Predictive Science Academic Alliance Program (PSAAP)

The slides that follow were presented at the PSAAP Bidder's Meeting May 16-17, 2006 and represent the ASC Trilab authors and interests as presented in the associated White Paper for this subject area.

PSAAP: Computer Science

Xabier Garaizar, garaizar@llnl.gov (LLNL)
David Jefferson, jefferson6@llnl.gov (LLNL)
Marv Alme, alme@lanl.gov (LANL)
Daniel Rintoul, mdrinto@sandia.gov (SNL)

May 16, 2006

National Nuclear Security Administration

UCRL-PRES-221275

ASC™

Preliminaries

- **MSCs**
  - **focus on large-scale, multidisciplinary, scalable and integrated simulations**
  - **have as primary goal to develop a verified and validated predictive capability for an application**

- **Avoiding a red herring**
  - **Computer Science Research is not a primary goal of the PSAAP**
  - **Computer Science in support of ASC applications is a component of the PSAAP**

Advanced Simulation & Computing

Background

- ASC codes are "conservative" on issues relating to
  - Code architecture
  - Computing paradigms
  - Computer languages
  - …
- How could we advance the science of prediction if we were given a clean slate, with freedom to re-invent scientific computation?

Thrust Areas

- Scalable algorithms
- Algorithms and programming technology specific to parallel simulation
- New parallel programming models
- Parallel componentization technology
- Software fault avoidance / detection / recovery
- OS support for capability and capacity machines
- Scalable I/O technology and abstractions

Advanced Simulation & Computing

Scalable Algorithms

- New scalable algorithms at application and systems level.
- Must be novel in some way, or cut across many application areas.
- Examples:
  - better performance, better error estimators, faster convergence, better conservation
  - algorithms that are MPMD, better balanced, use interval arithmetic, etc.
  - algorithms that address OS, I/O, fault tolerance, or other systems problems
  - algorithms that scale to 100,000 processors or more

Programming technology specific to parallel simulation

- Componentization, formal interfaces for simulations as objects, scriptable simulations (external control)
- Algorithms for coupling simulations
- Unification of continuous and discrete simulation
- Physical units (kg, watts, Hz) as part of language type system
- Domain specific constructs, e.g. support for grad, curl, or tensor ops
- Efficient execution engines for complex models with disparate time and length scales
- Techniques for fully unstructured space-time meshes in 3+1 or more dimensions (i.e. arbitrarily variable time steps and arbitrary time-varying meshes)

Parallel programming models

- **Programming models express parallelism at nine orders of magnitude of scale, from pipelined vector ops (10^9 Hz, 1 byte) to wide-area transactions (1 Hz, 10^9 bytes). We need technologies such as**
  - **nestable, composable parallel abstractions, classes and objects**
  - **componentization (composable units of separately-developed code)**
  - **migratable units (load balancing, fault avoidance)**
  - **checkpoint/restart, replication, rollback, redundancy, or retry mechanisms for handling faults at all levels**
  - **parallel high-level communication primitives (e.g. parallel remote procedure call)**
  - **speculative or optimistic algorithms**
  - **parallel instrumentation, optimization, debugging at all levels**
  - **new software build tools -- less error prone and more parallel**

Parallel componentization technology

- Simulation codes should not be standalone executables; they should be packaged as components to be used as units larger computations
- They should be dynamically instantiable and launchable in parallel
- The should have language-independent interfaces that go beyond traditional APIs to include also mesh information, physical units, etc.
- Components should be migratable, checkpointable, and should provide introspection and external control capability
- Components should be internally parallel, and communicate with each other in parallel
  - requires solutions to the "MxN problem"

ADVANCED SIMULATION & COMPUTING

Fault management

- **All scalable software must be designed with fault management in mind**
  - — **new algorithms, with internal algorithmic redundancy for fault detection/correction**
  - — **support for checkpoint/restart, retry, replication, rollback, etc. in programming languages, compilers, and especially libraries**
  - — **OS or runtime system support for anticipation of, and migration away from, hardware faults**
  - — **communication routing around faults**
  - — **modularized management of faults, i.e. recovery confined to the component where the fault occurs, without affecting other components**

OS support

- Capability and capacity machines need OS support for fault handling, load balancing, synchronization, componentization, I/O etc.
  - — boot different OS's in different partitions to allow richer mix of jobs to share capacity machines
  - — parallel boot, job launch, and DLL mechanisms
  - — support for load migration, job compaction, fault prediction, avoidance, and recovery
  - — dynamic node allocation for expanding and contracting jobs on capacity machines
  - — collective system calls
  - — efficient, preemptive and priority gang scheduling
  - — one-sided, interrupting communication

Parallel I/O

- Parallel I/O traditionally traditionally lags other aspects of parallel computation, but many ASC applications ahead may be dominated by I/O. That could justify research in:
  - parallel file systems and abstractions
  - parallel relational databases
  - parallel geometric and temporal databases
  - parallel input from sensor arrays, including asynchronous and real time input
  - parallel visualization systems

Conclusion

- Successful proposals
  - will not treat these as independent computer science research areas
  - will strongly connect them to the simulation capability and ASC application requirements

- This is not a prescribed list of topics, but an illustration of some issues that might be addressed in a successful proposal. Other topics not mentioned may be supported as long as the connection to ASC applications is clear.

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract no. W-7405-Eng-48.

UCRL-PRES-221275

Advanced Simulation & Computing