Performance Engineering Research Institute (PERI)

Presented by

Philip C. Roth

Future Technologies Group
Computer Science and Mathematics Division

Performance engineering: enabling petascale science

Petascale computing is about delivering performance to scientists

Maximizing performance is getting harder

- Systems are more complicated
  - O (100 K) processors
  - Multicore with SIMD extensions
- Scientific software is more complicated
  - Multidisciplinary and multiscale

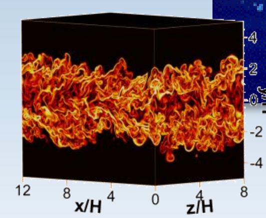
IBM BG/P at ANL

Photo courtesy of [Argonne National Laboratory]

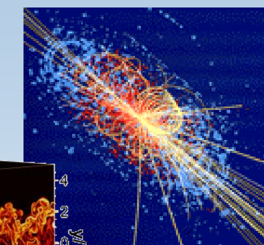
Cray XT5 at ORNL

PERI addresses this challenge in three ways

- Model and predict application performance
- Assist SciDAC scientific code projects with performance analysis and tuning
- Investigate novel strategies for automatic performance tuning



S3D turbulent combustion modeling



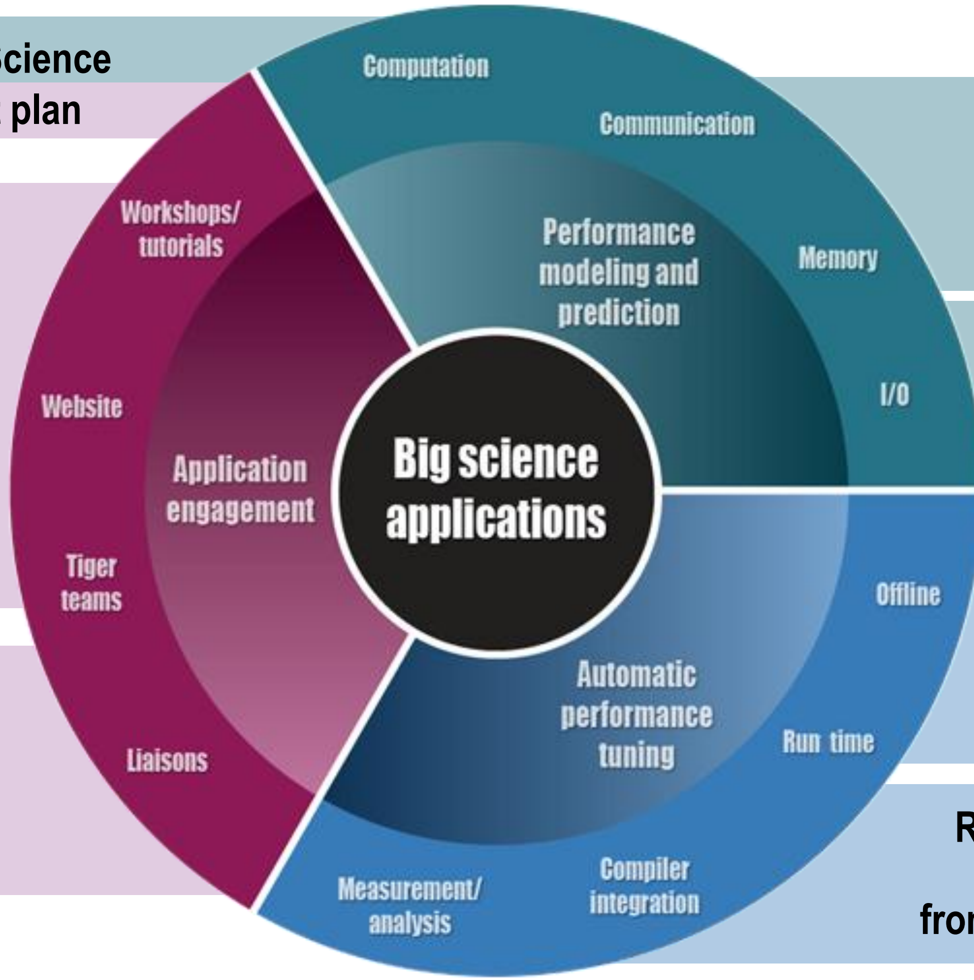
BeamBeam3D accelerator modeling

SciDAC-2 Performance Engineering Research Institute (PERI)

Informing Office of Science 10 year procurement plan

Providing near-term impact on the performance optimization of SciDAC applications

Informing long-term automated tuning efforts



Evaluating architectures and algorithms in preparation for move to petascale

Providing guidance in automatic tuning

Long-term research goal to improve performance portability

Relieving the performance optimization burden from scientific programmers

Engaging SciDAC software developers

Application survey

- Collect data on SciDAC-2 and INCITE code characteristics and performance requirements
- Use data to determine PERI engagement activities and to direct PERI research

Application liaisons

- Long-term partnerships between PERI researchers and scientific code teams
- Currently working actively with several application teams

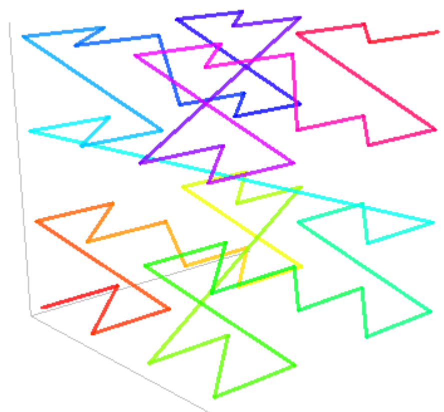
Tiger teams

- Focus on DOE's highest priorities: SciDAC-2, INCITE, JOULE
- Currently building models to estimate performance at scale and on new architectures

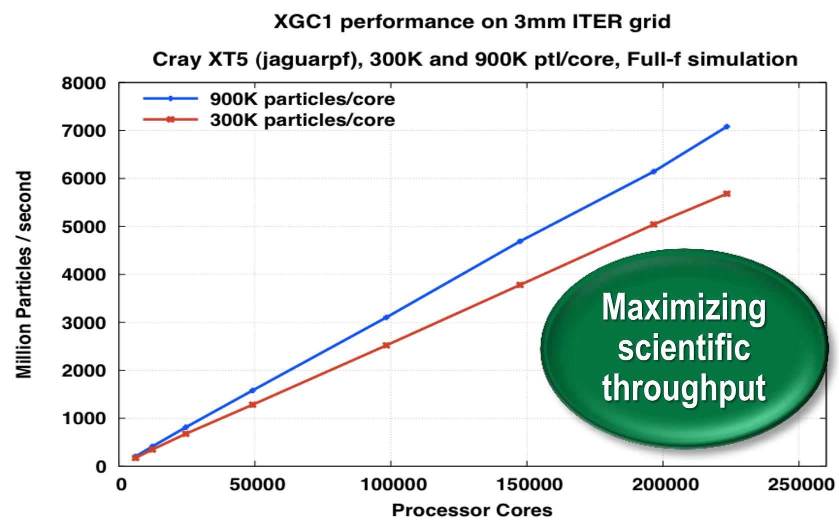
http://icl.cs.utk.edu/peri/

Optimizing kernels

Optimizing PFLOTRAN Jacobian initialization: using Morton space-filling curve to order initialization reduces L3 cache misses by 26%, TLB misses by 34% [source: Marin, ORNL]



Weak Scaling Graph for XGC1
Cray XT5 (jaguarpf), 900K ptl/thread, Full-f simulation
12 cores per node, 2 MPI processes per node



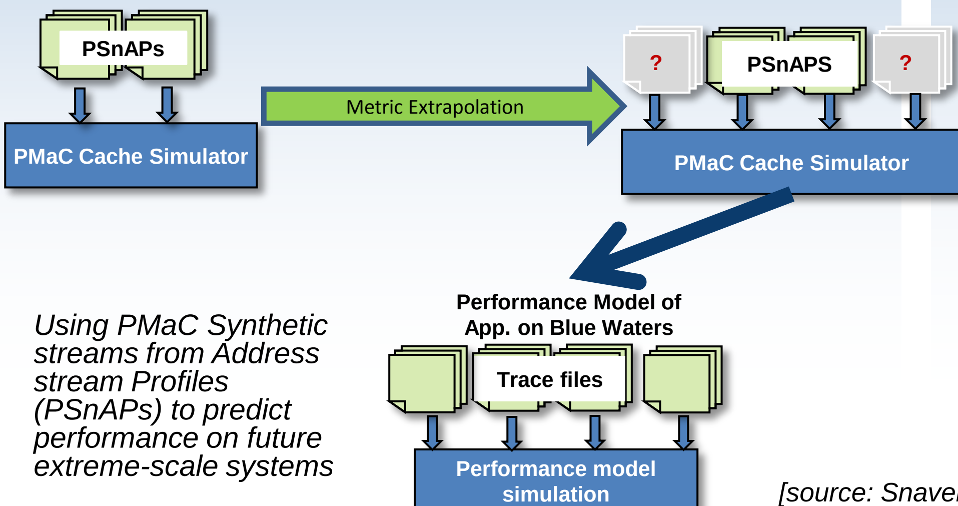
Maximizing scientific throughput

[source: Worley, ORNL]

Performance modeling

Modeling is critical for automation of performance tuning

- Guidance to the developer
  - New algorithms, systems, etc.
- Need to know where to focus effort
- Need to know when we are done tuning
- Predictions for new or hypothetical systems

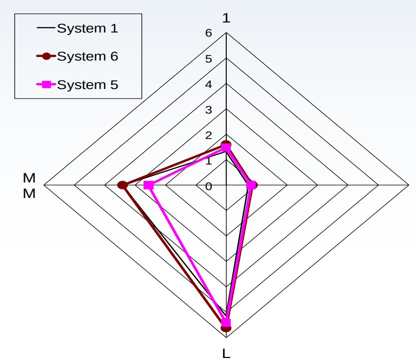


Using PMaC Synthetic streams from Address stream Profiles (PSnAPs) to predict performance on future extreme-scale systems

[source: Snavely, SDSC]

Recent progress

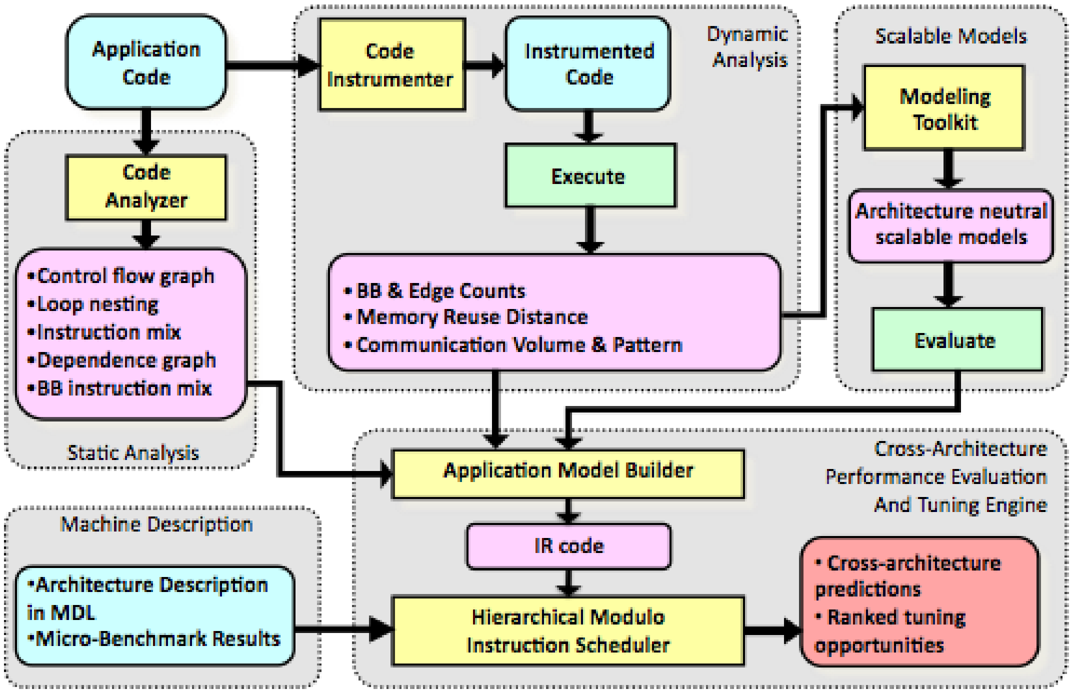
- Trace extrapolation techniques to enable performance prediction on larger systems
- HPCToolkit, PAPI, and PerfTrack extended to better support performance modeling
- Modeling Assertions extended to support performance predictions of workloads containing I/O activity
- Improved characterization of memory performance in multicore processors



Projections for FLASH performance on several future systems

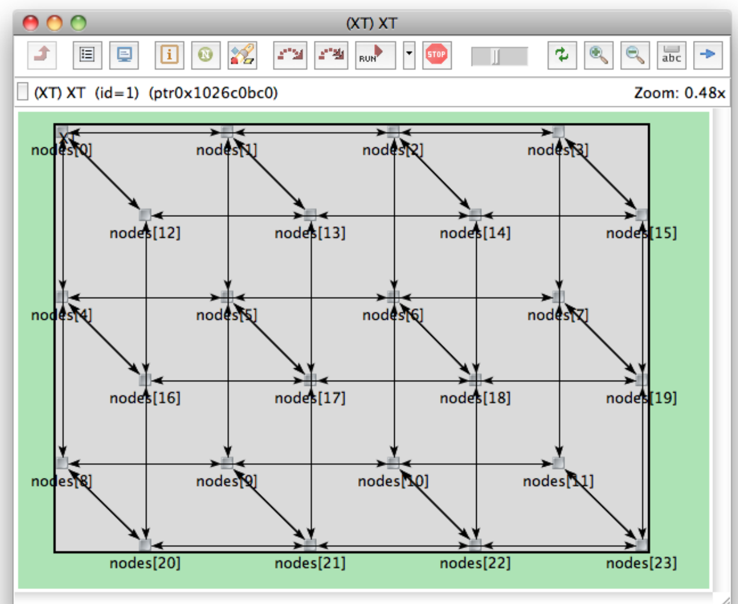
Modeling efforts contribute to procurements and other activities beyond PERI automatic tuning

PERI performance modeling at ORNL



Design of Machine Independent Application Performance Modeling Infrastructure (MIAMI) toolkit for producing scalable, cross-architecture performance models [source: Marin, ORNL]

Graphical user interface for discrete-event simulation of small Cray XT-like system with 3D torus interconnection network [source: Roth, ORNL]



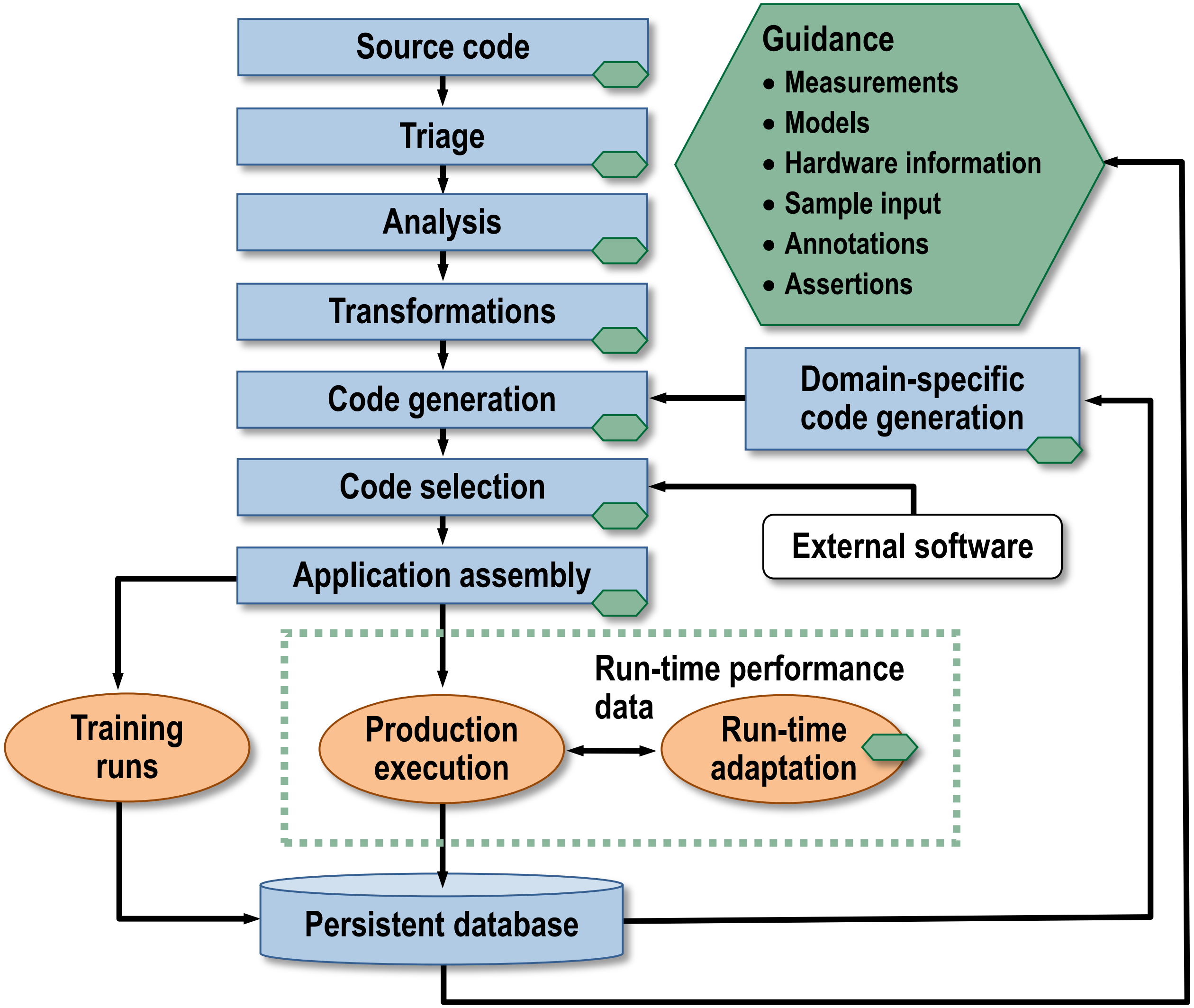
Automatic performance tuning of scientific code

Long-term goals for PERI

- Obtain hand-tuned performance from automatically generated code for scientific applications
  - General loop nests
  - Key application kernels
- Reduce the performance portability challenge facing computational scientists
  - Adapt quickly to new architectures
- Integrate compiler-based and empirical search tools into a framework accessible to application developers
- Run-time adaptation of performance-critical parameters

Automatic tuning workflow

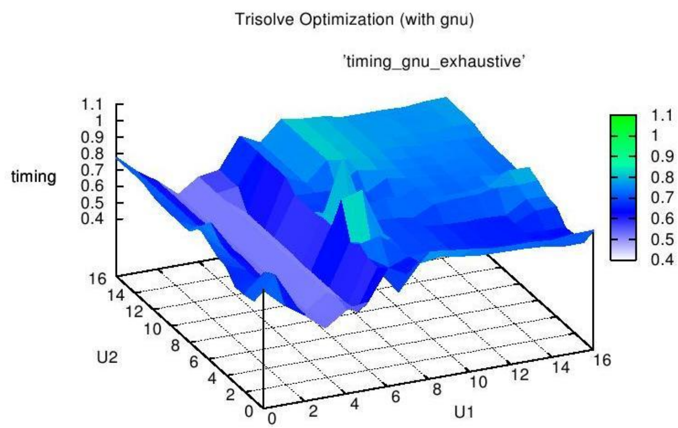
1: Triage	Where to focus effort
2: Semantic analysis	Traditional compiler analysis
3: Transformation	Code restructuring
4: Code generation	Domain-specific code
5: Code selection	Modeling and empirical search
6: Assembly	Choose the best components
7: Training runs	Performance data for feedback
8: Run-time adaptation	Optimize long-running jobs

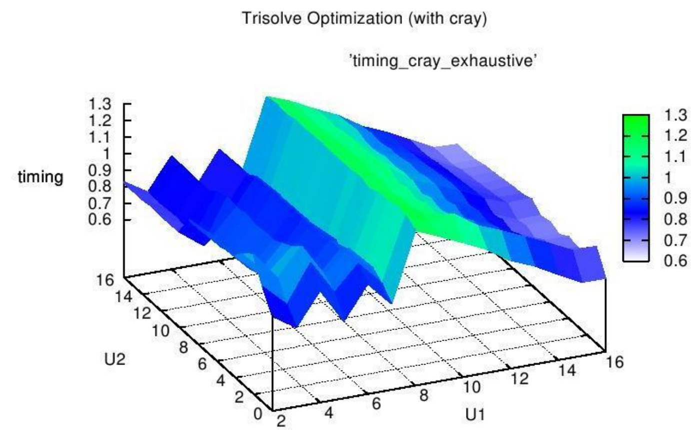


[source: Norris, ANL]

Automatic tuning examples

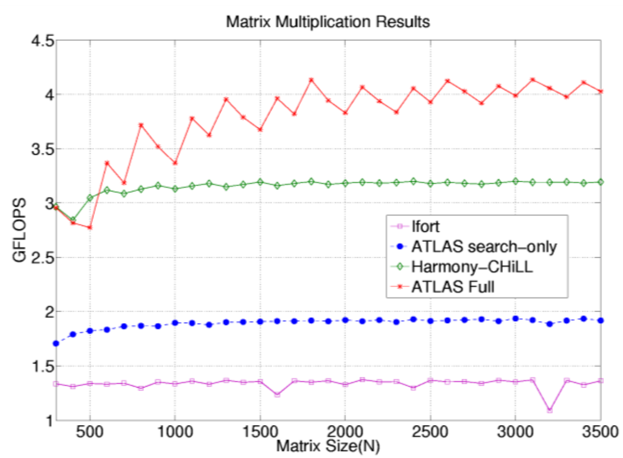
PFLOTRAN PETSc trisolve on 4K Jaguar XT5 nodes

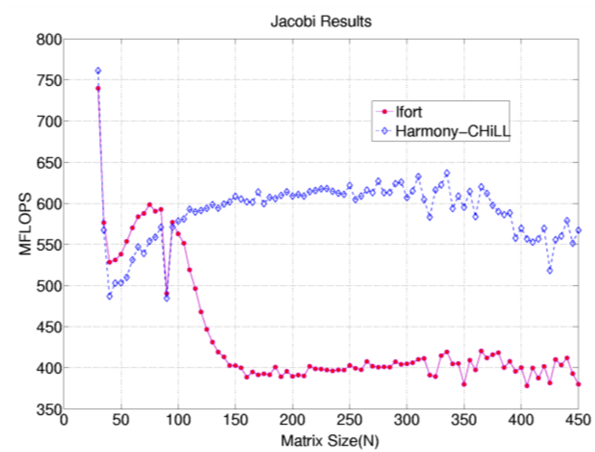




Outlined trisolve routine, used CHiLL (Utah) and Active Harmony (Maryland) to identify algorithm parameters and compiler that yield best performance

Matrix multiply on Intel Core Duo





Generated and evaluated different optimizations ***that would have been*** *prohibitively time consuming for a programmer to explore manually*

[source: Hall, University of Utah and Hollingsworth, University of Maryland]

The team

ANL
Paul Hovland
Boyana Norris

LBNL
David Bailey
Katherine Yelick

LLNL
Bronis de Supinski
Daniel Quinlan

NCSU
G. Mahinthakumar

ORNL
Philip Roth
Jeffrey Vetter
Patrick Worley

Rice
John Mellor-Crummey

UCSD
Allan Snavely
Laura Carrington

UMD
Jeffrey Hollingsworth

UNC
Rob Fowler

Oregon
Allen Malony

USC
Jacqueline Chame
Robert Lucas (PI)

UTK
Jack Dongarra
Shirley Moore

Utah
Mary Hall

Contacts

Philip C. Roth
Future Technologies Group
Computer Science and Mathematics Division
(865) 241-1543
rothpc@ornl.gov

Daniel Hitchcock
Office of Advanced Scientific Computing Research
DOE Office of Science