Screencast: What is [Open] MPI?

Jeff Squyres
May 2008

CISCO
What is MPI?

- Message Passing Interface
  - “De facto” standard
  - Not an “official” standard (IEEE, IETF, …)
- Written and ratified by the MPI Forum
  - Body of academic, research, and industry representatives
- MPI is two spec documents:
  - MPI-1 and MPI-2
  - Specified interfaces in C, C++, Fortran 77/90
MPI Forum

• Published MPI-1 spec in 1994
• Published MPI-2 spec in 1996
  ▪ Additions to MPI-1
• Recently reconvened (Jan 2008)
  ▪ Working on MPI-2.1 (small bug fixes)
    • Will issue a single document for MPI 1+2
  ▪ Also working on MPI-2.2 (bigger bug fixes)
  ▪ Also working on MPI-3 (entirely new stuff)
What is MPI?

- Software implementations of spec
  - Mostly host-side software
- “Middleware”
  - Sits between the application and network
  - Simplifies network activity to the application
- Source code portability
  - Run apps on commodity clusters and “big iron” supercomputers
MPI High-Level View

User application

MPI API

Operating System
What is MPI?

• Intended to deliver very high performance
  ▪ Low latency, high bandwidth

• Examples
  ▪ 2 servers + switch, user-level processes
  ▪ DDR InfiniBand
    • ~1-2μs half-round trip 0-byte ping pong
    • ~14Gbps bandwidth for large messages
  ▪ 10Gbps Ethernet
    • ~5-7μs half-round trip 0-byte ping pong
    • ~10Gbps bandwidth for large messages
MPI Implementations

• Many exist / are available for customers
  ▪ Vendors: HP MPI, Intel MPI, Scali MPI
    • Have their own support channels
  ▪ Open source: Open MPI, MPICH[2], …
    • Rely on open source community for support
    • But also have some vendor support

• Various research-quality implementations
  ▪ Proof-of-concept
  ▪ Not usually intended for production usage
Why So Many MPI’s?

• A complicated question…
  ▪ Some aim to make money (closed source)
  ▪ Some targeted at specific platforms
  ▪ Others aimed at research (open source)
  ▪ History and politics also involved (yuck)

• Open MPI is a fascinating blend of research and industry
Target Audience

• Scientists and engineers
  ▪ Don’t know or care how network works
  ▪ Not computer scientists
  ▪ Sometimes not even [very good] programmers

• Parallel computing
  ▪ Using tens, hundreds, or thousands of servers in a single computational program
  ▪ Intended for high-performance computing
Parallel Computing

• Use 10’s, 100’s, 1000’s of processors
  ▪ When the computation is too big for one server

• Spread the job across multiple servers
  ▪ Individual user processes running in concert
  ▪ Acting together as a single application

• More RAM

• More processing power

• Divide and conquer
MPI Abstracts the Network

• Sockets? Shared memory? Ethernet? InfiniBand? …something else?
  ▪ Doesn’t matter
• Application calls MPI_SEND / MPI_RECV
  ▪ The Right magic happens
• Connections are made automatically
  ▪ Sockets (IP address/port)
  ▪ Shared memory (e.g., mmap file)
  ▪ InfiniBand (queue pair setup)
Example: 4 Servers

Server
User application
MPI API
Operating System
Server
User application
MPI API
Operating System
Server
User application
MPI API
Operating System
Server
User application
MPI API
Operating System

Network
Example: 2 Servers

- User application
  - MPI API
  - Operating System

Server

Network

- User application
  - MPI API
  - Operating System

Server
Example: 1 Server

User application | User application | User application | User application
---|---|---|---
MPI API | MPI API | MPI API | MPI API

Operating System

Server
Runtime

• MPI implementations also include a runtime environment
  ▪ Need to start processes on multiple servers simultaneously
  ▪ Typically requires some user-level setup
  ▪ Common source of errors
int rank, size, message = -1, tag = 11111;
MPI_Init(NULL, NULL); /* Startup */
MPI_Comm_rank(..., &rank); /* Who am I? */
MPI_Comm_size(..., &size); /* How many peers do I have? */
to = (rank + 1) % size;
from = (rank + size - 1) % size;
/* Send a trivial message around in a ring */
if (0 == rank) {
    message = 42;
    MPI_Send(&message, 1, MPI_INT, to, tag, ...);
    MPI_Recv(&message, 1, MPI_INT, from, tag, ...);
} else {
    MPI_Recv(&message, 1, MPI_INT, from, tag, ...);
    MPI_Send(&message, 1, MPI_INT, to, tag, ...);
}
MPI_Finalize();
Trivial MPI Application

Process 0
- to=1
- from=3

Process 2
- to=3
- from=1

Process 1
- to=2
- from=0

Process 3
- to=0
- from=2

MPI_Send(…)
MPI_Recv(…)

MPI_Send(…)
MPI_Recv(…)

MPI_Send(…)
MPI_Recv(…)

MPI_Send(…)
MPI_Recv(…)

MPI_Send(…)
MPI_Recv(…)

MPI_Send(…)
MPI_Recv(…)

MPI_Send(…)
MPI_Recv(…)

MPI_Send(…)
MPI_Recv(…)

May 2008
Screencast: What is [Open] MPI? 18
Open MPI

- YAMPI (yet another MPI)
  - ...but not really
  - Replaces several prior MPI’s
- Collaborate = great MPI implementation
  - What a concept!
  - Lots of “MPI-smart” people out there
- Open source project
  - Influenced by both research and industry
Open MPI

• It’s two words!
  - Open MPI
  - NOT “OpenMPI”

• Frequently abbreviated “OMPI”
  - Pronounced “oom-pee”
Open MPI

• Fundamentally based on plugins
  ▪ A.k.a. “components” or “modules”

• Plugins for everything
  ▪ Back-end resource manager
  ▪ Back-end network
  ▪ Back-end checkpointer
  ▪ …etc.
  ▪ Currently ~30 types of plugins in Open MPI

• Recurring theme: run-time decisions
Resources

- MPI Forum
  - http://www.mpi-forum.org/

- Open MPI
  - General web site: http://www.open-mpi.org/
  - FAQ: http://www.open-mpi.org/faq/

- Magazine columns about MPI
  - http://cw.squyres.com/