MISSIONE 4
ISTRUZIONE
RICERCA

SPOKE 1
FUTURE HPC & BIG DATA
FLAGSHIP 5:
Survey of of candidate prototypes on benchmarking tools and mini-applications

Codesign - application + SW + HW targeting, benchmarking, patterns, microkernels

Finanziato dall'Unione europea
NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

EXECUTIVE SUMMARY

The main aim of this flagship is the design, modeling and simulation of heterogeneous accelerators for HPC-cloud systems and edge servers. In this report we focus on one of the objectives that have been planned to achieve the main aim, namely the selection of mini-applications and benchmarking from multiple domains, such as: AI/ML, Big Data, fluid dynamics, multi-scale simulations, data analysis for astrophysics, N-body dynamics, social media network analysis, graph analytics multi-particle, long-range interacting systems; computational geometry; light-matter interaction, low dimensionality systems, quantum materials, numerical analysis, etc.), algorithmic prototyping, and algorithmic co-design: simulation, modeling, optimization. The objective is also related with codesign HW Development and exploitation of next generation of HPC systems: VLSI and FPGA-based architectures; CPUGPU algorithms, VLSI and FPGA-based architectures; data-driven parallelism, data affinity and data locality, streaming computation.

Flagship 5 is carried out by the following units: UNICT(Leader); UNIBO, UNITO, UNIPI, UNIPD, ROMA-TOV, UNINA, PoliMI, UNICAL, INAF, CINECA, ENEA, IIT, UNIFE.

In this report we provide a selection of candidate prototypes on benchmarking tools and mini-applications (Section 1). We also report a brief summary of the cooperation between the different units and with external research groups (Section 2), the deliverables (Section 3) and a reference list (Section 4).

For each subsection with one subindex, namely Section 1.x, we listed at the end of the title the units involved in that specific research. The units are not repeated for subsections involving more than one subindex, such as Section 1.x.y.

Contributors:

Sebastiano Battiato, Sebastiano Boscarino, Armando Coco, Patrizia Daniele, Orazio Muscato, Giovanni Nastasi, Alessandro Ortis, Vittorio Romano, Giovanni, Russo, Corrado Santoro, Laura Rosa Maria Scrimali, *Università di Catania* (UNICT)

Giulio Malenza, Marco Aldinucci, Marco Beccuti, Simone Pernice, *Università di Torino* (UNITO)

Fabio Durastante, Luca Gemignani, Beatrice Meini, *Università di Pisa* (UNIPI)

Daniele Bertaccini, Gianfranco Bocchinfuso, Andrea Clementi, Salvatore Filippone, Ugo Locatelli, Francesco Quaglia, Roberto Verzicco, *Università di Tor Vergata* (UNITOV)

Donato D'Ambrosio, William Spataro, *Università di Cagliari* (UNICAL)

Carlo de Falco, Aldo Ghisi, *Politecnico di Milano* (POLIMI)

Giulia Bertaglia, Walter Boscheri, Giacomo Dimarco, Cristiano Guidorzi, Elisa Iacomini, Fabio Schifano, Lorenzo Pareschi, Luca Tomassetti, *Università di Ferrara* (UNIFE)

Alberto Aloisio, Giovanni De Lellis, Guido Russo, *Università di Napoli* (UNINA)

Walter Rocchia, Sergio Decherchi, Cristian Ciracì, Pietro Vidossich, Massimiliano Pontil, *Istituto Italiano di Tecnologia* (IIT)

Eva Sciacca, Nicola Tuccari, Luca Tornatore, Vincenzo Antonuccio, Alessandro Costa, Gian Luigi Granato, Giuseppe Murante, Alberto Vecchiato, Valentina Cesare, Claudio Zanni. *Istituto Nazionale di Astrofisica* (INAF)

Gianfranco Bilardi, Carlo Fantozzi, Carlo Janna, Leonardo Pellegrina, Andrea Alberto Pietracaprina, Geppino Pucci, Michele Scquizzato, Francesco Silvestri, Fabio Vandin, *Università di Padova* (UNIPD)

Section 1: Selection of candidate prototypes on benchmarking tools and mini-applications

In FutureHPC, benchmarking tools can be used to evaluate the performance of different computing systems, including processors, accelerators, and interconnects.
In addition, mini-applications can be used to evaluate the performance of specific algorithms and architectures, such as finite element analysis, molecular dynamics, and graph analysis.
These benchmarks can help identify candidate prototypes that can deliver the best performance for specific applications and workloads.
In Big Data, benchmarking tools can be used to evaluate the performance of different data storage and retrieval systems, including distributed storage systems, NoSQL databases, and object stores.
In addition, mini-applications can be used to evaluate the performance of specific data processing and analysis algorithms and architectures, such as graph analysis and distributed computing.
These benchmarks can help identify candidate prototypes that can deliver the best performance and efficiency for specific data processing and analysis workloads.

Section 1.1: simulation and distribution of energy from renewable sources, supply chain models, real-time simulations, continuous input data flow, real-time aggregation of real distribution data (UNICT)

Among the mini-applications of interest, we want to analyse further the algorithms for solving supply chain models, also in the case of evolutionary data.

Another scientific activity carried out in the context of the PNRR HPC project was the design and development of new solutions for real-time simulations, with high performance and continuous input data flow, related to topics such as the simulation and distribution of energy from renewable sources, real-time aggregation of real distribution data, and their reuse to improve and optimize energy distribution, as well as a high-performance physical simulation system.

In relation to the first activity, a distributable multi-agent simulator was developed, capable of simulating multiple renewable energy production and distribution sites and managing large data flows from external hardware and within the simulations themselves. To enable the launch of multiple distributed simulation environments, an orchestrator was also developed to manage the real-time launch and synchronization of these distributed simulations. Within each simulated environment, there are Devices, agents capable of altering their own state and that of other agents depending on the configuration used. The environment is also equipped with a graphical interface through which the various simulations can be monitored and managed, and can be configured before launch. Additionally, the machines used for the distributed system can be managed through this interface. Furthermore, a real-time aggregator is being studied and designed to further manage the various simulations in real-time, considering a high volume of data coming from the real distribution network and simulating possible behaviours to achieve specific objectives. Finally, the development of a first version of a real-time physical simulator has been completed, mostly concerning a game table where a ball can move at speeds over 100 km/h. The purpose of this simulator is to provide a pseudo-real environment for testing and allow the use of screen rendering for training computer vision algorithms capable of tracking a small object like a coloured ball at an average rendering speed of 200fps within a restricted and controlled space. The result will lead to the development of a control algorithm

capable of controlling four independent robotic systems that can play against a human opponent.

Currently, the results obtained from these tools are more than acceptable. Regarding the last physical simulator, the development of a more performant solution is currently underway, using a new rendering engine (Vulkan) and developing an algorithm that is performant and distributable on multiple embedded computer systems.

Section 1.2: Applications to Nanotechnology (UNICT, POLIMI, UNINA)
UNICT group involved in HPC has developed simulation codes for a resonant GaAs diode with two GaAlAs potential barriers using a drift-diffusion model where the presence of the Bohm potential for quantum effects has been included. Comparing the results with those obtained through a Monte Carlo approach based on signed particles for the Wigner equation, the parameters present in the quantum drift-diffusion model (DDQ) were optimized. As expected, the qualitative trend for relatively low applied voltages is well understood. For high voltages the DDQ model seems not accurate enough.
This analysis produced a preprint that will soon be submitted for publication: Giovanni Nastasi, Orazio Muscato, Vittorio Romano, Giorgia Vitanza "Optimized Quantum Drift Diffusion Model applied to a Resonant Tunneling Diode".
At the moment, UNICT is trying to improve the above considering instead of the DDQ model, a model based on isentropic gas dynamics, also in this case with quantum corrections. The resulting system of equations, consisting of a hyperbolic system coupled to the Poisson equation for the electrostatic potential, was numerically integrated using a Discontinuous Galerkin (DG) type approach, which is particularly suitable for taking into account potential barriers. Preliminary simulations seem to indicate that the path taken is appropriate but the analysis needs further investigation.
It should be emphasized that a collaboration with the POLIMI group (Carlo De Falco) is underway on the above topics in view of generalizations also to organic semiconductors.
UNINA is evaluating the deployment of a ARM microprocessor farm in real-time control of radiation detectors and sophisticated physics equipment [2-5] which require a continuous survey of their working points. We have developed a custom board interfaced with high-resolution 24-bit ADCs, monitoring bias voltages, temperature and humidity. The farm is made by 16 boards sending data over Internet. We are investigating the reliability of such an architecture and the most reliable software framework capable to assure the best quality of service in terms of uninterrupted runtime and low network and CPU overhead. UNINA also investigated the deployment of ARM-based front-end in the design of data-acquisition systems for characterization of both silicon devices as well as innovative Materials.

Section 1.3: Cosmological and Computational Fluid Dynamics codes (INAF)
This section is related with Section 1.1 of the report CN_HPC_FL5_D4_survey, where the main codes were introduced.

OpenGADGET
Our main target is the backbone of the code, that consists in the following "services":
- tree-bulding, that returns the global spatial representation of the particles distribution
- tree-walking, that returns the kNN of a particle
- domain decomposition, that is based on the top-region of the global tree, as detailed in the code description
These services are omni-present in every step and strongly determine the global scalability and performance of the code.
Our purpose is to extract them as a stand-alone app, to accurately profile and characterise

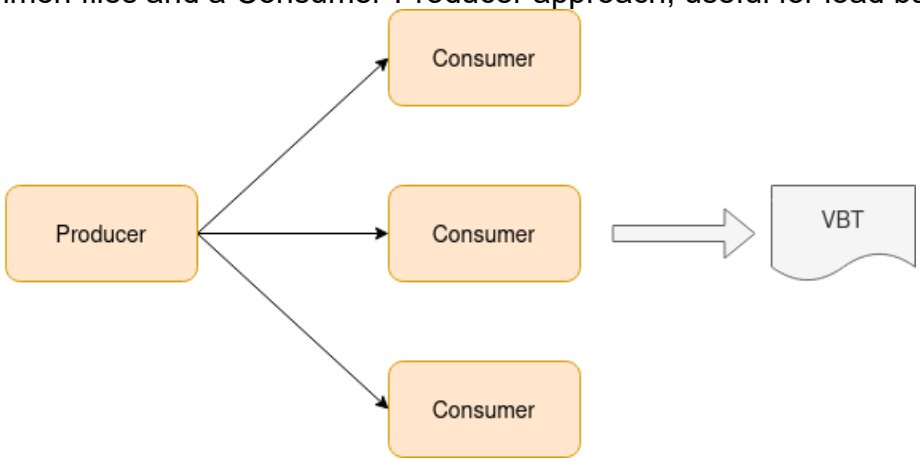
it on the different platforms and architectures provided by Spoke-1. We aim to optimize and evolve its capability as follows:
- enhancing the vectorization, possibly by changing the data layout
- introducing the threadization
- introducing an explicit NUMA-awareness through techniques that we have already developed

Pluto

Currently the development of the PLUTO code is oriented towards the optimal exploitation of computational platforms of the exascale class. In this context, energy efficiency has become a priority objective and the increasingly frequent use of accelerators in parallel computing systems, such as graphics processors (GPUs), represents a successful development in this direction. With this perspective, the developers of the PLUTO code are actively collaborating with NVIDIA Corporation which has recognized the importance of the work related to this numerical code in the research on astrophysical plasmas. The proposing group brings together the main developers, testers and users belonging to INAF actively engaged in the porting and testing of the PLUTO code on computational platforms of the exascale class.

Section 1.3.1: Astrophysics pipelines, Data analysis and scientific visualization

VisIVO will be extended and optimized for the real-time visualization of cosmological simulated data, giving the opportunity to compare with observational multiwavelength data, exploiting the available HPC platforms.
The importing modules are being parallelized for multi node/multi thread platforms using MPI. Specifically the importing modules will use MPI-IO to parallelize multiple reads and writes on common files and a Consumer-Producer approach, useful for load balancing.



The filtering modules will be extended to exploit multi GPU platforms investigating CUDA and OpenACC. Depending on the underlying complexity of the filter modules, some of them will instead employ MPI(e.g. the filters to merge VBTs or add new tabular columns).
The viewer modules core technology, based on VTK, is already optimized for emerging processor architectures and will be tailored to support the fine-grained concurrency for data analysis and visualization algorithms required to drive extreme scale computing by providing abstract models for data and execution that can be applied to a variety of algorithms across many different processor architectures. Finally a client-server based architecture (employing Paraview) will be exploited to avoid large scale data movements and to set up the render engine close to the data.

Section 1.3.2: Mini-applications and benchmarking of Least Square Solutions of Big Systems of linearized equations from astrophysics domains (GAIA)
To further study the Gaia AVU-GSR application, the following tests are proposed:

Weak scalability
The satellite will produce an increasing amount of data across the year, which will require an increasing amount of computing resources. For this reason, it is interesting to investigate the weak scalability of the code. In a MPI code, running on more nodes increases the parallelisation degree but also increases the number of communications. Specifically, we are interested in studying how the ratio between computation and communication changes by running the code for increasingly larger systems on an increasing number of nodes while keeping the amount of work per node constant.

Numerical stability
We aim to study the numerical stability of the code with the growth of the parallelism degree and the problem size. This is important e.g. for the Gaia mission, which aims to find astrometric parameters with a [10, 100] μarcsec accuracy, which implies very precise kinematic profiles, such as the rotation curve, a key observable to constrain different theories of gravity.
We aim to run a fixed-size system on a growing number of resources since the solution must not depend on the parallelism degree to test its numerical stability. Assuming as true the solution obtained on one MPI task, we will compute the relative error between the solutions derived on more resources and this reference solution and we will infer a relation between the relative error and the number of resources. Then, we will estimate the numerical stability as a function of the input data size, assuming the validity of the same relation.

Green Computing
An increasingly efficient parallelization on next-generation machines while maintaining a low environmental impact for the considered volumes of data is of interest for the national and international community. We will study if our proposed parallelization method is sufficiently “green” by comparing the power balance of the CUDA and OpenMP codes. In the OpenMP code, we do not have memory transfers between CPU and GPU, which might be quite power consuming, but in the CUDA code the computation is ~14 times faster, which might compensate for the consumption due to the data transfers. We will study which factor affects the most consumption by considering a growing system size proportional to the number of resources.

Further optimization
We aim to examine the code, to verify if there is room for further optimization, for example by better exploiting the different memory areas of the GPU and by adopting different parallelization languages.

In these tests, we will consider systems with an increasing size up to the size expected for the final Gaia dataset (~[10, 100] TB).
These tests are not only useful for the Gaia purposes but to investigate the behavior, in perspective of (pre-)Exascale systems of other applications that involve the usage of LSQR algorithm. The LSQR is currently employed in several contexts, such as visualization, cosmology, geophysics, geodesy, medicine, industry, and tomography.

Section 1.3.3: Building an accelerated OpenFOAM Proof-of-Concept application using Modern C++

Motivations and Scope

This work contributes to the state-of-the-art in OpenFOAM development by showcase a proof-of-concept toy application built using modern ISO C++ parallel constructs. By adopting this approach and by leveraging an appropriate compiler runtime stack like the one provided by the NVIDIA HPC SDK, it is possible to accelerate well-defined kernels allowing multi-core execution and GPU offloading using a single code-base.

Background

Generally, every OpenFOAM simulation consists of two stages: (1) the assembly, during which linear systems of algebraic equations are derived from the discretization of a set of partial differential equations and (2) the solver during which the solution of these systems is obtained using various numerical methods (often provided by external libraries).

Previous works [9] have shown that the main bottleneck on the OpenFOAM simulation resides in the solver part for a broader set of use-cases. It has been observed that assembly too can become a dominant aspect of a simulation. Many efforts [10, 11] have been devoted to improving the performance of the solver part [12, 13] as well as non-solver parts [14]. Despite some recorded successes, previous attempts to add GPU acceleration to code have not been adopted by the main code-base due to its maintainability and intrusiveness.

Following OpenFOAM development principles, the ideal code contribution ready to be merged into the official code-base has to follow these two principles: (1) it should not add or require in a mandatory way any specialized programming model which is also vendor-specific; and (2) the code as-is must compile on any platform, with or without accelerators, using open-source compiler tool-chains (like GNU).

Starting from the ISO C++11 standard, a first set of parallel constructs and concurrency topics have been introduced as part of the C++ language. Further releases of the ISO C++ standards (C++14, C++17, C++20) have progressively introduced new concepts and refined the specification. The NVIDIA C++ compiler is able to compile modern C++ and generate an executable that can run on GPUs. This is achieved just by adding in the command-line the option -stdpar=gpu. Recent studies [15, 16] show that with standard parallelism cleaner code can be written and good performance can be achieved.

OpenFOAM is designed to be portable on different CPU architectures thanks to the portable nature of the C++ programming language. Many compilers support it and these compilers are available for many CPU architectures (not only x86). Therefore, our challenge was to modify the code-base to run on GPUs without limiting its portability and without introducing much duplicated and over-specialized code. For the proposed Proof-of-Concept, we focused only on native C++ techniques so that the portability of the code is not affected in any way.

One recurrent operator, evaluated in the assembly phase of a general OpenFOAM simulation, is the gradient operator. For this reason a mini-application which is focused on the evaluation of the gradient of a scalar field was created. This application is only focused on performing this simple operation different times.

Following a profiling exercise of the gradient operator, we have identified that 99% of the time is spent in three main functions: interpolate, gradf and correctBoundaryCondition. Using only standard C++ parallel constructs like std::for each and std::transform, these functions were adapted to expose parallelism where appropriate.

The most expensive of the three is the gradf function which, internally, calls several other functions. One of the most expensive steps is a loop over the cell faces of the elements. This loop leverages LDU addressing, which is unique in OpenFOAM, and unfortunately it cannot be simply parallelized due to data races. To overcome this, we introduced a new element list which re-writes the owner and neighbour elements corresponding to a cell label. With these modifications, parallelism can be exploited safely and the loop can be offloaded to the GPU transparently by the compiler runtime.

Implementation

We have chosen for our performance characterisation an Arm-based platform called NVIDIA Arm HPC Developer Kit. The platform is a 2U production-ready server comprising one single-socket 80-cores Ampere running at 3.0 GHz (based on Arm Neoverse N1) and two NVIDIA A100 PCIe GPUs with 40 GByte HBM2 (not connected via NVLink/NVBridge). CPU and GPU are connected via PCIe Gen 4. This platform represents a good on-ramp step toward future on-prem and cloud Arm-based HPC system designs, including the NVIDIA Grace Hopper Superchip.

As described in the previous section, we developed a mini-app from OpenFOAM modules to simplify a focused analysis of gradient operator. We design experimental cases to stress this operator and focus all timing and profiling exclusively on it.

For CPU profiling we mostly used a combination of gprof and callgrind. For GPU profiling, we used the NVIDIA Nsight System. All the meshes considered in this work are 2D meshes, for simplicity. We report in this abstract for brevity two cases: one with 16 millions of unknowns (labeled as Mesh-16M) and another with 64 millions of unknown (labeled as Mesh-64M).These problem sizes were chosen to make sure all data fit in memory within one single one node without requiring swapping.

Our performance evaluations are strictly focused on the gradient operator. Execution is repeated artificially multiple times for measurement purposes. Since the operator is composed of different routines, most GPU accelerated and other not, we have recorded a wide range of speed-ups by sub-routines varying from near 1x (performance parity) up to 16x. Table 1 and 2 show end-to-end speed-up of the first and second iterations for the full gradient computation. Timings include computation done on CPU, GPU and extra pre-fetching operations (if any). We separate the results in these two tables because there are some calculations that only occur on the first iteration and not on subsequent iterations, unless there are geometric changes in the mesh.

1st Iteration	HYBRID vs PURE MPI	2 MPI + 2 GPU vs PURE MPI	2 MPI + 2 GPU vs HYBRID
Mesh-16M	1.14	2.03	1.79
Mesh-64M	1.52	4.27	2.81

Table 1: Full speed-ups of gradient operator evaluation during the 1st Iteration (HYBRID = 10 MPI × 8 threads)

2nd Iteration	HYBRID vs PURE MPI	2 MPI + 2 GPU vs PURE MPI	2 MPI + 2 GPU vs HYBRID
Mesh-16M	1.02	2.82	2.78
Mesh-64M	0.73	1.66	2.26

Table 2: Full speed-ups of gradient operator evaluation during the 2nd Iteration (HYBRID = 10 MPI × 8 threads)

These speed-ups also take into consideration portions of code that have not been modified. Explicit pre-fetching techniques were used to transfer data between CPU and GPU before the gradient computation and avoid uncontrolled data swap during kernel execution. By doing so we have been able to establish an indicative performance upper bound assuming that in the future other sections of the code could be ported to modern C++ and eventually be offloaded to GPU and at the same time reducing unnecessary data movement in both directions (CPU to GPU and viceversa).

From these initial analysis it can be seen that executing the gradient operator on GPU leads to a variable but positive gain in performance (considering that not everything run on GPU). Adopting an HYBRID approach (MPI + OpenMP) does not lead to significant results versus PURE MPI. However it neither bring performance losses. Different combination MPI processes and OpenMP threads could have a different performance impact.

At present, the work continues. We are looking at extending our evaluations on dense x86 GPU platforms (NVIDIA HGX 4-way and 8-way), more and extensive testing varying number of MPI processes vs number of OpenMP threads vs number of GPUs, further pre-fetching and GPU kernel performance studies, extra routine porting (if required). We aim to publish this work in a leading journal and open source the code to the community.

This work will be presented at the 18th OpenFOAM conference in Genoa from 11 to 14 July.

Section 1.4: (bio)electrostatics and electromagnetism (IIT, UNICT, POLIMI)

The development of electromagnetic codes will be done on the open source platform FenicsX https://fenicsproject.org/.
Biomolecular electrostatics applications will originate from in house code, developed at the MOX group, with the analytical expressions derived at IIT [6].

Section 1.4.1: Template matching. tasks in protein-protein docking and cryo-em applications.

We are developing our in house code, starting from some different analytical expressions and expansions simplifying the rotational scheme.

Section 1.4.2: Quantum physics.
As anticipated the MaZe code does not yet take advantage of parallel computing. In this scenario we are planning hence to take advantage of High Performance Computing to further boost the performance of the software. We will first perform a profiling of the code to identify the key routines of the simulative approach, next we will optimize them first through serial code refactoring (data structures, functions etc..) and next through proper parallelization. The parallelization will both take advantage of multi-core CPUs and GPUs. We will investigate, among others, the performance achievable through the OpenACC framework as this one consents to preserve code readability.

Section 1.4.3: Computational geometry.
In IIT it was developed the NanoShaper software [7], a tool for building and analysing the surface of molecular systems at the nanoscale. In a few years, it became a widespread and well recognized tool [8]. One of the nice features of NanoShaper is the fact that uses an inherently parallel computational approach; NanoShaper is already parallelized for shared memory architectures (Boost Threading) but not for multi-process. We are planning to parallelize it in multi-processing terms leveraging the portable MPI library.

Section 1.5: Reusable computational kernels for simulations based on PDEs and IEs.
The POLIMI group are working on the definition of a set of case studies for the identification of reusable computational kernels whose advanced tuning and optimization can strongly impact the performance of simulation tools based on Partial Differential Equation (PDE) and Integral Equation (IE) models.

A set of applications and numerical methods to be targeted in our study have been identified based on the competences an

In this phase of the project, the POLIMI group are working on the definition of a set of case studies for the identification of reusable computational kernels whose advanced tuning and optimization can strongly impact the performance of simulation tools based on Partial Differential Equation (PDE) and Integral Equation (IE) models.

Section 1.6: Large sparse linear systems, algebraic and geometric multigrid (UNIPD, UNICT)

UNPD activities have focused on the following directions: benchmarking and performance modeling of accelerators; fundamental primitives with hardware accelerators and processing in memory; acceleration of Algebraic Multigrid (AMG); efficient computation of statistical properties of the edit distance. Below we provide a brief description for each direction.

The contribution aims at capturing the performance of accelerators with a middle-level model that is not vendor specific. To this aim, it will be important to quantitatively compare the prediction of the ML model with
a computational model (high-level model), to determine how much predictivity is gained,
a register-level model (low-level model), to ascertain how much predictivity is lost.
During this phase of the project, the following two models have been selected for reference.
High-level model: TCU (https://doi.org/10.1145/3350755.3400252).
Low-level model: Accel-Sim (https://doi.org/10.1109/ISCA45697.2020.00047).
Accel-Sim also includes a benchmarking framework. As for the applications to be included in the benchmarks, linear algebra primitives will be part of the set. We are also evaluating the inclusion of DFT algorithms, where we can leverage the experience of the group on the topic (https://dl.acm.org/doi/abs/10.5555/2461196.2461210). Some of the application code will be custom-developed, and some will come from benchmark suites for heterogeneous computing (e.g., https://doi.org/10.1109/IISWC.2009.5306797).

We will focus on the design of efficient algorithms that exploit the memory hierarchy and memory accelerators, in particular processing-in-memory. We will address fundamental primitives for e.g., linear algebra and graph processing. We will consider the problem of frequent pattern mining, including frequent subgraph mining. Frequent subgraph mining is a fundamental task in the analysis of collections of graphs. While several exact approaches have been proposed, it remains computationally challenging on large graph datasets due to its inherent link to the subgraph isomorphism problem and the huge number of candidate patterns even for fairly small subgraphs. In recent work, we have studied two statistical learning measures of complexity, VC-dimension and Rademacher averages, for subgraphs, and derived efficiently computable bounds for both. We showed how such bounds can be applied to devise efficient sampling- based approaches for rigorously approximating the solution of the frequent subgraph mining problem. We also showed that our bounds can be used for true frequent subgraph mining, which requires identifying subgraphs generated with probability above a given threshold from an unknown generative process using samples from such a process. Our extensive experimental evaluation on real datasets showed that our bounds lead to efficiently computable, high-quality approximations for both applications. We will study how to employ accelerators in frequent pattern mining, including sampling approaches.

The goal of this task is the acceleration of Algebraic Multigrid (AMG) solvers in industrial applications of significant interest. Hence, on one side, we focused on the selection of the most demanding simulation tools that are currently used by scientists and engineers. After a survey, we decided to focus on structural mechanics, geomechanics and computational fluid dynamics. On the other side, we deeply analyzed AMG, which is a very complex method, to better understand which of its functional components really requires an acceleration. From this analysis, we decided to concentrate ourselves on coarsening, prolongation set-up and the sparse matrix times sparse matrix kernels.

We are investigating the efficient computation of statistical properties of the edit distance, a metric of dissimilarity between strings, widely applied in computational biology, speech recognition, and machine learning. We target specifically the average edit distance between random strings, proposing new algorithms to obtain deterministic upper and lower bounds as well as high-confidence statistical estimates. These algorithms yield improvements on most
previously published numerical values as well as results for alphabet sizes and string lengths not reported before. These results appear in one of our publications below.

UNICT implemented a multigrid approach to solve the Poisson equation on MPI architectures using the PETSc library suite. The code is available at https://github.com/armcoco/Poisson3DMG.git.
As a preliminary result, UNICT verified that weak and strong scalability are satisfactory for regular domains (ie rectangles in 2D and parallelepipeds in 3D), while some degradation is observed in the presence of curved domains. The last aspect is being investigated within the group, allowing the iterative solver to have more flexible relaxation parameters to optimize in terms of efficiency and parallel scalability.
Other applications of the Poisson solver that are being worked on are:
- computational fluid dynamics, concerning the solution of the Navier-Stokes equations to simulate the interaction between incompressible fluids and rigid bodies;
- gravitational problems in volcanic areas, concerning the evaluation of gravitational signals generated by underground magmatic migrations;
- degradation of monuments, related to the simulation of marble sulphation caused by atmospheric pollution.

Section 1.7: Construction and testing of parallel numerical algorithms for multiscale simulations in plasma physics (UNIFE, UNICT)
Constructing and testing parallel numerical algorithms for multiscale simulations in plasma physics is an active area of research, as plasma simulations often involve a large number of particles and the integration of multiple scales [18]. One approach to constructing parallel numerical algorithms is to use domain decomposition techniques, where the computational domain is divided into smaller subdomains that can be solved in parallel. Another approach is to use adaptive mesh refinement, where the mesh resolution is adjusted based on the local physics and computational resources available. In the presence of collisions the computational challenges increase dramatically and one needs either to use Monte Carlo or particle methods or suitable fast algorithms to integrate the Landau collision term. Both approaches are naturally integrated into a parallel structure thanks to splitting algorithms [17].
Testing these parallel algorithms involves assessing their scalability, accuracy, and efficiency on different computational architectures, such as multi-core CPUs and GPUs. Additionally, benchmark problems, such as simulations of plasma waves and instabilities, magnetohydrodynamic simulations of plasma confinement and stability, and fluid simulations of plasma sheaths and boundary layers, can be used to compare the performance of different parallel algorithms and assess their suitability for specific plasma physics applications.

Section 1.8: Cellular Automata (CA) models (UNICAL)
In this initial stage, in order to test the library and LB solutions in the case of Cellular Automata parallel execution, a first set of mini applications have been selected, aiming to show the goodness of the approach from both a qualitative and quantitative point of view.

Regarding the qualitative assessment, we have selected two mini application examples, namely “BallCell” and “SciddicaT”. While the first one is a CA model which simulates the motion of a set of non-stationary cells (of circular shape) where a customizable pseudo-computational function can be set for different pseudo workloads, by simulating unbalanced conditions over the computational domain, the second one is an example of semi empirical fluid flow model for the simulation of slow-moving landslides where the computational unbalance is due to the fact that the simulated phonomenon is connected and does not affect the entire domain. Both of them are able to show how the nodes workload remains balanced throughout the execution of the simulation.
A third mini application is “etero-genius”, developed to quantitatively analyse the benefits of the proposed LB by measuring the improvements achieved in terms of execution time reduction. The model is based on the direct discrete formulation of the Richards’ equation as formalized in [1], and describes double-phases flow in porous media by combining the mass conservation equation with the Darcys’ law.

Section 1.9: Design of efficient simulation codes for uncertainty quantification in traffic flow problems and kinetic equations (UNIFE, UNICT)
UNIFE contribution is related with the design of efficient simulation codes for uncertainty quantification in traffic flow problems. This is a challenging task, as traffic flow simulations typically involve a large number of uncertain parameters and the integration of multiple scales. Additionally, the design of efficient simulation codes for uncertainty quantification in traffic flow problems requires a combination of expertise in traffic flow modeling, uncertainty quantification, and high-performance computing. One approach to designing efficient simulation codes is to use surrogate models, which are simplified models that approximate the behavior of the full traffic flow model. Therefore, instead of using the full simulation of the individual cars motion one can take advantage of fluid-dynamic models like Lighthill-Whitham-Richards (LWR) or Aw-Rascle-Zhang (ARZ) models [19].
These surrogate models can be trained using a combination of simulation data and machine learning techniques, such as artificial neural networks and Gaussian process regression, and can be used to propagate uncertainties in the model inputs to outputs with reduced computational cost. Another approach is to use parallel computing techniques, such as domain decomposition, to distribute the simulation workload across multiple processors and improve simulation performance.

UNICT contributes with the study of methods for the numerical solution of kinetic equations. Such equations present a challenge to solve, mainly due to the size of the problem. The distribution function, in fact, depends on three spatial coordinates, three velocity components and time. Methods for the Boltzmann equation of the rarefied gas dynamics based on a semi-Lagrangian discretization of the transport term are currently under study, which guarantees a better computational efficiency thanks to the possibility of using larger time steps from the corresponding Eulerian schemes. The calculation of the collision operator is performed using a spectral method, which allows to reach high accuracy in speed and a remarkable efficiency compared to other types of discretization. In a preliminary phase, code prototypes will be developed using Matlab, on reduced-dimension models (for example one spatial dimension and two in velocity), to then move on to parallel implementations which will allow tackling more realistic problems.

Section 1.10: Modelling and simulation of space dependent epidemic dynamics (UNIFE)
The development of models and algorithms for the simulation of space-dependent epidemic dynamics is a rapidly growing field that seeks to understand how infectious diseases spread and evolve in different geographic locations. In these simulations, spatial heterogeneities in population density, demographics, and mobility patterns are taken into account. Mathematical models, such as compartmental models and agent-based models, are used to describe the spread of infectious diseases, and numerical simulations are used to test different scenarios and interventions. These simulations can help to identify high-risk areas and populations, optimize resource allocation, and evaluate the effectiveness of different public health interventions, such as vaccination campaigns and travel restrictions. We will consider models described by complex systems taking into account also social features such as age and number of contacts of individuals and test the use of physically informed neural networks (PINN) on HPC platforms on realistic scenarios concerning Covid-19 under uncertain data.

Section 2: Cooperations between groups
In this section we summarize some of the connections that have been created or strengthened within the project. Most connections have already been detailed in Section 1, therefore we provide a brief sketch here. We refer to the connections between units of the project, as well as with external research groups.

INAF collaborates with UNICT, UNITO, CINECA.

UNIPD collaborates with:
ETH
IT Univ. of Copenhagen
with Massimo Bernaschi, CNR, on the development of a GPU-accelerated sparse matrix times sparse matrix kernel for distributed memory systems able to scale to hundreds of compute nodes by overlapping computation and communication

Regarding UNIPI, part of the survey work on the algorithms for the solution of sparse linear systems and on the implementation of the new functions of the PSCToolkit library was carried out in collaboration with the research unit at the University of Rome "Tor Vergata".

IIT cooperates with other groups within the scope of Spoke 1:
MOX POLIMI (Prof. De Falco)
UNITO (Prof. Aldinucci)
UNIPI (Prof. Ferragina)

UNIFE cooperates with:
RWTH Aachen University (Prof. Michael Herty, collaboration on optimization method and applications to traffic flows)
TU Munich (Prof. Massimo Fornasier, collaboration on optimization methods and machine learning)
U. Lille (Prof. Thomas Rey, collaboration on numerical simulations of the Boltzmann equation)
U. Toulouse (Prof. F. Filbet, collaboration on numerical methods in plasma physics)
GSSI L'Aquila (Prof. N. Guglielmi, collaboration on numerical methods in biomedical and epidemiological applications)

U. Catania (Prof. G. Russo, collaboration on numerical methods for kinetic equations with multiple scales)

UNICAL cooperates with:
UPC Barcelona TECH – Spain
BSC Supercomputing Center – Barcelona - Spain
ICAR – CNR - Italy

Section 3: Outputs
Most outputs at this stage consists of a selection of published papers, as well as submitted papers and numerical codes uploaded in appropriate repositories.

Section 3.1: Published and Accepted Papers

UNICT
Fargetta G., Scrimali, L. A Sustainable Dynamic Closed-Loop Supply Chain Network Equilibrium for Collectibles Markets, Computational Management Science 20, 19, 2023.

UNIPD
Gianfranco Bilardi, Michele Schimd, Bounds and Estimates on the Average Edit Distance, arXiv:2211.07644, November 16, 2022 (pp 1-41; funding ack. to CN1, p.38).

Shiyuan Deng, Francesco Silvestri and Yufei Tao.The I/O Complexity of Enumerating Subgraphs of Constant Sizes. In Proc. 26th International Conference on Database Theory (ICDT), 2023.

Paolo Pellizzoni and Fabio Vandin. VC-dimension and Rademacher Averages of Subgraphs, with Applications to Graph Mining. In Proc. 39th IEEE International Conference on Data Engineering (ICDE), 2023.

UNIPI
P. D'Ambra; F. Durastante; S. Filippone; Parallel Sparse Computation Toolkit. Software Impacts 15 (2023), no. 100463.
V. Besozzi; M. Della Bartola; L. Gemignan; Experimental Study of a Parallel Iterative Solver for Markov Chain Modeling. International Conference on Computational Science, ICCS-2023. Springer (2023). In press
F. Durastante, L. Aceto; Efficient computation of the sinc matrix function for the integration of second-order differential equations. (2023) Submitted.

UNIFE
Boscheri, Walter; Dimarco, Giacomo ; Pareschi, Lorenzo . Locally structure-preserving div-curl operators for high order discontinuous Galerkin schemes. J. Comput. Phys. 486 (2023), Paper No. 112130.
Medaglia, Andrea ; Pareschi, Lorenzo ; Zanella, Mattia . Stochastic Galerkin particle methods for kinetic equations of plasmas with uncertainties. J. Comput. Phys. 479 (2023), Paper No. 112011, 22 pp.
Borghi, Giacomo ; Herty, Michael ; Pareschi, Lorenzo . Constrained consensus-based optimization. SIAM J. Optim. 33 (2023), no. 1, 211--236.

Bertaglia, Giulia ; Lu, Chuan ; Pareschi, Lorenzo ; Zhu, Xueyu . Asymptotic-preserving neural networks for multiscale hyperbolic models of epidemic spread. Math. Models Methods Appl. Sci. 32 (2022), no. 10, 1949--1985.
Pareschi, Lorenzo ; Rey, Thomas . Moment preserving Fourier-Galerkin spectral methods and application to the Boltzmann equation. SIAM J. Numer. Anal. 60 (2022), no. 6, 3216--3240.
Herty, Michael ; Iacomini, Elisa ; Visconti, Giuseppe . Recent trends on nonlinear filtering for inverse problems. Commun. Appl. Ind. Math. 13 (2022), no. 1, 10--20.
Guglielmi, Nicola ; Iacomini, Elisa ; Viguerie, Alex . Delay differential equations for the spatially resolved simulation of epidemics with specific application to COVID-19. Math. Methods Appl. Sci. 45 (2022), no. 8, 4752--4771.
Herty, Michael ; Iacomini, Elisa . Uncertainty quantification in hierarchical vehicular flow models. Kinet. Relat. Models 15 (2022), no. 2, 239--256.

UNICAL
Tailoring load balancing of cellular automata parallel execution to the case of a two-dimensional partitioned domain, A De Rango, A Giordano, G Mendicino, R Rongo, W Spataro, The Journal of Supercomputing, 1-15
Towards efficient GPGPU Cellular Automata model implementation using persistent active cells, P Renc, T Pęcak, A De Rango, W Spataro, G Mendicino, J Wąs, Journal of Computational Science 59, 101538
Opencal system extension and application to the three-dimensional richards equation for unsaturated flow A De Rango, L Furnari, A Giordano, A Senatore, D D'Ambrosio, W Spataro, S Straface, G Mendicino, Computers & Mathematics with Applications 81, 133-158

Section 3.2: Numerical codes

UNICT
https://github.com/armcoco/Poisson3DMG.git

Section 4: Reference list

Reference list

1. G. Mendicino, A. Senatore, G. Spezzano, and S. Straface, "Three-dimensional unsaturated flow modeling using cellular automata," Water Resources Research, vol. 42, no. 11, pp. 2332–2335, 2006.
2. ALOISIO, Alberto, et al. uSOP: A microprocessor-based service-oriented platform for control and monitoring. IEEE Transactions on Nuclear Science, 2017, 64.6: 1185-1190.
3. Di Capua, F., Aloisio, A., Ameli, F., Anastasio, A., Branchini, P., Giordano, R., ... &; Tortone, G. (2017). Monitoring complex detectors: the uSOP approach in the Belle II experiment. Journal of Instrumentation, 12(08), C08003.
4. G. Tortone, A. Anastasio, V. Izzo, A. Aloisio uSOP: AN EMBEDDED LINUX BOARD FOR THE BELLE2 DETECTOR CONTROLS, 16th Int. Conf. on Accelerator and Large Experimental Control Systems ICALEPCS2017, Barcelona, Spain JACoW Publishing ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUSH302
5. Di Capua, F., Aloisio, A., Ameli, F., Anastasio, A., Branchini, P., Giordano, R., ... &; Tortone, G. (2018). A Service-Oriented Platform for Embedded Monitoring Systems in Belle II Experiment. In Proceedings of International Conference on Technology and

Instrumentation in Particle Physics 2017: Volume 1 (pp. 54-57). Springer Singapore.
6. SV Siryk, W Rocchia, “Arbitrary-Shape Dielectric Particles Interacting in the Linearized Poisson–Boltzmann Framework: An Analytical Treatment” J. Phys. Chem. B 126 (49), 10400-10426, 2022
7. S. Decherchi and W. Rocchia, "A general and robust ray-casting-based algorithm for triangulating surfaces at the nanoscale" PLoS ONE 8(4): e59744, 2013
8. S. Decherchi et al. “NanoShaper-VMD interface: computing and visualizing surfaces, pockets and channels in molecular systems”. Bioinformatics, 35(7), 1241-1243, 2019
9. M. Culpo, “Current Bottlenecks in the Scalability of OpenFOAM on Massively Parallel Clusters,” Aug. 2012. [Online]. Available: https://doi.org/10.5281/zenodo.807482
10. I. Spisso, G. Amati, V. Ruggero, and C. Fiorina, “Porting, optimization and bottleneck of OpenFOAM in KNL,” Intel eXtreme Performance Users Group (IXPUG), Tech. Rep., 2018.
11. I. Spisso and G. Amati, “HPC Comparison of Hypre vs Pstream as external linear algebra library for OpenFOAM,” 2018.
12. S. Bna, I. Spisso, M. Olesen, and G. Rossi, “PETSc4FOAM: a library to plug-in PETSc into the OpenFOAM framework,” Jun. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3923780
13. M. Martineau, S. Posey, and F. Spiga, “AmgX GPU Solver Developments for OpenFOAM,” 2021.

14. D. Molinero Hernandez, S.-R. Galván-González, J. Pacheco, and N. Herrera, Multi GPU Implementation to Accelerate the CFD Simulation of a 3D Turbo-Machinery Benchmark Using the RapidCFD Library, 12 2019, pp. 173–187.

15. W.-C. Lin, T. Deakin, and S. McIntosh-Smith, “Evaluating ISO C++ Parallel Algorithms on Heterogeneous HPC Systems,” in International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems held in conjunction with Supercomputing (PMBS). IEEE, 2022, in press.

16. J. Latt, C. Coreixas, and J. Beny, “Cross-platform programming model for many-core lattice Boltzmann simulations,” PLOS ONE, vol. 16, no. 4, p. e0250306, apr 2021. [Online]. Available: https://doi.org/10.1371%2Fjournal.pone.0250306
17. “Numerical methods for kinetic equations” by G. Dimarco, L. Pareschi, Acta Numerica (2015)
18. “Numerical methods for plasma physics in collisional regimes”, G. Dimarco, Q. Li, L. Pareschi, B. Yan, Journal of Plasma Physics, (2015)
19. “Models for Vehicular Traffic on Networks”, M. Garavello, K. Han, B. Piccoli, AIMS on Applied Mathematics, (2017)