Center for Scalable Application Development Software

John Mellor-Crummey, Keith Cooper (Rice)
Peter Beckman, Ewing Lusk (ANL)
Jack Dongarra (UTK)
Bart Miller (Wisconsin)
Katherine Yelick (UCB/LBNL)
DOE SciDAC-2 Mission

• Develop comprehensive scientific computing software infrastructure to enable petascale science

• Develop new generation of data management and knowledge discovery tools for large data sets
DOE SciDAC-2 Program Investments

• Enabling technologies
  – Computer science
  – Applied math
  – Visualization and data mgmt.

• Science application areas
  – Physics
  – Climate
  – Groundwater
  – Fusion energy
  – Life sciences
  – Materials and chemistry

Participants
– 17 labs
– 55 universities
– 3 companies
Center for Scalable Application Development Software

• The Center was created to facilitate the scalability of applications to the petascale and beyond while fostering the development of new tools by the computer science community through support of common software infrastructures and standards.
Community Engagement
CScADS Summer Workshop Series

• Goals
  – identify challenges and open problems for leadership computing
  – brainstorm on promising approaches
  – foster collaborations between computer and application scientists
  – engage the broader community of enabling technology researchers

• Workshops to foster development of enabling technologies
  – Performance Tools for Petascale Computing - July 20-23
    • Organizers: Bart Miller and John Mellor-Crummey
    • Organizers: Rusty Lusk and Pete Beckman
  – Scientific Data and Analytics for Petascale Computing - August 3-6
    • Organizers: Rob Ross, Rusty Lusk, and Pete Beckman
  – Libraries and Autotuning for Petascale Applications - August 10-12
    • Organizers: Jack Dongarra, Keith Cooper, Rich Vuduc, and Kathy Yelick
Workshop on Libraries and Autotuning for Petascale Applications

Jack Dongarra

University of Tennessee
Oak Ridge National Laboratory
University of Manchester
Looking at the Gordon Bell Prize
(Recognize outstanding achievement in high-performance computing applications and encourage development of parallel processing)

- 1 GFlop/s; 1988; Cray Y-MP; 8 Processors
  - Static finite element analysis

- 1 TFlop/s; 1998; Cray T3E; 1024 Processors
  - Modeling of metallic magnet atoms, using a variation of the locally self-consistent multiple scattering method.

- 1 PFlop/s; 2008; Cray XT5; 1.5x10^5 Processors
  - Superconductive materials

- 1 EFlop/s; ~2018; ?; 1x10^7 Processors (10^9 threads)
Performance Development in Top500

- 1 Eflop/s
- 1 Gflop/s
- 1 Tflop/s
- 100 Mflop/s
- 100 Gflop/s
- 100 Tflop/s
- 10 Gflop/s
- 10 Pflop/s
- 1 Pflop/s
- 10 Pflop/s
- 100 Pflop/s
- Gordon Bell Winners

Graph showing performance development from 1994 to 2020.
33rd List: The TOP10 (core overloaded term)
Most likely be a hybrid design
Think standard multicore chips and accelerator (GPUs)
Today accelerators are attached
Next generation more integrated
Intel’s Larrabee in 2010
8, 16, 32, or 64 x86 cores
AMD’s Fusion in 2011
Multicore with embedded graphics ATI
Nvidia’s plans?
Exascale Computing

- Exascale systems are likely feasible by 2017±2
- 10-100 Million processing elements (cores or mini-cores) with chips perhaps as dense as 1,000 cores per socket, clock rates will grow more slowly
- 3D packaging likely
- Large-scale optics based interconnects
- 10-100 PB of aggregate memory
- Hardware and software based fault management
- Heterogeneous cores
- Performance per watt — stretch goal 100 GF/watt of sustained performance ⇒ >> 10 - 100 MW Exascale system
- Power, area and capital costs will be significantly higher than for today’s fastest systems
Five Important Features to Consider When Developing Software at Scale

- **Effective Use of Many-Core and Hybrid architectures**
  - Dynamic Data Driven Execution
  - Block Data Layout
- **Exploiting Mixed Precision in the Algorithms**
  - Single Precision is 2X faster than Double Precision
  - With GP-GPUs 10x
- **Self Adapting / Auto Tuning of Software**
  - Too hard to do by hand
- **Fault Tolerant Algorithms**
  - With 1,000,000’s of cores things will fail
- **Communication Avoiding Algorithms**
  - For dense computations from $O(n \log p)$ to $O(\log p)$ communications
  - GMRES s-step compute ($x$, $Ax$, $A^2x$, ... $A^s x$)