jh200053-MDHI

Preparing for Exa-systems: Performance portable implementation and scalable data analysis

Yuuichi Asahi（Japan Atomic Energy Agency）

Abstract
We aim to explore a performance portable implementation for fusion plasma turbulence codes like GYSELA and establish a large scale data analysis approach. For this purpose, we have encapsulated the key feature of GYSELA such as the high dimensionality and the semi-Lagrangian scheme into a mini-application that makes use of MPI + OpenACC/OpenMP directives or MPI + Kokkos performance portable framework to run across multiple CPU and GPU platforms. Thanks to the optimization, it is shown that both implementations can offer good performance portability on CPUs and GPUs as long as the compiler auto-vectorization is effective. For scalable data analysis, we have accomplished an extreme scale Principal Component Analysis (PCA) on the time series of 5D distribution function data (> 10TB) based on the Dask framework. It is shown that 83 % of the variance of the original 5D data can be expressed with 64 principal components, corresponding to the compression of the degrees of freedom from 10^{12} to 10^{9}. Another advantage of the proposed analysis is the decoupling of 6D (1D time and 5D phase space) data into the combinations of 3D data that are visible to human eyes.

1. Basic Information

(1) Collaborating JHPCN Centers

Tokyo-Tech, Nagoya

(2) Research Areas

- ☑ Very large-scale numerical computation
- ☑ Very large-scale data processing
- ☐ Very large capacity network technology
- ☐ Very large-scale information systems

(3) Roles of Project Members

Project representative Yuuichi Asahi works for a performance portable implementation of a GYSELA mini-app. Shinya Maeyama performs the local plasma turbulence simulations. Julien Bigot works on the in-situ data analysis of GYSELA. Xavier Garbet works for theoretical analysis of non-local transport processes. Guillaume Latu gives the advice for the parallelization of GYSELA miniapp. Keisuke Fujii contributes on data-driven analysis. Kevin Obrejan performs the global plasma simulations. Tomohiko Watanabe comments on characteristics of local transport processes. Yasuhiro Idomura gives advice for the large scale simulations. Takahiro Katagiri supports the optimization on Flow, particularly the intranode parallelization. Takayuki Aoki gives advices about the usage of TSUBAME3.0.

2. Purpose and Significance of Research

In this project, we aim to explore a performance portable implementation for fusion plasma turbulence codes like GYSELA (developed in France) and establish a scalable approach for advanced data analysis. In FY2020, we have focused on the optimization and MPI parallelization of a performance portable version of a mini-app. We have also worked for establishing the data analysis method for large scale numerical simulation data.

Since high diversity is expected for the coming exa-scale supercomputers,

application developers are now facing the huge challenge to achieve, within reasonable time, a good performance on these machines which may require architecture specific parallelization strategies. Thus, application developers are starved of performance portable parallelization approaches which allow a single codebase to run efficiently on many different architectures, otherwise it is too demanding and is hardly sustainable to utilize multiple exa-scale supercomputers. Establishing a performance portable approach over multiple architectures therefore contributes on many kinds of computational fluid dynamics applications.

In addition, there is an ever-increasing demand for the so-called big-data analysis dedicated to the fluid simulations. Firstly, the simulation data size is getting larger and larger as the computational capability is improved on the way to exa-scale supercomputing. Secondly, an emergence of the data science brings about the capability to extract features and reduce the dimensionality of the fluid simulation data. This kind of scalable data analysis method is beneficial not only for the HPC community but also for the academia and industries.

3. Significance as JHPCN Joint Research Project

In this project, we deal with large scale data analyses and performance portable implementations both of which require specialized knowledge.

For the large-scale data analyses, we investigate the phase space pattern formation from the time series of 5D fusion turbulence data. To manage the huge data over 10 TB without MPI parallelization, Japanese group has developed an Dask based incremental PCA. As usual for the data-driven approach, however, there is no straight forward way to interpret the outcome of the analysis. The theoretical background of French group on the pattern formation and transport processes has greatly helped to give the physical understanding for the obtained data. Thus, an expert level interaction between French and Japanese groups is necessary to establish our data-driven analyses.

To establish and improve a performance portable implementation dedicated to fusion codes is also a very challenging task which needs a good understanding of the large variety of architecture characteristics and programming models. In FY2019, we have already found that the OpenACC is the most promising directive for GPUs and Kokkos supports portable implementations without code duplication. A remaining question, of crucial importance, is how to optimize the performance portable codes. In FY2020, we have focused on this aspect. For this purpose, Japanese group can offer the GPU optimization strategies with Kokkos and directives, and French group can offer the optimization techniques for many-core CPUs, particularly x86 CPUs.

A deep collaboration based on JHPCN is thus essential to accomplish the objectives described above. In FY2020, we newly added a HPC specialist from CEA, JAEA and Nagoya Univ. and a data scientist from Kyoto Univ. Since we need a GPU environment like TSUBAME3.0 and a many-core CPU environment like Flow

TypeI subsystem, multi-platforms offered by JHPCN framework are also necessary to evaluate the performance portability.

4. Outline of Research Achievements up to FY2019

In FY2019, we extracted the key features of GYSELA such as high dimensionality and the semi-Lagrangian scheme, and encapsulated them into a mini-application which solves the similar but a simplified Vlasov-Poisson system (2D space and 2D velocity space).

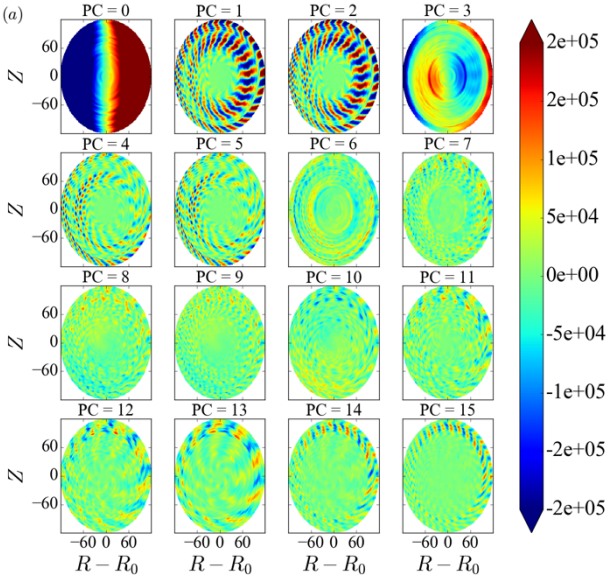
We implemented the mini-app in C++ language with a mixed OpenACC/OpenMP approach and Kokkos [3, 8]. By comparing these three approaches, we clarified the advantages and disadvantages of the directive-based approach and the higher level abstraction approach from the viewpoints of readability, performance portability, and productivity. We found that the Kokkos version with 3D MDrange policy to parallelize the 4D loops achieved better performance than the directive version. Some kernels have achieved almost ideal performance. Based on our experience, Kokkos can offer a readable, productive and performance portable codes at the cost of initial porting efforts. This result was published as a peer-reviewed paper [3].

5. Details of FY2020 Research Achievements

Scalable Data Analysis

5D (3D space + 2D velocity space) gyrokinetic simulations are powerful tools to analyze and predict turbulent transport in magnetic confined fusion plasmas. From the simulation results, the pattern formation in 3D real space (radial r, poloidal θ, toroidal φ directions) and its relationship to the transport properties have been discussed. Although the simulation itself is 5D and solves the time evolution of 5D distribution functions, less attention has been paid for the pattern formation in the full 5D phase space. There are fundamentally two difficulties for this type of pattern extraction: the huge data size (>10 TB) and the high-dimensionality more than 3D.

In this project, we have developed a new method to extract patterns in the 5D phase space based on Principal component analysis (PCA). As shown in Fig. 1, we have constructed the phase scape bases in $(\varphi, v_{\parallel}, w)$ and spatial coefficients in (r, θ) from the 5D time series data. Here the 2D velocity dimensions $(v_{\parallel}, w)$ represent the parallel and perpendicular directions with respect to the magnetic field line.



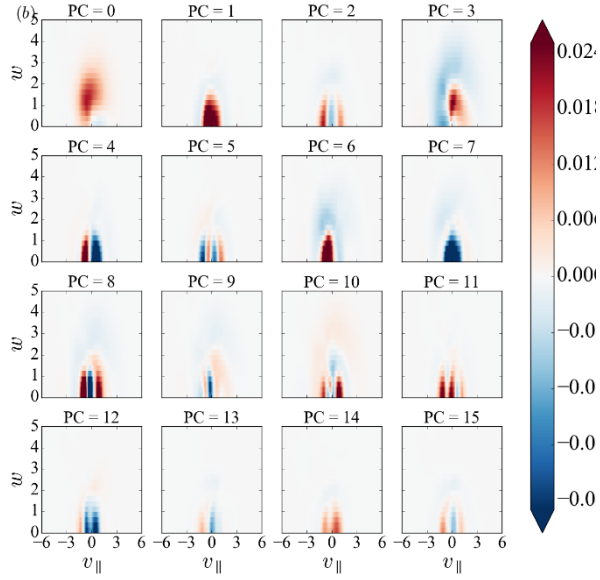


Fig. 1 First 16 principal components (PCs) of (a) of spatial coefficients and (b) phase space bases.

It was shown that 83 % of the variance of the original 6D data can be expressed with 64 principal components, where the compression of the degrees of the freedom from 10^{12} to 10^{9} is achieved. Another advantage of this method is the decoupling of 6D data into the combinations of 3D data which are visible to human eyes as shown in Fig. 1. We found that the extracted structures are similar to the structures known from the past plasma turbulence studies. For example, we found that PC0 represents a magnetic geometry. PCs 1 and 2 correspond to the so-called ballooning mode structures. Through the detailed analysis of the contribution of each PC to the energy flux, we found that the avalanche like energy transport is driven by coherent mode structures in the phase space. This work has been published as a peer-reviewed article [1]. This work was also presented in the international conference [6] and domestic conferences.

In parallel to this analysis, we have developed a new approach to analyze the nonlinear interactions in the plasma turbulence. The proposed approach is applied to understand the multi-scale interactions between ion and electron scale turbulence. This work is presented as an invited talk in the international conference [7], and the methodology is presented as a peer-reviewed article [2]. The developed library for this analysis is available on the GitHub page [10].

High Performance Computing

To further evaluate the effectiveness of directive-based implementation, we have tried the OpenMP4.5 for GPU offloading [4, 8]. Fig 2. shows the elapsed time of multiple kernels in mini-app with OpenMP4.5, OpenACC and Kokkos for one iteration on Nvidia V100. As shown, the OpenMP4.5 version is slower compared to Kokkos or OpenACC versions. This may be partially explained by the STREAM Triad bandwidth of 540 GB/s which is 30 % smaller compared to Kokkos or OpenACC versions. Based on this test, we decided to use Kokkos and OpenACC for intra-node parallelization.

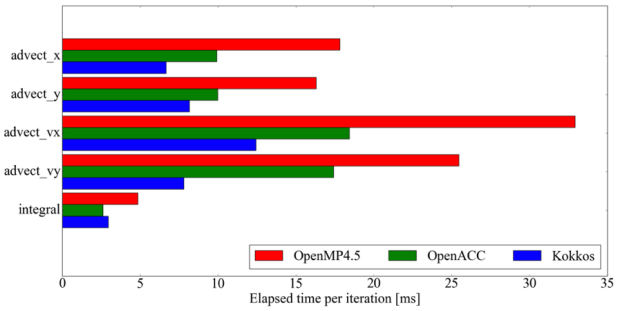
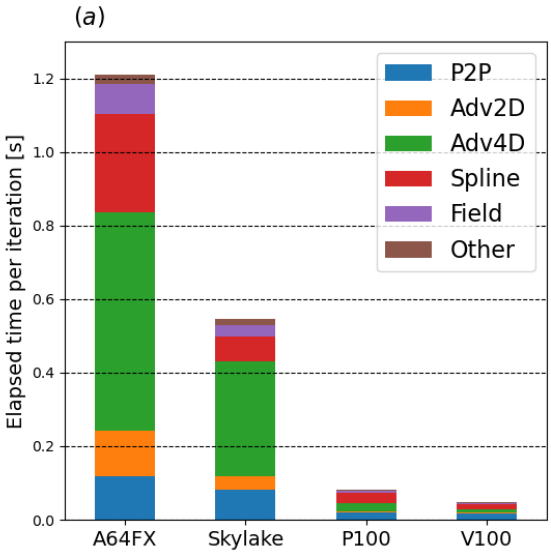


Fig. 2 The elapsed time of the kernels in mini-app with OpenMP4.5, OpenACC and Kokkos for one iteration on Nvidia V100.

In order to move forward to more complex setting, we have upgraded the

mini-app to use spline interpolation for the backward semi-Lagrangian scheme, and we have added MPI domain decomposition. Due to the increased complexity, the performance of MPI version degrades indicating that optimizations are necessary. As a case study for optimizing the performance portable version of kinetic plasma simulation codes, we have investigated the strategy to extract high performance over x86-based architecture, arm-based architecture and GPUs, while keeping the performance portability. For optimizations, we have tested directive-based SIMD tuning for Kokkos and OpenACC/OpenMP versions and View Layout tuning for the Kokkos version. After the optimizations, we have measured the performance of the mini-app on A64FX, Skylake, P100 and V100.



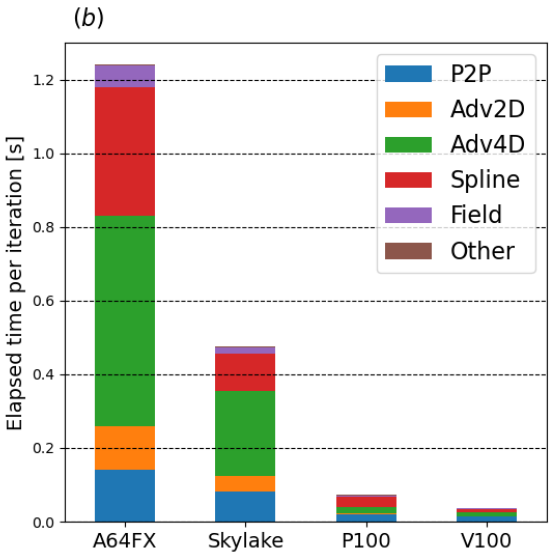


Fig. 3 The elapsed time for one iteration with (a) Kokkos and (b) OpenACC/OpenMP. The problem size is set as 128^4 and 16 MPI processes are mapped. On each supercomputer, we use 4 nodes with 4 MPI processes per node, where 2 and 4 MPI processes are mapped to a single CPU socket on Skylake and A64FX, respectively.

Fig. 3 shows the cost distribution of multiple kernels on A64FX, Skylake, P100 and V100. Here, 'P2P' denotes the Peer-to-peer communication costs of distribution function including packing, MPI communication and unpacking costs. 'Adv2D' and 'Adv4D' correspond to the 2D and 4D advection kernels with spline interpolation, which are compute intense. 'Spline' kernel refers to the total costs to construct spline coefficients. 'Field' consists of the 'Integral' kernel, MPI_Allreduce communication, and the Poisson solver with FFTs. As expected, we have obtained the better performance for computation rich kernels like Adv2D and Adv4D on P100 and V100 compared to CPUs.

We have also measured the

performance of the most time-consuming computational kernels based on the roofline analysis as shown in Tables 1 and 2. The achieved GFlops to the ideal performance are presented in the parentheses.

Table 1. Achieved performance on Skylake (8 sockets) and A64FX (4 sockets).

	Kernel	f/b	Achieved Performance GFlops	GBytes/s
A64FX (Kokkos)	adv2D	61/32	8.17 (2.14%)	4.29
	adv4D	845/32	23.94 (2.83%)	0.91
	spline 2D	18/16	2.53 (1.12%)	2.25
	integral	1/8	0.46 (1.84%)	3.69
A64FX (OpenMP)	adv2D	61/32	8.65 (2.27%)	4.54
	adv4D	845/32	24.75 (2.93%)	0.937
	spline 2D	18/16	2.07 (0.92%)	1.84
	integral	1/8	0.84 (3.37%)	6.73
Skylake (Kokkos)	adv2D	61/32	27.01 (35.43%)	14.17
	adv4D	845/32	45.57 (5.93%)	1.73
	spline 2D	18/16	16.77 (37.26%)	14.91
	integral	1/8	1.36 (27.25%)	10.90
Skylake (OpenMP)	adv2D	61/32	24.09 (31.60%)	12.64
	adv4D	845/32	61.40 (7.99%)	2.33
	spline 2D	18/16	11.06 (24.59%)	9.83
	integral	1/8	3.60 (72.03%)	28.81

Table 2. Achieved performance on P100 (16 GPUs) and V100 (16 GPUs).

	Kernel	f/b	Achieved Performance GFlops	GBytes/s
P100 (Kokkos)	adv2D	61/32	372.74 (36.21%)	195.54
	adv4D	845/32	630.1 (11.89%)	23.86
	spline 2D	18/16	23.31 (3.84%)	20.72
	integral	1/8	31.71 (46.98%)	253.72
P100 (OpenACC)	adv2D	61/32	292.1 (28.38%)	153.25
	adv4D	845/32	814.5 (15.4%)	30.8
	spline 2D	18/16	25.96 (4.27%)	23.08
	integral	1/8	53.94 (79.91%)	431.50
V100 (Kokkos)	adv2D	61/32	571.1 (35.7%)	299.62
	adv4D	845/32	1942.1 (24.90%)	73.55
	spline 2D	18/16	41.75 (4.42%)	37.11
	integral	1/8	32.53 (30.98%)	260.27
V100 (OpenACC)	adv2D	61/32	750.00 (47.8%)	393.4
	adv4D	845/32	1531.96 (19.64%)	58.01
	spline 2D	18/16	85.43 (9.04%)	75.93
	integral	1/8	73.17 (69.69%)	585.36

We have achieved good performance portability (more than 30 % to the ideal on Skylake, P100 and V100) for simpler kernels, 'Adv2D' and 'Integral'. The 'Adv4D' kernel is compute intense but hard to optimize due to the indirect memory accesses from the advection by electrostatic fields. The 'Spline 2D' kernel includes the unparallelizable operations which bothers vectorization on CPUs and reduces the parallelism on GPUs. The performance on A64FX is currently quite unsatisfactory judging from the roofline estimate. It has turned out that the Fujitsu C/C++ compiler fails to produce vectorized code if inline functions are called inside the innermost loop. Unfortunately, this happens frequently in our Kokkos and OpenACC/OpenMP implementations, both of which highly relies on the derived data structures and inline functions to access them. It is shown that both of our implementations can exploit a good performance portability on CPUs and GPUs as long as the compiler auto-vectorization is effective. This work is presented in the international workshop and conference [4, 5]. The source codes used in this work are found at the GitHub page [11].

6. Progress during FY2020 and Future Prospects

Scalable Data Analysis

As planned, we have developed a new method to extract a phase space structure from large scale data (>10 TB). In order to apply PCA on 10 TB scale data, we need to perform the incremental PCA based on Dask. With Dask, we can handle the out-of-memory data without MPI parallelization. The subproduct, Incremental PCA based on Dask, is merged to dask-ml [9]. As promised in the proposal, this is a contribution to the data science community.

In the research plan, we planned to apply PCA on the GYSELA simulation data. However, the GYSELA simulation data size is expected to be around 100 TB which is larger than our storage capacity. Thus, we have employed GT5D code whose simulation data size can be suppressed to

be around 50 TB.

In order to apply our method to GYSELA, we need to develop the in-situ version of this analysis in order to avoid storing the huge data which is not allowed in most of the latest supercomputers. This issue will be addressed in FY2021.

High Performance Computing

We have planned to develop the MPI version of the mini-app and test OpenMP4.5 for GPU offloading and tasking. As planned, we have developed the MPI version of the mini-app with mixed OpenACC/OpenMP and Kokkos and evaluated its performance on Intel Skylake, Fujitsu A64FX, Nvidia P100 and Nvidia V100. With the optimizations, both OpenACC/OpenMP and Kokkos versions exhibit good performance on each architecture except for A64FX, i.e., the good performance portability for most of the kernels. We developed the optimization strategies dedicated to each approach without violating the guiding principal "single code on many architectures."

We have initially planned to use OpenMP4.5 for GPU offloading and task level parallelizations on the MPI version of the mini-app, but it turned out that the communication and computation in the mini-app cannot be overlapped due to dependencies. Therefore, we just tested the OpenMP4.5 offloading on the non-MPI version of mini-app and compared its performance with respect to OpenACC and Kokkos implementations [4]. In order to test OpenMP4.5 offloading and task level parallelization at the same time, we need to redesign the mini-app which remains as a future work.

7. List of Publications and Presentations

(1) Journal Papers (Refereed)

[1] Yuuichi Asahi, Keisuke Fujii, Dennis Manuel Heim, Shinya Maeyama, Xavier Garbet (+), Virginie Grandgirard, Yanick Sarazin, Guilhem Dif-Pradalier, Yasuhiro Idomura, and Masatoshi Yagi, "Compressing the time series of five dimensional distribution function data from gyrokinetic simulation using Principal component analysis", Physics of Plasmas **28**, 012304 (2021).

[2] S. Maeyama, M. Sasaki, K. Fujii, T. Kobayashi, R. O. Dendy, Y. Kawachi, H. Arakawa, and S Inagaki, "On the triad transfer analysis of plasma turbulence: symmetrization, coarse graining, and directional representation", New Journal of Physics 23, 043049 (2021).

(2) Proceedings of International Conferences (Refereed)

[3] Asahi Y, Latu G (+), Grandgirard V, and Bigot J (+), "Performance Portable Implementation of a Kinetic Plasma Simulation Mini-App", In: Wienke S, Bhalachandra S., eds. Accelerator Programming Using Directive series. Springer International Publishing; 2020; Cham: 117-139.

[4] Y. Asahi, G. Latu (+), V. Grandgirard, and J. Bigot (+), "Performance portable implementation of a kinetic plasma simulation mini-app with a higher level abstraction and directives", Proceedings of SNA+MC 2020, May 2020, Japan.

(3) International conference Papers (Non-refereed)

[5] Y. Asahi, G. Latu (+), V. Grandgirard, and J. Bigot (+), "Accumulating knowledge for a performance portable kinetic plasma simulation code", IFERC GPU workshop, December 2020, online.

[6] Y. Asahi, K. Fujii, and Y. Idomura, "Phase-Space Pattern Extraction from Terabyte-Scale Plasma Turbulence Simulation Data with Principal Component Analysis", 3rd International Conference on Data Driven Plasma Science, March 2021, online.

[7] S. Maeyama, "Toward a systematic understanding of multi-scale interactions between ion and electron-scale turbulence", 4th Asia-Pacific Conference on Plasma Physics (AAPPS-DPP2020), October 2020, online (invited talk).

(4) Presentations at domestic conference (Non-refereed)

(5) Published library and relating data

[8] https://github.com/yasahi-hpc/vlp4d.

[9] Incremental PCA in dask-ml. (https://github.com/dask/dask-ml/pull/619)

[10] https://github.com/smaeyama/triadgraph

[11] https://github.com/yasahi-hpc/vlp4d_mpi.

(6) Other (patents, press releases, books and so on)