SC23 DENVER NOV 12–17

Overview of the SCC23 Benchmarks

Amiya K Maji

Purdue University

Benchmark List

Required

- HPL
- HPCG
- MLPerf Inference

Optional (Bonus)

- Stream
- OSU microbenchmarks

HPL

- One of the most popular benchmarks in the HPC world
- Solves a (random) dense linear system in double precision
  - https://www.netlib.org/benchmark/hpl/index.html
- Used to measure “performance” of a computer or a cluster
- Output: No. of FLOP/s
- TOP-500 listing of world’s fastest supercomputers use HPL

HPL

- Internally uses BLAS libraries for LA subroutines
  - Intel MKL
  - OpenBLAS
- Uses MPI for distributed memory parallelism
- Uses OpenMP for shared memory parallelism
- Guarantees error bound on the results
- Can take variable amount of time based on problem size
  - Size of the matrix
- Problem size is typically determined by the memory (RAM) size

Building HPL

- Download and untar the HPL source tarball
    - tar -xf hpl-2.3.tar.gz
- Need to load the necessary libraries for BLAS routines
- Verify that you have all the dependencies (compiler, MPI, BLAS)
- Now compile HPL
    - ./configure --prefix= ...
    - make -j16 && make install
- Add the binary location to your PATH
    - export PATH=/path/to/xhpl/binary:$PATH
- Time to run HPL

HPL Input File (HPL.dat)

- Copy HPL.dat to your working directory
- Edit the input file to reflect your setup
- Important parameters
  - N #Array size
  - NB #Block size for LA operations
  - P #Factorization rows
  - Q #Factorization columns. PxQ must equal your MPI processes
  - Change # of algorithms to test
- Now run HPL
  - mpirun -np 2 xhpl

HPL Tuning Guide

- Typically, HPL should occupy 80-90% of your memory for optimal performance
- NB (Block size) impacts performance
    - Try different NB sizes (128-512)
    - Empirically find out which one is better
- Exact layout of MPI processes/OpenMP threads impact performance
    - Optimal layout depends on the processor architecture
-

HPL Resources

- https://www.netlib.org/benchmark/hpl/faqs.html
- https://frobnitzem.github.io/hpl-hpcg/
- https://www.advancedclustering.com/act_kb/tune-hpl-dat-file/
- https://ulhpc-tutorials.readthedocs.io/en/latest/parallel/mpi/HPL/
- https://developer.amd.com/spack/hpl-benchmark/

HPCG Benchmark

- High-performance Conjugate Gradient
  - Create a new benchmark for ranking HPC systems
  - Uses challenging patterns of execution, memory access, and communication
- Download HPCG source code from
  - https://www.hpcg-benchmark.org/software/
-

Building and running HPCG

- Dependencies:
  - Need a compiler and MPI libraries
- Can build HPCG with both MPI and OpenMP support
- Rules for submitting HPCG results
  - Must occupy 25% of main memory or higher
  - Must run for at least 30 minutes
-

Configuring and tuning HPCG

- HPCG reads inputs from HPCG.dat file
  - You can specify size of the 3D array
  - You can specify run time
- Try different compilers and MPI libraries
- Try different MPI/OpenMP binding options

- Results obtained from HPCG is typically orders of magnitude lower than HPL
  - “Lower bound”

HPCG Resources

- https://www.hpcg-benchmark.org/software/
- https://www.hpcg-benchmark.org/
- https://ulhpc-tutorials.readthedocs.io/en/latest/parallel/hybrid/HPCG/

MLPerf Inference

- Measure how fast systems can run models in different deployments
- Uses MLCommons cm automation framework to automatically configure and run benchmarks
- Follow SCC22 instructions for MLPerf
  - https://studentclustercompetition.us/2022/Instructions/mlperf.pdf
  - Object detection with retinanet model
  - Openimages dataset
- Updated instructions will be shared later

MLPerf Inference Resources

- https://github.com/mlcommons/ck/blob/master/docs/tutorials/sc22-scc-mlperf.md
- https://github.com/mlcommons/inference
- https://www.nvidia.com/en-us/data-center/resources/mlperf-benchmarks/

Stream

- Benchmark to measure memory bandwidth on a single node
  - How fast can I read data from main memory?
- Memory bandwidth is a key factor for good performance
  - Memory hierarchies and access latencies
  - Hardware Cache vs. main memory
  - How fast can you feed data to the processor
  - Why GPUs have such high memory bandwidth
- You can run stream on a single core or multiple cores
- Uses OpenMP while running on multiple cores

Important things to remember

- The array size must be 2x the cache size or larger
- Binding of threads to physical cores can impact performance
- Choice of compiler can also make an impact
  - Try with gcc
  - Try with intel
  - See the difference
-

Running stream

- export OMP_NUM_THREADS=128
- ./stream_c.exe
- ./stream.icc

- If you have heterogeneous hardware
    - Only submit stream results from the compute node with best performance

STREAM Resources

- https://www.cs.virginia.edu/stream/
- https://www.intel.com/content/www/us/en/developer/articles/technical/optimizing-memory-bandwidth-on-stream-triad.html

OSU Microbenchmarks

- The OSU benchmarks measure performance of various MPI operations
  - How good is your network
  - How good is your MPI library
- Three primary types of operations
  - Point-to-point operations
  - Collective operations
  - One-sided operations
- We will focus on point-to-point performance
  - Latency
  - Bandwidth

Building and running OSU benchmarks

- Download link
  - https://mvapich.cse.ohio-state.edu/download/mvapich/osu-micro-benchmarks-7.2.tar.gz
- Dependencies: Compiler and MPI libraries
- configure and make
- Make sure that the path to binaries is added to your $PATH
- Run with mpirun/mpiexec
  - mpirun -np 2 osu_latency
- Make sure that MPI ranks are actually distributed across both nodes
- If you have heterogeneous compute nodes
  - Identify two compute nodes that have identical or closest specs
  - Run OSU benchmarks across these nodes

OSU Benchmark resources

- https://mvapich.cse.ohio-state.edu/benchmarks/
- https://ulhpc-tutorials.readthedocs.io/en/latest/parallel/mpi/OSU_MicroBenchmarks/
- https://hpcadvisorycouncil.atlassian.net/wiki/spaces/HPCWORKS/pages/1284538459/OSU+Benchmark+Tuning+for+2nd+Gen+AMD+EPYC+using+HDR+InfiniBand+over+HPC-X+MPI

Benchmarking Notes

- Detailed submission instructions will be shared later
- Follow submission instructions carefully

- Familiarize yourself with the benchmarks ahead of time
- Write scripts and automate
- Teams can use Spack/Easybuild to build the benchmark applications
  - https://spack.readthedocs.io/en/latest/

Questions