

SC2005 – Seattle, WA

ACTS

An Infrastructure for Maintaining DOE Software

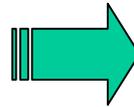
Osni Marques and Tony Drummond

Lawrence Berkeley National Laboratory (LBNL)

[OAMarques,LADrummond]@lbl.gov

Challenges in the Development of Scientific Codes

- Research in computational sciences is fundamentally interdisciplinary
- The development of complex simulation codes on high-end computers is not a trivial task
- Productivity
 - Time to the first solution (prototype)
 - Time to solution (production)
 - Other requirements
- Complexity
 - Increasingly sophisticated models
 - Model coupling
 - Interdisciplinarity
- Performance
 - Increasingly complex algorithms
 - Increasingly complex architectures
 - Increasingly demanding applications



- Libraries written in different languages
- Discussions about standardizing interfaces are often sidetracked into implementation issues
- Difficulties managing multiple libraries developed by third-parties
- Need to use more than one language in one application
- The code is long-lived and different pieces evolve at different rates
- Swapping competing implementations of the same idea and testing without modifying the code
- Need to compose an application with some other(s) that were not originally designed to be combined

What About Software Selection?

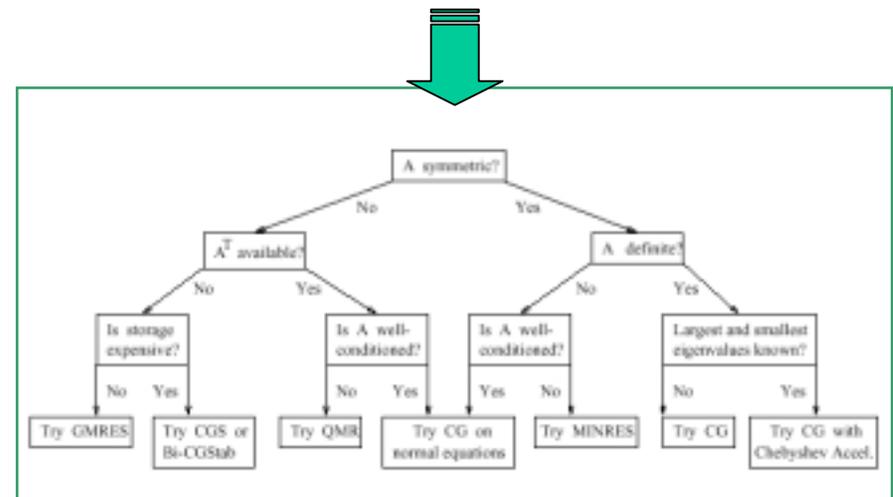
Example: $Ax = b$

$$A = LU$$

$$x_k = x_{k-1} + \alpha_k p_{k-1}$$

- Use a direct solver if
 - Time and storage space acceptable
 - Iterative methods don't converge
 - Many b 's for same A
- Criteria for choosing a direct solver
 - Symmetric positive definite (SPD)
 - Symmetric
 - Symmetric-pattern
 - Unsymmetric
- Row/column ordering schemes available
 - MMD, AMD, ND, graph partitioning
- Hardware

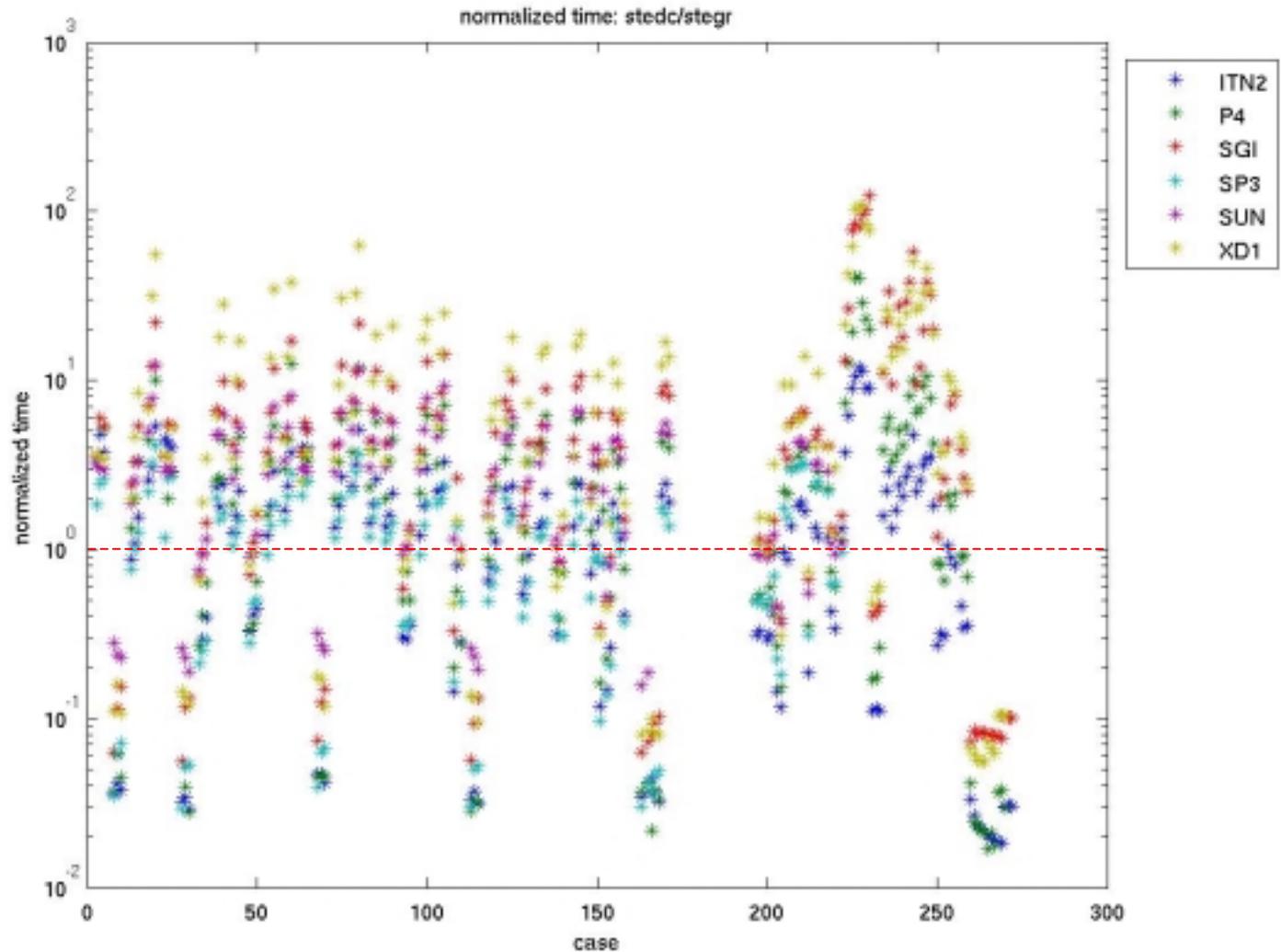
Build a preconditioning matrix K such that $Kx=b$ is much easier to solve than $Ax=b$ and K is somehow “close” to A (incomplete LU decompositions, sparse approximate inverses, polynomial preconditioners, element-by-element preconditioners, etc).



Software Selection: *Performance Issues*

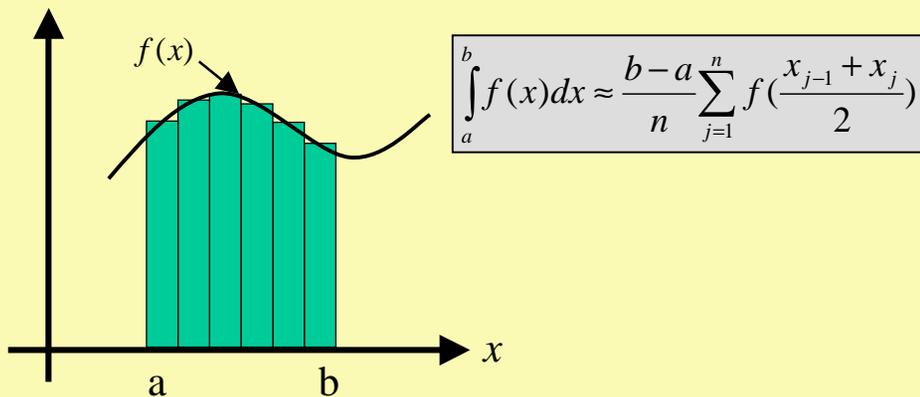
Relative performance of two tridiagonal eigensolvers on 6 platforms:

Performance varies depending on the application and the architecture!

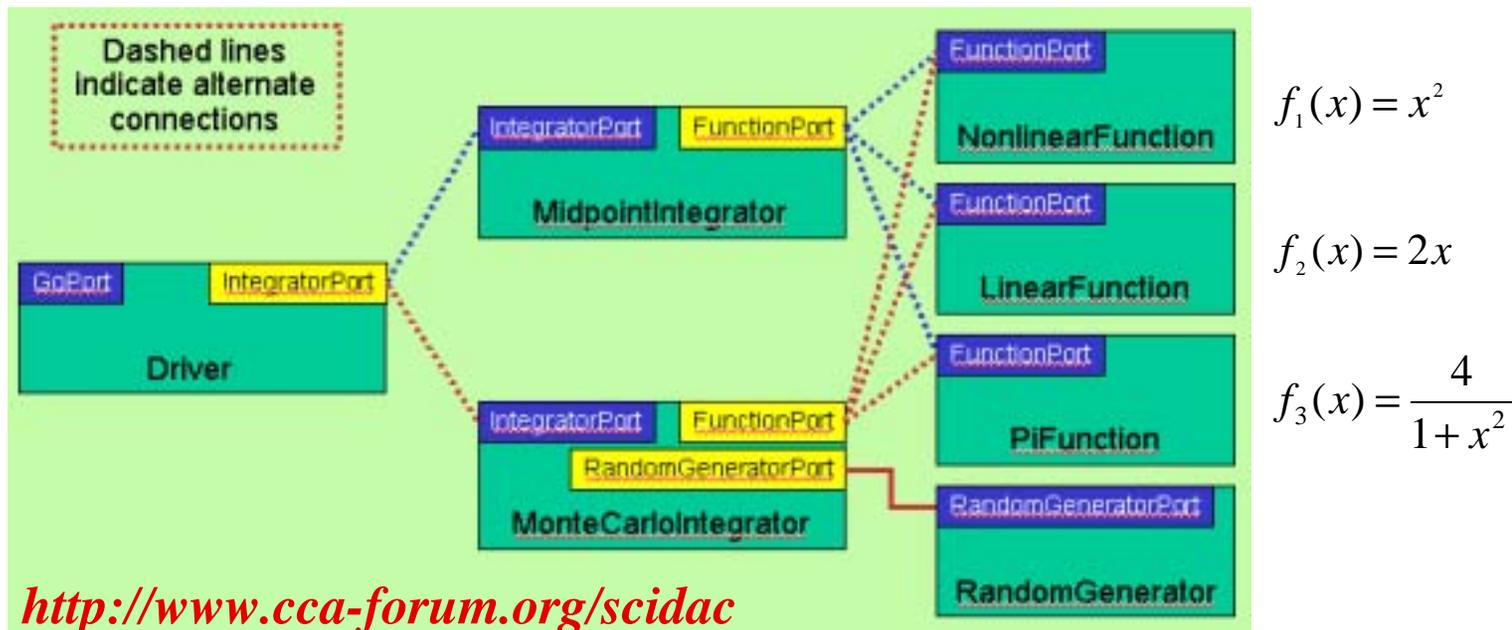
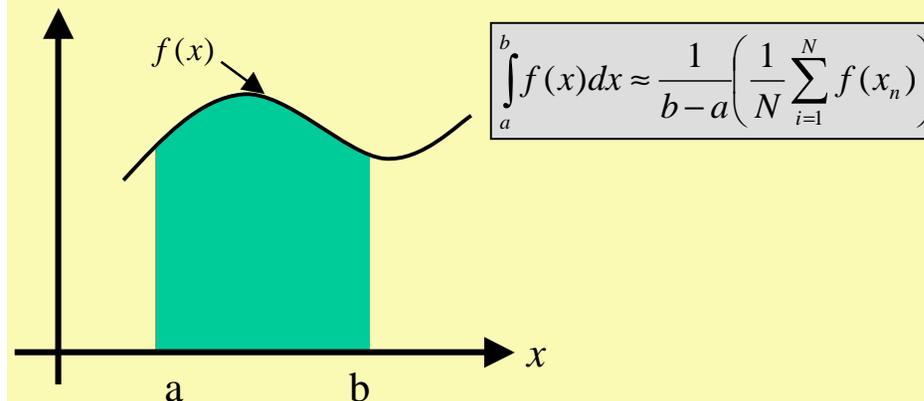


Components: *simple example*

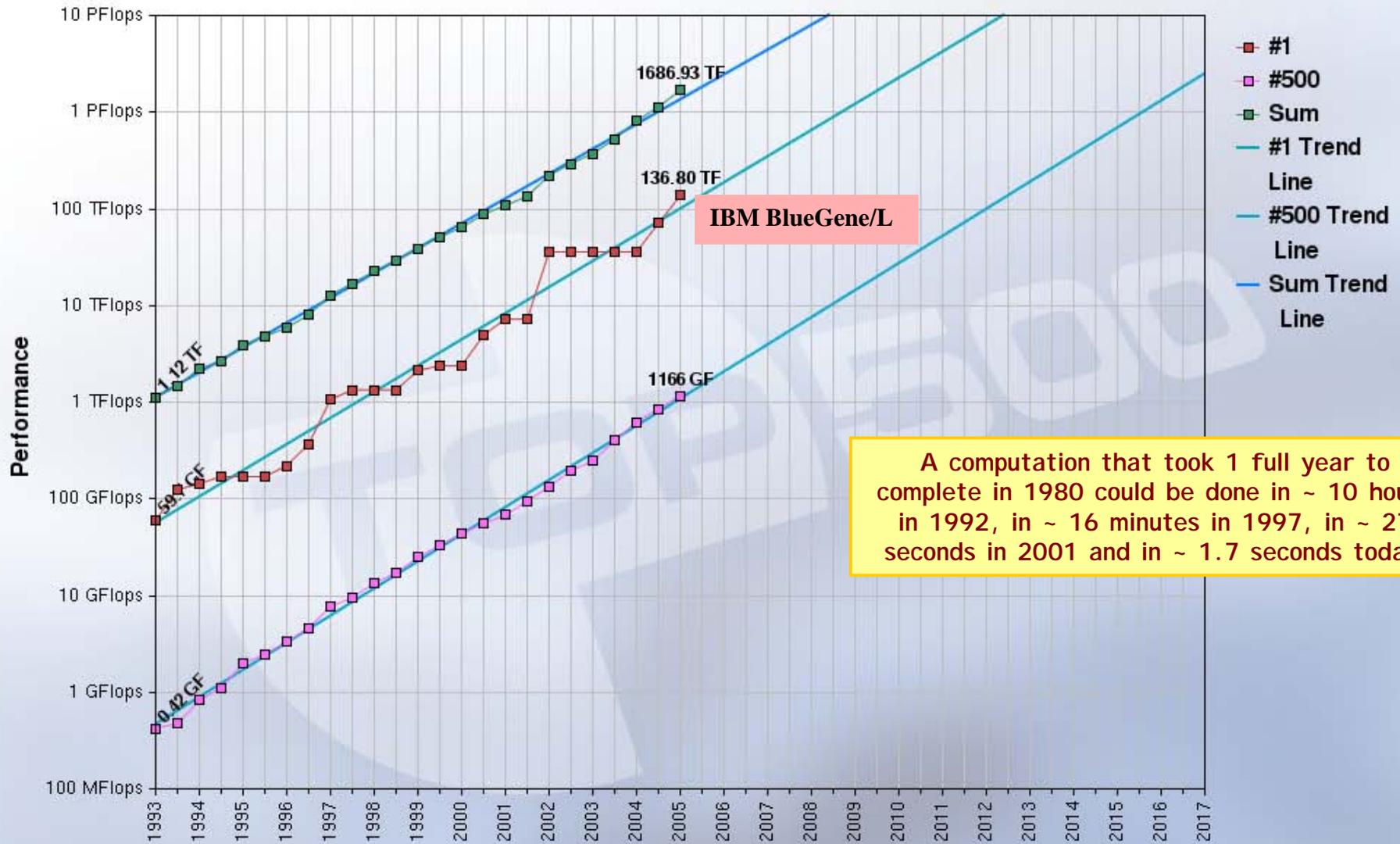
Numerical integration: midpoint



Numerical integration: Monte Carlo



Projected Performance Development

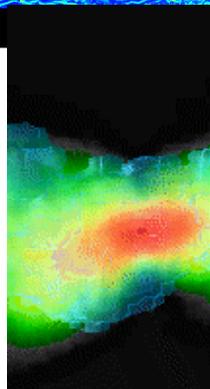
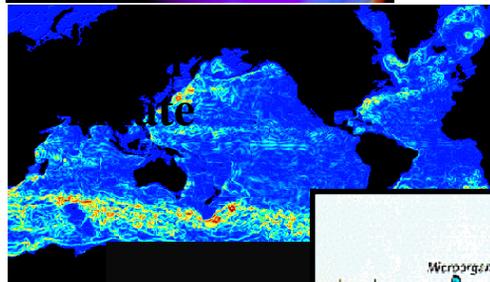
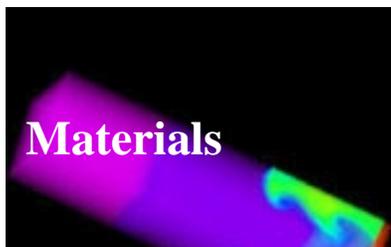


A computation that took 1 full year to complete in 1980 could be done in ~ 10 hours in 1992, in ~ 16 minutes in 1997, in ~ 27 seconds in 2001 and in ~ 1.7 seconds today!

A Sample of Applications of Interest

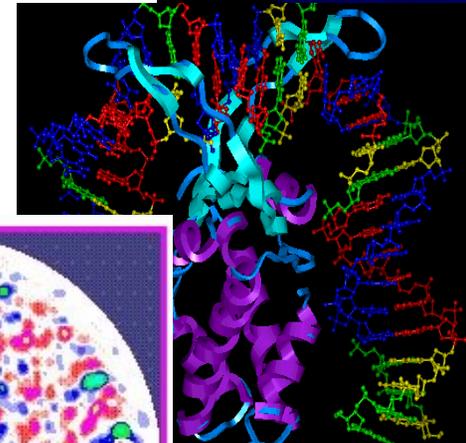
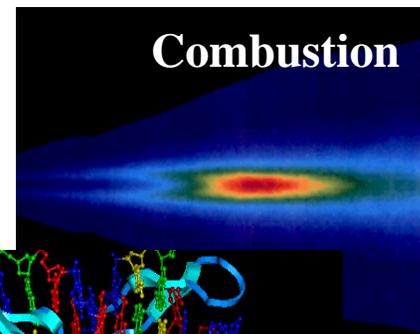
Scientific Computing – *Third Pillar of Science*

<http://www.science.doe.gov/scidac>

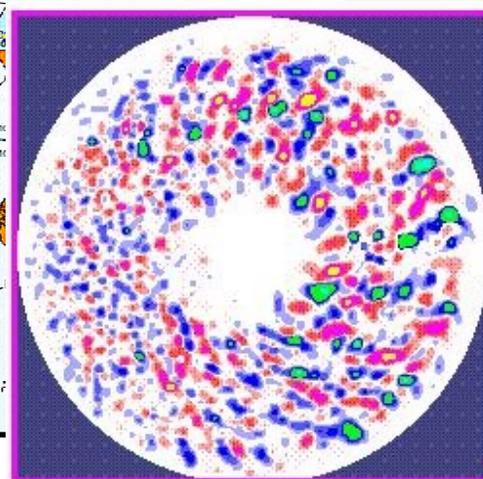
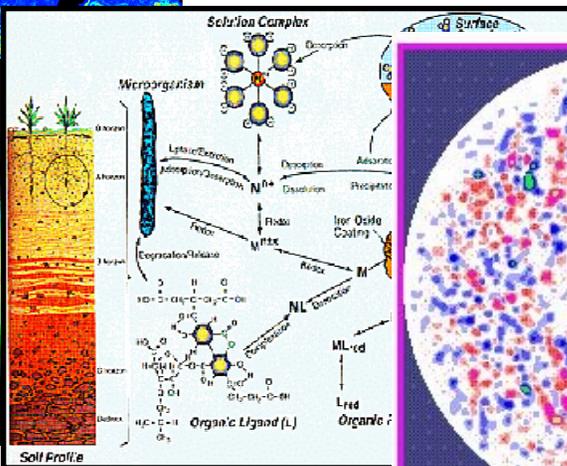


Components of Matter

Many SC programs need dramatic advances in simulation capabilities to meet their mission goals



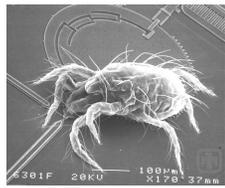
Health Effects, Bioremediation



Fusion Energy

The Scale of Things – Nanometers and More

Things Natural

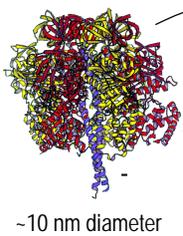
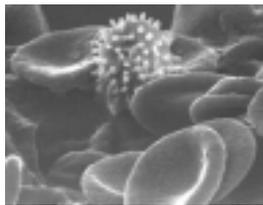


Dust mite
200 μm

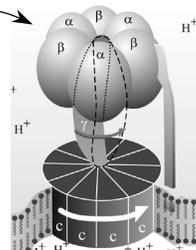


Human hair
~ 60-120 μm wide

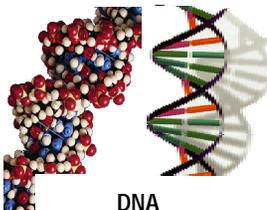
Red blood cells with white cell
~ 2-5 μm



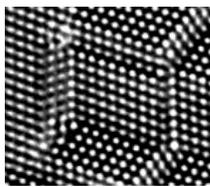
~10 nm diameter



ATP synthase



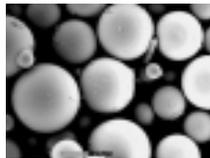
DNA
~2-1/2 nm diameter



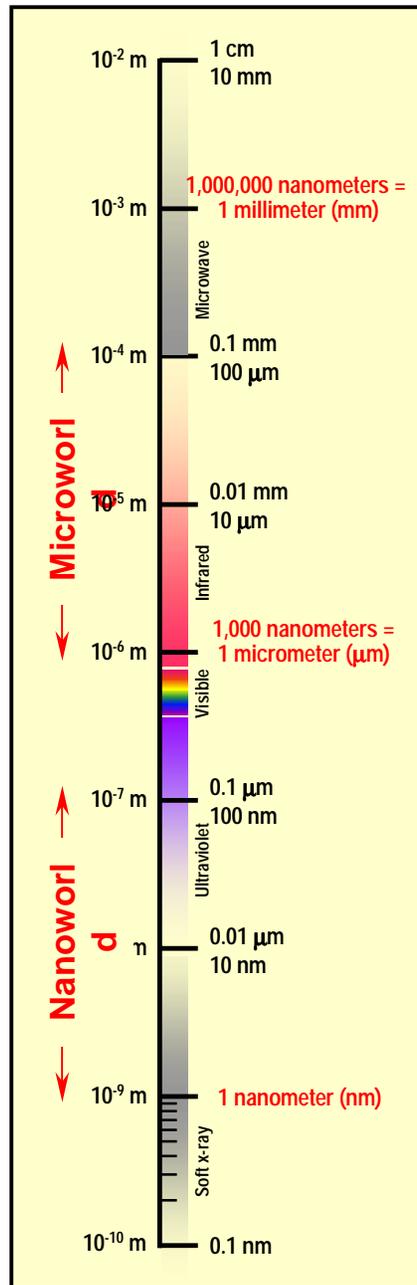
Atoms of silicon
spacing ~tenths of nm



Ant
~ 5 mm



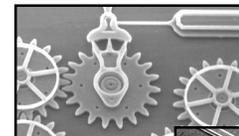
Fly ash
~ 10-20 μm



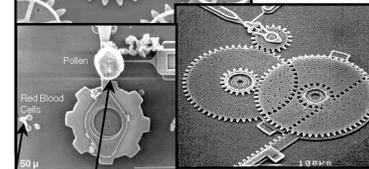
Things Manmade



Head of a pin
1-2 mm

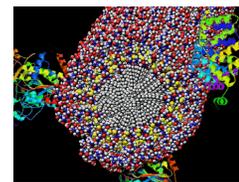


MicroElectroMechanical (MEMS) devices
10 -100 μm wide

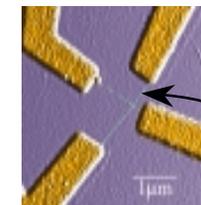


Pollen grain
Red blood cells

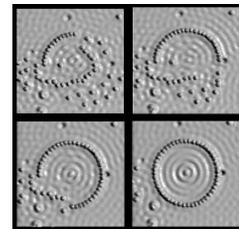
Zone plate x-ray "lens"
Outer ring spacing ~35 nm



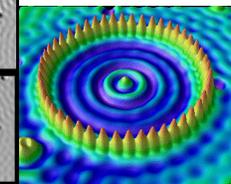
Self-assembled, Nature-inspired structure
Many 10s of nm



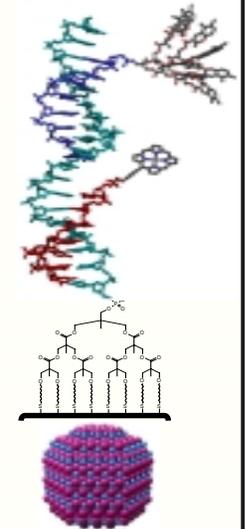
Nanotube electrode



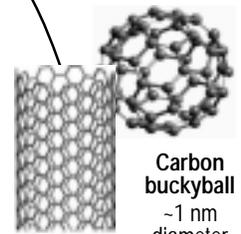
Quantum corral of 48 iron atoms on copper surface
positioned one at a time with an STM tip
Corral diameter 14 nm



The Challenge



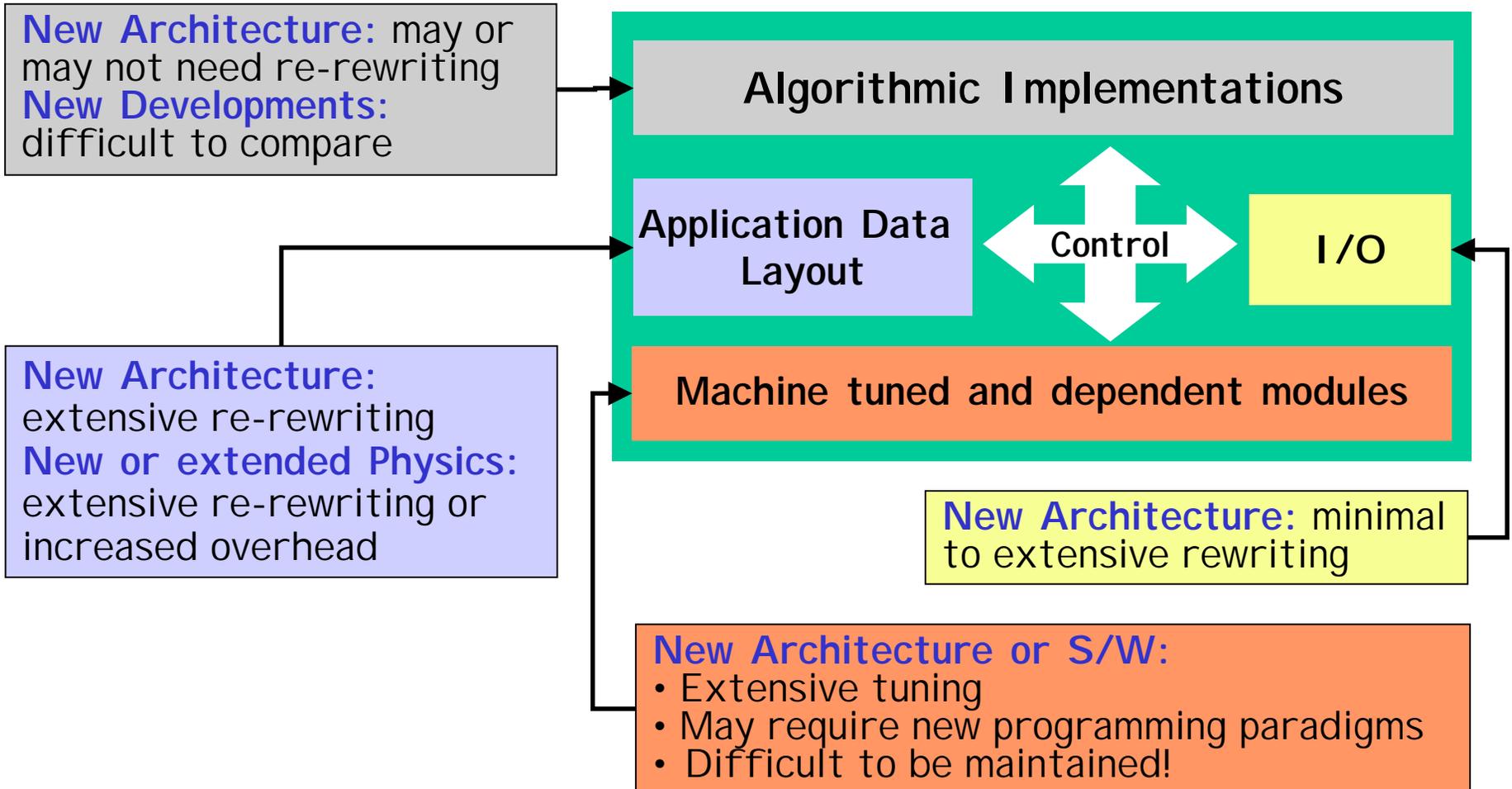
Fabricate and combine nanoscale building blocks to make useful devices, e.g., a photosynthetic reaction center with integral semiconductor storage.



Carbon buckyball
~1 nm diameter

Carbon nanotube
~1.3 nm diameter

Why do we need software libraries?



Key Lesson Learned

“We need to **move away from a coding style** suited for serial machines, **where every macro step of an algorithm needs to be thought** about and explicitly coded, to a **higher-level style, where the compiler and library tools take care of the details.** And the remarkable thing is, if we adopt this higher-level approach right now, **even on today's machines, we will see immediate benefits in our productivity.**”

W. H. Press and S. A. Teukolsky, 1997

Numerical Recipes: Does This Paradigm Have a future?

**There is a Need for a
Reliable Software
Infrastructure for Scientific
Computing**

The DOE ACTS Collection

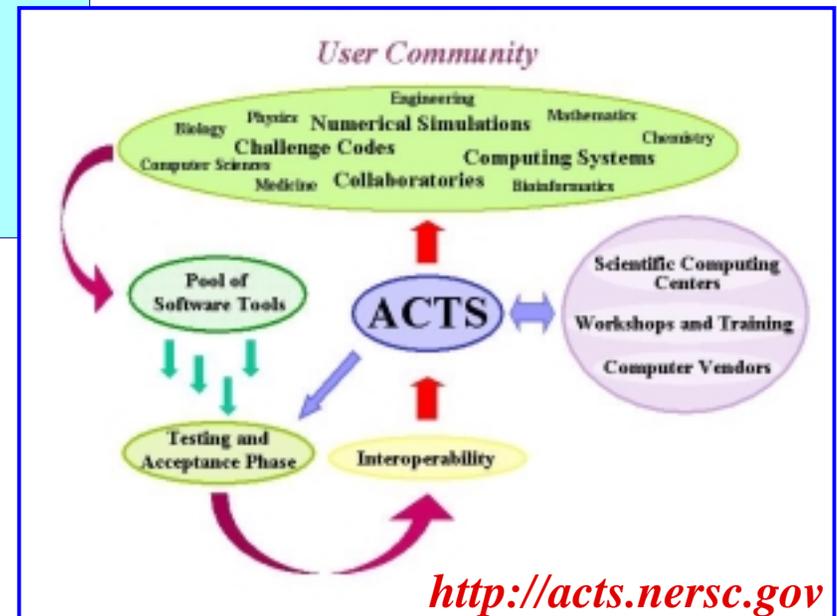
Goals

- ❑ *Collection of tools for developing parallel applications*
- ❑ *Extended support for experimental software*
- ❑ *Make ACTS tools available on DOE computers*
- ❑ *Provide technical support (acts-support@nersc.gov)*
- ❑ *Maintain ACTS information center (<http://acts.nersc.gov>)*
- ❑ *Coordinate efforts with other supercomputing centers*
- ❑ *Enable large scale scientific applications*
- ❑ *Educate and train*

- High Performance Tools
 - *portable*
 - *library calls*
 - *robust algorithms*
 - *help code optimization*
- More code development in less time
- More simulation in less computer time

Levels of Support

- **High**
 - Intermediate level
 - Tool expertise
 - Conduct tutorials
- **Intermediate**
 - Basic level
 - Higher level of support to users of the tool
- **Basic**
 - Help with installation
 - Basic knowledge of the tools
 - Compilation of user's reports

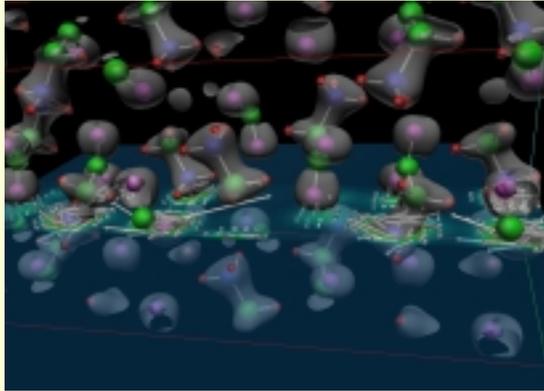


Current ACTS Tools and their Functionalities

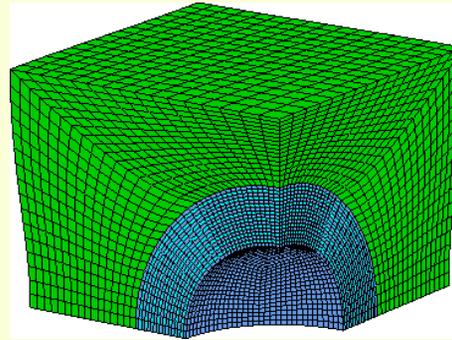
Category	Tool	Functionalities
Numerical $Ax = b$ $Az = \lambda z$ $A = U\Sigma V^T$ PDEs ODEs ⋮	Aztec	Algorithms for the iterative solution of large sparse linear systems.
	Hypre	Algorithms for the iterative solution of large sparse linear systems, intuitive grid-centric interfaces, and dynamic configuration of parameters.
	PETSc	Tools for the solution of PDEs that require solving large-scale, sparse linear and nonlinear systems of equations.
	OPT++	Object-oriented nonlinear optimization package.
	SUNDIALS	Solvers for the solution of systems of ordinary differential equations, nonlinear algebraic equations, and differential-algebraic equations.
	ScaLAPACK	Library of high performance dense linear algebra routines for distributed-memory message-passing.
	SuperLU	General-purpose library for the direct solution of large, sparse, nonsymmetric systems of linear equations.
	TAO	Large-scale optimization software, including nonlinear least squares, unconstrained minimization, bound constrained optimization, and general nonlinear optimization.
Code Development	Global Arrays	Library for writing parallel programs that use large arrays distributed across processing nodes and that offers a shared-memory view of distributed arrays.
	Overture	Object-Oriented tools for solving computational fluid dynamics and combustion problems in complex geometries.
Code Execution	CUMULVS	Framework that enables programmers to incorporate fault-tolerance, interactive visualization and computational steering into existing parallel programs.
	TAU	Set of tools for analyzing the performance of C, C++, Fortran and Java programs.
Library Development	ATLAS	Tools for the automatic generation of optimized numerical software for modern computer architectures and compilers.

Next tools: CCA, SLEPc (eigenvalues), Trilinos (iterative solvers)...

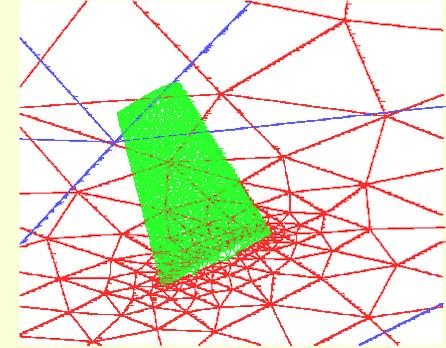
Use of ACTS Tools



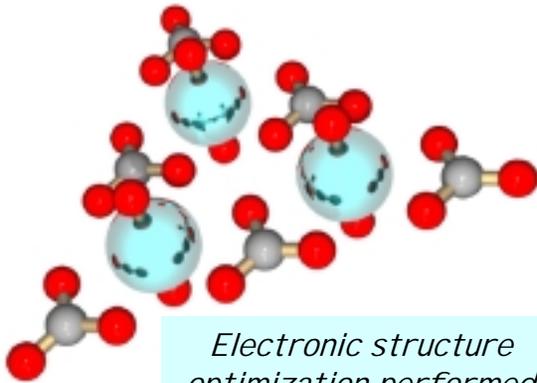
Induced current (white arrows) and charge density (colored plane and gray surface) in crystallized glycine due to an external field (Louie, Yoon, Pfrommer and Canning), eigenvalue problems solved with **ScaLAPACK**.



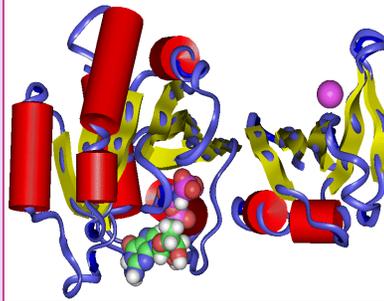
Model of a "hard" sphere included in a "soft" material, 26 million d.o.f. Unstructured meshes in solid mechanics using Prometheus and **PETSc** (Adams and Demmel).



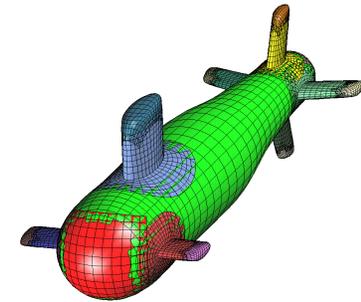
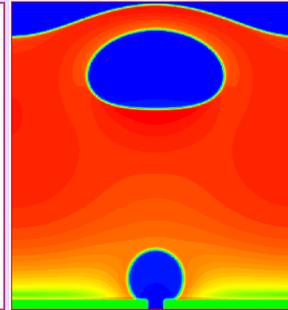
3D incompressible Euler, tetrahedral grid, up to 11 million unknowns, based on a legacy NASA code, FUN3d (W. K. Anderson), fully implicit steady-state, parallelized with **PETSc** (courtesy of Kaushik and Keyes).



Electronic structure optimization performed with **TAO**, $(UO_2)_3(CO_3)_6$ (courtesy of deJong).

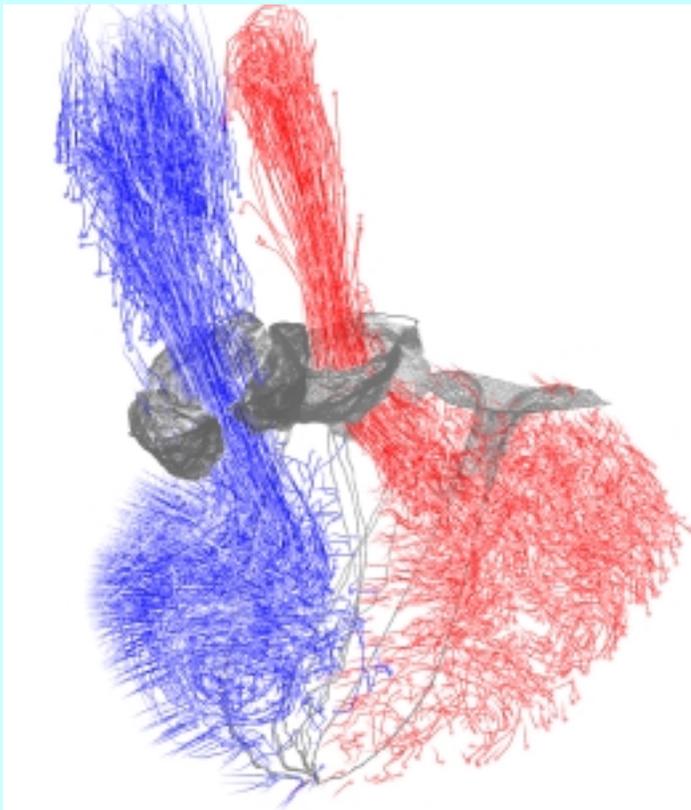


Molecular dynamics and thermal flow simulation using codes based on **Global Arrays**. GA have been employed in large simulation codes such as NWChem, GAMESS-UK, Columbus, Molpro, Molcas, etc.

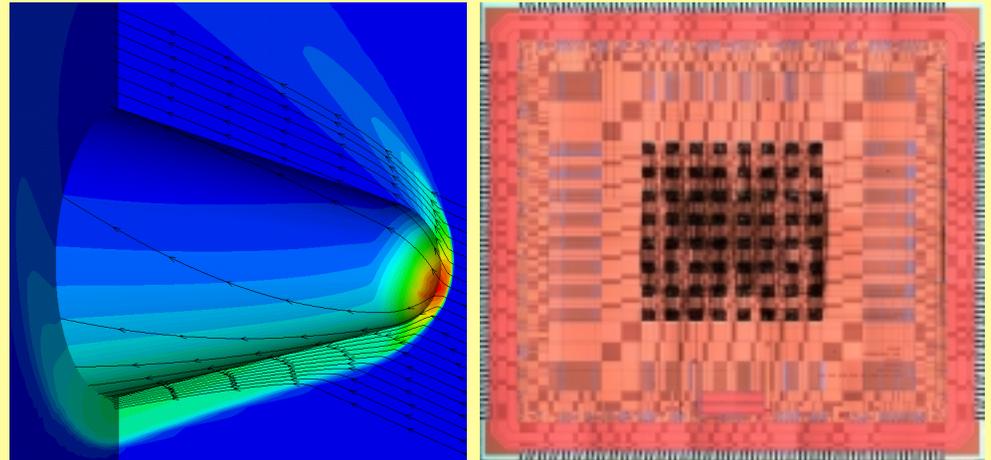


3D overlapping grid for a submarine produced with **Overture's** module ogen.

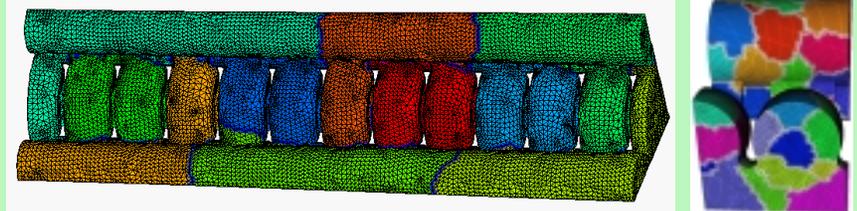
Use of ACTS Tools



Model of the blood-muscle-valve mechanics of the heart by an adaptive and parallel version of the immersed boundary method, using *PETSc*, *Hypr* and *SAMRAI*. Courtesy of Boyce Griffith, New York University.



Applications of *Trilinos*



Omega3P is a parallel distributed-memory code intended for the modeling and analysis of accelerator cavities, which requires the solution of generalized eigenvalue problems. A parallel exact shift-invert eigensolver based on *PARPACK* and *SuperLU* has allowed for the solution of a problem of order 7.5 million with 304 million nonzeros. Finding 10 eigenvalues requires about 2.5 hours on 24 processors of an IBM SP.

Use of ACTS Tools: Performance Analysis

```

#####
#*
#* This makefile shows how to use TAU to automatically instrument and
#* compile a simple Fortran program that calls the ScalAPACK routine
#* PDSGESV to solve a system of linear equations. It requires the
#* modules "tau" and "scalapack", as well as "gmake"
#*
#####

.SUFFIXES : .f90

# The following defines the SCALAPACK macros
# include $(SCALAPACKROOTDIR)/SLmake.inc

# The following defines the TAU macros
# For other options see $(TAUROOTDIR)/rs6000/lib
include $(TAUROOTDIR)/rs6000/lib/Makefile.tau-mpi-pdt-profile-trace

F90      = $(TAU_COMPILER) $(TAU_F90)
F90SUFFIX = -qsuffix=f90
LINKER   = $(TAU_LINKER)
LIBS     = $(TAU_MPI_FLIBS) $(TAU_LIBS) $(TAU_FORTRANLIBS)
LDFLAGS  = -brtl -binittfni:poe_remote_main
STLIBS   = $(SCALAPACK) $(PBLAS) $(BLACS) -lesslp2

TARGET = psgesvdriver.x
OBSJ = psgesvdriver.o

$(TARGET): $(OBSJ)
$(LINKER) $(LDFLAGS) $(OBSJ) -o

.f90.o:
$(F90) $(F90SUFFIX) -c $(F90)

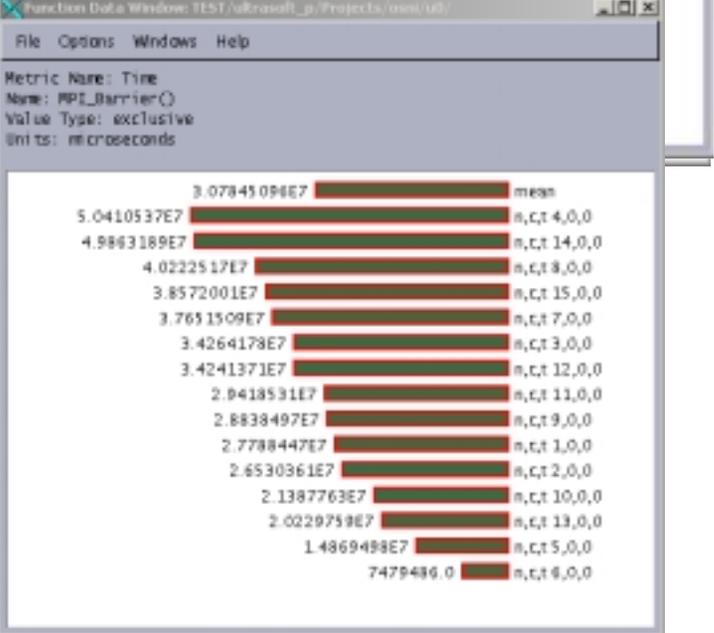
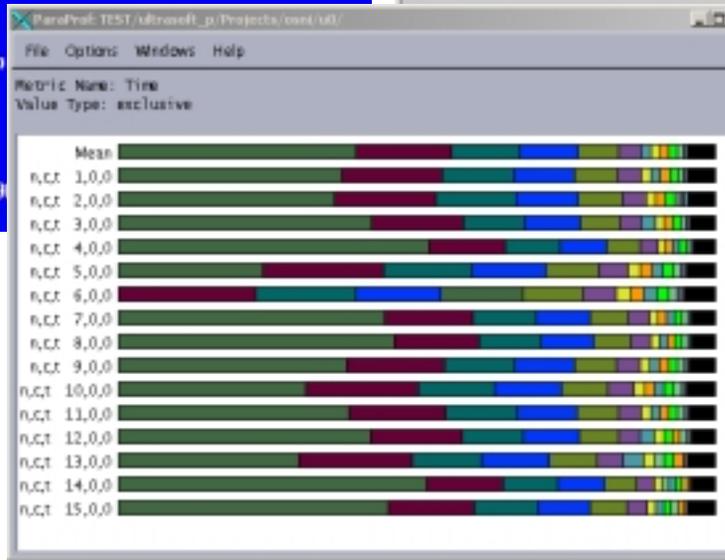
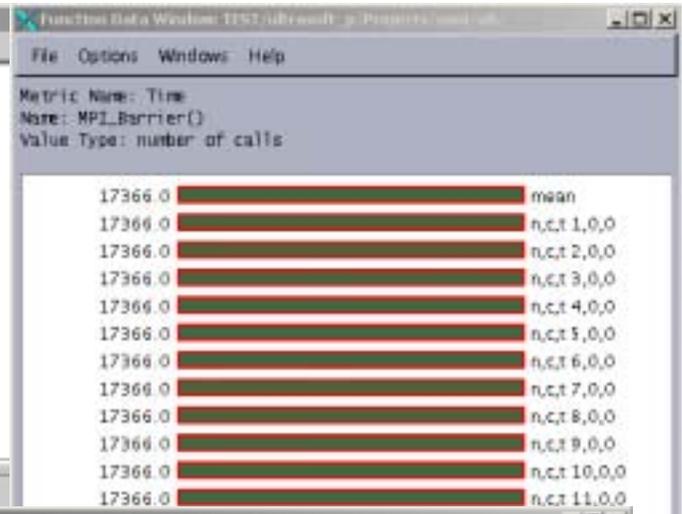
clean:
-@rm -f $(TARGET) *.o *.inst.f90
*.trc *.edf *.pv *.pprof *.txt
    
```

ParaProf Manager

- File Options Help
- Applications
 - Standard Applications
 - Default App
 - Default Exp
 - Time
- Runtime Applications
- DB Applications

Argument 1: Argument 1 (x.x.x.x)

Argument 2: Argument 2 (x.x.x.x)



Outreaching Efforts



Yearly Workshop at LBNL

- *Building Robust Scientific and Engineering High-End Computing Applications, August 23-26, 2005*
- Previous (five) workshops (and other events) attended by more than 300 participants from DOE labs, academia and industry
- Tutorials and hands-on (on NERSC computers) delivered by tool developers

- Accumulation of expertise and user feedback on the use of the software tools and scientific applications that benefited from them
- Gallery of applications: <http://acts.nersc.gov/MatApps>
- Collaboration with computer vendors, early access to new architectures allowing for porting and benchmarking

Application	Computational Problem	Software Tools	Highlights
MADCAP	Matrix factorization and triangular solves	ScaLAPACK	<ul style="list-style-type: none"> • 50% peak performance on an IBM SP • Nearly perfect scalability on 1024, 2048, 3072 and 4096 processors • Fast implementation of numerical algorithms
3-Charged Particles	Solution of large, complex unsymmetric linear systems	SuperLU	<ul style="list-style-type: none"> • Solves systems of equations of order 8.4 million on 64 processors in 1 hour of wall clock time • 30 GFLOPs
NWChem	Distribute large data arrays, collective operations	Global Arrays and LAPACK	<ul style="list-style-type: none"> • Very good scaling for large problems



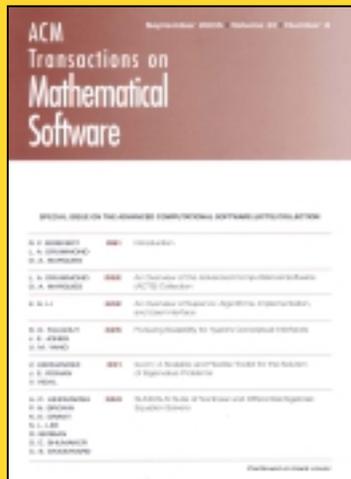
Enabling sciences and discoveries with high performance and scalability

Recent Short Courses and Minisimposia

- *Short Course on the ACTS Collection*, SIAM CSE05 Conference, Orlando, FL.
- *Numerical Software for Solving Problems in Computational Science and Engineering*, MS48, SIAM CSE05 Conference, Orlando, FL.
- *The ACTS Collection: Robust and High Performance Libraries for Computational Sciences*, SIAM PP04 Conference, San Francisco, CA.

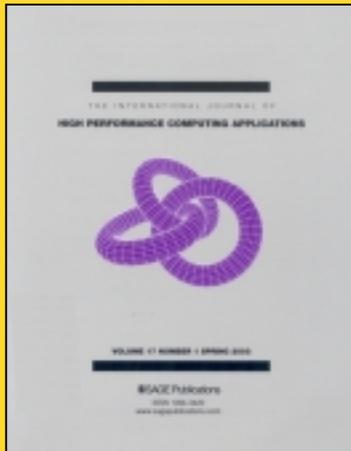
Journals Featuring ACTS Tools

September 2005 Issue



- **An Overview of the Advanced Computational Software (ACTS) Collection**, by T. Drummond and O. Marques
- **SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation Solvers**, by A. Hindmarsh, P. Brown, K. Grant, S. Lee, R. Serban, D. Shumaker and C. Woodward.
- **An Overview of SuperLU: Algorithms, Implementation, and User Interface**, by X. Li.
- **SLEPc: A Scalable and Flexible Toolkit for the Solution of Eigenvalue Problems**, by V. Hernandez, J. Roman and V. Vidal.
- **An Overview of the Trilinos Project**, by M. Heroux, R. Bartlett, V. Howle, R. Hoekstra, J. Hu, T. Kolda, R. Lehoucq, K. Long, R. Pawlowski, E. Phipps, A. Salinger, H. Thornquist, R. Tuminaro, J. Willenbring, A. Williams and K. Stanley.
- **Pursuing Scalability for hypre's Conceptual Interfaces**, by R. Falgout, J. Jones and U. Yang.

Spring 2006 Issue



- **A Component Architecture for High-Performance Scientific Computing**, by D. Bernholdt, B. Allan, R. Armstrong, F. Bertrand, K. Chiu, T. Dahlgreen, K. Damevski, W. Elwasif, T. Epperly, M. Govindaraju, D. Saltz, J. Kohl, M. Krishnan, G. Kumfert, J. Larson, S. Lefantzi, M. Lewis, A. Malony, L. McInnes, J. Nieplocha, B. Norris, S. Parker, J. Ray, S. Shende, T. Windus and S. Zhou.
- **CUMULVS: Interacting with High-Performance Scientific Simulations, for Visualization, Steering and Fault Tolerance**, by J. Kohl, T. Wilde and D. Bernholdt.
- **Advances, Applications and Performance of the Global Arrays Shared Memory Programming Toolkit**, by J. Nieplocha, B. Palmer, V. Tipparaju, M. Krishnan, H. Trease and E. Aprà.
- **The TAU Parallel Performance System**, by S. Shende and A. Malony.
- **High Performance Remote Memory Access Communication: The ARMCI Approach**, by J. Nieplocha, V. Tipparaju, M. Krishnan and D. Panda.

Future Directions: Building up on Lessons Learned

- Requirements for reusable high quality software tools
- Integration, maintenance and support
- Interfaces using script languages
- Software automation

⋮



<http://acts.nerisc.gov>