Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2020-91-SC5, 2020

Interactive comment on "A new distributed algorithm for routing network generation in model coupling and its evaluation based on C-Coupler2" *by* Hao Yu et al.

Li Liu

liuli-cess@tsinghua.edu.cn

Received and published: 16 May 2020

Dear Mr. Vijay Mahadevan,

Thanks a lot for your review. The suggestions and comments in the review will further help us to improve the manuscript.

We feel that, there are some misunderstandings arising from the current manuscript and we may not fully understand some of your comments and suggestions. We therefore wish more discussions with you before revising the manuscript.

In the following, we'd like to pose discussions about some review points. Please show us if some points in the discussions are wrong or incomplete. Thanks a lot.

1. “However, I fail to see enough conclusive evidence that improvement of just this initialization phase of the solver would lead to a signiïňĄcant impact on the total time to compute the overall coupled climate solution.” “The algorithmic modiïňĄcations proposed in the current manuscript is an incremental update to an existing algorithm, and does not provide a signiïňĄcant enough impact on the overall runtime of the climate solver.”

RESPONSE: The impact of initialization phase on total time of a model run is relative. Initialization phase will be trivial when simulating a long time (e.g., hundreds of model days or even hundreds of model years), but may be significant for a short simulation (e.g., several model days or even several model hours generally in weather forecasting). It can be required to start to run a model just for only several model hours in data assimilation for weather forecasting. Regarding an operational model, there is generally a time limitation of producing forecasting result (for example, finishing 5-day forecasting in two hours), no matter of the model resolution, and thus developers always have to carefully optimize various software modules especially when the model resolution gets finer.

C-Coupler2 has been used in an operational air-sea coupled model that must finish 7-day forecasting in two hours. This coupled model consists of 3 component models with about 6 million points in the largest model grid, running on over 3000 cores in operational forecasting. C-Coupler2 took about 1200 seconds for initialization at the beginning, and thus we have been asked for accelerating the initialization stage. That’s a motivation for this manuscript. We are sorry that we could not include the corresponding results in the manuscript because the codes of this coupled model cannot be open (according to GMD policies, the code used in the manuscript must be available).

Several months ago, a model developer in China asked us whether C-Coupler2 could couple an Ocean model with about 72,000,000 points in the horizontal grid, as he was being involved in developing such a model. C-Coupler2 cannot handle such coupling because of not only the significant initialization cost but also the huge memory consumption. Similar to some other couplers, C-Coupler2 utilizes the global grid representation, which means that each process keeps all points of each grid. Corresponding to 72,000,000 points, the memory for keeping such a grid in a process will be more than 6GB (given that each point has four vertexes and the data type is double precision). To address this challenge, we started to develop a distributed grid representation a few months ago. We find that it is always required to generate a new distribution of a grid from an existing distribution of the grid. As a grid distribution is essentially a parallel decomposition, the redistribution of grid points can also be handled by the data transfer functionality. Thus, routing network generation will be more frequently used when initializing the coupler, and the impact of its overhead will be more significant. That's another motivation for this manuscript.

We will try to include these motivations when revising the manuscript.

2. "Additionally, the proposed modiïňĄcations to the global routing table generation scheme is incremental in nature, and does not aim to minimize communication times between source and destination processes."

RESPONSE: The gather/broadcast based global routing table generation can be viewed as a sequential implementation while this manuscript focuses on how to parallelize it for acceleration. Given K processes, the communication times of gather/broadcast is O(K) for each process. Although the implementation proposed in this manuscript retains communication times at O(K), it decreases the message size per process. In other words, it does not change the overall complexity of routing table generation but makes processes cooperatively work together for acceleration. That is a general way how a parallel optimization accelerates a program. So, this manuscript should not be a new algorithm but a distributed implementation. We will modify the title and the corresponding content when revising the manuscript.

3. “However, since the algorithm does not aim to provide a better partitioning strategy, or make use of architecture layout to minimize communication latency (using say task mapping algorithms), the resulting communication graph between processes still remain the same as the one generated from global routing network algorithm.”

RESPONSE: As this manuscript can be view as a parallel optimization of the global routing network generation, it should not change the resulting communication graph between processes. The communication graph is generally determined by the parallel decompositions of the source and target component models and the use of architecture layout to lower communication latency can be further determined by the mapping between the process layout and the architecture layout. For a coupler working as a library, it generally can only input the parallel decompositions of models, but cannot change the parallel decompositions, communication graph and process layout.

4. “First, the authors claim that "existing couplers employ an ineﬁňĄcient and unscalableglobalimplementationforroutingnetworkgenerationthatreliesoncollective communications. That’s a main reason why the initialization cost of a coupler increases rapidly when using more processor cores."." “Can you provide some actual timings from the fully coupled climate solver runs to put the actual setup costs in perspective ? The scalability shown in Fig. (3) still indicate about 3.5s of compute time (since speedup for global routing network case is ≈1) for the 16M grid case on 1600 processes. If this accounts for say over 5% of the actual runtime of the solver, or a non-trivial percentage of total time to simulate a year (or days for high-res) of climate interactions, then 20x improvements in the setup cost could be quite impactful. However, such one time costs get amortized with physics setup costs for high-res runs, in addition to long-term temporal integration of the actual coupled simulation. Hence, I think the manuscript lacks a strong motivation, and provides only an incremental update to avoid the collective algorithms in the coupled simulation invoked during the initial setup phase. ”

RESPONSE: For the motivations of this manuscript, please refer to the response of the first point in this reply. We will add some performance data about the impact of routing network generation on the total time of coupler initialization.

We use 16M grid for evaluation to show that DaRong can significantly improve routing network generation under different grid size. We know that it is not a real case to run a 16M-grid simulation on only 1600 processes (cores). According to our experiences, a real model with 16M grid can effectively utilizes 40,000 cores (4,00 points per core) or more. We are sorry that we can only use at most 1600 cores per component model in the evaluation. The results in Fig. 3a, Fig. 3b and Table 7 can indicate that the global routing network generation for 16M grid will take much longer than 3.5s when using such as 40,000 cores.

There are always a number of routing networks generated for different data transfers when initializing a coupled model, and generally more routing networks corresponding to more component models in a coupled model (two-way coupling between a pair of component models generally introduces at least two times of routing network generation). Moreover, as stated in the response of the first point in this reply, the distributed grid representation can make routing network generation more frequently used. The parallel read of a remapping weight file can also utilize routing network generation. We have developed a framework for weakly coupled ensemble data assimilation based on C-Coupler2 recently (please refer to its manuscript https://www.geosci-model-dev-discuss.net/gmd-2020-75/), where a specific C-Coupler2 internal component model of the model ensemble will generate routing networks with each ensemble member. This technique will significantly enlarge the number of cores used by a component model and make routing network generation more frequently used. For example, given that there are 20 ensemble members each of which runs on 4,000 processes (cores), the model ensemble runs on 80,000 processes and it will be involved in at least 40 times of routing network generation with the ensemble members.

5. “Secondly, the global ID based partitioning strategy used in the distributed sort with DaRong to determine the communication pattern is not an innovative concept. There have been several algorithmic ideas based on graph partitioning strategies used in

the parallel Sparse Matrix-Vector (SpMV) linear algebra context [1].” “In a simpliïňĄed sense, without a constraint on the message volume, data locality or latency of communication (such strategies may require repartitioning and/or task mapping), the globally unique ID space can be used in a round-robin type partitioning scheme. For instance, if the source component data are distributed on M processes, and destination on N processes, an implicit decomposition can be determined a-priori based on the global ID numbering that leads to MxN data redistribution. Such an implicit ID decomposition establishes a direct point-to-point communication pattern after which the CLGMT table can be created on both source and destination processes for further send/receive of DoF data at runtime. There may be a need for multiple rounds of rendezvous communication to establish message size for buffer allocation etc, but such an algorithm can eliminate collectives like broadcast and allreduce operations as necessary for better scalability.”

RESPONSE: There are a lot of algorithms or optimizations based on the butterfly communication network. It is not an innovative concept indeed. That’s why we just give a simple example (Fig. 2) in the manuscript. The key point of this manuscript is how to develop a distributed implementation to accelerate routing network generation. We will correct the misleading title and the related content when revising the manuscript.

The sparse MxN data transfer has been widely used in existing couplers, and this manuscript focuses on how to accelerate the routing network generation for initializing the data transfer. This manuscript might correspond to “There may be a need for multiple rounds of rendezvous communication to establish message size for buffer allocation”. We are sorry as we may not fully understand your point here.

Wish your further comments.

Many thanks again,

Li

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2020-91, 2020.