Sandia National Laboratories

U.S. DEPARTMENT OF ENERGY

Program for "Algorithms and Abstractions for Assembly in PDE Codes" Workshop at Sandia National Laboratories*

May 12-14, 2014

FORMAT: All talks will be in CSRI/90. There will be three one hour long (with ten minutes for questions) keynote talks, and 24 "poster blitz" talks. Each blitz speaker is given 15 minutes to summarize the main points of their poster that will be presented. The idea is to stimulate deeper conversation on these topics. The poster may only be displayed during the allocated poster session following the poster blitz at the end of the day. Presenters are responsible for making sure their own poster is displayed.

SCHEDULE:

Monday - May 12, 2014

Start Time	Speaker and Title
9:00a-9:15	Opening Remarks
9:15a-10:15	Keynote: **Martin Berzins** - University of Utah *Software Abstractions for Extreme-Scale Scalability of Computational Frameworks*
10:15a-10:30	Break
10:30a-12:00	Poster Blitz **Andreas Kloeckner** - University of Illinois *Operator transformation and code generation for FEM* **Roy Stogner** - University of Texas *C++14 Generic Programming as a Domain-Specific Language for PDEs* **Irina Demeshko** - Sandia National Laboratories *A performance-portable implementation of the Finite Element Assembly: preliminary results of using Kokkos in the Albany code* **James Sutherland** - University of Utah *Flexible, Efficient Abstractions for High Performance Computation on Current and Emerging Architectures* **Roger Pawlowski** - Sandia National Laboratories *Template-based Generic Programming Techniques for Finite Element Assembly* **Janine Bennett** - Sandia National Laboratories *Fault-tolerant programming at the extreme-scale*
12:00-1:30p	Lunch
1:30p-2:30	Poster Blitz **David Andrs** - Idaho National Laboratory *Massive Hybrid Parallelism for Fully Implicit Multiphysics* **David Moulton** - Los Alamos National Laboratory *Amanzi and the Arctic Terrestrial Simulator: Flexible Multiphysics Simulators for Environment and Ecosystem Applications* **Ken Franko** - Sandia National Laboratories *MiniAero* **Eric Phipps** - Sandia National Laboratories *Improving PDE Assembly Performance Through Embedded Uncertainty Quantification*
2:30p-5:00	Poster Session

*Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Tuesday - May 13, 2014

Time	Speaker and Title
9:00a-10:00	Keynote: **Mike Heroux** - Sandia National Laboratories
	Challenges and Opportunities for Scalable Finite Element Assembly
10:00a-10:30	Break
10:30a-12:00	Poster Blitz
	Rob Kirby - Baylor University
	Fine-grained finite element parallelism
	Shawn Pautz - Sandia National Laboratories
	Matrix Assembly Tasks in the Sceptre Deterministic Radiation Transport Code
	Tim Warburton - Rice University
	OCCA: A Unified Approach to Multi-Threading Languages
	Tzanio Kolev - Lawrence Livermore National Laboratory
	Scalable high-order finite elements with MFEM, hypre and BLAST
	Onkar Sahni - RPI
	Abstractions and algorithms for adaptive methods on boundary layer meshes
	Rich Drake - Sandia National Laboratories
	The ALEGRA Production Application: Strategy, Challenges and Progress Toward Next Generation Platforms
12:00-1:30p	Lunch
1:30p-2:30	Poster Blitz
	Dan Sunderland - Sandia National Laboratories
	Thread Scalable CRS Graph Construction
	Ulrike Yang - Lawrence Livermore National Laboratory
	Matrix and Vector Assembly in hypres Conceptual Interfaces
	Mark Hoemmen - Sandia National Laboratories
	Tpetra interface changes to support thread-parallel fill
	Bruno Turcksin - Texas A&M
	Multithreaded matrix assembly for finite elements
2:30p-5:00	Poster Session

Wednesday - May 14, 2014

Start Time	Speaker and Title
9:00a-10:00	Keynote: **Paul Fischer** - Argonne National Laboratory
	Scaling PDE solvers beyond a million cores
10:00a-10:15	Break
10:15a-11:30	Poster Blitz
	Jed Brown - Argonne National Laboratory
	High-performance matrix-free operator application and preconditioning
	Carter Edwards - Sandia National Laboratories
	MiniFENL: Fully Hybrid Parallel and Performance Portable Nonlinear Finite Element Miniapp using MPI+Kokkos
	Christian Trott - Sandia National Laboratories
	A migration strategy for utilizing the Kokkos many-core programming model
	Kendall Pierson - Sandia National Laboratories
	Efficient Block Sparse Assembly with SIMD
	Eric C. Cyr - Sandia National Laboratories
	Global Unknown Numbering for Fully-Coupled Mixed Finite Element Methods
11:30p-12:30	Poster Session

ABSTRACTS: Speakers in bold

David Andrs, Derek Gaston, Cody Permann, John Peterson, Andrew Slaughter, Richard Martineau
Title: *Massive Hybrid Parallelism for Fully Implicit Multiphysics*
Abstract: As hardware advances continue to modify the supercomputing landscape, traditional scientific software development practices will become more outdated, ineffective, and inefficient. The process of rewriting/retooling existing software for new architectures is a Sisyphean task, and results in substantial hours of development time, effort, and money. Software libraries which provide an abstraction of the resources provided by such architectures are therefore essential if the computational engineering and science communities are to continue to flourish in this modern computing environment. The Multiphysics Object Oriented Simulation Environment (MOOSE) framework enables complex multiphysics analysis tools to be built rapidly by scientists, engineers, and domain specialists, while also allowing them to both take advantage of current HPC architectures, and efficiently prepare for future supercomputer designs. MOOSE employs a hybrid shared-memory and distributed-memory parallel model and provides a complete and consistent interface for creating multiphysics analysis tools. A brief discussion of the mathematical algorithms underlying the framework and the internal object-oriented hybrid parallel design are given. Representative massively parallel results from several applications and a brief discussion of future areas of research for the framework will be presented.

Janine Bennett, John Floren, Hemanth Kolla, Nicole Slattengren, Keita Teranishi, Jeremiah Wilke
Title: *Fault-tolerant programming at the extreme-scale*
Abstract: It is widely acknowledged that performant software at exasacle will require significant increases in fine-grained parallelism and resiliency. Asynchronous, many-task programming models are acknowledged to provide the desired levels task- and data-level parallelism and, furthermore, show promise at sustaining performance despite node degradation and failures. Asynchronous task models introduce a challenging distributed consistency problem both within and amongst several interacting components (e.g. scheduler, global address server, transport layer), which demands a large set of programming model and runtime tools to address process failures. Existing many-task solutions often have nascent resilience support, addressing a subset of the resilience problem. In this poster we outline a holistic resilience approach based on a deferred consistency model. As much as possible, we isolate the resilience problem to a distributed hash table and library of resilient collective communications, transforming the massive challenge of resilient many-task scheduling and execution into related, but better understood resilience problems.

Martin Berzins
Title: *Software Abstractions for Extreme-Scale Scalability of Computational Frameworks*
Abstract: Abstractions play a key role in the development of both computer and computational science. Often the choice of the abstraction is of key importance in enabling performance at the required level. At the same time the choice of abstraction alone may not be enough to guarantee that performance. A key abstraction in the move to extreme-scale computing is sometime stated to be that of basing execution around the concept of multiple directed acyclic graphs of tasks. We will show that using such an approach within the Utah Uintah makes it possible to separate the user specification and the runtime that executes the resulting tasks. This separation then makes it possible to scale the same (unchanged) applications code from 700 to 700K cores. The mechanism for making such an abstraction work is the constant re-engineering of the runtime system, based on a careful analysis of its performance. The techniques that make it possible for the Uintah software framework to run complex engineering applications at such scales will be described and their use illustrated in the context of problems such as modeling energetic materials, clean-coal turbulent combustion and multiscale materials by design. Finally the challenge of extending such an abstraction to present and future heterogeneous machines will be considered.

This work is joint with Alan Humphrey, Qingyu Meng and John Schmidt from the runtime system and Jacqueline Beckvermit, Todd Harman and Jeremy Thornock from the applications side.

Jed Brown, Dave May, Matt Knepley
Title: *High-performance matrix-free operator application and preconditioning*
Abstract: Assembled sparse matrices lead to algorithms with extremely low arithmetic intensity, thus using hardware inefficiently. The same linear systems can often be represented using less memory by storing information at quadrature points or flux points. In this form, operator application looks more like residual assembly. Preconditioning techniques need to be adapted to these representations. Techniques will be compared on the basis of generality and performance (up to 30% of FPU peak for some variants).

Eric C. Cyr, Ben Seefeldt, Roger Pawlowski
Title: *Global Unknown Numbering for Fully-Coupled Mixed Finite Element Methods*
Abstract: TBD

Irina Demeshko, H. Carter Edwards, Michael A. Heroux, Eric T. Phipps, Andrew G. Salinger
Title: *A performance-portable implementation of the Finite Element Assembly: preliminary results of using Kokkos in the Albany code*
Abstract: The diversity of modern HPC architectures and programming models introduces a performance portability issue: parallel code needs to be executed correctly and performant despite variation in the architecture, operating system and software libraries. In this poster we present our progress towards a performance portable implementation of Finite Element Assembly in the Albany code, based on using the Kokkos programming model from Trilinos.

Rich Drake
Title: *The ALEGRA Production Application: Strategy, Challenges and Progress Toward Next Generation Platforms*
Abstract: ALEGRA is a large, highly capable, option rich, production application solving coupled multi-physics PDEs modeling magnetohydrodynamics, electromechanics, stochastic damage modeling and detailed interface mechanics in high strain rate regimes on unstructured meshes in an ALE framework. Nearly all the algorithms must accept dynamic, mixed-material elements, which are modified by remeshing, interface reconstruction, and advection components. Recent trends in computing hardware have forced application developers to think about how to address and improve performance on traditional CPUs and to look forward to next generation platforms. Core to the ALEGRA performance strategy is to improve and rewrite loop bodies to be conformant with the requirements of high performance kernels, such as accessing data in array form, no pointer dereferencing, no function calls, and thread safety. Necessary to achieve this, however, are changes to the underlying infrastructure. We report on recent progress in the infrastructure to support array-based data access and on iteration of mesh objects. The effects on performance on traditional platforms will be shown. We also discuss the practical realities and cost estimates for attempting to move an existing full featured production application like ALEGRA toward running effectively on future platforms and being maintainable at the same time.

H. Carter Edwards
Title: *MiniFENL: Fully Hybrid Parallel and Performance Portable Nonlinear Finite Element Miniapp using MPI+Kokkos*
Abstract: MiniFENL is a miniapplication which solves a nonlinear system of equations generated from a finite element discretization. MiniFENL is implemented with MPI+Kokkos for performance-portability to heterogeneous platforms with manycore CPUs and accelerators such as Intel Xeon Phi and Nvidia GPUs. Every phase of miniFENL is hybrid parallel: internal generation of the finite element mesh, construction of the sparse linear system graph from the finite element mesh connectivity, computation of per-element nonlinear residuals and Jacobians, assembly of these per-element

contributions into the global sparse linear system, and two level Newton / conjugate gradient iterative solution of the nonlinear problem. We use miniFENL to explore hybrid parallel algorithms and performance tradeoffs across the entire solution process. For example, we recently demonstrated that a global linear system assembly scatter-add approach has better performance than a gather-sum approach on both Xeon Phi and Nvidia Kepler accelerators. The scatter-add approach uses atomic-fetch-and-add operations for thread safety whereas the gather-sum approach saves per-element contributions into a temporary array and then mines this array for a thread-safe one-thread-per-row assembly.

Paul Fischer
Title: *Scaling PDE solvers beyond a million cores*
Abstract: We discuss design and performance of communication kernels in the context of PDE based simulation at petascale and beyond. In the first part of the talk, we present a gather-scatter (GS) framework that has a particularly simple interface and has demonstrated scaling to beyond 6 million MPI ranks. The interface requires a "setup" phase in which each participating rank supplies a vector of 64-bit integers that map local degrees of freedom to their global index. In subsequent "execute" phases, ranks supply a vector, an operand type (32- or 64-bit real/int), and an associative/commutative operator (+,*,min,max) that is applied across sets of scalar or vector operands sharing the same global index. Depending on the density of the underlying graph, GS will choose one of three exchange strategies: pairwise, crystal-router (CR), or all-reduce. The latter two nominally have log P complexity, save for all-reduce on BG/L-P-Q, where all-reduce is essentially P-independent out to a million ranks. The CR is a scalable generalized all-to-many that is also used in GS-setup. We discuss the performance of these kernels in the context of billion-point simulations on over 100,000 cores.

In the second half of the presentation we examine fundamental issues that will be critical for strong scaling at exascale given current trends in compute/communication ratios. We propose hardware supported reduce-scan stategies for essential kernels (e.g., algebraic multigrid) that could mitigate the internode latency that ultimately limits strong scalability and, thus, utility of exascale platforms.

Michael A. Heroux
Title: *Challenges and Opportunities for Scalable Finite Element Assembly*
Abstract: Emerging computer architectures are forcing the finite element community to consider disruptive algorithmic and software changes in order to exploit new commodity performance curves, and address expected resilience issues at extreme scales.

In this presentation we characterize the architectural trends that pose the most significant algorithmic challenges and opportunities for the design and implementation of the next generation of finite element computations. In particular, we discuss strategies for exploiting new performance trends, issues of latency and bandwidth, and abstract models for resilient algorithm development. Finally we discuss practical consideration for developing the next generation of library software in this area, including reproducibility, data structures and mixed threading model concerns.

Mark Hoemmen
Title: *Tpetra interface changes to support thread-parallel fill*
Abstract: Tpetra is Trilinos' next-generation sparse linear algebra package. It provides sparse graphs and matrices and dense vectors, and has a parallel data redistribution facility which applications can use. Tpetra lets users choose the type of values in its matrices and vectors, has been demonstrated to solve problems with well over two billion unknowns, and supports "hybrid" MPI + X parallelism for several different shared-memory parallel programming models X. This poster will show our work in progress to improve Tpetra's support for thread-parallel fill. By "fill," we mean constructing and modifying Tpetra data structures, like sparse matrices and dense vectors, as for example in finite element assembly. This work builds on the new Kokkos thread-parallel programming model, but does not require that applications use Kokkos. Our interface changes

will help applications gradually adopt threads, and guide application developers with performant idioms that support different data structure fill patterns.

Ken Franko
Title: *MiniAero*
Abstract: Kokkos was used to develop a mini-application for gas dynamics applications, miniAero. miniAero is an explicit cell-centered finite-volume code that is MPI enabled and uses Kokkos for thread and GPU execution of kernels. Performance numbers for MPI+X for a variety of platforms will be presented along with lessons learned.

Rob Kirby
Title: *Fine-grained finite element parallelism*
Abstract: This poster will demonstrate available concurrency in elementwise finite element kernels, as well as using expressing certain global operations in terms of shared-memory primitives. Preliminary numerical results will be given using PyOpenCL.

Andreas Kloeckner
Title: *Operator transformation and code generation for FEM*
Abstract: The present talk and poster discuss three software components designed to ease and automate tasks encountered in FEM assembly. The first, named 'pymbolic', is an expression tree with extensive traversal and rewriting capabilities. Both its mathematical vocabulary and its traversal operations are easily extended. This functionality is demonstrated in action in the context of operator description and transformation for discontinuous Galerkin (dG) FEM and high-order integral equation codes, with special attention paid to the transformation pipeline implemented and the design constraints imposed by each environment. A brief mention is made of 'PyOpenCL' that, in addition to providing a friendly interface to heterogeneous, shared-memory parallel computing hardware, incorporates an array container and implementations of a variety of parallel primitives, including scan, sort, and reduction. Making use of these foundations, a generic code generator is shown. 'Loo.py' targets shared-memory, massively parallel machines, and based upon a mathematical description of a computation along with a sequence of transformations, generates efficient, low-level code. Its use is shown in the context of FEM assembly and dG operator evaluation. All tools are hosted in the Python programming language, which, by its design, enables and encourages reuse, abstraction, and modularization. The tools are available under the MIT license and straightforwardly incorporated into user code.

Tzanio Kolev, Veselin Dobrev, Michael Kumbera, Robert Rieben
Title: *Scalable high-order finite elements with MFEM, hypre and BLAST*
Abstract: The finite element method (FEM) is a powerful discretization technique that can utilize general unstructured grids to approximate the solutions of many PDEs. High-order finite elements, in particular, are ideally suited to take advantage of the changing computational landscape, because their order can be used to tune the performance, by increasing the FLOPs/bytes ratio, or to adjust the algorithm for different hardware. In this poster we present our work on scalable high-order finite element software that combines the modular finite element library MFEM [1], the hypre library of scalable linear solvers [2], and the high-order shock hydrodynamics research code BLAST [3]. We will first discuss the finite element abstractions provided by MFEM, which include arbitrary high-order H1-conforming, discontinuous (L2), H(div)-conforming, H(curl)-conforming and NURBS elements, defined on general high-order meshes. We will then explain how the MPI-based version of MFEM uses data structures and kernels from the hypre library to enable scalable finite element assembly in parallel. Finally, we will describe the efficient implementation of high-order force matrices in the MFEM-based BLAST application, where we will also demonstrate the benefits of our approach with respect to strong scaling and GPU acceleration.
[1] https://mfem.googlecode.com
[2] https://www.llnl.gov/casc/hypre

[3] https://www.llnl.gov/casc/blast This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, LLNL-ABS-652336.

David Moulton, Ethan Coon, Markus Berndt, Rao Garimella
Title: *Amanzi and the Arctic Terrestrial Simulator: Flexible Multiphysics Simulators for Environment and Ecosystem Applications.*
Abstract: Climate and environmental simulations present a rich set of challenges for multiphysics and multiscale tools. The Advanced Simulation Capability for Environmental Management (ASCEM) program is tasked with addressing these challenges for the effective and defensible cleanup and closure of legacy nuclear waste sites. ASCEM initiated development of a flexible and extensible multiprocess simulator, dubbed Amanzi, as part of its open-source suite of tools. This simulator provides a flow and reactive transport capability on general unstructured polyhedral meshes using Mimetic Finite Difference (MFD) discretizations, services from Trilinos, and multigrid solvers from HYPRE. Its capabilities include variably saturated flow and reactive transport, including a wide range of chemical reactions. Recently, it's modular and flexible design was leveraged and extended in a project to model the climate impacts of a warming arctic through changes in microtopography and its coupling to the hydrology. This extension, the Arctic Terrestrial Simulator (ATS), is significant because the sheer number of processes that are coupled (potentially tightly coupled) defies hand coding and manual testing. Current capabilities include coupled surface and subsurface thermally dependent flows, along with a surface energy balance model including snow. In this poster we highlight the design and implementation approach we used to represent this complex system. Specifically, we refer to the mathematical description of a process as a process model, and its discrete representation as a process kernel (PK). We use a hierarchical representation of the complete system as a graph of PKs with a hierarchy of couplers (the Multi-Process couplers or MPCs). This PK/MPC graph provides a natural structure for the system, and hence we create a discrete distributed vector, a tree vector, that mimics this structure for use by solvers and time integrators. We developed a dynamic data manager, represented as a directed acyclic graph (DAG), to create complex models at runtime and manage the dependencies of their variables. To support an accurate representation of the physical model including polygonal ground, troughs, pinch outs, and ice wedges, we use the MFD method with polyhedral meshes using the MSTK mesh infrastructure. The challenge created by MFD methods is the need for scalar degrees of freedom on the faces of mesh elements. This leads to a block system of cell-based and face-based unknowns, even in scalar models such as thermal energy. To mimic this structure we create composite vectors, which are used naturally as leaves of the tree vector. These abstractions and structures provide flexible building blocks, and are collected in a package named Arcos. We are now beginning to explore performance and optimization. This includes investigating both the local assembly of element matrices as well as the assembly of the complete global system in more general matrices. The lack of a block interface to the HYPRE AMG solver leads to the explicit creation of Schur complements or the copying of block matrices into a unblocked form. Moreover, we have been focused on MPI based parallelism to this point and are now beginning to investigate threading options as well. Here we will demonstrate existing capabilities of Arcos and its use in Amanzi and ATS, and highlight the challenges and potential for future development of these codes.

Shawn Pautz, Clif Drumm, Wesley Fann, Bill Bohnhoff
Title: *Matrix Assembly Tasks in the Sceptre Deterministic Radiation Transport Code*
Abstract: The Sceptre radiation transport code implements discretizations of two different forms of the linear Boltzmann transport equation. Solvers for these discretizations are divided into two different classes. In one class of solvers the full matrix is formed, which is then solved with either CG or GMRES. In the other class we use a wavefront sweep algorithm to solve a block-lower triangular system, which allows assembly of numerous small on-node systems when needed. We describe the various operations that we perform in order to create either type of linear system.

Roger P. Pawlowski, Eric C. Cyr, Eric T. Phipps, and Andrew G. Salinger
Title: *Template-based Generic Programming Techniques for Finite Element Assembly*
Abstract: Modeling and simulation are used to understand, analyze, predict, and design increasingly complex physical, biological, and engineered systems. Because of this complexity, significant investments must be made, both in terms of manpower and programming environments, to develop simulation capabilities capable of accurately representing the system at hand. At the same time, modern analysis approaches such as stability analysis, sensitivity analysis, optimization, and uncertainty quantification require increasingly sophisticated capabilities of those complex simulation tools. Often simulation frameworks are not designed with these kinds of analysis requirements in mind, which limits the efficiency, robustness, and accuracy of the resulting analysis.

In this work, we describe an approach for building simulation code capabilities that natively support the requirements of many types of analysis algorithms. This approach leverages compile-time polymorphism and generic programming through C++ templates to insulate the code developer from the need to worry about the requirements of advanced analysis, yet provides hooks within the simulation code so that these analysis techniques can be added later. The ideas presented here build on operator overloading-based automatic differentiation techniques to transform a simulation code into one that is capable of providing analytic derivatives. However we extend these ideas to compute quantities that aren't derivatives such as polynomial chaos expansions, floating point counts, and extended precision calculations. The capabilities in this work have been released in the open-source Trilinos packages Sacado, Stokhos and Phalanx.

Eric Phipps, H. Carter Edwards
Title: *Improving PDE Assembly Performance Through Embedded Uncertainty Quantification*
Abstract: Achieving high performance for PDE assembly on emerging multicore architectures (such as GPUs, multi-core CPUs, and many-core accelerators) is often difficult due to memory access and code design patterns that are not commensurate with architectural capabilities. These architectures require accesses of wide regions of contiguous memory to achieve good performance, which is often challenging for PDE assembly on unstructured meshes. Furthermore, Intel-based architectures require consistent vectorization to achieve good performance, which is difficult for complex PDE codes. To address these issues, we explore opportunities for improving memory access patterns and vectorization by simultaneously propagating ensembles of PDE samples relevant to uncertainty quantification. Here we leverage the fact that memory access patterns and instructions are often very similar for PDE evaluations across samples in an uncertainty quantification calculation. We use template-based generic programming techniques to replace each scalar in the PDE assembly with a small array tuned to the natural vector length of the architecture, and organize data structure layouts so that data corresponding to each sample instance are stored contiguously in memory. The performance and scalability of this approach will be investigated on a variety of contemporary multicore architectures.

Kendall Pierson, Micah Howard, Michael Tupek
Title: *Efficient Block Sparse Assembly with SIMD*
Abstract: High Mach fluid regimes are critical environments to simulate, understand, and predict for the NW mission. Conchas is our high Mach application code built upon the Sierra toolkit, a custom block compressed sparse row data structure and a native point-implicit solver. This work describes the transformation of the data structures to take advantage of SIMD instructions to improve current performance through vectorization which is a necessary step towards multi-core, GPU, and next-generation platforms.

Onkar Sahni
Title: *Abstractions and algorithms for adaptive methods on boundary layer meshes*
Abstract: A set of tools and techniques are presented for general unstructured meshes with a focus on adaptive methods for boundary layer meshes. Such meshes are useful, for example, in wall-bounded turbulent flows that require tightly controlled mesh spacing and structure near the walls.

An adaptive approach for such meshes must maintain highly anisotropic, graded, and layered elements near the walls while error estimators or indicators must incorporate the structure of the flow boundary layer and associated physics. Similarly, parallel procedures must account for mixed element types, i.e., in mesh modifications and dynamic load balancing. We present abstractions and algorithms that address these needs. We also present high-order discretization techniques for boundary layer meshes including use of higher interelement continuity in the wall-normal direction.

Roy Stogner
Title: *C++14 Generic Programming as a Domain-Specific Language for PDEs*
Abstract: Abstractions and techniques are shown for employing expression template class hierarchies in C++ to provide users with a natural way to express physics kernels and solve Initial Boundary Value Problems. Basic compile-time metaprogramming is used to construct an API which recasts PDE expressions in a syntax which is valid C++ yet also natural to write. Topics include the use of expression templates to generate GPGPU code and automatically differentiated Jacobian matrices, the use of C++14 return type deduction to enable kernel fusion within a modular code, and the use of generic programming to maintain flexibility of design and ease of debugging. Challenges relating to optimization, hybrid meshes, and mesh adaptivity will be discussed.

Daniel Sunderland, H. Carter Edwards
Title: *Thread Scalable CRS Graph Construction*
Abstract: Our portable thread scalable pattern for CRS graph construction consists of four simple steps: 1) parallel counting the non-zeros, 2) allocating storage, 3) parallel filling, and 4) parallel post-procssing each row. Counting the non-zeros can be one of the more difficult algorithms to correctly implement in a scalable way. We demonstrate a simple solution for parallel counting which uses a Kokkos UnorderedMap to achieve good scalability. We also show how the Kokkos UnorderedMap implements a portable, scalable, and lock-free insert.

James Sutherland, Matthew Might, Tony Saad, Christopher Earl, Abhishek Bagusetty
Title: *Flexible, Efficient Abstractions for High Performance Computation on Current and Emerging Architectures*
Abstract: Complexity for large-scale simulation software stems from two primary sources: the physics being simulated and the language abstractions for various hardware targets. Multiplicity of physical modeling options, each of which may introduce unique nonlinear coupling and dependencies, can create rigid, fragile software that isn't easily maintained or modified. Changes in hardware (e.g., multicore or GPU architectures) can require different computational kernels to be maintained for each hardware target. Handling these two general challenges together to produce efficient, scalable software can be a daunting challenge. This poster discusses two abstractions that work in tandem to address the aforementioned challenges. First, the software is written to represent nodes that can be self-assembled into a directed, acyclic graph (DAG) which exposes the structure of the calculation. This facilitates automated scheduling of nodes in the DAG, and reasoning about efficient management of CPU and GPU. Second, a domain-specific language, embedded in C++, is under development to allow the application programmer to specify high-level intent while allowing highly efficient back-ends targeting various hardware (CPU, GPU) to be generated at compile time. These two abstractions combine to create a powerful environment where application developers can increase productivity and deploy complex software across a variety of hardware environments.

Christian Trott
Title: *A migration strategy for utilizing the Kokkos many-core programming model*
Abstract: In order to support many-core architectures in Trilinos many packages have started to explore the utilization of Kokkos. Here a migration strategy will be presented for an incremental transition to using Kokkos, starting with simple thread-parallelism, continuing with GPU support and finishing with two and three-level parallelism employing thread teams and vectorization. A particular focus is put on software which already uses Tpetra, Trilinos' next-generation sparse linear

algebra package.

Bruno Turcksin, Martin Kronbichler, Wolfgang Bangerth
Title: *Multithreaded matrix assembly for finite elements*
Abstract: We present a design pattern that can be applied to any operation requiring to be done independently on every cell and which is followed by a reduction of the local result into a global data structure. This design pattern can be directly applied to multithreaded matrix assembly and implemented using the parallel pipeline design pattern. When assembling a global matrix for finite elements, a local matrix is assembled on each cell; this step is embarrassingly parallel. However, when the local matrices are incorporated into the global matrix, it is necessary to ensure that two processors do not attempt to write simultaneously in the same global matrix element. To prevent this, a colorization algorithm is used before the reduction operation; all the elements of a given color can be simultaneously written into. It is important for the colorization algorithm to produce few colors, but it is more so that the size of these colors are similar; small colors would degrade the scalability of the algorithm. This design pattern was implemented in the deal.II library and was shown to significantly speed up matrix assembly

Tim Warburton, David Medina, Amik St-Cyr
Title: *OCCA: A Unified Approach to Multi-Threading Languages*
Abstract: Currently there are a number of relatively popular APIs for multi-threading programming including but not limited to CUDA, OpenCL, OpenACC, and OpenMP. Initially it might appear that many-core programming forces programmers to lock into a specific API. Additionally simulation codes, frameworks, and libraries have lifetimes measured in decades that might outlive a specific API. To address these issues we developed the lightweight OCCA API in a way that allows a programmer to write single source kernel implementations that are portable and can be dynamically compiled and executed at run-time as CUDA, OpenCL, or OpenMP. Example performance results from our OCCA based finite difference, discontinuous Galerkin, and spectral element method based PDE solvers will show that it is possible to develop efficient and portable many-core code for CPUs and GPUs.

Ulrike Yang, Rob Falgout, Tzanio Kolev, Jacob Schroder
Title: *Matrix and Vector Assembly in hypres Conceptual Interfaces*
Abstract: The hypre software library provides high performance preconditioners and solvers for the solution of large sparse linear systems on massively parallel computers. One of its attractive features is the provision of conceptual interfaces, which include a structured, a semi-structured, and a traditional linear-algebra based interface. These interfaces give application users a more natural means for describing their linear systems, and provide access to methods such as structured multigrid solvers, which require additional information beyond just the matrix. We discuss the assembly of matrices and vectors within the various interfaces in hypre as well as current efforts to increase the use of OpenMP threads in the interfaces.