Joint Math/CS Institutes

Summary

Mike Heroux

Key Topics Discussed

1. Effective use of many core and hybrid architecture.
2. Exploiting mixed precision. Single/double and double/quad. Sub-single/XXX.
3. Addressing complexities of node architectures.
4. Fault detection and tolerant algorithms, resilience.
5. Communication-avoiding and communication-computation concurrent algs.
6. Sensitivity analysis (broad definition)
7. Multiscale/multiphysics modeling
8. Fast implicit solves.
9. Performance degradation at scale due to load imbalance exposed by synchronization.
10. Algorithm advances: Magneto-compressive wave reformulation. Time parallel algorithms
11. Efficient multigrid, efficient multi-grid-like time algorithms
12. Effective use of new and emerging memory systems
13. Debugging of correctness and performance.
14. Motifs, interoperable motifs.
15. New capabilities to promote efficient development of optimized code.
16. New discrete optimization methods for computer system resource management.

Problems

1. **Inability to efficiently develop straight-forward, high-performance portable code.**
2. **Using machines efficiently:**
   a) Using computational units (multicore, GPUs).
   b) Using memory system efficiently.
   c) Using switch-level system efficiently (e.g. ICN).
   d) Using system power efficiently.
   e) Using synchronizations efficiently.
3. **Fault detection, tolerance and management**
4. **Sensitivities, UQ, QMU, etc.**
5. **Multiscale/Multiphysics.**
6. **Fast implicit solves.**
   – esp: Small coarse problem on big dedicated machine.
7. **Numerical stability of transient problems at scale.**
8. **Debugging of correctness and performance is untenable.**
9. **Suboptimal algorithms for computer system resource management.**

Cross-cutting tools

1. **Code, algorithm and model transformations.**
2. **Motifs and their interoperability.**
3. **Portable programming model, execution model.**
4. **Algorithms.**
   a) Implicit methods.
   b) Reformulations for larger time steps. Parareal.
   c) Discrete optimization for page mapping, router management, etc.
5. **Mixed precision, reduced data representations.**
6. **Libraries can be a test-bed, proof-of-concept for new programming/execution models.**

Joint Math/CS Institutes

Discussion Jack Dongarra

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Effective use of manycore and hybrid architecture.**
- **What are the primary bottlenecks facing computational scientists?**
  - Using them *at all* is the issue.
  - MPI is the only thing, with a little OpenMP.
  - Productivity is issue: Is libs the answer?
  - Two main issues:
    - *Algs can be mapped but CS infrastructure is missing to express algs.*
    - *Current algs don't have concurrency, we need new ones or need to refocus to higher up the alg tree (for new concurrency).*
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Proof of concept of new languages, new algs in libraries and proxies for large-scale applications.
  - We don't have a suitable programming model and (a pre-requisite) an execution model.
  - Dealing with the data layout issue: Conceptual vs. optimize layouts.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Exploiting mixed precision.**
  - Single/double and double/quad. Sub-single/XXX.
  - *Integer range also an issue: dynamic ordinal ranges.*
  - *Benefits: Reduced data movement, faster FP execution.*
- **What are the primary bottlenecks facing computational scientists?**
  - Stability and convergence.
    - *Practical use of SP at scale.*
    - *Automation of detection/correction.*
  - Expressibility of mixed data types in current languages.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Development of automated detection/correction capabilities.
  - Programming model support for mixed precision.
  - Note: Already some work started on tools for debugging numerical sensitivity (UC-Berkeley).

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Addressing complexities of node architectures.**
- **What are the primary bottlenecks facing computational scientists?**
  - Have 8-10 different approaches. Which, if any, to use?
  - Large collection of legacy code.
  - Scope of current analyses is too narrow and focused on compute-rich algorithms. Coupled to storage association problem.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Self adapting / auto-tuning of software.
  - ID appropriate representations of app needs.
  - Libraries and proxies can provide first experiences.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Fault detection and tolerant algorithms, resilience.**
- **What are the primary bottlenecks facing computational scientists?**
  - Two branches:
    - *Suitable algs, no way to express them.*
    - *Moving up the execution hierarchy to express them.*
  - Current RT/OS tools and system policies insufficient for detecting faults.
  - Issues exists within the complete hierarchy of the system from HW to App. It's a CS problem to define the interfaces.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Improve checkpointing process: E.g., Reduce the footprint of the checkpoint.
  - Expanded definition of checkpointing, improved resilience, not just today's definition, e.g., diskless w/ checksum.
  - Programming Model/Language/library level support for fault detection/recovery.
    - *Augmented programming to include algorithm "sanity checks".*
    - *Mechanism to express the relative importance of getting the right answer.*
    - *Interval arithmetic?*
    - *Automation of sanity checks.*
  - Synergy with transactional programming model? May be a natural fit with some numerical codes.
  - Needs to be solved in the next gen of systems: MTBF is getting too bad.
  - Solution needs an integrated/consistent approach through the HW/SW stack.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Communication-avoiding and communication-computation concurrent algs.**
- **What are the primary bottlenecks facing computational scientists?**
  - Lots of systems don't support this in reality. Can't measure impact.
  - Algorithms that allow concurrency (lack of these).
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Development of new algs with greater concurrency/overlap.
    - *Note: Need 100X-1MX over current concurrency.*
    - *CS Tools to allow discovery of concurrent algorithms.*
  - Full system (RT, OS, HW, …) support for exploiting concurrency.
  - Development of consistency model that supports concurrent communication.

Joint Math/CS Institutes

Discussion Trey White

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Summary of summaries.**

Town Halls

- Big solvers,
- Big data,
- Big SW
- Long simulations
- UQ, sensitivity, V&V
- Multiscale
- Hierarchical parallelism
- Design Optimization

Exascale Survey summary

- Automated diagnostics
- HW Latency
- Hierarchical algs
- Parallel programming models
- Solver technology and innovative solution techniques
- Accelerated time integration
- Model coupling
- Maintaining current libraries

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Sensitivity analysis (broad definition):**
  - Model definition variability.
  - Parameter sensitivity.
  - UQ and QMU.
- **What are the primary bottlenecks facing computational scientists?**
  - Automatic tools (AD) are deficient:
    - *Interlanguage support*
    - *C++ support*
    - *F9X support*
  - Libraries lack sensitivity interfaces.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Multi-lingual compilation environment.
  - Coupling with multi-precision computations.
  - Library APIs for sensitivities.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Multiscale modeling**
  - A wide range of scales.
  - Primary model at one scale, assimilation of subscale models.
- **What are the primary bottlenecks facing computational scientists?**
  - Data structures:
    - *Data exchange between scales.*
  - Load balancing:
    - *Balance of computation at each scale.*
    - *Work and Data repartitioning.*
  - Mathematical properties.
    - *Stability*
    - *Accuracy.*
    - *Translation between particle <-> continuum.*
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - A way to express inhomogeneous parallelism, a framework for expression.
  - Tools to support inhomogeneous parallelism.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Fast implicit solves.**
- **What are the primary bottlenecks facing computational scientists?**
  - Poor single core performance (relative to peak).
  - Inefficient use of memory bandwidth.
  - Challenging scaling (load imbalance, algorithm complexity)
  - Algorithm robustness in the presence of faults.
  - Memory constraints, esp. in forming matrix.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Algorithm resilience, esp for iterative methods.
  - Quantification of the gap between observed and achievable performance:
    - *Prediction is difficult, but bounding is easier.*
  - Quantification of performance benefit/loss for implicit over explicit.
  - Quantification architectural impediments to better algorithm performance:
    - *HW design, accessibility and control of hardware features (e.g., cache control)*

Joint Math/CS Institutes

Discussion Brian van Straalen

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Performance degradation at scale due to load imbalance exposed by synchronization.**
  - Ragged entry into collectives.
    - HW fault recovery (persistent bad actor) (Strong and weak problem).
    - Model variability (application level).
    - Memory system anomalies (transient, migratory) (Just strong problem).
- **What are the primary bottlenecks facing computational scientists?**
  - Dependence on flat SPMD model at scale. But not clear that something else is better.
  - Dependence on single-level MPI.
  - HW: Is the dependence on MPI helping or hurting our bottleneck?
- **To what areas can a tightly integrated Math/CS effort contribute?**

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Algorithm advances**
  - Magneto-compressive wave reformulation.
  - Time parallel algorithms.
- **What are the primary bottlenecks facing computational scientists?**
  - Small time steps hinder large-scale runs.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Choice of math formulation is coupled with motif selection which is coupled with parallel programming models.
  - Outreach and training: pushing awareness of new math techniques to the modeling community.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Efficient multigrid, efficient multi-grid-like time algorithms.**
- **What are the primary bottlenecks facing computational scientists?**
  - Coarse grid solve.
  - Algorithm with great amount of work, depends on small problem: serial fraction.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Highly efficient coarse solves.
  - Better distributed memory interconnect networks.
  - Resource sharing so that serial bottlenecks are not an issue?

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Effective use of new and emerging memory systems.**
- **What are the primary bottlenecks facing computational scientists?**
  - Cache memory systems poorly serve many science apps.
  - Gather/scatter, indirect mem copy seem like important opportunities for improvement.
  - Users have little control over HW/runtime system behavior.
  - Flat SPMD, MPI-only model does not scale, or make effective use of multicore.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Studies of "what-if" scenarios for our important computations.
  - Low bandwidth algorithms. Reduced sync pt algorithms.
  - Memory system aware access (eg., multicore sharing of mem. system)
  - Characterization of essential bandwidth needs and elaboration of types (streaming, G/S).
  - Development and use of simulators.
  - Study of parallel programming models, software structure for hierarchical memory systems (3-4 levels: distributed, shared 1-2, SIMD).
  - Math contribution: Exploration of alternative algorithms and formulations that are equivalent but may have more favorable performance.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Debugging of correctness and performance.**
- **What are the primary bottlenecks facing computational scientists?**
  - Debugging at scale is hard.
  - Computing environment is complex, difficult to characterize expected performance.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Mathematical assertions: identities, equivalences, …
  - Framework for detecting and responding to assertion and model violations.

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: Motifs, interoperable motifs.**
- **What are the primary bottlenecks facing computational scientists?**
  - Difficulty in communicating between domain scientist and library/CS expert. No common language.
  - Optimization efforts and communication between domain scientist and computer scientist is done at the source code level.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Identify and catalogue motifs.
  - Identify motifs used in an application/library.
  - Educate community about motif concept and catalog.
  - Explore efficient implementations for motifs, consider alternative motifs.

Joint Math/CS Institutes

Discussion Barry Smith

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: New capabilities to promote efficient development of optimized code.**
- **What are the primary bottlenecks facing computational scientists?**
  - Hand optimizations error-prone, static.
  - Conceptual data layout non-optimal, e.g., ijk grid indexing fused to storage.
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Code transformation tools, allowing user intervention, source to source
    - *Loop unrolling for specific sizes: sparse SV, LAPACK for small sizes.*
  - Compiler transformations to reduce synchronization points: e.g., Bi-CGSTAB transformation.

Joint Math/CS Institutes

Contribution: Rich Graham, Ron Brightwell

Challenges Facing Scalable Applications:

Where are the 'gaps' between potential and achieved performance?

- **Topic: New discrete optimization methods for computer system resource management.**
- **What are the primary bottlenecks facing computational scientists?**
  - Ad hoc methods for computer system resource management.
    - *Page placement, router scheduling, …*
- **To what areas can a tightly integrated Math/CS effort contribute?**
  - Discrete optimization algorithms for system tools.

Joint Math/CS Institutes

Institute Description

Identify the ‘optimal’ end state in 10 years time . . .

- Libraries and tools will have built-in support for sensitivity information.
- Libraries and tools will have built-in predictive performance models.
- Libraries are fault-aware; resilience or informative or both.
- Libraries will have performance portability.
- Real application performance will match achievable performance on state-of-the-art scalable systems.
- Sensitivity, UQ and QMU will have penetrated application areas where the forward problem has sufficient fidelity.
- Multiscale/multiphysics modeling will be ubiquitous.
- The timestep limit will be set by accuracy considerations, not for stability needs.
- We will have a portable parallel programming and execution model.

What does a joint Math/CS Institute look like?

- **Institute is:**
  - Staffing of Math and CS, Labs and Universities, continuum of skills.
  - Approximately 10-20 members, single PI.
  - Single theme with multiple projects.
  - Integrated Math and CS effort.
- **Size:**
  - $1M too small. $3M OK.
- **How do we obtain an integrated effort?**
  - Focus on problems that require synergistic Math & CS effort.
- **How do we introduce joint accountability?**
  - Proposed work must clearly demonstrate need for combined Math & CS research to succeed.
  - Milestones must depend on joint effort.

Elements of a successful program

- **What are signs of success?**
  - Science teams beating down the door to get what we produce.
  - Vendors picking up concepts we develop.
  - Application codes rely more on libraries.
  - Successful multiscale/multiphysics applications.
- **Are there external dependencies that must be taken into account?**
  - Architecture roadmaps.
  - Industry language standards, programming APIs.

Additional Comments

- **Are there additional items that arose in discussion that need to be brought to light?**
  - How will we get long-term support for tools and software we produce?
  - Institutes must be long-lived, 5-10 years needed to realize the vision.