Creating Interoperable Meshing and Discretization Software: The Terascale Simulation Tools and Technology Center

David Brown
Lawrence Livermore National Laboratory
Livermore, CA 94550 USA
dlb@llnl.gov

Lori Freitag
Argonne National Laboratory
Argonne, IL 60490 USA
freitag@mcs.anl.gov

James Glimm
SUNY Stony Brook and Brookhaven National Laboratory
Stony Brook, NY 11794 USA
glimm@ams.sunysb.edu

Abstract

We present an overview of the technical objectives of the Terascale Simulation Tools and Technologies center. The primary goal of this multi-institution collaboration is to develop technologies that enable application scientists to easily use multiple mesh and discretization strategies within a single simulation on terascale computers. The discussion focuses on our efforts to create interoperable mesh generation tools, high-order discretization techniques, and adaptive meshing strategies.

Introduction

Terascale computing provides an unprecedented opportunity to achieve numerical simulations at levels of detail and accuracy previously unattainable. Scientists in many different application areas can reach new levels of understanding through the use of high-fidelity calculations based on multiple coupled physical processes and multiple interacting physical scales. Adaptive, composite, and hybrid approaches offer powerful methodologies for achieving results for these types of PDE-based simulations.

In today's environment many tools are available that generate a variety of mesh types ranging from unstructured meshes to overlapping structured meshes and hybrid meshes that include both structured and unstructured

components. Approximation techniques used on these meshes include finite difference, finite volume, finite element, spectral element, and discontinuous Galerkin methods. Any combination of these mesh and approximation types may be used to solve PDE-based problems. The fundamental concepts are the same for all approaches: some discrete representation of the geometry (the mesh) is used to approximate the physical domain, and some discretization procedure is used to represent approximate solutions and differential operators on the mesh. In addition, the concepts of adaptive mesh refinement for local resolution enhancement, time-varying meshes to represent moving geometry, data transfer between different meshes, and parallel decomposition of the mesh for computation on advanced computers are the same regardless of their implementation. In each case, the software tools providing these advanced capabilities are becoming increasingly accepted by the scientific community, but their application interfaces are not compatible. Thus, interchanging technology is often a labor intensive and error prone code modification process that must be endured by the application scientist. This typically results in a lengthy diversion from the central scientific investigation and severely inhibits experimentation with improved mesh and discretization technologies.

The Terascale Simulation Tools and Technologies (TSTT) center was recently funded by the DOE Scientific Discovery through Advanced Computing (SciDAC) Program [7] to address the technical and human barriers preventing the effective use of powerful adaptive, composite, and hybrid methods. The TSTT center brings together expertise from eight institutions in mesh generation, adaptive technologies, high-order discretization techniques, and terascale computing through projects such as CUBIT [4], NWGrid [9], Overture [2], and Trellis [1] (see Table 1 for a list of the principal investigators and their institutions). The pervading theme of the TSTT center is the development of interoperable and interchangeable meshing and discretization software. We are formulating a broad, comprehensive design that encompasses many aspects of the meshing and discretization process, but we are working toward that goal through incremental insertions of existing and newly developed technologies into targeted applications. Our efforts focus on three primary areas: advanced meshing technologies (see Section 2), high-order discretization techniques (see Section 3), and terascale computing issues such as dynamic load balancing and single processor performance optimization (not discussed in this paper because of space constraints).

Name	Institution	Email
David Brown	LLNL	dlb@llnl.gov
Ed D'Azevedo	ORNL	e6d@ornl.gov
Joe Flaherty	RPI	flaherje@cs.rpi.edu
Lori Freitag	ANL	freitag@mcs.anl.gov
Jim Glimm	SUNY Stony Brook, BNL	glimm@ams.sunysb.edu
Patrick Knupp	SNL	pknupp@sandia.gov
Xiaolin Li	SUNY Stony Brook	linlin@ams.sunysb.edu
Mark Shephard	RPI	shephard@scorec.rpi.edu
Harold Trease	PNNL	het@pnl.gov

Table 1. The principal investigators of the TSTT center

Advanced Meshing Technologies

Our current emphasis in the area of advanced meshing technologies is the development of techniques that allow existing TSTT Center technologies to interoperate or work with each other. To achieve this goal, we are creating common interfaces for accessing existing tools and will develop new capabilities as needed within these tools to provide compatible functionality, ensure mesh quality, and support complex geometries, high-order discretization techniques, and adaptive methods.

The first step in creating interoperable meshing technologies is to recognize that the underlying tools must take into account a conceptual hierarchy of domain representations (see Figure 1). The original problem specification can be given in terms of a high-level geometric representation such as a CAD description, image data, or possibly a previous mesh, along with the physical attributes of the application (level 0). This geometry has a global representation of some kind (such as a nonmanifold solid model or pixel intensity information) that properly represents the physical domain. The geometric model can be decomposed into subregions (level 1), each of which can be discretized by using a possibly different fundamental mesh type. Because the simulations will be run on terascale parallel computers, a further decomposition (level 3) is defined to support the partitioning of the levels 0-2 entities across the thousands of processors involved. All the levels shown in Figure 1 can be considered different resolutions of the domain and fit into a hierarchy of representations.

Analogous to the geometry hierarchy, we also use the concept of a mesh data hierarchy (Figure 2). The highest level in this hierarchy (level A) again contains a description of the geometric domain. Access to this information is provided, for example, by generic interfaces such as the Common Geometry Module (CGM) [8], direct functional interfaces to solid modeling

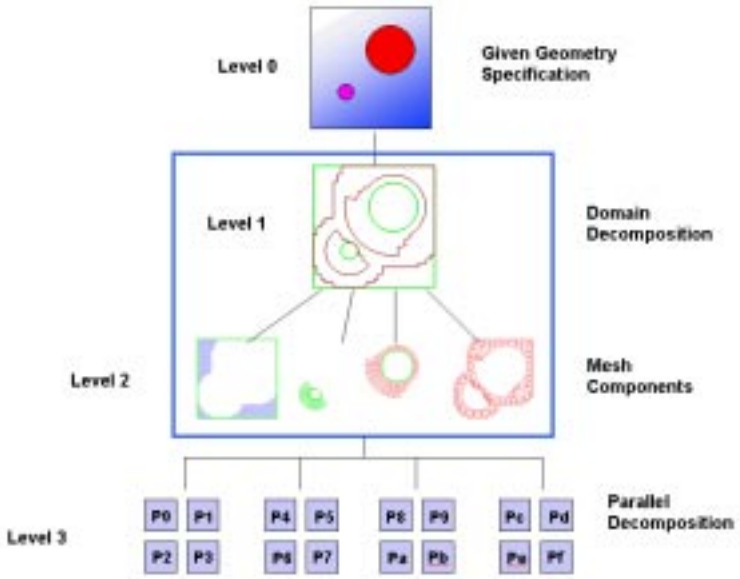

Figure 1. The geometry hierarchy

kernels, or data file representations such as IGES and STEP. The lowest level (C) shown in the figure again represents the mesh components, while the intermediate level (B) represents the combination of the level C mesh components into an overall hybrid mesh, together with the communication mechanisms that couple their data. We note that this hierarchy allows the possibility of multiple mesh representations (level B) for the complete geometry to be used within a single application or in coupled applications.

These hierarchical descriptions will be constructed by leveraging existing TSTT center tools. In particular, we will use graphically based tools available in Overture and CGM to partition a complex geometry into level 1 subregions that can be meshed by using TSTT mesh generators. A key new area of research will be the coupling of the level C submeshes to provide a complete discretization of the computational geometry at level B; for this purpose we will develop tools similar to the automatic overlap and hybrid mesh-stitching algorithms currently provided by Overture. The mesh structure and data hierarchy must also facilitate the needs of the discretization and solver procedures. For example, discretization procedures will employ information from levels 1 and 2 (Figure 1) to control the transfer of information.

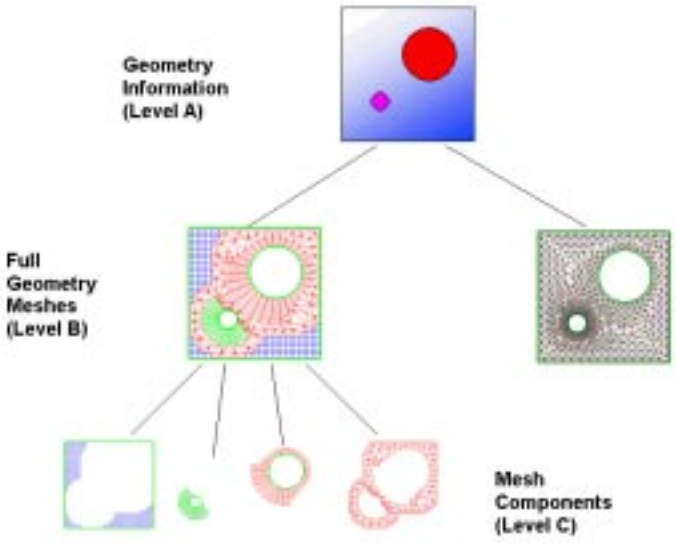

Figure 2. The mesh data hierarchy

To accomplish our interoperable software goal, we are defining common interfaces for accessing data at each level of the hierarchy shown in Figure 2. The interface design will be driven by application requirements and the need for intuitive, easy-to-use interfaces at multiple levels of sophistication. Thus, we will provide both high-level abstractions (e.g., representations of an entire complex mesh structure, and operations on that mesh) appropriate for new application development and low-level access functions (e.g., approximations of derivatives at a single point on a mesh) appropriate for incremental insertion of new technologies into existing applications. Initially we are focusing on query interfaces for accessing information pertaining to low-level mesh objects such as vertices, edges, faces, and regions, for accessing coordinate and adjacency information, and for setting and retrieving user-defined tag information on mesh entities. Discussions are under way to determine interfaces for mesh services, canonical ordering of entities, and query interfaces for distributed meshes in a parallel computing environment. These low-level interfaces will account for differences in how mesh data is typically stored for structured, unstructured, and hybrid meshes. For example, structured meshes can be represented compactly with node point locations and an indexing scheme, whereas unstructured meshes require explicit representation of the connectivity information. We will therefore group our interface definitions into broad categories appro-

priate for each mesh type; they will be general enough to support a broad range of tools but specific enough to support high-performance, efficient implementations.

Once this initial work on low-level interfaces is complete, we will develop high-level interfaces to support access to the entire grid hierarchy shown in Figure 2. At this level, users can call functions that provide, for example, partial differential operator discretizations, adaptive mesh refinement or multilevel data transfer over the entire mesh; these will enable the rapid development of new mesh-based applications. In addition, we will include support for advanced geometric capabilities such as generalized geometry-based tensor attributes with general distributions and dependencies; these will allow each mesh generation tool to have a common view of the geometry and physical attributes associated with the level A physical domain. We then will define the common interfaces needed to support various forms of adaptation. These interfaces will include access to error estimators and directionally dependent refinement indicators, accounting for high-order curved element mesh entities, and adaptive refinement that remains true to the geometric definition. To ensure the appropriateness of our interfaces, we will work closely with application, solver, and component technology researchers working in the Common Component Architecture (CCA) forum [3]. We invite participation by other interested researchers; by cooperating in this effort, a larger group of tool developers will provide input to, and eventually adopt, the proposed common interfaces that in turn will promote broader community usage and acceptance.

In addition to the development of common interface specifications, new technology development is needed in several areas to enable the effective creation and use of hybrid meshing technologies. In this paper, we comment on our work in two such areas: mesh quality improvement and adaptive technologies.

Mesh Quality Improvement. New mesh quality improvement algorithms and software are needed for the transition regions in conforming hybrid meshes obtained from multiple meshing codes. This need is especially critical for high-order methods that employ relatively few macro-elements; the conditioning of the governing systems can be impacted significantly by the deformation of just a few elements. To address this need, we are developing a stand-alone mesh improvement tool called MESQUITE that will work on all TSTT mesh types and that incorporates a broad spectrum of state-of-the-art approaches. MESQUITE will use the low-level common interfaces for accessing mesh geometry and topology and will be compatible with all of the existing and proposed TSTT mesh generation tools.

Adaptive Technologies. In many applications, the mesh description of geometry evolves during a computation. Reasons for mesh evolution include adaptive mesh refinement/coarsening, internally tracked interfaces or fronts arising in simulation physics, or motion of domain boundaries. A significant amount of basic research will be done to provide interoperable adaptive techniques within the context of the mesh hierarchy shown in Figure 1. We will consider adaptive procedures that account for mesh modifications locally (e.g., on an element-by-element basis), in selected subdomains, or in the entire domain employing information from the mesh hierarchy. In particular, we will develop adaptive techniques that address the following issues: maintenance of the true geometry of the domain to preserve the convergence rates of high order discretization techniques, effective abstraction of adaptive analysis procedures to promote "plug and play" interoperability, incorporation of advanced front tracking algorithms to allow for the explicit representation of solution discontinuities and material interfaces, mesh modification procedures that account for multiple refinement criteria to extend their applicability, and automatic selection and application of optimal adaptive strategies to improve ease of use.

High-Order Discretization

The complexities of discretizing new applications on unstructured and adaptively evolving grids have hampered widespread use of many powerful tools, leading to suboptimal strategies or to lengthy implementation periods. To overcome these limitations and to complement the mesh generation effort, we are developing a *Discretization Library* that will simplify the use of numerous mathematical operators that are common to PDEs arising in many applications. This library will support commonly used boundary conditions, will be extensible to provide application specific customization, and will be independent of the underlying mesh type and therefore interoperable with all TSTT meshing technology. Such techniques have been developed for structured mesh topology as part of the Overture project and for variational discretization (finite element, spectral element, partition of unity, etc.) as part of the Trellis project. We will use and expand on these approaches to lower the time, cost, and effort needed to effectively deploy modern discretization tools.

Initially we will include various arithmetic and differentiation operators as well as interpolation and projection operators to support multilevel solution strategies and preserve local conservation. Other common vector (div, grad, curl), tensor and boundary condition operators will be added to this initial set. The Discretization Library will define operators for finite difference finite volume, finite element, spectral element, discontinuous Galerkin,

and partition of unity methods. Operators of all orders will be considered, but our emphasis will be on creating high-order and variable order methods for use with adaptive methods. The library will be extensible so that one can easily include application-specific discrete operators such as projection operators to handle incompressibility constraints and stabilization operators for convection-dominated viscous flow problems. The design of the Discretization Library will support for temporal discretization strategies will range from the commonly used method-of-lines formulation where time steps and temporal methods are spatially independent, to local refinement methods, where time steps and methods vary in space, to space-time techniques where unstructured meshes are used in space and time.

Once a basic library of spatial and temporal operators is created, we will expand our efforts to support adaptive methods by including error estimation and interpolation procedures. *À* posteriori estimates of discretization errors are the best tools to guide adaptive approaches that use automatic mesh refinement/coarsening (h-refinement), variation of method order (p-refinement), and mesh motion (r-refinement). Some error estimates can be provided within the Discretization Library with relatively little difficulty. For example, simple ad hoc error indicators, such as solution gradients and various vorticity metrics, can be composed from tools within the Discretization Library. More complex methods that create estimates based on h- or p-refinement will also be straightforward to implement by using existing library tools. As the mesh is adapted or if different meshes discretize the same portion of the computational domain, the solution fields must be transferred among the mesh entities. Generalized procedures will be developed to determine the interacting mesh entities and to perform interpolations meeting such requirements as local conservation properties. We will consider searching structures and parametric inversion procedures for curved domain problems using both low- and high-order element geometries.

As with mesh generation tools, we will develop common interfaces that support a wide variety of discretization technologies. Interfaces will be exposed at different levels so that practitioners wishing to rapidly implement a new application, software developers wishing to add new operators to the library, and applications specialists seeking to incorporate a particular technology into their software can all benefit from the library. Low-level interfaces will be of immediate use for existing applications. These interfaces are simple to construct, although more tedious to use because they reveal many details that will be hidden at higher levels. The higher-level interfaces greatly simplify the construction of new applications and give users the opportunity to experiment with various combinations of discretization

technologies and optimize performance for their application. At all levels, the simplification afforded by hiding the complex grid-dependent details from users will shorten development times, reduce costs, enhance software reuse, and improve reliability of applications implemented and maintained in this manner.

It is critical that the kernel operations contained in the Discretization Library achieve optimal performance on terascale computing architectures. To accomplish this goal, we will investigate mechanisms for compile-time optimization using the ROSE preprocessing mechanisms [6]. This work will address the hierarchical memory performance and cache usage of the operators defined within the Discretization Library and user-defined operators. The concepts of generic programming where algorithms are separated from particular data structures is also an important factor in implementing distinct discretization strategies on the diverse mesh structures [5]. In addition, the algorithms for creating and composing the discrete operators and (for implicit operators) assembling them into a global algebraic system must scale well on distributed-memory architectures. In most cases, because the discrete operators are local, we will be able to accomplish this goal easily. Interprocessor communication is required only for entities on the boundaries of processor partitions (Level 3, Figure 2).

Interaction with Applications

One of the primary tenets of the SciDAC program is a tight coupling between software infrastructure centers such as TSTT and SciDAC application teams in the fusion, climate, accelerator design, and combustion areas. We have close interactions with scientists in each of these areas and feel strongly that only by addressing the needs of application scientists can we make substantial progress on the technical objectives of the TSTT center. In the near term, we are working to insert existing TSTT center technology into application codes. As one example, we are working with scientists in the fusion area to insert high-order finite element methods and advanced mesh alignment techniques into the parM3D software for resistive and two-fluid MHD models. Through short-term interactions such as this, we will be better able to abstract the necessary interfaces for our long-term interoperability goals.

Further Information

More information on the TSTT center's goals, research agenda, and investigators can be found at `www.tstt-scidac.org`.

Acknowledgments

This work was performed under the auspices of the U.S. Department of Energy, Office of Advanced Scientific Computing, SciDAC program by the University of California, Lawrence Livermore National Laboratory under contract number W-7405-Eng-48, by Brookhaven National Laboratory under contract number DE-AC02-98CH10886 and by Argonne National Laboratory under contract number W-31-109-ENG-38.

References

[1] M. W. Beall and M. S. Shephard. An object-oriented framework for reliable numerical simulations. *Engineering with Computers*, 15(1):61–72, 1999.

[2] D. L. Brown, Geoffrey S. Chesshire, William D. Henshaw, and Daniel J. Quinlan. Overture: An object oriented software system for solving partial differential equations in serial and parallel environments. In *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*, 1997.

[3] Common Component Architecture Forum. see `http://www.-cca-forum.org`.

[4] CUBIT mesh generation environment. Technical Report SAND94-1100, Sandia National Laboratories, 1999.

[5] J.E. Flaherty and J.D. Teresco. Software for parallel adaptive computation. In M. Deville and R. Owens, editors, *Proceedings of the 16th IMACS World Congress on Scientific Computing, Applied Mathmatics and Simulation*, pages 174–176. IMACS, 2000.

[6] D. Quinlan. ROSE: Compiler support for object-oriented frameworks. In *Proceedings of Conference on Parallel Compilers (CPC2000)*, Aussois, France, January 2000.

[7] SciDAC: Scientific discovery through advanced computing, 2002. http://www.scidac.org.

[8] T. J. Tautges. CGM: A geometry interface for mesh generation, analysis and other applications. to appear *Engineering with Computers*, 2001.

[9] H.E. Trease and L.L. Trease. NWGrid: A multi-dimensional, hybrid, unstructured, parallel mesh generation system, 2001. http://www.emsl.pnl.gov/nwgrid.