Performance Evaluation with the HPCC Benchmarks as a Guide on the Way to Peta Scale Systems

Rolf Rabenseifner, Michael M. Resch, Sunil Tiyyagura, Panagiotis Adamidis

rabenseifner@hlrs.de **resch@hlrs.de** **sunil@hlrs.de** **adamidis@hlrs.de**

University of Stuttgart
High-Performance Computing-Center Stuttgart (HLRS)
www.hlrs.de

Dagstuhl Seminar 06071

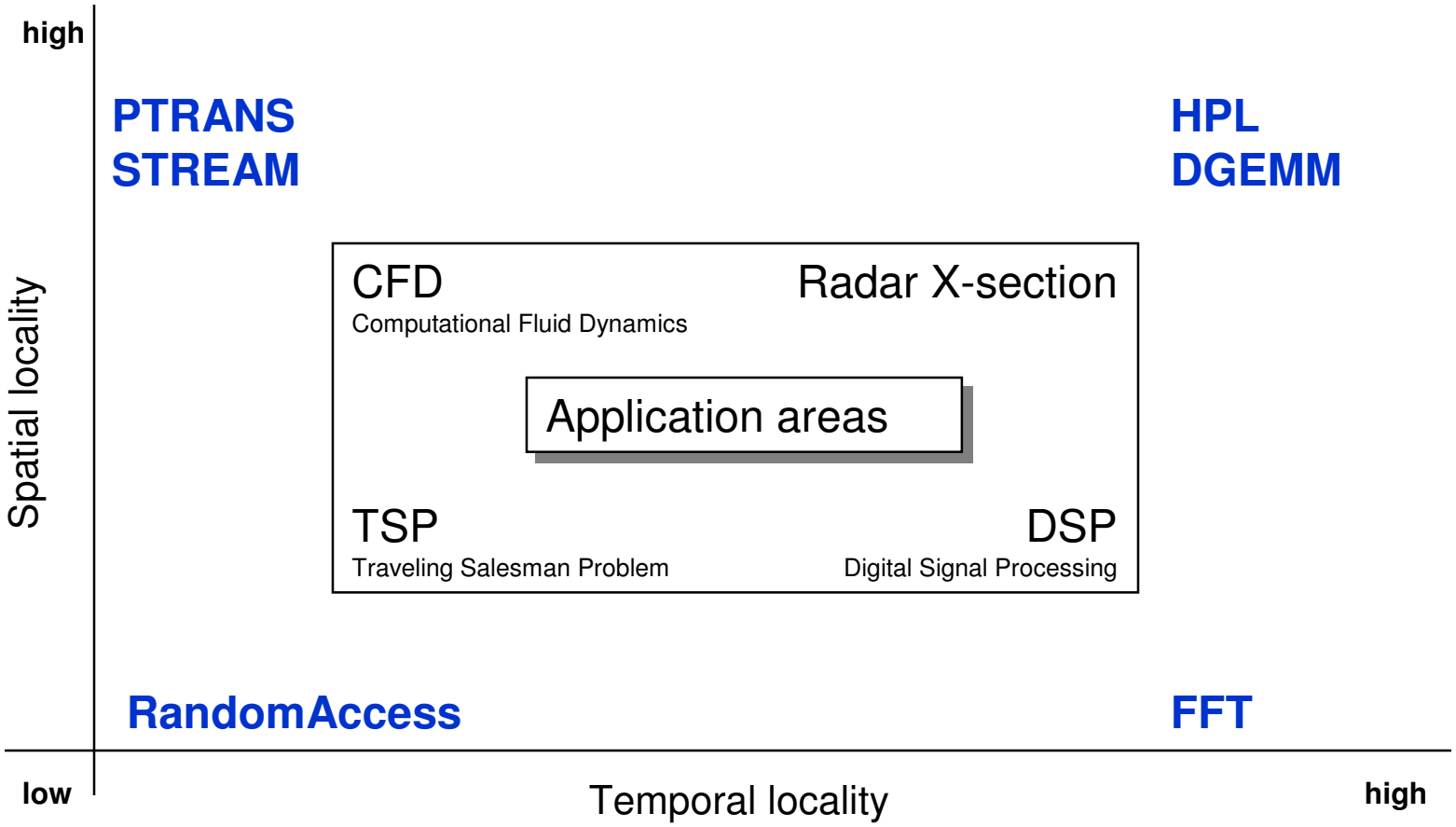
Schloß Dagstuhl, Wadern, Germany, Feb. 12-17, 2006

(HPCC data status Feb. 6, 2006)

Outline

- How HPC Challenge Benchmark (HPCC) data can be used to analyze the balance of HPC systems
  - **Details on ring based benchmarks**
- Resource based ratios
  - **Inter-node bandwidth and**
  - **memory bandwidth**
  - **versus computational speed**
  - **Comparison mainly based on public HPCC data**
- Towards Petaflop computing
  - **Total cost of ownership**
  - **Programmability**
  - **Usability**
- Conclusions

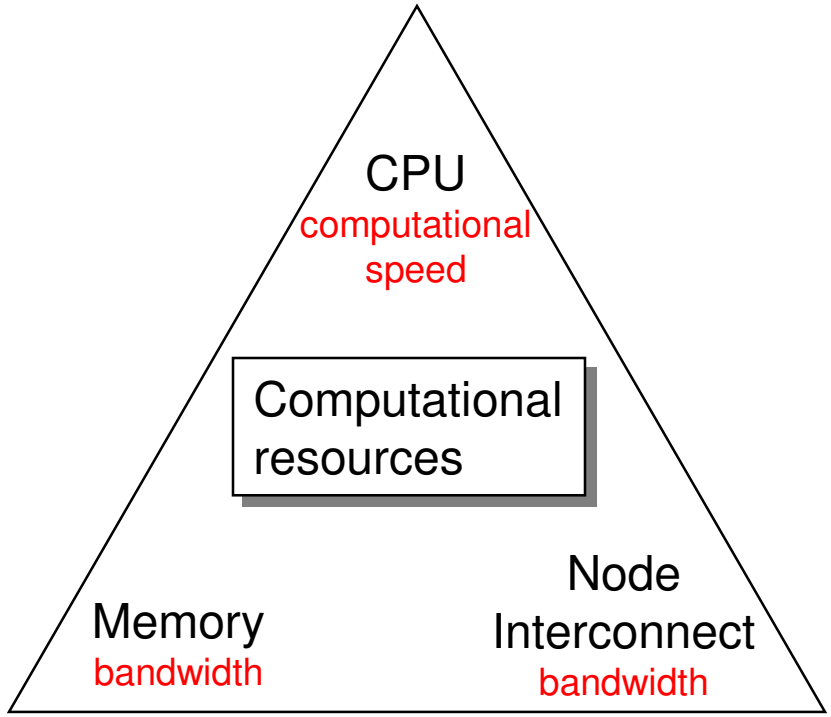
Application areas & HPC Challenge Benchmarks



Courtesy of Jack J. Dongarra and Piotr Luszczek

Computational Resources & HPC Challenge Benchmarks

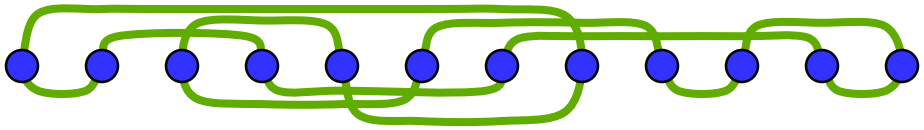
HPL
(Jack Dongarra)



STREAM
(John McCalpin)

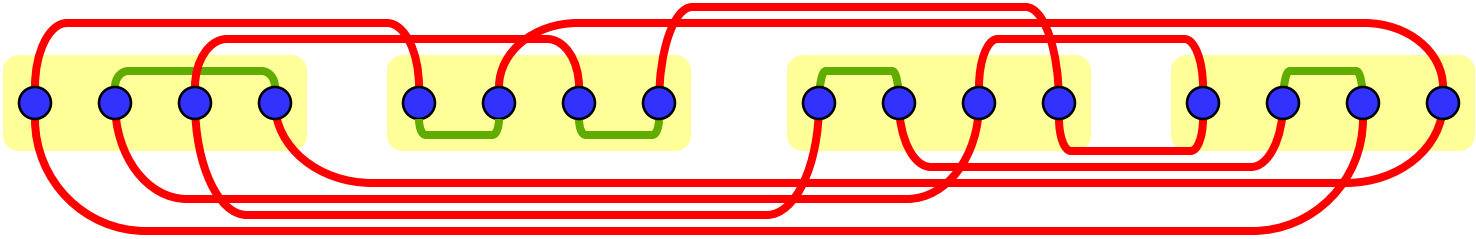
Random & Natural
Ring
Bandwidth & Latency
(my part of the
HPCC Benchmark Suite)

Random & natural ring bandwidth & latency

- Parallel communication pattern on all MPI processes (●)
  - Natural ring 
  - Random ring 
- Bandwidth per process
  - Accumulated message size / wall-clock time / number of processes
  - On each connection messages in both directions
  - With *2xMPI_Sendrecv* and *MPI non-blocking* → best result is used
  - Message size = 2,000,000 bytes
- Latency
  - Same patterns, message size = 8 bytes
  - Wall-clock time / (number of sendrecv per process)

Inter-node bandwidth on clusters of SMP nodes

- Random Ring
  - Reflects the other dimension of a Cartesian domain decomposition and
  - Communication patterns in unstructured grids
  - Some connections are inside of the nodes
  - Most connections are inter-node
  - Depends on #nodes and #MPI processes per node



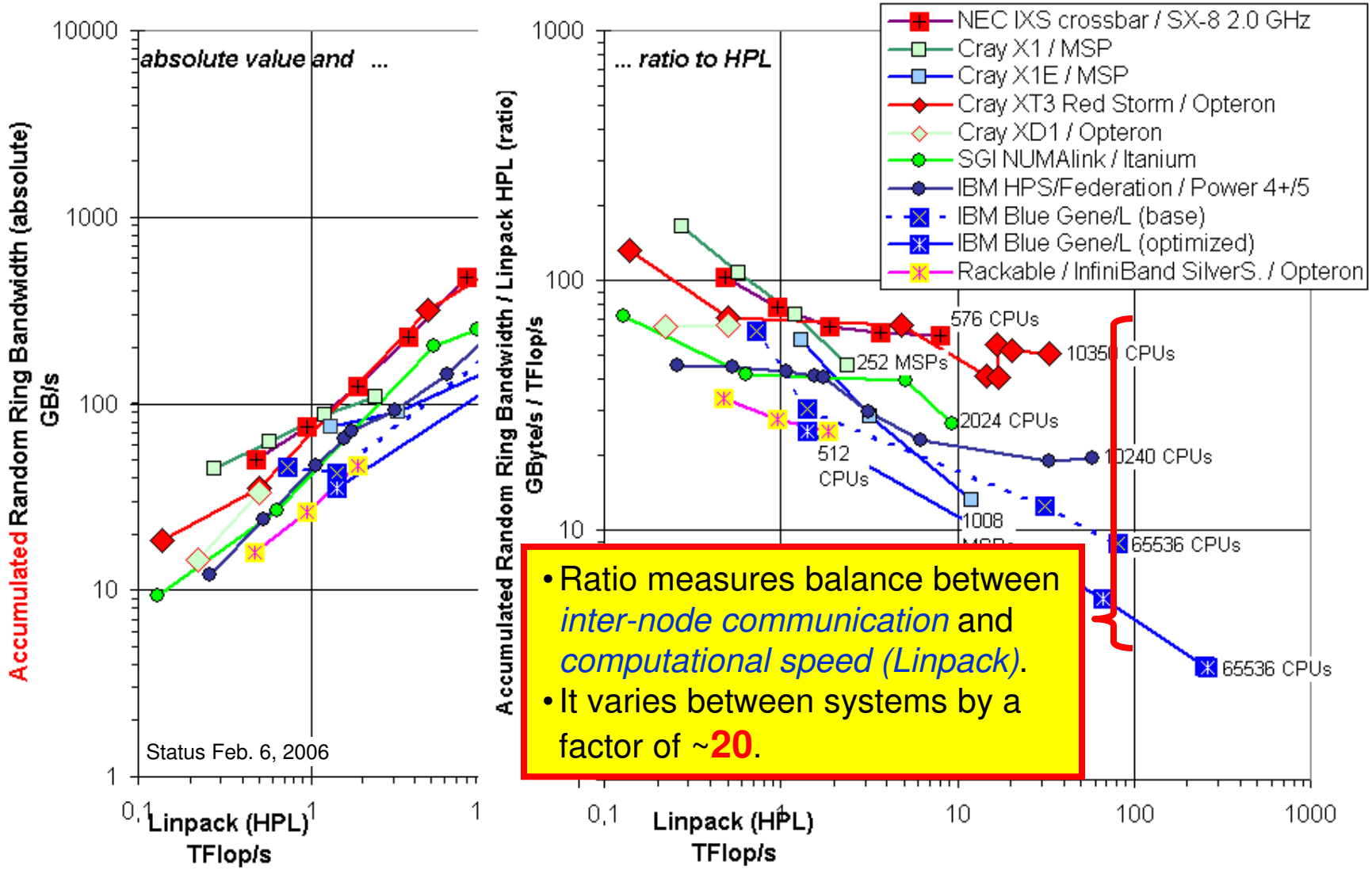
  - Accumulated bandwidth
    := bandwidth per process x #processes
  - ~= accumulated inter-node bandwidth x $(1 - 1 / \#nodes)^{-1}$

similar to bi-section bandwidth

Balance Analysis with HPC Challenge Benchmark Data

- Balance can be expressed as a set of ratios
  - e.g., accumulated memory bandwidth / accumulated Tflop/s rate
- Basis
  - **Linpack (HPL)** → **Computational Speed**
  - **Random Ring Bandwidth** → **Inter-node communication**
  - **Parallel STREAM Copy or Triad** → **Memory bandwidth**
- Be careful:
  - Some data are presented for the **total system**
  - Some per **MPI process** (HPL processes)
  - i.e., balance calculation always with accumulated data on the total system

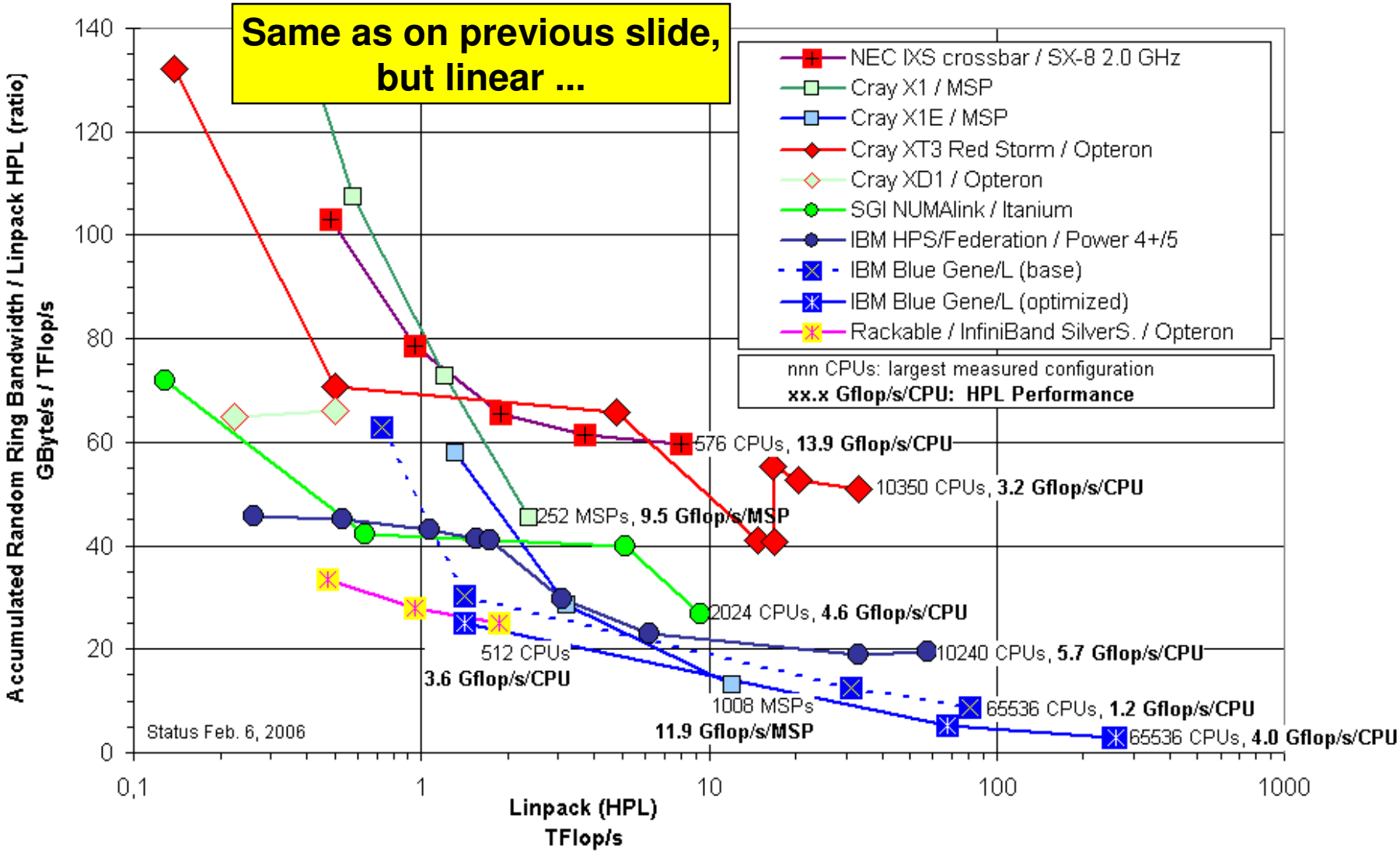
Balance between Random Ring Bandwidth (network b/w) and HPL Benchmark (CPU speed)



- Ratio measures balance between *inter-node communication* and *computational speed (Linpack)*.
- It varies between systems by a factor of ~**20**.

Measurements with smaller #CPUs on XT3 and SX-8 courtesy to Nathan Wichmann, Cray, CUG 2005, and Sunil Tiyyagura, HLRS.

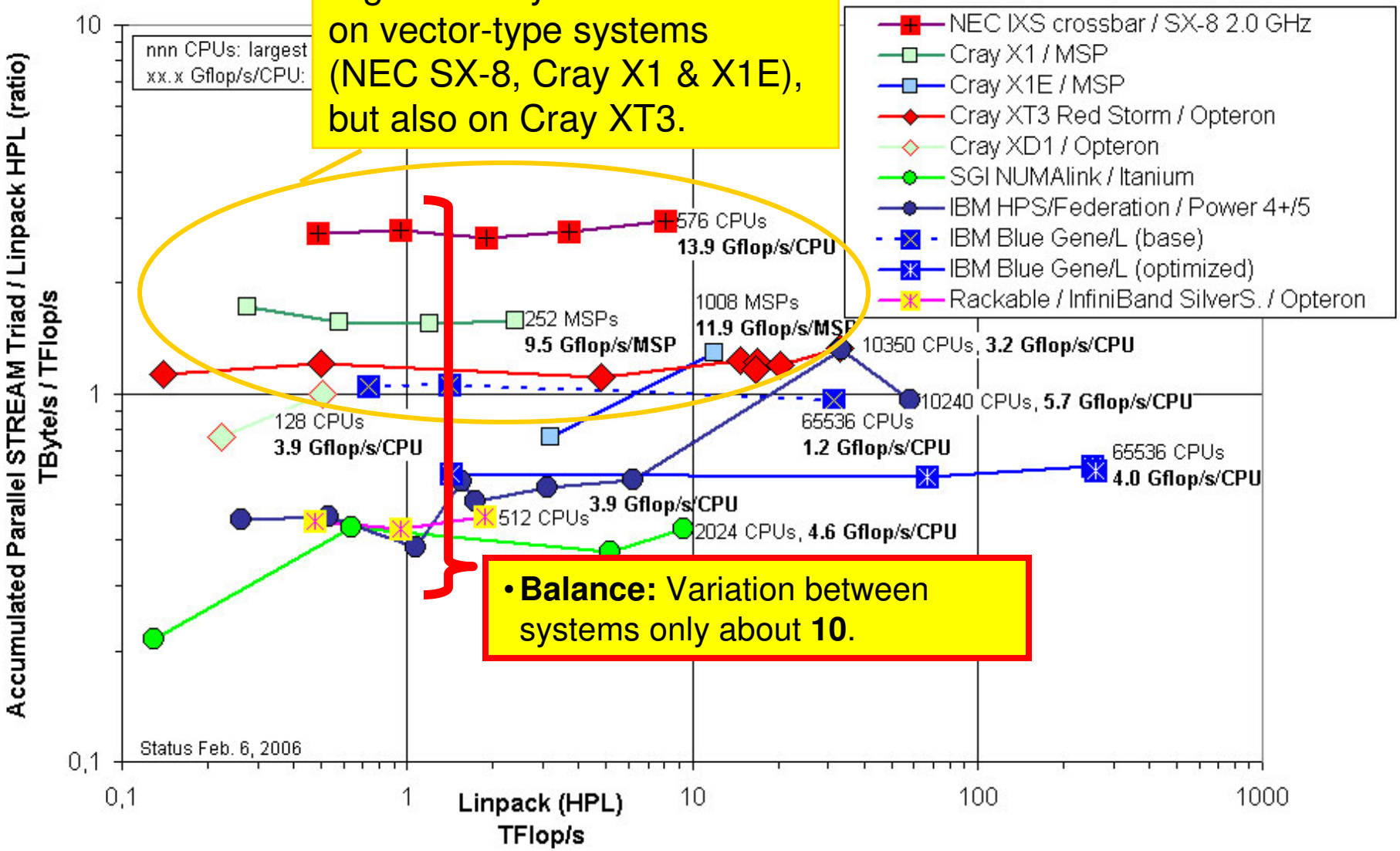
Balance between Random Ring Bandwidth and CPU speed



Measurements with smaller #CPUs on XT3 and SX-8 courtesy to Nathan Wichmann, Cray, CUG 2005, and Sunil Tiyyagura, HLRS.

Balance between memory and CPU speed

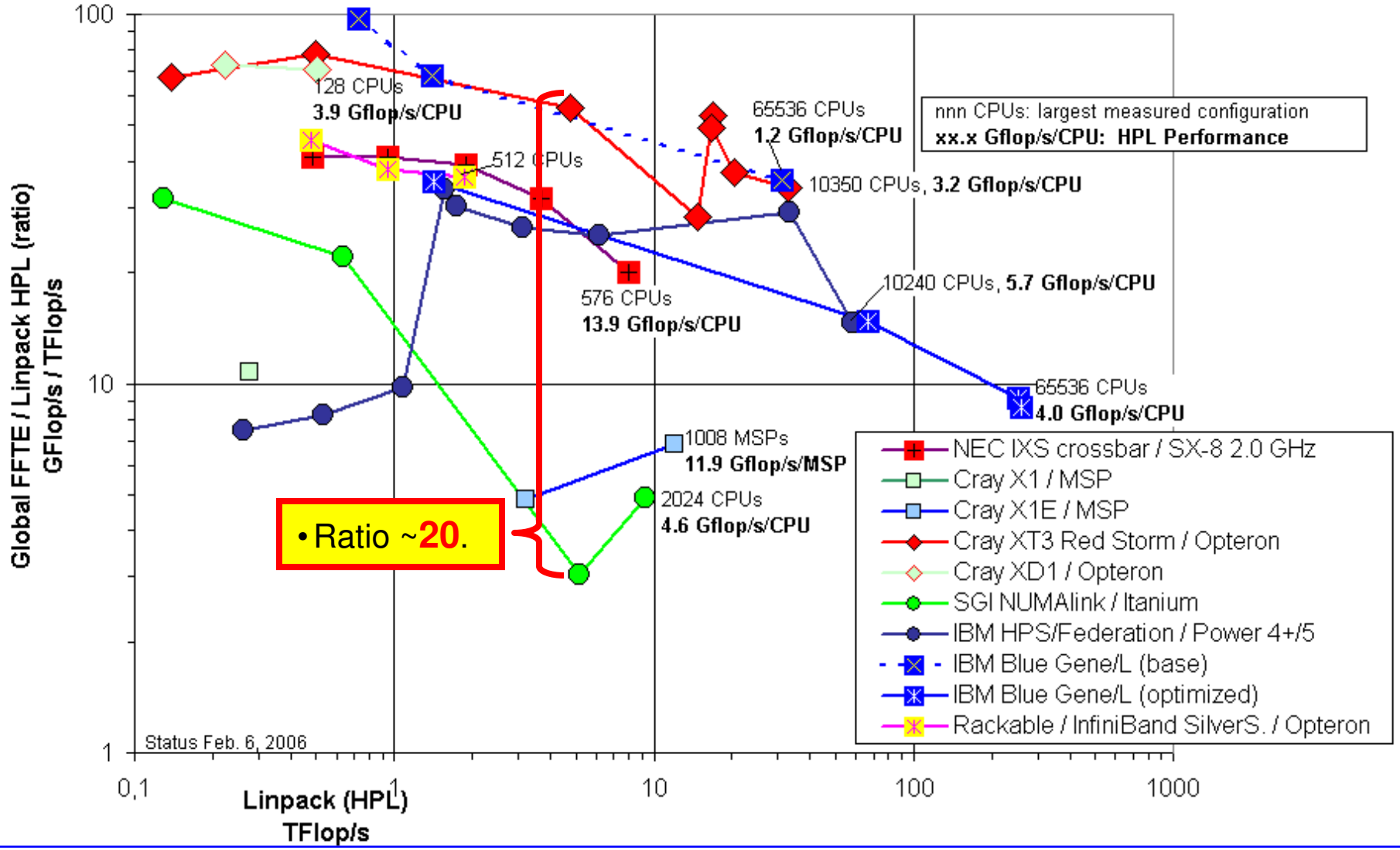
High memory bandwidth ratio on vector-type systems (NEC SX-8, Cray X1 & X1E), but also on Cray XT3.



- **Balance:** Variation between systems only about **10**.

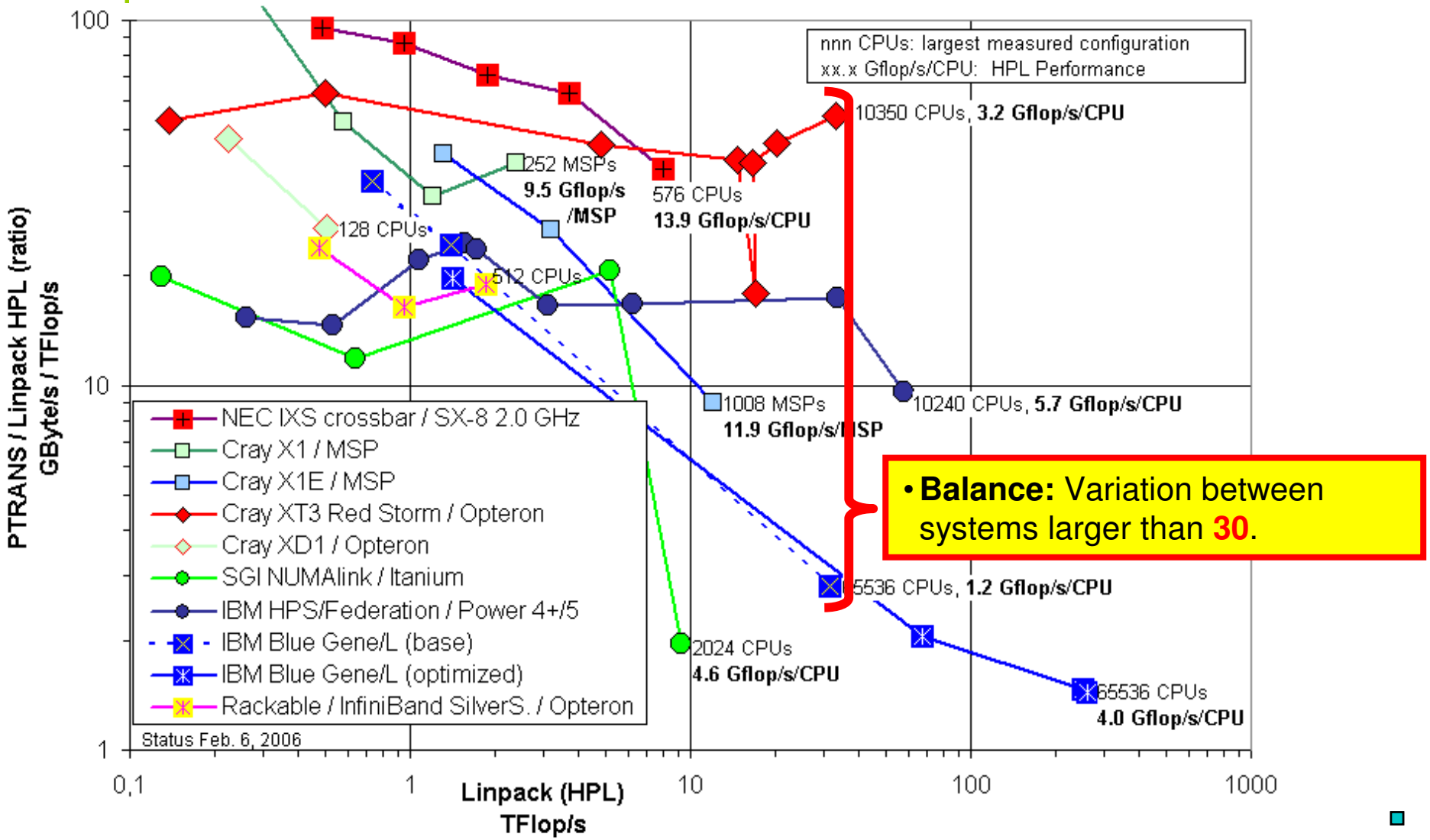
Measurements with smaller #CPUs on XT3 and SX-8 courtesy to Nathan Wichmann, Cray, CUG 2005, and Sunil Tiyyagura, HLRS.

Balance between Fast Fourier Transform (FFTE) and CPU



Measurements with smaller #CPUs on XT3 and SX-8 courtesy to Nathan Wichmann, Cray, CUG 2005, and Sunil Tiyyagura, HLRS.

Balance between Matrix Transpose (PTRANS) and CPU



- **Balance:** Variation between systems larger than **30**.

Measurements with smaller #CPUs on XT3 and SX-8 courtesy to Nathan Wichmann, Cray, CUG 2005, and Sunil Tiyyagura, HLRS.

Towards PetaFlop/s Systems

E.g., with commodity processors:

- An estimate:
  - 2 GHz clock rate
  - * 4 floating point operations / cycle
  - * 8 CPU-cores per chip
  - * 2 chips per SMP node
  - * 32 SMP nodes per cabinet
  - * 300 cabinets
  - = **1.23 PFlop/s** theoretical peak performance with **153,600 CPUs**
- With 240 W/Chip (i.e., 30 W/CPU)
  - → 15.4 kW / rack
  - → **4.6 MW** in total

PFlop/s systems foreseen
- **within the BlueGene project**
- **and in the Japanese project**
- **for the years 2008 – 2010**

Costs:
- 5 k$ per node * 9600 nodes = **48 Mio.$** hardware
- 4.6 MW * 5 year * 365 day/year * 24 h/day * 0.114 $/kWh*) = **23 Mio.$** for 5y power consum.

H L R S

*) 0.114 $/kWh = 1 M$/MWa

Total Cost of Ownership

Costs:
- 5 k$ per node * 9600 nodes = **48 Mio.$** hardware
- 4.6 MW * 5 year * 365 day/year * 24 h/day * 0.114 $/kWh = **23 Mio.$** for 5y power consum.

- Including also maintenance
- → **operational costs** may be **50% of total cost of ownership**
- i.e. hardware only 50%

Mean Time Between Failure

- 18,200 multi-CPU chips

  → MTBF ~ 7 days
  (better than BlueGene/L with 65536 dual-CPU chips?)

Balance

- Estimates range from 100,000 to 1,000,000 processors
- Inter-node communication may become critical bottle-neck
- Memory bandwidth is also tough
  with, e.g., 8 cores x 4 floating point units per chip
- On vector systems, caching will be necessary to overcome the problem that n-point-stencil data is n times reloaded from memory
- Today, balance factors are in a range of
  - **20** ← inter-node communication / HPL-TFlop/s
  - **10** ← memory speed / HPL-TFlop/s
  - **20** ← FFTE / HPL-TFlop/s
  - **30** ← PTRANS / HPL-TFlop/s

  → The value of 1 HPL TFlop/s may be reduced by such balance factors

Programmability

- Estimates range from 100,000 to 1,000,000 processors

- MPI works on such numbers
  - Bandwidth typically used with a small amount of neighbors for coarse grained domain decomposition
  - Global latencies should appear only in collective MPI routines
  - Collectives should internally be at least tree or butterfly based
    → O(log(#processors)),
    i.e., latency only doubled from 1,000 to 1,000,000 processors,
    i.e., from O(10) to O(20)
  - Most MPI optimizations can be done based on current MPI standard
    → lightweight MPI is not necessary

Programmability (continued)

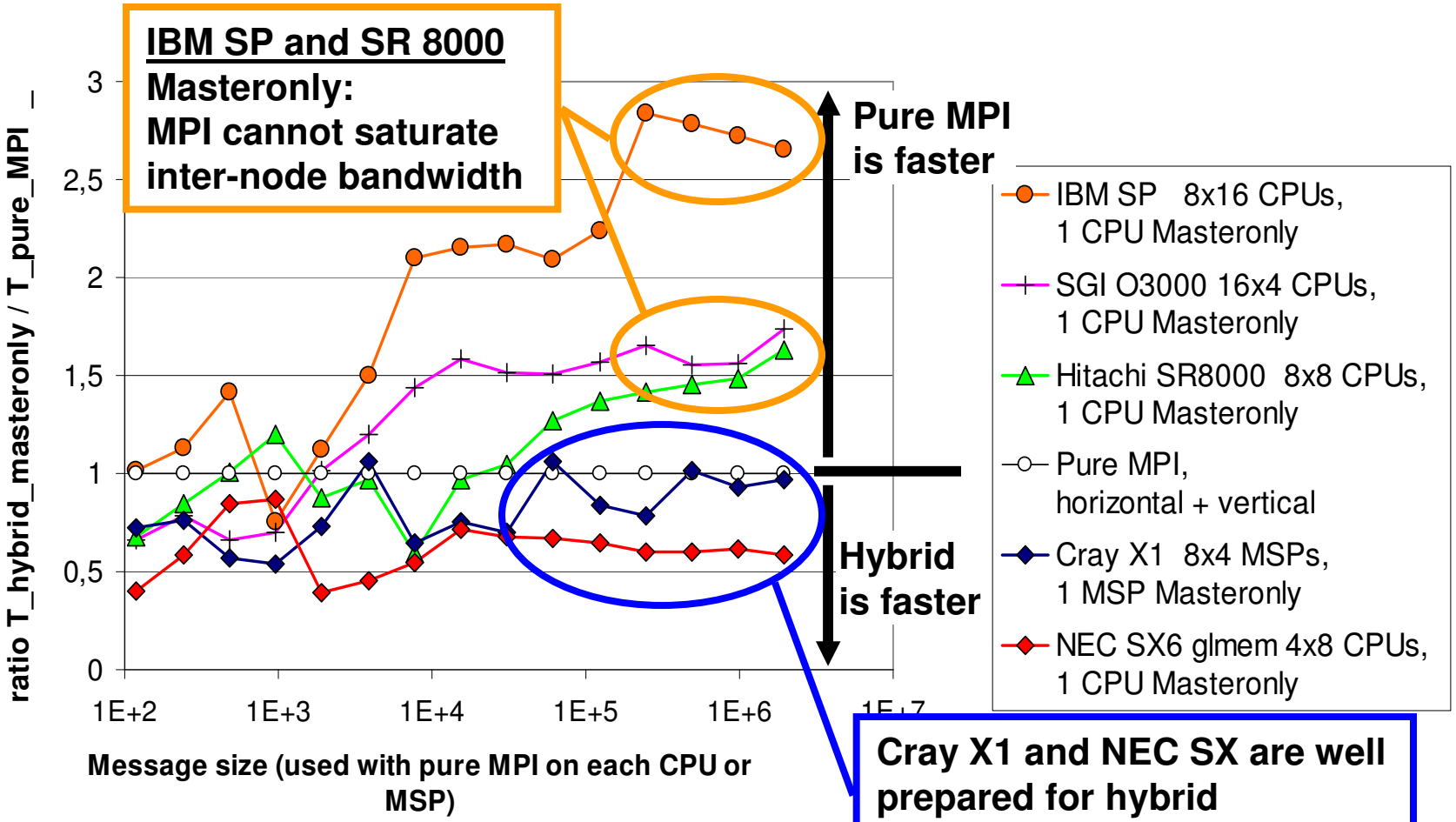
- Estimates range from 100,000 to 1,000,000 processors

- Hardware is always a **cluster of SMP nodes**
- Hybrid (mixed model) MPI+OpenMP involves additional problems:
  - Typical programming style:
    Master-only = MPI outside of OpenMP parallel regions
  - Is one CPU per SMP node able to saturate the inter-node network?
  - Is this still valid with 16 CPUs per SMP node?
- Must application domain decomposition be matched on inter-node network topology?
- Overhead in MPI and OpenMP → reduced speed-up

For further reading:

- Rolf Rabenseifner: **Hybrid Parallel Programming on HPC Platforms.**
  In proceedings of the Fifth European Workshop on OpenMP, EWOMP '03, Aachen, Germany, Sept. 22-26, 2003, pp 185-194.

Ratio on several platforms



Cray X1 and NEC SX are well prepared for hybrid *masteronly* programming

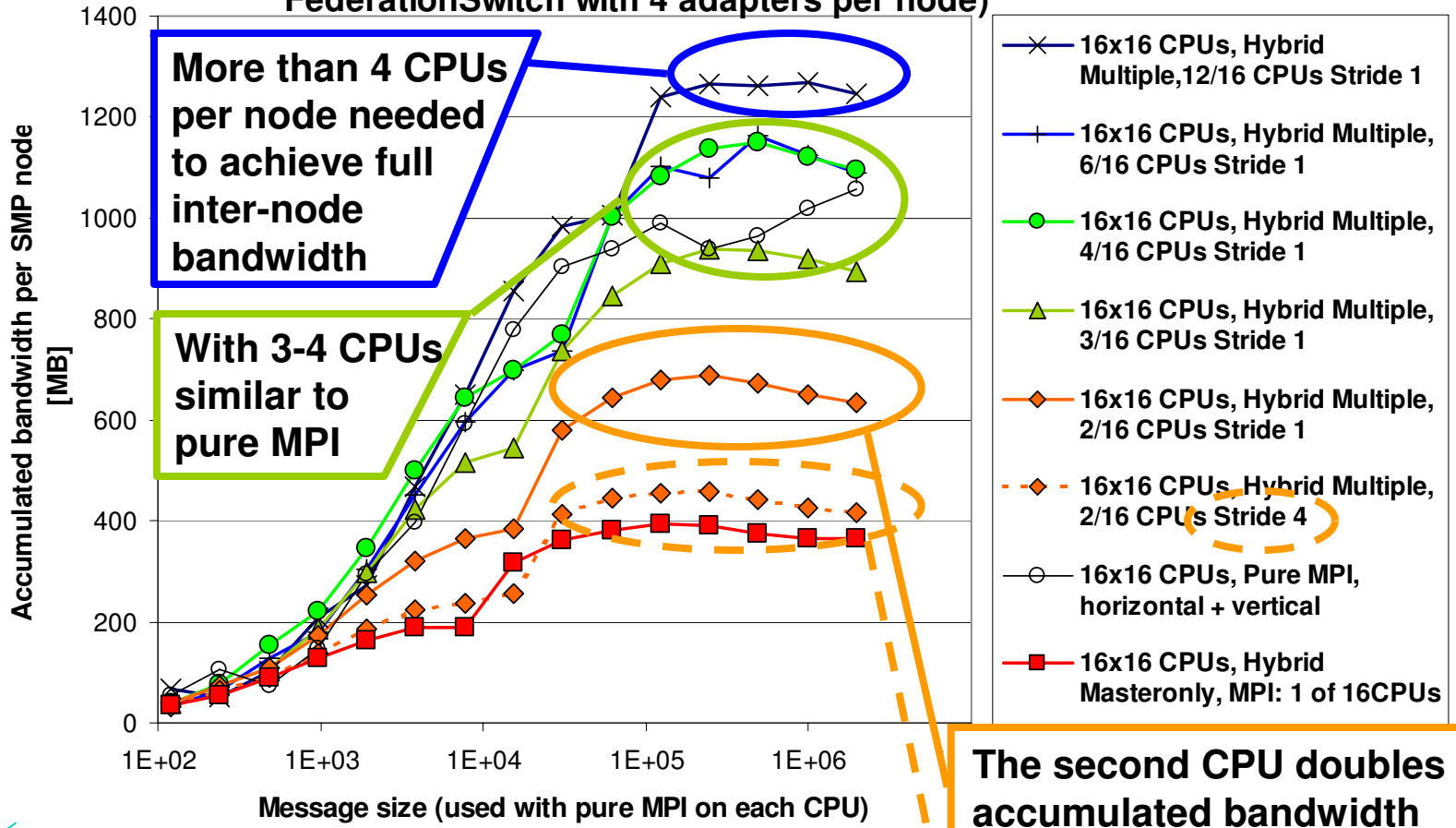
Further information:
www.hlrs.de/organization/par/par_prog_ws/
→ [23] MPI on hybrid systems / MPI + OpenMP

H L R S

Cray X1 and SGI results are preliminary

Multiple inter-node communication paths: IBM SP

Inter-node bandwidth per SMP node, accumulated over its CPUs,*)
on IBM at Juelich (32 Power4+ CPUs/node,
FederationSwitch with 4 adapters per node)



More than 4 CPUs per node needed to achieve full inter-node bandwidth

With 3-4 CPUs similar to pure MPI

The second CPU doubles the accumulated bandwidth

But only if second process is located on CPU connected with 2nd adapter!

Measurements: Thanks to Bern Mohr, ZAM, FZ Lülich

*) Bandwidth per node: totally transferred bytes on the inter-node network / wall clock time / number of nodes

Programmability (continued)

- Estimates range from 100,000 to 1,000,000 processors

- Load balancing in multi physics codes
- With mixtures of several types of grids

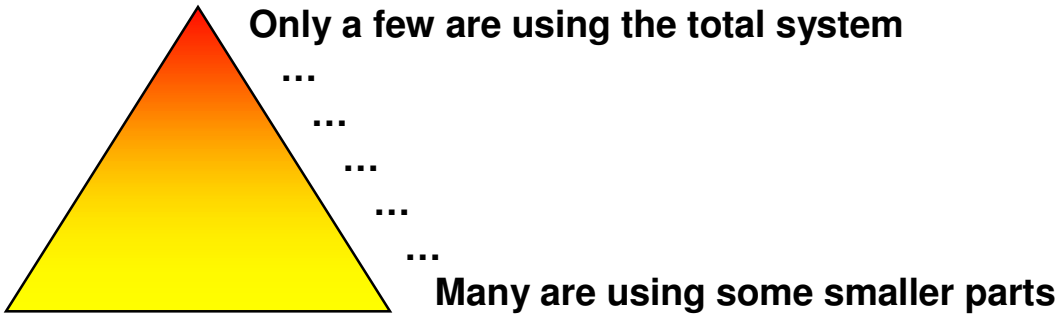
- Doing ALL in parallel is hard
- Omitting all serial parts (**e.g. some I/O**) is crucial and hard on 1,000,000 processors

For further reading:

- Rolf Rabenseifner, Alice E. Koniges, Jean-Pierre Prost, and Richard Hedges: **The Parallel Effective I/O Bandwidth Benchmark: b_eff_io.** In Christophe Cerin and Hai Jin (Eds.), Parallel I/O for Cluster Computing, Chap. 4. (pp 107-132), Kogan Page Ltd., Feb. 2004, ISBN 1-903996-50-3.

Usability

- Estimates range from 100,000 to 1,000,000 processors
- Systems should be still available to a large user community



- Ease of use is crucial for the future of super-computing
- What is the benefit from new parallel programming models on real complex applications?

Acknowledgments

- Thanks to
  - all persons and institutions that have uploaded HPCC results.
  - Jack Dongarra and Piotr Luszczek
    for inviting me into the HPCC development team.
  - Matthias Müller, Sunil Tiyyagura and Holger Berger
    for benchmarking on the SX-8 and SX-6 and discussions on HPCC.
  - Nathan Wichmann from Cray for Cray XT3 and X1E data.

- References

  - S. Saini, R. Ciotti,B. Gunney, Th. Spelce, A. Koniges, D. Dossa, P. Adamidis, R. Rabenseifner, S. Tiyyagura, M, Müller, and R. Fatoohi: **Performance Evaluation of Supercomputers using HPCC and IMB Benchmarks.**
    Will be published in the proceedings of the IPDPS 2006 Conference.
  - R. Rabenseifner, S. Tiyyagurra, M. Müller: **Network Bandwidth Measurements and Ratio Analysis with the HPC Challenge Benchmark Suite (HPCC).**
    Proceedings of the 12th European PVM/MPI Users' Group Meeting, EuroPVM/MPI 2005
  - **HPC Benchmark Suite** → http://icl.cs.utk.edu/hpcc/

Conclusions

- Peta scale systems are coming.
- HPCC is an interesting basis for
  - **benchmarking computational resources**
  - **analyzing the balance of a system**
  - **scaling with the number of processors**
  - **with respect to application needs**
- HPCC helps to show the strength and weakness of super-computers
- Future super computing should not focus only on PFlop/s in the TOP 500
  - **Memory and network bandwidth are as same as important to predict real application performance**
- Usability for broader user community
  - **Important for wide acceptance of the importance of super-computing**
- Benefit of new programming models for a large community and their real applications?