PROGRAMACIÓN DE SISTEMAS DISTRIBUIDOS

Simon Pickin
Alberto Núñez

Instalación y Ejecución (MPI)

Índice

Instalación de OpenMPI

- Descargar el release estable de OpenMPI más reciente
  - http://www.open-mpi.org/software/ompi/v1.10/downloads/openmpi-1.10.1.tar.gz
- Descomprimir en directorio para los fuentes <fuentes>, p.ej. en /home/***user***/lib/
  - cd /home/***user***/lib/
    tar xfz openmpi-1.10.1.tar.gz o bien gunzip -c openmpi-1.10.1.tar.gz | tar xf -
- Crear la variable de entorno MPI_HOME con valor el directorio de instalación
  - Por ejemplo:
    mkdir /home/***user***/openmpi-1.10.1; export MPI_HOME=/home/***user***/openmpi-1.10.1
- Configurar (ver doc. o usar configure --help), o bien en un directorio temporal:
  - mkdir /tmp/***user***/openmpi-1.10.1; cd /tmp/***user***/openmpi-1.10.1
    <fuentes>/openmpi-1.10.1/configure –prefix=$MPI_HOME

  o bien no:
  - cd $MPI_HOME; <fuentes>/openmpi-1.10.1/configure
- Compilar e instalar
  - cd $MPI_HOME; make
    sudo make install
- Para instalar en un cluster, hay 3 opciones
  - Compartir el directorio donde se ha instalado, p.ej. por NFS/MOUNT
  - Replicar el directorio de instalación en todos los nodos (por defecto es /usr/local/bin)
  - Realizar la instalación en cada nodo

Instalación de OpenMPI

- Añadir al fichero .bashrc
  - `export MPI_HOME=<el camino del directorio de instalación>`
  - `export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$MPI_HOME/lib`
  - `export PATH=$PATH:$MPI_HOME/bin`
- Instalar el servidor de ssh
  - Si va a ejecutar programas en múltiples ordenadores

Compilar y ejecutar programas MPI

- Compilar programas OpenMPI
  - `$> mpicc -o nombreEjecutable ficheroFuente.c`
    - Por ejemplo: `mpicc –o hello_c hello_c.c`
- Ejecutar programas OpenMPI
  - `$> mpiexec -hostfile ficheroHosts -np numProc nombreEjecutable`
- Documentación de OpenMPI
  - Páginas man
    - `http://www.open-mpi.org/doc/v1.10/`

Ejecutar una prueba en el localhost

- Alternativa 1 (ejecución en el localhost implícito)
  - `mpiexec -np 3 $MPI_HOME/examples/hello_c`
- Alternativa 2 (ejecución en máquina nombrada explícitamente)
  - `touch ficheroHosts; echo "localhost" > ficheroHosts`
  - `mpiexec -np 3 -host ficheroHosts $MPI_HOME/examples/hello_c`
- Alternativa 3 (ejecución con núm de procesos explícito en máquina explícita)
  - `touch ficheroHosts; echo "localhost slots=3" > ficheroHosts`
  - `mpiexec -np 3 -host ficheroHosts $MPI_HOME/examples/hello_c`
- Alternativa 4 (como alternativa 3 pero info. dada en la línea de comandos)
  - `mpiexec -np 3 -host localhost $MPI_HOME/examples/hello_c`
- Otras alternativas
  - En OpenMPI, `mpiexec` ≡ `mpirun` ≡ `orterun`

Ejecutar una prueba en 2 máquinas

- Contenido del fichero file1:

```
ordenador1
ordenador2
```

- Contenido del fichero file2

```
ordenador1 slots=2
ordenador2
```

- Por defecto (el comportamiento es configurable) el comando openmpi
  - `$> mpiexec -np 3 –hostfile file1 $MPI_HOME/examples/hello_c`

  ejecuta procesos 0 y 2 en ordenador1 y proceso 1 en ordenador2

- Por defecto (el comportamiento es configurable) el comando openmpi
  - `$> mpiexec -np 3 –hostfile file2 $MPI_HOME/examples/hello_c`

  ejecuta procesos 0 y 1 en ordenador1 y proceso 2 en ordenador2

Mapeo procesos-cores en OpenMPI

- Mapeo de procesos a nodos/cores
  - Usa `hwloc`, nombre del paquete y del subproyecto del proyecto OpenMPI
    - `http://www.open-mpi.org/projects/hwloc/`
  - Tres de las posibilidades para mapeo permanente - *binding* o afinidad – (hay más):
    - Prohibir al SO operativo mover los procesos entre cores
    - Prohibir al SO operativo mover procesos entre cores de distintos *sockets* del mismo nodo
    - No restringir al SO (puede mover los procesos entre cualesquieres cores del mismo nodo)

    Ver: `https://www.open-mpi.org/doc/v1.10/man1/mpiexec.1.php#sect8`
  - Opción (de `mpiexec`) por defecto:
    - versión actual de OpenMPI (1.10.1): `--bind-to core –-mapby socket`
      - Salvo si núm procesos <= 2 en cuyo caso: `--bind-to core --mapby core`
    - algunas versiones anteriores de OpenMPI: `-bind-to-none -bycore`
- *Binding* de procesos
  - Ver: `https://www.open-mpi.org/doc/v1.10/man1/mpiexec.1.php#sect9`
  - Opción para ver los *binding* usados en una ejecución dada: `-report-bindings`
  - Más explicación sobre estas opciones (algo anticuado pero útil todavía):
    - `http://blogs.cisco.com/performance/open-mpi-v1-5-processor-affinity-options/`

Instalación de MPICH

- Descargar el *release* estable de MPICH más reciente (en Ubuntu: use 3.1)
  - `http://www.mpich.org/static/downloads/3.2/mpich-3.2.tar.gz`
- Descomprimir en directorio para los fuentes <*fuentes*>, p.ej. en `/home/`***user***`/lib/`
  - `cd /home/`***user***`/lib/`
    `tar xfz mpich-3.2.tar.gz` o bien `gunzip -c mpich-3.2.tar.gz | tar xf -`
- Crear la variable de entorno `MPI_HOME` con valor el directorio de instalación
  - Por ejemplo: `mkdir /home/`***user***`/mpich-3.2; export MPI_HOME=/home/`***user***`/mpich-3.2`
  - El directorio de instalación por defecto es: `/usr/local/bin`
- Configurar (ver doc. o usar `configure --help`), o bien en un directorio temporal:
  - `mkdir /tmp/`***user***`/mpich-3.2; cd /tmp/`***user***`/mpich-3.2`
    `<`*fuentes*`>/mpich-3.2/configure --disable-f77 --disable-fc –prefix=$MPI_HOME`

  o bien no:
  - `cd $MPI_HOME; <fuentes>/mpich-3.2/configure --disable-f77 --disable-fc`
- Compilar e instalar
  - `cd $MPI_HOME; make`
    `sudo make install`
- Para instalar en un cluster, hay 3 opciones
  - Compartir el directorio donde se ha instalado, p.ej. por NFS/MOUNT
  - Replicar el directorio de instalación en todos los nodos (por defecto es `/usr/local/bin`)
  - Realizar la instalación en cada nodo

Instalación de MPICH

- Añadir al fichero .bashrc
  - `export MPI_HOME=<el camino del directorio de instalación>`
  - `export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$MPI_HOME/lib`
  - `export PATH=$PATH:$MPI_HOME/bin`
- Instalar el servidor de ssh
  - Si va a ejecutar programas en múltiples ordenadores
- Hydra: el gestor de procesos por defecto en las últimas versiones de MPICH
  - El gestor de procesos MPD (el defecto hasta v1.2) está deprecado y el gestor de procesos SMPD (el único que funcionaba en Windows) lo estará pronto
  - Con Hydra, no hace falta lanzar el gestor de procesos explícitamente
  - Hydra también gestiona la asignación de procesos a cores (*process-core binding*)
    - Ver transparencia más adelante para más detalles
  - MPICH permite el uso de otros gestores de procesos

Compilar y ejecutar programas MPI

- Compilar programas MPICH
  - `$> mpicc -o nombreEjecutable ficheroFuente.c`
    - Por ejemplo: `mpicc –o hello_c hello_c.c`
- Ejecutar programas MPICH
  - `$> mpiexec -f ficheroHosts -n numProcesos nombreEjecutable`
- Documentación de MPICH
  - Guía del instalador
    - `http://www.mpich.org/static/downloads/3.2/mpich-3.2-installguide.pdf`
  - Guía del usuario
    - `http://www.mpich.org/static/downloads/3.2/mpich-3.2-userguide.pdf`

Ejecutar una prueba en el localhost

- Alternativa 1 (ejecución en el localhost implícito)
  - `mpiexec -n 3 $MPI_HOME/examples/cpi`
- Alternativa 2 (ejecución en máquina nombrada explícitamente)
  - `touch ficheroHosts; echo "localhost" > ficheroHosts`
  - `mpiexec -n 3 –f ficheroHosts $MPI_HOME/examples/cpi`
- Alternativa 3 (ejecución con núm de procesos expícito en máquina explícita)
  - `touch ficheroHosts; echo "localhost:3" > ficheroHosts`
  - `mpiexec -n 3 –f ficheroHosts $MPI_HOME/examples/cpi`
- Alternativa 4 (como alternativa 3 pero info. dado en la línea de comandos)
  - `mpiexec -n 3 -hosts {localhost:3} $MPI_HOME/examples/cpi`
- Otras alternativas
  - En MPICH se puede usar la opción `-machinefile` en vez de `-f`

Ejecutar una prueba en 2 máquinas

- Contenido del fichero ficheroHosts1:

```
ordenador1
ordenador2
```

- Contenido del fichero ficheroHosts2

```
ordenador1:2
ordenador2
```

- Por defecto (el comportamiento es configurable) el comando MPICH
  - `$> mpiexec -n 3 –f ficheroHosts1 $MPI_HOME/examples/cpi`

  ejecuta `cpi` con procesos 0 y 2 en ordenador1 y proceso 1 en ordenador2
- Por defecto (el comportamiento es configurable) el comando MPICH
  - `$> mpiexec -n 3 –f ficheroHosts2 $MPI_HOME/examples/cpi`

  ejecuta `cpi` con procesos 0 y 1 en ordenador1 y proceso 2 en ordenador2

Mapeo procesos-cores en MPICH

- Mapeo de procesos a nodos con Hydra
  - Uso del colon en el hostfile (ver transparencia anterior)
- Mapeo de procesos a cores con Hydra
  - Usa `hwloc`, nombre del paquete y del subproyecto del proyecto OpenMPI
  - Para usar mapeo permanente (*binding*) explícito en vez del mapeo por defecto:
    - Utiliza la opción siguiente al configurar MPICH `-enable-hydra-procbind`
  - Algunas versiones anteriores de MPICH y de Hydra: la opción `–binding` del comando `mpiexec` permitía elegir entre 5 posibles binding:
    - *round-robin buddy-allocation, closest packing,* definidos por usuario en la linea de comandos, definidos por usuario en fichero
  - Versión actual de MPICH y Hydra: reemplazar `–binding` con `-bind-to` y `map-by` que permiten más flexibilidad (gracias a `hwloc`), ver OpenMPI
    - Para más detalle: `mpiexec –bind-to -help`
  - Opción (de `mpiexec`) por defecto: `-map-by`, lo mismo que `-bind-to`, que es mandatorio
  - Opción (de `mpiexec`) para ver el mapeo usado en una ejecución dada:
    - habría que acceder directamente a `hwloc`, aunque ver programa `print_cpus_allowed`

Claves ssh

- Por defecto, MPI utiliza el programa SSH para comunicarse con los nodos
- Generalmente, en cada acceso con SSH se pide la clave al usuario
- Para que no pida la clave en cada ejecución, ejecutar en consola:
  - `$>cd $HOME/.ssh`
  - `$>ssh-keygen -t dsa`
  - `$>ssh-keygen -t rsa`
  - `$>cat *.pub > authorized_keys`

MPI en los laboratorios de informática

- Actualmente, hay ~~dos~~ una instalac~~iones~~ión de MPI en los laboratorios:
  - Versión 1.10.0 de OpenMPI (ver con `mpiexec --version`; más actual es 1.10.1)
    - Ver `http://www.open-mpi.org/`
    - Para ejecutar: `mpiexec` o `mpirun`
    - Para usar un hostfile: opción `-machinefile` o `-hostfile` de `mpiexec/mpirun`
    - Para especificar cuantos procesos en un host del hostfile: `hostname slots=<num>`
    - Bibliotecas dinámicas para `LD_LIBRARY_PATH`: `/usr/lib/openmpi/lib`
  - Versión 1.2.1 de MPICH (más actual es 3.2) **DESINSTALADA**
    - Ver `http://www.mpich.org/`
    - Para ejecutar: `mpiexec.mpich2` o `mpirun.mpich2`
    - Para usar un hostfile: opción `-machinefile` o `–f` de `mpiexec/mpirun`
    - Para especificar cuantos procesos en un host del hostfile: `hostname:<num>`
    - Bibliotecas dinámicas para `LD_LIBRARY_PATH`: `/usr/lib/mpich/lib`