Solver Algorithms R&D for Scalable Applications & Architectures

Michael A. Heroux
Sandia National Laboratories

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy under contract DE-AC04-94AL85000.

Sandia National Laboratories

Take Home Points

- Trilinos is a large and growing “project of projects”.
- Software Engineering processes (when properly adapted) can positively impact algorithms R&D.
- New node architectures offer opportunities and challenges.
- We prepared for them from the beginning.
- We are working toward effective use today.

Sandia National Laboratories

Highlighted Projects

- Trilinos:
  - 40+ packages in repository. Multiple distribution groupings.
  - 8.0 Release 8/312007. 1200 downloads in 5 months. 6000 since Mar '05
  - Growing external collaborations: ORNL, LBL, INL, Boeing, XOM.
  - Trilinos 9.0: Fuller vertical SW stack, fuller support for Windows, Mac, more customers.
- TOPS-2:
  - Establishing ties to climate community:
    - Implicit solvers in Homme (dycore component) for petascale systems (Kate Evans ORNL).
    - Implicit POP (Wilbert Weijer LANL).
  - Fortran, PETSc interfaces ramping up.

- Tramonto:
  - Tramonto 2.1: First public Release March 2007. 100 downloads. Dow API.
  - Proposals: R&D 100.
  - New scalable preconditioners for several state-of-the-art DFTs.
  - Peta-scalability of Tramonto.

- Mantevo:
  - Mantevo*: Five microapps (phdMesh, HPCCG, pHPCCG, Beam, Prolego) + framework.
  - HPCCG: Publicly available. Part of Sequoia benchmark.
    - "Closest thing to an unstructured FEM/FVM code in 500 semi-colons or fewer."
    - Ports to nVidia, Clovertown, Sun 8x8 core/threads, RedStorm, Sequoia RFP, …
    - Rewritten in BEC, Qthreads.
    - 25K core runs on Redstorm.

  - pHPPCG: Parametrized HPCCG - arbitrary int/float types, data structure base class.
  - phdMesh part of Trilinos…Beam exercises vertical stack in Trilinos…Prolego basic research.

* Greek: augur, guess, predict, presage

Sandia National Laboratories

Libraries for Science & Engineering Applications: Trilinos

Sandia National Laboratories

Trilinos Contributors (past 3 years)

Chris Baker
Developer of Anasazi, RBGen

Ross Bartlett
Lead Developer of MOOCHO, Stratimikos, RTOp, Thyra
Developer of Rythmos

Pavel Bochev
Lead Developer Intrepid

Paul Boggs
Developer of Thyra

Erik Boman
Lead Developer Isorropia
Developer Zoltan

Todd Coffey
Lead Developer of Rythmos

Karen Devine
Lead Developer Zoltan

Clark Dohrmann
Lead Developer of CLAPS

Carter Edwards
Lead Developer phdMesh

Michael Gee
Developer of ML, Moertel, NOX

Bob Heaphy
Lead developer of Trilinos SQA
Developer Zoltan

Mike Heroux
Trilinos Project Leader
Lead Developer of Epetra, AztecOO, Kokkos, IFPACK, Tpetra
Developer of Amesos, Belos, EpetraExt

Ulrich Hetmaniuk
Developer of Anasazi

Robert Hoekstra
Lead Developer of EpetraExt
Developer of Epetra

Russell Hooper
Developer of NOX

Vicki Howle
Lead Developer of Meros

Jonathan Hu
Developer of ML

Joe Kotulski
Lead Developer of Pliris

Rich Lehoucq
Developer of Anasazi and Belos

Kevin Long
Developer of Thyra

Roger Pawlowski
Lead Developer of NOX

Michael Phenow
Trilinos Webmaster
Developer WebTrilinos

Eric Phipps
Lead developer Sacado, Stokhos
Developer of LOCA, NOX

Dennis Ridzal
Lead Developer of Aristos, Intrepid

Marzio Sala
Lead Developer of Didasko, Galeri, IFPACK, WebTrilinos
Developer of ML, Amesos

Andrew Salinger
Lead Developer of LOCA, Capo

Paul Sexton
Developer of Epetra and Tpetra

Bob Shuttleworth
Developer of Meros.

Chris Siefert
Developer of ML

Bill Spotz
Lead Developer of PyTrilinos
Developer of Epetra, New_Package

Ken Stanley
Lead Developer of Amesos and New_Package

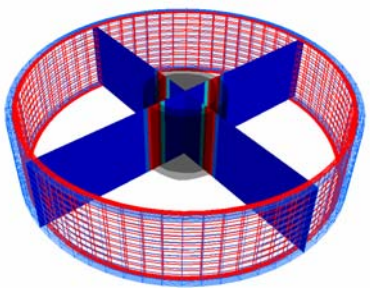
Heidi Thornquist
Lead Developer of Anasazi, Belos, RBGen and Teuchos

Ray Tuminaro
Lead Developer of ML and Meros

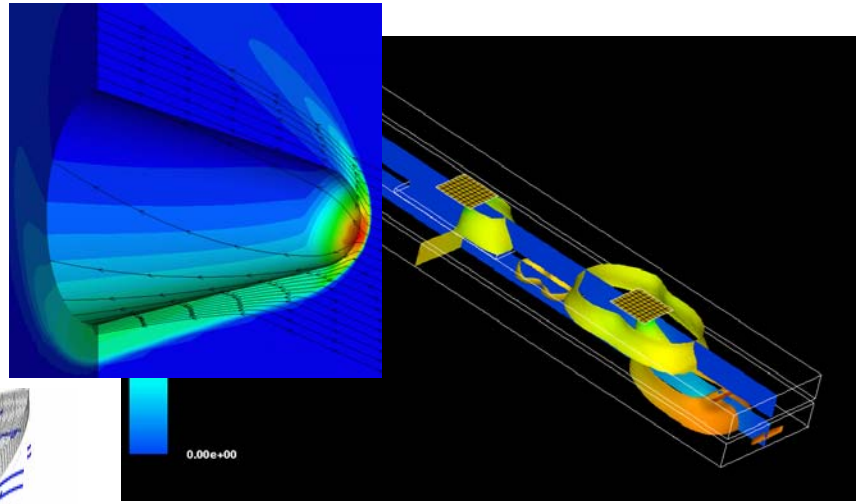
Jim Willenbring
Developer of Epetra and New_Package.
Trilinos library manager

Alan Williams
Lead Developer Isorropia, FEI
Developer of Epetra, EpetraExt, AztecOO, Tpetra

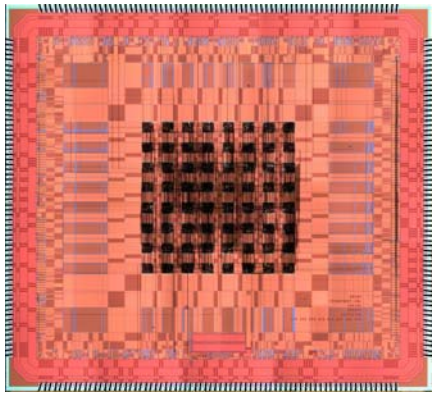
Target Problems: PDES and more…

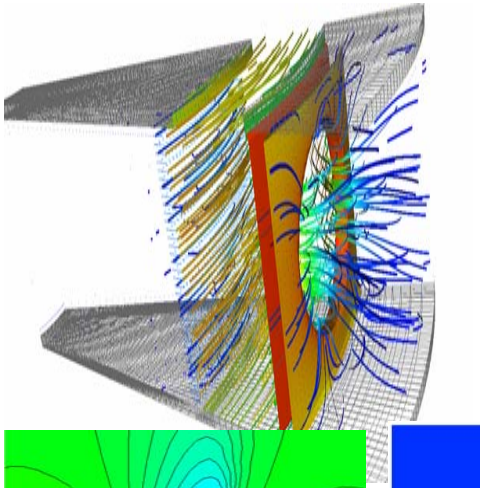


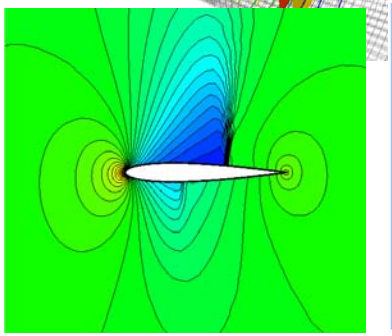
PDES

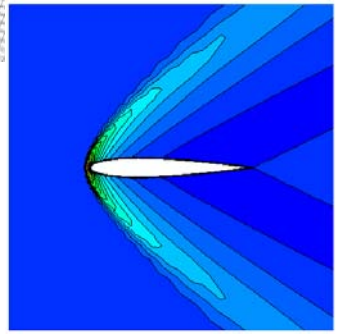


Circuits



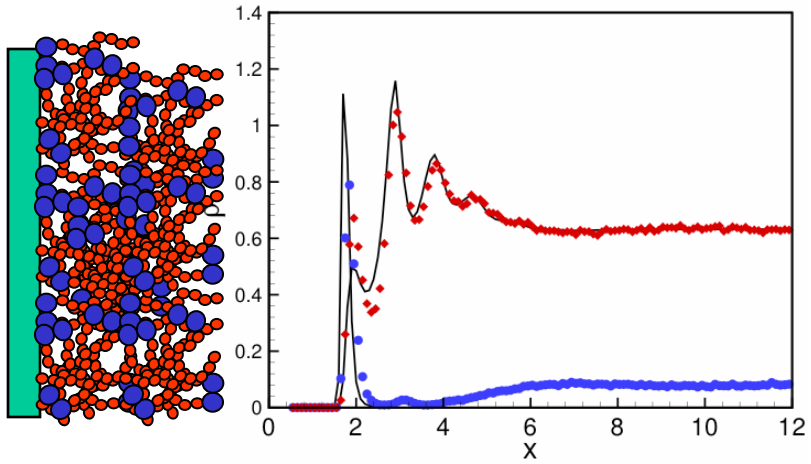






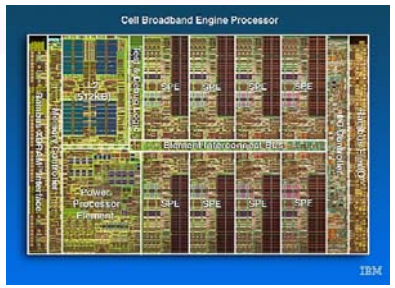
And More…

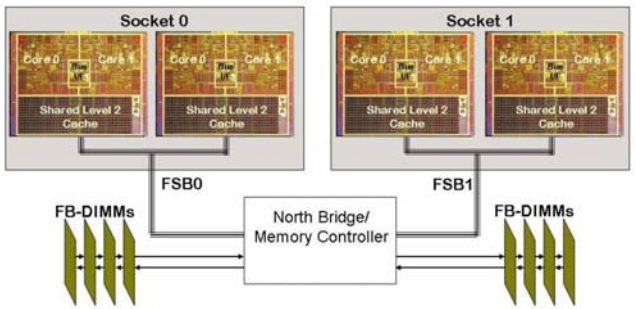
Inhomogeneous Fluids



Target Platforms: Any and All
(Now and in the Future)

- Desktop: Development and more…
- Capability machines:
  - Redstorm (XT3), Clusters
  - Roadrunner (Cell-based).
  - Multicore nodes.
- Parallel software environments:
  - MPI of course.
  - UPC, CAF, threads, vectors,…
  - Combinations of the above.
- User “skins”:
  - C++/C, Python
  - Fortran.
  - Web, CCA.





Motivation For Trilinos

- Sandia does LOTS of solver work.
- When I started at Sandia in May 1998:
  - Aztec was a mature package. Used in many codes.
  - FETI, PETSc, DSCPack, Spooles, ARPACK, DASPK, and many other codes were (and are) in use.
  - New projects were underway or planned in multi-level preconditioners, eigensolvers, non-linear solvers, etc…
- The challenges:
  - Little or no coordination was in place to:
    - Efficiently reuse existing solver technology.
    - Leverage new development across various projects.
    - Support solver software processes.
    - Provide consistent solver APIs for applications.
  - ASCI (now ASC) was forming software quality assurance/engineering (SQA/SQE) requirements:
    - Daunting requirements for any single solver effort to address alone.

Sandia National Laboratories

Evolving Trilinos Solution

- Trilinos[1] is an evolving framework to address these challenges:
  - Fundamental atomic unit is a *package*.
  - Includes core set of vector, graph and matrix classes (Epetra/Tpetra packages).
  - Provides a common abstract solver API (Thyra package).
  - Provides a ready-made package infrastructure (new_package package):
    - Source code management (cvs, bonsai).
    - Build tools (autotools).
    - Automated regression testing (queue directories within repository).
    - Communication tools (mailman mail lists).
  - Specifies requirements and suggested practices for package SQA.
- In general allows us to categorize efforts:
  - Efforts best done at the Trilinos level (useful to most or all packages).
  - Efforts best done at a package level (peculiar or important to a package).
  - **Allows package developers to focus only on things that are unique to their package.**

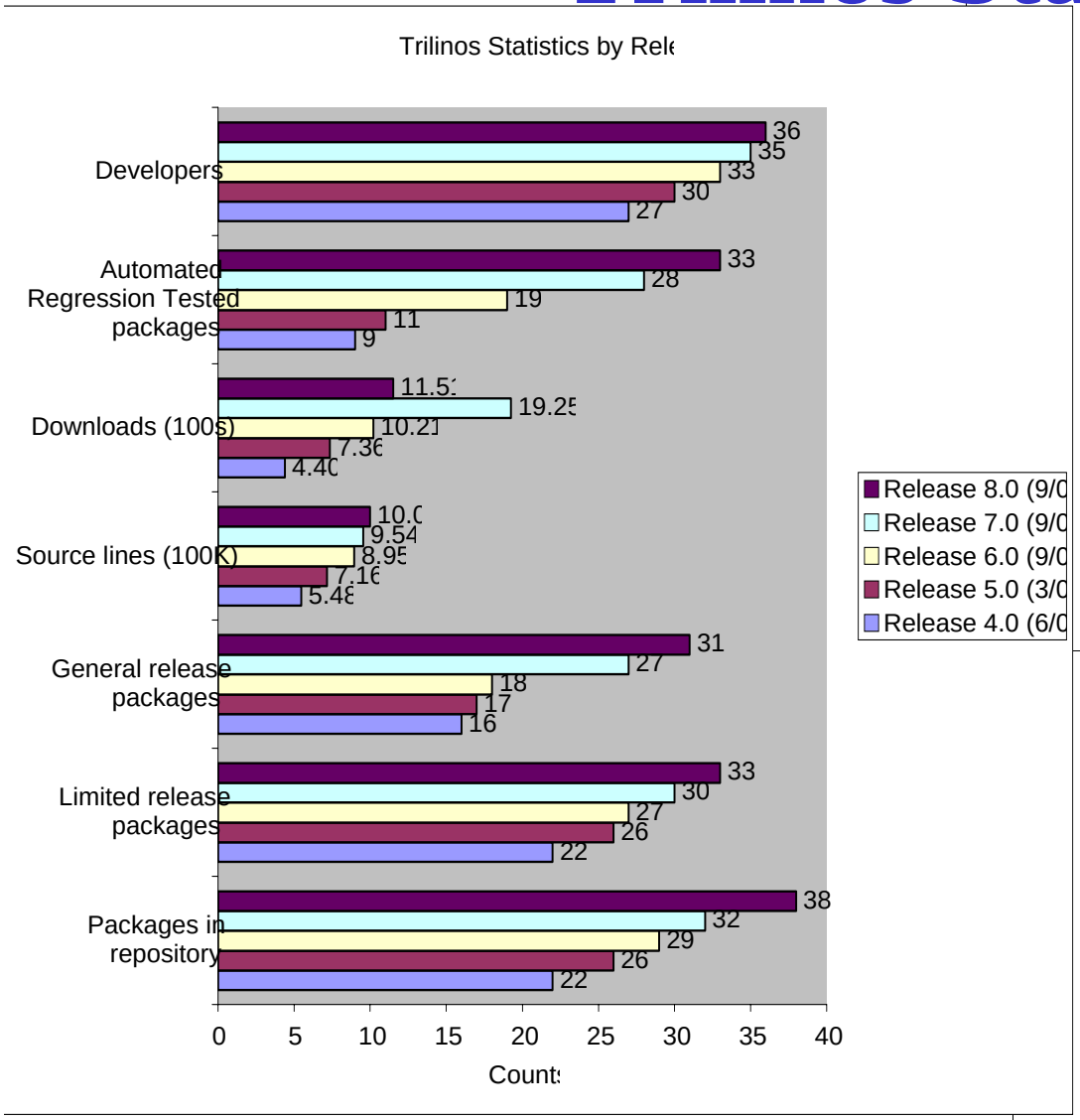
1. Trilinos loose translation: “A string of pearls”

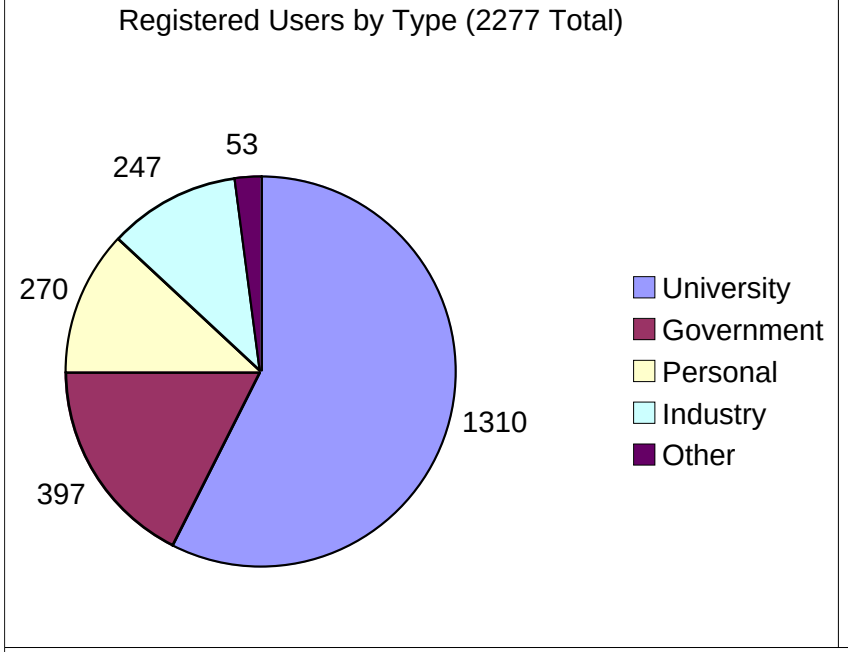
Sandia National Laboratories

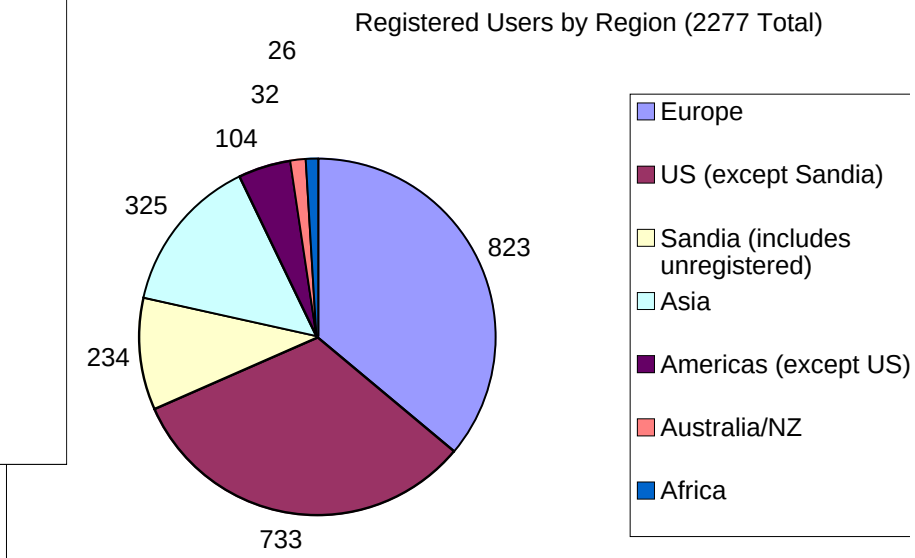
Trilinos Package Summary

	Objective	Package(s)
Discretizations	Meshing & Spatial Discretizations	phdMesh, Intrepid
	Time Integration	Rythmos
Methods	Automatic Differentiation	Sacado
	Mortar Methods	Moertel
Core	Linear algebra objects	Epetra, Jpetra, Tpetra
	Abstract interfaces	Thyra, Stratimikos, RTOp
	Load Balancing	Zoltan, Isorropia
	“Skins”	PyTrilinos, WebTrilinos, Star-P, ForTrilinos, CTrilinos
	C++ utilities, I/O, thread API	Teuchos, EpetraExt, Kokkos, Triutils, TPI
Solvers	Iterative (Krylov) linear solvers	AztecOO, Belos, Komplex
	Direct sparse linear solvers	Amesos
	Direct dense linear solvers	Epetra, Teuchos, Pliris
	Iterative eigenvalue solvers	Anasazi
	ILU-type preconditioners	AztecOO, IFPACK
	Multilevel preconditioners	ML, CLAPS
	Block preconditioners	Meros
	Nonlinear system solvers	NOX, LOCA
	Optimization (SAND)	MOOCHO, Aristos
	Stochastic PDEs	Stokhos

Trilinos Statistics







Stats: Trilinos Download Page 02/06/2008.

Trilinos Scope Changing

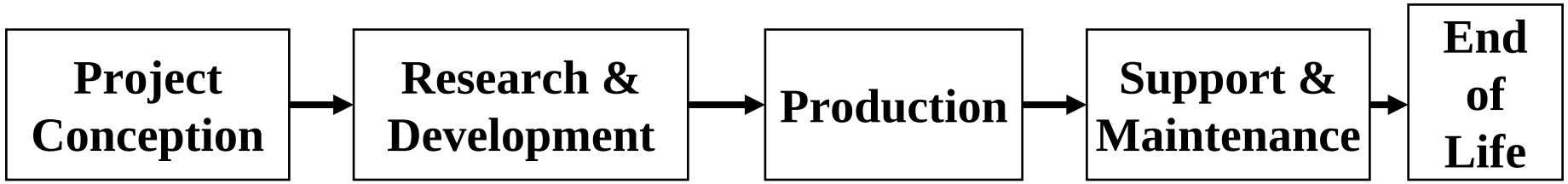
- Six Capability Areas/Leaders:
  - Framework, Tools & Interfaces (J. Willenbring).
  - Discretizations (P Bochev).
  - Geometry, Meshing & Load Balancing (K. Devine).
  - Scalable Linear Algebra (M. Heroux).
  - Linear & Eigen Solvers (J. Hu).
  - Nonlinear, Transient & Optimization Solvers (A. Salinger).
- Described in *The Changing Scope of Trilinos*, M. Heroux Sandia Tech report SAND2007-7775.

Sandia National Laboratories

SW Engineering in Scientific SW

- Much to Learn from SW Engineering discipline, but…
- Simple translation of techniques is inappropriate.
- Important goal for Trilinos: Adapt industry techniques to our needs.
- One important step forward: A Lifecycle Model.

(Typical) Project Lifecycle



Sandia National Laboratories

Scientific Research and Life Cycle Models

- Life Cycle Models are generally developed from the point of view of business software.
- Little consideration is given to algorithmic development.
- Traditional business execution environment is traditional mainframe or desktop, not parallel computers.
- Traditional development “techniques” are assumed.

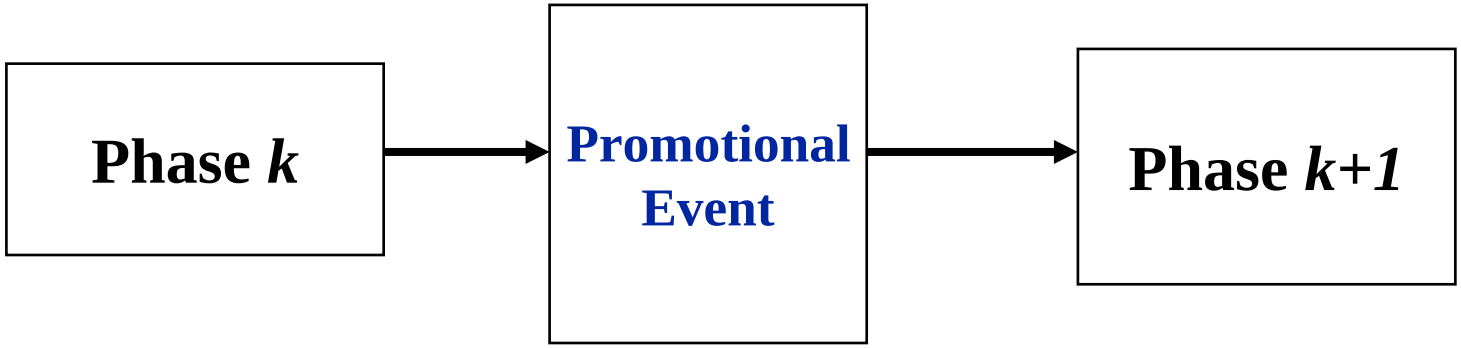
Sandia National Laboratories

Research Software needs a different model

- Research should be “informal”:
  - Allow external collaborators, students, post-docs, etc.
  - Allow changes of direction without seeking permission
  - Should use modern software development paradigms
    - i.e. Lean/Agile methods
  - Must be verified more than validated

- Production code must:
  - Have formality appropriate to risks,
  - Be Complete (documentation, testing, …),
  - Be “user proofed”,
  - Be platform independent (as necessary),
  - Be validated not just verified.

“Promotional” Model



- Lower formality
- Fewer Artifacts
- Lean/Agile

- Higher formality
- Sufficient Artifacts
- Bullet proof
- Maintainable

Sandia National Laboratories

Trilinos Lifecycle Model

- Three phases:
  - Research.
  - Production Growth.
  - Production Maintenance.
- Each phase contains its own lifecycle model.
- Promotional events:
  - Required for transition from one phase to next.
  - Signify change in behaviors and attitude.
- Phase assigned individually to each package.

The Trilinos Software Lifecycle Model, James M. Willenbring and Michael A. Heroux and Robert T. Heaphy, *Proceeding of the 29th International Conference on Software Engineering*, May 2007

Sandia National Laboratories

Lifecycle Phase 1: Research

- Conducting research is the primary goal.
- Producing software is potentially incidental to research.
- Any software that is produced is typically a “proof of concept” or prototype.
- Software that is in this phase may only be released to selected internal customers to support their research or development and should not be treated as production quality code.

Sandia National Laboratories

Phase 1 Required Practices

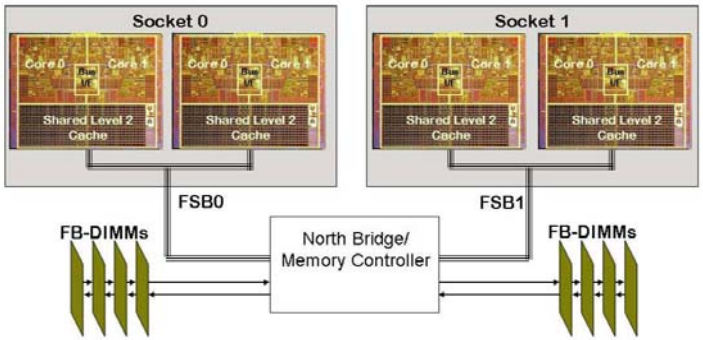
- The research proposal is the project plan.
- Software is placed under configuration control as needed to prevent loss due to disaster.
- Peer reviewed published papers are primary verification and validation.
- The focus of testing is a proof of correctness, not software.
- Periodic status reports should be produced, presentation is sufficient.
- A lab notebook, project notebook, or equivalent is the primary artifact.

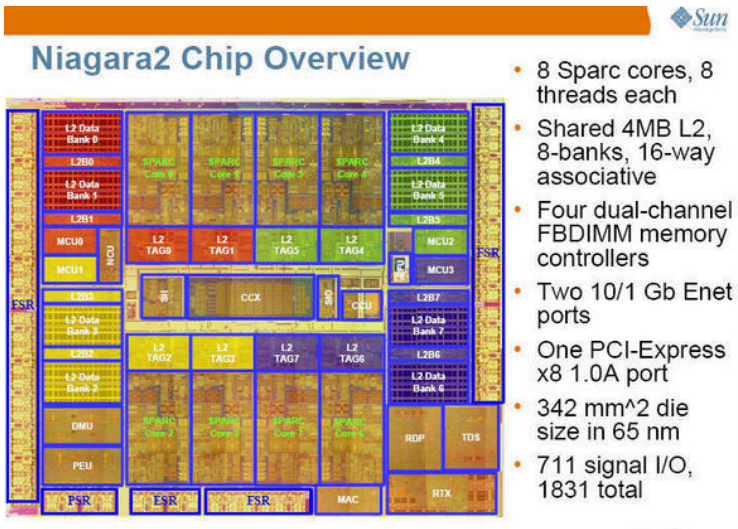
Node Architecture Studies: Mantevo Project

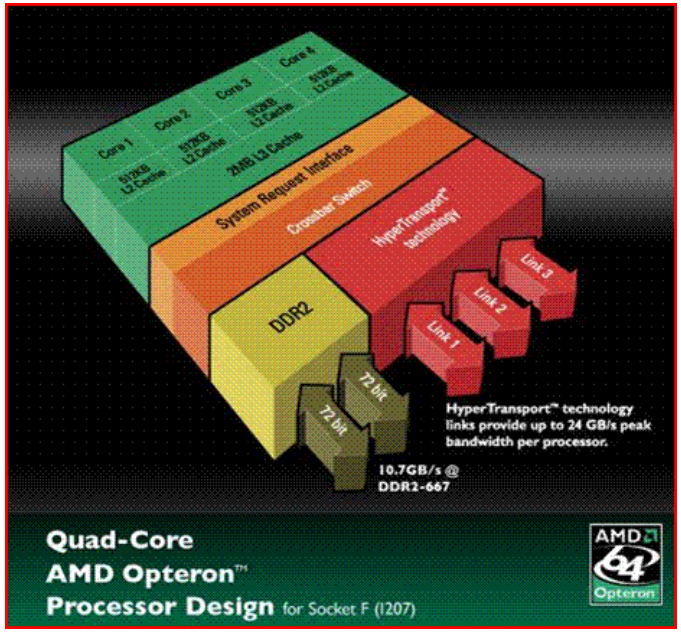
Sandia National Laboratories

Node Architectures

- Intel Clovertown
- AMD Barcelona
- Sun Niagara2







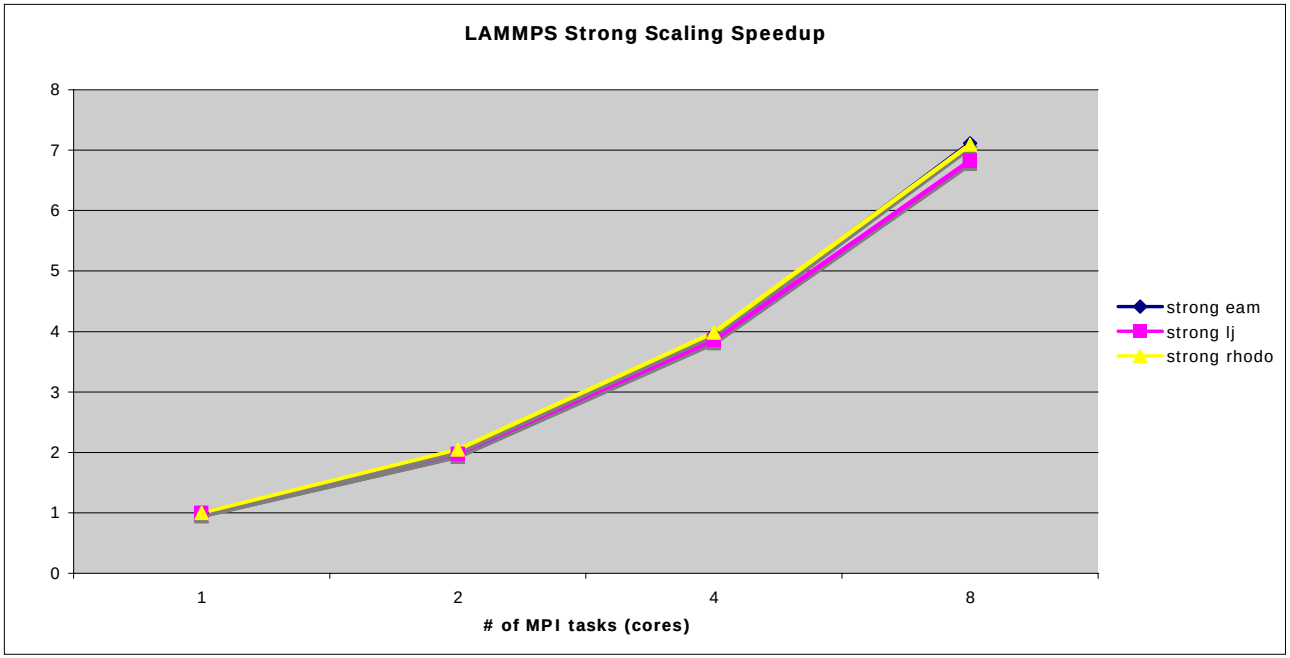
Sandia National Laboratories

Node Architecture Codes

- HPCCG:
  - “Closest thing to an unstructured FEM/FVM code in 500 semi-colons or fewer.”
- pHPCCG:
  - Compile-time parametrized FP (float, double, etc) and int (32, 64, etc).
- Epetra Benchmark Tests:
  - Trilinos Performance-determining kernels.
- Vector Multi-update:
  - Basic kernel in explicit dynamics (generalized DGEMV).
- Tramonto:
  - Polymer test case.
- LAMMPS: Molecular Dynamics.

Sandia National Laboratories

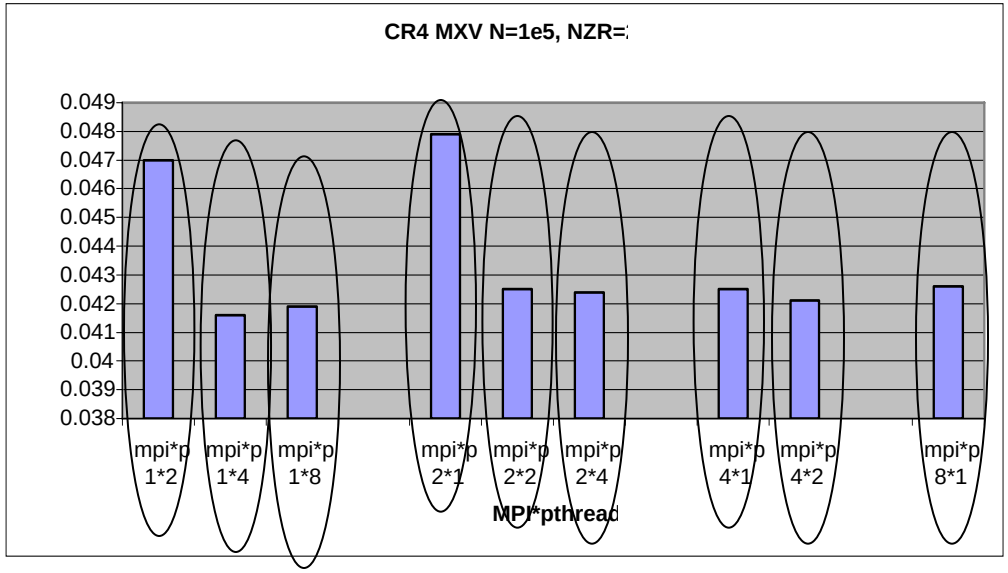
MPI-Only



- The incumbent: Always present.
- Sometimes sufficient.
- Hybrid: MPI-only + MPI/threads?

Sandia National Laboratories

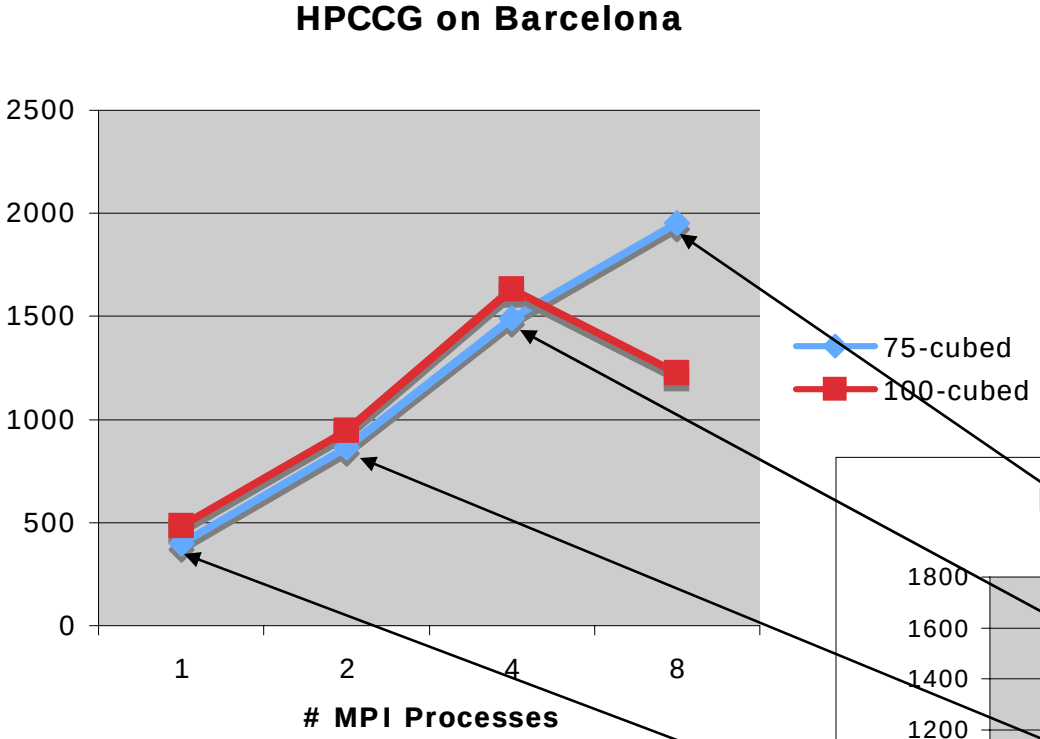
Programming Model Translation



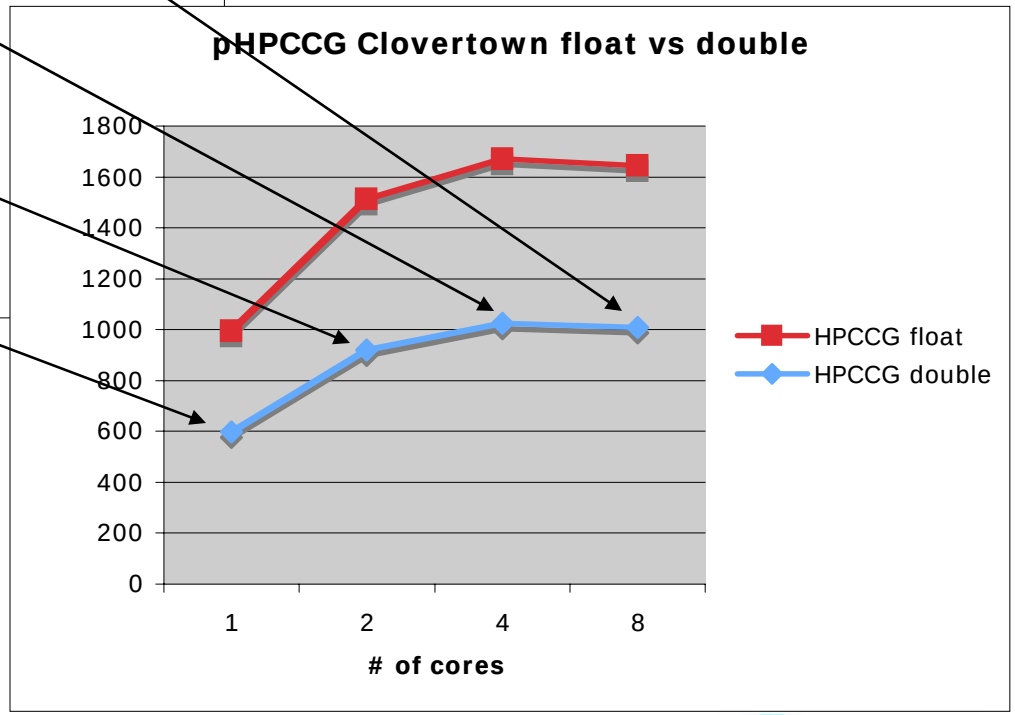
- Been here before:
  - 12-15 years ago: SMP nodes.
  - MPI vs. MPI/OpenMP/Pthreads.
- Lesson learned:
  - Nothing magic about programming model.
  - For SMP model to matter: Algorithms must exploit shared memory.

Sandia National Laboratories

AMD Barcelona vs Intel Clovertown



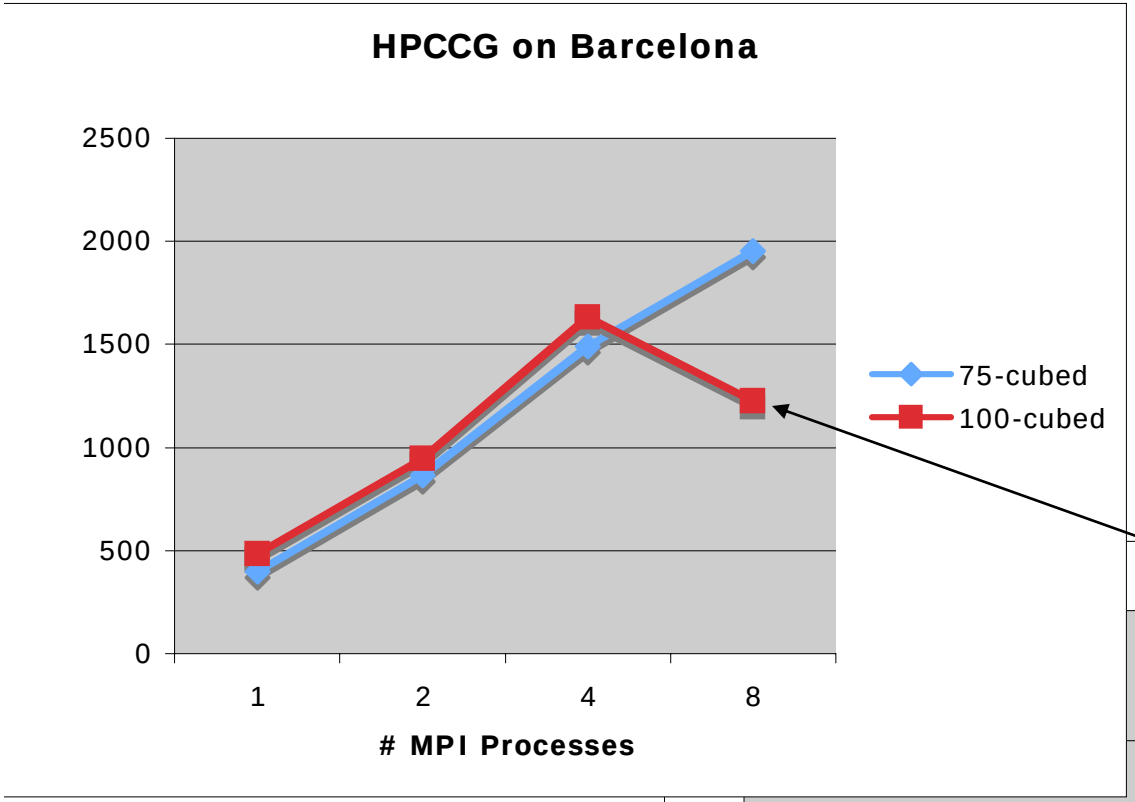
- Bandwidth penalty.
  - Always an issue, but still improvement with single core.
  - Multicore: Wasted cores.
- AMD better balanced.
- Page placement…



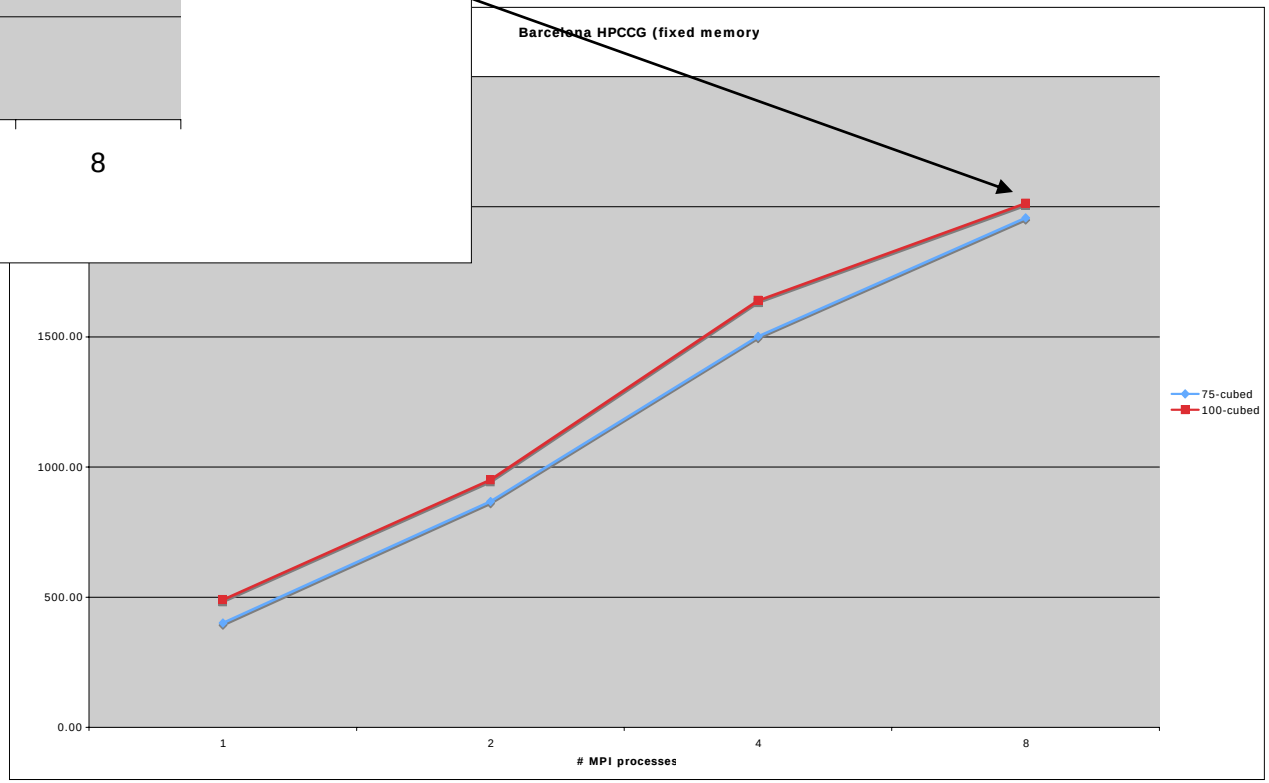
Barcelona HPCCG
Courtesy Kevin Pedretti

Sandia National Laboratories

AMD Barcelona Memory Placement

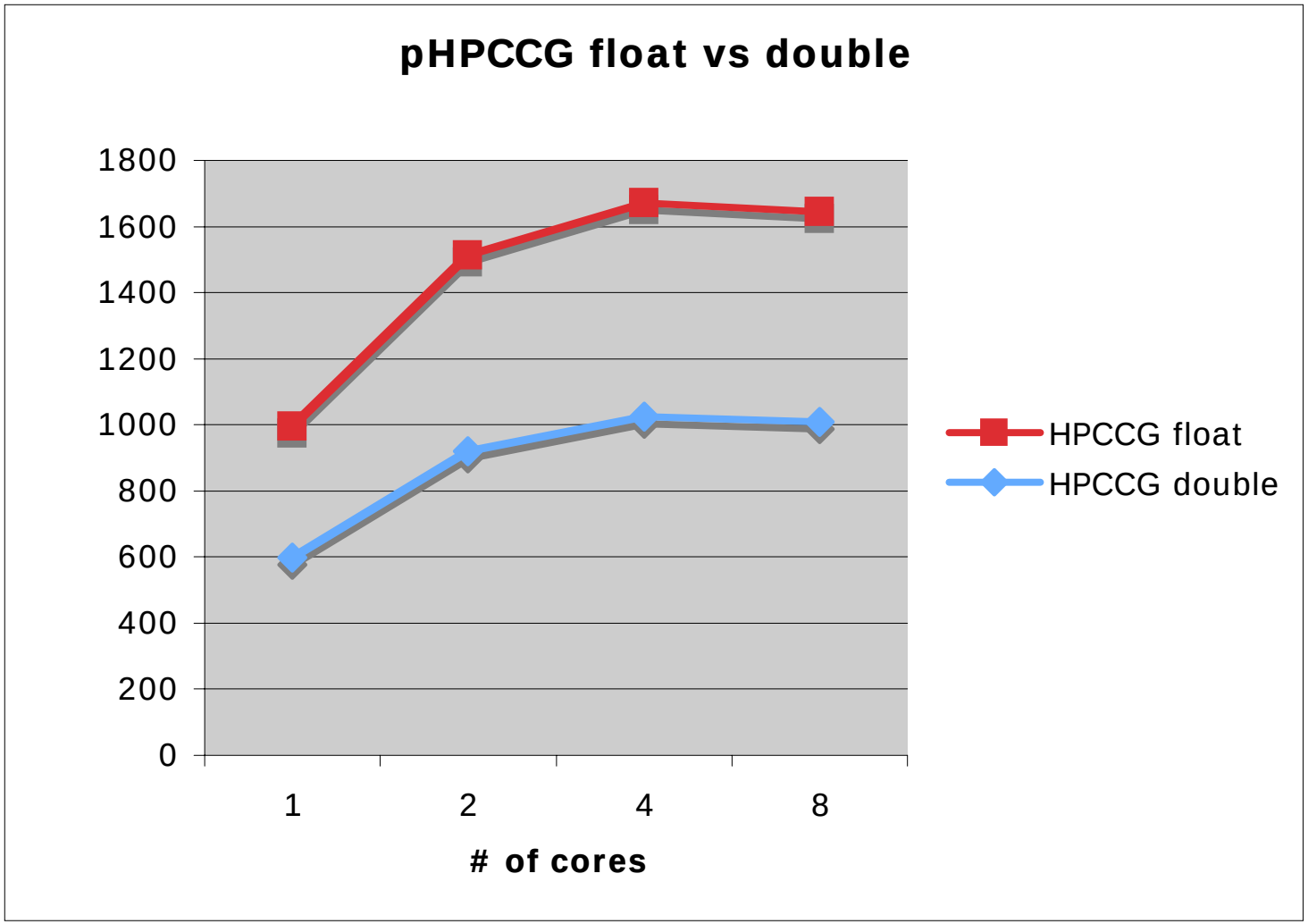


- Poor page placement impact.
- Memory policies: Need work.
- I/O swap space.



Barcelona HPCCG
Courtesy Kevin Pedretti

pHPCCG Clovertown Results

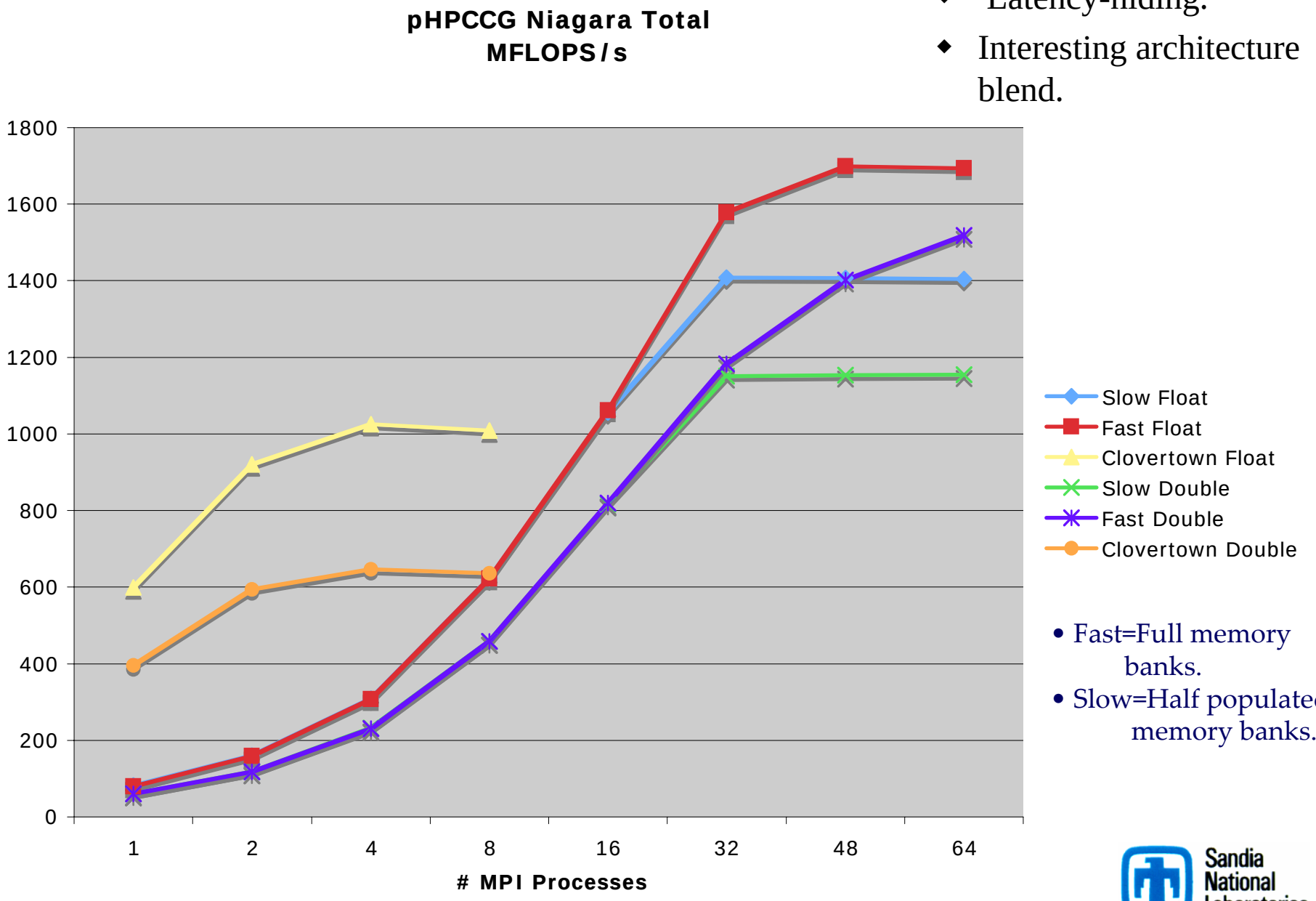


Float: better rates and scaling.

Sandia National Laboratories

Niagara2 Results

- Multi-threading provides:
  - Latency-hiding.
  - Interesting architecture blend.

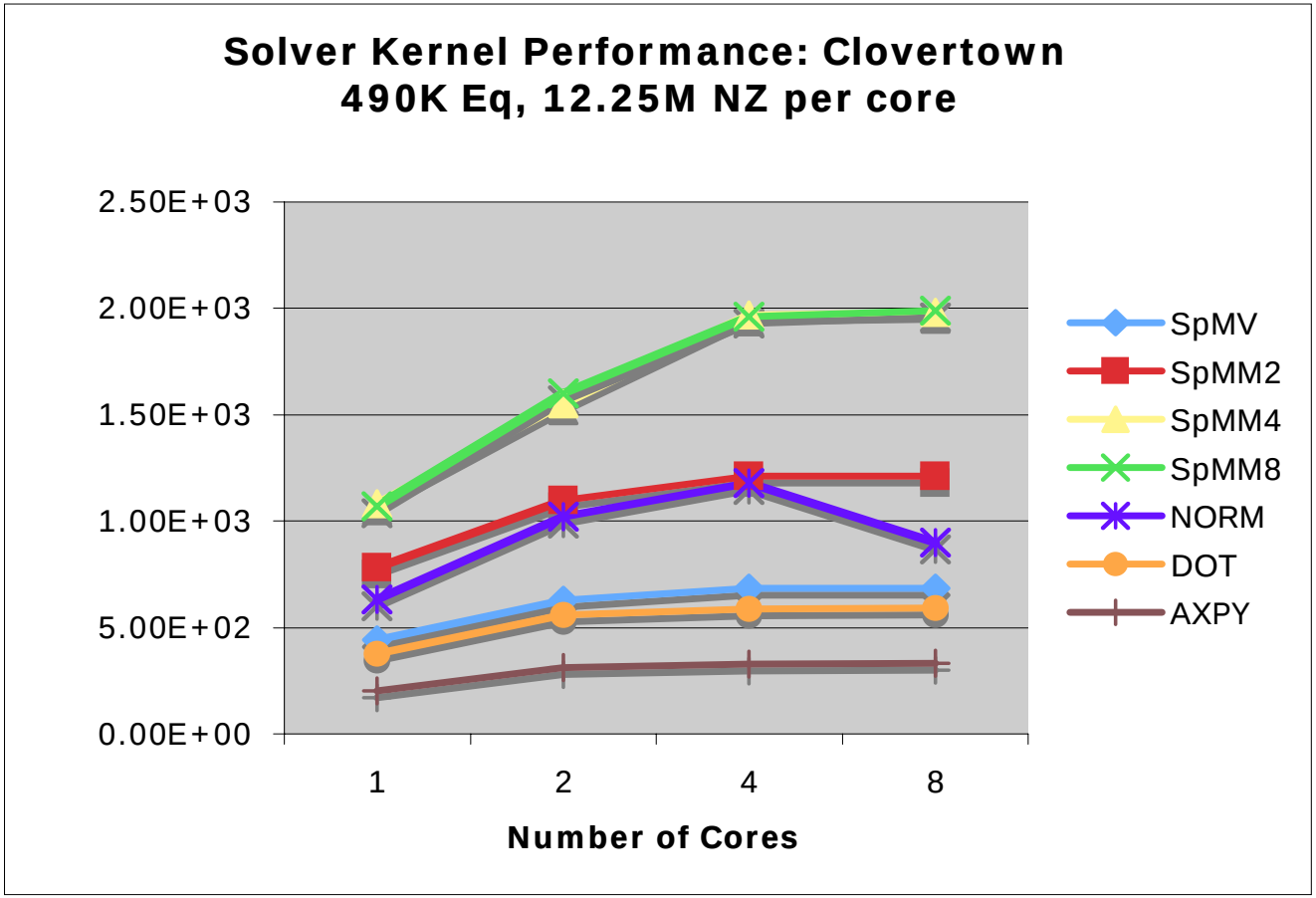


- Fast=Full memory banks.
- Slow=Half populated memory banks.

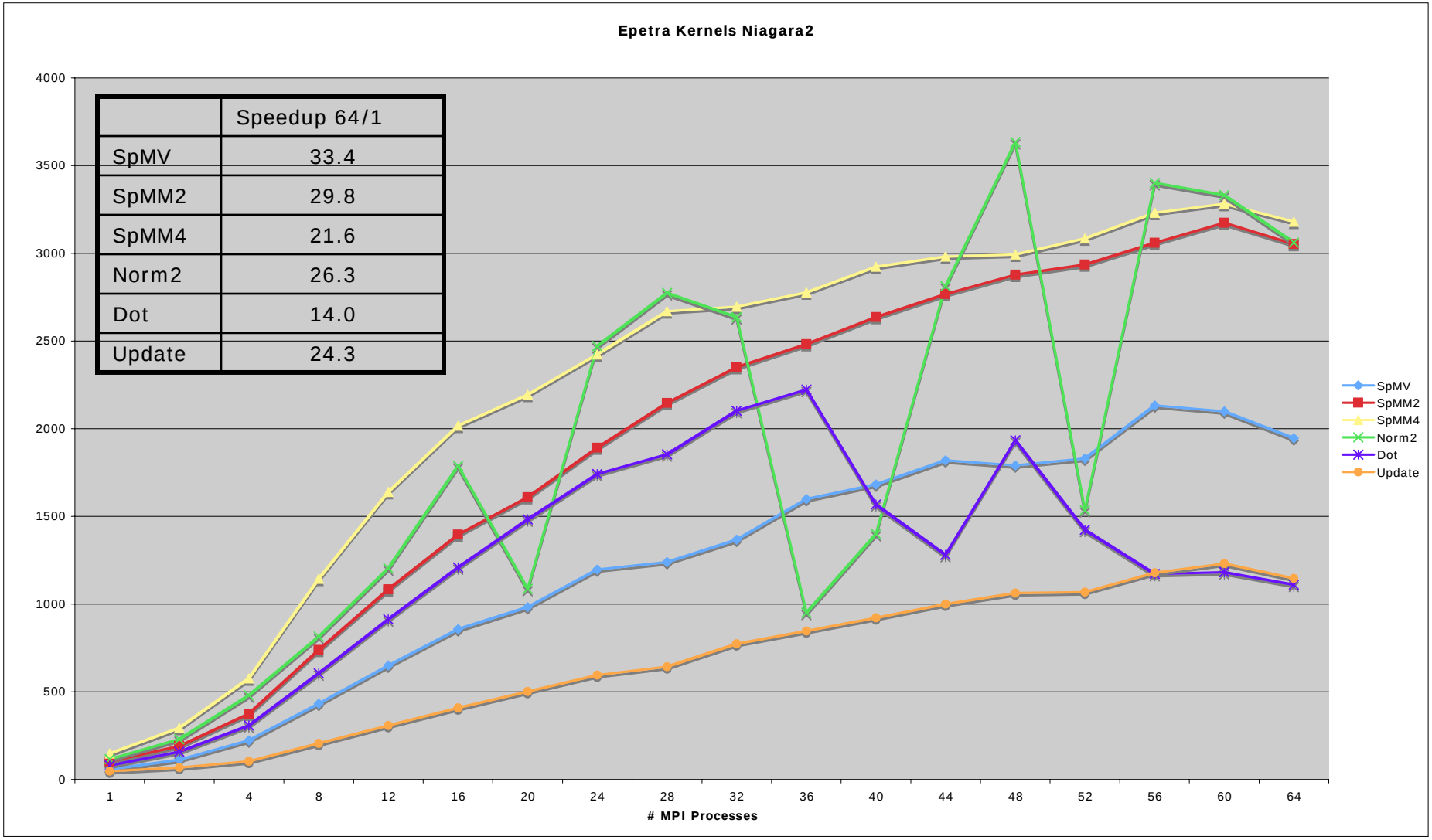
Sandia National Laboratories

Epetra Benchmark Tests

- Focused on core Epetra kernels:
  - Sparse MV, MM.
  - Dot products, norms, daxpy's.
- spMM:
  - Better performance.
  - Better core utilization.



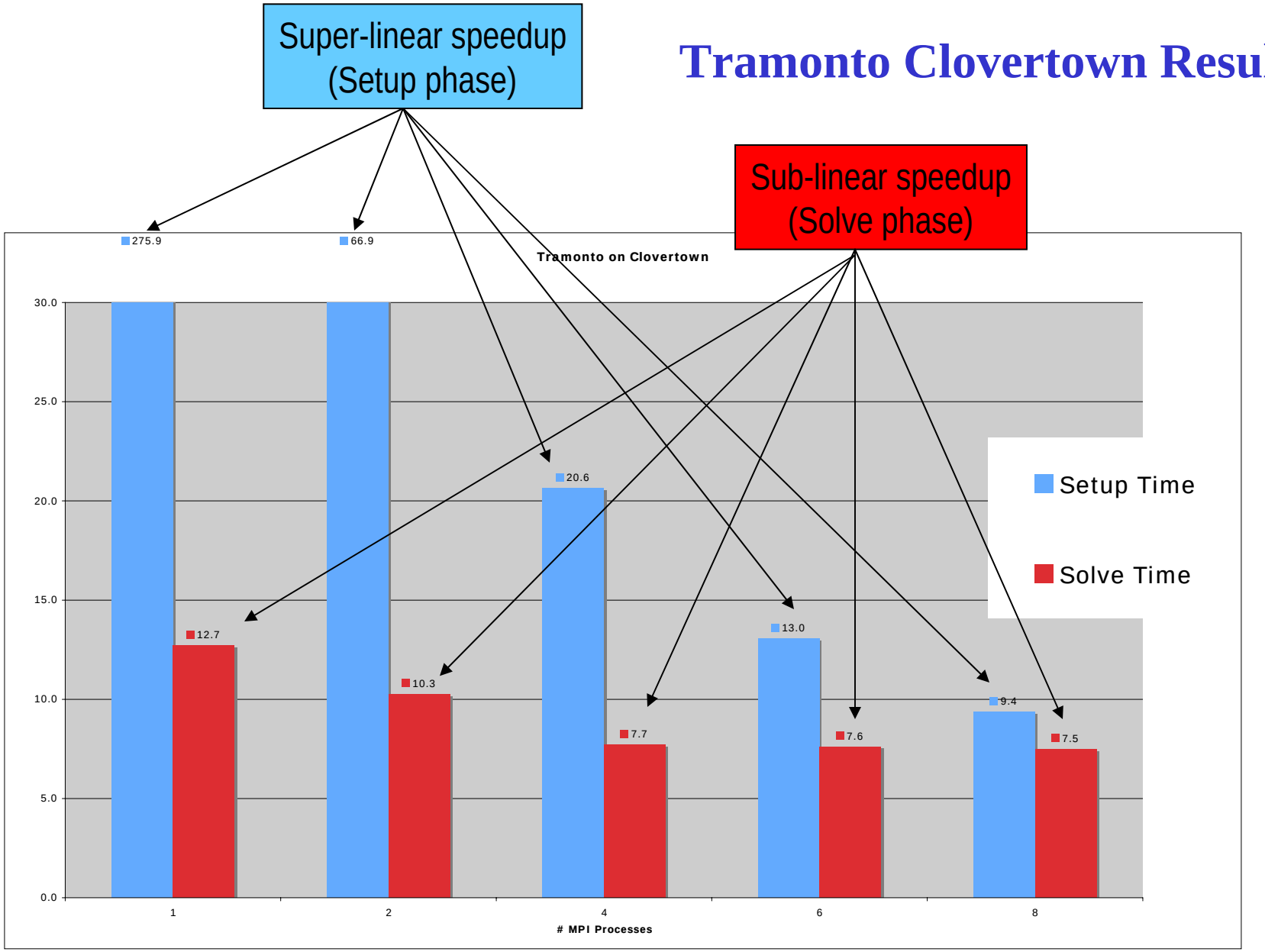
Epetra Kernels on Niagara2



	Speedup 64/1
SpMV	33.4
SpMM2	29.8
SpMM4	21.6
Norm2	26.3
Dot	14.0
Update	24.3

Sandia National Laboratories

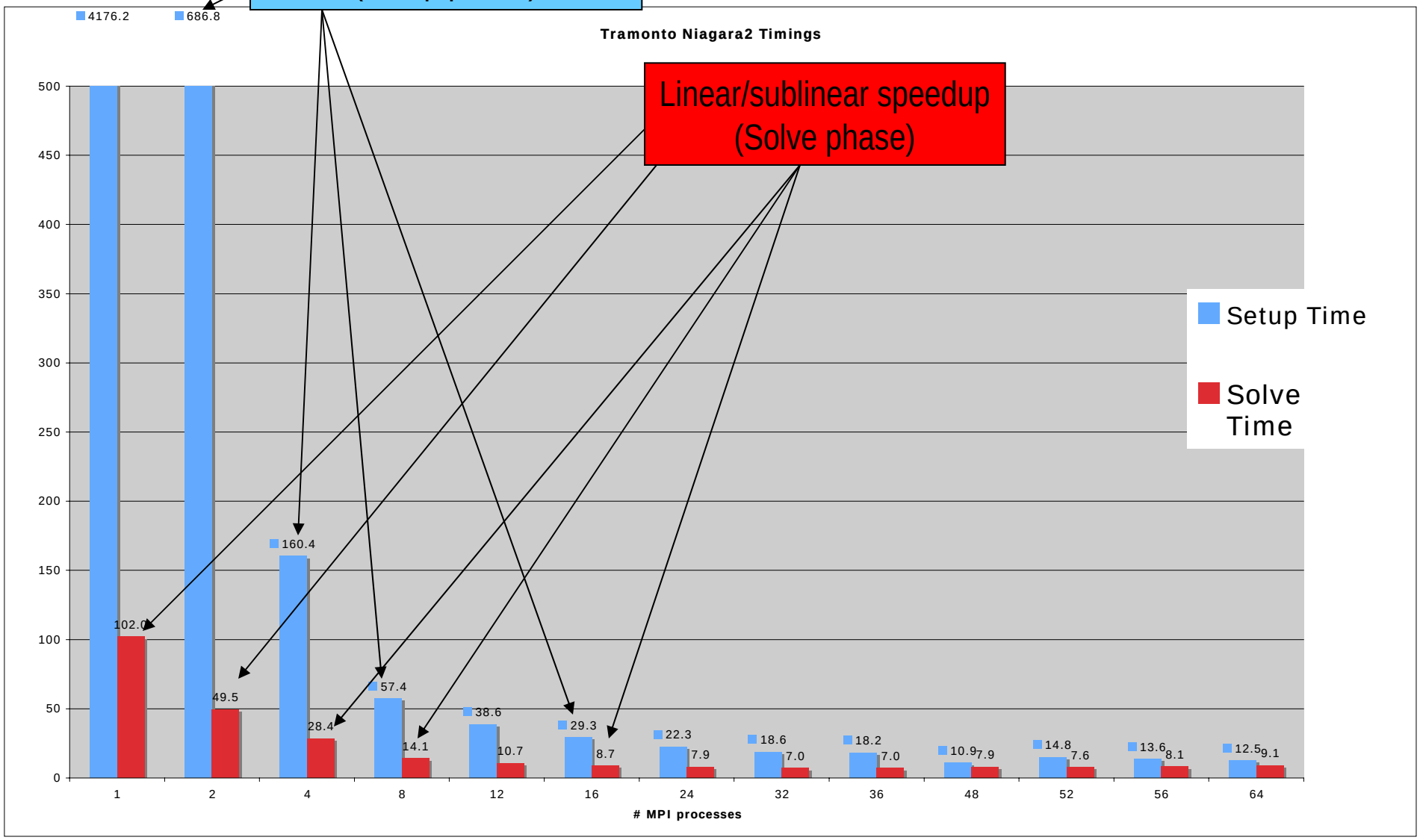
Tramonto Clovertown Results



- Setup (The application code itself): Excellent MPI-only.
- Solve (Trilinos libraries): Much poorer.

Sandia National Laboratories

Tramonto Niagara2 Results

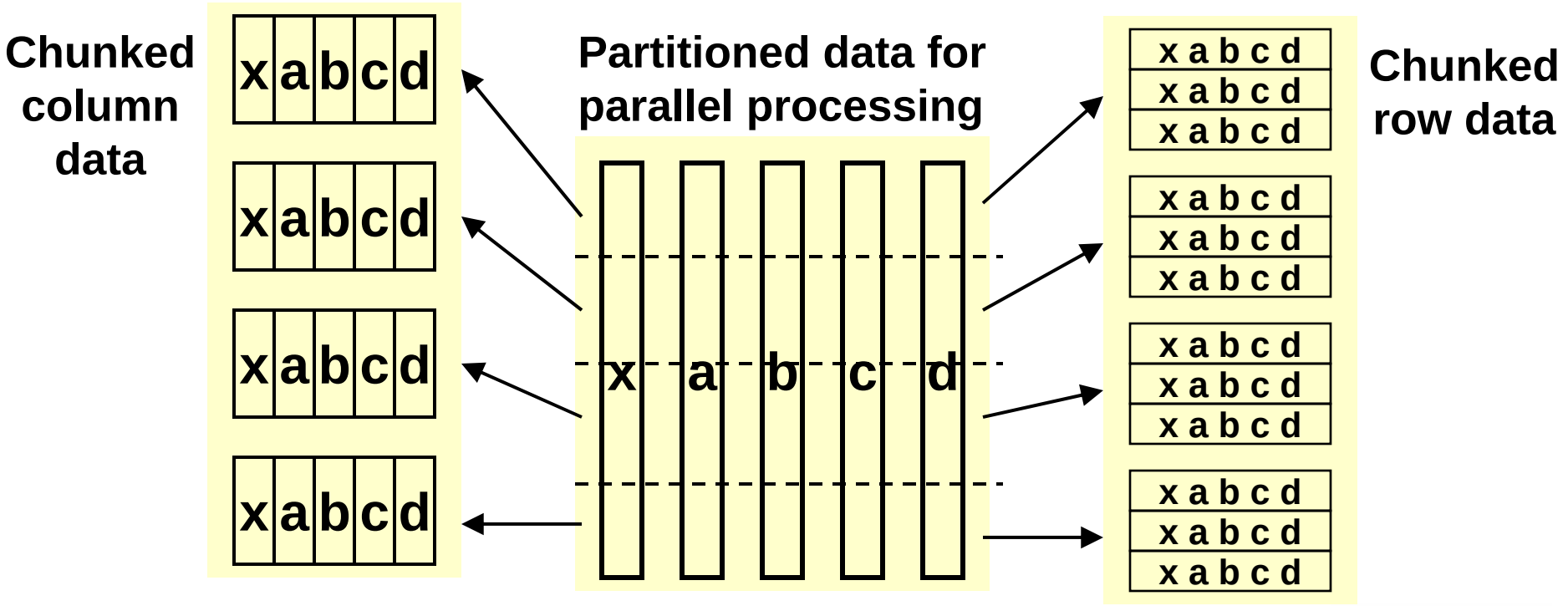


Sandia National Laboratories

Vector Multi-update

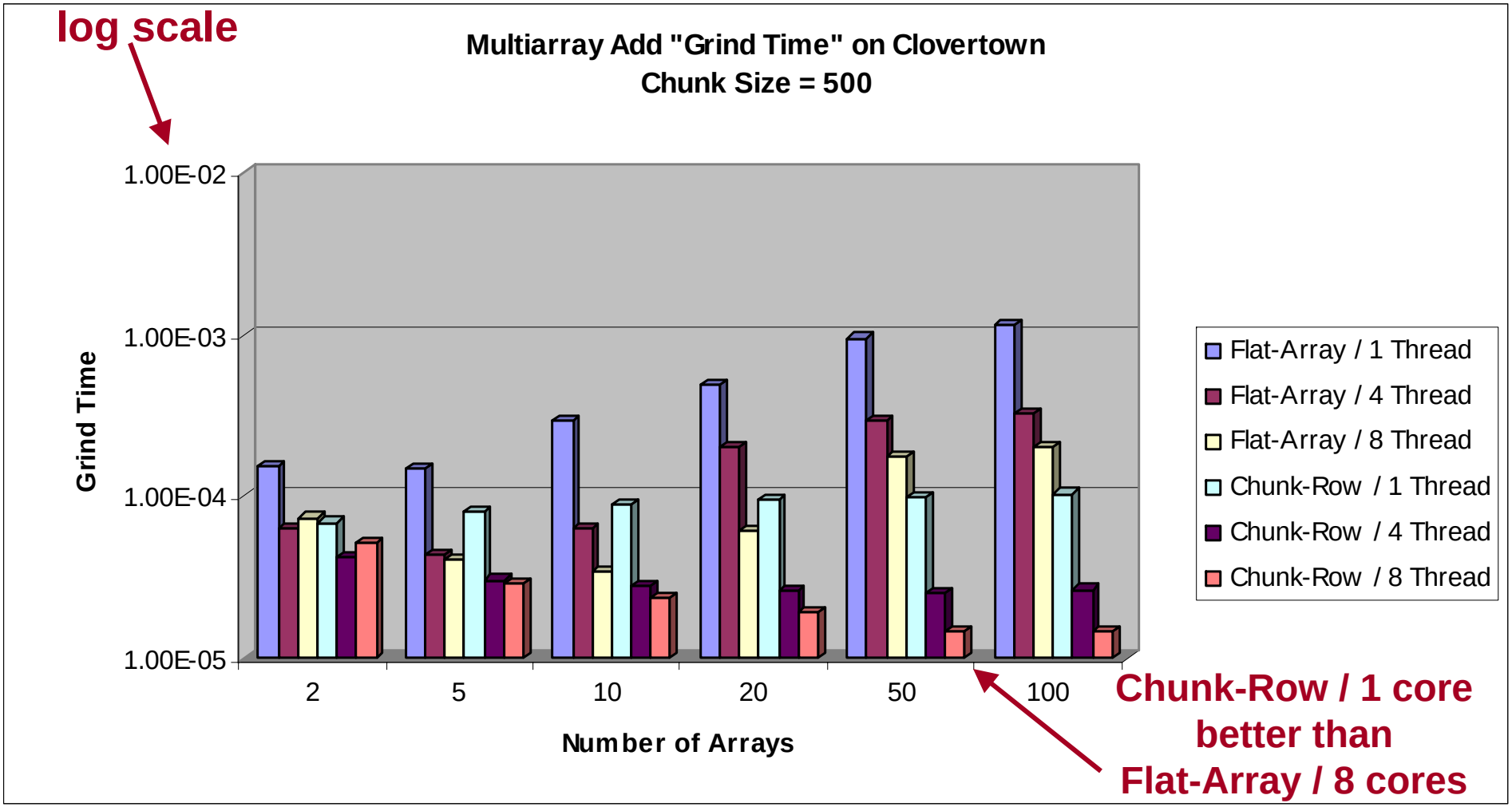
(courtesy H.C. Edwards)

- Cores compete for access to main memory
- Consider: x[i] = f(a[i], b[i], c[i], d[i], …); parallel on ‘i’
  - Compare performance of ‘Array’ versus ‘Chunk’ data structures



Sandia National Laboratories

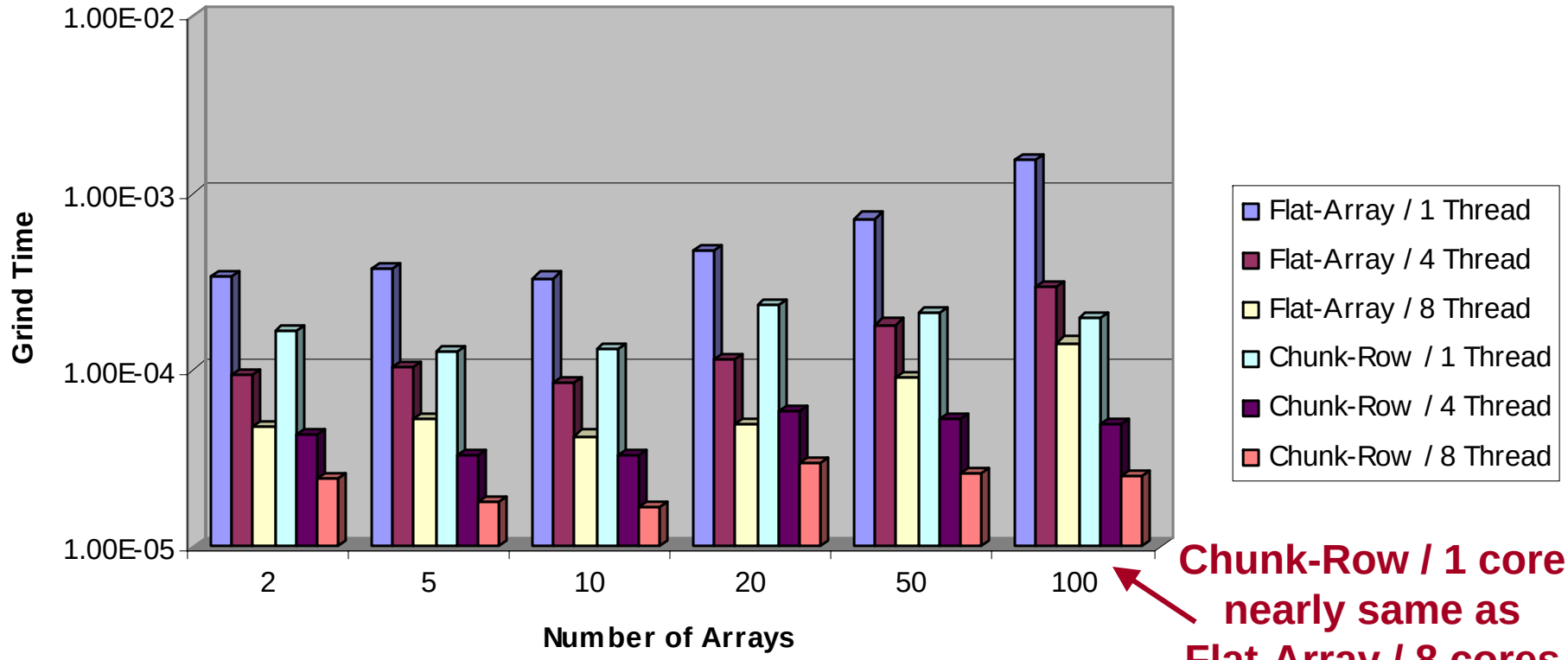
Chunked Data Structures Experiment
Clovertown – Scaling
Flat-Array 1,4,8 threads vs. Chunk-Row 1,4,8 threads



Sandia National Laboratories

Chunked Data Structures Experiment
Barcelona – Scaling
Flat-Array 1,4,8 threads vs. Chunk-Row 1,4,8 threads

log scale

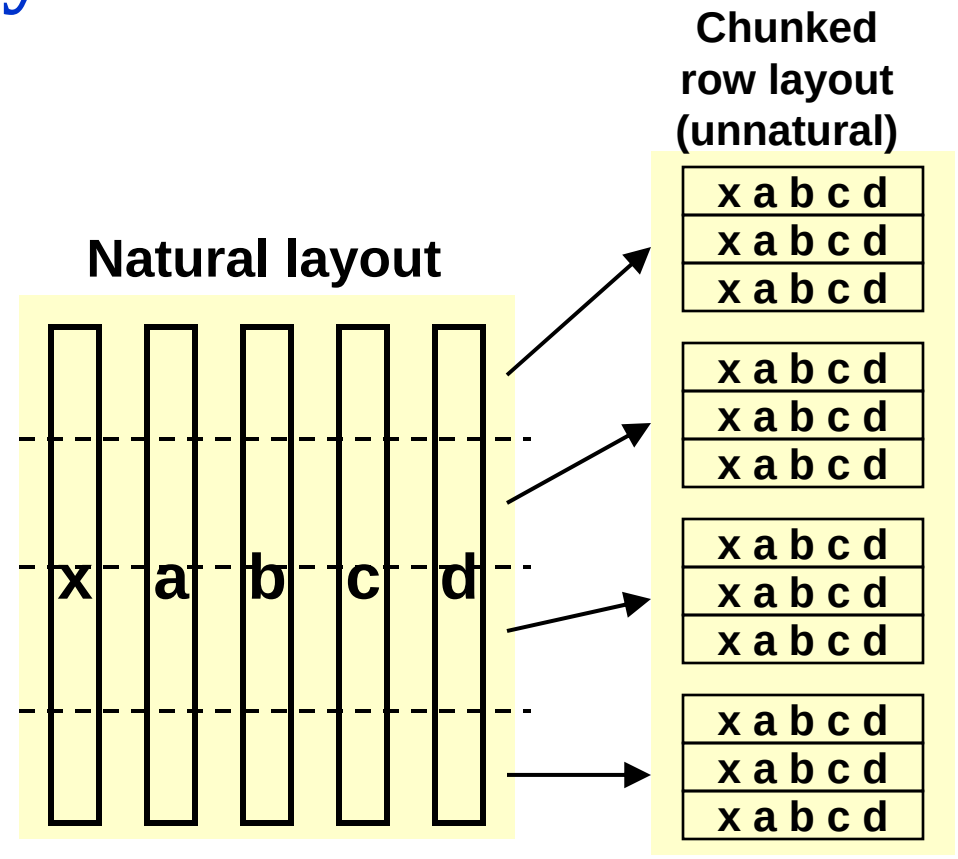


Chunk-Row / 1 core nearly same as Flat-Array / 8 cores

Sandia National Laboratories

Unnatural Data Layouts: Observations

- Unnatural layouts are troublesome.
- Have been around a long time: Dense BLAS
  - Actual compute layout different than user's
  - Compute rich: Translation done in real time.
- Sparse, vector computations much more challenging:
  - Translation (from natural to unnatural) cannot be done in real time.
  - Forces:
    - User to deal with unnatural layout or
    - Abstraction layer with temporal or spatial overheads.
  - Unnatural layout may have fastest kernel performance, but:
    - Overhead of translation.
    - Complexity of use.
  - Require careful interface design.



Sandia National Laboratories

Observations (So Far) for MPI Applications

1. MPI-only is a legitimate approach and the default.
2. Multicore will (probably, almost for-sure, very, very likely) change how we program the node.
   - Opinions on this are at both ends of spectrum and everywhere in between.
   - Uncomfortable defendinng MPI but: Bold predictions of MPI-only demise so far have proved false.
3. Simple programming model translation is ineffective.
4. Runtime environment is fragile: process/memory placement.
5. Bandwidth-intensive code problematic: Ineffective core use.

Opportunities (So Far) for MPI Applications

- Node architectures have the following trends:
  - Affinity for unit-stride memory access.
  - Essential benefit for using 32-bit float vs. 64-bit:
    - Storage/bandwidth: halved/doubled, resp.
    - Ops/sec: Double (increasing use of vector units, GPUS).
  - Opportunities for fine-grain shared memory parallel node algorithms.
- What we can do:
  - Access by unit stride if possible, even long vectors.
  - Explore the use of 32-bit float for steps not needing 64-bit.
  - Explore shared memory algorithms:
    - Not just a simple recasting of MPI processes (unless load imbalance an issue).
    - Improved algorithm, scheduling, etc.

Library Preparations for New Node Architectures (Decision Made Years Ago)

- We knew node architectures would change…
- Abstract Parallel Machine Interface: Comm Class.
- Abstract Linear Algebra Objects:
  - Operator class: Action of operator only, no knowledge of how.
  - RowMatrix class: Serve up a row of coefficients on demand.
  - Pure abstract layer: No unnecessary constraints at all. Temp
- Model Evaluator:
  - Highly flexible API for linear/non-linear solver services.
- Templated scalar and integer types:
  - Compile-time resolution float, double, quad,… int, long long,…
  - Mixed precision algorithms.

Library Effort in Response to Node Architecture Trends

- Block Krylov Methods (Belos & Anasazi):
  - Natural for UQ, QMU, Sensitivity Analysis…
  - Superior Node and Network complexity.
- Templated Kernel Libraries (Tpetra & Tifpack):
  - Choice of float vs double made when object created.
  - High-performance multiprecision algorithms.
- Threaded Comm Class (Tpetra):
  - Intel TBB support, compatible with OpenMP, Pthreads, …
  - Clients of Tpetra::TbbMpiComm can access static, ready-to-work thread pool.
  - Code above the basic kernel level is unaware of threads.
- Specialized sparse matrix data structures:
  - Sparse diagonal, sparse-dense, composite.
- MPI-only+MPI/PNAS
  - Application runs MPI-only (8 flat MPI processes on dual quad-core)
  - Solver runs:
    - MPI-only when interfacing with app using partitioned nodal address space (PNAS).
    - 2 MPI processes, 4 threads each when solving problem.

Sandia National Laboratories

Take Home Points

- Trilinos is a large and growing “project of projects”.
  - Trilinos 9.0 (Sep 2008) will contain many new packages.
  - Trilinos has new strategic leadership in place to handle growth.
- Software Engineering processes (when properly adapted) can positively impact algorithms R&D.
  - Trilinos Lifecycle Model is important milestone for us.
- New node architectures offer opportunities and challenges.
  - Characterization continues.
- We prepared for them from the beginning.
  - Vast majority of solver code will transparently port to new nodes.
- We are working toward effective use today.
  - Trilinos 9.0 will contain:
    - Thread programming support.
    - Full multi-precision support.
    - Mixed precision algorithms.
  - Availability of Trilinos 9.0 will enable variety of node programming approaches for new algorithm development.

Sandia National Laboratories

For More Information

- Trilinos:
  - http://trilinos.sandia.gov

- Tramonto:
  - http://software.sandia.gov/tramonto

- Mantevo:
  - http://software.sandia.gov/mantevo (under construction).

- My website:
  - http://www.cs.sandia.gov/~maherou

Sandia National Laboratories

Extra Slides

Sandia National Laboratories

Fluid Density Functional Theories: Tramonto

Sandia National Laboratories

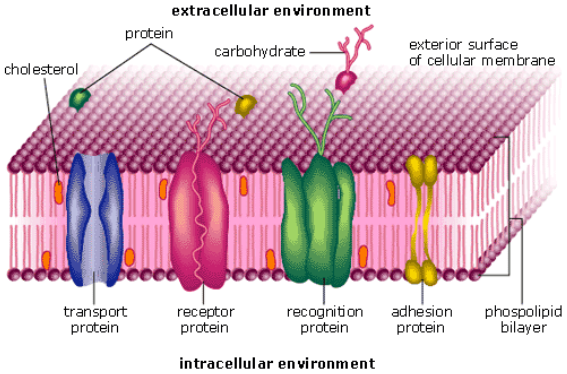
Collaborators

- Laurie Frink:
  - Primary developer of Tramonto.
  - Expertise: computational modeling of inhomogeneous fluids.
- Andy Salinger:
  - Other primary Tramonto developer.
  - Expertise: discretization methods and parallel application design.

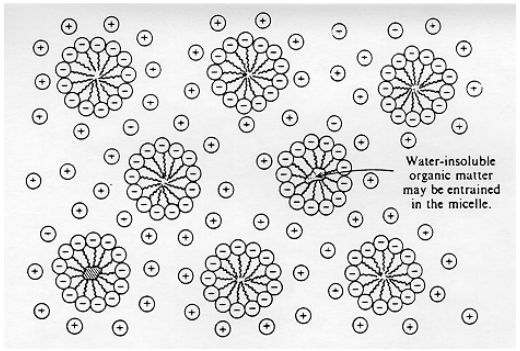
- My Role:
  - Solver algorithms.
  - Parallel implementation.

Sandia National Laboratories

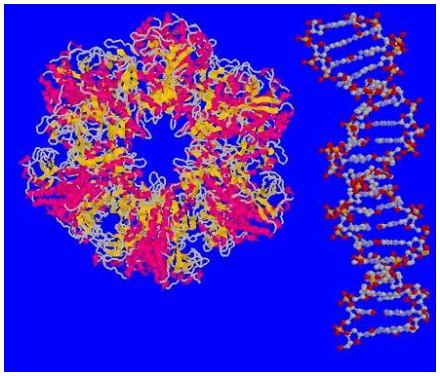
Complex fluid systems…



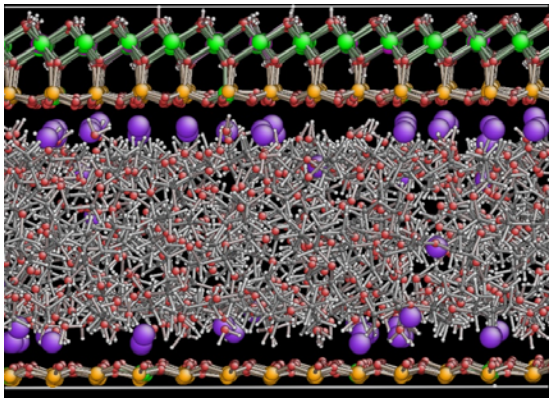
Cell membranes



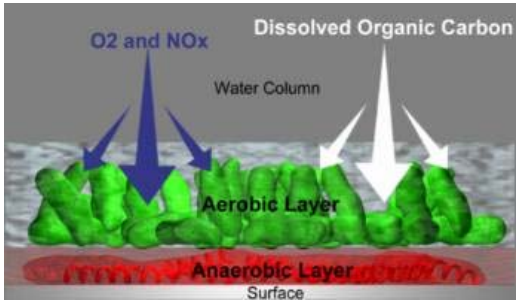
Colloidal/Amphiphilic systems
(www.science.duq.edu)



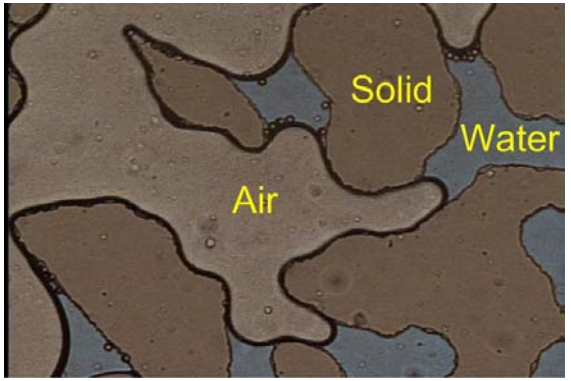
Biological Macromolecules
(www.hmi.de/people/kroy/rota.html)



Clay-polymer nanocomposites
(Univ. College London exclaim.org.uk)



Biofilms
(www.zetacorp.com)



Porous Media
(www2.bren.ucsb.edu/~keller/micromodels.html)

Problem characteristics
Interfacial fluids
Multiple length scales
Phase complexity

Sandia National Laboratories

DFT Acronym

- Quantum mechanical DFTs (QM-DFTs):
  - aka electronic DFTs.
  - Related but not discussed today.
- Fluid DFTs (F-DFTs):
  - aka classical DFTs.
  - We focus on this.
- Discrete Fourier Transforms (DFTs for F-DFTs):
  - Possible to use Fourier Transforms to work in frequency space.
  - FastTram: A version of Fourier Transform version of Tramonto (Mark Sears).
  - Restricted applicability: BCs, preconditioning.
- Real-space approach: Use spatial variables.
  - We focus on this.
  - From this point on: DFT means real-space F-DFTs.

DFT for fluids

The free energy functional … (a) The theory is exact, but the precise nature of the equations often cannot be derived. (b) Approximate functionals have been developed often as perturbations to a hard sphere reference system.

Coulomb interactions

Associations (H-bonding)

[Applied field]

$$\Omega[\rho(r)] = F_{id} + F_{hs} + F_{vdW} + F_c + F_{assoc} + \int \rho(r)[V(r) - \mu]$$

Ideal gas

Hard sphere

Dispersion attractions

Legendre Transform from Canonical to Grand canonical ensemble

We seek the the stationary states of the free energy functional with the understanding that the thermodynamically relevant state should be found at the global free energy minimum.

$$\frac{\delta\Omega}{\delta\rho(r)}_{\mu,T} = 0$$

Sandia National Laboratories

Properties of F-DFT systems

DFT - Integral equations of finite range
(matrix density is system size dependent)

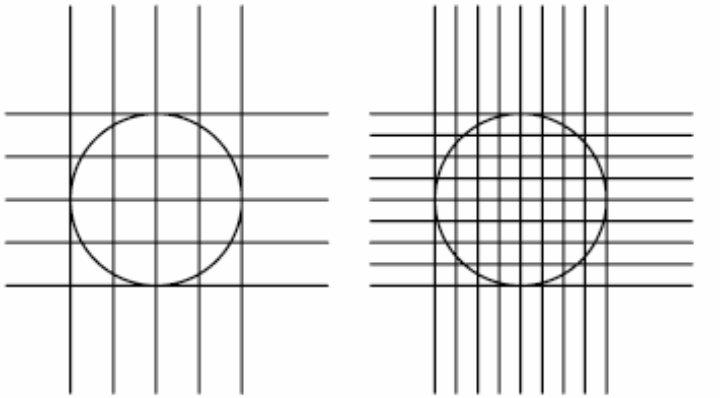
PDE - matrix density independent of system size.

DFT- Inter-physics coupling dominates

PDE - Inter-nodal coupling dominates

DFT - Stencils based on physical constants

PDE - Stencils based on nearest neighbors



DFT - May have large numbers of DOF per node
- HS (3D) 10+
- Polymer (20 beads) 42+
- Most DOFs are “constraints” on densities.

PDE - Usually a few DOFs per node

Sandia National Laboratories

General Segregation Strategy

- Observations:
  - Internodal coupling is weak.
  - Some DOFs have simple “one-way” dependence.
- Idea:
  - Organize DOFs physics-first.
  - Reorder blocks of DOFs to expose one-way dependences.
  - Apply 2-by-2 block partitioning such that one-way dependencies are in A_{11} block.
  - Use Schur complement on A_{22} block.

General Strategy

$$\left(\begin{array}{ccc|ccc} A_{11}^{11} & \cdots & A_{11}^{1j} & A_{12}^{1,j+1} & \cdots & A_{12}^{1k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ A_{11}^{j1} & \cdots & A_{11}^{jj} & A_{11}^{j1} & \cdots & A_{11}^{jj} \\ \hline A_{21}^{j+1,1} & \cdots & A_{21}^{j+1,j} & A_{22}^{j+1,j+1} & \cdots & A_{22}^{j+1,k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ A_{21}^{k1} & \cdots & A_{21}^{kj} & A_{22}^{k,j+1} & \cdots & A_{22}^{kk} \end{array}\right) \left(\begin{array}{c} x_1^1 \\ \vdots \\ x_1^j \\ \hline x_2^{j+1} \\ \vdots \\ x_2^k \end{array}\right) = \left(\begin{array}{c} b_1^1 \\ \vdots \\ b_1^j \\ \hline b_2^{j+1} \\ \vdots \\ b_2^k \end{array}\right)$$

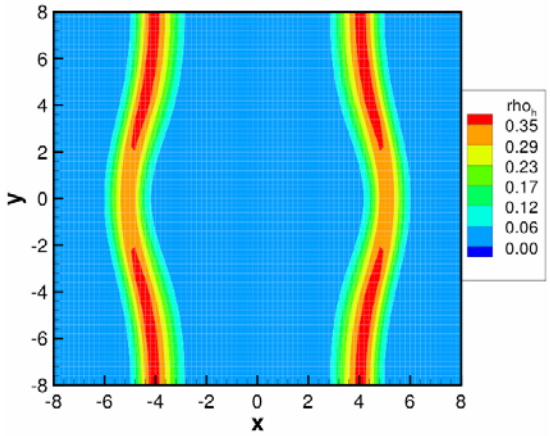
- Identify and order DOFs in A_{11} block so that A_{11}^{-1} easy to apply.
- Implicitly (or explicitly in some instances) form Schur complement system:

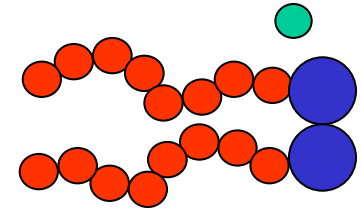
$$Sx_2 = (A_{22} - A_{21}A_{11}^{-1}A_{12})x_2 = b_2 - A_{21}A_{11}^{-1}b_1$$

- Solve Schur system via preconditioned GMRES.
- Solve finally for x_1.

Sandia National Laboratories

Second class of Problems: Self-assembly of lipid bilayers





8-2-8 Chain

Sandia National Laboratories

A 2nd Case…CMS-DFT / polymers

- developed for polymers
- chains are flexible
- 2nd order density expansion

Chandler, McCoy, Singer (1986);
McCoy et al. (1990s)

$$\rho_\alpha(r) = \frac{\rho_\alpha^b}{N_\alpha} \sum_{s=1}^{N_\alpha} \frac{G_s(r) G_s^i(r)}{e^{-\beta U_\alpha(r)}}$$

Chain density distribution

$$U_\alpha(r) = V_{ext}(r) - \sum_\gamma \int c_{\alpha\gamma}(r - r')[\rho_\gamma(r') - \rho_\gamma^b]dr'$$

Mean field

$$c(r) = c_{rep}(r) - u_{att}(r)$$

PRISM Theory **RPM Approx**

$$G_s(r) = e^{-\beta U_{\alpha,s}} \int w(r - r') G_{s-1}(r') dr'$$

$$G_s^i(r) = e^{-\beta U_{\alpha,s}} \int w(r - r') G_{s+1}^i(r') dr'$$

$$G_1 = G_N^i = e^{-\beta U(r)}$$

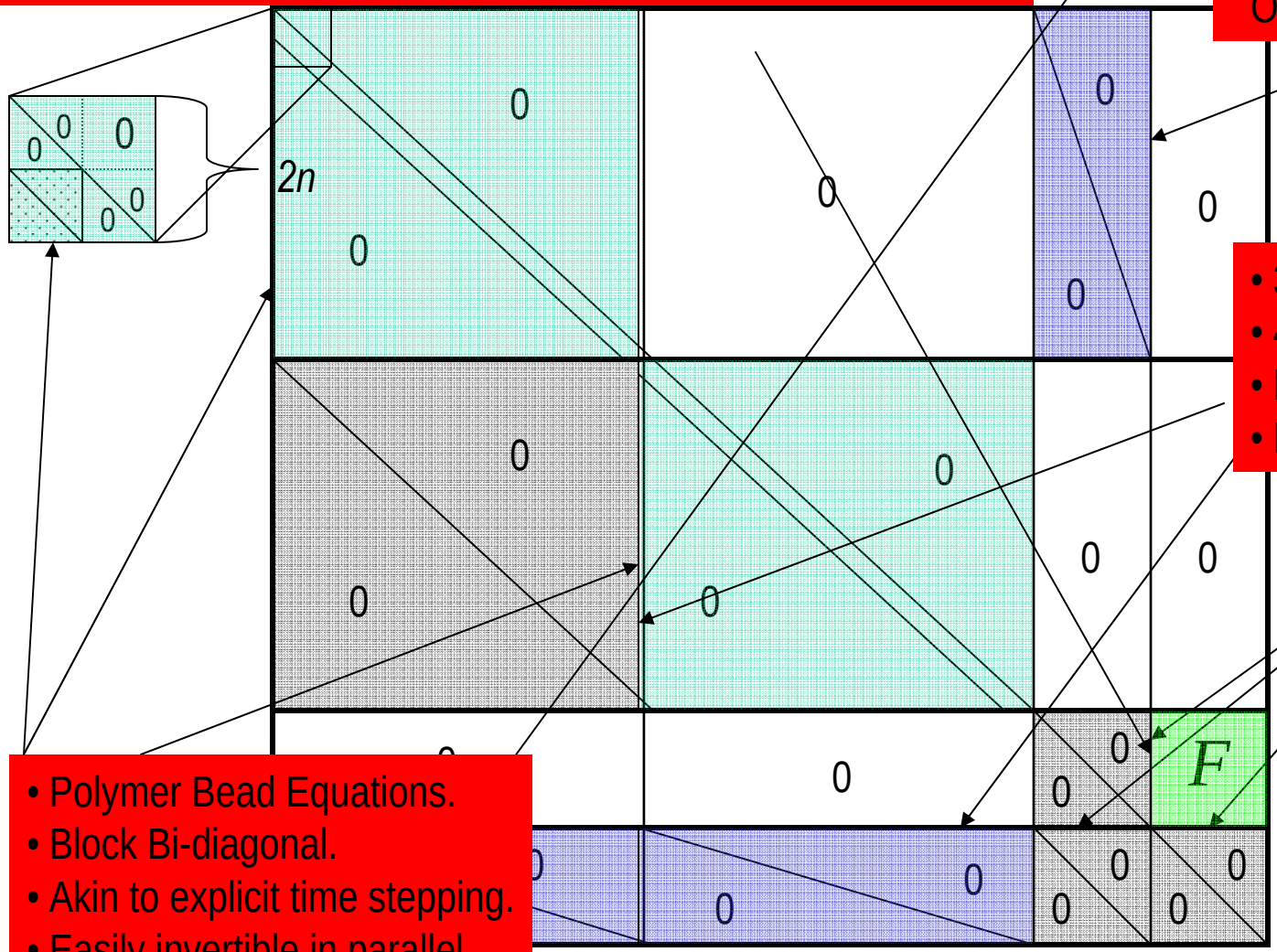
$$w(r) = \frac{1}{4\pi\sigma^2} \delta(|r| - \sigma)$$

Chain Architecture (freely-jointed chains)

Sandia National Laboratories

- There is only ONE interesting block in this whole matrix.
- *F* describes CMS field dependence on primitive densities.
- 2.5 σradius integral at each grid node (mesh independent).
- Not sparse, nor dense. Constant coefficient.

- Diagonal-like.
- One non-zero per row/col in long dimension.
- Like Prolongation/restriction Operators?



- 3rd block: CMS Field
- 4th block: Prim Densities
- Diagonal matrices.
- No spatial coupling.

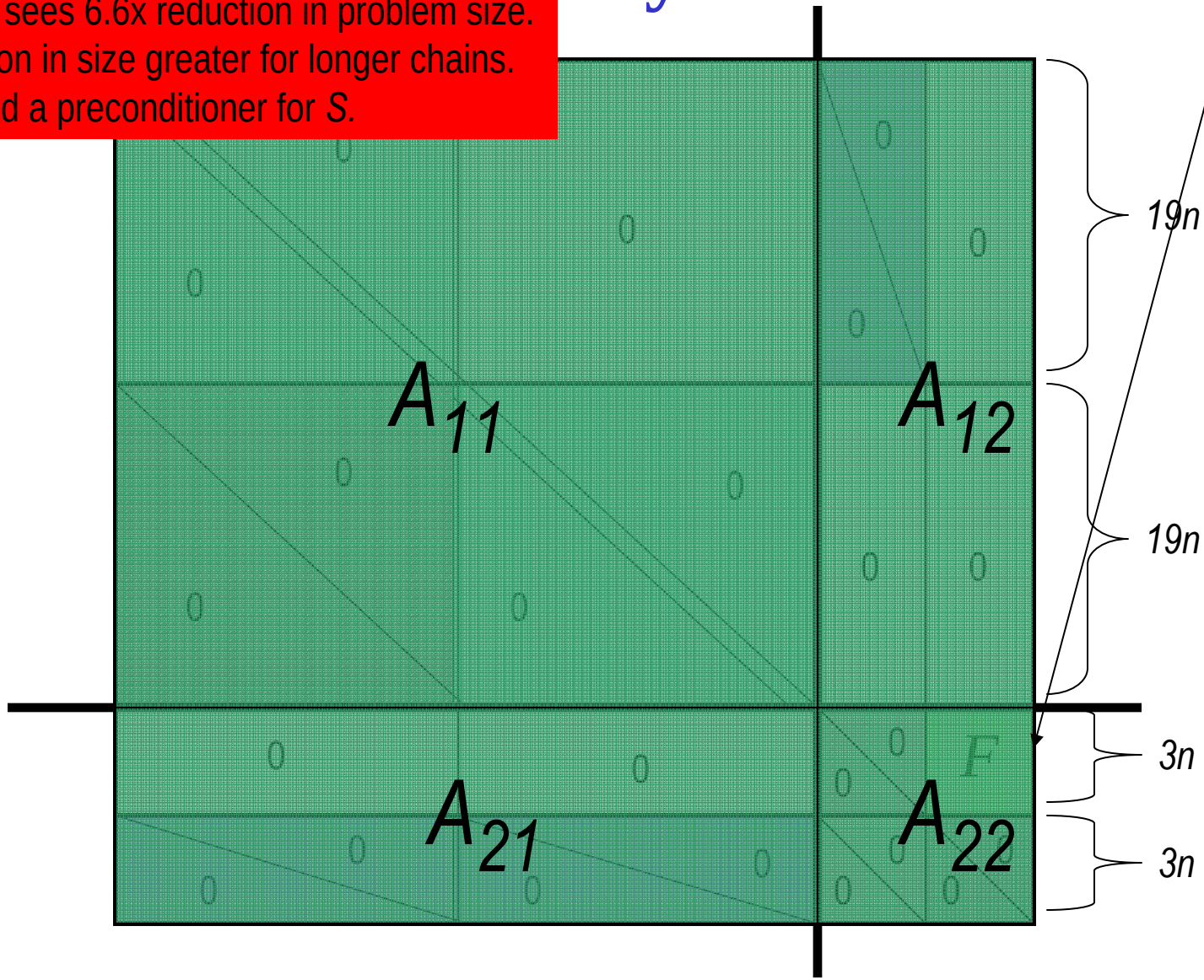
- Polymer Bead Equations.
- Block Bi-diagonal.
- Akin to explicit time stepping.
- Easily invertible in parallel.

Sandia National Laboratories

Layer Problem

- Last layer of structure: 2-by-2 partitioning.
- A_{11} solve easily applied in parallel.
- Apply GMRES to $S = A_{22} - A_{21}*inv(A_{11})*A_{12}$
- GMRES sees 6.6x reduction in problem size.
- Reduction in size greater for longer chains.
- Still need a preconditioner for S.

- F has strong overlap:
  Distribute separate from rest of problem.



Sandia National Laboratories

Preconditioner for *S*

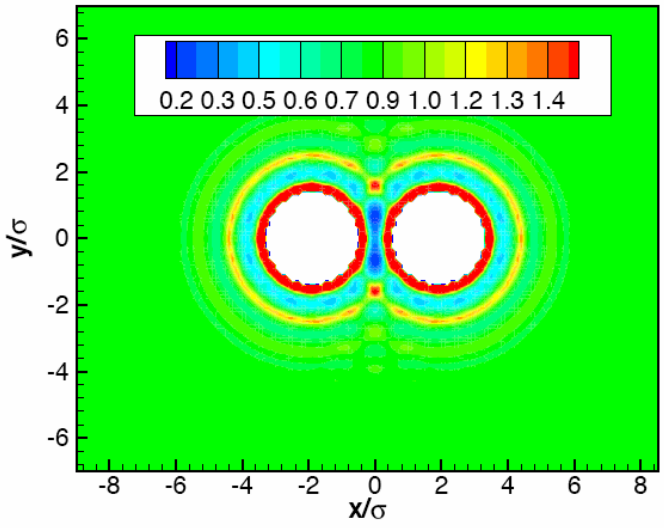
$$A_{22} = \begin{bmatrix} D_{11} & F \\ D_{21} & D_{22} \end{bmatrix}$$

$$A_{22} \approx \tilde{A}_{22} = \begin{bmatrix} D_{11} & F \\ 0 & D_{22} \end{bmatrix}$$

- $D_{11}, D_{22} = O(1), \; D_{21} = O(1e\text{-}10)$
- Ignore D_{21} for preconditioning.
- *P(S)* requires
  - 2 diagonal scalings,
  - matvec with F.
- All distributed operations.

Sandia National Laboratories

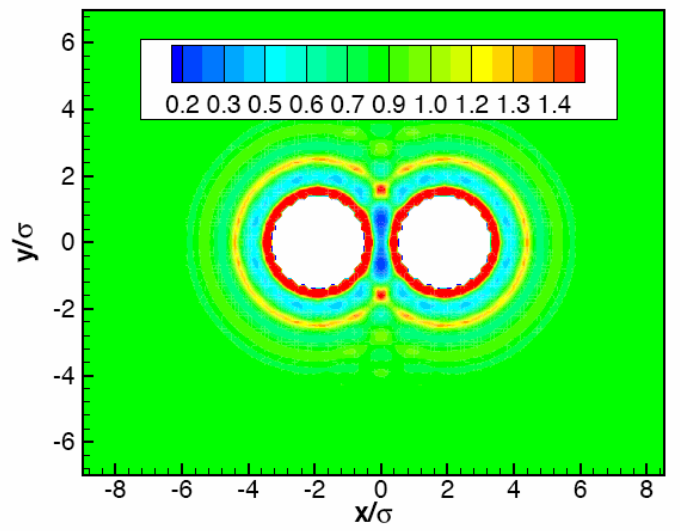
Parallel Scaling - CMS-DFT

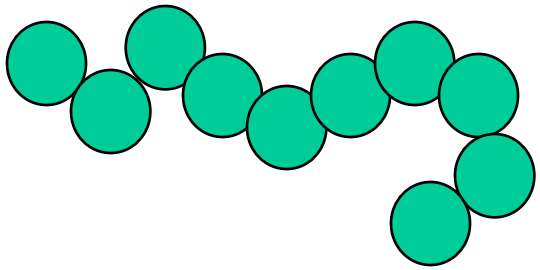


#Procs	N_{iter}	<# Lin iters>		Time/N_{iter}		T_{1Proc}/T	T_{FullA}/T_{SA22}
		FullA	*SA22*	*FullA*	*SA22*	(*SA22*)	
1	9		70	-	88	1	-
2	9		70	-	45	2.0	-
4	9	25	70	290*(8)	26.2	3.4	11.0
8	9	28	70	77	12	7.3	6.4
16	10	33	70	23	6.7	13.1	3.4
32	8	40	70	7.1	3.8	23.2	1.9
64	9	46	69	2.8	2.2	40.0	1.3
128	9	58	69	1.4	1.8	49.0	0.8

Sandia National Laboratories

Scaling with chain length





N_{seg}	N_{iter}	<# Lin iters>		Time/N_{iter}		$T/T_{N_{seg}=10}$ $(SA22)$	T_{FullA}/T_{SA22}
		FullA	*SA22*	*FullA*	*SA22*		
10	8	40	70	7.1	3.7	1	1.9
20	8	48	69	21.7	8.9	2.4	2.4
40	8	62	70	58.7	15.1	4.1	3.9
80	7	93	68	257.4	30.9	8.4	8.3

Sandia National Laboratories

Stats for Polymer A_{11} block

- 38 DOFs in A_{11} : 37 Epetra_CrsMatrix objects.
- ApplyInverse call requires 38 parallel vector updates interleaved with 37 matvecs.
- This is for 18-length polymer chain.
- 100-length chain: 199 Epetra_CrsMatrices.

Summary

Emphasis on algorithms has impacted applications work in a significant way.

Many complex 3D systems can be studied **now.**

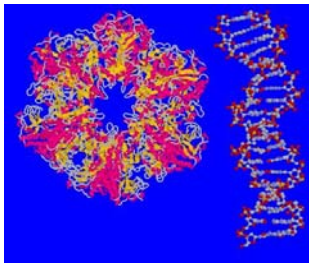
Much more work to be done

- Parallel Partitioning
- DFTs with greater complexity
- Optimization of preconditioners
- Solution complexity and physical phases
- Design applications
- Coupled (multiscale) methods
- Other better approaches

Sandia National Laboratories

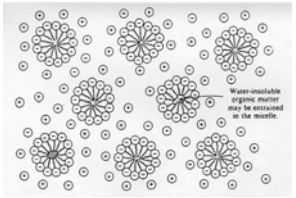
Summary, cont.

- New family of scalable solvers for complex fluid systems in Tramonto.
- Properties:
  - No tuning parameters.
  - Robust to processor count increase.
  - 5-20 times memory use reduction over previous approaches.
  - *O(10)-O(100)* reduced implicit problem size.
  - Nearly linear scalability in: processor count, mesh density, polymer chain length.
  - Candidate for petascale class computing.

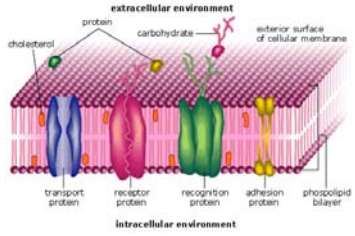


Biological Macromolecules
(www.hmi.de/people/kroy/rota.html)

- Enables:
  - Fundamentally new calculations for important bio problems. Quotes from *Physical Review Letters* referees on computations using these solvers:
    - "This is (to my knowledge) the first time [Fluid] DFT has been used to analyze the important problem of pore structure in biological membranes."
    - "This appears to me to be a highly significant advance in theoretical biophyics, even by the high standards of Physical Review Letters. I suspect that this Sandia group is the only one in the world to have developed classical DFT methods sufficiently sophisticated to deal with such a remarkably complex problem in colloidal physics…"
    - "…I would then recommend at least a footnote that gives some introductory hint as to how they have managed to cope numerically with such complex structures; presumably a 3d finite element method with **all manner of tricks**?
  - The "tricks" are the solvers.
  - *Parallel Segregated Schur Complement Methods for Fluid Density Functional Theories, M. Heroux, L. Frink, A. Salinger* to appear in SIAM SISC.
  - Tramonto first public release this year.



Colloidal/Amphiphilic systems
(www.science.duq.edu)



Cell membranes

http://software.sandia.gov/tramonto

Welcome to the Tramonto home page: Software for Nanostructured Fluids in materials and biology

Project goals

This project is based at Sandia National Laboratories, and is focused on developing molecular theory based computational tools for predicting the structure and properties of fluids at the nanoscale near surfaces and macromolecules. At this length scale fluids are inhomogeneous and common approximations for bulk fluids such as incompressibility do not apply. The specific capabilities of Tramonto and the related FasTram software packages are detailed in the Capability links to the left. In both cases, the molecular theories treated by the codes are fluid density functional theories (F-DFTs). These theories compute fluid structure near surfaces or as a result of self-assembly in contrast to quantum density functional theories (Q-DFTs) which are widely used to compute electronic structure of materials.

Applications of Fluids-DFTs

Fluids Density Functional Theory approaches have been used to study a wide range of physical systems. Some examples are: fluids at interfaces, surface forces, colloidal fluids, wetting, porous media, capillary condensation, interfacial phase transitions, nucleation phenomena, freezing, self-assembly, lipid bilayers, ion channel proteins, solvation of surfaces and molecules. The characteristic particle size in F-DFT models ranges from atoms (e.g Argon) to colloidal particles, proteins, or cells. Thus these F-DFT approaches provide a multiscale framework for studying the physcis of many complex fluid systems. Some of these applications are represented in the publication list on the left, others may be found in a very diverse literature. The Tramonto code does not capture all of the F-DFT approaches that have been developed to date, but can be extended to new theories and models.

Motivation - a Scientific Computing perspective.

Until recently, application of F-DFTs to problems in inhomogeneous fluids was limited primarily to systems with two dimensions of symmetry allowing for 1-dimensional computations. In that domain, fast calculations can be performed on single processer computers using algorithms of limited sophistication (e.g. Picard iterations).

Two and three dimensional calculations for F-DFTs are much more costly due to the integral nature of the systems of equations. To understand the computational cost, consider the differences between partial differential equations (PDEs) and the integral equations associated with F-DFTs. The nodes in PDEs generally interact only with nearest neighbors or next nearest neighbors often resulting in diagonally dominant sparse matrices. As the mesh is refined the number of interactions remains constant although there are more nodes to process. In the case of DFTs the range of the integration stencils is significantly longer based on the underlying physics included in