1967-17

Advanced School in High Performance and GRID Computing

3 - 14 November 2008

How to benchmark your application

COZZINI Stefano

*CNR-INFM Democritos
c/o SISSA
via Beirut 2-4, 34014
Trieste
ITALY*

Advanced School in
High Performance
and GRID Computing

eLab
Joint DEMOCRITOS/SISSA
Laboratory for e-Science

The art of benchmarking

Stefano Cozzini

CNR-INFM DEMOCRITOS, Trieste

ICTP HPC School 2008 – Trieste, Italy - November 03-14, 2008

Agenda/ Aims

- Give you the feeling how much is important to know how your system/ application/computational experiment is performing..
- Name a few standard benchmarks that can help you in making/taking a decision
- Show you some tricks and tips how to make your own benchmarking procedure
- Stop in less than 30 minutes.

benchmark: a definition

a benchmark is the act of running a computer program, a set of programs, or other operations, in order to assess the relative performance of an object, normally by running a number of standard tests and trials against it

from wikipedia

three important notes:

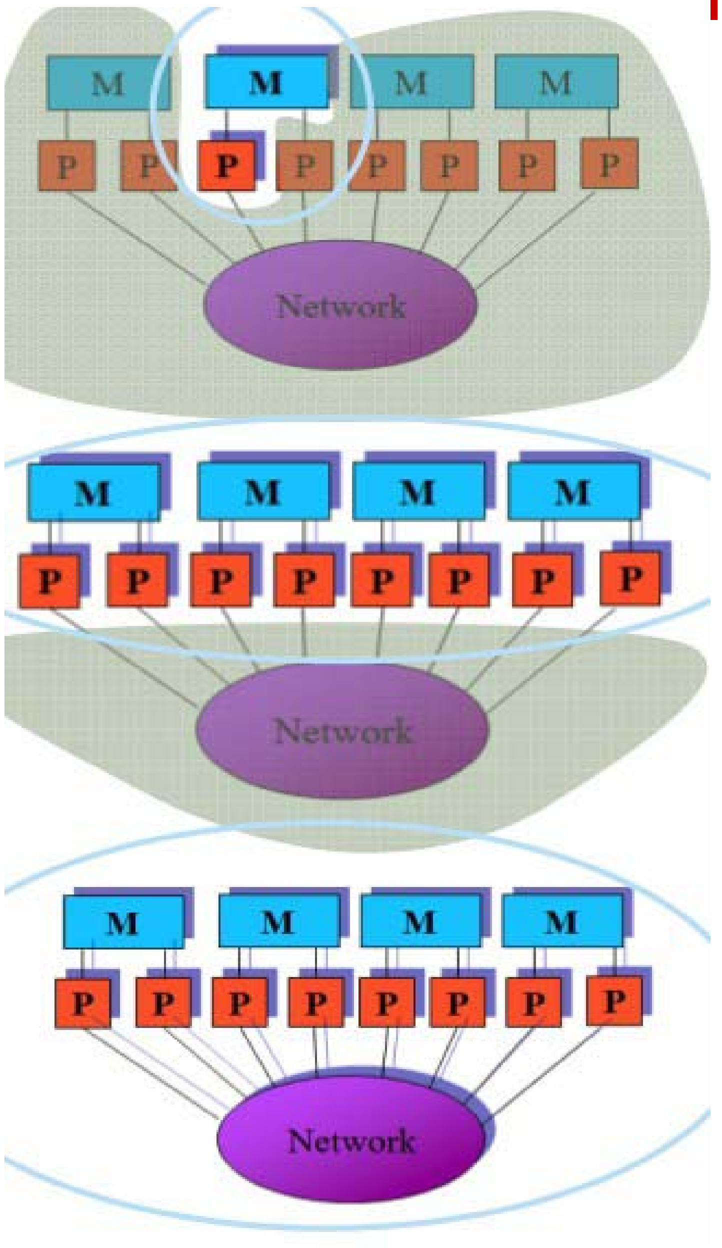
- no single number can reflect overall performance
- the only benchmark that matters is the intended workload.
- The purpose of benchmarking is not to get the best results, but to get consistent repeatable accurate results that are also the best results.

a few challenges in benchmarking:

- Benchmarking is not easy and often involves several iterative rounds in order to arrive at predictable, useful conclusions. Interpretation of benchmarking data is also extraordinarily difficult.
  - Vendors tend to tune their products specifically for industry-standard benchmarks. Use extreme caution in interpreting their results.
  - Many benchmarks focus entirely on the speed of computational performance, neglecting other important features of a computer system.
  - Benchmarks seldom measure real world performance of mixed workloads — running multiple applications concurrently in a full, multi-department environment

What we need to benchmark on a modern system

- Local: only a single processor (core) is performing computations.
- Embarrassingly Parallel -each processor (core) in the entire system is performing computations but they do no communicate with each other explicitly.
- Global -all processors in the system are performing computations and they explicitly communicate with each other.



Type of code for benchmark

- Synthetic codes
  - Basic hardware and system performance tests
  - Meant to determine expected future performance and serve as surrogate for workload not represented by application codes
  - useful for performance modeling
- Application codes
  - Actual application codes as determined by requirements and usage
  - Meant to indicate current performance
  - Each application code should have more than one real test case

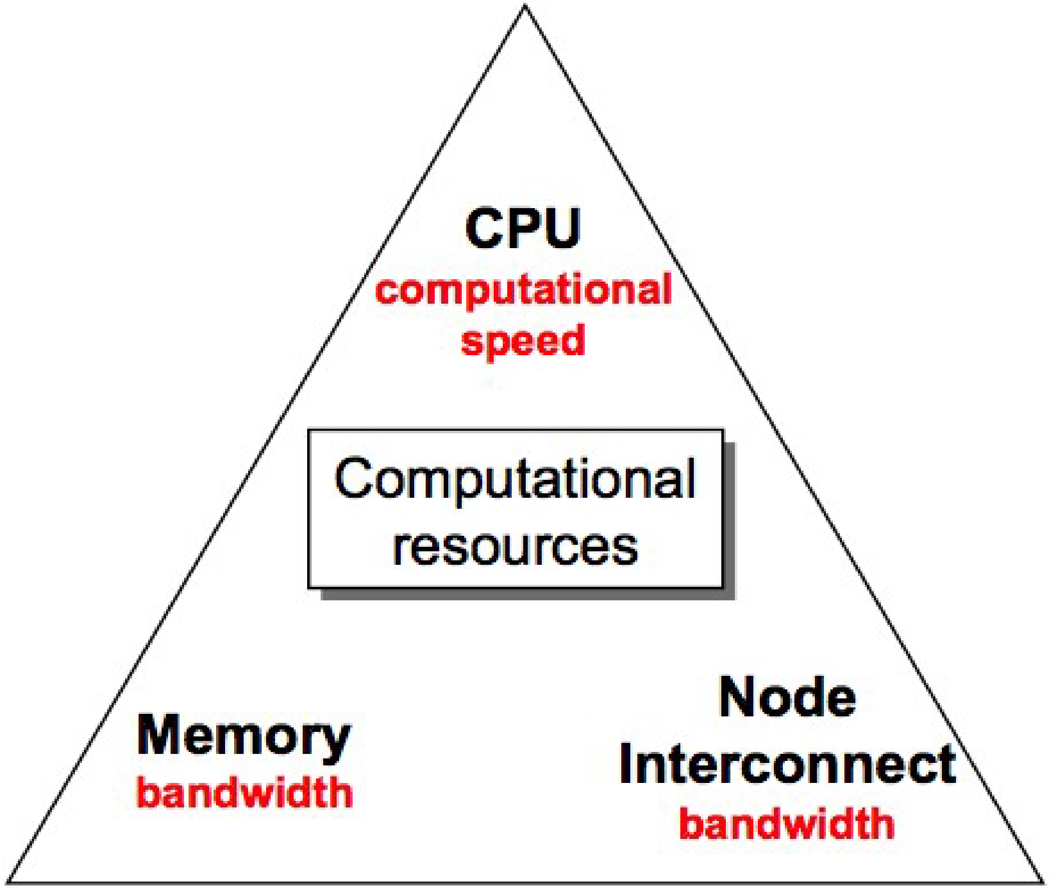
A very incomplete list of freely available benchmark:

- General benchmark:
  - HPL Linpack (for Top500)
  - HPC Challenge Benchmark:
    - a collection of basic benchmark beyond HPL
  - NAS benchmark suite
    - math kernel implemented both in MPI and openMP
- Network benchmark:
  - Netpipe /Netperf
    - tcp/ip protocol and more
  - IMB
    - MPI protocol
- I/O benchmarks: Iozone /bonnie etc..

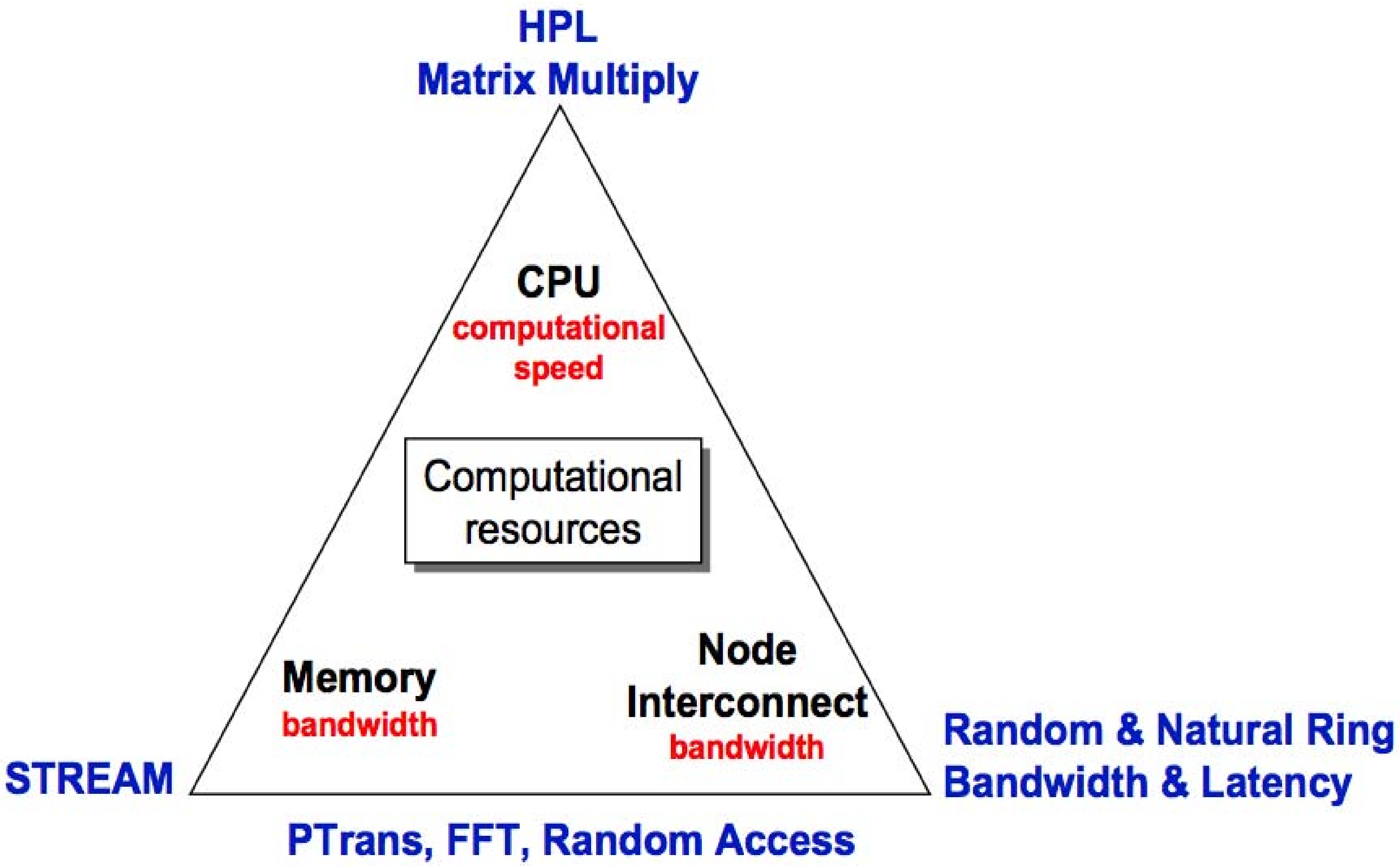
HPCC benchmark

- The HPC Challenge benchmark consists of basically 7 tests:
- 1. HPL - the Linpack TPP benchmark which measures the floating point rate of execution for solving a linear system of equations.
- 2. DGEMM - measures the floating point rate of execution of double precision real matrix-matrix multiplication.
- 3. STREAM - a simple synthetic benchmark program that measures sustainable memory bandwidth (in GB/s) and the corresponding computation rate for simple vector kernel.
- 4. PTRANS (parallel matrix transpose) - exercises the communications where pairs of processors communicate with each other simultaneously. It is a useful test of the total communications capacity of the network.
- 5. RandomAccess - measures the rate of integer random updates of memory (GUPS).
- 6. FFT - measures the floating point rate of execution of double precision complex one-dimensional Discrete Fourier Transform (DFT).
- 7. Communication bandwidth and latency - a set of tests to measure latency and bandwidth of a number of simultaneous communication patterns; based on b_eff (effective bandwidth benchmark).

Computational resources to benchmark



HPCC components



Remember:

- THERE IS NO BENCHMARK THAT SUBSTITUTES your own code on your dataset
- Measurement should be done by you on your code !

a few tips to benchmark your application. (1)

- use /usr/bin/time and take note of all times
  - wall time/ user time /sys time
- repeat the same run at least a few time to estimate the fluctuations of the numbers (this should be generally within a few percent)
- be sure to be alone on the system you are using and with no major perturbation on your cluster
-

a few tips to benchmark your application. (2)

- execution runs should be at least in the order of tens of minutes
- always check the correctness of your scientific output
- be sure to be alone on the system you are using and with no major perturbation on your cluster
-

Finally..

- Did I mention you need not to trust vendors ?
- Did I mention you need to use your application ?