MPI Internals Interface
&
Miscellaneous Topics WG

Bronis R. de Supinski, David Goodell, William D. Gropp, Adam Moody, **Martin Schulz,** Rajeev Thakur

Overview

- MPI Internals Interface (PIMPI)
  - Motivation
  - Approach and Features
    - Environment Variables
    - Runtime state and counters
    - Tracing support
- Miscellaneous Working Group
  - Proposed by Adam Moody
  - Cover additional related topics
    - Establish list
  - Format

Query Interface to Access MPI Internal Information

Goal: Standardized interface to gain access to internal MPI performance and state information

Proposal in preparation by Bronis R. de Supinski, David Goodell, William D. Gropp, Martin Schulz, Rajeev Thakur (plus discussions at CsCADS meeting)

Motivation

- Access for performance tools necessary
  - PMPI tools only provide partial information
  - Additional insight for application developers
  - Standardization of library performance information
- Performance Information interface (PIMPI)
  - Lightweight mechanism
  - Track information specific performance aspects
- Target scenarios
  - Investigate load imbalance
  - Detect resource/connection limitations

Contrast to PERUSE

- Similar ideas, but …
  - No fixed event model
  - Purely based on publish/subscribe interface
  - No required information
- Callbacks
  - Should we keep them?
  - What is the performance impact?
- Keep concept of environment variable queries
  - Detect all relevant variables
  - Additional: scope for changes

Environment Variables

- Routine to query all environment variables that impact performance (incl. description)

  `PIMPI_Iter_get_env`

  - Returns
    - Name
    - Description
    - Default Value
    - Scope (can it be changed and, if so, after MPI_Init)
  - Need to be able to get current value?
  - Need to be able to set value?

Publish/Subscribe Interface

- Initialize and Finalize routines
  - Required call to use the interface?
  - Before/after MPI_Init/Finalize?
- Each MPI implementation provides a set of performance variables
  - No predefined variables
  - Defined naming scheme?
- Applications/Tools can query for all available variables
  - Implementation provides description
  - Managed through (predefined) handles?

Possible Information Types

- State (usable through sampling)
  - Operations: query
  - Example: Compute/Block/Communicate
- Statistical state
  - Operations: query, reset
  - Example: high/low watermark of queue lengths
- Counter
  - Operations: start, stop, query, reset
  - Example: number of messages sent
  - Could be wrapped into a PAPI implementation
- Individual events (see next slide)
  - Example: unexpected message arrival

Event Delivery

- Callbacks
  - Invoked on every event
  - Impact on implementation/Overhead?
  - What can be allowed inside a callback?
  - Callback sampling?
- Alternative: Trace Ring Buffer
  - Write all events into a ring buffer
  - Including local timestamp
  - Query and reset ring buffer information through calls
  - What happens when buffer is full?

Open Issues

- Activation
  - On by default?
  - Explicit start/stop routines?
- Should this be its own WG?

Working Group for Miscellaneous Topics

Goal: Cover topics not included into other WGs, but that are not worth their own WG

Proposed and led by Adam Moody (LLNL)

(1) Standardization of MPI Compiler Wrappers

- Recommend standard names for compiler wrappers
  - Non binding
  - mpicc, mpif77, etc.
  - Similar to mpiexec in MPI 2.0
- Motivation: increased portability

- Additional suggestions for the MPI environment:
  - Standardized return values for mpiexec
  - Standardized mechanism to pass arguments through mpicc, … (--showme)

(2) Public MPI Test Suite

- Testing framework
  - Publicly available
  - Intended for MPI users and implementors
  - Open SVN repository?
- What to use as starting points?
  - Intel MPI tests?
  - Other suites?
- Main target: Correctness w.r.t. the MPI standard
- Other (optional) issues (?):
  - Composability (performance "guidelines")
  - Unexpected (although correct) behavior (MPITEST)

(3) MPI User Survey

- Organize survey on user requirements
  - Targeted for MPI 3.0
  - Unified survey for all user/code developer groups
- Starting point: existing surveys by
  - Bronis de Supinski and Adam Moody
  - Rainer Keller

(4) MPI Debugging Interface

- Automatic Process Acquisition Interface (APAI)
  - Query MPI tasks in a job
  - Currently mostly used by Totalview, but also other tools
  - Dubbed as "Debugger Interface"
- Standardization would be beneficial
  - Currently each MPI has slight variations in the API
  - Difficult for tools
- Varying locations
  - MPI tasks/runtime
  - Communication subsystem (e.g., OpenRTE)
  - External resource manager (e.g., SLURM)

Discussion

- Support
  - More than four members already agreed to WG
- Topics
  - Any other topics to include?
  - Any topics that should not be in here?
  - Add the PIMPI interface to this WG?
- Format
  - Phone discussions?