Review of "An Effective Communication Topology for Performance Optimization: A Case Study of the Finite Volume WAve Modeling (FVWAM)" by Renbo Pang, Fujiang Yu, Yuanyong Gao, Ye Yuan, Liang Yuan, and Zhiyi Gao

This paper describes implementation and performance benchmarks of neighborhood exchanges in the FVWAM model. It compares the performance of a standard implementation of halo exchanges based on point-to-point communication with the performance of an implementation based on MPI distributed graph topology interface and neighborhood collectives. Using the distributed graph topology interface, the authors obtained a maximum communication time speed-up of 5.63 and 40.2% reduction in the total FVWAM run time with 1,024 processes. This is an interesting and significant result, especially because the use of the distributed graph topology interface and neighborhood collectives is not common in earth system models. I believe the article is suitable for GMD after my comments are addressed.

Specific comments

- In section 2.2 the authors describe potential benefits of using the MPI distributed topology interface and present its ability to optimize process mappings as its main advantage. However, the benchmark results of FVWAM show that, while using this interface provides consistent speedups over the point-to-point implementation, setting the reorder flag has only minor performance impacts. What is then the main reason for the observed speedups ? Can section 2.2 be expanded to discuss other potential performance benefits ?
- The presentation of the distributed graph topology workflow in section 3.1 could be improved. The first three paragraphs, related to Figure 3, describe the process of creating MPI graph topology starting from domain partitioning. Most of this material is then repeated in subsequent paragraphs, which pertain to Figure 4. If the authors' intention was to first present the workflow at a high level and then go into details specific to FVWAM this needs to be clearly stated and better organized to remove some of the repetition.
- Are the results of both communication mechanisms bit-for-bit identical ? How was the correctness of the implementation verified ?
- All of the paper performance results were obtained using Intel MPI on one computing system. I imagine that the performance of a high-level interface like the distributed graph topology can strongly depend on the quality of implementation of the underlying MPI library. At minimum, this should

be discussed, but showing results using a different MPI implementation would be a great addition to the paper. Do the authors expect that their results would generalize to other platforms ?

Minor comments

- Throughout the paper, the authors refer to cells and cell indices as grids and grid IDs. This terminology is very non-standard and can be confusing. I strongly suggest replacing "grids" with "cells" and either replacing "grid IDs" with "cell IDs" or adding a sentence that in this paper "grid IDs" mean cell IDs.
- Line 156: the variable "cellsOnCell" has already been introduced on line 134, where it is spelled "CellsonCell".
- Line 181: "MPI_DIST_GRAPH_CREATE_ADJACENT" - why are some MPI function names written in all-caps and some are not ? I suggest using the C interface names consistently throughout the paper.
- Lines 212-213: "MPI_Isend (...) is infrequently utilized ...". Can the authors back-up this claim ? All of the models I worked on used "MPI_Isend".
- Table 1: Change "Compiling Option" to "Compilation Options".