Breakout Group 1:
Technical challenges and needs of academic and industrial software infrastructure research and development

Breakout Session Chair: Satoshi Matsuoka (Tokyo Tech./NII)
Breakout Session Secretary: Michael Heroux (Sandia NL)
Slides: Satoshi Matsuoka and Jack Dongarra (UTK/ORNL)

IESP Workshop 2, June 28-29, Paris, France

Objectives of Group1 at This Meeting

- Roadmaps for open software R&D towards exascale
  - From 2009 to 2020
- Important Software Component Identifications
  - Existing Components – how far do they scale?
  - Missing Components – how do we start R&D?
- (Collaboration scenarios)
- (Vendor relations – acceptance, support, etc.)
  - Sufficient “exascale” market?
  - Or, more widespread demand and community forces leveraged?

Dimensions of petascale software roadmaps

- Time – Petascale now 2009 to Exascale 2018-2020
  - Yearly timeline
- System and other software components/concerns
  - E.g., programming, HA/FT, libraries, I/O & filesystems
- Architectural Diversities
  - Homogeneous vs. heterogeneous cores/ISA
  - Multithreaded shared memory vs. distributed memory
- Target system sizes---peta to exascale
  - # of compute cores
  - # of nodes
  - Other parameters, such as memory, NW, and I/O

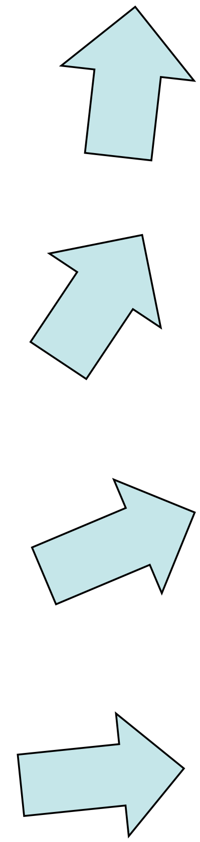
Issues facing the software components

- Extreme parallelism and Scale
- Tightening memory/bandwidth bottleneck
- High Availability despite continuous faults
- Huge power / facility requirements
- Heterogeneity and other complexities

Tightening memory/bandwidth bottleneck

Future Architecture Trends, or
the "n^2 (component density) vs. n (I/O BW) problem "

- Very Dense computation
  - Vector/SIMD/Multithreading arch.
  - Power consumption and programming the issue
- Good absolute local memory BW
  - 1TB/s per chip soon, fast/opto signaling, 3-D packaging
  - but deepening memory hierarchy
- Relatively poor node I/O channel and NW BW
  - (only) 40Gb/100Gb soon, long distance signaling hard
  - There might be breakthrus, (e.g. planar laser diode emisson), but…
- Very poor Disk Storage BW
  - SSDs are just boosts, no exception to the laws of physics



Architecture analysis and strawman targets

- Target system size and market
  - 1999/6: 2 systems Rpeak > 1TF, 70 machines > 100GF, 1 rack ~= 64GFlops (80 PentiumIII-S's), x20
  - 2009/6: 3 systems Rpeak > 1PF, 44 machines > 100TF, 1 rack ~= 10TF (200 Nehalem EPs), x100
  - 2019/6: a few exaflops (Rpeak > 1EF) ~28 machines > 100PF, 1 rack 1PF? , x500~1000
- Strawman Architectures circa 2018-20 (need to add memory)
  - Assume scaling down 45nm => 13~15nm, x10 transistors
  - Homogeneous arch: 100 cores/chip (x10), 16 FP issues/core (x4), 3.5 Ghz clock (x1.3) => 5TF/Chip, 10TF/node, 100 nodes/rack
    - 1 rack: 1 PetaF, 20,000 cores, 100 nodes
    - 1 Exa system: 20,000,000 cores (x100 BG)/ 100,000 nodes (x10 BG) / 1,000 racks (x4 ES)
  - Heterogeneous arch: 2500 (simple SIMD-Vector) cores/chip, 4 FP issues/core (x2), 2Ghz (x1.3) => 20TF/chip, 40TF node, 50 nodes/rack
    - 1 rack: 2 PetaF, 250,000 cores / 50 nodes
    - 1Exa system: 125 million cores / 25,000 nodes / 500 racks
  - May want to consider memory, power and other parameters

Software Components

- High Availability/Fault Management
  - Prevention, Tolerance, Detection, Recovery e.g., checkpointing
- Programming Languages and Models
  - Traditional: OpenMP + MPI
  - PGAS languages and variants
  - Accelerator languuages: CUDA, OpenCL
  - Others? (Locally-synchronous languages
- Compilers and Runtime Systems
  - Support for speculative computing, transactional memories, high asynchrony
  - Performance monitoring and feedback, auto-tuning
  - Debugging, verification, etc.
- IO and Filesystems (or perhaps more generally persistent storage models)
- Low level OS and Systems issues
  - virtualization, fault mgmt, memory mgmt, power mgmt, jitter/timing, …
- Numerical Libraries
  - (Lots of issues here, but should be prioritized by applications requirements)
- Systems Management and Configuration
  - (Goal should be to make future systems easier to manage than current Petascale systems)
- Networking and Integration with Broader Infrastructures
  - (Making these systems integrate with other things in the environment … whether its clouds, global filesystems, real-time data streams, etc.)

Group 1 plans

- Initially roughly agree on architectural roadmaps circa 2012 (10-30PF), 2015-6(100-300PF), 2018-20(1EF) (10-15 min)
- Split into two groups: (5 min => until 3:50 PM)
  - Intra-node issues: programming models, concurrency and fine-grained resource mgmt., languages, node OS
  - Inter-node issues: HA/resiliency/FT, Power, global config. Mgmt., I/O
  - Prioritarize discussions, identify cross-subgroup issues
- Reconvene at the end of the day and identify the cross-subgroup issues, to prepare for tomorrow (10 min)

Technology Assumptions

Year	Technology	#cores / socket	Issues /core	Nodes	Memory	Storage	Network
2012	32-28nm	8-16	4-8	10,000		HDD +SSD	100Gb
		500-1000	2	5000			
2015-16	18-15nm	24-48	8-16	30,000	3-D SOC?	SSD +HDD	SOC?
		1000-2000	2	15,000			
2018-2020	9-13nm?	100-200	8-16	100,000	3-D SOC	SSD	SOC
		2000-4000	2	50,000			

Roadmap Formulation Strategy (strawman) for IESP2

- Consider each software component / area, in operation at centers or close to deployment
- If standard / open source component exists
  - Then investigate status quo circa 2009 wrt scalability
  - If project exists to enhance scalability
    - Then identify roadmap until project termination
    - If need to continue then identify the timeline gap till 2018-20/exa
  - Else (R&D gap identification)
    - Identify research challenges envision project req.
    - Attempt to create scalability timeline to 2018-20 exa
- Else (component does not exist in open source)
  - Identify why the component does not exist
  - Conduct R&D gap identification as above

Collaboration Scenarios

1. Almost no collaboration
   – Periodic workshops, status reports of regions
   – Voluntary and ad-hoc usage of products of various projects
2. Loosely coupled collaboration
   – Focused meetings & workshops covering respective components & concerns of the software stack
   – Comparison of technical milestones, esp. for similar developments and application usage
   – Cross pollinating deployments of
3. Collaboration with Standardization
   • Definition of standards, test suites, and benchmarks, and their public availability
4. Tightly Coupled collaboration
   • International governance & funding structure
   • Cross-continental development teams (e.g., LHC/EGEE)

Roadmap Requirements (by Jack)

- Specify ways to re-invigorate the computational science software community throughout the international community.
- Include the status of computational science software activities across industry, government, and academia.
- Be created and maintained via an open process that involves broad input from industry, academia and government.
- Identify quantitative and measurable milestones and timelines.
- Be evaluated and revised as needed at prescribed intervals.
- Roadmap should specify opportunities for cross-fertilization of various agency activities, successes and challenges
- Agency strategies for computational science should be shaped in response to the roadmap
- Strategic plans should recognize and address roadmap priorities and funding requirements.

Research Topics to consider (by Jack)

- Contributors
- Priorities
- Existing expertise
- SW sustainability
- Developing new programming models and tools that address extreme scale, multicore, heterogeneity and performance
- Develop a framework for organizing the software research community
- Encourage and facilitate collaboration in education and training

Roadmap/Milestone

	2009	2010	2011	2012	2013	2014	2015	2016
Software/ Language Issues								
Sustainability								
Collaborative workshops								
Coordinated research								
Educational activities								
Standards activities								
Priorities								
Staffing								