Energy-Efficient Coupled Simulations with Dynamic Voltage and Frequency Scaling

Valentina Schüller
Centre for Mathematical Sciences
Lund University
Lund, Sweden
0000-0002-4579-4217

Filip Vaverka
IT4Innovations
VSB – Technical University of Ostrava
Ostrava, Czech Republic
0000-0002-7960-5752

Ondřej Vysocký
IT4Innovations
VSB – Technical University of Ostrava
Ostrava, Czech Republic
0000-0001-7849-2744

Philipp Birken
Centre for Mathematical Sciences
Lund University
Lund, Sweden
0000-0002-6706-3634

Roman Iakymchuk
Department of Information Technology
Uppsala University
Uppsala, Sweden
0000-0003-2414-700X

***Abstract*—Dynamic inter-solver load imbalance in coupled multiphysics simulations leaves faster solvers idle at synchronization points, wasting energy. We investigate whether dynamic voltage and frequency scaling can exploit this slack without increasing time-to-solution. Our runtime framework combines preCICE for coupling and timing with MERIC for energy monitoring and online CPU-frequency control. A controller based on the Jitter algorithm maps each solver's active fraction to a target frequency, while an energy model estimates savings from activity, power, and frequency-induced slowdown. We evaluate the method on a partitioned, iteratively coupled 2D air-water heat transfer problem using DUNE on the Barbora and Karolina clusters. On Barbora, total energy-to-solution fell by 13 % with negligible runtime overhead, while the faster air solver used 28 % less energy. On Karolina, solver runtime is dominated by communication overhead, causing DVFS to unnecessarily slow the simulation and significantly increase energy-to-solution.**

***Index Terms*—energy efficiency, coupled solvers, dynamic load imbalance, preCICE, MERIC**

I. Introduction

We investigate runtime energy savings in coupled scientific codes without compromising time-to-solution or accuracy. We focus on time-dependent multiphysics applications, such as climate models, whose components often have low arithmetic intensity (AI) and are therefore memory-bound. These components synchronize at regular intervals, referred to as time windows.

Coupled simulations exhibit both static and dynamic inter-component load imbalance. At the end of a time window, faster components may wait for slower ones. Static imbalance can be mitigated using automatic load-balancing tools, e.g., [1], whereas dynamic imbalance persists due to nonlinearities, varying I/O behavior, and adaptive numerical methods.

Dynamic voltage and frequency scaling (DVFS) reduces processor power and slows execution once frequency becomes performance-limiting. It can therefore exploit waiting time to reduce energy-to-solution without increasing overall time-to-solution [2]. The recent study [3] demonstrated this potential for static load balancing. Additionally, GEOPM [4] is a runtime system that uses DVFS to balance load across nodes in a job allocation. In this work, we extend the idea to dynamic imbalance in coupled simulations. We introduce a simple model that estimates the energy-saving potential and show how to implement the strategy at runtime.

II. Preliminaries

A. Inter-Solver Load Balance

Consider N solvers $\mathcal{S}_1, \ldots, \mathcal{S}_N$ with time-to-solution $t_1, \ldots, t_N$, *excluding coupling overhead*. We define inter-solver load balance as

$$b = \frac{\langle t \rangle}{t_{\max}}, \tag{1}$$

where

$$t_{\max} = \max_{i \in \{1,\ldots,N\}} t_i, \quad \langle t \rangle = \frac{1}{N} \sum_{i=1}^{N} t_i.$$

Thus, $b \in \left(\frac{1}{N}, 1\right]$, with $b = 1$ indicating perfect balance. The active fraction of each solver:

$$c_i = \frac{t_i}{t_{\max}}, \tag{2}$$

where $c_i \in (0, 1]$, and $c_i = 1$ for the slowest solver.

A low b indicates underutilized resources. Time-to-solution-based load balancing either allocates more resources to slow solvers or removes resources from fast ones [5]. DVFS follows the latter principle from an energy perspective: It reduces the CPU frequency of fast solvers, increasing their runtime while ideally leaving $t_{\max}$ unchanged. Increasing b then serves as a proxy for reducing energy-to-solution.

B. Expected Energy Gains

For a fast solver $\mathcal{S}_i$, baseline energy at frequency f_{high} is

$$E_{\text{base},i} = P(f_{\text{high}})t_i + (t_{\max} - t_i)P_{\text{idle}}, \tag{3}$$

where $P(f)$ is processors power consumption while running the solver and P_{idle} is idle power. Lowering the frequency to f_{low} changes both power and runtime by a penalty factor $\alpha(f)$:

$$E_{\text{low},i} = P(f_{\text{low}})\alpha(f_{\text{low}})t_i + P_{\text{idle}}(t_{\max} - \alpha(f_{\text{low}})t_i). \quad (4)$$

The power and time penalty responses depend on the hardware and solver AI. Platform measurements, therefore, estimate potential savings for different active fractions c_i, see Fig. 1. The solver has a low AI of 0.02 FLOPs/Byte, obtained by [6] on Barbora, so frequency reduction can save energy with little time-to-solution penalty even without imbalance. The potential benefit increases for (more) load-imbalanced simulations.

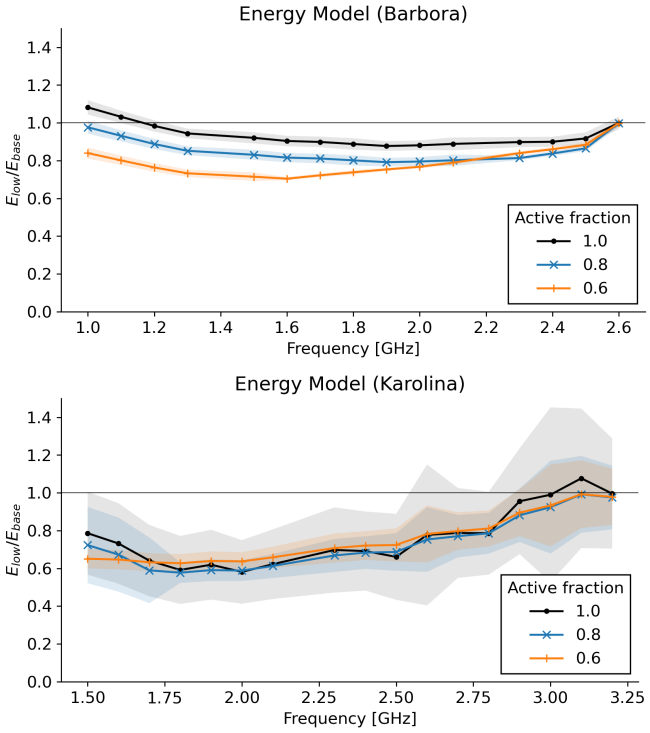


Fig. 1. Energy savings for the DUNE heat equation solver at different active fractions c_i, based on MERIC measurements on Barbora (Intel Cascade Lake) and Karolina (AMD Zen2).

III. Software Stack

We estimate b and c_i at runtime from preCICE measurements [7]. MERIC provides energy measurements and CPU-frequency control [8]. We extended MERIC with an online interface for real-time monitoring and reconfiguration. The Jitter algorithm [9] maps each active fraction obtained from preCICE to a target CPU frequency. At every coupling iteration, MERIC adjusts the frequency by one step to reduce imbalance.

IV. Experimental Results

We solve coupled 2D heat equations for air and water using a partitioned, parallel iterative scheme [10]. Both solvers advance concurrently over coupling windows of size Δt_{cpl}, exchange data, and repeat until convergence. We choose a final time of 30 s, $\Delta t_{\text{cpl}} = 10$ s, and five iterations per window.

The partial differential equations are discretized using linear finite elements in space and implicit Euler in time, implemented with DUNE [11]. Linear systems are solved using GMRES with SOR preconditioning. Both solvers use $\Delta t_i = 0.1$ s, with $n_x = n_y = 500$ unknowns for air and $n_x = n_y = 600$ for water to create an imbalance.

We run experiments on the Barbora and Karolina clusters at IT4Innovations, equipped with Intel Cascade Lake and AMD Zen2 CPUs, respectively. Each run uses two nodes, one solver per node, and hybrid parallelism: 18 MPI processes on Barbora or 64 on Karolina, with two OpenMP threads per process.

Tables I and II report time-to-solution (TtS), b, and energy-to-solution (EtS), with five runs per setup. On Barbora, the method reduces total EtS by 13 % with negligible time-to-solution overhead. The mean active fraction of the faster air solver is increased from 64 % to 74 %, corresponding to a load balance improvement of five percentage points. These results confirm that runtime DVFS can reduce energy consumption while preserving time-to-solution.

On the more powerful Karolina system, the two solvers are more load-balanced for this problem size from the outset; time-to-solution is dominated not by the linear solves, but by communication and solver setup overhead. The number of unknowns per time step and core drops from around 7×10^3 on Barbora to 2×10^3 on Karolina for $n_x = n_y = 500$, while time-to-solution more than doubles. In this case, the frequency adjustments from the Jitter algorithm do not directly lead to energy savings (cf. the large spread in the energy response in the right of Fig. 1). Instead, they create significant time-to-solution and energy-to-solution overheads.

TABLE I
Results on the Barbora cluster.

	TtS [s]	**Mean b**	**EtS [kJ] (air solver)**	**EtS [kJ] (total)**
Baseline	180 ± 1	0.82	37.0 ± 0.2	77.6 ± 0.3
Jitter	181 ± 2	0.87	26.6 ± 0.2	67.4 ± 0.5

TABLE II
Results on the Karolina cluster.

	TtS [s]	**Mean b**	**EtS [kJ] (air solver)**	**EtS [kJ] (total)**
Baseline	386 ± 5	0.92	110 ± 2	223 ± 4
Jitter	468 ± 1	0.89	138 ± 0	279 ± 1

Acknowledgment

This work was partially supported by the Swedish e-Science (eSSENCE) research program. The compute resources were provided by the EuroHPC Development Access grant, EHPC-DEV-2025D02-126. This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).

References

[1] S. Palomas, M. C. Acosta, G. Utrera, and E. Tourigny, "Reducing time and computing costs in EC-Earth: An automatic load-balancing approach for coupled Earth system models," *Geosci. Model Dev.*, vol. 18, no. 12, pp. 3661–3679, 2025, ISSN: 1991-959X. DOI: 10.5194/gmd-18-3661-2025.

[2] J. Corbalan, L. Alonso, J. Aneas, and L. Brochard, "Energy Optimization and Analysis with EAR," in *2020 IEEE International Conference on Cluster Computing (CLUSTER)*, Sep. 2020, pp. 464–472. DOI: 10.1109/CLUSTER49012.2020.00067.

[3] J. van Dijk, G. Zavodszky, A.-L. Varbanescu, and A. Pimentel, "Embracing Load Imbalance for Energy Optimizations: A Case-Study," in *2025 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, Jun. 2025, pp. 405–412. DOI: 10.1109/IPDPSW66978.2025.00066.

[4] J. Eastep et al., "Global extensible open power manager: A vehicle for hpc community collaboration on co-designed energy management solutions," in *High Performance Computing*, J. M. Kunkel, R. Yokota, P. Balaji, and D. Keyes, Eds., Cham: Springer International Publishing, 2017, pp. 394–412, ISBN: 978-3-319-58667-0.

[5] M. Garcia, J. Labarta, and J. Corbalan, "Hints to improve automatic load balancing with LeWI for hybrid applications," *J. Parallel Distrib. Comput.*, vol. 74, no. 9, pp. 2781–2794, Sep. 2014, ISSN: 0743-7315. DOI: 10.1016/j.jpdc.2014.05.004.

[6] A. Ilic, F. Pratas, and L. Sousa, "Cache-aware Roofline model: Upgrading the loft," *IEEE Comput. Archit. Lett.*, vol. 13, no. 1, pp. 21–24, Jan. 2014, ISSN: 1556-6064. DOI: 10.1109/L-CA.2013.6.

[7] G. Chourdakis et al., "preCICE v2: A sustainable and user-friendly coupling library," *Open Res. Europe*, vol. 2, no. 51, 2022. DOI: 10.12688/openreseurope.14445.2.

[8] O. Vysocky, M. Beseda, L. Říha, J. Zapletal, M. Lysaght, and V. Kannan, "MERIC and RADAR Generator: Tools for Energy Evaluation and Runtime Tuning of HPC Applications," in *High Performance Computing in Science and Engineering*, Springer, Cham, 2018, pp. 144–159, ISBN: 978-3-319-97136-0. DOI: 10.1007/978-3-319-97136-0_11.

[9] V. W. Freeh, N. Kappiah, D. K. Lowenthal, and T. K. Bletsch, "Just-in-time dynamic voltage scaling: Exploiting inter-node slack to save energy in MPI programs," *Journal of Parallel and Distributed Computing*, vol. 68, no. 9, pp. 1175–1185, Sep. 2008, ISSN: 0743-7315. DOI: 10.1016/j.jpdc.2008.04.007.

[10] B. Rüth, B. Uekermann, M. Mehl, P. Birken, A. Monge, and H.-J. Bungartz, "Quasi-Newton Waveform Iteration for Partitioned Surface-Coupled Multiphysics Applications," *Int. J. Numer. Methods Eng.*, vol. 122, no. 19, pp. 5236–5257, 2021, ISSN: 1097-0207. DOI: 10.1002/nme.6443.

[11] P. Bastian et al., "The DUNE framework: Basic concepts and recent developments," *Comput. Math. Appl.*, vol. 81, pp. 75–112, Jan. 2021. DOI: 10.1016/j.camwa.2020.06.007.