UL HPC School 2017
PS5: HPC workflow with MPI
Parallel/Distributed jobs (OSU Microbenchmarks, HPL)

UL High Performance Computing (HPC) Team
Sebastien Varrette

University of Luxembourg (UL), Luxembourg
http://hpc.uni.lu
Latest versions available on Github:

UL HPC tutorials:  
https://github.com/ULHPC/tutorials

UL HPC School:  
http://hpc.uni.lu/hpc-school/

PS5 tutorial sources:  
https://github.com/ULHPC/tutorials/tree/devel/advanced/OSU_MicroBenchmarks
1 Introduction

2 OSU Micro-Benchmarks

3 High-Performance Linpack (HPL)
Introduction

Main Objectives of this Session

- See how to use the MPI suit available on the UL HPC platform:
  - Intel MPI and the Intel MKL
  - OpenMPI
  - MVAPICH2
  - MPI-3 over OpenFabrics-IB, Omni-Path, OpenFabrics-iWARP, PSM, and TCP/IP

- Build and run MPI code (through the provided launcher scripts)
- Test case on reference parallel MPI benchmarks:
  - OSU micro-benchmarks:
    - measure the performances of various MPI operations
  - High-Performance Linpack (HPL)
OSU Micro-Benchmarks

Summary

1. Introduction

2. OSU Micro-Benchmarks

3. High-Performance Linpack (HPL)
Pre-requisites: get an interactive job for compilation

Question: what is the interest of requesting multiple cores?

```bash
### Iris cluster
(access)\$> si -n 14
# iris (long version)
(access)\$> srun -p interactive --qos qos-iteractive -n 14 --pty bash
# iris (long version, best-effort mode)
(access)\$> srun -p interactive --qos qos-besteffort -n 14 --pty bash

### On gaia, chaos
(access)\$> oarsub -I -l enclosure=1/nodes=1,walltime=4
```
OSU Micro-Benchmarks

We will build version 5.3.2 of the OSU micro-benchmarks
Focusing on (only) two one-sided benchmarks:
  → osu_get_latency - Latency Test
  → osu_get_bw - Bandwidth Test

Pre-requisites:
  → clone ULHPC/tutorials and ULHPC/launcher-scripts repositories
  → Preparing your working directory

```bash
$ mkdir -p ~/git/ULHPC && cd ~/git/ULHPC
$ git clone https://github.com/ULHPC/launcher-scripts.git
$ git clone https://github.com/ULHPC/tutorials.git
# Preparing your working directory
$ mkdir -p ~/tutorials/OSU-MicroBenchmarks
$ cd ~/tutorials/OSU-MicroBenchmarks
# Keep a symlink to the reference tutorial
$ ln -s ~/git/ULHPC/tutorials/advanced/OSU_MicroBenchmarks ref.ulhpc.d
```
Building the Benchmarks

Your Turn!

- Get the sources
- Uncompress them
- Compilation based on the Intel MPI suit
- Compilation based on the Open MPI suit
- Compilation based on the Open MPI suit over Ethernet interface
  - highlight performance drops compared to Infiniband
Running the Benchmarks

Your Turn!

- Build directory:
  libexec/osu-micro-benchmarks/mpi/one-sided/
- Prepare a batch launcher
  - copy and adapt the default SLURM launcher
- Run it in batch mode

```bash
$ cd ~/tutorials/OSU-MicroBenchmarks/runs
### On iris
$ sbatch ./launcher-OSU.intel.sh osu_get_bw
$ sbatch ./launcher-OSU.intel.sh osu_get_latency
### On gaia, chaos
$ oarsub -S ./launcher-OSU.intel.sh
```
Based on OSU Micro-benchmarks

OSU One Sided MPI Get latency Test v5.3.2

- Latency (µs) - LOGSCALE
- Packet size (bits) - LOGSCALE

OpenMPI (Ethernet only)
- OpenMPI
- Intel MPI

Sebastien Varrette (University of Luxembourg)
Interconnect Performances

Based on OSU Micro-benchmarks

![Graph showing OSU MPI One Sided MPI Get Bandwidth Test v5.3.2](image)

- IB EDR Theoretical Max
- Intel MPI
- OpenMPI
- OpenMPI (Ethernet on 10GbE only)
Summary

1. Introduction

2. OSU Micro-Benchmarks

3. High-Performance Linpack (HPL)
High-Performance Linpack (HPL)

Pre-requisites: get an interactive job for compilation

Question: what is the interest of requesting multiple cores?

### Iris cluster

(access)$> si -n 14

# iris (long version)

(access)$> srun -p interactive --qos qos-iteractive -n 14 --pty bash

# iris (long version, best-effort mode)

(access)$> srun -p interactive --qos qos-besteffort -n 14 --pty bash

### On gaia, chaos

(access)$> oarsub -I -l enclosure=1/nodes=1,walltime=4
High-Performance Linpack (HPL)

Portable implem. of High-Performance Linpack (HPL) Benchmark

→ for Distributed-Memory Computers, ref. benchmark for Top500

We will build **version 2.2 of the HPL**

→ Focusing (only) on Intel MPI+MKL build

**Pre-requisites:**

→ clone ULHPC/tutorials and ULHPC/launcher-scripts repositories

→ Preparing your working directory

```bash
$ mkdir -p ~/git/ULHPC && cd ~/git/ULHPC
$ git clone https://github.com/ULHPC/launcher-scripts.git
$ git clone https://github.com/ULHPC/tutorials.git

# Preparing your working directory
$ mkdir -p ~/tutorials/HPL
$ cd ~/tutorials/HPL

# Keep a symlink to the reference tutorial
$ ln -s ~/git/ULHPC/tutorials/advanced/HPL ref.ulhpc.d
```
Building HPL

Your Turn!

- Get the sources
- Uncompress them
- Compilation based on the Intel MPI suit
  - Prepare and adapt src/hpl-2.2/Make.intel64
- Compile it!

```
$ cd ~/tutorials/HPL/src/hpl-2.2
$ cp setup/Make.Linux_Intel64 Make.intel64
$ vim Make.intel64
# [...] change TOPdir and MP{dir,inc,lib} (at least)
$ make arch=intel64 clean_arch_all
$ make arch=intel64
```
Preparing the HPL Benchmark Run

Your Turn!

- Build directory: `bin/intel64`
- Prepare a batch launcher
  - copy and adapt the default SLURM launcher
- Prepare an input `HPL.dat` file
  - use Tuning HPC Online for some default settings

Main HPL parameters constraints

- `PxQ = <nodes>*<cores> = $SLURM_NTASKS`
- Problem size: `N` (to be as large as possible)
  - `N = α√#nodes * RAM * 1024` where RAM is expressed in GiB
- `NB`: depends on processor architecture (Ex: Intel MKL notes)
  - `NB = 192` on iris cluster
Example HPL.dat

HPLinpack benchmark input file
Innovative Computing Laboratory, University of Tennessee
HPL.out output file name (if any)
6 device out (6=stdout, 7=stderr, file)
1 # of problems sizes (N)
24650 Ns
1 # of NBs
192 NBs
0 PMAP process mapping (0=Row-, 1=Column-major)
2 # of process grids (P x Q)
2 4 Ps
14 7 Qs

Targeting 1 node in this case on 2 sets of parameters (P x Q = 28)

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>NB</th>
<th>P</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run 1</td>
<td>24650</td>
<td>192</td>
<td>2</td>
<td>14</td>
</tr>
<tr>
<td>Run 2</td>
<td>24650</td>
<td>192</td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>
Adapt the default SLURM launcher
Run it

```bash
$ cd ~/tutorials/HPL/runs
$ cp ../ref.ulhpc.d/HPL.dat .
```

### On iris
```bash
$ sbatch ./launcher-HPL.intel.sh
```

### On gaia, chaos
```bash
$ oarsub -S ./launcher-HPL.intel.sh
```

Grab the HPL results from the output logs

```bash
$ grep WR slurm-2758.out
```

<table>
<thead>
<tr>
<th>#</th>
<th>T/V</th>
<th>N</th>
<th>NB</th>
<th>P</th>
<th>Q</th>
<th>Time</th>
<th>Gflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WR11C2R4</td>
<td>24650</td>
<td>192</td>
<td>2</td>
<td>14</td>
<td>13.51</td>
<td>7.392e+02</td>
</tr>
<tr>
<td>2</td>
<td>WR11C2R4</td>
<td>24650</td>
<td>192</td>
<td>4</td>
<td>7</td>
<td>12.69</td>
<td>7.869e+02</td>
</tr>
</tbody>
</table>
Based on High-Performance Linpack (HPL)

reference benchmark for Top 500

HPL 2.2 - ULHPC iris cluster - 100 Nodes

$R_{\text{max}} = 78.47$ TFlops

$R_{\text{peak}} = 107.52$ TFlops

$N = 1169500$

$N = 1182490$

$N = 1156500$

$N = 1156500$

$N = 1091530$
The UL High Performance Computing (HPC) Team
University of Luxembourg, Belval Campus:
Maison du Nombre, 4th floor
2, avenue de l’Université
L-4365 Esch-sur-Alzette
mail: hpc@uni.lu

1 Introduction

2 OSU Micro-Benchmarks

3 High-Performance Linpack (HPL)