Advanced Technical Skills (ATS) North America

Running MPI applications on Linux over Infiniband cluster with mvapich / mvapich2

IBM High Performance Computing
February 2010

Y. Joanna Wong
yjw@us.ibm.com

MVAPICH: MPI over InfiniBand and iWarp

- MVAPICH
  - Open-source High Performance MPI Library for InfiniBand and 10GigE/iWARP (Internet Wide Area RDMA Protocol) clusters
  - Project started in 2001 at Ohio State University, Network-Based Computing Laboratory, headed by Prof. Dhabaleswar K Panda
  - MVAPICH (MPI-I) and MVAPICH2 (MPI-2)
  - Downloads from more than 840 organizations in 44 countries

    http://mvapich.cse.ohio-state.edu
  - Distributed by many IBA, 10GigE, server vendors and systems integrators in their software stacks, including OFED
  - Available with the OpenFabrics/Gen2 stack and supports uDAPL device on linux and solaris to work with network supporting uDAPL
  - Will build support for one interconnect at a time
  - Latest releases: mvapich 1.2

mvapich

- MPI-1 implementation based on MPICH (latest MPICH 1.2.7) and MVAICH
- Latest version 1.2RC1, released Jan 29, 2010
- Single code base for multiple architectures and underlying transport interfaces
  - Support for OpenFabrics/Gen2, OpenFabrics/Gen2-Hybrid (use Unreliable Datagram UD, Reliable Connection RC, and eXtended Reliable Connection XRC transports of InfiniBand), Shared-memory only channel, Qlogic InfiniPath, TCP/IP
  - Scalable and robust job startup with mpirun_rsh
  - Runtime and compile time tunable parameters (at MPI and network layers) for large scale systems and future platforms

mvapich…

- Compiler commands (mpicc, mpiCC, mpicxx, mpif77, mpif90) are wrapper scripts that will generate the correct flags, compiler options, includes, defines and libraries to be added to the compile and link commands
- The underlying Fortran, C, and C++ compilers are the compilers that were used to build mvapich.
- Can override the compilers and linkers using environment variables.

	mvapich compiler	default compiler variable	override environment variable	default linker variable	override environment variable
C	mpicc	CCBASE	MPICH_CC	CLINKERBASE	CLINKER
C++	mpiCC	CCCBASE	MPICH_CCC	CCLINKERBASE	CCLINKER
C++	mpicxx	CCCBASE	MPICH_CCC	CCLINKERBASE	CCLINKER
F77	mpif77	F77BASE	MPICH_F77	F77LINKERBASE	F77LINKER
F90	npif90	F90BASE	MPICH_F90	F90LINKERBASE	F90LINKER

Running MPI applications with mvapich 1.1

- MPI requires fully automatic logins on remote hosts
  - ssh authentication should be enabled between all hosts to allow password-less login
- Set up the environment variables to be included in the shell environment (e.g. .bashrc)

  export PATH=$MVAPICH_HOME/bin:$PATH

  export LD_LIBRARY_PATH=$MVAPICH_HOME/lib:$LD_LIBRARY_ PATH

  where $MVAPICH_HOME (e.g. /usr/mpi/intel/mvapich-1.1) is the top-level installed directory where mvapich 1.1 is installed
- Compile with MPI compiler e.g. in $MVAPICH_HOME/examples

  mpicc –o cpi cpi.c –lm
- Run the MPI program using mpirun or mpirun_rsh:

  mpirun_rsh –ssh -np 16 –hostfile hosts ./cpi

  where hosts is the file with 16 hostnames, one per line

mvapich…

- Enhanced more scalable mpirun_rsh framework is introduced in mvapich 1.1 to significantly cut down job start-up time on large clusters.

  usage: mpirun_rsh [-v] [-rsh|-ssh] [-paramfile=pfile] [-debug] -[tv] [-xterm] [-show] [-legacy] [-use_xlauncher][xlauncher-width W] -np N (-hostfile hfile | h1 h2 ... hN) a.out args

  Where:

  v => Show version and exit
  rsh => to use rsh for connecting
  ssh => to use ssh for connecting
  paramfile => file containing run-time MVICH parameters
  debug => run each process under the control of gdb
  tv => run each process under the control of totalview
  xterm => run remote processes under xterm
  show => show command for remote execution but don't run it
  legacy => use old startup method (1 ssh/process)
  np => specify the number of processes
  h1 h2... => names of hosts where processes should run
  or hostfile => name of file containing hosts, one per line
  a.out => name of MPI binary
  args => arguments for MPI binary

mvapich…

- MVAPICH runtime environment variable can be specified in a file to be included with the mpirun_rsh option –paramfile *pfile* or the variable name-value pair be included just before the executable, e.g.

  mpirun_rsh –np 4 –hostfile hosts ENV1=value1 ENV2=value2 ./cpi

- Some interesting MVAPICH parameters
  - VIADEV_CPU_MAPPING: process to CPU mapping
    - Only take effect with VIADEV_USE_AFFINITY=1 (VIADEV_USE_AFFINITY does not take effect with _AFFINITY_ is not defined)
    - For memory intensive applications where number of processes per node is less than the number of cores, it is preferable to distribute the processes on different sockets to minimize memory contention. For example running 4 processes on a server with 2 quad-core sockets:

      VIADEV_CPU_MAPPING=0:2 (compatible with mvapich2)

      or VIADEV_CPU_MAPPING=0,2 (mvapich 1.0 syntax)

mvapich…

- VIADEV_USE_APM : enable to support network fault recovery by using InfiniBand Automatic Path Migration mechanism (for OFED/IB adapters)

  mpirun_rsh –np 16 VIADEV_USE_APM=1 ./cpi

- Shared memory aware collectives:
  - Shared memory implementations of collectives (MPI_Allreduce, MPI_Reduce, MPI_Barrier, MPI_Bcast, enhanced MPI_Allgather) are enabled by default**.**
  - All can be disabled with : VIADEV_USE_SHMEM_COLL=0
  - Or selectively disabled with:
    - VIADEV_USE_SHMEM_ALLREDUCE=0
    - VIADEV_USE_SHMEM_REDUCE=0
    - VIADEV_USE_SHMEM_BARRIER=0
    - VIADEV_USE_SHMEM_BCAST=0
    - VIADEV_USE_NEW_ALLGATHER=0

mvapich2 (MPI-2 over InfiniBand)

- MPI-2 implementation based on MPICH2 ADI3 layer (device aimed at higher performance networks)
- Work closely with ANL to incorporate latest updates from MPICH2
- Current stable version, released Oct 29, 2009.
  - mvapich2 1.4 based on mpich2 1.0.8p1
- MVAPICH2 Features (source: MPICH2 BOF at SC08)
  - Unified design over OFED to support InfiniBand and 10GigE/iWARP
  - Scalable and robust daemon-less job startup with the new mpirun_rsh framework
  - High performance and scalable implementations: RDMA write and RDMA read
  - Multi-threading support
  - Optimized support for two-sided and one-sided operations (put, get, and accumulate)

mvapich2…

- MVAPICH2 Features (cont'd)
  - Integrated multi-rail support
    - Multiple adapters, queue pairs, multiple ports/adapter
  - Efficient shared-memory point-to-point communication
  - Optimized collectives – optimizations for multi-core platforms
  - Two different communication progress : polling and blocking
  - On-demand connection management for large clusters
  - Multiple-solutions for Fault-tolerances
    - Network-level FT support with Automatic Path Migration (APM)
    - Process-level FT with systems-level checkpoint-restart with shared-memory and shared-memory collectives
  - Hot-spot avoidance mechanism for alleviating network congestion in large clusters
  - Totalview Debugger support

mvapich2…

- Support for Multiple Interfaces/Adapters
  - OpenFabrics / Gen2-IB
    - All InfiniBand adapters (SDR, DDR, QDR) supporting Gen2
    - Mellanox ConnectX adapters
  - uDAPL
    - Linux-IB
    - Solaris-IB
  - OpenFabrics / Gen2-iWARP
    - Chelsio 10GigE

mvapich2…

- Compiler commands (mpicc, mpicxx, mpif77, mpif90) are wrapper scripts that will generate the correct flags, compiler options, includes, defines and libraries to be added to the compile and link commands
- The underlying Fortran, C, and C++ compilers are the compilers that were used to build mvapich2
  - Can print all the MPI Library information (configuration options) with:

    mpiname -a
- Can override the compilers and linkers using environment variables.

	mvapich2 compiler	default compiler variable	override environment variable
C	mpicc	CC	MPICH_CC
C++	mpicxx	CXX	MPICH_CXX
F77	mpif77	FC	MPICH_F77
F90	mpif90	F90	MPICH_F90

Running MPI applications with mvapich2-1.2

- MPI requires fully automatic logins on remote hosts
  - ssh authentication need to be set up to allow password-less login
- Set up the environment variables to be included in the shell environment (e.g. .bashrc)

  export PATH=$MVAPICH2_HOME/bin:$PATH

  export LD_LIBRARY_PATH=$MVAPICH2_HOME/lib:$LD_LIBRARY_ PATH

  where $MVAPICH2_HOME (e.g. /usr/mpi/intel/mvapich2-1.2) is the top-level installed directory where mvapich2 1.2 is installed
- Compile with the MPI compilers e.g.

  mpicc –o *app* *app*.c –lm
- Run the MPI program using new daemonless startup mpirun_rsh (new in version 1.2):

  mpirun_rsh –ssh -np 16 –hostfile *hfile app*

  where *hfile* is the name of the hostfile with 4 hostnames, one per line

mvapich2…

- Can run the MPI program with mpiexec, managing the processes with the mpd daemon
- Setup the mpd environment
  - create the hidden file .mpd.conf in home directory with “a secret word” stored is required. The file cannot be accessed by other users.

    echo “secretword=*mpd.secret.word*” > ~/.mpd.conf

    chmod 600 ~/.mpd.conf

    where my.secret.word is any word.
- Start the mpd daemons on the compute nodes with the mpdboot command.

  For example, on 4 hosts listed in file *hfile*, with 8 cpus on each host

  mpdboot –r ssh –n 4 –f *hfile* –-ncpus=8

  The file *hfile* has one host per line

mvapich2…

- Can use the command mpdtrace to list hostname of the mpd's in the ring
- Start the MPI program with mpiexec For example, running *app* on 4 nodes with 4 processes per node:

  $MVAPICH2_HOME/bin/mpiexec -machinefile *hfile* –np 16 –envall *./app*

  where *hfile* is a file with the name of the 4 nodes, e.g.

  node001:4
  node002:4
  node003:4
  node004:4

  mpiexec is the soft link to mpiexec.py script

MVAPICH2 parameters

- MVAPICH2 parameters can be set as environment variable in the shell startup script or as command line arguments in mpirun_rsh and mpiexec :

  mpirun_rsh –np *16* –hostfile *hfile* MV2_ENABLE_AFFINITY=1 *./app*

  or

  export MV2_ENABLE_AFFINITY=1

  mpiexec –np 16 –hostfile *hfile* -envall ./*app*

  or

  mpiexec –np 16 –hostfile **hfile** –env MV2_ENABLE_AFFINITY 1 ./*app*

- MV2_ENABLE_AFFINITY
  - enable CPU affinity: MV2_ENABLE_AFFINITY=1
    - For better performance for single-threaded programs
  - disable CPU affinity: MV2_ENABLE_AFFINITY=0
    - Need to be disabled in MPI+OpenMP so that all threads of a process will not be scheduled to run on the same CPU/core

mvapich2…

- MV2_CPU_MAPPING
  - Allows users to specify process to CPU/core mapping
  - Particularly useful on multi core systems where performance can be different if processes are mapped to different ones, e.g.

    MV2_CPU_MAPPING 0:1:4:5
  - Will take effect only if MV2_ENABLE_AFFINITY=1