



## High Fidelity Computational Model for Fluidized Bed Experiments

A. Rodriguez<sup>1</sup>, A. Schiaffino<sup>2</sup>, A. **Chattopadhyay**<sup>2</sup>, V. Kottedda<sup>3</sup>, **V. Kumar**<sup>4</sup> , W. Spotz<sup>5</sup>

<sup>1</sup>UG RA, <sup>2</sup>Graduate Student, <sup>3</sup>Postdoc fellow

<sup>4</sup>Associate Professor, Mechanical Engineering  
The University of Texas, El Paso

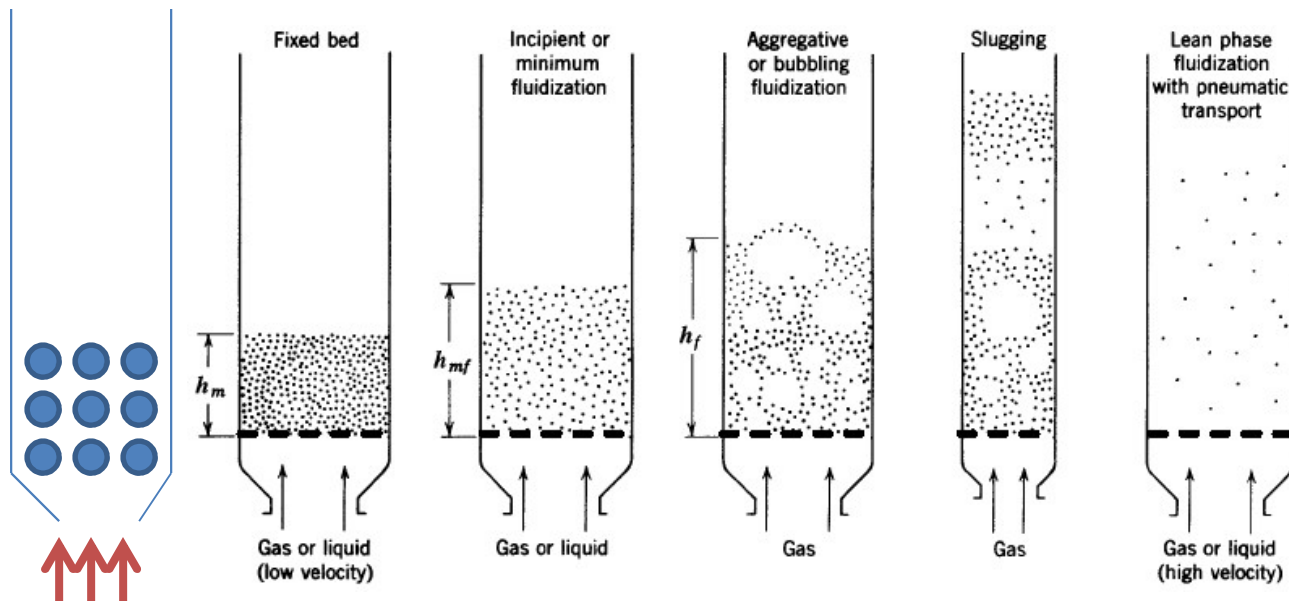
<sup>5</sup>Senior Staff Scientist, Sandia National Labs - NM



# Outline

- Background
- Goals and Objectives
- Project tasks, milestones, and schedule
- Technical approach
- Project status – results and discussions
- Accomplishments
- Concluding remarks

# Fluidized Bed Reactor



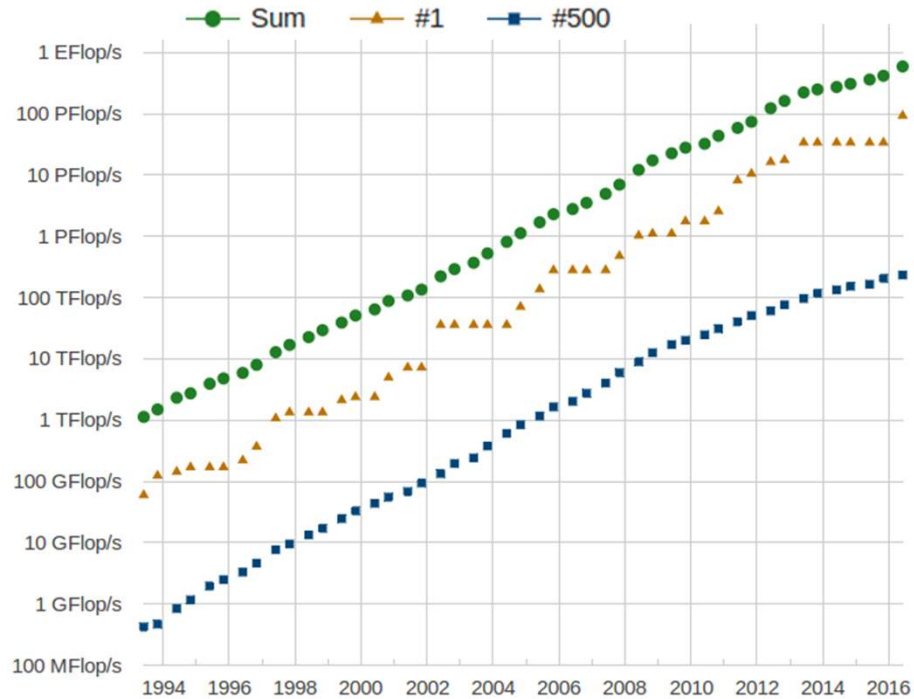
- Governing dynamics is very complicated.
- Multiple phases, interaction between particles and fluid, varying sizes of the particles makes it very difficult to predict the behavior of the bed.

*The arrival time of a space probe traveling to Saturn can be predicted more accurately than the behavior of a fluidized bed chemical reactor! (Geldart, 1986)*



- Greater surface area of contact for fluid and solid allowing for better mixing.
- Applications in process industry and nuclear engineering

# Supercomputers



<https://www.top500.org/>

**Cray X1**  
3 TF  
2004

**Cray XT3**  
Single-core  
26 TF  
2005

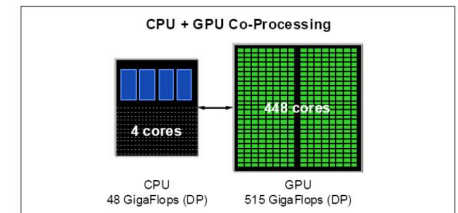
**Cray XT3**  
Dual-core  
54 TF  
2006

**Cray XT4**  
119 TF  
2007

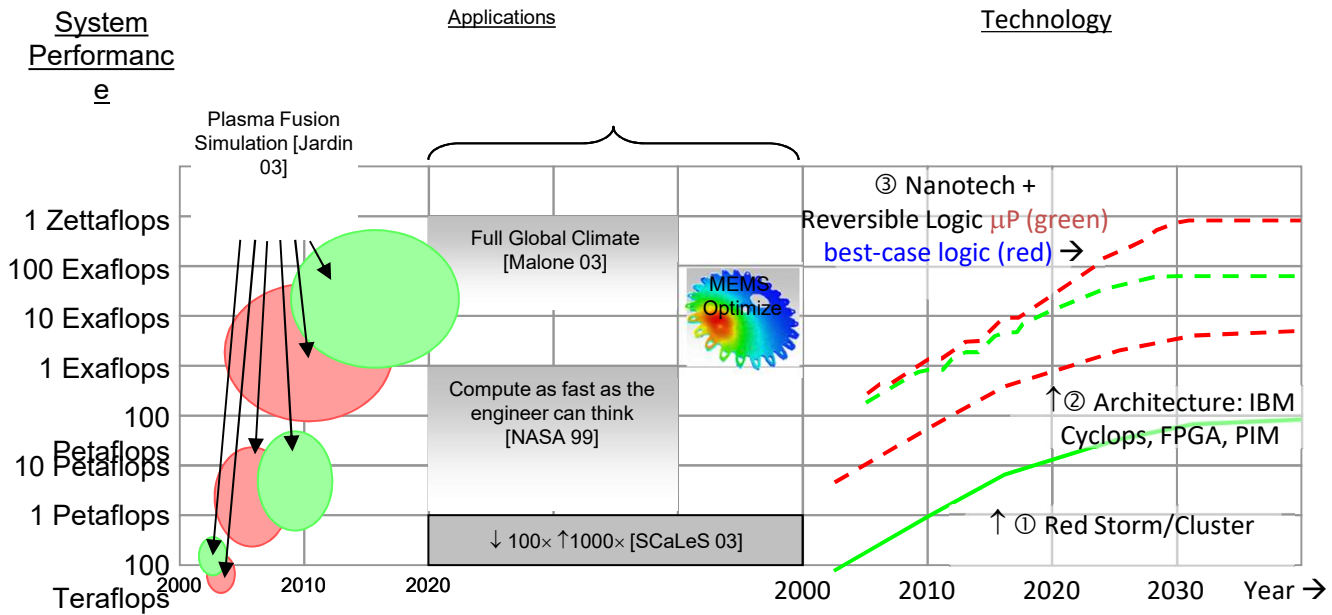
**Cray XT4**  
Quad-core  
263 TF  
2008

**Cray "Baker"**  
6-core, dual-socket SMP  
~1000 TF  
100TB, 2.5PB  
2009

<b>CRAY</b>	<b>NUDT</b>
<b>AMD Opteron</b>	<b>Intel Phi</b>
<b>Nvidia Tesla</b>	<b>54 TF</b>
<b>20 PF</b>	<b>2015</b>
<b>2012</b>	



# Towards Extreme Scale Computing



Slide Credits: DeBenedictis, Erik P, ["Petaflops Exaflops and Zettaflops for Climate Modeling,"](#)

# Why Exa-Scale?

## •Incrementally Advanced Computing

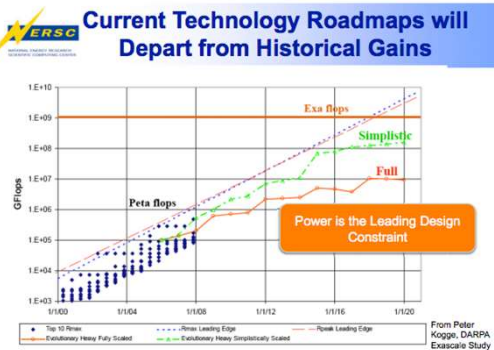
Increasing the computational domain *incrementally* increases resolution of physics.

*Computing a bit better gains results that are a bit better.*- ASCAC Subcommittee on Scale Computing, DOE, 2010

- Parallelism
- Memory
- Power

- Turbulence
- Climate Modeling
- Combustion
- Fluidized Bed Reactor

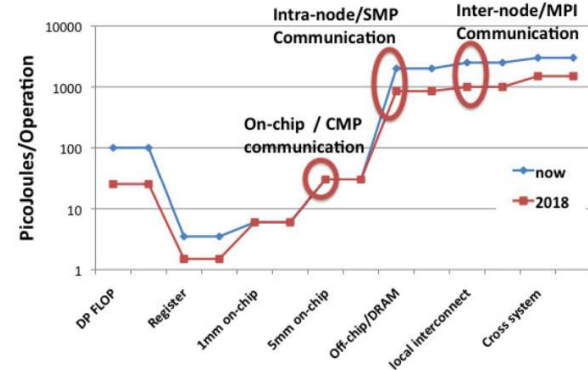
## Challenges



## •Transformational Computing

A “sufficiently large” computational domain resolves newer physics, helps in solving the

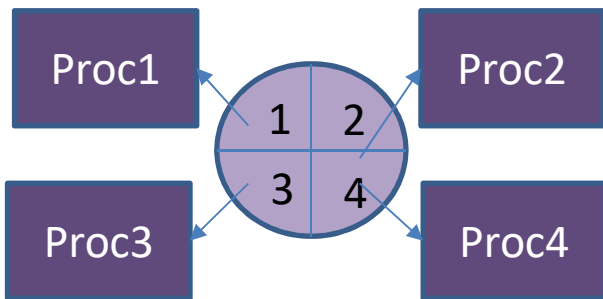
problem almost fully- resolved. - ASCAC Subcommittee on Scale Computing, DOE, 2010



NERSC, Lawrence Berkeley

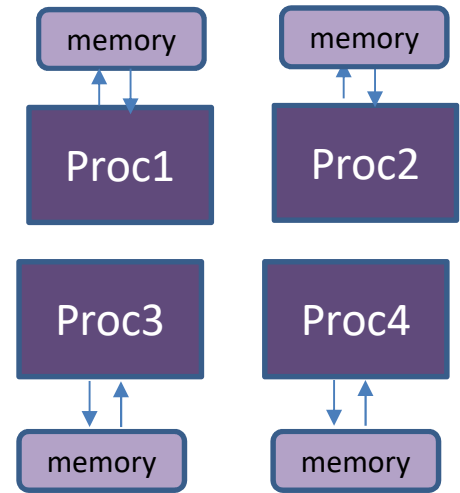
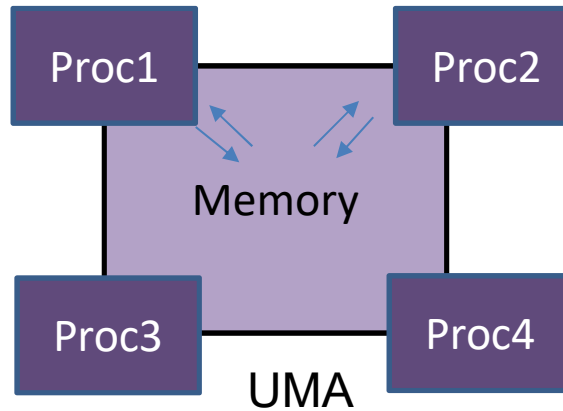
## High Performance Computing Models

Distributed Memory Model

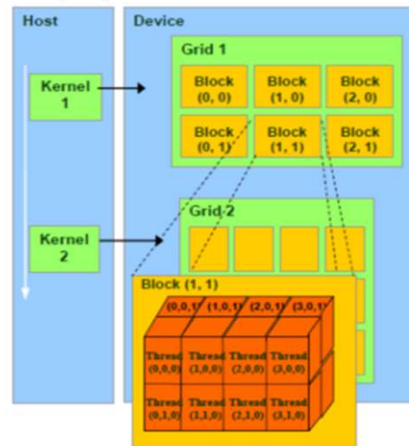


$$S_p = \frac{T_p}{T_s} = \frac{1}{F_s + \frac{F_p}{p} + T}$$

Shared Memory Model

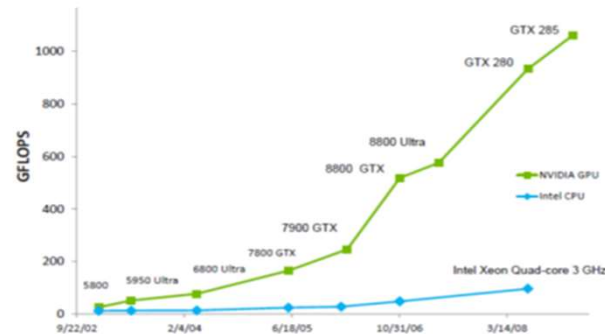


NUMA

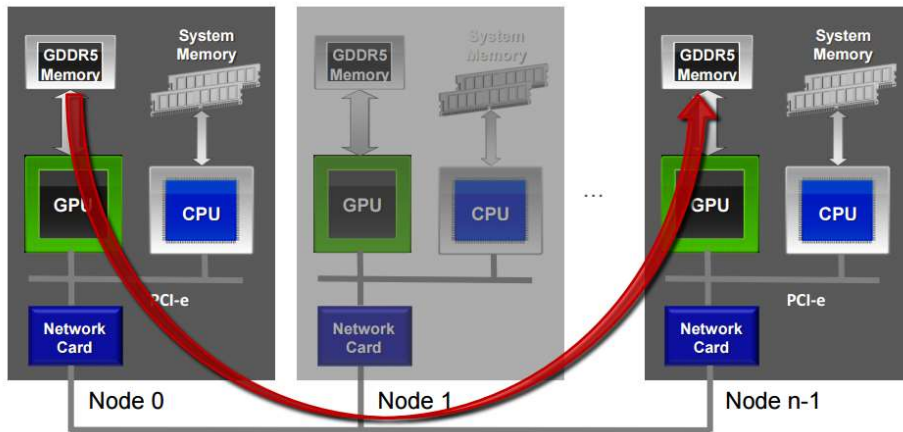


### Accelerators

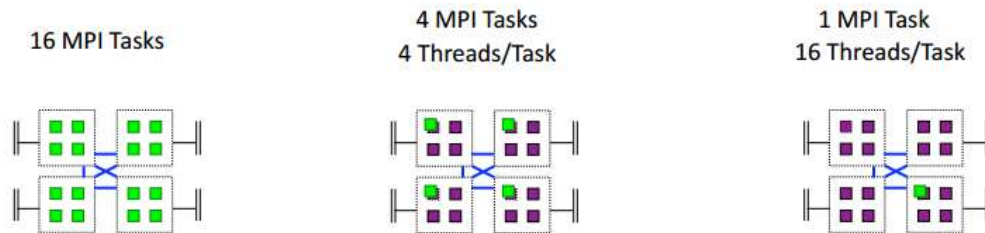
- Light weight computing
- Thousands of threads
- Huge speed up
- Host-Device transfer is the bottleneck



## Hybrid-Parallel Models



- Each MPI node performs fine grained parallelism.
- Thread overhead can be a bottle neck
- In MPI+CUDA data transfer across the PCI bus interconnect slows down computations





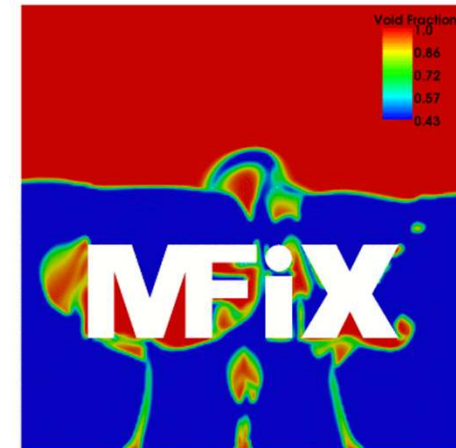
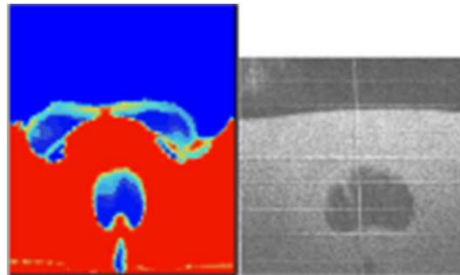
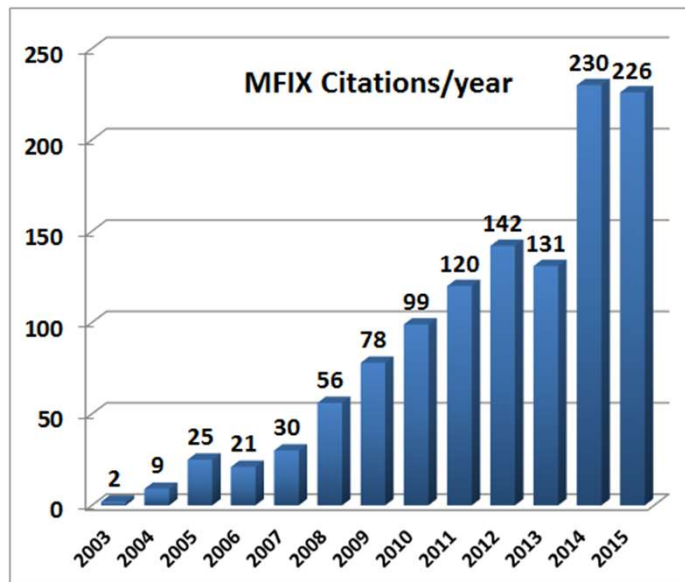
## Multiphase Flow with Interphase eXchange

The Multiphase Flow with Interphase eXchanges (MFiX) software package is

- a multiphase Computational Fluid Dynamics (CFD) software
- developed by NETL (Open-source)
- a legacy code written in Fortran

# Multiphase Flow with Interphase eXchange (MFiX)

- CFD code for modeling reacting multiphase systems.
- 30 years of development history with a wide range of customers.
- Successfully modeling the complexity of fluidized bed reactor flow.



[mfix.netl.doe.gov](http://mfix.netl.doe.gov)

•40% of value added by the U.S chemical industry is related to particle technology- Ennis et al. Chem Eng .Progress 1994.

•Scale up of fluidized beds is a daunting task- Knowlton et al. Powder technology, 2005.

## Multiphase Flow with Interphase eXchange (MFiX)

Gas-solids are addressed by solving

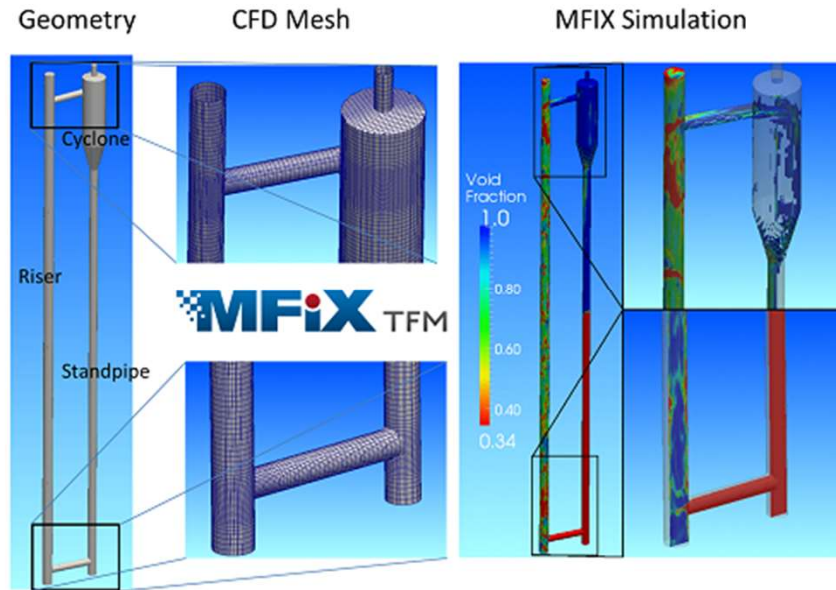
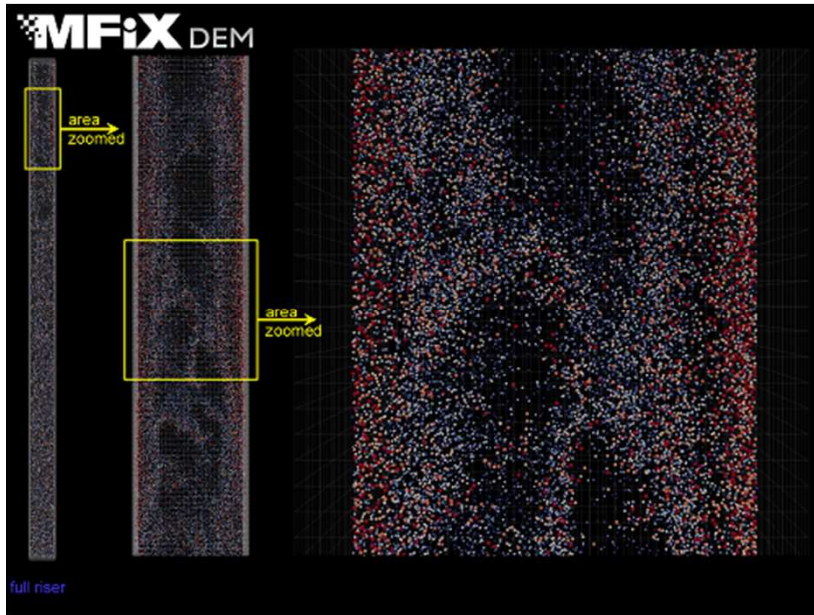
- coupled continuity
- momentum conservation
- energy equations, and
- parameterizing many effects such
  - drag force, buoyancy,
  - virtual mass effect,
  - lift force,
  - Magnus force,
  - Basset force,
  - Faxen force, etc.

Provides a suite of models that treat the carrier phase (gas phase) and disperse phase (solids phase) differently.

- MFiX-TFM (Two-Fluid Model)
- MFiX-DEM (Discrete Element Model)
- MFiX-PIC (Multiphase Particle in Cell)

In nutshell, though MFiX is widely used by the fossil fuel reactor communities to model and understand the multiphase physics in a circulating fluidized, its overall utility of the multiphase models remains **limited** due to the **computational expense** of large scale simulations. The time-to-solution however can be reduced by **leveraging** state-of-the-art **preconditions and linear solver libraries** where majority of processor-level time is spent in solving large systems of linearized equations.

# Challenges with MFiX



$$\frac{\partial}{\partial t} (\varepsilon_g \rho_g) + \nabla \cdot (\varepsilon_g \rho_g \vec{v}_g) = \sum_{n=1}^{N_g} R_{gn}$$

$$\frac{\partial}{\partial t} (\varepsilon_g \rho_g \vec{v}_g) + \nabla \cdot (\varepsilon_g \rho_g \vec{v}_g \vec{v}_g) = \nabla \cdot \bar{S}_g + \varepsilon_g \rho_g \vec{g} - \sum_{m=1}^M \bar{I}_{gm} + \bar{f}_g$$

$$\frac{\partial}{\partial t} (\varepsilon_{sm} \rho_{sm}) + \nabla \cdot (\varepsilon_{sm} \rho_{sm} \vec{v}_{sm}) = \sum_{n=1}^{N_{sm}} R_{smn}$$

$$A x = B$$

- Poor Convergence, especially in complex non-linear problems.
- Increased number of Iterations as a result of poor convergence.
- Basic preconditioners.
- Not as scalable as one would like.

# Technical goal

The technical goal of this project is to develop, validate and implement **advanced linear solvers** to replace the **existing linear solvers** that are used by the National Energy Technology Laboratory's (NETL) open source software package Multiphase Flow with Interphase eXchanges (**MFIX**). This goal will be achieved by integrating **Trilinos**, a publicly available open-source linear equation solver library developed by **Sandia** National Laboratory. The project will **demonstrate scalability** of the Trilinos- MFIX interface on various high-performance computing (HPC) facilities including the ones funded by the Department of Energy (DOE).

The expected results of the project will be **reduction of computational time** when solving complex gas-solid flow and reaction problems in MFIX, and reduction in time and cost of adding new algorithms and physics based models into MFIX

# Objectives

- Create a framework to integrate the existing MFIX linear solver with Trilinos linear solver packages,
- Evaluate the performance of the state-of-the-art preconditions and linear solver libraries in Trilinos with MFIX, and
- Test three dimensional (3D) MFIX suites of problems on massively parallel computers with and without GPU acceleration.

# **Proposed Tasks and subtasks**



# Tasks

## **Task 1.0 – Project Management and Planning**

## **Task 2.0 – Assembly of Optimum Trilinos Linear Equation Package for Integration with MFIX**

Subtask 2.1: Setup a GIT/version-control repository

Subtask 2.2: Select the optimum Trilinos software package

Subtask 2.3: Develop a ForTrilinos based Fortran interface for MFIX

## **Task 3.0 – Performance Evaluation of Preconditions and Linear Solver Libraries**

Subtask 3.1: Test and compare the linear equation solver packages in Trilinos

Subtask 3.2: Perform a scalability analysis of Trilinos-MFIX

Subtask 3.3: Improve performance of Trilinos-MFIX

## **Task 4.0 – Performance Evaluation of MFIX with the Trilinos Linear Solver on Massively Parallel Computers**

Subtask 4.1: Secure computational time on massively parallel computers

Subtask 4.2: Compile Trilinos-MFIX on the selected massively parallel computer(s)

Subtask 4.3: Run simulations of a fluidized bed test problems with various particle sizes and shapes

# Project milestones, budget and schedule

Title	Description	Related task or subtask	Expected Completion Date	Success Criteria	
<b>Budget Year 1:</b>					
Milestone 1.1	GIT repository setup completed	Setup GIT/version-control repository for Trilinos MFIX	Subtask 2.1	Q1	A working repository tested by at least three researchers
Milestone 1.2	Best Trilinos linear solver package decided	Choose the best Trilinos linear solver package for MFIX	Subtask 2.2	Q2	Source code for the decided package uploaded to the GIT repository
Milestone 1.3	Fortran interface for Trilinos MFIX created	Develop Fortran interface for the Trilinos linear solver to communicate with MFIX	Subtask 2.3	Q3-4	A working version of the Fortran interface uploaded
<b>Budget Year 2:</b>					
Milestone 2.1	>30% Linear solved speedup achieved	Test the linear solvers for its performance	Subtask 3.1	Q5	20% or better linear solver speedup achieved
Milestone 2.2	Scalability issues identified	Perform scalability analysis of Trilinos-MFIX	Subtask 3.2	Q6	Scalability testing for up to 1024 cores performed
Milestone 2.3	Bottlenecks to scalability identified and removed	Perform code profiling and identify bottlenecks	Subtask 3.3	Q7-8	Code profiling on Trilinos profiler completed and bottlenecks addressed for one HPC system
<b>Budget Year 3:</b>					
Milestone 3.1	Trilinos MFIX compiled on various OS/architectures	Compile Trilinos-MFIX on various cloud/HPC computers	Subtask 4.3	Q9	Trilinos MFIX compiled on 3 HPC (UTEP, DOE-Sandia, and one more)
Milestone 3.2	MFIX tests suites completed	Run simulations with various particle sizes and shapes of fluidized bed riser test problems	Subtask 4.3	Q10	All 2D runs and one 3D tests validated
Milestone 3.2	Trilinos MFIX performance analysis completed	Analyze Trilinos-MFIX performance for various computing architectures and fluidized-test problems	Subtask 4.3	Q11-12	Report submitted



# Project risks & risk management plan

Risk Category	Description of the Risk	Probability of Occurring	Impact	Overall Level of the Risk	Risk Mitigation Strategy
Internal Data Storage Array	How will the internal data storage arrays (a setpadiagonal matrix) be reconciled with native Trilinos data structures	Med	Low	Low	<ol style="list-style-type: none"> <li>1) Define a class that directly copy the data</li> <li>2) Compressed sparse row matrix storage</li> <li>3) Discussions with experts at Sandia to determine which approach would be best</li> </ol>
ForTrilinos	Trilinos Fortran interfaces	High	Low	Low	<ol style="list-style-type: none"> <li>1) Consider Python as a glue language since it provides automatic wrapper generators.</li> <li>2) Leverag from collaborators' existing PyTrilinos project</li> <li>3) Integrate the fortran interface fully with Stratimikos</li> </ol>
External supercomputer access	Securing access to the external supercomputers	Low	Low	Low	<p>Added as subtask to Secure external resources. Strategies are</p> <ol style="list-style-type: none"> <li>1) Work with Sandia Collaborator to secure DOE's HPC for Trilinos MFX performance testing</li> <li>2) Request for more allocation on TACC HPC through UT System HPC initiatives</li> <li>3) Write proposal to secure computing hours on XSEDE HPC resources</li> <li>4) PI already has access to Linux and IBM clusters through UTEP's HPC facilities</li> </ol>

# Project status

- Setup ~~GIT/version control~~ repository
- Select optimum Trilinos software package
- Develop ForTrilinos based Fortran interface for MFIX
- Test and compare the linear solver packages in Trilinos
  - Run test cases of fluidized bed simulations in MFIX using the various Trilinos linear equations preconditioner. Compare the performance (computing time and accuracy) of the MFIX- Trilinos package with solutions that use MFIX and its existing linear solvers
- Perform scalability analysis of Trilinos-MFIX
  - Conduct scalability tests of the Trilinos-MFIX package on single node and multi-core computer clusters using distributed/shared or in hybrid environment HPC systems. Test multi-core clusters containing 4, 16, 64, 128, 512, 1024, 8192(?) cores.
- Address Trilinos-MFIX performance bottlenecks via profiling and debugging tools in Trilinos
- Run fluidized bed test problems (various particle sizes and shapes, 2D/3D, Small/Large Scale, etc.)

# Technical approach

One of the main challenges for any software development is keeping the computer code up-to-date with the advancement in applied mathematics, software and hardware development in computational science and engineering. Realizing the challenge, the Computer Science Research Institute (**CSRI**) group at Sandia National Laboratories (**Sandia**) has developed and continues to develop scalable solver algorithms and software through **next-gen** (exa-scale, peta-scale, extreme-scale, etc.) computing investment. The project is called **Trilinos** project.



*Funded by various DOE entities mainly NNSA - Advanced Simulation and Computing (ASC)/DOE Office of Science (SciDAC), Advanced Scientific Computing Research (ASCR)*

*Note: Slides in this topic mostly borrowed from M.Heroux & other trilinos members*

# Trilinos

*The Trilinos Project is an effort to develop and implement **robust algorithms** and **enabling technologies** using modern object-oriented software design, while still leveraging the value of established libraries such as **PETSc, Metis/ParMetis, SuperLU, Aztec, the BLAS and LAPACK**. It emphasizes **abstract interfaces** for maximum **flexibility** of component interchanging, and provides a full-featured set of concrete classes that implement all abstract interfaces. Research efforts in **advanced solution algorithms** and **parallel solver** libraries have historically had a large impact on engineering and scientific computing. Algorithmic advances increase the range of tractable problems and reduce the cost of solving existing problems. Well-designed solver libraries provide a mechanism for leveraging solver development across a broad set of applications and minimize the cost of solver integration. Emphasis is required in both new algorithms and new software (Heroux et.al., <http://trilinos.sandia.gov/>).*

# What is Trilinos?

- Object-oriented software framework for...
- Solving big complex science & engineering problems
- More like LEGO™ bricks than Matlab™



**Trilinos** provides the state-of-the-art preconditions and **linear solver** libraries

- demonstrate **scalability** on **current HPC systems**
- illustrate plans for **continued maintenance**
- include **support for new hardware technologies**



# Target Platforms



Desktop: Development and more...

Capability machines:

Redstorm (XT3), Clusters

Roadrunner (Cell-based).

**Multicore nodes.**

Parallel software environments:

MPI

UPC, CAF, threads, vectors,...

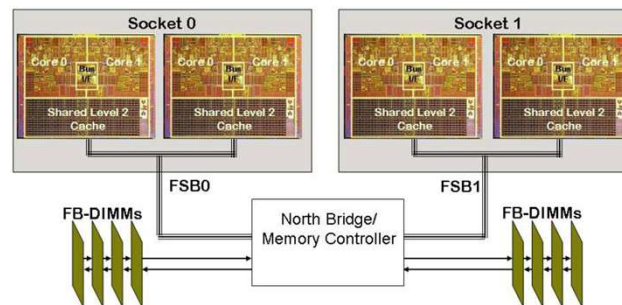
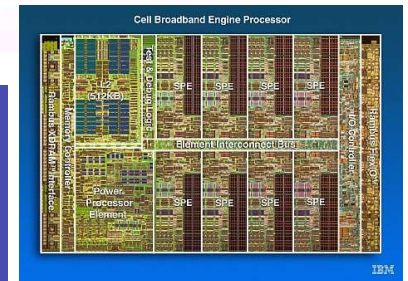
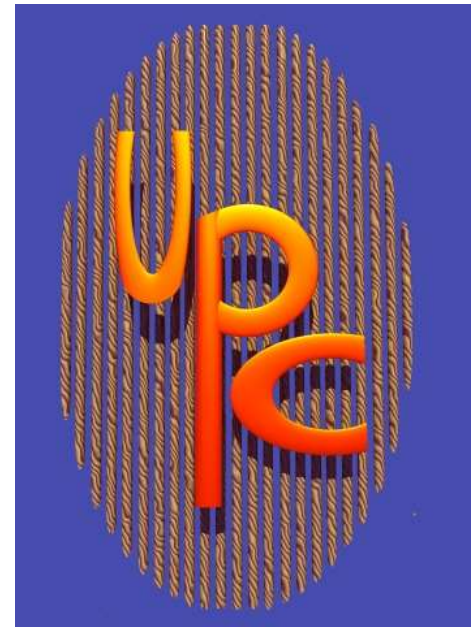
Combinations of the above.

User “skins”:

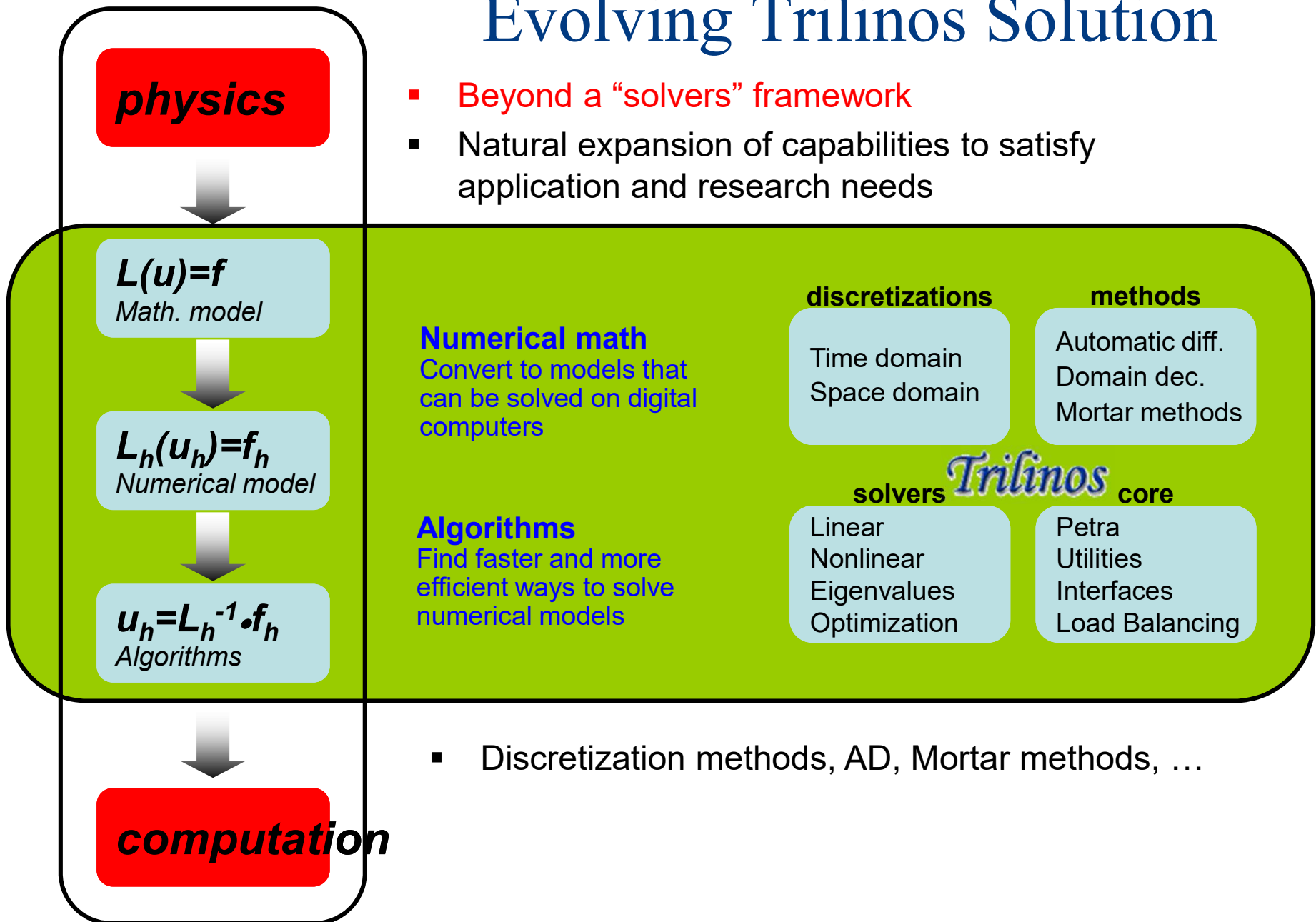
C++/C, Python

Fortran.

**Web**, CCA.



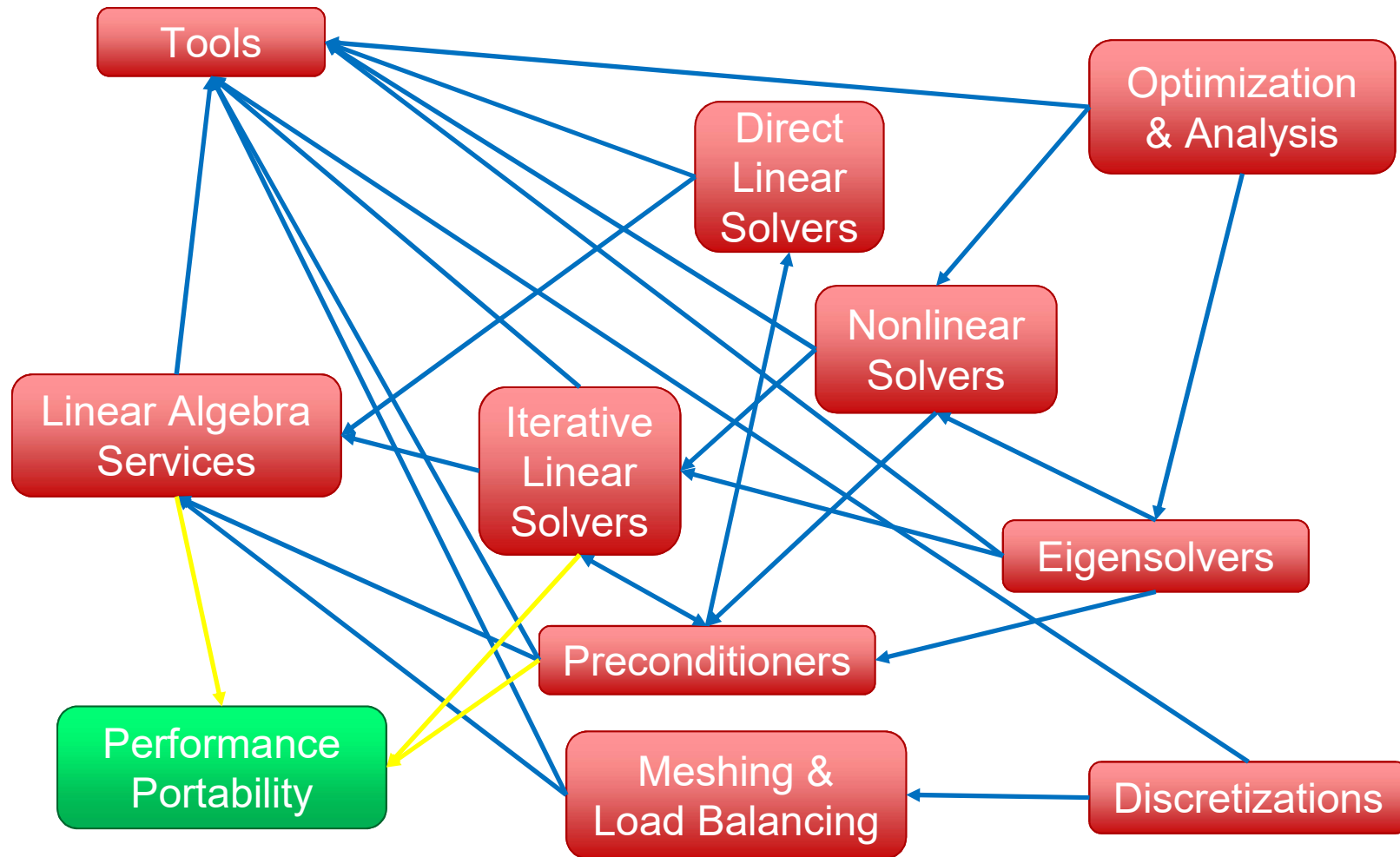
# Evolving Trilinos Solution



- Beyond a “solvers” framework
- Natural expansion of capabilities to satisfy application and research needs

- Discretization methods, AD, Mortar methods, ...

# Trilinos Package categories



# Trilinos Package Advancement

Category	1 <sup>st</sup> Generation	2 <sup>nd</sup> Generation	3 <sup>rd</sup> Gen
Linear Algebra Services	Epetra, EpetraExt, Komplex	Tpetra, Xpetra, Domi, RTOp, Thyra	Tpetra, Xpetra
Tools	Teuchos, Triutils, Galeri, Optika, Trios	Teuchos, Sacado, Trios	
Direct Linear Solvers	Amesos, Pliris	Amesos2	
Iterative Linear Solvers	AztecOO	Belos, Stratimikos	Belos
Preconditioners	IFPACK, ML	IFPACK2, MueLu, ShyLU	IFPACK2, MueLu
Nonlinear Solvers	NOX, LOCA	NOX, LOCA	
Eigensolvers	Anasazi	Anasazi	
Optimization & Analysis	MOOCHO	MOOCHO, OptiPack, Phalanx, Piro, ROL	
Meshing & Load Balance	STK, Zoltan, Isorropia, Mesquite, Moertel	STK, Zoltan2, Pamgen	
Discretizations	Intrepid, Shards, Rythmos	Intrepid, Shards, Tempus	
Performance Portability			Kokkos

# Trilinos Strategic Goals

**Scalable Computations:** As problem size and processor counts increase, the cost of the computation will remain nearly fixed.

**Hardened Computations:** Never fail unless problem essentially intractable, in which case we diagnose and inform the user why the problem fails and provide a reliable measure of error.

**Full Vertical Coverage:** Provide leading edge enabling technologies through the entire technical application software stack: from problem construction, solution, analysis to optimization.

Algorithmic  
Goals

**Grand Universal Interoperability:** All Trilinos packages will be interoperable, so that any combination of packages that makes sense algorithmically will be possible within Trilinos and with compatible external software.

**Universal Accessibility:** All Trilinos capabilities will be available to users of major computing environments: C++, Fortran, Python and the Web, and from the desktop to the latest scalable systems.

**Universal Capabilities RAS:** Trilinos will be:

The leading edge hardened, efficient, scalable solution for each of these applications (**Reliability**).

Integrated into every major application at Sandia (**Availability**).

Easy to maintain and upgrade within the application environment (**Serviceability**).

Software  
Goals

C++ Objected  
Oriented  
Framework

Parallel/Distribut  
ed Templated  
classes for matrix  
generations



Flexibility in  
terms of  
generic  
programming  
with Tpetra

State-of-The -  
Art Linear and  
Non-Linear  
Solvers

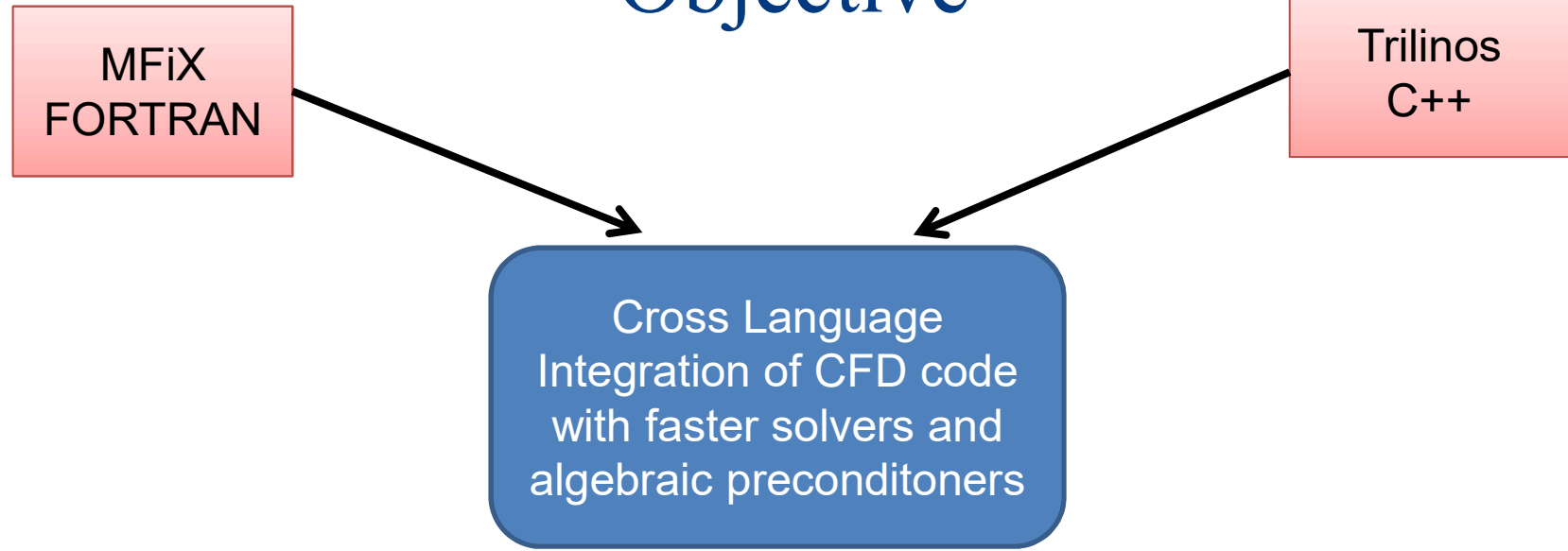
Run time load  
balancing  
between CPU  
and GPU via  
Kokkos

Parallel Data  
Decomposition  
data structures  
*Map*

Native support for  
exascale(>2  
billion unknowns)  
computing

Hardware  
Exploitation for  
large scale  
number  
crunching

# Objective



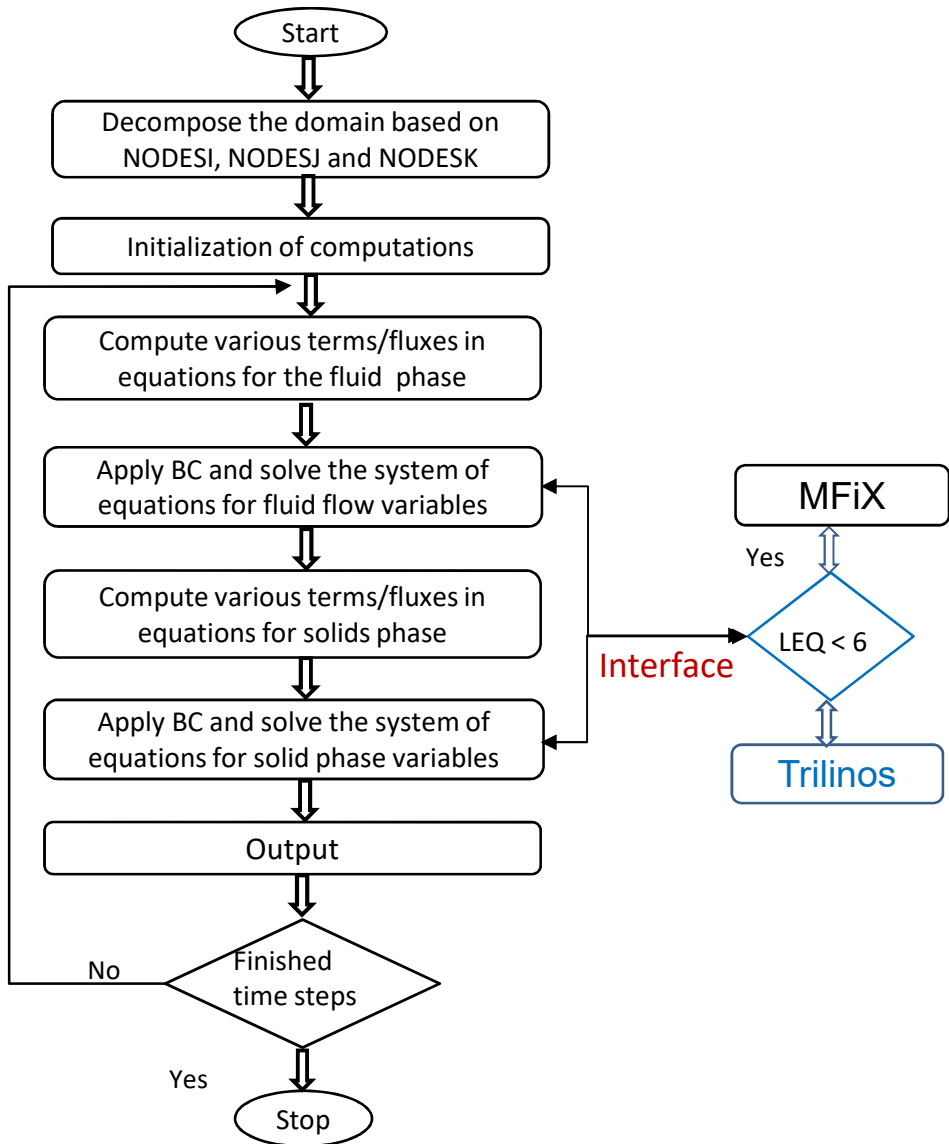
## Advantages

- Exploit legacy codes' expertise in setting up large scale problems
- Use Trilinos as a faster and modern solver platform integrated with legacy codes.

## Challenges

- FORTRAN has no objected oriented framework.
- Trilinos has been developed in C++(objected oriented) framework.
- No semantic support for C++ in FORTRAN.

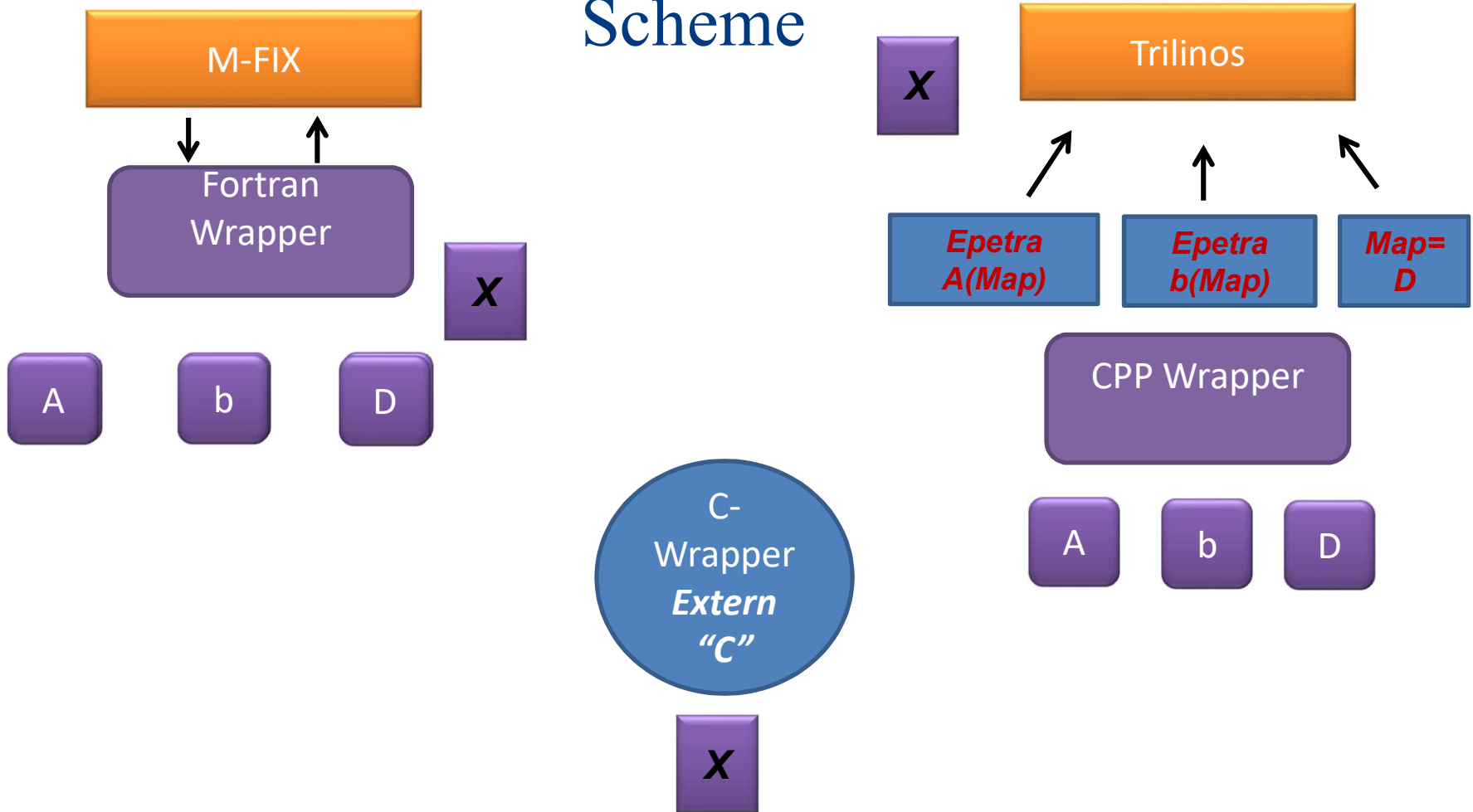
# Flow chart



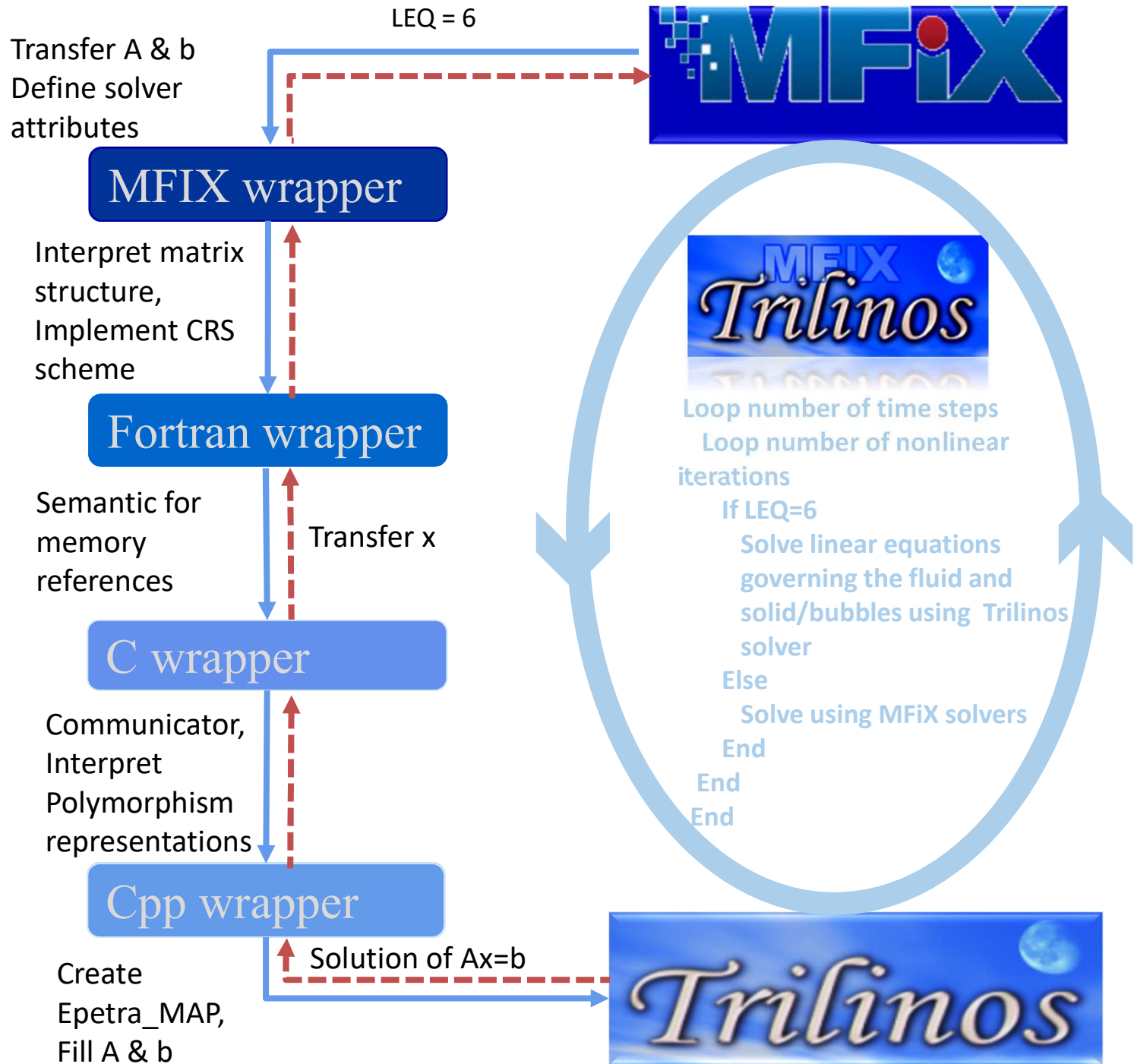
LEQ	Method	Description
1	SOR	Point Successive Over Relaxation
2	BiCGSTAB	Bi-Conjugate Gradient STABILized method
3	GMRES	Generalized Minimal RESidual algorithm
4	BICGSTAB + GMRES	
5	CG	Conjugate Gradient
6	BiCGSTAB/GMRES/CG/Direct/....	Trilinos



# Cross Language Integration Scheme



A language independent interface to  
integrate legacy codes



# MFIX Wrapper

```
nstart = istart;      nend   = iend

ie = 0
do k = kstart, kend
  do i = nstart, nend
    do j = jstart, jend
      IJK = funijk(i, j, k)
      IJK_GL = funijk_gl(i, j, k)
      ie = ie + 1
      Anew(ie, 1) = A_M(IJK, -3)
      Anew(ie, 2) = A_M(IJK, -1)
      Anew(ie, 3) = A_M(IJK, -2)
      Anew(ie, 4) = A_M(IJK, 0)
      Anew(ie, 5) = A_M(IJK, 2)
      Anew(ie, 6) = A_M(IJK, 1)
      Anew(ie, 7) = A_M(IJK, 3)
      Bn(ie)      = B_M(IJK)
      locgl(ie)   = IJK_GL
      pos(ie, 1)  = KM_OF(IJK) - IJK
      pos(ie, 2)  = IM_OF(IJK) - IJK
      pos(ie, 3)  = JM_OF(IJK) - IJK
      pos(ie, 4)  = JP_OF(IJK) - IJK
      pos(ie, 5)  = IP_OF(IJK) - IJK
      pos(ie, 6)  = KP_OF(IJK) - IJK
    enddo
  enddo
enddo
```

# Cpp Wrapper

```
Tpetra_Map map(numGlobalElements, numMyElements, indexBase, comm)

Tpetra_crsMatrix A(map,7);

Tpetra_multivector x (map,1);
Tpetra_multivector b (map,1);

for (LO i = 0; i < static_cast<LO> (numMyElements); ++i) {
    Values = Anew[][];
    Indices = pos[][];
    A->insertGlobalValues (gblRow, NumEntries, Values, Indices);
}
A->fillComplete (map,map);

for (LO i= 0; i < static_cast<LO> (numMyElements); ++i) {
    const GO gblRow = map->getGlobalElement (i);
    b->sumIntoGlobalValue(gblRow, 0, Bn[i]);
}

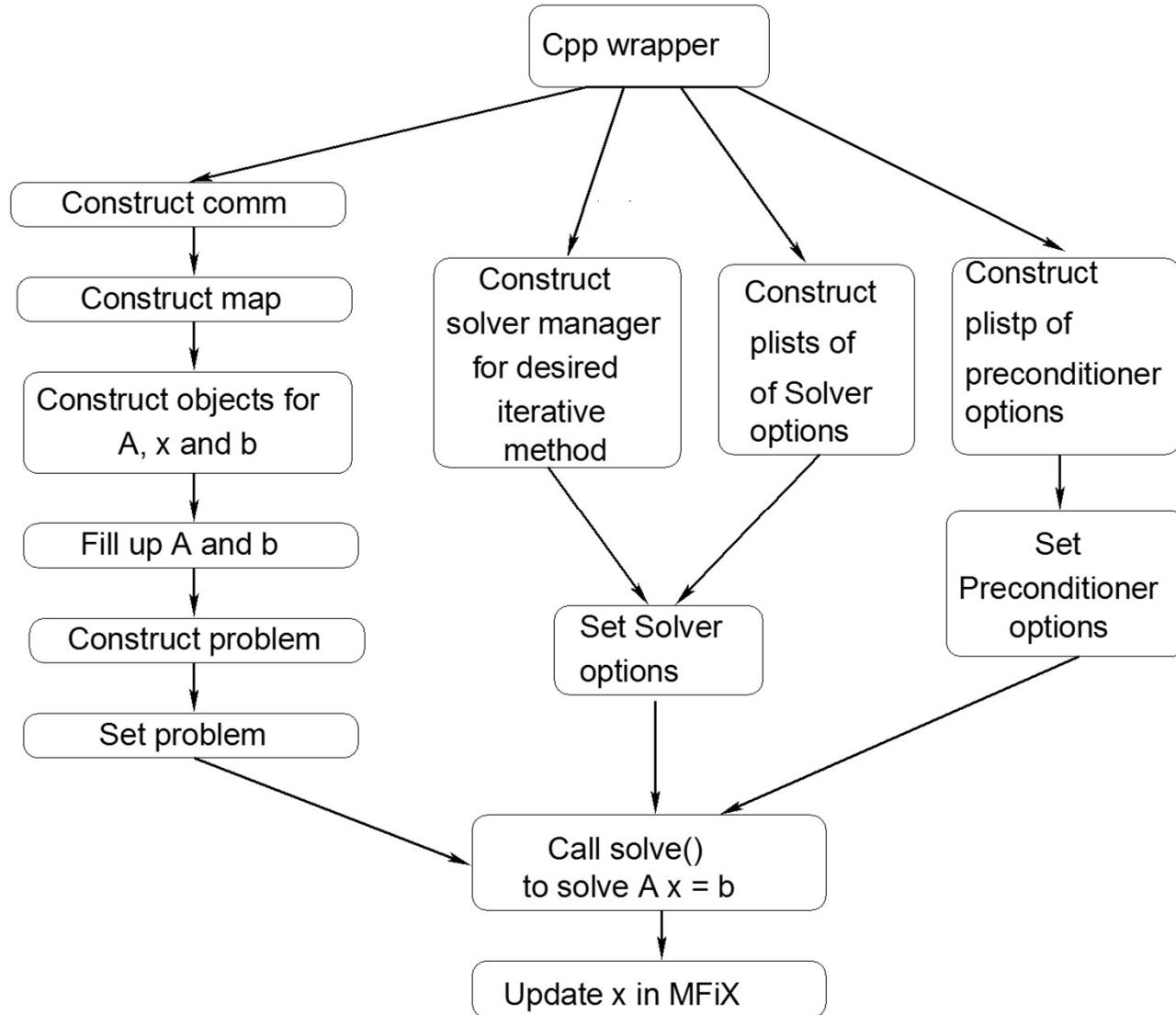
Tpetra_LinearProblem problem(&A, &x, &b);

Problem->setRightPrec (plistp);

belos_bicgstab_manager_type solver(Problem, plists));

solver->solve();
```

# Flow chart of Cpp wrapper



# V&V

Problem:

Bubbling flow

Small size (2D: 10x100, 3D: 10x100x10)

# Gas-solid flow in a 2D bubbling fluidized bed

Cartesian vessel: 10cm length and 100cm height

Sand particle diameter	= 0.04cm
Sand particle density	= 2.0 g/cm <sup>3</sup>
Restitution co-efficient	= 0.80
Angle of internal friction	= 30
The minimum void fraction	= 0.42
Fluid viscosity	= 0.00018 g/cm s
Fluid density	= 0.0012 g/cm <sup>3</sup>

## Boundary conditions

Inlet: constant mass inflow

124.6 cm/s for  $4.3 < x < 5.7$ ; 25.9cm/s for  $0 < x < 4.3$ ,  $5.7 < x < 10$

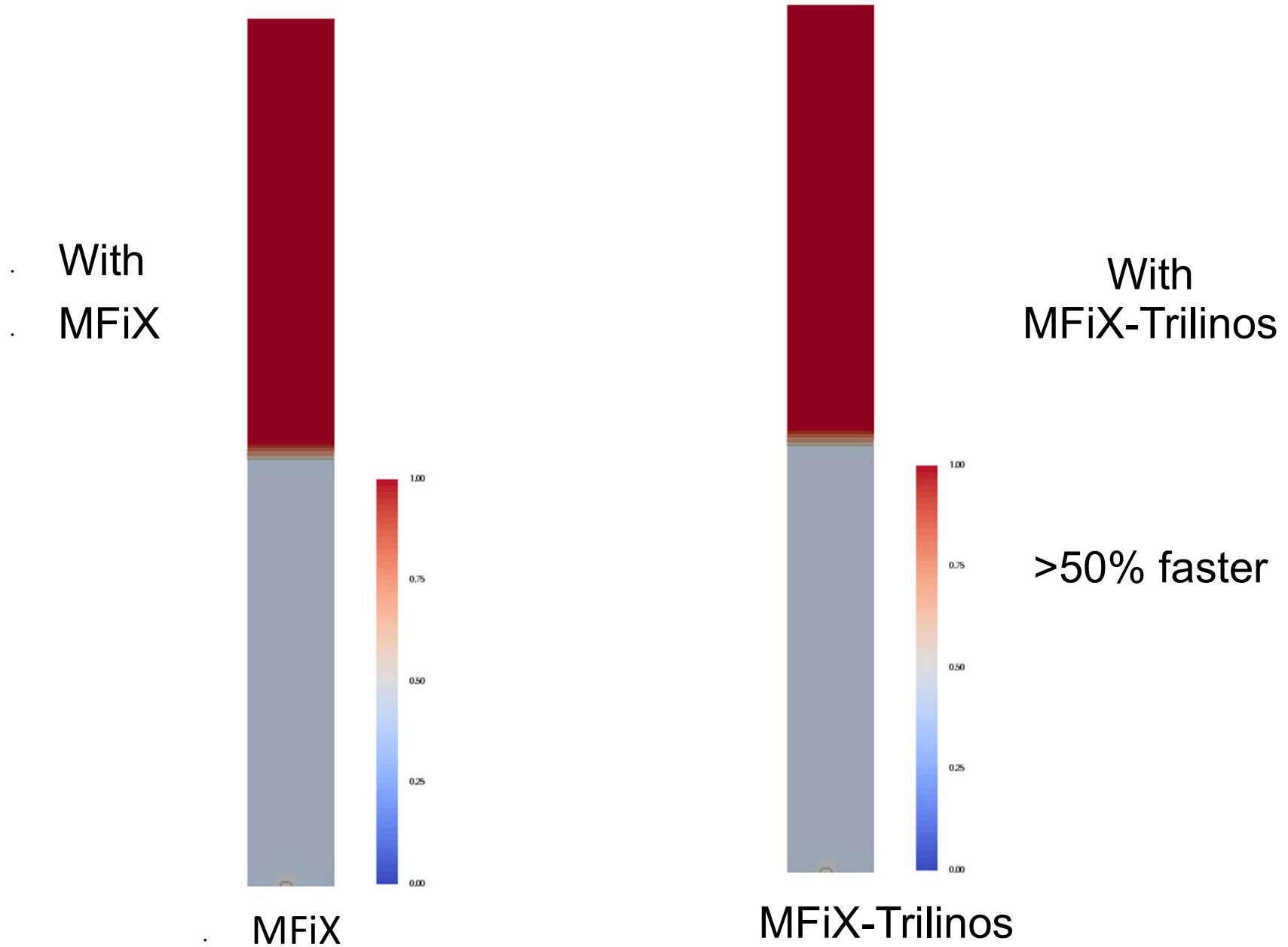
Sidewalls: slip condition

Outlet : pressure outflow condition ( $p = 0$ )

## Mesh

Structured: IMAX = 10, JMAX= 100

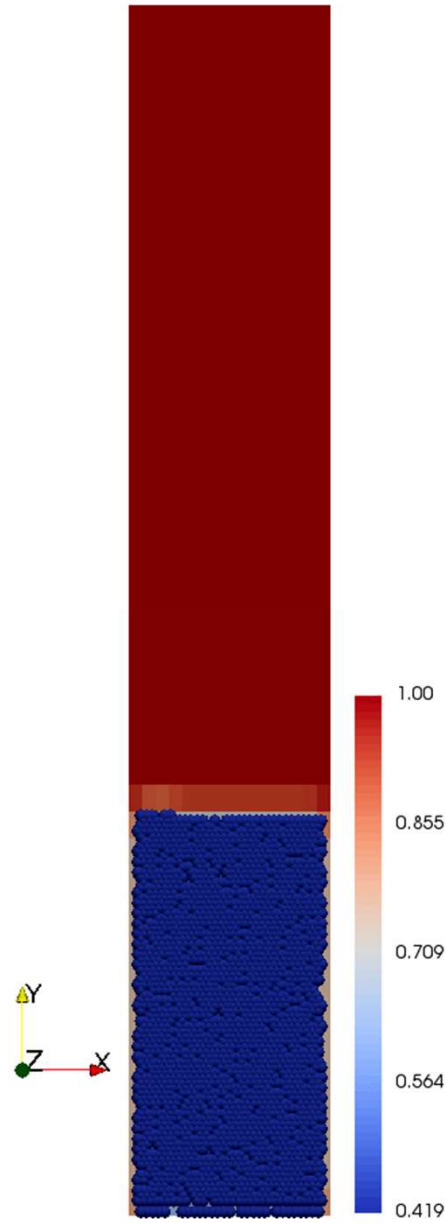
# Fluidized bed 2D (TFM, gas-solid)





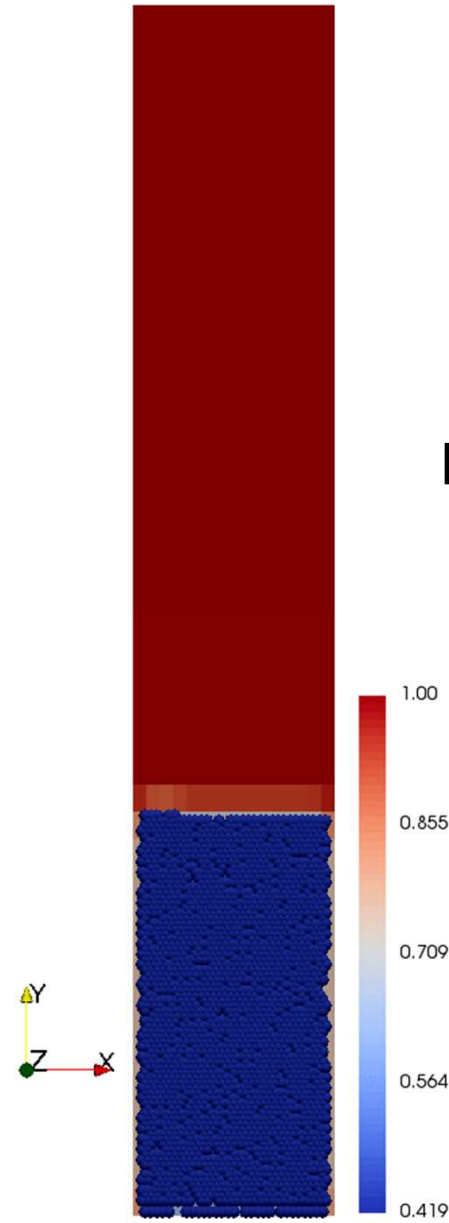
# Fluidized bed , 2D (DEM, gas-solid)

With  
MFiX



MFiX

With  
MFiX-Trilinos



MFiX-Trilinos

>50% faster

# Gas-solid flow in a 3D bubbling fluidized bed

Cartesian vessel: 10cm length, 10cm width and 100cm height.

Sand particle diameter	= 0.04cm
Sand particle density	= 2.0 g/cm <sup>3</sup>
Restitution co-efficient	= 0.80
Angle of internal friction	= 30
The minimum void fraction	= 0.42
Fluid (gas) viscosity	= 0.00018 g/cm s
Fluid density	= 0.0012 g/cm <sup>3</sup>

## Boundary conditions

Inlet: constant mass inflow (124.6 cm/s for  $4.3 < x < 5.7$ ,  $4.3 < z < 5.7$ )

Sidewalls: slip condition

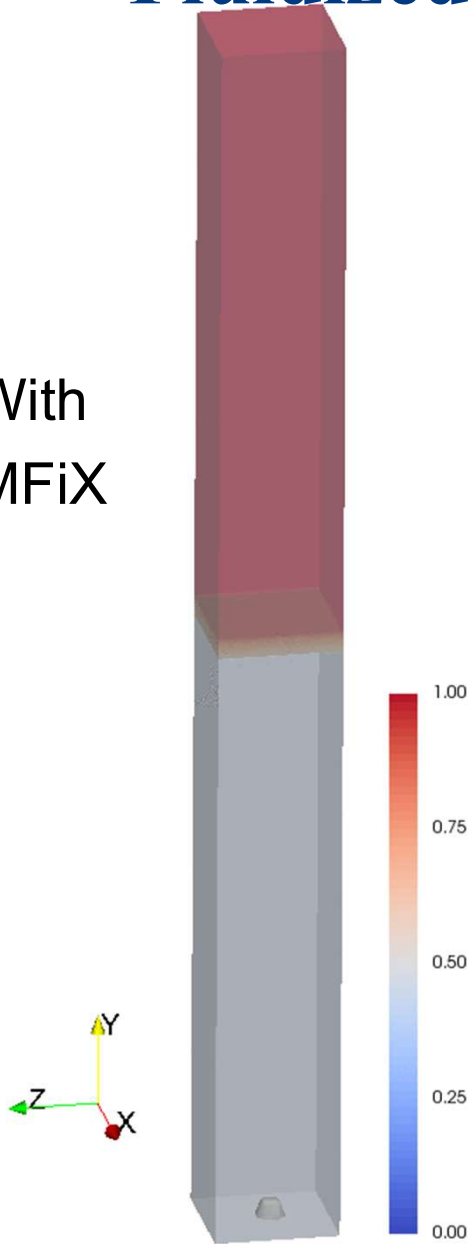
Outlet : pressure outflow condition ( $p = 0$ )

## Mesh

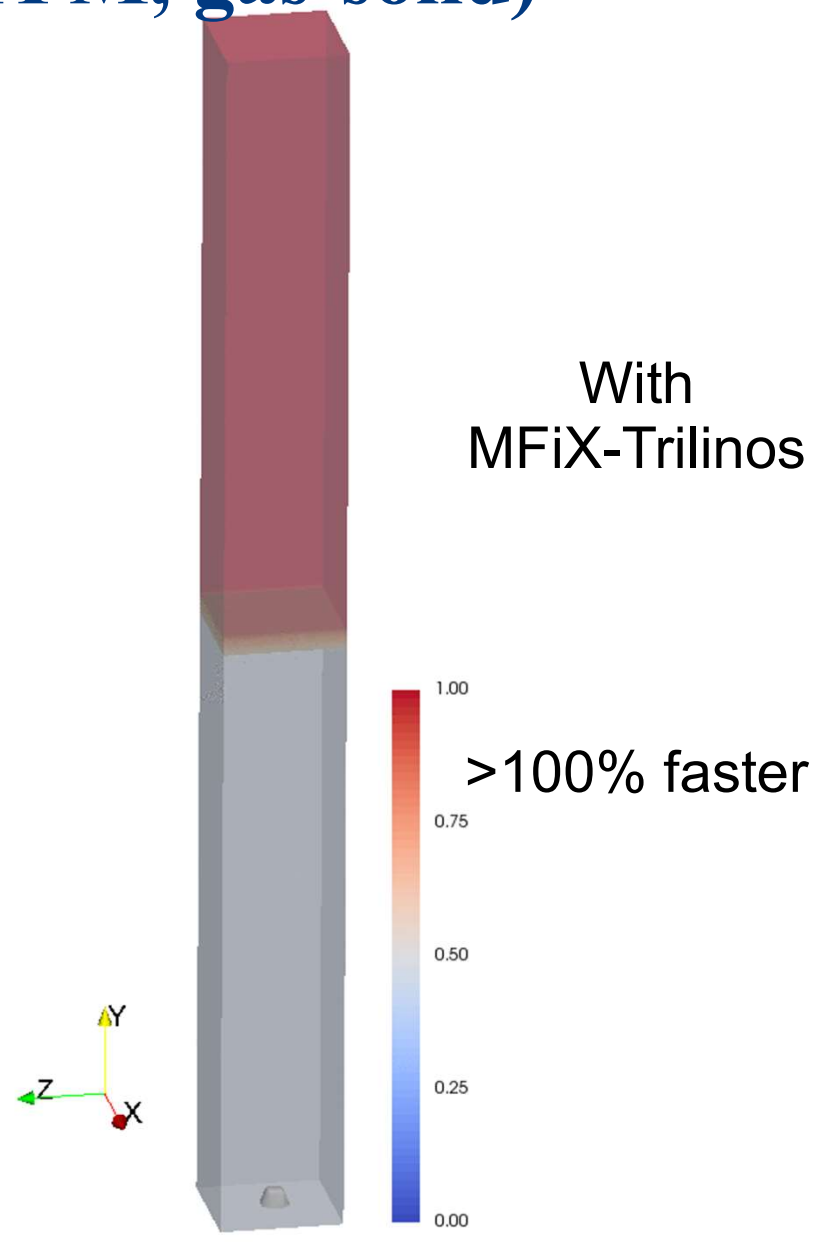
Structured: IMAX = 10, JMAX = 100, KMAX = 10

# Fluidized bed 3D (TFM, gas-solid)

With  
MFiX

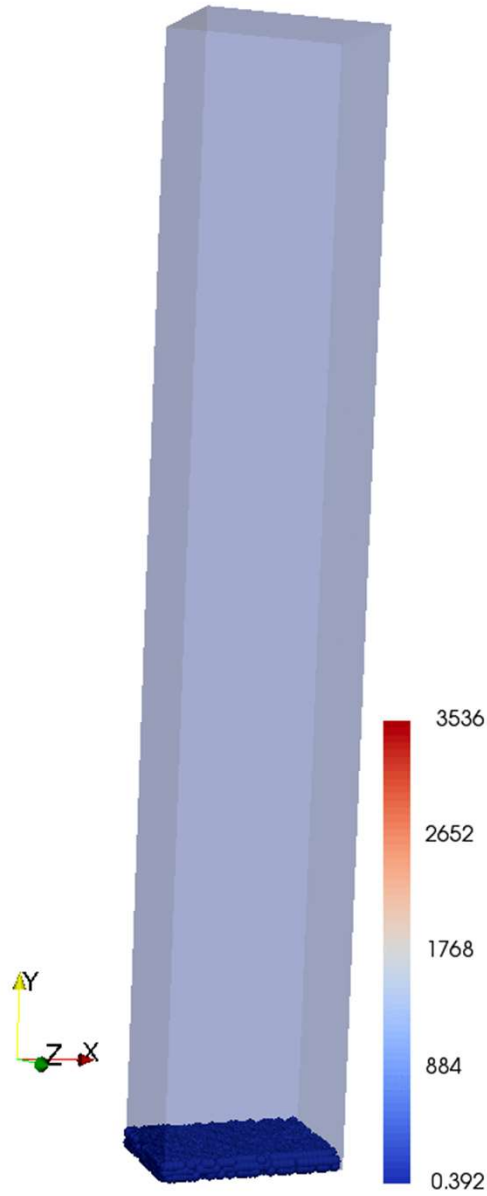


With  
MFiX-Trilinos

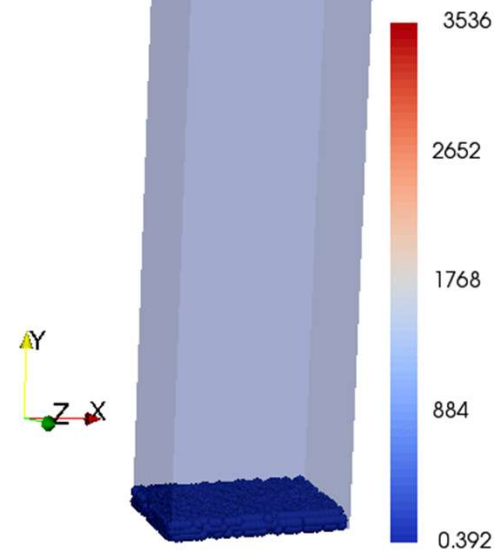


# Fluidized bed 3D (DEM, gas-solid)

With  
MFiX



With  
MFiX-Trilinos



>100% faster

# Gas-solid flow in a circulating fluidized bed

Cylindrical vessel: 10cm diameter 100cm height.

Sand particle diameter	= 0.04cm
Sand particle density	= 2.0 g/cm <sup>3</sup>
Restitution co-efficient	= 0.80
Angle of internal friction	= 0
The minimum void fraction	= 0.42
Fluid (gas) viscosity	= 0.00018 g/cm s
Fluid density	= 0.0012 g/cm <sup>3</sup>

## Boundary conditions

Inlet: constant mass inflow (124.6 cm/s for  $0 < r < 0.7$ )

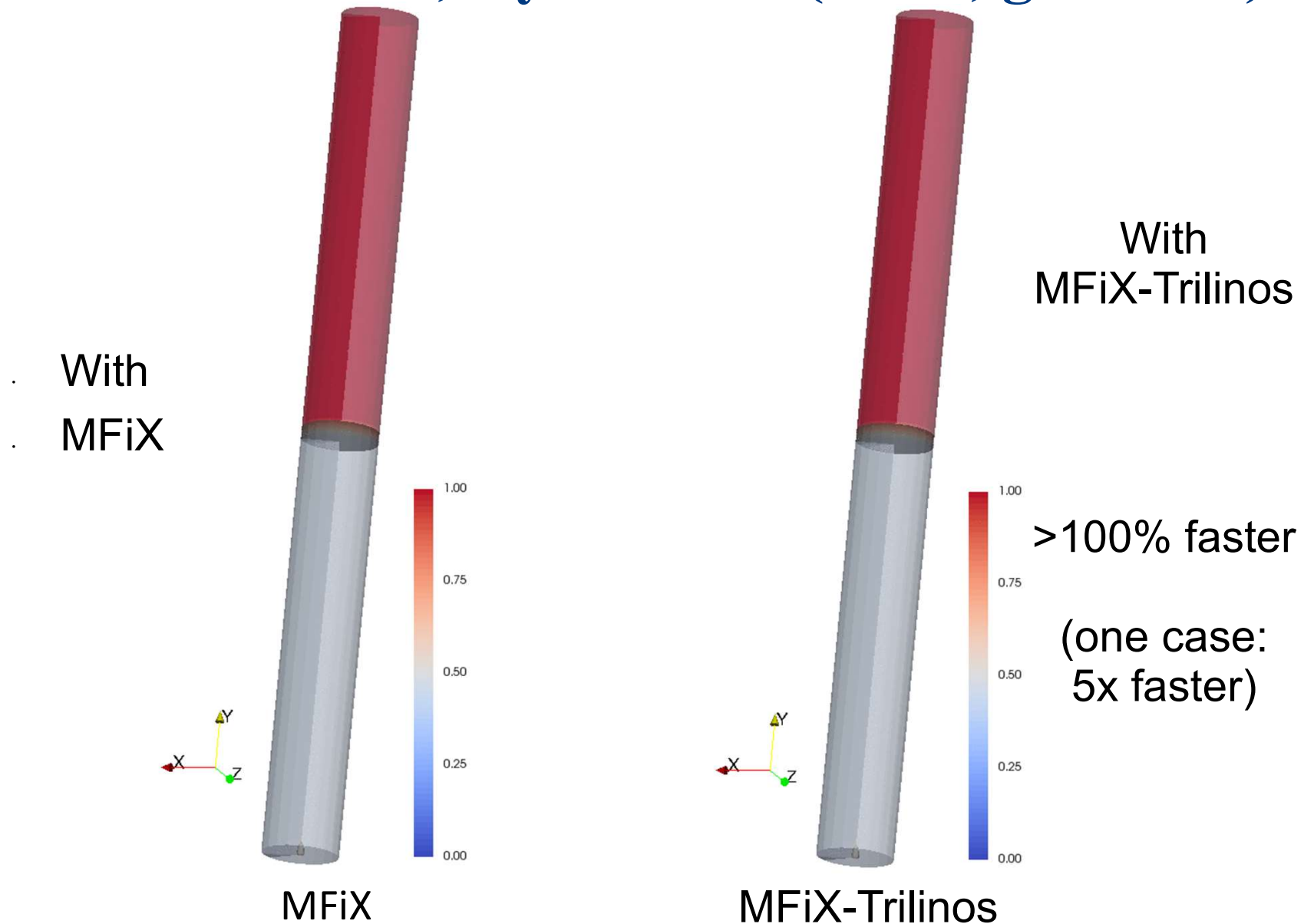
Sidewalls: slip condition

Outlet: pressure outflow condition ( $p = 0$ )

## Mesh

Structured: IMAX = 10, JMAX = 100, KMAX = 10

# Fluidized bed, Cylindrical (TFM, gas-solid)



# Speedup: Mfix-Trilinos v/s MFiX

Problem:

3D Bubbling flow

Larger size

Case 1: Approximate  $n_{\text{dof}} = 10\text{M}$  (100x100x1000)

Case 2: Approximate  $n_{\text{dof}} = 200\text{M}$  (100x500x4000)

Computers:

**Stampede:** Texas Advanced Computing Center (TACC)

**Bridges:** Pittsburgh Supercomputing Center (PSC)

**Comet:** San Diego Supercomputer Center (SDSC)

# Various computer architectures used for the performance analysis study

## 3D Bubbling flow Problem

Case 1: Mesh Size = 10M

Case 2: Mesh size = 200M

## Computers:

**Stampede:** Texas Advanced Computing Center (TACC)

**Bridges:** Pittsburgh Supercomputing Center (PSC)

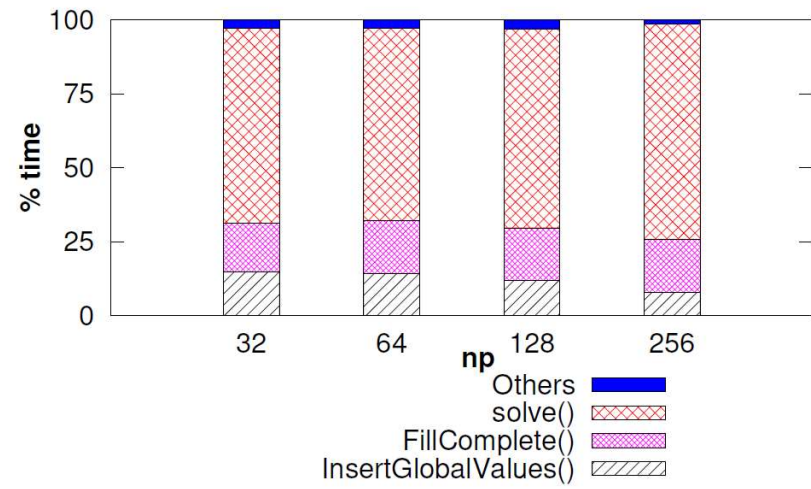
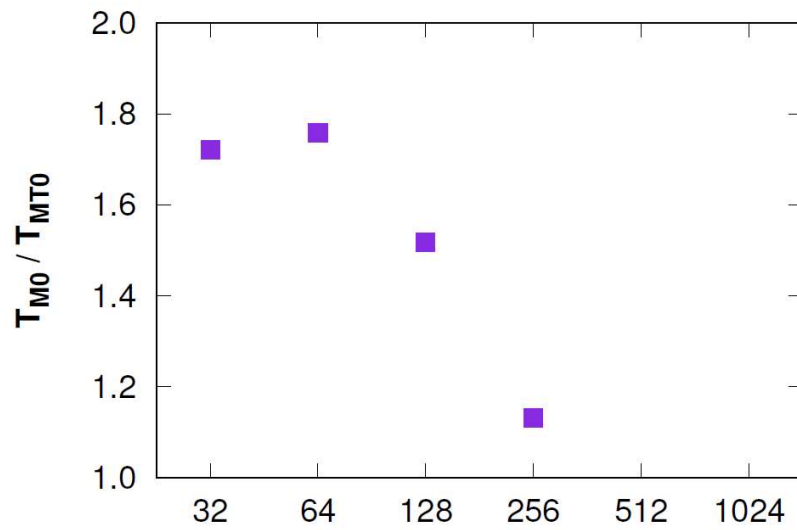
**Comet:** San Diego Supercomputer Center (SDSC)

## Various supercomputers used for the performance analysis study

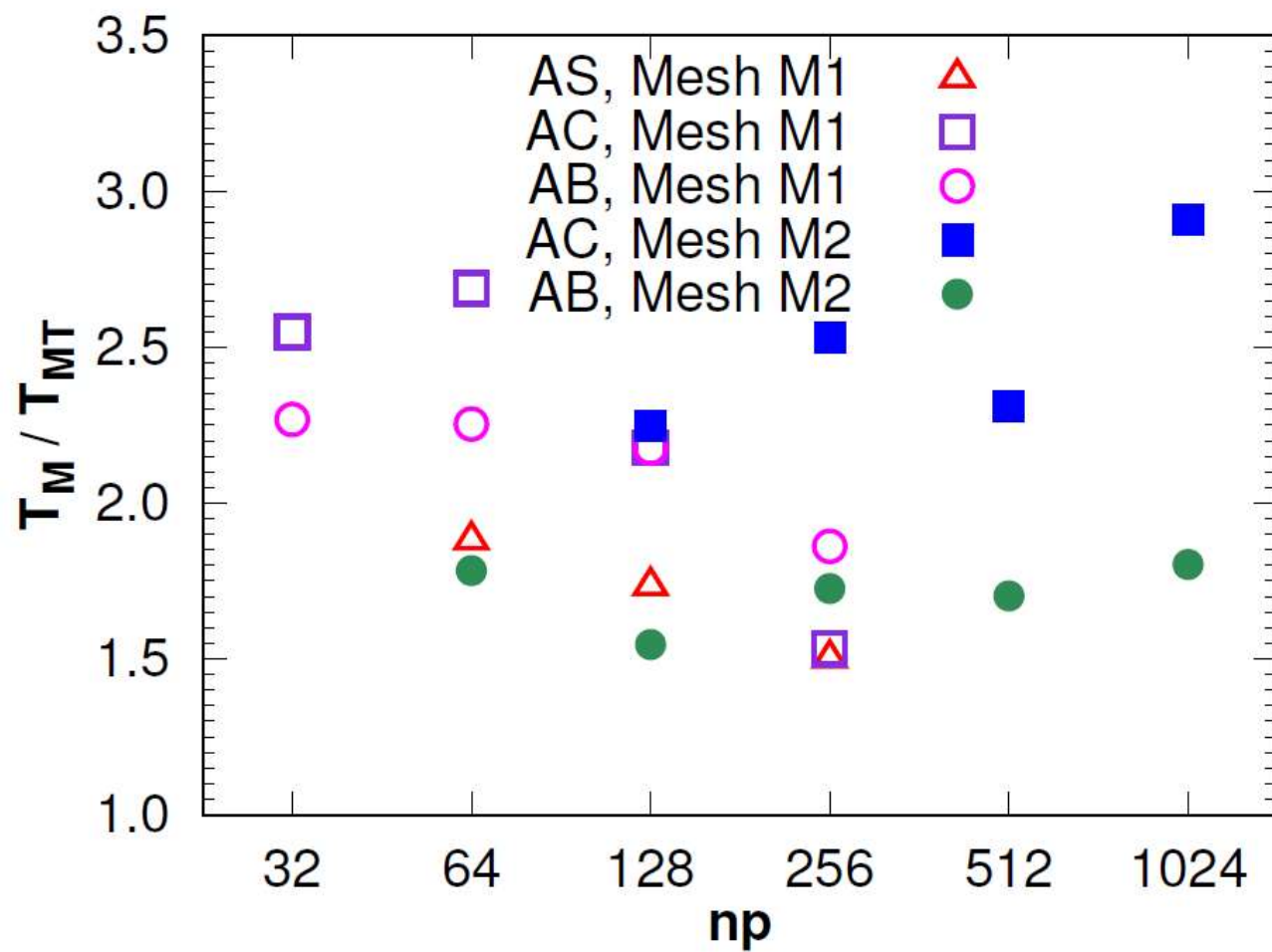
	Stampede (AS)	Comet (AC)	Bridges (AB)
Model (Intel Xeon)	E5-2680 2.7GHz	E5-2695 2.5GHz	E5-2695 2.30 GHz
Cores per socket	8	12	14
Sockets	2	2	2
L1 cache (KB)	32	32	32
L2 cache (KB)	256	256	256
L3 cache (KB)	20480	30720	35840
RAM (GB)	32	128	130



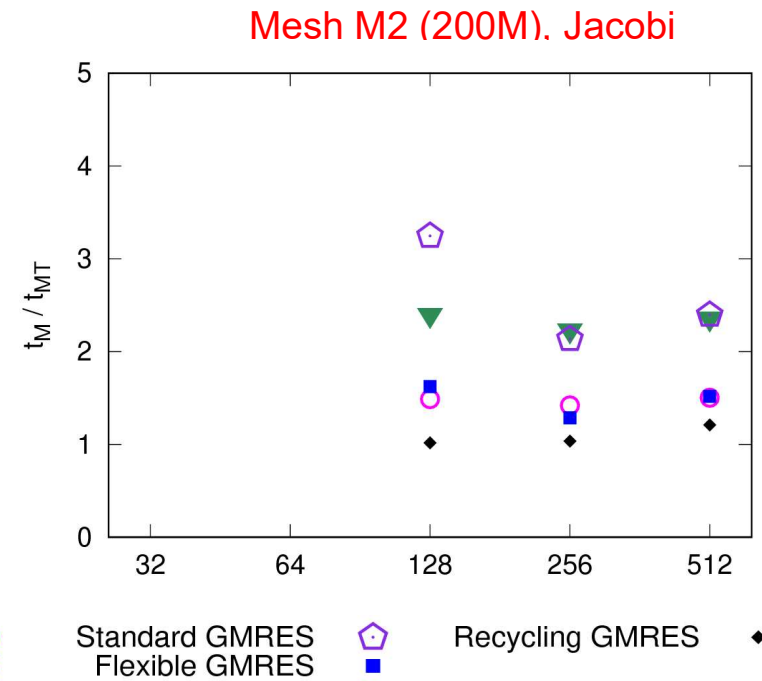
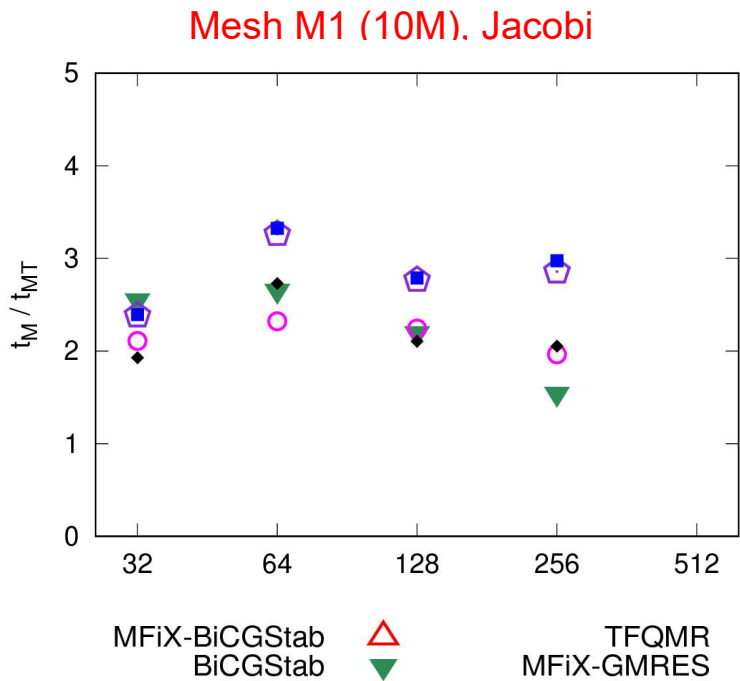
# Performance of MFiX-Trilinos



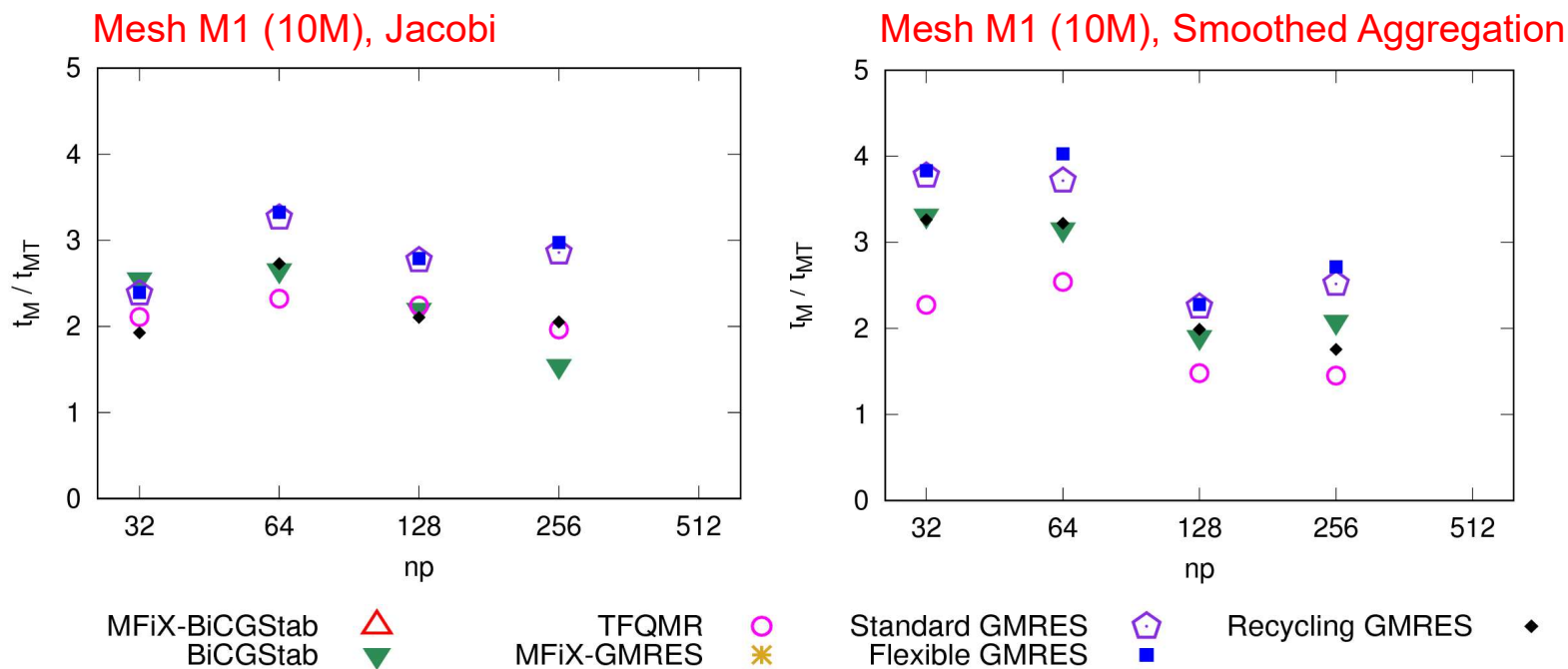
# Performance of MFiX-Trilinos



# Iterative solvers



# Preconditioners



# First and second generation solvers stacks

## First Generation

Iterative Solvers: BiCGStab, GMRES

Packages: Epetra (obj), Aztec(solver), ML

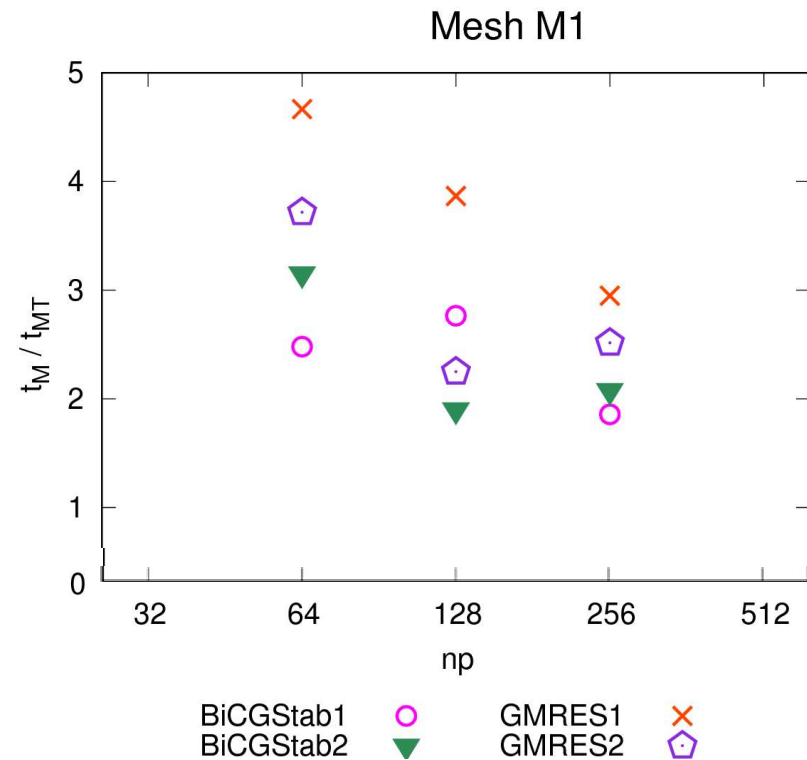
PC: Smoothed Aggregation

## Second generation

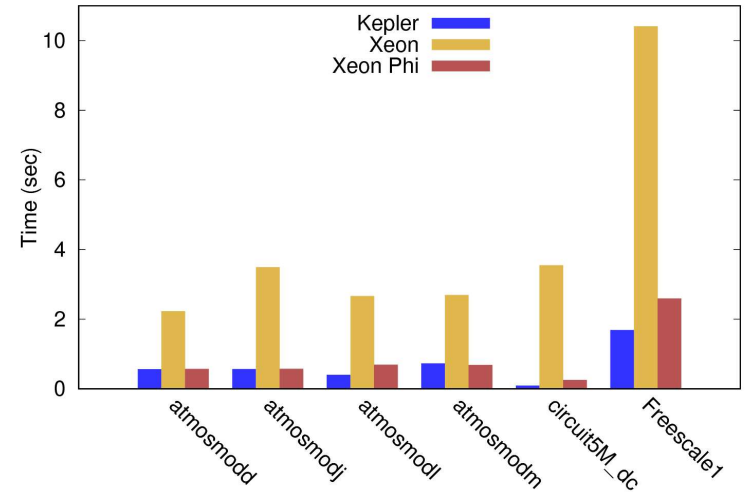
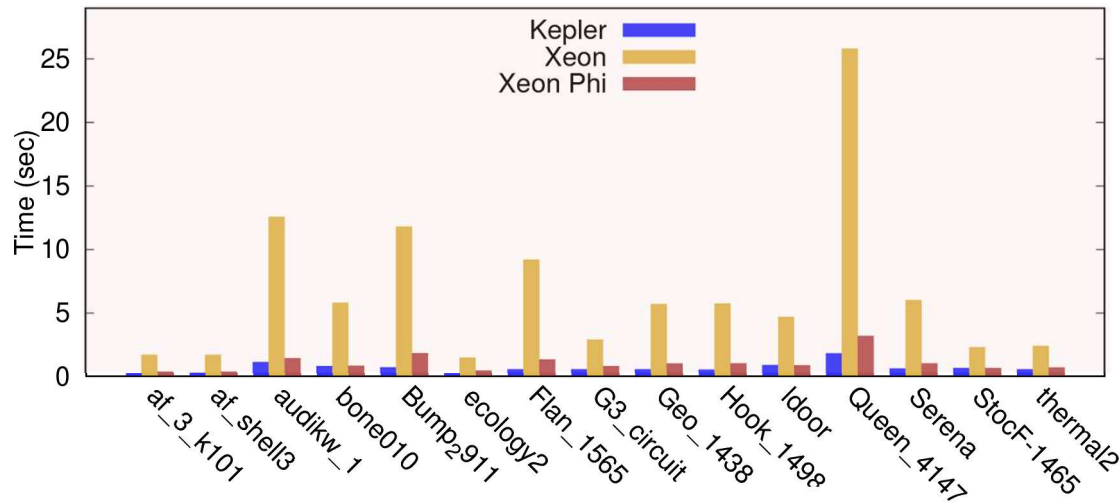
Iterative Solvers: BiCGStab, GMRES

Packages: Tpetra (obj), Belos (solver), MueLu

PC: Smoothed Aggregation



# Portability



Matrix	N	NNZ	NNZ/N
af_3_k101	503,625	17,550,675	34.8
af_shell3	504,855	17,562,051	34.8
audikw_1	943,695	77,651,847	82.3
bone010	986,703	47,851,783	48.5
Bump_2911	2,911,419	127,729,899	43.9
ecology2	999,999	4,995,991	5.0
Flan_1565	1,564,794	114,165,372	73.0
G3_circuit	1,585,478	7,660,826	4.8
Geo_1438	1,437,960	60,236,322	41.9
Hook_1498	1,498,023	59,374,451	39.6
ldoor	952,203	42,493,817	44.6
Queen_4147	4,147,110	316,548,962	76.3
Serena	1,391,349	64,131,971	46.1
StocF-1465	1,465,137	21,005,389	14.3
thermal2	1,228,045	8,580,313	7.0

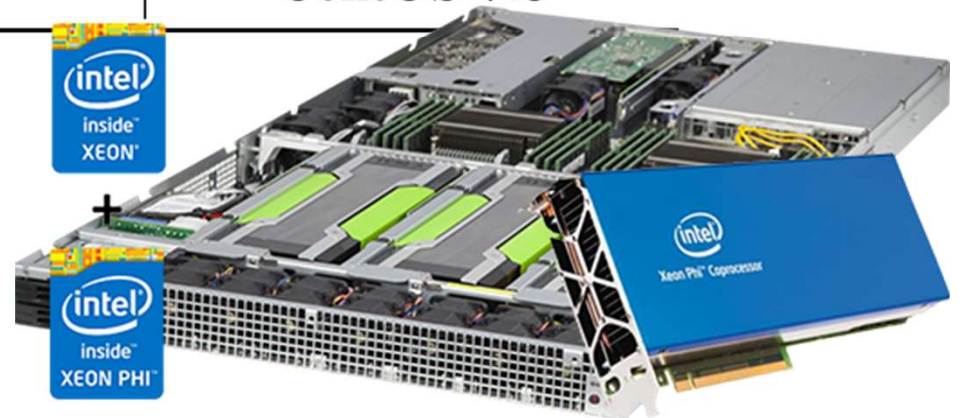
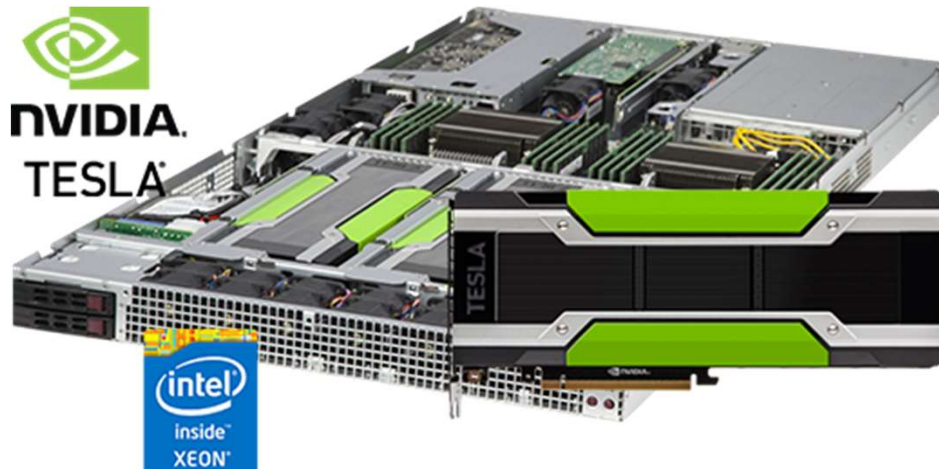
Matrix	N	NNZ	NNZ/N
atmosmodd	1,270,432	8,814,880	6.9
atmosmodj	1,270,432	8,814,880	6.9
atmosmodl	1,489,752	10,319,760	6.9
atmosmodm	1,489,752	10,319,760	6.9
circuit5M-dc	3,523,317	14,865,409	4.2
Freescale1	3,428,755	17,052,626	5.0

<https://www.cise.ufl.edu/>

**Trilinos** Solver: BiCGStab, Tpetra (obj), Belos (solver), Ifpack2 (pre)

Name	Hansen	Stampede2
Nodes	3	32
CPU	2 Xeon E5-2698 2.30GHz HT-on	<b>Xeon Phi</b> 7250 1.4GHz HT-on
Co-Processor	<b>NVIDIA Kepler 80</b>	
Interconnect	FDR IB	OPA
Memory	512 GB	96GB DDR4 + 16GB MCDRAM
Compiler	gcc 4.9.3 + CUDA 8.0	ICC 17.0.4
MPI	open MPI 1.10.6	Intel MPI 2017 3
OS	RedHat 6.5	CentOS 7.3

  
**NVIDIA.**  
**TESLA**



# Gas-solid flow in a 3D bubbling fluidized bed

Cartesian vessel: 10cm length, 10cm width and 100cm height.

Sand particle diameter	= 0.04cm
Sand particle density	= 2.0 g/cm <sup>3</sup>
Restitution co-efficient	= 0.80
Angle of internal friction	= 30
The minimum void fraction	= 0.42
Fluid (gas) viscosity	= 0.00018 g/cm s
Fluid density	= 0.0012 g/cm <sup>3</sup>

## Boundary conditions

Inlet: constant mass inflow (124.6 cm/s for  $4.3 < x < 5.7$ ,  $4.3 < z < 5.7$ )

Sidewalls: slip condition

Outlet : pressure outflow condition

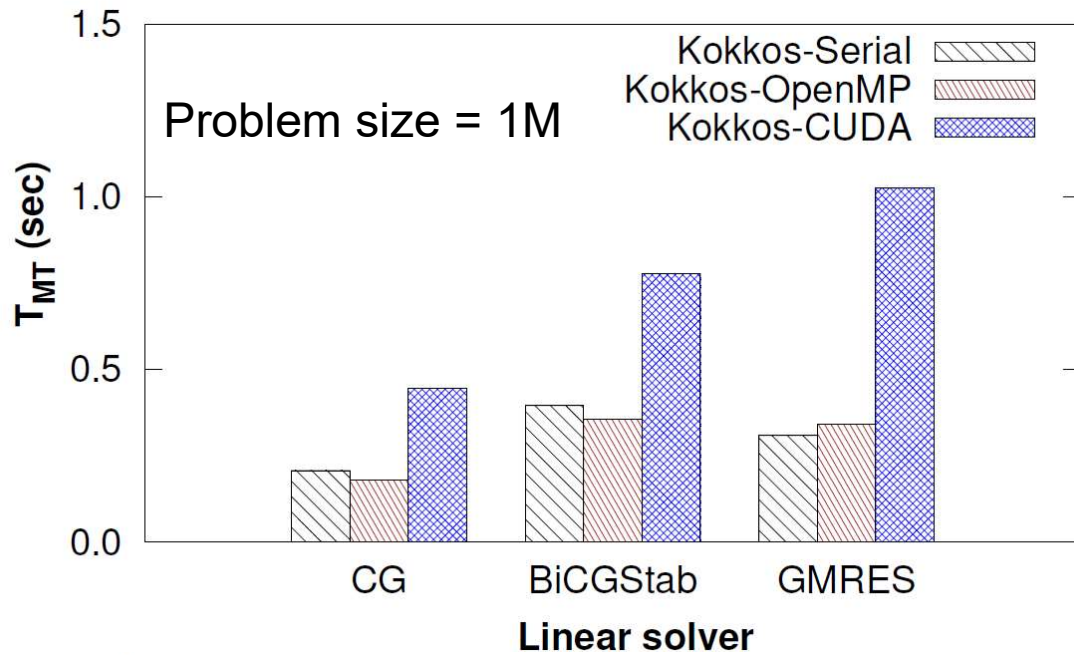
**Mesh1** - Structured: IMAX = 100, JMAX= 100, KMAX = 100

**Mesh2** - Structured: IMAX = 100, JMAX= 100, KMAX = 1000

**Mesh3** - Structured: IMAX = 100, JMAX=100, KMAX = 2000



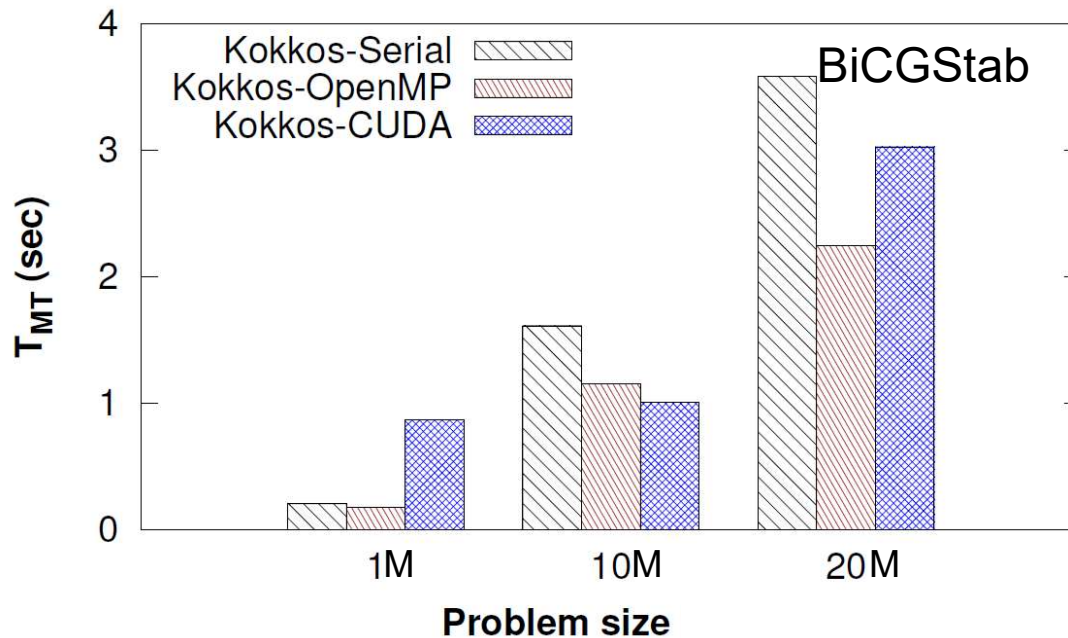
# Flow in a 3D fluidized bed problem



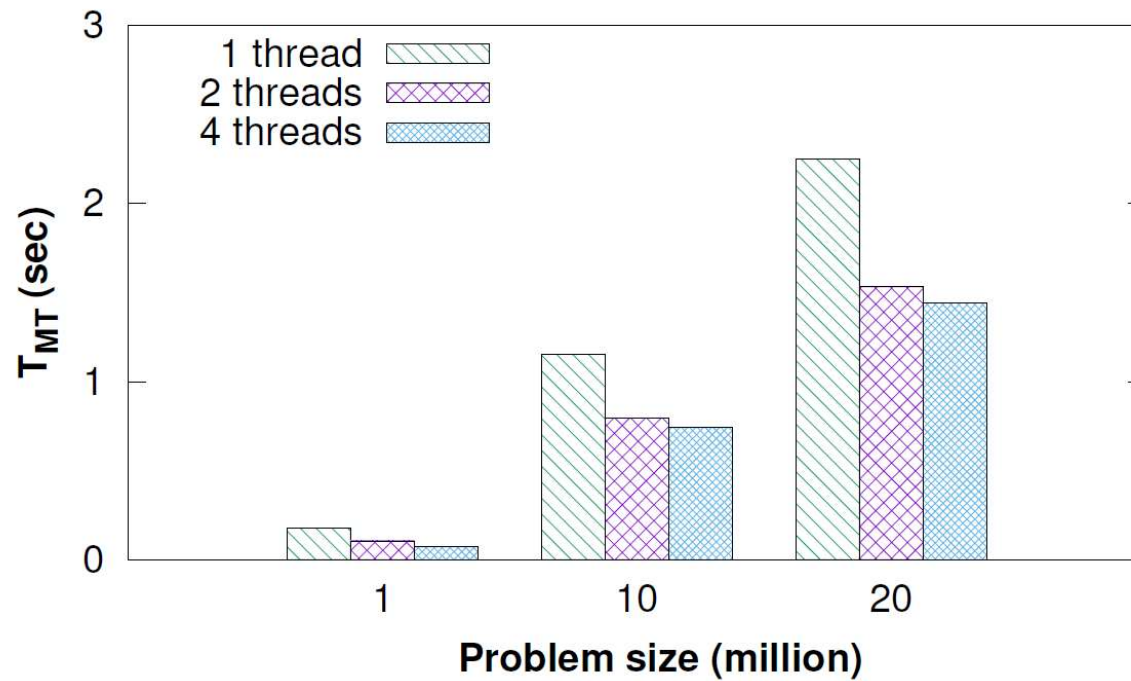
Kokkos-Serial – Haswell

Kokkos-OpenMP – Xeon Phi

Kokkos-CUDA – k80 GPU



# Flow in a 3D fluidized bed problem



Kokkos-OpenMP – Xeon Phi  
BiCGStab solver

## Remarks:

- Trilinos linear solver was integrated with MFiX
- Scalability and speed-up tests for 2D & 3D bubbling flow problems were performed for upto 4096 processors
- Upto 5x speedup is observed -- further investigation is required to examine any biases in the solver parameters

## Scholarly dissemination

- V. Kottedda, A. Chattopadhyay, V. Kumar, W. Spatz, “Linear solver performance analysis of MFiX integrated with a next generation computational framework”, The International Journal of Computational Fluid Dynamics (2018). (In review)
- VMK Kottedda, V Kumar and W Spatz, “performance of preconditioned iterative solvers in MFiX-Trilinos for fluidized beds”, Journal of Supercomputing , (2018). (In review)
- Chattopadhyay, Ashesh Kumar, Next Generation Exa-Scale Capable Software for High Fidelity Physics Aware Simulations, MS Thesis (2017)
- A Rodriguez, A Schiaffino, VMK Kottedda and V Kumar, Neural network approach to predict the flow rate for the immisible two phase flow at pore scale for enhance oil recovery application, FEDSM2018
- V. Kottedda, A. Chattopadhyay\*, V. Kumar, W. Spatz, “Next-generation multiphase flow solver for fluidized bed applications”, ASME FEDSM2017-69555 (2017)
- A. Chattopadhyay\*, V. Kumar, V. Kottedda, W. Spatz, “Next generation exascale capable multiphase solver with Trilinos”, ASME IMECE2016-67962 (2016)
- A. Chattopadhyay\*, V. Kumar, V. Kottedda, W. Spatz, Leveraging Trilinos’s Next Generation Computing Framework for an Exa-Scale Poro-Elastic Network Simulator Implementation, 2016 IEEE High Performance Extreme Computing (HPEC) (2016)
- V. Kottedda, A. Chattopadhyay\*, V. Kumar, W. Spatz, A Framework to Integrate MFiX with Trilinos for High Fidelity Fluidized Bed Computations, 2016 IEEE High Performance Extreme Computing (HPEC) (2016)

# Q & A ?

We would like to acknowledge National Energy Technology Laboratories (**NETL**), Sandia National Laboratories (**Sandia**), Texas Advanced Computing Center (**TACC**), eXtreme Science and Engineering Discovery Environment (**XSEDE**), Research cloud team at UT El Paso, Mechanical Engineering and Computational Science Programs at UT El Paso. This material is based upon work supported by the Department of Energy under Award Number DE-FE0026220.