Geosci. Model Dev. Discuss., 6, C1803–C1805, 2013
www.geosci-model-dev-discuss.net/6/C1803/2013/

Geoscientific Model Development Discussions

Interactive comment on "A distributed computing approach to improve the performance of the Parallel Ocean Program (v2.1)" *by* B. van Werkhoven et al.

M. Govett (Referee)

Mark.W.Govett@noaa.gov

Received and published: 9 December 2013

I liked this paper. It describes a new domain partitioning strategy that minimizes interprocess communications across multiple nodes and yields significant performance benefit. The paper also describes efforts to parallelize select routines to run on the GPU. The time required to copy data between host (CPU) and device (GPU) is significantly more than the computation time of each routine, so several strategies were tried to reduce the copy time (explicit), by using GPU mapped memory (implicit) and cuda streams that overlap communications and computations. GPU runtimes were marginally faster but would benefit much more if more routines were run on the device

to avoid or reduce communications time. It is noted the future work will be done to parallelize more routines which will reduce inter CPU-GPU communications.

Further comments:

Section 2.3 describes the partitioning algorithm and references figures describing an example of the domain decomposition. In particular, Figure 7 shows significant improvement (in blue) for the two cluster runtimes. However, it seems there are edge cases that might lead to points in the domain that would be stranded in the two-stage (horizontal and vertical) partitioning scheme. This would be potentially exacerbated by land points that could surround ocean points at the edge of the model domain. For example if the domain were 180W to 180E, could there, in some cases, be ocean points at the corner of the domain (eg. NE corner) that are isolated (surrounded by land points) that might be contiguous with ocean points in the NW portion of the domain? Are there other edge cases that might require more complexity in the algorithm than was either explained or shown to incorporate potentially stranded or isolated ocean points?

Section 2.4: You do not explain the motivation for exploring tri-pole grids. Are the scientific results better? Do you avoid certain scientific, computational or other issues? Please explain.

Section 4: The POP is a widely used community model written in Fortran 90. It appears select routines were rewritten in CUDA-C (assume you did not use CUDA Fortran but you did not say). You reference the CUDA programming model but that is not sufficient, please give more details. In both cases, the original source code would need to be modified so it can run on the NVIDIA GPU, which means it is less useful to the community of Fortran-based modelers and researchers - because they don't want to learn CUDA or maintain two copies (Fortran and CUDA) of the code. Have you considered using the openACC compilers to parallelize the routines, and based on your success parallelize other routines in POP with these compilers? That would be most beneficial

to the community in my view.

You do not report on speedup when communications times (with explicit, implicit or streams) is not included. This would be useful to know particularly if you or others wish to parallelize the rest of the model for GPU. Given there is such a low computational intensity, and the subroutine profile is fairly flat, parallelization of the entire model may be the best approach and as you state, may be a target for future work.

Section 5.2: Page 4726, lines 25-29 The results you show using more MPI tasks with less data per GPU block outperforms larger blocks and fewer MPI tasks. Can you explain the configuration more clearly? Are you scheduling multiple MPI tasks per GPU device? If so, you may benefit from the reduced synchronization overhead, because when one MPI task is waiting for block (kernel) level synchonization, another MPI task can be executing on the same GPU device.

Interactive comment on Geosci. Model Dev. Discuss., 6, 4705, 2013.