Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2018-27-AC1, 2018

Geoscientific Model Development Discussions

Interactive comment on "C-Coupler2: a flexible and user-friendly community coupler for model coupling and nesting" *by* Li Liu et al.

Li Liu et al.

liuli-cess@tsinghua.edu.cn

Received and published: 2 May 2018

We thank Reviewer #1 for the comments and suggestions. We will modify the manuscript according to them in the revision stage. In the following, we will reply them one by one.

1. I would encourage the authors to publish additional results in the future detailing the performance cost of higher resolution and higher core counts tests and sharing the performance of 3D weight generation and coupling.

Response: Although it is difficult for us to find more processor cores to evaluate the performance cost of higher resolution and higher core counts tests, we will try to add the performance cost of dynamic 3-D weight generation and coupling in the revised

manuscript.

2. page23, lines25-29. The ability to run with lags properly is critical. Lags almost always create additional requirements on restart ïňĄles and the ability to restart a model exactly (bit-for-bit) with lagged coupling ïňĄelds should be a requirement if lags are going to be supported. It sounds like this is not currently supported in C-Coupler2? Maybe rather than saying "We therefore propose", it would be clearer to say something like "Lags are not fully supported in the current version, but in the future, the C-Coupler2 will ..."

Response: It is true that the capability of flexible lags setting will introduce significant technical challenges to achieve exact (bit-for-bit) restart at any case. In the latest code version of C-Coupler2 which will be publicly and formally released before the end of this month, these technical challenges have been fully resolved through a new feature of C-Coupler2: adaptive restart capability. In other words, C-Coupler2 can conveniently achieve exact restart no matter the setting of coupling lags, without any additional requirements. We will detail the implementation of the adaptive restart capability in the revised manuscript.

3. page 26, Section 4.5. Is the only reason MPI_Put and MPI_Get is used is to avoid possible exhaustion of MPI buffer space? That should be very rare in practice. Are there other reasons? Performance, ease of implementation, etc? Based on the description on Section 4.5, the MPI_Put/Get implementation sounds slower and more complicated than well managed MPI_ISend/IRecv implementations with MPI_Wait implemented appropriately. Are the authors happy with this implementation? Section 5.3 answers this question in part, but it might be nice to add a few more words in either section 4.5 or 5.3. I think one-side communication potentially helps with both MPI buffer usage and ability to have greater ïňĆexibility in coupling lags, but does not improve performance? How about implementation complexity?

Response: Originally, we thought that the MPI_Put/MPI_Get implementation was the

unique solution for achieving ïňĆexibility in coupling lags setting. Although it will not obviously increase the performance cost compared to the MPI_ISend/IRecv implementation in most cases, we encountered a significantly slow down case on a IBM system. After adding an MPI_ISend/IRecv implementation into C-Coupler2, we find that the MPI_ISend/IRecv implementation can also achieve ïňĆexibility in coupling lags. Therefore, in the latest C-Coupler2, there are both an MPI_Put/MPI_Get implementation and an MPI_ISend/IRecv implementation, where the latter one is used as default. Users are proposed to try the MPI_Put/MPI_Get implementation if an unpredictable deadlock is encountered. The complexity of the MPI_Put/MPI_Get implementation is much higher. The manuscript and user manual will be modified accordingly.

4. page 28, line 8. This is a nice feature. One has to be concerned about memory usage but this provides a nice way to allow extra ïňĆexibility in lags compared to other implementations.

Response: Extra memory usage and higher cost in achieving exact restart capability are unavoidable when coupling lags are big. We will remind this point in the revised manuscript.

5. page 22, paragraph beginning at line 6. I believe the comparison between Oasis3MCT_3.0, CESM, and C-Coupler2 is not particularly clear. The authors compare how components interact in different systems, but the deïňĄnition of the component is not the same in each system. In CESM and the C-Coupler2, the component is deïňĄned by separation of scientiïňąc models. In Oasis, the component is deïňĄned by the separation of MPI tasks. In addition, CESM is more than a coupling layer, it also includes a top level driver that supports the ability to call multiple components from the same MPI tasks in a single executable but only to couple via the driver layer. Oasis3-MCT_3.0 does not have a driver layer and is driven by calls from inside the models. In practice, users could implement a top level driver using Oasis3-MCT_3.0, so Oasis3-MCT_3.0 can behave just like CESM plus it can behave in other ways. I am still a little unclear about whether the C-Coupler2 consists of a driver. If so, is it just

a single executable system or does is support multiple executables? I believe none of the coupled systems discussed in this paragraph support multiple MPI tasks running on a single processor, and otherwise they are very similar in capabilities. The main difference is that CESM does not support coupling within a component compared to the other two. I think this paragraph should be clariïňĄed. It's difïňĄcult to read and the similarities and differences should be more clearly qualiïňĄed.

Response: Similar to Oasis3-MCT_3.0, C-Coupler2 also does not have a driver layer and is driven by calls from inside the models. C-Coupler2 has a definition of component model similar to CESM but different from Oasis3-MCT_3.0. C-Coupler2 improves its debugging capability based on its definition of component model. As a grid, a parallel decomposition or a field instance must be across all processes of the corresponding component model, C-Coupler2 can check whether all processes of a component model call the corresponding API at the same time, when registering a grid, a parallel decomposition or a field instance. We will improve this paragraph in the revised manuscript.

6. page 22, paragraph beginning at line 22. It seems C-Coupler2 is using a ïňĄle to coordinate MPI tasks between components. While this may be simpler than synchronizing with MPI, there is still the equivalent of a global barrier in the interaction. A component cannot know the tasks of other components until other components have written to the ïňĄle. How does the C-Coupler2 ensure that other components have written to the ïňĄle before the information is needed? What is "ïňĄle" synchronization chosen over MPI?

Response: Given that the coupled model covers 1000 MPI processes, while a component model A covers No. 0∼99 MPI processes and a component model B covers No. 900∼999 MPI processes, the coupling generation between A and B as well as the corresponding "file" synchronization only introduce a partial barrier among process 0∼99 and 900∼999, but not a global barrier. A (or B) will wait until the "file" of B (or A) is ready. We will clarify this point in the revised manuscript.

7. page 34, line 30. The issues with 3D conservative coupling are the same as 2D. Even with areas, model areas and conservation method areas can differ and this needs to be taken into account with 2D conservative mapping. I do not believe there are any fundamental hurdles to extend 2D conservative coupling to 3D and there may be tools that already accomplish that.

Response: In fact, we still do not have a good idea for achieving 3-D conservative coupling.

8. Is the C-Coupler a hub coupler a component, is it just a layer in the system, is it the driver? I think C-Coupler1 was a hub and C-Coupler2 is a coupling layer, is that correct? It might be good to discuss this in the introduction and in regard to Figure 1.

Response: Both C-Coupler1 and C-Coupler2 are libraries but not a hub coupler component. This will be stated clearly in the revised manuscript.

9. With self-coupling or self-nesting on the same pes with the same executable and multiple grids, how does the C-Coupler2 address the issue of multi-instance data privacy within the executable? It may not be enough just to instantiate a new domain or a new state. The underlying model has to meet speciïňĄc and complex requirements to support that feature with regard to fully separating the memory of the two instances and most models do not. Does the C-Coupler2 actually support this and does it introduce any requirements on components to support that capability? For example, running multiple instances on concurrent pes does not create the same problems. Also, using the Coupler2 to couple internal data within a model that supports nesting is not difïňĄcult. It's not clear whether the C-Coupler2 supports self nesting on overlapping pes between a component model and another instance of the same component model. Section 4.6 suggests it can. How can that be? Maybe that could be clariïňĄed. This comes up in Section 3.6 and Section 4.6.

Response: C-Coupler2 does not allocate memory space for component models. Therefore, the issue of multi-instance data privacy within the same executable must

be addressed by the model itself. However, C-Coupler2 can easily identify different instances of the same field (the field names are the same) that are from different grid domains of the same component model or from different component models with the same type, so as to facilitate the implementation of model coupling. For example, given a self-nesting atmosphere model A with three levels of grid domains A1=>A2=>A3, and a elf-nesting ocean model O with the corresponding three levels of grid domains O1=>O2=>O3, C-Coupler2 can easily achieve atmosphere-ocean coupling at each grid level through the corresponding partial coupling generation (a coupling generation between A3 and O3, a coupling generation between A2 and O2, and a coupling generation between A1 and O1), while the configuration files for different partial coupling generations can be almost the same and very simple. In other words, C-Coupler2 can also facilitate the implementation of nesting of a coupled model. We will make clarification in the revised manuscript.

10. Does the C-Coupler2 support unstructured grids in 2D or 3D such as cubed sphere, non quadrilaterals, and other complex geometries? Does the on-line remapping support weight generation for those grids? Please indicate in the text.

Response: As C-Coupler2 still uses the remapping software CoR1 for remapping weight generation, the on-line remapping weight generation supports unstructured 2-D grids. We will state that in the revised manuscript.

11. The results show reasonable performance at moderately high resolution and pecounts. I think these results are adequate at this point, but it would be nice if there were an opportunity to test and publish results at higher resolution and higher task counts in the future, and I agree with the ïňĄnal statement on page 35, line 6.

Response: We only used moderately high resolution and pecounts in this paper mainly due to limited computing resource. Currently we are afraid of that we may not be able to successfully apply more computing resource.

12. I think section 4.1.1.1 to 4.1.1.8 could be removed and the user guide could be

referenced instead. I think the API details are not needed in this paper. 4.1.1 could just a paragraph that provides a few sentences about the API and points to the user guide plus 4.1.1.9. That would be my recommendation, but will allow the authors to respond to this point.

Response: We describe the APIs with details in order to introduce our consideration in how to design the APIs. We will try to shrink the content of section 4.1.1.1.

13. Technical Comments

Response: We will modify the manuscript according to the technical comments.

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2018-27, 2018.