STATUS OF OPENFABRICS INTERFACES (OFI) SUPPORT IN MPICH

Yanfei Guo, Assistant Computer Scientist

Argonne National Laboratory
AGENDA

- What is MPICH?
- Why OFI?

Current support
  - MPICH 3.3 series (CH4)
  - MPICH 3.4 series (CH4)

Ongoing work
  - New Collective Framework
  - GPU Support
WHAT IS MPICH?

- MPICH is a high-performance and widely portable open-source implementation of MPI
- It provides all features of MPI that have been defined so far (up to and include MPI-3.1)
- Active development led by Argonne National Laboratory and University of Illinois at Urbana-Champaign
  - Several close collaborators who contribute features, bug fixes, testing for quality assurance, etc.
    - IBM, Microsoft, Cray, Intel, Ohio State University, Queen’s University, Mellanox, RIKEN AICS and others
- Current stable release is MPICH-3.3.2
- Latest release is MPICH-3.4a2
- [www.mpich.org](http://www.mpich.org)
MPICH: GOAL AND PHILOSOPHY

- MPICH aims to be the preferred MPI implementation on the top machines in the world
- Our philosophy is to create an “MPICH Ecosystem”
MOTIVATION

Why OFI/OFI WG?

- Support for diverse hardware through a common API
- Actively, openly developed
  - Bi-weekly calls
  - Hosted on Github
- Close abstraction for MPI
  - MPI community engaged from the start
- Fully functional sockets provider
  - Prototype code on a laptop
- Strong Vendor Support
Introducing the CH4 device

- Replacement for CH3, but we will maintain CH3 till all of our partners have moved to CH4
- Co-design effort
  - Weekly telecons with partners to discuss design and development issues
- Two primary objectives:
  - Low-instruction count communication
    - Ability to support high-level network APIs (OFI, UCX)
    - E.g., tag-matching in hardware, direct PUT/GET communication
  - Support for very high thread concurrency
    - Improvements to message rates in highly threaded environments (MPI_THREAD_MULTIPLE)
    - Support for multiple network endpoints (THREAD_MULTIPLE or not)
MPICH WITH CH4 DEVICE OVERVIEW

Application
MPI Interface

MPI Layer
- Machine-independent Collectives
- Derived Datatype Management
- Group Management

Abstract Device Interface (ADI)

CH4
- CH4 Core
  - Architecture-specific Collectives
  - Active Message Fallback
  - GPU Support Fallback

Netmods
- OFI
- UCX

Shmmods
- POSIX
- XPMEM

© OpenFabrics Alliance
- **Lightweight communication**
  - Reducing overhead in instruction count and memory usage
  - Inline Libfabric with MPICH further reduces overhead

- **Improvements in MPI one-sided communication**
  - Enabling HW accelerated RMA

- **Communication hints**
  - Allowing user to tell MPI to optimize for the crucial subset of features
**Virtual Network Interface (VNI)**
- Each VNI abstracts a set of network resources
- Some networks support multiple VNIs: InfiniBand contexts, scalable endpoints over Intel Omni-Path
- Traditional MPI implementation uses single VNI
  - Serializes all traffic
  - Does not fully exploit network hardware resources

**Utilizing multiple VNIs to maximize independence in communication**
- Separate VNIs per communicator or per RMA window
- Distribute traffic between VNIs with respect to ranks, tags, and generally out-of-order communication
- M-N mapping between Work-Queues and VNIs
MPI+THREAD HYBRID PROGRAMMING PERFORMANCE

Multithreaded Transfer Model Current MPI (3.1)

User Expose Parallelism with COMM/TAG

Work-Queue Data Transfer Model with MPI Endpoints

Message size (B)

-5 0 5 10 15 20 25 30 35 40 45 50 55

Messages/s (x 10^6)

4 16 64 256 1024 4096 16384 65536

- MPI_THREAD_SINGLE
- MPI_THREAD_MULTIPLE with MPI_COMM_WORLD
- MPI_THREAD_MULTIPLE with separate COMMs
UPCOMING MPICH-3.4 AND FUTURE PLANS

- **New Collective Framework**
  - Optimizing collective based on communication characteristic and availability of HW acceleration
  - JSON configuration generated by external profiler

- **GPU Support**
  - Communication using GPU-resident buffers
  - Non-contiguous datatypes

© OpenFabrics Alliance
Thanks to Intel for the significant work on this infrastructure

Two major improvements:

• C++ Template-like structure (still written in C)
  • Allows collective algorithms to be written in template form
  • Provides “generic” top-level instantiation using point-to-point operations
  • Allows device-level machine specific optimized implementations (e.g., using triggered operations for OFI or HCOLL for UCX)

• Several new algorithms for a number of blocking and nonblocking collectives (performance tuning still ongoing)

Contributed by Intel (with some minor help from Argonne)
SELECTING COLLECTIVE ALGORITHM

- **Choose Optimal Collective Algorithms**
  - Optimized algorithm for certain communicator size, message size
  - Optimized algorithm using HW collective support
  - Making decision on each collective call

- **Generated Decision Tree**
  - JSON file describing choosing algorithms with conditions
  - JSON file created by profiling tools
  - JSON parsed at MPI_Init time and applied to the library

*Contributed by Intel (with some minor help from Argonne)*
GPU SUPPORT PLAN

- **Internode**
  - Native GPU support through Librabric and UCX
  - Developing fallback path for no native GPU support

- **Intranode**
  - GPU support in SHM

- **Intranode**
  - Supporting non-contiguous datatype for GPU
  - Packing/Unpacking using host/device buffer

*Partnership with Intel, Cray, Mellanox, NVIDIA and AMD*
**CURRENT STATE OF GPU SUPPORT**

- **Native Internode**
  - With Libfabric, UCX and supported GPUs

- **Yaksa Datatype Engine** ([https://github.com/pmodels/yaksa](https://github.com/pmodels/yaksa))
  - Support H2D, D2H, D2D
  - Packing to appropriate GPU or CPU stage buffer for either native or fallback route
  - 1.0 release with CUDA backend
  - Intel is contributing on the Intel Xe backend

---

Packing the Y-Z plane of a 3D matrix (2x2x<dims>)

© OpenFabrics Alliance
- CH4 already in at http://github.com/pmodels/mpich
- MPICH-3.4 GA coming out this summer
  - Multi-VNI support
  - Collective Selection Framework
THANK YOU

Yanfei Guo, Assistant Computer Scientist
Argonne National Laboratory