Author's consolidated review comment response on "Improving climate model coupling through a complete mesh representation: a case study with E3SM (v1) and MOAB (v5.x)" *by* Vijay S. Mahadevan et al.

mahadevan@anl.gov

1 Rebuttals for Reviewer 1 comments

1.1 Needed clarifications

- Referee: p.4 l.29 explain "consistently respecting the underlying discretization" or remove the sentence.
  Authors: Modified. Please review.
  Manuscript Diff: Changes in p5. l.9.

- Referee: p.6 l.6 define (or cite) "component architecture". Versus what?
  Authors: The following reference has been added.
  Zhou, S. J. "Coupling climate models with the earth system modeling framework and the common component architecture." Concurrency and Computation: Practice and Experience 18.2 (2006): 203-213.
  Manuscript Diff: Changes in p5. l.28.

- Referee: p.6 l.21 does "Fig. 1 (right)" apply to OASIS3-MCT also? Or does the mere-library approach (no separate N_x for the coupler) defines another work-

flow?
Authors: Yes. We could have $N_x = N_{atm}$ and still have Fig. (1) (left), the hub-and-spoke model, just to be clear. But what the distributed coupled model refers to is that the components can directly communicate with each other without an additional hop through the coupler (hence no explicit N_x). This is much more efficient in terms of total reduced data-transfers and optimizations that can be performed on pair-wise meshes, which are not available on a many-to-many scenario through a global coupler. In the workflow we have defined in the manuscript, the MBTR workflow allows both.
Manuscript Diff: Changes in p6. l.21.

- Referee: p.7 l.25 what does "field [...] aware" mean?
Authors: Being field aware indicates that regridder needs to understand the discretization types. Our aim is to provide an online remapper that supports consistent and conservative projection of **FV, cGLL, dGLL** source field data to target meshes, and fill the gap with ESMF based offline remapping workflows that require dual meshes and only support FV-type discretizations.
Manuscript Diff: No changes.

- Referee: p.7 l.29 "during the setup phase" is in contradiction with the aim of allowing for adaptive and moving meshes. Indicate whether it is just a practical choice in the current implementation.
Authors: In the current implementation, we are computing the remapping operators only in the setup phase since most existing E3SM workflows do not support adaptive grids. But the MBTR workflow can fully support remapping with moving meshes by recomputing the weight matrices at run-time. The text has been slightly rephrased to clarify this.
Manuscript Diff: Changes in p8. l.30-l.31.

- Referee: p.7 ll.30-32 in what exactly is the MBTR stack an improvement w.r.t. the

MCT view, since MCT is able to handle decomposed meshes?
Authors: MBTR is an improvement because it also stores the connectivity of the mesh. In particular, MOAB knows that the neighbor of a point might be on another processor. MCT does not have any of that information, which is essential when performing online remapping computation or performance optimizations/load rebalancing based on mesh topology.
Manuscript Diff: No changes.

- Referee: p.8 l.3 indicate under which scheduling assumptions the N_x processes can share with the $N_{c,l}$ processes part of the processor resources as implied later by Fig. 4.
Authors: Sharing resources would be appropriate when the physics of the system requires that a calculation performed in the coupler must happen before the solver in the next component is invoked. Figure 4 doesn't indicate that the coupler is sometimes invoked multiple times as individual components are executed, and it specifically doesn't show the global "driver" layer which controls the overall flow of execution and data transfer.
Manuscript Diff: No changes.

- Referee: p.8 l.14 please define a "DoF": since it is not a word used for cell-centered couplers, it is not a common term for all the readers.
Authors: DoF has been expanded.
Manuscript Diff: Changes in p9. l.26.

- Referee: p.8 whole section 3.1 (and following) please include references or links for HOMME, MPAS, VisIt and in general do so for all mentioned models, libraries and other software tools (Zoltan, ParMetis, Eigen3, etc).
Authors: These references have now been added.
Manuscript Diff: Changes in p9. l.29-l.30, p11. l.9.

- Referee: p.8 l.26 why "replicated" meshes. Isn't it rather "partitioned"?

Authors: Corrected.
Manuscript Diff: Changes in p11. l.8-l.9.

- Referee: p.8 l.29 "in terms of a 'Tag'." is a useless statement unless you make it clear to the reader.
  Authors: Small description of a tag has been added.
  Manuscript Diff: Changes in p11. l.11-l.13.

- Referee: p.8 l.29 n_p has not been defined and is not trivial.
  Authors: This sentence has been rephrased
  Manuscript Diff: Changes in p11. l.13-l.15.

- Referee: p.9 Algorithm 1. The formulation is too compact and missing some previous definition. Insert references to following sections for details.
  Authors: Appropriate references to other sections have now been added.
  Manuscript Diff: Added several references to sections and relevant citations in p10. Additional descriptions as needed.

- Referee: p.11 ll.2-3 state here (or anticipate) the rational for replacing MCT as a broker.
  Authors: We expect MOAB to perform data transfers faster an more efficiently (fewer overall messages) then MCT at scale because of MOAB's crystal router. MOAB will also have better memory scaling because, unlike MCT, it does not have datatypes that can grow with grid or processor size. Finally MOAB will allow a simplified workflow by removing the need for directories of mapping weight files.
  Manuscript Diff: No changes.

- Referee: p.12 l.8 Kd-tree is a relatively common technique (already mentioned at p.4) BVH-tree deserves a reference here (only provided at p.20).
  Authors: Added references for both tree structures
  Manuscript Diff: p.14 l.8-9.

- Referee: p.12 l.11 why "unique" ?
  Authors: Removed. It is clearer now.
  Manuscript Diff: p.14 l.12.

- Referee: p.12 l.17 the same consideration as for p.7 l.29 applies.
  Authors: Removed reference to the setup phase.
  Manuscript Diff: p.14 l.18-l.19.

- Referee: p.12 ll.26-29 Fig 6. is not immediate to read without some further "step to step" details in the text.
  Authors: This comment is unclear. Should the advancing front algorithm be explained better ? We have added references to the front intersection video illustrations that are added as supplementary materials.
  Manuscript Diff: No changes.

- Referee: pp.12-13 subsection 3.3.1 does the seed determination can be fully automated or its efficiency depend on user tuning?
  Authors: The determination is fully automated. However, there may be cases with failures when dealing with meshes with holes where a seed in say an atmosphere mesh may not be able to find a corresponding element containing point in the MPAS mesh (if it falls in a land geographical area). Such cases could require more than one attempt in each partition to get the front computation started.
  Manuscript Diff: No changes.

- Referee: p.13 l.11 what does the sentence "without approximations" refer to (especially w.r.t what alternative)?
  Authors: The sentence "without approximation" refers to the fact that the intersection can be computed to machine precision as the edges become straight lines in a gnomonic plane (projected from great circle arcs on a sphere). If curves on a sphere are not great circle arcs (splines, for example), the intersection be-

tween those curves has to be computed using some nonlinear iterations such as Newton Raphson for example (depending on the representation of the curve).

We wanted to indicate in the mauscript that intersection in gnomonic plane is simple to do and "exact"; When you have more general curves on a sphere, you might even have multiple points of intersections, which could test the robustness and stability of the intersection algorithm. However, note that a latitude arc can intersect a great circle arc in 2 places (this can still be computed exactly to machine precision, without any approximations coming from an iteration).

Manuscript Diff: No changes.

- Referee: p.14 l.6 computing a meaningful bounding box is not trivial in polar or periodicity regions for lon/lat grids.
  Authors: MOAB stores explicit 3-d bounding boxes, since it is a general mesh query/manipulation library. We have not particularly encountered difficulties in handling lat/lon grids.
  Manuscript Diff: No changes.

- Referee: p.14 l.7 does "to all tasks" refer to tasks (or rather processes) on the source side?
  Authors: This refers to processes on the coupler processing elements. The source/target meshes are already in coupler PEs, and a coverage mesh is computed by appropriately moving only elements required to completely cover target elements in current process.
  Manuscript Diff: No changes added. Description in rebuttal is detailed.

- Referee: p.14 l.8 "Cells [...] are sent": how are they represented? What's the size of the communications? Is any packing strategy used to avoid latency in separate small communications?
  Authors: MOAB utilizes the aggregated crystal router to efficiently send small data between processes. In an all-to-all communication strategy, with log(N)

steps of communication, all the processes get access to the data they need. This is used once during the setup phase to establish point-to-point communication links, which is then used later to pack and send data directly.

During the field transfer from components to coupler, we pack multiple fields together in a single array to send the data to coupler, apply weight matrices on the vectors and transmit back the fields (in a packed and aggregated fashion) to the target component. The size of such communication is on the order of DoFs in source + target.
Manuscript Diff: No changes added. Description in rebuttal is detailed.

- Referee: p.14 l.10 please clarify the term "superset": a superset usually refers to inclusion of similat objects. Does it imply that the after representation in MOAB - through the definition of the supermesh - the source and the target side share the same spatial discretisation?
Authors: The MOAB view of the supermesh includes the union of all vertices and (elements formed by) edges in both source and target grids. Hence the supermesh is typically the superset of either the source or the target grid. This is only with respect to the actual topology of the grid, and has no correlation to the underlying discretization of field data.
Manuscript Diff: No changes added. Description in rebuttal is detailed.

- Referee: p.14 l.14 is the "crystal" router explained in Tautges et al. (2012) [N.B. reference not freely available] or does it need an extra reference?
Authors: Added.
Manuscript Diff: p.16 l.21.

- Referee: p.15 l.5 how expensive can be the communication of ghost intersection elements on highly distributed components?
Authors: The communication is typically among nearest neighbors and requires 1-2 rings of elements on average depending on the relative resolution between

source and target grids. Since these are direct nearest neighbor computations that are performed only once during the setup phase, the actual impact on overall runtime is small.
Manuscript Diff: p.16 l.21.

- Referee: p.15 l.13 does "has the potential to" mean that is just an idea or is there a prototype?
Authors: This is an idea and a work in progress at the moment. We expect to spend more time hardening the implementation in the coming year.
Manuscript Diff: No changes added.

- Referee: p.16 l.28 "it is non-trivial to": did you find a way?
Authors: There are ways that could ensure bit-for-bit reproducibility at the cost of heavy sub-optimizations. There are internal discussions to better understand whether the non-BFB algorithmic parts can be isolated together.
Manuscript Diff: No changes added.

- Referee: p.21 l.8 reference to NE11 configuration not known to the reader.
Authors: NE refers to the number of elements on an edge of a cubed-sphere grid. This has been clarified in the manuscript.
Manuscript Diff: p.23 l.1, p.25 l.4, p.25 l.6

- Referee: p.26 Fig.14(b) provide an explanation for the difference of behaviour when going beyond 64 processors
Authors: This was an interesting transition in the communication timings as we expanded from intra-node to inter-node regime on Cori that has 64 Haswell cores per node. The overall message passing latency as we cross the 64-core barrier is certainly evident in the figure, especially since we are only communicating one solution data field from the component to coupler and vice-versa. We have added some text in the revised paper to discuss this further.
Manuscript Diff: p.28 l.13-l.16

1.2 Technical Corrections

- Referee: p.2 l.5 vs l.31 (and elsewhere) make the use of "donor" or "source" consistent p.2 l.9 should probably be "conservation for critical quantities" p.3 l.20 the subject of "that nonlinearly couple" should not be "solution fields" (they are just exchanged in the nonlinear coupling process)
  Authors: Done
  Manuscript Diff: p.2 l.6

- Referee: p.6 l.6 unclear (if not useless) reference "Section (1)"
  Authors: Introduced a subsection 1.1 to clarify.
  Manuscript Diff: p.2 l.27 - new subsection

- Referee: p.6 l.32 the "GLL acronym" is used before definition which is given a few lines later
  Authors: Done
  Manuscript Diff: p.7 l.6

- Referee: p.7 l.29 "an in-memory" instead of "a in-memory" p.8 l.26 remove "a" before "replicated SE and MPAS" meshes
  Authors: Removed "a"
  Manuscript Diff: p.8 l.29

- Referee: p.9 Algorithm 1. - Step 1: if l can only be s or t - as in step 4: - indicate $l \in [s,t]$ also in step 1: otherwise if the formulation is generic for more than one mesh for component, the naming should be consistent. - Step 2: if you indicate W_{ij} instead of W_{st} you should not define i,j as a mesh pair. Later at step 20: i takes a specific meaning.
  Authors: Revised and included suggested changes
  Manuscript Diff: p.10 l.3 in Algorithm. 1

- Referee: p.12 l.1 "partitioner" instead of "repartioner" p.12 l.23 remove "is" before "results" p.12 l.26 "each" instead of "Each" p.14 Fig.7 caption: "fully covers" instead of "fully cover" p.15 l.3 "the intersection vertices [...] need" instead of "needs"
  Authors: Done
  Manuscript Diff: p.10 l.1-l.3 in Algorithm. 1

2 Rebuttals for Reviewer 2 comments

2.1 General Comments

- Referee: The main body text where references are given needs to be reformatted. The references and text are not clearly separated and make them difficult to read. There is also a reference to "Section. (1)" on Page 6, line 6, that I believe suffers from the same formatting problem?
  Authors Done. We replaced "cite" with "citep". Manuscript Diff: All citations/references in the manuscript have been modified due to this change.

- Referee: The outstanding questions that are not answered in the paper are (1) can the weights generated online be counted on to produce error free interpolation (conservation, monotonicity, etc) without first being reviewed and validated offline? (2) is the weights generation capability robust and reliable enough to run on different platforms and expect the same results to at least roundoff? (3) is it faster to generate weights online vs reading them in? (4) Is there some benefit to generating the weights online and then being able to reuse them as compared to regenerating them each time the model is run with regard to performance or reproducibility? It would be helpful if the paper addressed these issues if possible. These issues are partly raised in a few places in the paper, at least Page 6,

Lines 25-27 and Page 18, Lines 9-10. Some addiitonal discussion/results might be interesting.
Authors We have made changes to the manuscript in the Background, Software and Results sections to raise these questions and to address the solutions appropriately as needed. Detailed discussions have also been provided in a previous reply to the reviewer comments.
Manuscript Diff: p.7 l.25-l.31, p.20 l.21-l.22, p.25 l.20 - p.26 l.23.

- Referee: The MCT gsmap is generally compressed significantly because the information can be defined via a single start and end ID for certain kinds of decompositions. Since the MOAB mesh carries more info, I assume that compression is not possible and that the mesh consists of "n" fields of data for each gridpoint/corner/edge/etc? Is that a lot of data? Does the memory scale at all at higher resolutions and higher pe counts? I'm sure much of this is documented in MOAB papers, but it would be nice to add a sentence or two about it in this paper.
  Authors Some discussions about the mesh storage and memory requirements for serializing field DoF data on the MOAB mesh has been added. Again, a detailed discussion was provided in the previous response and we can add to it if further clarifications are needed.
  Manuscript Diff: p.9 l.7-l.12.

- Referee: In Figure 1, it looks like there is no longer a coupler. Where are the non-coupling non-mapping coupler operations (merging, atm/ocn flux, diagnostics, etc) being computed? In text, it sounds like the coupler component still exists but that the underlying MCT datatypes were swapped for MOAB datatypes, an additional set of calls were added in the component coupling layer to more fully describe the meshes, the online weights generation was added, and the online sparse matrix multiply was converted from MCT calls to MOAB calls. But then at page 17, line 12-15, it sounds like the coupling is between pairs of components excluding a coupler. It would be good if this were clarified.

Authors Quoting from our previous response: "The hub coupler still exists. We are currently duplicating the MCT calls alongside the MOAB based coupler in order to fully verify and validate both the accuracy and performance at runtime. After full validation, the MCT coupler will be completely removed from E3SM. The MOAB coupler allows the possibility for ATM to directly compute the remapping weights to project field data to OCN since the intersection will then be carried out through migration of OCN mesh to ATM pes. Hence this pair-wise coupling leads to a more distributed coupling strategy in the future. However, we do envision that there will still be a thin layer of a global coupler, even in the distributed case, to drive the subcycling, to compute merging with weighted combinations of fluxes, for validation and other diagnostics data outputs. We understand that Fig. (1) is somewhat misleading in this context and intend to make modifications to make it clearer.". Clarifications have been added to the text along these lines.
Manuscript Diff: p.9 l.33-l.34, p.12 l.6-l.11

2.2 Technical Corrections

- Referee: Page 6, line 20 "oas (2018)" ?
  Authors Fixed.
  Manuscript Diff: p.6 l.34.
- Referee: Page 7, line 29 fix "a in-line", should be "an in-line"
  Authors Changed "a in-memory" to "an in-memory"
  Manuscript Diff: p.8 l.29.
- Referee: Page 8, line 7 Alg. 1 -> Algorithm 1
  Authors Fixed.
  Manuscript Diff: p.9 l.18.

- Referee: Page 11, Fig 5b. It seems unlikely that the trivial decompostion would be someone's first guess for best performing decomposition with knowledge of how the coupling/ mapping work. Having said that, I'm surprised it performs as well as it does in Figure 14. There are lots of other resonable decompositions, why were Trival and Zoltan chosen to be highlighted in this paper? And why does the trivial decomposition perform so well in Figure 14.
  Authors This was another particularly interesting result from our scaling studies. The triival partitioner is not particularly the best strategy, but from an implementation stand-point, easiest to get working. However, we expected the Zoltan repartitioner to provide much better scaling and overall speedup (in terms of time) when computing the remapping weights by minimizing the source coverage mesh communication time. But, this problem is particularly tricky, since there are two parts that have to be optimized simultaneously.

  1. Migration from component to coupler requires repartitioning,
  2. computing coverage mesh requires moving source mesh elements to cover local target elements.

  So even if one partitioner is optimal for migration, it may still require moving lot of elements for coverage computation. There are ways where we could simultaneously optimize the partition for all components (source/target combinations) while at the same time taking into account the PE layouts, but this implementation is more involved, and is a work in progress at this stage.
  Manuscript Diff: p.13 l.29 - p.14 l.6.

- Referee: Page 12 line 23, remove "is" in "is results in"
  Authors Done.
  Manuscript Diff: p.14 l.8

- Referee: Page 16, line 28, bit-for-bit capability is sometimes important to achieve, certainly for identical runs, also for runs on different pe counts (sometimes with a performance penalty via an optional flag). This sentence left me asking what the bit-for-bit capabilities are and what risks are introduced when computing online versus reusing.

Authors Agreed. If the performance penalties are not an issue, potentially exact bit-for-bit runs can be performed with the MOAB intersection. While we have not noticed any variation in the actual supermesh computation, the element sequence in the resulting supermesh will have to be re-sorted so that it is always partition agnostic. Currently, this is not strictly enforced. Additionally, any and all reductions in remapping weight computations, enforcing conservation and performing A*x, where A is the weight matrix and x is the solution vector to be projected need to be handled carefully to preserve unique order of arithmetic necessary for bit-for-bit reproducibility. Hence our statement that this is non-trivial, though necessary in the longer run as an explicit option.

Manuscript Diff: No changes.

- Referee: Page 19, Fig 9. it would be nice if the scale were not so ad-hoc and instead something more like (0,80,4). Scales like the one shown just make the figure more difficult to digest and in this case, there is no benefit to have the breaks defined as they are relative to something simpler and easier to read. Also, I'm not sure color adds anything, I think the same could be shown via a contour plot, possibly clearer and simpler still.

  Authors We originally made use of contour plots but it made the appearance much less easier on the eye. The issue with presenting this data is that its a 2-D data set showing the timings for combination of source/target element combinations. We could use 3-D plots to show surfaces aligned to the computation time but drawing conclusion from such a description was not obvious. The reasoning for the chosen scale in Fig. 9 is that around 80 secs was the maximum amount of time (upper bound) for the largest source-target element combination to run ESMF in our case. The coloring provides a relative comparison with respect to this upper bound, and shows as you have lower target elements, all libraries perform well relatively; but when there are lot more target elements, the algorithmic differences become much more obvious.

The loop over target elements is typically the sequential part in the computation. We have stressed in multiple places how we can accelerate by using OpenMP threading or task-based programming models specifically for intersection computation and also in the TempestRemap online weight matrix generation. While we don't have any results at the moment to show performance gains with such hybrid implementations, we expect to leverage the finer grain parallelism in the next iteration of the implementation refinements.
Manuscript Diff: No changes.

- Referee: Page 18, line 29 "serial runs" vs page 20, line 1 "better performance in MBTempest : : : offers avenues to incorporate task level parallelism : : :". Are these serial runs or something else? Serial in MPI but using shared memory parallelism? Is that still serial?
  Authors Yes we are referring to serial in MPI but parallelism introduced either through threads or task-based programming. Since TempestRemap is a pure serial code (no MPI/OpenMP support), we had to compare serial performance on the same architectures to draw computational throughput conclusions. As mentioned above, we will include shared memory parallelism as a separate future study when we have implemented threading and/or task-based parallelism in both MOAB and perhaps TempestRemap.
  Manuscript Diff: No changes.

- Referee: Page 20, lines 18-19. For the 1024^3 test case, are weights being generated in 2d or 3d? If 3d, is this test case an order of magitude (or more) larger than the largest climate model grids? Might be worth clairfying in text.
  Authors This was a full 3-D test case. Yes we used a very high-res run to showcase strong scalability of the point location algorithm in MOAB. While current pro-

duction level runs still have lower DoFs compared to this study, there has been a lot of interest in doing sub-Km atmosphere resolution studies, which will push the boundaries of what is required from remapping libraries.
Manuscript Diff: No changes.

- Referee: Page 20, lines 24-26. I agree that the initialization cost is amortized for long production climate runs. But in your example, that init cost is order (hundreds) of seconds (see fig 10c). That is for a single set (pair?) of weights. In coupled climate model, there are often order (10) of these to be done. Now we're talking 1000s of seconds which starts to sound expensive in production but is certainly very expensive for short development test runs. Would it be cheaper to store the weights in a file and read them in the next time? (see general comments). Having said that, please confirm that weights are generated on each of the 1 billion gridcells (1024^3) in 3d. And if so, that's a lot of gridcells.
  Authors Agreed. This is a deficiency of the Kd-tree datastructure and as mentioned in the manuscript, we intend to add BVH implementations where the overall cost for the tree construction is much smaller. $O(nlog(n))$ in Kd-tree vs $O(log(n))$ in BVH-tree. The BVH implementation is a little complex and so we do not have this working correctly for large cases yet in MOAB.
  Manuscript Diff: No changes.

- Referee: Page 21, Figure 10, I am struggling to read the axes and other text on the plots
  Authors At 100% zoom in the pdf using our Adobe reader, the axes are clearly visible. However, we can try to modify the fonts slightly to get better resolution in the images.
  Manuscript Diff: Slightly zoomed image.

- Referee: Page 21, Figure 10b shows scaling to 512k pes for a problem size of 1024^3. The final point has 2000 gridcells per process which is still relatively big.

What if you chose a problem size of 128^3 or 256^3 and tried to scale to 512k cores?
Authors The complexity scales as O(nlog(n)). So if we decrease the total n, the total work required reduces as well, which will transition more into the memory bandwidth bound regime. So while the overall time to solution may be much lower, the strong scalability may be lower as well as expected.
Manuscript Diff: No changes.

- Referee: Page 25, figure 13. Is there benefit to showing the three results (colocated plus two disjoint). The results are very similar for the three cases, at least as presented. And there is no discussion of the differences/similarities in text.
Authors One of the key points that we wanted to highlight was the relative indifference of the algorithms to the type of PE partitioning. When we originally looked at this study, it was our belief that the fully disjoint case would perform the worst and having any level of overlap with the coupler PEs would reduce the total amount of communication for both the mesh and data. While this may be true with really strict partitioning strategies, giving the components control over how the underlying grid is partitioned results in a nearly independent rate of scalability; this is especially evident when you look at the coverage mesh computation time that shows similar trends in all three cases.

  We will add additional text in the manuscript to point out this particular conclusion from the study, which was non-intuitive at first during our experimentation.
Manuscript Diff: p.25 l.20 - p.26 l.23

- Referee: Page 26, figure 14b. I am surprised there is so little scaling of the send/recv at NE120 and the core counts presented. I understand the claim that the absolute cost is small in all cases. I guess you are only redistributing 86k (NE120) elements, maybe that's expected then? At 128 cores, you should be transferring over 500 elements per core. Do you expect no scaling beyond that given the message size? Do you want to mention any of this in the paper?
Authors Fig. 14 (a) shows the actual mesh migration timing. And Fig. 14(b)

shows the send/receieve scaling for the actual field data from component to the coupler PEs. After the initial setup phase through the Crystal router algorithm during the mesh migration, all communications for field data are performed point-to-point from component to coupler PEs. The lack of scaling beyond 128 cores may be related to the size of the messages here. Since we are only measuring scalability of only one field transfer here, the latency for message creation and sending (non-blocking) still dominates the actual scaling timings; we intend to follow up this study with aggregated, multi-field transfers between atm-ocn, which should show better (lower uncertainty) point-to-point communication scaling.
Manuscript Diff: p.25 l.20 - p.26 l.23, p.28 l.8-l.9

- Referee: The jump between 64 and 128 must be a machine thing, going offnode or something?
  Authors Yes this is correct. We have added additional discussions related to these results in the paper.
  Manuscript Diff: p.28 l.13 - l.16

- Referee: Please confirm that you describe which machine the tests are run on in text and it might be beneficial to include that information in the figure captions. For page 27, figure 15, maybe remind us that it's case B of Table 1 (I think that's correct) in text.
  Authors These have been mentioned in each corresponding section for serial, parallel runs e.g., 4.1, 4.2, 4.3.1. We have also modified other sections and figures where this was not clear.
  Manuscript Diff: p.25 l.19, p.27 Fig.13 caption, p.28 l.14, p.29 Fig. 14 caption, p.30 Fig. 15 caption

Improving climate model coupling through a complete mesh representation: a case study with E3SM (v1) and MOAB (v5.x)

Vijay S. Mahadevan[1], Iulian Grindeanu[1], Robert Jacob[1], and Jason Sarich[1]

[1]Argonne National Laboratory, 9700 S. Cass Avenue, Lemont, IL

Correspondence: V. S. Mahadevan (mahadevan@anl.gov)

Abstract.

One of the fundamental factors contributing to the spatiotemporal inaccuracy in climate modeling is the mapping of solution field data between different discretizations and numerical grids used in the coupled component models. The typical climate computational workflow involves evaluation and serialization of the remapping weights during the pre-processing step, which is then consumed by the coupled driver infrastructure during simulation to compute field projections. Tools like Earth System Modeling Framework (ESMF) ~~Hill et al. (2004) and TempestRemap Ullrich et al. (2013)~~ (Hill et al., 2004) and TempestRemap (Ullrich et al., 2013) offer capability to generate conservative remapping weights, while the Model Coupling Toolkit (MCT) ~~Larson et al. (2001)~~ (Larson et al., 2001) that is utilized in many production climate models exposes functionality to make use of the operators to solve the coupled problem. However, such multi-step processes present several hurdles in terms of the scientific workflow, and impedes research productivity. In order to overcome these limitations, we present a fully integrated infrastructure based on the Mesh Oriented datABase (MOAB) ~~Tautges et al. (2004); Mahadevan et al. (2015)~~ (Tautges et al., 2004; which allows for a complete description of the numerical grids, and solution data used in each submodel. Through a scalable advancing front intersection algorithm, the supermesh of the source and target grids are computed, which is then used to assemble the high-order, conservative and monotonicity preserving remapping weights between discretization specifications. The Fortran compatible interfaces in MOAB are utilized to directly link the submodels in the Energy Exascale Earth System Model (E3SM) to enable online remapping strategies in order to simplify the coupled workflow process. We demonstrate the superior computational efficiency of the remapping algorithms in comparison with other state-of-science tools and present strong scaling results on large-scale machines for computing remapping weights between the spectral-element atmosphere and finite-volume discretizations on the polygonal ocean grids.

1 Introduction

Understanding Earth's climate evolution through robust and accurate modeling of the intrinsically complex, coupled ocean-atmosphere-land-ice-biosphere models requires extreme-scale computational power ~~Washington et al. (2008)~~ (Washington et al., 2008) . In such coupled applications, the different component models may employ unstructured spatial meshes that are specifically generated to resolve problem-dependent solution variations, which introduces several challenges in performing a consistent solution coupling. It is known that operator decomposition and unresolved coupling errors in partitioned atmosphere and ocean model simulations ~~Beljaars et al. (2017)~~ (Beljaars et al., 2017) , or physics and dynamics components of an atmosphere, can lead to large approximation errors that cause severe numerical stability issues. In this context, one factor contributing to the spatiotemporal accuracy is the mapping between different discretizations of the sphere used in the components of a coupled climate model. Accurate remapping strategies in such multi-mesh problems are critical to preserve higher order resolution, but are in general computationally expensive given the disparate spatial scales across which conservative projections are calculated. Since the primal solution or auxiliary derived data defined on a ~~source~~ donor physics component mesh (~~donor~~ source model) needs to be transferred to its coupled dependent physics mesh (target model), robust numerical algorithms are necessary to preserve discretization accuracy during these operations ~~Grandy (1999); de Boer et al. (2008)~~ (Grandy, 1999; de Boer et al., 2008) , in addition to conservation and monotonicity properties in the field profile.

An important consideration is that in addition to maintaining the overall discretization accuracy of the solution during remapping, global conservation, and sometimes local element-wise conservation for ~~quantities Jiao and Heath (2004)~~ critical quantities (Jiao and Heath, 2004) needs to be imposed during the workflow. Such stringent requirements on key flux fields that couple components along boundary interfaces is necessary in order to mitigate any numerical deviations in coupled climate simulations. Note that these physics meshes are usually never embedded or include trivial linear transformations, which render existence of exact projection or interpolation operators unfeasible, even if the same continuous geometric topology is discretized in the models. Additionally, the unique domain decomposition used for each of the component physics meshes complicates the communication pattern during intra-physics transfer, since aggregation of point location requests need to be handled efficiently in order to reduce overheads during the remapping workflow ~~Plimpton et al. (2004); Tautges and Caceres (2009)~~ (Plimpton et al., 2004; Tau

Adaptive block-structured cubed-sphere or unstructured refinement of icosahedral/polygonal meshes ~~Slingo et al. (2009)~~ (Slingo et al., 2 often used to resolve the complex fluid dynamics behavior in atmosphere and ocean models efficiently. In such models, conservative, local flux-preserving remapping schemes are critically important ~~Berger (1987)~~ (Berger, 1987) to effectively reduce multimesh errors, especially during computation of tracer advection such as water vapor or CO_2 ~~Lauritzen et al. (2010)~~ (Lauritzen et al., 20 This is also an issue in atmosphere models where physics and dynamics are computed on non-embedded grids ~~Dennis et al. (2012)~~ (Dennis e and the improper spatial coupling between these multi-scale models could introduce numerical artifacts. Hence, the availability of different consistent and accurate remapping schemes under one flexible climate simulation framework is vital to better understand the pros and cons of the adaptive multiresolution choices ~~Reichler and Kim (2008)~~ (Reichler and Kim, 2008) .

1.1 Hub-and-Spoke vs Distributed Coupling Workflow

The hub-and-spoke centralized model as shown in Fig. 1 (left) is used in the current Exascale Earth System Model (E3SM) driver, and relies on several tools and libraries that have been developed to simplify the regridding workflow within the climate community. Most of the current tools used in E3SM and the Community Earth System Model (CESM) ~~Hurrell et al. (2013)~~ (Hurrell et al., 2 included in a single package called the Common Infrastructure for Modeling the Earth (CIME), which builds on previous couplers used in CESM ~~Craig et al. (2005, 2012)~~ (Craig et al., 2005, 2012) . These modeling tools approach the problem in a two-step computational process~~:~~:

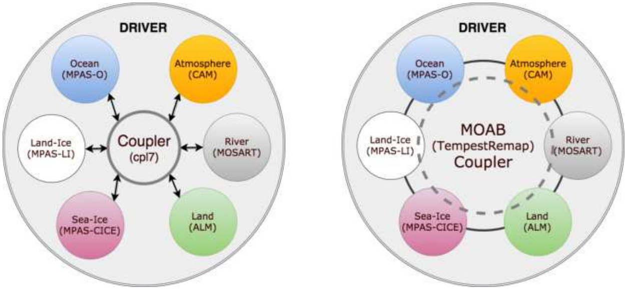


Figure 1. E3SM Coupled Climate Solver: (a) Current model (left), (b) Newer MOAB based coupler (right).

1. Compute the projection or remapping weights for a solution field from a source component physics to a target component physics as an offline process

2. During runtime, the CIME coupled solver loads the remapping weights from a file, and handles the partition-aware communication and weight matrix application to project coupled fields between components

The first task in this workflow is currently accomplished through a variety of standard state-of-science tools such as the Earth Science Modeling Framework (ESMF) ~~Hill et al. (2004)~~ (Hill et al., 2004) , Spherical Coordinate Remapping and Interpolation Package (SCRIP) ~~Jones (1999) , TempestRemap Ullrich et al. (2013); Ullrich and Taylor (2015)~~ (Jones, 1999) , TempestRemap (Ullrich et al., 2013; Ullrich and Taylor, 2015) . The Model Coupling Toolkit (MCT) ~~Larson et al. (2001); Jacob et al. (200~~ in the CIME solver provides data structures for the second part of the workflow. Traditionally the first workflow phase is executed decoupled from the simulation driver during a pre-processing step, and hence any updates to the field discretization or the underlying mesh resolution immediately necessitates recomputation of the remapping weight generation workflow with updated inputs. This process flow also prohibits the component solvers from performing any runtime spatial adaptivity, since the remapping weights have to be re-computed dynamically after any changes in grid positions. To overcome such deficiencies, and to accelerate the current coupling workflow, recent efforts have been undertaken to implement a fully integrated remapping weight generation process within E3SM using a scalable infrastructure provided by the topology, decomposition and data-aware Mesh Oriented datABase (MOAB) ~~Tautges et al. (2004); Mahadevan et al. (2015) and TempestRemap Ullrich et al. (2013)~~ (Tautges et al., 2004; Mahadevan et al., 2015) and TempestRemap (Ullrich et al., 2013) software libraries as shown in Fig. 1 (right). Note that whether a hub-and-spoke or distributed coupling model is used to drive the simulation, a minimal layer of driver logic is necessary to compute weighted combination of fluxes, validation metrics, and other diagnostic outputs.

The paper is organized as follows. In Section. (2), we present the necessary background and motivations to develop an online remapping workflow implementation in E3SM. Section. (3) covers details on the scalable, mesh and partition aware, conservative remapping algorithmic implementation to improve scientific productivity of the climate scientists, and to simplify the overall computational workflow for complex problem simulations. Then, the performance of these algorithms are first evaluated in serial for various grid combinations, and the parallel scalability of the workflow is demonstrated on large-scale machines in Section. (4).

2 Background

Conservative remapping of ~~solution fields that nonlinearly couple multiple physics components~~ nonlinearly coupled solution fields is a critical task to ensure consistency and accuracy in climate and numerical weather prediction simulations ~~Slingo et al. (2009)~~ (Sling While there are various ways to compute a projection of a solution defined on a source grid Ω_S to a target grid Ω_T, the requirements related to global or local conservation in the remapped solution reduces the number of potential algorithms that can be employed for such problems.

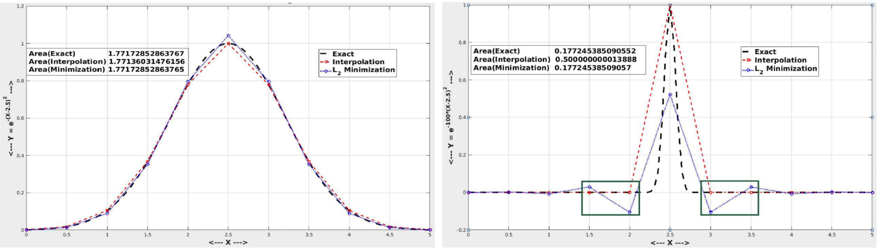
Depending on whether (global or local) conservation is important, and if higher-order, monotone interpolators are required, there are several consistent algorithmic options that can be used ~~de Boer et al. (2008)~~ (de Boer et al., 2008) . All of these different remapping schemes usually have one of these characteristic traits: non-conservative (**NC**), globally-conservative (**GC**) and locally-conservative (**LC**).

1. **NC/GC**: Solution interpolation approximations
   - **NC**: (Approximate or exact) nearest neighbor interpolation
   - **NC/GC**: Radial Basis Function (RBF) ~~Flyer and Wright (2007)~~ (Flyer and Wright, 2007) interpolators and patch-based Least Squares reconstructions ~~Zienkiewicz and Zhu (1992); Fleishman et al. (2005)~~ (Zienkiewicz and Zhu, 1992; Fleishı
   - **GC**: Consistent ~~FEM~~ Finite Element (FE) interpolation and area re-normalization
2. **LC/GC**: L_2 or H_1 projection
   - **LC/GC**: Embedded ~~FEM~~FE/FD/FV meshes in adaptive computations
   - **LC**: Intersection-based field integrators with consistent higher-order discretization ~~Jones (1999)~~ (Jones, 1999)
   - **LC**: Constrained projections to ensure conservation ~~Berger (1987); Aguerre et al. (2017) and monotonicity Rančić (1995)~~ (Berg monotonicity (Rančić, 1995)

Typically in climate applications, flux fields are interpolated using first-order (locally) conservative interpolation, while other scalar fields use non-conservative but higher-order interpolators (e.g. bilinear or biquadratic). For scalar solutions that do not need to be conserved, consistent ~~FEM~~ FE interpolation, patch-wise reconstruction schemes ~~Fornberg and Piret (2008)~~ (Fornberg and Piret, even nearest neighbor interpolation ~~Blanco and Rai (2014)~~ (Blanco and Rai, 2014) can be performed efficiently using Kd-tree based search and locate infrastructure. Vector fields like velocities or wind stresses are interpolated using these same routines by separately tackling each Cartesian-decomposed component of the field. However, conservative remapping of flux fields require computation of a supermesh ~~Farrell and Maddison (2011)~~ (Farrell and Maddison, 2011) , or a global intersection mesh that can be viewed as $\Omega_S \bigcup \Omega_T$, which is then used to compute projection weights that contain additional conservation and monotonicity constraints embedded in them.

In general, remapping implementations have three distinct steps to accomplish the solution field projection between grids. First, the target points of interest are identified and located in the source grid, such that, the target cells are a subset of the covering (source) mesh. Next, an intersection between this covering (source) mesh and the target mesh is performed, in order to calculate the individual weight contribution to each target cell, ~~while consistently respecting the underlying discretization of the field data~~without approximations to the component field discretizations (type $\in [FV, FEM]$ and order). Finally, application of the weight matrix yields the projection required to conservatively transfer the data onto the target grid.

To illustrate some key differences between some **NC** to **GC** or **LC** schemes, we show a 1-D Gaussian hill solution, projected onto a coarse grid through linear basis interpolation and L_2 minimization, as shown in Fig. 2. While the point-wise linear interpolator is computationally efficient, and second-order accurate (Fig. 2-(a)) for smooth profiles, it does not preserve the exact area under the curve. In contrast, the L_2 minimizer conserves the global integral area, but can exhibit spurious oscillatory modes as shown in Fig. 2-(b), when dealing with solutions with strong gradients (Gibbs phenomena ~~Gottlieb and Shu (1997)~~ (Gottlieb and Shu, 1997)). This demonstration confirms that even for the simple 1-D example, a conservative and monotonic projector is necessary to preserve both stability and accuracy for repeated remapping operator applications, in order to accurately transfer fields between grids with very different resolutions. These requirements are magnified manyfold when dealing with real-world climate simulation data.



(a) Gaussian hill $(x_0, \delta) = (2.5, 1.0)$ (b) Gaussian hill $(x_0, \delta) = (2.5, 0.1)$

Figure 2. An illustration: comparing point interpolation vs L_2 minimization; impact on conservation and monotonicity properties.

While there is a delicate balance in optimizing the computational efficiency of these operations without sacrificing the numerical accuracy or consistency of the procedure, several researchers have implemented algorithms that are useful for a variety of problem domains. In the recent years, the growing interest to rigorously tackle coupled multiphysics applications has led to research efforts focused on developing new regridding algorithms. The Data Transfer Kit (DTK) ~~Slattery et al. (2013)~~ (Slattery et al., 2013) fr

Oak Ridge National Labs was originally developed for Nuclear engineering applications, but has been extended for other problem domains through custom adaptors for meshes. DTK is more suited for non-conservative interpolation of scalar variables with either mesh-aware (using consistent discretization bases) or RBF-based meshless (point-cloud) representations ~~Slattery (2016)~~ (Slattery, 2016) that can be extended to model transport schemes on a sphere ~~Flyer and Wright (2007)~~ (Flyer and Wright, 2(The Portage library ~~Herring et al. (2017)~~ (Herring et al., 2017) from Los Alamos National Laboratory also provides several key capabilities that are useful for geology and geophysics modeling applications including porous flow and seismology systems. Using advanced clipping algorithms to compute the intersection of axis-aligned squares/cubes against faces of a triangle/tetrahedron in 2-d and 3-d respectively, general intersections of arbitrary convex polyhedral domains can be computed efficiently ~~Powell and Abel (2015)~~ (Powell and Abel, 2015) . Support for conservative solution transfer between grids and bound-preservation (to ensure monotonicity) ~~Certik et al. (2017)~~ (Certik et al., 2017) has also been recently added. While Portage does support hybrid level parallelism (MPI + OpenMP), demonstrations on large-scale machines to compute remapping weights for climate science applications has not been pursued previously. ~~It is also unclear whether DTK and Portage support~~ Based on the software package documentation, support for remapping of vector fields with conservation constraints ~~to be of direct~~ in DTK and Portage is not directly available for use in climate workflows. Additionally, unavailability of native support for projection of high-order spectral-element data on a sphere onto a target mesh restricts the use of these tools for certain component models in E3SM.

In earth science applications, the state-of-science regridding tool that is often used by many researchers is the ESMF library, and the set of utility tools that are distributed along with it ~~Collins et al. (2005); Dunlap et al. (2013)~~ (Collins et al., 2005; Dunlap et al., 201 to simplify the traditional offline-online computational workflow as described in Section. (1.1). ESMF is implemented in a component architecture (Zhou, 2006) and provides capabilities to generate the remapping weights for different discretization combinations on the source and target grids in serial and parallel. ESMF provides a standalone tool, ESMF_REGRIDWEIGHTGEN, to generate *offline* weights that can be consumed by climate applications such as E3SM and OASIS3-MCT. ESMF also exposes interfaces that enable drivers to directly invoke the remapping algorithms in order to enable the fully-online workflow as well.

Currently, the E3SM components are integrated together in a hub-and-spoke model (Fig. 1 (left)), with the inter-model communication being handled by the Model Coupling Toolkit (MCT) ~~Larson et al. (2001); Jacob et al. (2005a)~~ (Larson et al., 2001; Jacob et al. CIME. The MCT library consumes the offline weights generated with ESMF or similar tools, and provides the functionality to interface with models, decompose the field data, and apply the remapping weights loaded from a file during the setup phase. Hence, MCT serves to abstract the communication of data in the E3SM ecosystem. However, without the offline remapping weight generation phase for fixed grid resolutions and model combinations, the workflow in Fig. 1 (a) is incomplete.

Similar to the CIME-MCT driver used by E3SM, OASIS3-MCT ~~Valcke (2013); Craig et al. (2017)~~ (Valcke, 2013; Craig et al., 2017) is a coupler used by many European climate models, where the interpolation weights can be generated offline through SCRIP (included as part of OASIS3-MCT). An option to call SCRIP in an online mode is also available. The OASIS team have recently parallelized SCRIP to speed up its calculation time ~~?~~ (OASIS3-MCT v4.0, 2018) . OASIS3-MCT also supports application of global conservation operations after interpolation, and does not require a strict hub-and-spoke coupler. Similar to the coupler in CIME, OASIS3-MCT utilizes MCT to perform both the communication of fields between components and for application of the pre-computed interpolation weights in parallel.

ESMF and SCRIP traditionally handle only cell-centered data that targets Finite Volume discretizations (FV to FV projections), with first-order conservation constraints. Hence, generating remapping weights for atmosphere-ocean grids with a Spectral Element (SE) source grid definition requires generation of an intermediate and spectrally equivalent, '*dual*' grid, which matches the areas of the polygons to the weight of each ~~GLL node~~Gauss-Lobatto-Legendre (GLL) nodes. Such procedures add more steps to the offline process and can degrade the accuracy in the remapped solution since the original spectral order is neglected (transformation from p-order to first order). These procedures may also introduce numerical uncertainty in the coupled solution that could produce high solution dispersion ~~Ullrich et al. (2016)~~ (Ullrich et al., 2016) .

To calculate ~~offline~~ remapping weights directly for high-order Spectral Element grids, E3SM uses the TempestRemap C++ library ~~Ullrich et al. (2013)~~ (Ullrich et al., 2013) . TempestRemap is a uni-process tool focused on the mathematically rigorous implementations of the remapping algorithms ~~Ullrich and Taylor (2015); Ullrich et al. (2016)~~ (Ullrich and Taylor, 2015; Ullrich et al., 201 provides higher order conservative and monotonicity preserving interpolators with different discretization basis such as (Finite Volume (FV), the spectrally equivalent continuous Galerkin ~~Finite Element with Gauss-Lobatto quadrature~~ FE with GLL basis (cGLL), and dis-continuous Galerkin ~~Finite Element with Gauss-Lobatto quadrature~~ FE with GLL basis (dGLL)). This library was developed as part of the effort to fill the gap in generating consistent remapping operators for non-FV discretizations without a need for intermediate dual meshes. Computation of conservative interpolators between any combination of these discretizations (FV, cGLL, dGLL) and grid definitions are supported by TempestRemap library. However, since this regridding tool can only be executed in serial, the usage of TempestRemap prior to the work presented here has been restricted primarily to generating the required mapping weights in the offline stage.

Even though ESMF and OASIS3-MCT have been used in online remapping studies, weight generation as part of a pre-processing step currently remains the preferred workflow for ~~most~~ many production climate models. While this decoupling provides flexibility in terms of choice of remapping tools, the data management of the mapping files for different discretizations, field constraints and grids can render provenance, reproducibility and experimentation a difficult task. It also precludes the ability to handle moving or dynamically adaptive meshes in coupled simulations. However, it should be noted that the shift of the remapping computation process from a pre-processing stage in the workflow, to the simulation stage, imposes additional onus on the users to better understand the underlying component grid properties, their decompositions, the solution fields being transferred and the preferred options for computing the weights. This also raises interesting workflow modifications to ensure verification of the online weights such that consistency, conservation and dissipation of key fields are within user-specified constraints. In the implementation discussed here, the online remapping computation uses the exact same input grids, and specifications along with ability to write the weights to file, which can be used to run offline checks as needed.

There are several challenges in scalably computing the regridding operators in parallel, since it is imperative to have both a mesh- and partition-aware datastructure to handle this part of the regridding workflow. A few climate models have begun to calculate weights online as part of their regular operation. The ICON GCM ~~Wan et al. (2013) uses YAC Hanke et al. (2016) and FGOALS Li et al. (2013)~~ (Wan et al., 2013) uses YAC (Hanke et al., 2016) and FGOALS (Li et al., 2013) uses the C-Coupler

~~Liu et al. (2014, 2018)~~ (Liu et al., 2014, 2018) framework. These codes expose both offline and online remapping capabilities with parallel decomposition management similar to the ongoing effort presented in the current work for E3SM. Both of these packages provide algorithmic options to perform in-memory search and locate operations, interpolation of field data between meshes with first order conservative remapping, higher-order patch-recovery ~~Zienkiewicz and Zhu (1992)~~ (Zienkiewicz and Zhu, 1992) and RBF schemes and the NC nearest-neighbor queries. The use of non-blocking communication for field data in these packages align closely with scalable strategies implemented in MCT ~~Jacob et al. (2005b)~~ (Jacob et al., 2005b) . While these capabilities are used routinely in production runs for their respective models, the motivation for the work presented here is to tackle coupled high-resolution runs on next generation architectures with scalable algorithms (the high resolution E3SM coupler routinely runs on 13,000 mpi tasks), without sacrificing numerical accuracy for all discretization descriptions (FV, cGLL, dGLL) on unstructured grids.

In the E3SM workflow supported by CIME, the ESMF-regridder understands the component grid definitions, and generates the weight matrices (offline). The CIME driver loads these operators at runtime and places them in MCT datatypes, which treats them as discrete operators to compute the interpolation or projection of data on the target grids. Additional changes in conservation requirements or monotonicity of the field data cannot be imposed as a runtime or post-processing step in such a workflow. In the current work, we present a new infrastructure with scalable algorithms implemented using the MOAB mesh library and TempestRemap package to replace the ESMF-E3SM-MCT remapper/coupler workflow. A detailed review of the algorithmic approach used in the MOAB-TempestRemap (MBTR) workflow, along with the software interfaces exposed to E3SM is presented next.

3 Algorithmic approach

Efficient, conservative and accurate multi-mesh solution transfer workflows ~~Jacob et al. (2005b); Tautges and Caceres (2009)~~ (Jacob et al., a complex process. This is due to the fact that in order to ensure conservation of critical quantities in a given norm, exact cell intersections between the source and target grids have to be computed. This is complicated in a parallel setting since the domain decompositions between the source and target grids may not have any overlaps, making it a potentially all-to-all collective communication problem. Hence, efficient implementations of regridding operators need to be mesh, resolution, field and decomposition aware in order to provide optimal performance in emerging architectures.

Fully online remapping capability within a complex ecosystem such as E3SM requires a flexible infrastructure to generate the projection weights. In order to fullfill these needs, we utilize the MOAB mesh datastructure combined with the TempestRemap libraries in order to provide ~~a~~ an in-memory remapping layer to dynamically compute the weight matrices during the setup phase of the simulations ~~.~~ for static source-target grid combinations. For dynamically adaptive and moving grids, the remapping operator can be recomputed at runtime as needed. The introduction of such a software stack allows higher order conservation of fields while being able to transfer and maintain field relations in parallel, within the context of the fully decomposed mesh view. This is an improvement to the E3SM workflow where MCT is oblivious to the underlying mesh datastructure in the component models. Having a fully ~~mesh aware datastructure~~ mesh-aware implementation also provides opportunities to implement dynamic load-balancing algorithms to gain optimal performance on large-scale machines. YAC interpolator ~~Hanke et al. (2016)~~ (Hanke et al., 2016) and the multidimensional Common Remapping software (CoR) in C-Coupler2 ~~Liu et al. (2018)~~ (Liu et al., 2018) provide similar capabilities to perform a parallel tree-based search for point location and interpolation through various supported numerical schemes.

~~Let $N_{c,s}$~~ MOAB is a fully distributed, compact, array-based mesh datastructure, and the local entity lists are stored in ranges along with connectivity and ownership information, rather than explicit lists, thereby leading to a high degree of memory compression. The memory constraints per process scales well in parallel Tautges and Caceres (2009) , and is only proportional to the number of entities in the local partition, which reduces as number of processes increases (strong scaling limit). This is similar to the Global Segment Map (GSMap) in MCT, which in contrast is stored in every processor, leading to $O(N_x)$ memory requirements.

In order to illustrate the online remapping algorithm implemented with the MOAB-TempestRemap infrastructure, we define the following terms. Let $N_{c,S}$ be the component processes for source mesh, ~~$N_{c,t}$~~ $N_{c,T}$ be the component processes for target mesh and N_x be the coupler processes where the remapping operator is computed. More generally, the problem statement can be defined as: transfer a solution field U defined on the domain Ω_S and processes ~~$N_{c,s}$~~ $N_{c,S}$, to the domain Ω_T and processes ~~$N_{c,t}$~~ $N_{c,T}$, through a centralized coupler with domain information $\Omega_S \bigcup \Omega_T$ defined on N_x processes. Such a complex online remapping workflow for projecting the field data from a source to target mesh follows the algorithm shown in ~~Alg~~Algorithm. 1.

In the following sections, the new E3SM online remapping interface implemented with a combination of the MOAB and TempestRemap libraries is explained. Details regarding the algorithmic aspects to compute conservative, high-order remapping weights in parallel, without sacrificing discretization accuracy on next generation hardware are presented.

3.1 Interfacing to Component Models in E3SM

Within the E3SM simulation ecosystem, there are multiple component models (atmosphere-ocean-land-ice-runoff) that are coupled to each other. While the MCT infrastructure only allowed for a numbering of the grid points, the new MOAB-based coupler infrastructure provides the ability to natively interface to the underlying mesh, and understand the field ~~DoF~~ Degree-of-Freedom (DoF) data layout associated with each model. MOAB can understand the difference between values on a cell center and values on a cell edge or corner. In the current work, the MOAB mesh database has been used to create the relevant integration abstraction for the HOMME atmosphere model (Thomas and Loft, 2005; Taylor et al., 2007) (cubed-sphere SE grid) and the ~~MPAS ocean model~~ Model for Prediction Across Scales (MPAS) ocean model (Ringler et al., 2013; Petersen et al., 2015) (polygonal meshes with holes representing land and ice regions). Since ~~the~~ details of the mesh are not available at the level of the coupler interface, additional MOAB ~~calls~~ (Fortran) calls via the `iMOAB` interface are added to HOMME and MPAS component models to describe the details of the unstructured mesh to MOAB with explicit vertex and element connectivity information, in contrast to MCT coupler that is oblivious to the underlying grid. The atmosphere-ocean coupling requires the largest computational effort in the coupler (since they cover about 70% of the coupled domain), and hence ~~the bulk of the~~ bulk of discussions in the current work will focus on remapping and coupling between these two component models.

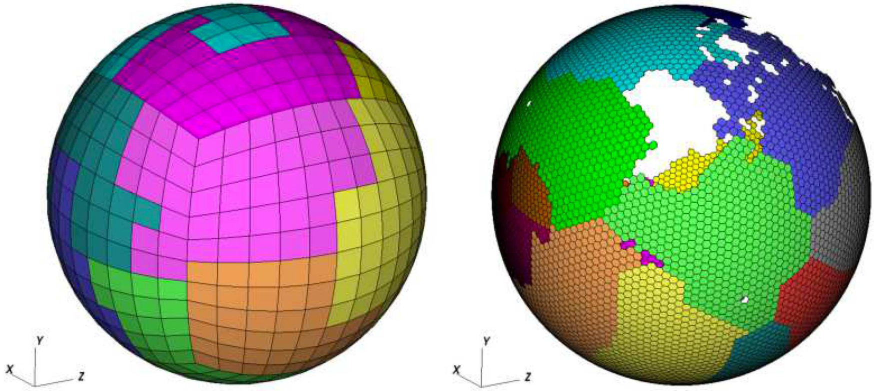
Algorithm 1 MOAB-TempestRemap parallel regridding workflow

```
Input: Partitioned and distributed native component meshes on ~~$N_{c,l}$~~ $N_{c,S}$ source and $N_{c,T}$ target processes
Result: Remapping weight matrix ~~$W_{ij}$~~ $W_{S\to T}$ computed for a source (~~$i$~~$S$) and target (~~$j$~~$T$) mesh pair on $N_x$ coupler processes

Scope: Coupler $N_x \leftarrow$ component mesh $N_{c,l}$, where $l \in [S,T]$
for each component $l \in [S,T]$ do
    – create in-memory copy of component ~~mesh/data with MOAB~~ unstructured mesh and data using MOAB interfaces (Section. 3.1)
    – migrate MOAB component mesh to coupler; repartition from $N_{c,l} \to N_x$ (Section. 3.2)
end for

Scope: Compute pair-wise intersection mesh on coupler processes $N_x$
for each mesh pair to be regridded: $\Omega_S$ and $\Omega_T$ in $N_x$ do
    Ensure: {local source mesh fully covers target mesh}
    if $(\Omega_T - \Omega_T \cap \Omega_S) \neq 0$ then
        collectively gather coverage mesh $\Omega_{Sc}$ on $N_x \mid (\Omega_T - \Omega_T \cap \Omega_{Sc}) = 0$ (Section. 3.4.1)
    end if
    – store communication graph to send/receive between $N_{c,l}$ ~~/~~and $N_x$
    – compute ~~$\Omega_{ST} = \Omega_{Sc} \bigcup \Omega_T$~~ $\Omega_{ST} = \Omega_{Sc} \cap \Omega_T$ through an advancing-front algorithm ~~Löhner and Parikh (1988); Gander and Japhet (2009)~~ (Löhner and Parikh, 1988; Gander and Japhet, 2009) (Section. 3.3.1)
    – evaluate source/target element mapping for $e_i \in \Omega_{ST}$
    – exchange ghost cell information for $\Omega_{ST}$
end for

Scope: Integrate over $\Omega_{ST}$ to compute remapping weights
for each intersection polygon element $e_i \in \Omega_{ST}$ do
    – Tessellate $e_i$ into triangular elements with reproducible ordering
    – Compute projection integral with consistent Triangular quadrature rules
    – Determine row/col DoF coupling through $e_i$ parent association to $\Omega_S/\Omega_T$
    – Assemble local matrix weights such that ~~$W_{ij} = \sum_1^{N_x} w_{ij}$~~ $W_{S\to T} = \sum_1^{N_x} w_{ij}$, where $w_{ij}$ represents the coupling between local target DoF (row $i$) and source DoF (col $j$) in projection operator (Section. 3.5)
end for
```



(a) HOMME-SE mesh

(b) MPAS mesh

Figure 3. MOAB representation of partitioned component meshes.

MOAB can handle the finite-element zoo of elements on a sphere (triangles, quadrangles, and polygons) making it an appropriate layer to store both the mesh layout (vertices, elements, connectivity, adjacencies) and the parallel decomposition for the component models along with information on shared and ghosted entities. While having a uniform partitioning methodology across components may be advantageous for improving the efficiency of coupled climate simulations, the parallel partition of the meshes are chosen according to the requirements in individual component solvers. Fig. 3 shows ~~an example of a replicated~~ examples of partitioned SE and MPAS meshes, visualized through the native MOAB plugin for VisIt (VisIt, 2005) .

The coupled field data that is to be remapped from the source grid to the target grid also needs to be serialized as part of the MOAB mesh database in terms of an internally contiguous, MOAB data storage structure named a 'Tag' (Tautges et al., 2004) . For E3SM, we use element-based tags to store ~~$n_p{}^2$ values per element , where for the atmosphere n_p~~ the partitioned field data that is required to be remapped between components. Typically, the number of DoF per element ($nDoF_e$) is determined based on the underlying discretization; $nDoF_e = p^2$ values in HOMME where p is the order of SE discretization~~and for MPAS ocean, $n_p = 1$~~, and $nDoF_e = 1$ for the FV discretization in MPAS ocean. With this complete description of the mesh and associated data for each component model, MOAB contains the necessary information to proceed with the remapping workflow.

3.2 Migration of Component Mesh to Coupler

E3SM's driver supports multiple modes of partitioning the various components in the global processor space. This is usually fine tuned based on the estimated computational load in each physics, according to the problem case definition. A sample

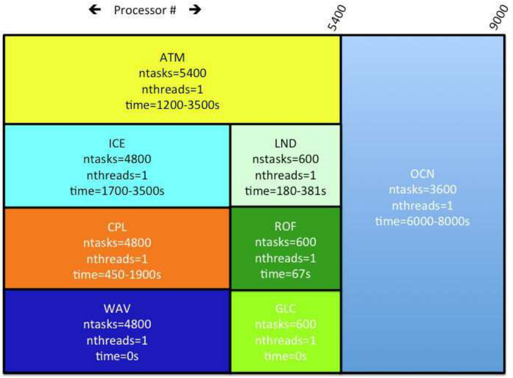
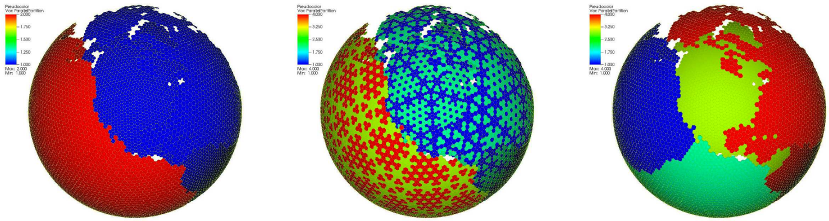


Figure 4. Example E3SM process execution layout for a problem case

~~process~~ process-execution (PE) layout for a E3SM run on 9000 processes with ATM on 5400 and OCN on 3600 tasks is shown in Fig. 4. In the case shown in the schematic, ~~$N_{c,atm} = 5400$, $N_{c,ocn} = 3600$~~ $N_{c,ATM} = 5400$, $N_{c,OCN} = 3600$ and $N_x = 4800$. In such a ~~processor execution~~ PE layout, the atmosphere component mesh from HOMME, distributed on ~~$N_{c,atm}$~~ $N_{c,ATM}$ (5400) tasks needs to be migrated and redistributed on N_x (4800 tasks). Similarly, from ~~$N_{c,ocn}$~~ $N_{c,OCN}$ (3600) to N_x (4800) tasks for the MPAS ocean mesh. ~~Since the remapping process~~ In the hub-and-spoke coupling model as shown in Fig. 1, the remapping computation is performed only in the coupler ~~processing elements within the hub-and-spoke model Fig. 1~~ processors. Hence, inference of a communication pattern becomes necessary to ensure scalable data transfers between the components and the coupler. In the existing implementation, MCT handles such communication, which is being replaced by point-to-point communication kernels in MOAB to transfer mesh and data between different components or component-coupler PEs. Note that in a distributed coupler, source and target components can communicate directly, without any intermediate transfers (through the coupler). Under the unified infrastructure provided by MOAB, minimal changes are required to enable either the hub-and-spoke or the distributed coupler for E3SM runs, which offers opportunities to minimize time to solution without any changes in spatial coupling behavior.

For illustration, let N_c be the number of component processing elements, and N_x be the number of coupler processing elements. In order to migrate the mesh and associated data from N_c to N_x, we first compute a trivial partition of elements that map directly in the partition space, the same partitioning as used in the CIME-MCT coupler. In MOAB, we have exposed parallel graph and geometric repartitioning schemes through interfaces to Zoltan ~~or ParMetis~~(Devine et al., 2002) or ParMetis (Karypis et al., 1997) , in order to evaluate optimized migration patterns to minimize the volume of data communicated between



(a) Component mesh on 2 tasks (b) Migrated mesh on 4 tasks (Trivial partitioner) (c) Migrated mesh on 4 tasks (Zoltan partitioner)

Figure 5. Migration strategies to repartition from $N_c \rightarrow N_x$

component and coupler~~processing units~~. We intend to analyze the impact of different migration schemes on the scalability of the remapping process in Section. (4). These optimizations have the potential to minimize data movement in the MOAB-based remapper, and to make it a competitive data broker to replace the current MCT ~~Jacob et al. (2005a)~~ (Jacob et al., 2005a) coupler in E3SM.

We show an example of a decomposed ocean mesh (polygonal MPAS mesh) that is replicated in a E3SM problem case run on two processes in Fig. 5. Fig. 5-(a) is the original decomposed mesh on 2 tasks $\in N_c$, while Fig. 5-(b) and Fig. 5-(c) show the impact of migrating a mesh from 2 N_c tasks to 4 tasks $\in N_x$ with a trivial linear partitioner and a Zoltan based geometric online ~~repartitioner~~partitioner. The decomposition in Fig. 5-(b) shows that the element ID based linear partitioner can produce bad data locality, which may require large number of nearest neighbor communications when computing a source coverage mesh. The resulting communication pattern ~~also makes the migration~~can also make the migration, and coverage computation process non-scalable on larger core counts. In contrast, in Fig. 5-(c), the Zoltan partitioners produce much better load balanced decompositions with Hypergraph (PHG), Recursive Coordinate Bisection (RCB) or Recursive Inertial Bisection (RIB) algorithms to reduce communication overheads in the remapping workflow. In order to better understand the impact of online decomposition strategies on the overall remapping process, we need to better understand the impact of the repartitioner on two communication-heavy steps.

1. Mesh migration from component to coupler involving communication between $N_{c,s/t}$ and N_x.

2. Computing the coverage mesh requiring gather/scatter of source mesh elements to cover local target elements.

In a hub-and-spoke model with online remapping, the best coupler strategy will require a simultaneous partition optimization for all grids such that mesh migration includes constraints on geometric coordinates of component pairs. While such extensions can be implemented within the infrastructure presented here, the performance discussions in Section 4 will only focus on the trivial and Zoltan-based partitioners. It is also worth noting that in a distributed coupler, pair-wise migration optimizations can be performed seamlessly using a master(*target*)-slave(*source*) strategy to maximize partition overlaps.

3.3 Computing the Regridding Operator

Standard approaches to compute the intersection of two convex polygonal meshes involve the creation of a Kd-tree (Hunt et al., 2006) or BVH-tree datastructure (Ize et al., 2007) to enable fast element location of relevant target points. In general, each target point of interest is located on the source mesh by querying the tree datastructure, and the corresponding (source) element is then marked as a contributor to the remapping weight computation of the target DoF. This process is repeated to form a list of ~~unique~~ source elements that interact directly according to the consistent discretization basis. TempestRemap, ESMF and YAC use variations of this search-and-clip strategy tailored to their underlying mesh representations.

3.3.1 Advancing Front Intersection – A Linear Complexity Algorithm

The intersection algorithm used in this paper follows the ideas from ~~Löhner and Parikh (1988); Gander and Japhet (2013)~~ (Löhner and Paril in which two meshes are covering the same domain. At the core is an advancing front method that aims to traverse through the source and target meshes to compute a union (super) mesh. First, two convex cells from the source coverage mesh and the target meshes that intersect are identified by using an adaptive Kd-tree search tree ~~constructed during the setup phase. This~~ datastructure. This process also includes determination of the seed for the advancing front. Advancing in both meshes using face adjacency information, incrementally all possible intersections are computed ~~Březina and Exner (2017)~~ (Březina and Exner, 2017) accurate to a user defined tolerance (default = $1e-15$).

While the advancing front algorithm is not restricted to convex cells, the intersection computation is simpler if they are strictly convex. If concave polygons exist in the initial source or target meshes, they are recursively decomposed into simpler convex polygons, by splitting along interior diagonals. Note that the intersection between two convex polygons ~~is~~ results in a strictly convex polygon. Hence, the underlying intersection algorithm remains robust to resolve even arbitrary non-convex meshes covering the same domain space.

Fig. 6 ~~shows~~ illustrates how the algorithm advances~~;~~. Each target cell is resolved by building a local queue of source cells that intersect the target cell. Source cells are added to ~~the~~ a local queue incrementally, using adjacency information. At the same time, a global queue with seeds is formed, ~~and it contains~~ containing pairs of source/target cells that have the ~~potential~~ probability to intersect. When there are no more source cells in the local queue, the algorithm advances to the next seed from the global queue, and the algorithm repeats. This workflow has been illustrated in both serial (Mahadevan et al., 2018a) and in parallel with partitioned meshes (Mahadevan et al., 2018b) .

This flooding-like advancing front needs a stable and robust methodology of intersecting edges/segments in two cells that belong to different meshes. Any pair of segments that intersect can appear in four different pairs of cells. A list of intersection points is maintained on each target edge, so that the intersection points are unique. Also, a geometric tolerance is used to merge intersection points that are close to each other, or if they are proximal to the original vertices in both meshes. Decisions regarding whether points are inside, outside or at the boundary of a convex enclosure are handled separately. If necessary, more

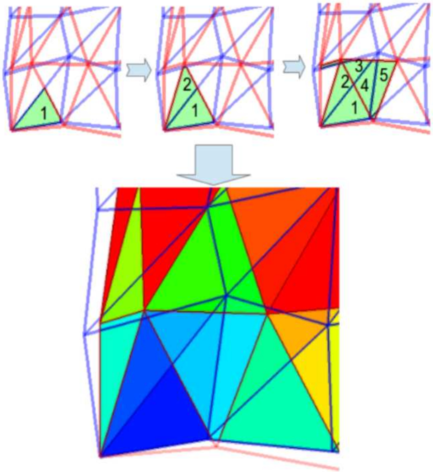


Figure 6. Illustration of the advancing front intersection algorithm.

robust techniques such as adaptive precision arithmetic procedures used in *Triangle* ~~Shewchuk (1996)~~ (Shewchuk, 1996) , can be employed to resolve the fronts more accurately. Note that the advancing front strategy can be employed for meshes with topological holes (e.g. ocean meshes, in which the continents are excluded) without any further modifications by using a new seed for each disconnected region in the target mesh.

Note on Gnomonic Projection for Spherical Geometry

Meshes that appear in climate applications are often on a sphere. Cell edges are considered to be great circle arcs. A simple gnomonic projection is used to project the edges on one of the six planes parallel to the coordinate axis, and tangent to the sphere ~~Ullrich et al. (2013)~~ (Ullrich et al., 2013) . With this projection, all curvilinear cells on the sphere are transformed to linear polygons on a gnomonic plane, which simplifies the computation of intersection between multiple grids. Once the intersection points and cells are computed on the gnomonic plane, these are projected back on to the original spherical domain without approximations. This is possible due to the fact that intersection can be computed to machine precision as the edges become straight lines in a gnomonic plane (projected from great circle arcs on a sphere). If curves on a sphere are not great circle arcs (splines, for example), the intersection between those curves have to be computed using some nonlinear iterative procedures such as Newton Raphson (depending on the representation of the curves).

3.4 Parallel Implementation Considerations

Existing Infrastructure from MOAB ~~Tautges et al. (2004)~~ (Tautges et al., 2004) was used to extend the advancing front algorithm in parallel. The expensive intersection computation can be carried out independently, in parallel, once we redistribute the source mesh to envelope the target mesh areas fully, in a step we refer to as ‘source coverage mesh’ computation.

3.4.1 Computation of a Source Coverage Mesh

We select the target mesh as the driver for redistribution of the source mesh. On each task, we first compute the bounding box of the local target mesh. This information is then gathered and communicated to all tasks, and used for redistribution of the local source mesh. Cells that intersect the bounding boxes of other processors are sent to the corresponding owner task. This workflow guarantees that the target mesh on each processor is completely enveloped by the covering mesh repartitioned from its original source mesh decomposition, as shown in Fig. 7. In other words, the covering mesh is a superset of the target mesh in each task. It is important to note that some source coverage cells might be sent to multiple processors during this step, depending on the target mesh resolution and decomposition. The parallel infrastructure in MOAB is heavily leveraged ~~Tautges et al. (2012)~~ (Tautges et al., 2012) to utilize the scalable, *crystal router algorithm* *(Fox et al., 1989; Schliephake and Laure, 2015)* in order to scalably communicate the covering cells to different processors.

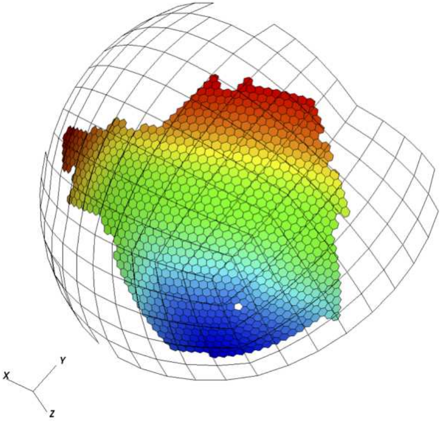


Figure 7. Source coverage mesh fully ~~cover~~ covers local target mesh; local intersection proceeds between atmosphere (Quadrangle) and ocean (Polygonal) grids.

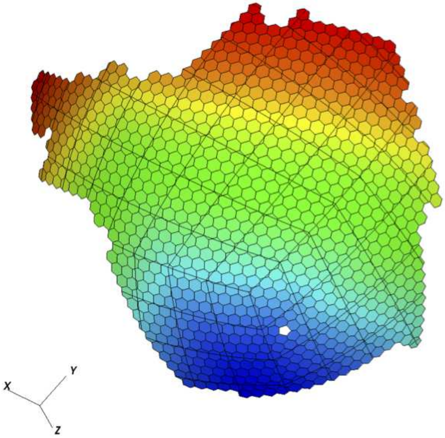


Figure 8. Intersection mesh computed with the coverage and target mesh in a single process.

Once the relevant covering mesh is accumulated locally on each process, the intersection computation can be carried out in parallel, completely independently, using the advancing front algorithm (Section. (3.3.1)), as shown in Fig. 8. ~~Once each task computes its share of~~ After computation of the local intersection polygons, the ~~intersection~~ vertices on the shared edges between processes ~~needs to be~~ are communicated to avoid duplication. In order to ensure consistent local conservation constraints in the weight matrix in the parallel setting, there might be a need for additional communication of ghost intersection elements to nearest neighbors. This extra communication step is only required for computing interpolators for flux variables~~and can~~ , and can generally be avoided when transferring scalar fields with non-conservative bilinear or higher-order interpolations~~in this workflow~~.

The parallel advancing front algorithm presented here to globally compute the intersection supermesh can be extended to expose finer grained parallelism ~~with~~ using hybrid-threaded (OpenMP) programming or a task-based execution ~~models~~model, where each task handles a unique front in the computation queue. ~~With a local mesh decomposition with Metis or through coloring, each~~ Such task or hybrid threaded parallelism can be employed in combination with the MPI-based mesh decompositions. Using local partitions computed with Metis and through standard coloring approaches, each thread or task can then proceed to compute the intersection elements until the front collides with another~~from a different thread,~~ , and until all the overlap elements have been computed in each process. Such a ~~hybrid MPI and threading~~ parallel hybrid algorithm has the potential to scale well even on heterogeneous architectures and provides options to improve the computational throughput of the regridding process ~~Löhner (2014)~~ (Löhner, 2014) .

3.5 Computation of Remapping operator with TempestRemap

For illustration, consider a scalar field $\boldsymbol{U}$ discretized with standard Galerkin FEM on source Ω_1 and target Ω_2 meshes with different resolutions. The projection of the scalar field on the target grid is in general given as follows.

$$\boldsymbol{U}_2(\boldsymbol{\Omega_2}) = \boldsymbol{\Pi}_1^2 \boldsymbol{U}_1(\boldsymbol{\Omega_1}) \tag{1}$$

where, $\boldsymbol{\Pi}_1^2$ is the discrete solution interpolator of $\boldsymbol{U}$ defined on $\boldsymbol{\Omega_1}$ to $\boldsymbol{\Omega_2}$. This interpolator $\boldsymbol{\Pi}_1^2$ in Eq. (1) is often referred to as the remapping operator, which is pre-computed in the coupled climate workflows using ESMF and TempestRemap. For embedded meshes, the remapping operator can be calculated exactly as a restriction or prolongation from the source to target grid. However, for general unstructured meshes and in cases where the source and target meshes are topologically different, the numerical integration to assemble $\boldsymbol{\Pi}_1^2$ needs to be carried out on the supermesh ~~Ullrich and Taylor (2015)~~ (Ullrich and Taylor, 2015) . Since a unique source and target parent element exists for every intersection element belonging to the supermesh $\boldsymbol{\Omega_1} \bigcup \boldsymbol{\Omega_2}$, $\boldsymbol{\Pi}_1^2$ is assembled as the sum of local mass matrix contributions on the intersection elements, by using the consistent discretization basis for the source and target field descriptions ~~Ullrich et al. (2016)~~ (Ullrich et al., 2016) . The intersection mesh typically contains arbitrary convex polygons and hence subsequent triangulation may be necessary before evaluating the integration. This global linear operator directly couples source and target DoFs based on the participating intersection element parents ~~Ullrich et al. (2009)~~ (Ullrich et al., 2009) .

MOAB supports point-wise FEM interpolation (bilinear and higher-order spectral) with local or global subset normalization ~~Tautges and Caceres (2009)~~ (Tautges and Caceres, 2009) , in addition to a conservative first-order remapping scheme. But higher order conservative monotone weight computations are currently unsupported natively. To fill this gap for climate applications, and to leverage existing developments in rigorous numerical algorithms to compute the conservative weights, interfaces to TempestRemap in MOAB were added to scalably compute the remap operator in parallel, without sacrificing field discretization accuracy. The MOAB interface to the E3SM component models provides access to the underlying type and order of field discretization, along with the global partitioning for the DoF numbering. Hence the projection or the weight matrix can be assembled in parallel by traversing through the intersection elements, and associating the appropriate source and target DoF parent to columns and rows respectively. The MOAB implementation uses a sparse matrix representation using the Eigen3 library (Guennebaud et al., 2010) to store the local weight matrix. Except for the particular case of projection onto a target grid with cGLL description, the matrix rows do not share any contributions from the same source DoFs. This implies that for FV and dGLL target field descriptions, the application of the weight matrix does not require global collective operations and sparse matrix vector applications scale ideally (still memory bandwidth limited). In the cGLL case, we perform a reduction of the parallel vector along the shared DoFs to accumulate contributions exactly. However, it is non-trivial to ensure full bit-for-bit reproducibility during such reductions.

It is also possible to use the transpose of the remapping operator computed between a particular source and target component combination, to project the solution back to the original source grid. Such an operation has the advantage of preserving the consistency and conservation metrics originally imposed in finding the remapping operator and reduces computation cost by avoiding recomputation of the weight matrix for the new directional pair. For example, when computing the remap operator between atmosphere and ocean models (with holes), it is advantageous to use the atmosphere model as the source grid, since the advancing front seed computation may require multiple iterations if the front begins within a hole. Additionally, such transpose vector applications can also make the global coupling symmetric, which may have favorable implications when pursuing implicit temporal integration schemes.

3.6 Note on MBTR Remapper Implementation

The remapping algorithms presented in the previous section are exposed through a combination of implementations in MOAB and TempestRemap libraries. Since both the libraries are written in C++, direct inheritance of key datastructures such as the GridElements (mesh) and OfflineMap (projection weights) are available to minimize data movement between the libraries. Additionally, Fortran codes such as E3SM can invoke computations of the intersection mesh and the remapping weights through specialized language-agnostic interfaces in MOAB: `iMOAB` ~~Mahadevan et al. (2015)~~ (Mahadevan et al., 2015) . These interfaces offer the flexibility to query, manipulate and transfer the mesh between groups of processes that represent the component and coupler processing elements.

Using the iMOAB interfaces, the E3SM coupler can coordinate the online remapping workflow during the setup phase of the simulation, and compute the projection operators for component and scalar or vector coupled field combinations. For each pair of coupled components, the following sequence of steps are then executed to consistently compute the remapping operator and transfer the solution fields in parallel.

1. `iMOAB_SendMesh` and `iMOAB_ReceiveMesh`: Send the component mesh (defined on $N_{c,l}$ processes), and receive the complete unstructured mesh copy in the coupler processes (N_x). This mesh migration undergoes an online mesh repartition either through a trivial decomposition scheme or with advanced Zoltan algorithms (geometric or graph partitioners)

2. `iMOAB_ComputeMeshIntersectionOnSphere`: The advancing front intersection scheme is invoked to compute the overlap mesh in the coupler processes

3. `iMOAB_CoverageGraph`: Update the parallel communication graph based on the (source) coverage mesh association in each process

4. `iMOAB_ComputeScalarProjectionWeights`: The remapping weight operator is computed and assembled with discretization-specific (FV, SE) calls to TempestRemap, and stored in Eigen3 SparseMatrix object

Once the remapping operator is serialized in-memory for each coupled scalar and flux fields, this operator is then used at every timestep to compute the actual projection of the data.

1. `iMOAB_SendElementTag` and `iMOAB_ReceiveElementTag`: Using the coverage graph computed previously, direct one-to-one communication of the field data is enabled between $N_{c,l}$ and N_x, before and after application of the weight operator

2. iMOAB_ApplyScalarProjectionWeights: In order to compute the field interpolation or projection from the source component to the target component, a matvec product of the weight matrix and the field vector defined on the source grid is performed. The source field vector is received from source processes $N_{c,s}$ and after weight application, the target field vector is sent to target processes $N_{c,l}$

Additionally, to facilitate offline generation of projection weights, a MOAB based parallel tool mbtempest has been written in C++, similar to ESMF and TempestRemap (serial) standalone tools. mptempst can load the source and target meshes from files, in parallel, and compute the intersection and remapping weights through TempestRemap. The weights can then be written back to a SCRIP-compatible file format, for any of the supported field discretization combinations in source and destination components. Added capability to apply the weight matrix onto the source solution field vectors, and native visualization plugins in VisIt for MOAB, simplify the verification of conservation and monotonicity for complex remapping workflows. This workflow allows users to validate the underlying assumptions for remapping solution fields across unstructured grids, and can be executed in both a serial and parallel setting.

4 Results

Evaluating the performance of the in-memory, MOAB-TempestRemap (MBTR) remapping infrastructure requires recursive profiling and optimization to ensure scalability for large-scale simulations. In order to showcase the advantage of using the mesh-aware MOAB datastructure as the MCT coupler replacement, we need to understand the per task performance of the regridder in addition to the parallel point locator scalability, and overall time for remapping weight computation. Note that except for the weight application for each solution field from a source grid to a target grid, the in-memory copy of the component meshes, migration to coupler PEs, computation of intersection elements and remapping weights are done only once during the setup phase in E3SM, per coupled component model pair.

4.1 Serial Performance

We compare the total cost for computing the supermesh and the remapping weights for several source and target grid combinations through three different methods to determine the serial computational complexity.

1. ESMF: Kd-tree based regridder and weight generation for first/second order FV→FV conservative remapping
2. TempestRemap: Kd-tree based supermesh generation and conservative, monotonic, high-order remap operator for FV→FV, SE→FV, SE→SE projection
3. MBTempest: Advancing front intersection with MOAB and conservative weight generation with TempestRemap interfaces

Fig. 9 shows the serial performance of the remappers for computing the conservative interpolator from Cubed-Sphere grids to polygonal MPAS grids of different resolutions for a FV→FV field transfer. This total time includes the computation of

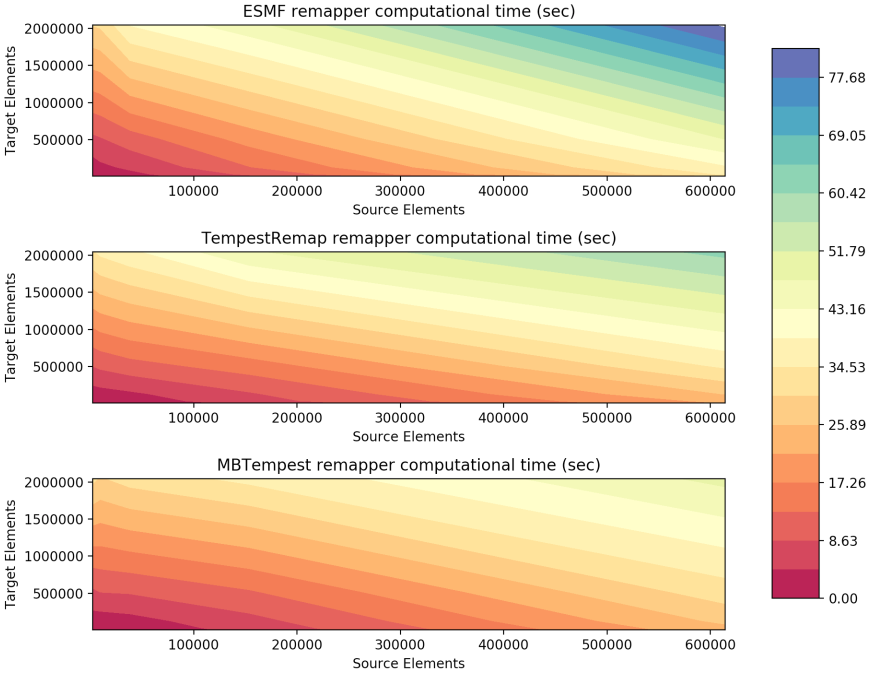


Figure 9. Comparison of serial regridding computation (supermesh and projection weight generation) between ESMF, TempestRemap, and MBTempest

intersection mesh or supermesh, in addition to the remapping weights with field conservation specifications. These serial runs were executed on a machine with 8x Intel Xeon(R) CPU E7-4820 @ 2.00GHz (total of 64 cores) and 1.47 TB of RAM. As the source grid resolution increases, the advancing front intersection with linear complexity outperforms the Kd-tree intersection algorithms used by TempestRemap and ESMF. The time spent in the remapping task, including the overlap mesh generation, provides an overall metric on the single task performance when memory bandwidth or communication concerns do not dominate in a parallel run. In this comparison with three remapping software libraries, the total computational time in the fine resolution limit as $\frac{nele(source)}{nele(target)} \approx 1$ consistently increases (going diagonally from left to right in Fig. 9). We note that the serial version of TempestRemap is comparable to ESMF and can even provide better timings on the highly refined cases, while the MBTempest remapper consistently outperforms both the tools, with a 2x speedup on average. The relatively better performance in MBTempest is accomplished through the linear complexity advancing front algorithm, which further offers avenues to incorporate ~~task~~ finer grain task or thread level parallelism to accelerate the on-node performance on multicore and GPGPU ~~hybrid~~ architectures.

4.2 Scalability of the MOAB Kd-tree Point Locator

In addition to being able to compute the supermesh between Ω_S and Ω_T, MOAB also offers datastructures to query source elements containing points that correspond to the target DoFs locations. This operation is critical in evaluating bilinear and biquadratic interpolator approximations for scalar variables when conservative projection is not required by the underlying coupled model. The solution interpolation for the multi-mesh case involves two distinct phases.

1. Setup phase: Use Kd-tree to build the search datastructure to locate points corresponding to vertices in the target mesh on the source mesh

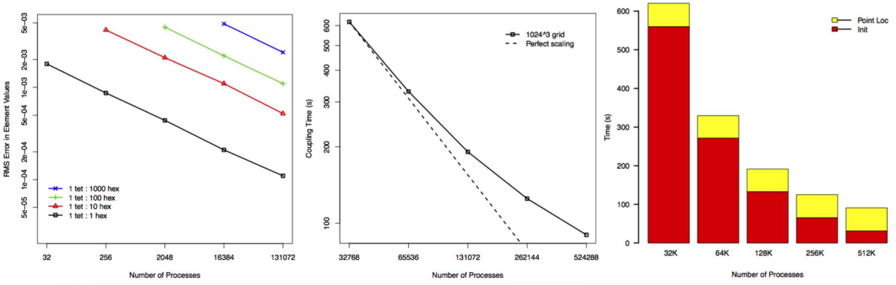
2. Run phase: Use the elements containing the located points to compute consistent interpolation onto target mesh vertices

Studies were performed on the BlueGene-Q machine (Mira) at ANL to evaluate the strong and weak scalability of the parallel Kd-tree point search implementation in MOAB. The scalability results were generated with the CIAN2 coupling mini-app ~~Morozov and Peterka (2016)~~ (Morozov and Peterka, 2016) , which links to MOAB to handle traversal of the unstructured grids and transfer of solution fields between the grids. For this case, a series of hexahedral and tetrahedral meshes were used to interpolate an analytical solution. By changing the basis interpolation order, and mesh resolutions, the convergence of the interpolator was verified to provide theoretical accuracy orders of convergence in the asymptotic fine limit.

The performance tests were executed on the IBM BlueGene/Q Mira at 16 MPI ranks per node, with 2GB RAM per MPI rank, at up to 500K MPI processes. The strong scaling results and error convergence were computed with a grid size of 1024^3. The solution interpolation on varying mesh resolutions were performed by projecting an analytical solution from a Tetrahedral→Hexahedral→Tetrahedral grid, with total number of points/rank varied between [2K, 32K] in the study.

Fig. 10 shows a strong scaling efficiency of around 50% is achieved on a maximum of 512K cores (66% of Mira). We note that the computational complexity of the Kd-tree data structure scales as $O(nlog(n))$ asymptotically, and the point location phase during initial search setup dominates the total cost on higher core counts. This is evident in the timing breakdown for each phase shown in Fig. 10-(c). Since the point location is performed only once during simulation startup, while the interpolation is performed multiple times per timestep during the run, we expect the total cost of the projection for scalar variables to be amortized over transient climate simulations with fixed grids. Further investigations with optimal BVH-tree ~~Larsen et al. (1999)~~ (Larsen et al., 1999) or R-tree implementations for these interpolation cases could help reduce the overall cost.

The full 3-D point location and interpolation operations provided by MOAB are comparable to the implementation in Common Remapping component used in the C-Coupler ~~Liu et al. (2013)~~ (Liu et al., 2013) and provide relatively much stronger scalability on larger core counts ~~Liu et al. (2014)~~ (Liu et al., 2014) for the remapping operation. Such higher-order interpolators



(a) Tet→Hex RMS Error

(b) Strong Scaling

(c) Strong Scaling by Component

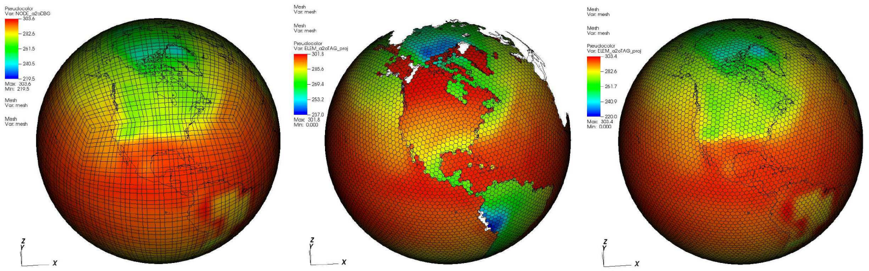
Figure 10. MOAB 3-d Kd-tree Point Location: Strong scaling on Mira (BG/Q)

for multicomponent physics variables can provide better performance in atmospheric chemistry calculations. Currently, only the NC bilinear or biquadratic interpolation of scalar fields with subset normalization ~~Tautges and Caceres (2009)~~ (Tautges and Caceres, 20 supported directly in MOAB (via Kd-tree point location and interpolation), and advancing front intersection algorithm does not make use of these data-structures. In contrast, TempestRemap and ESMF use a Kd-tree search to not only compute the location of points, but also to evaluate the supermesh $\Omega_S \bigcup \Omega_T$, and hence the computational complexity for the intersection mesh determination scales as $O(nlog(n))$, in contrast to the linear complexity ($O(n)$) of the advancing front intersection algorithm implemented in MOAB.

4.3 The Parallel MBTR Remapping Algorithm

The MBTR online weight generation workflow within E3SM was employed to verify and test the projection of real simulation data generated during the coupled atmosphere-ocean model runs. A choice was made to use the model-computed temperature on the lowest level of the atmosphere, since the heat fluxes that nonlinearly couples the atmosphere and ocean models are directly proportional to this interface temperature field. By convention, the fluxes are computed on the ocean mesh, and hence the atmosphere temperature must be interpolated onto MPAS polygonal mesh. We use this scenario as a test case for demonstrating the strong scalability results in this section.

The ~~NE11 (~~atmosphere run with approximately 4 degree grid size ~~) atmosphere run~~ and 11 elements per edge on a cubed-sphere (NE11) in E3SM, and the projection of its lowest level temperature onto two different MPAS meshes (with approximate grid size of 240km) are shown in Fig. 11. The conservative projection from SE→FV on a mesh with holes (Fig. 11-(b)) and without holes (Fig. 11-(c)) corresponding to land regions, is presented here to show the difference in the remapped solutions.



(a) Original SE field on CS atmosphere mesh

(b) Remapped field on MPAS mesh with holes

(c) Remapped field on MPAS mesh without holes

Figure 11. Projection of the NE11 SE bottom atmospheric temperature field onto the MPAS ocean grid

4.3.1 Scaling Comparison of Conservative Remappers (FV→FV)

The strong scaling studies for computation of remapping weights to project a FV solution field between CS grids of varying resolutions was performed on the Blues large-scale cluster (with 16 Sandy Bridge Xeon E5-2670 2.6GHz cores and 32 GB RAM per node) at ANL, and the Cori supercomputer at NERSC (with 64 Haswell Xeon E5-2698v3 2.3GHz cores and 128 GB RAM per node). Fig. 12 shows that the MBTR workflow consistently outperforms ESMF on both the machines as the number of processes used by the coupler is increased. The timings shown here represent the total remapping time i.e., cumulative computational time for generating the super mesh and the (conservative) remapping weights.

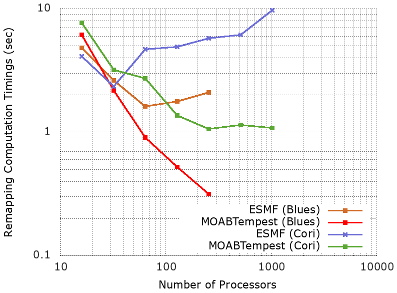


Figure 12. CS (E=614400 quads) → CS (E=153600 quads) remapping (-m conserve) on LCRC/ALCF and NERSC machines

The relatively better scaling for MOAB on the Blues cluster is due to faster hardware and memory bandwidth compared to the Cori machine. The strong scaling efficiency approaches a plateau on Cori Haswell nodes as communication costs for the coverage mesh computation start dominating the overall remapping processes, especially in the limit of $\frac{nele}{process} \rightarrow 1$ at large node counts.

4.3.2 Strong Scalability of Spectral Projection (SE→FV)

To further evaluate the characteristics of in-memory remapping computation, along with cost of application of the weights during a transient simulation, a series of further studies were executed on the NERSC Cori system to determine the spectral projection of a real dataset between atmosphere and ocean components in E3SM. The source mesh contains 4th order spectral element temperature data defined on Gauss-Lobatto quadrature nodes (cGLL discretization) of the CS mesh, and the projection is performed on a MPAS polygonal mesh with holes (FV discretization). A direct comparison to ESMF was unfeasible in this study since the traditional workflow requires the computation of a dual mesh transformation of the spectral grid. Hence, only timings for MBTR workflow is shown here.

Two specific cases were considered for this SE→FV strong scaling study with conservation and monotonicity constraints.

1. **Case A (NE30):** 1-degree CS (30 edges per side) SE mesh (nele=5400 quads) with $p = 4$ to MPAS mesh (nele=235160 polygons)

2. **Case B (NE120):** 0.25-degree CS (120 edges per side) SE mesh (nele=86400 quads) with $p = 4$ to MPAS mesh (nele=3693225 polygons)

The performance tests for each of these cases were launched with three different process execution layouts for the atmosphere, ocean components and the coupler.

(a) Fully colocated PE layout: $N_{atm} = N_x$ and $N_{ocn} = N_x$

(b) Disjoint-ATM model PE layout: $N_{atm} = N_x/2$ and $N_{ocn} = N_x$

(c) Disjoint-OCN model PE layout: $N_{atm} = N_x$ and $N_{ocn} = N_x/2$

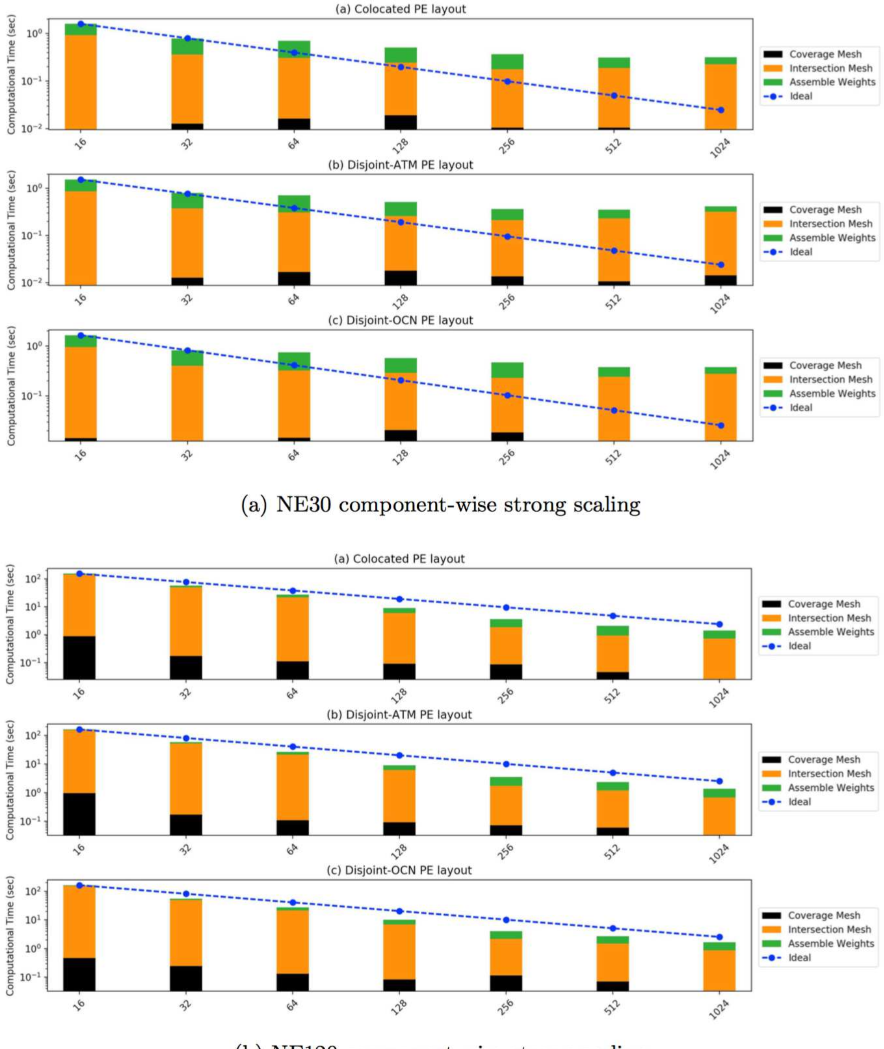
A breakdown of computational time for key tasks on Cori with up to 1024 processes for both the cases is tabulated in Table 1 on a fully colocated decomposition i.e., $N_{ocn} = N_{atm} = N_x$. It is clear that the computation of parallel intersection mesh strong scales well for these production cases, especially for larger mesh resolutions (Case B). For the smaller source and target mesh resolution (Case A), we notice that the intersection time hits a lower bound that is dominated by the computation of the coverage mesh to enclose the target mesh in each task. It is important to stress that this one time setup call to compute remap operator, per component pair, is relatively much cheaper compared to individual component and solver initializations and get amortized over longer transient simulations.

It is also worth noting that as the I/O bandwidth in emerging architectures are not scaling in line with the compute throughput, such an online workflow can generally be faster than parallel I/O for reading the weights from file at scale. The MBTR

Table 1. Strong scaling on Cori for SE→FV projection with two different resolutions

Number of processors	**Case A**		**Case B**	
	Intersection (sec)	**Compute Weights (sec)**	**Intersection (sec)**	**Compute Weights (sec)**
16	0.936846	0.64983	145.623	9.732
32	0.449022	0.429028	53.1244	5.78093
64	0.377767	0.373476	22.7167	4.92151
128	0.255154	0.270574	6.70485	2.79397
256	0.180136	0.18272	2.26435	1.71835
512	0.162388	0.104737	1.25471	0.928622
1024	0.203354	0.0932475	0.680122	0.618943

implementation is also flexible to allow loading the weights from file directly in order to preserve the existing coupler process with MCT. In comparison to the computation of the intersection mesh, the time to assemble the remapping weight operator in parallel is generally smaller. Even though both of these operations are performed only once during the setup phase of the E3SM simulation, the weight operator computation involves several validation checks that utilize collective MPI operations, which do destroy the embarrassingly parallel nature of the calculation, once appropriate coverage mesh is determined in each task.



(a) NE30 component-wise strong scaling

(b) NE120 component-wise strong scaling

Figure 13. Strong scaling study for the NE30 and NE120 cases for spectral projection with Zoltan repartitioner on Cori

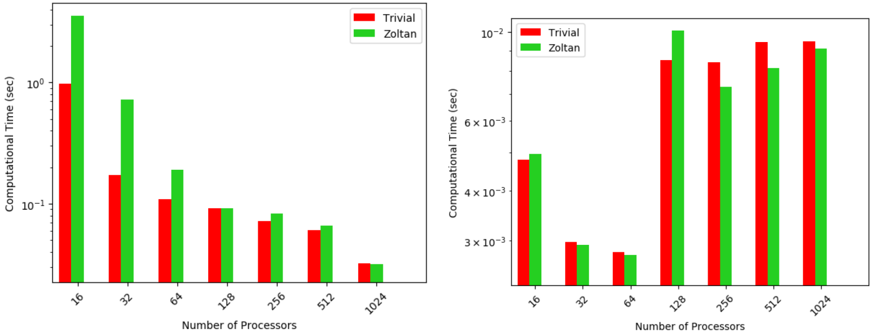
The component-wise breakdown for the advancing front intersection mesh, the parallel communication graph for sending and receiving data between component and coupler, and finally, the remapping weight generation for the SE→FV setup for NE30 and NE120 cases are shown in Fig. 13. The cumulative time for this remapping process is shown to scale linearly for NE120 case, even if the parallel efficiency decreases significantly in the NE30 case, as expected based on the results in Table 1. Note that the MBTR workflow provides a unique capability to consistently and accurately compute SE→FV projection weights in parallel, without any need for an external pre-processing step to compute the dual mesh (as required by ESMF) or running the entire remapping process in serial (TempestRemap). Also note that Fig. 13 confirms that the overall scaling of the remapping algorithm is nearly independent of the PE layout for the simulation.

~~Strong scaling study for the NE30 and NE120 cases for spectral projection with Zoltan repartitioner~~

4.4 Effect of partitioning strategy

In order to determine the effect of partitioning strategies described in Fig. 5, the NE120 case with the trivial decomposition and Zoltan geometric partitioner (RCB) were tested in parallel. Fig. 14 compares the two strategies for optimizing the mesh migration from the component to coupler. These strategies play a critical role in task mapping and data locality for the source coverage mesh computation, in addition to determining the communication graph complexity between the components and the coupler. This comparison highlights that the coverage mesh cost reduces uniformly at scale, while the trivial partitioning scheme behaves better on lower core counts ~~.~~as shown in Fig. 14-(a). The communication of field data between the atmosphere component and the coupler resulting from the partitioning strategy is a critical operation during the transient simulation, and generally stays within network latency limits in Cori, as the message size reduces. Eventhough the communication kernel does not show good ideal scaling on increasing node counts, the relative cost of the operation is insignificant in comparison to total time spent in individual component solvers. Note that production climate model solvers require multiple data fields to be remapped at every rendezvous timestep, and hence the size of the packed messages may be larger for such simulations (volume should remain similar to Fig. 14-(b)).

We also note that there is a factor of 3 increase in the communication time to send and receive data, which occurs after the 64 process count on Cori in Fig. 14-(b). This is an artifact of the additional communication latency due to the transition from an intra-node (each Haswell node in Cori accommodates 64 processes) to inter-node nearest neighbor data transfer when using multiple nodes.



(a) Source coverage mesh computation

(b) Send/Receive field data between N_x and N_{atm}

Figure 14. Scaling of the communication kernels driven with the parallel graph computed with a trivial redistribution and the Zoltan geometric (RCB) repartitioner for the NE120 case with $N_{ocn} = N_x$ and $N_{atm} = N_x/2$ on Cori

4.5 Note on Application of Weights

Generally, operations involving Sparse Matrix-Vector (SpMV) products are memory bandwidth limited ~~Bell and Garland (2009)~~ (Bell and G and occur during the application of remapping weights operator on to the source solution field vector, in order to compute the field projection onto the target grid. In addition to the communication of field data shown in Fig. 14-(b), the cost of remapping weight application in parallel (presented in Fig. 15) determines the total cost of the remapping operation during runtime. Except for the case of cGLL target discretizations, the parallel SpMV operation during the weight application do not involve any global collective reductions. In the current E3SM and OASIS3-MCT workflow, these operations are handled by the MCT library. In high resolution simulations of E3SM, the total time for the remapping operation in MCT is primarily dominated by the communication costs based on the communication graph, similar to the MBTR workflow. However, a direct comparison between these two workflows is not yet possible, but we expect the aggregated communication strategies in the crystal router algorithm ~~Fox et al. (1989)~~ (Fox et al., 1989) in MOAB, to provide relatively better performance at scale.

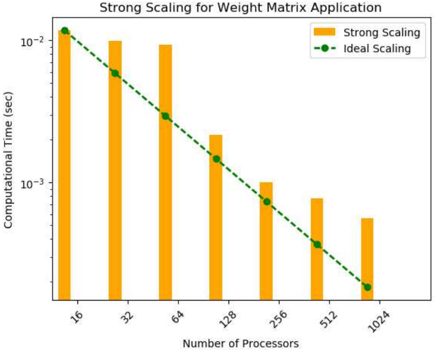


Figure 15. SE→FV remapping weight operator application on Cori

5 Conclusion

Understanding and controlling primary sources of errors in a coupled system dynamically, will be key to achieving predictable and verifiable climate simulations on emerging architectures. Traditionally, the computational workflow for coupled climate simulations has involved two distinct steps, with an offline pre-processing phase using remapping tools to generate solution field projection weights (ESMF, TempestRemap, SCRIP), which is then consumed by the coupler to transfer field data between the component grids.

The offline steps include generating grid description files and running the offline tools with the problem-specific options. Additionally many of state-of-science tools such as ESMF and SCRIP require additional steps to specially handle interpolators from SE grids. Such workflows create bottlenecks that do not scale, and can inhibit scientific research productivity. When experimenting with refined grids, a goal for E3SM, this tool chain has to excercised repeatedly. Additionally, when component meshes are dynamically modified, either through mesh adaptivity or dynamical mesh movement to track moving boundaries, the underlying remapping weights must be recomputed on the fly.

To overcome some of these limitations, we have presented scalable algorithms and software interfaces to create a direct component coupling with online regridding and weight generation tools. The remapping algorithms utilize the numerics exposed by TempestRemap, and leverage the parallel mesh handling infrastructure in MOAB to create a scalable in-memory remapping infrastructure that can be integrated with existing coupled climate solvers. Such a methodology invalidates the need for dual grids, preserves higher-order spectral accuracy, and locally conserves the field data, in addition to monotonicity constraints, when transferring solutions between grids with non-matching resolutions.

The serial and parallel performance of the MOAB advancing front intersection algorithm with linear complexity ($O(n)$) was demonstrated for a variety of source and target mesh resolution combinations, and compared with the current state-of-science regridding tools such as ESMF (serial/parallel) and TempestRemap (serial) that have a $O(nlog(n))$ complexity using the Kd-tree datastructure. The MOAB-TempestRemap (MBTR) software infrastructure yields a balance of both the scalable performance on emerging architectures without sacrificing discretization accuracy for component field interpolators. There are also several optimizations in the MBTR algorithms that can be implemented to improve finer-grained parallelism on ~~hybrid architectures~~heterogeneous architectures, and to minimize data movement with better partitioning in combination with load rebalancing strategies. Such a software infrastructure provides a foundation to build a new coupler to replace the current offline-online, hub-and-spoke MCT-based coupler in E3SM, and offer extensions to enable a fully distributed coupling paradigm (without the need for a centralized coupler) to minimize computational bottlenecks in a task-based workflow.

Code availability. Information on the availability of source code for the algorithmic infrastructure and models featured in this paper is tabulated below.

Short name	Code availability
E3SM	E3SM Project (2018) is under active development funded by the US Department of Energy. E3SM version 1.1 has been publicly released under an open-source 3-clause BSD license in August 2018, and available at GitHub.
MOAB	MOAB Tautges et al. (2004) is an open-source library under the umbrella of the SIGMA toolkit (2014) Mahadevan et al. (2015) , and is publicly available under the Lesser GNU Public License (v3) on BitBucket. v5.1.0 was released on Jan 07, 2019 and available here. DOI: 10.5281/zenodo.2584863.
TempestRemap	The TempestRemap Ullrich and Taylor (2015); Ullrich et al. (2016) source code is available under a BSD open-source license and hosted in GitHub. v2.0.2 was released on Dec 19, 2018 and available here.

Video supplement. The video supplements for the serial and parallel advancing front mesh intersection algorithm to compute the supermesh ($\mathbf{\Omega_S} \bigcup \mathbf{\Omega_T}$) of a source ($\mathbf{\Omega_S}$)and target ($\mathbf{\Omega_T}$) grid is demonstrated.

Short name	Video description and availability
Serial advancing front mesh intersection	Intersection between CS and MPAS grids on a single task is illustrated. DOI:10.6084/m9.figshare.7294901
Parallel advancing front mesh intersection	Simultaneous parallel Intersection between CS and MPAS grids on two different tasks are illustrated side by side. DOI:10.6084/m9.figshare.7294919

Author contributions. VM and RJ wrote the paper (with comments from IG and JS). VM and IG designed and implemented the MOAB integration with TempestRemap library, along with exposing the necessary infrastructure for online remapping through iMOAB interfaces. IG and JS configured the MOAB-TempestRemap remapper within E3SM, and verified weight generation to transfer solution fields between atmosphere and ocean component models. VM conducted numerical verification studies and executed both the serial and parallel scalability studies on Blues and Cori LCF machines to quantify performance characteristics of the remapping algorithms. The broader project idea was conceived by Andy Salinger (SNL), RJ, VM, and IG.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. Support for this work was provided by the Climate Model Development and Validation - Software Modernization project, and partially by the SciDAC Coupling Approaches for Next Generation Architectures (CANGA) project, which are funded by the US Department of Energy (DOE), Office of Science Biological and Environmental Research Program. CANGA is also funded by the DOE Office of Advanced Scientific Computing Research. This research used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357, and resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. We gratefully acknowledge the computing resources provided on Blues, a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory. We would also like to thank Dr. Paul Ullrich at University of California, Davis for several helpful discussions regarding remapping schemes and, in particular, implementations in TempestRemap. We would also like to thank the anonymous reviewers for their indepth comments that helped clarify and improve the manuscript.

References

Aguerre, H. J., Damián, S. M., Gimenez, J. M., and Nigro, N. M.: Conservative handling of arbitrary non-conformal interfaces using an efficient supermesh, Journal of Computational Physics, 335, 21–49, 2017.

Beljaars, A., Dutra, E., Balsamo, G., and Lemarié, F.: On the numerical stability of surface-atmosphere coupling in weather and climate models, Geoscientific Model Development Discussions, 10, 977–989, 2017.

Bell, N. and Garland, M.: Implementing sparse matrix-vector multiplication on throughput-oriented processors, in: Proceedings of the conference on high performance computing networking, storage and analysis, p. 18, ACM, 2009.

Berger, M. J.: On conservation at grid interfaces, SIAM journal on numerical analysis, 24, 967–984, 1987.

Blanco, J. L. and Rai, P. K.: nanoflann: a C++ header-only fork of FLANN, a library for Nearest Neighbor (NN) wih KD-trees, https://github.com/jlblancoc/nanoflann, 2014.

Březina, J. and Exner, P.: Fast algorithms for intersection of non-matching grids using Plücker coordinates, Computers & Mathematics with Applications, 74, 174 – 187, https://doi.org/https://doi.org/10.1016/j.camwa.2017.01.028, http://www.sciencedirect.com/science/article/pii/S0898122117300792, 5th European Seminar on Computing ESCO 2016, 2017.

Certik, O., Ferenbaugh, C., Garimella, R., Herring, A., Jean, B., Malone, C., and Sewell, C.: A Flexible Conservative Remapping Framework for Exascale Computing, https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-17-21749, sIAM Conference on Computational Science & Engineering Feb 2017, 2017.

Collins, N., Theurich, G., Deluca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang, W., Hill, C., and Da Silva, A.: Design and implementation of components in the Earth System Modeling Framework, The International Journal of High Performance Computing Applications, 19, 341–350, 2005.

Craig, A., Valcke, S., and Coquart, L.: Development and performance of a new version of the OASIS coupler, OASIS3-MCT_3.0, Geoscientific Model Development, 10, 3297–3308, https://doi.org/10.5194/gmd-10-3297-2017, https://www.geosci-model-dev.net/10/3297/2017/, 2017.

Craig, A. P., Jacob, R., Kauffman, B., Bettge, T., Larson, J., Ong, E., Ding, C., and He, Y.: CPL6: The New Extensible, High Performance Parallel Coupler for the Community Climate System Model, The International Journal of High Performance Computing Applications, 19, 309–327, https://doi.org/10.1177/1094342005056117, https://doi.org/10.1177/1094342005056117, 2005.

Craig, A. P., Vertenstein, M., and Jacob, R.: A new flexible coupler for earth system modeling developed for CCSM4 and CESM1, The International Journal of High Performance Computing Applications, 26, 31–42, https://doi.org/10.1177/1094342011428141, https://doi.org/10.1177/1094342011428141, 2012.

de Boer, A., van Zuijlen, A., and Bijl, H.: Comparison of conservative and consistent approaches for the coupling of non-matching meshes, Computer Methods in Applied Mechanics and Engineering, 197, 4284–4297, https://doi.org/10.1016/j.cma.2008.05.001, http://dx.doi.org/10.1016/j.cma.2008.05.001, 2008.

Dennis, J. M., Edwards, J., Evans, K. J., Guba, O., Lauritzen, P. H., Mirin, A. A., St-Cyr, A., Taylor, M. A., and Worley, P. H.: CAM-SE: A scalable spectral element dynamical core for the Community Atmosphere Model, The International Journal of High Performance Computing Applications, 26, 74–89, 2012.

Devine, K., Boman, E., Heaphy, R., Hendrickson, B., and Vaughan, C.: Zoltan Data Management Services for Parallel Dynamic Applications, Computing in Science and Engineering, 4, 90–97, 2002.

Dunlap, R., Rugaber, S., and Mark, L.: A feature model of coupling technologies for Earth System Models, Computers & geosciences, 53, 13–20, 2013.

E3SM Project: Energy Exascale Earth System Model (E3SM), [Computer Software] https://dx.doi.org/10.11578/E3SM/dc.20180418.36, https://doi.org/10.11578/E3SM/dc.20180418.36, https://dx.doi.org/10.11578/E3SM/dc.20180418.36, 2018.

Farrell, P. and Maddison, J.: Conservative interpolation between volume meshes by local Galerkin projection, Computer Methods in Applied Mechanics and Engineering, 200, 89 – 100, https://doi.org/http://dx.doi.org/10.1016/j.cma.2010.07.015, http://www.sciencedirect.com/science/article/pii/S0045782510002276, 2011.

Fleishman, S., Cohen-Or, D., and Silva, C. T.: Robust moving least-squares fitting with sharp features, ACM transactions on graphics (TOG), 24, 544–552, 2005.

Flyer, N. and Wright, G. B.: Transport schemes on a sphere using radial basis functions, Journal of Computational Physics, 226, 1059–1084, 2007.

Fornberg, B. and Piret, C.: On choosing a radial basis function and a shape parameter when solving a convective PDE on a sphere, Journal of Computational Physics, 227, 2758–2780, 2008.

Fox, G., Johnson, M., Lyzenga, G., Otto, S., Salmon, J., Walker, D., and White, R. L.: Solving Problems On Concurrent Processors Vol. 1: General Techniques and Regular Problems, Computers in Physics, 3, 83–84, 1989.

Gander, M. J. and Japhet, C.: An algorithm for non-matching grid projections with linear complexity, in: Domain Decomposition Methods in Science and Engineering XVIII, pp. 185–192, Springer, 2009.

Gander, M. J. and Japhet, C.: Algorithm 932: PANG: software for nonmatching grid projections in 2D and 3D with linear complexity, ACM Transactions on Mathematical Software (TOMS), 40, 6, 2013.

Gottlieb, D. and Shu, C.-W.: On the Gibbs phenomenon and its resolution, SIAM review, 39, 644–668, 1997.

Grandy, J.: Conservative remapping and region overlays by intersecting arbitrary polyhedra, J. Comput. Phys., 148, 433–466, 1999.

Guennebaud, G., Jacob, B., et al.: Eigen v3, http://eigen.tuxfamily.org, 2010.

Hanke, M., Redler, R., Holfeld, T., and Yastremsky, M.: YAC 1.2.0: new aspects for coupling software in Earth system modelling, Geoscientific Model Development, 9, 2755–2769, https://doi.org/10.5194/gmd-9-2755-2016, https://www.geosci-model-dev.net/9/2755/2016/, 2016.

Herring, A. M., Certik, O., Ferenbaugh, C. R., Garimella, R. V., Jean, B. A., Malone, C. M., and Sewell, C. M.: (U) Introduction to Portage, Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-17-20831, 2017.

Hill, C., DeLuca, C., Balaji, Suarez, M., and Silva, A. d.: The architecture of the earth system modeling framework, Computing in Science & Engineering, 6, 18–28, 2004.

Hunt, W., Mark, W. R., and Stoll, G.: Fast kd-tree construction with an adaptive error-bounded heuristic, in: Interactive Ray Tracing 2006, IEEE Symposium on, pp. 81–88, IEEE, 2006.

Hurrell, J. W., Holland, M. M., Gent, P. R., Ghan, S., Kay, J. E., Kushner, P. J., Lamarque, J.-F., Large, W. G., Lawrence, D., Lindsay, K., et al.: The community earth system model: a framework for collaborative research, Bulletin of the American Meteorological Society, 94, 1339–1360, 2013.

Ize, T., Wald, I., and Parker, S. G.: Asynchronous BVH construction for ray tracing dynamic scenes on parallel multi-core architectures, in: Proceedings of the 7th Eurographics conference on Parallel Graphics and Visualization, pp. 101–108, Eurographics Association, 2007.

Jacob, R., Larson, J., and Ong, E.: M× N communication and parallel interpolation in Community Climate System Model Version 3 using the model coupling toolkit, The International Journal of High Performance Computing Applications, 19, 293–307, 2005a.

Jacob, R., Larson, J., and Ong, E.: M× N communication and parallel interpolation in Community Climate System Model Version 3 using the model coupling toolkit, The International Journal of High Performance Computing Applications, 19, 293–307, 2005b.

Jiao, X. and Heath, M. T.: Common-refinement-based data transfer between non-matching meshes in multiphysics simulations, International Journal for Numerical Methods in Engineering, 61, 2402–2427, 2004.

Jones, P. W.: First-and second-order conservative remapping schemes for grids in spherical coordinates, Monthly Weather Review, 127, 2204–2210, 1999.

Karypis, G., Schloegel, K., and Kumar, V.: Parmetis: Parallel graph partitioning and sparse matrix ordering library, Version 1.0, Dept. of Computer Science, University of Minnesota, p. 22, 1997.

Larsen, E., Gottschalk, S., Lin, M. C., and Manocha, D.: Fast proximity queries with swept sphere volumes, Tech. rep., Technical Report TR99-018, Department of Computer Science, University of North Carolina, 1999.

Larson, J. W., Jacob, R. L., Foster, I., and Guo, J.: The model coupling toolkit, in: International Conference on Computational Science, pp. 185–194, Springer, 2001.

Lauritzen, P. H., Nair, R. D., and Ullrich, P. A.: A conservative semi-Lagrangian multi-tracer transport scheme (CSLAM) on the cubed-sphere grid, Journal of Computational Physics, 229, 1401 – 1424, https://doi.org/http://dx.doi.org/10.1016/j.jcp.2009.10.036, http://www.sciencedirect.com/science/article/pii/S002199910900597X, 2010.

Li, L., Lin, P., Yu, Y., Wang, B., Zhou, T., Liu, L., Liu, J., Bao, Q., Xu, S., Huang, W., Xia, K., Pu, Y., Dong, L., Shen, S., Liu, Y., Hu, N., Liu, M., Sun, W., Shi, X., Zheng, W., Wu, B., Song, M., Liu, H., Zhang, X., Wu, G., Xue, W., Huang, X., Yang, G., Song, Z., and Qiao, F.: The flexible global ocean-atmosphere-land system model, Grid-point Version 2: FGOALS-g2, Advances in Atmospheric Sciences, 30, 543–560, https://doi.org/10.1007/s00376-012-2140-6, https://doi.org/10.1007/s00376-012-2140-6, 2013.

Liu, L., Yang, G., and Wang, B.: CoR: a multi-dimensional common remapping software for Earth System Models, in: The Second Workshop on Coupling Technologies for Earth System Models (CW2013), available at: https://wiki.cc.gatech.edu/CW2013/index.php/Program (last access: 8 May 2014), 2013.

Liu, L., Yang, G., Wang, B., Zhang, C., Li, R., Zhang, Z., Ji, Y., and Wang, L.: C-Coupler1: a Chinese community coupler for Earth system modeling, Geoscientific Model Development, 7, 2281–2302, https://doi.org/10.5194/gmd-7-2281-2014, https://www.geosci-model-dev.net/7/2281/2014/, 2014.

Liu, L., Zhang, C., Li, R., Wang, B., and Yang, G.: C-Coupler2: a flexible and user-friendly community coupler for model coupling and nesting, Geoscientific Model Development, 11, 3557–3586, https://doi.org/10.5194/gmd-11-3557-2018, https://www.geosci-model-dev.net/11/3557/2018/, 2018.

Löhner, R.: Recent advances in parallel advancing front grid generation, Archives of Computational Methods in Engineering, 21, 127–140, 2014.

Löhner, R. and Parikh, P.: Generation of three-dimensional unstructured grids by the advancing-front method, International Journal for Numerical Methods in Fluids, 8, 1135–1149, 1988.

Mahadevan, V., Grindeanu, I. R., Ray, N., Jain, R., and Wu, D.: SIGMA Release v1.2 - Capabilities, Enhancements and Fixes, https://doi.org/10.2172/1224985, 2015.

Mahadevan, V., Grindeanu, I., Jacob, R., and Sarich, J.: MOAB: Serial Advancing Front Intersection Computation, https://doi.org/10.6084/m9.figshare.7294901.v1, https://figshare.com/articles/MOAB_Serial_Advancing_Front_Intersection_Computation/7294901, 2018a.

Mahadevan, V., Grindeanu, I., Jacob, R., and Sarich, J.: MOAB: Parallel Advancing Front Mesh Intersection Algorithm, https://doi.org/10.6084/m9.figshare.7294919.v2, https://figshare.com/articles/MOAB_Parallel_Advancing_Front_Mesh_Intersection_Algorithm/72949 2018b.

Morozov, D. and Peterka, T.: Block-Parallel Data Analysis with DIY2, 2016.

OASIS3-MCT v4.0: OASIS3-MCT 4.0 official release, https://portal.enes.org/oasis/news/oasis3-mct_4-0-official-release, [Online; accessed 10-October-2018], 2018.

Petersen, M. R., Jacobsen, D. W., Ringler, T. D., Hecht, M. W., and Maltrud, M. E.: Evaluation of the arbitrary Lagrangian–Eulerian vertical coordinate method in the MPAS-Ocean model, Ocean Modelling, 86, 93–113, 2015.

Plimpton, S. J., Hendrickson, B., and Stewart, J. R.: A parallel rendezvous algorithm for interpolation between multiple grids, Journal of Parallel and Distributed Computing, 64, 266–276, 2004.

Powell, D. and Abel, T.: An exact general remeshing scheme applied to physically conservative voxelization, Journal of Computational Physics, 297, 340 – 356, https://doi.org/https://doi.org/10.1016/j.jcp.2015.05.022, http://www.sciencedirect.com/science/article/pii/S0021999115003563, 2015.

Rančić, M.: An efficient, conservative, monotonic remapping for semi-Lagrangian transport algorithms, Monthly Weather Review, 123, 1213–1217, 1995.

Reichler, T. and Kim, J.: How well do coupled models simulate today's climate?, Bulletin of the American Meteorological Society, 89, 303–312, 2008.

Ringler, T., Petersen, M., Higdon, R. L., Jacobsen, D., Jones, P. W., and Maltrud, M.: A multi-resolution approach to global ocean modeling, Ocean Modelling, 69, 211–232, 2013.

Schliephake, M. and Laure, E.: Performance Analysis of Irregular Collective Communication with the Crystal Router Algorithm, in: Solving Software Challenges for Exascale, edited by Markidis, S. and Laure, E., pp. 130–140, Springer International Publishing, Cham, 2015.

Shewchuk, J. R.: Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator, in: Applied computational geometry towards geometric engineering, pp. 203–222, Springer, 1996.

SIGMA toolkit: Scalable Interfaces for Geometry and Mesh based Applications (SIGMA) toolkit, http://sigma.mcs.anl.gov, http://sigma.mcs.anl.gov, 2014.

Slattery, S., Wilson, P., and Pawlowski, R.: The data transfer kit: a geometric rendezvous-based tool for multiphysics data transfer, in: International conference on mathematics & computational methods applied to nuclear science & engineering (M&C 2013), pp. 5–9, 2013.

Slattery, S. R.: Mesh-free data transfer algorithms for partitioned multiphysics problems: Conservation, accuracy, and parallelism, Journal of Computational Physics, 307, 164–188, 2016.

Slingo, J., Bates, K., Nikiforakis, N., Piggott, M., Roberts, M., Shaffrey, L., Stevens, I., Vidale, P. L., and Weller, H.: Developing the next-generation climate system models: challenges and achievements, Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 367, 815–831, 2009.

Tautges, T. J. and Caceres, A.: Scalable parallel solution coupling for multiphysics reactor simulation, Journal of Physics: Conference Series, 180, 2009.

Tautges, T. J., Meyers, R., Merkley, K., Stimpson, C., and Ernst, C.: MOAB: A Mesh-Oriented datABase, SAND2004-1592, Sandia National Laboratories, 2004.

Tautges, T. J., Kraftcheck, J., Bertram, N., Sachdeva, V., and Magerlein, J.: Mesh Interface Resolution and Ghost Exchange in a Parallel Mesh Representation, IEEE, Shanghai, China, 2012.

Taylor, M., Edwards, J., Thomas, S., and Nair, R.: A mass and energy conserving spectral element atmospheric dynamical core on the cubed-sphere grid, in: Journal of Physics: Conference Series, vol. 78, p. 012074, IOP Publishing, 2007.

Thomas, S. J. and Loft, R. D.: The NCAR spectral element climate dynamical core: Semi-implicit Eulerian formulation, Journal of Scientific Computing, 25, 307–322, 2005.

Ullrich, P. A. and Taylor, M. A.: Arbitrary-order conservative and consistent remapping and a theory of linear maps: Part I, Monthly Weather Review, 143, 2419–2440, 2015.

Ullrich, P. A., Lauritzen, P. H., and Jablonowski, C.: Geometrically Exact Conservative Remapping (GECoRe): Regular latitude–longitude and cubed-sphere grids, Monthly Weather Review, 137, 1721–1741, 2009.

Ullrich, P. A., Lauritzen, P. H., and Jablonowski, C.: Some considerations for high-order 'incremental remap'-based transport schemes: edges, reconstructions, and area integration, International Journal for Numerical Methods in Fluids, 71, 1131–1151, 2013.

Ullrich, P. A., Devendran, D., and Johansen, H.: Arbitrary-order conservative and consistent remapping and a theory of linear maps: Part II, Monthly Weather Review, 144, 1529–1549, 2016.

Valcke, S.: The OASIS3 coupler: a European climate modelling community software, Geoscientific Model Development, 6, 373–388, 2013.

VisIt: VisIt User's Guide, Tech. Rep. UCRL-SM-220449, Lawrence Livermore National Laboratory, 2005.

Wan, H., Giorgetta, M. A., Zängl, G., Restelli, M., Majewski, D., Bonaventura, L., Fröhlich, K., Reinert, D., Rípodas, P., Kornblueh, L., and Förstner, J.: The ICON-1.2 hydrostatic atmospheric dynamical core on triangular grids â€" Part 1: Formulation and performance of the baseline version, Geoscientific Model Development, 6, 735–763, https://doi.org/10.5194/gmd-6-735-2013, https://www.geosci-model-dev.net/6/735/2013/, 2013.

Washington, W., Bader, D., and Collins, B, e. a.: Challenges in climate change science and the role of computing at the extreme scale, in: Proc. of the Workshop on Climate Science, 2008.

Zhou, S.: Coupling climate models with the earth system modeling framework and the common component architecture, Concurrency and Computation: Practice and Experience, 18, 203–213, 2006.

Zienkiewicz, O. C. and Zhu, J. Z.: The superconvergent patch recovery and a posteriori error estimates. Part 1: The recovery technique, International Journal for Numerical Methods in Engineering, 33, 1331–1364, 1992.

Government License

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2018-280, 2018.

Improving climate model coupling through a complete mesh representation: a case study with E3SM (v1) and MOAB (v5.x)

Vijay S. Mahadevan[1], Iulian Grindeanu[1], Robert Jacob[1], and Jason Sarich[1]

[1]Argonne National Laboratory, 9700 S. Cass Avenue, Lemont, IL

Correspondence: V. S. Mahadevan (mahadevan@anl.gov)

Abstract.

One of the fundamental factors contributing to the spatiotemporal inaccuracy in climate modeling is the mapping of solution field data between different discretizations and numerical grids used in the coupled component models. The typical climate computational workflow involves evaluation and serialization of the remapping weights during the pre-processing step, which is then consumed by the coupled driver infrastructure during simulation to compute field projections. Tools like Earth System Modeling Framework (ESMF) (Hill et al., 2004) and TempestRemap (Ullrich et al., 2013) offer capability to generate conservative remapping weights, while the Model Coupling Toolkit (MCT) (Larson et al., 2001) that is utilized in many production climate models exposes functionality to make use of the operators to solve the coupled problem. However, such multi-step processes present several hurdles in terms of the scientific workflow, and impedes research productivity. In order to overcome these limitations, we present a fully integrated infrastructure based on the Mesh Oriented datABase (MOAB) (Tautges et al., 2004; Mahadevan et al., 2015) library, which allows for a complete description of the numerical grids, and solution data used in each submodel. Through a scalable advancing front intersection algorithm, the supermesh of the source and target grids are computed, which is then used to assemble the high-order, conservative and monotonicity preserving remapping weights between discretization specifications. The Fortran compatible interfaces in MOAB are utilized to directly link the submodels in the Energy Exascale Earth System Model (E3SM) to enable online remapping strategies in order to simplify the coupled workflow process. We demonstrate the superior computational efficiency of the remapping algorithms in comparison with other state-of-science tools and present strong scaling results on large-scale machines for computing remapping weights between the spectral-element atmosphere and finite-volume discretizations on the polygonal ocean grids.

1 Introduction

Understanding Earth's climate evolution through robust and accurate modeling of the intrinsically complex, coupled ocean-atmosphere-land-ice-biosphere models requires extreme-scale computational power (Washington et al., 2008). In such coupled applications, the different component models may employ unstructured spatial meshes that are specifically generated to resolve problem-dependent solution variations, which introduces several challenges in performing a consistent solution coupling. It is known that operator decomposition and unresolved coupling errors in partitioned atmosphere and ocean model simulations (Beljaars et al., 2017), or physics and dynamics components of an atmosphere, can lead to large approximation errors that cause severe numerical stability issues. In this context, one factor contributing to the spatiotemporal accuracy is the mapping between different discretizations of the sphere used in the components of a coupled climate model. Accurate remapping strategies in such multi-mesh problems are critical to preserve higher order resolution, but are in general computationally expensive given the disparate spatial scales across which conservative projections are calculated. Since the primal solution or auxiliary derived data defined on a donor physics component mesh (source model) needs to be transferred to its coupled dependent physics mesh (target model), robust numerical algorithms are necessary to preserve discretization accuracy during these operations (Grandy, 1999; de Boer et al., 2008), in addition to conservation and monotonicity properties in the field profile.

An important consideration is that in addition to maintaining the overall discretization accuracy of the solution during remapping, global conservation, and sometimes local element-wise conservation for critical quantities (Jiao and Heath, 2004) needs to be imposed during the workflow. Such stringent requirements on key flux fields that couple components along boundary interfaces is necessary in order to mitigate any numerical deviations in coupled climate simulations. Note that these physics meshes are usually never embedded or include trivial linear transformations, which render existence of exact projection or interpolation operators unfeasible, even if the same continuous geometric topology is discretized in the models. Additionally, the unique domain decomposition used for each of the component physics meshes complicates the communication pattern during intra-physics transfer, since aggregation of point location requests need to be handled efficiently in order to reduce overheads during the remapping workflow (Plimpton et al., 2004; Tautges and Caceres, 2009).

Adaptive block-structured cubed-sphere or unstructured refinement of icosahedral/polygonal meshes (Slingo et al., 2009) are often used to resolve the complex fluid dynamics behavior in atmosphere and ocean models efficiently. In such models, conservative, local flux-preserving remapping schemes are critically important (Berger, 1987) to effectively reduce multimesh errors, especially during computation of tracer advection such as water vapor or CO_2 (Lauritzen et al., 2010). This is also an issue in atmosphere models where physics and dynamics are computed on non-embedded grids (Dennis et al., 2012), and the improper spatial coupling between these multi-scale models could introduce numerical artifacts. Hence, the availability of different consistent and accurate remapping schemes under one flexible climate simulation framework is vital to better understand the pros and cons of the adaptive multiresolution choices (Reichler and Kim, 2008).

1.1 Hub-and-Spoke vs Distributed Coupling Workflow

The hub-and-spoke centralized model as shown in Fig. 1 (left) is used in the current Exascale Earth System Model (E3SM) driver, and relies on several tools and libraries that have been developed to simplify the regridding workflow within the climate community. Most of the current tools used in E3SM and the Community Earth System Model (CESM) (Hurrell et al., 2013) are included in a single package called the Common Infrastructure for Modeling the Earth (CIME), which builds on previous couplers used in CESM (Craig et al., 2005, 2012). These modeling tools approach the problem in a two-step computational process:

1. Compute the projection or remapping weights for a solution field from a source component physics to a target component physics as an offline process

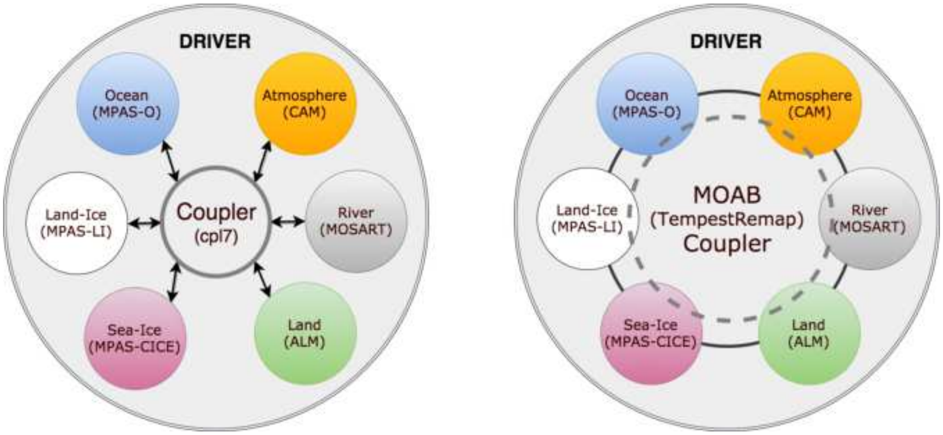


Figure 1. E3SM Coupled Climate Solver: (a) Current model (left), (b) Newer MOAB based coupler (right).

2. During runtime, the CIME coupled solver loads the remapping weights from a file, and handles the partition-aware communication and weight matrix application to project coupled fields between components

The first task in this workflow is currently accomplished through a variety of standard state-of-science tools such as the Earth Science Modeling Framework (ESMF) (Hill et al., 2004), Spherical Coordinate Remapping and Interpolation Package (SCRIP) (Jones, 1999), TempestRemap (Ullrich et al., 2013; Ullrich and Taylor, 2015). The Model Coupling Toolkit (MCT) (Larson et al., 2001; Jacob et al., 2005a) used in the CIME solver provides data structures for the second part of the workflow. Traditionally the first workflow phase is executed decoupled from the simulation driver during a pre-processing step, and hence any updates to the field discretization or the underlying mesh resolution immediately necessitates recomputation of the remapping weight generation workflow with updated inputs. This process flow also prohibits the component solvers from performing any runtime spatial adaptivity, since the remapping weights have to be re-computed dynamically after any changes in grid positions. To overcome such deficiencies, and to accelerate the current coupling workflow, recent efforts have been undertaken to implement a fully integrated remapping weight generation process within E3SM using a scalable infrastructure provided by the topology, decomposition and data-aware Mesh Oriented datABase (MOAB) (Tautges et al., 2004; Mahadevan et al., 2015) and TempestRemap (Ullrich et al., 2013) software libraries as shown in Fig. 1 (right). Note that whether a hub-and-spoke or distributed coupling model is used to drive the simulation, a minimal layer of driver logic is necessary to compute weighted combination of fluxes, validation metrics, and other diagnostic outputs.

The paper is organized as follows. In Section. (2), we present the necessary background and motivations to develop an online remapping workflow implementation in E3SM. Section. (3) covers details on the scalable, mesh and partition aware, conservative remapping algorithmic implementation to improve scientific productivity of the climate scientists, and to simplify the overall computational workflow for complex problem simulations. Then, the performance of these algorithms are first evaluated in serial for various grid combinations, and the parallel scalability of the workflow is demonstrated on large-scale machines in Section. (4).

2 Background

Conservative remapping of nonlinearly coupled solution fields is a critical task to ensure consistency and accuracy in climate and numerical weather prediction simulations (Slingo et al., 2009). While there are various ways to compute a projection of a solution defined on a source grid Ω_S to a target grid Ω_T, the requirements related to global or local conservation in the remapped solution reduces the number of potential algorithms that can be employed for such problems.

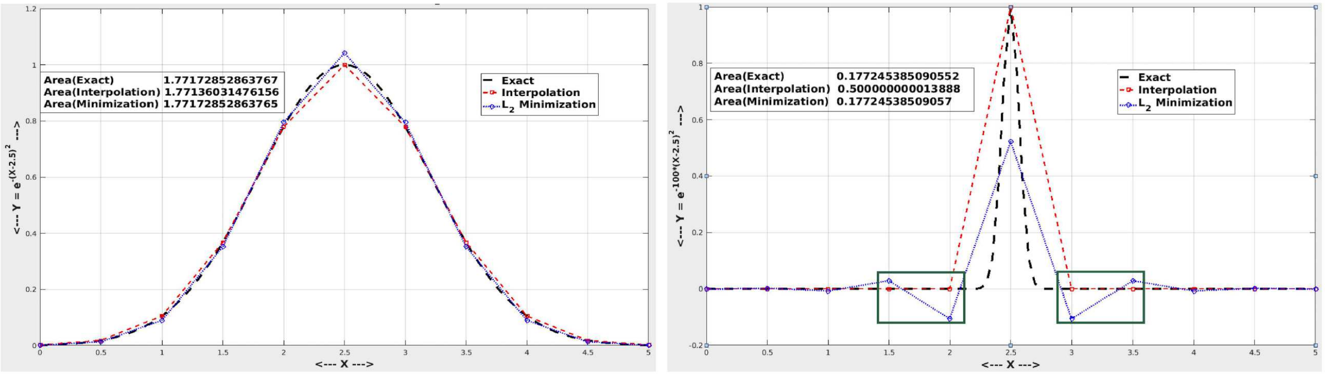
Depending on whether (global or local) conservation is important, and if higher-order, monotone interpolators are required, there are several consistent algorithmic options that can be used (de Boer et al., 2008). All of these different remapping schemes usually have one of these characteristic traits: non-conservative (**NC**), globally-conservative (**GC**) and locally-conservative (**LC**).

1. **NC/GC**: Solution interpolation approximations
   - **NC**: (Approximate or exact) nearest neighbor interpolation
   - **NC/GC**: Radial Basis Function (RBF) (Flyer and Wright, 2007) interpolators and patch-based Least Squares reconstructions (Zienkiewicz and Zhu, 1992; Fleishman et al., 2005)
   - **GC**: Consistent Finite Element (FE) interpolation and area re-normalization
2. **LC/GC**: L_2 or H_1 projection
   - **LC/GC**: Embedded FE/FD/FV meshes in adaptive computations
   - **LC**: Intersection-based field integrators with consistent higher-order discretization (Jones, 1999)
   - **LC**: Constrained projections to ensure conservation (Berger, 1987; Aguerre et al., 2017) and monotonicity (Rančić, 1995)

Typically in climate applications, flux fields are interpolated using first-order (locally) conservative interpolation, while other scalar fields use non-conservative but higher-order interpolators (e.g. bilinear or biquadratic). For scalar solutions that do not need to be conserved, consistent FE interpolation, patch-wise reconstruction schemes (Fornberg and Piret, 2008) or even nearest neighbor interpolation (Blanco and Rai, 2014) can be performed efficiently using Kd-tree based search and locate infrastructure. Vector fields like velocities or wind stresses are interpolated using these same routines by separately tackling each Cartesian-decomposed component of the field. However, conservative remapping of flux fields require computation of a supermesh (Farrell and Maddison, 2011), or a global intersection mesh that can be viewed as $\Omega_S \bigcup \Omega_T$, which is then used to compute projection weights that contain additional conservation and monotonicity constraints embedded in them.

In general, remapping implementations have three distinct steps to accomplish the solution field projection between grids. First, the target points of interest are identified and located in the source grid, such that, the target cells are a subset of the covering (source) mesh. Next, an intersection between this covering (source) mesh and the target mesh is performed, in order to calculate the individual weight contribution to each target cell, without approximations to the component field discretizations (type $\in [FV, FEM]$ and order). Finally, application of the weight matrix yields the projection required to conservatively transfer the data onto the target grid.

To illustrate some key differences between some **NC** to **GC** or **LC** schemes, we show a 1-D Gaussian hill solution, projected onto a coarse grid through linear basis interpolation and L_2 minimization, as shown in Fig. 2. While the point-wise linear interpolator is computationally efficient, and second-order accurate (Fig. 2-(a)) for smooth profiles, it does not preserve the exact area under the curve. In contrast, the L_2 minimizer conserves the global integral area, but can exhibit spurious oscillatory modes as shown in Fig. 2-(b), when dealing with solutions with strong gradients (Gibbs phenomena (Gottlieb and Shu, 1997)). This demonstration confirms that even for the simple 1-D example, a conservative and monotonic projector is necessary to preserve both stability and accuracy for repeated remapping operator applications, in order to accurately transfer fields between grids with very different resolutions. These requirements are magnified manyfold when dealing with real-world climate simulation data.



(a) Gaussian hill $(x_0, \delta) = (2.5, 1.0)$

(b) Gaussian hill $(x_0, \delta) = (2.5, 0.1)$

Figure 2. An illustration: comparing point interpolation vs L_2 minimization; impact on conservation and monotonicity properties.

While there is a delicate balance in optimizing the computational efficiency of these operations without sacrificing the numerical accuracy or consistency of the procedure, several researchers have implemented algorithms that are useful for a variety of problem domains. In the recent years, the growing interest to rigorously tackle coupled multiphysics applications has led to research efforts focused on developing new regridding algorithms. The Data Transfer Kit (DTK) (Slattery et al., 2013) from Oak Ridge National Labs was originally developed for Nuclear engineering applications, but has been extended for other problem domains through custom adaptors for meshes. DTK is more suited for non-conservative interpolation of scalar variables with either mesh-aware (using consistent discretization bases) or RBF-based meshless (point-cloud) representations (Slattery, 2016) that can be extended to model transport schemes on a sphere (Flyer and Wright, 2007). The Portage library (Herring et al., 2017) from Los Alamos National Laboratory also provides several key capabilities that are useful for geology and geophysics modeling applications including porous flow and seismology systems. Using advanced clipping algorithms to compute the intersection of axis-aligned squares/cubes against faces of a triangle/tetrahedron in 2-d and 3-d respectively, general intersections of arbitrary convex polyhedral domains can be computed efficiently (Powell and Abel, 2015). Support for conservative solution transfer between grids and bound-preservation (to ensure monotonicity) (Certik et al., 2017) has also been recently added. While Portage does support hybrid level parallelism (MPI + OpenMP), demonstrations on large-scale machines to compute remapping weights for climate science applications has not been pursued previously. Based on the software package documentation, support for remapping of vector fields with conservation constraints in DTK and Portage is not directly available for use in climate workflows. Additionally, unavailability of native support for projection of high-order spectral-element data on a sphere onto a target mesh restricts the use of these tools for certain component models in E3SM.

In earth science applications, the state-of-science regridding tool that is often used by many researchers is the ESMF library, and the set of utility tools that are distributed along with it (Collins et al., 2005; Dunlap et al., 2013), to simplify the traditional offline-online computational workflow as described in Section. (1.1). ESMF is implemented in a component architecture (Zhou, 2006) and provides capabilities to generate the remapping weights for different discretization combinations on the source and target grids in serial and parallel. ESMF provides a standalone tool, ESMF_REGRIDWEIGHTGEN, to generate *offline* weights that can be consumed by climate applications such as E3SM and OASIS3-MCT. ESMF also exposes interfaces that enable drivers to directly invoke the remapping algorithms in order to enable the fully-online workflow as well.

Currently, the E3SM components are integrated together in a hub-and-spoke model (Fig. 1 (left)), with the inter-model communication being handled by the Model Coupling Toolkit (MCT) (Larson et al., 2001; Jacob et al., 2005a) in CIME. The MCT library consumes the offline weights generated with ESMF or similar tools, and provides the functionality to interface with models, decompose the field data, and apply the remapping weights loaded from a file during the setup phase. Hence, MCT serves to abstract the communication of data in the E3SM ecosystem. However, without the offline remapping weight generation phase for fixed grid resolutions and model combinations, the workflow in Fig. 1 (a) is incomplete.

Similar to the CIME-MCT driver used by E3SM, OASIS3-MCT (Valcke, 2013; Craig et al., 2017) is a coupler used by many European climate models, where the interpolation weights can be generated offline through SCRIP (included as part of OASIS3-MCT). An option to call SCRIP in an online mode is also available. The OASIS team have recently parallelized SCRIP to speed up its calculation time (OASIS3-MCT v4.0, 2018). OASIS3-MCT also supports application of global conservation operations after interpolation, and does not require a strict hub-and-spoke coupler. Similar to the coupler in CIME, OASIS3-MCT utilizes MCT to perform both the communication of fields between components and for application of the pre-computed interpolation weights in parallel.

ESMF and SCRIP traditionally handle only cell-centered data that targets Finite Volume discretizations (FV to FV projections), with first-order conservation constraints. Hence, generating remapping weights for atmosphere-ocean grids with a Spectral Element (SE) source grid definition requires generation of an intermediate and spectrally equivalent, '*dual*' grid, which matches the areas of the polygons to the weight of each Gauss-Lobatto-Legendre (GLL) nodes. Such procedures add more steps to the offline process and can degrade the accuracy in the remapped solution since the original spectral order is neglected (transformation from p-order to first order). These procedures may also introduce numerical uncertainty in the coupled solution that could produce high solution dispersion (Ullrich et al., 2016).

To calculate remapping weights directly for high-order Spectral Element grids, E3SM uses the TempestRemap C++ library (Ullrich et al., 2013). TempestRemap is a uni-process tool focused on the mathematically rigorous implementations of the remapping algorithms (Ullrich and Taylor, 2015; Ullrich et al., 2016) and provides higher order conservative and monotonicity preserving interpolators with different discretization basis such as (Finite Volume (FV), the spectrally equivalent continuous Galerkin FE with GLL basis (cGLL), and dis-continuous Galerkin FE with GLL basis (dGLL)). This library was developed as part of the effort to fill the gap in generating consistent remapping operators for non-FV discretizations without a need for intermediate dual meshes. Computation of conservative interpolators between any combination of these discretizations (FV, cGLL, dGLL) and grid definitions are supported by TempestRemap library. However, since this regridding tool can only be executed in serial, the usage of TempestRemap prior to the work presented here has been restricted primarily to generating the required mapping weights in the offline stage.

Even though ESMF and OASIS3-MCT have been used in online remapping studies, weight generation as part of a preprocessing step currently remains the preferred workflow for many production climate models. While this decoupling provides flexibility in terms of choice of remapping tools, the data management of the mapping files for different discretizations, field constraints and grids can render provenance, reproducibility and experimentation a difficult task. It also precludes the ability to handle moving or dynamically adaptive meshes in coupled simulations. However, it should be noted that the shift of the remapping computation process from a pre-processing stage in the workflow, to the simulation stage, imposes additional onus on the users to better understand the underlying component grid properties, their decompositions, the solution fields being transferred and the preferred options for computing the weights. This also raises interesting workflow modifications to ensure verification of the online weights such that consistency, conservation and dissipation of key fields are within user-specified constraints. In the implementation discussed here, the online remapping computation uses the exact same input grids, and specifications along with ability to write the weights to file, which can be used to run offline checks as needed.

There are several challenges in scalably computing the regridding operators in parallel, since it is imperative to have both a mesh- and partition-aware datastructure to handle this part of the regridding workflow. A few climate models have begun to calculate weights online as part of their regular operation. The ICON GCM (Wan et al., 2013) uses YAC (Hanke et al., 2016) and FGOALS (Li et al., 2013) uses the C-Coupler (Liu et al., 2014, 2018) framework. These codes expose both offline and online remapping capabilities with parallel decomposition management similar to the ongoing effort presented in the current work for E3SM. Both of these packages provide algorithmic options to perform in-memory search and locate operations, interpolation of field data between meshes with first order conservative remapping, higher-order patch-recovery (Zienkiewicz and Zhu, 1992) and RBF schemes and the NC nearest-neighbor queries. The use of non-blocking communication for field data in these packages align closely with scalable strategies implemented in MCT (Jacob et al., 2005b). While these capabilities are used routinely in production runs for their respective models, the motivation for the work presented here is to tackle coupled high-resolution runs on next generation architectures with scalable algorithms (the high resolution E3SM coupler routinely runs on 13,000 mpi tasks), without sacrificing numerical accuracy for all discretization descriptions (FV, cGLL, dGLL) on unstructured grids.

In the E3SM workflow supported by CIME, the ESMF-regridder understands the component grid definitions, and generates the weight matrices (offline). The CIME driver loads these operators at runtime and places them in MCT datatypes, which treats them as discrete operators to compute the interpolation or projection of data on the target grids. Additional changes in conservation requirements or monotonicity of the field data cannot be imposed as a runtime or post-processing step in such a workflow. In the current work, we present a new infrastructure with scalable algorithms implemented using the MOAB mesh library and TempestRemap package to replace the ESMF-E3SM-MCT remapper/coupler workflow. A detailed review of the algorithmic approach used in the MOAB-TempestRemap (MBTR) workflow, along with the software interfaces exposed to E3SM is presented next.

3 Algorithmic approach

Efficient, conservative and accurate multi-mesh solution transfer workflows (Jacob et al., 2005b; Tautges and Caceres, 2009) are a complex process. This is due to the fact that in order to ensure conservation of critical quantities in a given norm, exact cell intersections between the source and target grids have to be computed. This is complicated in a parallel setting since the domain decompositions between the source and target grids may not have any overlaps, making it a potentially all-to-all collective communication problem. Hence, efficient implementations of regridding operators need to be mesh, resolution, field and decomposition aware in order to provide optimal performance in emerging architectures.

Fully online remapping capability within a complex ecosystem such as E3SM requires a flexible infrastructure to generate the projection weights. In order to fullfill these needs, we utilize the MOAB mesh datastructure combined with the TempestRemap libraries in order to provide an in-memory remapping layer to dynamically compute the weight matrices during the setup phase of the simulations for static source-target grid combinations. For dynamically adaptive and moving grids, the remapping operator can be recomputed at runtime as needed. The introduction of such a software stack allows higher order conservation of fields while being able to transfer and maintain field relations in parallel, within the context of the fully decomposed mesh view. This is an improvement to the E3SM workflow where MCT is oblivious to the underlying mesh datastructure in the component models. Having a fully mesh-aware implementation also provides opportunities to implement dynamic load-balancing algorithms to gain optimal performance on large-scale machines. YAC interpolator (Hanke et al., 2016) and the multidimensional Common Remapping software (CoR) in C-Coupler2 (Liu et al., 2018) provide similar capabilities to perform a parallel tree-based search for point location and interpolation through various supported numerical schemes.

MOAB is a fully distributed, compact, array-based mesh datastructure, and the local entity lists are stored in ranges along with connectivity and ownership information, rather than explicit lists, thereby leading to a high degree of memory compression. The memory constraints per process scales well in parallel Tautges and Caceres (2009), and is only proportional to the number of entities in the local partition, which reduces as number of processes increases (strong scaling limit). This is similar to the Global Segment Map (GSMap) in MCT, which in contrast is stored in every processor, leading to $O(N_x)$ memory requirements.

In order to illustrate the online remapping algorithm implemented with the MOAB-TempestRemap infrastructure, we define the following terms. Let $N_{c,S}$ be the component processes for source mesh, $N_{c,T}$ be the component processes for target mesh and N_x be the coupler processes where the remapping operator is computed. More generally, the problem statement can be defined as: transfer a solution field U defined on the domain Ω_S and processes $N_{c,S}$, to the domain Ω_T and processes $N_{c,T}$, through a centralized coupler with domain information $\Omega_S \bigcup \Omega_T$ defined on N_x processes. Such a complex online remapping workflow for projecting the field data from a source to target mesh follows the algorithm shown in Algorithm. 1.

In the following sections, the new E3SM online remapping interface implemented with a combination of the MOAB and TempestRemap libraries is explained. Details regarding the algorithmic aspects to compute conservative, high-order remapping weights in parallel, without sacrificing discretization accuracy on next generation hardware are presented.

3.1 Interfacing to Component Models in E3SM

Within the E3SM simulation ecosystem, there are multiple component models (atmosphere-ocean-land-ice-runoff) that are coupled to each other. While the MCT infrastructure only allowed for a numbering of the grid points, the new MOAB-based coupler infrastructure provides the ability to natively interface to the underlying mesh, and understand the field Degree-of-Freedom (DoF) data layout associated with each model. MOAB can understand the difference between values on a cell center and values on a cell edge or corner. In the current work, the MOAB mesh database has been used to create the relevant integration abstraction for the HOMME atmosphere model (Thomas and Loft, 2005; Taylor et al., 2007) (cubed-sphere SE grid) and the Model for Prediction Across Scales (MPAS) ocean model (Ringler et al., 2013; Petersen et al., 2015) (polygonal meshes with holes representing land and ice regions). Since details of the mesh are not available at the level of the coupler interface, additional MOAB (Fortran) calls via the `iMOAB` interface are added to HOMME and MPAS component models to describe the details of the unstructured mesh to MOAB with explicit vertex and element connectivity information, in contrast to MCT coupler that is oblivious to the underlying grid. The atmosphere-ocean coupling requires the largest computational effort in the coupler (since they cover about 70% of the coupled domain), and hence bulk of discussions in the current work will focus on remapping and coupling between these two component models.

MOAB can handle the finite-element zoo of elements on a sphere (triangles, quadrangles, and polygons) making it an appropriate layer to store both the mesh layout (vertices, elements, connectivity, adjacencies) and the parallel decomposition for the component models along with information on shared and ghosted entities. While having a uniform partitioning methodology across components may be advantageous for improving the efficiency of coupled climate simulations, the parallel partition of the meshes are chosen according to the requirements in individual component solvers. Fig. 3 shows examples of partitioned SE and MPAS meshes, visualized through the native MOAB plugin for VisIt (VisIt, 2005).

The coupled field data that is to be remapped from the source grid to the target grid also needs to be serialized as part of the MOAB mesh database in terms of an internally contiguous, MOAB data storage structure named a ‘Tag’ (Tautges et al., 2004). For E3SM, we use element-based tags to store the partitioned field data that is required to be remapped between components. Typically, the number of DoF per element ($nDoF_e$) is determined based on the underlying discretization; $nDoF_e = p^2$ values in HOMME where p is the order of SE discretization, and $nDoF_e = 1$ for the FV discretization in MPAS ocean. With this

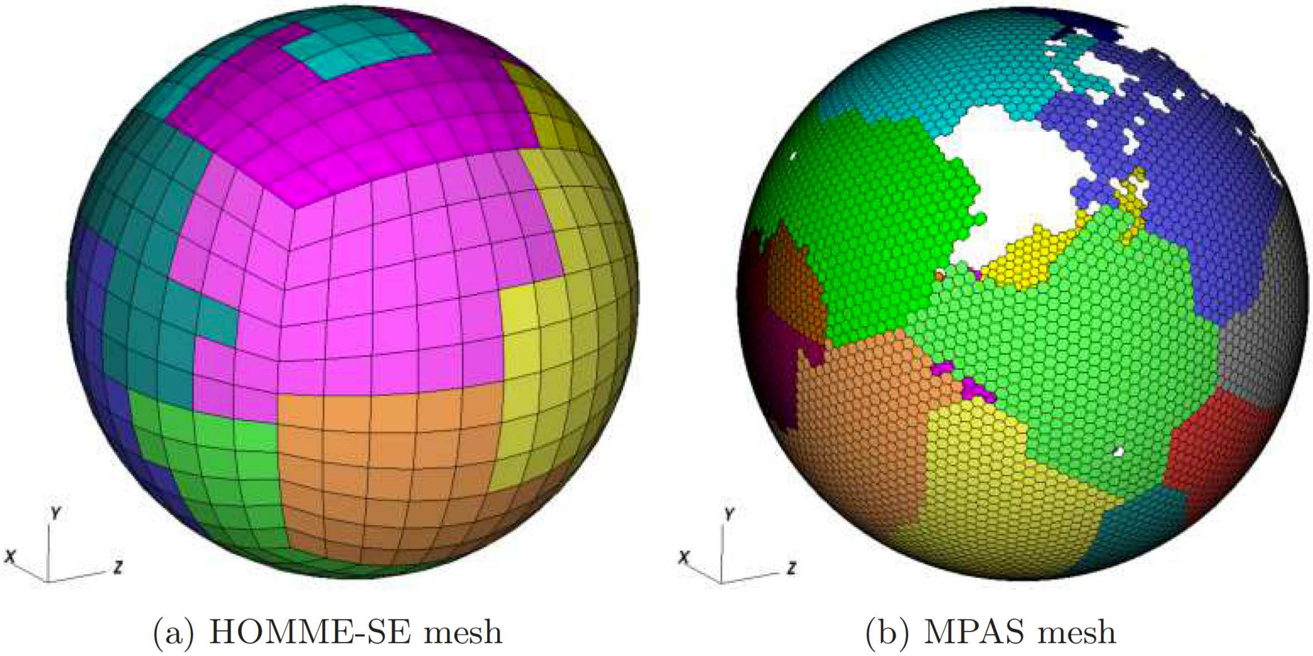
Algorithm 1 MOAB-TempestRemap parallel regridding workflow

```
Input: Partitioned and distributed native component meshes on $N_{c,S}$ source and $N_{c,T}$ target processes
Result: Remapping weight matrix $W_{S \to T}$ computed for a source ($S$) and target ($T$) mesh pair on $N_x$ coupler processes

Scope: Coupler $N_x \leftarrow$ component mesh $N_{c,l}$, where $l \in [S,T]$
for each component $l \in [S,T]$ do
    – create in-memory copy of component unstructured mesh and data using MOAB interfaces (Section. 3.1)
    – migrate MOAB component mesh to coupler; repartition from $N_{c,l} \to N_x$ (Section. 3.2)
end for

Scope: Compute pair-wise intersection mesh on coupler processes $N_x$
for each mesh pair to be regridded: $\Omega_S$ and $\Omega_T$ in $N_x$ do
    Ensure: {local source mesh fully covers target mesh}
    if $(\Omega_T - \Omega_T \cap \Omega_S) \neq 0$ then
        collectively gather coverage mesh $\Omega_{Sc}$ on $N_x \mid (\Omega_T - \Omega_T \cap \Omega_{Sc}) = 0$ (Section. 3.4.1)
    end if
    – store communication graph to send/receive between $N_{c,l}$ and $N_x$
    – compute $\Omega_{ST} = \Omega_{Sc} \cap \Omega_T$ through an advancing-front algorithm (Löhner and Parikh, 1988; Gander and Japhet, 2009) (Section. 3.3.1)
    – evaluate source/target element mapping for $e_i \in \Omega_{ST}$
    – exchange ghost cell information for $\Omega_{ST}$
end for

Scope: Integrate over $\Omega_{ST}$ to compute remapping weights
for each intersection polygon element $e_i \in \Omega_{ST}$ do
    – Tessellate $e_i$ into triangular elements with reproducible ordering
    – Compute projection integral with consistent Triangular quadrature rules
    – Determine row/col DoF coupling through $e_i$ parent association to $\Omega_S / \Omega_T$
    – Assemble local matrix weights such that $W_{S \to T} = \sum_1^{N_x} w_{ij}$, where $w_{ij}$ represents the coupling between local target DoF (row $i$) and source DoF (col $j$) in projection operator (Section. 3.5)
end for
```



(a) HOMME-SE mesh

(b) MPAS mesh

Figure 3. MOAB representation of partitioned component meshes.

complete description of the mesh and associated data for each component model, MOAB contains the necessary information to proceed with the remapping workflow.

3.2 Migration of Component Mesh to Coupler

E3SM's driver supports multiple modes of partitioning the various components in the global processor space. This is usually fine tuned based on the estimated computational load in each physics, according to the problem case definition. A sample process-execution (PE) layout for a E3SM run on 9000 processes with ATM on 5400 and OCN on 3600 tasks is shown in Fig. 4. In the case shown in the schematic, $N_{c,ATM} = 5400$, $N_{c,OCN} = 3600$ and $N_x = 4800$. In such a PE layout, the atmosphere component mesh from HOMME, distributed on $N_{c,ATM}$ (5400) tasks needs to be migrated and redistributed on N_x (4800 tasks). Similarly, from $N_{c,OCN}$ (3600) to N_x (4800) tasks for the MPAS ocean mesh. In the hub-and-spoke coupling model as shown in Fig. 1, the remapping computation is performed only in the coupler processors. Hence, inference of a communication pattern becomes necessary to ensure scalable data transfers between the components and the coupler. In the existing implementation, MCT handles such communication, which is being replaced by point-to-point communication kernels in MOAB to transfer mesh and data between different components or component-coupler PEs. Note that in a distributed coupler, source and target components can communicate directly, without any intermediate transfers (through the coupler). Under the unified infrastructure provided by MOAB, minimal changes are required to enable either the hub-and-spoke or the

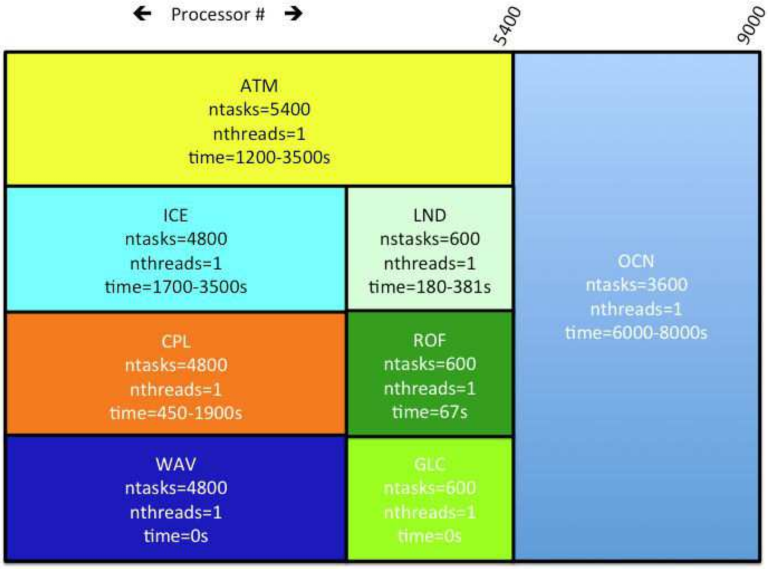
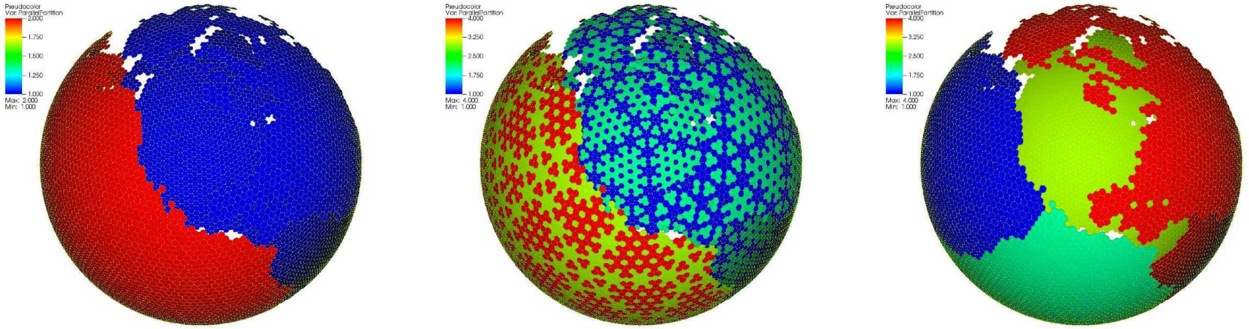


Figure 4. Example E3SM process execution layout for a problem case

distributed coupler for E3SM runs, which offers opportunities to minimize time to solution without any changes in spatial coupling behavior.

For illustration, let N_c be the number of component processing elements, and N_x be the number of coupler processing elements. In order to migrate the mesh and associated data from N_c to N_x, we first compute a trivial partition of elements that map directly in the partition space, the same partitioning as used in the CIME-MCT coupler. In MOAB, we have exposed parallel graph and geometric repartitioning schemes through interfaces to Zoltan (Devine et al., 2002) or ParMetis (Karypis et al., 1997), in order to evaluate optimized migration patterns to minimize the volume of data communicated between component and coupler. We intend to analyze the impact of different migration schemes on the scalability of the remapping process in Section. (4). These optimizations have the potential to minimize data movement in the MOAB-based remapper, and to make it a competitive data broker to replace the current MCT (Jacob et al., 2005a) coupler in E3SM.

We show an example of a decomposed ocean mesh (polygonal MPAS mesh) that is replicated in a E3SM problem case run on two processes in Fig. 5. Fig. 5-(a) is the original decomposed mesh on 2 tasks $\in N_c$, while Fig. 5-(b) and Fig. 5-(c) show the impact of migrating a mesh from 2 N_c tasks to 4 tasks $\in N_x$ with a trivial linear partitioner and a Zoltan based geometric online partitioner. The decomposition in Fig. 5-(b) shows that the element ID based linear partitioner can produce bad data locality, which may require large number of nearest neighbor communications when computing a source coverage mesh. The resulting communication pattern can also make the migration, and coverage computation process non-scalable on larger core counts. In contrast, in Fig. 5-(c), the Zoltan partitioners produce much better load balanced decompositions with Hypergraph (PHG), Recursive Coordinate Bisection (RCB) or Recursive Inertial Bisection (RIB) algorithms to reduce communication overheads in



(a) Component mesh on 2 tasks (b) Migrated mesh on 4 tasks (Trivial partitioner) (c) Migrated mesh on 4 tasks (Zoltan partitioner)

Figure 5. Migration strategies to repartition from $N_c \rightarrow N_x$

the remapping workflow. In order to better understand the impact of online decomposition strategies on the overall remapping process, we need to better understand the impact of the repartitioner on two communication-heavy steps.

1. Mesh migration from component to coupler involving communication between $N_{c,s/t}$ and N_x,

2. Computing the coverage mesh requiring gather/scatter of source mesh elements to cover local target elements.

In a hub-and-spoke model with online remapping, the best coupler strategy will require a simultaneous partition optimization for all grids such that mesh migration includes constraints on geometric coordinates of component pairs. While such extensions can be implemented within the infrastructure presented here, the performance discussions in Section 4 will only focus on the trivial and Zoltan-based partitioners. It is also worth noting that in a distributed coupler, pair-wise migration optimizations can be performed seamlessly using a master(*target*)-slave(*source*) strategy to maximize partition overlaps.

3.3 Computing the Regridding Operator

Standard approaches to compute the intersection of two convex polygonal meshes involve the creation of a Kd-tree (Hunt et al., 2006) or BVH-tree datastructure (Ize et al., 2007) to enable fast element location of relevant target points. In general, each target point of interest is located on the source mesh by querying the tree datastructure, and the corresponding (source) element is then marked as a contributor to the remapping weight computation of the target DoF. This process is repeated to form a list of source elements that interact directly according to the consistent discretization basis. TempestRemap, ESMF and YAC use variations of this search-and-clip strategy tailored to their underlying mesh representations.

3.3.1 Advancing Front Intersection – A Linear Complexity Algorithm

The intersection algorithm used in this paper follows the ideas from (Löhner and Parikh, 1988; Gander and Japhet, 2013), in which two meshes are covering the same domain. At the core is an advancing front method that aims to traverse through the source and target meshes to compute a union (super) mesh. First, two convex cells from the source coverage mesh and the target meshes that intersect are identified by using an adaptive Kd-tree search tree datastructure. This process also includes determination of the seed for the advancing front. Advancing in both meshes using face adjacency information, incrementally all possible intersections are computed (Březina and Exner, 2017) accurately to a user defined tolerance (default = $1e-15$).

While the advancing front algorithm is not restricted to convex cells, the intersection computation is simpler if they are strictly convex. If concave polygons exist in the initial source or target meshes, they are recursively decomposed into simpler convex polygons, by splitting along interior diagonals. Note that the intersection between two convex polygons results in a strictly convex polygon. Hence, the underlying intersection algorithm remains robust to resolve even arbitrary non-convex meshes covering the same domain space.

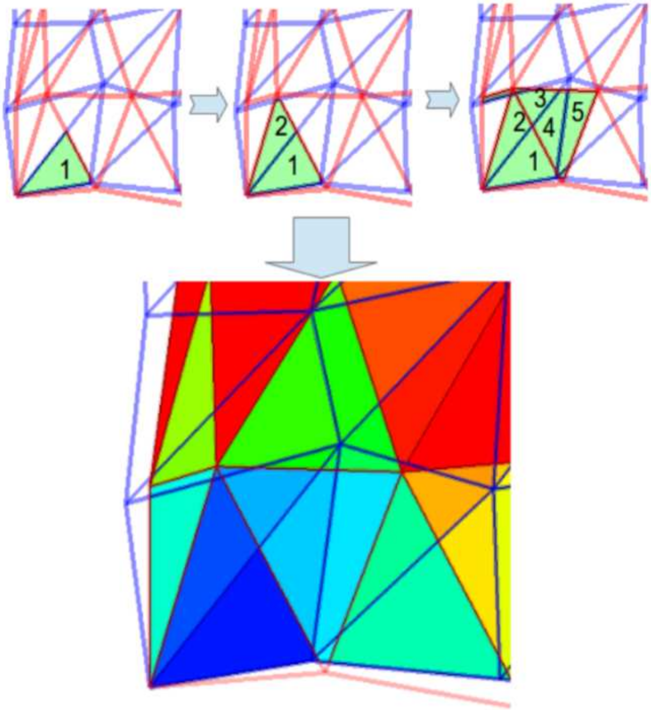


Figure 6. Illustration of the advancing front intersection algorithm.

Fig. 6 illustrates how the algorithm advances. Each target cell is resolved by building a local queue of source cells that intersect the target cell. Source cells are added to a local queue incrementally, using adjacency information. At the same time, a global queue with seeds is formed, containing pairs of source/target cells that have the probability to intersect. When there are no more source cells in the local queue, the algorithm advances to the next seed from the global queue, and the algorithm repeats. This workflow has been illustrated in both serial (Mahadevan et al., 2018a) and in parallel with partitioned meshes (Mahadevan et al., 2018b).

This flooding-like advancing front needs a stable and robust methodology of intersecting edges/segments in two cells that belong to different meshes. Any pair of segments that intersect can appear in four different pairs of cells. A list of intersection points is maintained on each target edge, so that the intersection points are unique. Also, a geometric tolerance is used to merge intersection points that are close to each other, or if they are proximal to the original vertices in both meshes. Decisions regarding whether points are inside, outside or at the boundary of a convex enclosure are handled separately. If necessary, more robust techniques such as adaptive precision arithmetic procedures used in *Triangle* (Shewchuk, 1996), can be employed to resolve the fronts more accurately. Note that the advancing front strategy can be employed for meshes with topological holes (e.g. ocean meshes, in which the continents are excluded) without any further modifications by using a new seed for each disconnected region in the target mesh.

Note on Gnomonic Projection for Spherical Geometry

Meshes that appear in climate applications are often on a sphere. Cell edges are considered to be great circle arcs. A simple gnomonic projection is used to project the edges on one of the six planes parallel to the coordinate axis, and tangent to the sphere (Ullrich et al., 2013). With this projection, all curvilinear cells on the sphere are transformed to linear polygons on a gnomonic plane, which simplifies the computation of intersection between multiple grids. Once the intersection points and cells are computed on the gnomonic plane, these are projected back on to the original spherical domain without approximations. This is possible due to the fact that intersection can be computed to machine precision as the edges become straight lines in a gnomonic plane (projected from great circle arcs on a sphere). If curves on a sphere are not great circle arcs (splines, for example), the intersection between those curves have to be computed using some nonlinear iterative procedures such as Newton Raphson (depending on the representation of the curves).

3.4 Parallel Implementation Considerations

Existing Infrastructure from MOAB (Tautges et al., 2004) was used to extend the advancing front algorithm in parallel. The expensive intersection computation can be carried out independently, in parallel, once we redistribute the source mesh to envelope the target mesh areas fully, in a step we refer to as ‘source coverage mesh’ computation.

3.4.1 Computation of a Source Coverage Mesh

We select the target mesh as the driver for redistribution of the source mesh. On each task, we first compute the bounding box of the local target mesh. This information is then gathered and communicated to all tasks, and used for redistribution of the local source mesh. Cells that intersect the bounding boxes of other processors are sent to the corresponding owner task. This workflow guarantees that the target mesh on each processor is completely enveloped by the covering mesh repartitioned from its original source mesh decomposition, as shown in Fig. 7. In other words, the covering mesh is a superset of the target mesh in each task. It is important to note that some source coverage cells might be sent to multiple processors during this step, depending on the target mesh resolution and decomposition. The parallel infrastructure in MOAB is heavily leveraged (Tautges et al., 2012) to utilize the scalable, *crystal router algorithm (Fox et al., 1989; Schliephake and Laure, 2015)* in order to scalably communicate the covering cells to different processors.

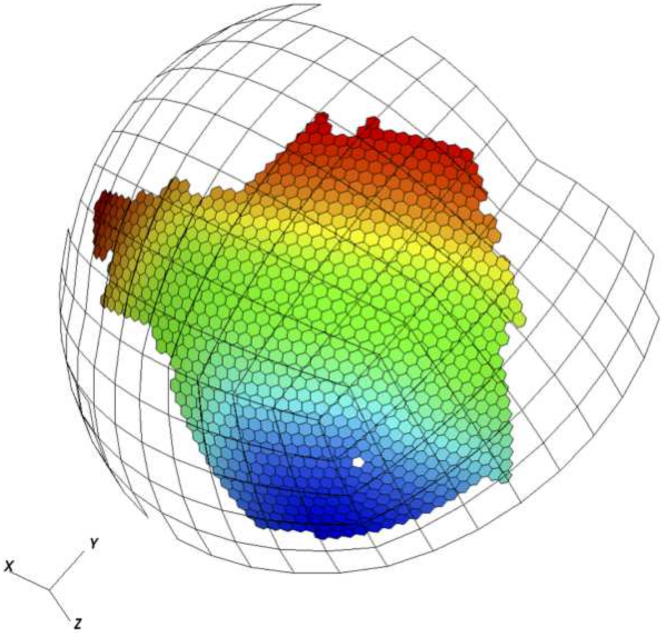


Figure 7. Source coverage mesh fully covers local target mesh; local intersection proceeds between atmosphere (Quadrangle) and ocean (Polygonal) grids.

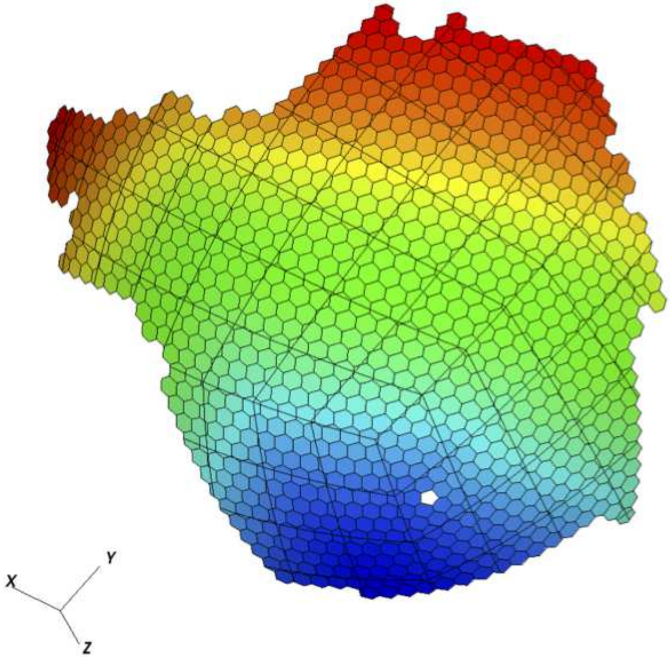


Figure 8. Intersection mesh computed with the coverage and target mesh in a single process.

Once the relevant covering mesh is accumulated locally on each process, the intersection computation can be carried out in parallel, completely independently, using the advancing front algorithm (Section. (3.3.1)), as shown in Fig. 8. After computation of the local intersection polygons, the vertices on the shared edges between processes are communicated to avoid duplication. In order to ensure consistent local conservation constraints in the weight matrix in the parallel setting, there might be a need for additional communication of ghost intersection elements to nearest neighbors. This extra communication step is only required for computing interpolators for flux variables, and can generally be avoided when transferring scalar fields with non-conservative bilinear or higher-order interpolations.

The parallel advancing front algorithm presented here to globally compute the intersection supermesh can be extended to expose finer grained parallelism using hybrid-threaded (OpenMP) programming or a task-based execution model, where each task handles a unique front in the computation queue. Such task or hybrid threaded parallelism can be employed in combination with the MPI-based mesh decompositions. Using local partitions computed with Metis and through standard coloring approaches, each thread or task can then proceed to compute the intersection elements until the front collides with another, and until all the overlap elements have been computed in each process. Such a parallel hybrid algorithm has the potential to scale well even on heterogeneous architectures and provides options to improve the computational throughput of the regridding process (Löhner, 2014).

3.5 Computation of Remapping operator with TempestRemap

For illustration, consider a scalar field $\boldsymbol{U}$ discretized with standard Galerkin FEM on source Ω_1 and target Ω_2 meshes with different resolutions. The projection of the scalar field on the target grid is in general given as follows.

$$\boldsymbol{U}_2(\boldsymbol{\Omega_2}) = \boldsymbol{\Pi}_1^2 \boldsymbol{U}_1(\boldsymbol{\Omega_1}) \tag{1}$$

where, $\boldsymbol{\Pi}_1^2$ is the discrete solution interpolator of $\boldsymbol{U}$ defined on $\boldsymbol{\Omega_1}$ to $\boldsymbol{\Omega_2}$. This interpolator $\boldsymbol{\Pi}_1^2$ in Eq. (1) is often referred to as the remapping operator, which is pre-computed in the coupled climate workflows using ESMF and TempestRemap. For embedded meshes, the remapping operator can be calculated exactly as a restriction or prolongation from the source to target grid. However, for general unstructured meshes and in cases where the source and target meshes are topologically different, the numerical integration to assemble $\boldsymbol{\Pi}_1^2$ needs to be carried out on the supermesh (Ullrich and Taylor, 2015). Since a unique source and target parent element exists for every intersection element belonging to the supermesh $\boldsymbol{\Omega_1} \bigcup \boldsymbol{\Omega_2}$, $\boldsymbol{\Pi}_1^2$ is assembled as the sum of local mass matrix contributions on the intersection elements, by using the consistent discretization basis for the source and target field descriptions (Ullrich et al., 2016). The intersection mesh typically contains arbitrary convex polygons and hence subsequent triangulation may be necessary before evaluating the integration. This global linear operator directly couples source and target DoFs based on the participating intersection element parents (Ullrich et al., 2009).

MOAB supports point-wise FEM interpolation (bilinear and higher-order spectral) with local or global subset normalization (Tautges and Caceres, 2009), in addition to a conservative first-order remapping scheme. But higher order conservative monotone weight computations are currently unsupported natively. To fill this gap for climate applications, and to leverage existing developments in rigorous numerical algorithms to compute the conservative weights, interfaces to TempestRemap in MOAB were added to scalably compute the remap operator in parallel, without sacrificing field discretization accuracy. The MOAB interface to the E3SM component models provides access to the underlying type and order of field discretization, along with the global partitioning for the DoF numbering. Hence the projection or the weight matrix can be assembled in parallel by traversing through the intersection elements, and associating the appropriate source and target DoF parent to columns and rows respectively. The MOAB implementation uses a sparse matrix representation using the Eigen3 library (Guennebaud et al., 2010) to store the local weight matrix. Except for the particular case of projection onto a target grid with cGLL description, the matrix rows do not share any contributions from the same source DoFs. This implies that for FV and dGLL target field descriptions, the application of the weight matrix does not require global collective operations and sparse matrix vector applications scale ideally (still memory bandwidth limited). In the cGLL case, we perform a reduction of the parallel vector along the shared DoFs to accumulate contributions exactly. However, it is non-trivial to ensure full bit-for-bit reproducibility during such reductions.

It is also possible to use the transpose of the remapping operator computed between a particular source and target component combination, to project the solution back to the original source grid. Such an operation has the advantage of preserving the consistency and conservation metrics originally imposed in finding the remapping operator and reduces computation cost by avoiding recomputation of the weight matrix for the new directional pair. For example, when computing the remap operator between atmosphere and ocean models (with holes), it is advantageous to use the atmosphere model as the source grid, since the advancing front seed computation may require multiple iterations if the front begins within a hole. Additionally, such transpose vector applications can also make the global coupling symmetric, which may have favorable implications when pursuing implicit temporal integration schemes.

3.6 Note on MBTR Remapper Implementation

The remapping algorithms presented in the previous section are exposed through a combination of implementations in MOAB and TempestRemap libraries. Since both the libraries are written in C++, direct inheritance of key datastructures such as the GridElements (mesh) and OfflineMap (projection weights) are available to minimize data movement between the libraries. Additionally, Fortran codes such as E3SM can invoke computations of the intersection mesh and the remapping weights through specialized language-agnostic interfaces in MOAB: `iMOAB` (Mahadevan et al., 2015). These interfaces offer the flexibility to query, manipulate and transfer the mesh between groups of processes that represent the component and coupler processing elements.

Using the iMOAB interfaces, the E3SM coupler can coordinate the online remapping workflow during the setup phase of the simulation, and compute the projection operators for component and scalar or vector coupled field combinations. For each pair of coupled components, the following sequence of steps are then executed to consistently compute the remapping operator and transfer the solution fields in parallel.

1. `iMOAB_SendMesh` and `iMOAB_ReceiveMesh`: Send the component mesh (defined on $N_{c,l}$ processes), and receive the complete unstructured mesh copy in the coupler processes (N_x). This mesh migration undergoes an online mesh repartition either through a trivial decomposition scheme or with advanced Zoltan algorithms (geometric or graph partitioners)

2. `iMOAB_ComputeMeshIntersectionOnSphere`: The advancing front intersection scheme is invoked to compute the overlap mesh in the coupler processes

3. `iMOAB_CoverageGraph`: Update the parallel communication graph based on the (source) coverage mesh association in each process

4. `iMOAB_ComputeScalarProjectionWeights`: The remapping weight operator is computed and assembled with discretization-specific (FV, SE) calls to TempestRemap, and stored in Eigen3 SparseMatrix object

Once the remapping operator is serialized in-memory for each coupled scalar and flux fields, this operator is then used at every timestep to compute the actual projection of the data.

1. `iMOAB_SendElementTag` and `iMOAB_ReceiveElementTag`: Using the coverage graph computed previously, direct one-to-one communication of the field data is enabled between $N_{c,l}$ and N_x, before and after application of the weight operator

2. `iMOAB_ApplyScalarProjectionWeights`: In order to compute the field interpolation or projection from the source component to the target component, a matvec product of the weight matrix and the field vector defined on the source grid is performed. The source field vector is received from source processes $N_{c,s}$ and after weight application, the target field vector is sent to target processes $N_{c,l}$

Additionally, to facilitate offline generation of projection weights, a MOAB based parallel tool `mbtempest` has been written in C++, similar to ESMF and TempestRemap (serial) standalone tools. `mptempst` can load the source and target meshes from files, in parallel, and compute the intersection and remapping weights through TempestRemap. The weights can then be written back to a SCRIP-compatible file format, for any of the supported field discretization combinations in source and destination components. Added capability to apply the weight matrix onto the source solution field vectors, and native visualization plugins in VisIt for MOAB, simplify the verification of conservation and monotonicity for complex remapping workflows. This workflow allows users to validate the underlying assumptions for remapping solution fields across unstructured grids, and can be executed in both a serial and parallel setting.

4 Results

Evaluating the performance of the in-memory, MOAB-TempestRemap (MBTR) remapping infrastructure requires recursive profiling and optimization to ensure scalability for large-scale simulations. In order to showcase the advantage of using the mesh-aware MOAB datastructure as the MCT coupler replacement, we need to understand the per task performance of the regridder in addition to the parallel point locator scalability, and overall time for remapping weight computation. Note that except for the weight application for each solution field from a source grid to a target grid, the in-memory copy of the component meshes, migration to coupler PEs, computation of intersection elements and remapping weights are done only once during the setup phase in E3SM, per coupled component model pair.

4.1 Serial Performance

We compare the total cost for computing the supermesh and the remapping weights for several source and target grid combinations through three different methods to determine the serial computational complexity.

1. ESMF: Kd-tree based regridder and weight generation for first/second order FV→FV conservative remapping
2. TempestRemap: Kd-tree based supermesh generation and conservative, monotonic, high-order remap operator for FV→FV, SE→FV, SE→SE projection
3. MBTempest: Advancing front intersection with MOAB and conservative weight generation with TempestRemap interfaces

Fig. 9 shows the serial performance of the remappers for computing the conservative interpolator from Cubed-Sphere grids to polygonal MPAS grids of different resolutions for a FV→FV field transfer. This total time includes the computation of

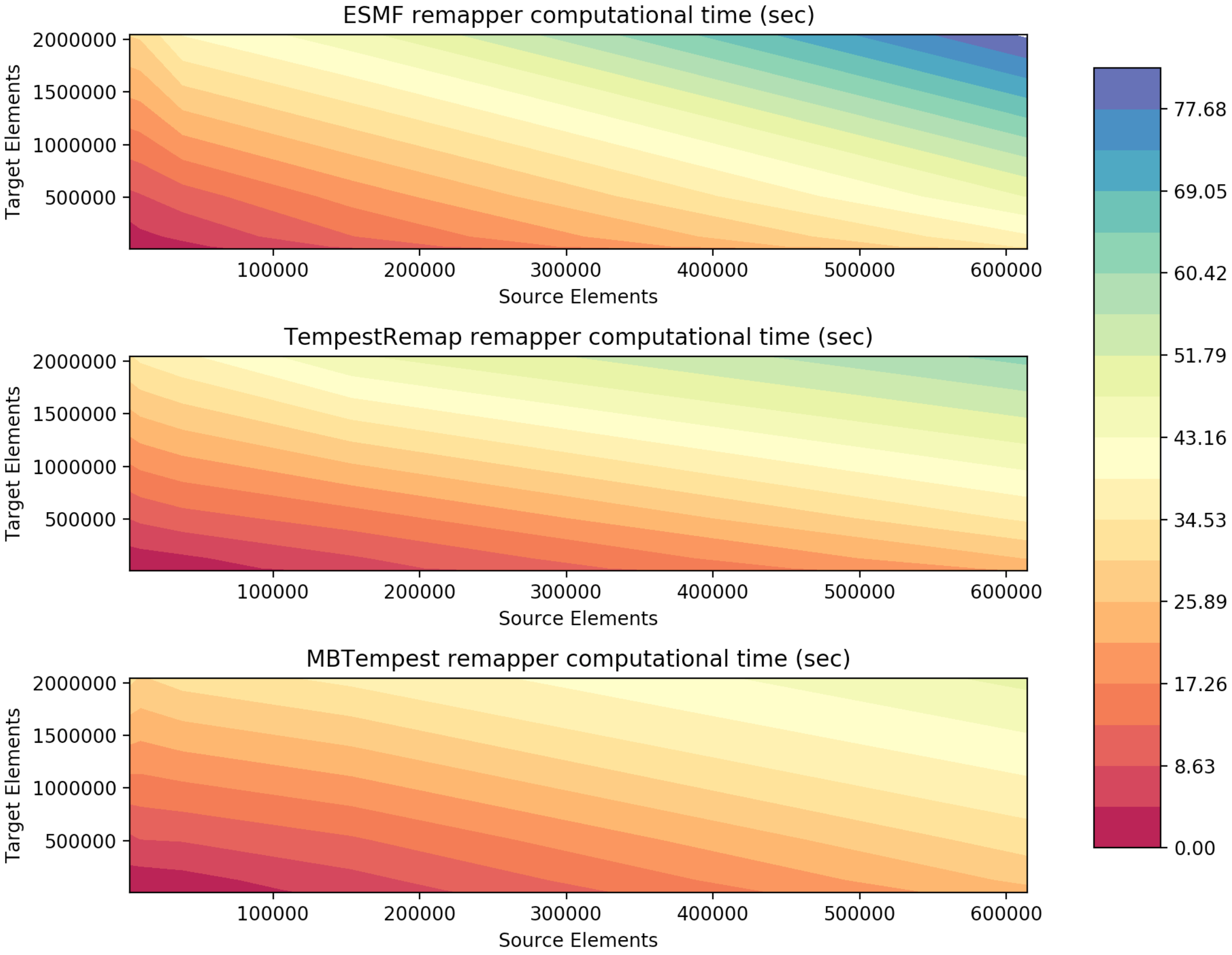


Figure 9. Comparison of serial regridding computation (supermesh and projection weight generation) between ESMF, TempestRemap, and MBTempest

intersection mesh or supermesh, in addition to the remapping weights with field conservation specifications. These serial runs were executed on a machine with 8x Intel Xeon(R) CPU E7-4820 @ 2.00GHz (total of 64 cores) and 1.47 TB of RAM. As the source grid resolution increases, the advancing front intersection with linear complexity outperforms the Kd-tree intersection algorithms used by TempestRemap and ESMF. The time spent in the remapping task, including the overlap mesh generation, provides an overall metric on the single task performance when memory bandwidth or communication concerns do not dominate in a parallel run. In this comparison with three remapping software libraries, the total computational time in the fine resolution limit as $\frac{nele(source)}{nele(target)} \approx 1$ consistently increases (going diagonally from left to right in Fig. 9). We note that the serial version of TempestRemap is comparable to ESMF and can even provide better timings on the highly refined cases, while the MBTempest remapper consistently outperforms both the tools, with a 2x speedup on average. The relatively better performance in MBTempest is accomplished through the linear complexity advancing front algorithm, which further offers avenues to incorporate finer grain task or thread level parallelism to accelerate the on-node performance on multicore and GPGPU architectures.

4.2 Scalability of the MOAB Kd-tree Point Locator

In addition to being able to compute the supermesh between Ω_S and Ω_T, MOAB also offers datastructures to query source elements containing points that correspond to the target DoFs locations. This operation is critical in evaluating bilinear and biquadratic interpolator approximations for scalar variables when conservative projection is not required by the underlying coupled model. The solution interpolation for the multi-mesh case involves two distinct phases.

1. Setup phase: Use Kd-tree to build the search datastructure to locate points corresponding to vertices in the target mesh on the source mesh

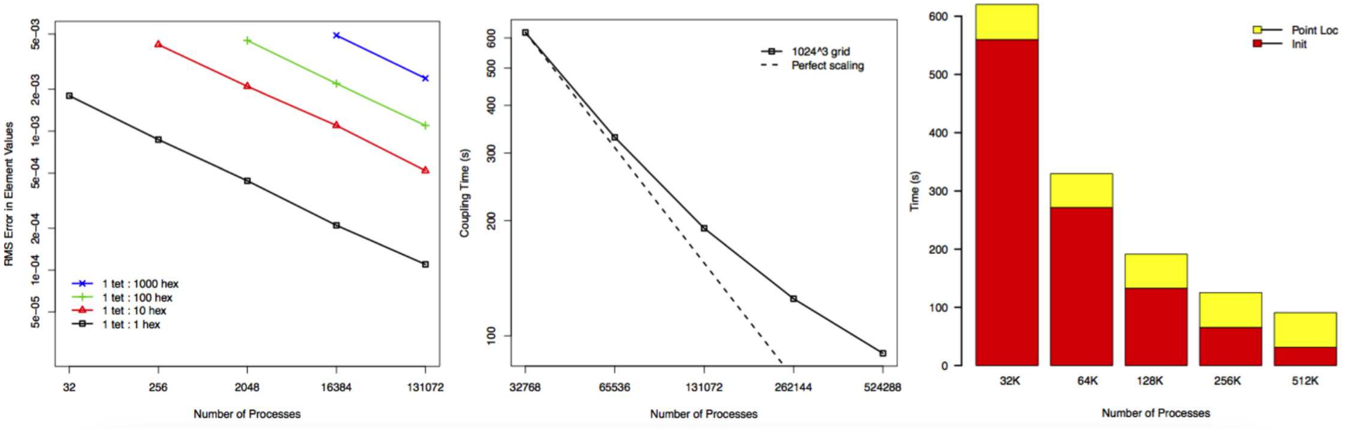
2. Run phase: Use the elements containing the located points to compute consistent interpolation onto target mesh vertices

Studies were performed on the BlueGene-Q machine (Mira) at ANL to evaluate the strong and weak scalability of the parallel Kd-tree point search implementation in MOAB. The scalability results were generated with the CIAN2 coupling mini-app (Morozov and Peterka, 2016), which links to MOAB to handle traversal of the unstructured grids and transfer of solution fields between the grids. For this case, a series of hexahedral and tetrahedral meshes were used to interpolate an analytical solution. By changing the basis interpolation order, and mesh resolutions, the convergence of the interpolator was verified to provide theoretical accuracy orders of convergence in the asymptotic fine limit.

The performance tests were executed on the IBM BlueGene/Q Mira at 16 MPI ranks per node, with 2GB RAM per MPI rank, at up to 500K MPI processes. The strong scaling results and error convergence were computed with a grid size of 1024^3. The solution interpolation on varying mesh resolutions were performed by projecting an analytical solution from a Tetrahedral→Hexahedral→Tetrahedral grid, with total number of points/rank varied between [2K, 32K] in the study.

Fig. 10 shows a strong scaling efficiency of around 50% is achieved on a maximum of 512K cores (66% of Mira). We note that the computational complexity of the Kd-tree data structure scales as $O(nlog(n))$ asymptotically, and the point location phase during initial search setup dominates the total cost on higher core counts. This is evident in the timing breakdown for each phase shown in Fig. 10-(c). Since the point location is performed only once during simulation startup, while the interpolation is performed multiple times per timestep during the run, we expect the total cost of the projection for scalar variables to be amortized over transient climate simulations with fixed grids. Further investigations with optimal BVH-tree (Larsen et al., 1999) or R-tree implementations for these interpolation cases could help reduce the overall cost.

The full 3-D point location and interpolation operations provided by MOAB are comparable to the implementation in Common Remapping component used in the C-Coupler (Liu et al., 2013) and provide relatively much stronger scalability on larger core counts (Liu et al., 2014) for the remapping operation. Such higher-order interpolators for multicomponent physics variables can provide better performance in atmospheric chemistry calculations. Currently, only the NC bilinear or biquadratic



(a) Tet→Hex RMS Error

(b) Strong Scaling

(c) Strong Scaling by Component

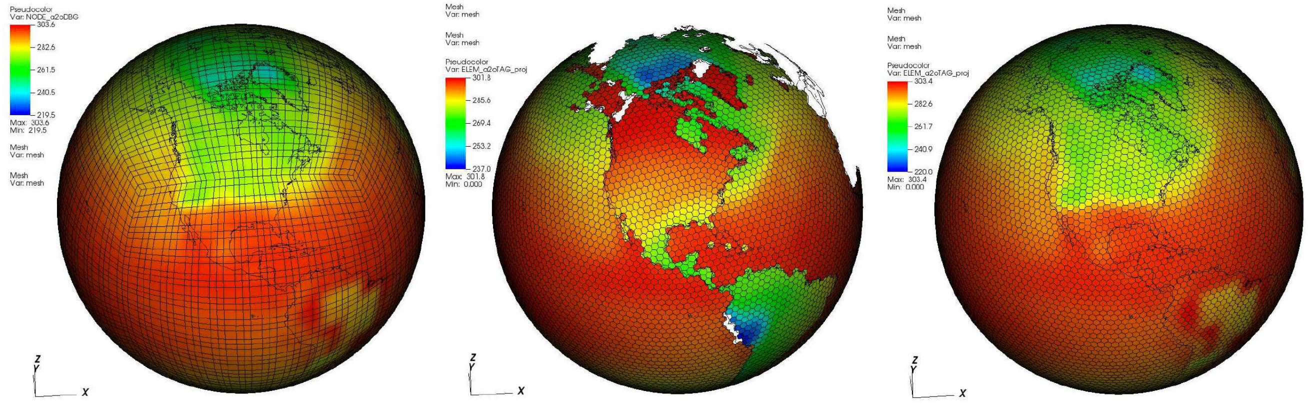
Figure 10. MOAB 3-d Kd-tree Point Location: Strong scaling on Mira (BG/Q)

interpolation of scalar fields with subset normalization (Tautges and Caceres, 2009) is supported directly in MOAB (via Kd-tree point location and interpolation), and advancing front intersection algorithm does not make use of these data-structures. In contrast, TempestRemap and ESMF use a Kd-tree search to not only compute the location of points, but also to evaluate the supermesh $\Omega_S \bigcup \Omega_T$, and hence the computational complexity for the intersection mesh determination scales as $O(nlog(n))$, in contrast to the linear complexity ($O(n)$) of the advancing front intersection algorithm implemented in MOAB.

4.3 The Parallel MBTR Remapping Algorithm

The MBTR online weight generation workflow within E3SM was employed to verify and test the projection of real simulation data generated during the coupled atmosphere-ocean model runs. A choice was made to use the model-computed temperature on the lowest level of the atmosphere, since the heat fluxes that nonlinearly couples the atmosphere and ocean models are directly proportional to this interface temperature field. By convention, the fluxes are computed on the ocean mesh, and hence the atmosphere temperature must be interpolated onto MPAS polygonal mesh. We use this scenario as a test case for demonstrating the strong scalability results in this section.

The atmosphere run with approximately 4 degree grid size and 11 elements per edge on a cubed-sphere (NE11) in E3SM, and the projection of its lowest level temperature onto two different MPAS meshes (with approximate grid size of 240km) are shown in Fig. 11. The conservative projection from SE→FV on a mesh with holes (Fig. 11-(b)) and without holes (Fig. 11-(c)) corresponding to land regions, is presented here to show the difference in the remapped solutions.



(a) Original SE field on CS atmosphere mesh

(b) Remapped field on MPAS mesh with holes

(c) Remapped field on MPAS mesh without holes

Figure 11. Projection of the NE11 SE bottom atmospheric temperature field onto the MPAS ocean grid

4.3.1 Scaling Comparison of Conservative Remappers (FV→FV)

The strong scaling studies for computation of remapping weights to project a FV solution field between CS grids of varying resolutions was performed on the Blues large-scale cluster (with 16 Sandy Bridge Xeon E5-2670 2.6GHz cores and 32 GB RAM per node) at ANL, and the Cori supercomputer at NERSC (with 64 Haswell Xeon E5-2698v3 2.3GHz cores and 128 GB RAM per node). Fig. 12 shows that the MBTR workflow consistently outperforms ESMF on both the machines as the number of processes used by the coupler is increased. The timings shown here represent the total remapping time i.e., cumulative computational time for generating the super mesh and the (conservative) remapping weights.

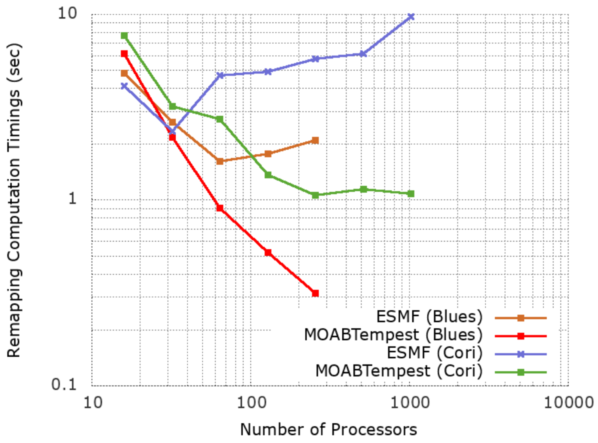


Figure 12. CS (E=614400 quads) → CS (E=153600 quads) remapping (-m conserve) on LCRC/ALCF and NERSC machines

The relatively better scaling for MOAB on the Blues cluster is due to faster hardware and memory bandwidth compared to the Cori machine. The strong scaling efficiency approaches a plateau on Cori Haswell nodes as communication costs for the coverage mesh computation start dominating the overall remapping processes, especially in the limit of $\frac{nele}{process} \rightarrow 1$ at large node counts.

4.3.2 Strong Scalability of Spectral Projection (SE→FV)

To further evaluate the characteristics of in-memory remapping computation, along with cost of application of the weights during a transient simulation, a series of further studies were executed on the NERSC Cori system to determine the spectral projection of a real dataset between atmosphere and ocean components in E3SM. The source mesh contains 4th order spectral element temperature data defined on Gauss-Lobatto quadrature nodes (cGLL discretization) of the CS mesh, and the projection is performed on a MPAS polygonal mesh with holes (FV discretization). A direct comparison to ESMF was unfeasible in this study since the traditional workflow requires the computation of a dual mesh transformation of the spectral grid. Hence, only timings for MBTR workflow is shown here.

Two specific cases were considered for this SE→FV strong scaling study with conservation and monotonicity constraints.

1. **Case A (NE30):** 1-degree CS (30 edges per side) SE mesh (nele=5400 quads) with $p = 4$ to MPAS mesh (nele=235160 polygons)

2. **Case B (NE120):** 0.25-degree CS (120 edges per side) SE mesh (nele=86400 quads) with $p = 4$ to MPAS mesh (nele=3693225 polygons)

The performance tests for each of these cases were launched with three different process execution layouts for the atmosphere, ocean components and the coupler.

(a) Fully colocated PE layout: $N_{atm} = N_x$ and $N_{ocn} = N_x$

(b) Disjoint-ATM model PE layout: $N_{atm} = N_x/2$ and $N_{ocn} = N_x$

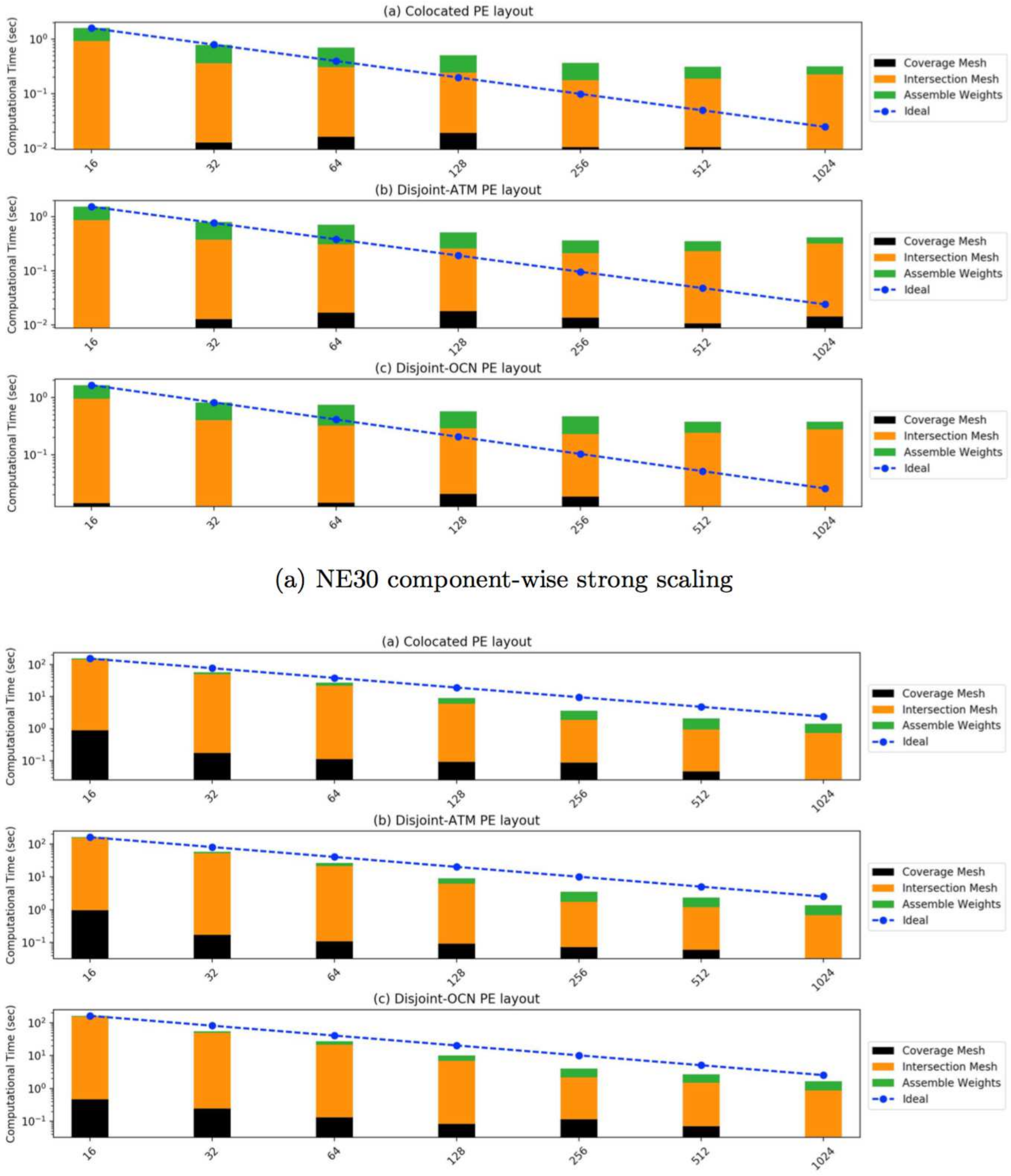
(c) Disjoint-OCN model PE layout: $N_{atm} = N_x$ and $N_{ocn} = N_x/2$

A breakdown of computational time for key tasks on Cori with up to 1024 processes for both the cases is tabulated in Table 1 on a fully colocated decomposition i.e., $N_{ocn} = N_{atm} = N_x$. It is clear that the computation of parallel intersection mesh strong scales well for these production cases, especially for larger mesh resolutions (Case B). For the smaller source and target mesh resolution (Case A), we notice that the intersection time hits a lower bound that is dominated by the computation of the coverage mesh to enclose the target mesh in each task. It is important to stress that this one time setup call to compute remap operator, per component pair, is relatively much cheaper compared to individual component and solver initializations and get amortized over longer transient simulations. It is also worth noting that as the I/O bandwidth in emerging architectures are not scaling in line with the compute throughput, such an online workflow can generally be faster than parallel I/O for reading the weights from file at scale. The MBTR implementation is also flexible to allow loading the weights from file directly in order

Table 1. Strong scaling on Cori for SE→FV projection with two different resolutions

Number of processors	Case A		Case B	
	Intersection (sec)	Compute Weights (sec)	Intersection (sec)	Compute Weights (sec)
16	0.936846	0.64983	145.623	9.732
32	0.449022	0.429028	53.1244	5.78093
64	0.377767	0.373476	22.7167	4.92151
128	0.255154	0.270574	6.70485	2.79397
256	0.180136	0.18272	2.26435	1.71835
512	0.162388	0.104737	1.25471	0.928622
1024	0.203354	0.0932475	0.680122	0.618943

to preserve the existing coupler process with MCT. In comparison to the computation of the intersection mesh, the time to assemble the remapping weight operator in parallel is generally smaller. Even though both of these operations are performed only once during the setup phase of the E3SM simulation, the weight operator computation involves several validation checks that utilize collective MPI operations, which do destroy the embarrassingly parallel nature of the calculation, once appropriate coverage mesh is determined in each task.



(a) NE30 component-wise strong scaling

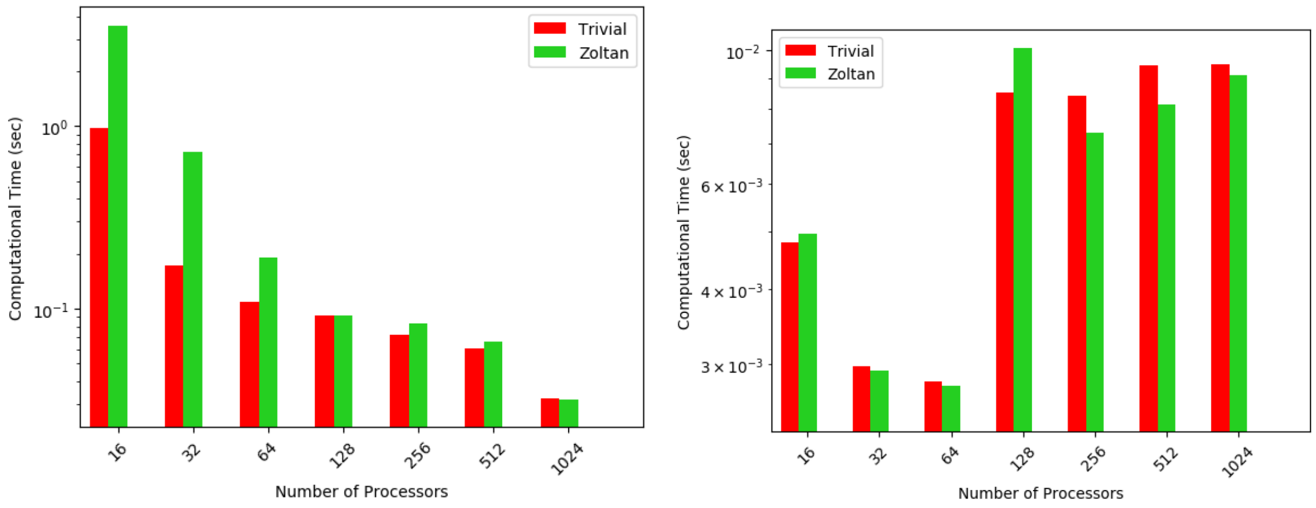
(b) NE120 component-wise strong scaling

Figure 13. Strong scaling study for the NE30 and NE120 cases for spectral projection with Zoltan repartitioner on Cori

The component-wise breakdown for the advancing front intersection mesh, the parallel communication graph for sending and receiving data between component and coupler, and finally, the remapping weight generation for the SE→FV setup for NE30 and NE120 cases are shown in Fig. 13. The cumulative time for this remapping process is shown to scale linearly for NE120 case, even if the parallel efficiency decreases significantly in the NE30 case, as expected based on the results in Table 1. Note that the MBTR workflow provides a unique capability to consistently and accurately compute SE→FV projection weights in parallel, without any need for an external pre-processing step to compute the dual mesh (as required by ESMF) or running the entire remapping process in serial (TempestRemap). Also note that Fig. 13 confirms that the overall scaling of the remapping algorithm is nearly independent of the PE layout for the simulation.

4.4 Effect of partitioning strategy

In order to determine the effect of partitioning strategies described in Fig. 5, the NE120 case with the trivial decomposition and Zoltan geometric partitioner (RCB) were tested in parallel. Fig. 14 compares the two strategies for optimizing the mesh migration from the component to coupler. These strategies play a critical role in task mapping and data locality for the source coverage mesh computation, in addition to determining the communication graph complexity between the components and the coupler. This comparison highlights that the coverage mesh cost reduces uniformly at scale, while the trivial partitioning scheme behaves better on lower core counts as shown in Fig. 14-(a). The communication of field data between the atmosphere component and the coupler resulting from the partitioning strategy is a critical operation during the transient simulation, and generally stays within network latency limits in Cori, as the message size reduces. Eventhough the communication kernel does not show good ideal scaling on increasing node counts, the relative cost of the operation is insignificant in comparison to total time spent in individual component solvers. Note that production climate model solvers require multiple data fields to be remapped at every rendezvous timestep, and hence the size of the packed messages may be larger for such simulations (volume should remain similar to Fig. 14-(b)). We also note that there is a factor of 3 increase in the communication time to send and receive data, which occurs after the 64 process count on Cori in Fig. 14-(b). This is an artifact of the additional communication latency due to the transition from an intra-node (each Haswell node in Cori accommodates 64 processes) to inter-node nearest neighbor data transfer when using multiple nodes.



(a) Source coverage mesh computation

(b) Send/Receive field data between N_x and N_{atm}

Figure 14. Scaling of the communication kernels driven with the parallel graph computed with a trivial redistribution and the Zoltan geometric (RCB) repartitioner for the NE120 case with $N_{ocn} = N_x$ and $N_{atm} = N_x/2$ on Cori

4.5 Note on Application of Weights

Generally, operations involving Sparse Matrix-Vector (SpMV) products are memory bandwidth limited (Bell and Garland, 2009), and occur during the application of remapping weights operator on to the source solution field vector, in order to compute the field projection onto the target grid. In addition to the communication of field data shown in Fig. 14-(b), the cost of remapping weight application in parallel (presented in Fig. 15) determines the total cost of the remapping operation during runtime. Except for the case of cGLL target discretizations, the parallel SpMV operation during the weight application do not involve any global collective reductions. In the current E3SM and OASIS3-MCT workflow, these operations are handled by the MCT library. In high resolution simulations of E3SM, the total time for the remapping operation in MCT is primarily dominated by the communication costs based on the communication graph, similar to the MBTR workflow. However, a direct comparison between these two workflows is not yet possible, but we expect the aggregated communication strategies in the crystal router algorithm (Fox et al., 1989) in MOAB, to provide relatively better performance at scale.

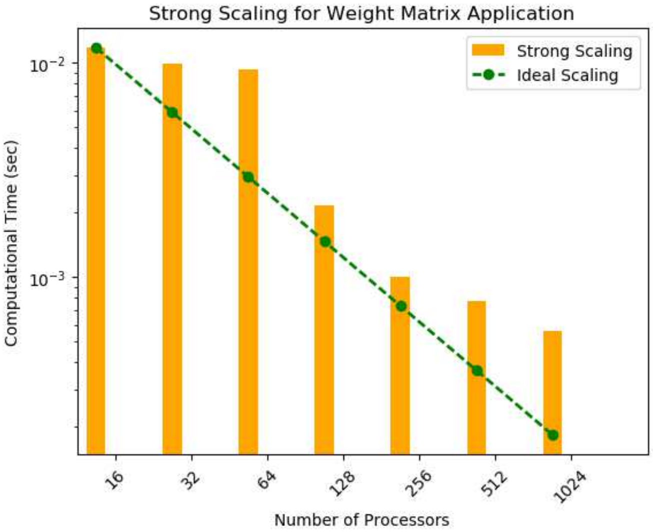


Figure 15. SE→FV remapping weight operator application on Cori

5 Conclusion

Understanding and controlling primary sources of errors in a coupled system dynamically, will be key to achieving predictable and verifiable climate simulations on emerging architectures. Traditionally, the computational workflow for coupled climate simulations has involved two distinct steps, with an offline pre-processing phase using remapping tools to generate solution field projection weights (ESMF, TempestRemap, SCRIP), which is then consumed by the coupler to transfer field data between the component grids.

The offline steps include generating grid description files and running the offline tools with the problem-specific options. Additionally many of state-of-science tools such as ESMF and SCRIP require additional steps to specially handle interpolators from SE grids. Such workflows create bottlenecks that do not scale, and can inhibit scientific research productivity. When experimenting with refined grids, a goal for E3SM, this tool chain has to excercised repeatedly. Additionally, when component meshes are dynamically modified, either through mesh adaptivity or dynamical mesh movement to track moving boundaries, the underlying remapping weights must be recomputed on the fly.

To overcome some of these limitations, we have presented scalable algorithms and software interfaces to create a direct component coupling with online regridding and weight generation tools. The remapping algorithms utilize the numerics exposed by TempestRemap, and leverage the parallel mesh handling infrastructure in MOAB to create a scalable in-memory remapping infrastructure that can be integrated with existing coupled climate solvers. Such a methodology invalidates the need for dual grids, preserves higher-order spectral accuracy, and locally conserves the field data, in addition to monotonicity constraints, when transferring solutions between grids with non-matching resolutions.

The serial and parallel performance of the MOAB advancing front intersection algorithm with linear complexity ($O(n)$) was demonstrated for a variety of source and target mesh resolution combinations, and compared with the current state-of-science regridding tools such as ESMF (serial/parallel) and TempestRemap (serial) that have a $O(nlog(n))$ complexity using the Kd-tree datastructure. The MOAB-TempestRemap (MBTR) software infrastructure yields a balance of both the scalable performance on emerging architectures without sacrificing discretization accuracy for component field interpolators. There are also several optimizations in the MBTR algorithms that can be implemented to improve finer-grained parallelism on heterogeneous architectures, and to minimize data movement with better partitioning in combination with load rebalancing strategies. Such a software infrastructure provides a foundation to build a new coupler to replace the current offline-online, hub-and-spoke MCT-based coupler in E3SM, and offer extensions to enable a fully distributed coupling paradigm (without the need for a centralized coupler) to minimize computational bottlenecks in a task-based workflow.

Code availability. Information on the availability of source code for the algorithmic infrastructure and models featured in this paper is tabulated below.

Short name	Code availability
E3SM	E3SM Project (2018) is under active development funded by the US Department of Energy. E3SM version 1.1 has been publicly released under an open-source 3-clause BSD license in August 2018, and available at GitHub.
MOAB	MOAB Tautges et al. (2004) is an open-source library under the umbrella of the SIGMA toolkit (2014) Mahadevan et al. (2015), and is publicly available under the Lesser GNU Public License (v3) on BitBucket. v5.1.0 was released on Jan 07, 2019 and available here. DOI: 10.5281/zenodo.2584863.
TempestRemap	The TempestRemap Ullrich and Taylor (2015); Ullrich et al. (2016) source code is available under a BSD open-source license and hosted in GitHub. v2.0.2 was released on Dec 19, 2018 and available here.

Video supplement. The video supplements for the serial and parallel advancing front mesh intersection algorithm to compute the supermesh ($\mathbf{\Omega_S} \bigcup \mathbf{\Omega_T}$) of a source ($\mathbf{\Omega_S}$)and target ($\mathbf{\Omega_T}$) grid is demonstrated.

Short name	Video description and availability
Serial advancing front mesh intersection	Intersection between CS and MPAS grids on a single task is illustrated. DOI:10.6084/m9.figshare.7294901
Parallel advancing front mesh intersection	Simultaneous parallel Intersection between CS and MPAS grids on two different tasks are illustrated side by side. DOI:10.6084/m9.figshare.7294919

Author contributions. VM and RJ wrote the paper (with comments from IG and JS). VM and IG designed and implemented the MOAB integration with TempestRemap library, along with exposing the necessary infrastructure for online remapping through iMOAB interfaces. IG and JS configured the MOAB-TempestRemap remapper within E3SM, and verified weight generation to transfer solution fields between atmosphere and ocean component models. VM conducted numerical verification studies and executed both the serial and parallel scalability studies on Blues and Cori LCF machines to quantify performance characteristics of the remapping algorithms. The broader project idea was conceived by Andy Salinger (SNL), RJ, VM, and IG.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. Support for this work was provided by the Climate Model Development and Validation - Software Modernization project, and partially by the SciDAC Coupling Approaches for Next Generation Architectures (CANGA) project, which are funded by the US Department of Energy (DOE), Office of Science Biological and Environmental Research Program. CANGA is also funded by the DOE Office of Advanced Scientific Computing Research. This research used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357, and resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. We gratefully acknowledge the computing resources provided on Blues, a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory. We would also like to thank Dr. Paul Ullrich at University of California, Davis for several helpful discussions regarding remapping schemes and, in particular, implementations in TempestRemap. We would also like to thank the anonymous reviewers for their indepth comments that helped clarify and improve the manuscript.

References

Aguerre, H. J., Damián, S. M., Gimenez, J. M., and Nigro, N. M.: Conservative handling of arbitrary non-conformal interfaces using an efficient supermesh, Journal of Computational Physics, 335, 21–49, 2017.

Beljaars, A., Dutra, E., Balsamo, G., and Lemarié, F.: On the numerical stability of surface-atmosphere coupling in weather and climate models, Geoscientific Model Development Discussions, 10, 977–989, 2017.

Bell, N. and Garland, M.: Implementing sparse matrix-vector multiplication on throughput-oriented processors, in: Proceedings of the conference on high performance computing networking, storage and analysis, p. 18, ACM, 2009.

Berger, M. J.: On conservation at grid interfaces, SIAM journal on numerical analysis, 24, 967–984, 1987.

Blanco, J. L. and Rai, P. K.: nanoflann: a C++ header-only fork of FLANN, a library for Nearest Neighbor (NN) wih KD-trees, https://github.com/jlblancoc/nanoflann, 2014.

Březina, J. and Exner, P.: Fast algorithms for intersection of non-matching grids using Plücker coordinates, Computers & Mathematics with Applications, 74, 174 – 187, https://doi.org/https://doi.org/10.1016/j.camwa.2017.01.028, http://www.sciencedirect.com/science/article/pii/S0898122117300792, 5th European Seminar on Computing ESCO 2016, 2017.

Certik, O., Ferenbaugh, C., Garimella, R., Herring, A., Jean, B., Malone, C., and Sewell, C.: A Flexible Conservative Remapping Framework for Exascale Computing, https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-17-21749, sIAM Conference on Computational Science & Engineering Feb 2017, 2017.

Collins, N., Theurich, G., Deluca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang, W., Hill, C., and Da Silva, A.: Design and implementation of components in the Earth System Modeling Framework, The International Journal of High Performance Computing Applications, 19, 341–350, 2005.

Craig, A., Valcke, S., and Coquart, L.: Development and performance of a new version of the OASIS coupler, OASIS3-MCT_3.0, Geoscientific Model Development, 10, 3297–3308, https://doi.org/10.5194/gmd-10-3297-2017, https://www.geosci-model-dev.net/10/3297/2017/, 2017.

Craig, A. P., Jacob, R., Kauffman, B., Bettge, T., Larson, J., Ong, E., Ding, C., and He, Y.: CPL6: The New Extensible, High Performance Parallel Coupler for the Community Climate System Model, The International Journal of High Performance Computing Applications, 19, 309–327, https://doi.org/10.1177/1094342005056117, https://doi.org/10.1177/1094342005056117, 2005.

Craig, A. P., Vertenstein, M., and Jacob, R.: A new flexible coupler for earth system modeling developed for CCSM4 and CESM1, The International Journal of High Performance Computing Applications, 26, 31–42, https://doi.org/10.1177/1094342011428141, https://doi.org/10.1177/1094342011428141, 2012.

de Boer, A., van Zuijlen, A., and Bijl, H.: Comparison of conservative and consistent approaches for the coupling of non-matching meshes, Computer Methods in Applied Mechanics and Engineering, 197, 4284–4297, https://doi.org/10.1016/j.cma.2008.05.001, http://dx.doi.org/10.1016/j.cma.2008.05.001, 2008.

Dennis, J. M., Edwards, J., Evans, K. J., Guba, O., Lauritzen, P. H., Mirin, A. A., St-Cyr, A., Taylor, M. A., and Worley, P. H.: CAM-SE: A scalable spectral element dynamical core for the Community Atmosphere Model, The International Journal of High Performance Computing Applications, 26, 74–89, 2012.

Devine, K., Boman, E., Heaphy, R., Hendrickson, B., and Vaughan, C.: Zoltan Data Management Services for Parallel Dynamic Applications, Computing in Science and Engineering, 4, 90–97, 2002.

Dunlap, R., Rugaber, S., and Mark, L.: A feature model of coupling technologies for Earth System Models, Computers & geosciences, 53, 13–20, 2013.

E3SM Project: Energy Exascale Earth System Model (E3SM), [Computer Software] https://dx.doi.org/10.11578/E3SM/dc.20180418.36, https://doi.org/10.11578/E3SM/dc.20180418.36, https://dx.doi.org/10.11578/E3SM/dc.20180418.36, 2018.

Farrell, P. and Maddison, J.: Conservative interpolation between volume meshes by local Galerkin projection, Computer Methods in Applied Mechanics and Engineering, 200, 89 – 100, https://doi.org/http://dx.doi.org/10.1016/j.cma.2010.07.015, http://www.sciencedirect.com/science/article/pii/S0045782510002276, 2011.

Fleishman, S., Cohen-Or, D., and Silva, C. T.: Robust moving least-squares fitting with sharp features, ACM transactions on graphics (TOG), 24, 544–552, 2005.

Flyer, N. and Wright, G. B.: Transport schemes on a sphere using radial basis functions, Journal of Computational Physics, 226, 1059–1084, 2007.

Fornberg, B. and Piret, C.: On choosing a radial basis function and a shape parameter when solving a convective PDE on a sphere, Journal of Computational Physics, 227, 2758–2780, 2008.

Fox, G., Johnson, M., Lyzenga, G., Otto, S., Salmon, J., Walker, D., and White, R. L.: Solving Problems On Concurrent Processors Vol. 1: General Techniques and Regular Problems, Computers in Physics, 3, 83–84, 1989.

Gander, M. J. and Japhet, C.: An algorithm for non-matching grid projections with linear complexity, in: Domain Decomposition Methods in Science and Engineering XVIII, pp. 185–192, Springer, 2009.

Gander, M. J. and Japhet, C.: Algorithm 932: PANG: software for nonmatching grid projections in 2D and 3D with linear complexity, ACM Transactions on Mathematical Software (TOMS), 40, 6, 2013.

Gottlieb, D. and Shu, C.-W.: On the Gibbs phenomenon and its resolution, SIAM review, 39, 644–668, 1997.

Grandy, J.: Conservative remapping and region overlays by intersecting arbitrary polyhedra, J. Comput. Phys., 148, 433–466, 1999.

Guennebaud, G., Jacob, B., et al.: Eigen v3, http://eigen.tuxfamily.org, 2010.

Hanke, M., Redler, R., Holfeld, T., and Yastremsky, M.: YAC 1.2.0: new aspects for coupling software in Earth system modelling, Geoscientific Model Development, 9, 2755–2769, https://doi.org/10.5194/gmd-9-2755-2016, https://www.geosci-model-dev.net/9/2755/2016/, 2016.

Herring, A. M., Certik, O., Ferenbaugh, C. R., Garimella, R. V., Jean, B. A., Malone, C. M., and Sewell, C. M.: (U) Introduction to Portage, Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-17-20831, 2017.

Hill, C., DeLuca, C., Balaji, Suarez, M., and Silva, A. d.: The architecture of the earth system modeling framework, Computing in Science & Engineering, 6, 18–28, 2004.

Hunt, W., Mark, W. R., and Stoll, G.: Fast kd-tree construction with an adaptive error-bounded heuristic, in: Interactive Ray Tracing 2006, IEEE Symposium on, pp. 81–88, IEEE, 2006.

Hurrell, J. W., Holland, M. M., Gent, P. R., Ghan, S., Kay, J. E., Kushner, P. J., Lamarque, J.-F., Large, W. G., Lawrence, D., Lindsay, K., et al.: The community earth system model: a framework for collaborative research, Bulletin of the American Meteorological Society, 94, 1339–1360, 2013.

Ize, T., Wald, I., and Parker, S. G.: Asynchronous BVH construction for ray tracing dynamic scenes on parallel multi-core architectures, in: Proceedings of the 7th Eurographics conference on Parallel Graphics and Visualization, pp. 101–108, Eurographics Association, 2007.

Jacob, R., Larson, J., and Ong, E.: M× N communication and parallel interpolation in Community Climate System Model Version 3 using the model coupling toolkit, The International Journal of High Performance Computing Applications, 19, 293–307, 2005a.

Jacob, R., Larson, J., and Ong, E.: M× N communication and parallel interpolation in Community Climate System Model Version 3 using the model coupling toolkit, The International Journal of High Performance Computing Applications, 19, 293–307, 2005b.

Jiao, X. and Heath, M. T.: Common-refinement-based data transfer between non-matching meshes in multiphysics simulations, International Journal for Numerical Methods in Engineering, 61, 2402–2427, 2004.

Jones, P. W.: First-and second-order conservative remapping schemes for grids in spherical coordinates, Monthly Weather Review, 127, 2204–2210, 1999.

Karypis, G., Schloegel, K., and Kumar, V.: Parmetis: Parallel graph partitioning and sparse matrix ordering library, Version 1.0, Dept. of Computer Science, University of Minnesota, p. 22, 1997.

Larsen, E., Gottschalk, S., Lin, M. C., and Manocha, D.: Fast proximity queries with swept sphere volumes, Tech. rep., Technical Report TR99-018, Department of Computer Science, University of North Carolina, 1999.

Larson, J. W., Jacob, R. L., Foster, I., and Guo, J.: The model coupling toolkit, in: International Conference on Computational Science, pp. 185–194, Springer, 2001.

Lauritzen, P. H., Nair, R. D., and Ullrich, P. A.: A conservative semi-Lagrangian multi-tracer transport scheme (CSLAM) on the cubed-sphere grid, Journal of Computational Physics, 229, 1401 – 1424, https://doi.org/http://dx.doi.org/10.1016/j.jcp.2009.10.036, http://www.sciencedirect.com/science/article/pii/S002199910900597X, 2010.

Li, L., Lin, P., Yu, Y., Wang, B., Zhou, T., Liu, L., Liu, J., Bao, Q., Xu, S., Huang, W., Xia, K., Pu, Y., Dong, L., Shen, S., Liu, Y., Hu, N., Liu, M., Sun, W., Shi, X., Zheng, W., Wu, B., Song, M., Liu, H., Zhang, X., Wu, G., Xue, W., Huang, X., Yang, G., Song, Z., and Qiao, F.: The flexible global ocean-atmosphere-land system model, Grid-point Version 2: FGOALS-g2, Advances in Atmospheric Sciences, 30, 543–560, https://doi.org/10.1007/s00376-012-2140-6, https://doi.org/10.1007/s00376-012-2140-6, 2013.

Liu, L., Yang, G., and Wang, B.: CoR: a multi-dimensional common remapping software for Earth System Models, in: The Second Workshop on Coupling Technologies for Earth System Models (CW2013), available at: https://wiki.cc.gatech.edu/CW2013/index.php/Program (last access: 8 May 2014), 2013.

Liu, L., Yang, G., Wang, B., Zhang, C., Li, R., Zhang, Z., Ji, Y., and Wang, L.: C-Coupler1: a Chinese community coupler for Earth system modeling, Geoscientific Model Development, 7, 2281–2302, https://doi.org/10.5194/gmd-7-2281-2014, https://www.geosci-model-dev.net/7/2281/2014/, 2014.

Liu, L., Zhang, C., Li, R., Wang, B., and Yang, G.: C-Coupler2: a flexible and user-friendly community coupler for model coupling and nesting, Geoscientific Model Development, 11, 3557–3586, https://doi.org/10.5194/gmd-11-3557-2018, https://www.geosci-model-dev.net/11/3557/2018/, 2018.

Löhner, R.: Recent advances in parallel advancing front grid generation, Archives of Computational Methods in Engineering, 21, 127–140, 2014.

Löhner, R. and Parikh, P.: Generation of three-dimensional unstructured grids by the advancing-front method, International Journal for Numerical Methods in Fluids, 8, 1135–1149, 1988.

Mahadevan, V., Grindeanu, I. R., Ray, N., Jain, R., and Wu, D.: SIGMA Release v1.2 - Capabilities, Enhancements and Fixes, https://doi.org/10.2172/1224985, 2015.

Mahadevan, V., Grindeanu, I., Jacob, R., and Sarich, J.: MOAB: Serial Advancing Front Intersection Computation, https://doi.org/10.6084/m9.figshare.7294901.v1, https://figshare.com/articles/MOAB_Serial_Advancing_Front_Intersection_Computation/7294901, 2018a.

Mahadevan, V., Grindeanu, I., Jacob, R., and Sarich, J.: MOAB: Parallel Advancing Front Mesh Intersection Algorithm, https://doi.org/10.6084/m9.figshare.7294919.v2, https://figshare.com/articles/MOAB_Parallel_Advancing_Front_Mesh_Intersection_Algorithm/72949 2018b.

Morozov, D. and Peterka, T.: Block-Parallel Data Analysis with DIY2, 2016.

OASIS3-MCT v4.0: OASIS3-MCT 4.0 official release, https://portal.enes.org/oasis/news/oasis3-mct_4-0-official-release, [Online; accessed 10-October-2018], 2018.

Petersen, M. R., Jacobsen, D. W., Ringler, T. D., Hecht, M. W., and Maltrud, M. E.: Evaluation of the arbitrary Lagrangian–Eulerian vertical coordinate method in the MPAS-Ocean model, Ocean Modelling, 86, 93–113, 2015.

Plimpton, S. J., Hendrickson, B., and Stewart, J. R.: A parallel rendezvous algorithm for interpolation between multiple grids, Journal of Parallel and Distributed Computing, 64, 266–276, 2004.

Powell, D. and Abel, T.: An exact general remeshing scheme applied to physically conservative voxelization, Journal of Computational Physics, 297, 340 – 356, https://doi.org/https://doi.org/10.1016/j.jcp.2015.05.022, http://www.sciencedirect.com/science/article/pii/S0021999115003563, 2015.

Rančić, M.: An efficient, conservative, monotonic remapping for semi-Lagrangian transport algorithms, Monthly Weather Review, 123, 1213–1217, 1995.

Reichler, T. and Kim, J.: How well do coupled models simulate today's climate?, Bulletin of the American Meteorological Society, 89, 303–312, 2008.

Ringler, T., Petersen, M., Higdon, R. L., Jacobsen, D., Jones, P. W., and Maltrud, M.: A multi-resolution approach to global ocean modeling, Ocean Modelling, 69, 211–232, 2013.

Schliephake, M. and Laure, E.: Performance Analysis of Irregular Collective Communication with the Crystal Router Algorithm, in: Solving Software Challenges for Exascale, edited by Markidis, S. and Laure, E., pp. 130–140, Springer International Publishing, Cham, 2015.

Shewchuk, J. R.: Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator, in: Applied computational geometry towards geometric engineering, pp. 203–222, Springer, 1996.

SIGMA toolkit: Scalable Interfaces for Geometry and Mesh based Applications (SIGMA) toolkit, http://sigma.mcs.anl.gov, http://sigma.mcs.anl.gov, 2014.

Slattery, S., Wilson, P., and Pawlowski, R.: The data transfer kit: a geometric rendezvous-based tool for multiphysics data transfer, in: International conference on mathematics & computational methods applied to nuclear science & engineering (M&C 2013), pp. 5–9, 2013.

Slattery, S. R.: Mesh-free data transfer algorithms for partitioned multiphysics problems: Conservation, accuracy, and parallelism, Journal of Computational Physics, 307, 164–188, 2016.

Slingo, J., Bates, K., Nikiforakis, N., Piggott, M., Roberts, M., Shaffrey, L., Stevens, I., Vidale, P. L., and Weller, H.: Developing the next-generation climate system models: challenges and achievements, Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 367, 815–831, 2009.

Tautges, T. J. and Caceres, A.: Scalable parallel solution coupling for multiphysics reactor simulation, Journal of Physics: Conference Series, 180, 2009.

Tautges, T. J., Meyers, R., Merkley, K., Stimpson, C., and Ernst, C.: MOAB: A Mesh-Oriented datABase, SAND2004-1592, Sandia National Laboratories, 2004.

Tautges, T. J., Kraftcheck, J., Bertram, N., Sachdeva, V., and Magerlein, J.: Mesh Interface Resolution and Ghost Exchange in a Parallel Mesh Representation, IEEE, Shanghai, China, 2012.

Taylor, M., Edwards, J., Thomas, S., and Nair, R.: A mass and energy conserving spectral element atmospheric dynamical core on the cubed-sphere grid, in: Journal of Physics: Conference Series, vol. 78, p. 012074, IOP Publishing, 2007.

Thomas, S. J. and Loft, R. D.: The NCAR spectral element climate dynamical core: Semi-implicit Eulerian formulation, Journal of Scientific Computing, 25, 307–322, 2005.

Ullrich, P. A. and Taylor, M. A.: Arbitrary-order conservative and consistent remapping and a theory of linear maps: Part I, Monthly Weather Review, 143, 2419–2440, 2015.

Ullrich, P. A., Lauritzen, P. H., and Jablonowski, C.: Geometrically Exact Conservative Remapping (GECoRe): Regular latitude–longitude and cubed-sphere grids, Monthly Weather Review, 137, 1721–1741, 2009.

Ullrich, P. A., Lauritzen, P. H., and Jablonowski, C.: Some considerations for high-order 'incremental remap'-based transport schemes: edges, reconstructions, and area integration, International Journal for Numerical Methods in Fluids, 71, 1131–1151, 2013.

Ullrich, P. A., Devendran, D., and Johansen, H.: Arbitrary-order conservative and consistent remapping and a theory of linear maps: Part II, Monthly Weather Review, 144, 1529–1549, 2016.

Valcke, S.: The OASIS3 coupler: a European climate modelling community software, Geoscientific Model Development, 6, 373–388, 2013.

VisIt: VisIt User's Guide, Tech. Rep. UCRL-SM-220449, Lawrence Livermore National Laboratory, 2005.

Wan, H., Giorgetta, M. A., Zängl, G., Restelli, M., Majewski, D., Bonaventura, L., Fröhlich, K., Reinert, D., Rípodas, P., Kornblueh, L., and Förstner, J.: The ICON-1.2 hydrostatic atmospheric dynamical core on triangular grids â€“ Part 1: Formulation and performance of the baseline version, Geoscientific Model Development, 6, 735–763, https://doi.org/10.5194/gmd-6-735-2013, https://www.geosci-model-dev.net/6/735/2013/, 2013.

Washington, W., Bader, D., and Collins, B, e. a.: Challenges in climate change science and the role of computing at the extreme scale, in: Proc. of the Workshop on Climate Science, 2008.

Zhou, S.: Coupling climate models with the earth system modeling framework and the common component architecture, Concurrency and Computation: Practice and Experience, 18, 203–213, 2006.

Zienkiewicz, O. C. and Zhu, J. Z.: The superconvergent patch recovery and a posteriori error estimates. Part 1: The recovery technique, International Journal for Numerical Methods in Engineering, 33, 1331–1364, 1992.

Government License