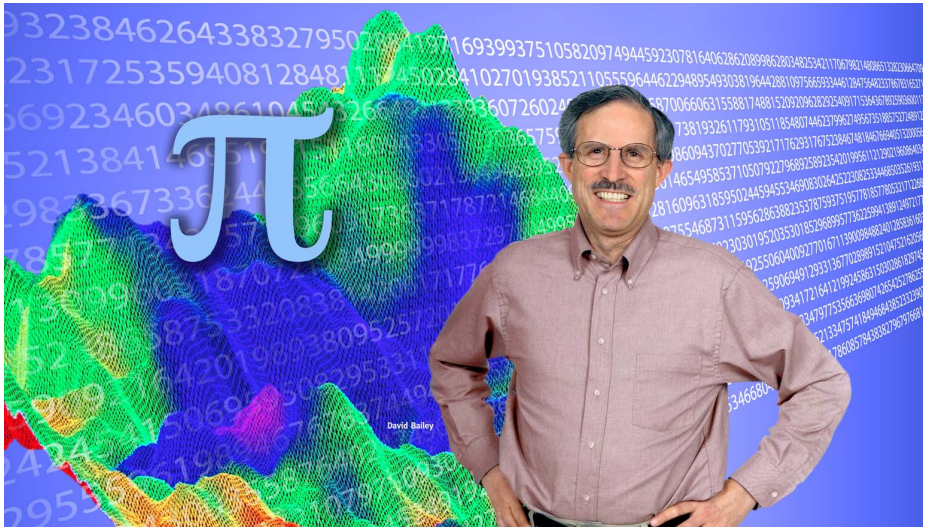
The SciDAC-3 Institute for Sustained Performance, Energy, and Resilience (SUPER)

David H Bailey, Lawrence Berkeley National Lab, USA

http://crd.lbl.gov/~dhbailey



Why is performance important

- Peta- and exa-scale computing has opened up new vistas for scientific computing.
- BUT: Achieved performance is often poor – typically only 1-5% of peak.
- Systems are more complicated:
  - 100,000 or more multi-core CPUs.
  - GPU accelerators.
- Codes are more complicated:
  - Multi-disciplinary.
  - Multi-scale.
- Low performance is unacceptable, not only because of the high purchase cost of state-of-the-art systems, but also because of the increasing cost of maintenance and electrical power for these systems.

Predecessor organizations

Performance Evaluation Research Center (PERC):

- 8-institution SciDAC-1 consortium, in operation 2001-2006.
- Led by DHB of LBNL.
- Focus areas:
  - Benchmarking.
  - Modeling.
  - Understanding

Performance Engineering Research Institute (PERI):

- 10-institution SciDAC-2 collaboration, in operation 2006-2011.
- Led by Robert F. Lucas of USC / ISI and David H. Bailey of LBNL.
- Focus areas:
  - Performance modeling.
  - Application engagement.
  - Semiautomatic tuning research.

The Sustained Performance, Energy and Resilience (SUPER) Institute

- Started September 2011, with funding from DOE SciDAC-3 program.
- Director: Robert F. Lucas of USC/ISI and David H. Bailey of LBNL.
- Focus areas:
  - Automatic performance tuning.
  - Energy efficient-computing.
  - Resilient computing.
  - Combined optimization of performance, energy efficiency and resilience.
  - Engagement with other institutes and scientific application projects.
  - Tool integration.
  - Outreach and tutorials.

The SUPER team

ANL
Paul Hovland
Boyana Norris
Stefan Wild

LBNL
David Bailey
Lenny Oliker
Sam Williams

LLNL
Bronis de Supinski
Daniel Quinlan

Oregon
Allen Malony
Sameer Shende

ORNL
Gabriel Marin
Philip Roth
Patrick Worley

UCSD
Laura Carrington

UMD
Jeffrey Hollingsworth

UNC
Rob Fowler
Allan Porterfield

USC
Jacque Chame
Robert Lucas (PI)

UTK
Shirley Moore
Dan Terpstra

Utah
Mary Hall
Chun Chen

Energy minimization

- New activity, led by Laura Carrington of UC San Diego.
- Objective: Help users and centers to conserve energy while at the same time achieve good performance.
- Approach:
  - Develop new energy-aware application-programmer interfaces (APIs) for users.
  - Obtain more precise data regarding energy consumption.
  - Extend PAPI to sample hardware energy monitors.
  - Build some prototype PowerMon devices.
  - Extend performance models to encompass energy consumption.
  - Transform codes to minimize energy consumption.
  - Inform systems to exploit dynamic voltage frequency scaling (DVFS).

Resilence

- New activity, led by Bronis de Supinski of LLNL.
- Objective: Automate vulnerability assessment for scientific computing.
- Approach:
  - Build on success of PERI autotuning.
  - Conduct fault injection experiments.
  - Determine which code regions or data structures fail catastrophically.
  - Determine what transformations enable them to survive.
  - Extend ROSE compiler to implement the transformations.
  - Investigate directive-based API for users.
  - Augments empirically derived vulnerability assessment.

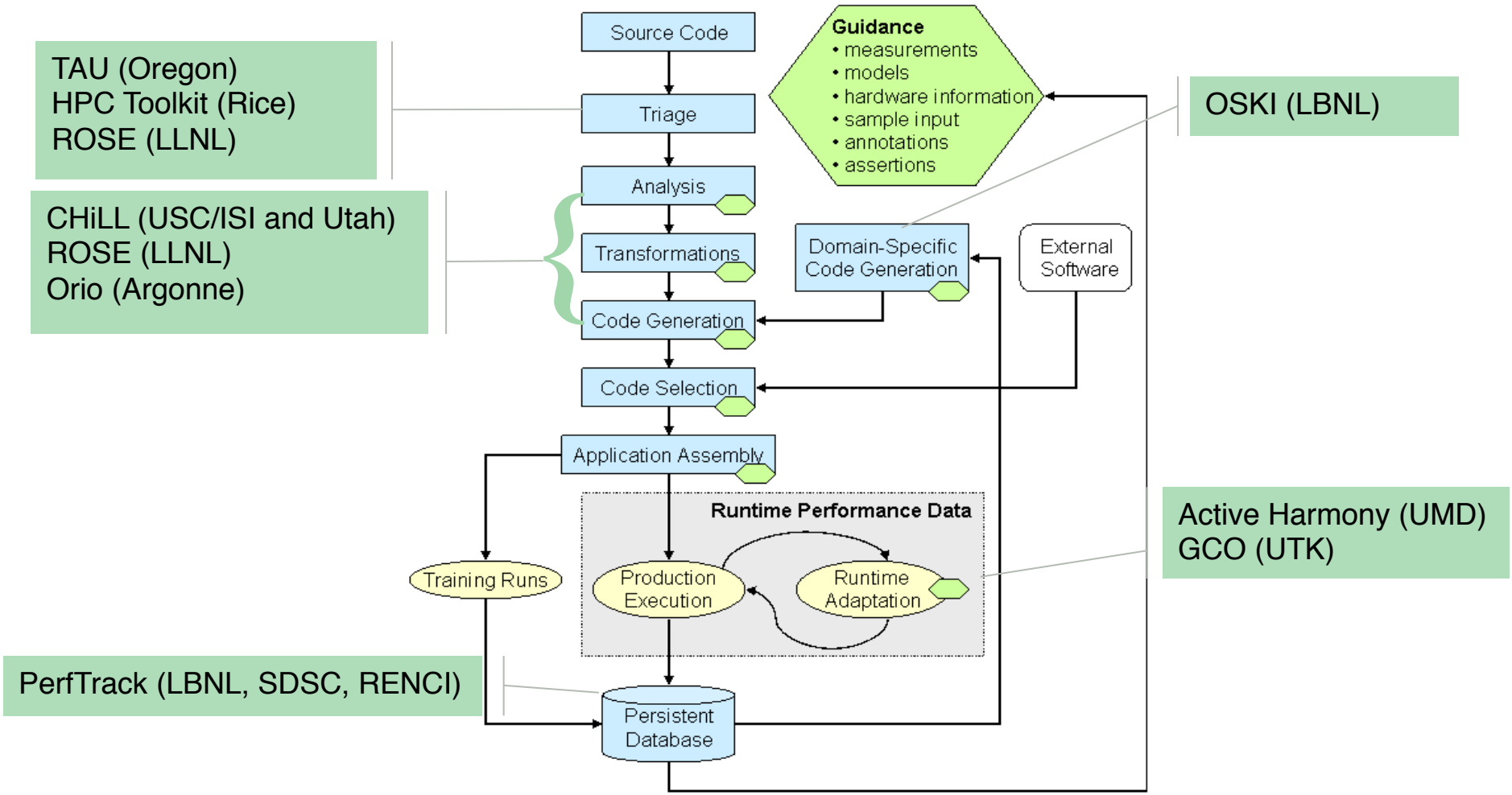
Performance is our middle name

- Performance measurement:
  - Adopt University of Oregon's Tau system.
  - Extend PerfDMF to enable online collection and analysis.
  - Collaborate with Rice and HPCToolkit.
  - Develop and utilize a performance database.
- Performance modeling:
  - Refine ANL's PBound and LBNL's Roofline performance models.
  - Extend MIAMI to model impact of architectural variation.
  - Extend UCSD's PSINS to model communication.

Automatic performance tuning

- Started in PERI project; continuing in SUPER, led by Mary Hall.
- Challenge: We have found that application scientists are reluctant to learn and use performance tools in day-to-day research work.
  - We called a party, but nobody came.
- Solution: Extend semi-automatic performance tuning methods, such as those developed for FFTW (FFTs) and ATLAS (dense matrices), to general large-scale scientific codes.
- Approach:
  - Extend basic autotuning technology developed in PERI to a broader set of applications and architectures.
  - Use Tau-based front-end for “triage.”
  - Extend OpenMP-CHiLL framework to multicore systems.
  - Use Active Harmony as the search engine.
  - Targeted autotuning: users write simple code, leaving tuning to us.
  - Whole program autotuning: parameters, algorithm choice, libraries, etc.

The SUPER autotuning framework

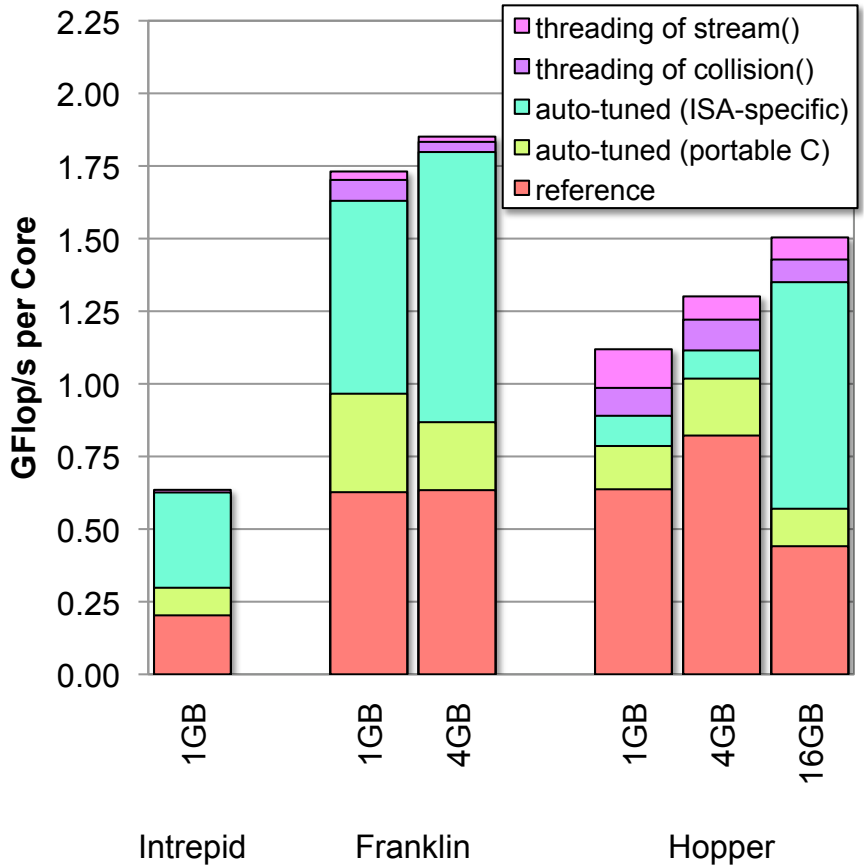


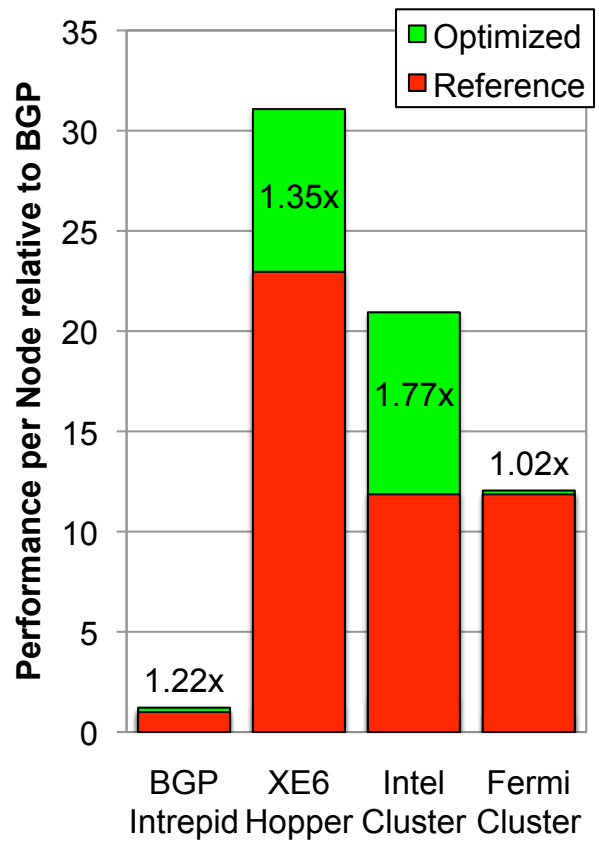
Applications of SUPER-PERI research

- LBHMD (lattice Boltzmann) and GTC (plasma toroidal):
  - LBMHD: Up to 3X speedup via autotuning.
  - GTC: Up to 1.77X speedup via autotuning.
- S3D (combustion):
  - 12.7% overall performance improvement.
  - 762,000 CPU-hours are potentially saved each year.
- PFLOTRAN (subsurface reactive flows):
  - 2X speedup on two key PETSc routines via autotuning.
  - 40X speedup in initialization; 4X improvement in I/O stage; overall 5X.
- SMG2000 (groundwater diffusion):
  - 2.37X speedup on one key kernel; overall 27% improvement.
- Nek5000 (turbulence):
  - Up to 1.93X speedup.
- LS3DF (electronic structure):
  - Increased scalability from 1000-2000 to over 160,000 cores.
  - Achieved 442 Tflop/s on Jaguar.

Autotuning LBMHD and GTC

- LBMHD (left): Implements a lattice Boltzmann method for magnetohydrodynamic plasma turbulence simulation.
- GTC (right): A gyrokinetic toroidal code for plasma turbulence modeling.

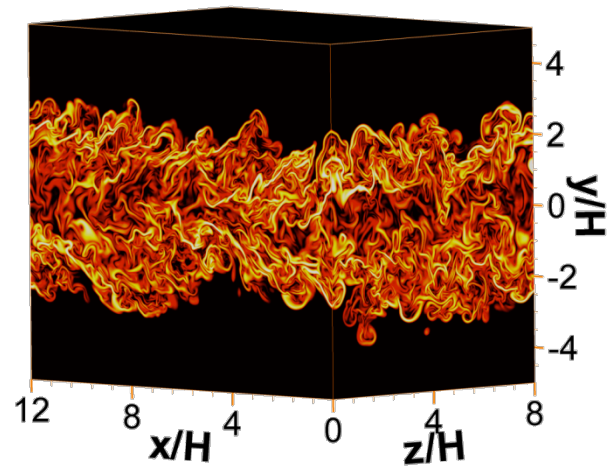


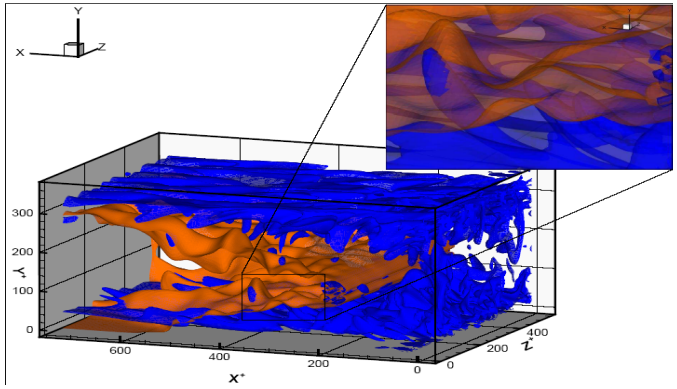


Credit: Samuel Williams, LBNL

S3D

- Performs direct numerical simulation of turbulent combusion.
- Developed at CRF/Sandia.
- Multiphysics (sprays, radiation, soot).
- Consumes 6,000,000 CPU-hours at NCCS.
- Tier 1 pioneering application for Jaguar.
- Compressible Navier-Stokes equations.
- High fidelity numerical methods:
  - 8th order finite difference.
  - 4th order explicit Runge-Kutta integrator.
- Hierarchy of molecular transport models.





Figures courtesy of S3D PI, Jacqueline H. Chen, SNL

Performance tuning of S3D

- Used performance models to identify improved compiler settings.
- Identified central importance of exp intrinsic:
  - Developed custom exp function than ran faster than library exp.
  - Subsequently replaced with improved Cray library exp.
- Used tools from Rice University to improve performance:
  - Used HPCToolkit to identify and to assess bottlenecks.
  - LoopTool helped automate tedious code transformations.

Bottom line:

- Achieved 12.7% overall improvement.
  - Node performance increased from 15% of peak to 17.4% of peak.
  - Estimated potential annual savings: 762,000 CPU-hours.

SMG2000

- SMG2000: Semicoarsening multigrid code, used for various applications, including modeling of groundwater diffusion.
- PERI researchers integrated several tools, then developed a “smart” search technique to find an optimal tuning strategy among 581 million different choices.
- Achieved 2.37X performance improvement on one key kernel.
- Achieved 27% overall performance improvement.

Autotuning the central SMG2000 kernel

Outlined code (from ROSE outliner)

```
for (si = 0; si < stencil_size; si++)
    for (kk = 0; kk < hypre__mz; kk++)
        for (jj = 0; jj < hypre__my; jj++)
            for (ii = 0; ii < hypre__mx; ii++)
                rp[((ri+ii)+(jj*hypre__sy3))+(kk*hypre__sz3)] -=
                    ((Ap_0[((ii+(jj*hypre__sy1))+ (kk*hypre__sz1))+
                     (((A->data_indices)[i])[si])])*
                     (xp_0[((ii+(jj*hypre__sy2))+(kk*hypre__sz2))+(( *dxp_s)[si])]));
```

CHiLL transformation recipe

permute([2,3,1,4])
tile(0,4,TI)
tile(0,3,TJ)
tile(0,3,TK)
unroll(0,6,US)
unroll(0,7,UI)

Constraints on search

$0 \leq TI, TJ, TK \leq 122$
$0 \leq UI \leq 16$
$0 \leq US \leq 10$
compilers $\in$ {gcc, icc}

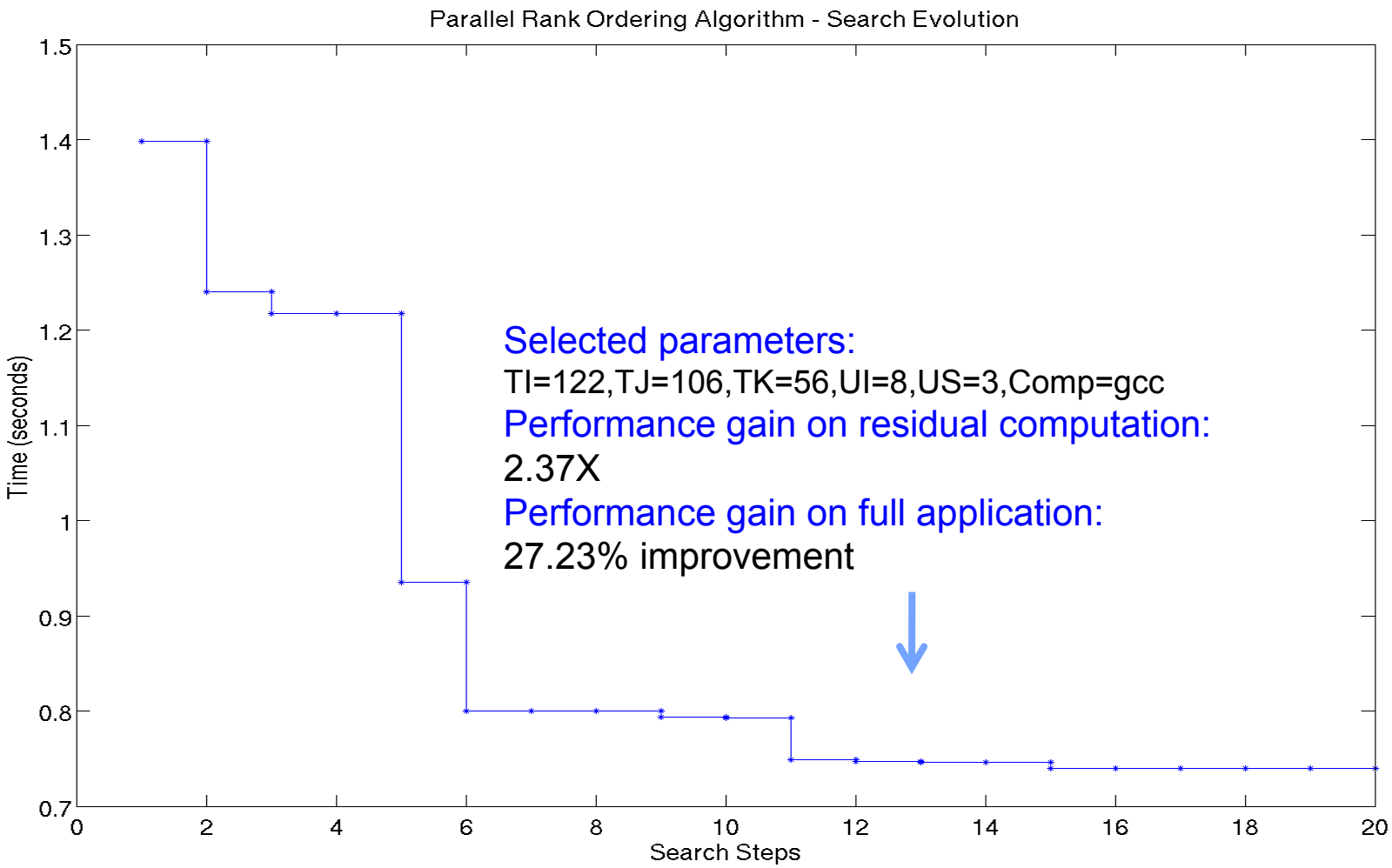
Search space:

$122^3 \times 16 \times 10 \times 2 = 581{,}071{,}360$ points

Credit: Mary Hall, Utah

Search for optimal tuning parameters in SMG200 kernel

Parallel heuristic search (using Active Harmony) evaluates 490 total points and converges in 20 steps.

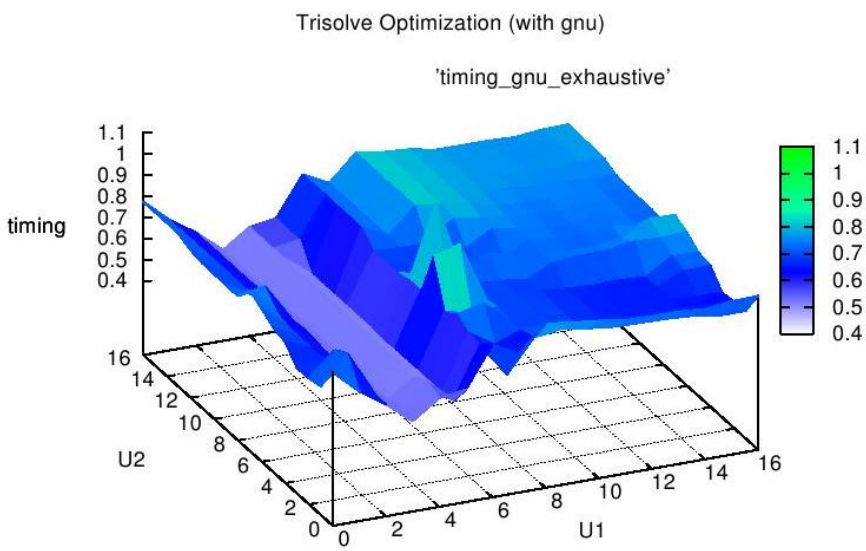


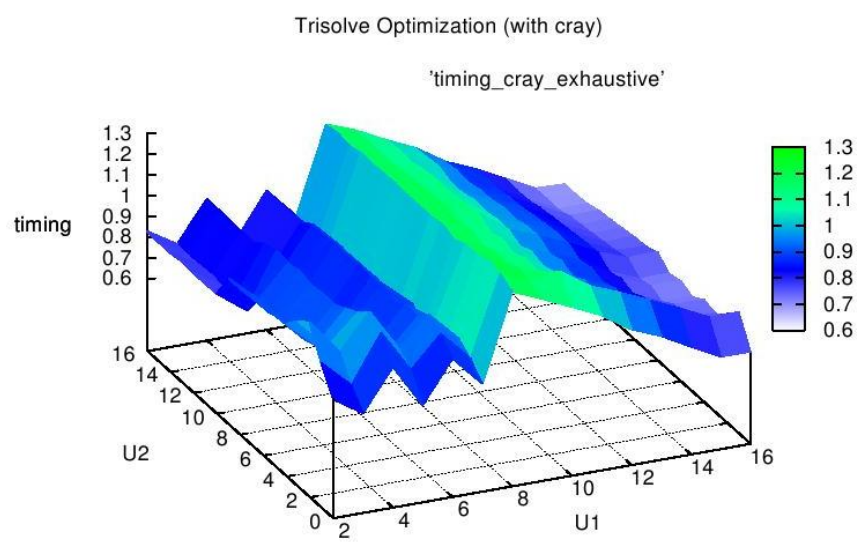
Credit: Mary Hall, University of Utah

17

Autotuning the triangular solve kernel of the Nek5000 turbulence code

Compiler	Original	Active Harmony			Exhaustive		
	Time	Time	(u1,u2)	Speedup	Time	(u1,u2)	Speedup
pathscale	0.58	0.32	(3,11)	1.81	0.30	(3,15)	1.93
gnu	0.71	0.47	(5,13)	1.51	0.46	(5,7)	1.54
pgi	0.90	0.53	(5,3)	1.70	0.53	(5,3)	1.70
cray	1.13	0.70	(15,5)	1.61	0.69	(15,15)	1.63





Credit: Jeff Hollingsworth, University of Maryland

LS3DF

- LS3DF: "linearly scaling 3-dimensional fragment" code for electronic structure calculation.
- Developed at LBNL by Lin-Wang Wang and several collaborators.
- Numerous applications in materials science and nanoscience.
- Employs a novel divide-and-conquer scheme including a new approach for patching the fragments together.
- Achieves nearly linear scaling in *computational cost versus size of problem*, compared with n^3 scaling in many other comparable codes.
- Potential for nearly linear scaling in *performance versus number of cores*.

Challenge:

- Initial implementation of LS3DF had disappointingly low performance and parallel scalability.

Performance analysis of LS3DF

LBNL researchers (funded through PERI) applied performance monitoring tools to analyze run-time performance of LS3DF. Key issues uncovered:

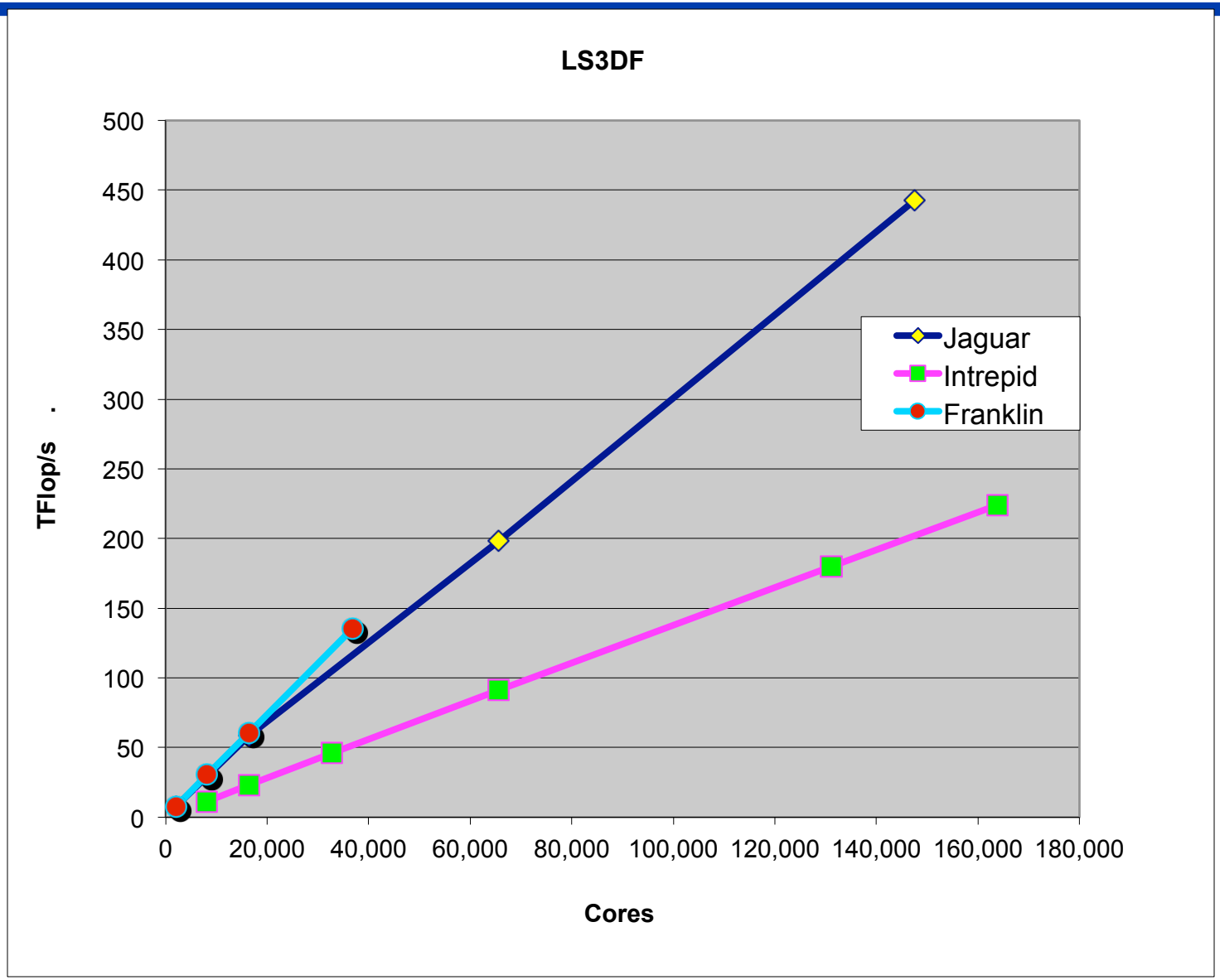
- Limited concurrency in a key step, resulting in a significant load imbalance between processors.
  - Solution: Modify code for two-dimensional parallelism.
- Costly file I/O operations were used for data communication between processors.
  - Solution: Replace all file I/O operations with MPI send-receive operations.

Resulting performance of tuned LS3DF

- 135 Tflops/s on 36,864 cores of the Cray XT4 Franklin system at LBNL.
  - 40% efficiency on 36,864 cores.
- 224 Tflops/s on 163,840 processors of the BlueGene/P Intrepid system at Argonne Natl. Lab.
  - 40% efficiency on 163,840 cores.
- 442 Tflops/s on 147,456 processors of the Cray XT5 Jaguar system at Oak Ridge Natl. Lab.
  - 33% efficiency on 147,456 cores.

2008 ACM Gordon Bell Prize in a special category for "algorithm innovation."

Near-linear parallel scaling up to 163,840 cores and up to 442 Tflop/s



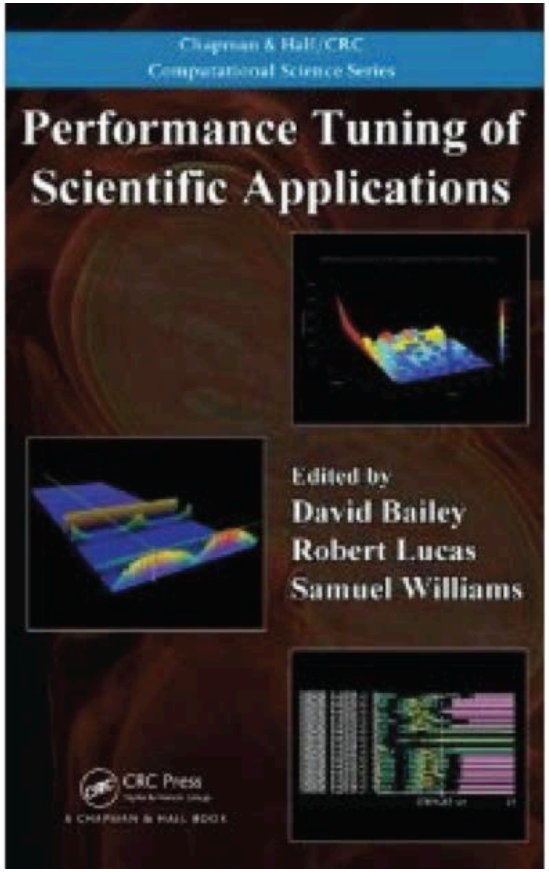
Performance Tuning of Scientific Applications

Editors: Bailey (LBNL), Lucas (USC/ISI), Williams (LBNL); numerous individual authors of various chapters.

Publisher: Taylor and Francis / CRC, Jan 2011.

Contents:

1. Introduction
2. Parallel computer architecture
3. Software interfaces to hardware counters
4. Measurement and analysis of parallel program performance using TAU and HPCToolkit.
5. Trace-base tools
6. Large-scale numerical simulations on high-end computational platforms
7. Performance modeling: the convolution approach
8. Analytic modeling for memory access patterns based on Apex-MAP
9. The roofline model
10. End-to-end auto-tuning with active harmony
11. Languages and compilers for auto-tuning
12. Empirical performance tuning of dense linear algebra software
13. Auto-tuning memory-intensive kernels for multicore
14. Flexible tools supporting a scalable first-principles MD code
15. The community climate system model
16. Tuning an electronic structure code



Disclosure: I receive a modest royalty (shared with other editors and LBNL) from sales.