Optimization of Multiscale Fusion Plasma Simulations within the ComPat Framework

O.O. Luk[1], O. Hoenen[1], A. Bottino[1], B.D. Scott[1], D.P. Coster[1]

[1] *Max-Planck-Institut für Plasmaphysik, Boltzmannstr. 2, D-85748, Garching, Germany*

Fully simulating the impact of turbulence on the performance of fusion devices such as ITER is challenging, especially when fusion plasmas exhibit highly disparate spatio-temporal scales. Currently, there are single-scale models developed to study turbulence (gyrokinetic models) and transport (large-scale simplified models) separately. To go beyond single-scale simulations, the Computing Patterns for High Performance Multiscale Computing (ComPat) project [1] takes the component based approach to construct multiscale simulations by connecting existing single-scale models (submodels) together into a workflow. This approach has simpler algorithm and codebase, therefore each submodel is easier to validate, verify, maintain and optimize. In addition, ComPat incorporates the concept of Multiscale Computing Patterns (MCP) [2] into its framework, so that applications can run efficiently when one or several submodels require computing capabilities at the petascale.

In order to couple these existing single-scale models together, a generic and non-intrusive method was used to build a multiscale workflow. A data wrapper is implemented around a submodel, which is a physics routine to handle the inputs and outputs in the form of consistent physical objects CPOs. Around this data wrapper is the component layer that connects the wrapper to the multiscale coupling environment (e.g. Multiscale Coupling Library and Environment MUSCLE2).

In this work, the ComPat framework is utilized to build a multiscale fusion application that couples four major submodels enclosed in the component:

- TRANSP: a 1D transport model (ETS) that evolves temperature and other profiles in time
- EQUIL: a 2D fixed boundary equilibrium model (CHEASE) that updates geometrical information
- TURB: a 3D gyrofluid turbulence model (GEM) to compute heat and particle fluxes
- F2DV: a module (IMP4DV) that converts fluxes coming from turbulence submodel into transport coefficients compatible with transport submodels

Topology of such workflow is illustrated in FIG 1a, with each box represents a component of the multiscale simulation. INIT is a component that imports initial data into the workflow. Dup

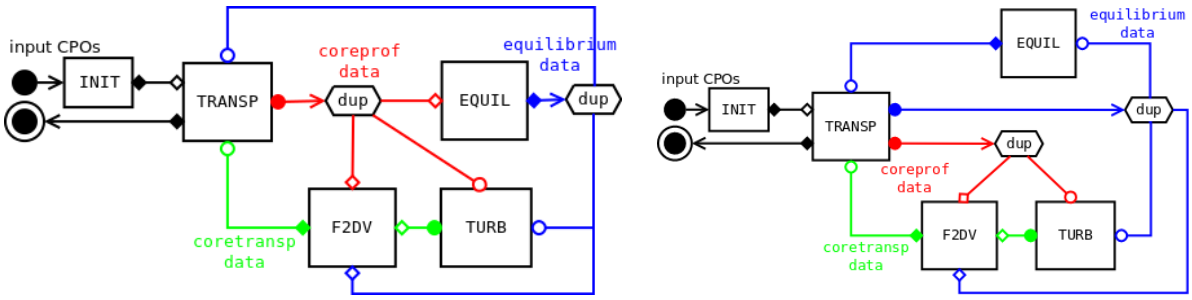


Figure 1: (a) Serial workflow vs. (b) parallel workflow, with equilibrium (EQUIL) and turbulence (TURB) components run in parallel.

is a generic duplication data mappers (a part of the MUSCLE2 standard library) that scatters data to several destinations.

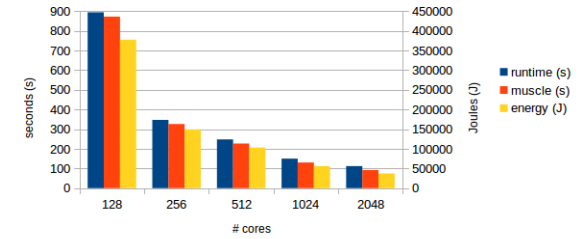
Using the unified datastructure developed in EUROfusion and the MUSCLE2, one can set up a reliable multiscale fusion plasma simulation based on existing single scale codebases. A significant advantage to such approach, is the ease with which individual submodel can be replaced by another that offers similar functionality. However, there are major challenges arise to such framework. Several of these challenges, including time bridging between turbulence and transport models, defining quasi-steady state of core plasma, implementing global gyrokinetic code into current simulation framework, and optimizing the overall simulation runtime are addressed in Luk et. al. [3]. All the simulation results are based on initial conditions from an ASDEX Upgrade experiment.

Luk et. al. demonstrate one way to optimize the overall simulation runtime. They introduce a workflow configuration that allows the EQUIL component to run concurrently with the TURB component. The topology of such parallel workflow is displayed in FIG 1b. A 4000-time-iteration simulation of the parallel workflow is compared to the serial workflow (FIG 1a). Both cases use 1024 cores on Cineca Marconi computing cluster, and the parallel workflow shows an overall runtime improvement of 5.7%. It is beneficial to study in further detail on the performance of such setup, in particular the runtime within each component of the parallel workflow, in order to run an optimal setup of multiscale workflow. One of the major goals of this project is to develop optimization schemes for every MCP to improve the overall performance of the multiscale simulations. Methods such as smart scheduling and load balancing can lead to improvement on simulation runtime, efficiency, and energy consumption on targeted execution platforms. For this work, the fusion application utilizes the extreme scaling MCP, in which one submodel (primary model) takes up the majority of the computing resources while the rest of the submodels (auxiliary models) do not require much computing resources. For this fusion

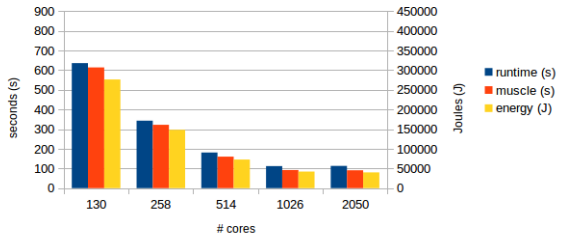
application, the turbulence model is the primary model.

In ComPat project, the Arm MAP profiling tool [4] is available to measure multiscale simulation performances. It can provide the runtime, time spent in the MUSCLE2 operations, time spent in MPI communications, and energy consumption of each component. This is very useful when analyzing various workflow setup and finding optimal workflow configurations. To demonstrate, the workflows in FIG 1 are tested on the the thin sized nodes of SuperMUC supercomputer located in Leibniz-Rechenzentrum (LRZ). There are 16 cores per node with hyperthreading technology. Each workflow is simulated under a set of resource counts: 128, 256, 512, 1024, and 2048 cores, and each simulation run completes 4 time iterations of the workflow. These resource counts are equivalent to amount of resources needed for the primary model. The runtimes are being compared for all components, and the results show that when the EQUIL (CHEASE) and TURB (GEM) run in parallel, the runtime for EQUIL component improves by 0.3% on 256 cores to 3.8% on 1024 cores, to 39.8% on 2048 cores. Similar trend is observed for the runtime on F2DV (IMP4DV) and TRANSP (ETS): 1% on 256 cores, 5-6% on 1024 cores, 38% on 2048 cores. As for the runtime for TURB, it improves by 1.1% on 256 cores, 7.9% on 1024 cores, and 12.9% on 2048 cores. The runtime comparison shows scalability to the performance improvement by core count when running EQUIL and TURB concurrently. However, there was no improvement on the performance when running on 128 and 512 cores, as a matter of fact, the performance worsen by 30-40%. This may have to do with the way the thin nodes run simulations on those core counts, and that is something to keep in mind in future simulation runs.

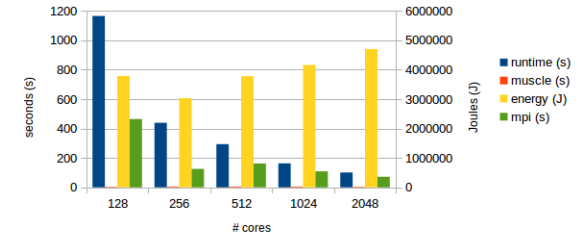
While there is an improvement on performance (e.g. runtime) for two components (EQUIL and TURB) running concurrently, the total number of cores for the simulation is equivalent to the number of cores needed for TURB (GEM). Hence, it is interesting to study the performance of the parallel workflow when more resources are allotted. In particular, the auxiliary models (ETS and CHEASE) shall receive their own resource (e.g. 1 core for each). To do a quantitative comparison, the set of runs with parallel workflow is conducted again. However, this time two additional cores are added to the total core count (e.g. simulation on 128 cores becomes 130 cores), and each additional core is given to one auxiliary model. The performance measurements on the parallel workflow, with and without the additional cores, are illustrated in FIG 2. There is improvement on runtime (blue) for every kernel of the auxiliary models. While an improvement of approximately 25-30% is observed for runs with two additional cores added to the 128-, 512-, and 1024-core runs, we see approximately 1% improvement at 256 cores. On the contrary, two additional cores results in longer runtime for 2048-core run. Again, this may have to do with



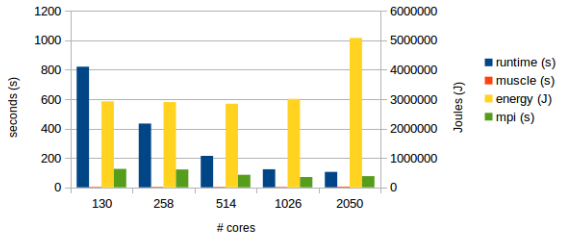
(a) CHEASE



(b) CHEASE (with 2 additional cores)



(c) GEM



(d)GEM (with 2 additional cores)

Figure 2: Figure 2: Performance measurements, including runtime (blue), time spent in the MUSCLE2 operations (orange), time spent in MPI communications (green), and energy consumption (yellow) of the EQUIL and TURB components on parallel workflow with (left) and without (right) two additional cores.

the characteristics of the SuperMUC thin nodes. As for time spent in MUSCLE2 operations (orange) and energy consumption (yellow) in every component, they are qualitatively similar to the observation for runtime. This information will be saved in a database, in order to give better statistics when searching for optimal scenario for the application.

References

[1] http://compat-project.eu

[2] S. Alowayyed, D. Groen, P.V. Coveney, and A.G. Hoekstra, Journal of Computational Science **22**, 15-25 (2017)

[3] O.O. Luk, O. Hoenen, A. Bottino, B.D. Scott, and D.P. Coster, ComPat Framework for Multiscale Simulations Applied to Fusion Plasmas, Computer Physics Communications Special Issue Numerical Plasma Simulation at the Dawn of Exascale, submitted.

[4] https://developer.arm.com