Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2020-257-RC2, 2020

Geoscientific Model Development Discussions

Interactive comment on "Parallelizing a serial code: open–source module, EZ Parallel 1.0, and geophysics examples" *by* Jason Louis Turner and Samuel N. Stechmann

Anonymous Referee #2

Received and published: 23 December 2020

This paper presents the authors' efforts on developing a tool that can help the process of parallelizing a serial code, at the level of MPI-based parallelization.

This is a traditionally interesting topic, even more interesting at the current stage of migrating from homogeneous multi-core clusters to heterogeneous clusters with GPUs or other many-core accelerators.

This is overall a quite interesting work. The authors have also done a good job to provide not only a tool but also demonstrations in a number of different applications or kernels.

My major question is about the comparison with other similar efforts in the field. Par-

allelization, especially MPI-based parallelization, has been around for many decades. Groups from both computer science and application domains have existing projects that try to derive languages, compiler, tools to support better and easier parallelization. Therefore, an introduction of such a tool should come with a comprehensive overview of existing efforts. Also, for demonstrating the efficiency and performance of the proposed tool, comparisons should be made on both the parallel performance achieved, and the extra coding efforts needed. In the current paper, we only see results of the parallel code in the proposed tool, but not sure how good it is when compared to other similar tools, or languages, such as the Unified Parallel C project from Berkeley.

Another minor complaint is about the lack of support for parallelism at thread level. My understanding is that the backbone of the tool is based on MPI, which is still the best way to go for parallelization across different nodes. But the current hardware trend is to have more and more computing power within a node, and you can easily have a heavy CPU with around 50 cores, or a GPU with thousands of CUDA cores. It would be a more interesting tool if these cases can be covered, or at least discussed. For example, for a node with around 100 cores, how would such a tool perform against an OpenMP approach?

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2020-257, 2020.