Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2017-150-AC1, 2017

Geoscientific Model Development Discussions
Open Access
EGU

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.

Andrew Porter et al.

andrew.porter@stfc.ac.uk

Received and published: 6 October 2017

The authors thank the reviewers for their detailed feedback on our paper. We address the points raised below, beginning with the first (anonymous) referee and then moving on to those from Carlos Osuna.

Anonymous referee:

1/ The reviewer is right to be concerned with seeing how much performance may be lost by the re-structuring required for PSyKAl. However, the original form of the NEMO-Lite2D code that we started with for this work was unoptimised. This is emphasised

Printer-friendly version

Discussion paper

by the results in Figure 4 which show that, for the majority of cases, the final PSyKAl version was significantly faster. In contrast, our previous paper (Porter et al, 2016) tackled the "Shallow" code which has had optimisations applied to it over a period of some twenty years and was therefore a much more demanding test. There, the initial PSyKAl re-structuring did significantly harm performance. However, we showed that it was possible to recover the performance of the optimised original by applying code transformations that were PSyKAl-compliant. We therefore do not think that showing results for the optimisations applied to the original form of NEMOLite2D is necessary.

2/ The PSyKAl approach is in fact applicable to both Finite Element and Finite Difference codes (and in fact, LFRic, the largest project making use of it is using Finite Elements). We have updated the title to make this clear.

3/ A new section, "Related Approaches", has been added to the paper. In this we bring together a comparison of various other approaches including OCCA. This makes explicit the differences and makes clearer the advantages of PSyKAl.

4/ We also use the new "Related Approaches" section to emphasise that although the tool to generate the PSy layer exists ("PSyclone"), the current paper is about code structuring.

Carlos Osuna's general review:

We have added a few key lines of source code from the Momentum kernel in section 3.3. We have also added text that explains that the apparently poor performance of this kernel is down to the number of division operations. On the Ivy Bridge, division operations require at least eight times as many cycles as a multiplication which can be very significant.

The suggestion of a "Related Work" section is a good one and we have added this just after the Introduction.

We attribute the better performance of the SSE version of the code to the fact that the

poor SIMD efficiency of the code is failing to amortize the larger cost of the peel and remainder loops associated with the greater vector width of AVX. We have expanded upon this in the text.

Carlos Osuna's specific comments:

(Top of page 8.) We have clarified why allocating larger arrays was necessary in order to use the safe_address directive.

Changed "Intel does..." to "Intel can..." as requested.

Sec 2.1.5 - the grid is not attached to the field, rather each field keeps a pointer back to the grid upon which it is defined. This approach was chosen in order to support applications which use more than one grid.

Sec 2.1.8 - it is not clear why in-lining manually helps (especially since one would expect the Intel compiler in particular to do this). We suspect that the change in the PSy-layer code causes the compiler to make some different optimisation decisions and these happen to be beneficial. A proper investigation of this is outside the scope of this paper. We have added a sentence about this.

Sec 2.2.1 - yes, other strategies might be beneficial, especially blocking. The key point is that adopting the PSyKAl structure does not limit these possibilities.

Figure 3 - The Gnu results are much slower because gfortran is only SIMD vectorising the loop containing the Continuity kernel. For some reason it does not vectorise any others. Unlike Cray, there is no directive that can be inserted to mandate that a loop be SIMD vectorised. We have added text to this effect.

Page 15, line 5 - we were referring to the L3 cache. This has been clarified.

Figure 5 - it is a reflection of the fact that Gnu is free while the Intel and Cray compilers have performance of the generated code as a key part of their respective business models.

Page 18, line 9 - the NEMOLite2D performance for the 32x32 domain is significantly greater than for the other domains because it largely fits within L2 cache. We have calculated that the working-set size for the momentum kernel is 136*nˆ2. For 32x32 this gives ∼136KB while for 64x64 this gives ∼544KB. The former will fit within L2 cache while the latter will spill to L3 cache. In addition, recall that we only parallelise the outermost loop (Sec. 2.2.1) so, for the 32x32 domain, there are only 32 items to be shared amongst the threads. We have added a note to the text regarding this.

Figure 11 - the size of the Momentum kernels meant that their resource use limited the occupancy that could be achieved. Therefore COLLAPSE might only help the performance of the other kernels and they only account for some 30% of the runtime. In addition, the kernels had been parallelised by enclosing them within a !$acc kernels region which leaves the details to the compiler. It would seem that in this case the decisions made by the compiler were good and COLLAPSE provided little or no benefit. The text has been altered to cover both of these points.

Page 22, line 11 - we agree with the reviewer's point that there was no proof that the CUDA version was performing well. However, it is also not a matter of comparing memory bandwidths on the CPU and GPU - our work on the roofline model has demonstrated that the Momentum kernel is not memory-bandwidth bound. Instead, we have added a paragraph that demonstrates we have used profiling to investigate the performance limitations on the GPU and are confident that we can do no better without changing kernel code.

Figure 13 - font size and line widths have been increased. Labels have been rearranged to improve clarity.

Finally, the software discussed in this paper has been published with a DOI (http://dx.doi.org/10.5286/edata/707) and the title updated, as required.

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