Programming of Supercomputers

Assignment 1:

Single Node Performance

Prof. Michael Gerndt, Madhura Kumaraswamy

Technische Universität München

Informatik 10: Lehrstuhl für Rechnerarchitektur & Parallele Systeme

26.10.2018

Single-Thread Performance

- Gprof
  - Flat Profile
  - Call graph
- Compiler flags
  - GCC
  - ICC
- Optimization pragmas
  - GCC pragmas
  - ICC pragmas

Multi-Thread Performance

- OpenMP
  - Single process
  - Scaling with threads
    - Shared address space
    - Direct loads and stores
    - Coherency, locks
- MPI
  - Multiple processes
  - Only certain process counts valid
  - Scaling with processes
    - Separate address space
    - Messages
- MPI + OpenMP
  - Hybrid

SuperMUC

https://www.welt.de/wissenschaft/article143276645/SuperMUC-beeindruckt-nun-mit-doppelter-Leistung.html

Login to SuperMUC, Documentation

- First change the standard password
  - https://idportal.lrz.de/r/entry.pl
- Login via
  - lxhalle due to restriction on connecting machines
  - First ssh <TUMuserID>@lxhalle.in.tum.de, then ssh <userid>@hw.supermuc.lrz.de for Phase 2 nodes (Haswell)
  - For Assignment 1, login to Phase 1 fat nodes (Westmere): ssh <userid>@wm.supermuc.lrz.de
- You can't simply copy external files to SuperMUC!
  - Use SFTP or scp to transfer files from your localhost to SuperMUC and vice-versa
- Documentation
  - http://www.lrz.de/services/compute/supermuc/
  - http://www.lrz.de/services/compute/supermuc/loadleveler/

Building the Benchmark

- List all modules available to be loaded:
  ```
  module avail
  ```
- Load the required modules:
  ```
  module unload mpi.ibm
  module load mpi.intel
  ```
- Update your Makefile (refer to the provided instructions)
  - Serial
  - OpenMP
  - MPI
  - Hybrid
- Build and verify that the binaries were created
- Run the benchmark in the login node
- Identify your performance metric from the output!
  - More is better or less is better?
  - Check the benchmark's documentation online

Batch Scripts

- Advantages
  - Reproducible performance
  - Run larger and longer running jobs

- Several job classes available:
  - **fattest** (*recommended for this assignment‘s tasks*)
    - Phase 1:
      - Max 1 island, max 4 nodes, max 2 hours, 1 job in queue
  - **fat**
    - Phase 1:
      - Max 1 island, max 52 nodes, max 48 hours, 8 jobs in queue
  - **test** (*recommended for this assignment‘s tasks*)
    - Phase 2:
      - Max 1 island, 20 nodes, 30 minutes, 1 job in queue
  - **micro**
    - Phase 2:
      - Max 1 island, 20 nodes, 48 hours, 8 jobs in queue

Submitting a Batch Job

- llsubmit ll.sh
  - Submission to batch system
- llq –u $USER
  - Check status of own jobs
- llcancel <jobid>
  - Kill job if no longer needed
  - Obtain the <jobid> from the llq output

Example OpenMP Batch Jobs

Phase 1 fat nodes. Also see: https://www.lrz.de/services/compute/supermuc/loadleveler/examples_fat_nodes/

```
#!/bin/bash
#@ wall_clock_limit = 00:20:00
#@ job_name = pos-lulesh-openmp
#@ job_type = MPICH
#@ class = fattest
#@ output = pos_lulesh_openmp_$(jobid).out
#@ error = pos_lulesh_openmp_$(jobid).out
#@ node = 1
#@ total_tasks = 40
#@ node_usage = not_shared
#@ energy_policy_tag = lulesh
#@ minimize_time_to_solution = yes
#@ island_count = 1
#@ queue

. /etc/profile
. /etc/profile.d/modules.sh

export OMP_NUM_THREADS=40
./lulesh2.0
```

Phase 2 Haswell nodes. Also see: https://www.lrz.de/services/compute/supermuc/loadleveler/examples_haswell_nodes/

```
#!/bin/bash
#@ wall_clock_limit = 00:20:00
#@ job_name = pos-lulesh-openmp
#@ job_type = MPICH
#@ class = test
#@ output = pos_lulesh_openmp_$(jobid).out
#@ error = pos_lulesh_openmp_$(jobid).out
#@ node = 1
#@ total_tasks = 28
#@ node_usage = not_shared
#@ energy_policy_tag = lulesh
#@ minimize_time_to_solution = yes
#@ island_count = 1
#@ queue

. /etc/profile
. /etc/profile.d/modules.sh

export OMP_NUM_THREADS=28
./lulesh2.0
```

Use CPU hours responsibly!

- Specify job execution as tight as possible
  - For this assignment, 15 minutes are sufficient
- Only request the number of nodes required
  - 1 node is sufficient for all tasks in assignment 1
- Small tests can be done in the login node
  - Create a batch only after you are ready to collect results
  - Running in a batch eliminates interference from other users