

Performance optimization of parallel applications in diverse ad-hoc development teams

Hristo Iliev¹, Marc-André Hermanns², Jens Henrik Göbbert², René Halver², Christian Terboven¹, Bernd Mohr², and Matthias Müller¹

¹ Chair for High Performance Computing, IT Center, RWTH Aachen University
{iliev,terboven,mueller}@itc.rwth-aachen.de

² Jülich Supercomputing Centre, Forschungszentrum Jülich
{m.a.hermanns,j.goebbert,r.halver,b.mohr}@fz-juelich.de

Abstract.

Current supercomputing platforms and scientific application codes have grown rapidly in complexity over the past years. Multi-scale, multi-domain simulations on one hand and deep hierarchies in large-scale computing platforms on the other make it exceedingly harder to map the former onto the latter and fully exploit the available computational power. The complexity of the software and hardware components involved calls for in-depth expertise that can only be met by diversity in the application development teams. With its model of simulation labs and cross-sectional groups, JARA-HPC enables such diverse teams to form ad hoc to solve concrete development problems. This work showcases the effectiveness of this model with two application studies involving the JARA-HPC cross-sectional group “Parallel Efficiency” and simulation labs and domain-specific development teams. For one application, we show the results of a completed optimization and the estimated financial impact of the combined efforts. For the other application, we present initial performance measurements, preliminary conclusions, and plan to describe the full optimization cycle in the final version of the paper.

1 Introduction

In the past decade, concurrency in supercomputing platforms has risen exponentially. With increasing transistor density resulting in new multi- and many-core processor node architectures and large-scale network topologies, such as multi-dimensional tori and variations of fat-tree architectures, users are faced with deep NUMA hierarchies on the node and complex network topologies. Reasoning about application performance in such these environments becomes increasingly difficult and requires more and more expertise. Homogeneous development teams comprised of scientists of the same domain easily reach the boundaries of their expertise.

Simulation codes have undergone a similar transformation in complexity. The evermore computing power available to simulation scientists have led to unprecedented detail with multi-scale, multi-domain simulations that expose a complexity inaccessible to most developers outside that specific domain. Modifications to such complex codes can often only be performed from the core developers of a specific simulation component.

From this status quo arises the need for scientific development teams with a diverse knowledge base, in order to sustain scientific productivity and code maintainability. However, such requirements are often not attainable by small to mid-sized development teams, common to university research groups. Moreover, not all expertise is needed all the time during the application life cycle. As such, much of the available expertise is left untapped during the development process. Furthermore, the synergism between domain and HPC experts is claimed to be an often overlooked means to reducing the operating costs of institutions providing computing services [2].

JARA-HPC addresses this problem by establishing two different types of research groups: (1) domain-specific research groups (simulation labs) focused on a specific domain, its approaches, and algorithms; and (2) cross-sectional groups of fields of expertise needed by all of the simulation labs. Simulation labs serve as beacons for a specific community, assisting research groups from their respective fields at RWTH Aachen University and Forschungszentrum Jülich, i.e., smaller research groups can call on a simulation labs expertise and manpower for a specific problem. Likewise, the cross-sectional groups assist simulation labs and research groups in the area of their scientific field.

2 Mission Statement

The mission of the cross-sectional group “Parallel Efficiency” is the creation and dissemination of methods and tools in the area of software engineering of massively parallel application that enable the efficient use of HPC resources. One of the core ideas is a synergistic exchange of knowledge between the cross-sectional group and developers of scientific simulation software. The former brings in the existing tools and expertise in parallel efficiency while the close collaboration leads to concrete insights on how the tools are used, how existing functionality can be improved, and which functionality is needed to investigate a specific application scenario.

2.1 Related Activities

Major US national labs implemented so-called computational endstations, such as the Climate-science Computational End Station (CCES) [13] or the Performance Evaluation and Analysis Consortium (PEAC) End Station. In comparison to the JARA-HPC cross-sectional group “Parallel Efficiency”, the latter group focuses more on the proliferation of scalable performance tool support and is less directly engaged in explicit optimization efforts of other end stations. However, end stations, such as CCES, have kept close ties to individual members of the PEAC group to drive the optimization of the community codes on large-scale systems, as in the context of the G8 project ECS, targeting climate simulations at exascale [4].

The EU Horizon 2020 center of excellence Performance Optimization and Productivity (POP) [1] targets performance analysis and optimization of third-party application codes on broader European scale. The main goal is the development of a unified set of performance metrics, a collection of structured methods and workflows for performance analysis and optimization, and a standard set of reports for presenting the results to the

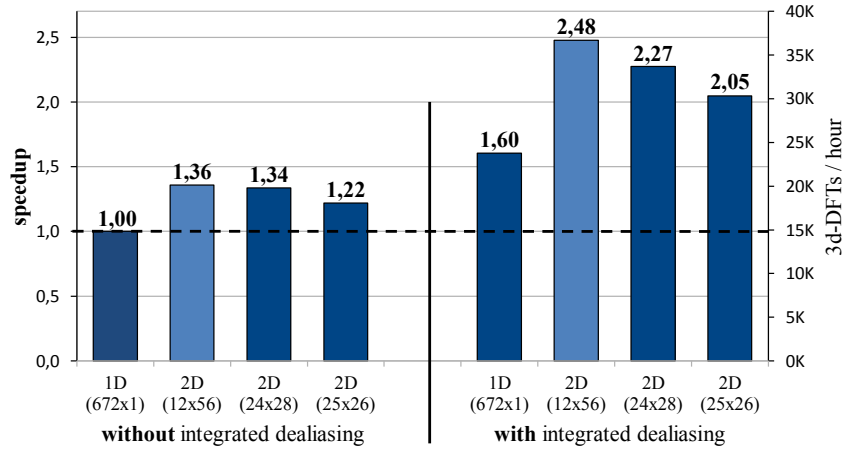


Fig. 1. Speed-up of with three different pencil (2D) domain decompositions relative to the slab (1D) decomposition for a domain of size 1024^3 . Results without (left) and with integrated dealiasing (right) are shown.

end user. POP offers several levels of involvement ranging from performance audits of the user code to proof-of-concept modifications that improve the performance and scalability of the code. The cross-sectional group shares many of POP's goals and therefore actively collaborates with it.

3 Application Case Study: psOpen

The *psOpen* simulation code [6] is a parallel solver for the Navier-Stokes equations for incompressible fluid flows with high Reynolds numbers using a pseudo-spectral direct numerical simulation (DNS) method. *psOpen* is developed by the Institute for Technical Combustion (ITV) of RWTH Aachen University. It is written in modern Fortran and is member of the High-Q Club of extremely scalable applications that run on the IBM Blue Gene/Q system JUQUEEN at the Jülich Supercomputing Centre (JSC) of Forschungszentrum Jülich.

Spectral methods [3] solve the equations in inverse space where the differential operators reduce to local multiplications. Certain non-linear terms in the Navier-Stokes equations involve field products that turn into expensive convolutions in inverse space, thus the product is computed in real space instead with multiple Fourier transforms to and from real space needed per iteration. As the point operations are pretty optimally implemented by the vectorizing compiler installed on the cluster system, the main efforts were focused on analyzing the forward and backward Fourier transforms, which take significant amount of time in each iteration. *psOpen* utilizes the open-source *P3DFFT* library [9], which implements the 3D Fast Fourier Transform (3D-FFT) and is parallelized with the Message Passing Interface (MPI). Both slab (1D) and pencil (2D) domain decompositions are supported. The library makes use of the MPI all-to-all operations to perform the global array transpose. The pencil domain decomposition

Setup	1D (672x1)	2D (12x56)
	separate dealiasing	integrated dealiasing
Hardware costs	250 000 EUR	
Electricity	0.14 EUR/kWh	
Scenario 1		
Fixed no. 3D-FFTs	100 000 000	
3D-FFTs/kWh	152	277
Electricity costs	92 105 EUR	50 525 EUR
Personnel costs	0 EUR	10 000 EUR
Savings	31 580 EUR	
Scenario 2		
Fixed lifetime	5 years	
3D-FFTs/hour	2273	4215
3D-FFTs/EUR	291	523
Efficiency improvement	+80%	

Table 1. Total cost of ownership analysis for a domain size of 2048³.

technique needs two all-to-all operations, one before the FFT in the Y direction and one before the FFT in the Z direction, while the slab decomposition needs a single all-to-all operation before the FFT in the Z direction, therefore the pencil decomposition might seem more appealing from a performance point of view. The typical simulation domain is a cube of certain number of points along each dimension and that number determines the maximum scalability in terms of MPI processes that can be used.

Performance optimization in that case was a two step process. A domain expert implemented an integration of a part of the algorithm known as spectral dealiasing (filtering off certain parts of the spectrum) into the 3D-FFT library, which significantly reduces the amount of data being communicated [7]. In addition, we used a simple network model to show that the pencil decomposition is more efficient, scales better given the same number of MPI processes, and that certain process distributions can take advantage of the specifics of the 3D-FFT algorithm and the network topology to gain an increase in performance. Based on the models, the application was adapted and the achieved speedups are shown in Figure 1. The figure also shows that the proper process distribution becomes even more relevant for the case with integrated dealiasing.

We quantified the financial impact of our efforts by performing a total cost of ownership (TCO) analysis and estimated the price of each 3D-FFT in EUR taking into account the price of the hardware used, the cost of electricity and facility maintenance, and the development costs. The results are summarized in Table 1.

Note to reviewers: The final article will include a more detailed description of the performance model and the methodology used to obtain the values in Table 1.

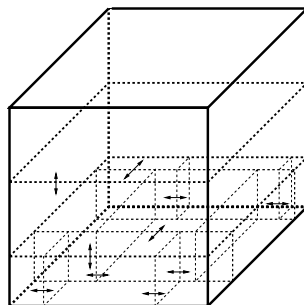


Fig. 2. Geometry of the staggered load-balance scheme: (1) the work of one Cartesian plane is collected and the border between neighboring planes adjusted according to the differences in work load between them; (2) the procedure is repeated for Cartesian columns and (3) for individual cells. This leads to a staggered grid, as the columns of neighboring planes are likely to be shifted against each other, as are the cells of neighboring columns.

4 Case Study: MP2C

The Massively Parallel Multi-Particle Collision Dynamics (MP2C) [12] simulation code is a particle based parallel solver in the field of mesoscopic hydrodynamics. It combines Multi Particle Collision Dynamics (MPC) with Molecular Dynamics (MD) to simulate solute systems, such as polymers, embedded in a fluid. The parallelization approach is based on a domain decomposition where data are exchanged via the Message Passing Interface (MPI). For data intensive applications, a highly-scalable parallel I/O library, SIONlib [5], developed at Jülich Supercomputing Centre (JSC) is available. The code is developed by the simulation laboratory *Molecular Systems* [11] at JSC and has successfully run at extreme scale on the IBM Blue Gene/Q system JUQUEEN of Forschungszentrum Jülich [10], as well as on other architectures, e.g. CRAY XT4/XT5. The 3D domain-decomposition relies on disjunct spatial sub-domains which are administered by individual processes. Due to the dynamic nature of the simulated systems, the particles are free to move between domains, which demands for data exchange between the processors. While the fluid fills the simulation box almost homogeneously, the embedded particle systems can cluster which might create load-imbalances in the MD computations. To tame the performance degradation due to load imbalance MP2C implements different load-balancing strategies to adapt to these dynamically changing work loads: (1) regular and (2) staggered domain geometries. The former implements load balancing along the regular three-dimensional Cartesian grid, while the latter hierarchically adjusts the load within the cells as is shown in Figure 2.

Both of these load-balancing strategies have their own advantages and disadvantages, e.g. the adjustment of work is better for the staggered scheme, as the cells can be fitted to clusters of work in a more optimal way than in the plain scheme. This advantage comes at the price of a much more complex communication scheme, as the the number of neighbors for each cell can increase in a non-predictable way, while for the regular scheme the number of neighbors is conserved. Figure 3 shows that the regular load-balancing scheme can achieve better results in terms of runtime than the staggered

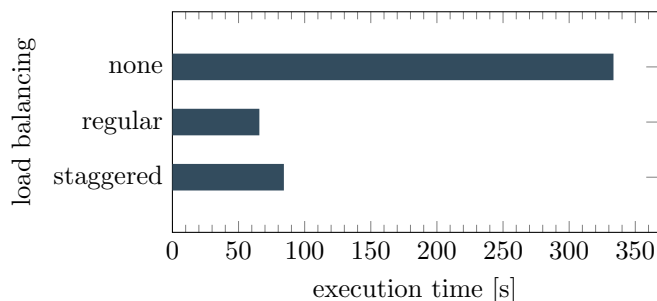


Fig. 3. Total execution time in seconds for (1) no, (2) regular and (3) staggered load balance scheme. The staggered load balancing seems to suffer from additional overheads due to its more complex communication scheme.

version for a given example of 50,000 MD particles without inclusion of the solvent. The goal of the collaboration of the cross-sectional group *Parallel Efficiency* and the simulation laboratory *Molecular Systems* is to identify the cause for this difference and to locate optimization targets in the staggered load-balancing scheme. Previous measurements indicate that the difference in quality of the schemes is related to the more complex communication scheme in the staggered version, which calls for an improvement in latency and asynchronous execution.

An initial performance screening confirmed the increase in communication complexity. Figure 4 reveals that the total bytes transferred using point-to-point communication significantly increases with the staggered load balancing. To verify the correlation of bytes transferred and the increase in communication time, we need to conduct more in-depth measurements.

Note to reviewers: To provide meaningful insights, such measurements require minimal runtime dilation and careful configuration of the measurement system, which we could not optimize in due time. We plan to present such measurements at the symposium and the final paper.

Using the measurement results obtained so far (not all shown here), we can come to the following preliminary conclusions: (1) the number of point-to-point communications increases by a factor of 2.7 from the regular to the staggered load balancing; (2) the amount of bytes transferred via point-to-point communication increases by a comparable factor; (3) the increase in both number of communications and bytes transferred correlates with an increase in execution time from the regular to the staggered scheme; (4) spacial distribution of calls to point-to-point communication in the staggered load balancing shows a much greater variation than in the regular load balancing; (5) imbalances within an iteration should not affect further iterations, but manifest as waiting time at implicit and explicit global synchronization after each iteration.

Note to reviewers: As this case study is not finished at the time of submission, a detailed analysis is planned to be presented at the symposium as well as described in the final version of the paper. It is planned to include (1) identified communication wait states, (2) their root causes, and (3) an overall assessment of optimization potential.

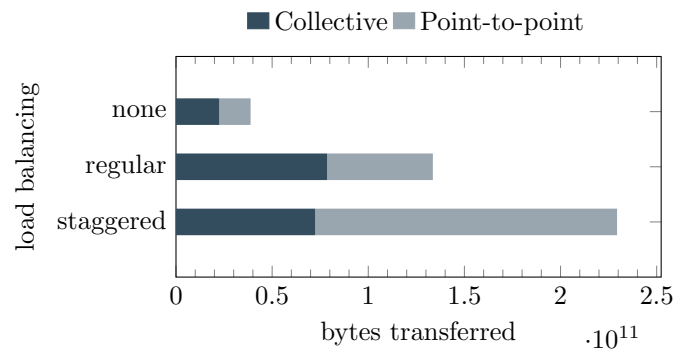


Fig. 4. Total bytes transferred by communication paradigm for (1) no, (2) regular and (3) staggered load balance scheme. The staggered load balancing exchanges significantly more bytes via point-to-point communication.

5 Conclusion

Diverse development teams are needed to meet the challenges of the ever increasing complexity of modern high-performance computing infrastructures and simulation codes. While maintaining such diverse teams over the full application life cycle is often unfeasible, drawing from a set of existing domain-specific groups to form such teams ad hoc for specific development tasks is feasible. In this work we have shown two examples of application optimization performed by such teams. We have also shown through total-cost-of-ownership analysis that funding such environments is economically justified, as their work pays off and can greatly impact the overall code efficiency in terms of cost per simulation.

Acknowledgments

The authors gratefully acknowledge the computing time granted by the JARA-HPC Vergabegremium and provided on the two JARA-HPC Partition systems – the supercomputer JUQUEEN at Forschungszentrum Jülich and the RWTH Compute Cluster at RWTH Aachen University.

References

1. CoE Performance Optimisation and Productivity. <https://pop-coe.eu/> (2016)
2. Bischof, C., anMey, D., Iwainsky, C.: Brainware for green HPC. *Computer Science - Research and Development* 27(4), 227–233 (2012), <http://dx.doi.org/10.1007/s00450-011-0198-5>
3. Canuto, C., Hussaini, M.Y., Quarteroni, A., Zang, T.A.: *Spectral methods*. Springer (2006)
4. Cappello, F., Wuebbles, D.: G8 ECS: Enabling climate simulation at extreme scale, 2012. In: *G8 Exascale Projects Workshop-12 November* (2012)

5. Freche, J., Frings, W., Sutmann, G.: High throughput parallel-I/O using SIONlib for mesoscopic particle dynamics simulations on massively parallel computers. In: Chapman, B., Desprez, F., Joubert, G.R., Lichnewsky, A., Peters, F.J., Priol, T. (eds.) *Parallel Computing: From Multicores and GPU's to Petascale*. *Advances in Parallel Computing*, vol. 19, pp. 371–378. IOS Press (2010)
6. Göbbert, J.H.: psOpen. http://www.fz-juelich.de/ias/jsc/EN/Expertise/High-Q-Club/psOpen/_node.html (2015), [Online; accessed 4-July-2016]
7. Goebbert, J.H., Gauding, M., Ansoerge, C., Hentschel, B., Kuhlen, T., Pitsch, H.: Direct numerical simulation of fluid turbulence at extreme scale with psOpen. *Advances in Parallel Computing* 27, 777–785 (2016)
8. an Mey, D., Biersdorff, S., Bischof, C., Diethelm, K., Eschweiler, D., Gerndt, M., Knüpfer, A., Lorenz, D., Malony, A.D., Nagel, W.E., Oleynik, Y., Rössel, C., Saviankou, P., Schmidl, D., Shende, S.S., Wagner, M., Wesarg, B., Wolf, F.: Score-P: A unified performance measurement system for petascale applications. In: *Proc. of the CiHPC: Competence in High Performance Computing, HPC Status Konferenz der Gauß-Allianz e.V.*, Schwetzingen, Germany, June 2010. pp. 85–97. Gauß-Allianz, Springer (2012), <http://www.springerlink.com/content/t041605372024474/?MUD=MP>
9. Pekurovsky, D.: P3DFFT: A framework for parallel computations of Fourier transforms in three dimensions. *SIAM Journal on Scientific Computing* 34(4), C192–C209 (2012)
10. Sutmann, G.: MP2C. http://www.fz-juelich.de/ias/jsc/EN/Expertise/High-Q-Club/MP2C/_node.html (2015), [Online; accessed 4-July-2016]
11. Sutmann, G.: Simulation Lab Molecular Systems. http://www.fz-juelich.de/ias/jsc/EN/AboutUs/Organisation/ComputationalScience/Simlabs/slms/_node.html (2015), [Online; accessed 4-July-2016]
12. Sutmann, G., Westphal, L., Bolten, M.: Particle based simulations of complex systems with MP2C: Hydrodynamics and electrostatics. *AIP Conference Proceedings* 1281(1) (2010)
13. Washington, W.M., Drake, J., Buja, L., Anderson, D., Bader, D.C., Dickinson, R., Erickson, D., Gent, P., Ghan, S., Jones, P., Jacob, R.L.: The use of the Climate-Science Computational End Station (CCES) development and grand challenge team for the next IPCC assessment: An operational plan (Dec 2007)