowski)., ladislav.foltyn@vsb.cz (Ladislav Foltyn)., jan.martinovic@vsb.cz (Jan Martinovic)., andrea.beccari@dompe.com (Andrea R. Beccari)., gianluca.palemro@polimi.it (Gianluca Palermo).

The virtual screening stage estimates the interaction strength between a drug candidate, named *ligand*, and at least one binding site of the target protein(s), representing the target disease. Domain experts can use this information to rank a chemical library for selecting the most promising candidates to test in vitro. To estimate the interaction strength we need to solve two well-known problems: molecular docking and scoring [17, 19, 21]. The former estimates the 3D displacement of the ligand atoms when they interact with the protein, which is a requirement for the latter to evaluate the intermolecular forces. Both tasks are complex and compute-intensive. Since the computation of each protein-ligand pair is independent of the others, this is an embarrassingly parallel problem. For these reasons, HPC systems are the ideal machines for a virtual screening campaign. In the context of urgent computing [11], where it is important to find a solution in a short time frame, HPC usage becomes of paramount importance. For example, the largest virtual screening campaign ranked 72 billions of ligands, against 15 binding sites of 12 viral proteins of the SARS-CoV-2 [7]. The computation lasted 60 hours, using a significant fraction of the computation nodes of Marconi100 at CINECA, and of HP5 at ENI, which have a combined sustained throughput of 81 petaflops. The results are under analysis and publicly available [1].

In this paper we focus on LiGen, the virtual screening application of the EXSCALATE platform [7], that has been designed from scratch to hinge on all the computation resources provided by an HPC system. In particular, we report the main two challenges of extreme-scale virtual screening campaigns from the computation perspective, and how we are addressing them. The first challenge is posed by hardware heterogeneity. If we look at the TOP500 list[2], which ranks the most powerful HPC systems in terms of FLOPS, we can notice how the top 4 systems use different architectures. While 3 of them rely on GPUs to accelerate computation, they are made by different vendors, each with its own native programming language. For the virtual screening campaign, we aim for performance portability, not only functional portability, because the probability of finding good candidates increases with the number of evaluated molecules, for any given computation budget. Indeed, we can spend the time that we spare from an increase in computation efficiency by evaluating more ligands. In the context of the EuroHPC project LIGATE[3], we address this challenge by porting in SYCL 2020 the computation kernels [6] and out-of-kernel optimizations [2]. We validated LiGen using NVIDIA and AMD-based GPUs [1].

The second challenge is posed by the operational effort required to carry out the computation. LiGen, as most of the HPC applications [4], hinge on MPI to scale over the available nodes. When an MPI process aborts due to a fault, the default response by the standard is to kill the remaining processes. In the context of an extreme-scale virtual screening campaign, it means that we will lose all the results that have not been stored in the file system. Since IO is a scarce resource in the HPC system, it is common to accumulate the output, either in RAM [12] or in the node local storage [8], before storing it on the shared file system. To solve this problem, the EXSCALATE platform divides the virtual screening campaign into smaller tasks, using custom reactive tools to execute them through the system job scheduler. While effective, this solution needs to be tailored for each HPC center and requires human supervision with technical skills that are not common among domain experts. For this reason, in the context of the EuroHPC project LIGATE, we integrated LiGen into the LEXIS Platform [9]. The latter enables access to HPC systems and automatizes workflows through an intuitive web interface. Moreover, it can abstract the execution across different HPC systems, either on-premises or European supercomputer centers. In the case of an urgent computing scenario, it is possible to spawn the virtual screening campaign across multiple locations without requiring a dedicated queue.

The remainder of the paper is structured as follows. Section 2 focuses on the performance portability challenges, providing more technical details on how LiGen can run efficiently on a wide range of HPC-grade GPUs and scale over a large number of nodes. Then, Section 3 focuses on the second challenge, describing the technologies behind the LEXIS Platform and how it can be used to carry out a virtual screen campaign. Finally, Section 4 concludes the paper.

[1] Scientific papers and campaign results: `https://www.exscalate4cov.eu/contribute.html`
[2] TOP500 list: `https://www.top500.org`
[3] Website: `https://www.ligateproject.eu`

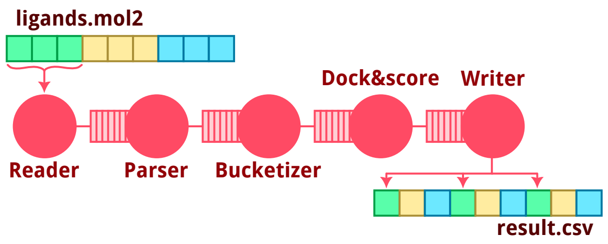


Fig. 1. Example of an MPI process that implements a LiGen computation pipeline for virtual screening. All the stages are represented by red circles, while input queues are by slim red rectangles. The input files are partitioned by a single IO request, and color-coded according to the rank of the MPI process that issues the request. In the example, we consider the MPI process related to the green color.

2. Performance portability in virtual screening

This section describes the LiGen application and how it can scale across the supercomputing resources. Then, we focus on the computation kernels and their SYCL porting, highlighting their optimization on the Karolina supercomputer at IT4i. Finally, we measure how LiGen can run efficiently on different HPC-grade GPUs.

2.1. The LiGen virtual screening application

LiGen is a `C++17` application that uses MPI to scale across different nodes. The idea is to use a single MPI process within each node that spawns the same asynchronous computation pipeline. Each stage of the pipeline uses at least one CPU thread to take a ligand from its input queue, perform a computation, and then enqueue the ligand in the next stage. The input queues have a maximum size. This means that the slower stage generates back-pressure on the queues stalling the previous stages. If we assign to the slower stage a number of CPU threads equal to the number of hardware threads, on average, we will use the node to compute the most expensive operation, which is the desired scenario. The only stages that require synchronization with the other MPI processes are the ones that perform IO operations.

In the virtual screening scenario, we have two types of input files. One type includes all the files required to describe the target protein(s). The second one represents the chemical library that we need to virtual screen. The output file decorates the input molecules with the interaction strength against the target protein(s). LiGen considers the target protein(s) as constant, thus it will read them using MPI IO collective functions at the beginning of the execution, providing the same information to all the MPI processes. In this way, all the stages can have access to target(s) without any need for synchronization. On the other hand, LiGen statically splits the ligand input file among the available processes based on the file size. For performance reason [12], each MPI process uses private IO operations to read the input file at a pace that depends on the throughput of the slower pipeline stage. To minimize the imbalance between different MPI processes, the EXSCALATE platform uses a pre-processing step to uniform the complexity within the input file [7]. However, we need to synchronize the write operation, to avoid that MPI processes will overwrite each other. In this case, we hinge on the fact that the order among ligands of the input file is not important. In particular, the user can select how many MPI processes will collect the output and issue the actual IO operation to the file system.

Figure 1 depicts an example of a pipeline to perform virtual screening. In the example, three MPI processes replicate the same pipeline that works on different slabs of the input file. The first stage reads a chunk of text that contains the description of zero, one, or more ligands, depending on the chunk size, which is a configuration parameter [12]. The next stage parses the description of each ligand to create and populate the related data structures that can be used by all the other stages. Then, a bucketizer stage uses input features to cluster ligands that have similar sizes, in terms number of heavy atoms and rotatable bonds. Once the bucketizer fills a bucket of ligands it will forward them to the dock & score stage that performs the actual computation. Finally, the writer stage will generate a CSV file that relates a molecule with the interaction strength against the target(s).

LiGen uses MPI to scale across different nodes, and `std::thread` to scale within the cores of a node. In particular, it is possible to configure the number of threads that carry out the computation of each stage. They use work stealing for the input queue of ligands. Usually, it is the dock & score stage that defines the pace of the whole pipeline. This implies that the thread that executes the writer stage will be idle waiting for input, while all the threads involved in the

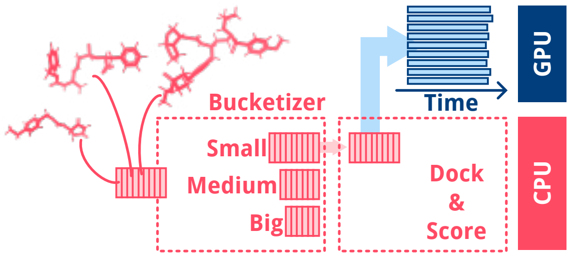


Fig. 2. Interaction between the bucketizer and dock & score stage. The red part represents CPU computation, while the blue part represents the GPU computation.

reader, parser, and bucketizer stages will be waiting due to the back pressure. In this scenario, all the nodes involved in the computation will be busy docking and scoring for most of the time, which is the desired outcome.

To address the hardware heterogeneity, the dock & score stage may use different implementations to carry out the computation. Besides the `C++17` one, we can offload computation to GPUs using a CUDA and a SYCL implementation. At the beginning of the execution, LiGen will allocate memory on each available GPU, setting up a double buffering approach. Then, when a CUDA or SYCL implementation needs to compute a bucket, it will handle memory transfer and computation on the first available GPU. As a rule of thumb, we need to use a number of dock & score threads, that use a CUDA or SYCL implementation, equal to twice the number of available GPUs, to benefit from double buffering. By using this approach we can hinge on all the resources available in a computation node.

2.2. The computation kernel structure and SYCL implementation

The first LiGen version that could offload computation on a GPU, was using a CUDA implementation that spread the computation of a ligand on all the GPU resources. It seldom happens that a ligand is big enough to use all the resources of an HPC-grade GPU. For this reason, each thread that uses the CUDA implementation in the dock & score stage computes the ligand in its own CUDA stream. The main issue with this approach is that the synchronization required by some kernels hindered the computation efficiency[23].

To achieve full GPU occupancy, we shifted from this latency-oriented approach to a throughput one. In the throughput version, a batch of ligands is computed in parallel at a given point in time. The main difference is that we use few resources to compute each ligand. While the execution time required to process a single ligand increases, the overall throughput is better [23]. To improve computation efficiency, ligands with similar resource usage are packed into the same batch, also called buckets. At each moment in time, the GPU is computing only one bucket. This approach can yield a throughput that is 5× the classical one if we can fine-tune batches to reach full GPU occupancy [2]. In our case, input features like the number of ligand atoms, influence the resources required to compute a ligand. Therefore, the size and number of batches depend on the input feature and how we choose to cluster them. To reach full GPU occupancy, we need to compute as many ligands in parallel as they fit in the GPU hardware and make sure that their computation will last for a similar amount of time [2].

Figure 2 shows how the bucketizer and dock & score stage solve this problem. In particular, the bucketizer bundles all the ligands that have a similar number of heavy atoms and rotatable bonds in the same bucket. Indeed, these input features have a strong correlation with the execution time [7]. When we look at the distribution of the number of atoms in a dataset, it is very common to find ligands with 30-40 atoms, but there are also few ligands with more than 150 atoms. The higher the number of ligand atoms, the higher the resource usage. For this reason, the computation kernels have non-type template parameters that define the maximum number of atoms. To increase the number of ligands that run in parallel, each kernel is instantiated for a suitable maximum number of atoms. The idea is that given the same GPU resource, batches containing smaller ligands may pack a higher number of ligands[2]. At compile time, LiGen fixes the ranges of atom numbers a kernel has to compute, thus the batch number. While at runtime, LiGen evaluates the running environment and the resources available, selecting how many ligands we can pack in a batch [2]. This step is of paramount importance to automatically tailor the execution on the underlying hardware.

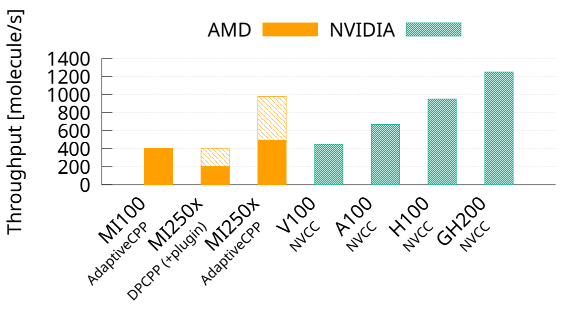


Fig. 3. LiGen average throughput, by varying target GPU, language, and compiler. The performance refers to a single physical card. For the MI250x GPU, we need to consider two virtual GPUs to make a fair comparison.

In the LIGATE European project, we developed a SYCL porting of LiGen [5] to unlock performance portability on several EuroHPC supercomputing centers. This development and optimization activity has been carried out using mainly NVIDIA cards. To validate the performance portability approach, we had access to LUMI-G, which ships AMD cards [1].

2.3. Performance evaluation

If we consider the best-performing supercomputers ranked in the TOP500 list, we can notice how the largest fraction of their throughput hinge on accelerators. In almost all cases, the accelerator is a GPU. In the past only NVIDIA cards were available, leading to an effort of porting existing code in the CUDA language. However, modern supercomputers use GPUs from different vendors, with their own native language.

In the case of urgent computing, where it is important to run a virtual screening campaign on a supercomputer, we need to handle this hardware heterogeneity. This section aims at evaluating and comparing the LiGen performance on a wide range of HPC-grade GPUs, to prove that we can reach performance portability. Figure 3 reports the average throughput, in terms of ligands screened per second. This measures the time of the whole execution, taking into account also communication overhead with the file system and the GPU memory. To make a fair comparison, we consider only a single physical card. In the case of the AMD MI250x, we need to take into account that each physical card is composed of two logical ones, that's why we have highlighted the related bar with two different patterns. The main takeaway message from this experiment is the relative difference in throughput between the different HPC-grade GPUs. When we focus on each vendor, we can notice how the performance will grow after each new generation. If we compare the GPU of different vendors, we can notice how the AMD MI250x, which is the GPU card equipped by the LUMI-G supercomputer, is greater than the NVIDIA A100, which is the GPU card equipped by the Karolina supercomputer. This result confirms that we reached performance portability across European supercomputers since the two cards were launched in a similar period of time. Moreover, we had a grant for a development access call on the Karolina supercomputer that we used to optimize and tune the LiGen performance, in terms of both quality and performance. Furthermore, we used a benchmark access call on LUMI to measure the performance and check whether the performance was portable, on a completely different supercomputer [1]. However, if we compare the FLOPS declared by the two GPUs, we can notice how the AMD card is supposed to yield a much higher throughput than the NVIDIA one. For example, we noticed how the SYCL compiler produces a high register pressure [5]. This hints that there is still room for improvement in the SYCL implementation.

3. Orchestrate an extreme-scale virtual screening campaign

This section describes more in detail how the LEXIS Platform can orchestrate a virtual screening campaign. We begin with a general description of the LEXIS Platform [4] [9] and its features, including generic workflow orchestration

[4] LEXIS Platform documentation - `https://docs.lexis.tech`

and data movement across multiple sites. Finally, we show how the user launches the screening campaign through the platform's web interface.

3.1. The LEXIS Platform

The LEXIS Platform provides services for complex computing workflow orchestration and distributed data management across connected HPC sites. The platform provides an API and a web-based user interface for easy and unified access to multiple powerful HPC resources. The workflows orchestrated by the platform can be described as directed acyclic graphs (DAGs) that consist of various dependent tasks that trigger and monitor data movement, Kubernetes container, and HPC batch job operations. The orchestrator used by the platform is built on top of Apache Airflow [10], which uses Python files to describe the workflow task dependencies and their parameters. The platform also allows a declarative approach to workflow description based on YAML files using the TOSCA standard [18]. The Apache Airflow has a concept of DAG runs, which can be seen as instances of a particular workflow with a set of parameter values defined at the beginning of the execution. Thus, executions of the same workflow with various parameter values can be tracked in a unified way.

The LEXIS Platform defines an abstraction to work with different compute projects, user groups, and resource allocations. The basic element of the platform is a LEXIS Project, which has its own set of users with various associated roles and a set of resource allocations. Each project has its own datasets, workflows, and users. The resource abstraction is used by an HPC-as-a-Service middleware called HEAppE [22] deployed near each connected HPC cluster to manage credentials and access to particular HPC applications through HEAppE Command Templates. These templates are essentially parameterized job scripts used to launch only a precisely defined set of commands on the clusters through the HEAppE API which is used by the LEXIS Orchestrator. In this way, the user is completely isolated from the cluster-specific configuration and does not have the possibility to directly access the executable files of the application. Technical details of this implementation are out of the scope of this paper and can be found in [9]. These templates are usually created by an HPC application specialist who can use them to hide site-specific options (i.e. MPI parameters) inside from the regular users of the application.

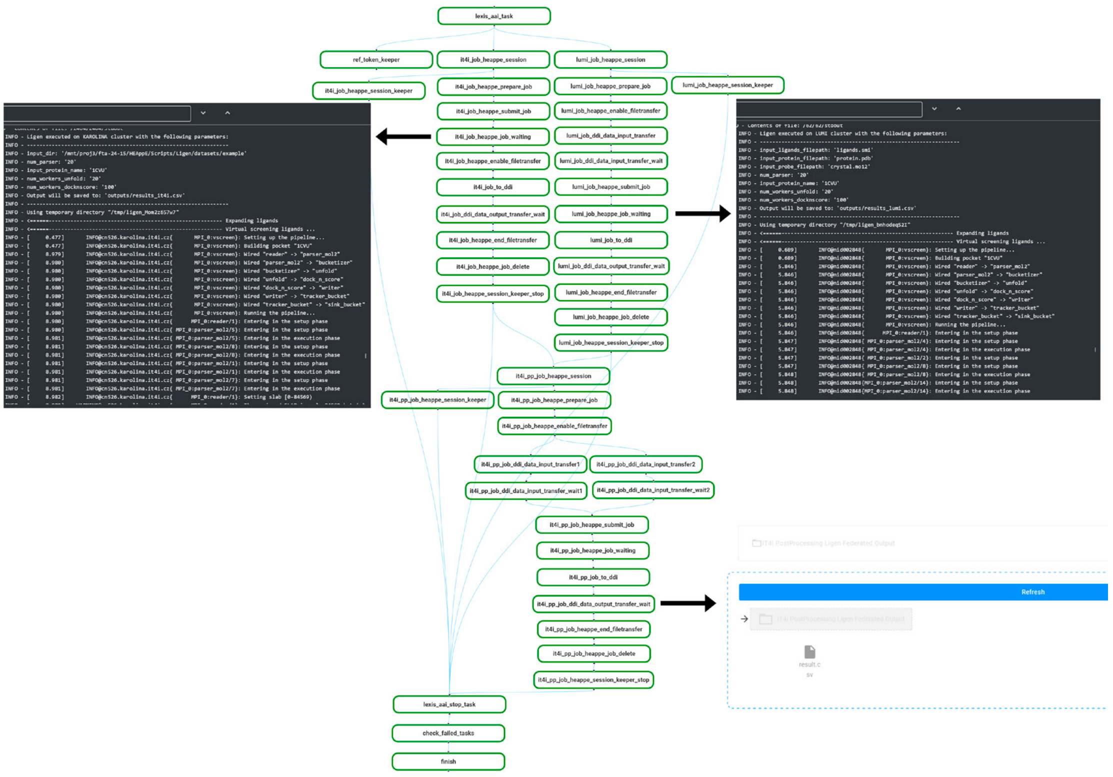
The described infrastructure is used by the LEXIS Orchestrator during the execution of the workflow. Thanks to the branching capability provided by the Airflow DAGs, various situations can be handled, such as job failures at a particular site. The DAG can contain a branch that handles this situation by triggering a transfer of all available checkpoint data to another site and triggering a job submission there. The simulation then loads the transferred checkpoints and continues with its execution. The workflow also contains cleanup tasks which are usually triggered only for successful execution, leaving the logs and potential by-products to be examined by a specialist in case of job failure (displayed in Figure 4). The user has the possibility, not only in case of DAG failure, to check the *stdout* generated by the applications through the graphical user interface and a part of the DAG can be re-executed if needed.

3.2. Data movement

The LEXIS Platform provides data movement capabilities through its distributed data interface (DDI). The DDI has functions for data transfer between the user and a target site, as well as staging features for moving data to and from HPC clusters and other compute resources. The data in the DDI are defined as datasets, which can be seen as a classical directory with a rich set of metadata [16]. These metadata are indexed in a centralized Elastic Search index and at the same time in iRODS zones which are typically hosted by the HPC sites. The data is stored inside the iRODS collections. The DDI then uses a set of custom-built APIs and asynchronous worker processes that perform the actual data movement using native protocols. For example, data transfers between locations use native iRODS protocol, which leverages parallel TCP streams, native POSIX protocols common in the HPC environment such as SFTP or RSync and HTTPS chunked transfers for web-based front-ends or similar clients.

These functionalities are leveraged by the multi-site workflow of LiGen, where the input data are stored as DDI datasets, and reference to them is passed to the workflow at execution time. The workflow then uses the DDI capabilities to trigger dataset movement to a target location. The same mechanism is used to store the result dataset along with the necessary metadata. The DDI recognizes multiple levels of access to the datasets:

- *user* – datasets are visible only by a particular user in a given LEXIS Project;


lexis_aai_task
ref_token_keeper
it4i_job_heappe_session
lumi_job_heappe_session
it4i_job_heappe_session_keeper
it4i_job_heappe_prepare_job
lumi_job_heappe_prepare_job
lumi_job_heappe_session_keeper
it4i_job_heappe_submit_job
lumi_job_heappe_enable_filetransfer
it4i_job_heappe_job_waiting
lumi_job_ddi_data_input_transfer
it4i_job_heappe_enable_filetransfer
lumi_job_ddi_data_input_transfer_wait
it4i_job_to_ddi
lumi_job_heappe_submit_job
it4i_job_ddi_data_output_transfer_wait
lumi_job_heappe_job_waiting
it4i_job_heappe_end_filetransfer
lumi_job_to_ddi
it4i_job_heappe_job_delete
lumi_job_ddi_data_output_transfer_wait
it4i_job_heappe_session_keeper_stop
lumi_job_heappe_end_filetransfer
lumi_job_heappe_job_delete
lumi_job_heappe_session_keeper_stop
it4i_pp_job_heappe_session
it4i_pp_job_heappe_session_keeper
it4i_pp_job_heappe_prepare_job
it4i_pp_job_heappe_enable_filetransfer
it4i_pp_job_ddi_data_input_transfer1
it4i_pp_job_ddi_data_input_transfer2
it4i_pp_job_ddi_data_input_transfer_wait1
it4i_pp_job_ddi_data_input_transfer_wait2
it4i_pp_job_heappe_submit_job
it4i_pp_job_heappe_job_waiting
it4i_pp_job_to_ddi
it4i_pp_job_ddi_data_output_transfer_wait
it4i_pp_job_heappe_end_filetransfer
it4i_pp_job_heappe_job_delete
it4i_pp_job_heappe_session_keeper_stop
lexis_aai_stop_task
check_failed_tasks
finish
Refresh
result.csv

```
INFO - Ligen executed on KAROLINA cluster with the following parameters:
INFO - ---------------------------------------------------------------------
INFO - input_dir: '/mnt/proj3/fta-24-15/HEAppE/Scripts/Ligen/datasets/example'
INFO - num_parser: '20'
INFO - input_protein_name: '1CVU'
INFO - num_workers_unfold: '20'
INFO - num_workers_docknscore: '100'
INFO - Output will be saved to: 'outputs/results_it4i.csv'
INFO - ---------------------------------------------------------------------
INFO - Using temporary directory "/tmp/ligen_Mom2z657w7"
INFO - <======------------------------------------------------------------ Expanding ligands
INFO - <======------------------------------------------------------------ Virtual screening ligands ...
INFO - [     0.477]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Setting up the pipeline...
INFO - [     0.477]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Building pocket "1CVU"
INFO - [     8.979]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Wired "reader" -> "parser_mol2"
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Wired "parser_mol2" -> "bucketizer"
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Wired "bucketizer" -> "unfold"
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Wired "unfold" -> "dock_n_score"
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Wired "dock_n_score" -> "writer"
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Wired "writer" -> "tracker_bucket"
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Wired "tracker_bucket" -> "sink_bucket"
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{       MPI_0:vscreen}: Running the pipeline...
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{      MPI_0:reader/1}: Entering in the setup phase
INFO - [     8.980]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/5}: Entering in the setup phase
INFO - [     8.981]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/5}: Entering in the execution phase
INFO - [     8.981]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/8}: Entering in the setup phase
INFO - [     8.981]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/8}: Entering in the execution phase
INFO - [     8.981]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/1}: Entering in the setup phase
INFO - [     8.981]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/1}: Entering in the execution phase
INFO - [     8.981]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/7}: Entering in the setup phase
INFO - [     8.981]       INFO@cn526.karolina.it4i.cz{ MPI_0:parser_mol2/7}: Entering in the execution phase
INFO - [     8.982]       INFO@cn526.karolina.it4i.cz{      MPI_0:reader/1}: Setting slab [0-84569)
```

```
INFO - Ligen executed on LUMI cluster with the following parameters:
INFO - ---------------------------------------------------------------------
INFO - input_ligands_filepath: 'ligands.smi'
INFO - input_protein_filepath: 'protein.pdb'
INFO - input_probe_filepath: 'crystal.mol2'
INFO - num_parser: '20'
INFO - input_protein_name: '1CVU'
INFO - num_workers_unfold: '20'
INFO - num_workers_docknscore: '100'
INFO - Output will be saved to: 'outputs/results_lumi.csv'
INFO - ---------------------------------------------------------------------
INFO - Using temporary directory "/tmp/ligen_bnhodeq52I"
INFO - <======------------------------------------------------------------ Expanding ligands
INFO - <======------------------------------------------------------------ Virtual screening ligands ...
INFO - [     0.689]       INFO@nid002848{       MPI_0:vscreen}: Setting up the pipeline...
INFO - [     0.689]       INFO@nid002848{       MPI_0:vscreen}: Building pocket "1CVU"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Wired "reader" -> "parser_mol2"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Wired "parser_mol2" -> "bucketizer"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Wired "bucketizer" -> "unfold"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Wired "unfold" -> "dock_n_score"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Wired "dock_n_score" -> "writer"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Wired "writer" -> "tracker_bucket"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Wired "tracker_bucket" -> "sink_bucket"
INFO - [     5.846]       INFO@nid002848{       MPI_0:vscreen}: Running the pipeline...
INFO - [     5.846]       INFO@nid002848{      MPI_0:reader/1}: Entering in the setup phase
INFO - [     5.847]       INFO@nid002848{ MPI_0:parser_mol2/4}: Entering in the setup phase
INFO - [     5.847]       INFO@nid002848{ MPI_0:parser_mol2/4}: Entering in the execution phase
INFO - [     5.847]       INFO@nid002848{ MPI_0:parser_mol2/2}: Entering in the setup phase
INFO - [     5.847]       INFO@nid002848{ MPI_0:parser_mol2/8}: Entering in the setup phase
INFO - [     5.848]       INFO@nid002848{ MPI_0:parser_mol2/8}: Entering in the execution phase
INFO - [     5.848]       INFO@nid002848{MPI_0:parser_mol2/14}: Entering in the setup phase
INFO - [     5.848]       INFO@nid002848{MPI_0:parser_mol2/14}: Entering in the execution phase
```

Fig. 4. Example LEXIS workflow with HPC application logs, and workflow results stored in LEXIS DDI

- *project* – shared among all project users;
- *public* – available to all logged-in users of the LEXIS Platform.

Each site used in execution has an HEAppE Middleware and a DDI worker process deployed (usually on a virtual machine). The HEAppE is used apart from the HPC job management and also for data transfer credentials management by the DDI worker process.

3.3. *Interaction with the user*

The user interaction with the LiGen through the LEXIS Platform is done using a web-based interface. A regular user of LiGen wants to have the capability to specify input data, launch the LiGen application on multiple HPC sites without the need to interact with a particular HPC cluster or job scheduler and download and examine the results. The user can log in to the LEXIS Portal using one of the identity providers (EUDAT B2ACCESS [5] or MyAccessID [6]) which cover most of the European institutional identities as well as some of the public commercial ones.

Once logged in, the user can request access to a particular LEXIS Project, which has the LiGen workflow associated, as well as the allocation on one of the desired HPC clusters. The owner of the LEXIS Project needs to approve the user's request. Once approved, the user sees the LiGen workflow in the workflow list, as well as project and public datasets. The user may choose to upload their own dataset or use one of the existing ones.

[5] B2ACCESS: https://eudat.eu/service-catalogue/b2access
[6] MyAccessID: https://wiki.geant.org/display/MyAccessID

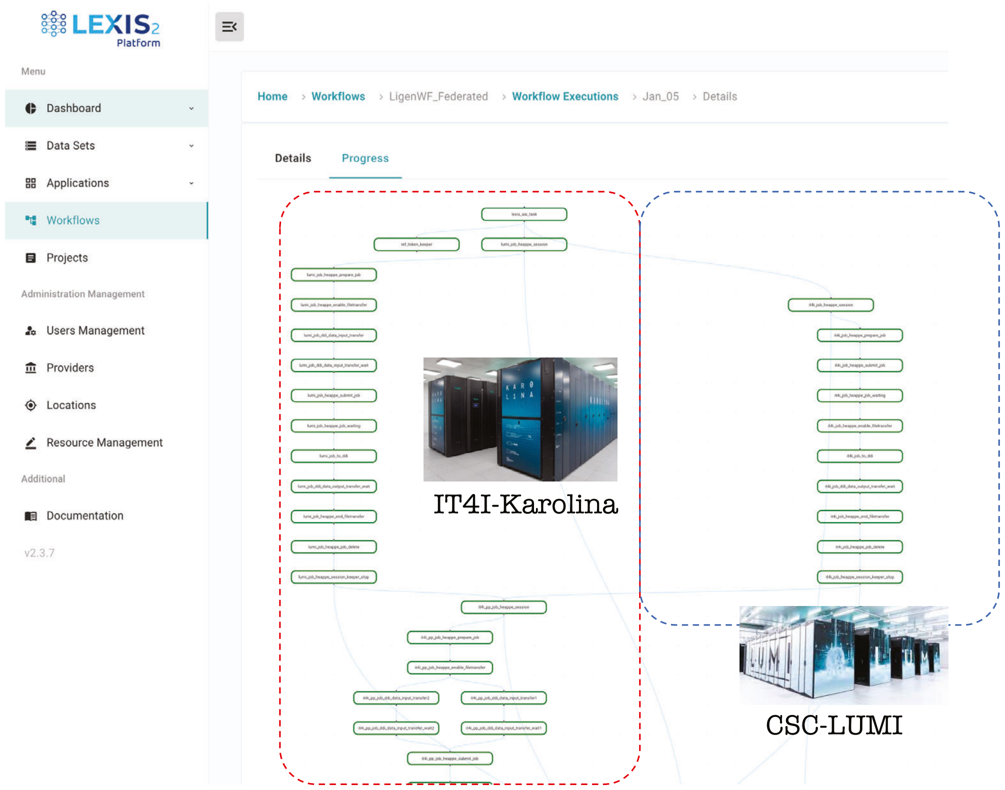


Fig. 5. LEXIS multisite execution of the LiGen-based Drug-Discovery workflow on IT4Innovations-Karolina and CSC-LUMI supercomputers

Once done, the user selects a particular workflow for execution, selects the input datasets, and may change a set of the input parameter values of the application. This workflow is tied to two locations, the Karolina petascale (located at IT4Innovations in Ostrava, Czech Republic) and LUMI pre-exascale (located at CSC's data center in Kajaani, Finland) supercomputers. The principal investigator for each resource allocation must approve this association in advance. Details of this process are beyond the scope of this paper. The screenshot from the LEXIS Portal containing workflow execution is displayed in Figure 5. It is also possible to use a dynamic selection of locations using smart scheduler capabilities, through the selection of a scheduling policy. In both cases, the LEXIS Project must have compute time allocations on all involved HPC sites associated. When we consider the virtual screening workflow for urgent computing, we could reach up to 12M ligands per second of throughput if we target all computation nodes on the LUMI and Karolina supercomputing centers. In an extreme-scale experiment for urgent computing, the size of the input ligands is in the order of TBs in SMILES format [7]. However, the data staging on each location happens before starting the computation, without impacting the throughput.

Finally, the user fills out the necessary parameters and triggers the execution. The execution progress can be monitored in the web-based UI as well as logs from a particular HPC job can be read for debugging purposes if needed. Once finished, the execution detail contains a link to a result dataset, which can be downloaded through the web browsers and examined. This approach enables an end-user to use HPC resources and third-party software without having direct access to them. This permits on one side to offer a virtual screening-as-a-service platform to researchers. On the other side, it protects the intellectual properties of the pharmaceutical company, as in our scenario.

4. Conclusion

Virtual screening is becoming an important stage in drug discovery for suggesting which are the candidates to the test *in-vitro*. Since the problem is embarrassingly parallel and the size of the chemical library that we want to virtual screen is limited only by the available computation power, HPC systems are the ideal candidate to carry out the computation. In this paper, we focus on the challenge of performance portability across heterogeneous hardware and, in the context of urgent computing, orchestrate an extreme-scale virtual screening campaign. In particular, we show how we are addressing them in LiGen, the EXSCALATE platform virtual screening application. From experimental results, we can notice how a SYCL implementation enables LiGen to be deployed on supercomputers that also have a non-NVIDIA GPU. If we compare its performance with the native hand-optimized CUDA version on NVIDIA GPU, we can notice that LiGen run uses the resources efficiently, even if there is still room for improvement. Moreover, the usage of the LEXIS Platform provides an elegant way for domain experts to perform an extreme-scale virtual screening campaign on one or more European supercomputers.

Acknowledgements

This project has received funding from EuroHPC-JU - the European High-Performance Computing Joint Undertaking - under grant agreement No 956137 (LIGATE). The JU receives support from the European Union's Horizon 2020 research and innovation program and Italy, Sweden, Austria, the Czech Republic, and Switzerland. We also acknowledge EuroHPC-JU for awarding us access to Karolina at IT4Innovations, Czech Republic (EHPC-DEV-2021D02-049), and to LUMI-G at CSC, Finland (EHPC-BEN-2022B12-001). This work was also partly-supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).

References

[1] Accordi, G., Gadioli, D., Palermo, G., Crisci, L., Carpentieri, L., Cosenza, B., Beccari, A.R., 2024a. Unlocking performance portability on lumi-g supercomputer: A virtual screening case study, in: Proceedings of the 12th International Workshop on OpenCL and SYCL, Association for Computing Machinery, New York, NY, USA. URL: https://doi.org/10.1145/3648115.3648125, doi:10.1145/3648115.3648125.

[2] Accordi, G., Gadioli, D., Vitali, E., Crisci, L., Cosenza, B., Beccari, A., Palermo, G., 2024b. Out of kernel tuning and optimizations for portable large-scale docking experiments on GPUs. The Journal of Supercomputing , 1–18.

[3] Allegretti, M., Cesta, M.C., Zippoli, M., Beccari, A., Talarico, C., Mantelli, F., Bucci, E.M., Scorzolini, L., Nicastri, E., 2022. Repurposing the estrogen receptor modulator raloxifene to treat SARS-CoV-2 infection. Cell Death & Differentiation 29, 156–166.

[4] Bernholdt, D.E., Boehm, S., Bosilca, G., Gorentla Venkata, M., Grant, R.E., Naughton, T., Pritchard, H.P., Schulz, M., Vallee, G.R., 2020. A survey of mpi usage in the us exascale computing project. Concurrency and Computation: Practice and Experience 32, e4851. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4851, doi:https://doi.org/10.1002/cpe.4851, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.4851. e4851 cpe.4851.

[5] Crisci, L., Carpentieri, L., Cosenza, B., Accordi, G., Gadioli, D., Vitali, E., Palermo, G., Beccari, A.R., 2024. Enabling performance portability on the ligen drug discovery pipeline. Future Generation Computer Systems URL: https://www.sciencedirect.com/science/article/pii/S0167739X24001195, doi:https://doi.org/10.1016/j.future.2024.03.045.

[6] Crisci, L., Salimi Beni, M., Cosenza, B., Scipione, N., Gadioli, D., Vitali, E., Palermo, G., Beccari, A., 2022. Towards a portable drug discovery pipeline with sycl 2020, in: Proceedings of the 10th International Workshop on OpenCL, Association for Computing Machinery, New York, NY, USA. URL: https://doi.org/10.1145/3529538.3529688, doi:10.1145/3529538.3529688.

[7] Gadioli, D., Vitali, E., Ficarelli, F., Latini, C., Manelfi, C., Talarico, C., Silvano, C., Cavazzoni, C., Palermo, G., Beccari, A.R., 2022. EXSCALATE: an extreme-scale virtual screening platform for drug discovery targeting polypharmacology to fight SARS-CoV-2. IEEE Transactions on Emerging Topics in Computing 11, 170–181.

[8] Glaser, J., Vermaas, J.V., Rogers, D.M., Larkin, J., LeGrand, S., Boehm, S., Baker, M.B., Scheinberg, A., Tillack, A.F., Thavappiragasam, M., Sedova, A., Hernandez, O., 2021. High-throughput virtual laboratory for drug discovery using massive datasets. The International Journal of High Performance Computing Applications 35, 452–468. doi:10.1177/10943420211001565.

[9] Golasowski, M., Martinovič, J., Křenek, J., Slaninová, K., Levrier, M., Harsh, P., Derquennes, M., Donnat, F., Terzo, O., 2022. The lexis platform for distributed workflow execution and data management, in: HPC, Big Data, and AI Convergence Towards Exascale. Taylor & Francis.

[10] Harenslak, B., de Ruiter, J., 2021. Data Pipelines with Apache Airflow. Manning Publications. ISBN: 9781617296901.

[11] López, N., Debbio, L.D., Baaden, M., Praprotnik, M., Grigori, L., Simões, C., Bogaerts, S., Berberich, F., Lippert, T., Ignatius, J., Lavocat, P., Pineda, O., Giuffreda, M.G., Girona, S., Kranzlmüller, D., Resch, M.M., Scipione, G., Schulthess, T., 2021. Lessons learned from urgent computing in europe: Tackling the covid-19 pandemic. Proceedings of the National Academy of Sciences 118, e2024891118. doi:10.1073/pnas.2024891118.

[12] Markidis, S., Gadioli, D., Vitali, E., Palermo, G., 2021. Understanding the i/o impact on the performance of high-throughput molecular docking, in: 2021 IEEE/ACM Sixth International Parallel Data Systems Workshop (PDSW), pp. 9–14. doi:10.1109/PDSW54622.2021.00007.

[13] Matter, H., Sotriffer, C., 2011. Applications and Success Stories in Virtual Screening. John Wiley & Sons, Ltd. chapter 12. pp. 319–358. ISBN: 9783527633326.

[14] Mayr, L.M., Fuerst, P., 2008. The future of high-throughput screening. SLAS Discovery 13, 443–448. URL: https://www.sciencedirect.com/science/article/pii/S2472555222082260, doi:https://doi.org/10.1177/1087057108319644.

[15] Morgan, S., Grootendorst, P., Lexchin, J., Cunningham, C., Greyson, D., 2011. The cost of drug development: A systematic review. Health Policy 100, 4–17. URL: https://www.sciencedirect.com/science/article/pii/S0168851010003659, doi:https://doi.org/10.1016/j.healthpol.2010.12.002.

[16] Munke, J., Hayek, M., Golasowski, M., García-Hernández, R.J., Donnat, F., Koch-Hofer, C., Couvee, P., Hachinger, S., Martinovič, J., 2022. Chapter 4 Data System and Data Management in a Federation of HPC/Cloud Centers. Taylor & Francis.

[17] Murugan, N.A., Podobas, A., Gadioli, D., Vitali, E., Palermo, G., Markidis, S., 2022. A review on parallel virtual screening softwares for high-performance computers. Pharmaceuticals 15, 63.

[18] OASIS, 2020. Tosca simple profile in yaml version 1.3. URL: https://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.3/TOSCA-Simple-Profile-YAML-v1.3.pdf.

[19] Pagadala, N.S., Syed, K., Tuszynski, J., 2017. Software for molecular docking: a review. Biophysical reviews 9, 91–102.

[20] Polishchuk, P.G., Madzhidov, T.I., Varnek, A., 2013. Estimation of the size of drug-like chemical space based on gdb-17 data. Journal of computer-aided molecular design 27, 675–679.

[21] Su, M., Yang, Q., Du, Y., Feng, G., Liu, Z., Li, Y., Wang, R., 2019. Comparative assessment of scoring functions: The casf-2016 update. Journal of Chemical Information and Modeling 59, 895–913. doi:10.1021/acs.jcim.8b00545.

[22] Svaton, V., Martinovic, J., Krenek, J., Esch, T., Tomancak, P., 2020. Hpc-as-a-service via heappe platform, in: Barolli, L., Hussain, F.K., Ikeda, M. (Eds.), Complex, Intelligent, and Software Intensive Systems, Springer International Publishing. pp. 280–293. doi:10.1007/978-3-030-22354-0_26.

[23] Vitali, E., Ficarelli, F., Bisson, M., Gadioli, D., Accordi, G., Fatica, M., Beccari, A.R., Palermo, G., 2024. GPU-optimized approaches to molecular docking-based virtual screening in drug discovery: A comparative analysis. Journal of Parallel and Distributed Computing 186, 104819. URL: https://www.sciencedirect.com/science/article/pii/S0743731523001892, doi:https://doi.org/10.1016/j.jpdc.2023.104819.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 52–60

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Quantum ESPRESSO towards performance portability: GPU offload with OpenMP ☆

Fabrizio Ferrari Ruffino[a], Laura Bellentani[b], Giacomo Rossi[c], Fabio Affinito[b], Stefano Baroni[d], Oscar Baseggio[d], Pietro Delugas[d], Paolo Giannozzi[e,a], Jakub Kurzak[f], Ye Luo[g], Ossian O'Reilly[f], Sergio Orlandini[b], Ivan Carnimeo[d,*]

[a]*CNR-IOM Istituto dell'Officina dei Materiali, area SISSA, via Bonomea 265, 34136 Trieste, Italy*
[b]*CINECA, Via Magnanelli 6/3, 40033 Casalecchio di Reno, BO, Italy*
[c]*Intel Corporation Italia S.p.A., via Santa Maria Valle 3, 20123 Milano, Italy*
[d]*SISSA, Scuola Internazionale Superiore di Studi Avanzati, via Bonomea 265, 34136 Trieste, Italy*
[e]*Dipartimento di Scienze Matematiche, Informatiche e Fisiche (DMIF), Università degli Studi di Udine, via delle Scienze 206, 33100 Udine, Italy*
[f]*Advanced Micro Devices, Inc., 2485 Augustine Drive, Santa Clara, CA 95054, USA*
[g]*Computational Science Division, Argonne National Laboratory, 9700 S Cass Ave, Lemont, IL 60559, USA*

Abstract

In recent years, significant strides have been made to enable Quantum ESPRESSO (QE) to operate on heterogeneous HPC architectures, effectively harnessing the computational capabilities of GPUs. Initially, emphasis has been placed on NVIDIA-based hardware and software infrastructure. Nonetheless, the HPC landscape is multifaceted and intricate, prompting the project to prioritize achieving peak performance across various GPU models. To this end, efforts are being directed towards empowering QE to leverage AMD and Intel GPUs through OpenMP offload directives. In this paper, coding approach and benchmark tests of the OpenMP porting of Plane-Wave Self-Consistent Field (`PWSCF`) code is presented and discussed.

Keywords: GPU; HPC; OpenMP offload; PWSCF; QE; Quantum ESPRESSO

☆ The submitted manuscript has been co-authored by UChicago Argonne, LLC, Operator of Argonne National Laboratory ("Argonne"). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan. http://energy.gov/downloads/doe-public-access-plan.

* Corresponding author.
E-mail address: icarnimeo@sissa.it

1. Introduction

Heterogeneous architectures based on GPU-accelerators have become a *de facto* standard in High Performance Computing (HPC), and the majority of the world's top exascale and pre-exascale machines are now equipped with GPUs [1]. In this context, molecular and material sciences are evolving in parallel with the advances in computer science and most electronic structure codes have either been adapted to accelerated architectures or are in the process of being so. Enabling quantum materials discovery and design towards exascale and extreme scaling performance on present and future HPC machines is one of the pillars of the MaX "MAterials design at the eXascale" Centre of Excellence for Supercomputing applications [2], of which Quantum ESPRESSO [3, 4, 5, 6, 7] stands as one of the lighthouse codes.

A significant challenge hindering this enabling is the varied landscape of hardware and software stacks provided by different vendors. Notably, each vendor offers specific technologies such as compilers, libraries, profilers, and languages, which must be incorporated into the codes to be optimized for a particular architecture. In this complex environment, a primary goal of scientific software development is to enhance the performance portability of the code across different architectures. An overview of performance comparison among different GPU porting approaches on different architectures can be found in the paper by Davis et al. [8].

The first porting of the Quantum ESPRESSO suite dates back to several years ago [4] and was fully based on CUDA Fortran in view of deployment on NVIDIA hardware and software stack. At a later stage [3], the suite was refactored using a mixed CUDA Fortran/OpenACC language model and all the most important codes of the suite (`PWSCF`, `CP` [9], `PHonon` [10, 11], `turbo_eels` [12, 13, 14, 15], `HP` [16, 17, 18]) were accelerated with a performance CPU-GPU speed-up[1] ranging between 4x and 6x for linear response codes, higher for ground-state codes. At the time of writing this paper, the porting of `turbo_davidson` [19], `turbo_lanczos` [20] and `turbo_magnons` [21, 22, 23] codes has been also finalized. Noticeably, the simplicity and minimal intrusiveness nature of OpenACC directives have simplified the code development process, lowered entry barriers for new developers, and facilitated software stack maintenance.

Recently, in order to target a more diverse range of architectures, in particular based on Intel and AMD hardware and software stack, a new porting of the Quantum ESPRESSO suite based on OpenMP offload has been initiated. The OpenMP offload has been developed using directives based on the OpenMP API 5.1 standard [24], and from here onwards will be referred to as "OpenMP5" porting model. Currently, the development of `PWSCF` is about to be completed (branch `develop_omp5` in the official Quantum ESPRESSO repository [25]). The new porting model is designed to seamlessly accommodate both OpenACC and OpenMP5 directives in a single source code, enabling deployment on various hardware architectures (CPU, NVIDIA GPUs, and Intel/AMD GPUs) determined at compile time. This approach aims at achieving consistent performance across different GPU cards at runtime.

In this communication, we provide a brief overview of the porting model and discuss the computational performance and speed-up of `PWSCF`, based on calculation tests conducted on Leonardo (CINECA) and LUMI (CSC) EuroHPC superclusters.

2. Code development

The new OpenACC/OpenMP5 porting model for `PWSCF` is based on one unique source code, where two partially separated execution paths have been defined with directives. The first path is oriented to NVIDIA hardware and software stack and is based on an high level layer of OpenACC directives, an intermediate layer of CUDA Fortran libraries (e.g., FFTXlib, LAXlib and other libraries managing MPI and BLAS/LAPACK wrappers) and a low level layer of vendor's numerical libraries (e.g., cuSOLVER, cuFFT, cuBLAS, cuRAND). The second path is oriented to AMD and Intel hardware and software stack and is fully based on OpenMP5 directives, with the noticeable exception of a small HIP code portion in FFTXlib, that will be discussed in the following. The low level layer of vendor's

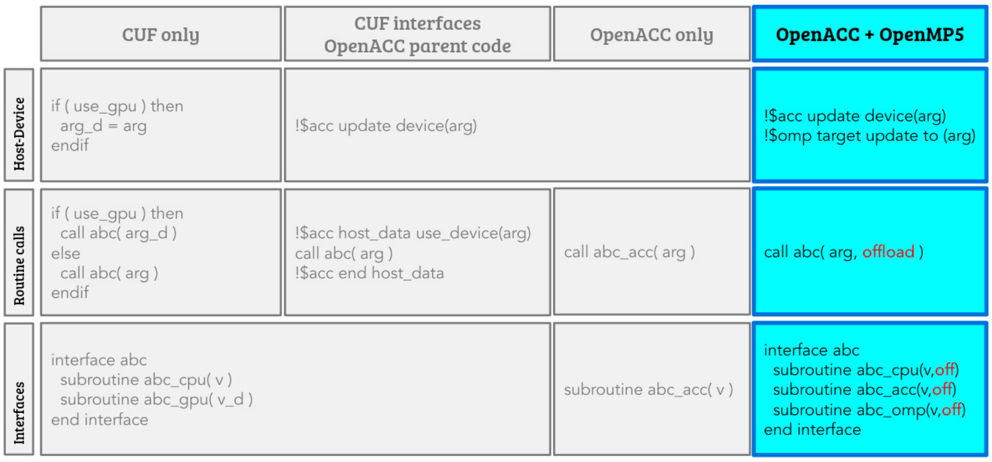
[1] The CPU-GPU speed-up was calculated by comparing executions on a CPU-based HPC machine (Galileo100 at CINECA) with GPU-based ones (Marconi100 at CINECA and Selene at Nvidia), utilizing an equal number of nodes for each comparison.

numerical libraries relies especially on oneAPI and ROCm utilities, that provide efficient Fast Fourier Transform (FFT) and linear algebra operations on GPU on AMD and Intel accelerators.

In order to simplify code development and deployment, the OpenACC and OpenMP5 execution paths are selected at compile time and are currently mutually exclusive, being not interfaced with each other (i.e., it is not possible to link oneAPI and ROCm backends when `PWSCF` is compiled with OpenACC, and, vice versa, it is not possible to link CUDA libraries when `PWSCF` is compiled with OpenMP5). Both `configure` and `cmake` compilation tools have been enabled.

A critical point in developing the mixed OpenACC/OpenMP5 porting model was how to tailor execution for CPU, CUF/OpenACC, and OpenMP5 paths, especially when each path necessitates routines specifically designed for a particular architecture. For instance, let us consider the case where distinct specialized algorithms are employed for CPU and GPU executions, and GPU execution is then further customized with a CUDA Fortran (or OpenACC) implementation, tailored for NVIDIA architectures, and an OpenMP5 implementation, for Intel/AMD architectures. Additionally, since variables can be stored in host or device memories, the CPU execution should be kept accessible also when the code is compiled with GPU flags. In the Quantum ESPRESSO suite, this scenario typically occurs, for example, in the FFTXlib library [26], for 3d FFTs, and in the `becmod` module, that contains many numerically intensive procedures with heavy matrix-matrix multiplications.

In Figure 1 very schematic pseudo-codes are represented to illustrate different strategies followed to cope with this issue. In the old purely CUDA Fortran approach [4], Fortran interfaces could recognize the `DEVICE` attribute of variables, invoking the right subroutine procedure accordingly. For instance, in the leftmost block of Figure 1, `v_d` is declared `DEVICE` in `abc_gpu`, and the type of the `arg/arg_d` variable in the parent code is sufficient to determine whether to invoke `abc_gpu` or `abc_cpu`. In order to resolve Fortran interfaces with OpenACC and OpenMP5 based models (rightmost block of Figure 1), we have defined three distinct derived types, one for each different execution path (CPU, CUF/OpenACC, OpenMP5). The `off` argument in `abc_cpu`, `abc_acc`, `abc_omp` is then declared with the specific type. Consequently, depending on the type of the `offload` flag in the parent code, a particular execution path is chosen.



	CUF only	CUF interfaces OpenACC parent code	OpenACC only	OpenACC + OpenMP5

Host-Device	if (use_gpu) then
	  arg_d = arg
	endif
		!$acc update device(arg)		!$acc update device(arg)
				!$omp target update to (arg)

Routine calls	if (use_gpu) then
	  call abc(arg_d)
	else
	  call abc(arg)
	endif
		!$acc host_data use_device(arg)
		call abc(arg)
		!$acc end host_data
			call abc_acc(arg)	call abc(arg, offload)

Interfaces	interface abc
	  subroutine abc_cpu(v)
	  subroutine abc_gpu(v_d)
	end interface
			subroutine abc_acc(v)	interface abc
				  subroutine abc_cpu(v,off)
				  subroutine abc_acc(v,off)
				  subroutine abc_omp(v,off)
				end interface

Fig. 1. Schematic representation of different GPU code implementation schemes.

Regarding performance, the majority of the computational cost of a typical `PWSCF` calculation is due to FFTs. Three-dimensional FFTs in the Quantum ESPRESSO suite are executed using FFTXlib [26], a specialized library tailored to accommodate the internal data distribution of the wavefunction (and charge density). It leverages the properties of DFT-related datasets, such as band structure, cutoff, and dual parameter and supports both slab and pencil decompositions. The former method involves a slab-based partition of the direct space, where the input function undergoes transformation across the entire x-y domain for a subset of values along the z-axis. Each subset is allocated to a specific processor, and the Fourier transform along the z-direction is carried out based on the 'z-stick' distribution of the reciprocal space: for each x-y point falling within the cutoff circle, the function is transformed across the entire z-range. This algorithm entails one 2d FFT operation for x-y slab transforms, one stage of collective communication (typically MPI all to all), which involves distributing the z-sticks among processors, and one final 1d FFT operation on the z-sticks. The uniqueness of the Quantum ESPRESSO version of this algorithm lies in the z-stick decomposition of the reciprocal space and the utilization of the energy cutoff. This mapping of the square grid covering the direct space into a smaller one inscribed within the cutoff sphere allows for memory and data movement optimizations.

Pencil decomposition works similarly to slab decomposition, but transforms along the x and y directions separately, involving only 1d FFT operations. While it enables more efficient memory distribution, it requires an additional stage of communication compared to slab decomposition. Consequently, with the increasing memory availability of modern GPU devices, slab decomposition has become the preferred method in recent years.

Figure 2a provides a schematic overview of the main steps of an inverse FFT computation, performed with slab decomposition on 4 MPI processes: even on small systems, the time spent in communication and data movement significantly exceeds that related to actual 1d and 2d FFT computations.

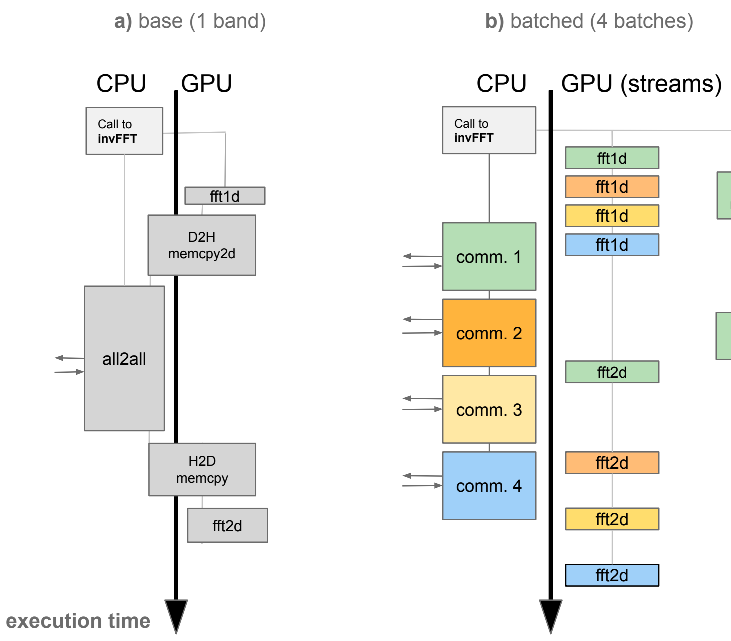


Fig. 2. Illustrative scheme of FFT with slab decomposition: a) base accelerated inverse FFT algorithm; b) asynchronous batched inverse FFT algorithm with 5 streams and 4 sub-batches (one for each color). The main data movements are done with memcpy2d and memcpy routines. MPI communications without GPU-aware MPI are performed among CPUs and require host-device synchronizations, whereas when GPU-aware MPI is enabled they involve GPU-direct device-device communications. The time scale of the two diagrams is not the same.

The GPU porting of FFTs for NVIDIA-based architectures [4, 3] leveraged the extensive internal parallelism of GPU cards to concurrently process many bands at a time within each FFT operation. This was achieved by segmenting the wavefunction data structures into batches and sub-batches, each containing a fixed number of bands, and using streams to process sub-batches asynchronously. This algorithm, initially developed using CUDA Fortran [4], has not undergone rewriting in OpenACC during the recent refactoring [3]. Instead, it has been expanded to support AMD-based architectures using the HIP language, since OpenMP5 does not allow to work on multiple GPU streams within a single CPU task. For Intel architectures, FFTXlib has been currently ported using the base algorithm shown in Figure 2a, and work is currently underway to include the asynchronous streams.

In the batched algorithm, schematically summarized in Figure 2b, all the 1d and 2d FFTs, needed to fully Fourier transform a batch, are performed on one dedicated stream (usually stream 0), while data movement and copies are performed on different streams, one for each sub-batch. Other small operations are performed on stream 0 too, by means of explicit HIP kernels. MPI communications are called for each sub-batch asynchronously with respect to the computation and the data movements related to all the other sub-batches. In this way, the overlap in time between communication and computation is brought out at its maximum, taking into account the constraints coming from the finite bandwidth of CPU and GPU communications.

Work is currently still in progress to finalize the OpenMP5 porting, in particular to optimize eigensolver operations.

3. Numerical results

The developments described in the previous section allowed to significantly accelerate FFTs and basic matrix operations in PWSCF on Intel and AMD GPU-based machines. Such operations are especially relevant to compute the application of the Kohn-Sham Hamiltonian to a generic function (h_psi subroutine in PWSCF), that is the core of the iterative methods used in plane-wave based electronic structure codes. For example, in the CPU execution of PWSCF shown in Figure 3, h_psi took around 75% of one SCF interation. In this section we will discuss numerical performance of PWSCF, comparing CPU and GPU executions on Galileo100, Leonardo and LUMI machines, whose main technical features are briefly summarized in Table 1.

HPC cluster	Galileo100	Leonardo	LUMI
Processors	2 x Intel Xeon E5-2697	Intel Ice Lake	AMD EPYC™ 7A53
Cores	36	32	64
RAM	128 GB	512 GB DDR4	512 GiB DDR4
Accelerators	-	4 x NVIDIA A100 GPUs with 64 GB HBM2	4 x AMD Instinct™ MI250X GPU modules with 2 x 64 GB of HBM memory
Intra-node links	-	NVLink 3.0 (200 GB/s)	In-package Infinity Fabric™ (400 GB/s), and single/double Infinity Fabric (100/200 GB/s)
Network	Intel OmniPath, 100 Gb/s	DragonFly+ 200 Gbps of 2x dual port NVIDIA Mellanox Infiniband HDR100	PCIe links to the Slingshot-11 (200 GB/s)

Table 1. Systems specification for the CPU (Galileo100) and GPU (Leonardo and LUMI) partitions

Calculations on CPU have been performed on Galileo100 with the official release `qe-v7.2` [27], compiled with Intel Ifort 2021.5.0, Intel MPI 2021.5 and Intel MKL version 2022.0.0; GPU runs have been performed on Leonardo and LUMI machines. Simulations on Leonardo have been done with the official release `qe-v7.3.1` [28], compiled with NVHPC 23.1 and CUDA 11.8. On LUMI we employed the development version of PWSCF in the `develop_omp5` branch [25], based on the `qe-v7.2` release, and available on the official Quantum ESPRESSO repository on GitLab. The code has been compiled with CRAY HPE 15.0.1 and ROCm 5.2.5.

Figure 3 shows that the performance of the OpenMP5 version of h_psi on LUMI is equivalent to that of the corresponding OpenACC version on Leonardo. The test case is a reduced version of the CrI_3 system previously benchmarked in Ref.[3], with an orthorhombic cell of 480 atoms, for a total of 3240 electrons, with 1944 bands and a per-band FFT grid of dimension (120,192,640), which make it a good candidate to assess the effectiveness of the FFT parallelization via *R&G* distribution and band-batching. The executions on LUMI are performed by progressively enabling improvements of the FFT algorithm with respect to the "base" one (cf. Figure 2a). Calculations labeled as "many" use the batched FFT algorithm (cf. Figure 2b); GPU-MPI awareness is enabled for the "many aware" results. The CPU-GPU speed-up of the last two bars is also in line with other calculations published in another work [3], using the same machine (Galileo100) as CPU reference.

In Figure 4 the scaling over plane waves (*R&G*) on LUMI, at fixed number of pools, is shown for the same system, and it is noteworthy that using the batched FFT algorithm and GPU-aware MPI it is possible to decrease communication bottlenecks and scale the calculations beyond 2 nodes.

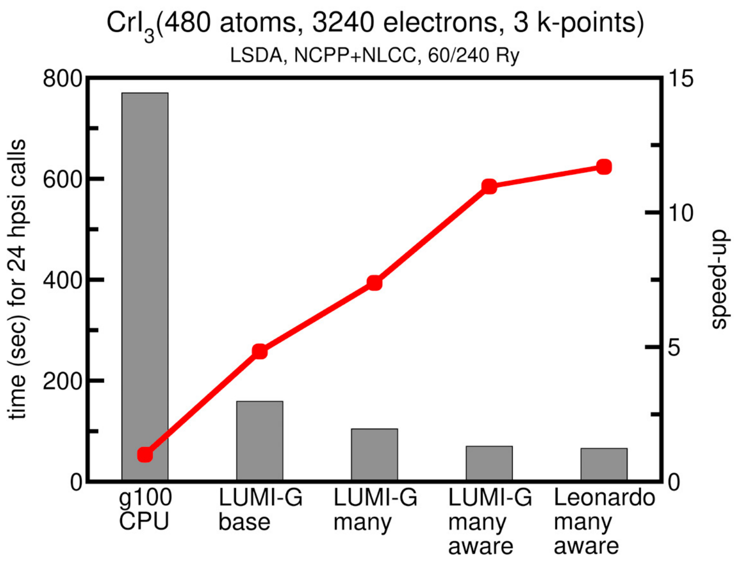


Fig. 3. Wall time of `h_psi` calls in one SCF iteration of CrI_3 system, performed on different HPC clusters using 2 nodes. Speed-up on the right axis is computed with respect to CPU time. "many" refers to the batched FFT algorithm, "aware" refers to GPU-aware MPI, "base" is without neither "many" nor "aware".

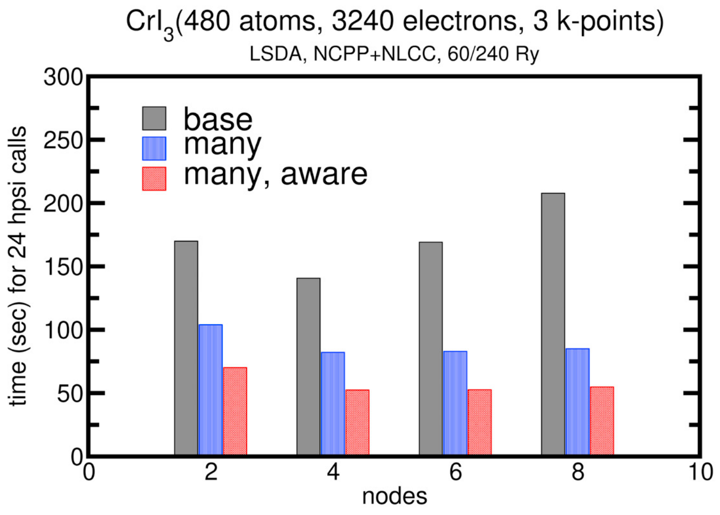


Fig. 4. *R&G* scaling of `h_psi` over the number of nodes in LUMI cluster. "many" refers to the batched FFT algorithm, "aware" refers to GPU-aware MPI, "base" is without neither "many" nor "aware".

In Figure 5 the performance of `h_psi` with respect to the number of pools (independent groups of k-points distributed among MPI processes) is shown, using one pool per node, for the CsI test case[29]. This system has a cubic CsCl crystal structure, with 768 electrons, 461 bands and a per-band FFT grid of dimension `(180,180,135)`. Its limited memory footprint allows distributing the workload with R&G parallelization intra-node only, while, thanks to the large number of available k-points, the system can scale with pools up to 89 nodes.

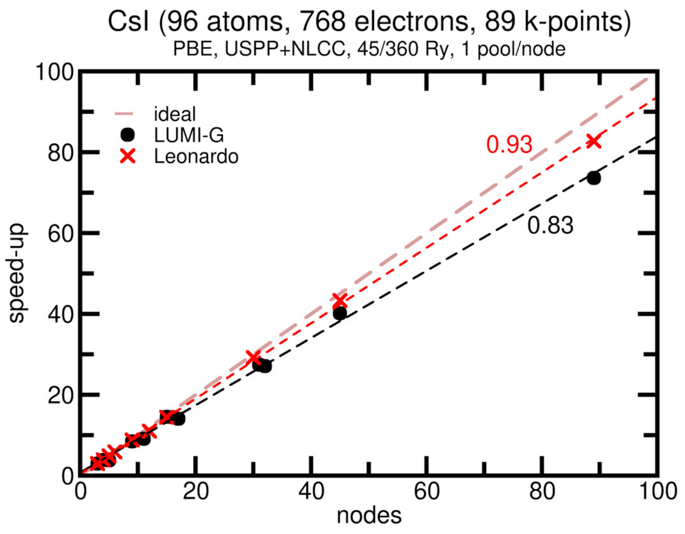


Fig. 5. Speed-up of pool scaling of `h_psi` in LUMI and Leonardo cluster (using one pool per node). Slope of the linear regression is reported for each dataset.

The scaling shown here is in line with the speed-up of pool parallelism shown in another previous work [3], and the slope of the regression is reasonably similar between Leonardo and LUMI executions.

4. Conclusions

The main numerically intensive parts of `PWSCF`, in particular FFT and basic linear algebra operations such as matrix multiplications, have been ported to GPU using OpenMP5, in order to address Intel and AMD cards and software stack. OpenMP5 directives have been seamlessly integrated in the previous CUF/OpenACC code, resulting in a unique source code that can be tailored at compile time for execution on CPU, NVIDIA and Intel/AMD hardware and software stack. Performance portability of `h_psi`, one of the most computationally heavy steps of the SCF calculation (i.e., the application of the Kohn-Sham Hamiltonian to a generic function), has been tested with calculations on Galileo100, LUMI and Leonardo clusters. In such tests, very similar computational times and speed-ups have been found between the two GPU execution paths of `PWSCF` (CUF/OpenACC and OpenMP5), both for *R&G* and pools parallelism.

The porting of other parts of `PWSCF` with OpenMP5, particularly with regard to the eigensolver, is still underway. When using the accelerated version of `PWSCF`, the main bottleneck of OpenMP5 executions on LUMI remains the eigensolver, and overcoming this step is crucial to approaching performance portability between the OpenACC and OpenMP5 versions of the code, although absolute parity may be limited by intrinsic differences in hardware architectures.

Concerning Intel GPUs, an asynchronous algorithm for batched FFTs fully based on OpenMP offload is currently under development. Since explicit streams are not supported in OpenMP, it is based on `nowait` and `depend` clauses, and the batch related workload is distributed among OpenMP tasks. GPU streams associated to each sub-batch are implicitly defined according to each task.

Acknowledgements

This work was funded by the European Union through the MaX ”MAterials design at the eXascale” Centre of Excellence for Supercomputing applications (Grant agreement No. 101093374, co-funded by the European High Performance Computing joint Undertaking (JU) and participating countries) and by the Italian MUR, through the PRIN project ARES (grant number 2022W2BPCK) and the Italian National Centre for HPC, Big Data, and Quantum Computing (grant No. CN00000013), founded within the Next Generation EU initiative. The authors acknowledge EuroHPC JU Development Access Call for granting computational resources on LUMI cluster (EHPC-DEV-2023D06-013 and EHPC-DEV-2021D04-129 projects). The authors IC, FFR, LB, OB acknowledge CSC for technical support received at the LUMI GPU / Nomad CoE hackathon Sept 4-6, 2023. YL was supported by the Office of Science, U.S. Department of Energy, under contract DE-AC02-06CH11357.

References

[1] Top 500 list of supercomputers. https://www.top500.org (last access Apr, 16th 2024).

[2] MaX: Materials at the eXascale. An EU Centre of Excellence for Supercomputing Applications. https://www.max-centre.eu (last access Apr, 16th 2024).

[3] Ivan Carnimeo, Fabio Affinito, Stefano Baroni, Oscar Baseggio, Laura Bellentani, Riccardo Bertossa, Pietro Davide Delugas, Fabrizio Ferrari Ruffino, Sergio Orlandini, Filippo Spiga, and Paolo Giannozzi. Quantum ESPRESSO: One further step toward the exascale. *Journal of Chemical Theory and Computation*, 19(20):6992–7006, 2023. PMID: 37523670.

[4] Paolo Giannozzi, Oscar Baseggio, Pietro Bonfà, Davide Brunato, Roberto Car, Ivan Carnimeo, Carlo Cavazzoni, Stefano de Gironcoli, Pietro Delugas, Fabrizio Ferrari Ruffino, Andrea Ferretti, Nicola Marzari, Iurii Timrov, Andrea Urru, and Stefano Baroni. Quantum ESPRESSO toward the exascale. *J. Chem. Phys.*, 152(15):1–11, 04 2020. 154105.

[5] P. Giannozzi, O. Andreussi, T. Brumme, O. Bunau, M. Buongiorno Nardelli, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, M. Cococcioni, N. Colonna, I. Carnimeo, A. Dal Corso, S. de Gironcoli, P. Delugas, R. A. DiStasio, Jr., A. Ferretti, A. Floris, G. Fratesi, G. Fugallo, R. Gebauer, U. Gerstmann, F. Giustino, T. Gorni, J. Jia, M. Kawamura, H-Y Ko, A. Kokalj, E. Kucukbenli, M. Lazzeri, M. Marsili, N. Marzari, F. Mauri, N. L. Nguyen, H-V Nguyen, A. Otero-de-la Roza, L. Paulatto, S. Ponce, D. Rocca, R. Sabatini, B. Santra, M. Schlipf, A. P. Seitsonen, A. Smogunov, I. Timrov, T. Thonhauser, P. Umari, N. Vast, X. Wu, and S. Baroni. Advanced capabilities for materials modelling with Quantum ESPRESSO. *J. Phys. Condens. Matter*, 29(46):1–30, 2017. 465901.

[6] Paolo Giannozzi, Stefano Baroni, Nicola Bonini, Matteo Calandra, Roberto Car, Carlo Cavazzoni, Davide Ceresoli, Gui do L Chiarotti, Matteo Cococcioni, Ismaila Dabo, Andrea Dal Corso, Stefano de Gironcoli, Stefano Fabris, Guido Fratesi, Ralph Gebauer, Uwe Gerstmann, Christos Gougoussis, Anton Kokalj, Michele Lazzeri, Layla Martin-Samos, Nicola Marzari, Francesco Mauri, Riccardo Mazzarello, Stefano Paolini, Alfredo Pasquarello, Lorenzo Paulatto, Carlo Sbraccia, Sandro Scandolo, Gabriele Sclauzero, Ari P Seitsonen, Alexander Smogunov, Paolo Umari, and Renata M Wentzcovitch. Quantum ESPRESSO: a modular and open-source software project for quantum simulations of materials. *J. Phys. Condens. Matter*, 21(39):1–19, sep 2009. 395502.

[7] Sandro Scandolo, Paolo Giannozzi, Carlo Cavazzoni, Stefano de Gironcoli, Alfredo Pasquarello, and Stefano Baroni. First-principles codes for computational crystallography in the Quantum ESPRESSO package. *Zeitschrift für Kristallographie*, 220(5-6):574–579, 2005.

[8] Joshua H. Davis, Pranav Sivaraman, Joy Kitson, Konstantinos Parasyris, Harshitha Menon, Isaac Minn, Giorgis Georgakoudis, and Abhinav Bhatele. Taking GPU programming models to task for performance portability, 2024.

[9] P. Giannozzi, F. De Angelis, and R. Car. First-principle molecular dynamics with ultrasoft pseudopotentials: Parallel implementation and application to extended bioinorganic systems. *The Journal of Chemical Physics*, 120(13):5903–5915, 04 2004.

[10] S. Baroni, S. de Gironcoli, A. Dal Corso, and P. Giannozzi. Phonons and related crystal properties from density-functional perturbation theory. *Rev. Mod. Phys.*, 73(2):515–562, 2001.

[11] P. Giannozzi, S. De Gironcoli, P. Pavone, and S. Baroni. Ab-initio calculation of phonon dispersions in semiconductors. *Phys. Rev. B*, 43(9):7231–7242, 1991.

[12] Iurii Timrov, Nathalie Vast, Ralph Gebauer, and Stefano Baroni. Electron energy loss and inelastic x-ray scattering cross sections from time-dependent density-functional perturbation theory. *Phys. Rev. B*, 88:064301, Aug 2013.

[13] Iurii Timrov, Nathalie Vast, Ralph Gebauer, and Stefano Baroni. turboeels-a code for the simulation of the electron energy loss and inelastic x-ray scattering spectra using the Liouville-Lanczos approach to time-dependent density-functional perturbation theory. *Comp. Phys. Comm.*, 196:460–469, 2015.

[14] Oleksandr Motornyi, Michele Raynaud, Andrea Dal Corso, and Nathalie Vast. Simulation of electron energy loss spectra with the turboeels and thermo_pw codes. In *XXIXTH IUPAP Conference on Computational Physics CCP2017*, volume 1136 of *J. Phys. Conf. Ser.* IUPAP, 2018. 39th IUPAP Conference on Computational Physics (CCP), Univ Pierre Marie Curie Sorbonne, Paris, France, JUL 09-13, 2017.

[15] Oleksandr Motornyi, Nathalie Vast, Iurii Timrov, Oscar Baseggio, Stefano Baroni, and Andrea Dal Corso. Electron energy loss spectroscopy of bulk gold with ultrasoft pseudopotentials and the Liouville-Lanczos method. *Phys. Rev. B*, 102:1–12, 2020. 035156.

[16] Iurii Timrov, Nicola Marzari, and Matteo Cococcioni. HP – A code for the calculation of Hubbard parameters using density-functional perturbation theory. *Comp. Phys. Comm.*, 279:1–17, 2022. 108455.

[17] Iurii Timrov, Nicola Marzari, and Matteo Cococcioni. Hubbard parameters from density-functional perturbation theory. *Phys. Rev. B*, 98:1–15, Aug 2018. 085127.

[18] Iurii Timrov, Nicola Marzari, and Matteo Cococcioni. Self-consistent Hubbard parameters from density-functional perturbation theory in the ultrasoft and projector-augmented wave formulations. *Phys. Rev. B*, 103:1–14, Jan 2021. 045141.

[19] Brent Walker, A. Marco Saitta, Ralph Gebauer, and Stefano Baroni. Efficient approach to time-dependent density-functional perturbation theory for optical spectroscopy. *Phys. Rev. Lett.*, 96:1–4, Mar 2006. 113001.

[20] Dario Rocca, Ralph Gebauer, Yousef Saad, and Stefano Baroni. Turbo charging time-dependent density-functional theory with Lanczos chains. *J. Chem. Phys.*, 128(15):1–14, 2008. 154105.

[21] Tommaso Gorni, Oscar Baseggio, Pietro Delugas, Stefano Baroni, and Iurii Timrov. Turbomagnon – A code for the simulation of spin-wave spectra using the Liouville-Lanczos approach to time-dependent density-functional perturbation theory. *Comp. Phys. Comm.*, 280:1–15, 2022. 108500.

[22] Tommaso Gorni, Oscar Baseggio, Pietro Delugas, Iurii Timrov, and Stefano Baroni. First-principles study of the gap in the spin excitation spectrum of the CrI_3 honeycomb ferromagnet, 2022.

[23] Pietro Delugas, Oscar Baseggio, Iurii Timrov, Stefano Baroni, and Tommaso Gorni. Magnon-phonon interactions enhance the gap at the Dirac point in the spin-wave spectra of CrI_3 2D magnets, 2021.

[24] OpenMP Architecture Review Board. OpenMP application program interface version 5.1, 2020.

[25] Quantum ESPRESSO development branch `develop_omp5`. https://gitlab.com/QEF/q-e/-/tags/qe-7.2-omp5-1.0 (last access Apr, 16th 2024).

[26] Michael Wagner, Victor Lopez, Julian Morillo, Carlo Cavazzoni, Fabio Affinito, Judit Gimenez, and Jesus Labarta. Performance analysis and optimization of the fftxlib on the intel knights landing architecture. In *2017 46TH International Conference on Parallel Processing Workshops (ICPPW)*, International Conference on Parallel Processing Workshops, pages 243–250. CRAY Supercomp Company; Intel; ARM; OCF; SGI; Univ Bristol; VirginiaTech; Int Assoc Comp & Communications, 2017. 46th International Conference on Parallel Processing Workshops (ICPPW), Bristol, England, AUG 14-17, 2017.

[27] Quantum ESPRESSO release 7.2. https://gitlab.com/QEF/q-e/-/tags/qe-7.2 (last access Apr, 16th 2024).

[28] Quantum ESPRESSO release 7.3.1. https://gitlab.com/QEF/q-e/-/tags/qe-7.3.1 (last access Apr, 16th 2024).

[29] Procurement repository of CINECA.
https://gitlab.hpc.cineca.it/procurement/tier1-tecnopolo/benchmarks/-/blob/main/QuantumESPRESSO/LEONARDO-datacentric/inputfiles/csi.in (last access Apr, 16th 2024).

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 61–69

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

An overview of the Legio fault resilience framework for MPI applications

Roberto Rocco[a,*], Elisabetta Boella[b], Daniele Gregori[b], Gianluca Palermo[a]

[a] *Politecnico di Milano - Dipartimento di Elettronica, Informazione e Bioingegneria, Via Giuseppe Ponzio 34, 20133 Milano, Italy*
[b] *E4 Computer Engineering SpA, Viale Martiri della Libertà 66, 42019 Scandiano, Italy*

Abstract

While the computational power of HPC clusters breached through the exascale milestone, it highlighted the need for some critical features like fault management. The current de facto standard for inter-process communication at scale, MPI, lacks proper fault management functionalities, precluding result production in the presence of faults. Many efforts have already addressed the problem, with the ULFM extension and Reinit proposal being the two works receiving the most attention. While powerful and effective, they require expertise from the user and significantly impact the application code. For this reason, we presented the Legio fault resilience framework, which can introduce graceful degradation properties in MPI applications with minimal changes. This work summarises the development and evolution of Legio, highlighting successful use cases and providing some tips for users who want to leverage it for their executions.

Keywords: MPI; Fault Tolerance; ULFM; Legio

1. Introduction

HPC clusters have just broken through the exascale milestone. This result has been achieved thanks to developments that pushed the boundaries of computational power, but also underlined the importance of some other aspects: fault management is one of these. While HPC clusters feature reliable and tested components operating in a stable environment, their sheer amount makes the probability that any failure happens non-negligible. HPC executions must be aware of this possibility and plan countermeasures. Otherwise, faults will escalate, compromising the production of any meaningful result, with a subsequent waste of computational resources and energy.

While fault management is an essential feature for HPC tools and executions, it is not currently included in the Message Passing Interface (MPI) [5], the de facto standard for inter-process communication at scale. Even the latest version of the MPI standard (v4.1), which introduced some fault isolation restrictions, does not specify the execution

* Corresponding author
E-mail address: roberto.rocco@polimi.it

behaviour after fault insurgence. On the other hand, dealing with faults presence is becoming a more and more concerning topic for MPI users, as shown in an MPI users survey for the US Exascale Computing Project [2]. In that survey, only 12% of the participants showed no worry about fault tolerance.

Alongside this result, the survey also showed how users intend to deal with fault presence in their applications. The approach receiving the most votes is Checkpoint and Restart (C/R), which saves the execution state periodically so that the run can resume upon a fault [12, 1]. Checkpointing is an effective solution to mitigate the damage caused by a fault: rather than restarting from the beginning, thus losing all the computation, we lose what follows the last checkpoint. Nonetheless, C/R does not provide a method to continue with the execution, nor does it have negligible overhead, nor is it scalable: coordinated checkpointing causes a remarkable load on the disk, while uncoordinated checkpointing provides fewer guarantees, so we may have to store multiple checkpoints per process. All these reasons imply that C/R, while being a versatile and effective solution, does not entirely solve the problem of fault presence in HPC.

In the literature, many efforts addressed the problem and proposed solutions to overcome the limitations of the MPI standard. While the earliest ones tried to focus on constructing a fault-tolerant MPI implementation [4, 8, 24], they received limited efforts and are almost unused. Currently, the main focuses of the MPI fault tolerance Working Group (responsible for the introduction of fault tolerance functionalities in the upcoming versions of the standard) are the User Level Fault Mitigation (ULFM) extension [3] and the Reinit proposal [13]. The first one provides new functions that can repair the damage caused by faults in MPI communicators, while the latter allows for a complete re-initialisation of the MPI communication layer. Both efforts enable the execution to continue past fault detection, introducing a potential benefit also in terms of energy consumption (avoiding the re-initialisation of the job). However, they require user expertise, as the application code must feature the new function calls in precise areas to handle faults correctly, and they also must plan the monitoring and management functionalities carefully, or they risk losing all the benefits in terms of performance and energy consumption. This requirement for expertise, added to the fact that MPI applications tend to be stable and have a large code base (i.e., making them difficult to change), implies that leveraging the fault-tolerance functionalities is a non-trivial task even in the presence of ULFM or Reinit.

The difficulties in adopting the possibilities of ULFM and Reinit led to the development of frameworks that combine these solutions with C/R [9, 23, 25, 14]. Alongside those, we proposed the Legio fault resilience framework [22], not leveraging C/R but embracing graceful degradation for faster and more scalable recovery from faults. Using Legio, applications' executions featuring a fault will produce a different (approximated) result, as the work of the failed processes gets discarded. This behaviour is valid only for some MPI applications, like Monte Carlo solvers, but it performs better than a C/R solution because it removes the need to save and recover the application state.

In this paper, we want to summarise the developments of the Legio fault resilience framework over the last few years, pinpointing the principles that drove its development. We highlight all the MPI functionalities supported by Legio, with a brief overview of the techniques and algorithms that made it possible. We also show some successful Legio use cases, proving the tool's effectiveness in introducing fault resilience in MPI applications. Lastly, we offer some application analysis tips to potential users to simplify the integration of Legio with a future MPI application.

Overall, the contributions of this effort are the following:

- We overview the design process of the Legio fault resilience framework, showing all the new features introduced during the years;
- We show some successful use cases of integration between applications and Legio;
- We provide users with useful tips to integrate Legio with their MPI application.

The rest of the paper is structured as follows: Section 2 shows the design and implementation process of the Legio framework. Section 3 discusses all the evolutions and improvements of the framework over the years. Section 4 illustrates some successful use cases of introducing fault resilience properties with Legio. Section 5 provides valuable suggestions to the users, and Section 6 concludes the paper.

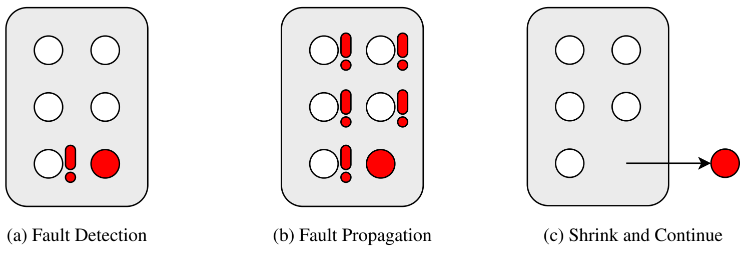


Fig. 1: An overview of the repair procedure. Fault repair starts when a process detects a fault in a collective function. After all processes become aware of the fault, Legio invokes the shrink function, which removes the failed process from the communicator. Communication can restart after recreating all the structures generated from the replaced communicator. Rank translation may be needed from now on.

2. The Legio framework rationale

While fault management is becoming increasingly critical in MPI applications, it comes with a cost. The complexity needed to integrate fault mechanisms in MPI applications is non-negligible, as programmers need knowledge about parallel and distributed algorithms. Moreover, the vastity of HPC applications' codes implies that a refactor to introduce fault mechanisms is a cumbersome task. Lastly, HPC applications are stable and validated; introducing changes would compromise these properties. For all these reasons, we focused on the simplicity of integration through transparency without strictly requiring changes to the code: with Legio, the application can remain unaware of faults in its executions, as they are all handled and masked by the framework.

Alongside preserving the stability and validation of HPC applications, we must also consider their performance. HPC applications are remarkably optimised and scalable, so our framework must not introduce excessive overheads and scalability issues, or we would compromise their usability. For this reason, we always kept the complexity of our recovery algorithms in mind and avoided the introduction of background threads as much as possible.

With transparency and efficiency, we also considered flexibility a target for our framework development. Rather than supporting a limited subset of MPI functionalities, we envisioned our framework to be modular and configurable. This decision helped us to develop the framework and ensure support for the MPI functionalities exploited by users' applications. This aspect also differentiates our solution from most other fault management frameworks in the literature, which would plan the supported functionalities considering some target applications rather than envisioning the entire MPI standard.

Following these three principles [19], we implemented the Legio fault resilience framework as a C++ library. Legio uses the PMPI profiling layer defined within the MPI standard, which allows us to catch all the MPI calls performed by the application. The main idea behind Legio is to substitute the structures handled by the application with others managed by the framework. This stratagem ensures that faults do not affect the application directly but stay within Legio, lowering the impact on the application code. Legio leverages ULFM functionalities to deal with faults but hides the complexity within the PMPI layer. Integration between Legio and MPI applications happens entirely through the linking phase, as the application does not have to perform specific calls to the Legio API.

What Legio does under the hood is relatively simple, thus introducing limited overhead. Legio replaces the MPI data structures created and used by the application with duplicates handled within the framework. In the absence of faults, the overhead is minimal (replacement happens through hash table lookups, so with constant complexity). In the presence of faults (i.e., abrupt process terminations like the ones considered by ULFM), Legio repairs its structures, excluding the failed processes and resuming the execution. Figure 1 illustrates the repair procedure. Afterwards, the execution continues as usual, except for some rank manipulation due to the different sizes of the application structures (they will still contain the failed processes) and Legio ones.

While the primary behaviour of Legio is relatively simple, we had to extend it to consider some tricky recovery policies and some complex MPI functionalities. The following Section covers all the improvements we made to the Legio framework over the years.

3. The Legio framework extensions

Starting from the basic concept described in the previous section, we developed the Legio fault resilience framework following the needs of applications and users, introducing support for MPI functionalities and addressing scalability and usability issues that we were experiencing using it. These efforts led us to support an extensive set of features with no counterpart in the literature. An outstanding example of this trend is the support for one-sided communication and file operations, which has been present in Legio since its defining paper [22]. We achieved support for those functionalities by discarding and recreating the affected data structures after repairing the communicators. While never explored in the literature, this solution allowed us to support these functionalities for applications using them.

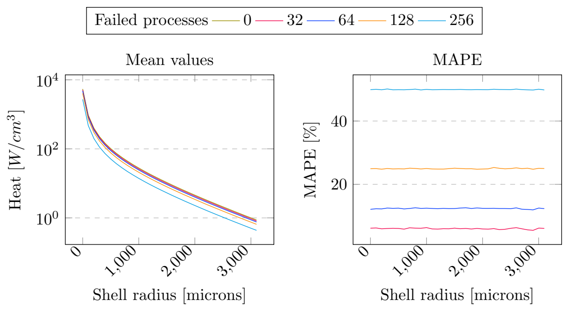
Our first extension to the Legio fault resilience framework proposal wanted to tackle the reported poor scalability of the shrink operation present in ULFM [10]. We use the shrink function to remove failed processes from communicators, which is critical in Legio's recovery procedure [22]. Its lack of scalability could compromise the time to repair executions, making stopping the execution more convenient. We addressed the issue by splitting the communicators into sub-groups, with the idea of repairing smaller communicators upon faults. We ensure communication using a star-shaped hierarchical communication topology: each sub-group has one leader, and all the leaders are part of a leader communicator that we can use to forward messages traveling through different sub-groups. Repairing the leader communicator is a rather complex task, but it is still faster than fixing the entire communicator, assuming the poor scalability of the shrink operation. Results show that this solution can reduce the recovery times at scale.

Alongside the analysis on reducing the scalability issues from the shrink ULFM function, we considered the group-collective communicator creation function, which is helpful in MPI applications to contain the synchronisation among processes. We found an issue preventing the repair in MPI applications leveraging those functionalities, and we developed a Liveness Discovery Algorithm (LDA), able to detect the presence of failed processes within a group without using collective operations [18]. With Legio and the LDA, we overcame the issue and opened the path towards group-collective agreement and shrink, possibly enhancing the possibilities of ULFM. The experimental results proved the efficiency and (logarithmic) scalability of our solution, and we are evaluating alternative implementations of the LDA (rather than leveraging a binomial tree, we are exploring the possibility of VCube [7] and k-nomial tree-shaped algorithms).

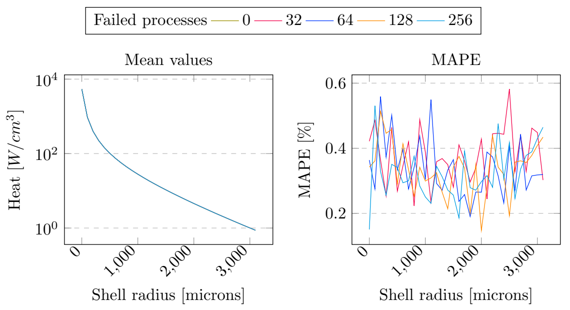
Implementing the LDA allowed us to explore new MPI features to support. In particular, we focused on the session model, a new initialisation and isolation paradigm that will become of significant importance in the upcoming versions of MPI. We addressed a possible problem that prevented the completion of MPI applications leveraging the session model, and we contained it with the Horizon communicator set concept [20]. Leveraging the Horizon set, we have a poll of communicators we can use to discover faults with the LDA, reducing the possibility of unrepairable faults in the executions. With the Horizon communicator set, Legio can introduce fault resilience properties in applications leveraging the new session model.

Legio repair procedure cannot handle all the faults by continuing only with the non-failed processes. In particular, an application may feature some processes whose completion is fundamental for producing a meaningful result. We refer to these processes as *critical* ones, and we developed the Subsidia extension to Legio, able to regenerate them upon fault [21]. Legio, empowered by the Subsidia extension, makes a compromise between a pure graceful degradation solution and a fault-tolerant one (achieved by C/R). This compromise allows us to reduce the amount of state to be stored while preserving the validity of the results. Overall, the introduction of the Subsidia extension changes the recovery procedure, as we now have to repair all communicators in case of faults. This change slightly impacts the performance of Legio, especially in the presence of unused communicators that would otherwise not be repaired. Nonetheless, the repair procedure for critical process faults is not particularly costly, except for the `MPI_Comm_spawn` call, whose OpenMPI implementation proved not scalable.

All these extensions to the Legio fault resilience framework made it what it is today. Legio is highly configurable, as the users can turn off the functionality they do not need for their executions, preserving performance. We will expand on configurability in Section 5 after showing some successful use cases in the following.



(a) Fault unaware



(b) Fault aware

Fig. 2: Accuracy analysis of the photon simulation application with different amounts of faults before (a) and after (b) the fault-awareness changes. We run those experiments on four IT4I Karolina nodes, each featuring 128 MPI processes. Source: [17]

4. Application use cases

During our studies on the Legio fault resilience framework, we applied its functionalities to many applications to measure the overhead it introduces in terms of execution times and to assess the impact on result accuracy. This section will cover some of our analyses, starting with a Monte Carlo Photon simulation and following with a stencil Particle In Cell (PIC) solver.

4.1. Monte Carlo photon simulation

The first application we considered simulates the behaviour of light (modeled as a collection of photons) in an infinite medium with isotropic scattering [16]. The application simulates the path followed by a group of photons leaving a source and heating the medium in a non-deterministic way. It achieves the result by computing the non-deterministic path of each photon and combining their effects. Photons do not interact with each other, making the application potentially embarrassingly parallel. Moreover, being a Monte Carlo simulation, the final result is already an approximated one, so losing accuracy to have a faster recovery time is acceptable as long as the loss is not significant.

In our work [17], we analysed the application, integrated it with Legio and assessed the impact of faults on the result accuracy. We injected faults through software signals, simulating the effect of hardware failures. Our first measurements showed that faults affected the result in two ways: they reduced the amount of photons considered by the simulations and introduced a drift in the result. While we expected the first aspect due to the graceful degradation approach, the latter had deeper roots in how the application handled communication. In particular, the application assumes that every process part of the execution produces a result, but this assumption is invalid in the presence of faults since failed processes do not participate in MPI calls. Thus, they behave like they produce null values. When the application tries to mediate the values obtained, those null values move the result towards zero, introducing the drift, as seen in the result in Figure 2a.

To solve this issue, the user must code its application with fault awareness: it must be conscious that processes may not participate due to faults and act accordingly. In this case, the application must count the number of values received by processes, thus discarding null values from the computation. It is possible to implement this change using MPI functions only, making the application fault-aware while not introducing third-library APIs. We followed that path, removing the drift in the result and achieving accurate results, as shown in Figure 2b. The Mean Average Percentage Error (MAPE) achieved by our fault-aware implementation remained below 0.6% even when half of the processes failed.

While these results prove the effectiveness of the Legio fault resilience framework, they also highlight the importance of making the application fault-aware. A simple (yet working) integration between Legio and the application compromises the results too much, so the users must analyse their application for no longer valid assumptions in the presence of fault resilience properties. In addition to these concerns about the accuracy of the results, the failure of the process that collects the data from all the others compromises the production of any result: this process is critical and users must take care of it using the Subsidia extension.

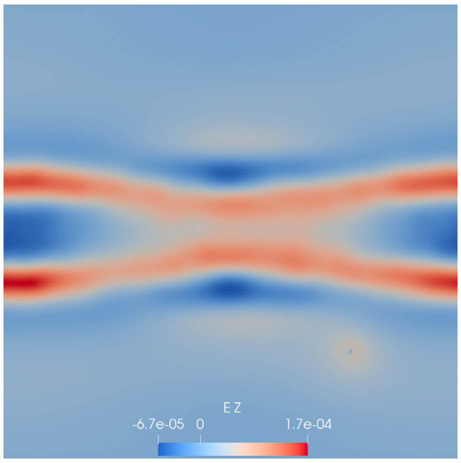
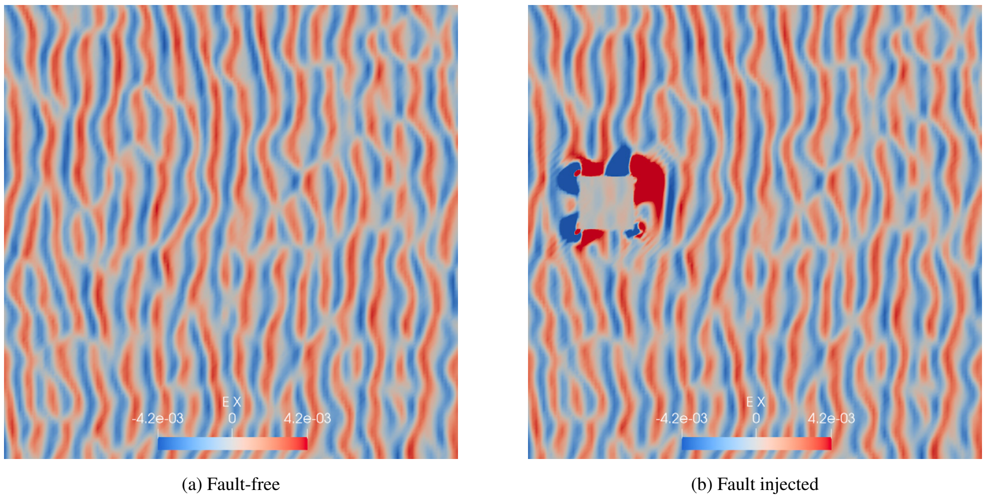


Fig. 3: Result of the iPIC3D application for the GEM problem. We injected the fault outside the area of interest (region in red) 20 iterations before the shown values. We executed this experiment using 32 Karolina IT4I nodes, each with 128 MPI processes. The fault reveals itself as an anomaly in the bottom right but does not interfere with the region of interest.



(a) Fault-free (b) Fault injected

Fig. 4: Result of the iPIC3D application for the Weibel problem. In this case, the whole space is meaningful for the analysis, and the fault affects the execution 50 iterations before the shown values. We executed this experiment using a single Karolina IT4I node, running 64 MPI processes. The fault reveals itself as an anomaly on the left but does not interfere with the values far from it

4.2. *iPIC3D plasma simulation*

The iPIC3D application [15, 11] simulates the plasma behaviour under the effect of an electromagnetic field. It performs the simulation by dividing the space into cells and considering charged probe particles in the space. For each timestep, the application computes the electric field generated by those particles, then the magnetic field, and moves the particles according to the forces they are subject to. Overall, the application follows a stencil pattern: each process manages some cells and communicates the field values at the border to their neighbours. Additionally, processes exchange particles moving in the space, sending them to the appropriate cell.

Graceful degradation in stencil applications is frowned upon, as the communication pattern implies that the fault will affect neighbour processes and eventually compromise the entire execution. In the literature, most efforts rely on C/R functionalities to deal with faults in stencil applications. While graceful degradation is not generally applicable to the stencil scenario, it can still produce partially usable results if the fault occurs in an irrelevant area of the computation or does not have enough iterations to spread. In those cases, the benefits of using graceful degradation may outperform a C/R approach.

Introducing graceful degradation properties in the iPIC3D application through Legio was not straightforward. In particular, the application featured a particle routing algorithm that worked with the assumption that all the processes are present and responsive: that assumption is no longer valid in the presence of faults handled with Legio. We had to change the algorithm to overcome this issue, moving from a greedy approach to a more flexible (but with higher overhead and computational complexity) A* solution. The idea is to use the original algorithm until we detect faults in the network and then switch to A* so that we can limit the impact on performance to the minimum. Additionally, iPIC3D leverages the virtual cartesian topology functionality of MPI, and we had to ensure that faults and the repair procedure preserve details about the topology underneath.

To validate our approach, we executed some experiments on the Karolina IT4I cluster, injecting faults and comparing the results with fault-free executions. Figures 3 and 4 show some of the results of our experiments. In Figure 3, the fault affects a small area of space, which is also outside of the region of interest (the central one with the red bands), so it does not significantly affect the usability of the results. In Figure 4, the fault affects a larger area (we executed with fewer processes, so each of them is responsible for a larger region), and we are interested in the whole space, but the fault effect stays close to it, and the regions far from it remain unchanged, thus valid.

The iPIC3D study shows that even applications that should not benefit from a graceful degradation solution like the one proposed by Legio can gain from its integration. While the analysis needed to integrate our framework was not trivial (we had to identify the problematic algorithm and implement an alternative), using Legio simplified the process, containing code changes to a few files. In the next section, we give users a few hints on how to analyse the application to make it fault-aware, together with some tips for proper Legio usage.

5. Using Legio: tips for the users

The Legio fault resilience framework is open-source and released under a GPL-3.0 license[1]. It does not require external libraries or dependencies, as it leverages only ULFM, which is part of the latest OpenMPI version and possibly the MPI standard's next version. It uses cmake as a building system, which also enables the framework's configuration. In particular, it is possible to specify which Legio functionalities to use and to redefine the behaviour of MPI calls upon fault. Additionally, it is possible to choose the logging level of the framework, affecting the amount of information shown by Legio during application execution.

After configuring, compiling, and installing Legio, the user can link it to MPI applications and leverage graceful degradation properties. As we have shown in the previous section, however, the user should take additional care to ensure the fault-awareness of the application. This consideration requires an analysis of the role of the information moving between processes, considering the possibility that the data movement may not happen. For this reason, the application should initialise receive-buffers so that missing data will maintain the validity of the entire result. Changing the application to include fault-awareness should not remarkably impact application performance, and it should be possible to perform it using only MPI functionalities. Additionally, the user can leverage the API provided by Legio directly, which helps determine the presence of faults within the execution.

Alongside the analysis of data movements, the user should assess the role of processes in creating the result. If a process is mandatory for generating a valid result, then the process is critical, and the user must use the Subsidia extension to handle it correctly. Using the Subsidia extension is slightly intrusive in the code and requires additional consideration on how to recover the state of the failed critical process after its recreation. While an application level C/R may be sufficient, the users should evaluate the possibility of performing other kinds of data recovery, like Algorithm-Based Fault Tolerance (ABFT) [6] or leveraging data redundancy, with the objective of getting the best out of their executions.

While all these considerations and analyses may seem to make it non-trivial to use Legio in a general use case, they are fundamental to extracting the best from the application executions. A more direct integration can already provide some benefits, but the fault-awareness analysis allows for even better accuracy results, as shown in Section 4.1.

[1] https://github.com/Robyroc/Legio

Moreover, Legio hides all the complexity of fault management within its code, leaving the application free of burdensome fault-handling routines. We think using Legio can be a significant step up for users dealing with faults in their executions, as it allows for valid results with minimal changes.

6. Conclusion

In this paper, we overviewed the current status of the Legio fault resilience framework, with all the improvements that enlarged the functionalities it provides. Using Legio, users can introduce graceful degradation properties in their applications, continuing the execution with the non-failed processes. This behaviour affects the results' correctness, but it can be an acceptable tradeoff for a fast fault recovery procedure. Moreover, Legio requires minimal changes to the application code for its integration, making it ideal for inexperienced users.

We showed the use of the Legio framework features in two use cases. The first one covered a Monte Carlo application, which can benefit significantly from a graceful degradation approach. Using Legio, we could continue the execution even when half of the processes failed, obtaining a minimal loss of accuracy after the fault awareness analysis. The second use case analysed a stencil application, which is usually incompatible with the possibility of losing part of the computation. Nonetheless, leveraging Legio allowed the execution to continue past fault detection and produce usable and valid results despite the damage of the fault.

We designed Legio with the HPC user in mind. This commitment led to the definition of the three core principles (efficiency, flexibility, and transparency) we thoroughly followed during our development. This work also contains a set of tips for HPC users so that they can take the best out of the framework. We think using Legio is an effective way to deal with faults in MPI applications and can be a good starting point for a deeper study of fault management.

Acknowledgements

This work was supported by the Italian Ministry of University and Research (MUR) under the PON/REACT project. We also acknowledge EuroHPC Joint Undertaking for awarding us access to Karolina at IT4Innovations, Czech Republic (EHPC-DEV-2023D10-018).

References

[1] Ansel, J., Arya, K., et al., 2009. DMTCP: Transparent checkpointing for cluster computations and the desktop, in: 2009 IEEE International Symposium on Parallel & Distributed Processing, IEEE. pp. 1–12.

[2] Bernholdt, D.E., Boehm, S., Bosilca, G., Gorentla Venkata, M., Grant, R.E., Naughton, T., Pritchard, H.P., Schulz, M., Vallee, G.R., 2020. A survey of MPI usage in the US exascale computing project. Concurrency and Computation: Practice and Experience 32, e4851.

[3] Bland, W., et al., 2013. Post-failure recovery of MPI communication capability: Design and rationale. The International Journal of High Performance Computing Applications 27, 244–254.

[4] Bouteiller, A., Herault, T., et al., 2006. MPICH-V project: A multiprotocol automatic fault-tolerant MPI. The International Journal of High Performance Computing Applications 20, 319–333.

[5] Clarke, L., Glendinning, I., et al., 1994. The MPI message passing interface standard, in: Programming environments for massively parallel distributed systems. Springer, pp. 213–218.

[6] Du, P., et al., 2012. Algorithm-based fault tolerance for dense matrix factorizations. Acm sigplan notices 47, 225–234.

[7] Duarte, E.P., Bona, L.C., Ruoso, V.K., 2014. VCube: A provably scalable distributed diagnosis algorithm, in: 2014 5th Workshop on latest advances in scalable algorithms for large-scale systems, IEEE. pp. 17–22.

[8] Fagg, G.E., Dongarra, J.J., 2000. FT-MPI: Fault tolerant MPI, supporting dynamic applications in a dynamic world, in: European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting, Springer. pp. 346–353.

[9] Gamell, M., et al., 2014. Exploring automatic, online failure recovery for scientific applications at extreme scales, in: Proceedings of SC'14, IEEE. pp. 895–906.

[10] Gamell, M., et al., 2015. Local recovery and failure masking for stencil-based applications at extreme scales, in: Proceedings of SC, IEEE. pp. 1–12.

[11] Gonzalez-Herrero, D., et al., 2018. Performance analysis and implementation details of the Energy Conserving Semi-Implicit Method code (ECsim). Computer Physics Communications 229, 162–169.

[12] Hargrove, P.H., Duell, J.C., 2006. Berkeley lab checkpoint/restart (BLCR) for linux clusters, in: Journal of Physics: Conference Series, p. 494.

[13] Laguna, I., Richards, D.F., et al., 2016. Evaluating and extending user-level fault tolerance in MPI applications. The International Journal of High Performance Computing Applications 30, 305–319.

[14] Losada, N., Cores, I., et al., 2017. Resilient MPI applications using an application-level checkpointing framework and ULFM. The Journal of Supercomputing 73, 100–113.

[15] Markidis, S., Lapenta, G., et al., 2010. Multi-scale simulations of plasma with iPIC3D. Mathematics and Computers in Simulation 80, 1509–1519.

[16] Prahl, S.A., 1989. A Monte Carlo model of light propagation in tissue, in: Dosimetry of laser radiation in medicine and biology, SPIE. pp. 105–114.

[17] Rocco, R., Palermo, G., 2023a. Exploit Approximation to Support Fault Resiliency in MPI-based Applications. 2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume .

[18] Rocco, R., Palermo, G., 2023b. Fault-Aware Group-Collective Communication Creation and Repair in MPI, in: European Conference on Parallel Processing, Springer. pp. 47–61.

[19] Rocco, R., Palermo, G., 2023c. POSTER: The Legio Fault Resilience Framework: Design and Rationale, in: Proceedings of the 20th ACM International Conference on Computing Frontiers.

[20] Rocco, R., Palermo, G., et al., 2023. Fault Awareness in the MPI 4.0 Session Model, in: Proceedings of the 20th ACM International Conference on Computing Frontiers.

[21] Rocco, R., Repetti, L., Boella, E., Gregori, D., Palermo, G., 2024. Extending the legio resilience framework to handle critical process failures in mpi, in: 2024 32nd Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), IEEE. pp. 44–51.

[22] Rocco, R., et al., 2021. Legio: fault resiliency for embarrassingly parallel MPI applications. The Journal of Supercomputing , 1–21.

[23] Shahzad, F., Thies, J., et al., 2018. CRAFT: A library for easier application-level checkpoint/restart and automatic fault tolerance. IEEE Transactions on Parallel and Distributed Systems 30, 501–514.

[24] Suo, G., Lu, Y., et al., 2013. NR-MPI: a Non-stop and Fault Resilient MPI, in: 2013 International Conference on Parallel and Distributed Systems, IEEE. pp. 190–199.

[25] Teranishi, K., Heroux, M.A., 2014. Toward local failure local recovery resilience model using MPI-ULFM, in: Proceedings of the 21st european MPI users' group meeting, pp. 51–56.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 70–81

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Towards a holistic view of the origin of multiple stellar populations in globular clusters

Elena Lacchin[a,b,c,*]

[a] *Physics and Astronomy Department Galileo Galilei, University of Padova, Vicolo dell'Osservatorio 3, I-35122 Padova, Italy*
[b] *INAF, Osservatorio Astronomico di Bologna, Via Gobetti 93/3, I-40129 Bologna, Italy*
[c] *Institut für Theoretische Astrophysik, ZAH, Universität Heidelberg, Albert-Ueberle-Str. 2, D-69120 Heidelberg, Germany*

Abstract

The presence of multiple stellar populations in globular clusters is now well accepted, however, very little is known regarding how they formed. Answering this question is one of the major challenges of stellar astrophysics, yet of great interest, given the importance of globular clusters in different fields, from stellar evolution to gravitational wave emission and galaxy assembly. We present a series of 3D hydrodynamic simulations both of isolated clusters and at a cosmological scale aimed at probing the formation of globular clusters and their multiple stellar populations. We focus first on the effects of both rotation and Type Ia supernovae on the formation of the second population with particular attention on the chemical composition of the newborn stars. Then, we present the first simulations of the SImulating the Environment where Globular clusters Emerged project aimed at probing the formation of globular clusters through sub-parsec cosmological zoom-in simulations.

Keywords: hydrodynamics; globular clusters; stellar feedback; star formation

1. Introduction

Until the beginning of the 21st century, globular clusters (GCs) were thought to be simple systems composed of gravitationally bound stars, sharing similar origins and properties. However, with the continuous development of observational facilities and new tools to analyze the incoming data, a different and way more complex picture arose. Now we know that in general, Galactic GCs have masses between $10^4 - 10^6 M_\odot$, with an average mass of $2 \times 10^5 M_\odot$, and are very compact, with a half mass radius of up to few tens of pc. In most cases, as in the Milky Way, they have ages greater than 10 Gyr, even though some younger ones are found in the Magellanic Clouds and in the Andromeda galaxy. From the chemical point of view, they generally host metal-poor stars but the most peculiar feature, discovered in the last decades, is that stars belonging to the same GC show different chemical compositions meaning that GCs

[*] Corresponding author.
E-mail address: elena.lacchin@inaf.it

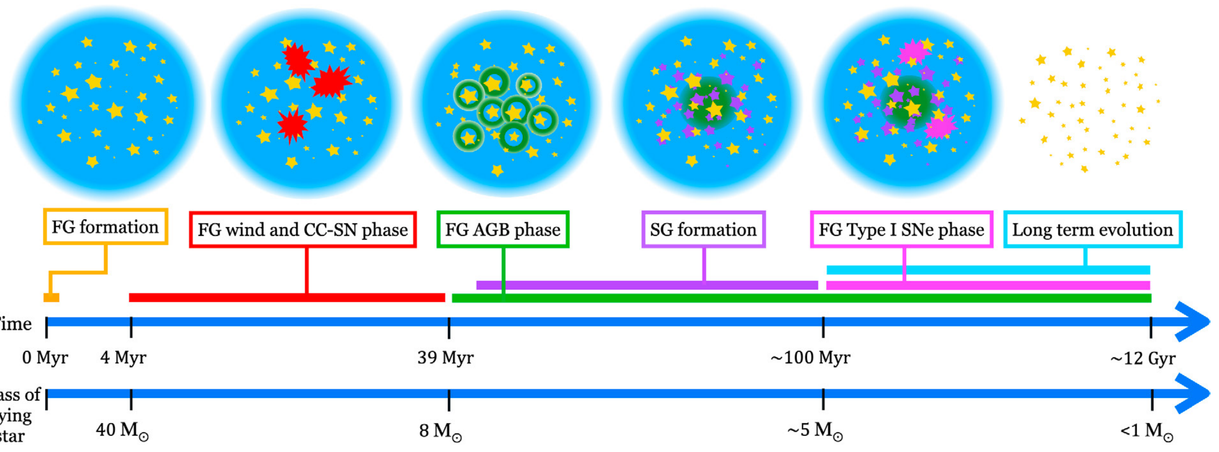


Fig. 1. Schematic visualization of the different phases of the formation of MPs in a GC in the AGB scenario, highlighting the main feedback sources in action as a function of time and associated mass of dying star.

are hosting multiple stellar populations (MPs). In particular, a fraction of their stars shows an “anomalous” chemistry which are labelled second-population stars at variance with first-population stars which show a chemical composition similar to the field stars.

From the observational point of view, in the local Universe, major recent advances have been made mainly thanks to the Hubble Space Telescope and Gaia observations of Galactic GCs. In the last years also James Webb Space Telescope (JWST) is providing us with fundamental follow-ups, not only allowing us to study with unprecedented detail the populations of local GCs, but also to probe the high-redshift Universe. It is there that most of the GCs are assumed to be formed and therefore exploring the very early Universe can provide hints about how GC looked like at their formation. So far several faint systems at redshift $z > 6$, with masses of $\sim 10^6 - 10^7 M_\odot$ and sizes of few tens of parsec have been revealed, compatible with the expectations for GCs precursors.

On the theoretical side, several investigations have studied the formation of GCs and their MPs but a clear picture is still lacking. Understanding the origin of MPs is crucial not only to reach a deeper knowledge of globular clusters but also to put constraints on stellar evolution and nucleosynthesis, to understand how star formation proceeded in dense early environments, to explore the role of stellar dynamics on the formation of exotic stellar objects such as the binary black holes responsible for the emission of gravitational waves and to understand the role of proto-GC in the cosmic reionization and galaxy formation.

Several scenarios were proposed in the last decades to explain the peculiar chemical composition of GCs stars and the other observational constraints [38, 1]. One of the most widespread scenarios is the asymptotic giant branch (AGB) one, where the first and second populations are formed at different times and are therefore called generations. In this scenario, the first generation is formed in the disk of a star-forming high-redshift galaxy. Then, massive FG stars are exploding carving a hole in the disk of the galaxy. After 40 Myr, as shown in Figure 1 stars are starting to undergo the AGB phase, polluting the system with their ejecta. This material, together with pristine gas (i.e. gas with the same chemical composition of the FG), is used to form the second generation (SG) stars [13]. The dilution of the AGB ejecta with pristine gas is fundamental to reproduce the observed chemical trends. As proposed by [14], the star formation of the second generation is supposed to be quenched when Type Ia supernovae are starting to explode, leaving the system gas-free.

So far, very few theoretical investigations have probed, by means of detailed hydrodynamic models, the formation of MPs, although they represent a crucial tool as they allow us to explore the interplay between gas and stars. For this reason, with the resources available for our EuroHPC project (14 million core hours on Discoverer) we performed a comprehensive series of high-resolution 3D hydrodynamic simulations of star-forming proto-GCs, aimed at modelling a realistic physical environment for GC formation. In particular, our focus has been twofold: on isolated GCs testing the effects of various physical processes on the star formation, and on the formation of faint star-forming stellar clumps

Table 1. Simulations parameters.

Parameter	Description	Values
$M_{\rm FG}$	FG stellar mass	10^7M$_\odot$
a	FG Plummer radius	23 pc
$\rho_{\rm pg}$	Density of the pristine gas	$10^{-24,-23}$g cm^{-3}
$v_{\rm pg}$	Pristine gas velocity relative to the cluster	20 km s^{-1}
$Z_{\rm pg}$	Metallicity of the pristine gas	0.001
Y_{pg}	Pristine gas helium mass fraction	0.246
$T_{\rm pg}$	Temperature of the pristine gas	10^4K
$T_{\rm floor}$	Minimum temperature	10^3K
$t_\star$	Star formation time-scale	0.1 Gyr

in cosmological zoom-in simulations at sub-pc resolution. All the simulations were performed using customized versions of the RAMSES [42] hydro code.

Modelling isolated clusters, due to their lower computational demand, permits the exploration of the parameter space which is fundamental to set new constraints and test the feasibility of the assumed scenario of MP formation. In particular, we have tested the effects of cluster rotation and Type Ia supernovae feedback on the second-generation (SG) star formation.

With cosmological zoom-in simulations, we aimed to achieve a comprehensive, ab initio view of the formation of GCs and their MPs. We performed the first simulations within a new project aimed at SImulating the Environment where Globular clusters Emerged (SIEGE) [6]. The simulations are one of the first-ever attempts with a sub-pc resolution, carried on until $z \sim 6.1$ (i.e. for a cosmic time interval of $\sim$ 1 Gyr) and including the feedback of individual stars.

2. Effects of cluster rotation of the star formation

Generally, globular clusters are described as non-rotating systems but observations in the last decade have shown that a large fraction of them are characterized by a slow internal rotation [36, 22, 46]. The rotation that we observe now is most likely the remnant of a higher early rotation [18, 29] which decreased under the effects of two-body relaxation [2, 22, 40] and tidal forces exerted by the host galaxy. It has been shown that internal rotation can strongly affect GC long-term evolution, in particular, reducing the relaxation timescale (or shortening the time of core collapse), and, therefore, accelerating its dynamical evolution and the mass-loss rate [15, 16, 32]. Moreover, rotation affects also the present-day morphology [19], as well as the dynamics of multiple stellar populations [30, 31, 43]. The observational study of the kinematics of multiple stellar populations is still in its early stages, but a few investigations have already revealed differences in the rotation amplitudes between stellar populations [25, 26, 12, 11, 41], with the SG component rotating faster than the FG. Rotation is, therefore, a fundamental physical process that needs to be included in theoretical models, however, very few studies have investigated its effects on the formation of multiple populations in GCs so far.

In this work, we extended the study of [4] to investigate the effects of FG rotation on the formation of SG stars in a massive proto-GC of mass of 10^7M$_\odot$, moving through a uniform pristine gas distribution for which we have assumed two different density values: $\rho_{pg} = 10^{-23}$g cm^{-3} and $\rho_{pg} = 10^{-24}$g cm^{-3}, through a series of 3D hydrodynamic simulations. We have also varied the inclination between the rotational axis and the orbital plane of the GC and the velocity profiles. We explored how the interplay between the kinematics of the AGB ejecta released by a rotating FG system and the accreted pristine gas affect the final kinematic properties of SG stars and their dependence on the chemical composition, for different SG subpopulations. For this reason, we have followed the evolution of the helium mass fraction (Y) which is equal to 0.246 for FG stars while it decreases with time for AGB ejecta starting from $\sim$ 0.36. The simulation parameters have been collected in Table 1.

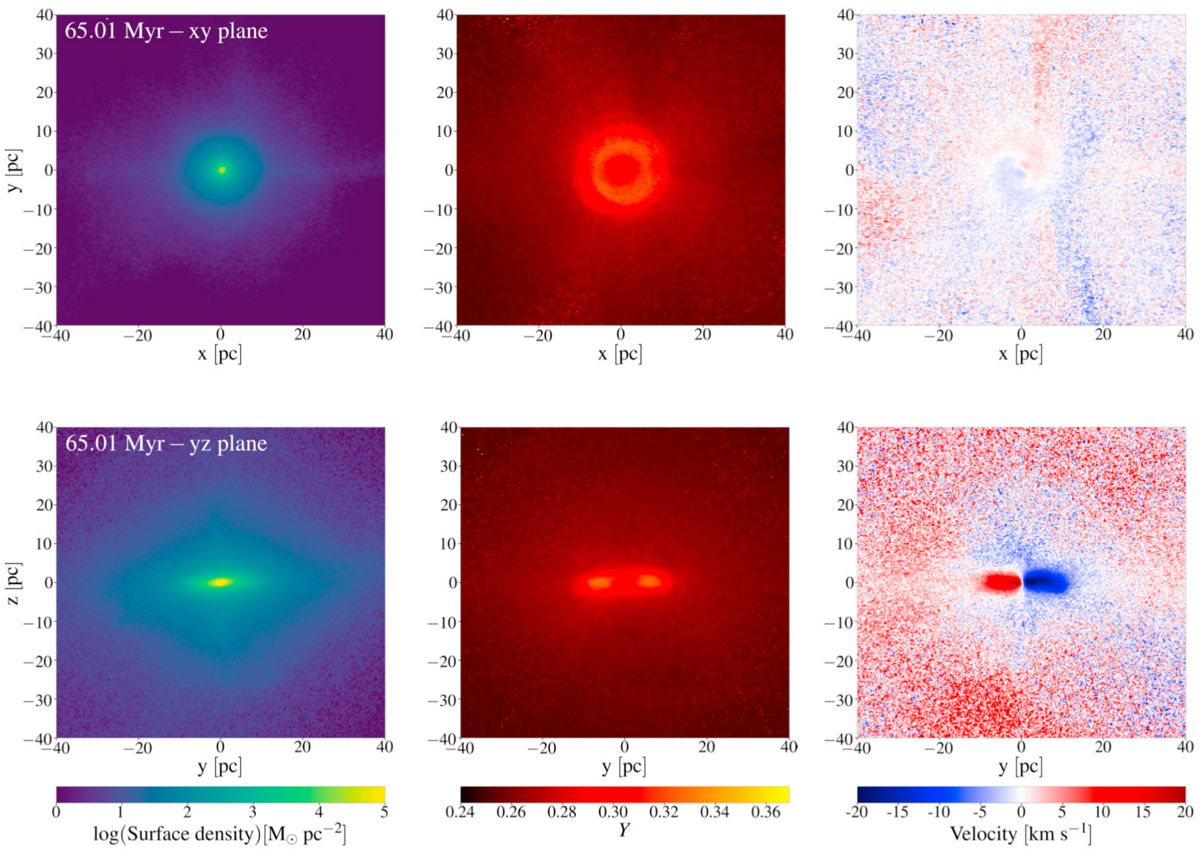


Fig. 2. Two-dimensional maps of the stellar component at 65 Myr for the model with $\rho_{pg} = 10^{-24}$g cm^{-3}, with the rotational profile described in Equation 2 and rotation about the z-axis on the x-y plane (top panels) and y-z plane (bottom panels). The first column shows the surface density, the second represents the helium mass fraction Y, and the third the line-of-sight velocity of the stars. From [24].

The feedback from FG AGB stars is modelled by adding a source term to the Eulerian mass and energy conservation equations assuming that the FG is distributed following the Plummer profile [37]:

$$\rho_\star(r) = \frac{3M_{\rm FG}}{4\pi a^3}\left(1 + \frac{r^2}{a^2}\right)^{-5/2} \tag{1}$$

with mass $M_{\rm FG}$ and Plummer radius a.

To model the cluster's internal rotation, we impart a rotation velocity to the AGB ejecta, following the rotational curve radial profile suggested by [28] :

$$v_{\rm rot} = \frac{2v_{\rm pk}R}{R_{\rm pk}}\left(1 + \left(\frac{R}{R_{\rm pk}}\right)^2\right)^{-1} \tag{2}$$

$R_{\rm pk}$ represents the location of the peak of the profile, while $v_{\rm pk}$ is the value of the rotational amplitude at the peak. Here, we have set $R_{\rm pk} = a$ and $v_{\rm pk} = 2.5$ km s^{-1}. We also performed a run with a solid body rotation obtaining very minor differences. In addition to these physical processes, we are also including star formation and cooling processes whose description can be found in [24].

The initial setup of this work is similar to the one of [4] with all our simulations starting at $t_{AGB} = 39$Myr after the FG formation, once all massive stars have already exploded as core-collapse supernovae and the system is completely cleared out of both SN-enriched ejecta and pristine gas [5]. At this time, the intermediate-mass stars are undergoing

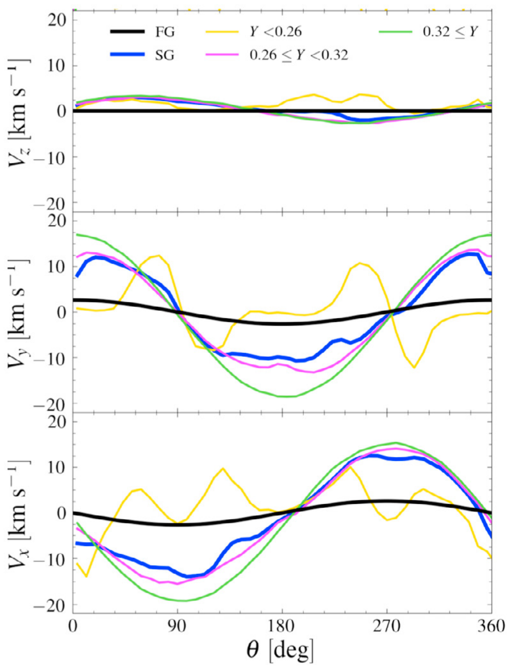


Fig. 3. Rotation amplitude of the Cartesian velocity components for SG stars as a function of θ, defined as the angle between the direction of each star projected on the xy plane and the x-axis, for three bins of the helium mass fraction Y (see the legend for more details) for the model with low density and rotation about the z-axis at 65 Myr. The FG rotation amplitudes are also shown by the black line. From [24]. .

their AGB phase, returning mass and energy in the gas-free system. The system is located at the centre of a cubic computation box which is divided into a uniform grid reaching a resolution of $\sim$ 0.6pc.

Our simulations have revealed the complex hydrodynamics/stellar dynamics of the SG formation phase in the presence of a rotational FG system. As derived in previous investigations, we find that the SG forms concentrated in the innermost regions of the FG cluster. Both the SG morphology and kinematics are significantly affected by rotation and the interaction between rotating AGB ejecta and non-rotation infalling external gas. In addition, we find that the evolution of the system strongly depends on the density of the infalling pristine gas: while for a low density, the rotating ejecta give birth to a stellar disk, mainly composed of extremely helium-enhanced stars as shown in Figure 2, for a pristine gas density one order of magnitude higher, the AGB ejecta are significantly more diluted with non-rotating pristine gas and, therefore, new stars are born with small-to-no memory of the FG rotational pattern. Focusing on the low-density model, as shown in Figure 3, the SG is rotating faster than the FG as a result of its higher concentration. In addition, we derived that the He-rich SG subgroup, which has formed earlier and is poorly diluted with pristine gas, rotates more rapidly than the He-poor subgroup, formed out of more diluted ejecta. The findings of our simulations are generally consistent with those of the first observational studies that have explored the kinematics of different SG subgroups and found that the more helium-enhanced SG stars rotate faster than the helium-poor ones [10, 21]. Changing the inclination and the velocity profiles leads to very minor differences.

3. Type Ia Supernova feedback in globular clusters

One of the still open questions regarding the AGB scenario deals with how the SG formation was quenched. However, this issue concerns more in general all the scenarios for the formation of multiple stellar populations since, even though most of them are not assuming any pollution from AGB stars, low and intermediate-mass stars will release enriched gas. To prevent AGB pollution, this gas must not be recycled to form new stars and therefore star formation has to be already quenched by some process. In the AGB framework, [14] ran simulations including Type

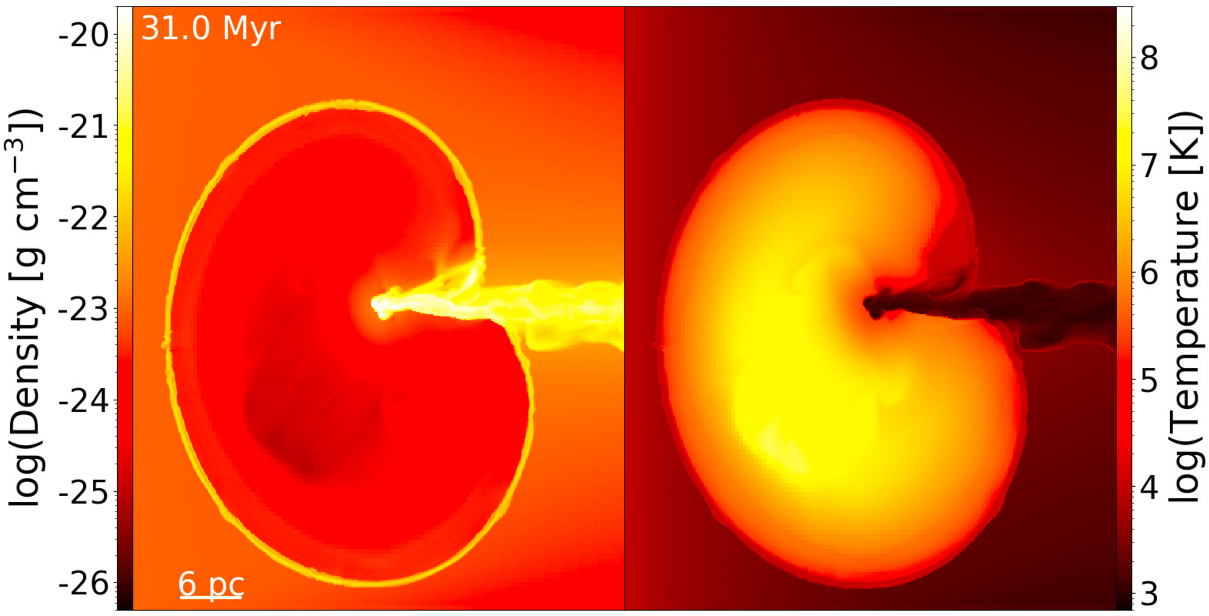


Fig. 4. Two-dimensional maps of the stellar component at 31 Myr for the high-density simulation with a mass of 10^6M$_\odot$ on the x-y plane. The first column shows the density, the second represents the temperature. A SN is just exploded and it is expanding all over the cluster. A dense and cold shell is formed which separates the still unperturbed gas from the swept-up one. On the right, cold and dense gas is being accreted by the cluster mainly coming from the infalling event.

Ia SNe feedback to test whether these stars could halt the star formation in a GC. They performed 1D hydrodynamic simulations comparing the results with and without SNe Ia feedback. They found that assuming a constant SN rate, the AGB ejecta are rapidly wiped out from the system almost immediately after the first SN Ia explosions, resulting in a sudden halt of the star formation. However, they located all SNe Ia at the centre of the cluster, a very strong assumption that could lead to significant overestimations of the effects of feedback, in particular in 1D simulations. On the other hand, from the chemical point of view, the bulk of GCs are characterized by a narrow internal iron dispersion of $\sigma_{[Fe/H]} < 0.1$dex [9], in particular among SG stars [27], suggesting that Type Ia SNe belonging to the FG, which are significant Fe producers, should not provide a significant contribution to the gas out of which SG stars are formed. This would be consistent with the results of [14] where SNe Ia are halting the SG formation very quickly. There are however about 20% of the Galactic GC (referred to as Type II clusters in [34]; see also [20] and Section 2.2.1) that are instead characterized by a significant dispersion in Fe; the origin of this spread and its link with the SG formation is still unknown. In [23] we have studied whether Type Ia SN feedback could prevent the star formation in a cluster of 10^7M$_\odot$, finding that the star formation was not quenched as expected by [14]. We have then followed up this work extending it to clusters of lower masses to check if the change in the cluster mass affects how SNe Ia regulate star formation. We therefore perform a series of 3D hydrodynamic simulations to explore the effects of the feedback of SN Ia explosions on the duration of the SG star formation phase and on the chemical properties of the SG stars. In particular, we study the effects of SNe Ia on the iron composition to understand how much of their ejecta is recycled to form new stars.

The initial configuration of the simulations is similar to the one described in the previous section where instead of rotation we model the feedback from Type Ia SNe (see [23]). At variance with the previous simulations, we have varied the mass of the FG, for which we have tested a case with $M_{FG} = 10^5$M$_\odot$ and a higher mass one with $M_{FG} = 10^6$M$_\odot$. The system is located as before at the centre of the computational box however we have here exploited the adaptive mesh refinement technique to reach higher resolution where SNe are exploding for a maximum resolution of 0.1 pc. Type I SNe are originated from the thermonuclear explosion of white dwarfs in binary systems. When a Type Ia explodes, we assume that one Chandrasekar mass (1.44M$_\odot$) is released into the interstellar medium (ISM) with an amount of iron of 0.5M$_\odot$[39] and a metallicity equal to 1. Each SN will also release 10^{51}erg of thermal energy in

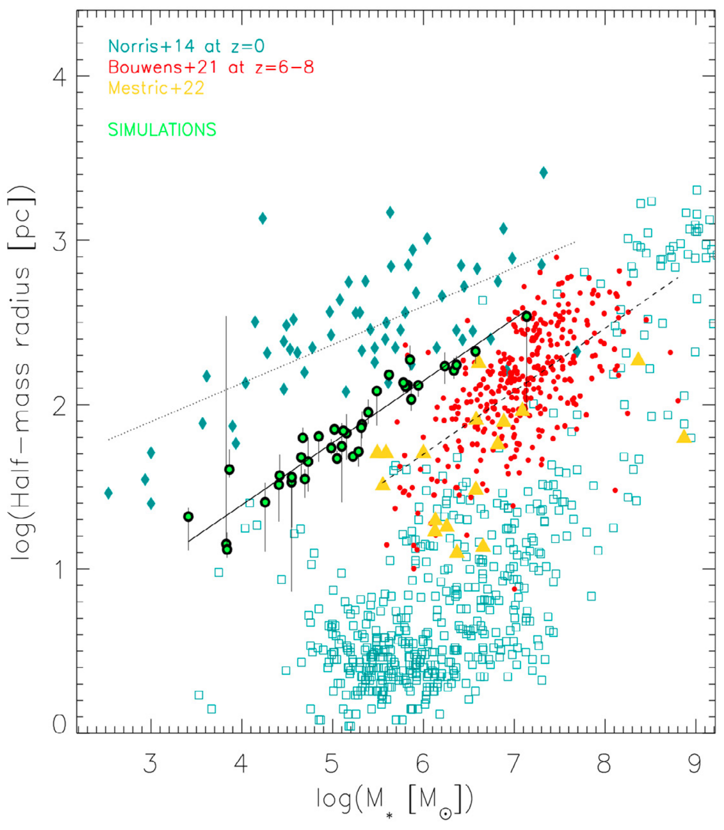


Fig. 5. Size as a function of stellar mass for the stellar clumps in [6] cosmological simulation (solid green circles) compared to observational data sets from the literature. We computed the size of each clump from the half-mass radius of its stellar surface density profile, considering three different viewing angles, i.e. along the x-, y-, and z-axis and taking the median values of the three projections. The error bar is computed as the distance between the minimum and the maximum size. The other symbols are observational values from a sample of lensed faint galaxies with photometric redshifts in the range $6 \leq z \leq 8$ [3], red solid circles), a set of spectroscopically confirmed lensed clumps at $z > 6$ ([33] yellow triangles), and from an extended local sample which includes globular clusters, spheroids, dwarf ellipticals, ultra-compact dwarfs (dark cyan solid diamonds), and dwarf spheroidals ([35], dark cyan open squares). From [6].

the ISM. To every Type Ia SN progenitor, we have associated an explosion time assuming the delay time distribution of [17] for the single degenerate scenario, due to its higher rate at short times. In the first 0.1 Gyr, the timespan we are interested in, $\sim$ 100 SNe explosions in $10^6 M_\odot$ cluster and 10 SNe explosions in a $10^5 M_\odot$ one would take place. Spatially, SNe have been distributed following the Plummer profile computed for a cluster mass of $10^{5-6} M_\odot$, and a Plummer radius of a = 3 pc.

In Figure 4 we show a snapshot of the high-density simulations right after the formation of one Type Ia supernova explosion. We find that the explosions of Type Ia SNe are mildly affecting the star formation rate both in the low-and high-density models at $M_{FG} = 10^6 M_\odot$ while a much stronger effect is seen for $M_{FG} = 10^5 M_\odot$. What is instead affected are the chemical compositions of the newborn SG stars, both in the helium and iron content. In the low-density model, the accretion of pristine ISM gas is prevented during a significant part of the SG formation, which leads to a very high helium concentration, not observed in present-day GCs. On the other hand, with a higher ISM density, the explosions do not prevent the accretion of external gas and the newborn stars are formed from more diluted material. The helium composition is therefore more in line with the observations. These results allow us to place constraints on the possible contribution of Type Ia supernovae to the formation of GCs, which is crucial for interpreting the iron spreads that have recently been detected in some of these objects.

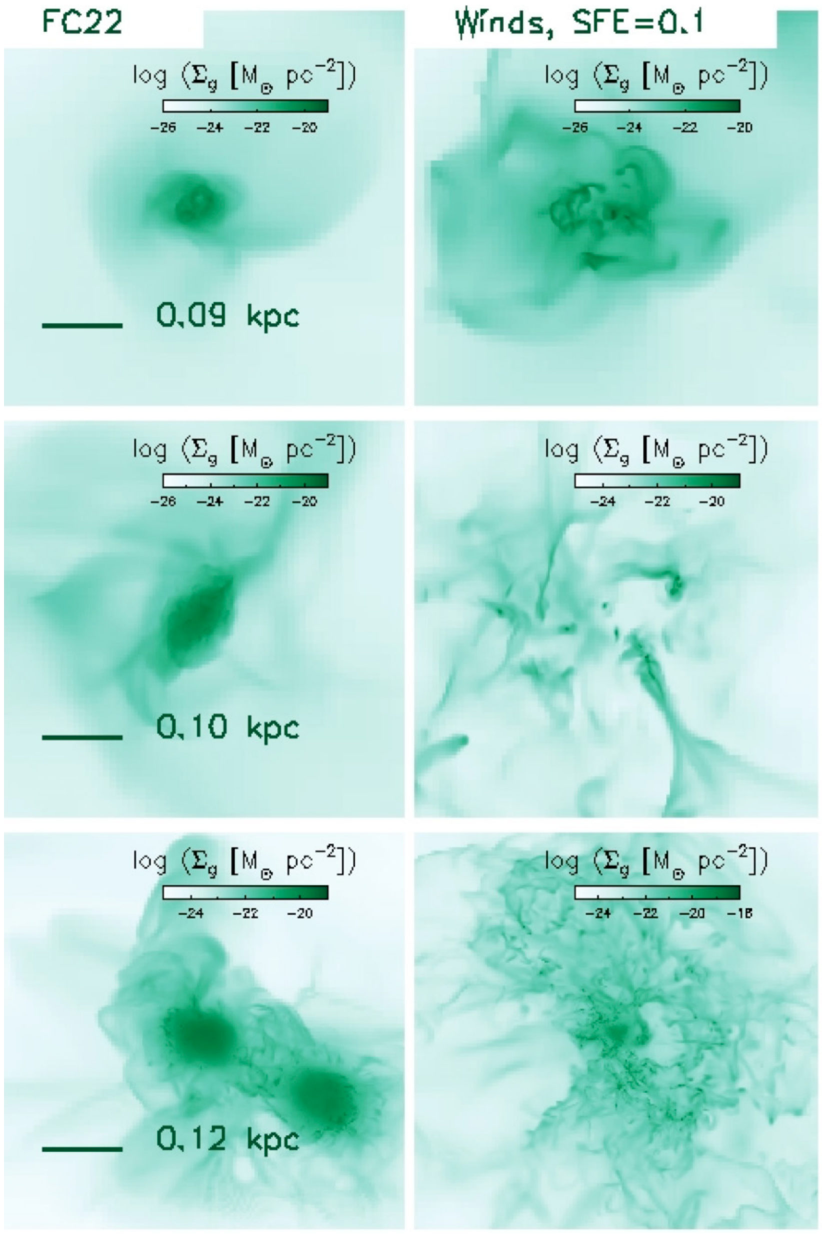


Fig. 6. Evolution of the surface density of the gas, Σ_g, in the highest resolution simulation of [6],on the left, and in the new set of simulations we have performed, specifically with new winds recipe and star formation efficiency equal to 0.1, on the right. The maps show the central, zoomed-in region of the box, computed at three different redshifts (from top-to bottom: z = 15, z = 13, z = 10). The horizontal green solid lines shown in the left-hand column panels indicate the physical scale.

4. Cosmological zoom-in simulations of star-forming clumps

Observations are now revealing smaller and smaller stellar clumps at increasingly larger distances from us which could be the progenitors of what we classify now as globular clusters. This wealth of data is significantly increasing with the JWST and will be extended even more once Extremely Large Telescope starts operating. It is therefore fundamental to interpret the current data to then guide future observations. For this reason, it is crucial to build a solid understanding of these faint sources and the environment in which they are forming and evolving. Cosmological zoom-in hydrodynamic simulations are the perfect tool to achieve this aim as they allow us to focus at a very high resolution on the region of interest without losing information of the large-scale structure, which is followed at a lower resolution.

As stellar clumps have sizes of ∼ 100pc, the typical resolutions adopted in these simulations of ∼ 10pc are too large to probe the internal structures of these systems. We are therefore forced to increase the resolution down to the sub-parsec scale to properly model the turbulent star-forming gas. Reaching sub-parsec resolutions implies that the masses available for star formation in a cell are comparable with the ones of single stars and therefore creating particles representing an entire stellar population, as it is done at lower resolution, is no more a good approximation. In this case, we need to properly model individual stars, from their formation where we implement a stochastic initial mass function-sampling to their feedback.

Our simulations are aimed at modelling a system with features similar to an extended star-forming complex at redshift $z = 6.14$ strongly magnified by the galaxy cluster MACS J0416.1–2403, which includes several clumps [44, 7] . The main components of the complex are D1 and T1: D1 has a stellar mass of $2.2 \times 10^7 M_\odot$ and size of 44 pc [44] while T1 is one of the faintest spectroscopically confirmed starforming objects ever identified at high redshift with an estimated stellar mass of $2 \times 10^6 M_\odot$ and a size of < 30 pc [45].

We first performed a simulation, where we included the feedback from both massive and intermediate-mass stars of a dark matter halo that is assumed to form a stellar component of a mass comparable with the one observed in the D1+T1 system. We found that the total mass formed inside the dark matter halo is $3.4 \times 10^7 M_\odot$ in agreement with the observed stellar mass of the D1 + T1 system. However, the most massive clumps formed have masses of $10^6 M_\odot$ and half-mass sizes of 100 pc. These sizes are larger than the observed ones and imply an average density one order of magnitude lower than those observed (see Figure 5). In addition, also the star formation rate reached in the simulation of about $0.1 M_\odot$ yr^{-1} is much lower than the $15 M_\odot$ yr^{-1} derived from observations. The observed value itself, on the other hand, deviates significantly from the tight SFR–L_{CII} relation observed in high-redshift systems [8], if considering the $L_{CII} = 2.9 \times 10^6 L_\odot$ derived by [6]. The one predicted from our simulation instead, is much more consistent with the observations of other high-redshift systems.

To understand why our stellar clumps are so diffuse we performed another series of simulations changing the star formation efficiency, the initial mass function and the feedback model. In general, the new feedback model which includes a more realistic treatment of stellar winds leads to smaller and much denser clumps with respect to the ones obtained by [6], as shown in Figure 6, and comparable with present-day young stellar clusters.

5. Conclusion

The EuroHPC project we were awarded of on the Discoverer supercomputer has been devoted to the study of the formation and evolution of globular clusters and their multiple stellar populations which have been widely probed but still poorly understood. We exploited a twofold approach focusing, on one side, on the modelization of isolated clusters and testing the effect of different physical processes on their star formation, on the other side, on cosmological simulations, to study the interplay between the stellar clusters and the surrounding environment. For both our investigations we performed detailed 3D hydrodynamic simulations.

Firstly, we have been able to study the behaviour of rotating clusters, particularly the link between the stellar dynamics and the chemical composition of second-generation stars finding good agreement between the results we have obtained and the observational trends.

In addition, we have explored how the second population formation is shaped by Type Ia supernova explosions in clusters of lower mass with respect to [23]. Such investigation has revealed that these explosions cannot quench the star formation as previously found and that they are limiting the dilution of the Asymptotic Giant Branch stars ejecta, leading to an unobserved helium-rich second generation. Therefore, it is likely that Type Ia SNe may have played a major role neither in the formation nor in the quenching of star formation in globular clusters.

Finally, we focused on the formation of stellar clumps at high-redshift to reproduce the D1+T1 system. We successfully obtained clumps with similar masses, although the simulated clump sizes are too large in comparison with the observed ones. In a follow-up series of simulations, we have improved our feedback recipe obtaining much more concentrated clumps significantly reducing the gap with the observations.

Nomenclature

GCs	Globular Clusters
MPs	Multiple Stellar Populations
JWST	James Webb Space Telescope
AGB	Asympthotic Giant Branch
SG	Second generation
FG	First generation
SN	Supernova
ISM	Interstellar medium
pc	parsec
$M_\odot$	Mass of the Sun
$L_\odot$	Luminosity of the Sun
z	Redshift
Y	Helium mass fraction

Acknowledgements

We acknowledge EuroHPC Joint Undertaking for awarding us access to Discoverer at Sofia Tech Park, Bulgaria.

References

[1] Bastian, N., Lardo, C., 2018. Multiple Stellar Populations in Globular Clusters. ARA&A 56, 83–136. doi:10.1146/annurev-astro-081817-051839, arXiv:1712.01286.

[2] Bianchini, P., van der Marel, R.P., del Pino, A., Watkins, L.L., Bellini, A., Fardal, M.A., Libralato, M., Sills, A., 2018. The internal rotation of globular clusters revealed by Gaia DR2. MNRAS 481, 2125–2139. doi:10.1093/mnras/sty2365, arXiv:1806.02580.

[3] Bouwens, R.J., Oesch, P.A., Stefanon, M., Illingworth, G., Labbé, I., Reddy, N., Atek, H., Montes, M., Naidu, R., Nanayakkara, T., Nelson, E., Wilkins, S., 2021. New Determinations of the UV Luminosity Functions from z 9 to 2 Show a Remarkable Consistency with Halo Growth and a Constant Star Formation Efficiency. AJ 162, 47. doi:10.3847/1538-3881/abf83e, arXiv:2102.07775.

[4] Calura, F., D'Ercole, A., Vesperini, E., Vanzella, E., Sollima, A., 2019. Formation of second-generation stars in globular clusters. MNRAS 489, 3269–3284. doi:10.1093/mnras/stz2055, arXiv:1906.09137.

[5] Calura, F., Few, C.G., Romano, D., D'Ercole, A., 2015. Feedback from Massive Stars and Gas Expulsion from Proto-Globular Clusters. ApJ 814, L14. doi:10.1088/2041-8205/814/1/L14, arXiv:1511.03277.

[6] Calura, F., Lupi, A., Rosdahl, J., Vanzella, E., Meneghetti, M., Rosati, P., Vesperini, E., Lacchin, E., Pascale, R., Gilli, R., 2022. Sub-parsec resolution cosmological simulations of star-forming clumps at high redshift with feedback of individual stars. MNRAS 516, 5914–5934. doi:10.1093/mnras/stac2387, arXiv:2206.13538.

[7] Calura, F., Vanzella, E., Carniani, S., Gilli, R., Rosati, P., Meneghetti, M., Paladino, R., Decarli, R., Brusa, M., Lupi, A., D'Amato, Q., Bergamini, P., Caminha, G.B., 2021. Constraints on the [C II] luminosity of a proto-globular cluster at z ∼ 6 obtained with ALMA. MNRAS 500, 3083–3094. doi:10.1093/mnras/staa3185, arXiv:2010.07302.

[8] Carniani, S., Maiolino, R., Amorin, R., Pentericci, L., Pallottini, A., Ferrara, A., Willott, C.J., Smit, R., Matthee, J., Sobral, D., Santini, P., Castellano, M., De Barros, S., Fontana, A., Grazian, A., Guaita, L., 2018. Kiloparsec-scale gaseous clumps and star formation at z = 5-7. MNRAS 478, 1170–1184. doi:10.1093/mnras/sty1088, arXiv:1712.03985.

[9] Carretta, E., Bragaglia, A., Gratton, R.G., Lucatello, S., Catanzaro, G., Leone, F., Bellazzini, M., Claudi, R., D'Orazi, V., Momany, Y., Ortolani, S., Pancino, E., Piotto, G., Recio-Blanco, A., Sabbi, E., 2009. Na-O anticorrelation and HB. VII. The chemical composition of first and second-generation stars in 15 globular clusters from GIRAFFE spectra. A&A 505, 117–138. doi:10.1051/0004-6361/200912096, arXiv:0909.2938.

[10] Cordero, M.J., Hénault-Brunet, V., Pilachowski, C.A., Balbinot, E., Johnson, C.I., Varri, A.L., 2017. Differences in the rotational properties of multiple stellar populations in M13: a faster rotation for the 'extreme' chemical subpopulation. MNRAS 465, 3515–3535. doi:10.1093/mnras/stw2812, arXiv:1610.09374.

[11] Cordoni, G., Milone, A.P., Mastrobuono-Battisti, A., Marino, A.F., Lagioia, E.P., Tailo, M., Baumgardt, H., Hilker, M., 2020. Three-component Kinematics of Multiple Stellar Populations in Globular Clusters with Gaia and VLT. ApJ 889, 18. doi:10.3847/1538-4357/ab5aee, arXiv:1905.09908.

[12] Dalessandro, E., Raso, S., Kamann, S., Bellazzini, M., Vesperini, E., Bellini, A., Beccari, G., 2021. 3D core kinematics of NGC 6362: central rotation in a dynamically evolved globular cluster. MNRAS 506, 813–823. doi:10.1093/mnras/stab1257, arXiv:2105.02246.

[13] D'Ercole, A., D'Antona, F., Vesperini, E., 2016. Accretion of pristine gas and dilution during the formation of multiple-population globular clusters. MNRAS 461, 4088–4098. doi:10.1093/mnras/stw1583, arXiv:1607.00951.

[14] D'Ercole, A., Vesperini, E., D'Antona, F., McMillan, S.L.W., Recchi, S., 2008. Formation and dynamical evolution of multiple stellar generations in globular clusters. MNRAS 391, 825–843. doi:10.1111/j.1365-2966.2008.13915.x, arXiv:0809.1438.

[15] Einsel, C., Spurzem, R., 1999. Dynamical evolution of rotating stellar systems - I. Pre-collapse, equal-mass system. MNRAS 302, 81–95. doi:10.1046/j.1365-8711.1999.02083.x.

[16] Ernst, A., Glaschke, P., Fiestas, J., Just, A., Spurzem, R., 2007. N-body models of rotating globular clusters. MNRAS 377, 465–479. doi:10.1111/j.1365-2966.2007.11602.x, arXiv:astro-ph/0702206.

[17] Greggio, L., 2005. The rates of type Ia supernovae. I. Analytical formulations. A&A 441, 1055–1078. doi:10.1051/0004-6361:20052926, arXiv:astro-ph/0504376.

[18] Hénault-Brunet, V., Gieles, M., Evans, C.J., Sana, H., Bastian, N., Maíz Apellániz, J., Taylor, W.D., Markova, N., Bressert, E., de Koter, A., van Loon, J.T., 2012. The VLT-FLAMES Tarantula Survey. VI. Evidence for rotation of the young massive cluster R136. A&A 545, L1. doi:10.1051/0004-6361/201219472, arXiv:1207.7071.

[19] Hong, J., Kim, E., Lee, H.M., Spurzem, R., 2013. Comparative study between N-body and Fokker-Planck simulations for rotating star clusters - II. Two-component models. MNRAS 430, 2960–2972. doi:10.1093/mnras/stt099, arXiv:1211.6527.

[20] Johnson, C.I., Rich, R.M., Pilachowski, C.A., Caldwell, N., Mateo, M., Bailey, John I., I., Crane, J.D., 2015. A Spectroscopic Analysis of the Galactic Globular Cluster NGC 6273 (M19). AJ 150, 63. doi:10.1088/0004-6256/150/2/63, arXiv:1507.00756.

[21] Kamann, S., Dalessandro, E., Bastian, N., Brinchmann, J., den Brok, M., Dreizler, S., Giesers, B., Göttgens, F., Husser, T.O., Krajnović, D., van de Ven, G., Watkins, L.L., Wisotzki, L., 2020. The peculiar kinematics of the multiple populations in the globular cluster Messier 80 (NGC 6093). MNRAS 492, 966–977. doi:10.1093/mnras/stz3506, arXiv:1912.06158.

[22] Kamann, S., Husser, T.O., Dreizler, S., Emsellem, E., Weilbacher, P.M., Martens, S., Bacon, R., den Brok, M., Giesers, B., Krajnović, D., Roth, M.M., Wendt, M., Wisotzki, L., 2018. A stellar census in globular clusters with MUSE: The contribution of rotation to cluster dynamics studied with 200 000 stars. MNRAS 473, 5591–5616. doi:10.1093/mnras/stx2719, arXiv:1710.07257.

[23] Lacchin, E., Calura, F., Vesperini, E., 2021. On the role of Type Ia supernovae in the second-generation star formation in globular clusters. MNRAS 506, 5951–5968. doi:10.1093/mnras/stab2061, arXiv:2107.07962.

[24] Lacchin, E., Calura, F., Vesperini, E., Mastrobuono-Battisti, A., 2022. The role of rotation on the formation of second generation stars in globular clusters. MNRAS 517, 1171–1188. doi:10.1093/mnras/stac2328, arXiv:2209.05178.

[25] Lee, J.W., 2015. Multiple Stellar Populations of Globular Clusters from Homogeneous Ca by Photometry. I. M22 (NGC 6656). ApJS 219, 7. doi:10.1088/0067-0049/219/1/7, arXiv:1506.00116.

[26] Lee, J.W., 2017. Multiple Stellar Populations of Globular Clusters from Homogeneous Ca-CN Photometry. II. M5 (NGC 5904) and a New Filter System. ApJ 844, 77. doi:10.3847/1538-4357/aa7b8c, arXiv:1706.07969.

[27] Legnardi, M.V., Milone, A.P., Armillotta, L., Marino, A.F., Cordoni, G., Renzini, A., Vesperini, E., D'Antona, F., McKenzie, M., Yong, D., Dondoglio, E., Lagioia, E.P., Carlos, M., Tailo, M., Jang, S., Mohandasan, A., 2022. Constraining the original composition of the gas forming first-generation stars in globular clusters. MNRAS 513, 735–751. doi:10.1093/mnras/stac734, arXiv:2203.07571.

[28] Lynden-Bell, D., 1967. Statistical mechanics of violent relaxation in stellar systems. MNRAS 136, 101. doi:10.1093/mnras/136.1.101.

[29] Mapelli, M., 2017. Rotation in young massive star clusters. MNRAS 467, 3255–3267. doi:10.1093/mnras/stx304, arXiv:1702.00415.

[30] Mastrobuono-Battisti, A., Perets, H.B., 2013. Evolution of Second-generation Stars in Stellar Disks of Globular and Nuclear Clusters: ω Centauri as a Test Case. ApJ 779, 85. doi:10.1088/0004-637X/779/1/85, arXiv:1304.6086.

[31] Mastrobuono-Battisti, A., Perets, H.B., 2016. Second-generation Stellar Disks in Dense Star Clusters and Cluster Ellipticities. ApJ 823, 61. doi:10.3847/0004-637X/823/1/61, arXiv:1506.08198.

[32] Mastrobuono-Battisti, A., Perets, H.B., 2021. Linking globular cluster structural parameters and their evolution: multiple stellar populations. MNRAS 505, 2548–2560. doi:10.1093/mnras/stab1407, arXiv:2011.12292.

[33] Meštrić, U., Vanzella, E., Zanella, A., Castellano, M., Calura, F., Rosati, P., Bergamini, P., Mercurio, A., Meneghetti, M., Grillo, C., Caminha, G.B., Nonino, M., Merlin, E., Cupani, G., Sani, E., 2022. Exploring the physical properties of lensed star-forming clumps at $2 \lesssim z \lesssim 6$. MNRAS 516, 3532–3555. doi:10.1093/mnras/stac2309, arXiv:2202.09377.

[34] Milone, A.P., Piotto, G., Renzini, A., Marino, A.F., Bedin, L.R., Vesperini, E., D'Antona, F., Nardiello, D., Anderson, J., King, I.R., Yong, D., Bellini, A., Aparicio, A., Barbuy, B., Brown, T.M., Cassisi, S., Ortolani, S., Salaris, M., Sarajedini, A., van der Marel, R.P., 2017. The Hubble Space Telescope UV Legacy Survey of Galactic globular clusters - IX. The Atlas of multiple stellar populations. MNRAS 464, 3636–3656. doi:10.1093/mnras/stw2531, arXiv:1610.00451.

[35] Norris, M.A., Kannappan, S.J., Forbes, D.A., Romanowsky, A.J., Brodie, J.P., Faifer, F.R., Huxor, A., Maraston, C., Moffett, A.J., Penny, S.J., Pota, V., Smith-Castelli, A., Strader, J., Bradley, D., Eckert, K.D., Fohring, D., McBride, J., Stark, D.V., Vaduvescu, O., 2014. The AIMSS Project - I. Bridging the star cluster-galaxy divide†‡§¶. MNRAS 443, 1151–1172. doi:10.1093/mnras/stu1186, arXiv:1406.6065.

[36] Pancino, E., Galfo, A., Ferraro, F.R., Bellazzini, M., 2007. The Rotation of Subpopulations in ω Centauri. ApJ 661, L155–L158. doi:10.1086/518959, arXiv:0704.2962.

[37] Plummer, H.C., 1911. On the problem of distribution in globular star clusters. MNRAS 71, 460–470. doi:10.1093/mnras/71.5.460.

[38] Renzini, A., D'Antona, F., Cassisi, S., King, I.R., Milone, A.P., Ventura, P., Anderson, J., Bedin, L.R., Bellini, A., Brown, T.M., Piotto, G., van der Marel, R.P., Barbuy, B., Dalessandro, E., Hidalgo, S., Marino, A.F., Ortolani, S., Salaris, M., Sarajedini, A., 2015. The Hubble Space Telescope UV Legacy Survey of Galactic Globular Clusters - V. Constraints on formation scenarios. MNRAS 454, 4197–4207. doi:10.1093/mnras/stv2268, arXiv:1510.01468.

[39] Scalzo, R.A., Ruiter, A.J., Sim, S.A., 2014. The ejected mass distribution of Type Ia supernovae: a significant rate of non-Chandrasekhar-mass progenitors. MNRAS 445, 2535–2544. doi:10.1093/mnras/stu1808, arXiv:1408.6601.

[40] Sollima, A., Baumgardt, H., Hilker, M., 2019. The eye of Gaia on globular clusters kinematics: internal rotation. MNRAS 485, 1460–1476.

doi:10.1093/mnras/stz505, arXiv:1902.05895.

[41] Szigeti, L., Mészáros, S., Szabó, G.M., Fernández-Trincado, J.G., Lane, R.R., Cohen, R.E., 2021. The rotation of selected globular clusters and the differential rotation of M3 in multiple populations from the SDSS-IV APOGEE-2 survey. MNRAS 504, 1144–1151. doi:10.1093/mnras/stab1007, arXiv:2104.04524.

[42] Teyssier, R., 2002. Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES. A&A 385, 337–364. doi:10.1051/0004-6361:20011817, arXiv:astro-ph/0111367.

[43] Tiongco, M.A., Vesperini, E., Varri, A.L., 2019. Kinematical evolution of multiple stellar populations in star clusters. MNRAS 487, 5535–5548. doi:10.1093/mnras/stz1595, arXiv:1907.05901.

[44] Vanzella, E., Calura, F., Meneghetti, M., Castellano, M., Caminha, G.B., Mercurio, A., Cupani, G., Rosati, P., Grillo, C., Gilli, R., Mignoli, M., Fiorentino, G., Arcidiacono, C., Lombini, M., Cortecchia, F., 2019. Massive star cluster formation under the microscope at z = 6. MNRAS 483, 3618–3635. doi:10.1093/mnras/sty3311, arXiv:1809.02617.

[45] Vanzella, E., Calura, F., Meneghetti, M., Mercurio, A., Castellano, M., Caminha, G.B., Balestra, I., Rosati, P., Tozzi, P., De Barros, S., Grazian, A., D'Ercole, A., Ciotti, L., Caputi, K., Grillo, C., Merlin, E., Pentericci, L., Fontana, A., Cristiani, S., Coe, D., 2017. Paving the way for the JWST: witnessing globular cluster formation at z ¿ 3. MNRAS 467, 4304–4321. doi:10.1093/mnras/stx351, arXiv:1612.01526.

[46] Vasiliev, E., Baumgardt, H., 2021. Gaia EDR3 view on galactic globular clusters. MNRAS 505, 5978–6002. doi:10.1093/mnras/stab1475, arXiv:2102.09568.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 82–90

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Performance and Scaling of PION for Modelling Colliding-Wind Binary Systems

Jonathan Mackey[a,*], Thomas A. K. Jones[a,b], Robert Brose[c,a], Luca Grassitelli[d], Brian Reville[e], Arun Mathew[a]

[a]*Dublin Institute for Advanced Studies, Astronomy & Astrophysics Section, DIAS Dunsink Observatory, Dunsink Lane, Dublin, D15 XR2R, Ireland*
[b]*School of Physics, Trinity College Dublin, The University of Dublin, Dublin 2, Ireland*
[c]*School of Physical Sciences and Centre for Astrophysics & Relativity, Dublin City University, Glasnevin, D09 W6Y4, Ireland.*
[d]*Argelander-Institut für Astronomie, Universität Bonn, Auf dem Hügel 71, D-53121 Bonn, Germany*
[e]*Max-Planck-Institut für Kernphysik, Saupfercheckweg 1, 69117 Heidelberg, Germany*

Abstract

Colliding-wind binaries are star systems with strong wind-wind interaction producing X-ray-, radio- and sometimes gamma-ray-emitting plasma. Computer modelling of these systems including magnetic fields and all relevant radiative processes is still in its infancy, but is important for constraining shock physics (including particle acceleration) and interstellar dust formation processes. Here we summarise some recent scientific results on these systems and present results of *EuroHPC* benchmarking tests of the astropysical fluid-dynamics code PION on the Czech supercomputer *Karolina*. Excellent strong and weak scaling are demonstrated, showing that PION can be used efficiently on Tier-0 systems for 3D simulations of colliding winds in binary systems. Further improvements were obtained in hybrid OpenMP/MPI parallelisation on the basis of the benchmarking test results.

Keywords: Computational Fluid Dynamics ; Astrophysics ; Stars: binaries ; High-Performance Computing ; shock waves

1. Introduction

Colliding-wind binary (CWB) systems are a class of binary star system containing two massive stars with sufficiently fast and dense winds that the radiation from the shocked plasma in the wind-collision zone may be detected above the radiation from the stars themselves, typically in thermal X-rays [12, 8] and radio synchrotron radiation [3]. The most extreme of them accelerate particles to TeV energies [6], almost certainly through diffuse shock acceleration [16]. Many CWBs have eccentric orbits, resulting in periodic variation of the wind-collision radiation as the shock properties change in a predictable way through the orbit, and the archetype of this phenomenon is WR 140 [13]. This

* Corresponding author. Tel.: +353-1-440-6656
E-mail address: jmackey@cp.dias.ie

10.1016/j.procs.2024.07.011

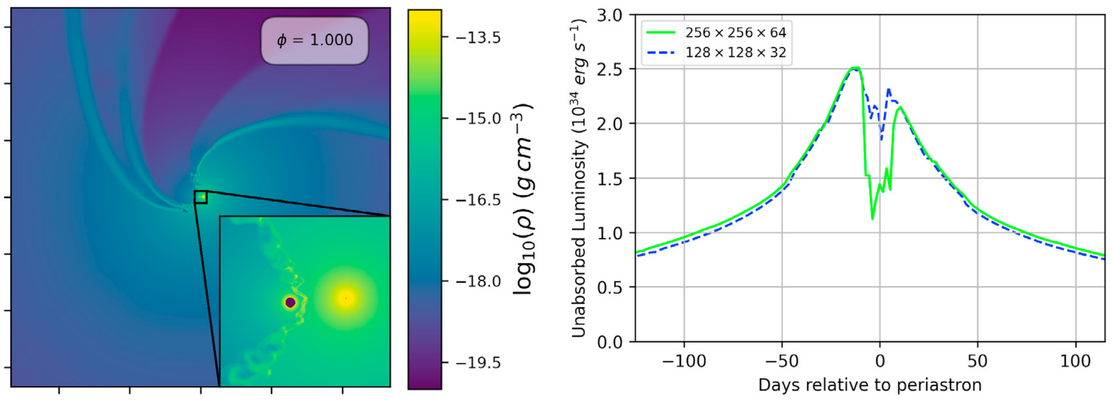


Fig. 1. **Left:** slice through the midplane of a simulation of WR 140 at periastron, showing log of gas density on the colour scale (adapted from [10], fig. 10). The tick marks are separated by 25 au (the Sun-Earth separation), and the inset is zoomed in on the system centre-of-mass by a factor of 32. The plot shows the deformation of the large-scale shocks by orbital motion, and the unstable radiative shock around the apex of the shock cone at periastron. The stellar interior is shown as a dark colour. **Right:** Resolution study showing X-ray luminosity from the wind-collision region as a function of time through periastron, for simulations with $128^2 \times 32$ and $256^2 \times 64$ cells per refinement level (from [10]). The plots are reproduced from figs. 10 and A1 of "Inverse-Compton cooling of thermal plasma in colliding-wind binaries", Mackey, J. *et al.*, Monthly Notices of the Royal Astronomical Society, Vol. 526, Issue 2, pp. 3099-3114.

system has a 7.9 year orbital period with an eccentricity of 0.9, and the wind-wind collision produces large quantities of interstellar dust for a short period around periastron (time of closest approach) on each orbit [7].

WR 140 consists of an O-type main-sequence star (powerd by hydrogen fusion in its core) with mass 30 $M_\odot$, and a Wolf-Rayet (WR) star of carbon type (WC) with mass 10 $M_\odot$ [15]. Although currently less massive than its companion, the WC star is more evolved and was the initially more massive and more luminous star. WR stars are defined by the strong emission lines of helium and other ions in their spectrum (the WC stars having strong carbon lines), produced by radiative transfer effects in their optically thick stellar wind [2]. In the case of WR 140, the WC star is a post-main-sequence star, most likely powered by helium fusion in its core, that has lost its outer hydrogen envelope to reveal the helium+carbon core. The evolutionary history of these extreme stars, that evolve close to the Eddington limit and are likely progenitors of black-hole binary systems, is still quite uncertain [14].

In the WR 140 system the two stars have similar wind speeds of around 3000 km s^{-1}, but the wind of the WC star is about 30 times denser that that of its companion O star. Ram-pressure balance dictates that the wind-collision region is very close to the O star, forming a bow-shaped region of shocked plasma bounded by two shocks [8]. The semimajor axis of the orbit is 14 au, with periastron and apastron separation of the two stars of 1.4 and 26.7 au, respectively [10]. The X-ray emission from the wind-wind collision has been extensively studied by space telescopes for at least 3 full orbits [13], giving us excellent data on the properties of the hot plasma. It is hoped that new missions such as *XRISM* will observe the upcoming periastron passage (in Nov. 2024) in even greater detail, giving new insights into shock physics and hydrodynamics [11].

The CWB systems are a valuable laboratory where we can constrain complex physical processes in extreme environments occuring in real time, namely the physics of collisionless shocks, particle acceleration, high-energy radiation processes, X-ray-emitting plasmas, radiative cooling and dust formation. In particular for systems with well-constrained orbits, detailed multi-wavelength observing campaigns can be planned precisely to answer specific questions. In [10] we introduced inverse-Compton (IC) cooling of the thermal electrons off the intense (but low-energy) radiation field of the two stars. We showed that this should be the dominant cooling process of the shocked plasma in the WR 140 system around periastron, with a shorter cooling timescale than collisional processes. This is important because the intermittent dust production in WR 140 implies that the shocks, which are adiabatic over almost all of the orbit, switch to becoming radiative for about a month around periastron [13]. A simple calculation shows that without IC cooling, the shock should remain adiabatic because the gas advection timescale (timescale to leave the wind-collision zone) is shorter than the cooling timescale.

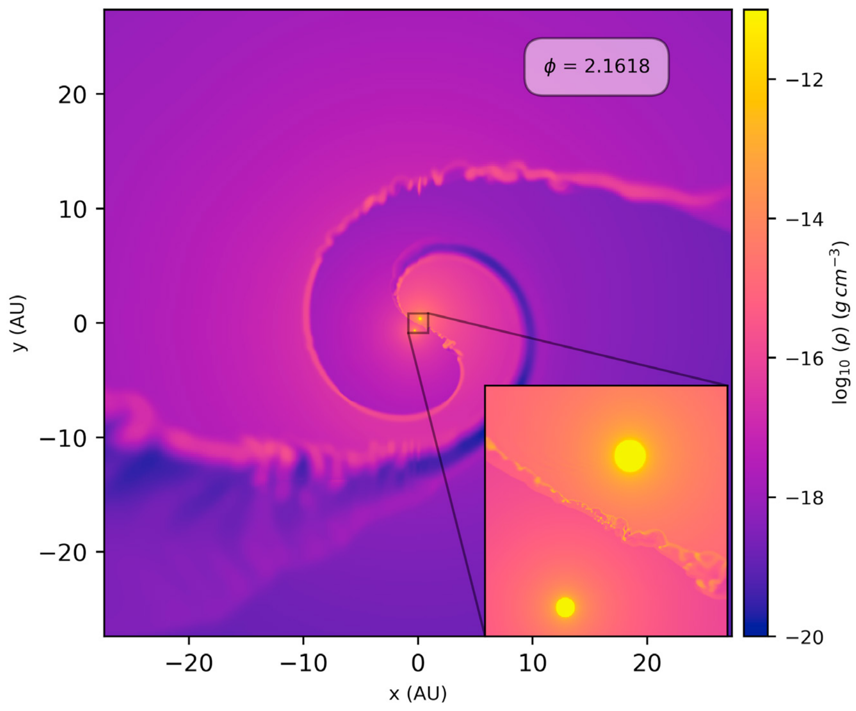


Fig. 2. Slice through the orbital plane $z = 0$ of a high-resolution 3D simulation of the colliding winds of the binary system WR 21a (section 3.1). The plot shows $\log_{10}$ of gas density at orbital phase 0.16, shortly after periastron. The main plot shows the full domain and the inset is zoomed in by a factor of 16. The stellar interior is bright yellow.

In [10] it was demonstrated with 2D and 3D magnetohydrodynamics (MHD) simulations that the addition of IC cooling triggers a runaway cooling instability in the shocked plasma of WR 140 around periastron, allowing the gas to cool from the post-shock temperature of $T \sim 10^7 - 10^8$ K down to $T \sim 10^4$ K (see Fig. 1, left panel). We obtained qualitative agreement with the observed X-ray lightcurve before, during and after periastron, although quantitatively the result had not converged with numerical resolution (Fig. 1, right panel). In particular the decrease in X-ray emission around periastron was deeper with increasing resolution, although still not deep enough to match observations [13]. This highlighted the need for more computationally intensive simulations with higher spatial resolution to better capture the plasma properties of the wind-collision zone, and motivated us to make some code improvements to enable better scaling on larger HPC systems.

In principle we could have continued to study WR 140 for the scaling tests, but we decided instead to use a model for the CWB system WR 21a. The motivation was that this is a very interesting and under-studied system (see the following subsection) where IC cooling should also be an important physical process, potentially allowing a better test of the impact of IC cooling than is possible with WR 140. We wanted to also use the code scaling tests to assess the viability of applying our wind-acceleration algorithm to other binary systems, especially shorter-period systems such as WR 21a which have smaller separation between the stars.

1.1. WR 21a: one of the most massive binary systems in the Galaxy

WR 21a is a binary system containing two main-sequence stars in an eccentric orbit (eccentricity $e \approx 0.7$) with a 32-day period [1]. The two stars have masses of 93 $M_\odot$ and 53 $M_\odot$, radii of 23 $_\odot$ and 14 $R_\odot$, and luminosities of $10^{6.2}$ $L_\odot$ and $10^{6.0}$ $L_\odot$ respectively [1] (where $M_\odot$, $R_\odot$ and $L_\odot$ are the mass, radius and luminosity of the Sun). The separation of the two stars at periastron is only 0.33 au, and the space between the two stellar surfaces is only 0.14 au (1 au is 215 $R_\odot$). At apastron (point of furthest separation) this increases to > 1.5 au. The mass-loss rate and wind terminal velocity of the primary star are estimated to be $\dot{M} = 10^{-5}\ M_\odot\,\mathrm{yr}^{-1}$ and $v_\infty = 2000\,\mathrm{km\,s}^{-1}$, and of the secondary

$\dot{M} = 2 \times 10^{-6}\,\mathrm{M}_\odot\,\mathrm{yr}^{-1}$ and $v_\infty = 3800\,\mathrm{km\,s}^{-1}$ [5]. We assume wind acceleration from the stellar surface according to the so-called β law: $v(r) = v_0 + (v_\infty - v_0)(1 - R_\star/r)^\beta$, where $R_\star$ is the radius of the star, v_0 is a reference velocity comparable to the sound speed in the photosphere, and β is a parameter that we take $\beta = 1$ (see e.g., [5]).

Remarkably, this system is among the most X-ray luminous of the CWBs containing main-sequence stars, on account of the tight orbit together with dense and fast winds [5]. The X-ray emission throughout the orbit is strongly variable, slowly increasing through the orbit, reaching a maximum shortly before periastron, followed by a sharp decrease to a minimum at or shortly after periastron [5]. There are no published hydrodynamic simulations of the wind-wind collision in this extreme binary system. Based on the extremely high luminosity of the two star, together with the short orbital period, we expect that inverse-Compton cooling of the shocked plasma should have an important effect in the shock dynamics of this system.

2. Methods and simulation setup

The project uses the radiation-MHD code PION [9] which implements a grid-based, finite-volume integration scheme that is accurate to 2nd order in time and space. Static mesh-refinement (in the form of recursively nested grids) is used to focus resolution and computational resources on specific regions of the simulation domain, with adaptive time-stepping for further efficiency gains. This method is particularly efficient when the regions requiring highest resolution can be predicted in advance, which is the case for CWBs with well-specified orbits. We showed that this method was effective for the simulations of the periastron passage of WR 140 [10], and our aim is to apply this to other binary systems, increasing the numerical resolution to obtain better-converged results. Higher resolution requires more computing resources, and so we applied for a *EuroHPC Benchmark Access* project on the Czech supercomputer *Karolina*[1] to test the parallel scaling of the algorithms on a large machine. The project[2] ran from June to September 2023 and involved porting PION to *Karolina*, setting up a series of simulations with varying resolution, and then performing weak- and strong-scaling tests on them.

A simulation was set up with two stars of mass, radius and luminosity consistent with measurements of the two components of WR 21a, and with the derived wind and orbital parameters (see section 1.1). The simulation domain is $[x, y, z] \in \pm[27.4, 27.4, 13.7]$ au, with the centre of mass of the binary system at the origin, and with 8 refinement levels, where each successive level is a factor of 2 smaller than the level above but with the same number of grid cells. The finest level has $[x, y, z] \in \pm[0.214, 0.214, 0.107]$ au, and contains both stars at periastron.

A slice through the orbital plane $z = 0$ of the highest-resolution simulation (see section 3.1 below) not long after periastron is shown in Fig. 2, showing $\log_{10}$ of gas density. The more massive and larger star has a denser and slower wind, and the effect of orbital motion on the shock structures is clear. The inset shows that the shock is radiative and unstable on the side of the more massive star, but the faster wind of the secondary star produces an adiabatic shock.

Wind acceleration, radiative cooling and inverse-Compton cooling of the plasma are implemented exactly as in [10], except that the two stars are assumed to have standard Galactic elemental abundances in line with their main-sequence status. In [10] the primary star in WR 140 is of WC type, with a wind composed mostly of helium and carbon, and radiative cooling appropriate for this plasma composition was used [4]. Here the two stars have the same composition and radiative cooling is the same in both cases.

3. Results

3.1. Strong scaling tests

For the strong-scaling test a high-resolution calculation was integrated from periastron to apastron, using $320^2 \times 160$ grid cells per refinement level, giving a cell diameter of $\Delta x = 2\times10^{10}$ cm, or $0.29\,\mathrm{R}_\odot$ on the finest level. The simulation was run on *Karolina* using 1, 2, 4, 8, 16, 32, and 64 nodes, corresponding to 128, 256, 512, 1024, 2048, 4096 and 8192 cores. Simulations were first run with only MPI parallelisation, i.e, 1 MPI process per core. Subsequent series of runs

[1] Operated by the IT4Innovations National Supercomputing Center of the Czech Republic.
[2] European High Performance Computing Joint Undertaking (EuroHPC JU) project `EHPC-BEN-2023B05-029`.

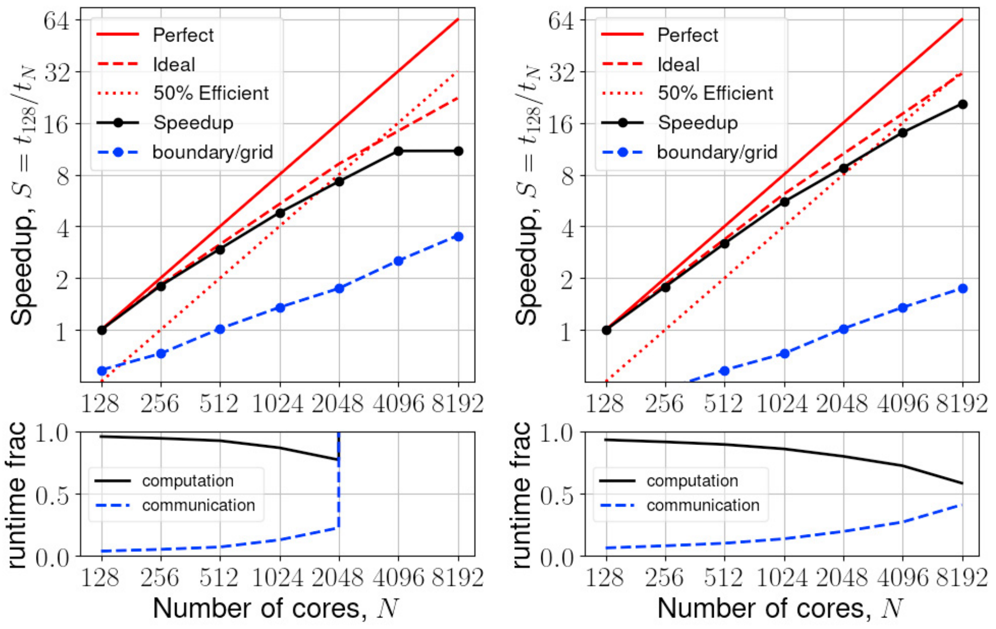


Fig. 3. Strong scaling of PION on *Karolina* for a 3D simulation of colliding winds. Left: Speedup for the case of pure MPI parallelisation, with 1 MPI process per core. Right: the case of hybrid MPI/OpenMP parallelisation with 1 MPI process for every 4 cores. The red solid line shows perfect scaling; red dashed line takes account of the extra work from calculating ghost zones in domain decomposition; red dotted line shows 50% efficiency, and black solid line shows the achieved speedup. The blue dashed line shows the ratio of ghost cells to grid cells. In the lower panels of each plot the black solid line shows the fraction of the time spent on computation (of grid and ghost cells) and the blue dashed line the time fraction on communication. No data are present for 4096 and 8192 cores on the left plot because these simulations did not progress far enough to report the information. Strong scaling shown for these two datapoints is an extrapolation based on the smaller number of timesteps completed by these runs.

were performed with hybrid MPI/OpenMP parallelisation, using 2, 4 or 8 cores per MPI process, and the same number of OpenMP threads per process. Tests with 16 OpenMP threads per MPI process showed relatively poor performance and were not pursued further.

Strong scaling results with 1 MPI process per core, and with 1 MPI process per 4 cores (with 4 OpenMP threads), are shown in Fig. 3. The total number of cells calculated increases with the number of MPI processes because of the buffers of ghost cells required to maintain solution consistency across sub-domains. This is reflected by the red dashed line, which shows the expected best possible scaling given the increasing computation caused by ghost cells. The scaling of the pure MPI runs is pretty good up to 2048 cores, a factor of 16 increase over the base run on a single node. After this it drops well below 50% efficiency because of the large quantity of extra calculation and communication required. The runs with 4096 and 8192 cores did not complete in the allocated walltime and so the breakdown between communications and computation was not available, but clearly these runs were inefficient.

Using 4 OpenMP threads on 4 cores per MPI process, shown on the right panel of Fig. 3, shows better strong scaling up to a larger number of cores. This is largely due to having less ghost-cell computation because the number of subdomains is 4 times smaller than in the MPI-only runs. Surprisingly, using 8 threads was less efficient than 4 threads, and an analysis of the timings spent in various subroutines revealed that there were parts of the code that were not efficiently multi-threaded, particularly those for calculating wind acceleration and inverse-Compton cooling.

3.2. Weak scaling test

For the weak-scaling test, we start with the same simulation as for the strong-scaling test, but ran different versions of this with fewer grid cells for running on fewer nodes. Specifically, simulations using $128^2 \times 64$, $160^2 \times 80$, $192^2 \times 96$ and $256^2 \times 128$ grid cells per level were evolved to the same starting point as was used for the strong-scaling test with $320^2 \times 160$ cells per level. These simulations were run on 1, 2, 4, 8 and 16 nodes of *Karolina*, respectively, in all cases with 8 OpenMP threads on 8 cores per MPI process. The relative number of grid cells per node for these 5 simulations

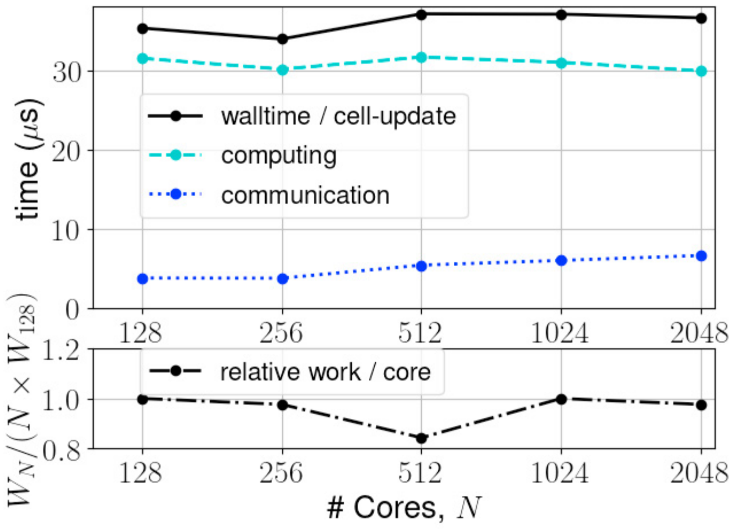


Fig. 4. Weak scaling of PION on *Karolina* for a 3D simulation of colliding winds, for the case of hybrid MPI/OpenMP parallelisation with 1 MPI process for every 8 cores (with 8 OpenMP threads). The black solid line shows the time required per cell-update per core; the cyan dashed line shows that most of this time is used for computation and the blue dotted line shows the time used for communication. The lower panel shows the relative amount of work per core for each of these simulations.

is 1.0 : 0.98 : 0.84 : 1.0 : 0.98 and so it is not a perfect weak-scaling test, but is as close as possible given the requirement that the domain should be subdivided in 2 many times along each dimension.

The speed of each of these simulations is shown in Fig. 4, presented as the walltime required per cell-update per core. The weak scaling is excellent, showing basically no decrease in speed over a factor of 16 range in core count. We did not have sufficient resources to extend this plot to an even larger simulation run on 4096 or 8192 cores. Overall performance is slower than running a pure hydrodynamics calculation, because here we also have the radiative cooling source-term, computation of the inverse-Compton cooling rate from the stellar radiation field, calculation of wind acceleration in each cell, and regular updates of the wind boundary regions because of orbital motion.

3.3. *Code optimization*

Building on these benchmarking tests with *Karolina*, we made some code optimizations to improve the multi-threading performance with larger numbers of OpenMP threads. The communication between refinement levels was improved, sub-domain boundary communication was optimized, and calculation of wind acceleration and the radiation field (for inverse-Compton cooling) was more efficiently multi-threaded. These improvements were tested in December 2023 and January 2024 on the supercomputer *Kay* of the Irish Centre for High-End Computing (ICHEC) on up to 2560 cores and up to 20 OpenMP threads per MPI process. The results are presented in Fig. 5, where the speedup on *Kay* with 20 OpenMP threads is compared with results from the *EuroHPC* benchmarking on *Karolina* with 1, 4, and 8 OpenMP threads. The speedup is normalised to running the calculation on 1 node of *Karolina*, and we can see that the older CPUs of *Kay* are somewhat slower. Despite this, the performance with 20 OpenMP threads on *Kay* is just as good as with 4 OpenMP threads on *Karolina* when run on 2560 cores, and it is better than running with other thread counts on *Karolina*. Unfortunately we could not test the performance on *Kay* up to larger numbers of cores due to queue-size limitations, but we will make this test on a larger computer for a forthcoming publication to accompany a new release of PION, version 3.0.

3.4. *Scientific results*

For the highest resolution simulation the predicted X-ray luminosity in the 2.5 – 10 keV waveband is shown in Fig. 6 as a function of orbital phase (in units such that 0 → 1 is a full orbit). This can be compared with the observational

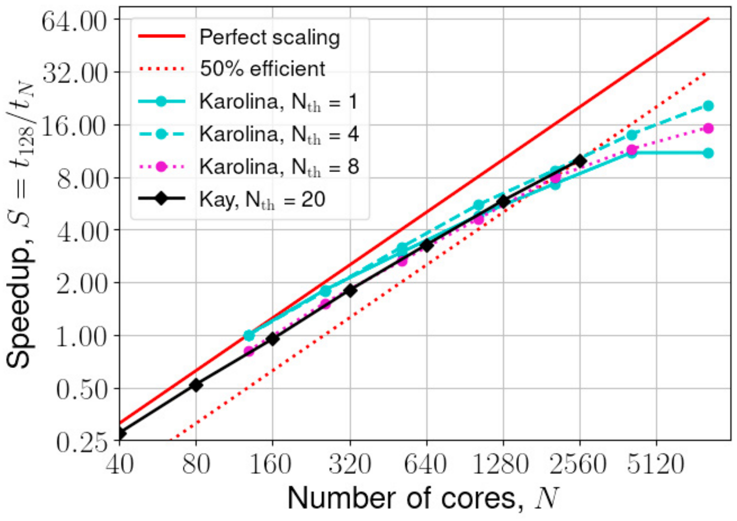


Fig. 5. Strong scaling of PION for a 3D simulation of colliding winds. The data for *Karolina* are the same as in Fig. 3 for 1 (cyan solid line) and 4 (cyan dasked line) threads per MPI process, with the addition of results using 8 OpenMP threads per MPI process (magenta dotted line). In addition the solid black line shows results using the ICHEC machine *Kay* with 20 OpenMP threads per MPI process, run with a version of PION optimized for larger thread counts. All curves are normalised to the runtime for running with 1 thread per MPI process on 1 node of *Karolina* (128 cores). *Kay* has somewhat less raw speed on account of its older CPUs.

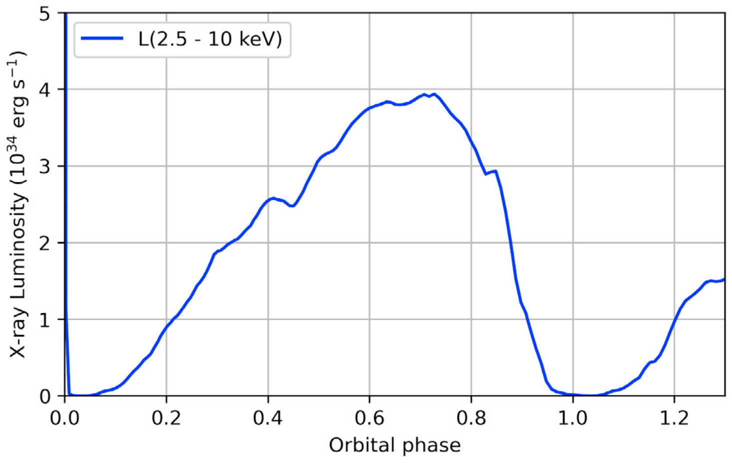


Fig. 6. X-ray luminosity of the colliding-wind region of WR 21a as a function of orbital phase, from a high-resolution hydrodynamical simulation. The calculation assumes collisional ionization equilibrium and is calculated in the energy range 2.5 – 10 keV.

data from [5]. The general features of the lightcurve are similar, namely the strong dip around periastron at phase 1, the steady increase during the orbit to a maximum before periastron, followed by a strong decrease to the minimum. There are quantitative differences, however, in particular that the observations show almost constant flux in the phase range 0.2 – 0.7, not seen in the simulation. Furthermore, the maximum of the emission is at phase 0.75 from the simulation, but at about phase 0.9 from the observations. The dip at periastron is effectively to zero emission in the simulation whereas it is clearly non-zero in the observations.

There may still be effects of local and phase-dependent X-ray absorption in the observed light-curve which are not taken into account in the simulated data. Alternatively, our model may not yet be a good description of the wind-wind interaction. The poorly constrained wind parameters ($\dot{M}$ and v_∞) of the two stars may be set incorrectly in the simulations, or the complex physics of radiation-driven wind acceleration may not be adequately modelled with our simplified prescription.

4. Conclusions

Using the astrophysics MHD code PION [9] we demonstrated a method for making high-resolution 3D simulations of the magnetised wind-wind collision in binary systems containing two massive stars, including effects of wind acceleration, orbital motion and inverse-Compton cooling [10]. With the aim of scaling up these simulations to higher spatial resolution on larger computers, we benchmarked the performance of PION on the *EuroHPC* system *Karolina* during Summer 2023. The strong scaling on a colliding-wind test calculation was demonstrated to run at $> 50\,\%$ efficiency on 16 nodes compared with running on 1 node of *Karolina*, using 4 OpenMP threads (one thread per core) per MPI process. Weak scaling was excellent again up to 16 nodes, and could not be tested further because the benchmarking resources were exhausted. Bottlenecks were found when run with 8 or more OpenMP threads per MPI process, and subsequent code optimizations were made with the result that PION now runs efficiently with at least 20 OpenMP threads per MPI process, demonstrated using the ICHEC machine *Kay*.

Analysing the simulations run for this project, the X-ray emission from the CWB system WR 21a is qualitatively well-modelled although there are quantitative differences in the emission as a function of orbital phase. These may be due to choosing incorrect parameters for the stellar winds or inadequacies in the modelling of the wind acceleration away from the surface of each star. Future work will address these uncertainties to construct a computational model for this extremely interesting binary system – one of the most extreme CWBs in the Galaxy.

Acknowledgements

JM acknowledges support from a Royal Society-Science Foundation Ireland University Research Fellowship. AM acknowledges support from a Royal Society Research Fellows Enhancement Award 2021. This work was supported by an Irish Research Council (IRC) Starting Laureate Award. RB acknowledges funding from the Irish Research Council under the Government of Ireland Postdoctoral Fellowship program.

References

[1] Barbá, R.H., Gamen, R.C., Martín-Ravelo, P., Arias, J.I., Morrell, N.I., 2022. The winking eye of a very massive star: WR 21a revealed as an eclipsing binary by TESS. MNRAS 516, 1149–1157. doi:10.1093/mnras/stac2173, arXiv:2109.06311.

[2] Crowther, P.A., 2007. Physical Properties of Wolf-Rayet Stars. ARA&A 45, 177–219. doi:10.1146/annurev.astro.45.051806.110615, arXiv:astro-ph/0610356.

[3] Dougherty, S.M., Beasley, A.J., Claussen, M.J., Zauderer, B.A., Bolingbroke, N.J., 2005. High-Resolution Radio Observations of the Colliding-Wind Binary WR 140. ApJ 623, 447–459. doi:10.1086/428494, arXiv:astro-ph/0501391.

[4] Eatson, J.W., Pittard, J.M., Van Loo, S., 2022. An exploration of dust grain growth within WCd systems using an advected scalar dust model. MNRAS 516, 6132–6144. doi:10.1093/mnras/stac2617, arXiv:2204.07397.

[5] Gosset, E., Nazé, Y., 2016. The X-ray light curve of the massive colliding wind Wolf-Rayet + O binary WR 21a. A&A 590, A113. doi:10.1051/0004-6361/201527051, arXiv:1604.01536.

[6] H.E.S.S. Collaboration, Abdalla, H., Adam, R., Aharonian, F., Ait Benkhali, F., Angüner, E.O., Arakawa, M., Arcaro, C., Armand, C., Armstrong, T., Ashkar, H., Backes, M., Barbosa Martins, V., Barnard, M., Becherini, Y., Berge, D., Bernlöhr, K., Blackwell, R., Böttcher, M., Boisson, C., Bolmont, J., Bonnefoy, S., Bregeon, J., Breuhaus, M., Brun, F., Brun, P., Bryan, M., Büchele, M., Bulik, T., Bylund, T., Caroff, S., Carosi, A., Casanova, S., Cerruti, M., Chand, T., Chandra, S., Chen, A., Colafrancesco, S., Cotter, G., Curyło, M., Davids, I.D., Davies, J., Deil, C., Devin, J., deWilt, P., Dirson, L., Djannati-Ataï, A., Dmytriiev, A., Donath, A., Doroshenko, V., Dyks, J., Egberts, K., Eichhorn, F., Emery, G., Ernenwein, J.P., Eschbach, S., Feijen, K., Fegan, S., Fiasson, A., Fontaine, G., Funk, S., Füßling, M., Gabici, S., Gallant, Y.A., Gaté, F., Giavitto, G., Giunti, L., Glawion, D., Glicenstein, J.F., Gottschall, D., Grondin, M.H., Hahn, J., Haupt, M., Heinzelmann, G., Henri, G., Hermann, G., Hinton, J.A., Hofmann, W., Hoischen, C., Holch, T.L., Holler, M., Hörbe, M., Horns, D., Huber, D., Iwasaki, H., Jamrozy, M., Jankowsky, D., Jankowsky, F., Jardin-Blicq, A., Joshi, V., Jung-Richardt, I., Kastendieck, M.A., Katarzyński, K., Katsuragawa, M., Katz, U., Khangulyan, D., Khélifi, B., King, J., Klepser, S., Kluźniak, W., Komin, N., Kosack, K., Kostunin, D., Kreter, M., Lamanna, G., Lemière, A., Lemoine-Goumard, M., Lenain, J.P., Leser, E., Levy, C., Lohse, T., Lypova, I., Mackey, J., Majumdar, J., Malyshev, D., Malyshev, D., Marandon, V., Marchegiani, P., Marcowith, A., Mares, A., Martí-Devesa, G., Marx, R., Maurin, G., Meintjes, P.J., Moderski, R., Mohamed, M., Mohrmann, L., Moore, C., Morris, P., Moulin, E., Muller, J., Murach, T., Nakashima, S., Nakashima, K., de Naurois, M., Ndiyavala, H., Niederwanger, F., Niemiec, J., Oakes, L., O'Brien, P., Odaka, H., Ohm, S., de Ona Wilhelmi, E., Ostrowski, M., Panter, M., Parsons,

R.D., Perennes, C., Petrucci, P.O., Peyaud, B., Piel, Q., Pita, S., Poireau, V., Priyana Noel, A., Prokhorov, D.A., Prokoph, H., Pühlhofer, G., Punch, M., Quirrenbach, A., Raab, S., Rauth, R., Reimer, A., Reimer, O., Remy, Q., Renaud, M., Rieger, F., Rinchiuso, L., Romoli, C., Rowell, G., Rudak, B., Ruiz-Velasco, E., Sahakian, V., Sailer, S., Saito, S., Sanchez, D.A., Santangelo, A., Sasaki, M., Scalici, M., Schlickeiser, R., Schüssler, F., Schulz, A., Schutte, H.M., Schwanke, U., Schwemmer, S., Seglar-Arroyo, M., Senniappan, M., Seyffert, A.S., Shafi, N., Shiningayamwe, K., Simoni, R., Sinha, A., Sol, H., Specovius, A., Spencer, S., Spir-Jacob, M., Stawarz, Ł., Steenkamp, R., Stegmann, C., Steppa, C., Takahashi, T., Tavernier, T., Taylor, A.M., Terrier, R., Tiziani, D., Tluczykont, M., Tomankova, L., Trichard, C., Tsirou, M., Tsuji, N., Tuffs, R., Uchiyama, Y., van der Walt, D.J., van Eldik, C., van Rensburg, C., van Soelen, B., Vasileiadis, G., Veh, J., Venter, C., Vincent, P., Vink, J., Völk, H.J., Vuillaume, T., Wadiasingh, Z., Wagner, S.J., Watson, J., Werner, F., White, R., Wierzcholska, A., Yang, R., Yoneda, H., Zacharias, M., Zanin, R., Zdziarski, A.A., Zech, A., Zorn, J., Żywucka, N., 2020. Detection of very-high-energy γ-ray emission from the colliding wind binary η Car with H.E.S.S. A&A 635, A167. doi:10.1051/0004-6361/201936761, arXiv:2002.02336.

[7] Lau, R.M., Hankins, M.J., Han, Y., Argyriou, I., Corcoran, M.F., Eldridge, J.J., Endo, I., Fox, O.D., Garcia Marin, M., Gull, T.R., Jones, O.C., Hamaguchi, K., Lamberts, A., Law, D.R., Madura, T., Marchenko, S.V., Matsuhara, H., Moffat, A.F.J., Morris, M.R., Morris, P.W., Onaka, T., Ressler, M.E., Richardson, N.D., Russell, C.M.P., Sanchez-Bermudez, J., Smith, N., Soulain, A., Stevens, I.R., Tuthill, P., Weigelt, G., Williams, P.M., Yamaguchi, R., 2022. Nested dust shells around the Wolf-Rayet binary WR 140 observed with JWST. Nature Astronomy 6, 1308–1316. doi:10.1038/s41550-022-01812-x, arXiv:2210.06452.

[8] Luo, D., McCray, R., Mac Low, M.M., 1990. X-Rays from Colliding Stellar Winds. ApJ 362, 267. doi:10.1086/169263.

[9] Mackey, J., Green, S., Moutzouri, M., Haworth, T.J., Kavanagh, R.D., Zargaryan, D., Celeste, M., 2021. PION: simulating bow shocks and circumstellar nebulae. MNRAS 504, 983–1008. doi:10.1093/mnras/stab781, arXiv:2103.07555.

[10] Mackey, J., Jones, T.A.K., Brose, R., Grassitelli, L., Reville, B., Mathew, A., 2023. Inverse-Compton cooling of thermal plasma in colliding-wind binaries. MNRAS 526, 3099–3114. doi:10.1093/mnras/stad2839, arXiv:2301.13716.

[11] Miyamoto, A., Sugawara, Y., Maeda, Y., Ishida, M., Hamaguchi, K., Corcoran, M., Russell, C.M.P., Moffat, A.F.J., 2024. X-ray plasma flow and turbulence in the colliding winds of WR140. MNRAS 527, 7121–7135. doi:10.1093/mnras/stad3635, arXiv:2401.00977.

[12] Pollock, A.M.T., 1987. The Einstein View of the Wolf-Rayet Stars. ApJ 320, 283. doi:10.1086/165539.

[13] Pollock, A.M.T., Corcoran, M.F., Stevens, I.R., Russell, C.M.P., Hamaguchi, K., Williams, P.M., Moffat, A.F.J., Weigelt, G., Shenavrin, V., Richardson, N.D., Espinoza, D., Drake, S.A., 2021. Competitive X-Ray and Optical Cooling in the Collisionless Shocks of WR 140. ApJ 923, 191. doi:10.3847/1538-4357/ac2430, arXiv:2109.10350.

[14] Shenar, T., 2022. Wolf-Rayet stars: recent advances and persisting problems, in: Mackey, J., Vink, J., St-Louis, N. (Eds.), Massive Stars Near and Far; Proceedings of the IAU, Vol. 361, in press, p. arXiv:2208.02614. doi:10.48550/arXiv.2208.02614, arXiv:2208.02614.

[15] Thomas, J.D., Richardson, N.D., Eldridge, J.J., Schaefer, G.H., Monnier, J.D., Sana, H., Moffat, A.F.J., Williams, P., Corcoran, M.F., Stevens, I.R., Weigelt, G., Zainol, F.D., Anugu, N., Le Bouquin, J.B., ten Brummelaar, T., Campos, F., Couperus, A., Davies, C.L., Ennis, J., Eversberg, T., Garde, O., Gardner, T., Fló, J.G., Kraus, S., Labdon, A., Lanthermann, C., Leadbeater, R., Lester, T., Maki, C., McBride, B., Ozuyar, D., Ribeiro, J., Setterholm, B., Stober, B., Wood, M., Zurmühl, U., 2021. The orbit and stellar masses of the archetype colliding-wind binary WR 140. MNRAS 504, 5221–5230. doi:10.1093/mnras/stab1181, arXiv:2101.10563.

[16] White, R., Breuhaus, M., Konno, R., Ohm, S., Reville, B., Hinton, J.A., 2020. Gamma-ray and X-ray constraints on non-thermal processes in η Carinae. A&A 635, A144. doi:10.1051/0004-6361/201937031, arXiv:1911.01079.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 91–98

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

New physics in the muon magnetic moment?

Kalman K. Szabo[a], Laurent Lellouch[b], Zoltan Fodor[c], Finn Stokes[a], Balint C. Toth[c,*], Gen Wang[b]

[a]Forschungszentrum Juelich, Juelich Supercomputing Centre, Wilhelm Johnen Strasse, 52425 Juelich, Germany
[b]CNRS and Aix-Marseille University, Centre de Physique Théorique, 163 Avenue de Luminy, 13288 Marseille, France
[c]University of Wuppertal, Faculty of Mathematics and Natural Sciences, Gaussstr 20, 42349 Wuppertal, Germany

Abstract

The muon, a short-lived cousin of the electron, has provided a longstanding discrepancy between the standard model of particle physics and experimental measurements. This suggests that a not-yet-known particle or force perturbs the muon. The discovery of such "new physics" would have profound consequences on our understanding of Nature.

In 2021, the attention of the world was drawn to this discrepancy when the announcement of the independent confirmation of the experiment by Fermilab [1] coincided with the publication of our ab-initio calculation (Nature [2]). Our result dramatically updated the theoretical prediction, bringing it significantly closer to the experimental value: it may be possible to explain the Fermilab measurement without any new physics, even with the latest Fermilab update [3]. Our result established a new standard of precision for such calculations that has yet to be challenged; with uncertainties comparable to the experimental measurement and the reference, data-driven computations.

We are carrying out new simulations to reduce both the systematic and statistical uncertainties. Both improvements are required in order to match the precision of the final Fermilab measurement, to be obtained in the coming years. As such, these simulations will be critical to determine whether the muon's magnetic moment harbours new fundamental physics.

Keywords: particle physics; beyond Standard Model; lattice field theory; gpu acceleration

1. Introduction

The muon is an elementary particle, a short-lived cousin of the electron. For many years the calculation of its magnetic moment has disagreed with its measurement, suggesting that a not-yet-known particle or force perturbs the muon. Such a discovery would have profound consequences on our understanding of Nature. This is an extremely interesting time as there is a race between theoretical predictions and experimental findings, which might eventually lead to the discovery of a new interaction. If not, even that case would be an unprecedented triumph for modern

* Corresponding author. Tel.: +49-202-439-3479 ; fax: +49-202-439-3860.
E-mail address: btoth@uni-wuppertal.de

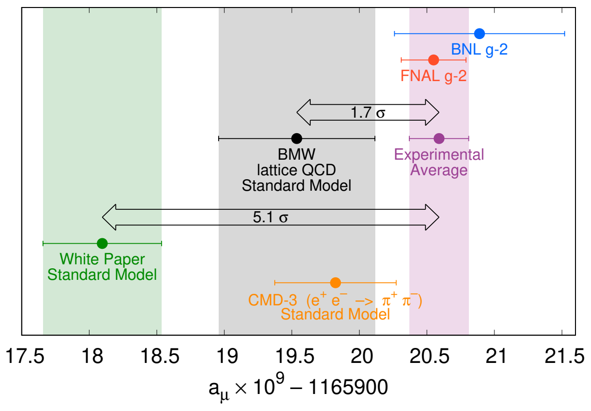


Fig. 1. Current status of theory and experiment for the anomalous contribution to the muon magnetic moment, denoted $a_\mu = \frac{1}{2}(g-2)$. Is there new physics or not? The blue dot shows the previous experimental result from Brookhaven National Laboratory (BNL) [4], while the red dot is the most recent result from the Fermilab experiment [3]. The purple band represents the current experimental average value. The green band represents the theoretical prediction given in the White Paper of the g-2 Theory Initiative [5], where the hadronic vacuum polarization (HVP) contribution is obtained using the data-driven approach. It has a 5.1σ discrepancy with the experimental value, indicating the existence of new physics, when taken at face value. The grey band shows the theoretical prediction, if the HVP contribution is replaced by the result of our lattice computation [2]. It reduces the tension between theory and experiment to 1.7σ. The orange point depicts the data-driven theoretical result, if the two-pion contribution to the HVP is taken from the recent CMD-3 measurements [6]. As of now it is not clear, how this new data should be included into an update of the theory prediction for the muon $g-2$.

physics, since the prediction of the Standard Model with three fundamental interactions would be proved up to an extremely high precision.

The experimental measurement at Fermilab National Laboratory is performing according to plan and in terms of precision has already outperformed the previous experiment from Brookhaven National Laboratory (BNL) [4]. As discussed above a new result from the Fermilab experiment was announced (called Fermilab/Run-2-3) [3]. These together with the earlier Fermilab (called Fermilab/Run-1) and BNL results are shown in Figure 1. In the next few years it is expected that Fermilab will reduce the uncertainty by a further factor of two. In addition, there is a facility under construction in Japan (called J-PARC) [7], which operates at a smaller beam energy but aims at the same precision.

The gold-standard of the theory computation is given by the White Paper of the g-2 Theory Initiative [5]. Its result is shown as the green dot and its error as the green band in Figure 1. This value differs from the combined Fermilab/Run1-2-3 and BNL experiments by 5.1σ, which is usually considered to be a very strong signal for physics beyond the Standard Model. As we discuss later the experiment did not announce this measurement as a sign of new physics, mainly because of our lattice result. The contribution to the Standard Model prediction with the largest uncertainty by far is the so-called hadronic vacuum polarization (HVP), shown in Figure 3. In the White Paper this is obtained from a phenomenological approach, sometimes called the R-ratio method. It uses a vast amount of data collected from electron-positron annihilation experiments.

There is an important new development in the direct measurements of the electron-positron cross-section. The CMD collaboration from Novosibirsk, Russia released their latest data (CMD-3) in the energy window 0.60–0.88 GeV for the two pion channel [6]. This channel provides approximately 70% of the hadronic vacuum polarization contribution into a_μ. The new result is shown in Figure 2, and it is in a strong, 4.4σ tension with the previous world average, which enters into the White Paper prediction. As of now it is not clear, how this new data should be included into an update of the theory prediction for the muon $g-2$. As can be seen in Figure 1, there is no tension with the direct magnetic moment measurement, if one takes the CMD-3 data at face value.

Finally, there is a soaring of lattice QCD determinations of the hadronic vacuum polarization contribution. These are ab-initio computations, and obviously do not rely on experimental determinations of the R-ratio. The first full

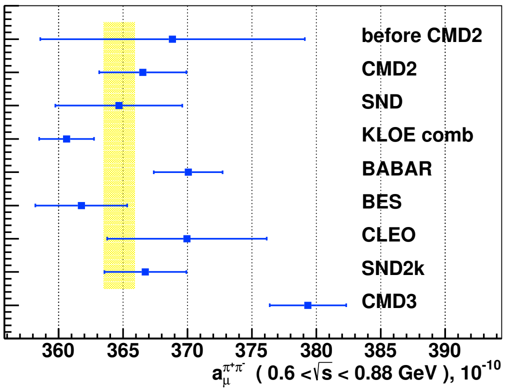


Fig. 2. Two-pion contribution to the hadronic vacuum polarization (HVP) obtained in different electron-positron annihilation experiments. Figure taken from [6].

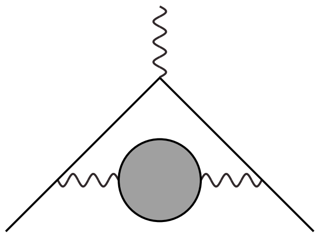


Fig. 3. The hadronic vacuum polarization (HVP) contribution to the muon magnetic moment: Before the muon (depicted as the straight line) interacts with the external magnetic field (shown as the vertical wavy line in the top of the figure), it emits a virtual photon (horizontal wavy line). This virtual photon polarizes the vacuum, temporarily creating the hadronic blob represented by the grey circle, then finally is recaptured by the leaving muon.

computation was done by our group and was published in Nature in 2021 [2]. This result is shown in Figure 1 by a black dot and its error by the grey band. It reduces the 5.1σ discrepancy to a mere 1.7σ.

To fully exploit the potential for new physics searches of the future Fermilab and J-PARC experiments, lattice computations must reach the same precision. To that end we have to have an even finer lattice than we have had until now: this is the aim of the current project.

2. Computation of the HVP contribution

The leading order, hadronic vacuum polarization (LO-HVP) describes how the propagation of a virtual photon is modified by the presence of quark and gluon fluctuations in the vacuum. The corresponding contribution to the muon magnetic moment, shown in Figure 3, dominates the uncertainty on the theoretical determination. Here we compute this LO-HVP contribution $a_\mu^{\text{LO-HVP}}$, using ab initio simulations in quantum chromodynamics (QCD) and quantum electrodynamics (QED). In the present work, we include both QED and QCD in a lattice formulation, including four non-degenerate quark flavors (up, down, strange and charm). We also consider the tiny contributions of the bottom and top quarks.

In lattice quantum field theory a space-time grid is introduced, and both the fields as well as the equations of the theory are discretized. The resolution of the lattice is determined by the smallest distance a between neighbouring grid points, which is called the lattice spacing. The so obtained finite number of degrees of freedom are distributed on a supercomputer, and then the discretized equations of QCD and QED are solved on this lattice. In order to obtain results

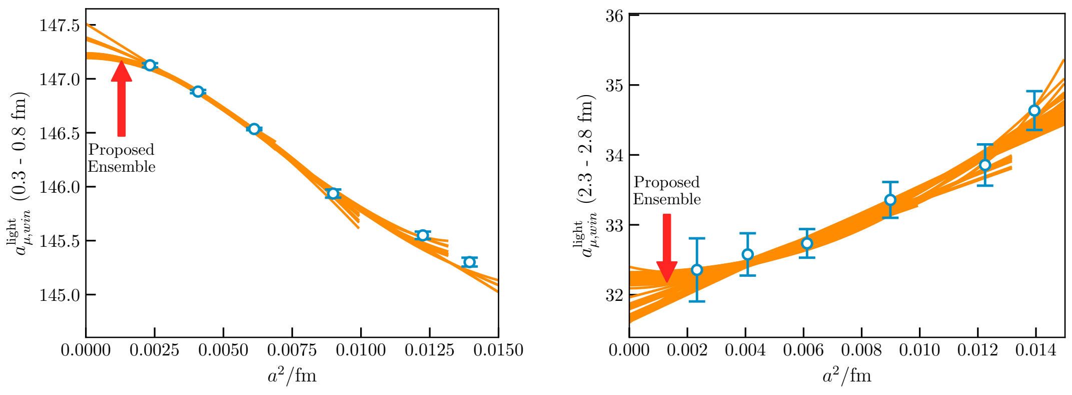


Fig. 4. Continuum limit in two different Euclidean time windows. On the left we have a short-distance contribution with no taste corrections applied. On the right we have a long-distance contribution with the SRHO taste correction from our previous work [2]. In both plots the orange curves show representative fits that contribute most strongly to the analysis. These curves are cut off where cuts have been applied to exclude the largest lattice spacings from the fit. The red arrows represent the position of the planned, additional lattice spacing, that helps reducing the uncertainty arising from the continuum limit procedure.

corresponding to the physical world, one has to take the continuum limit. This means that the computations have to be repeated at smaller and smaller lattice spacings, and the results have to be extrapolated to the $a = 0$ continuum case.

3. Continuum limit

The largest source of systematic error in our previous work [2] comes from the continuum extrapolation, 4.2 out of the total systematic error of 5.0, both in units of 10^{-10}. The continuum extrapolation error is obtained as a variation among various approaches to perform the continuum extrapolation. This includes changing the number of lattice spacings in the fit, changing the fit function from linear to quadratic in a^2, multiplying a^2 by powers of the strong coupling constant $\alpha_s(a)$, and applying different improvement procedures.

We can reduce the continuum extrapolation error by going to finer lattices. In a first step, we added a new lattice spacing $a = 0.048$ fm to our ensemble set, with lattice size $128^3 \times 192$. With previous computer time allocations we generated 5000 gauge configurations, a large part of which is already analysed for the hadron spectrum, a gradient flow scale and the current-current correlator, which gives the HVP contribution to a_μ. We are in the process of analysing these new data and preliminary results were reported at the lattice conferences in 2022 and 2023.

In the present phase of the project we are adding an even finer ensemble, $a = 0.036$ fm, with a $176^3 \times 264$ lattice size. This 30% decrease from our currently available finest lattice spacing will reduce the lattice artefacts by about 70% and will give an additional lever arm to our a^2 continuum extrapolations. With this new ensemble we will be able to use five different lattice spacings finer than $a = 0.1$ fm, with up to cubic continuum extrapolations in the a^2 scaling variable.

In Figure 4 we show the continuum extrapolation of the contribution to the full calculation within two different length scales. On the left, we show a short-distance contribution, where systematic errors dominate, and on the right a long-distance contribution which includes a model correction, and has larger statistical errors. In both cases, all existing lattice spacings including the new $a = 0.048$ fm are shown, and the proposed new lattice spacing is indicated by the red arrow. The location of the proposed point within a region where the representative fits (orange curves) are spreading out indicates that the new data at this point is expected to be very useful in constraining these fits.

4. Intermediate window

The RBC/UKQCD collaboration defined a particularly useful observable $a_{\mu,\text{win}}$, in which the current propagator is restricted to an intermediate Euclidean time window using a smooth weight function [8]. The advantage of $a_{\mu,\text{win}}$,

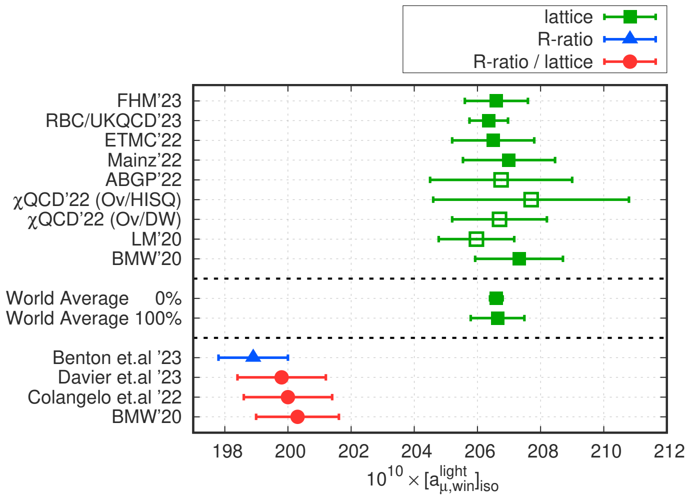


Fig. 5. Isospin-symmetric, light, connected component of the window observable, $[a^{\text{light}}_{\mu,\text{win}}]_{\text{iso}}$. The green squares denote lattice results. From each group only the most recent values are shown: FHM'23 [12], RBC/UKQCD'23 [13], ETMC'22 [14], Mainz'22 [15], ABGP'22 [16], χQCD'22 with overlap valence on HISQ and overlap valence on domain wall configurations [17], LM'20 [18], and our result BMW'20 [2]. The data points with filled squares give a set of independent results: each of these are obtained using a different set of configurations. The points World Average 0% and 100% represent their weigthed average, assuming 0% and 100% correlation between their systematics, respectively. The red dots show the R-ratio/lattice based determinations of Davier et.al. [10], Colangelo et.al. [9] and BMW'20 [2]. They are obtained by combining results from the data-driven approach for $a_{\mu,\text{win}}$ and lattice results for the non-light-connected contributions, as described in [2]. The blue triangle shows the R-ratio based determination of Benton et.al. [11]. This result is 6.9 sigmas away from the averaged lattice result assuming 0% correlation, and 5.6 sigmas away from the lattice average assuming 100% correlation.

which gives roughly 1/3 of the total HVP contribution, over a_μ, is that its calculation on the lattice is much less challenging than that of the full a_μ. The restriction to the intermediate window eliminates both the short-distance region, where large cutoff effects are present, and the long-distance region, where the statistical uncertainties and finite-size effects are large. Moreover, in the case of staggered fermions, the taste-breaking artefacts are reduced as well. This makes $a_{\mu,\text{win}}$ a strong benchmark quantity, allowing the comparison of the results of different lattice groups already at an early stage. The most recent value from each lattice collaboration is collected in Figure 5. As can be seen, all the lattice results agree nicely, and confirm our results of [2].

On the other hand, $a_{\mu,\text{win}}$ can be computed in the R-ratio approach as well [2, 9, 10, 11], which can then be compared to the values obtained on the lattice, providing a sharp comparison of the two methods. As Figure 5 shows, the R-ratio based determinations are in a strong tension with the lattice computations.

5. Code performance

In this project we use `dynqcd`, which has been developed continously for more than a decade. It includes state-of-the-art algorithms and has been deployed to several different architectures. We have optimized the code in a variety of ways:

- **Vectorization.** We designed our code from the beginning to enable auto-vectorization of time critical functions by the compiler. For this we fuse N sites in a given direction (`sitefusedir`) into a vector structure and all lattice fields are built up from such vectors. The GCC compiler does a good job when working with such structures: it automatically uses the SSE or AVX registers for operating on them.
- **Threading.** Currently the loops can be either parallelized using threads from the `pthreads` library or by OpenMP directives.

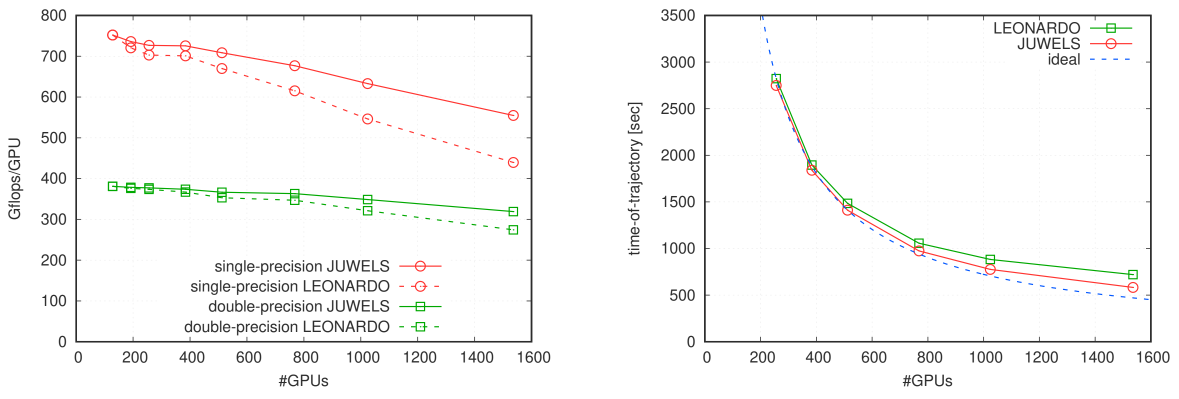


Fig. 6. Code performance as the function of the number of GPUs. The plots show strong scaling on a $176^3 \times 264$ lattice. Left: Gflops/GPU in multishift conjugate gradient iterations. Measurements performed on JUWELS Booster are connected with a solid line, estimates for the Leonardo Booster are connected with a dashed line. Right: time to solution in seconds, measurements on JUWELS and estimates on Leonardo are shown with different colors.

- **Communication.** The standard way of doing inter-node communication uses the MPI library. The code currently can split three out of the four directions for communication. We have also implemented the one-sided communication routines of the Global Address Space Programming Interface (GASPI).

We work at fixed lattice size, so only the strong scaling is relevant for our purposes. On the left panel of Figure 6 we show the performance of the multishift conjugate gradient inverter, which runs 90% of the time in the simulation. The numbers are obtained on JUWELS Booster, where we divided the lattice among the MPI processes to minimize the communication surface.

The local lattice size is large enough, so that the communication should not be an issue at least on the smaller partitions. This is also demonstrated by the fact that the performance numbers decrease very slowly with increasing the partition size, see the left panel of Figure 6.

Leonardo Booster nodes have half the number of network cards of that of the JUWELS Booster nodes. Therefore, we assume that the communication loss on Leonardo Booster is twice as large as on JUWELS Booster. This is how we obtained the numbers corresponding to Leonardo on Figure 6, denoted by dashed lines.

We also performed measurements on JUWELS, where we divided the lattice in a way to double the communication surface compared to the minimal case. In the case of the 384 and 512 GPU runs we saw a drop in performance below 1%. These are better than our estimates for the communication loss above.

6. Conclusions

This long term project was started nine years ago with the target to compute the leading order hadronic vacuum polarization contribution to the anomalous magnetic moment of the muon, $a_\mu^{\text{LO-HVP}}$. The project had two important milestones.

At the first milestone we were able to reach a relative precision of 2.7% and obtained the result

$$a_\mu^{\text{LO-HVP}} = 711.0(7.5)(17.3)[18.9] \tag{1}$$

with statistical, systematic and total errors. This was published in Physical Review Letters [19] and the article was highlighted as an Editors' Suggestion.

At the second milestone we were able to improve the precision by several means and obtained the result

$$a_\mu^{\mathrm{LO-HVP}} = 707.5(2.3)(5.0)[5.5]\ . \tag{2}$$

The relative uncertainty is 0.8%, which is still far less than that of other lattice determinations, and is comparable to the errors of R-ratio based computations. The result was published in Nature [2] and was selected with the experimental measurement as one of the 10 ”Breakthroughs of the Year 2021” in all science by the journal Science.

Lattice field theory has now important challenges ahead. We have to explain the difference between the R-ratio method and our lattice determination. This is obviously necessary before drawing any conclusion about physics beyond the Standard Model in the muon magnetic moment. This will be our third milestone. We also have to increase our precision by a factor of four to reach the precision of the future Fermilab experiment, which will be the fourth and final milestone of this project.

7. Acknowledgements

The computations were performed on HAWK at the High Performance Computing Center in Stuttgart, on SUPERMUC at the Leibniz Rechenzentrum in Munich, on JUWELS at Forschungszentrum Juelich and IRENE at Commissariat à l’Energie Atomique et aux énergies alternatives (CEA), France. We thank the Gauss Centre for Supercomputing, PRACE and GENCI (grant 52275) for awarding us computer time on these machines. We also acknowledge EuroHPC Joint Undertaking for awarding this project access to the EuroHPC supercomputer Leonardo hosted by CINECA in Bologna, Italy.

References

[1] B. Abi, et al., Measurement of the Positive Muon Anomalous Magnetic Moment to 0.46 ppm, Phys. Rev. Lett. 126 (14) (2021) 141801. arXiv:2104.03281, doi:10.1103/PhysRevLett.126.141801.

[2] S. Borsanyi, et al., Leading hadronic contribution to the muon magnetic moment from lattice QCD, Nature 593 (7857) (2021) 51–55. arXiv:2002.12347, doi:10.1038/s41586-021-03418-1.

[3] D. P. Aguillard, et al., Measurement of the Positive Muon Anomalous Magnetic Moment to 0.20 ppm, Phys. Rev. Lett. 131 (16) (2023) 161802. arXiv:2308.06230, doi:10.1103/PhysRevLett.131.161802.

[4] G. W. Bennett, et al., Final Report of the Muon E821 Anomalous Magnetic Moment Measurement at BNL, Phys. Rev. D 73 (2006) 072003. arXiv:hep-ex/0602035, doi:10.1103/PhysRevD.73.072003.

[5] T. Aoyama, et al., The anomalous magnetic moment of the muon in the Standard Model, Phys. Rept. 887 (2020) 1–166. arXiv:2006.04822, doi:10.1016/j.physrep.2020.07.006.

[6] F. V. Ignatov, et al., Measurement of the $e^+e^- \to \pi^+\pi^-$ cross section from threshold to 1.2 GeV with the CMD-3 detector, Phys. Rev. D 109 (11) (2024) 112002. arXiv:2302.08834, doi:10.1103/PhysRevD.109.112002.

[7] M. Abe, et al., A New Approach for Measuring the Muon Anomalous Magnetic Moment and Electric Dipole Moment, PTEP 2019 (5) (2019) 053C02. arXiv:1901.03047, doi:10.1093/ptep/ptz030.

[8] T. Blum, P. A. Boyle, V. Gülpers, T. Izubuchi, L. Jin, C. Jung, A. Jüttner, C. Lehner, A. Portelli, J. T. Tsang, Calculation of the hadronic vacuum polarization contribution to the muon anomalous magnetic moment, Phys. Rev. Lett. 121 (2) (2018) 022003. arXiv:1801.07224, doi:10.1103/PhysRevLett.121.022003.

[9] G. Colangelo, A. X. El-Khadra, M. Hoferichter, A. Keshavarzi, C. Lehner, P. Stoffer, T. Teubner, Data-driven evaluations of Euclidean windows to scrutinize hadronic vacuum polarization, Phys. Lett. B 833 (2022) 137313. arXiv:2205.12963, doi:10.1016/j.physletb.2022.137313.

[10] M. Davier, Z. Fodor, A. Gerardin, L. Lellouch, B. Malaescu, F. M. Stokes, K. K. Szabo, B. C. Toth, L. Varnhorst, Z. Zhang, Hadronic vacuum polarization: comparing lattice QCD and data-driven results in systematically improvable ways (8 2023). arXiv:2308.04221.

[11] G. Benton, D. Boito, M. Golterman, A. Keshavarzi, K. Maltman, S. Peris, Data-driven estimates for light-quark-connected and strange-plus-disconnected hadronic g-2 window quantities, Phys. Rev. D 109 (3) (2024) 036010. arXiv:2311.09523, doi:10.1103/PhysRevD.109.036010.

[12] A. Bazavov, et al., Light-quark connected intermediate-window contributions to the muon g-2 hadronic vacuum polarization from lattice QCD, Phys. Rev. D 107 (11) (2023) 114514. arXiv:2301.08274, doi:10.1103/PhysRevD.107.114514.

[13] T. Blum, et al., Update of Euclidean windows of the hadronic vacuum polarization, Phys. Rev. D 108 (5) (2023) 054507. arXiv:2301.08696, doi:10.1103/PhysRevD.108.054507.

[14] C. Alexandrou, et al., Lattice calculation of the short and intermediate time-distance hadronic vacuum polarization contributions to the muon magnetic moment using twisted-mass fermions, Phys. Rev. D 107 (7) (2023) 074506. arXiv:2206.15084, doi:10.1103/PhysRevD.107.074506.

[15] M. Cè, et al., Window observable for the hadronic vacuum polarization contribution to the muon g-2 from lattice QCD, Phys. Rev. D 106 (11) (2022) 114502. arXiv:2206.06582, doi:10.1103/PhysRevD.106.114502.

[16] C. Aubin, T. Blum, M. Golterman, S. Peris, Muon anomalous magnetic moment with staggered fermions: Is the lattice spacing small enough?, Phys. Rev. D 106 (5) (2022) 054503. arXiv:2204.12256, doi:10.1103/PhysRevD.106.054503.

[17] G. Wang, T. Draper, K.-F. Liu, Y.-B. Yang, Muon g-2 with overlap valence fermions, Phys. Rev. D 107 (3) (2023) 034513. arXiv:2204.01280, doi:10.1103/PhysRevD.107.034513.

[18] C. Lehner, A. S. Meyer, Consistency of hadronic vacuum polarization between lattice QCD and the R-ratio, Phys. Rev. D 101 (2020) 074515. arXiv:2003.04177, doi:10.1103/PhysRevD.101.074515.

[19] S. Borsanyi, et al., Hadronic vacuum polarization contribution to the anomalous magnetic moments of leptons from first principles, Phys. Rev. Lett. 121 (2) (2018) 022002. arXiv:1711.04980, doi:10.1103/PhysRevLett.121.022002.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 99–108

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

Digital twins, the journey of an operational weather system into the heart of Destination Earth

Thomas Geenen[a], Nils Wedi[a], Sebastian Milinski[a], Ioan Hadade[a], Balthasar Reuter[a], Simon Smart[a], James Hawkes[a], Emma Kuwertz[a], Tiago Quintino[a], Emanuele Danovaro[a] Domokos Sarmany[a], Razvan Aguridan[a], Pedro Maciel[a], Martin Suttie[a], Cristina Duma[a], Matthew Griffith[a], Paul Burton[a], Andrew Bennet[a], Tryggvi Horvjar[a], Bentorey Hernandez[a] Cruz[a], Johannes Bulin[a], Michael Lange[a], Ahmad Nawab[a], Mathilde Leuridan[a], Samet Demir[a], Antonino Bonanni[a], Nicolau Manubens[a], Phillip Geier[a], Christopher Bradley[a], Adam Warde[a], Mirco Valentini[a], Eugen Betke[a], Marcos Bento[a], Michael Sleigh[a] , Andreas Müller[a], Willem Deconinck[a], Olivier Marsden[a], Michael Staneker[a], Fatemeh Pouyan[a], Slavko Brdar[a], Zbigniew Piotrowski[a], Samuel Hatfield[a]

[a] *European Centre for Medium-Range Weather Forecasts (ECMWF), Shinfield Park, Reading RG2 9AX, United Kingdom*

Abstract

Moving a world leading numerical weather prediction system that runs on a dedicated, bespoke, high performance computing cluster and supporting infrastructure, into the heart of a digital twin for climate change adaptation and extreme weather events has been a challenging and exciting journey. In this paper we describe this journey with a focus on those aspects required to leverage the pre-exascale EuroHPC systems that have been made available to Destination Earth (DestinE) to run its computational representation[1]. EuroHPC systems can be effectively used for DestinE and are in fact key assets to deliver the computational power required for Earth system digital twins at global km-scale resolution. At the same time, EuroHPC systems were newly installed and procedures to run them efficiently are evolving. We find that each of these systems is operated by a national hosting entity that implements its own procedures, e.g. for identity and access management, specific system configuration like schedulers, filesystems, software management systems, and specific, sometimes vendor associated, toolchains, tooling, and container runtimes. In particular, the different scheduling policies encountered, required us to adapt our workflows for each site. We found that having dedicated resources available, which was trialed in a period from 16th February to 14th April on LUMI, allowed to achieve high occupation rates, with 92% on the reserved GPU allocation and greater than 97% efficiency on the CPU reservation. Also a stronger focus on federation of these systems, with a focus not only on federation of identities and accounts, but also in the areas of data ownership/transfer, observability, services and service accounts, maintenance coordination and performance portability is required. In general, it

should become much easier to transfer a workload, or a digital twin system, from one EuroHPC site to the next and run and maintain them across several sites concurrently.

Keywords: Type your keywords here, separated by semicolons ;

1. Introduction

DestinE is a flagship initiative of the European Commission to develop a highly accurate digital representation of the Earth (a digital twin of the Earth) to model, monitor and simulate natural phenomena, hazards, and the related human activities. Through a strategic allocation from EuroHPC JU, DestinE has been granted access to its pre-exascale systems, which now form the computational backbone of the DestinE system. Moving from a dedicated system to distributed EuroHPC systems has been a challenge. ECMWF's operations are designed to deliver medium and extended-range ensemble forecasts, supporting climate services and delivering meteorological products to ECMWF's 35 Member and Cooperating States, as well as to customers worldwide. By contrast, EuroHPC systems are designed and operated to support a large and heterogeneous science community. As a result, EuroHPC systems are lacking the service support levels to deliver an operational product reliably, but also the ability to quickly (re)dedicate system resources to deliver products on time, for instance through urgent computing mechanisms. This paper has a strong focus on the technical challenges of deploying DestinE on multiple EuroHPC systems. The main challenges can be broken down in the following components: porting to novel and emerging computer architectures, deploying a services infrastructure, handling the large volumes of data that are produced by the simulation codes, and combining these aspects to produce a service that serves information for downstream consumption.

DestinE has the ambition to run earth system models at unprecedented resolution and fidelity. But before we touch on the technical challenges of deploying DestinE on EuroHPC systems, we describe the main Digital twins that are being developed [2,3]. The first Digital twin is for Climate Change Adaptation; it delivers a substantial evolution of existing climate projection capabilities at multi-decadal timescales. The main breakthroughs lie in the operationalization and the regular production of high-quality climate information, and the streaming of this information to applications from important impact-sectors like forestry, urban environments, hydrology, hydro-meteorology, and energy, and in developing further interactivity elements in particular in support of performing 'what-if' scenarios. The DT will operationalize Europe's next generation of storm and eddy-resolving Earth-system

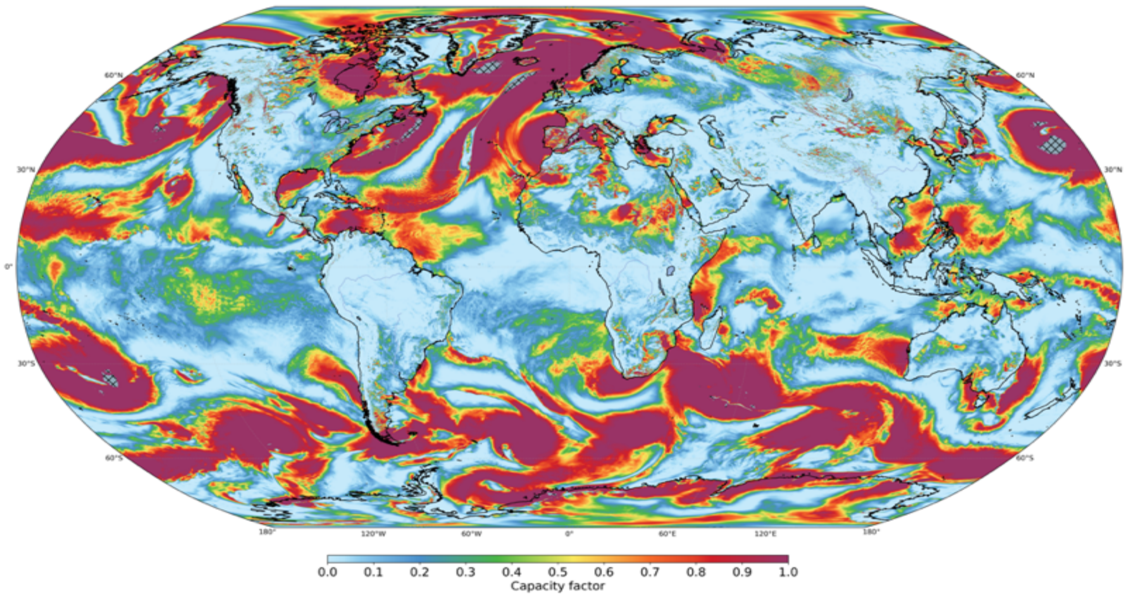


Fig. 1 Global map at a horizontal resolution of 5 km and 1-hourly frequency for 20/01/2020. It shows the capacity factor for a class S wind turbine (Vestas V164 - 9.5 MW). Data was obtained from the IFS-FESOM model. Credit: BSC

models, including an observation-based monitoring framework that can support model improvement. It will provide globally consistent and co-located climate, weather, and impact-sector information at much higher data output rates (5 to 10 km globally, hourly to monthly) than presently available for different emission scenarios for the next few decades. Fig. 1 illustrates a use case where a global climate model run produces wind speeds and associated capacity factors for wind energy generation.

The second Digital twin is for extreme weather events and is initialized using the best existing estimate of the atmosphere, land and ocean state created by optimally fusing simulations and millions of Earth system observations (e.g., the analyses produced operationally at ECMWF with the IFS for initializing its medium-range weather forecasts). These simulations have been run at km-scale (4.4 and 2.8 km) resolutions Fig. 2 for the atmosphere and

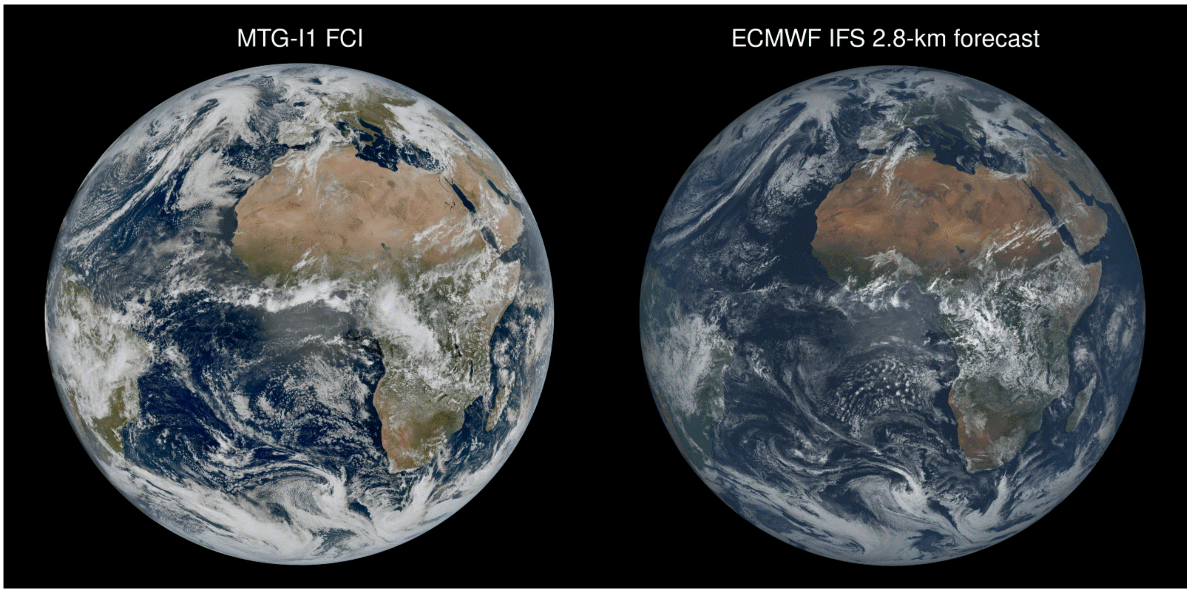


Fig. 2 The first official MTG-FCI visible image from EUMETSAT (left) and 12-hour simulation with ECMWF’s Integrated Forecasting System (IFS) at 2.8 km resolution (right), valid for 18 March 2023 at 12:00 UTC. Credit: EUMETSAT/ECMWF

land, and initially coupled to the wave, ocean and sea ice models at coarser resolutions. The length of these first simulations ranged between 2 and 5 days. This digital twin represents the first time ECMWF has run in a quasi-operational manner (as opposed to research runs) on an HPC facility not owned and operated by ECMWF, and the first time such a configuration has been run using GPUs (see section 3.1).

DestinE pushes the computational boundaries of current HPC systems and requires more computational resources than can be made available on a single EuroHPC system to a single initiative. Therefore, DestinE has been granted computation time on all the available pre-exascale EuroHPC systems. This however requires porting DestinE applications and services to each of them. ECMWF has an extensive track record in porting its main applications to new architectures [4] and managed to switch its whole simulation model from double to single precision in a very short timeframe[5]. A key ingredient in the ongoing adaptation strategy is the use of source-to-source translation[6] to allow transformation of a single source base to a compiler optimizable format, that can be compiled and run efficiently on several hardware architectures. This approach allows for the transformation of a code that is under continuous development, making sure that the latest features are optimized as soon as they are merged with the main code branch. However, for each new architecture or programming model, recipes must be adapted, and vendor or platform specific compilers and runtimes investigated. This is labor intensive and was not designed or foreseen to deal with the rapid exposure to new architectures that we see now with the addition of new EuroHPC systems every six months on average, each with its own specific architecture.

For DestinE we are developing and providing a set of components and services that we collectively call the Digital Twin Engine (DTE). These components are mainly focused on data handling and simulation model interactivity. Most of these services are deployed on a cloud stack that DestinE deploys next to the EuroHPC systems, but some of them need to interact with the EuroHPC system as well. We found that having a cloud stack available and managed by DestinE partners greatly facilitated the deployment and operation of the services. The interface between these services and operations on the HPC system turned out to require a bespoke solution for each individual hosting site and resulted in an expanding code base and maintainability challenges. So far, we managed to find a solution for each site that provides us with the functionality required.

The data volumes anticipated in the lifetime of DestinE are expected to be of the order of 1 PB per day per EuroHPC hosting site. This data volume is a significant amount to be written to the HPC parallel filesystems, but the systems have never been designed to support moving this amount of data outside of the system daily. We are developing a streaming concept to lower the impact on both the internal parallel filesystem, as well as the network connections to external systems. The developed data handling services will support a progressive data reduction from where the data is generated, in stages up to when the data is stored long term. They also include support for streaming the data to downstream applications without storing it at full resolution and/or fidelity for offline processing.

Making the data, produced by the simulation model in the digital twin (DT), available as information requires a bespoke data processing pipeline that allows for detailed, semantic, data extraction and transformation to ultimately feed into an interpretation layer, for instance an immersive visualization or augmented reality setup. The Digital Twin Engine provides the data handling backend that can operate on the large volumes of data and extract specific subsets close to where the data is stored and stream the outcome directly into remote running applications or Jupyter notebooks for further interactive analysis.

2. Porting to EuroHPC systems

2.1. The ECMWF strategy to port IFS to new computational architectures

At ECMWF, we developed a strategy to incrementally adapt the operational weather forecasting model to new architectures alongside ongoing other technical and scientific developments. This is built upon four core principles: (i) encapsulation with clean library interfaces to hide technical complexities, (ii) flexible data structures to provide abstractions for complex memory hierarchies, (iii) an increasingly flexible control flow to support different parallelisation paradigms, and (iv) the use of source-to-source translation to encode and apply hardware-specific optimizations of compute kernels.

As an example, the spectral transforms library, ecTrans, provides CPU and GPU backends behind a common interface. The first step in this work is often an extraction of a component into a standalone library and executable. These so-called mini-apps or dwarfs [7] can then be analysed to trial and implement sustainable optimisation strategies. For the bespoke IFS data layout, a FIELD API library[8] has been developed as a custom software abstraction with support for efficient host-device memory transfers. Longer term, it is foreseen that this functionality is delegated to the Atlas library[9], facilitating the use of its powerful parallel algorithms and improved interoperability. For frequently changed code parts, such as the physical parameterizations, a bespoke source-to-source translation tool, Loki [6], is being developed. Loki allows HPC experts to encode adaptation recipes to change the vectorized, CPU-optimized Fortran source into GPU-optimised code at build time. With this, hardware-specific optimization is separated from algorithmic and science developments, with the ability to modify and extend recipes for future hardware architectures. All libraries and tools are openly developed and made available on GitHub under an Apache-2 licence.

2.2. Porting the computational representation, IFS, to EuroHPC systems

IFS has been ported to all EuroHPC systems available to us, namely LUMI-C, LUMI-G, LEONARDO-BOOSTER. LUMI is a Cray system with AMD CPUs and AMD GPUs [10]. Leonardo is a Eviden system with Intel CPUs and Nvidia GPUs[11].The first step in porting the computational representation of the digital twin, the

ECMWF Integrated forecast system (IFS), to LUMI was to get it running on the CPU partition. This required overcoming a significant number of problems related to the compiler environment and the adjacent software stack as well as instabilities of the system. Regular contact with the LUMI support team and direct communication with HPE-Cray and AMD allowed us to resolve these or find workarounds to enable successful coupled forecast runs with IFS. As more users began to use LUMI, and in particular the CPU-only partition LUMI-C, queuing times became prohibitive to achieve any meaningful development or production throughput, at which point focus was shifted entirely to LUMI-G, which was possible thanks to the parallel progress in utilizing the AMD MI250X GPU architecture. The spectral transforms library, ecTrans, as the computationally most expensive part of IFS, has been adapted for offloading to AMD GPUs. Subsequently, this version of ecTrans was integrated into IFS and tested on LUMI-G for the Global Extremes and Climate Digital Twin configurations. Despite these issues, coupled IFS forecasts demonstrate the ability to scale to large node counts on LUMI-G Fig. 3. demonstrates this up to 512 nodes, i.e., 4096 MPI tasks.

Compared to LUMI-G, the host-side CPUs in LEONARDO are less powerful (a single low power, low core count CPU), which incurs a significant performance impact on the computational kernels that are not yet GPU-offloaded. With the stepwise integration of more GPU-enabled components it is expected that notable performance improvements can be seen within the next year.

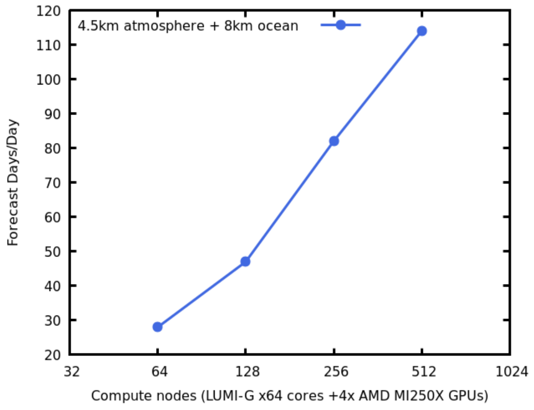


Fig. 3 Scalability study for 4.5km atmosphere coupled to 1/12 degree (10 km at equator) ocean on LUMI-? We use 8 MPI tasks per node and 7

Compared with the version for NVIDIA GPUs, the Cray compiler required many minor modifications in the OpenACC directives, and it requires HIP code (calling hipblas and hipfft) instead of the CUDA code (calling cublas and cufft). In principle, moving between CUDA and HIP is just a matter of replacing function names. After solving these problems, we now have one version of ecTrans that works equally well on both NVIDIA GPUs (with CUDA) and AMD GPUs (with HIP).

The optimization insights from the Climate DT have been transferred to the Continuous Global Extremes Digital Twin. Having run continuously on LUMI-G for the last months, it is now being transferred to LUMI-C to exploit the improved throughput on that partition also.

2.3. The role of containers in hardening the deployment

EuroHPC systems perform system upgrades on a regular basis, and we experienced that such system upgrades typically also involve the upgrade of the compilers, toolchains and drivers. In practice we need significant effort to get the model running performantly again after each upgrade. Running the application in a container can shield us from system changes to a certain extent[12]. Martinasso et al.. showed that they can successfully run an older version of the ICON simulation model inside a container with an upgraded version of the system software and toolchains. Also, for testing purposed they could run a version compiled against a newer version of the system software and toolchains from a container on a system with an older version of the system software and toolchains. These findings give us confidence that we can build a certain level of resilience against system changes when the simulation software runs inside a container. In fact, Martinasso etal managed to run a two-year-old version of the container image successfully on a recent system configuration.

3. Data handling and management

3.1. Generating the data stream

The data production of the digital twin for climate change adaptation comes from two main simulation codes. These produce their own specific native format which is passed into a common IO pipeline and processing system which can transform the output into a generic format that can be used for downstream processing or for direct consumption by downstream applications. The principal data pathways through the EuroHPC and ancillary systems deployed by ECMWF in Destination Earth are shown in Fig. 4.

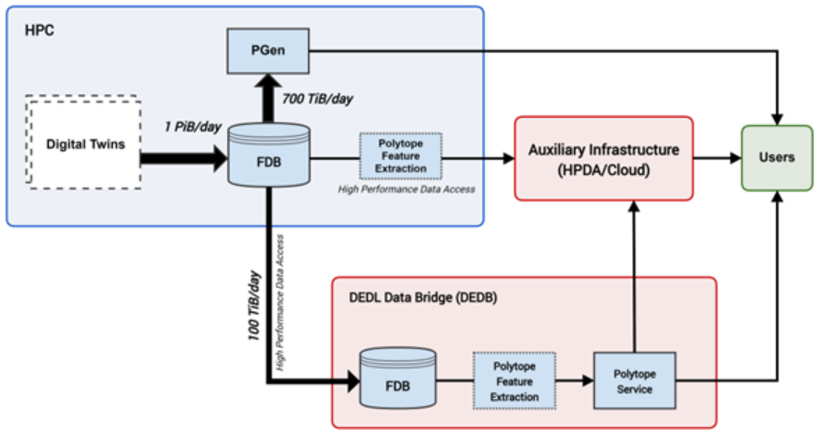

HPC
PGen
700 TiB/day
1 PiB/day
Digital Twins
FDB
Polytope Feature Extraction
High Performance Data Access
Auxiliary Infrastructure (HPDA/Cloud)
Users
100 TiB/day
High Performance Data Access
DEDL Data Bridge (DEDB)
FDB
Polytope Feature Extraction
Polytope Service

Fig. 4 In blue are components residing within the EuroHPC systems, green symbolizes external user interactions, and in red are services running in PaaS or IaaS cloud partitions. The software in the blue small boxes (PGen, FDB, Polytope) is part of the DTE deployed by ECMWF

3.2. Storing the simulated data in a domain specific object store, the FDB

Data that is generated by the simulation model is stored initially on the parallel filesystem of the EuroHPC system. DTE components are then able to, in stages, transfer and reduce a subset of this data to the DestinE data bridge, as shown in Fig. 4. To optimize data access and to be able to expose the data via an API rather that a path to a file, ECMWF has developed a domain specific object store (the FDB) that is used to store simulation model outputs from both models. The FDB deployment consist of several components; a library that controls how data is stored and accessed from disk-based infrastructure and can be built into models and other tooling; the configurations that control how those resources are used; a deployed distributed service for storing large volumes of data on the databridge; and an FDB client-server system which is able to make the data service on the databridge available.

For longer term storage of data, a distributed system is deployed on dedicated nodes running on the databridge. FDB servers running on the databridge make this data available to clients inside the EuroHPC systems and to Polytope for serving data to the outside world. An FDB server running on a dedicated node in the EuroHPC system in turn makes data available from the in-HPC FDB to the databridge. This allows for data exchange between the FDB instances and services and facilitates serving data externally.

3.3. Accessing data in a semantic way

Within the DTE ecosystem, and most significantly when data is stored in the FDB, each data object is uniquely identified by a semantically and scientifically meaningful key itself made up of a set of key-value pairs describing the data. A query may represent, for instance; give me the 2-meter temperature over Italy for today at 11:00 CET. These kind of queries makes it much more intuitive and less error prone to retrieve data, in comparison to using an opaque and arbitrary key, and allow for discovery services to translate meaningful data descriptions directly to the end users

An additional service, Polytope, has been deployed which is able to serve data from the FDB to external users over the internet, and comes with its own python library for smooth integration into downstream tools and applications, or interactive environments like jupyter.

3.4. Integration into an end-to-end service for DestinE

In the context of DestinE the above set of tools needs to be deployed across several platforms and connected via APIs. Specifically, a user that is logged into the central DestinE platform will have direct access to a polytope client interface that will allow for digital twin data retrieval directly from the databridge or even from the EuroHPC system, while respecting the roles and data authorization that has been granted on the central platform.

4. Deploying and operating IT-services

Digital twins, as has become apparent from the above description, require a rich set of services to allow users to interact with the data, the information and the simulation models themselves. These services are distributed over several system components that are themselves geographically distributed to be collocated with the different EuroHPC systems that are used for DestinE Fig. 5.

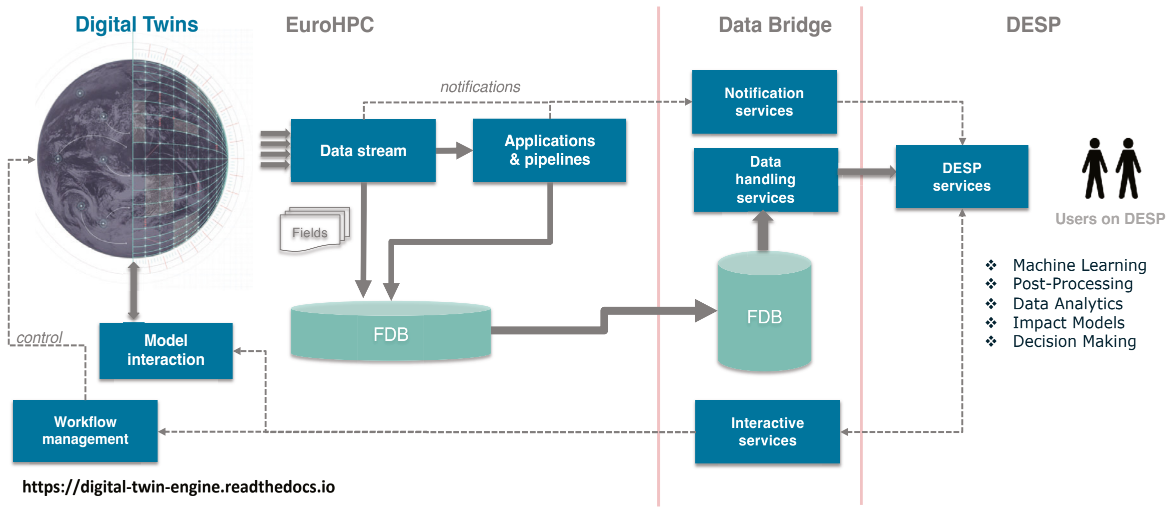


Fig. 5 The high level service architecture of the DTE illustrates the richness of the distributes services landscape across the DestinE system

4.1. Deploying services on EuroHPC systems

The deployment of services on EuroHPC systems comes with several challenges. Most EuroHPC sites are not set up to host these services and special arrangements had to be made on a site-by-site basis to allow for a service to be permanently up, and exposing an API that can be consumed by other services and users both on the system itself but also external to the system. The two main design patterns that start to emerge are the deployment of special service nodes that either run services directly under the control of systemd, but require therefore the support from the systems operational teams, or deploy a VM on these service nodes that can be managed (internally on the VM) by the DestinE developers and operational teams to deploy services. The latter approach seems to allow for more flexibility and allows the management of the system accounts under which the services are run by the DestinE teams. Obviously, a mapping or federation of service accounts to system users or system accounts on the EuroHPC must still be established, but this setup allows for a higher level of isolation that can decrease the attack surface and prevent data and information leakage from the system.

4.2. Interfaces between services on the DestinE databridge and the EuroHPC systems

In the context of DestinE we must establish a connection between the services running on the EuroHPC and services running on the databridge. Most EuroHPC sites are not setup for this scenario and typically only allow ssh access by known users to the login nodes of the system. In the context of DestinE, we negotiated access from the databridge directly to the services that we deployed as described in the above subsection over a dedicated set of ports. Unfortunately, we did not manage yet to agree on a common approach between all sites resulting in a diverging code base and associated operational and maintainability challenges. In addition, the introduction of two factor authentication did not improve the situation and special, site specific, arrangements must be put in place to respect the implemented security measures by each site.

5. Operational challenges

Operating a complex climate adaptation and weather induced extremes digital twin system across different systems that are each geographically distributed requires a tight integration and a high-level of visibility of the underlying systems and systems. In the context of the EuroHPC systems this means a strong federation implementation and access to monitoring and logging. This allows for in-depth observability of the system to understand where issues may arise and take preventive or reactive measures. In addition, there is a need for an operational urgent computing framework that allows for swift access to dedicated resources under well-defined conditions that warrant exclusive and imminent use of EuroHPC resources.

5.1. Federation

DestinE requires the implementation of a broad interpretation of a federation concept to allow its operation over a distributed set of EuroHPC systems. During the usage of EuroHPC systems up till now we have been challenged by the lack of federation capabilities, but with the increasing number of EuroHPC systems becoming available for DestinE these issues become more and more prominent. The first challenge is obviously associated with the lack of federation of identities and roles. Hundreds of users had to be onboarded manually on each system and accounting and roles had to be defined and set up on each hosting site. We understand and appreciate that each hosting site needs to implement its own user registration and comply with national laws, but a level of federation should be added on top. The next major challenge is the lack of federation of data. Users who have access under a certain project on one system do not have the same access rights to the same data on another system when the data gets migrated between systems. This also holds for system accounts or the mapping between system accounts and user accounts that should be consistent and federated between sites. When it comes to federation, we also want to include the federation of software and service deployment mechanisms. Users or projects should be able to deploy in a consistent and transparent way between EuroHPC hosting sites. Also, federation of systems with respect to monitoring, logging, accounting and in general observability is needed. A consistent view is required to understand how the different components running on different systems are performing and what proactive or reactive measures should or could be taken to recover from anomalies. Federation between accounting systems is needed in the context of meta scheduling where, depending on data locality and system availability, workload should be steered towards the most appropriate system. There is also a need for federation in the context of urgent computing. If a service has been labelled as urgent it should be able to run on the most suitable EuroHPC system without any need for ad-hoc (re)configurations of the urgent computing service of any hosting site. Finally, there is also the need to federate SLAs, service levels should be able to migrate with the application or service from one EuroHPC system to the next.

5.2. Observability and its role in operations

To allow for proactive or reactive responses to system anomalies, DestinE needs to establish in-depth visibility on the underlying systems. For the EuroHPC systems that would mean access to system monitoring and logging as well as detailed accounting and job management information. This system information would allow for the support teams of DestinE to assess whether an issue is related to the simulation application of one of the supporting services or is caused by a failure of one of the system components of the EuroHPC system. If the latter is the case, or if we observe degradation in the system performance and not only in the application performance, then corrective measures can be taken to restore the normal operational state. Such measures could be to restart the application with more resources to allow for faster throughput, migrating the application to another part of the system that might not be impacted by the system anomaly. Or, as a last resort, migrate the application to another EuroHPC system to rerun the simulation model. In this case also some of the data associated with the simulation model must be migrated and the system must be made aware that the results are generated in a different physical location also. Access to accounting and job information is required to make sure DestinE can do adequate resource management and to prevent running out of computational or storage resources during model execution.

5.3. Urgent computing

Urgent computing is both a policy mechanism as well as a technical implementation that can free up computational resources for simulations that are needed to support rapid alerting and forecasting frameworks. Examples are typically associated with disaster management but can be generalized as computations whose results are necessary for decision making under a strict deadline. In Europe several projects have been run and demonstrators deployed and tested to define the procedures and mechanisms required that are now well understood[13–15] . In the context of DestinE there are several scenarios, for instance in the context of the Extremes digital twin, but also for the climate change adaptation digital twin where results are necessary for decision making under a strict deadline. We therefore argue that these mechanisms should be activated on EuroHPC systems and the DestinE system prepared to leverage these mechanisms when the criteria are met to execute them. In practice what this means is that the applications and the scenarios that should be run are prepared and a special queue should be in place for these scenarios to be executed without, or with limited, human intervention. On a regular basis these scenarios should be tested to make sure that they will execute without any issues when triggered.

5.4. Support levels for operational services

With DestinE moving towards an operational service, there is a need for increased levels of support from the usual 5x7 support levels towards extended support hours also covering the weekends. These support levels could be provided by EuroHPC hosting sites directly or by a subcontractor of these entities but could also be supported by DestinE or its contractors. In the latter case it is foreseen that certain scenarios will be run on multiple sites at the same time to make sure that results will be delivered within the required timeframe. Operators from DestinE will oversee the progress of the runs and take proactive or reactive measures to restore operations when an anomaly is detected. This requires access to low level system monitoring but does not require elevated privileges to restore the system when an issue arises.

6. Conclusion

We have shown that DestinE is leveraging EuroHPC systems and their capabilities efficiently and effectively for running the computational representation for its digital twins. The capabilities and functionalities that have been developed for DestinE, as part of the Digital Twin Engine, to run efficiently on these systems, allow us to move the data from the system onto the wider DestinE system. We have highlighted some potential for future developments that would allow us to integrate better between the EuroHPC systems and the rest of the DestinE system, as well as those capabilities that would allow us to move workloads, data, and services faster between systems. Finally, we highlighted some options for improved operational capabilities including urgent computing.

7. Acknowledgements

We acknowledge the EuroHPC Joint Undertaking for awarding this project access to the EuroHPC supercomputer LUMI, hosted by CSC in Kajaani through a EuroHPC JU Special Access call. We would like to thank the EuroHPC hosting sites for their commitment in getting the models up and running and our (contract) partners for the implementation of the digital twins. We are looking forward to continuing our fruitful collaboration in the coming phases of DestinE.

[1] Mckee D. Platform Stack Architectural Framework: An Introductory Guide A Digital Twin Consortium White Paper Platform Stack Architectural Framework: An Introductory Guide ii. A Digital Twin Consortium White Paper 2023.
[2] Hoffmann J, Bauer P, Sandu I, Wedi N, Geenen T, Thiemert D, et al. Destination Earth - A digital twin in support of climate services. CliSe 2023;30:100394. https://doi.org/10.1016/J.CLISER.2023.100394.
[3] Wedi N, Bauer P, Sandu I, Hoffmann J, Sheridan S, Cereceda R, et al. Destination Earth: High-Performance Computing for Weather and Climate. Comput Sci Eng 2022;24:29–37. https://doi.org/10.1109/MCSE.2023.3260519.
[4] Wedi N, Bauer P, Quintino T. From the Scalability Programme to Destination Earth. ECMWF Newsletter 2022:15–22. https://doi.org/10.21957/pb2vnp59ks.
[5] Váňa F, Düben P, Lang S, Palmer T, Leutbecher M, Salmond D, et al. Single Precision in Weather Forecasting Models: An Evaluation with the IFS. Mon Weather Rev 2017;145:495–502. https://doi.org/10.1175/MWR-D-16-0228.1.
[6] Targett JS, Luk W, Lange M, Marsden O. Systematically migrating an operational microphysics parameterisation to FPGA technology. 2021 IEEE 29th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2021, p. 69–77. https://doi.org/10.1109/FCCM51124.2021.00016.
[7] Mengaldo G. Batch 1: Definition of several Weather & Climate Dwarfs. ArXiv 2019;abs/1908.06089.
[8] ecmwf-ifs/field_api n.d. https://github.com/ecmwf-ifs/field_api (accessed May 31, 2024).
[9] Deconinck W, Bauer P, Diamantakis M, Hamrud M, Kühnlein C, Maciel P, et al. Atlas : A library for numerical weather prediction and climate modelling. Comput Phys Commun 2017;220:188–204. https://doi.org/10.1016/J.CPC.2017.07.006.
[10] LUMI’s full system architecture revealed - LUMI n.d. https://www.lumi-supercomputer.eu/lumis-full-system-architecture-revealed/ (accessed May 31, 2024).
[11] Leonardo HPC System | Leonardo Pre-exascale Supercomputer n.d. https://leonardo-supercomputer.cineca.eu/hpc-system/ (accessed May 31, 2024).
[12] Martinasso M, Gila M, Sawyer W, Sarmiento R, Peretti-Pezzi G, Karakasis V. Cray programming environments within containers on Cray XC systems. Concurr Comput 2020;32:e5543. https://doi.org/https://doi.org/10.1002/cpe.5543.
[13] Boukhanovsky A, Bubak M. High Performance Computations for Decision Support in Critical Situations: Introduction to the Third Workshop on Urgent Computing. Procedia Comput Sci 2014;29:1644–5. https://doi.org/https://doi.org/10.1016/j.procs.2014.05.149.
[14] Kupczyk M, Lawenda M. Whitepaper Urgent Computing Policy recommendations n.d. https://doi.org/10.5281/zenodo.10953116.
[15] Talia D, Trunfio P. Urgent Computing for Protecting People From Natural Disasters. Computer (Long Beach Calif) 2023;56:131–4. https://doi.org/10.1109/MC.2023.3241733.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 240 (2024) 109–114

Procedia Computer Science

www.elsevier.com/locate/procedia

Proceedings of the First EuroHPC user day

EuroHPC JU Infrastructures and Their Use in Science and Technology

1. EuroHPC JU: Current state of the Infrastructures

The mission of EuroHPC JU is to develop, deploy, extend and maintain an integrated world-class supercomputing and data infrastructure in the European Union and to develop and support a highly competitive and innovative High Performance Computing (HPC) ecosystem.
Since 2021, EuroHPC has been providing scientists, SMEs and industry located in Europe with access to operational HPC Systems.
In 2023, the following EuroHPC supercomputers were operational: LUMI in Finland, Leonardo in Italy, Vega in Slovenia, MeluXina in Luxembourg, Karolina in Czech Republic, and Discoverer in Bulgaria.

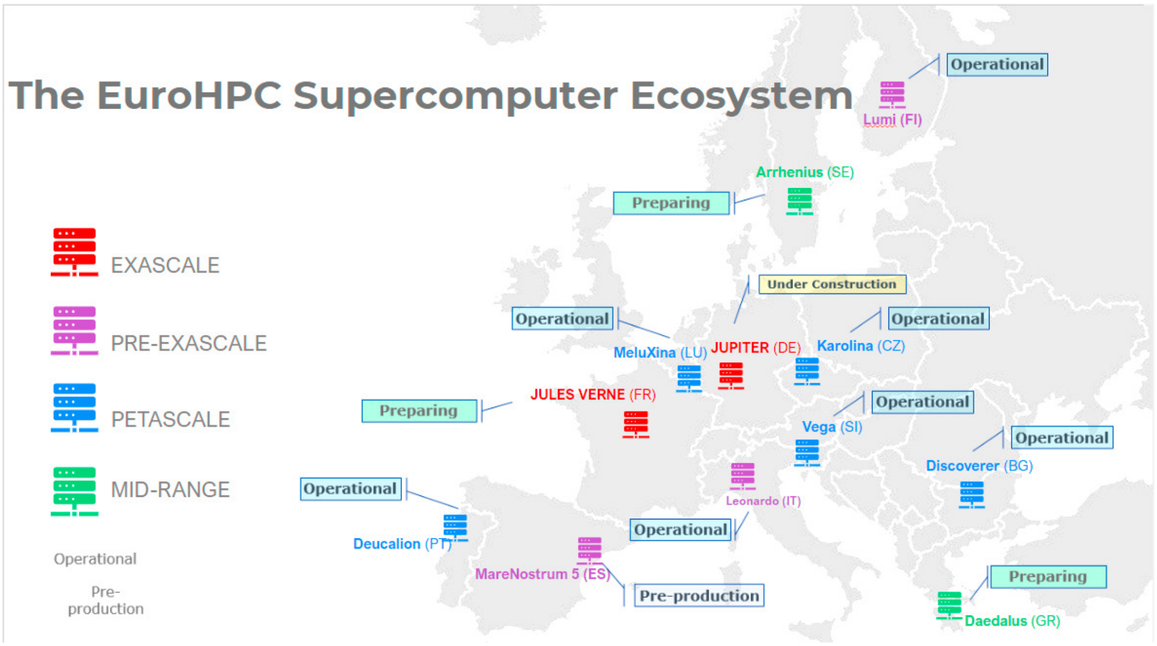
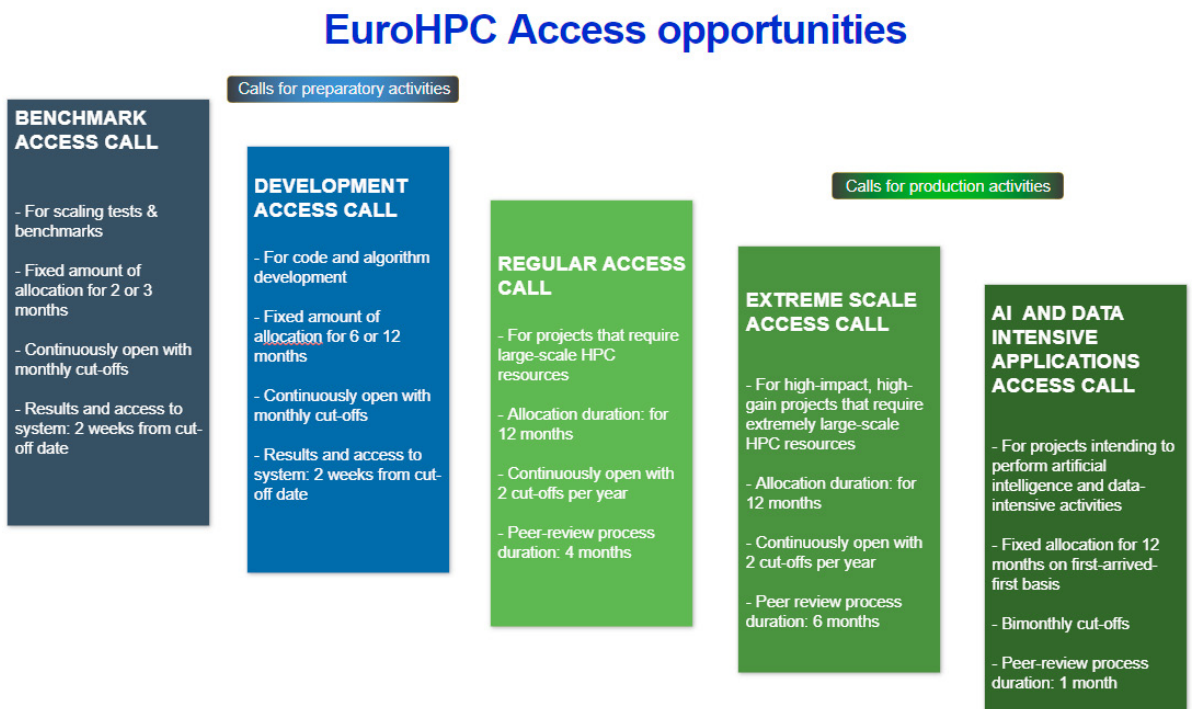


Figure 1: EuroHPC JU Supercomputers as of May 2024

10.1016/j.procs.2024.07.014

As of the beginning of 2024, the MareNostrum5 pre-exascale supercomputer in Spain has entered the pre-production phase, Jupiter, the first European Exascale system, is in the construction phase, and three more supercomputers are in preparation as depicted in Figure 1: the second European exascale system, to be hosted by the Jules-Verne consortium in France, Arrhenius, the mid-range system in Sweden, and Daedalus, another mid-range system in Greece.

There are five Access Modes, that is types of calls, for the EuroHPC supercomputers as shown in Figure 2, categorized according to several parameters such as the volume of resources offered, the complexity of the evaluation process that is applied, the type and maturity of applications targeted by each mode, and the periodicity of cut-off dates. A call for access involves an evaluation process. For Access Modes allocating large proportions of system resources, a Peer-Review evaluation is required to rank the applications based on the established evaluation criteria.


EuroHPC Access opportunities

Calls for preparatory activities

BENCHMARK ACCESS CALL

- For scaling tests & benchmarks
- Fixed amount of allocation for 2 or 3 months
- Continuously open with monthly cut-offs
- Results and access to system: 2 weeks from cut-off date

DEVELOPMENT ACCESS CALL

- For code and algorithm development
- Fixed amount of allocation for 6 or 12 months
- Continuously open with monthly cut-offs
- Results and access to system: 2 weeks from cut-off date

Calls for production activities

REGULAR ACCESS CALL

- For projects that require large-scale HPC resources
- Allocation duration: for 12 months
- Continuously open with 2 cut-offs per year
- Peer-review process duration: 4 months

EXTREME SCALE ACCESS CALL

- For high-impact, high-gain projects that require extremely large-scale HPC resources
- Allocation duration: for 12 months
- Continuously open with 2 cut-offs per year
- Peer review process duration: 6 months

AI AND DATA INTENSIVE APPLICATIONS ACCESS CALL

- For projects intending to perform artificial intelligence and data-intensive activities
- Fixed allocation for 12 months on first-arrived-first basis
- Bimonthly cut-offs
- Peer-review process duration: 1 month

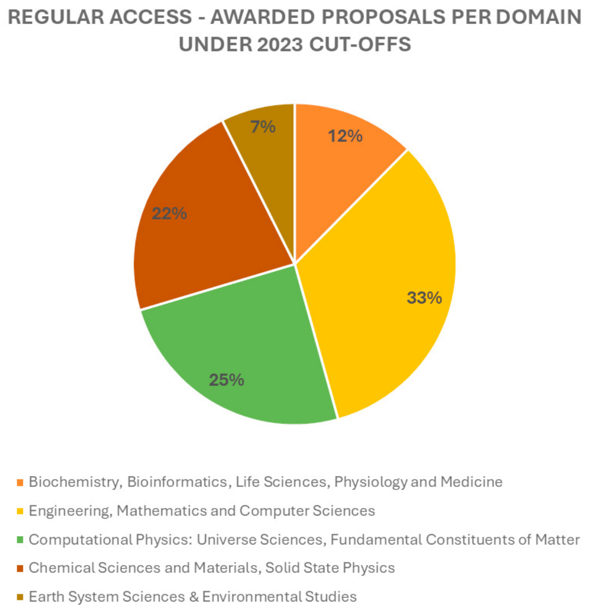
Figure 2: The Five Access Modes to EuroHPC JU Supercomputers

1.1. Benchmark and Development Access Modes

In 2023, over 400 projects were granted access to EuroHPC JU systems via Benchmark and Development calls. These calls allowed projects to test their software on different architectures of the system, improve them, benchmark and optimize them.

1.2. Regular and Extreme Access Modes

In 2023, 163 projects were submitted under the Regular and Extreme Scale Access calls and 96 were awarded access time on EuroHPC JU systems. These modes require the involvement of a large number of external experts which support the Access Resource Committee to conclude the final ranking of proposals based on the quality of the proposals. The result of this rigorous process is that a number of applications may be rejected due to achieving a lower ranking. This is because the JU must allocate resources in order of ranking, with the highest ranked proposals receiving resources first. Figure 3shows the division of these projects per scientific domain.



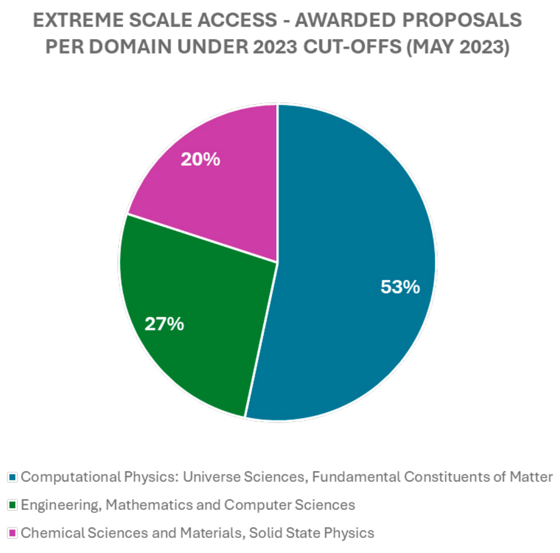


Figure 3: Division of 96 awarded projects form Regular and Extreme Access calls per scientific domain.

The proposals belong to various domains, including Biochemistry, Bioinformatics, Life Sciences, Physiology and Medicine, Chemical Sciences and Materials, Solid State Physics, Computational Physics: Universe Sciences, Fundamental Constituents of Matter and Engineering, Mathematics and Computer Sciences. EuroHPC JU has allocated 28,060,689 node hours and 2,210,009,322 core hours on the HPC systems to the successful projects in these two types of calls. Figure 4 and Figure 5 show the distribution of affiliation of main Principal Investigators per country for both Regular and Extreme access calls

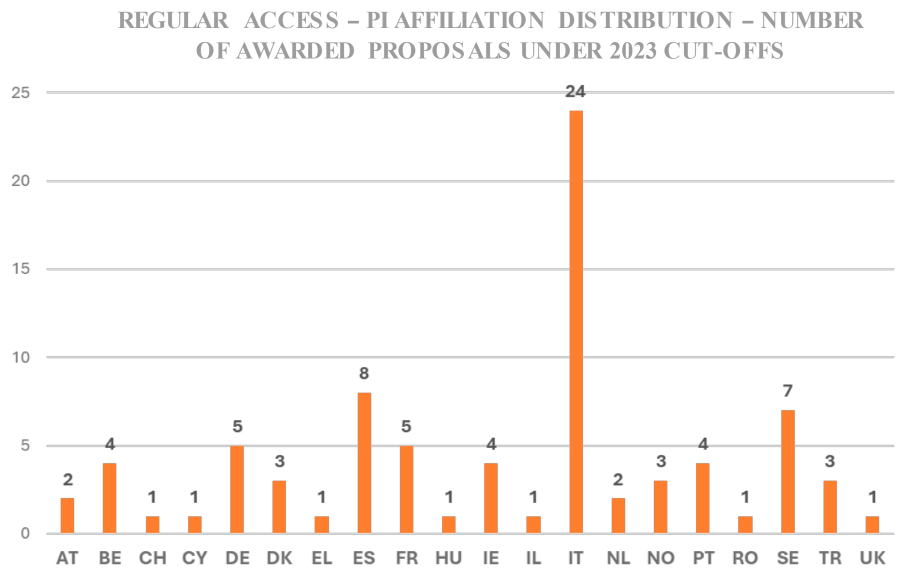


Figure 4: PI affiliation distribution of Regular access applications per country.

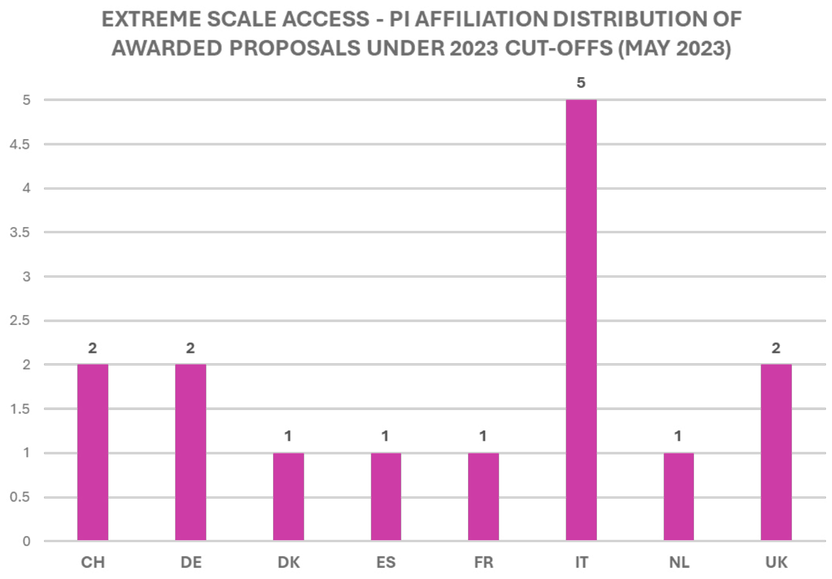


Figure 5: PI affiliation distribution of Extreme access applications per country.

1.3. AI and Data Intensive Access Modes

In 2024, EuroHPC JU launched a new access mode dedicated specifically to AI projects. More about this access mode will be detailed in our future books of proceedings.

1.4. Special Access Modes

EuroHPC JU can grant special access free of charge to strategic European Union initiatives considered to be essential for the public good, or in emergency and crisis management situations.

In 2022, the Destination Earth project, also known as DestinE, became the first such initiative to be granted Special Access. DestinE has used resources from LUMI, Leonardo, MareNostrum 5 and MeluXina EuroHPC supercomputers. The project aims to develop a highly accurate digital model of the complex Earth system – a digital twin (DT) – to monitor, simulate and predict environmental change and human impact to support more sustainable developments and support corresponding European policies supporting the European Green Deal.

2. Usage of systems by projects through EuroHPC JU Access calls

As shown in Figure 6, during the year 2023, 593 projects were given access to EuroHPC JU supercomputers coming from four different EuroHPC JU access call modes: Benchmark, Development, Regular and Extreme. Most of these projects were from Italy, Austria, Germany, France, Sweden and Spain.

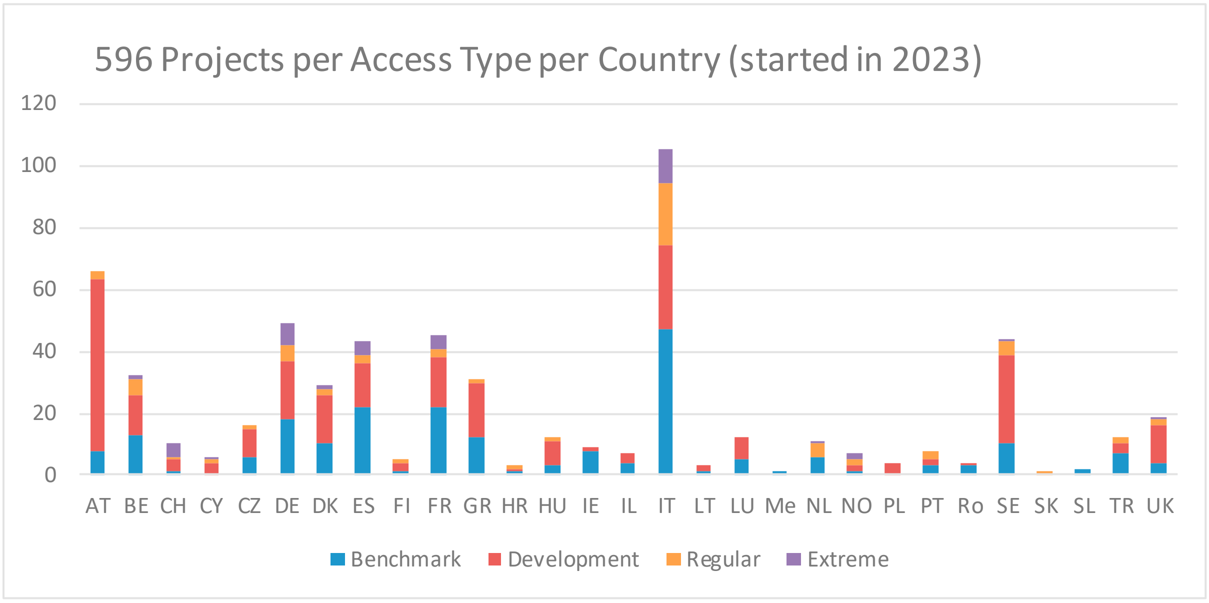


Figure 6: The active EuroHPC JU projects during the year 2023 on EuroHPC JU supercomputers.

Figure 7 shows the division of these active projects per type of Access mode. It should be noted that while the number of Regular and Extreme mode projects represents a small proportion of the projects, the amount of compute resources consumed by these projects is incomparably larger than the Benchmark and Development mode projects.

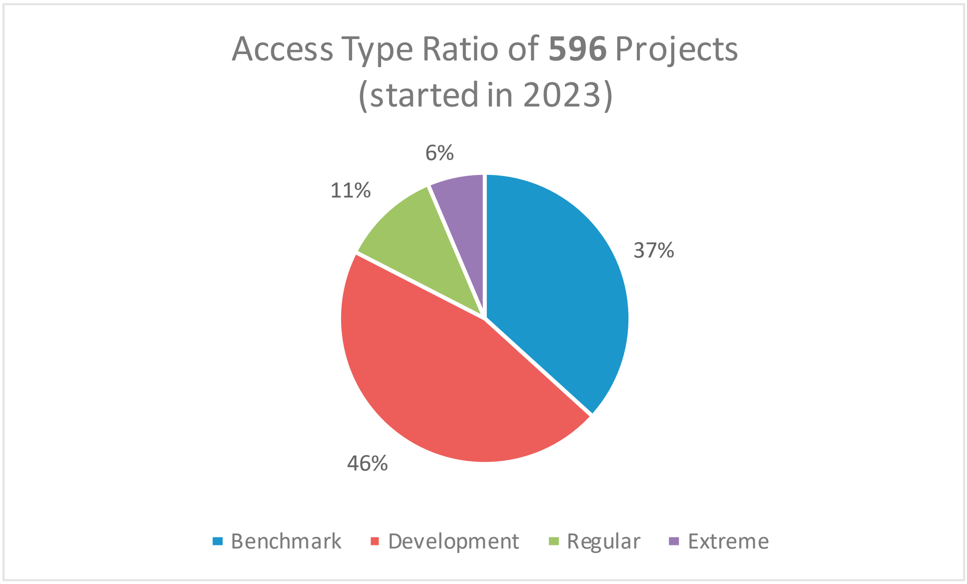


Figure 7: Distribution of 596 EuroHPC JU projects per types of access modes.

It should also be noted that that out of these projects, 167 were Artificial Intelligence (AI) projects and the distribution of these across countries is depicted in the Figure 8. The majority of these projects were from Austria and Sweden.

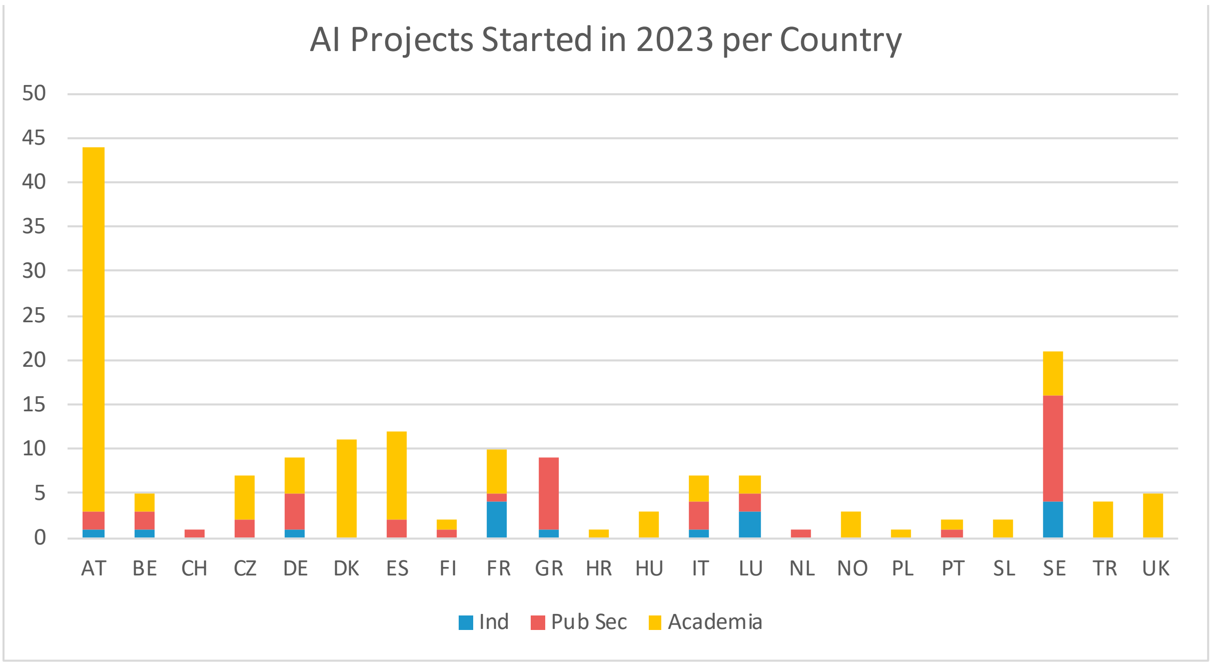


Figure 8: The 167 AI EuroHPC JU project active on EuroHPC JU systems during the year 2023.

3. Conclusion

To conclude, it is extremely exciting and rewarding to see such an immense uptake of the EuroHPC JU supercomputers for European science and innovation by academia, public and private sector in all domains and sectors. In this edition of EuroHPC Computer Science Proceedings, only a fraction of the excellent research examples that were presented during the EuroHPC JU User Day 2023 event are depicted.
In our future editions, we hope 163 projects were submitted under the Regular and Extreme Scale Access calls out of which 96 were awarded access time. present many more interesting examples of European research and innovation on EuroHPC JU supercomputers.