Benchmarks for the TOP500

HPCG Meeting, Washington, DC

Outline

- TOP500
  - Overview, current state
- Why Linpack in the First Place
  - Why it worked so well
- Problems with Linpack
- How to Fix the Ranking?
  - Approaches
  - The “Stunt-Machine” Test
- 3 Criteria for a Useful New Benchmark

TOP500

- We have published twice a year since June 1993
- Linpack (HPL) is so far the only official benchmark
- Small set of rules, which need occasional tweaking
- Hans recently passed away, but Prometheus will continue web management and publication of the list

Why Linpack in the First Place?

- The idea for a list ranking supercomputers was first discussed between Hans and myself in 1991
  - We wanted to improve his previous statistics based on # of certain systems – Vector Mainframes
  - We thought it would be impractical
  - We tried to use processor count for 2 years (91/92), did not 'work' either
- Started working on the list concept again early 1993
  - Based on other 'fantasy' lists of systems we decided a *real* benchmark was needed to rank systems
  - **Linpack had widest coverage by at least 2-3x**
  - ➢ So we invited Jack and started publishing the TOP500
- We looked at other benchmarks periodically but coverage was always an issue (e.g. NAS PB)

Why did Linpack Work so Well?

Many other reasons, here are 3 essentials for the ***business as a list***:

1) Easy and continuous scalable problem size

- Otherwise you never keep up with Moore's Law
- You would loose comparability across discrete sizes
- It provides ONE simple performance number
- **Simplicity!**

Why did Linpack Work so Well?

Many other reasons, here are 3 essentials for the *business as a list*:

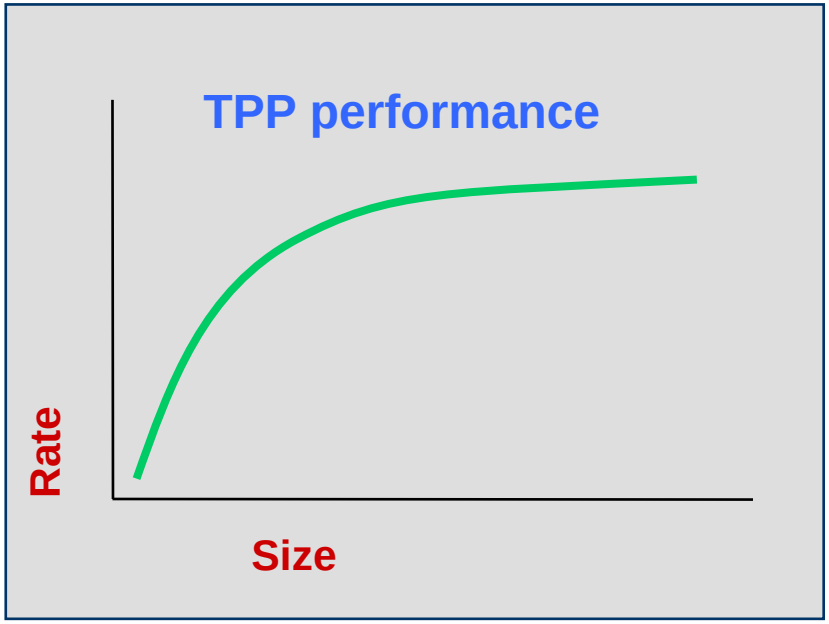
1) Easy and continuous scalable problem size

- Simplicity

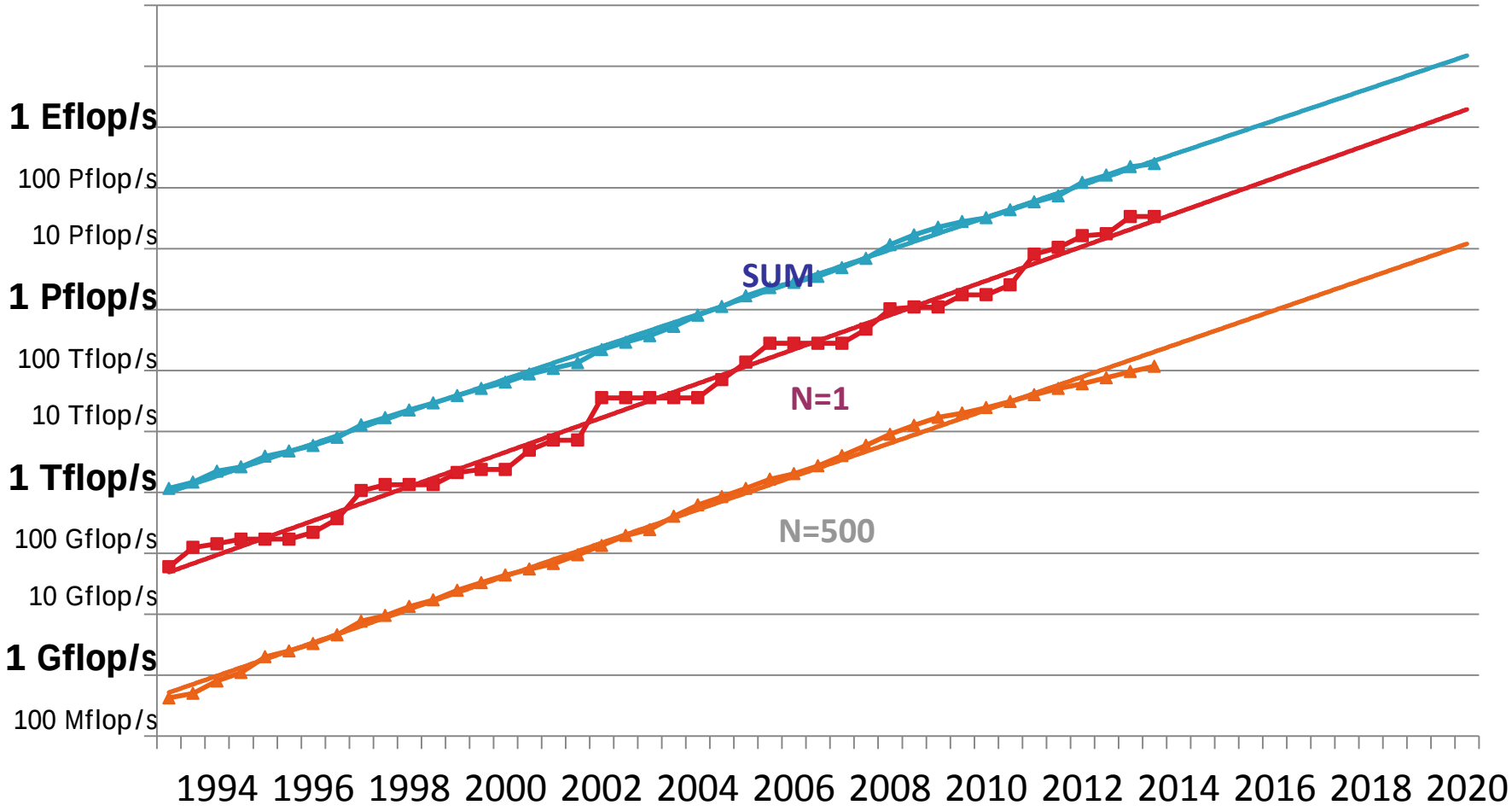
2) Asymptotically best performance

- For both **system size** and **problem size**
- This gets people to measure **full systems** and **fill up the memory** (no in-cache measurements)
- Preempts a boatload of bad tricks and games
- This also means your benchmark must scale aka cannot be too hard!!!
- **Brings out correct long term trends!**

Why did Linpack Work so Well?



Projected Performance Development



Why did Linpack Work so Well?

Many other reasons, here are 3 essentials for the *business as a list*:

1) Easy and continuous scalable problem size

- Simplicity

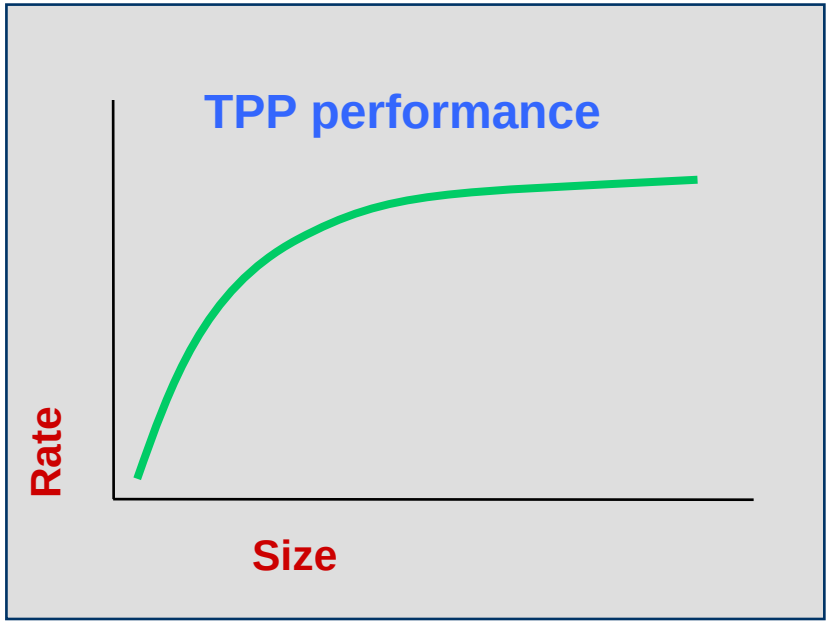
2) Asymptotically best performance

- For both **system size** and **problem size**
- Brings out correct long term trends

3) Convex performance curves over system size and problem size

- This allows a *safe* interpolation to smaller systems
- Important for coverage of large variety of installed system sizes
- This is probably a corollary to 2)
  (It looks like a more restricting requirement)

Why did Linpack Work so Well?



Problems with Linpack

***"Business as a List"* point of view – None:**

- No majors ones
  - Runtimes are fixable

Community point of view - A major one:

- It produces some unrealistic rankings, which do not reflect our common sense!
- Corollaries to this statement are:
  - It does not represent our codes/workloads
  - We can build **stunt-machines**:
    - Achieve high Linpack
    - Are not good for much else!

How to Fix the Ranking?

Different angles people take to fix these problems:

- Find a benchmark, which better **represents** our workloads and depends on all hardware aspects of our systems
- Find or define an **artificial benchmark** which achieves a re-ordering satisfying our common sense
- **Combine** several orthogonal benchmarks to construct a better ranking

In Benchmarking you always have to have a clear objective in mind and select code and metric according to it – otherwise you will fail!

Representative Benchmarks?

Find a benchmark, which better represents our workloads and depends on all hardware aspects of our systems

- Pros:
  - As application scientists and hardware architects, these are the features close to our heart and mind
  - This is crucial for community acceptance
- Cons:
  - It lends itself to unproductive philosophical “wars”
  - **There is no guarantee what-so-ever that it will fix the rankings !!!**
  - **It does not necessarily preclude stunt-machines**

Artificial Benchmarks?

Find or define an artificial benchmark which achieves a re-ordering satisfying our common sense

- Pros:
  - It would fix the ranking (by definition)
    - For now
  - It might (or might not) preclude stunt machines
- Cons:
  - It sounds and feels too artificial from the get-go
  - Community acceptance will be an issue
  - **Artificial benchmarks tend to have a very short shelf life and are more prone to be 'gamed'**
    - **Which opens up the possibilities of stunt machines all over again**

Aggregate Metrics?

Combine several orthogonal benchmarks to construct a better ranking

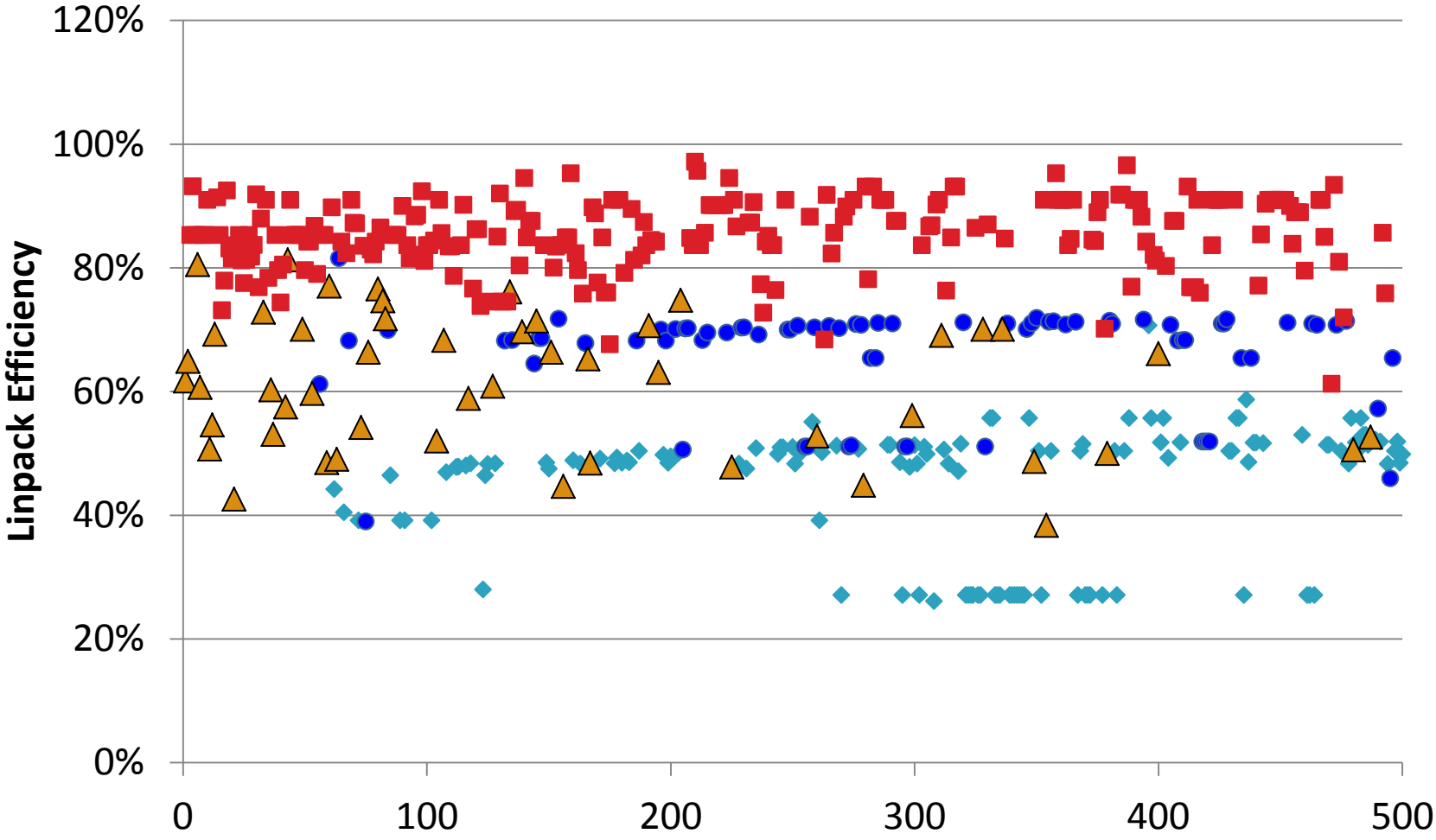
- Pros:
  - There are sound ways to do it
    - (I published some ;-)
  - It might fix the ranking
    - with proper weights
    - For now …
  - It might (or might not) preclude stunt machines
- Cons:
  - There will be free parameters=weights
  - It sounds and feels too artificial and complex from the get-go
  - Community acceptance will be an issue (confusing metrics)
  - Artificial benchmarks tend to have a very short shelf life and are more prone to be 'gamed'
    - Which opens up the possibilities of stunt machines all over again

3 Criteria for a Useful New Benchmark

- **Represents a real computational problem**:
  - Allows simple problem scaling
  - Performs asymptotically optimal for system and problem size
- **Changes relative rankings**:
  - To fit common sense
  - **By a sufficient magnitude !!!**

#	Site	Manufacturer	Computer	Country	Cores	Rmax [Pflops]	Power [MW]
1	National University of Defense Technology	NUDT	Tianhe-2 NUDT TH-IVB-FEP, Xeon 12C 2.2GHz, IntelXeon Phi	China	3,120,000	33.9	17.8
2	Oak Ridge National Laboratory	Cray	Titan Cray XK7, Opteron 16C 2.2GHz, Gemini, NVIDIA K20x	USA	560,640	17.6	8.21
3	Lawrence Livermore National Laboratory	IBM	Sequoia BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	1,572,864	17.2	7.89
4	RIKEN Advanced Institute for Computational Science	Fujitsu	K Computer SPARC64 VIIIfx 2.0GHz, Tofu Interconnect	Japan	795,024	10.5	12.7
5	Argonne National Laboratory	IBM	Mira BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	786,432	8.59	3.95
6	Swiss National Supercomputing Centre (CSCS)	Cray	Piz Daint Cray XC30, Xeon E5 8C 2.6GHz, Aries, NVIDIA K20x	Switzer-land	115,984	6.27	2.33
7	Texas Advanced Computing Center/UT	Dell	Stampede PowerEdge C8220, Xeon E5 8C 2.7GHz, Intel Xeon Phi	USA	462,462	5.17	4.51
8	Forschungszentrum Juelich (FZJ)	IBM	JuQUEEN BlueGene/Q, Power BQC 16C 1.6GHz, Custom	Germany	458,752	5.01	2.30
9	Lawrence Livermore National Laboratory	IBM	Vulcan BlueGene/Q, Power BQC 16C 1.6GHz, Custom	USA	393,216	4.29	1.97
10	Leibniz Rechenzentrum	IBM	SuperMUC iDataPlex DX360M4, Xeon E5 8C 2.7GHz, Infiniband FDR	Germany	147,456	2.90	3.52

Linpack Efficiency



3 Criteria for a Useful New Benchmark

- **Represents a real computational problem**:
  - Allows simple problem scaling
  - Performs asymptotically optimal for system and problem size
- **Changes relative rankings**:
  - To fit common sense
  - By a sufficient magnitude
- **Passes the stunt-machine test**:
  - We can not construct a stunt-machine to defeat it!