aasish-kumar.sharma@gwdg.de

Aasish Kumar Sharma

HPC Benchmarking

Practical Course on High Performance Computing (PCHPC)

Table of contents

Background

To this point you learned about

- What is High Performance Computing (HPC) system & what is HPC workload.
- What kind of tools and techniques you need to make such a system, and
- How you can configure these tools and techniques to assemble the system.

Now, tell me how much workload your system can handle?

- How do you benchmark your HPC system?

Objectives

To understand,

- To understand basics about Benchmark and benchmarking.
- To understand types of benchmarking and why they are necessary.
- To understand how benchmarking is done in HPC use case.

Basic Concepts: Benchmark & Benchmarking

Dictionary meaning

- Benchmark (noun)
  - A standard or point of reference against which things can be compared.
- Benchmarking (verb)
  - A process of comparing with a previously defined standards (benchmarks).

HPC Benchmarking

- Is related to measuring HPC system performance based on unit workload.
- Is related to system's computation, memory, I/O, and network standards.
- Mainly, to identify bottlenecks in computation, memory, I/O, and network.

Source: https://languages.oup.com/google-dictionary-en

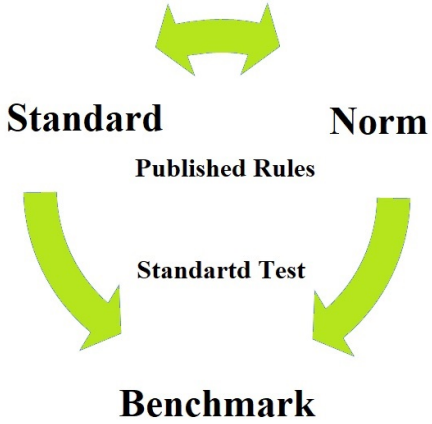
What is a Standard?

Standards!

- Standards are published rules.

Logically,

- We need Standards for Benchmarking, &
- Prerequisites for any standards are,
  - It should be technically mature, and
  - It should be beneficial for the users.



More on standards: https://de.wikipedia.org/wiki/Standard

Alert! Certified Standards Does Not Certify Benchmarking

Standard Certification

- Assures standard processes are followed.
- It is a published specification ratified by some organization.
- e.g., ISO 9001 - is an international standard for quality management.

Benchmarking

- Assures comparative results against standard tests are achieved.
- It is often established by general organizational acceptance.
- e.g., IO 500 Benchmark - is a comparison against standard tests.

Images source: https://www.iso.org/modules/isoorg-template/img/iso/iso-logo-print.gif, https://www.vi4io.org/io500/start, https://www.top500.org/news/chinas-tianhe-2-supercomputer-retains-top-spot-on-43rd-edition-of-the-top500-list/

Types of HPC Benchmarking and Key Metrics

Micro Benchmarking - Baseline performance

- Benchmark performance improvement against **unit** of system/workload.

Macro Benchmarking - Scaling

- Benchmark performance scaling the number of unit of system/workload.

Overlall - Performance

- It is a system performance measure running a unit workload & scaling them.

What is Measured?

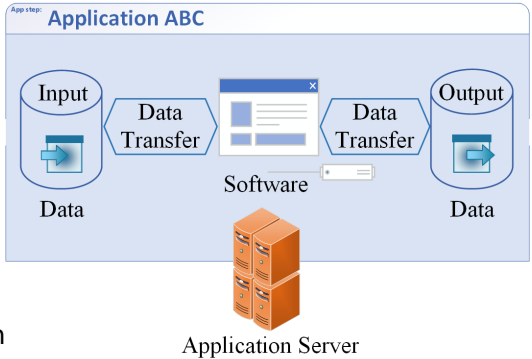
- *Timing*
  - A measure of full or partial run-time. Mainly, wall clock time.
- *Throughput*: (including parallel efficiency)
  - The ratio of measured scaling to the perfect scaling.

Source: Fleming and Wallace, “How Not to Lie with Statistics”

HPC Benchmarking

HPC Benchmarking Includes

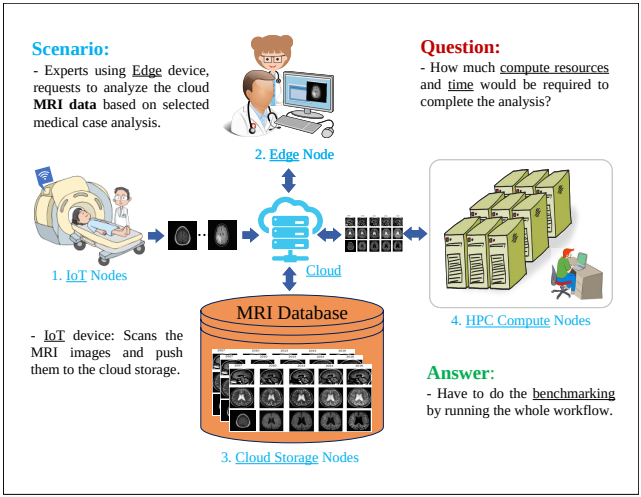
- **System Benchmark**:
  - Partition → Cluster → Nodes
- **Workload Benchmark**:
  - Workflow of HPC Applications:
    - App1 → App2 → App3 → ...



HPC Workload means:

- Application running on a HPC system
- Consumes system resources,
- Considering usages throughout the system,
- Consider Input/Output data and latency.

Scenario: MRI Use Case



Why Benchmarking is Necessary in HPC?

- To *calibrate* and *standardize* the system's and the workload's:
  - **Performance**, **Capacity**, **Stability**, & **Reliability**
  - e.g., Whenever there are concerns on performance, answer is benchmarking.
- Helps to identify bottlenecks (CPU, Memory, Storage, Network)
  - With different in/out parameters, or
  - Mainly, to find out the limits,
- To analyze different hardware/software configurations,
  - While scaling the system/software.
  - Mainly, while optimizing resource utilization.

Tools and Techniques for Benchmarking HPC Resources

Example: Performance Benchmarking

- *Compute Performance*: Execution time, CPU/GPU utilization.
- *Memory Performance*: Peak memory usage, swap behavior.
- *Storage I/O Performance*: Read/write speeds, file system latency.
- *Network Performance*: Data transfer speed, latency.

Compute Performance:

- `time` - Time package to measures execution time.
- `likwid-perfctr` - CPU FLOPS analysis.
- `nvidia-smi` - GPU utilization monitoring.

Memory Performance:

- `/usr/bin/time -v` - Tracks peak memory usage.
- `htop` - Real-time memory monitoring.

Storage I/O Performance:

- `dd` - **LinuxPerformanceTests**.
- `iostat` - Monitors disk activity.

Network Bandwidth:

- `iperf` - Tests network throughput.
- `rsync` - Benchmarks data transfer speed.

Macro Benchmarking - Scaling

- Measuring the performance by scaling the number of unit resources.
- Speedup Scalability in case of:
  - Hardware:
    - Is the ability to handle more workload by scaling computational resources.
    - e.g., adding more processors, memory, or storage.
  - Software:
    - Is the parallelization efficiency,
    - Which determines how well an application utilizes increased resources.
  - For example: For compute processors, its measures:
    - The ratio of the actual speedup and the ideal speedup for a number of processors.

$$Speedup(S) = T(1)/T(N) \longrightarrow T : time; N : no..of..processors. \quad (1)$$

Macro Benchmarking - Scaling Types

- Workload scalability is evaluated using two approaches:
  - **1. Strong Scaling**
    - The problem size is kept constant while the number of processors (N) is increased.
    - In Ideal case the runtime decreases with increase in processors.
    - Speedup is defined as:

$$S_{\text{strong}} = \frac{T_1}{T_N} \tag{2}$$

      where T_1 and T_N are the execution time with 1 and N number of processors.
    - **Amdahl's Law** (1967): places an upper bound on strong scaling efficiency:

$$S_{\text{Amdahl}} = \frac{1}{s + \frac{p}{N}} \tag{3}$$

      where s is the serial fraction and $p = 1 - s$ is the parallel fraction of the workload.

Macro Benchmarking - Scaling Types (Contd..)

- Workload scalability is evaluated using two approaches:
  - **2. Weak Scaling**
    - The problem size and the number of processors (N) are increased proportionally.
    - In Ideal case the runtime remains constant with growing workload and resources.
    - Speedup is defined as:

$$S_{\text{weak}} = \frac{T_N(P)}{T_1(P/N)} \tag{4}$$

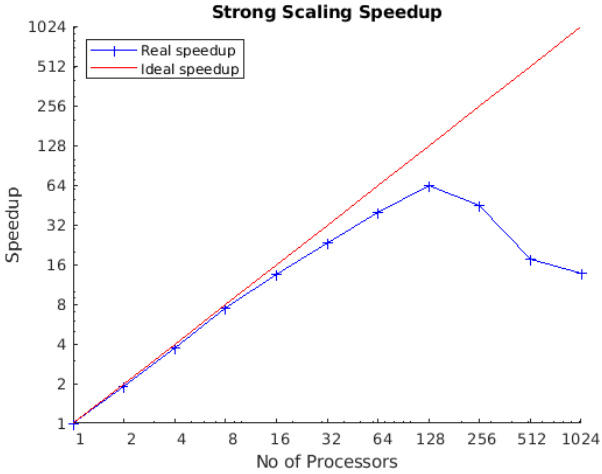
      where $T_N(P)$ is the execution time with N processors for problem size P, and $T_1(P/N)$ is the execution time for a scaled-down problem size per processor.
    - **Gustafson's Law** (1988): argues, weak scaling is more practical in real applications:

$$S_{\text{Gustafson}} = s + p \times N \tag{5}$$

      where s is the serial fraction and $p = 1 - s$ is the parallel fraction.

Macro Benchmarking: Strong Scaling Discussion

.



Scaling - HPC Wiki

In Summary

Macro Scaling Benchmark Guidelines

1. Measure using multiple system factors & independent runs per job size.
2. Considered following factors as well for multi-nodes:
   a) Interconnect speed & latency
   b) Max processors & memory per node
   c) System variables & restrictions (e.g. stack size)

Overall Guidelines:

- Should be alert and vigilante to the details,
- Should think critically to include all details,
- Use proper measures and charts to present,
- Should repeat benchmarking periodically.

Image: https://thefruitfultoolbox.com/dos-donts-disc/

References

Fleming, Philip J. and John J. Wallace. “How Not to Lie with Statistics: The Correct Way to Summarize Benchmark Results”. In: *Commun. ACM* 29.3 (Mar. 1986), pp. 218–221. ISSN: 0001-0782. DOI: 10.1145/5666.5673.

Scaling - HPC Wiki. URL: https://hpc-wiki.info/hpc/Scaling (visited on 04/18/2023).

Outline