Exceptional service in the national interest

Sandia National Laboratories

New Multiphysics Coupling Tools for Trilinos: PIKE and DTK

Roger Pawlowski

Sandia National Laboratories

Roscoe Bartlett, Stuart Slattery, Mark Berrill, Kevin Clarno, and Steve Hamilton

Oak Ridge National Laboratory

Trilinos User Group Meeting
Wednesday October 29th, 2014

SAND2014-19290 PE

U.S. DEPARTMENT OF ENERGY | NNSA National Nuclear Security Administration

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. SAND NO. 2011-XXXXP

Driving Project

- The Consortium for Advanced Simulation of LWRs (CASL) is a DOE program to improve modeling and simulation of nuclear reactors

- Flagship product is a “Virtual Reactor” Simulation Suite based on code-to-code couplings

- Integrating both modern and **legacy codes**
  - Familiar, validated codes are valued in the community
  - Providing residuals for Newton-based coupling can require significant redesign
  - Production tools use simple Picard iteration
  - CASL does use Newton-based solvers for research and assessment of coupling algorithms
  - New Trilinos multiphysics tools (packages) have been abstracted

A Domain Model

Sandia National Laboratories

A Theory Manual for Multiphysics Code Coupling in LIME, R. Pawlowski, R. Bartlett, R. Schmidt, R. Hooper, and N. Belcourt, SAND2011-2195

$$f(\dot{x}, x, \{p_l\}, t) = 0$$

Residual

$$\dot{x} = \frac{\partial x}{\partial t}$$

State (DOF)

Time

Set of parameters

$x \in \mathbb{R}^{n_x}$ is the vector of state variables (unknowns being solved for),
$\dot{x} = \partial x / \partial t \in \mathbb{R}^{n_x}$ is the vector of derivatives of the state variables with respect to time,
$\{p_l\} = \{p_0, p_1, \ldots, p_{N_p-1}\}$ is the set of N_p independent parameter sub-vectors,
$t \in [t_0, t_f] \in \mathbb{R}^1$ is the time ranging from initial time t_0 to final time t_f,

$$f(\dot{x}, x, \{p_l\}, t) : \mathbb{R}^{\left(2n_x + \left(\sum_{l=0}^{N_p-1} n_{p_l}\right)+1\right)} \rightarrow \mathbb{R}^{n_x}$$

$$g_j(\dot{x}, x, \{p_l\}, t) = 0, \text{ for } j = 0, \ldots, N_g - 1$$

Response Function

$$g_j(\dot{x}, x, \{p_l\}, t) : \mathbb{R}^{\left(2n_x + \left(\sum_{l=0}^{N_p-1} n_{p_l}\right)+1\right)} \rightarrow \mathbb{R}^{n_{g_j}} \text{ is the } j^{\text{th}} \text{ response function.}$$

- Input Arguments: state time derivative, state, parameters, time
- Output Arguments: Residual, Jacobian, response functions, etc…

Extension to Multiphysics

Sandia National Laboratories

Split parameters into “coupling” and truly independent.

$$f_i(\dot{x}_i, x_i, \{z_{i,k}\}, \{p_{i,l}\}, t) = 0$$

Set of *coupling* parameters

Set of *independent* parameters

Require transfer functions:

- Can be complex nonlinear functions themselves

$$z_{i,k} = r_{i,k}(\{x_m\}, \{p_{m,n}\})$$

Transfer Function

Response functions now dependent on z

- Can be used as coupling parameters (z) for other codes

$$g_{i,j}(\dot{x}_i, x_i, \{z_{i,k}\}, \{p_{i,l}\}, t)$$

Response Function

Application Classification

Sandia National Laboratories

Inputs and outputs are ***optionally*** supported by physics model → restricts allowed solution procedures

Name	Definition	Required Inputs	Required Outputs	Optional Outputs	Time Integration Control
Response Only Model (Coupling Elimination)	$p \rightarrow g(p)$	p	g		**Internal**
State Elimination Model	$p \rightarrow x(p)$	p	x	g	**Internal**
Fully Implicit Time Step Model	$f(x,p) = 0$	x, p	f	$W,\ M,\ g$	**Internal**
Transient Explicitly Defined ODE Model	$\dot{x} = f(x,p,t)$	x, p, t	f	$W,\ M,\ g$	**External**
Transient Fully Implicit DAE Model	$f(\dot{x},x,p,t) = 0$	$\dot{x}, x, p, t$	f	$W,\ M,\ g$	**External or Internal**

$$W = \alpha \frac{\partial f}{\partial \dot{x}} + \beta \frac{\partial f}{\partial x} \qquad M = \text{preconditioner}$$

Application Classification

Inputs and outputs are ***optionally*** supported by physics model → restricts allowed solution procedures

Name	Definition	Required Inputs	Required Outputs	Optional Outputs	Time Integration Control
Response Only Model (Coupling Elimination)	$p \rightarrow g(p)$	BlackBox	g		Internal
State Elimination Model	$p \rightarrow x(p)$	p	x	g	Internal
Fully Implicit Time Step Model					Internal
Transient Explicitly Define ODE Model					
Transient Fully Implicit DAE Model	$f(\dot{x}, x, p, t) = 0$	$\dot{x}, x, p, t$	f	$W,\ M,\ g$	External or Internal

Implicit (Invasive)
- Requires Advanced Solver abstractions:
- Vectors, Operators, … (Thyra)

$$W = \alpha \frac{\partial f}{\partial \dot{x}} + \beta \frac{\partial f}{\partial x} \qquad M = \text{preconditioner}$$

Trilinos Implicit Multiphysics Capabilities

Sandia National Laboratories

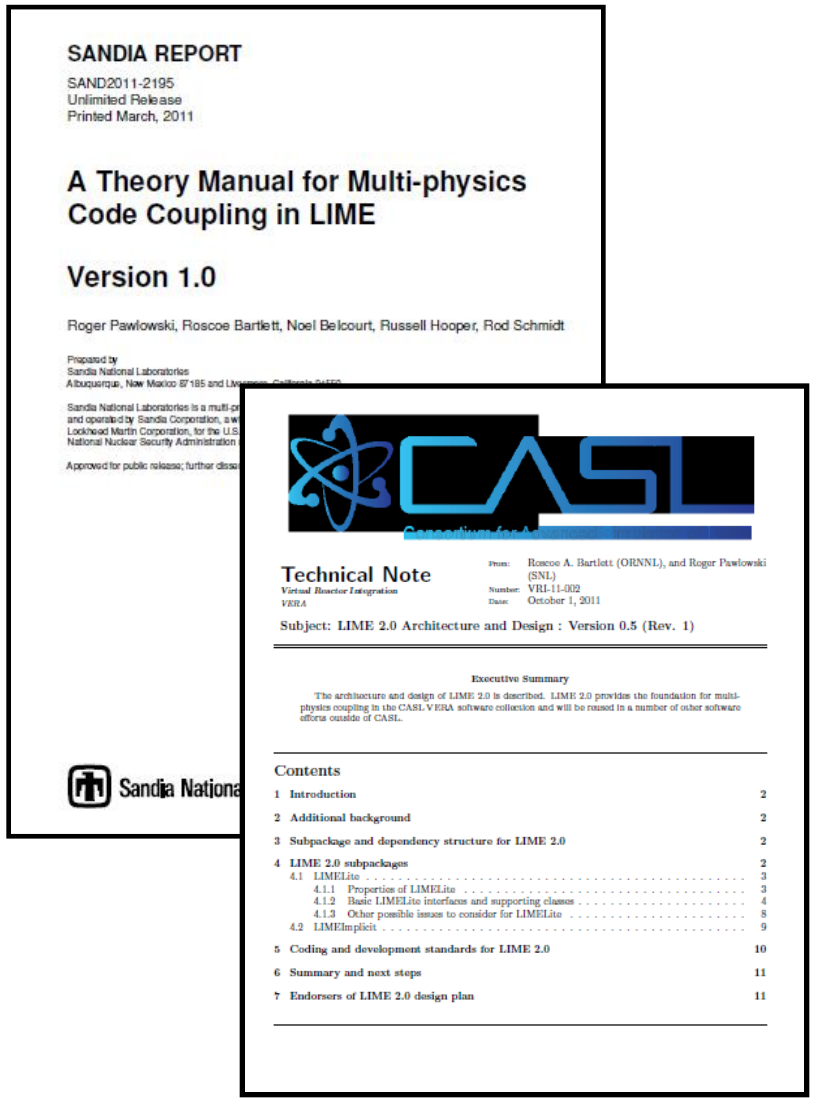
- Trilinos already supports Newton-based strong coupling
  - Block composite linear systems: **Thyra** product objects and Model Evalautor

$$\begin{bmatrix} \frac{\partial f_0}{\partial x_0} & \frac{\partial f_0}{\partial z_{0,0}} \frac{\partial r_{0,0}}{\partial x_1} \\ \frac{\partial f_1}{\partial z_{1,0}} \frac{\partial r_{1,0}}{\partial x_0} & \frac{\partial f_1}{\partial x_1} \end{bmatrix} \begin{bmatrix} \Delta x_0^{(k)} \\ \Delta x_1^{(k)} \end{bmatrix} = - \begin{bmatrix} f_0(x_0^{(k-1)}, r_{0,0}(x_1^{(k-1)})) \\ f_1(x_1^{(k-1)}, r_{1,0}(x_0^{(k-1)})) \end{bmatrix}$$

  - Sensitivities: **Sacado**
  - Blocked physics-based preconditioners: **Teko (ML, MeuLu, Ifpack, Ifpack2)**
  - Inexact Newton, Jacobian-Fee Newton-Krylov: **NOX**
  - Transient DAE: **Rythmos**
  - Multiphysics FE assembly engine: **Phalanx, Panzer**
    - DOF Manager: Fully coupled Newton with mixed basis (**Intrepid**), different equation sets in different element blocks
    - Basic framework for describing equation set, boundary conditions
    - Provides a **Thyra::ModelEvaluator** for solvers
- Demonstrated on leadership class machines (**Drekar**)!
- **See TUG 2011 (Hierarchical Toolchains for Nonlinear Analaysis, Panzer)**

Physics Integration KErnels (PIKE)

- Production coupled codes are Picard-based
- L3 milestone in PoR3 outlined the design of a "LIME 2"
- Generalization of Coupled drivers in CASL products
  - 2000 lines of c++
  - **Received DOE copyright for release in Trilinos**
- Benefits
  - Simplified and unified model interfaces
  - Explicit separation of global and local convergence
  - User defined convergence test hierarchy
  - User defined solvers
  - Unified control via observers
    - Unit testing, output summary, initialization
  - Timing control
  - Consistent with Trilinos coding guidelines



SANDIA REPORT

SAND2011-2195
Unlimited Release
Printed March, 2011

A Theory Manual for Multi-physics Code Coupling in LIME

Version 1.0

Roger Pawlowski, Roscoe Bartlett, Noel Belcourt, Russell Hooper, Rod Schmidt

Technical Note

Subject: LIME 2.0 Architecture and Design : Version 0.5 (Rev. 1)

Executive Summary

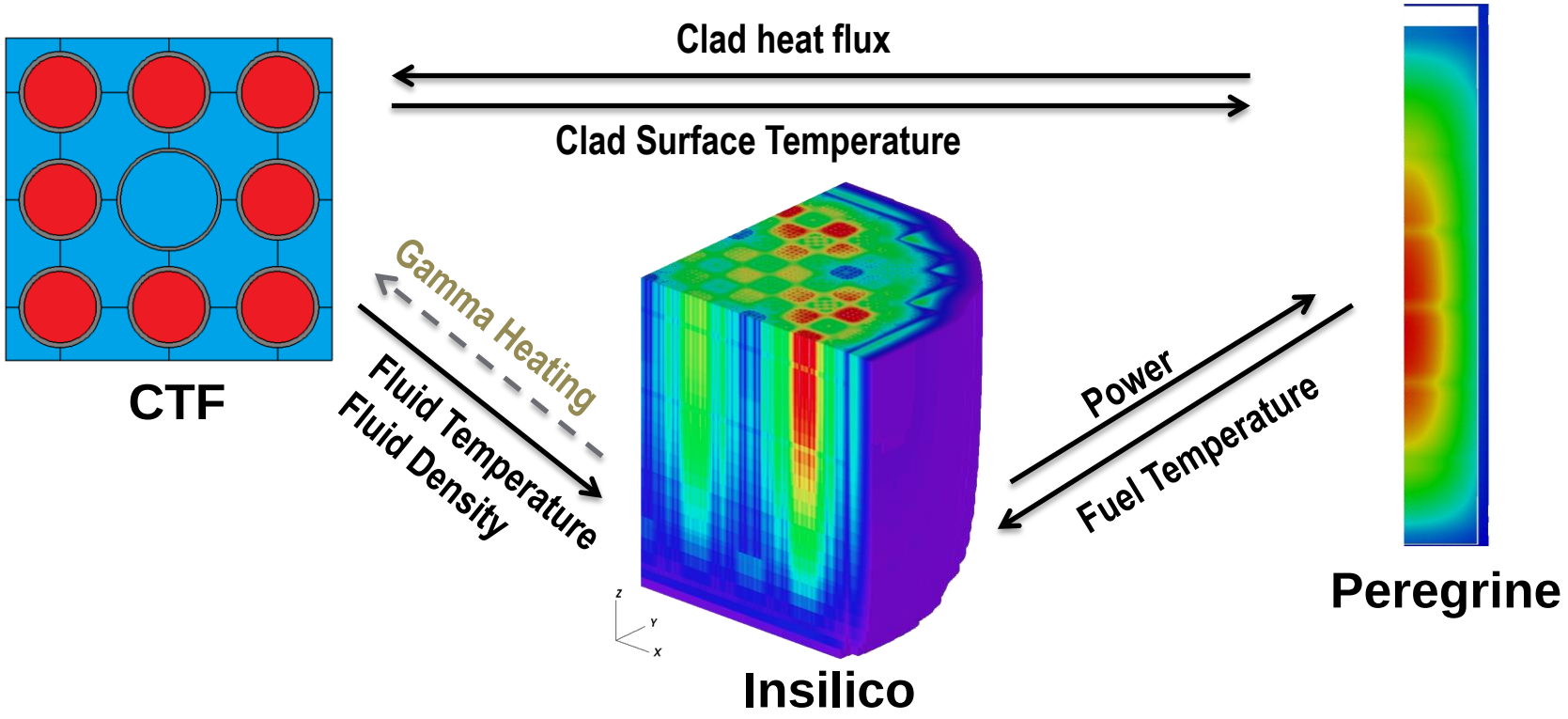
The architecture and design of LIME 2.0 is described. LIME 2.0 provides the foundation for multi-physics coupling in the CASL VERA software collection and will be reused in a number of other software efforts outside of CASL.

Contents

Tiamat: Core Simulator for Pellet Clad Interaction

Sandia National Laboratories

- CTF: Multiphase thermal hydraulics
- Insilico: SP_N neutronics
- Peregrine: Thermal conductivity, solid mechanics with contact



Example: Tiamat CPI

Sandia National Laboratories

- Application codes are a “black box”
  - Set parameters
  - Call Solve
  - Evaluate responses

$f_p(x_p, z_p) = 0$	Peregrine
$f_c(x_c, z_c) = 0$	CTF
$f_i(x_i, z_i) = 0$	Insilico
$z_{p,c} = r_{p,c}(x_c)$	Clad Temp
$z_{p,i} = r_{p,i}(x_i)$	Power
$z_{c,p} = r_{c,p}(x_p)$	Clad Heat Flux
$z_{i,p} = r_{i,p}(x_p)$	Fuel Temp
$z_{i,c} = r_{i,c}(x_c)$	Fluid Temp/density

- Jacobi and Gauss-Seidel options available.
- Transient, steady-state, and pseudo steady-state
- Strong coulping: All codes are subcycled to converge state within each time step!

Algorithm GS: Block Gauss-Seidel

GIVEN x_p^0, x_c^0, x_i^0.
FOR $k = 0, 1, \ldots$ (UNTIL CONVERGED) DO:
$k = k+1$
TRANSFER TO p:
$z_{p,c}^k = r_{p,c}(x_c^{k-1})$
$z_{p,i}^k = r_{p,i}(x_i^{k-1})$
SOLVE $f_p(x_p^k, r_{p,c}(x_c^{k-1}), r_{p,i}(x_i^{k-1})) = 0$ FOR x_p^k
TRANSFER TO c:
$z_{c,p}^k = r_{c,p}(x_p^k)$
SOLVE $f_c(x_c^k, r_{c,p}(x_p^k)) = 0$ FOR x_c^k
TRANSFER TO i:
$z_{i,c}^k = r_{i,c}(x_c^k)$
$z_{i,p}^k = r_{i,p}(x_p^k)$
SOLVE $f_i(x_p^k, r_{i,c}(x_c^k), r_{i,p}(x_p^k)) = 0$ FOR x_i^k

Pros and Cons of Picard

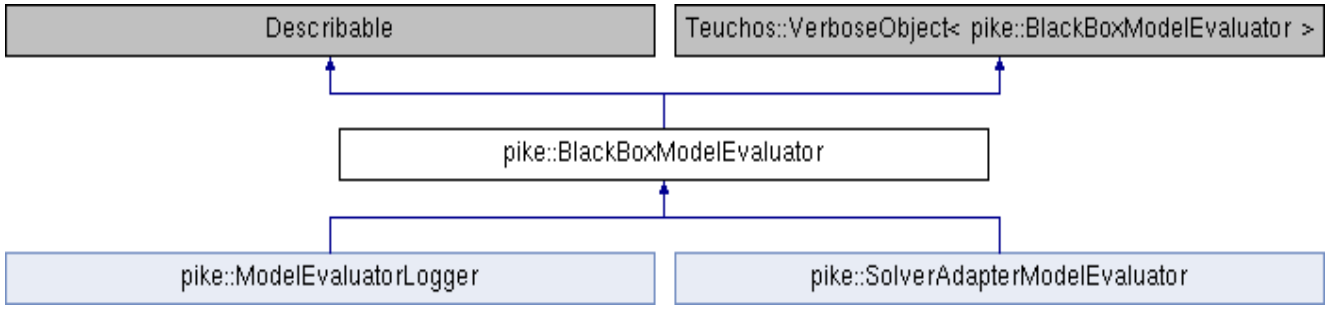
Sandia National Laboratories

- Advantages:
  - Simple to implement
  - Black-Box: Driver requires minimal knowledge of components (no solutions or residuals required)
  - Allows optimized/tuned solvers on individual physics
  - ***Easy for analysts to understand***

- Disadvantages:
  - Linear convergence rate
  - Robustness controlled by damping parameter selection
  - Inner/outer tolerance selection and convergence criteria
  - Sequential in physics domains (Gauss-Seidel)

Minimal Model Evaluator Interface

- Simplest steady-state Model Evaluator interface
- Low barrier to entry
- NO exposure to Thyra/Epetra Vectors or Thyra::Model Evaluators
  - Removed high entry barrier




```
namespace pike {
  class BlackBoxModelEvaluator :
     public Teuchos::Describable,
     public
     Teuchos::VerboseObject<pike::BlackBoxModelEvaluator> {

  public:
    virtual ~BlackBoxModelEvaluator();
    virtual std::string name() const = 0;
    virtual void solve() = 0;
    virtual bool isLocallyConverged() const = 0;
    virtual bool isGloballyConverged() const = 0;
        .
        .
        .
  };
}
```

Deprecated in favor of StatusTest objects

Expanded for Parameter, Response and Transient (support mixes pseudo SS)

Sandia National Laboratories

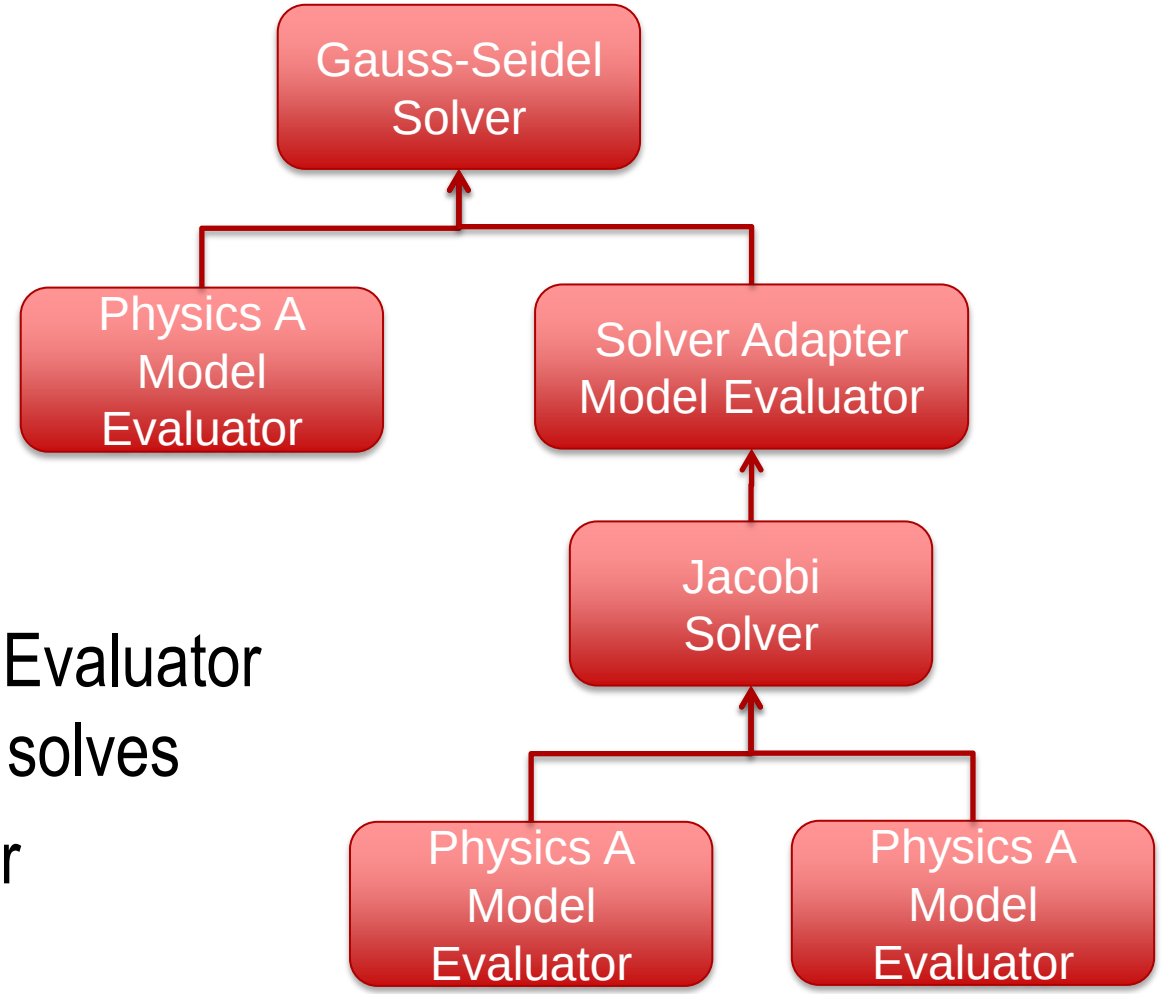
```
virtual bool supportsParameter(const std::string& pName) const;
virtual int getNumberOfParameters() const;
virtual std::string getParameterName(const int l) const;
virtual int getParameterIndex(const std::string& pName) const;
virtual void setParameter(const int l, const Teuchos::ArrayView<const double>& p);

virtual bool supportsResponse(const std::string& rName) const;
virtual int getNumberOfResponses() const;
virtual std::string getResponseName(const int j) const;
virtual int getResponseIndex(const std::string& rName) const;
virtual Teuchos::ArrayView<const double> getResponse(const int j) const;

virtual bool isTransient() const;
virtual double getCurrentTime() const;
virtual double getTentativeTime() const;
virtual bool solvedTentativeStep() const;
virtual double getCurrentTimeStepSize() const;
virtual double getDesiredTimeStepSize() const;
virtual double getMaxTimeStepSize() const;
virtual void acceptTimeStep();
```

Flexibility

Sandia National Laboratories

- Sovlers:
  - pike::SolverObserver for user injection of code at specific points in the solve
  - pike:AbstractSolverFactory for users to inject new solvers
  - pike::Factory for aggregating solver factories
- Status Tests
  - Splits convergence into local (application) and global (coupled problem)
  - Also defines Abstract Factory and Factory base classes
  - Follows NOX: abstract base class for StatusTest, combined into user defined hierarchy
- Utilities: Default Solver and ME classes, Logger Wrappers
- Hierarchical Solves are supported via SolverAdapterModelEvaluator wrapper

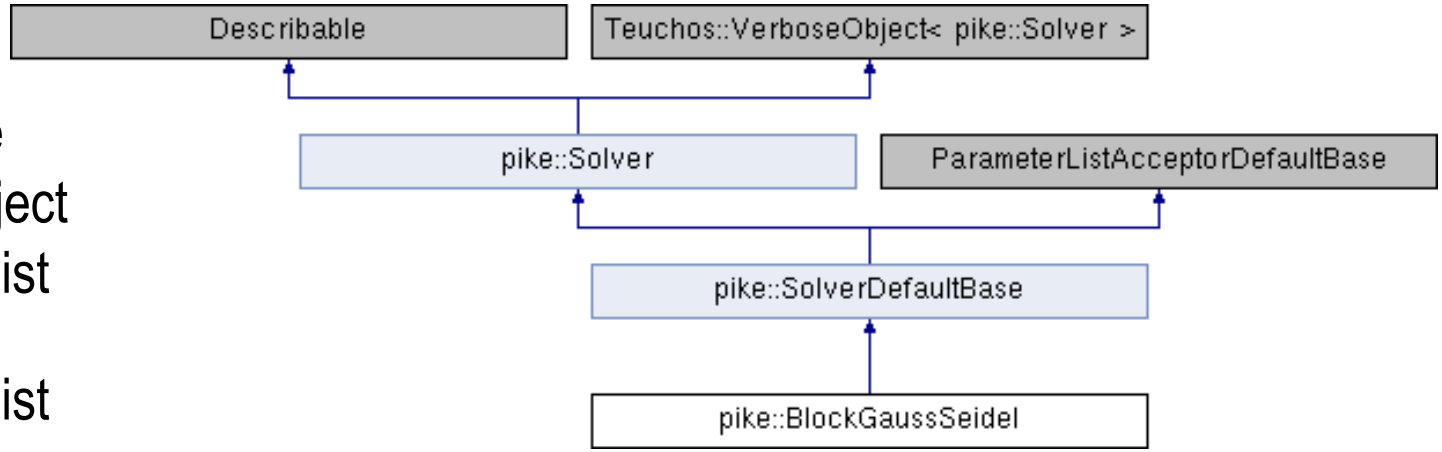
Hierarchic Solves

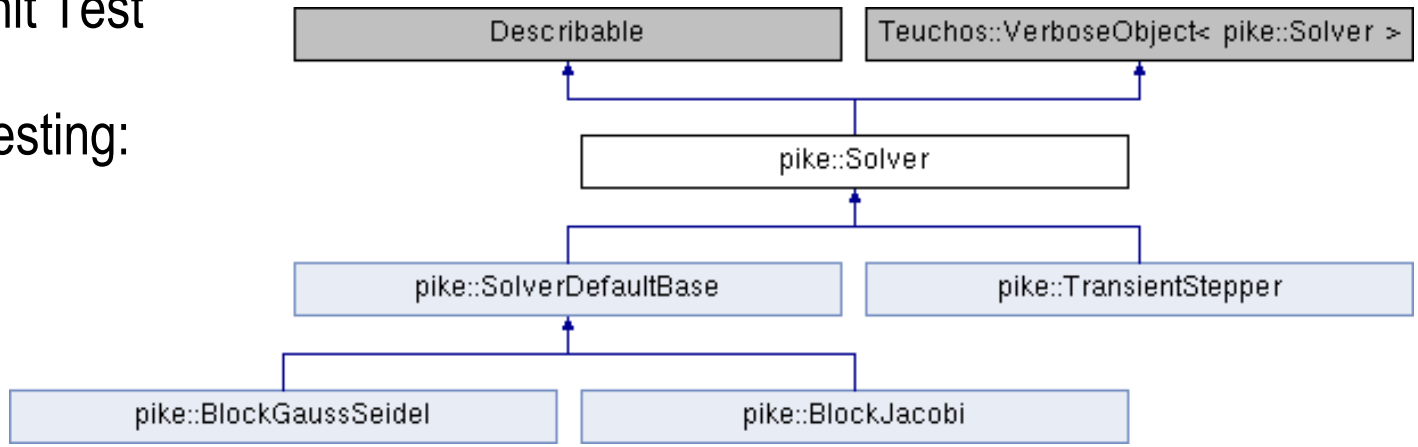


- SolverAdapterModelEvaluator allows for hierarchic solves
- Optimal data transfer

Follows Trilinos Coding Standards

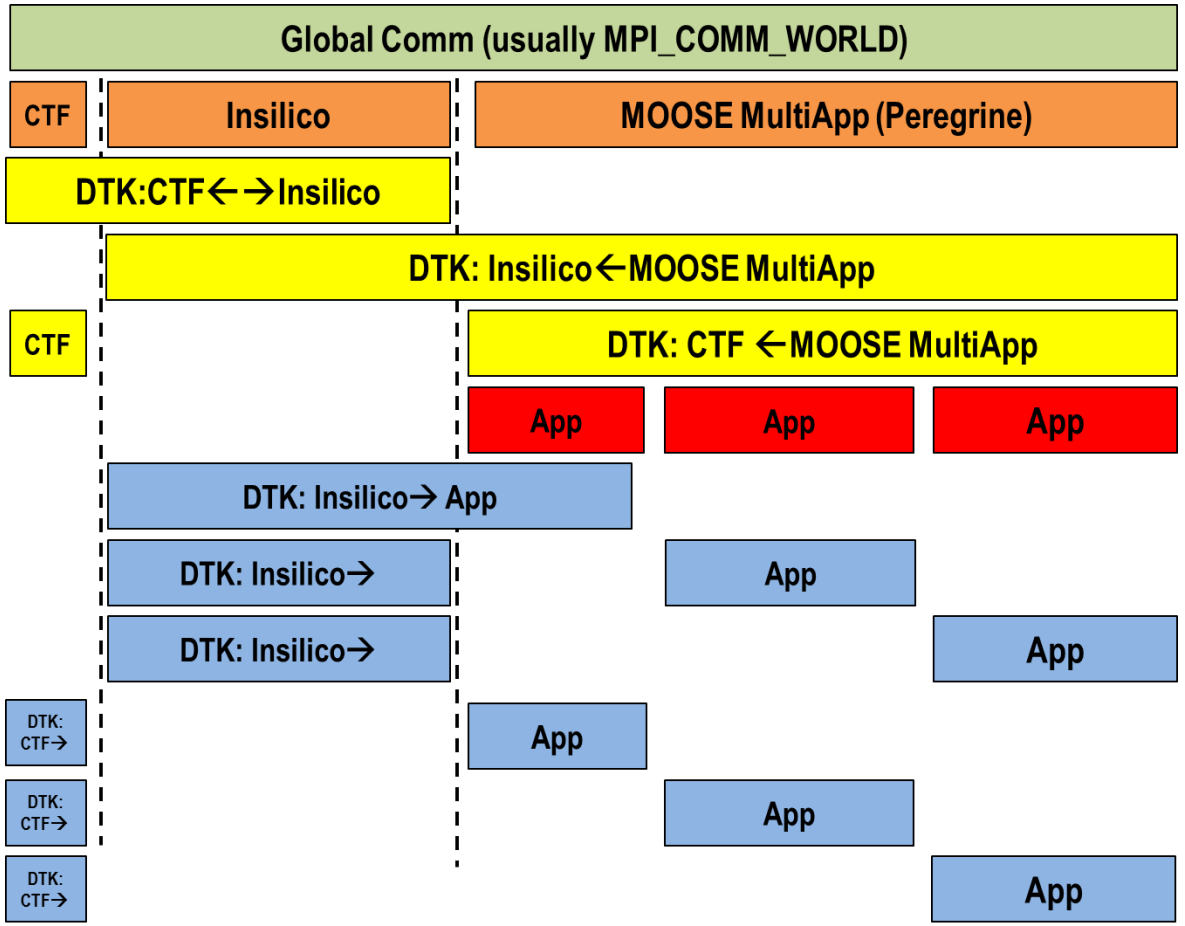
- Describable
- VerboseObject
- ParameterList Acceptor
- ParameterList Validation
- TimeMonitor
- Teuchos Unit Test Harness
- Coverage testing: 87%





The pike::MultiphysicsDistributor Class

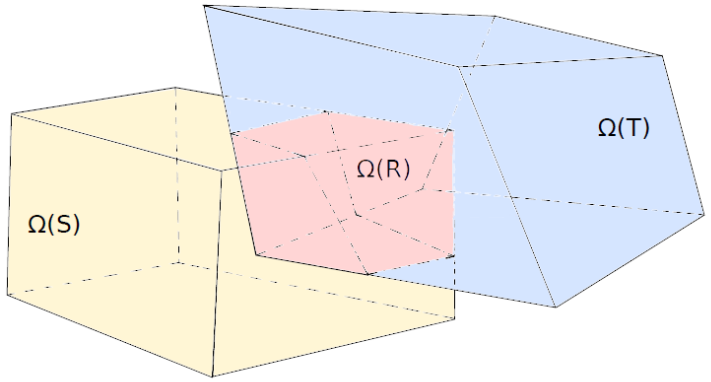
- Creates MPI Communicators and provides information for coupled problems
- Can overlap or segregate codes in MPI process space
- Data Transfers: DTK
  - In memory
  - Rendezvous algorithm



Data Transfer Kit (NEXT TALK: SLATTERY) (Slattery, Wilson, Pawlowski)

Sandia National Laboratories

- In-memory data transfers are critical for efficiency: NO file I/O for transfers
- Determines efficient point-to-point communication pattern for parallel transfer between codes
- Provides both volume and interfacial:
  - Volume-to-Volume: (Shared Domain)
    - Point Interpolation (post-scale for conservation)
    - Both mesh and Geometry based
  - Surface/Interfacial: Common Refinement (Jiao and Heath 2004)
  - Spline interpolation (de Boer et al. CMAME 2007)
- Uses Rendezvous algorithm (Plimpton et al., Journal of Parallel and Distributed Computing 2004)
  - N log(N) time complexity in parallel map generation



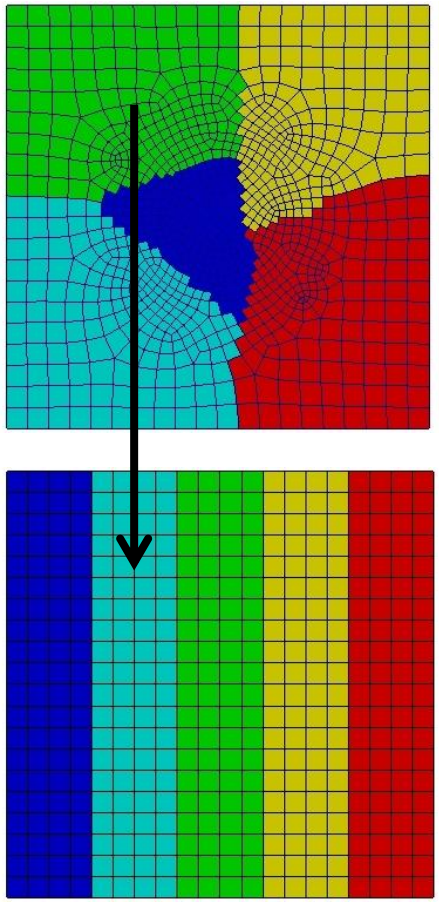
$$\mathbf{M} : \mathbb{R}^D \rightarrow \mathbb{R}^D, \forall r \in \Omega_R$$

$$\mathbf{G}(t) \leftarrow \mathbf{M}(\mathbf{F}(s))$$

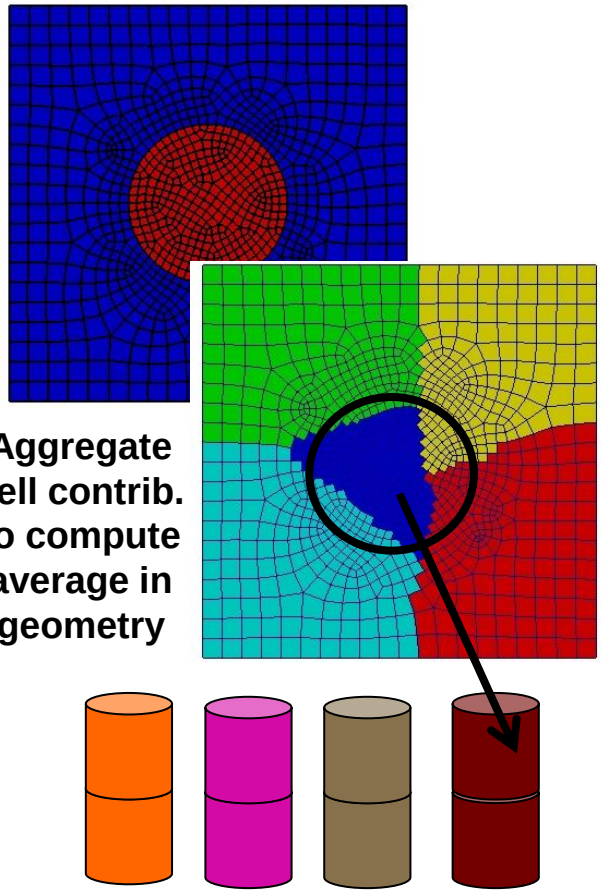
DTK Implements Mappings for Various Transfers (Rendezvous used by all Mappings)

Sandia National Laboratories

Shared Domain Map
Mesh →Point

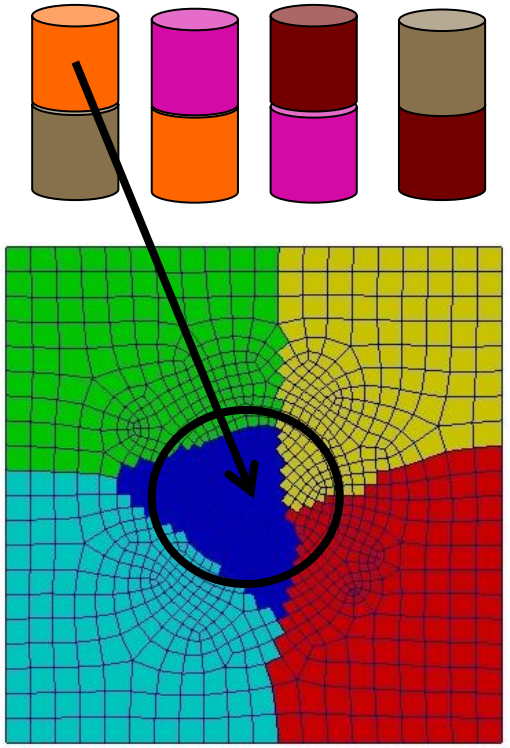


Integral Assembly Map
Mesh →Geometry



Aggregate cell contrib. to compute average in geometry

Shared VolumeMap
Geometry → Point



Colors represent different MPI processes

Pseudocode for MPD and DTK

Sandia National Laboratories

```
setupDTKAdapters {
  if (mpd->transferExistsOnProcess(DREKAR_TO_INSILICO)) {
    RCP<DataTransferKit::GeometryManager<…> > drekar_geom;
    RCP<DataTransferKit::FieldContainer<…> > insilico_target_points;

    if (mpd->appExistsOnProcess(DREKAR))
      drekar_geom = drekar_me->getSourceGeom();

    if (mpd->appExistsOnProcess(INSILICO))
      insilico_target_points = insilico_me->getTargetPoints();

    dtk_map->setup(drekar_geom, insilico_target_points);
  }
}
doTransfer() {
  …
}
```

Solver Comparisons

- Picard
  - Generalized Davidson eigensolver for SP_N
  - JFNK for thermal/subchannel solve
  - Damping factor = 0.4
- Anderson
  - Anderson(2)
  - Mixing parameter: β = -1.0
- JFNK-based methods
  - Block diagonal preconditioner on physics components:

$$\mathbf{P}\begin{pmatrix} f \\ \phi \\ T \end{pmatrix} = \begin{bmatrix} \mathbf{A}(T) & \mathbf{B}(T) & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \mathbf{L}(T) \end{bmatrix}$$

  - Diagonal blocks approximately inverted with Trilinos/ML
  - Trilinos/NOX JFNK solver with Belos GMRES
- Stopping tolerance: 10^{-4} nonlinear, 10^{-5} linear

Why Anderson Acceleration?

Sandia National Laboratories

- Minimal change to "black-box" codes: $x = g(x)$

Algorithm AA: Anderson Acceleration

GIVEN x_0 AND $m \geq 1$.
SET $x_1 = g(x_0)$.
FOR $k = 1, 2, \ldots$ (UNTIL CONVERGED) DO:
  SET $m_k = min\{m, k\}$.
  DETERMINE $\gamma^{(k)} = (\gamma_0^{(k)}, \ldots, \gamma_{m_k-1})^T$ THAT SOLVES
    $min_{\gamma^{(k)}=(\gamma_0^{(k)},\ldots,\gamma_{m_k-1})^T} \|f_k - \mathcal{F}_k\|_2$.
  SET $x_{k+1} = g(x_k) - \mathcal{G}_k\gamma^{(k)}$.

$f_i = g(x_i) - x_i$

$\mathcal{F}_k = (\Delta f_{k-m_k}, \ldots, \Delta f_{k-1})$ with $\Delta f_i = f(x_{i+1}) - f(x_i)$.

$\mathcal{G}_k = (\Delta g_{k-m_k}, \ldots, \Delta g_{k-1})$ with $\Delta g_i = g(x_{i+1}) - g(x_i)$.

Anderson, ACM 1965

Comparison Problem – CASL AMA Problem 6

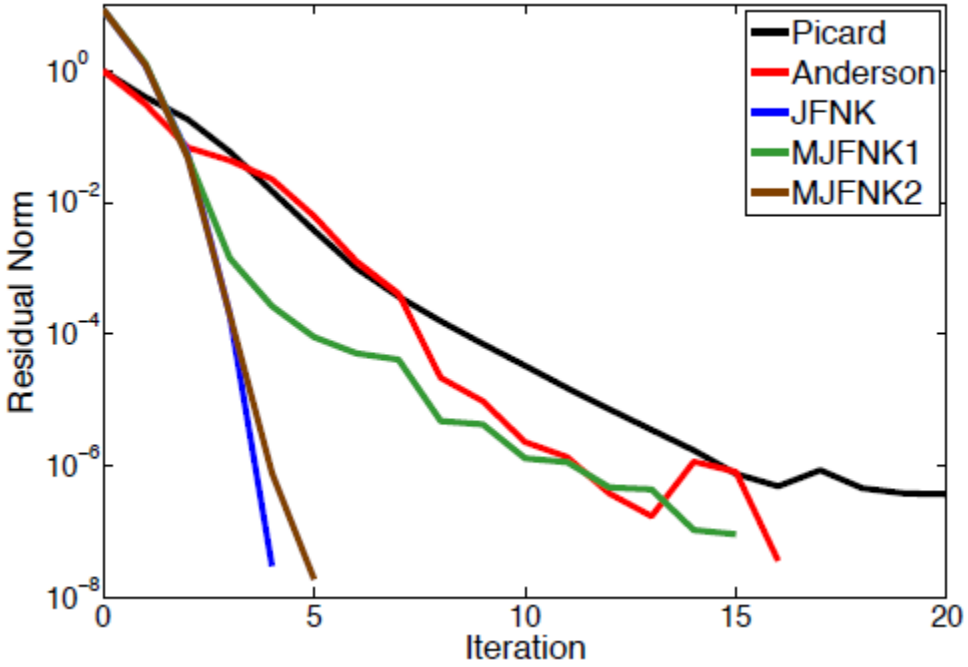
Sandia National Laboratories

- Standard 17x17 PWR Fuel Assembly
  - 264 fuel pins
  - 24 guide tubes, 1 instrumentation tube
  - 1.26 cm pitch, 365 cm active fuel height
- 3.1% enriched UO_2
  - water moderator w/650 ppm boron
  - Zirc-4 clad
- 8 spacer grids, top/bottom nozzle
- 30,000 W/kg operating power

- Finite volume simplified P_N (SP_N) transport
  - SP_3 in angle, P_1 scattering
  - 23 energy groups
  - Cross sections collapsed and homogenized from 56/252 groups on 49 axial levels per fuel pin
  - 290,000 mesh cells on Cartesian mesh
  - 13M DOFs, 524M nonzeros in matrix
- Unstructured mesh CFEM thermal diffusion
  - 4M mesh cells on unstructured hexahedral mesh
- Pin-by-pin subchannel flow model
- Spatially decomposed on 289 cores

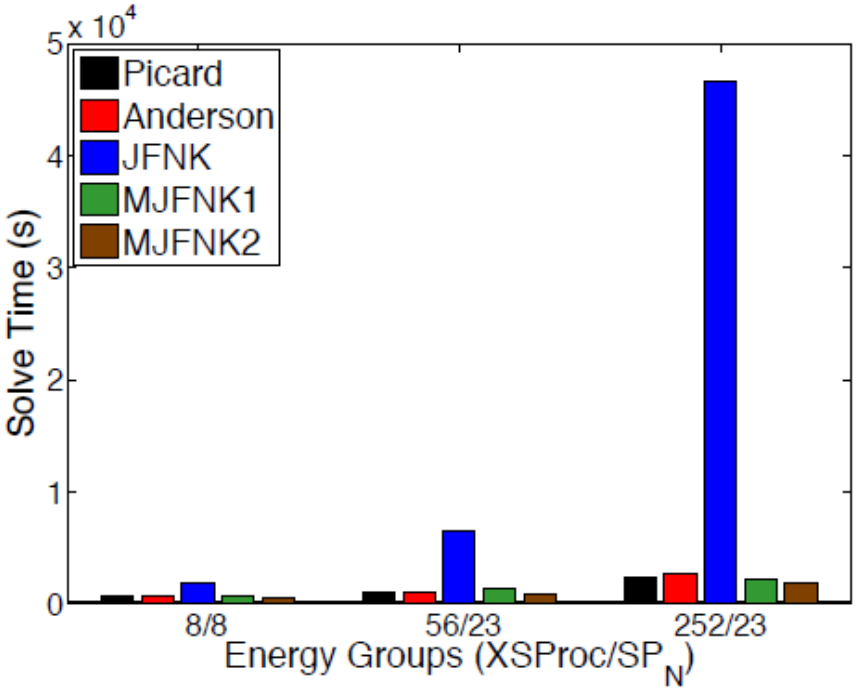
Comparison of Solvers

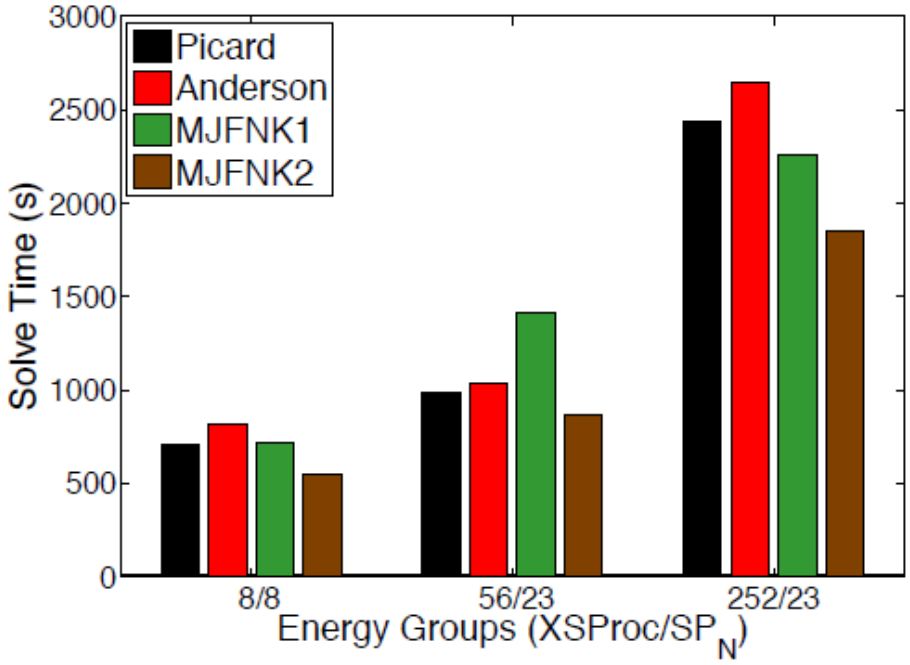
- Watts-Bar Cycle 1 single assembly
- Multiple Solvers:
  - Picard iteration (PIKE)
  - Anderson acceleration (NOX)
  - JFNK (NOX)
  - Modified JFNK 1 & 2 (NOX)
- Direct to steady-state
- Submitted to JCP special issue for CASL



Method	Total	SP_N	Thermal	XSProc
Picard	2437 (9)	81 (296)	470 (296)	1507 (9)
Anderson	2647 (9)	92 (332)	483 (305)	1673 (10)
JFNK	46653 (4)	173 (252)	417 (252)	45472 (252)
MJFNK1	2263 (5)	210 (323)	499 (323)	1003 (6)
MJFNK2	1846 (4)	145 (224)	346 (224)	835 (5)

Total Runtime Comparison

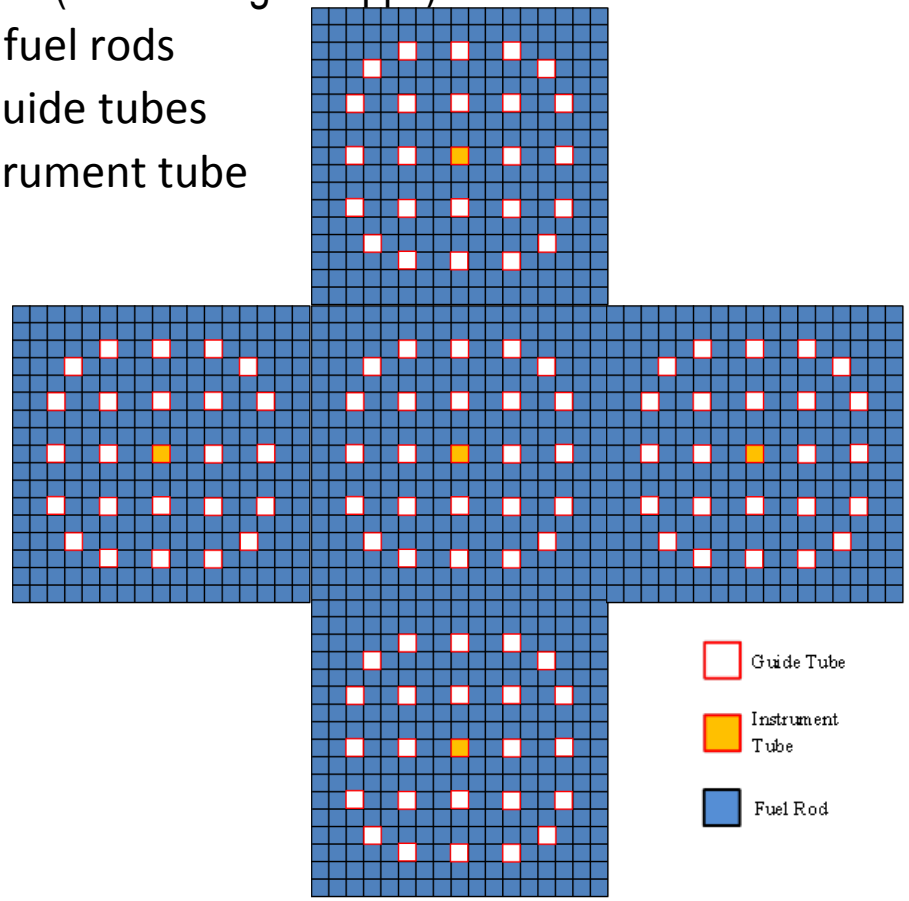




- > 90% of runtime at 252 energy groups is in the online cross section calculation
- Blind application of JFNK was terrible due to cross section recalculation in Jacobian-vector products
- **Better performance of JFNK is not enough to justify the effort needed in refactoring legacy codes (but for new codes JFNK is the preferable from a V&V/UQ standpoint!)**

5-Assembly Cross Results:
3x3 Assembly with 17x17 WEC Assembly
(AMA progression problem 6 and 5-Assebly cross)

- Based on Watts Barr Unit 1 Cycle 1
- **252-group neutronics!**
- Cross Layout (1445 Peregine Apps):
  - 1320 fuel rods
  - 120 guide tubes
  - 5 instrument tube



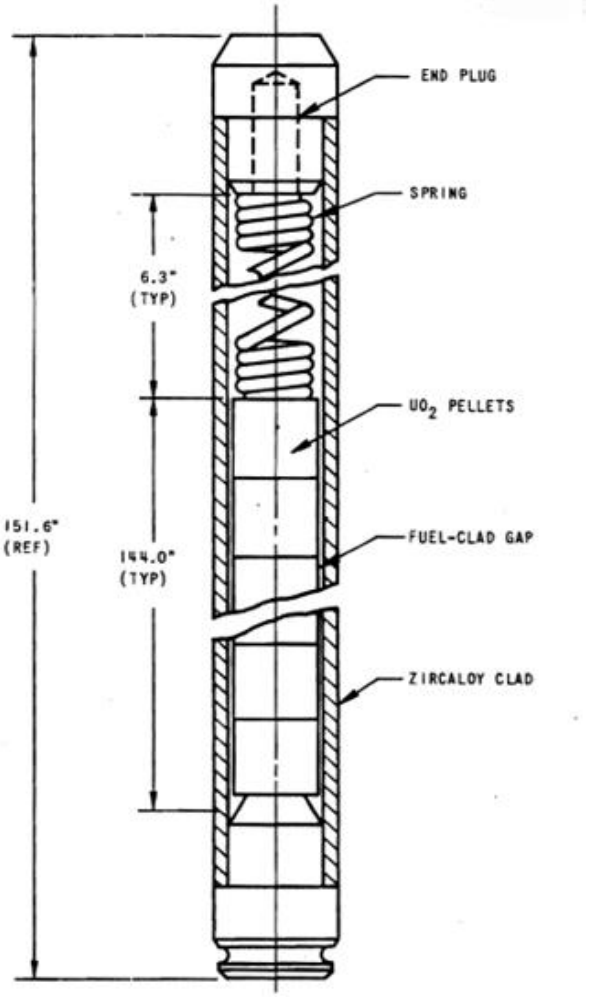


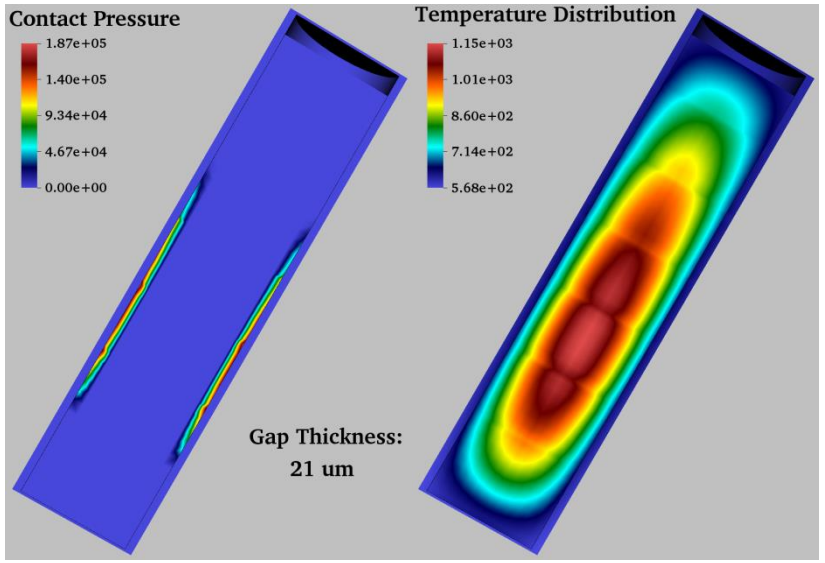
Figure from Watts Bar Unit 2 Final Safety Analysis Report (FSAR), Amendment 93, Section 4, ML091400651, April 30, 2009.
Figure 4.2-3

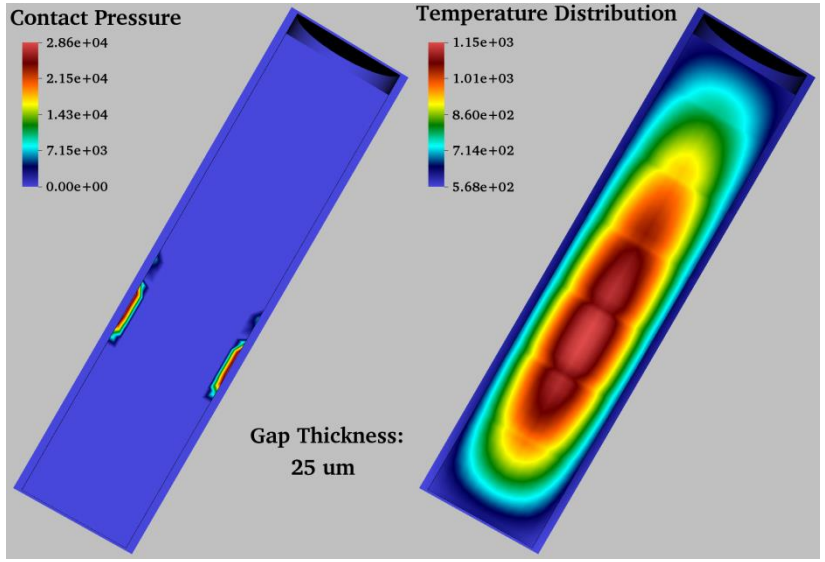
Contact Assessment for PCI

- Changed fuel pellet radius to initiate contact at early times.
- Peregrine (and thus Tiamat) robustly converged through contact events.

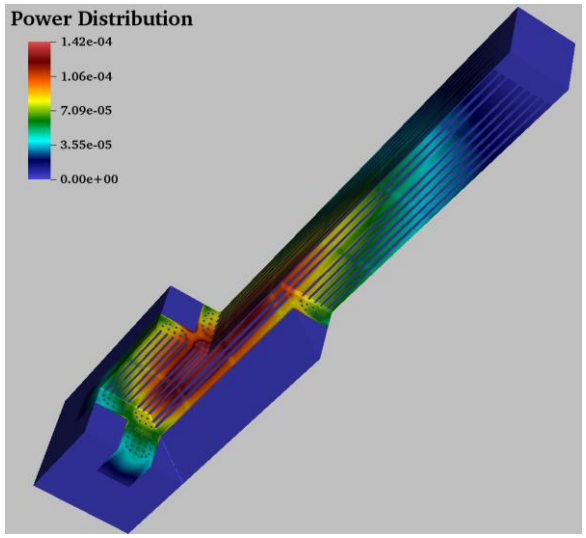
Initial Gap Thickness (um)	Maximum Contact Pressure at HFP	Maximum Fuel Temperature (C)	Simulation Phase at Initial Contact
84	0	1040	Never
42	0	911	Never
26	0	860	HZP->Estimated HFP
25	2.86E+04	859	HZP->Estimated HFP
24	6.37E+04	858	HZP->Estimated HFP
23	1.05E+05	857	HZP->Estimated HFP
22	1.53E+05	857	HZP->Estimated HFP
21	1.87E+05	856	HZP->Estimated HFP
11.5	6.89E+05	856	HZP->Estimated HFP
5.75	9.39E+05	856	CZP->HZP

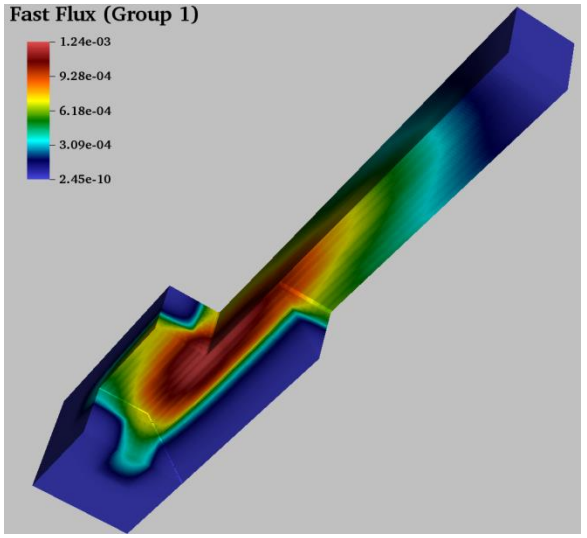
Contact is robust

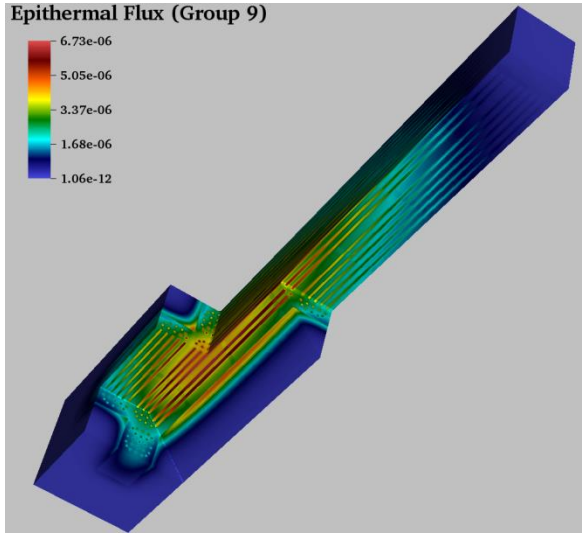


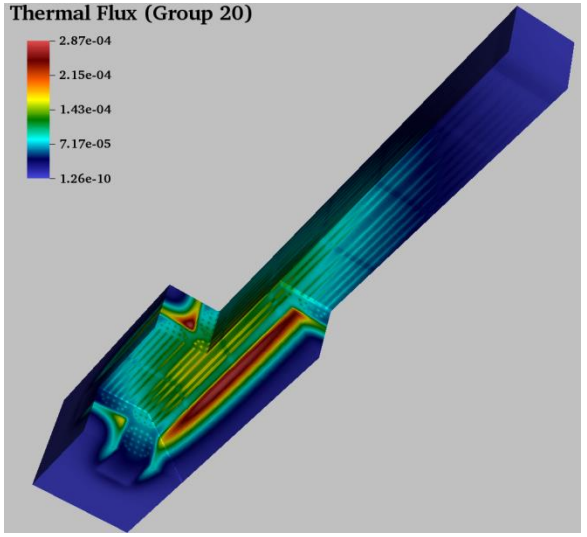


5-Assembly Cross: Power and Flux



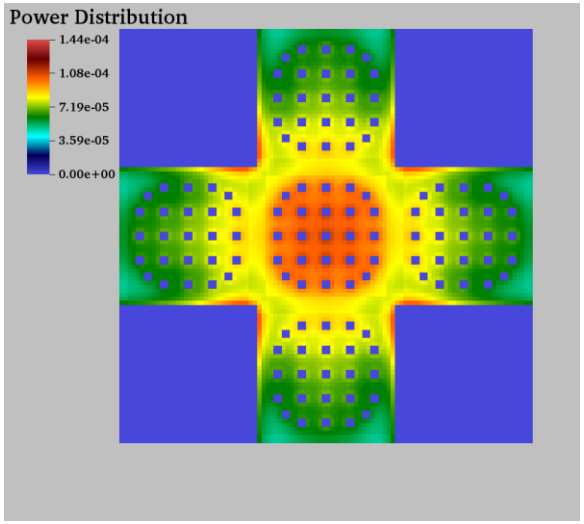


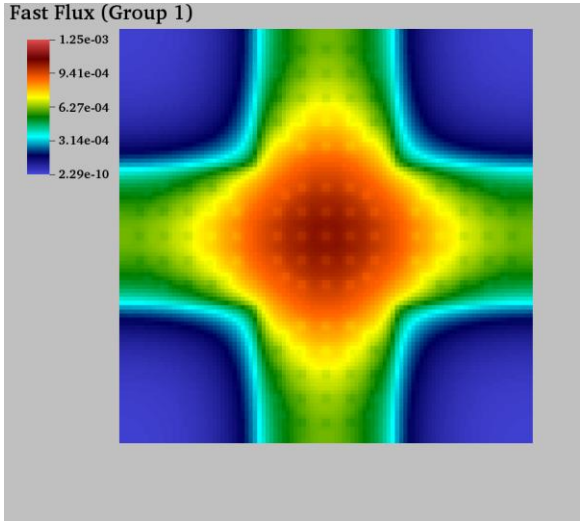


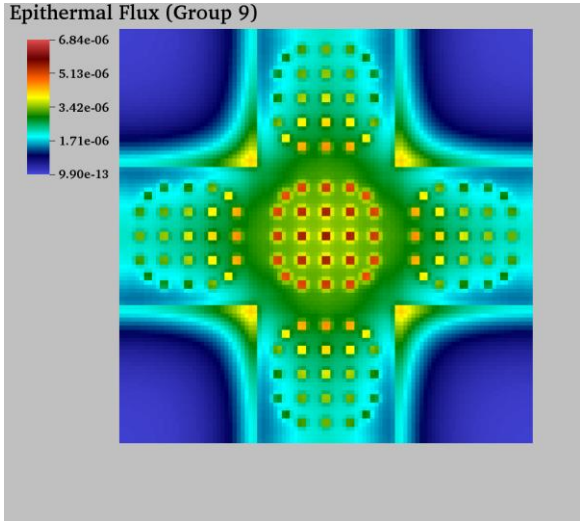


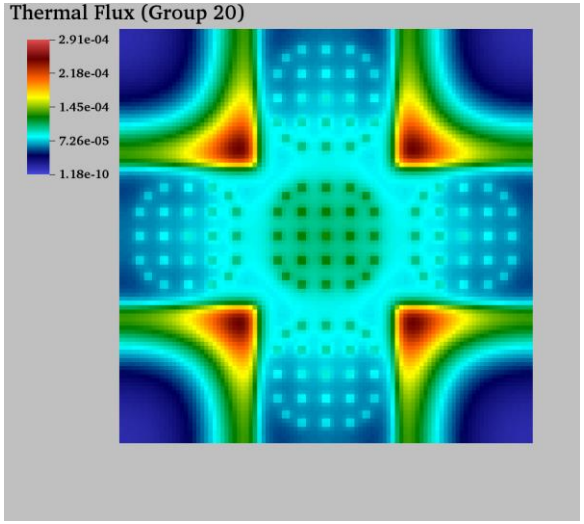
5-Assembly Cross: Power and Flux

Sandia National Laboratories

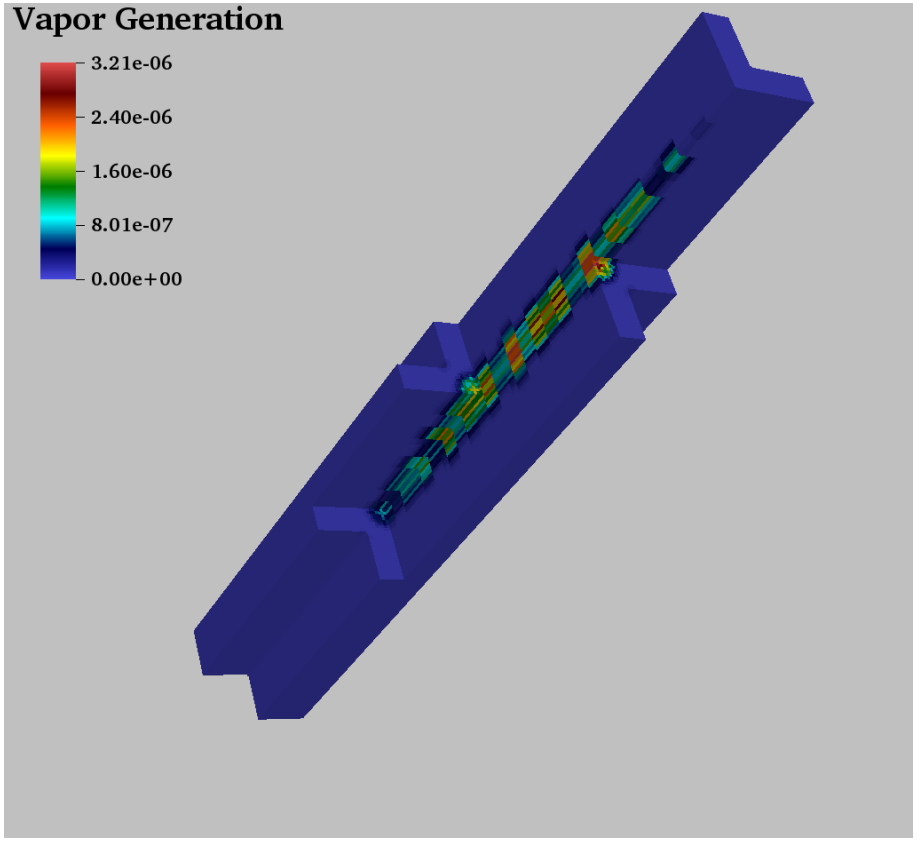


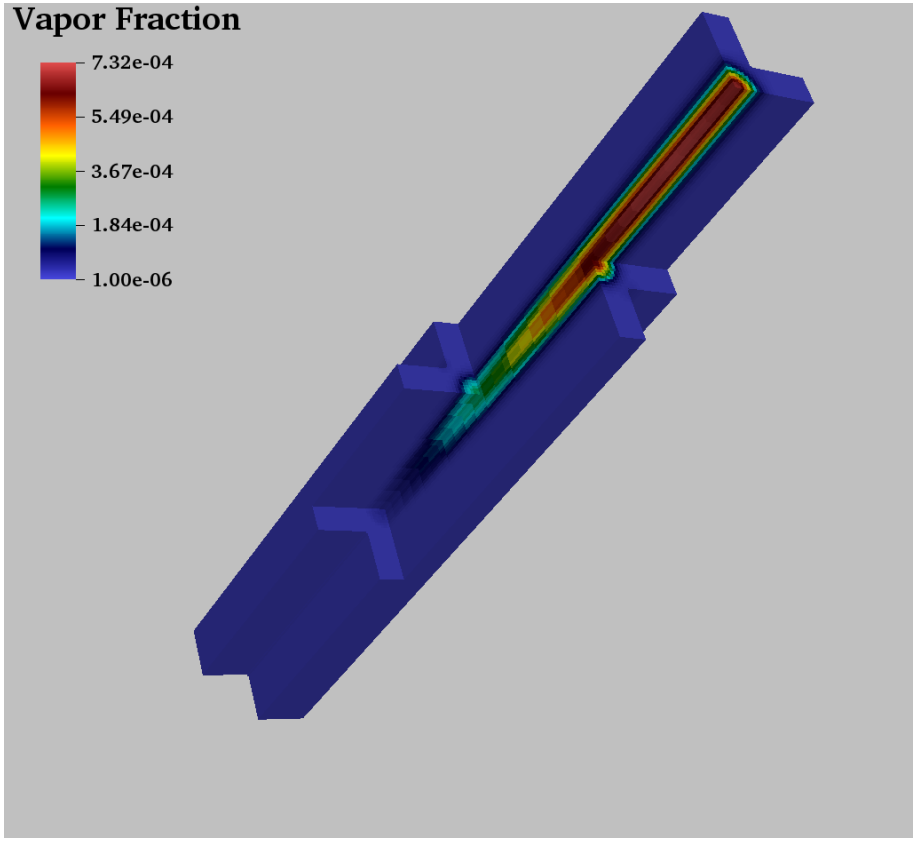






5-Assembly Cross
Vapor Generation and Vapor Fraction

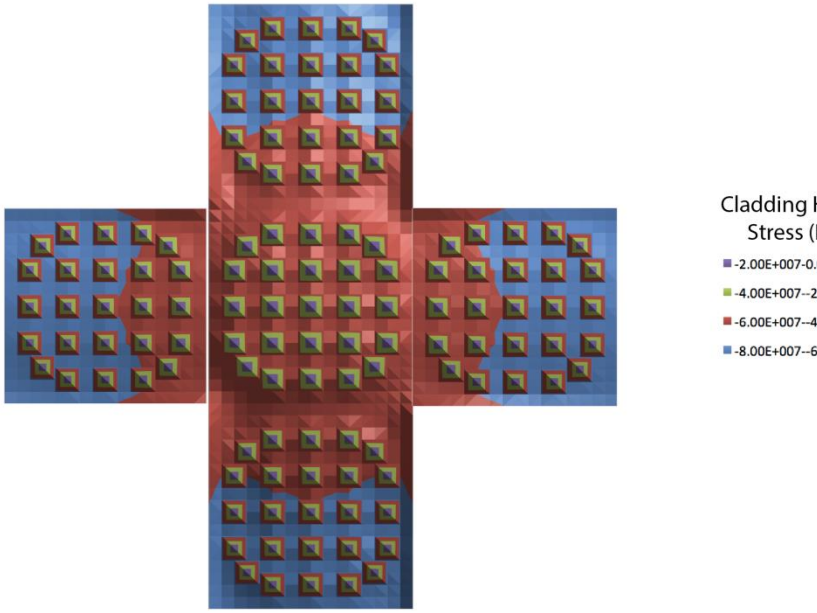


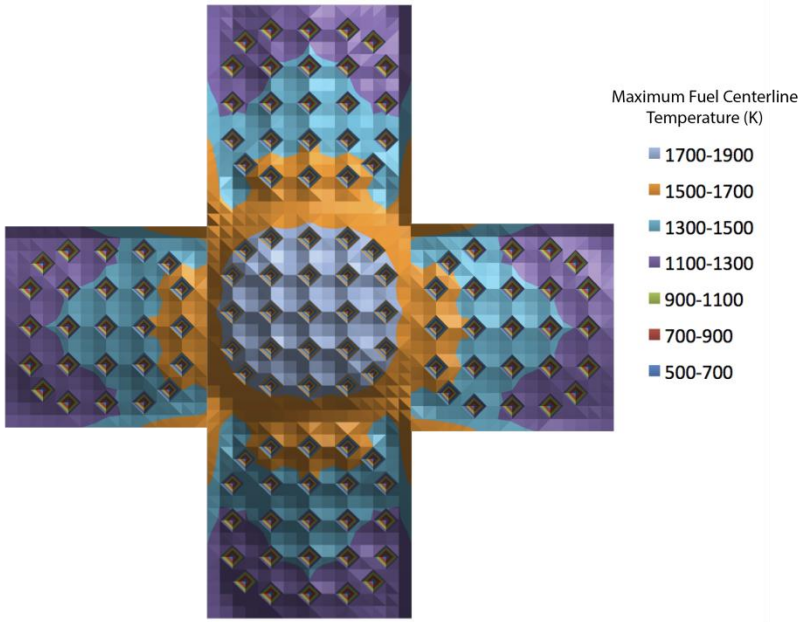


Minimal amount of boiling

PCI Capability Demo

- "A multiple assembly simulation of coupled CTF/Insilico/Peregrine that computes figures of merit for PCI analysis."
- Important features:
  - Fully coupled feedback in each time step
  - Using 252-group neutronics
  - Solid mechanics, w/ history effects
- Significant coordination/collaboration between SNL, ORNL, INL, and PNNL





5-Assembly Cross: Timings

Sandia National Laboratories

- Similar to single assembly layout
  - One MPI core per Peregrine pin
- Timings are dominated by Insilico cross section evaluations

Timings (S)	Tiamat	CTF/Ins
Num Cores	1735	289
Total time	22840	6919
Setup	2200	-
HFP Est/HFP Ramp	3641	-
Solve	16999	-
Num fixed-point Iterations	7	2

Application	Num Cores	Solve Time (s)	Num Solves	Avg/Solve (s)
CTF (HFP Est)	1	697	1	697
CTF (Solve)	1	963	7	138
Insilico (HFP Est)	289	2316	1	2316
Insilico (Solve)	289	15600	7	2229
Peregrine (HFP Ramp)	1445	96.4	12	8
Peregrine (Solve)	1445	94.48	7	14

Transfer	Phase	Time (s)				
		Number	Min	Mean	Max	Mean/Trans
CTF to Insilico	Setup	1	4.40E-01	19.46	1991	19.46
	Transfer	7	1.93E-01	1.93E-01	1.93E-01	2.75E-02
CTF to Peregrine	Setup	1	89.91	89.91	89.91	89.91
	Transfer	7	4.92	4.92	4.92	0.702
Peregrine to CTF	Setup	1	4.67E-01	4.67E-01	4.69E-01	4.67E-01
	Transfer	7	1.90E-03	2.36E-03	4.83E-02	3.38E-04
Insilico to Peregrine	Setup	1	102.3	102.3	102.3	102.3
	Transfer	7	332.9	332.9	333.2	47.56
Peregrine to Insilico	Setup	1	7.24E-02	7.24E-02	7.25E-02	7.24E-02
	Transfer	7	1.38E-03	1.47E-02	1.51E-02	2.10E-03

Summary and Future Work

Sandia National Laboratories

- Black Box coupling is not ideal algorithmically, but is used in many industries
  - Very practical when working with Legacy code!
- Utility of PIKE is to provide a consistent set of interfaces for multiple couplings → reuse of model evaluators and data transfer operators
- Future Work
  - PIKE and DTK will be integrated/snapshotted into the next release of Trilinos
  - PIKE: Finish up TransientSolver support
  - Newton-based Coupling: Thyra::ProductModelEvaluator is under development but not yet completed (currently using AMP for model composite)
  - DTK: Addition/Refactoring of interfaces before Trilinos release
  - DAKOTA interfaces to PIKE
  - Add new solver that mimics SIERRA solution control?
- Questions
  - Trilinos integration: should PIKE be a separate package? NOX?