

Lab 6a - mpirun e mpicc

<https://elly2023.smfi.unipr.it/mod/page/view.php?id=5002>

OBIETTIVO

Studiare e analizzare mpirun e mpicc, i comandi principali per l'utilizzo del modello a memoria distribuita con MPI nei sistemi HPC.

Attività svolte

Come prima cosa ho creato la directory di lavoro "mkdir -p ~/HPC2324/mpi/mpirun/" e copiato all'interno gli esercizi tramite "cp /hpc/home/roberto.alfieri/SHARE/mpi/mpirun/*."

mpirun

"mpirun" è il comando che si occupa della creazione e dell'esecuzione dei processi.

Ho provato i seguenti comandi:

- module load intel impi

```
[martina.genovese@ui01 mpirun]$ module load intel impi
```

- mpirun --host wn22,wn24 hostname

```
[martina.genovese@ui01 mpirun]$ mpirun --host wn22,wn24 hostname
wn24
wn24
wn24
wn24
wn22
wn22
wn22
wn22
```

- mpirun -np 6 hostname # esegue 6 istanze del comando hostname all'interno di task eseguiti sullo stesso host da cui è stato eseguito il comando

```
[martina.genovese@ui01 mpirun]$ mpirun -np 6 hostname
ui01.hpc.unipr.it
ui01.hpc.unipr.it
ui01.hpc.unipr.it
ui01.hpc.unipr.it
ui01.hpc.unipr.it
ui01.hpc.unipr.it
```

- mpirun -machine machine.txt hostname # machine.txt contiene l'elenco degli host

```
[martina.genovese@ui01 mpirun]$ mpirun -machine machine.txt hostname
[mpiexec@ui01.hpc.unipr.it] HYD_hostfile_parse (../../../../src/pm/i_hydra/libhydra/hostfile/hydra_hostfile.c:56): unable to open host file: machine.txt
[mpiexec@ui01.hpc.unipr.it] mfile_fn (../../../../src/pm/i_hydra/mpiexec/mpiexec_params.c:447): error parsing hostfile
[mpiexec@ui01.hpc.unipr.it] match_arg (../../../../src/pm/i_hydra/libhydra/arg/hydra_arg.c:52): match handler returned error
[mpiexec@ui01.hpc.unipr.it] HYD_arg_parse_array (../../../../src/pm/i_hydra/libhydra/arg/hydra_arg.c:74): argument matching returned error
[mpiexec@ui01.hpc.unipr.it] mpiexec_get_parameters (../../../../src/pm/i_hydra/mpiexec/mpiexec_params.c:1283): error parsing input array
[mpiexec@ui01.hpc.unipr.it] main (../../../../src/pm/i_hydra/mpiexec/mpiexec.c:1745): error parsing parameters
```

- mpirun -machine machine.txt -np 2 hostname

```
[martina.genovese@ui01 mpirun]$ mpirun -machine machine.txt -np 2 hostname
[mpiexec@ui01.hpc.unipr.it] HYD_hostfile_parse (../../../../../../src/pm/i_hydra/libhydra/hostfile/hydra_hostfile.c:56): unable to open host file: machine.txt
[mpiexec@ui01.hpc.unipr.it] mfile_fn (../../../../../../src/pm/i_hydra/mpiexec/mpiexec_params.c:447): error parsing hostfile
[mpiexec@ui01.hpc.unipr.it] match_arg (../../../../../../src/pm/i_hydra/libhydra/arg/hydra_arg.c:52): match handler returned error
[mpiexec@ui01.hpc.unipr.it] HYD_arg_parse_array (../../../../../../src/pm/i_hydra/libhydra/arg/hydra_arg.c:74): argument matching returned error
[mpiexec@ui01.hpc.unipr.it] mpiexec_get_parameters (../../../../../../src/pm/i_hydra/mpiexec/mpiexec_params.c:1283): error parsing input array
[mpiexec@ui01.hpc.unipr.it] main (../../../../../../src/pm/i_hydra/mpiexec/mpiexec.c:1745): error parsing parameters
```

- mpirun -machine machine.txt -np 2 -npernode 1 hostname

```
[martina.genovese@ui01 mpirun]$ mpirun -machine machine.txt -np 2 -npernode 1 hostname
[mpiexec@ui01.hpc.unipr.it] HYD_hostfile_parse (../../../../../../src/pm/i_hydra/libhydra/hostfile/hydra_hostfile.c:56): unable to open host file: machine.txt
[mpiexec@ui01.hpc.unipr.it] mfile_fn (../../../../../../src/pm/i_hydra/mpiexec/mpiexec_params.c:447): error parsing hostfile
[mpiexec@ui01.hpc.unipr.it] match_arg (../../../../../../src/pm/i_hydra/libhydra/arg/hydra_arg.c:52): match handler returned error
[mpiexec@ui01.hpc.unipr.it] HYD_arg_parse_array (../../../../../../src/pm/i_hydra/libhydra/arg/hydra_arg.c:74): argument matching returned error
[mpiexec@ui01.hpc.unipr.it] mpiexec_get_parameters (../../../../../../src/pm/i_hydra/mpiexec/mpiexec_params.c:1283): error parsing input array
[mpiexec@ui01.hpc.unipr.it] main (../../../../../../src/pm/i_hydra/mpiexec/mpiexec.c:1745): error parsing parameters
```

Analizzato ed eseguito lo script hostname.slurm

hostname.slurm

```
#!/bin/bash
#SBATCH --output=%x.o%j          # Nome del file per lo standard output
#SBATCH --partition=cpu_guest     # Nome della partizione
#SBATCH --qos=cpu_guest          # Nome della partizione
#SBATCH --nodes=2                # numero di nodi richiesti
#SBATCH --ntasks-per-node=2      # numero di cpu per nodo
##SBATCH --mem=2G                # massima memoria utilizzata
#SBATCH --time=0-00:10:00        # massimo tempo di calcolo

module purge
#module load intel impi
#module load gnu openmpi
module load gnu7 openmpi3
#module load gnu8 openmpi4

echo "#SLURM_JOB_NODELIST      : $SLURM_JOB_NODELIST"
echo "#SLURM_JOB_CPUS_PER_NODE  : $SLURM_JOB_CPUS_PER_NODE"

mpirun hostname
```

Output

```
[martina.genovese@ui01 mpirun]$ cat hostname.slurm.o2189940
#SLURM_JOB_NODELIST      : wn[21-22]
#SLURM_JOB_CPUS_PER_NODE : 2(x2)
wn22
wn21
wn22
wn21
```

In particolare lo script in questione:

1. Rimuove tutti i moduli attualmente caricati dall'ambiente tramite "module purge"
2. carica la versione 7 del compilatore GNU e la versione 3 di OpenMPI
3. stampa le variabili di ambiente che contengono le risorse assegnate da Slurm:
 - \$SLURM_JOB_NODELIST: elenco dei nodi assegnati da Slurm à wn[21-22]
 - \$SLURM_JOB_CPUS_PER_NODE: numero di cpu assegnate per nodo à 2(x2)

mpicc

"mpicc" è un wrapper per il compilatore C che semplifica il processo di compilazione di programmi MPI scritti in C. In altre parole, include automaticamente le librerie necessarie e imposta le opzioni del compilatore affinché il programma possa utilizzare le funzionalità di MPI.

Compilazione ed esecuzione di mpi_hello.c

Link: https://www.hpc.unipr.it/dokuwiki/doku.php?id=calcoloscientifico:userguide#mpi_jobs_on_cpu

Ho creato la directory di lavoro "mkdir -p ~/HPC2324/mpi/mpicc/" e copiato all'interno l'esercizio fornito nella guida tramite "cp /hpc/share/samples/mpi/mpi_hello.c ."

Dopodichè ho creato lo script mpi_hello.slurm per compilare ed eseguire il programma mpi_hello.c.

In sintesi, nel programma ogni processo stampa un messaggio di saluto che include il proprio ID (taskid) e il nome del processore (hostname); nel caso il processo è il master (ID 0) allora stampa anche il numero totale di processi.

Analisi del programma:

- MPI_Init(&argc, &argv): inizializza l'ambiente MPI.
- MPI_Comm_size(MPI_COMM_WORLD, &numtasks): ottiene il numero totale di processi nell'intercommunicator MPI_COMM_WORLD e lo memorizza in numtasks.
- MPI_Comm_rank(MPI_COMM_WORLD, &taskid): ottiene l'ID del processo corrente all'interno dell'intercommunicator MPI_COMM_WORLD e lo memorizza in taskid.
- MPI_Get_processor_name(hostname, &len): ottiene il nome del processore su cui sta girando il processo corrente e memorizza il nome in hostname e la lunghezza in len.
- printf("Hello from task %d on %s!\n", taskid, hostname): stampa un messaggio da parte di ogni processo che include il proprio ID (taskid) e il nome del processore (hostname).
- printf("MASTER: Number of MPI tasks is: %d ", numtasks): se taskid è uguale a MASTER (cioè se il processo corrente è il master, con ID=0), stampa anche il numero totale di task.
- MPI_Finalize(): termina l'ambiente MPI, liberando le risorse allocate.

mpi_hello.c

```
1 /*****
2  * FILE: mpi_hello.c
3  * DESCRIPTION:
4  *   MPI tutorial example code: Simple hello world program
5  * AUTHOR: Blaise Barney
6  * LAST REVISED: 03/05/10
7  *****/
8 #include "mpi.h"
9 #include <stdio.h>
10 #include <stdlib.h>
11 #define MASTER 0
12
13 int main (int argc, char *argv[])
14 {
15     int numtasks, taskid, len;
16     char hostname[MPI_MAX_PROCESSOR_NAME];
17
18     MPI_Init(&argc, &argv);
19     MPI_Comm_size(MPI_COMM_WORLD, &numtasks);
20     MPI_Comm_rank(MPI_COMM_WORLD, &taskid);
21     MPI_Get_processor_name(hostname, &len);
22     printf ("Hello from task %d on %s!\n", taskid, hostname);
23     if (taskid == MASTER)
24         printf("MASTER: Number of MPI tasks is: %d\n", numtasks);
25     MPI_Finalize();
26 }
27
```

mpi_hello.slurm

```
#!/bin/bash

#SBATCH --partition=cpu_guest
#SBATCH --qos=cpu_guest
#SBATCH --job-name="mpi_hello-GNU_compiler"
#SBATCH --output=%x.o%j
##SBATCH --error=%x.e%j #If error is not specified stderr is redirected to stdout
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4

## Uncomment the following line if you need an amount of memory other than default (512MB)
##SBATCH --mem=2G

## Uncomment the following line if your job needs a wall clock time other than default (1 hour)
## Please note that priority of queued job decreases as requested time increases
##SBATCH --time=0-00:30:00

## Uncomment the following line if you want to use an account other than your default account ( $
##SBATCH --account=<account>

echo "# SLURM_JOB_NODELIST      : $SLURM_JOB_NODELIST"
echo "# SLURM_CPUS_PER_TASK     : $SLURM_CPUS_PER_TASK"
echo "# SLURM_JOB_CPUS_PER_NODE: $SLURM_JOB_CPUS_PER_NODE"

# Gnu compiler
module load gnu openmpi
mpicc mpi_hello.c -o mpi_hello
mpirun mpi_hello

# Uncomment the following lines to compile and run using the INTEL compiler
#module load intel intelmpi
#mpicc mpi_mm.c -o mpi_mm_intel
#mpirun mpi_hello_intel

# Uncomment the following lines to compile and run using the PGI compiler
#module load pgi/2018 openmpi/2.1.2/2018
#mpicc mpi_hello.c -o mpi_hello_pgi
#mpirun --mca mpi_cuda_support 0 mpi_hello_pgi
```

Output

```
[martina.genovese@ui01 mpicc]$ cat mpi_hello-GNU_compiler.o2189943
# SLURM_JOB_NODELIST      : wn[21-22]
# SLURM_CPUS_PER_TASK     :
# SLURM_JOB_CPUS_PER_NODE: 4(x2)
Lmod has detected the following error: You can only have one compiler module
loaded at a time.
You already have intel loaded.
To correct the situation, please execute the following command:

$ module swap intel gnu/5.4.0

While processing the following module(s):
  Module fullname  Module Filename
  -----
  gnu/5.4.0        /opt/ohpc/pub/modulefiles/gnu/5.4.0

Hello from task 4 on wn22!
Hello from task 5 on wn22!
Hello from task 6 on wn22!
Hello from task 7 on wn22!
Hello from task 0 on wn21!
MASTER: Number of MPI tasks is: 8
Hello from task 1 on wn21!
Hello from task 2 on wn21!
Hello from task 3 on wn21!
[martina.genovese@ui01 mpicc]$
```