Technical University of Denmark

Computational Hydrodynamics: How Portable and Scalable Are Heterogeneous Programming Paradigms?

Pawlak, Wojciech ; Glimberg, Stefan Lemvig; Engsig-Karup, Allan Peter

Publication date:
2015

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Pawlak, W., Glimberg, S. L., & Engsig-Karup, A. P. (2015). Computational Hydrodynamics: How Portable and Scalable Are Heterogeneous Programming Paradigms?. Poster session presented at SIAM Conference on Computational Science and Engineering (SIAM CSE 2015), Salt Lake City, Utah, United States.

DTU Compute
Department of Applied Mathematics and Computer Science

Computational Hydrodynamics: How Portable and Scalable Are Heterogeneous Programming Paradigms?

Wojciech Pawlak[1], Stefan L. Glimberg[2], Allan P. Engsig-Karup[1]

[1] Technical University of Denmark (DTU) [2] Lloyds Register Consulting

Motivation

New many-core era applications at the interface of mathematics and computer science adopt modern parallel programming paradigms and expose parallelism through proper algorithms. We present new performance results for a novel massively parallel free surface wave model suitable for advanced simulations in arbitrary size Numerical Wave Tanks.

The application has already been studied in a series of works (see References) and is demonstrated to exhibit excellent performance portability and scalability using hybrid MPI-OpenCL/CUDA. Furthermore, it can be executed on arbitrary heterogeneous multi-device system sizes from desktops to large HPC systems such as superclusters and in the cloud utilizing heterogeneous devices like multi-core CPUs, GPUs, and Xeon Phi coprocessors.

The numerical efficiency is evaluated on heterogeneous devices like multi-core CPUs, GPUs and Xeon Phi coprocessors to test the performance with respect to both portability and scalability.

This study contributes to investigating the potential of code acceleration for reducing turn-around times of industrial CFD applications on heterogeneous hardware.

Contributions

- Massively parallel implementation of an advanced free surface wave model to enable fast engineering analysis and large-scale simulations of ocean waves.
- Accelerated computing. Evaluation of parallel programming paradigms with respect to performance and scalability across all major architectures using the GPUlab library for solving partial differential equations on heterogeneous many-core hardware.

Deterministic formulation

An advanced free surface model for description of nonbreaking irrotational ocean waves is this fully nonlinear and dispersive wave potential model. Dynamic and kinematic free surface boundary conditions are

$$\partial_t \zeta = -\nabla\zeta \cdot \nabla\tilde{\phi} + \tilde{w}(1 + \nabla\zeta \cdot \nabla\zeta),$$
$$\partial_t \tilde{\phi} = -g\zeta - \frac{1}{2}\left(\nabla\tilde{\phi} \cdot \nabla\tilde{\phi} - \tilde{w}^2(1 + \nabla\zeta \cdot \nabla\zeta)\right),$$

where $\tilde{\phi} = \phi(x, \zeta, t)$, $\zeta(\mathrm{x}, t)$ and $\tilde{w} = \partial_z\phi|_{z=\zeta}$ are free surface quantities and g gravitational acceleration. A Laplace problem needs to be solved

$$\begin{aligned} \phi &= \tilde{\phi}, & z &= \zeta(\mathrm{x}, t), \\ \nabla^2\phi + \partial_{zz}\phi &= 0, & -h &\le z < \zeta(\mathrm{x}, t), \\ \partial_z\phi + \nabla h \cdot \nabla\phi &= 0, & z &= -h. \end{aligned}$$

from which closure is obtained by $(\mathrm{u}, w) = (\nabla, \partial_z)\phi$.

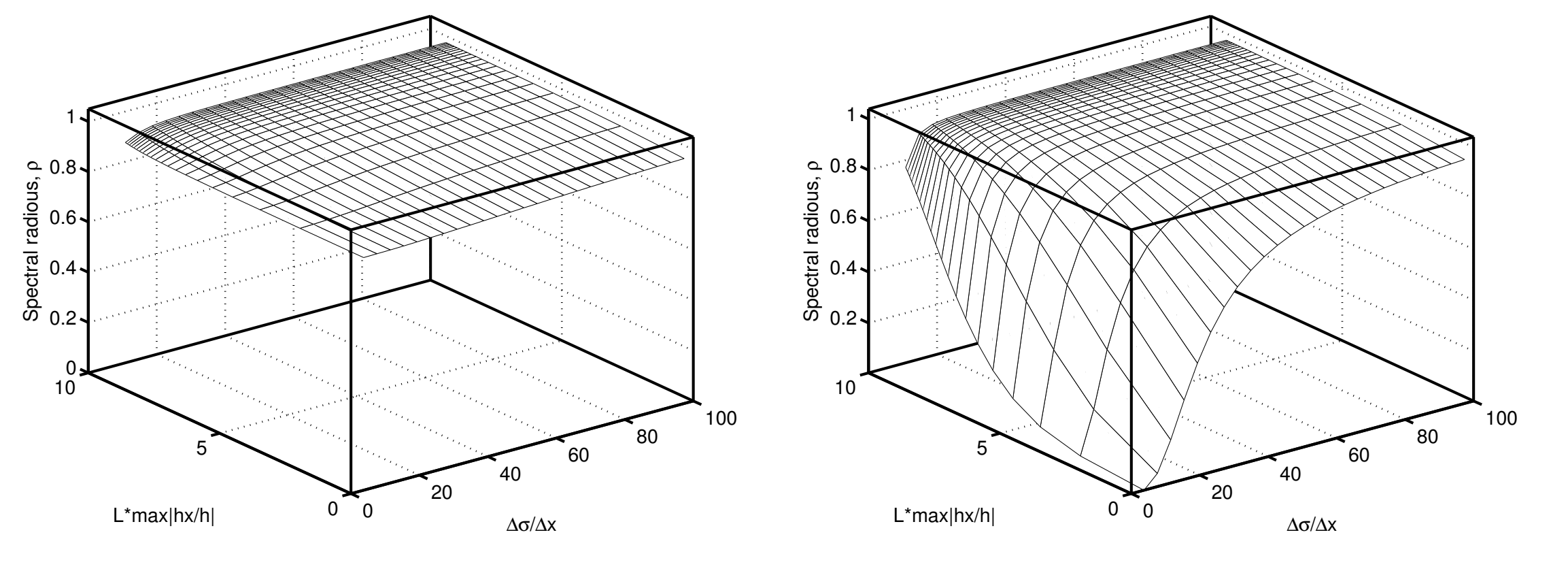


Figure 1 : *Using a multigrid PDC method, the eigenvalues of the iteration matrix confirms robustness in numerical experiments using smoothers; (a) Point-wise Gauss-Seidel, GS and (b) Block Gauss-Seidel, ZL-GS.*

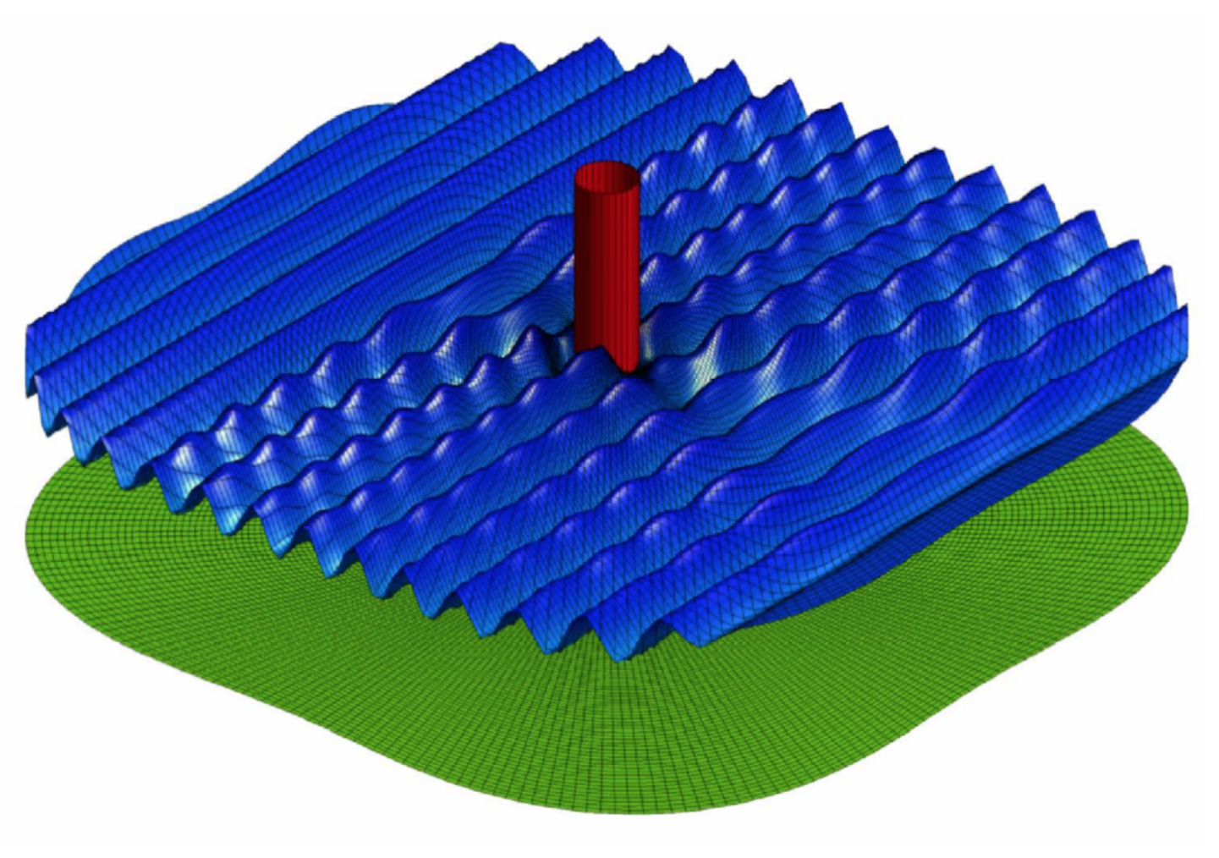


Figure 2 : *Snap shot of diffracted deterministic wave field about a cylinder.*

Iterative Preconditioned Defect Correction Method

To enable scalable massively parallel computations, an iterative preconditioned defect correction method can be used to generate a sequence of iterations, starting from an initial guess $\Phi^{[0]}$ by using the following short recurrence

$$\Phi^{[k]} = \Phi^{[k-1]} + \mathcal{M}^{-1}\mathrm{r}^{[k-1]},$$
$$\mathrm{r}^{[k-1]} = \mathrm{b} - \mathcal{A}\Phi^{[k-1]}, \quad k = 1, 2, \ldots,$$

until a given convergence criterium is satisfied.

The PDC method becomes scalable by using a multigrid preconditioner, where the ideas of nested grids and defect corrections are combined and smoothers $\mathcal{S}$ are introduced for ensuring convergence. A two-level multigrid method can be stated as

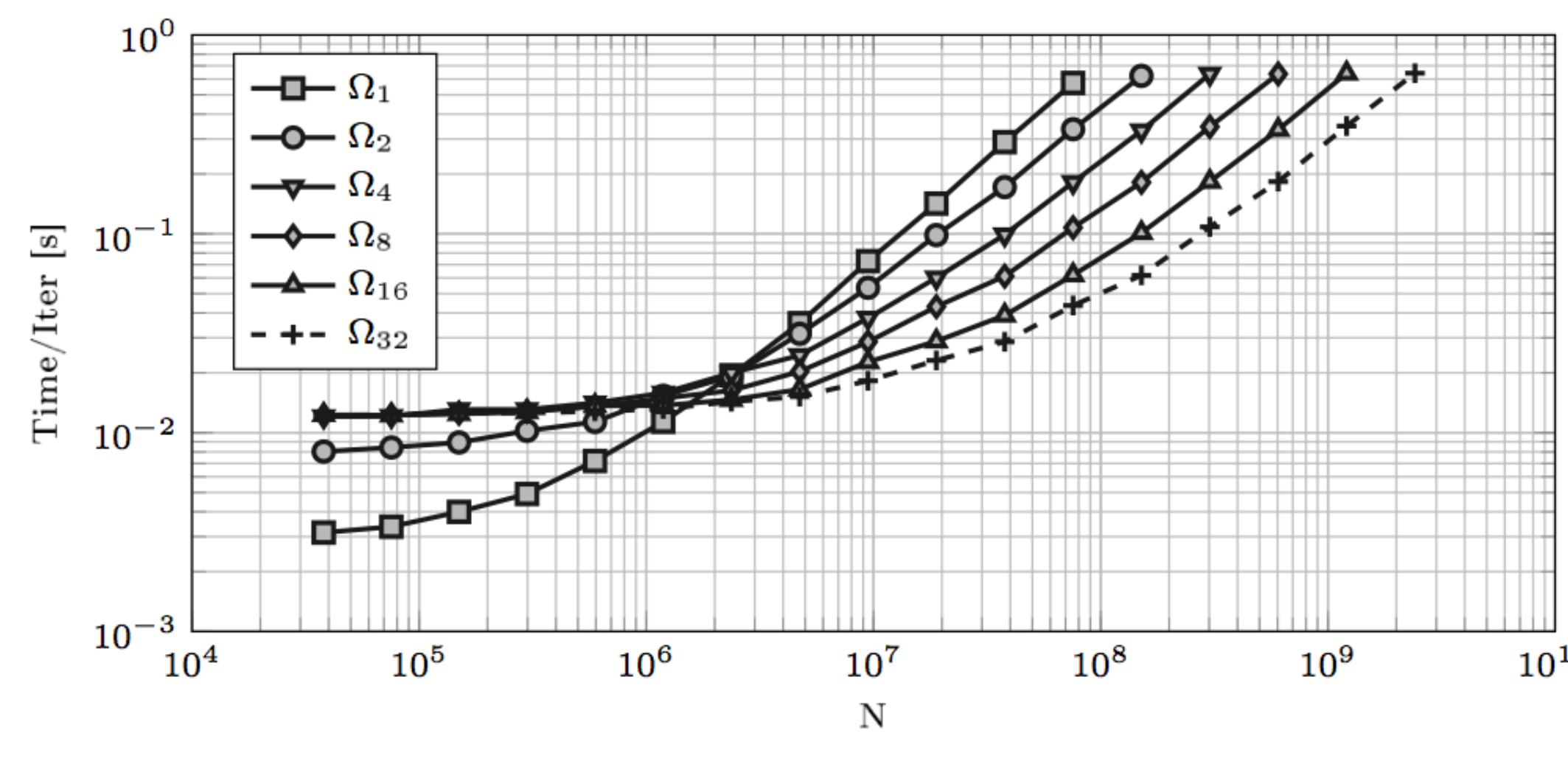
$$\Phi_k = \Phi_{k-1} + M^{-1}r_{k-1},$$
$$\mathcal{M} = S^{\mu_2}\left(\mathcal{I} - \mathcal{P}^h_{2h}\left(\mathcal{M}^{2h}\right)^{-1}\mathcal{R}^{2h}_h(\mathcal{M}^h)^{-1}\mathcal{A}\right)S^{\mu_1}$$

where $\mathcal{P}^h_{2h}$ and $\mathcal{R}^{2h}_h$ are prolongation and restriction operations, respectively.

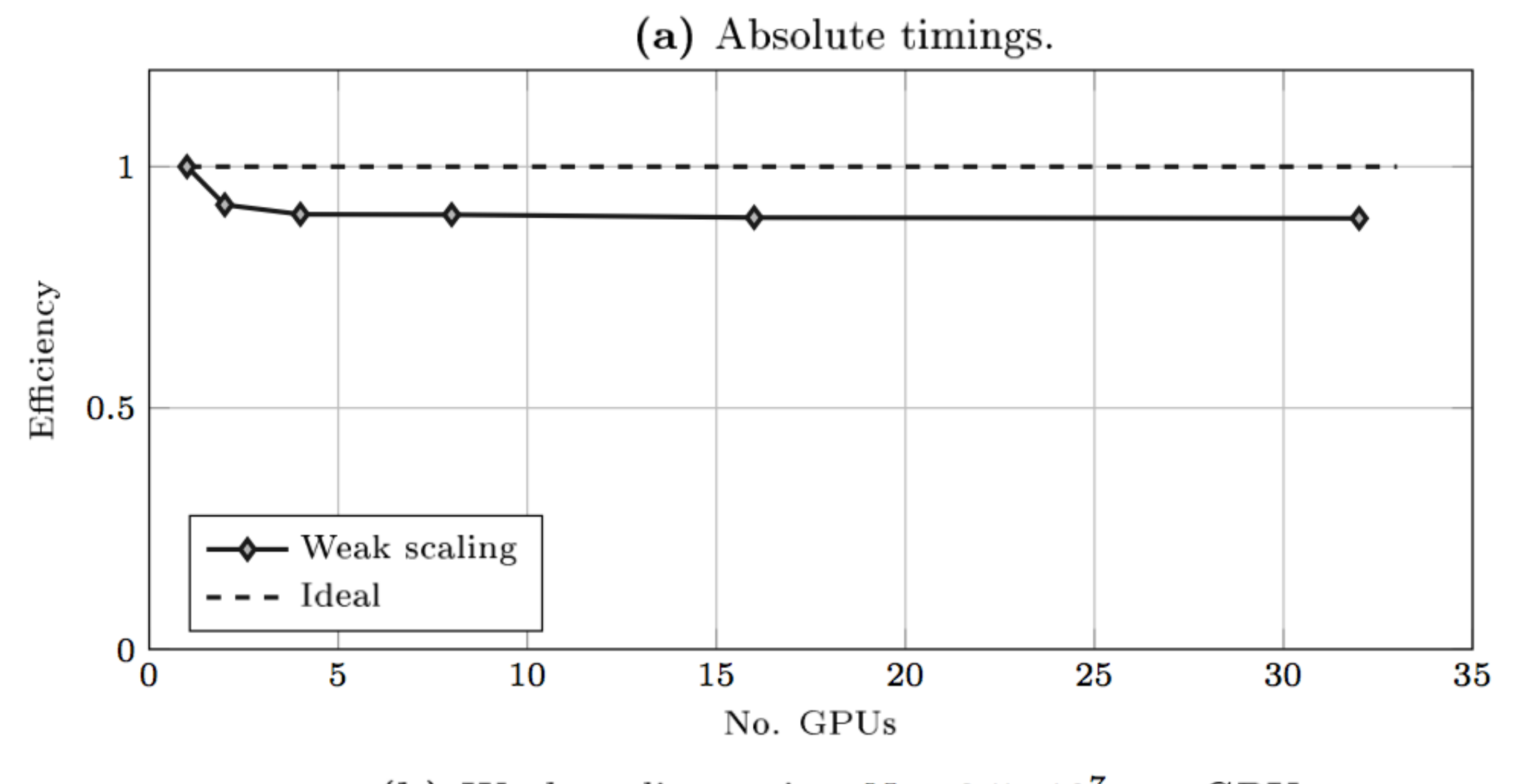
The operator

$$\mathcal{I} - \mathcal{P}^h_{2h}\left(\mathcal{M}^{2h}\right)^{-1}\mathcal{R}^{2h}_h(\mathcal{M}^h)^{-1}\mathcal{A}$$

is the *coarse grid correction* operator responsible for global coupling ($(\mathcal{M}^{2h})^{-1}$ may be a full matrix) and dictates the need for global communication. Without global communication the number of iterations may grow with number of subdomains for the outer iterative solver and the strategy will not be scalable.



(a) Absolute timings.

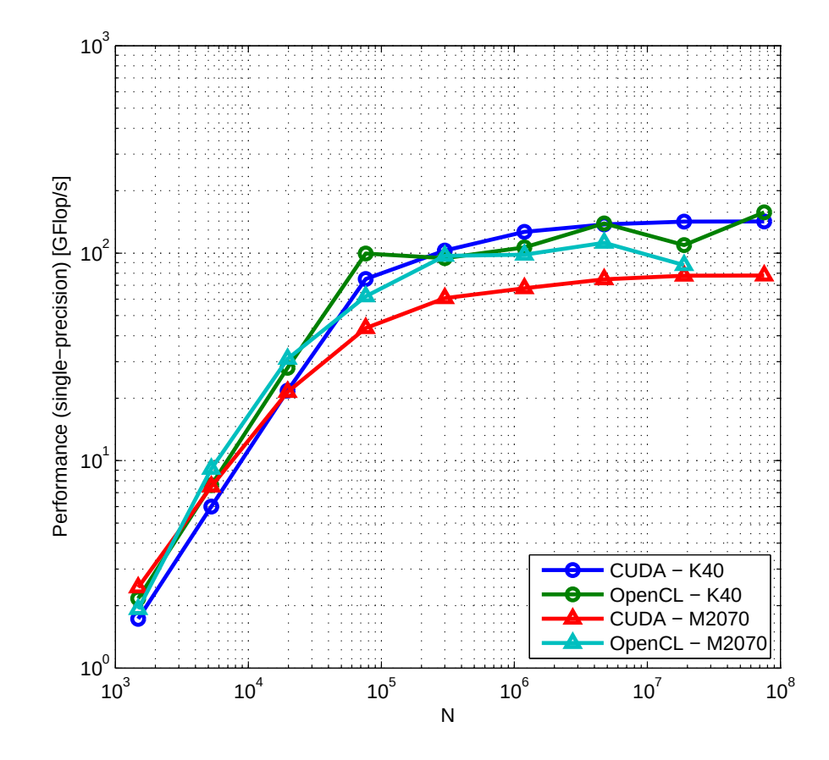


(b) Weak scaling, using $N \approx 3.5 \cdot 10^7$ per GPU.

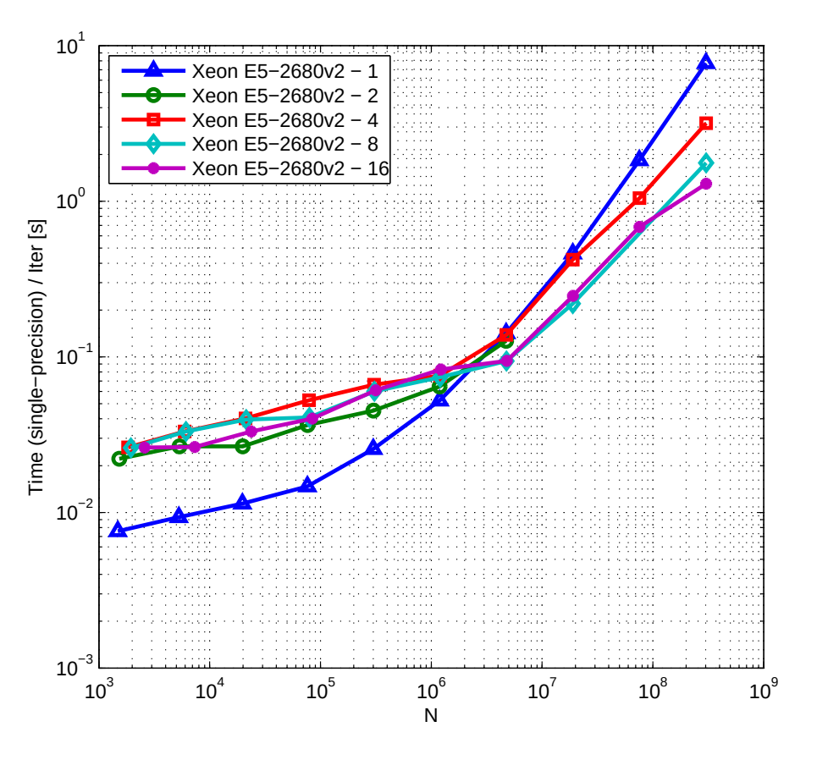
Figure 3 : *Performance scaling for the defect correction iteration, on STAMPEDE, single-precision.*

On scalability across architectures

Experiments proved that it is possible to achieve performance portability with OpenCL programming model over various architectures, types and generations. OpenCL implementation gave satisfactory results on par with or better than the algorithmically-identical CUDA implementation. The goal was to avoid architecture-specific optimization, but rather apply autotuning techniques that adapt to the given device during simulation runtime. Scalability was achieved using domain decomposition between nodes through MPI and OpenCL for heterogeneous devices on node. The final result is more clearly observable between GPUs and CPUs as optimal Xeon Phi performance needs more multithreading and vectorization optimizations.



(a) CUDA vs. OpenCL on NVIDIA hpc-oriented devices. Performance in GFlop/s. High-order residual kernel. Single precision.



(b) OpenCL on multi Xeon E5-2680v2. Execution time in ms. One DC iteration. Single precision.

Figure 4 : *Portability and scalability experiment results.*

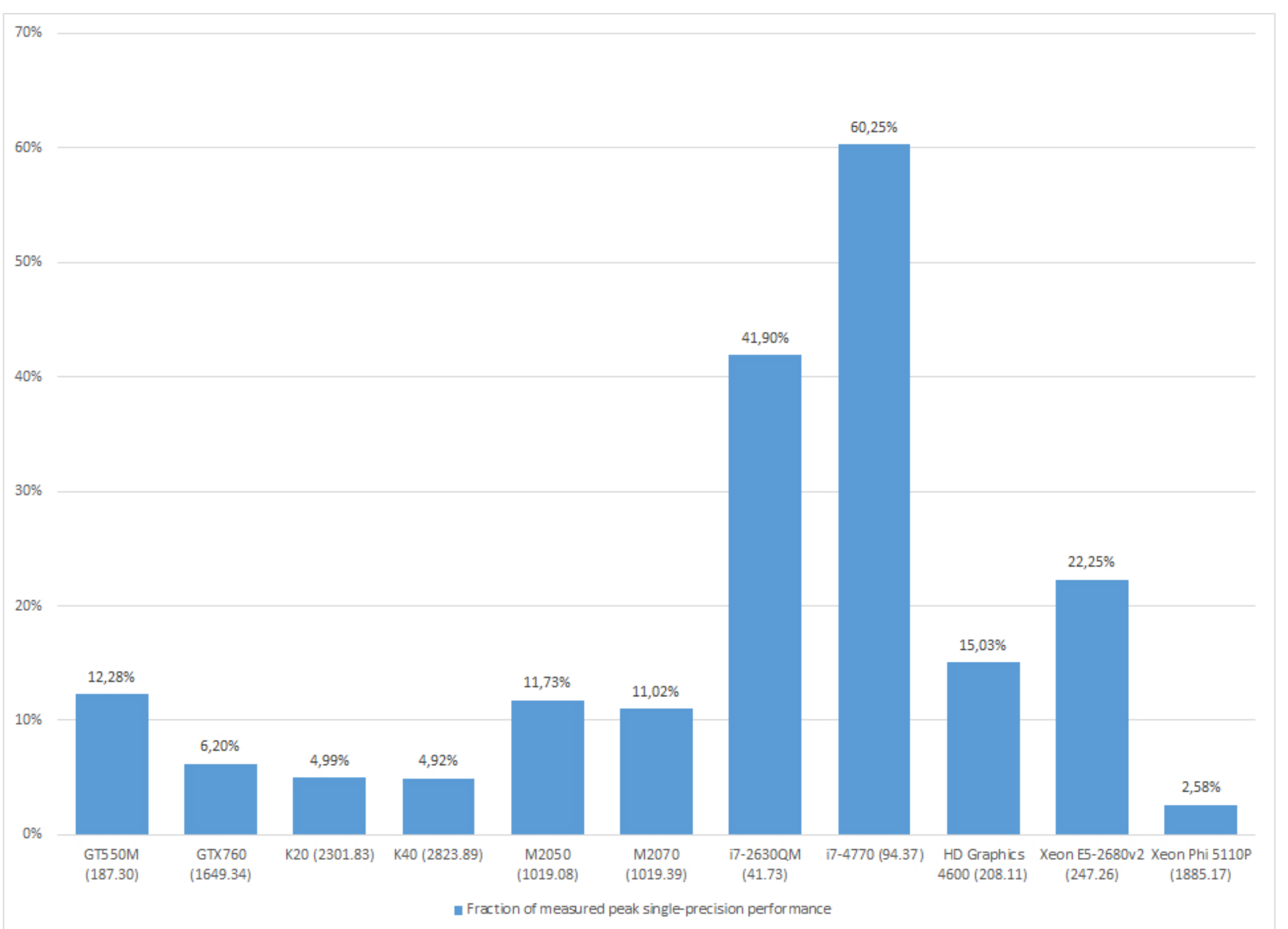


Figure 5 : *Sustained fraction of peak performance in % for all devices. Single precision.*

References

- Engsig-Karup, A. P., Glimberg, L. S., Nielsen, A. S. and Lindberg, O. 2013. Fast hydrodynamics on heterogenous many-core hardware. Part of: Raphäel Couturier (Ed). Designing Scientific Applications on GPUs, 2013, CRC Press / Taylor & Francis Group.
- Engsig-Karup, A. P., Madsen, M. G. and Glimberg, S. L. A massively parallel GPU-accelerated model for analysis of fully nonlinear free surface waves. E-published in *International Journal for Numerical Methods in Fluids*, July, 2011.
- Engsig-Karup, A.P., Bingham, H.B. and Lindberg, O. 2009 An efficient flexible-order model for 3D nonlinear water waves. *Journal of Computational Physics*, 288, pp. 2100–2118.

Contacts: Wojciech: wojciechpawlak@gmail.com, Stefan: Stefan.Glimberg@lr.org, Allan: apek@dtu.dk