

Applications and Libraries

- Areas of interest to this TC
 - Math & I/O Libraries
 - Novel Algorithm Research
 - Scientific toolkits / libraries
 - Profiling tools
 - Fault tolerance

- Relation to other TCs
 - Programming Models: Ease of programming & debugging
 - Performance: Enhance throughput & provide measurement tools
 - Architectures: same

Topic	Urgency	Duration	Responsiveness	Applicability	Timeline
Math & I/O Libraries	Critical	Medium	Moderate	Broad	Immediate
Novel Algorithm Research	Critical	Long	High	Broad	Soon
Scientific toolkits / Libraries	Useful	Long	High	Science	Eventually
Profiling tools	Important	Long	High	HPC	Eventually
Fault tolerance	Important	Long	High	HPC	Eventually

Math & I/O Libraries

Topic	Urgency	Duration	Responsiveness	Applicability	Timeline
Math & I/O Libraries	Critical	Medium	Moderate	Broad	Immediate

Description

Numeric libraries such as BLAS< LAPACK, Trilinos, FFTW, BGL, grid operations, AMR, etc. and I/O libraries to ease the pain of checkpoint/restarts.

Notes from Discussion

These are the building blocks of applications that developers are dependent upon. It is important to have an implementation in place first followed by high performance versions.

Some of this activity will be done by vendors anyway, however it is important to have some independent work in this area to assure completeness. These libraries need to be scalable all the way from desktop to HPC.

Knowledge gained during the development of these libraries regarding the models that worked, and those that didn't work should be documented in a knowledge base and be accessible to developers of new libraries on other platforms.

Portability across platforms is critical for these libraries.

Relations to other TCs

Performance
Programming Models
Architectures

Related Projects

MAGMA
cuBLAS
Trilinos
PETSc
Adios
PVFS, PLFS, GPFS, etc

Novel Algorithm Research

Topic	Urgency	Duration	Responsiveness	Applicability	Timeline
Novel Algorithm Research	Critical	Long	High	Broad	Soon

Description

Given the difference between hybrid architectures and traditional CPUs, algorithm development is critical to leverage the unique characteristics of each platform. This requires not only development of new algorithms, but also revisiting algorithms that were discarded in the past because they might have been too compute intensive.

Notes from Discussion

This is required to enable methods development on advanced architectures. Algorithm is some version of said method that we can implement. Implementation is a specific instantiation of that method. Implementations need to be architecture aware and should focus on reducing time to solution or energy to solution rather than maximizing metrics such as ops/load or ops/Watt.

Again, Knowledge gained during the development of these libraries regarding the models that worked, and those that didn't work should be documented in a knowledge base and be accessible to developers of new libraries on other platforms.

Portability across platforms is critical for these libraries.

Relations to other TCs

Programming Models

Related Projects

CFDNS on Cell

FEAST-GPU

Scientific Toolkits / Libraries

Topic	Urgency	Duration	Responsiveness	Applicability	Timeline
Scientific toolkits / Libraries	Useful	Long	High	Science	Eventually

Description

These toolkits allow for a high level expression of the mathematics / physics.

Notes from Discussion

These toolkits build on the Math and I/O libraries to provide a richer development environment for specific physics or domains. Some examples include Grid operation libraries, PDE libraries, Graph libraries. Success requires strong interaction between Computer Scientists and Subject Matter Experts. It was decided that the physics research resides in the Applications whereas the Computer Science resides in programming models.

Again, Knowledge gained during the development of these libraries regarding the models that worked, and those that didn't work should be documented in a knowledge base and be accessible to developers of new libraries on other platforms.

Portability across platforms is critical for these libraries.

Relations to other TCs

Programming Models

Related Projects

SCOUT

libMesh

netCDF

Toolkits within Matlab

BGL/PBGL

Profiling Tools

Topic	Urgency	Duration	Responsiveness	Applicability	Timeline
Math & I/O Libraries	Critical	Medium	Moderate	Broad	Immediate
Novel Algorithm Research	Critical	Long	High	Broad	Soon
Scientific toolkits / Libraries	Useful	Long	High	Science	Eventually
Profiling tools	Important	Long	High	HPC	Eventually
Fault tolerance	Important	Long	High	HPC	Eventually

Description

Tools to provide developers feedback on application performance and identify optimization possibilities.

Notes from Discussion

As the memory hierarchy gets deeper, and the node itself gets more complex, it becomes critical to have good profiling tools. These tools would provide feedback on data motion / location, cache hits/misses, instruction pipeline analysis, bus contention, latency, packet size, etc. and help optimize both time to solution and energy to solution. The ownership for this resides equally with the performance TC.

Again, Knowledge gained during the development of these libraries regarding the models that worked, and those that didn't work should be documented in a knowledge base and be accessible to developers of new libraries on other platforms.

Portability across platforms is critical for these libraries.

Relations to other TCs

Performance
Architectures

Related Projects

Open Speedshop
VTUNE
VAMPIR
oProfile
gProf
Tau

Fault Tolerance

Description

There needs to be a means for the application to interact with the system to obtain information regarding the state of a given node and the possibility of failure of the node, or a core within a node.

Notes from Discussion

As clusters increase in size, we need to move beyond checkpoint/restart mechanisms for fault tolerance. Any new methods developed should have minimal impact on resources. A generic API for interaction with the system needs to be defined so that hardware vendors can implement them and application developers can program to them. One possibility is to have a set of 'spare' nodes and when diagnostics indicate that a node may be close to failure, the system could automatically migrate the data on the bad node to a spare thus allowing the application to continue.

Portability across platforms is critical for these libraries.

Relations to other TCs

Programming models
Architectures

Related Projects

Erlang
OpenMPI
compilers