Final report for DE-SC0018275: Center for Tokamak Transient Simulations (RPI Unstructured Mesh Developments for FES SciDAC4 Partnerships)

Abstract

The goal of this project was the development of unstructured mesh technologies for fusion simulation codes" for FES SciDAC partnerships and to integrate those technologies into the simulation workflows of those partnerships. Specific developments were executed in support of the following FES SciDAC4 partnerships: Partnership Center for High-fidelity Boundary Plasma Simulation (HBPS), Center for Integrated Simulation of Fusion Relevant RF (RF-SciDAC), Center for Plasma Surface Interactions: Predicting the Performance and Impact of Dynamic PFC Surfaces (PSI2), and Center for Tokamak Transient Simulations (CTTS). The key unstructured mesh development areas addressed in this project include (i) methods to most effectively perform PIC calculations on unstructured meshes; (ii) creating meshes for fusion systems accounting for any desired level of geometric complexity and providing physics aware mesh configurations, (iii) adapting unstructured meshes to control the discretization errors, (iv) executing unstructured mesh calculations on GPU accelerated systems, (v) supporting physics/application-specific PIC operations including surface/wall interactions of particles, (vi) providing infrastructure tools to support the interactions of solvers with unstructured meshes, and (vii) providing advanced methods for coupling plasma physics codes.

Final report for DE-SC0018275: Center for Tokamak Transient Simulations (RPI Unstructured Mesh Developments for FES SciDAC4 Partnerships)

Major project goals

The goal of this project was the development of unstructured mesh technologies for fusion simulation codes" for FES SciDAC partnerships and to integrate those technologies into the simulation workflows of those partnerships. Specific developments were executed in support of the following FES SciDAC4 partnerships: Partnership Center for High-fidelity Boundary Plasma Simulation (HBPS), Center for Integrated Simulation of Fusion Relevant RF (RF-SciDAC), Center for Plasma Surface Interactions: Predicting the Performance and Impact of Dynamic PFC Surfaces (PSI2), and Center for Tokamak Transient Simulations (CTTS).

An increasing number of the high-performance fusion modeling simulation codes are taking advantage of the ability of unstructured mesh methods that can provide the required levels of accuracy using the fewest number of elements, often fewer by an order of magnitude or more. The advantages of unstructured meshes come with a cost of larger per element (cell) data structures, and more complex algorithms to achieve parallel scalability. Starting from a set of components for supporting unstructured meshes on parallel computers, the key unstructured mesh developments executed in this project addressed (i) methods to most effectively perform PIC calculations on unstructured meshes; (ii) creating meshes for full scale fusion systems accounting for any desired level of geometric complexity and providing physics aware mesh configurations, (iii) adapting unstructured meshes to control the discretization errors, (iv) executing unstructured mesh calculations on GPU accelerated systems including Frontier and

Aurora, (v) supporting physics/application-specific PIC operations including surface/wall interactions of particles, (vi) providing infrastructure tools to support the interactions of solvers with unstructured meshes, and (vii) providing advanced methods for coupling plasma physics codes.

A development area of importance to fusion applications is performing multiscale simulations in which particles model the fine scale behaviors and PDEs, discretized over unstructured meshes, represent coarser scale behaviors. A new mesh centric approach, referred to as PUMIPic, based on a new set of GPU friendly data structures and associated mesh/particle operators, was developed to support the execution of unstructured mesh PIC simulations. PUMIPic supports distributed meshes and has advantages for improving memory access patterns, performing geometric searches, and supporting dynamic load balancing.

A fully 3D mesh version of the PSI2 developed GITR impurity transport code, called GITRm, was developed building on PUMIPic. GITRm supports scalable and performant execution of impurity transport simulations using graded anisotropic 3D meshes that can be automatically generated in geometrically complex domains that fully represent key local components such divertors, antennas, tiles, and probes.

PUMIPic was also used in the development of a version of the HBPS developed XGC edge plasma code, called XGCm, which is fully in C++ and is focused on execution of all PIC/mesh interaction operations GPUs. XGCM demonstrated speed-up of from 21% to 300% depending on the physics included in the XGC simulation.

There are a number of unstructured mesh generation and mesh adaptation challenges faced by the SciDAC fusion partnerships. To address these challenges a number of both general purpose and specific purpose analysis geometry and mesh generations tools and workflows were developed. Procedures were developed to define complete analysis models of fusion plasma systems in which the analysis domains are combinations of detailed CAD models (e.g., antenna arrays, divertors, etc.), overall reactor wall, coupling interfaces (e.g., boundary for a coupled core plasma simulation), and physics geometry (e.g., flux surfaces/curves, separatrix surfaces/curves , X-points). Both general purpose and specialized mesh generation procedures were provided to support the generation of the meshes need by the analysis codes being developed by the SciDAC fusion partnerships. To support the RF-SciDAC advanced geometric modeling and meshing procedures were developed and integrated into the RF modeling and simulation codes. These same procedures are used, with specific extensions to couple to input fields, to generate the 3D meshes used in GITRm for PSI2. GITRm is now being used in DIII-D studies by General Atomics and West studies by OTNL. One special purpose mesh generation tool developed was the Tokamak Modeling and Meshing Software (ToMMS). ToMMS includes functions for creating the flux aligned, “field following” meshes required by the HBPS XGC code and the graded meshes needed by the CTTS M3D-C1 code. A second specialized mesh generation tool, MBBL, was developed to provided strongly graded, block structured, meshes for the PSI2 hPIC code.

Adaptive mesh control and analysis methods were developed to support both the PetraM/MFEM based RF wave plasma heating simulations for the RF-SciDAC and extend MHD simulations using M3D-C1 for CTTS. These efforts include the development of discretization error estimation procedures, tools to perform conforming curved mesh adaptation tools and integration of those tools into the appropriate analysis codes.

Efforts were also initiated near the end of the project on the development of generalized methods for the multiscale coupling of plasma physics codes for effective two-way code coupling.

The development of unstructured mesh technologies for FES SciDAC partnerships continues in SciDAC5 where the RPI team is part of the CEDA, RF, HifiStell and StellFoundry partnerships. These projects are addressing the unstructured mesh needs of both tokamak and stellarator systems.

All procedure developed in this project are open source (BSD) and maintained in GitHub repositories. Specific efforts were carried out to integrate the procedures with the specific SciDAC fusion partnership codes as well as installing and maintaining the software on the appropriate DOE and other computing systems.

The unstructured mesh tools developed as part of the project (with reference to their GitHubs) included are:

- GITRm [B]: PUMIPic based 3D unstructured mesh global impurity transport coder.
- MBBL [E]: Specialized mesh generation tool for hPIC and hPIC2 plasma wall material interaction codes.
- PUMIPic [H]: Parallel Unstructured Mesh Infrastructure for PIC calculations. A GPU enabled tool to support the data and operations required for unstructured mesh pic simulations.
- XCGm [L]: PUMIPic based edge plasma simulation code that supports a subset of the XGC physics.
- M3D-C1 mesh generation [D]: A tool to efficiently support the creation of M3D-C1 simulation domain geometries and specification of the desired mesh control information.
- PCMS [K]: Parallel Coupler for Multimodel Simulations. A generalized, GPU enabled, tool to support the effective coupling (1-way and 2-way) of plasma, and other, physics codes.

The unstructured mesh tools extended as part of this project (with reference to their GitHubs) included:

- EnGPar [A] and ParMa [J]: Tools for dynamic load balancing of partitioned meshes.
- M3D-C1 adaptive procedures [C]: Applied error estimation and mesh adaptation procedures for M3D-C1 simulations that were extended to support 2.5D mesh adaptation use in M3D-C1 3D simulations.
- Omega_h [F]: A GPU based infrastructure for the storage, manipulation and cavity-based mesh adaptation of unstructured meshes.
- PUMI [H]: Parallel Unstructured Mesh Infrastructure. A CPU based infrastructure for the storage, manipulation and cavity-based mesh adaptation of unstructured meshes.
- TOMMS [I]: Tokamak Modeling and Meshing System. Specialized mesh generation tool for XGC edge plasma code.

Summary of Project Accomplishments

Parallel Infrastructure for Unstructured Mesh PIC Calculations (PUMIPic)

Particle in cell (PIC) methods are used in multiple fusion plasma codes. The level of computation required for accurate PIC simulations of tokamak fusion systems can only be provided by massively parallel computing systems. Recent efforts have focused on the effective implementation of PIC calculations on GPU powered exascale computing systems. Standard PIC implementations use a distributed particle data structure and a spatial discretization in terms of a simple grid copied on all processes. As PIC codes started using unstructured meshes, they added a standalone mesh data structure and again copied the mesh on all processes. For this to work they also had to add linkages between particles and unstructured meshes that would require some level of local search in the execution of PIC operations. This limits the mesh scalability and does not support efficient implementations of graded unstructured meshes as are critical for cost effective solutions.

To support the ability to have both the mesh and particles distributed over multiple processes and the effective GPU execution of PIC operations on unstructured meshes, an alternative approach to support unstructured mesh PIC calculations has been developed [3] [H]. The implementation of codes based on this approach is supported by a library, PUMIPic, in which the mesh is the core data structure. The particles and mesh are directly related at all times. The unstructured mesh library used in PUMIPic, Omega h [F], maintains the mesh adjacencies needed by applications and supports a full range of mesh based operations. Omega h was specifically designed for effective execution of distributed mesh operations on GPU accelerated parallel computers. Both PUMIPic and Omega h are implemented using Kokkos to enable performance portability across multiple CPU and GPU architectures.

Since the number of particles per cell and their distribution throughout the domain varies based on the specific physics and PIC code being used, four alternative particle data structures have been developed. As discussed in reference [3], the first three stored the particles on a per element basis and included CSR, SCS and CabanaM. The particle data structures and mesh infrastructure, Omega_h are implemented using Kokkos. The effective use of PUMIPic in XGCm demonstrated substantial performance gains for supporting ion push operations where the number of cells a particle moved per time step. However, in the case of global impurity transport simulations executed with GITRm, the relative benefits of PUMIPic were less since the average numbers of particle per element is much less and the distribution of particles throughout the domain is highly nonuniform and evolves over time. To deal with the evolving nonuniform particle distributions, a dynamic load balancing procedure was developed [6]. To deal with the combination of non-uniform particle distributions and few particles per element on average, the fourth data structure, DPS, was introduced.

DPS, the disorganized particle structure, maintains the association of each particle to the element it exists within, but it does not group particles in memory by element. Consequentially, the memory accesses for each element to its particles is not coalesced on GPUs, and thus DPS sacrifices data access performance relative to CSR, SCS and CabanaM. But, the cost to maintain the particle-to-element association only requires writing a single integer, the element identifier, per particle as opposed to the coarse grained sorting operation and subsequent memory movement needed for CSR, SCS and CabanaM. Based on the relative advantages of the two overall approaches, DPS is more performant when the number of particles per element is low. In GITRm, the impurity particles only occupy a fraction of the domain at any given time with no particles in most elements and other elements with a maximum of tens of particles. In such a case, savings in maintaining the particle-to-element association outweighs the loss in performance during Particle Push and other operations. In GITRm, several cases have been carried out to compare computational performance between DPS and SCS options. Using the DPS option in GITRm cut the calculation time in half when compared to the SCS option.

Dynamic load balancing of particles is another crucial component to maintain computational efficiency for certain applications. For example, in GITRm impurity particle density varies significantly over the mesh at any given instance of time and it changes dramatically in time as the simulation progresses. To effectively account for spatiotemporal density variation of impurity particles, we apply dynamic load balancing. Our current work has focused on a fixed mesh partition and uses particle load balancing such that particles are re-distributed/migrated over the partitioned mesh, which was found to be sufficient in the current applications. The effective implementation of particle load balancing/migration step is supported by the inclusion of buffer parts in PUMIPic [37]. For the DIII-D case [6], a particle imbalance of around 300% is observed without load balancing, while with the particle load balancing, it is reduced to about 10% and is maintained at that level during the entire simulation.

In GITRm, the sheath electric field must also be accurately taken into account. The sheath electric field is a strong electric field with an exponential variation like a boundary-layer feature that forms near a plasma-facing material surface. It resides within only a few near-wall elements in the GITRm mesh, however, it plays a significant role in impurity particle trajectories as they erode from the surface or impact the surface. The sheath region is currently modeled using an analytic model, e.g., Chodura sheath based Brooks model. Note that the evaluation of the sheath electric field using such a model is computationally expensive since it requires the determination of the closest distance to a boundary for each particle in every push operation/step. GITRm addresses this need by performing a pre-processing step that stores a list of mesh faces residing on the surface/boundary that are closest to a volumetric mesh element and in the course of the simulation, the distance to boundary calculation between a particle and the nearest surface is limited to this relatively small subset of boundary mesh faces. Furthermore, it is only done once for a given mesh and only for those elements that are within a specified distance from the relevant surfaces around which the sheath electric field is dominant. To construct such a list of boundary mesh faces for each relevant mesh element (in the sheath region), we exploit topological adjacency information for mesh entities and association of the mesh entities to the geometric model entities, which is supported in Omega h. The pre-processing step involves a marching method that starts from the relevant boundaries of interest and for each mesh face on relevant boundaries it performs element walks away from the boundary by using mesh adjacency up to the specified distance. For any element in such a walk, the starting mesh boundary face is included in the pre-processed list.

PUMIPic based Impurity Transport Code (GITRm)

A completely new GPU-accelerated code, GITRm, was developed for global impurity transport in magnetically confined fusion devices/tokamaks [6][7]. It is built on top of the PUMIPic library and uses fully 3D unstructured mesh-based particle tracking procedures. It is designed to target complex geometries including non-axisymmetric local features such as bumpers, probes, limiters, antennas, etc., and uses strongly graded and anisotropic elements to accurately represent the plasma fields. It is a high-performance Monte Carlo particle (neutral atom and ion) tracking code, based on the trace approximation. It can simulate surface erosion (due to particle-surface/wall interactions), ionization, migration, and redistribution of impurity material emanating from plasma-facing components in fusion devices. It has been extended to account for multiple impurity species in a consistent fashion in a single simulation [7]. It has also been extended to incorporate the multi-scale near-wall sheath electric field that is computed by a sheath simulator such as the hPIC2 code. It operates on distributed meshes and is performant on multi GPU-accelerated computer systems. It has been used to perform fully 3D impurity transport simulations on several tokamaks, such as ITER, DIII-D and WEST, with billion of particles on hundreds of GPUs for physically-relevant problem cases.

GITRm is designed to be used by domain scientists for fusion devices involving fully 3D geometric features and on GPU-accelerated supercomputers. GITRm workflow has been well documented and shared with several domain scientists including those from ORNL and General Atomics (GA), and it has also been ported on several computer systems used by these scientists [7]. These interactions have enabled simulations and predictions that was previously extremely time consuming (e.g., due to geometric complexity) or even intractable (e.g., due to the computational scale/size of the problem). Specifically, GITRm has enabled GA researchers to perform fully 3D impurity transport simulations for the DIII-D device with collector probes and V-shaped small-angle slot (SAS-V) divertor including any tile misalignment that occurs during installation. Similarly, GITRm has enabled ORNL researchers to perform fully 3D simulations for the WEST device with detailed antenna geometry (with Faraday grids) and including impurity erosion from limiters.

PUMIPic based Edge Plasma Code (XGCm)

The XGCm code [17] is written in C++, employs PUMIPic structures and XGC [1] physics algorithms and runs on GPU powered supercomputers platforms including Perlmutter and Summit. XGCm has three levels of mesh partitions including: (i) toroidal direction partition along the Tokamak torus direction; (ii) within the poloidal plane the unstructured mesh is partitioned according to the flux curves; (iii) mesh partition within each poloidal plane for solving the gyro-kinetic Poisson equation.

Main component operations of the gyro-kinetic PIC method within XGCm, are performed on the GPU using a distributed mesh. These operations include:

- Ion and electron charge scatter processes (particle to mesh operation) are finished and performed on GPU.
- Electrostatic potential calculation (gyro-kinetic Poisson equation solve) performed on GPU with PETSc.
- Gyro-kinetic electric field calculation on unstructured mesh (gradient of electrostatic potential) performed on GPU with distributed mesh.
- Gyro-kinetic electric field gather operation (mesh to particle field operation, for both ion and electron) on GPU.
- Ion and electron push are performed on GPU, with electron sub-cycling capability. Particle initialization with perturbation field is performed on GPU.

Code verification has been performed. The Cyclone Ion Temperature Gradient (ITG) case serves as a benchmark for XGC overall operations and physics. Additional code verification has also been performed using the DIII-D tokamak geometry.

Weak scaling results using Summit computer have been obtained. In the weak scaling study, a distributed mesh is used with 48 MPI ranks corresponding to each poloidal plane. Each MPI rank corresponds to 1 GPU. At the minimum, 8 poloidal planes are used, corresponding to a total of 384 GPUs, or 64 Summit nodes. By increasing the toroidal mesh resolution, i.e. using more poloidal planes, we maintain a fixed amount of mesh size and simulation particles per GPU. In the largest run 2048 nodes, corresponding to 12288 GPUs, were used. Overall, good weak scaling is achieved.

Performance comparisons between XGCm and a version of XGC were carried out. For an adiabatic electron case XGCm took only 35% of the time of XGC while in the kinetic electron case XGCm took only 80% of the time of XGC. The performance improvement gained to date for the kinetic electron case are limited since XGCm is not yet realizing an advantage during the electron sub-cycling step that both XGC and XGCm is using. Investigation is currently underway to determine if the PUMIPic capabilities can be used to fuller advantage during electron subcysling.

Generation and Meshing of Analysis Geometries for Fusion Plasma Simulations

As overviewed in reference [11] the RPI team has developed a broad range of unstructured mesh tools to support FES SciDAC partnerships. These developments include:

- Specialized mesh generation tools meeting the needs of the edge plasma code XGC [10],[I], the extended MHD plasma code M3D-C1 [D], and the plasma-material interaction code hPIC2 [8][E].
- Analysis geometry construction and mesh generation workflows based on combinations of open source and commercial software tools for the impurity transport code GITRm [6] and RF code PetraM/MFEM [12].
- Development of adaptive simulation workflows for the RF code PetraM/MFEM [12] and the extended MHD plasma code M3D-C1 [C]
- Core unstructured mesh tools including parallel adaptive mesh infrastructures for CPUs [G] and GPUs [F] and parallel dynamic load balancing tools [13][J].[D]

In the case of M3D-C1, XCG and XGCm, the analysis geometric models sent to the automatic mesh generators are limited to 2D poloidal plane cross sections. The M3D-C1 3D mesh is created by extruding the poloidal plane mesh in the toroidal directions and in XGC and XGCm the poloidal plane mesh is replicated on a selected number of poloidal planes with finite difference methods applied to discretize the physics equations between poloidal planes.

To support the users of M3D-C1, the input to the geometry construction tool can include any combination of (i) the parameters of the analytic curve of the bounding the vacuum region, (ii) a discrete geometry loop of the wall, (iii) a thickness parameter if a finite thickness wall will be represented in the model, and (iv) discrete geometry of any other model entities to be represented (e.g. flux curves, magnet cross sections, etc.). The user also identifies the model faces to be meshed in terms of the set of loops that bound the face. The analysis geometry construction tool processes the discrete loops to define straight line and spline analysis model edges, and, when a finite thickness wall is to be represented, employs a geometry expansion procedure to define the outer wall loop model edges. The user also has the ability to specify mesh control information on the geometric model loops and faces[D].

XGC and XGCm have a set of analysis geometric model definition requirements dictated by the need to construct near-field following meshes. The analysis geometric model consists of an initial model, that is later modified to define the model provided to the mesh generation procedures [10][I]. The initial analysis model geometry includes the tokamak wall and a set of "physics geometry" in terms of critical points and flux curves extracted from the background flux field defined at a set of grid points that encompass the tokamak wall geometry. Due to the existence of conflicting meshing requirements associated with plasma physics simulation meshes, selected model faces need to be split into multiple faces where a mesh generation attribute is applied to each model face to indicate the specific triangulation algorithm to be applied to that face [D].

XGC and XGCm edge plasma codes have specific mesh configuration requirements that must be satisfied. The first requirement is mesh vertices be place on flux curves such that near field-following tracking can be exercised on the mesh. The second requirement is that there are no mesh vertices placed between flux curves for all closed flux curves and between pairs of open flux curves except around the separatrix and in areas near where open flux curves intersect the tokamak wall. In those areas, the continued use of the no mesh vertices between flux curves would create situations where a small number of mesh faces with areas much larger than the neighboring elements, and anisotropic mesh entities with the long mesh edges being in the "wrong direction", would be created. The solution to producing a satisfactory mesh required the isolation of the problems areas and application of a mesh doubling procedure to reduce mesh anisotropy, plus the application of isotropic triangulation to the remaining area [10]. The application of these additional meshing procedures is facilitated by the process of splitting the model faces between the flux curves in the problem areas with the addition of two new model faces and assigning an attribute indicating the mesh generation operations to be used for that face. The first face is a thin transition model face that is meshed with an element doubling procedure that reduces element anisotropy. The second model face is the remaining area to either the wall for open flux curves, or areas around an X-point, which are meshed with an isotropic triangulation procedure.

The RPI team has developed a novel meshing and particle tracking strategy for the hPIC2 code [8][E]. hPIC2 is a hardware-accelerated, hybrid particle-in-cell code for dynamic plasma-material interactions. The current strategy employs block-structured nonuniform meshes for multi-scale plasma simulations of plasma sheaths and scrape-off layers. The current strategy allows a flexible block arrangement and gradation technique to account for different physics regimes in different regions of the computational domain, e.g., in a sheath-to-presheath transition region exponential mesh gradation is used to go from a small element size (in sheath) to a very large element size (in bulk plasma). In addition, this strategy uses a highly efficient two-level particle tracking strategy (including within an exponentially graded block) and results in very little overheads as compared to (trivial) particle tracking on a uniform mesh. This strategy has been successfully used in hybrid Particle-in-Cell (PIC) schemes that considers kinetic ions and Boltzmann electrons for large-scale fusion plasma domains (spanning tens of meters). This strategy has resulted in reduction of degrees-of-freedom by order of magnitude as compared to a uniform mesh (which was only option available previously), while maintaining the same level of accuracy and keeping particle noise under control. The new developments have been used to perform one-of-its-kind sheath-resolved simulations in a poloidal cross-section of the ITER device. In summary, the new developments enable cost-effective, multi-scale fusion plasma simulations that were previously intractable.

Petra-M and GITRm employ fully 3D unstructured meshes where the analysis model geometries are general combinations of physical and physics model components. The physical model components are imported from CAD manufacturing models, defined in terms of parameterized geometries, and/or constructed from 2D planar information, as discussed above, and extruded in the toroidal direction. The physics geometry includes selected flux surfaces and "critical point" flux curves. A set of geometric modeling components developed to support simulation-based engineering simulation workflows are applied in the construction of the analysis geometry models for Petra-M and GITRm. The steps executed in the construction of the analysis geometry model for tokamak geometries include:

- Importing analysis model components defined in terms of a set of discrete line segments and/or surface triangles and processing them to define appropriate CAD model entities.
- Importing the background flux field defined over a grid and processing it to find the extreme points, define the separatrix curves and define specified flux curves. Extruding the constructed flux and separatrix curves to create the associated model surfaces.
- Importing CAD manufacturing models of components to be included in the analysis model and processing those models to eliminate small artifacts and undesired model features.
- Combining the model entities as needed to create the 3D non-manifold analysis geometric model.

It has been demonstrated that optimally implemented high-order finite elements are well suited for application in RF simulations of tokamaks. When applying high-order finite element methods, ensuring solution convergence requires using curved elements that maintain an order of geometric approximation to the curved domain boundaries that is dictated by the order of finite element basis used. Although theory indicates the geometric approximation order required to ensure convergence in the energy norm is one less than the basis order of the finite elements used, numerical studies of the convergence of selected quantities of interest to RF antenna designers indicate that the geometric approximation must be of the same order as the finite element basis being used. Although the meshes generated for accurate RF tokamak simulations have several million elements, even when using fifth order finite elements, the size of individual elements on the smaller curved model entities is large compared to the size of the model entity. The procedure used to generate the curved meshes employs an automatic mesh generator capable of generating quality quadratic geometry meshes. When higher than quadratic finite elements are used for the analysis, the mesh faces and edges on the curved model entities have their order of geometric approximation improved to the order of the finite elements. In some cases, increasing the order of geometric approximation degrades the element shapes and can create invalid elements. In those cases, local curved mesh modification procedures are applied to create acceptably shaped elements.

In the case of coupled simulations, the accurate transfer of solution fields between two mesh-based simulations requires the mesh resolution over the coupled portion of the domain be appropriately matched. When using the same order elements in both simulations, this requirement can be effectively met by matching the mesh resolution and anisotropy between the coupled meshes. In the case of the impurity transport code GITRm [9], background plasma profiles, such as density and temperature, are used by the physics operators that move the particles. Although a fully 3D mesh is used in GITRm to span the 3D domain of the plasma in the tokamak, the background plasma profiles are currently defined based on 2D axisymmetric field solvers. In particular, these are defined from edge-plasma codes, such as SOLPS and OEDGE, that employ a 2D block structured anisotropic mesh in a portion of the tokamak represented as a 3D surface within the volume of the GITRm domain and is used to generate highly graded, anisotropic elements as layered stacks in a local neighborhood. This is done to capture the sharp directional gradients in the plasma fields. Away from the layered stacks, unstructured (isotropic) tetrahedral elements are used in the GITRm mesh.

Adapting Unstructured Meshes Methods

When simulating complex physics over general geometries, it is not possible for the scientist or engineer performing the simulation to know the optimal mesh distribution to obtain the results to the desired levels of accuracy. Although methods to ensure a given level of accuracy in the physical quantities of interest remains somewhat allusive, methods to construct near optimal meshes based on the use of a posteriori error estimators and/or correction indicators are well known. As an example of the usefulness of these methods, consider a time dependent simulation of the injection of an impurity pellet into a plasma as executed with the M3D-C1 extended MHD code. In this simulation, the mesh needs to be substantially more refined in areas where the impurities are concentrated as they move from the boundary toward the O-point. When this problem was solved using the adaptive mesh control procedures developed in this project, the adaptively defined meshes contained two orders fewer elements than a uniform mesh providing the same level of accuracy.

Two key challenges in the development of adaptive mesh control procedures for PetraM/MFEM simulations are the development of effective error indicators for Nedelec finite elements and performing conforming curved mesh adaptation on coarse curved meshes. Reference [12] presents a patch recovery type error indicated designed for Nedelec elements.

The curve conforming mesh adaptation procedures used to adapt the high order curved meshes employ recently extended curved element mesh cavity remeshing procedures. The primary developments carried out included extending the collapse, swap and split operators to support high that quadratic interior mesh entity geometry. Recent enhancements to the collapse and split operators support effective cubic interior mesh entity geometry definition. Cubic interior mesh entities can support a change in sign of curvature which has been found to provide substantial flexibility in defining satisfactory element shapes. Due to the high cost of using higher order interior mesh entity shapes, the mesh modification procedures employ mesh cavity properties to attempt construction of quadratic and then cubic interior mesh entities. If those operations fail, the interior element entity order will be raised to the order of the finite element basis and quality improvement techniques will be applied at the cavity level.

Parallel Coupler for Multimodel Simulations (PCMS)

A number of tools have been, or are being, developed to support the transfer of physics field information between simulation codes. None of those tools meet the functional requirements and/or accuracy needs of the mesh-to-mesh physics fields transfers required by fusion plasma simulation codes and the codes to which they need to couple. To meet the needs of mesh-to-mesh solution transfer, the Parallel Coupler for Multimodel Simulations (PCMS) is being developed [14], [9], [K]. The combination of three aspects, each critical for coupling of fusion reactor codes, set PCMS apart: First, integration of coordinate and data transformations. This is critical for two reasons, most plasma codes use unique forms of field aligned, or physics based coordinate systems and many store field data in the frequency domain requiring extra processing if coupling is to be done in the spatial domain. Second, parallel control and transfer algorithms that account for details of mesh-based fields by allowing the use, or implementation of, native interrogation methods (such as interpolation) wherever possible. And, third a design that avoids intrusive code changes and makes use of existing data-structures.

The primary goal of PCMS is to make it as easy as possible for applied mathematicians and physicists to test a wide range of schemes without resorting to ad-hoc modifications of the coupled physics codes.

When using PCMS applications perform their operations on their unmodified data structures. PCMS interfaces are used to describe the data layout across the set of application ranks. Data is prepared using an adapter class that performs serialization, and deserialization operations. Additionally, field layout metadata is exchanged for construction of rank-to-rank mappings as needed for parallel control. All data and meta-data are sent over the network using ADIOS2 [14] which can be used with in-memory or file-based transport.

Once data is sent to the coupler, coordinate and data transformations are performed so that fields can be operated on in an intermediate representation. This helps reduce the burden of adding new coupling pairs since each new application only needs to develop transformation capabilities to a single intermediate representation. Although the coupler is shown as an independent utility, requiring extra communications, in practice it need not be an independent application. By maintaining separate application and coupler interfaces we do not need any of the components to share compilers or toolchains. This has been particularly helpful while running on leadership-class resources where toolchains are constantly evolving and compiler and hardware bugs abound that require application specific toolchain workarounds.

In general, constructing process-to-process communication maps in a scalable way is difficult. PCMS extends the rendezvous algorithm to work with geometric model classification, and/or mesh based partitioning such as recursive coordinate bisection (RCB) and graph-based methods. The basic idea of the rendezvous algorithm is to use a secondary process mapping to a set of rendezvous processes that each application can compute independently. Then, the true application-to-application process mapping can be constructed on the set of rendezvous processes and communicated back to the applications. We have developed mesh databases and model setup libraries that maintain a tight link to the analysis CAD models through geometric classifications.

Coordinate and data transformations are key to coupling plasma codes. In PCMS, we encode these transformations into data strong-types to provide a type-safe interface for programming new coupling schemes. We found that a traditional view on coordinate transformations requiring user input of a Jacobian worked well when all applications stored their data in the space domain. However, it is common for plasma codes to store data in the frequency domain. By using a spatial Jacobian we had to employ a two step procedure to first, transform the data from the frequency domain to the spatial domain, then perform coordinate transformation through the coordinate Jacobian. This was particularly inefficient because conversion from the frequency to spatial domain along the magnetic field lines requires a high mesh density. Also, it’s typically possible to write the transformation as a single step from the frequency domain of the first coordinate system to the spatial domain of the second. This single step processing is more computationally and memory efficient, as well as being more accurate. Therefore, instead of performing transformations through Jacobians, we allow the user to describe the transformation through a stateful (to allow caching) transformation object. Transformation objects can be used to perform all required coordinate and data transformations and can combine multiple operations through composition.

The PCMS field transfer operations efficiently execute in parallel on CPUs and GPUs employing generalized unstructured mesh representations and will shortly support specific structured mesh representations (typically block structured field-following mesh commonly employed by plasma simulation codes).

PCMS runs on a wide range of hardware from laptop computers to leadership-class supercomputers which make use of CPUs and GPUs from a variety of chip manufactuers (Intel, AMD, NVIDIA). To obtain reasonable performance, in- memory code coupling is critical. To enable this, PCMS uses various software packages such as Kokkos for performance portability, and Adios2 to enable efficient in-memory communication across applications.

The utility of PCMS has been demonstrated in a dynamic coupling framework "Benesh" to support its unstructured mesh capabilities. In particular, a gryrokinetic total-f to gyrokinetic delta-f coupling scheme was demonstrated at scale on the Summit, and Frontier supercomputers at the Oak Ridge Leadership Computing Facility (OLCF) [2].

Repository Links to Unstructured Mesh Tools and Components

[A] EnGPar: https://github.com/SCOREC/EnGPar
[B] GITRm: <https://github.com/SCOREC/gitrm>
[C] M3D-C1 mesh infrastructure and mesh adapation: https://github.com/PrincetonUniversity/M3DC1 see: PrincetonUniversity/M3DC1/meshgen - mesh generation and PrincetonUniversity/M3DC1/m3dc1_scorec
[D] M3D-C1 mesh generation: https://github.com/PrincetonUniversity/M3DC1 see PrincetonUniversity/M3DC1/meshgen - mesh generation
[E] MBBL: https://github.com/SCOREC/pumiMBBL
[F] Omega_h: https://github.com/SCOREC/omega_h
[G] PUMI: https://github.com/SCOREC/core
[H] PUMIPic: https://github.com/SCOREC/pumi-pic
[I] TOMMS: https://github.com/SCOREC/tomms
[J] ParMA: https://github.com/SCOREC/core/tree/master/parma
[K] PCMS: https://github.com/SCOREC/pcms
[L] XGCm: https://github.com/SCOREC/xgcm

Project Journal Papers

[1] E. D'Azevedo, S. Abbott, T. Koskela, P. Worley, S. Ku, S. Ethier, E. Yoon, M. Shephard, R. Hager, J. Lang, J. Choi, N. Podhorszki, S. Klasky, M. Parashar, C. Chang, The fusion code XGC: Enabling kinetic study of multiscale edge turbulent transport in ITER, *Exascale Scientific Applications: Scalability and Performance Portability*, CRC Press, 2017, Ch. 24, pp. 529–552, 2018.

[2] Davis, P.E., Merson, J., Subedi, P., Ricketson, L., Smith, C.W., Shephard, M.S. and Parashar, M., 2023, Dec.. Benesh: a Framework for Choreographic Coordination of In Situ Workflows. *2023 IEEE 30th Int. Conf. High Performance Computing, Data, and Analytics (HiPC)* (pp. 298-308). IEEE.

[3] Diamond, G., Smith, C.W., Zhang, C., Yoon, E. and Shephard, M.S., 2021. PUMIPic: A mesh-based approach to unstructured mesh Particle-In-Cell on GPUs. *J. Parallel and Dist. Comp.*, *157*, pp.1-12.

[4] Granzow, B.N., Oberai, A.A. and Shephard, M.S., 2020. An automated approach for parallel adjoint-based error estimation and mesh adaptation. *Engineering with Computers*, *36*, pp.1169-1188.

[5] Kolev, T., Fischer, P., Min, M., Dongarra, J., Brown, J., Dobrev, V., Warburton, T., Tomov, S., Shephard, M.S., Abdelfattah, A., Barra, V., and others, 2021. Efficient exascale discretizations: High-order finite element methods. *Int. J. High Performance Computing Applications*, *35*(6), pp.527-552.

[6] D.D. Nath, V.V. Srinivasaragavan, T.R. Younkin, G. Diamond, C.W. Smith, A. Hayes, M.S. Shephard, O.Sahni, An unstructured mesh 3D pic code for impurity transport simulation in fusion tokamaks, *Computer Physics Communications*, Vol. 292, 2023.

[7] D.D. Nath, T.R. Younkin, J. Guterl, A. Kumar, M.S. Shephard, O.Sahni, GITRm: A high-performance code for simulation of multi-species impurity transport and surface interaction in SOL of tokamaks, submitted 2024.

[8] Meredith, L.T., Rezazadeh, M., Huq, M.F., Drobny, J., Srinivasaragavan, V.V., Sahni, O. and Curreli, D., 2023. hPIC2: A hardware-accelerated, hybrid particle-in-cell code for dynamic plasma-material interactions. *Computer Physics Communications*, *283*, p.108569.

[9] Merson, J. and Shephard, M.S., 2021. model-traits: Model attribute definitions for scientific simulations in C++. *Journal of Open Source Software*, *6*(64), p.3389.

[10] Riaz, U., Seol, E.S., Hager, R. and Shephard, M.S., 2024. Modeling and meshing for tokamak edge plasma simulations. *Computer Physics Communications*, *295*, p.108982.

[11] M.S. Shephard, J. Merson, O. Sahni, A.E. Castillo, A.Y. Josh, D.D. Nath, U. Riaz, E.S. Seol, C.W. Smith, C. Zhang, MW. Beall, O. Klaas, R. Nastasia, and S. Tendulkar. 2024, Unstructured Mesh Tools for Magnetically Confined Fusion System Simulations, *Engineering with Computers*, accepted, 2024.

[12] Siboni, M.H. and Shephard, M.S., 2022. Adaptive workflow for simulation of RF heaters. *Computer Physics Communications*, *279*, p.108434.

[13] Smith, C.W., Rasquin, M., Ibanez, D., Jansen, K.E. and Shephard, M.S., 2018. Improving unstructured mesh partitions for multiple criteria using mesh adjacencies. *SIAM J. Scientific Computing*, *40*(1), pp.C47-C75.

[14] Suchyta, E., Klasky, S., Podhorszki, N., Wolf, M., Adesoji, A., Chang, C.S., Choi, J., Davis, P.E., Dominski, J., Ethier, S., Foster, I., Smith, C.W., Shephard, M.S., and others, 2022. The Exascale Framework for High Fidelity coupled Simulations (EFFIS): Enabling whole device modeling in fusion science. *Int. J. High Performance Computing Applications*, *36*(1), pp.106-128.

[15] Tendulkar, S., Yang, F., Nastasia, R., Beall, M.W., Oberai, A.A., Shephard, M.S. and Sahni, O., 2022. Geometry and Adaptive Mesh Update Procedures for Ballistics Simulations. *Mesh Generation and Adaptation* (pp. 209-231). Springer, Cham., 2022.

[16] Yang, F., Chandra, A., Zhang, Y., Tendulkar, S., Nastasia, R., Oberai, A.A., Shephard, M.S. and Sahni, O., 2022. A parallel interface tracking approach for evolving geometry problems. *Engineering with Computers*, *38*(5), pp.4289-4305.

[17] Zhang, C., Diamond, G., Smith, C.W. and Shephard, M.S., Development of an unstructured mesh gyrokinetic particle-in-cell code for exascale fusion plasma simulations on GPUs, *Computer Physics Communications*, Vol. 291, 2023.