Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2017-150-RC1, 2017

Geoscientific Model Development Discussions

Interactive comment on "Portable Multi- and Many-Core Performance for Finite Difference Codes; Application to the Free-Surface Component of NEMO" *by* Andrew Porter et al.

Anonymous Referee #1

Received and published: 18 August 2017

General summary and comments: This is a well written paper addressing the very real problem of performance portability of modeling codes with rapidly changing computer architectures. The paper describes the PSyKAl approach, separating the main algorithm, computational kernels, and parallel systems from each other, allowing domain scientists to focus on only the algorithm and kernels, and the computational scientist to optimize the parallel systems. In order to test this approach, the authors introduce the NEMOLite2D model, a small 2D finite-difference ocean model. The authors then thoroughly describe the code transformations needed to rewrite NEMOLite2D using the PSyKAl approach, which reduces to a single call to the PSy layer per timestep, where the PSy layer executes 8 pointwise update kernels at each gridpoint, and kernels for

each boundary point. Then, the paper details all of the optimizations performed on the PSy layer, citing evidence for each based on their previous work, adding further performance improvements via OpenMP or OpenACC parallelization directives. Finally, the paper surveys a large number of performance results, obtained using 3 different compilers for each stage of optimization, concluding that the PSyKAl approach allowed for a wide range of performance and parallelization opportunities without modification of the algorithm or kernel code.

Specific comments: 1. The optimization and performance sections are very well done, however, I would like to see one additional data point in the parallel performance section: all of the optimizations and directives applied to the original NEMOLite2D code. This will allow the reader to see if much or any performance is being left on the table by using the PSyKAl approach. 2. The title of the paper suggests the PSyKAl approach is limited to finite-difference models, yet there is barely any mention of this limitation in the paper. Can you describe what causes this limitation or if there will be attempts to extend the idea to more complex finite-element or finite-volume models? 3. I fully agree that a separation of concerns is necessary to achieve portable performance, however it is not clear to me how significantly different PSyKAl is from similar ideas, such as OCCA (http://libocca.org) where kernels are written in a simple C or Fortran based kernel language and translated into OpenMP, CUDA or OpenCL and invoked similarly to a CUDA kernel. Also, time-stepping research has long assumed that the spatial domain is computed using a single function call which iterates over all of points and executes the appropriate kernels. Could the authors be more specific about the contributions which the PSyKAl approach makes, citing specific differences from existing programming models, like CUDA and OpenMP? 4. The paper briefly mentions that generation of the PSy layer will be automated in the future. Perhaps the authors could emphasize this more (as part of any changes made #3), as this paper describes a number of lessons learned which will be applied to work on automation.

2017.

C3