Geosci. Model Dev. Discuss.,
doi:10.5194/gmd-2016-307-RC2, 2017

Geoscientific Model Development Discussions
Open Access
EGU

Interactive comment on "Accelerating the Global Nested Air Quality Prediction Modeling System (GNAQPMS) model on Intel Xeon Phi processors" *by* Hui Wang et al.

Anonymous Referee #2

Received and published: 25 March 2017

The manuscript presents results of efforts to improve the performance and scalability of a global atmospheric chemistry model, the QNAQPMS, using the Knights Landing version of the Intel Xeon Phi. Gains are relative to the original unoptimized code on both KNL and on Broadwell, Intel's latest-generation multi-core Xeon processor. The subject of the work, increasing simulation capability of QNAQPMS using next-generation processors to improve computational performance and scaling, represents a noteworthy contribution to modeling science within the scope of Geoscientific Model Development. The contribution comes in the form of new concepts (next generation processors) and methods (measuring performance, identification of bottlenecks, and optimization through restructuring of application code). The approach and methods

Printer-friendly version

Discussion paper

appear to be valid and reasonable, and the references to related work are appropriate and sufficient.

This reviewer did not evaluate the manuscript with respect to scientific reproducibility. The paper appears to present sufficient detail on the workload, configuration of the application, and the computational platforms tested that another scientist with access to the code, datasets and computer resources would be able to reproduce the performance and scaling results presented. However, the verification of model output from the optimized version appears to be based on merely eyeball comparisons of color contour plots. The lack of a suitable objective verification technique would make it impossible to be sure the results were reproduced.

Regarding the presentation of the results, the clarity and the level of detail are very poor. There are significant gaps in the explanation of, for example, the analysis of hotspots and bottlenecks in the code, and their relative impacts on performance.

The use of VTune is mentioned, but how do the authors determine whether a particular hotspot is performing inadequately, and if so, what is the specific bottleneck? Insufficient use of vectors? Hardware threads? Memory bandwidth or latency? Load imbalance?

The use of OpenMP threading is also insufficiently motivated or explained.

This reviewer also takes issue with including “changing global communication from interface-files writing/reading to MPI functions” among the list of five optimizations discussed, since this change appears to address a fundamental deficiency in the parallel design of the original application, not one specific to optimizing for the Intel Xeon Phi processor. This reviewer suggests removing discussion of this optimization from the paper and using the space to more fully explain the other four optimizations. If, however, the authors wish to include discussion and results of the global communication optimization in this paper, its effect on performance and scaling should be clearly separated from the other optimization results. As currently presented, the reader cannot

deterimine what effect this has relative to the effects of the KNL-specific optimizations.

Because of these problems, which go beyond language and grammatical issues, this reviewer recommends the paper be rejected and reconsidered only after significant revision.

Specific comments:

Pages 1 and 2, Introduction. Discussion of impacts is incomplete. What levels of performance does QNAQPMS currently provide for simulation science using the model? What scientific problems are currently possible? What scientific problems are beyond reach with current QNAQPMS performance and scaling. How much more performance will be is required to enable these larger problems. Be specific.

Page 3, Section 2.1, Model Description of GNAQPMS. (Suggestion) What is the difference between GNAQPMS and NAQMPS? Basing the description of GNAQPMS on the NAQPMS presupposes knowledge about the NAQPMS. There is already a reference to NAQPMS but also add a few sentences of background on the NAQPMS. Explain why the authors focus GNAQPMS and not at the NAQPMS.

Page 4, line 3: “Since the memory processor is not dominant”. Unclear, explain further: what do the authors mean specifically by “memory pressure”. Memory working set size? Memory bandwidth requirement of this application. How have they determined it is not dominant?

Page 4, line 9: Spell out first use of TLS to represent thread local storage. It is spelled out in the abstract, but needs to be spelled out again in main body of paper.

Page 5, paragraph beginning line 6 and all of Section 3.2.1. As noted in the general comments above, the use of files for global communication instead of MPI reductions and gathers is problematic for any modern parallel architecture, not just the KNL. Strongly recommend removing discussion of this optimization, including the entire Section 3.2.1, from the paper. It is relevant to optimizing for KNL.

Page 6, line 11. “Cyclic order”? Do the authors mean loop nesting order?

Page 6, line 12. “We cancelled the calling of the subroutine ... and made it an internal function in main program.” This technique is referred to as inlining. Did the authors look at inlining reports from the Intel compiler to see if this could have been accomplished automatically or with the help of directives and compiler options, without manually re-structuring the code? Also, the authors mean to say the "calling subroutine", not the "main program".

Page 6, line 14. “... using parameters to convert scalar structure to vector structure.” Unclear what "coverting scalar to vector" means, nor does it appear from step 3 Figure 3 that this is what is going on.

Page 6, line 15. “... we added the directives, clauses, declarations and syntax comment of OpenMP outside the outermost loop as shown in box 4.” This sentence appears to be the only discussion in the paper of how and why OpenMP threading was added to the code. Given that this is one of the five optimizations being presented, this is insufficient. It’s not clear why the authors felt it was important to add OpenMP threading nor whether, from the results, it provided a benefit.

Page 7, line 15. “2) updating some code segments to the the serial codes to construct vectorization.” What does this mean?

Page 7, lines 17-25. A reader might make a number of educated guesses about what the authors are saying was the bottleneck and how it was fixed, but it is not clear at all from the text. What codes were "added by the compiler?" Was the issue a copy-in/copy-out problem for thread-local variables that were listed as private/firstprivate in the OpenMP directives? If so, how did adding the CBZOBJ argument fix this? By passing-by-reference? If so, how were data races avoided? Were they avoided because the CBMZOBJ objects themselves were THREADPRIVATE? There’s a lot of important detail missing from this discussion.

Page 7, line 32. "...which spent 10 percentages and 8 percentages." One would prefer to see actual timings instead of percentage of runtime in discussion of performance improvement.

Page 7, line 36. "1.78 speedup on the CPU platform and 2.39 speedup on the KNL." How much of these speedups were from the manual vectorization and how much from the optimization of global communication? (Again, this reviewer suggests not considering global communication optimization at all, but if it is discussed, show effects separately from other optimizations).

Page 8, lines 7-10. "The horizontal resolution of the model is $1^\circ \times 1^\circ$, which indicates that the modelling domain contains 360×180 grids. And the number of vertical layers is 20, while the time step for integration is 600 seconds in the test case. The test case was designed to test the performance of GNAQPMS on single node of CPU and KNL platform, and multi-nodes on different platform clusters." This workload is very small and probably not suitable for KNL clusters, which require a high degree of parallelism to be efficient. (The authors make note of this on page 9, lines 32-33). The paper describe a representative workload for scientific simulations using the GNAQPMS and provide some discussion of how the performance results presented are relevant to an actual scientific workload.

Page 8, line 15. "The Intel Corporation provides the High Performance Computing environment for the test." Was this Intel's Endeavor cluster? Perhaps mention this as well.

Page 8, line 20. Opt-V GNAQPMS has been compiled on CPU and KNL platform, respectively." Not sure what this means. Are the authors actually compiling the codes on the respective platforms? If so, why? Is there some difference expected between a native-compiled and a cross-compiled executable?

Page 8, line 21. "... the -xCore-AVX2 and –xMIC-AVX512 compile flags were not used for the advection module..." This is troubling and bears further discussion. What did the

authors see that caused them to avoid these compiler flags for the advection module? Why is advection susceptible to differences but not other parts of the code such as the ODE solver?

Page 9, line 2. "...were confirmed to be identical." This is not a persuasive verification method. An eyeball comparison of plots would not be considered sufficient to confirm that results are identical. Provide difference plots and RMS difference statistics.

Page 9, line 20. "...when the computing scale is fixed." Right word? Instead maybe workload? problem size?

Page 9, line 24. "After optimization, the parallel scalability of GNAQPMS is greatly improved..." This seems counterintuitive, since vectorization and other node-performance optimizations that improve performance on each node from should make the code less well, assuming that interprocessor communication overheads are the same. It's important to distinguish here (1) what parts of the code are being timed for the scaling measurements. (2) Which optimizations are contributing to the improved scalability and which may be improving performance but working against strong scaling.

Page 9, line 33. "... OpenMP code segments on KNL..." What is the effect of varying the number of OpenMP threads with respect to MPI tasks? That is, using the same overall number of hardware threads across the job? What is the effect of running pure MPI? Pure OpenMP?

Page 10, Conclusion section. Future work and areas for improvement are discussed. Howeve,r the conclusion should also include discussion of what levels of performance and scaling are needed for simulation science using GNAQPMS. The conclusion should assess whether and how well the presented work helps the GNAQPMS users achieve these goals.

Figure 7. Shows speedup but not performance. Suggest adding a figure that shows performance as a function of the number of nodes. Plot as simulation seconds per

second of wall clock time.

Interactive comment on Geosci. Model Dev. Discuss., doi:10.5194/gmd-2016-307, 2017.

Printer-friendly version

Discussion paper