OVERVIEW OF MPC

Forum Teratec | Patrick CARRIBAULT, Julien JAEGGER, Marc PERACHE
CEA, DAM, DIF, F-91297 Arpajon, France

JUNE 24th 2015
Context

- **Starting point: programming model used today**
  - Generalization of hybrid programming model
  - Most used standards: MPI+OpenMP
  - Current architectures: petaflopic machines such as TERA100/TERA1000/Curie
  - Languages: C, C++ and Fortran
  - Large amount of application codes and libraries

- **Main target: transition to new programming models for Exascale**
  - Provide efficient runtime to evaluate mix of programming models
    - Unique programming model may be a non-optimal approach
  - Provide smooth/incremental way to change large codes and associated libraries
    - Avoid full rewriting before any performance results
    - Keep existing libraries at full current performance coupled with application using other programming model
      - Example: MPI application calling OpenMP-optimized schemes/libraries

- **Multi-Processor Computing (MPC)**
Team Activity Overview

- **Team overview**
  - Runtime system and software stack for HPC
  - Team as of June 2015 (CEA/DAM and CEA/Intel/UVSQ ECR Lab)
    - 3 research scientists, 5 PhD students, 1 apprentice, 1 engineer, 3 interns
    - Contact: marc.perache@cea.fr, patrick.carribault@cea.fr or julien.jaeger@cea.fr
  - Available software
    - MPC framework
    - MALP
    - JCHRONOSS
  - Website for team work: http://hpcframework.com

- **MPC framework**
  - Unified parallel runtime for clusters of NUMA machines
    - Idea: one process per node, compute units exploited by user-level threads
  - Integration with other HPC components
    - Parallel memory allocator, compilers, debuggers, topology tool…
  - Tool website: http://mpc.hpcframework.com
MPC Capability

- **Supported programming models**
  - Full MPI 1.3, parts of MPI 2 and MPI 3
  - Full OpenMP 3.1
  - Pthread

- **Networks**
  - Support of TCP
  - Support of InfiniBand with multirail

- **Resource Manager**
  - Slurm
  - Hydra

- **Architectures**
  - X86, x86-64, MIC

- **Compilers**
  - Compatible with GCC and ICC

- **Debuggers**
  - Compatible with GDB (patched GDB provided)
  - Allinea DDT

- **Topology**
  - Use HWLOC to detect topology
APIs Support
• **Goals**
  - Smooth integration with multithreaded model
  - Low memory footprint
  - Deal with unbalanced workload
• **MPI 1.3**
  - Fully MPI 1.3 compliant
• **Thread-based MPI**
  - Process virtualization
  - Each MPI rank is a thread
• **Thread-level features**
  - From MPI2 standard
  - Handle up to MPI_THREAD_MULTIPLE level (max level)
  - Unification with PThread representation
• **Inter-node communications**
  - TCP, InfiniBand
• **Tested up to 80,000 cores with various HPC codes**
MPC Execution Model: Example #1 (MPI)

Application with 4 MPI tasks
OpenMP

- **OpenMP 3.1**
  - OpenMP 3.1-compliant runtime integrated to MPC
  - Directive-lowering process done by provided patched GCC (C,C++,Fortran) or ICC
    - Generate calls to MPC ABI instead of GOMP (GCC OpenMP implementation)
    - MPC runtime now compatible with KMPC (Intel ABI for use with Intel’s icc, icpc and ifort)

- **Hierarchical Representation**
  - Organize threads of the same OpenMP team in a hierarchical manner
  - Use a tree-like structure to link the threads
    - NUMA-aware design
MPC Execution Model: Example #2 (OpenMP)

- Application with 1 MPI task + 4 OpenMP threads
Application with 2 MPI tasks + 4 OpenMP threads
Automatic Privatization

- **Global variables**
  - Expected behavior: duplicated for each MPI task
  - Issue with thread-based MPI: global variables shared by MPI tasks located on the same node

- **Solution: Automatic privatization**
  - Automatically convert any MPI code for thread-based MPI compliance
  - Duplicate each global variable

- **Design & Implementation**
  - Completely transparent to the user
  - When parsing or creating a new global variable: flag it as thread-local
  - Generate runtime calls to access such variables (extension of TLS mechanism)
    - Linker optimization for reduce overhead of global variable access

- **Compiler support**
  - New option to GCC C/C++/Fortran compiler (\texttt{-fmpc-privatize})
    - Patched GCC provided with MPC (4.8.0)
  - Intel's ICC support automatic privatization with same flag (\texttt{-fmpc-privatize})
    - ICC 15.0.2 and later
Conclusion
Conclusion

• **Runtime**
  - Provide widely spread standards
  - MPI 1.3+ (soon MPI-IO and non-blocking collectives), OpenMP 3.1, PThread
  - Available at [http://mpc.hpcframework.com](http://mpc.hpcframework.com) (version 2.5.2)
  - Optimized for manycore and NUMA architectures

• **Programming models**
  - Provide unified runtime for MPI + X applications
  - New mechanism to mix thread-based programming models: Extended TLS
  - Automatic privatization

• **Tools**
  - Paratools: TAU support for profiling
  - Allinea: DDT support for debugging
  - Intel: ICC/ICPC/IFORT support for automatic privatization
References

2015

• E. Saillard, P. Carribault and D. Barthou, *MPI Thread-level Checking for MPI+OpenMP Applications*. (To Appear in EuroPar’15)

2014

• S. Didelot, P. Carribault, M. Pérache and W. Jalby, *Improving MPI communication overlap with collaborative polling*. (Computing 2014)

• J. Jaeger, P. Carribault, and M. Pérache, *Fine-grain data management directory for OpenMP 4.0 and OpenACC*. (CCPE 2014)

• E. Saillard, P. Carribault, and D. Barthou. *PARCOACH: Combining static and dynamic validation of MPI collective communications*. (JHPCA 2014)

• J. Clet-Ortega, P, Carribault, and M, Pérache, *Evaluation of openmp task scheduling algorithms for large numa architectures*. (Euro-Par’14)

• A. Mahéo, P. Carribault, M. Pérache, and W. Jalby, *Optimizing collective operations in hybrid applications*. (EuroMPI ’14)

• E. Saillard, P. Carribault, and D. Barthou, *Static validation of barriers and worksharing constructs in openmp applications*. (IWOMP 2014)
References

2013

• J.-B. Besnard, M. Pérache and W. Jalby, Event streaming for online performance measurements reduction. (ICPP 2013)
• J. Jaeger, P. Carribault, M. Pérache, Data-Management Directory for OpenMP 4.0 and OpenACC, (HeteroPar’13)
• S. Didelot, P. Carribault, M. Pérache, W. Jalby, Improving MPI Communication Overlap With Collaborative Polling, (Springer Computing Journal)
• E. Saillard, P. Carribault, D. Barthou. Combining Static and Dynamic Validation of MPI Collective Communications. (EuroMPI’13)

2012

• S. Didelot, P. Carribault, M. Pérache, W. Jalby, Improving MPI Communication Overlap With Collaborative Polling, (EuroMPI’12)
• A. Maheo, S. Koliai, P. Carribault, M. Pérache, W. Jalby, Adaptive OpenMP for Large NUMA Nodes, (IWOMP’12)
• M. Tchiboukdjian, P. Carribault, M. Pérache, Hierarchical Local Storage: Exploiting Flexible User-Data Sharing Between MPI Tasks, (IPDPS’12)
• J.-Y. Vet, P. Carribault, A. Cohen, Multigrain Affinity for Heterogeneous Work Stealing, (MULTIPROG’12)
References

2011

• P. Carribault, M. Pérache, H. Jourdren, *Thread-Local Storage Extension to Support Thread-Based MPI/OpenMP Applications* (IWOMP’11)

2010

• P. Carribault, M. Pérache, H. Jourdren, *Enabling Low-Overhead Hybrid MPI/OpenMP Parallelism with MPC* (IWOMP’10)
• K. Pouget, M. Pérache, P. Carribault, H. Jourdren, *User Level DB: a Debugging API for User-Level Thread Libraries* (MTAAP’10)

2009

• M. Pérache, P. Carribault, H. Jourdren, *MPC-MPI: An MPI Implementation Reducing the Overall Memory Consumption* (EuroPVM/MPI'09)

2008

• F. Diakhaté, M. Pérache, H. Jourdren, R. Namyst, *Efficient shared-memory message passing for inter-VM communications* (VHPC'08)
• M. Pérache, H. Jourdren, R. Namyst, *MPC: A Unified Parallel Runtime for Clusters of NUMA Machines* (EuroPar'08)
• S. Zuckerman, M. Pérache, W. Jalby, *Fine tuning matrix multiplications on multicore*, (HiPC'08)