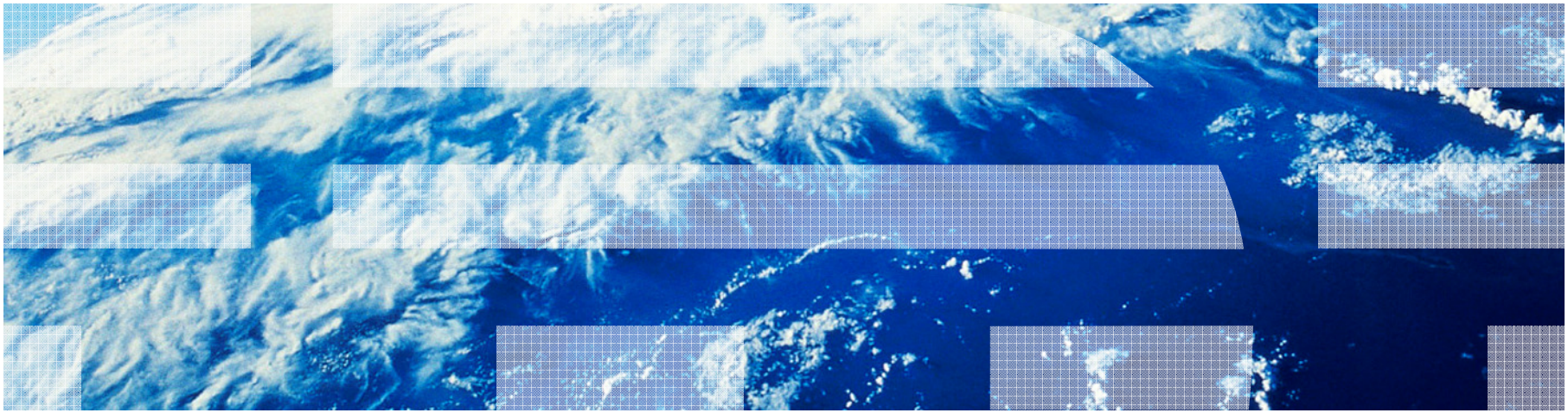
IBM

Interconnection Network Architectures for High-Performance Computing

Cyriel Minkenberg
IBM Research — Zurich



© 2013 IBM Corporation

HPC interconnection networks

- This lecture relates most closely to Lecture 9 (Apr. 23) “High-Performance Networking I”

- Main objective: Provide you with the basic concepts encountered in HPC networks and how these are applied in practice

© 2013 IBM Corporation

Outline

1. Brief introduction to HPC
2. Current HPC landscape (Top 500)
3. Role of the interconnect
4. Topology
5. Routing
6. Flow control & deadlock avoidance
7. Workloads
8. Performance evaluation
9. Outlook

© 2013 IBM Corporation

Brief intro to HPC

© 2013 IBM Corporation

HPC – a brief history

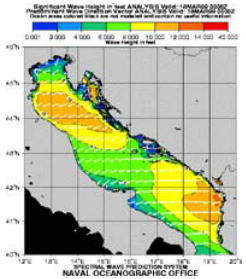
- Vector machines
- SMPs
- Clusters
- MPPs

© 2013 IBM Corporation

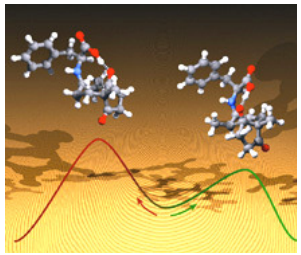
HPC use cases

- **Technical and scientific computing, for instance:**
  - Molecular modeling (e.g. CPMD)
  - Quantum mechanics
  - Weather forecasting
  - Climate research
  - Ocean modeling
  - Oil and gas exploration
  - Aerodynamics (airplanes, automobiles), aquadynamics (ships)
  - Nuclear stockpile stewardship
  - Nuclear fusion

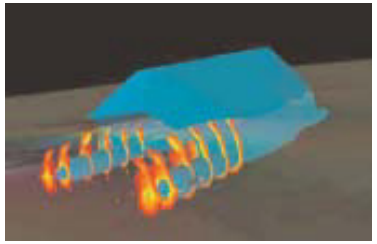
Weather forecasting



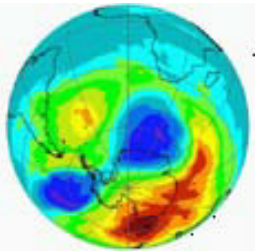
Ocean modeling



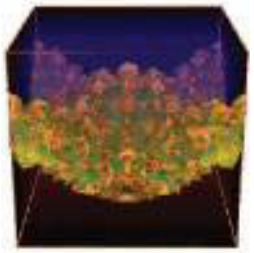
Molecular science



Ship design



Climate modeling



Nuclear stockpile stewardship

© 2013 IBM Corporation

Capability vs. capacity systems

- **A *capability* system aims to to solve a single huge problem as quickly as possible by applying its entire compute power to that one problem**
  - 1 machine = 1 job
  - For Top500 ranking purposes, a machine is typically operated in this mode

- **A *capacity* system aims to solve a plurality of problems simultaneously by partitioning its compute power and applying one partition to each job (in parallel)**
  - 1 machine = several large jobs or many small jobs
  - Many large "open" machines at government labs or universities are operated in this mode
  - Sophisticated job scheduling engines try to allocate resources to match job requirements in fair way

© 2013 IBM Corporation

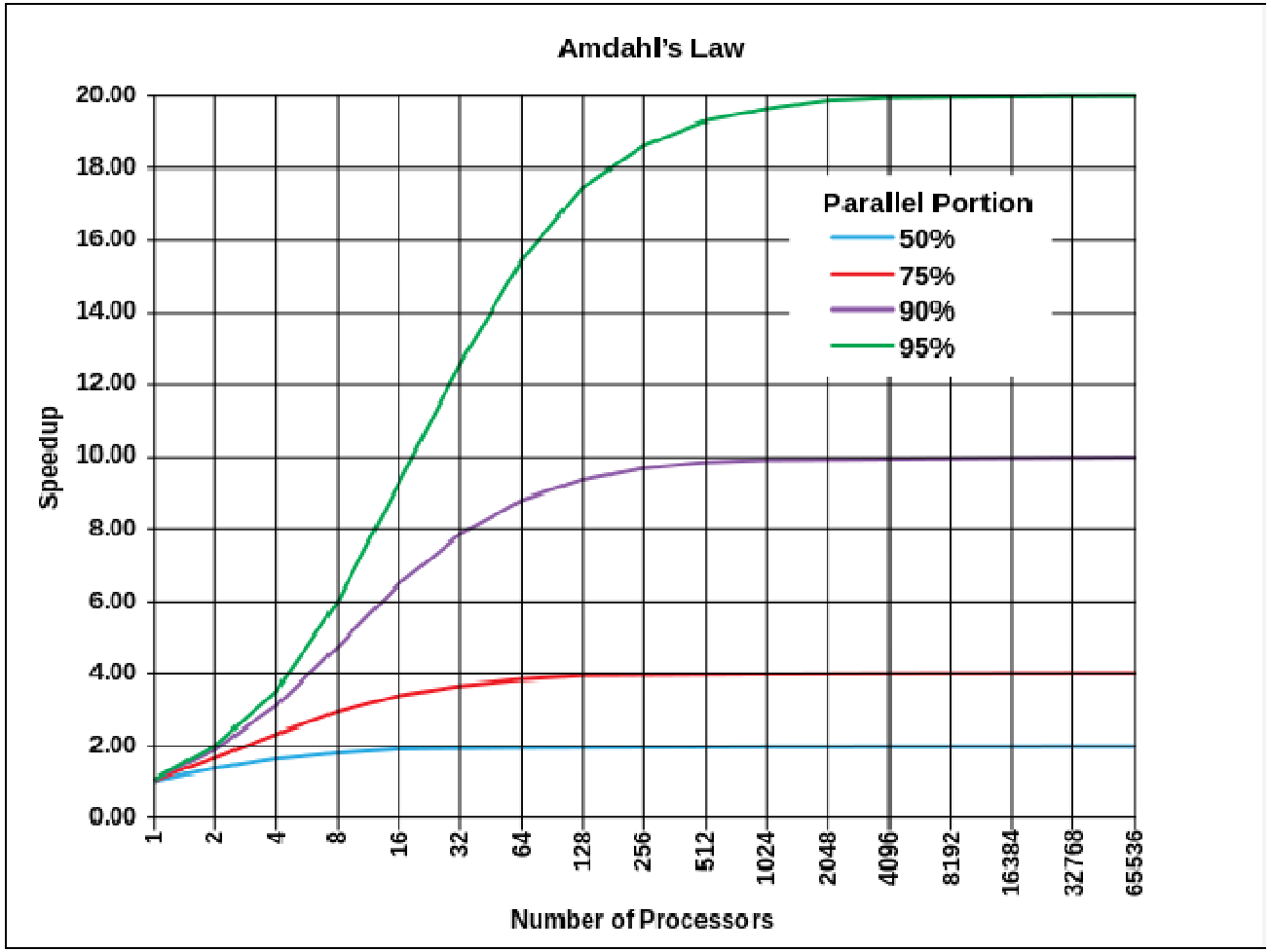
Strong scaling vs. weak scaling

- **Scaling: How much faster is a given problem solved by applying more processors?**
  - Speedup(n) = S(n) = ($t_{execution}$ with 1 processor) / ($t_{execution}$ with n processors)
  - Strong scaling: Increase the number of processor *n while keeping the problem size constant*
  - Weak scaling: Increase the number of processors *n while increasing the problem size (commensurately)*
  - Strong scaling is generally (much) harder than weak scaling

- **Amdahl's Law**
  - Speedup is limited by the serial (non-parallelizable part of the program)
  - S(n) ≤ 1 / (B+(1-B)/n), where B is the percentage of the program that is serial
  - For most codes, beyond a certain n performance stops improving
  - …and may get worse, because of increasing communication overheads!

© 2013 IBM Corporation

Amdahl's law



Source: wikipedia

© 2013 IBM Corporation

HPC vs Cloud

The main difference between HPC and cloud is not primarily in the hardware, it's in the WORKLOADS!

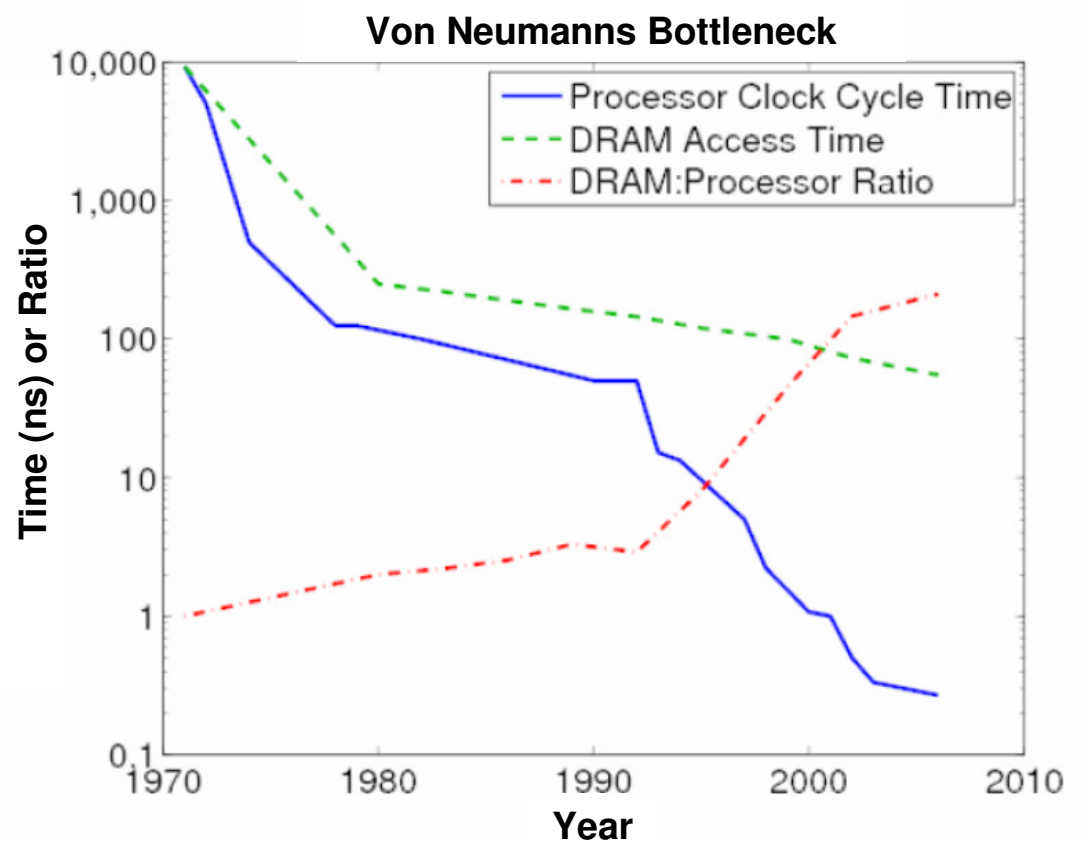
Different workloads means different requirements means a Cloud-optimized system looks quite different from an HPC-optimized system.

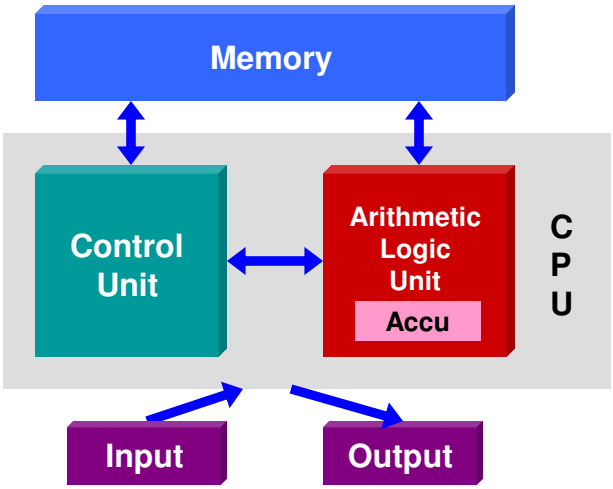
© 2013 IBM Corporation

Role of the interconnect

© 2013 IBM Corporation

Memory / CPU performance mismatch





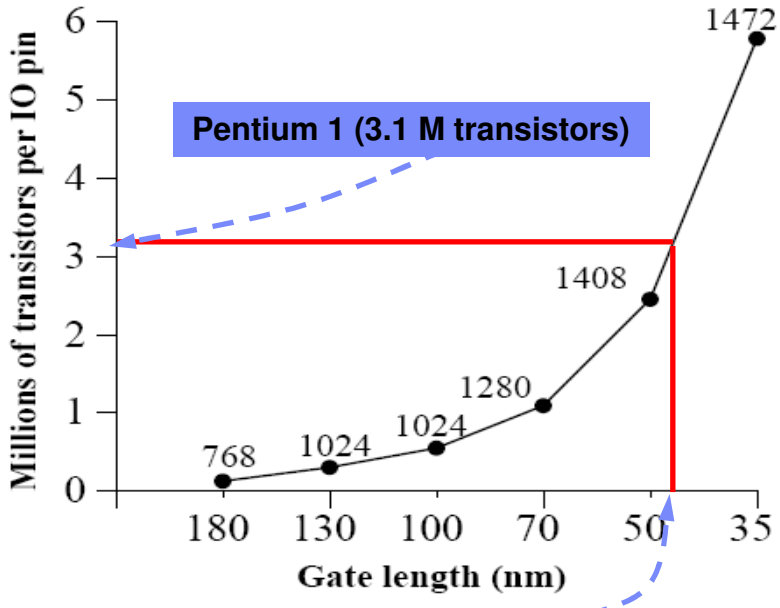
Source: ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems, Sept. 2008 (DARPA)

© 2013 IBM Corporation

The interconnect – no longer 2nd class citizen?

- **Conventional wisdom**
  - Computation = expensive
  - Communication = cheap
  - ➔Corollary: Processor is king

- **Then something happened…**
  - Computation
    - Transistors became "free" ➔ more parallelism: superscalar, SMT, multi-core, many-core
    - Huge increase in FLOPS/chip
  - Communication
    - Packaging & pins remained expensive
    - Scaling of per-pin bandwidth did not keep pace with CMOS density
  - ➔Consequence
    - Comp/comm cost ratio has changed fundamentally
    - Memory and I/O bandwidth now an even scarcer resource



For the 45nm node, all I/O and power & ground connections of an Pentium1-equivalent chip will have to be served by ONE package pin!

Source: Intel & ITRS

© 2013 IBM Corporation

Computation-communication convergence

- **Communication == data movement**
  - It's all about the data!
- **Convergence of computation and communication in silicon**
  - Integrated L1/L2/L3 caches; memory controller; PCIe controller; multi-chip interconnect (e.g. HT, QPI replacing FSB); network-on-chip
  - Integrated SMP fabric, I/O hub; NIC/CNA/HCA (taking PCIe out of the loop?); special-purpose accelerators
- **Convergence of multiple traffic types on a common physical wire**
  - LAN, SAN, IPC, IO, management
  - L2 encroaching upon L3 and L4 territory
  - Convergence Enhanced Ethernet

© 2013 IBM Corporation

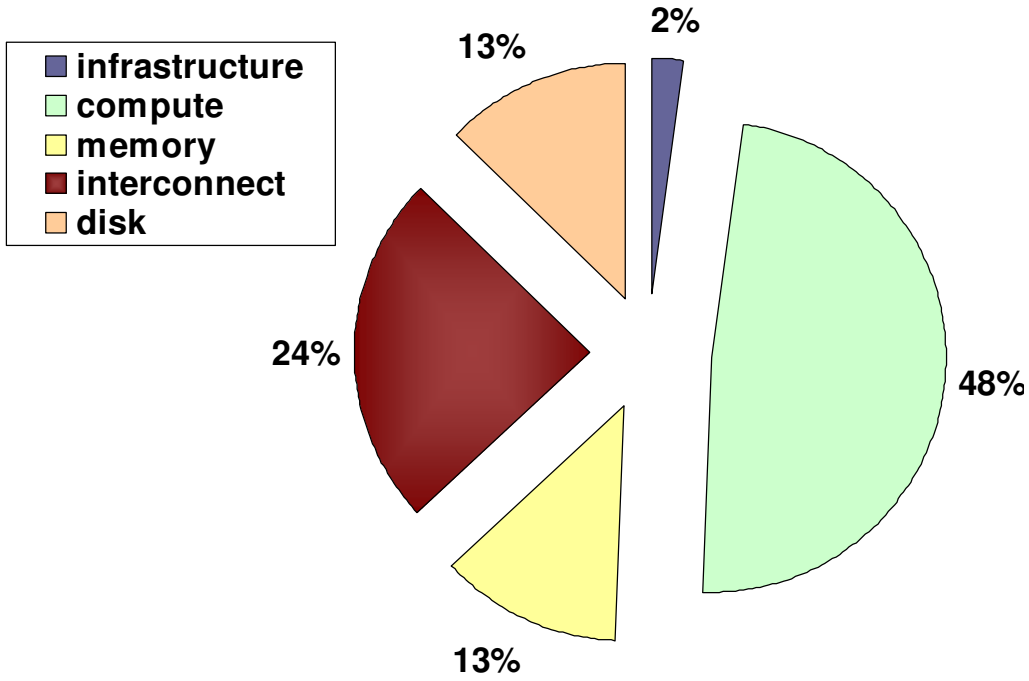
Consequences of scarce bandwidth

- **Performance of communication-intensive applications has scaled poorly**
  - because of lower *global* byte/FLOP ratio
- **Yet *mean* utilization is typically very low**
  - because of synchronous nature of many HPC codes; regularly alternating comp/comm phases
  - massive underutilization for computation-intensive applications (e.g. LINPACK)
- **Full-bisection bandwidth networks are no longer cost-effective**
- **Common practice of separate networks for clustering, storage, LAN has become inefficient and expensive**
  - File I/O and IPC can't share bandwidth
    - I/O-dominated initialization phase could be much faster if it could exploit clustering bandwidth: poor speedup, or even slowdown with more tasks…

© 2013 IBM Corporation

Interconnect becoming a significant cost factor

- Current interconnect cost percentage increases as cluster size increases
- About one quarter of cost due to interconnect for ~1 PFLOP/s peak system



	Fat Tree	*Torus 1*	*Torus 2*	*Torus 3*
Compute	63.3%	72.5%	76.5%	79.7%
Adapters + cable	10.4%	11.9%	12.6%	13.1%
Switches + cables	26.3%	15.6%	10.9%	7.2%
Total	100%	100%	100%	100%

© 2013 IBM Corporation

Current HPC landscape

© 2013 IBM Corporation

Top 500

- **List of 500 highest-performing supercomputers**
  - Published twice a year (June @ ISC, November @ SC)
  - Ranked by maximum achieved GFLOPS for a specific benchmark
  - See www.top500.org
  - "Highest-performing" is measured by sustained Rmax for the LINPACK (HPL) benchmark
  - LINPACK solves a large dense system of linear equations
  - Note that not all supercomputers are listed (not every supercomputer owner care to see their system appear on this list…)

If your problem is not represented by a dense system of linear equations, a system's LINPACK rating is not necessarily a good indicator of the performance you can expect!

© 2013 IBM Corporation

Caveat emptor!

#1: Not all workloads are like LINPACK

#2: FLOPS per se is a meaningless metric

1) HPL is embarrassingly parallel

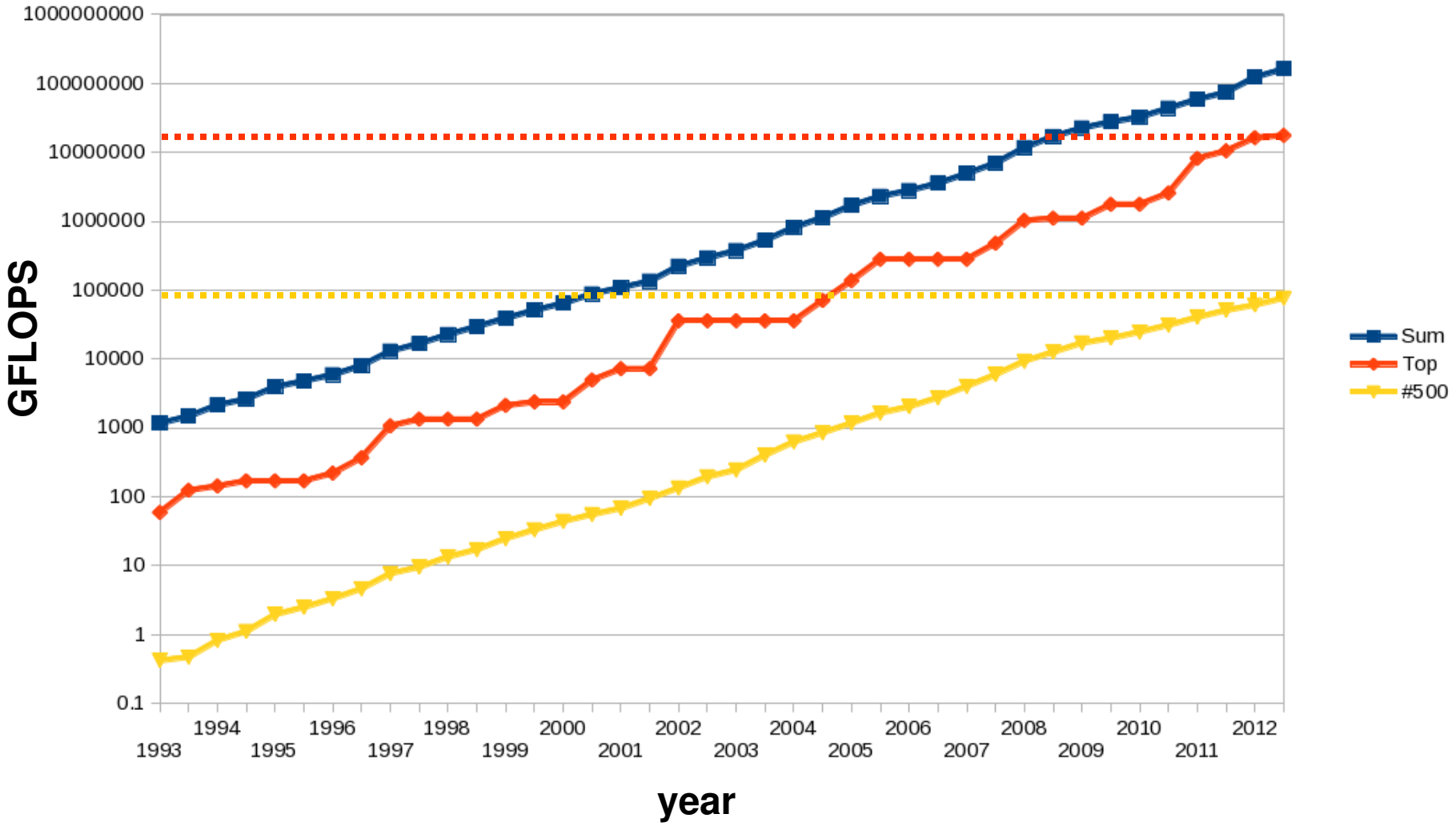
- Very little load on the network
- Highly debatable whether it is a good yardstick at all
- Yet the entire HPC community remains ensnared by its spell

2) FLOPS rating says NOTHING about usefulness or efficiency of computation

- The underlying algorithm is what matters!
- Consider two algorithms solving the same problem:
  - Alg A solves it in t time with f FLOPS
  - Alg B solves it in $10*t$ time with $10*f$ FLOPS
  - A is 100 times more efficient than B, yet has one-tenth of the FLOPS rating

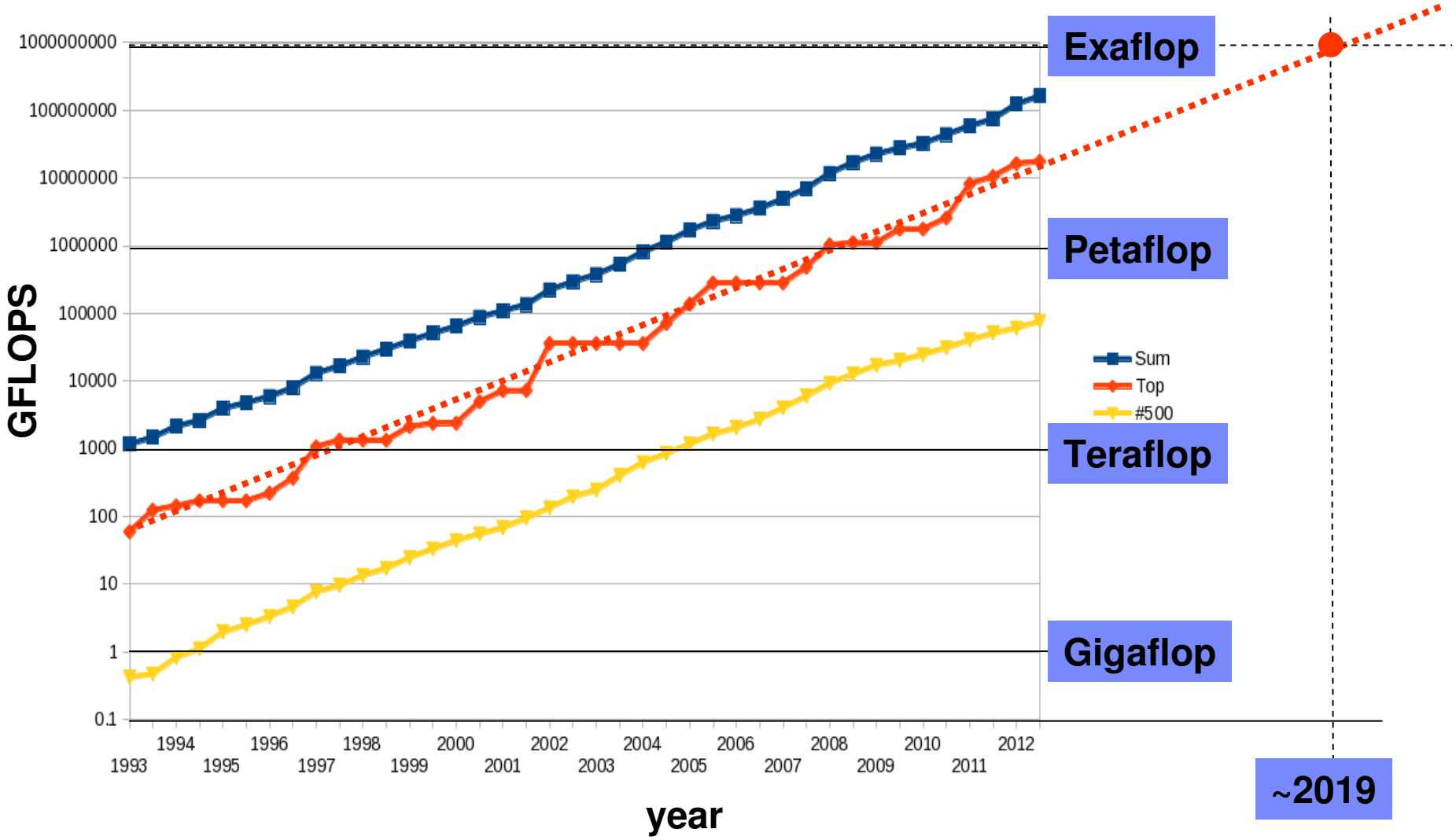
© 2013 IBM Corporation

Top 500 since 1993



© 2013 IBM Corporation

Top 500 since 1993



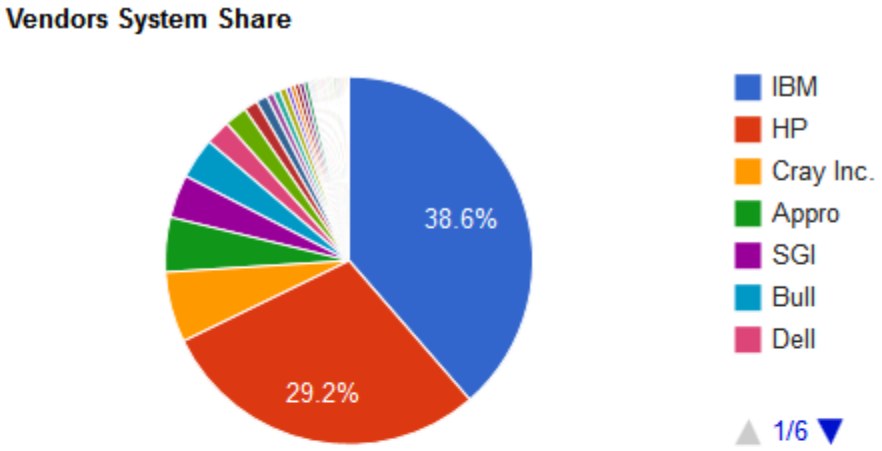
© 2013 IBM Corporation

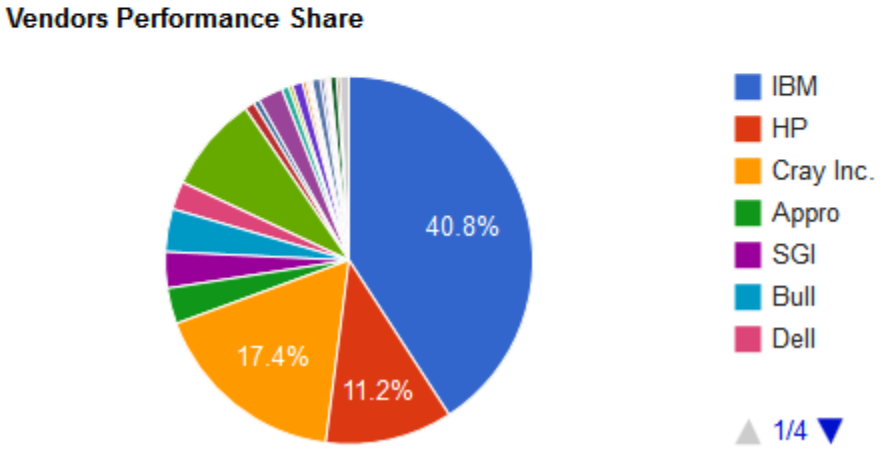
Top 10, November 2012

#	Rmax Rpeak (Pflops)	Name	Computer design Processor type, interconnect	Vendor	Site Country, year	Operating system
1	17.590 27.113	*Titan*	Cray XK7 Opteron 6274 + Tesla K20X, Custom	Cray	Oak Ridge National Laboratory (ORNL) in Tennessee United States, 2012	Linux (CLE, SLES based)
2	16.325 20.133	*Sequoia*	Blue Gene/Q PowerPC A2, Custom	IBM	Lawrence Livermore National Laboratory United States, 2011	Linux (RHEL and CNK)
3	10.510 11.280	*K computer*	RIKEN SPARC64 VIIIfx, Tofu	Fujitsu	RIKEN Japan, 2011	Linux
4	8.162 10.066	*Mira*	Blue Gene/Q PowerPC A2, Custom	IBM	Argonne National Laboratory United States, 2012	Linux (RHEL and CNK)
5	4.141 5.033	*JUQUEEN*	Blue Gene/Q PowerPC A2, Custom	IBM	Forschungszentrum Jülich Germany, 2012	Linux (RHEL and CNK)
6	2.897 3.185	*SuperMUC*	iDataPlex DX360M4 Xeon E5–2680, Infiniband	IBM	Leibniz-Rechenzentrum Germany, 2012	Linux
7	2.660 3.959	*Stampede*	PowerEdge C8220 Xeon E5–2680, Infiniband	Dell	Texas Advanced Computing Center United States, 2012	Linux
8	2.566 4.701	*Tianhe-1A*	NUDT YH Cluster Xeon 5670 + Tesla 2050, Arch[6]	NUDT	National Supercomputing Center of Tianjin China, 2010	Linux
9	1.725 2.097	*Fermi*	Blue Gene/Q PowerPC A2, Custom	IBM	CINECA Italy, 2012	Linux (RHEL and CNK)
10	1.515 1.944	*DARPA Trial Subset*	Power 775 POWER7, Custom	IBM	IBM Development Engineering United States, 2012	Linux (RHEL)

© 2013 IBM Corporation

System/performance share by vendor

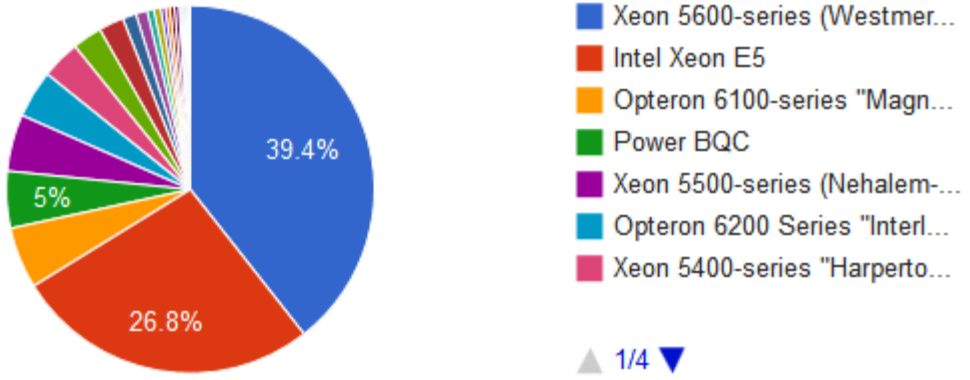




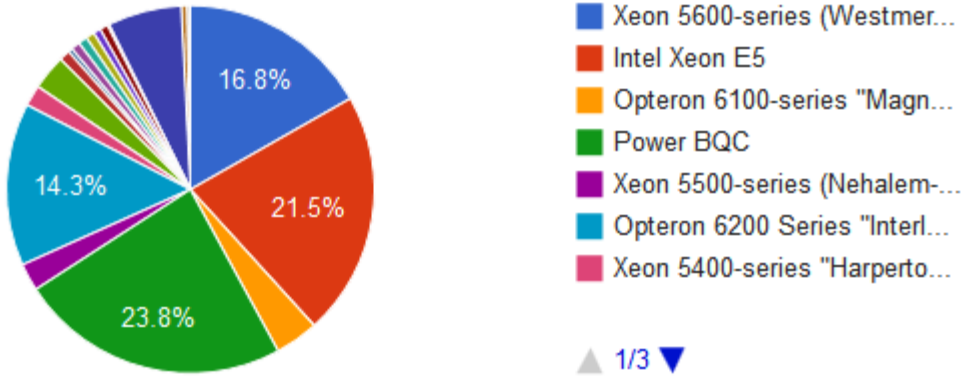
© 2013 IBM Corporation

System/performance share by processor

Processor Generation System Share

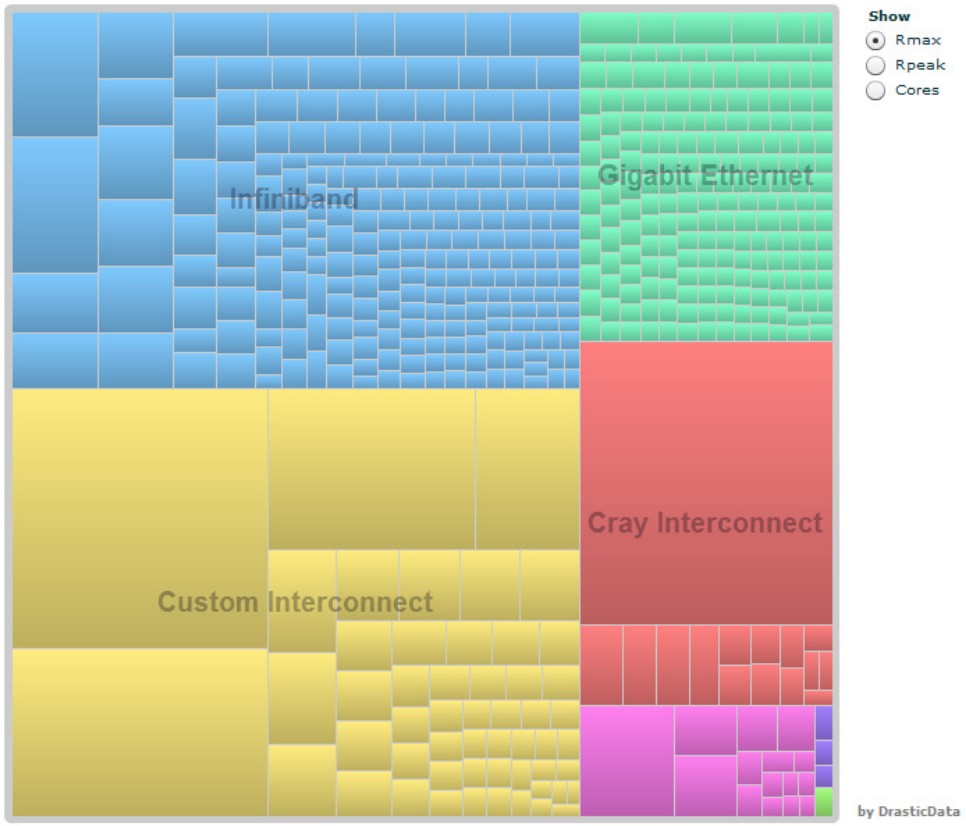


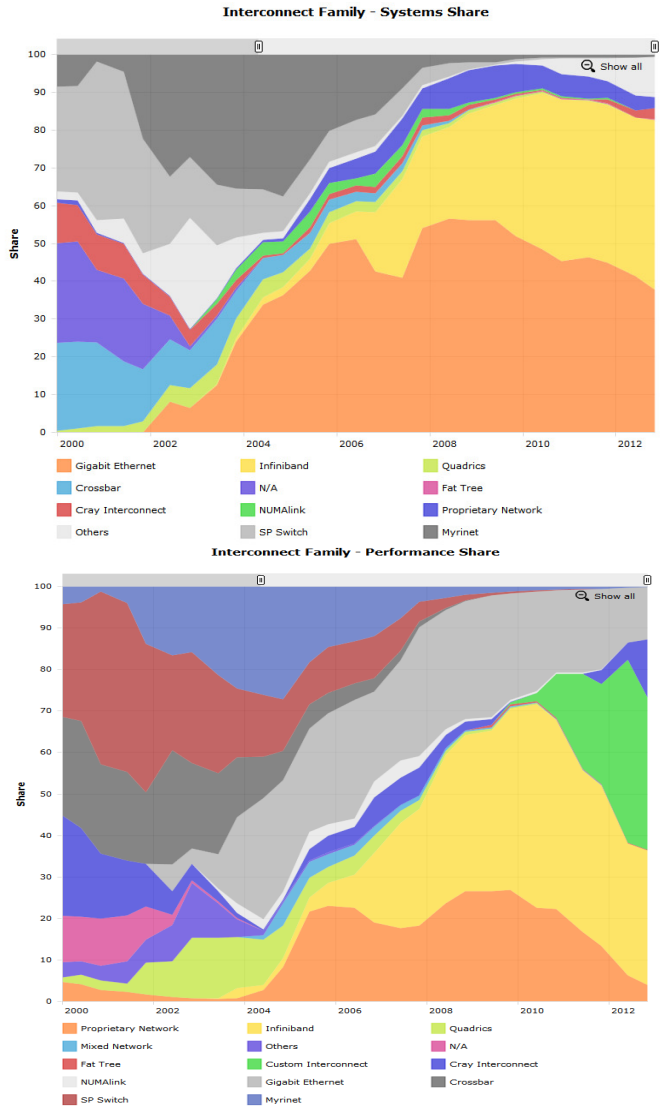
Processor Generation Performance Share



© 2013 IBM Corporation

HPC interconnect landscape, Nov. 2012





Source: www.top500.org

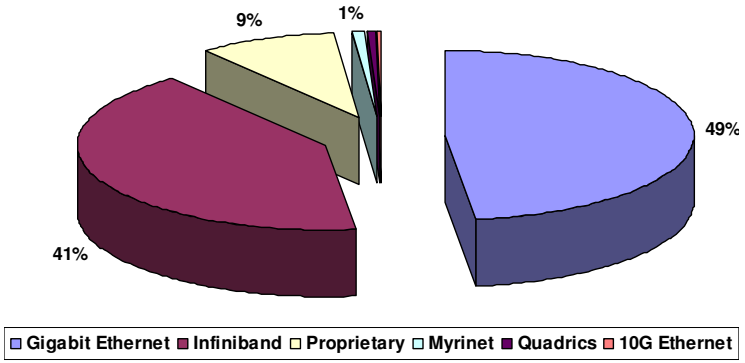
© 2013 IBM Corporation

Top 500 Interconnects

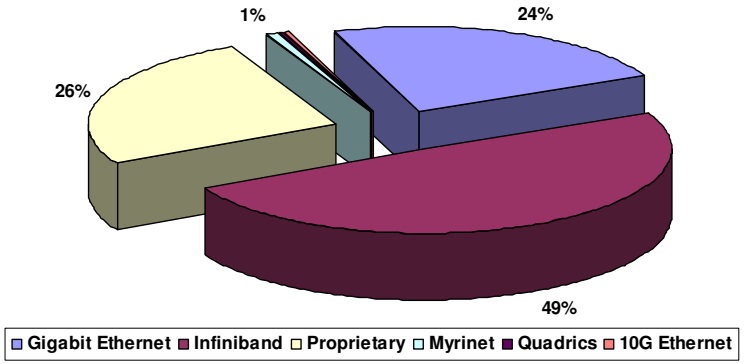
June 2010

- **List of June 2010**
- **Dominated by Ethernet & Infiniband**
  - Ethernet by volume (49%)
  - Infiniband by PFLOPs (49%)
- **Proprietary still plays a significant role**
  - 9% system share, 26% performance share
  - High-end HPC vendors
- **Myrinet, Quadrics dwindling**
- **10 GigE: two installations listed**

Number of systems



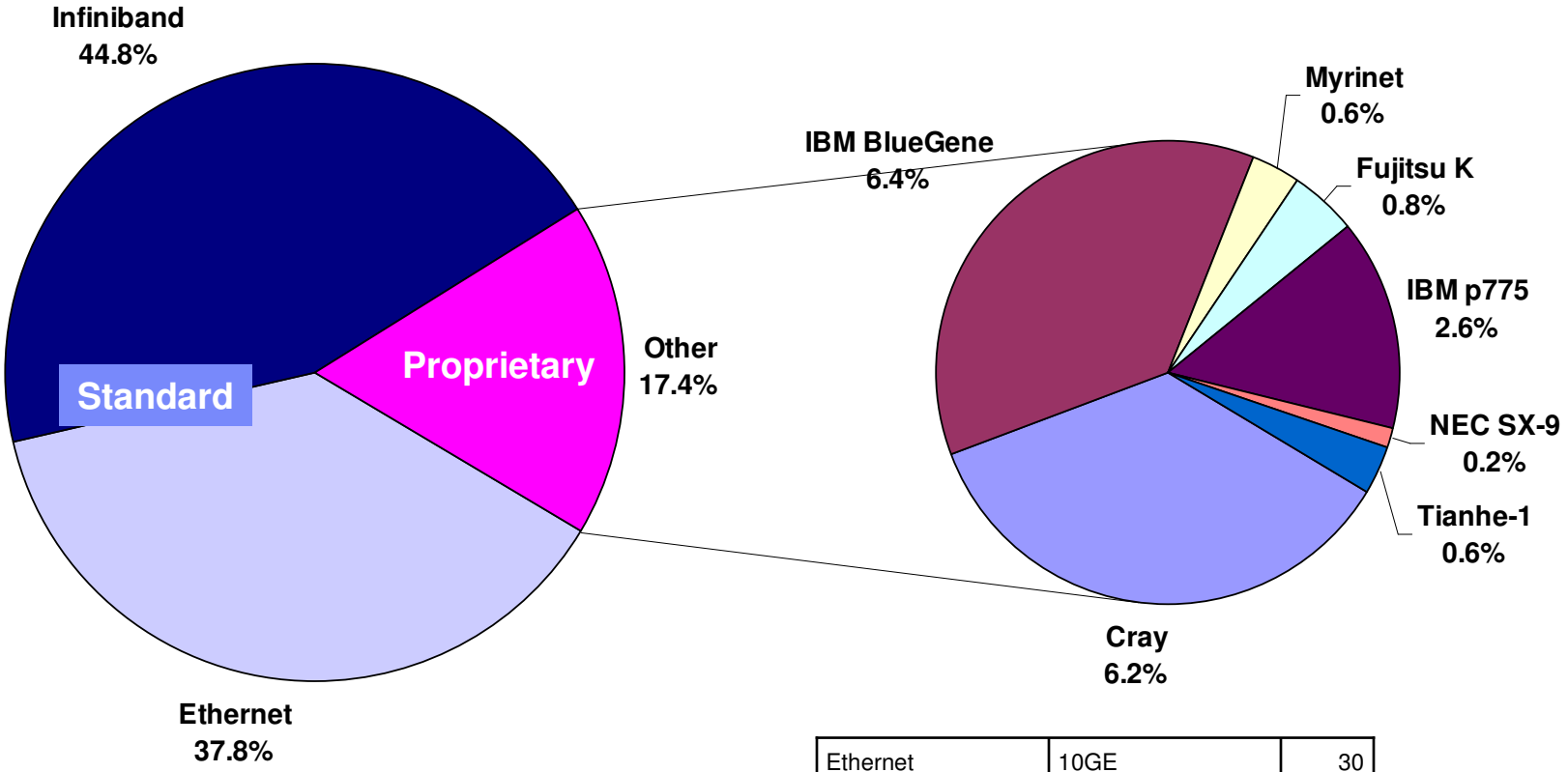
FLOPS (Rmax)



Source: www.top500.org

© 2013 IBM Corporation

Interconnects by system share



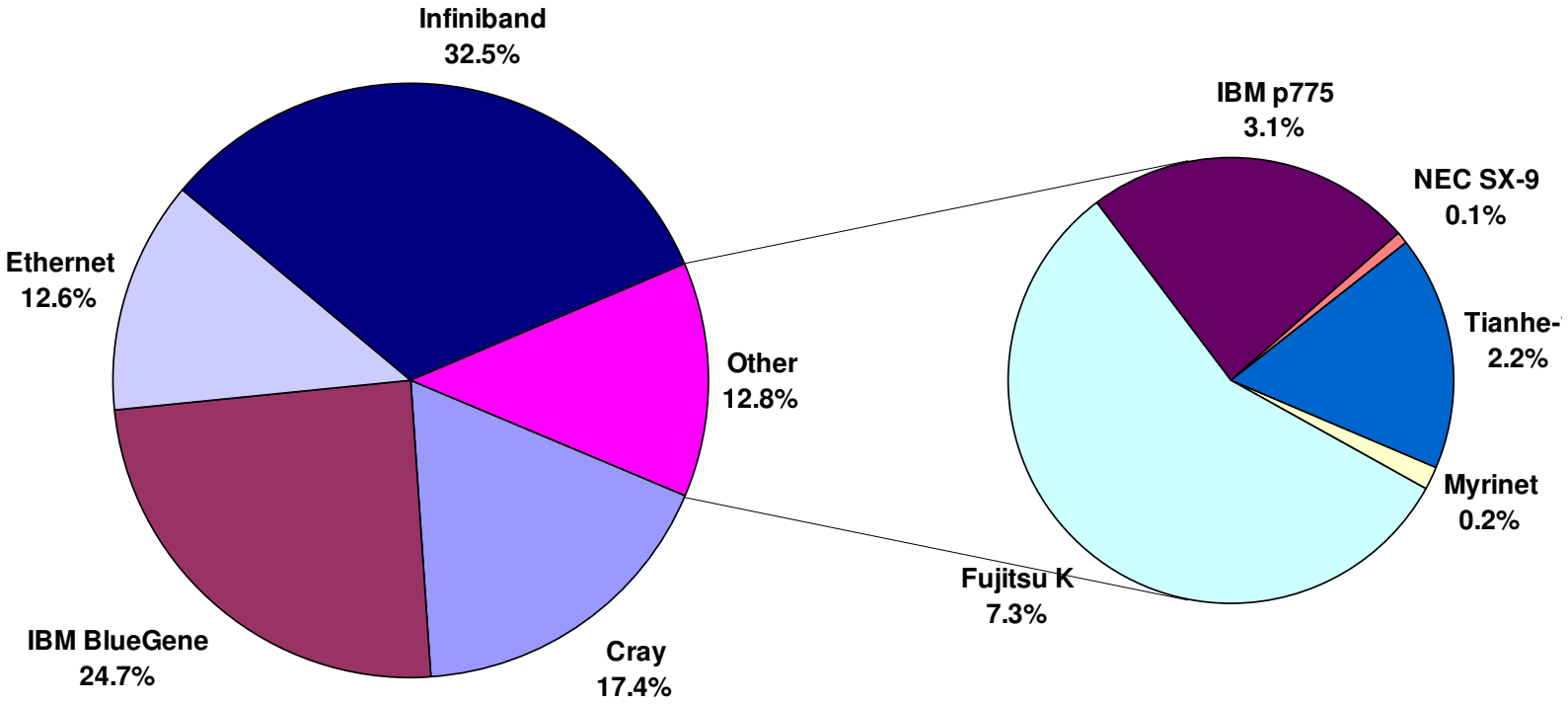
Nov. 2012

Based on Nov. 2012 Top 500 data

Ethernet	10GE	30
	1GE	159
Ethernet Total		**189**
Infiniband	Infiniband	59
	Infiniband DDR	13
	Infiniband FDR	45
	Infiniband QDR	107
Infiniband Total		**224**

© 2013 IBM Corporation

Interconnects by performance share



Nov. 2012

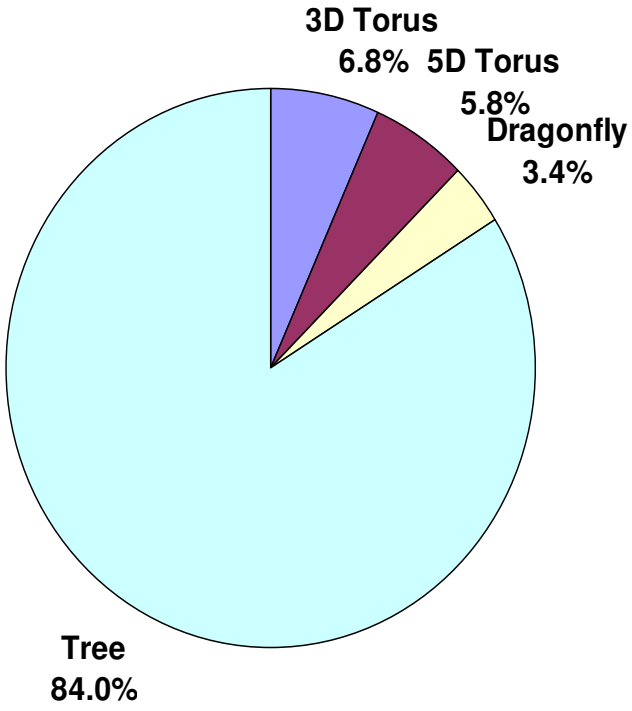
Based on Nov. 2012 Top 500 data

Network	Systems			Performance		
	Jun. '10	Nov. '11	Nov. '12	Jun. '10	Nov. '11	Nov. '12
Ethernet	49%	45%	**38%**	24%	19%	**13%**
Infiniband	41%	42%	**45%**	49%	39%	**32%**
Proprietary	10%	13%	**17%**	27%	42%	**55%**

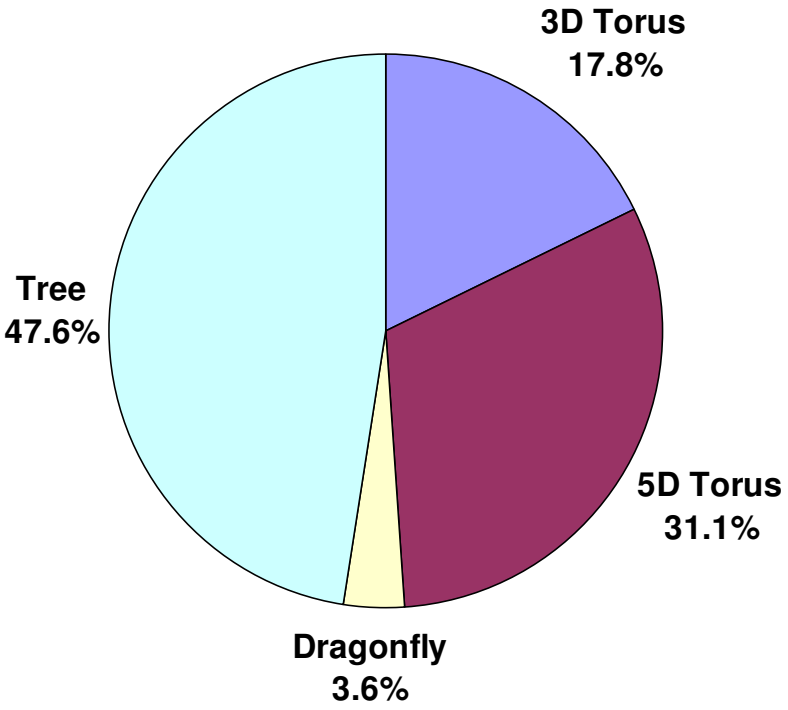
© 2013 IBM Corporation

Interconnects by topology

By system share



By performance share



© 2013 IBM Corporation

Interconnection network basics

© 2013 IBM Corporation

Interconnection network basics

- **Networks connect processing nodes, memories, I/O devices, storage devices**
  - System comprises network nodes (aka switches, routers) and compute nodes
- **Networks permeate the system at all levels**
  - Memory bus
  - On-chip network to connect cores to caches and memory controllers
  - PCIe bus to connect I/O devices, storage, accelerators
  - Storage area network to connect to shared disk arrays
  - Clustering network for inter-process communication
  - Local area network for management and “outside” connectivity
- **We’ll focus mostly on clustering networks here**
  - Don’t equate “clustering network” with “cluster” here
  - These networks tie many nodes together to create one large computer
  - Nodes are usually identical and include their own cores, caches, memory, I/O

© 2013 IBM Corporation

Interconnection network basics

- By now, buses have mostly been replaced by networks, off-chip as well as on-chip
- System performance is increasingly limited by communication instead of computation
- Hence, network has become a key factor determining system performance
- Therefore, we should choose its design wisely
- We review several options here (but not all by far)

© 2013 IBM Corporation

Main design aspects

- **Topology**
  - Rules determining how compute nodes and network nodes are connected
  - Unlike LAN or data center networks, HPC topologies are highly *regular*
  - Interconnection patterns can be described by simple algebraic expressions
  - All connections are full duplex
- **Routing**
  - Rules determining how to get from node *A* to node *B*
  - Because topologies are regular & known, routing algorithms can be designed a priori
  - Source- vs. table-based; direct vs. indirect; static vs. dynamic; oblivious vs. adaptive
- **Flow control**
  - Rules governing *link traversal*
  - Deadlock avoidance

© 2013 IBM Corporation

Interconnect classification

- **Direct vs. indirect**
  - Direct network: Each network node attaches to at least one compute node.
  - Indirect network: Compute nodes are attached at the edge of the network only; clear boundary between compute nodes & network nodes; many routers only connect to other routers.
- **Discrete vs. integrated**
  - Discrete: network nodes are physically separate from compute nodes
  - Integrated: network nodes are integrated with compute nodes
- **Standard vs. proprietary**
  - Standard ("open"): Network technology is compliant with a specification ratified by standards body (e.g., IEEE)
  - Proprietary ("closed"): Network technology is owned & manufactured by one specific vendor

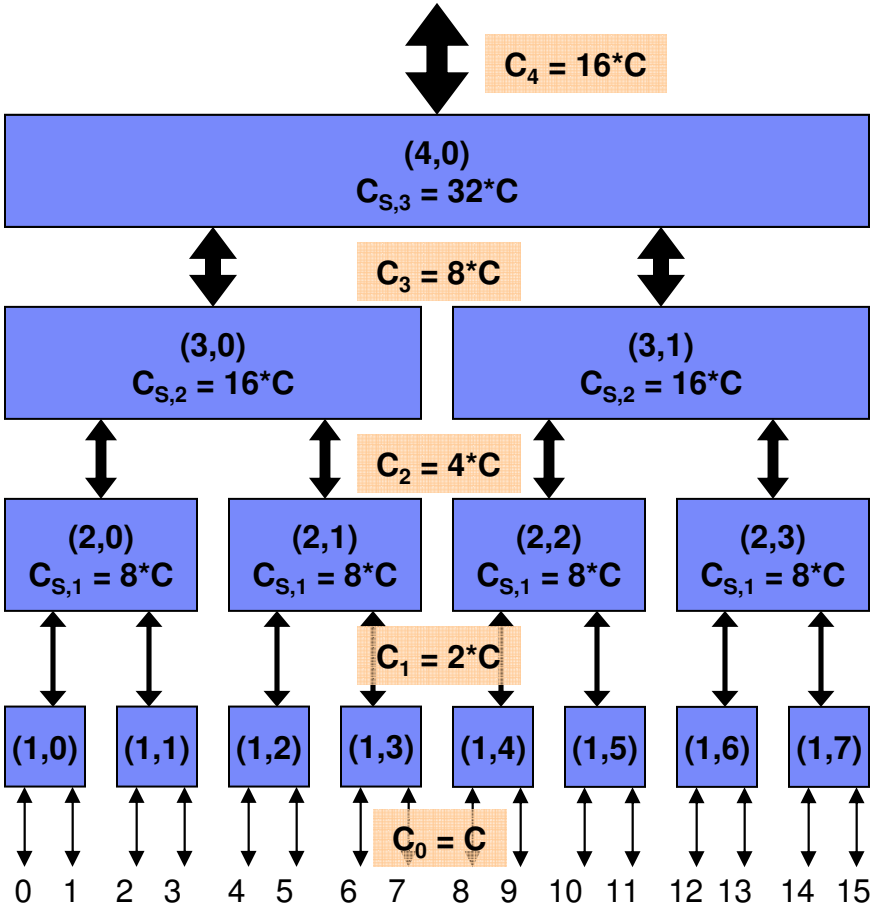
© 2013 IBM Corporation

Topologies

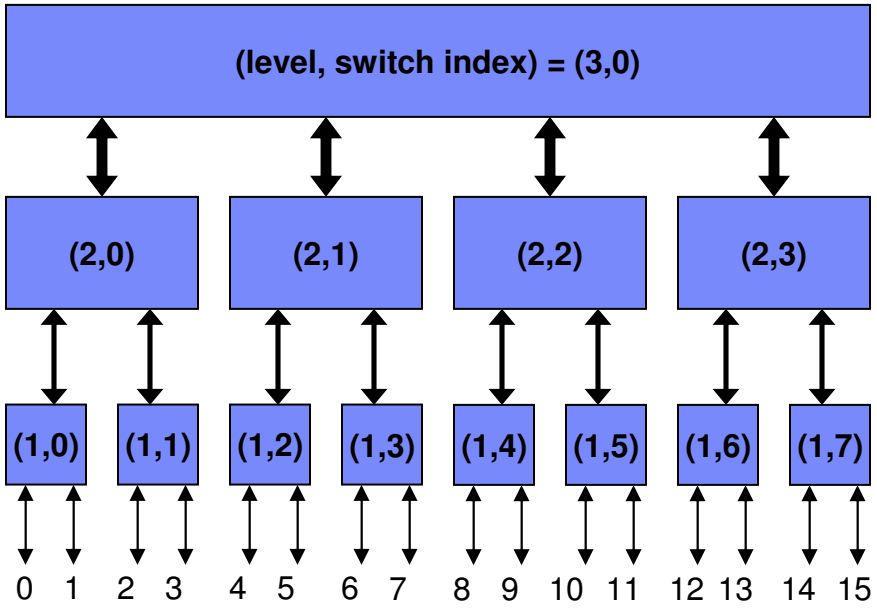
- **Trees [indirect]**
  - Fat tree
  - *k*-ary *n*-tree
  - Extended generalized fat tree (XGFT)
- **Mesh & torus [direct]**
  - *k*-ary *n*-mesh
  - *k*-ary *n*-cube
- **Dragonfly [direct]**
- **HyperX/Hamming Graph [direct]**

© 2013 IBM Corporation

The origin of the term “fat tree”



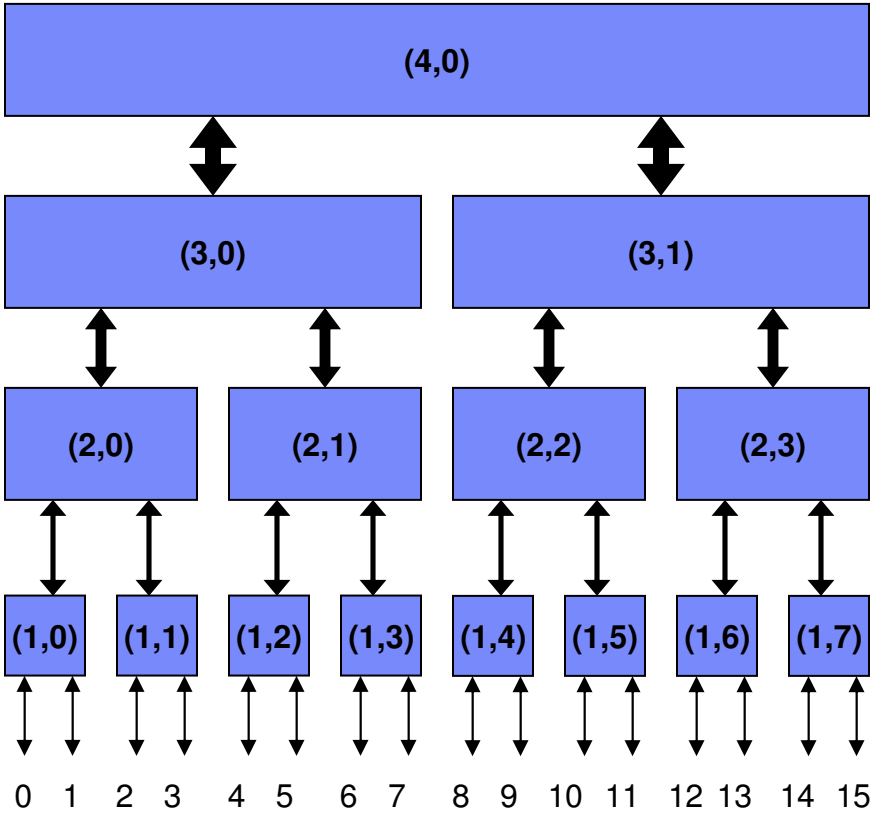
- First described by C. Leiserson, 1985.
- A fat tree of height m and arity k has k^m end nodes and k^{m-j} switches in level j ($1 \le j \le m$)
- Each switch has at level j has k “down” ports with capacity $C_{j-1} = k^{j-1}*C$ and 1 “up” port with capacity $C_j = k^j*C$

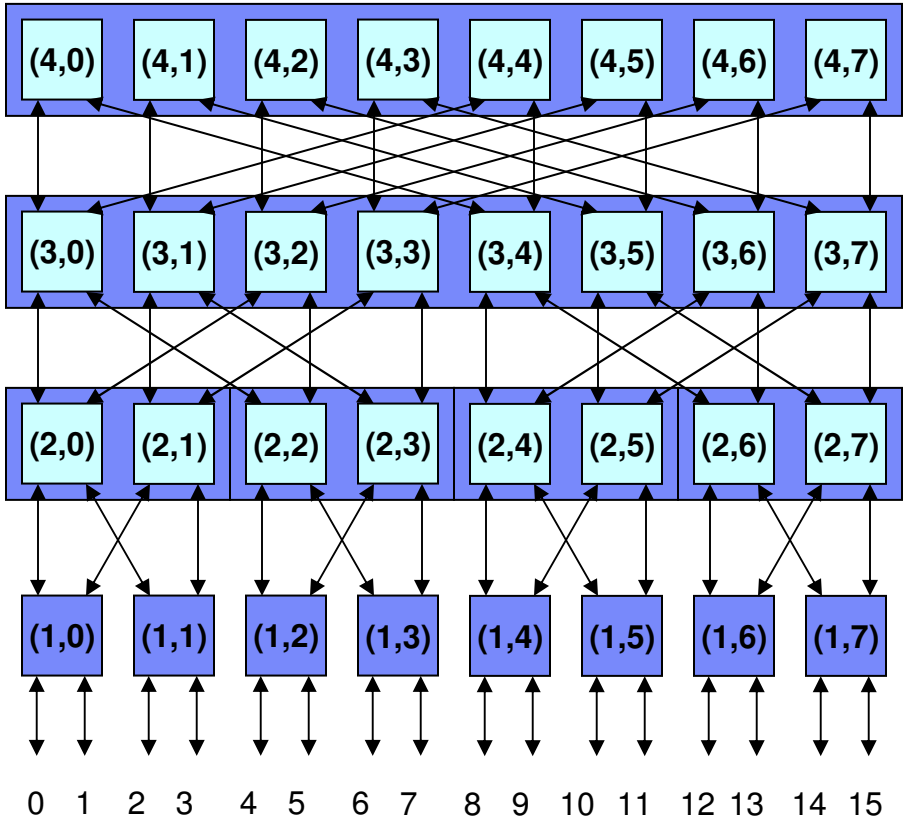


- **A fat tree’s main characteristic: Constant bisectional bandwidth**
- **Achieved by increasingly “fatter” links in subsequent levels**
- **Top level can be eliminated if further expansion not desired (“double sized”)**

© 2013 IBM Corporation

Transmogrification into *k*-ary *m*-tree



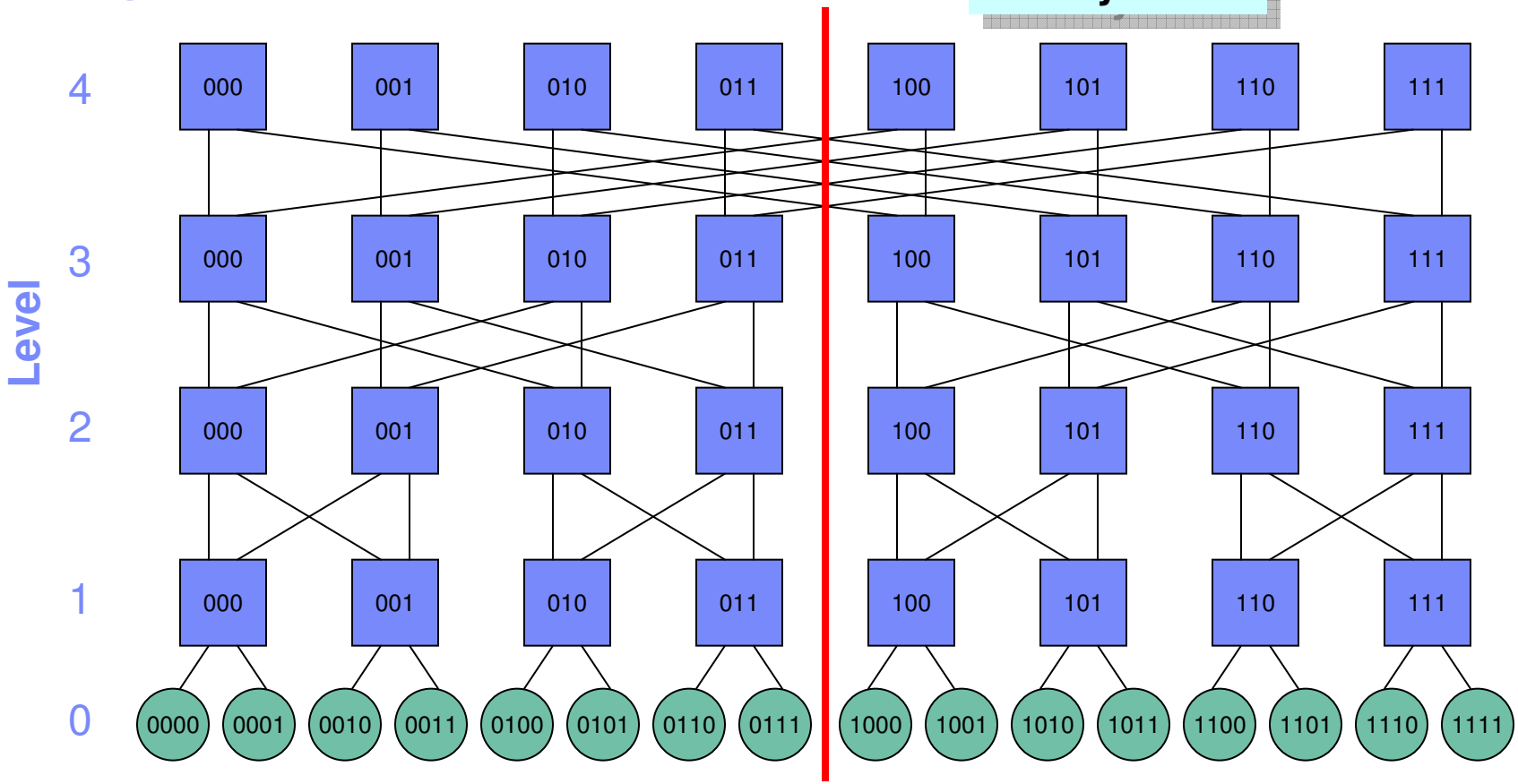


- **Switch capacity doubles with each level**
  - Constant radix, but increasing link speed
  - Not feasible in practice for large networks
  - Routing is trivial (single path)

- **Construct fat tree from fixed-capacity switches**
  - Constant radix, constant link speed
  - Requires specific interconnection pattern and routing rules (multi-path)
  - Top-level switches radix is *k*/2 (without expansion)

© 2013 IBM Corporation

k-ary *n*-tree network



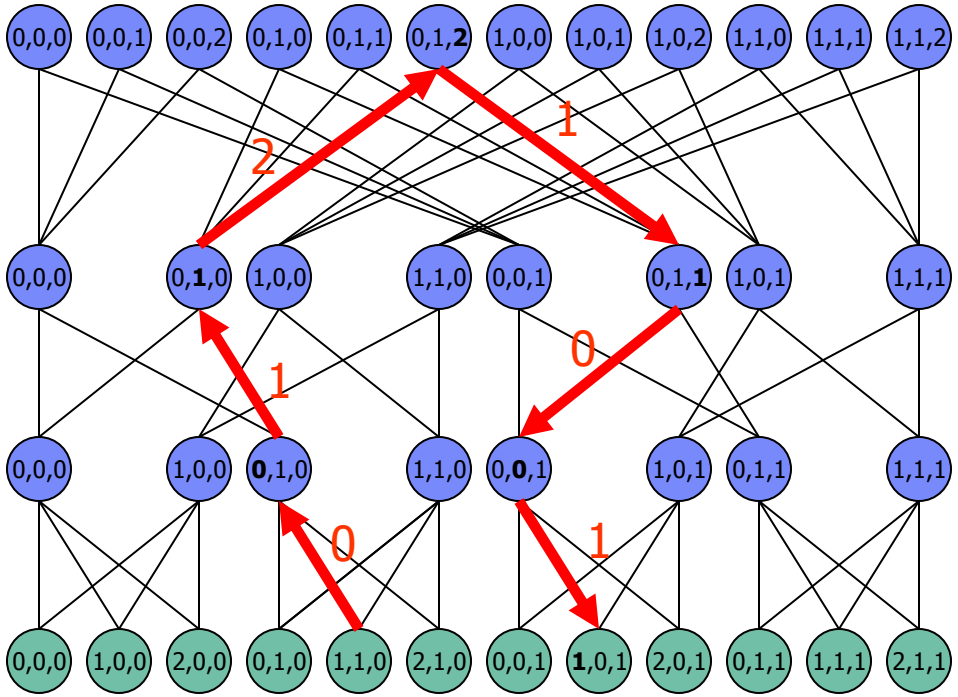
- $N = k^n$ end nodes
- $nk^{(n-1)}$ switches arranged in n stages
- $(nk^n)/2$ inter-switch links

- Diameter = $2n$-1 switch hops
- Bisection = $B_{bis} = ½k^n$ bidir links
- Relative bisection = $B_{bis}/(N/2) = 1$

© 2013 IBM Corporation

Extended Generalized Fat Tree

- **XGFT (h ; m_1, … , m_h; w_1, … , w_h)**
- **h = height**
  - number of levels-1
  - levels are numbered 0 through h
  - level 0 : compute nodes
  - levels 1 … h : switch nodes
- **m_i = number of children per node at level i, $0 < i \leq h$**
- **w_i = number of parents per node at level i-1, $0 < i \leq h$**
- **number of level 0 nodes = $\prod_i m_i$**
- **number of level h nodes = $\prod_i w_i$**



XGFT (3 ; 3, 2, 2 ; 2, 2 ,3)

number of children per level

number of parents per level

© 2013 IBM Corporation

Fat tree example: Roadrunner

- **First system to break the petaflop barrier**
  - Operational in 2008, completed in 2009
  - National Nuclear Security Administration, Los Alamos National Laboratory
  - Used to modeling the decay of the U.S. nuclear arsenal
- **First hybrid supercomputer**
  - 12,960 IBM **PowerXCell 8i** CPUs
  - 6,480 AMD Opteron dual-core processors
  - InfiniBand interconnect
- **Numbers**
  - Power 2.35 MW
  - Space 296 racks, 560 m2
  - Memory 103.6 TiB
  - Storage 1,000,000 TiB
  - Speed 1.042 petaflops
  - Cost US$100 million

© 2013 IBM Corporation

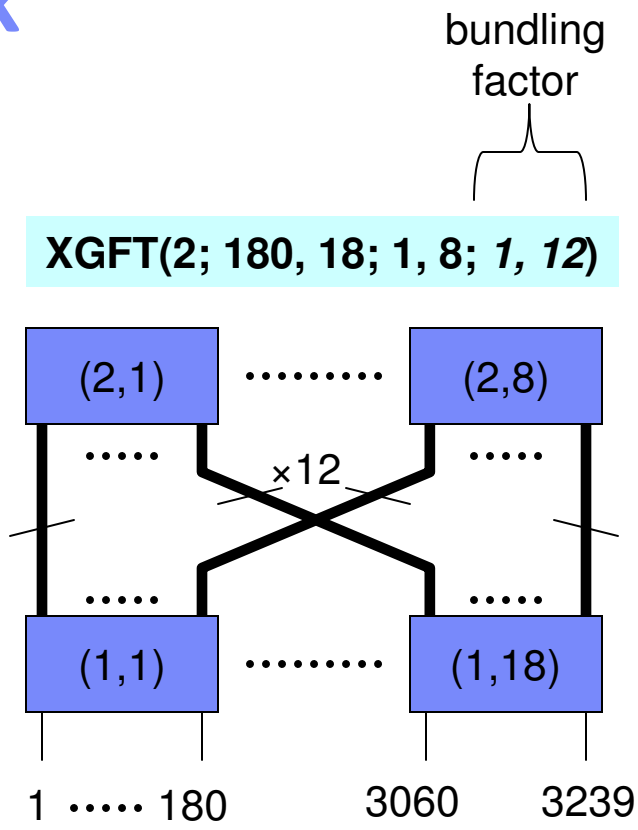
Cell BE commercial application

Source: Sony

© 2013 IBM Corporation

Roadrunner fat tree network

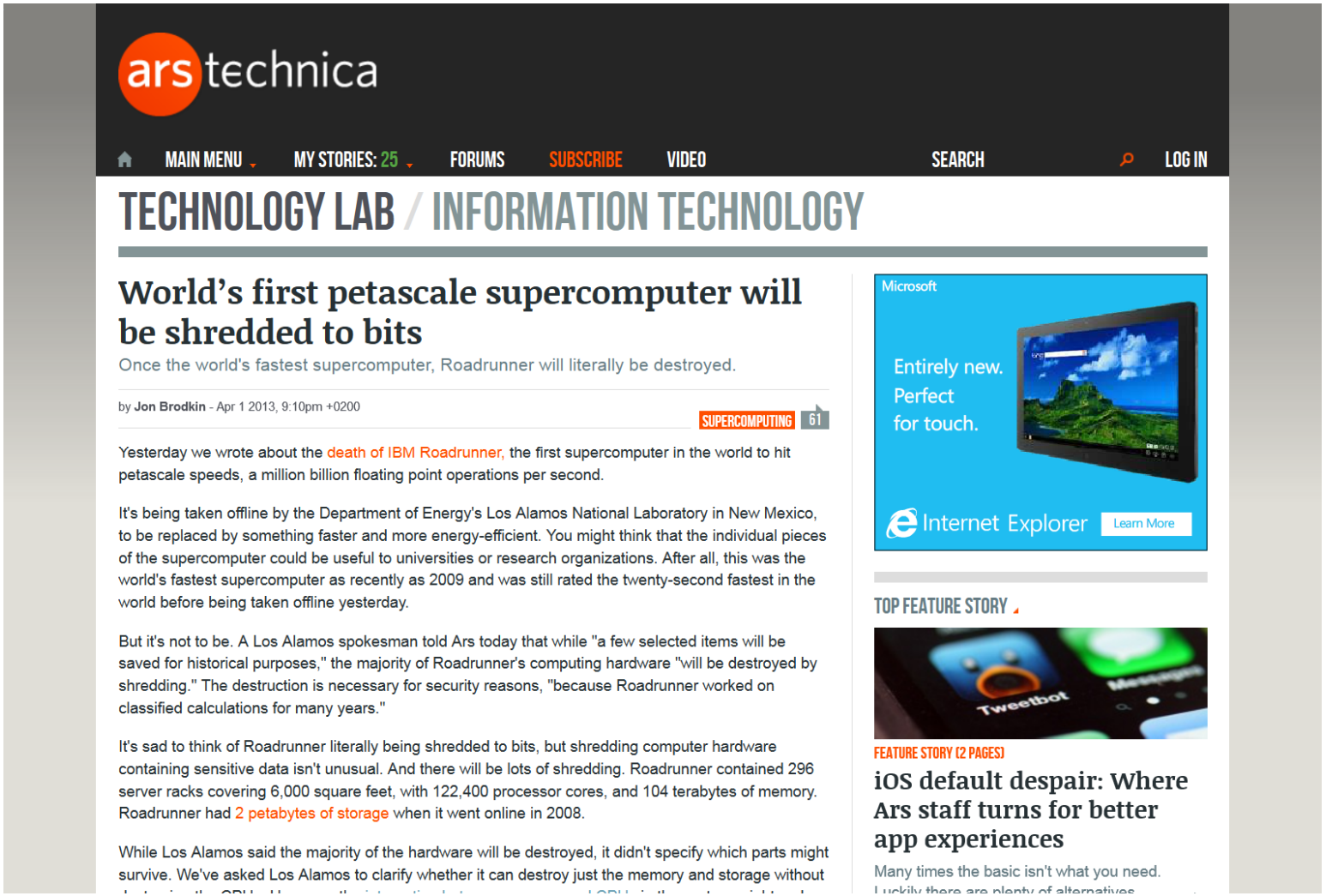
- **"TriBlade" hybrid compute node**
  - One IBM LS21 AMD Opteron blade: 2 dual-core AMD Opteron, 32 GB RAM
  - Two IBM QS22 Cell BE blades: 2x2 3.2 GHz IBM PowerXCell 8i processors, 32 GB RAM, 460 GFLOPS (SP)
- **One Connected Unit (CU) = 180 TriBlades**
- **Full system = 18 CUs = 3,240 compute nodes = 6,480 Opterons + 12,960 Cells**
- **InfiniBand 4x DDR network (2 GB/s/port) two-stage fat tree**
  - Stage 1: 18 switches: 180 ports down, 8x12 ports up
  - Stage 2: 8 switches, 18x12 ports down



***Slimmed* fat tree: half bisection bandwidth @ 2nd level**

© 2013 IBM Corporation

Ashes to ashes…



ars technica

MAIN MENU · MY STORIES: 25 · FORUMS · SUBSCRIBE · VIDEO · SEARCH · LOG IN

TECHNOLOGY LAB / INFORMATION TECHNOLOGY

World's first petascale supercomputer will be shredded to bits

Once the world's fastest supercomputer, Roadrunner will literally be destroyed.

by **Jon Brodkin** - Apr 1 2013, 9:10pm +0200

SUPERCOMPUTING 61

Yesterday we wrote about the death of IBM Roadrunner, the first supercomputer in the world to hit petascale speeds, a million billion floating point operations per second.

It's being taken offline by the Department of Energy's Los Alamos National Laboratory in New Mexico, to be replaced by something faster and more energy-efficient. You might think that the individual pieces of the supercomputer could be useful to universities or research organizations. After all, this was the world's fastest supercomputer as recently as 2009 and was still rated the twenty-second fastest in the world before being taken offline yesterday.

But it's not to be. A Los Alamos spokesman told Ars today that while "a few selected items will be saved for historical purposes," the majority of Roadrunner's computing hardware "will be destroyed by shredding." The destruction is necessary for security reasons, "because Roadrunner worked on classified calculations for many years."

It's sad to think of Roadrunner literally being shredded to bits, but shredding computer hardware containing sensitive data isn't unusual. And there will be lots of shredding. Roadrunner contained 296 server racks covering 6,000 square feet, with 122,400 processor cores, and 104 terabytes of memory. Roadrunner had 2 petabytes of storage when it went online in 2008.

While Los Alamos said the majority of the hardware will be destroyed, it didn't specify which parts might survive. We've asked Los Alamos to clarify whether it can destroy just the memory and storage without

Microsoft
Entirely new. Perfect for touch.
Internet Explorer Learn More

TOP FEATURE STORY

FEATURE STORY (2 PAGES)

iOS default despair: Where Ars staff turns for better app experiences

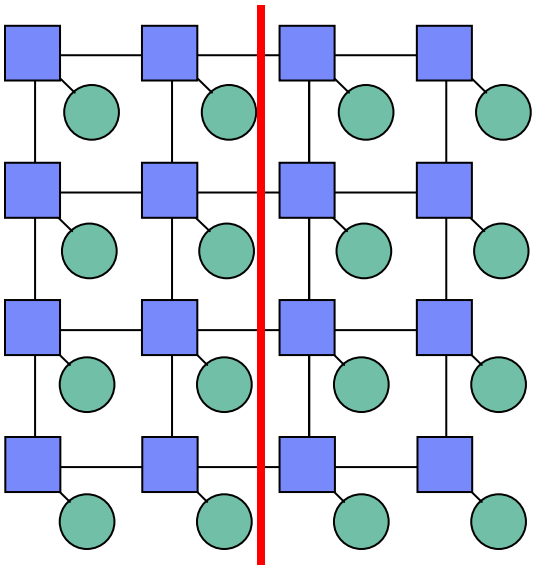
Many times the basic isn't what you need.

Source: arstechnica.com

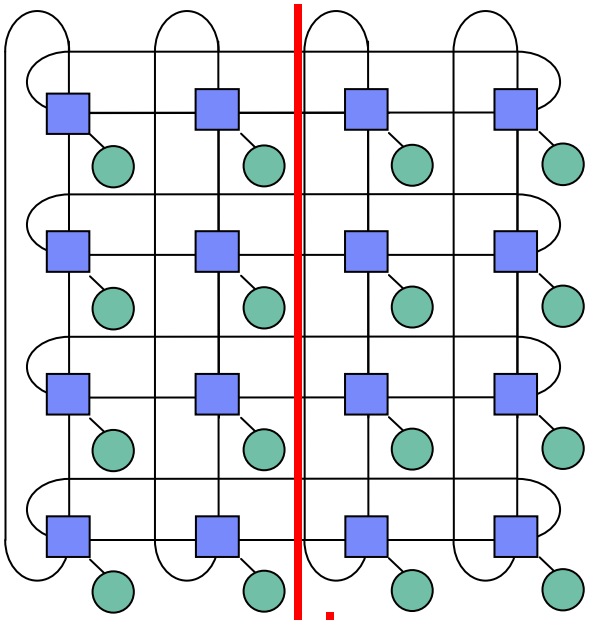
© 2013 IBM Corporation

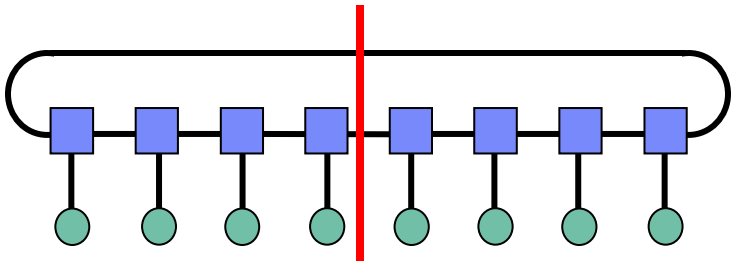
Mesh and torus

2D ***mesh*** : 4-ary 2-mesh

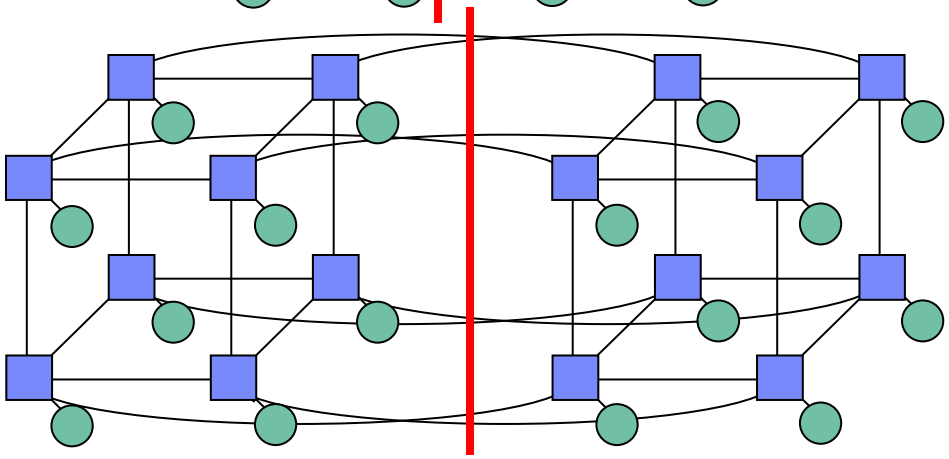


2D ***torus*** : 4-ary 2-cube





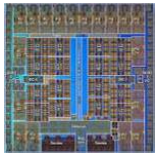
Ring : 8-ary 1-cube



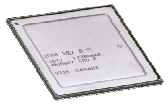
Hypercube : 2-ary 4-cube

Blue Gene/Q packaging

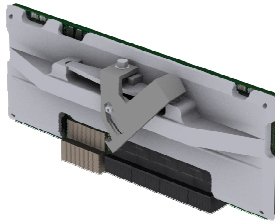
1. Chip
16+2 cores



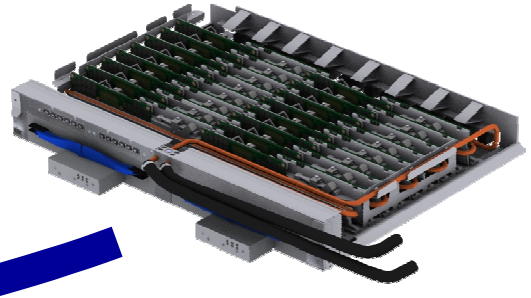
2. Module
Single chip



3. Compute Card
One single chip module,
16 GB DDR3 memory



4. Node Card
32 Compute Cards,
Optical Modules, Link Chips,
2x2x2x2x2 5D Torus



5b. I/O Drawer
8 I/O Cards
8 PCIe Gen2 slots

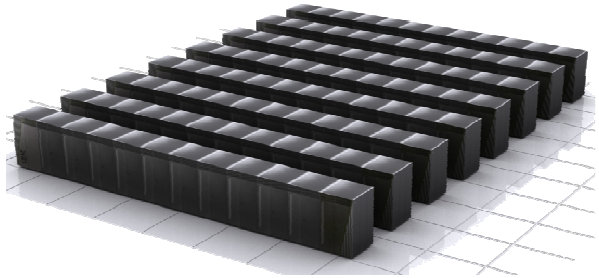
5a. Midplane
16 Node Cards

4x4x4x4x2

6. Rack
2 Midplanes
1, 2 or 4 I/O Drawers

4x4x4x8x2

7. System
96 racks, 20 PF/s



Sustained single node perf: 10x BG/P, 20x BG/L
MF/Watt: > 2 GFLOPS per Watt (6x BG/P, 10x BG/L)

Full system: 5D Torus 12x16x16x16x2

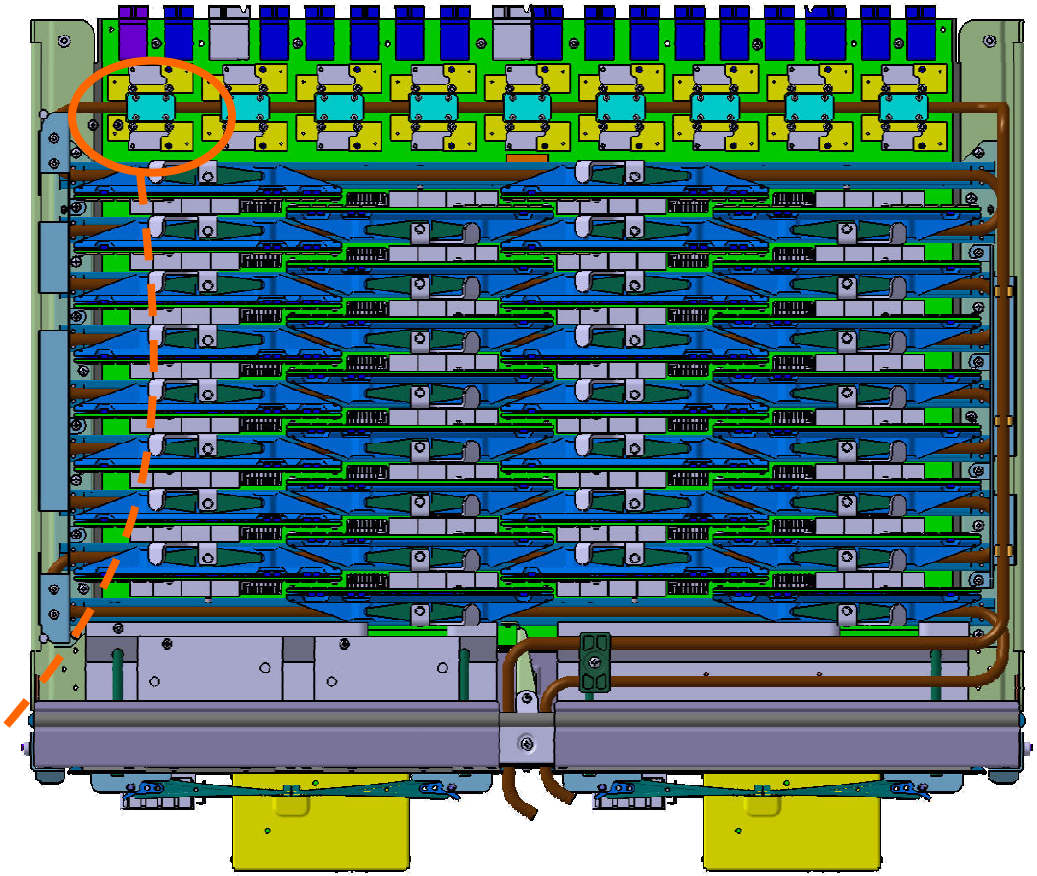
© 2013 IBM Corporation

BlueGene/Q

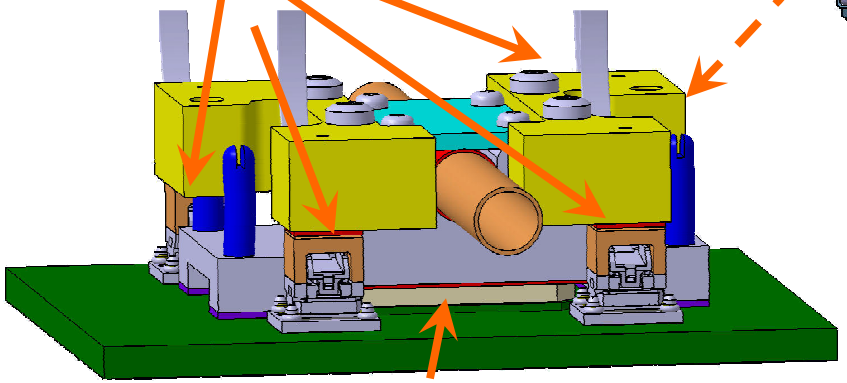
9 link modules, each having 1 link chip + 4 optical modules; total 384 fibers (256 torus + 64 ECC + 64 spare)

32 compute cards (nodes) forming a 2x2x2x2x2 torus

Node board



Optical modules (4x)



Link module

1 optical module = 12 TX + 12 RX fibers @ 10 Gb/s
with 8b/10b coding: eff. 1 GB/s per fiber
8 fibers for 4 external torus dimensions (2 GB/s/port)
2 fibers for ECC (1 per group of 4 fibers)
2 spares
1 link module = 4 optical modules

Courtesy of Todd Takken

© 2013 IBM Corporation

BG/Q Sequoia, Top500 #2

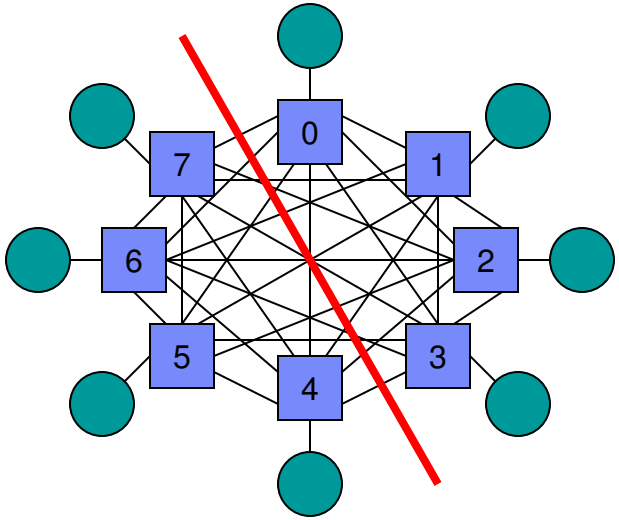
Source: LLNL

- **1,572,864 cores**
- **17 PF sustained performance (20 PF peak)**
- **8 MW power consumption**

© 2013 IBM Corporation

Full mesh aka "complete graph"

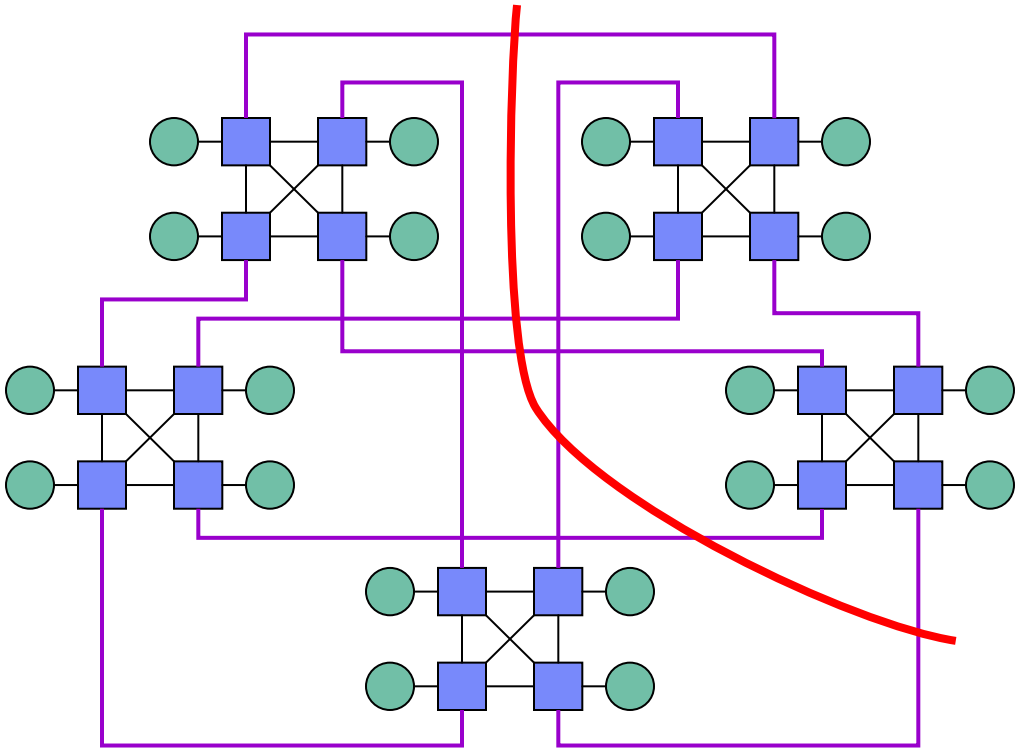
- **Fully connected or Complete Graph K_n: all nodes directly connected to all other nodes by bidirectional dedicated links**
  - *R* routers and *R*(*R*-1)/2 bidirectional links
  - Bisection bandwidth = $R^2/4$ if *R* even or $(R^2-1)/4$ if *R* odd
  - Assuming *n* = *R* and random traffic, $R^2/4$ packets in each direction for a total of $R^2/2$ bisection traffic



© 2013 IBM Corporation

Dragonfly topology

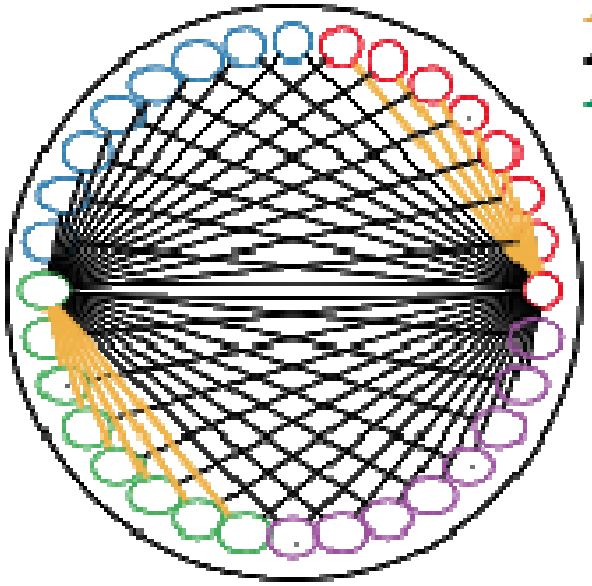
- **Hierarchical network**
  - Full mesh of routers at 1st level
  - Full mesh of groups at 2nd level
- **DF(*p*, *a*, *h*)**
  - *p*: #nodes/router
  - *a*: #routers/group
  - *h*: #remote links/router
  - #groups $ah+1$
  - #routers $a(ah+1)$
  - #nodes $pa(ah+1)$
  - #links $a(ah+1)(a-1+h)/2$
  - Diameter 3
  - Bisection $\sim((ah+1)^2-1)/4$



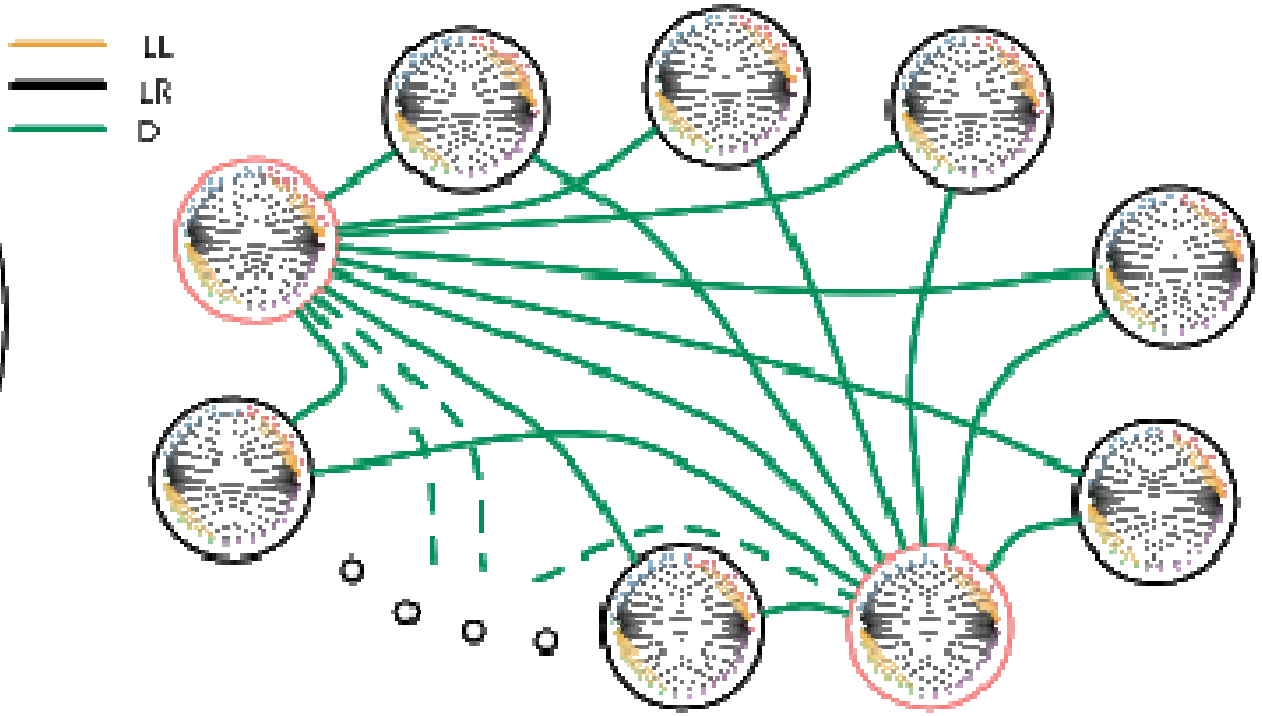
DF(1, 4, 1)

© 2013 IBM Corporation

HPCS PERCS topology (IBM p775)



One supernode of the PERCS topology

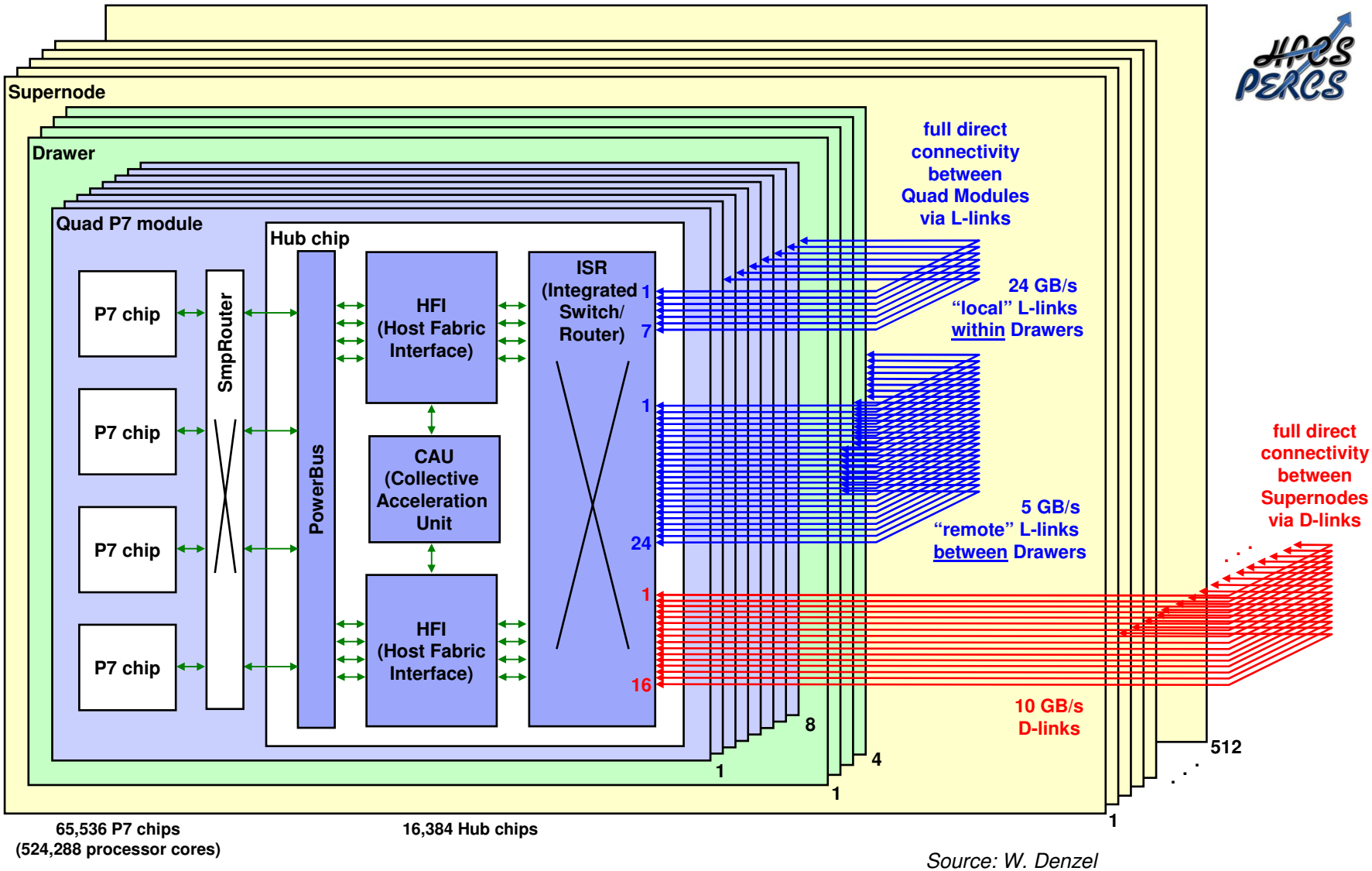


DF(4, 32, 16)

- 8 cores/processor
- *p* = 4 processors/router
- *a* = 32 routers/group (group = "supernode")
- *h* = 16 remote links/router
- Total #routers = 32*(32*16[+1]) = 16,416 [16,384]
- Total #processors = 4*32*(32*16[+1]) = 65,664 [65,536]
- Total #cores = 524,288

© 2013 IBM Corporation

HPCS PERCS interconnect (IBM p775)

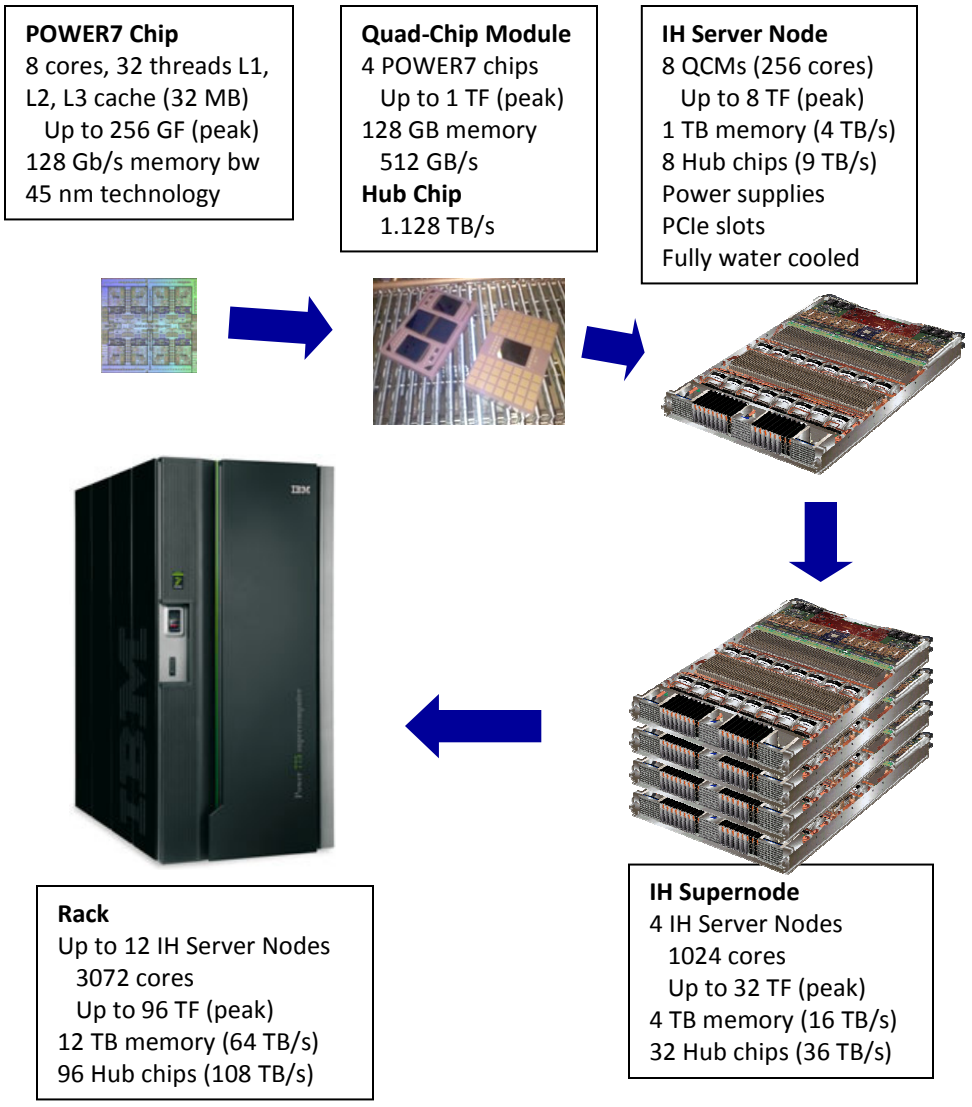


Source: W. Denzel

© 2013 IBM Corporation

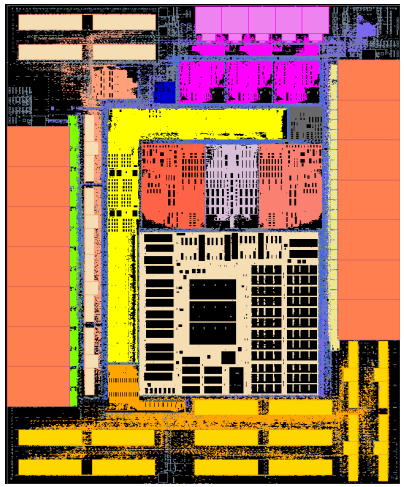
IBM Power 775 (aka P7-IH, PERCS, TSFKABW)

- **Packaging**
  - 1 P7 = 8 cores @ 3.84 GHz
  - 1 quad = 4 P7 = 32 cores
  - 1 2U drawer = 8 quads = 32 P7 = 256 cores
  - 1 supernode = 4 drawers = 32 quads = 128 P7 = 1'024 cores
  - 1 rack = 3 supernodes = 3'072 cores
  - 1 full-scale PERCS system = ~171 racks = 512 supernodes = 524'288 cores
- **Drawer is smallest unit**
  - 8 MCMs with 4 P7 chips each
  - 8 hub modules with optical modules & fiber connectors
  - Memory DIMMs (up to 2 TB)
  - Power, cooling
- **Integrated network**
  - MCM including 4 P7 chips plus hub chip
  - No external adapters or switches
- **96 TFLOPs per rack**
  - 24 TB memory
  - 230 TB storage
  - Watercooled

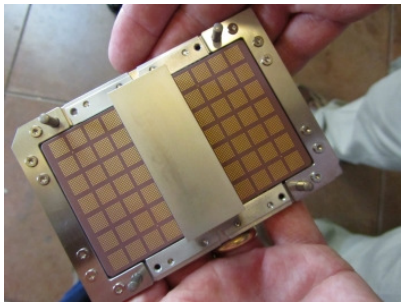


© 2013 IBM Corporation

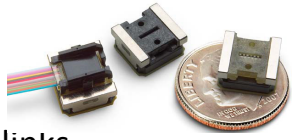
p775 hub chip & module



- **Hub chip for POWER7-based HPC machines**
  - Coherency bus for 4 POWER7 chips (“quad”)
  - Integrated adapter: Host Fabric Interface (HFI)
  - Integrated memory controllers
  - Integrated Collective Acceleration Unit (CAU)
  - Integrated PCIe controllers
  - Integrated switch/router (ISR)
  - Hardware acceleration (CAU, global shared memory, RDMA)

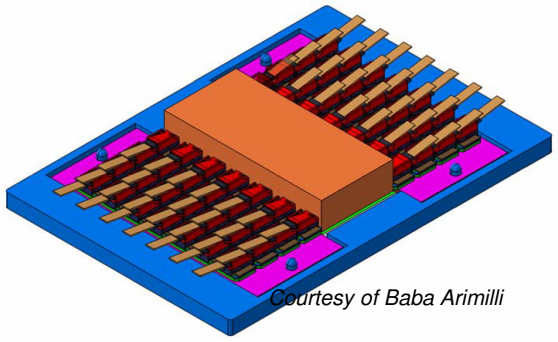


- **56-port integrated switch**
  - 7 electrical intra-drawer links (LL) to all other hubs within same drawer @ (24+24) GB/s per link
  - 24 optical inter-drawer links (LR) to all other hubs within other drawers of same supernode @ (5+5) GB/s per link
  - 16 optical inter-supernode links to hubs in other supernodes @ (10+10) GB/s per link



Source: Avago

- **336 × 10 Gb/s optical links in 56 modules**
  - 28 Tx/Rx pairs × 12 fibers/module = 336 bidir links
  - Avago MicroPOD™ 12× 10 Gb/s with PRIZM™ Light-Turn® optical connector
  - 8b/10b coding ➔ (2 × 336 × 10 × 8/10)/8 = 672 GB/s of aggregate optical link bandwidth



Courtesy of Baba Arimilli

© 2013 IBM Corporation

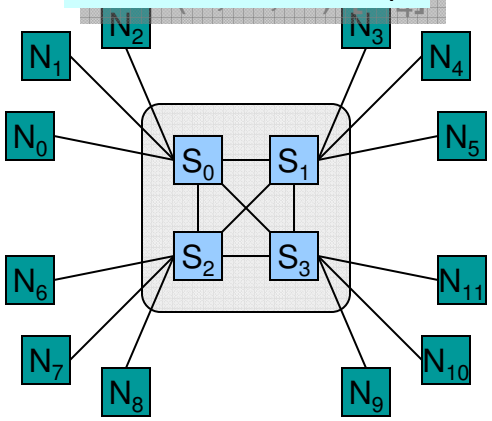
Hamming graph aka HyperX

- **Cartesian product of complete graphs**
- **HG(*p*, *g*, *h*)**
  - Hierarchical topology with *h* levels
  - Each level comprises *g* groups of switches
  - *p* end nodes are connected to each switch
  - Recursive construction
    - At level 0, each group comprises one switch
    - At level i+1, replicate the network of level i *g* times and connect each switch of each group directly to each switch at the same relative position in the other *g*-1 groups
- **Each group at a given level is directly connected to all other groups at that level**
  - Complete graph ("full mesh") among groups at each level
  - With g^{i-1} links in between each pair of groups at level *i*, $1 \leq i \leq h$
- **Total number of switches = g^h**
- **Total number of end nodes = pg^h**
- **Each switch has radix $r = n + h(g - 1)$**
- **Total number of internal links = $g^h \, h \, (g - 1) / 2$**
- **Diameter = $h+1$**

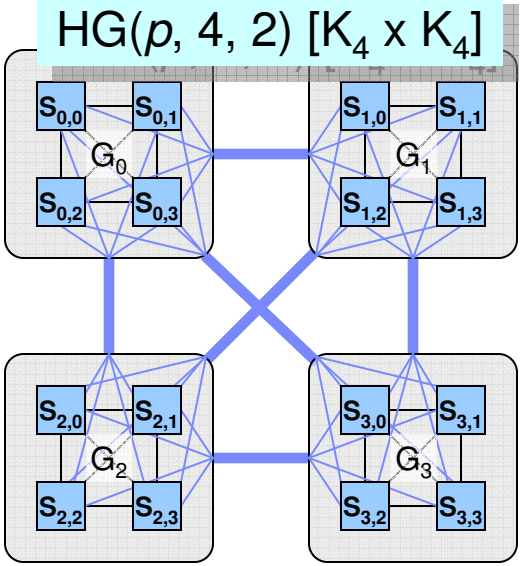
© 2013 IBM Corporation

Hamming graphs

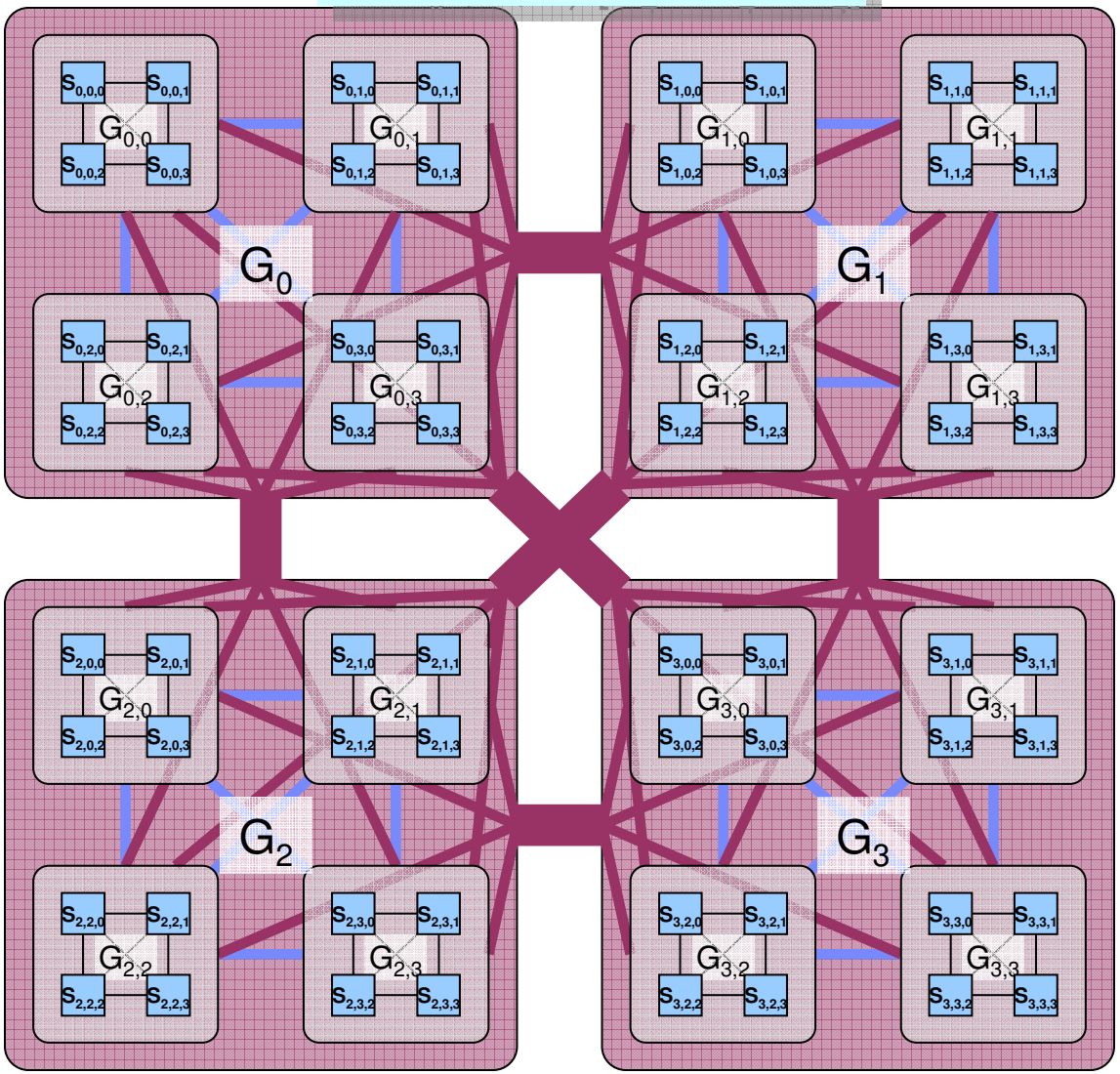
HG(3, 4, 1) [K_4]



HG(*p*, 4, 2) [K_4 x K_4]



HG(*p*, 4, 3) [K_4 x K_4 x K_4]

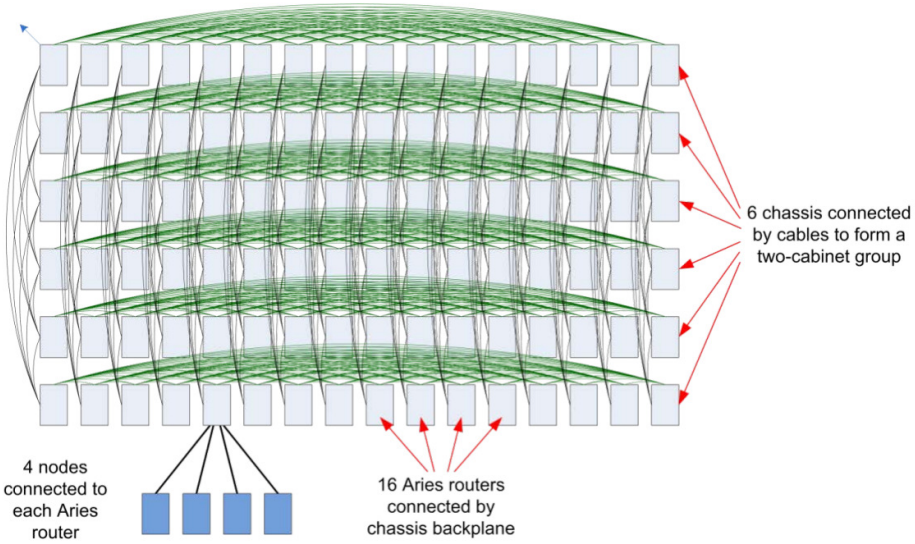


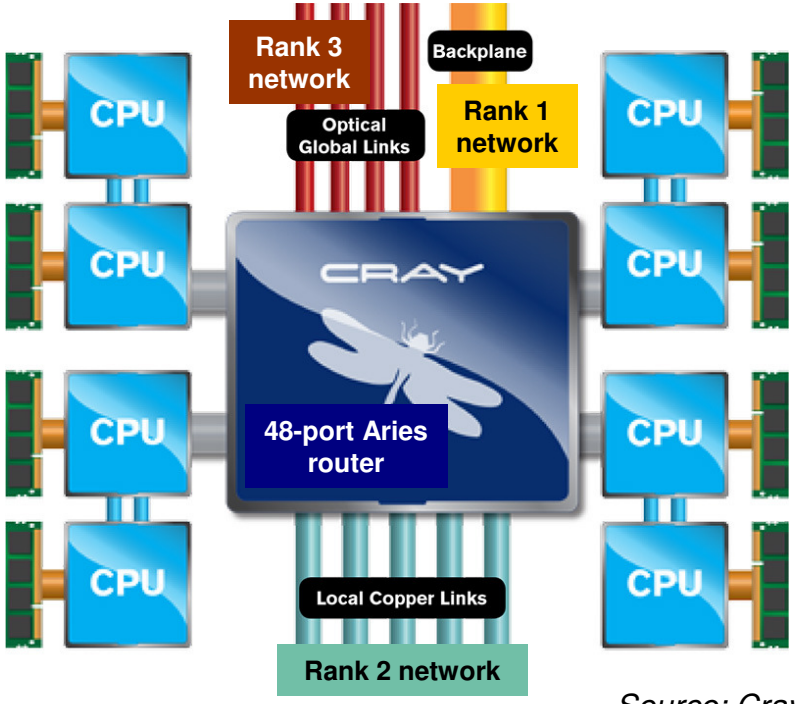
4 links 16 links

© 2013 IBM Corporation

Cray XC30

- **Aries interconnect: combination of Hamming graph & dragonfly**
- **Router radix = 48 ports**
  - node: 8 (2 ports per node)
  - rank-1: 15
  - rank 2: 15 (3 ports per connection)
  - rank 3: 10 (global)
- **Rank 1: complete graph of 16 routers**
  - 16 Aries, 64 nodes
- **Rank 2: group of 6 rank-1 graphs; Hamming graph K_{16} x K_6**
  - 96 Aries, 384 nodes
- **Up to 96x10+1 = 961 groups; in practice limited to 241 groups**
  - 23,136 Aries, 92,544 nodes





Source: Cray

© 2013 IBM Corporation

Routing

© 2013 IBM Corporation

Routing

- **How do we get from source node *A* to destination node *B*?**
- **HPC topologies are regular and static**
  - Optimal routing algorithms can be designed a priori
  - No need for flooding-based learning methods
  - Many of such algorithms could be implemented by algebraic manipulations on current & destination address
- **Examples**
  - Fat tree: destination[source]-modulo-radix
  - Mesh + torus: dimension order routing
  - Dragonfly: direct (local-global-local)
  - Hamming graph: level order routing
- **Variations/enhancments**
  - **Source-** (end node determines route) vs. **hop-by-hop** (routing decision at each router)
  - **Direct** (shortest path) vs. **indirect** (detour)
  - **Static** (path for a given src/dst pair is fixed) vs. **dynamic** (multiple paths may be used)
  - **Oblivious** (path alternative chosen obliviously) vs. **adaptive** (path alternative chosen based on network state)
- **This topic alone would be good for several hours**

© 2013 IBM Corporation

Flow control

© 2013 IBM Corporation

Message, packets & flits

- **A message is divided into multiple packets**
  - Each packet has header information that allows the receiver to re-construct the original message
  - Packets of same message may follow different paths

- **A packet may further be divided into *flow control units* (flits)**
  - Only head flits contains routing information
  - Flits of same packet must follow same path
  - Flit = finest granularity at which data flow is controlled (atomic unit of flow control)
  - Flits from different packets may (usually) not be interleaved ➔ ***wormhole routing***
  - If flit == packet size: ***virtual cut-through*** (VCT)

- **Segmentation into packets & flits allows the use of a large packets (low header overhead) and fine-grained resource allocation on a per-flit basis**

© 2013 IBM Corporation

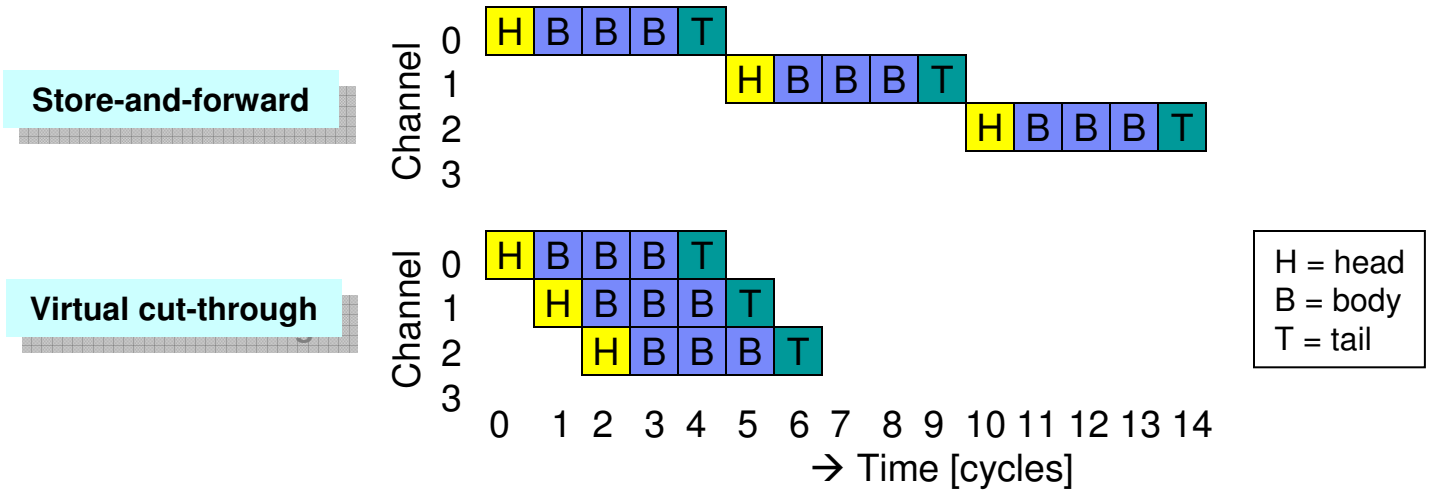
Flow control

- **HPC networks employ link-level flow control**
  - Avoid packet drops due to buffer overflows; retransmission latency penalty is considered too expensive
  - Enable buffer resource partitioning for deadlock prevention & service differentiation
  - This is a crucial difference from TCP/IP networks
- **Routing of a message requires allocation of various resources**
  - Channel (link bandwidth)
  - Buffer
  - Control state (virtual channel)
- **Non-packet-switched flow control**
  - Bufferless switching: drop & retransmit
  - Circuit switching: set up end-to-end circuit before sending data

© 2013 IBM Corporation

Buffered flow control

- **Buffers enable contention resolution in packet-switched networks**
  - Temporarily store packets contending for same channel
  - Decouple resource allocation for subsequent channels – buffers are cheaper bandwidth
- **Packet-buffer flow control: channels and buffers are allocated per packet**



© 2013 IBM Corporation

Flit-buffer flow control (Wormhole)

- **Wormhole flow control is virtual cut-through at the flit instead of packet level**
- **A head flit must acquire three resources at the next switch before being forwarded:**
  - Channel control state (virtual channel)
  - Buffer for one flit
  - Channel bandwidth for one flit
  - Body and tail flits use the same VC as the head, but must still acquire a buffer and physical channel individually
- **Can operate with much smaller buffers than VCT**
- **May suffer from channel blocking: channel goes idle when packet is blocked in the middle of traversing the channel**

© 2013 IBM Corporation

Virtual channel flow control

- **Multiple virtual channels per physical channel**
  - Partition each physical buffer into multiple logical buffer spaces, one per VC
  - Flow control each logical buffer independently
  - Flits are queued per VC
  - VCs may be serviced independently
  - Incoming VC may be different from outgoing VC
  - Each VC keeps track of the output port + output VC of head flit
  - Head flit must allocate the same three resources at each switch before being forwarded

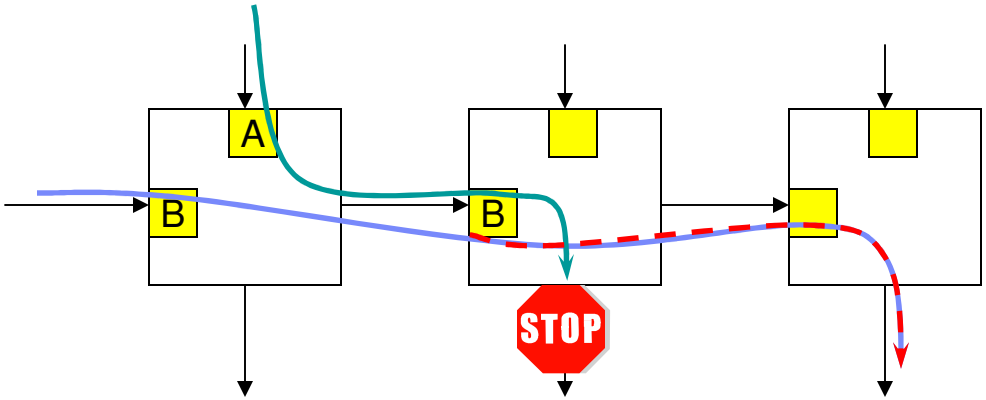
- **Advantages**
  - Alleviate channel blocking
  - Enable deadlock avoidance
  - Enable service differentiation

© 2013 IBM Corporation

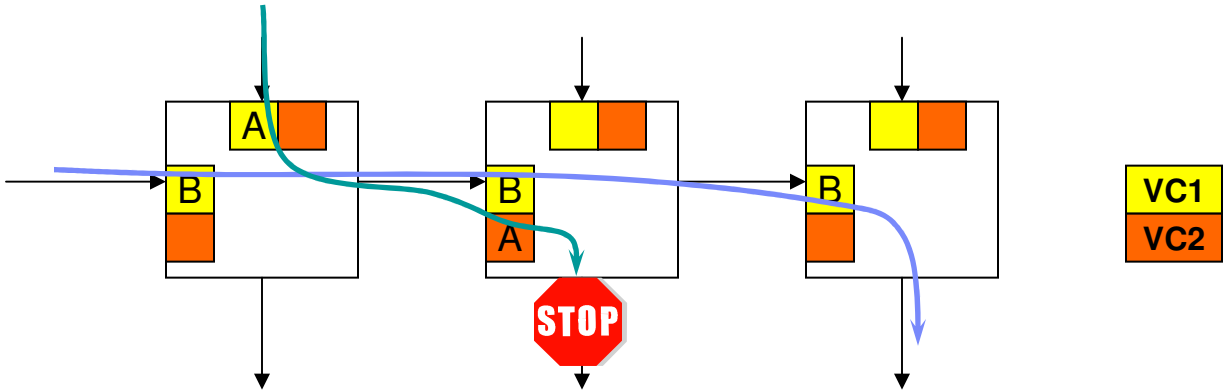
Example

Message A is blocked in the middle switch
Message B needs to traverse the middle switch

Wormhole



Virtual channel



© 2013 IBM Corporation

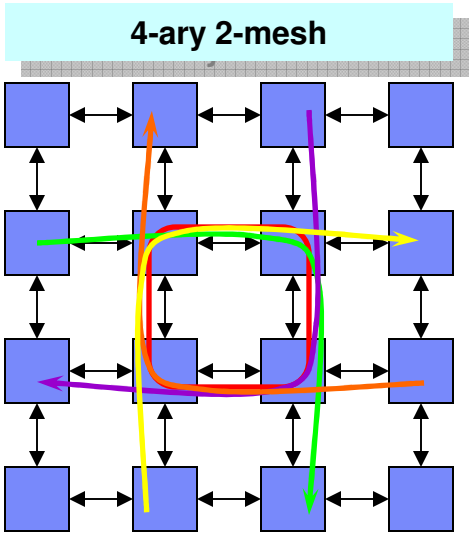
Deadlock avoidance

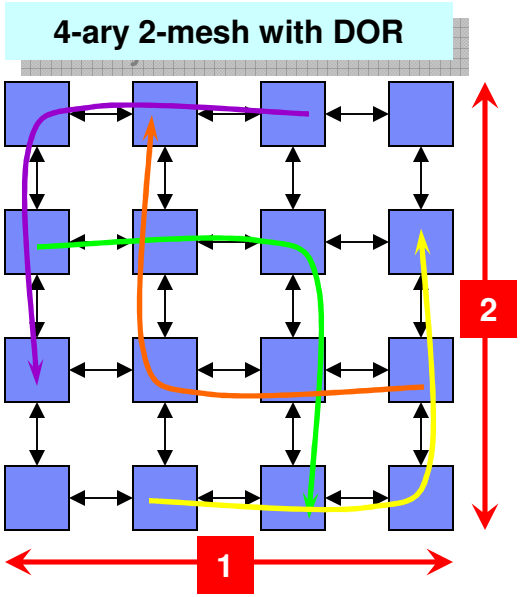
© 2013 IBM Corporation

Routing deadlocks

- **Most topologies (other than trees) have *loops***
  - Loops can be dangerous!
  - Packets routed along a loop would never get to their destination
  - Broadcast packets in a loop can cause devastating **broadcast storms**
  - For this reason, Ethernet network use the Spanning Tree Protocol: Overlay a loop-free logical topology on a loopy physical one
  - In regular topologies, routing loops can be avoided by a sound routing algorithm

- **However, because HPC networks also use link-level flow control, loops can induce *routing deadlocks***
  - In deadlock, no packet can make forward progress anymore
  - Formally, a routing deadlock can occur when there is a **cyclic dependency in the channel dependency graph**
  - Such dependencies can be broken
    - By restricting routing
    - By restricting injection (e.g., bubble routing)
    - By means of virtual channels

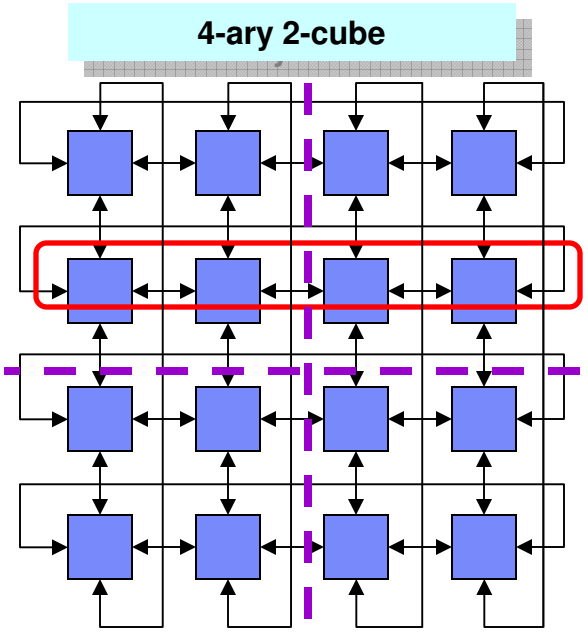




© 2013 IBM Corporation

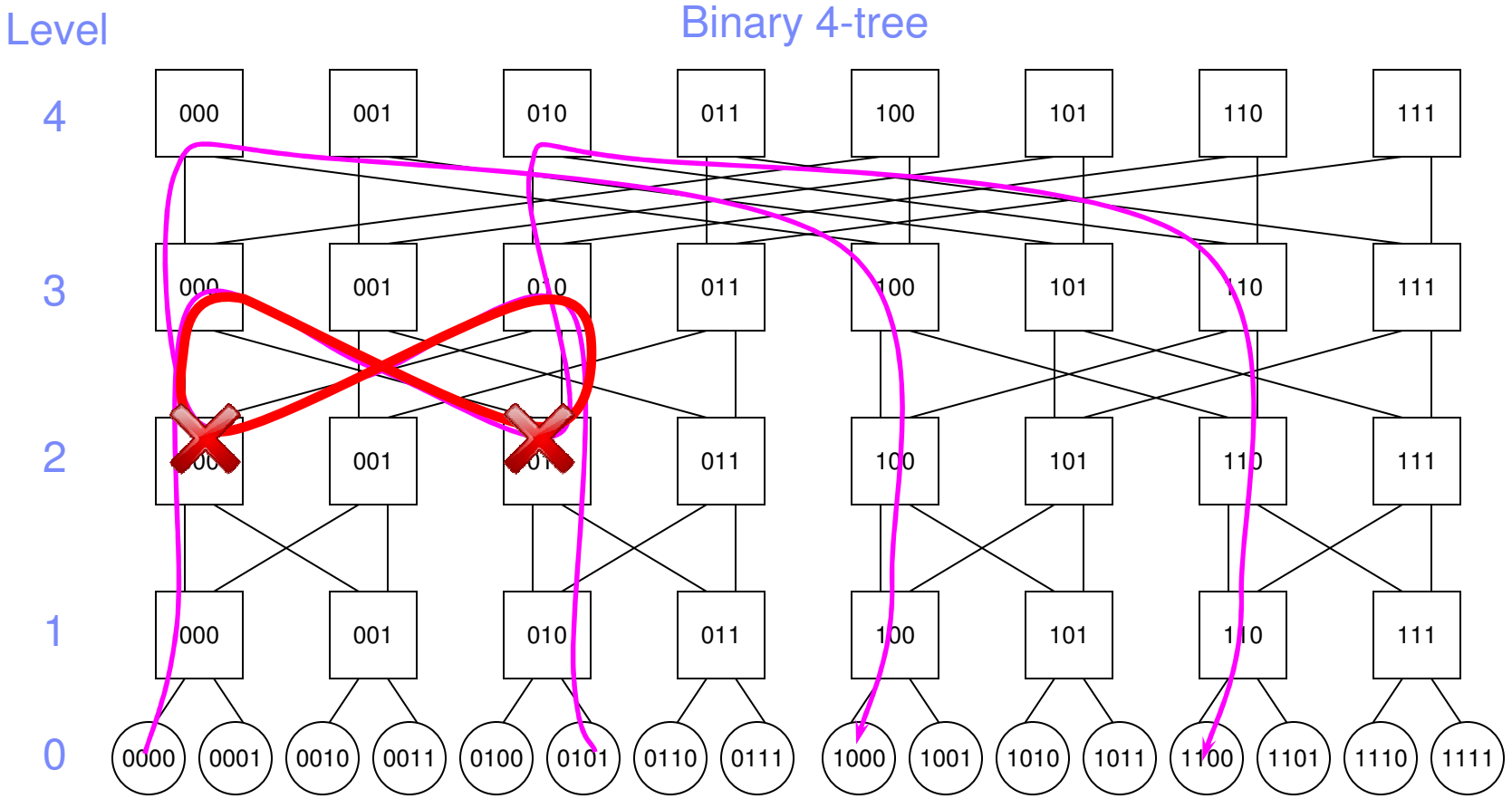
Deadlock-free routing in tori

- **Does dimension-ordered routing avoid deadlocks in a torus?**
- **No, because there is a cycle *within* each dimension**
- **Introduce a second VC to break the cyclic dependency: "Dateline" routing**
  - Each ring has a dateline link
  - Each packet starts on VC1
  - When a packet crosses the dateline, it moves to the VC2
  - Visit dimension in fixed order
  - When moving to another dimension, go back to VC1



© 2013 IBM Corporation

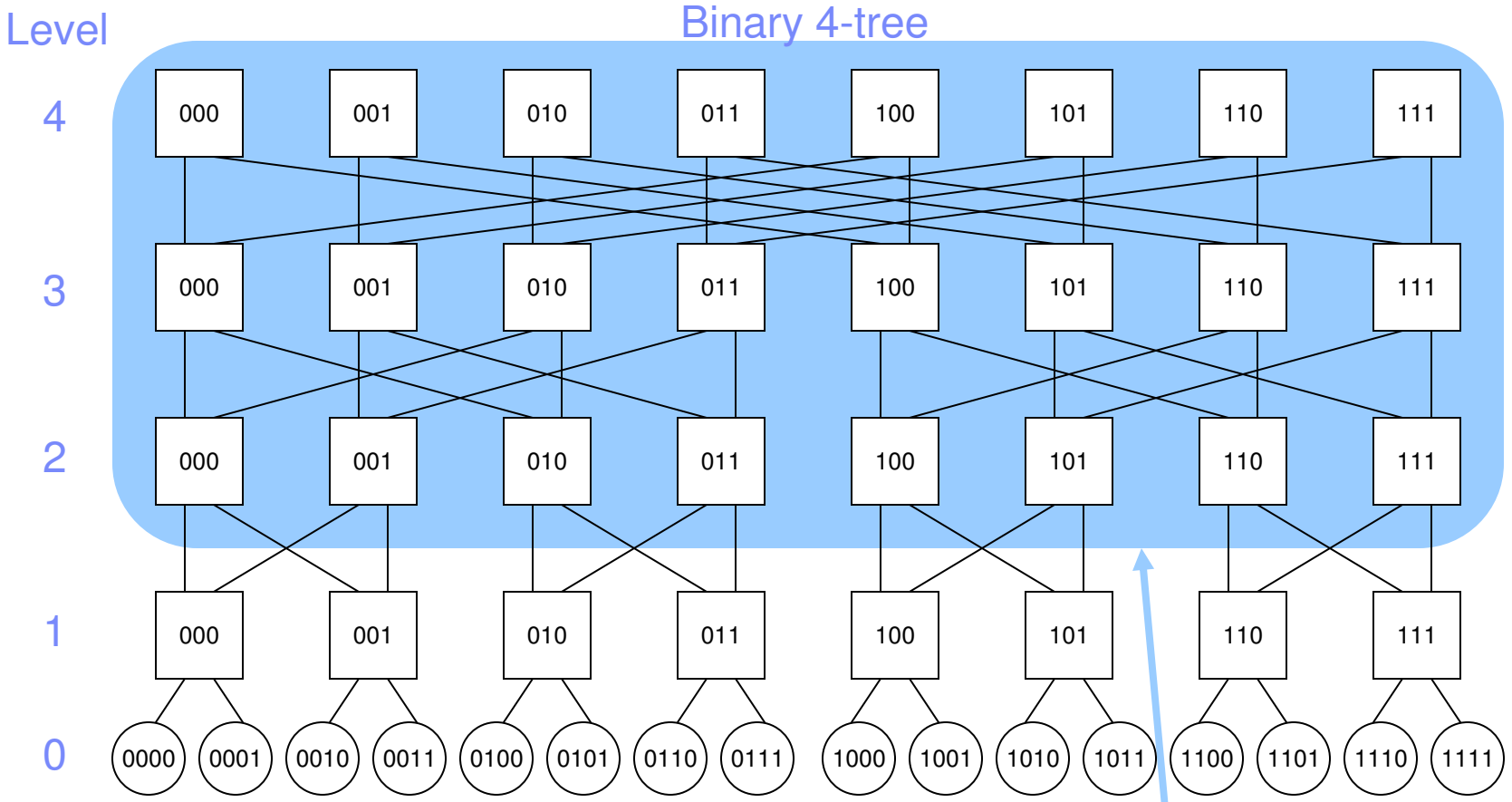
Recall the *k*-ary *n*-tree network



- **Are loops a problem in *k*-ary *n*-trees?**
- **Not really: Shortest-path routing never performs a down-up turn**
  - Always *n* hops up followed by *n* hops down
  - No cyclic dependencies in channel dependency graph

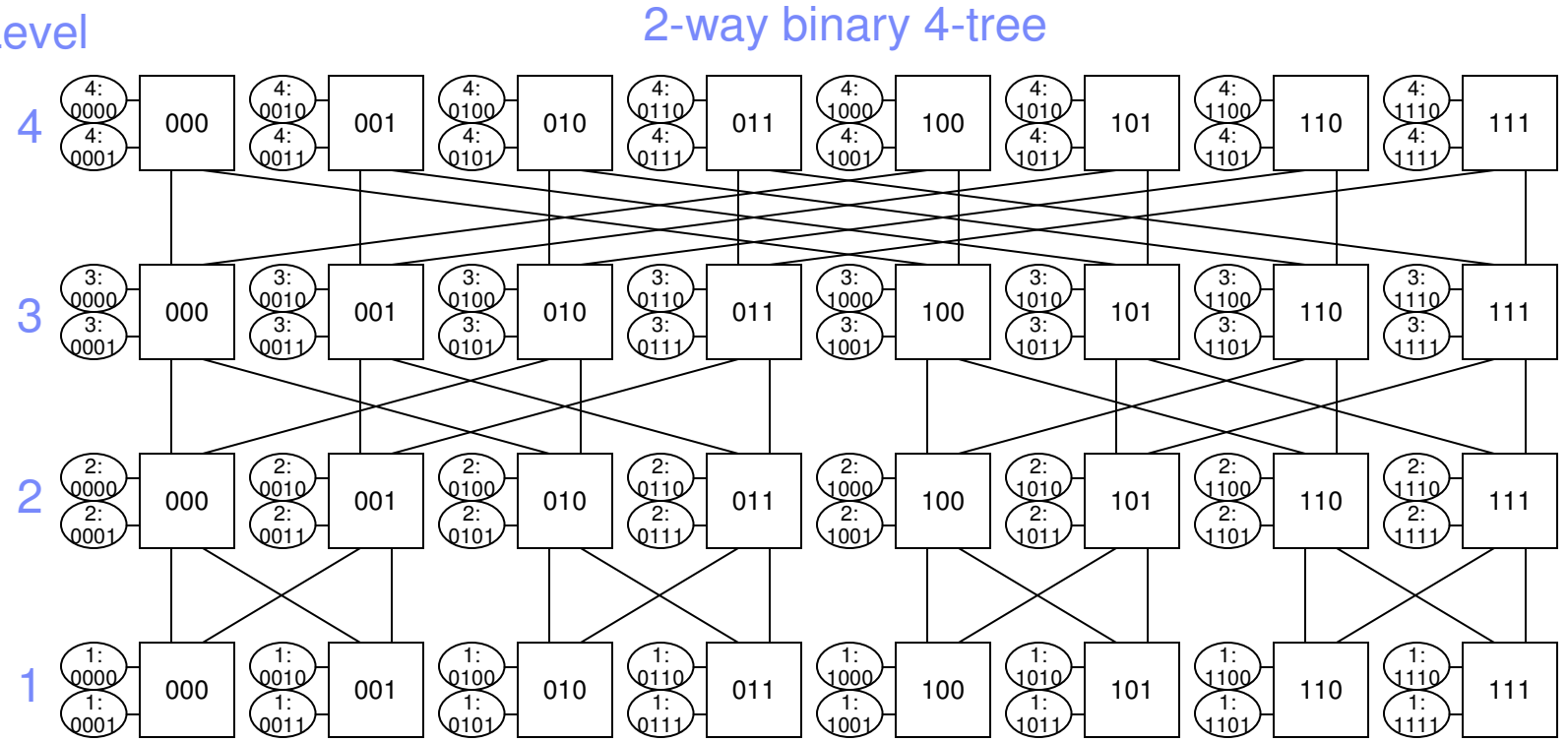
© 2013 IBM Corporation

A *k*-ary *n*-tree network with more nodes



© 2013 IBM Corporation

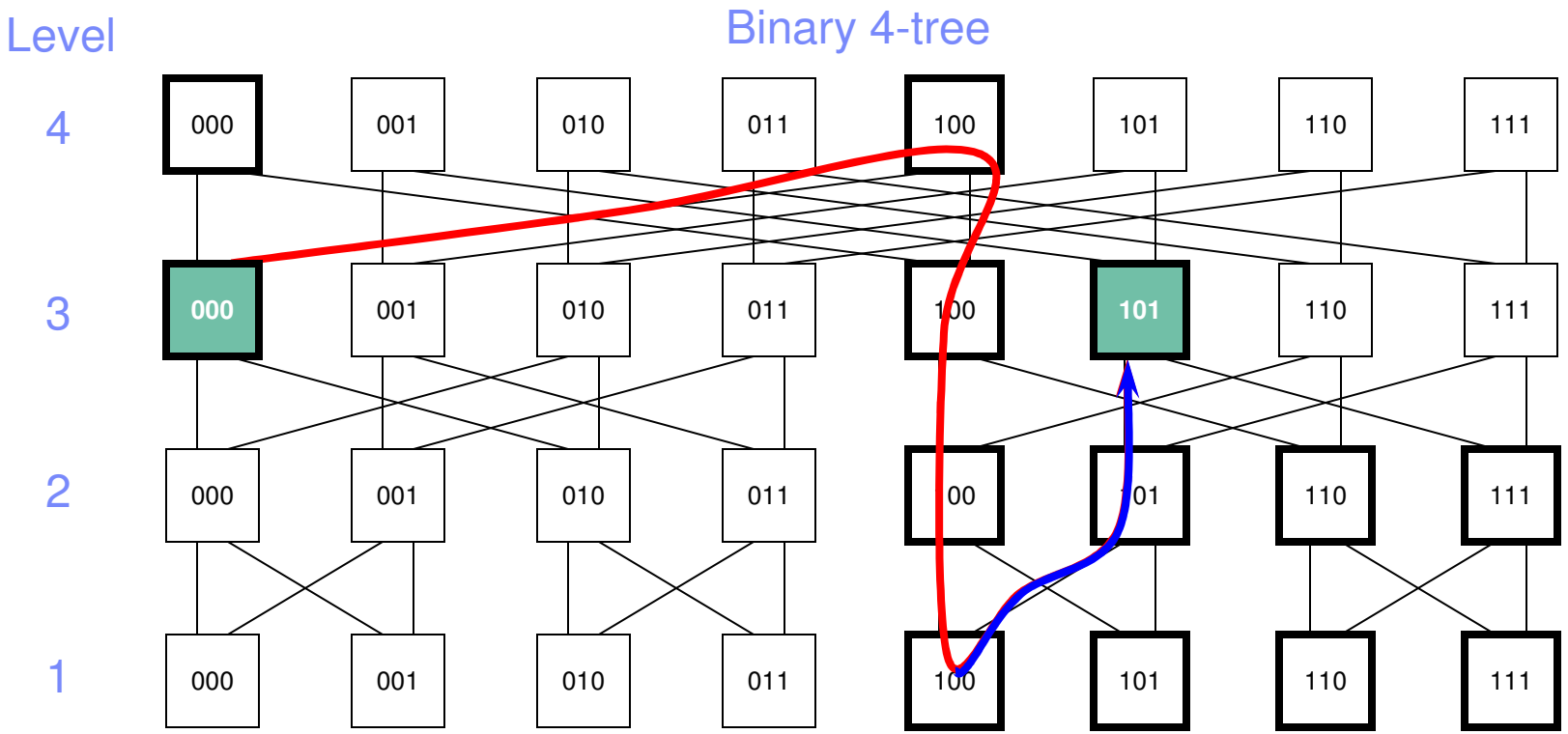
b-way *k*-ary *n*-direct-tree



- bnk^{n-1} end nodes
- nk^{n-1} switches arranged in n stages
- $nk^n/2$ inter-switch links

© 2013 IBM Corporation

Deadlock prevention in direct trees



- **NCA(S(3;000), S(3,101)) = S(4; x00)**
- **NCD(S(3;000), S(3,101)) = S(1; 1yz)**
- **8 alternative shortest paths**
- **Route on VC0 (red) until NCD, then switch to VC1 (blue)**

© 2013 IBM Corporation

Workloads

© 2013 IBM Corporation

Workload communication patterns

Understanding workloads characteristics is crucial to designing a cost-performance-optimal system.

For the interconnect designer, this implies understanding temporal, spatial, and causal patterns of inter-node communications.

© 2013 IBM Corporation

Let's look at bytes per FLOP

	Power 775	BG/Q
Cores/chip	8	16
Chips/node	4	1
Cores/node	32	16
GFLOPS/core	31.2	12.8
GFLOPS/chip	249.6	204.8
GFLOPS/node	998.4	204.8
Node escape bw [GB/s]	448	20
Memory/node [GB]	256	16
Memory/core [GB]	8	1
Topology	2D Dragonfly	5D Torus
Bytes/FLOP	**0.45**	**0.10**

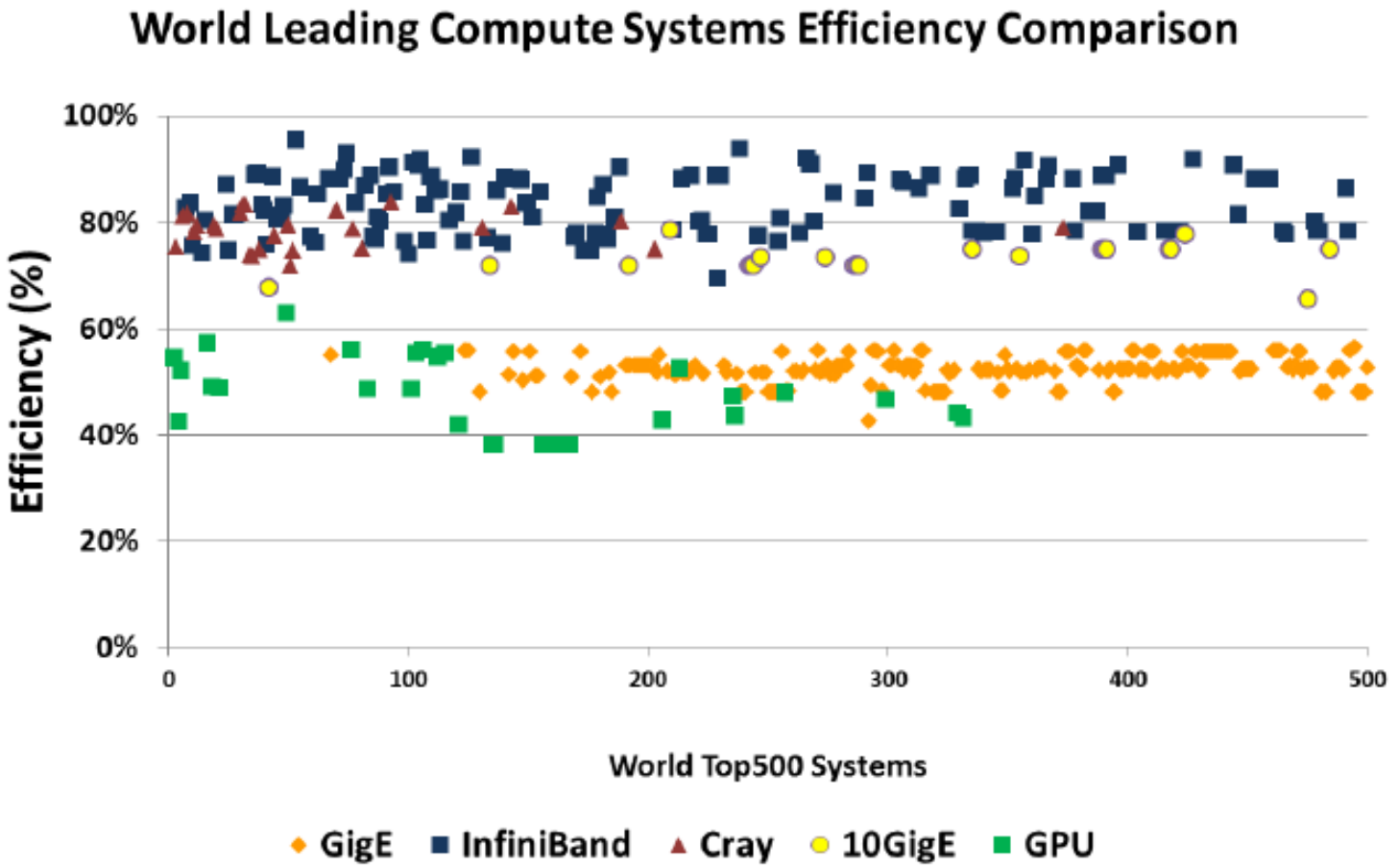
BG/Q achieves higher parallel efficiency at scale despite much lower B/F ratio! (and much less memory per core, and higher-diameter network…)
Therefore, for the LINPACK workload:
➔If problem size is large enough, then network bandwidth does not matter
➔Corollary: If bandwidth does matter, then the problem size is too small!

Rank	Site	Country	System	Cores	Rmax (TF/s)	Rpeak (TF/s)	Eff. (%)
23	IBM Development Engineering	United States	Power 775	39680	886.4	1217.7	73
34	ECMWF	United Kingdom	Power 775	24576	549	754.2	73
43	UK Meteorological Office	United Kingdom	Power 775	18432	411.7	565.6	73
51	UK Meteorological Office	United Kingdom	Power 775	15360	343.1	471.4	73
90	Environment Canada	Canada	Power 775	8192	185.1	251.4	74
113	IBM POK Benchmarking Center	United States	Power 775	6912	159.6	212.1	75
143	UK Meteorological Office	United Kingdom	Power 775	5120	124.8	157.1	79
335	Slovak Academy of Sciences	Slovak Republic	Power 775	3072	76.5	94.3	81
382	IBM POK Benchmarking Center	United States	Power 775	2816	70.8	86.4	82
463	University of Warsaw	Poland	Power 775	2560	64.3	78.6	82
1	DOE/NNSA/LLNL	United States	BlueGene/Q	1572864	16325	20133	81
3	DOE/SC/ANL	United States	BlueGene/Q	786432	8162.4	10066	81
7	CINECA	Italy	BlueGene/Q	163840	1725.5	2097.2	82
8	Forschungszentrum Juelich	Germany	BlueGene/Q	131072	1380.4	1677.7	82
13	Daresbury Laboratory	United Kingdom	BlueGene/Q	114688	1207.8	1468	82
20	University of Edinburgh	United Kingdom	BlueGene/Q	98304	1035.3	1258.3	82
29	EDF R&D	France	BlueGene/Q	65536	690.2	838.9	82
36	High Energy Accelerator Research Organization	Japan	BlueGene/Q	49152	517.6	629.1	82
48	DOE/NNSA/LLNL	United States	BlueGene/Q	32768	345.1	419.4	82
100	DOE/SC/ANL	United States	BlueGene/Q	16384	172.7	209.7	82
252	DOE/NNSA/LLNL	United States	BlueGene/Q	8192	86.3	104.9	82

Source: top500.org

© 2013 IBM Corporation

LINPACK performance: IB vs GigE



- TOP500 systems listed according to their efficiency
- InfiniBand is the key element responsible for the highest system efficiency
- Up to 96% efficiency

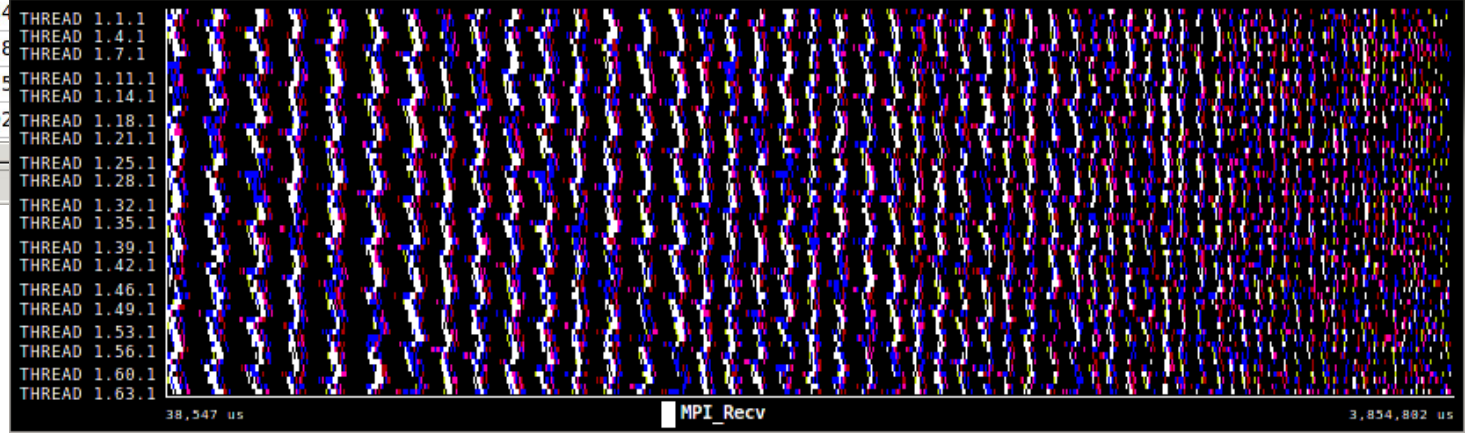
Brian Sparks *IBTA Marketing Working Group Co-Chair*

© 2013 IBM Corporation

MPI call profile @ HPL.orig.64.8192.128.8.8.chop1.prv (on tusa.zurich.ibm.com)

	Outside MPI	MPI_Send	MPI_Recv	MPI_Irecv	MPI_Wait	MPI_Iprobe
THREAD 1.1.1	59.38 %	15.25 %	18.25 %	0.50 %	6.60 %	0.03 %
THREAD 1.2.1	60.39 %	15.95 %	17.02 %	0.60 %	6.03 %	0.01 %
THREAD 1.3.1	59.48 %	16.71 %	16.42 %	0.59 %	6.79 %	0.01 %
THREAD 1.4.1	57.46 %	19.63 %	15.58 %	0.61 %	6.72 %	0.00 %
THREAD 1.5.1	55.60 %	17.98 %	18.26 %	0.69 %	7.45 %	0.02 %
THREAD 1.6.1	57.05 %	18.24 %	16.52 %	0.64 %	7.53 %	0.01 %
THREAD 1.7.1	55.53 %	19.05 %	17.02 %	0.75 %	7.64 %	0.01 %
THREAD 1.8.1	53.40 %	21.63 %	17.11 %	0.69 %	7.15 %	0.01 %
THREAD 1.9.1	59.65 %	16.77 %	17.25 %	0.45 %	5.86 %	0.02 %
THREAD 1.10.1	60.79 %	17.33 %	15.92 %	0.48 %	5.47 %	0.01 %
THREAD 1.60.1	59.72 %	21.66 %	12.73 %	0.57 %	5.32 %	0.01 %
THREAD 1.61.1	58.01 %	18.36 %	17.41 %	0.60 %	5.60 %	0.02 %
THREAD 1.62.1	59.88 %	18.83 %	14.59 %	0.70 %	5.99 %	0.01 %
THREAD 1.63.1	58.42 %	20.12 %	14.83 %	0.68 %	5.94 %	0.01 %
THREAD 1.64.1	56.50 %	22.30 %	15.35 %	0.69 %	5.16 %	0.01 %
Total	3,751.05 %	1,205.58				
Average	58.61 %	18.84				
Maximum	63.48 %	23.18				
Minimum	53.40 %	15.25				
StDev	2.55 %	2.02				

MPI call @ HPL.orig.64.8192.128.8.8.chop1.prv (on tusa.zurich.ibm.com)



High-Performance LINPACK
MareNostrum, 64 MPI tasks
N = 8192, BS = 128, P=Q=8

Problem too small:
60% computation
40% communication

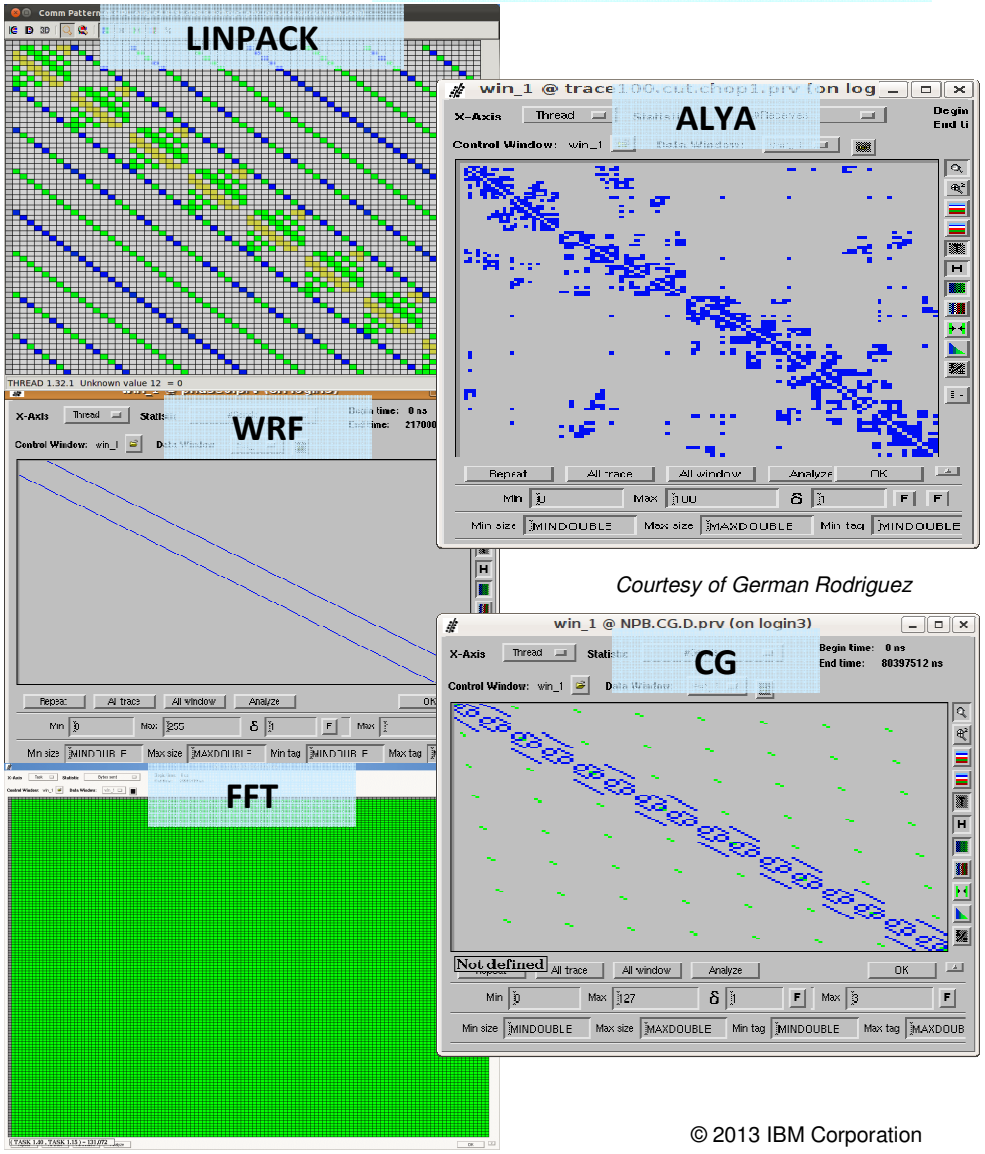
Courtesy of German Rodriguez

© 2013 IBM Corporation

The communication matrix

Each graph shows data volume per task pair (x,y)

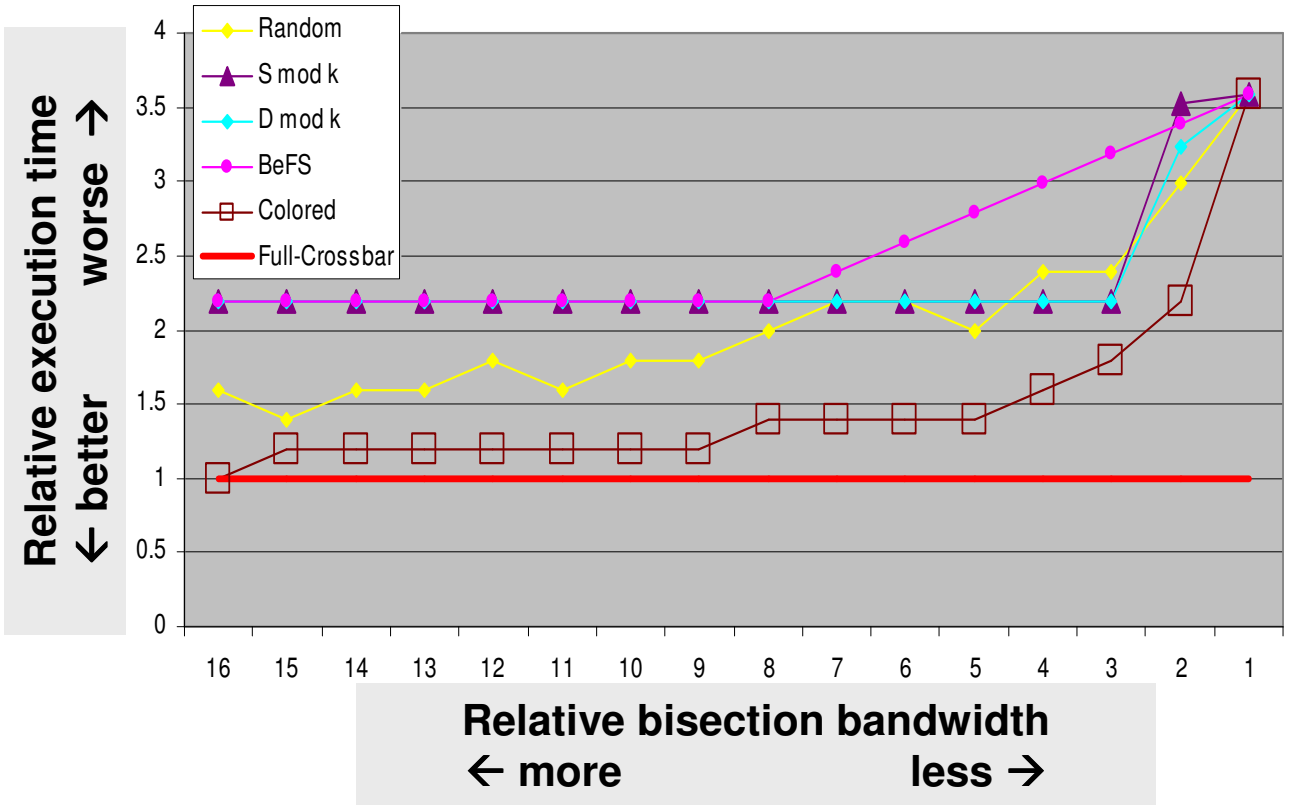
- **Each task communicates with a limited subset of other tasks**
- **Very regular structure**
- **This is typical for many scientific codes**
- **If one were to design an interconnect for a specific code, one could exploit this to great effect!**
- **However**
  - Depends on task placement
  - Patterns vary across codes
  - For some codes the matrix is even
  - Says nothing about comm/comp ratio!



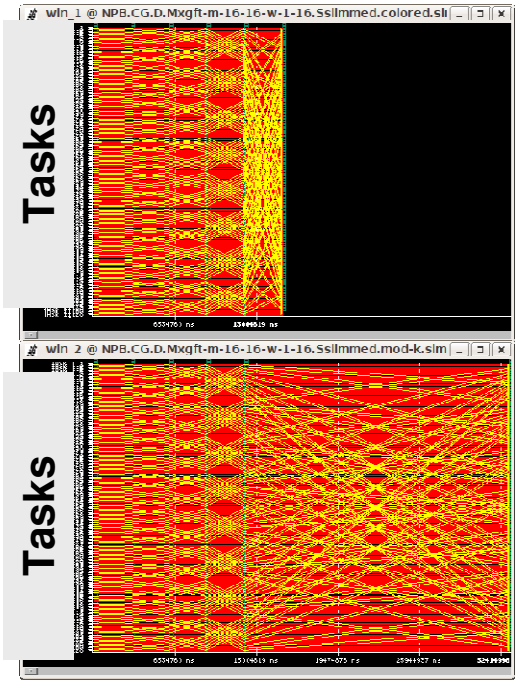
Courtesy of German Rodriguez

© 2013 IBM Corporation

Impact of topology & routing on CG



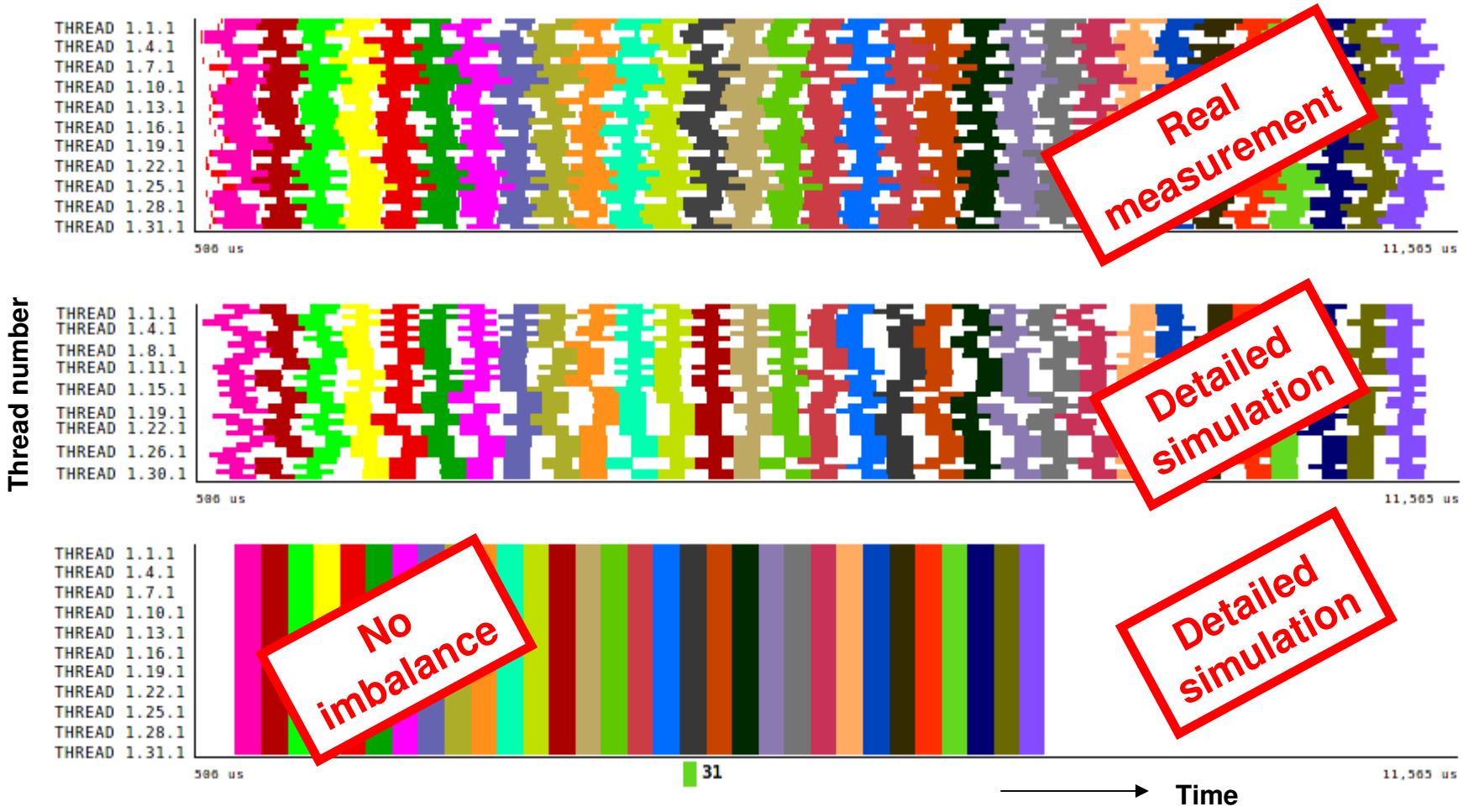
Execution traces



Performance of communication-intensive codes depends strongly on interconnect design: topology, routing, etc.

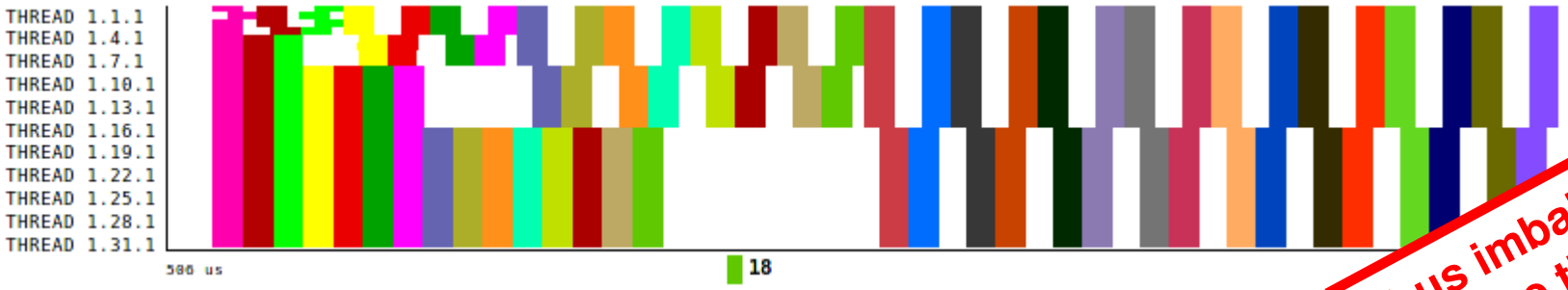
© 2013 IBM Corporation

All-to-all exchange pattern

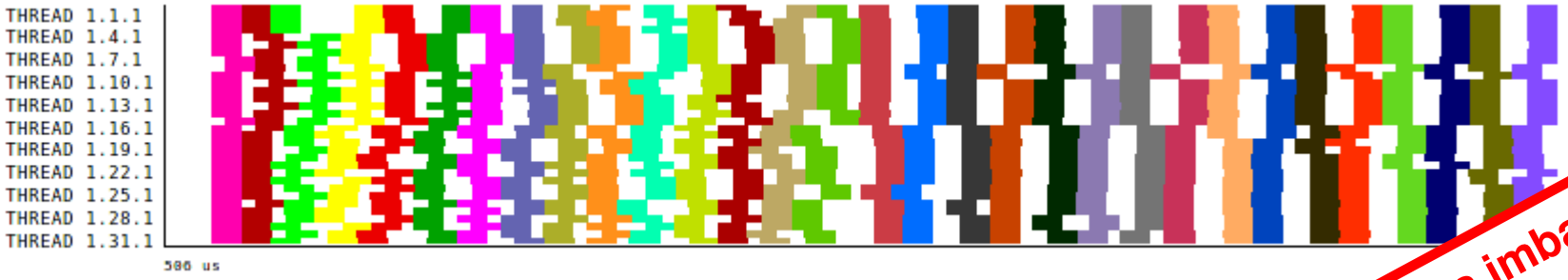


- Each color marks the physical reception of the payload of a message
- Gaps between messages are exchanges of rendez-vous protocol messages

© 2013 IBM Corporation



1 us imbalance in one thread



1 us imbalance in a few threads

- **Dependencies cause delays to accumulate and propagate**
- **From 1 microsecond imbalance to 1 millisecond loss!**

© 2013 IBM Corporation

Workload predictability

- **HPC systems are usually "closed"**
  - Not directly exposed to the outside world, only local research community
  - Jobs controlled by a scheduler
- **Scientific codes**
  - Comm patterns of many scientific code are quite static and regular
  - With some codes, comm depends on the input/problem (e.g., adaptive mesh refinement)
- **Capacity vs capability**
  - Need to control/be aware of task placement
- **Certain classes of data centers are "open"**
  - North-south vs east-west traffic
  - Much less predictable workloads
  - Virtualization will affect this negatively – but is also an opportunity

© 2013 IBM Corporation

Performance evaluation

© 2013 IBM Corporation

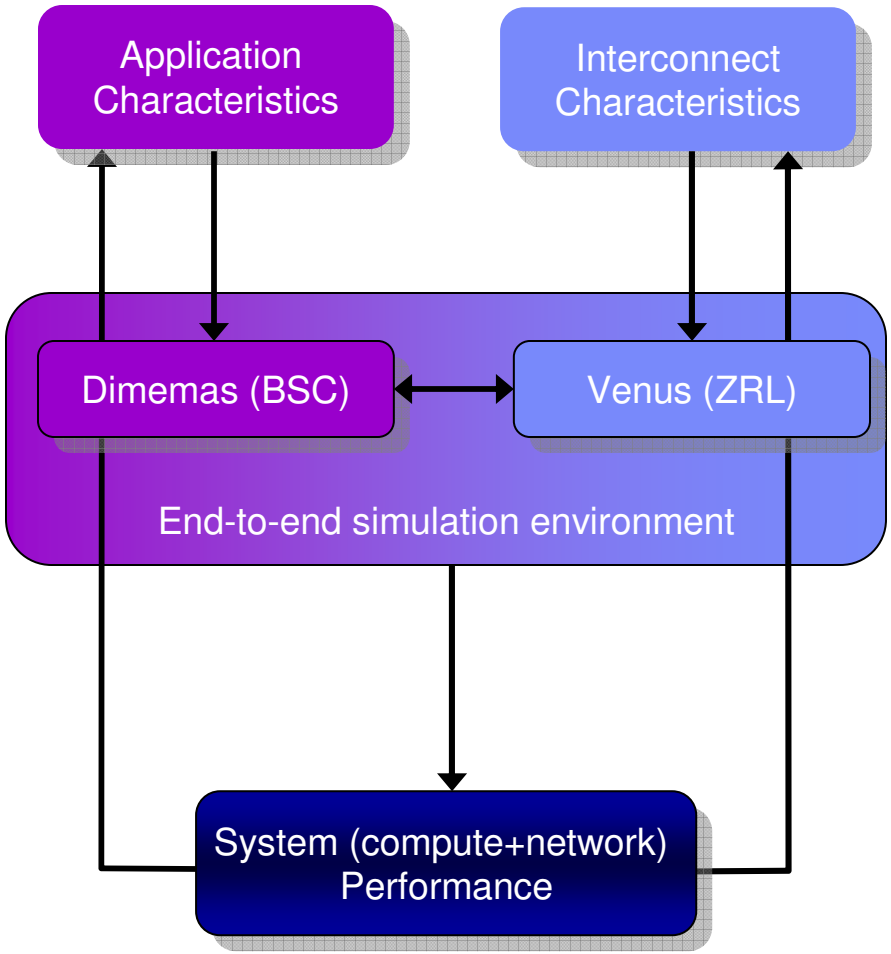
Modeling of large-scale HPC systems

- **Dichotomy in performance evaluation in computer architecture**
  - Node architecture
    - Execution-driven, cycle-accurate CPU (ISA), cache and memory models
    - Highly accurate
    - Too much detail to scale to large node counts
  - Network architecture
    - Highly accurate (flit level) at network level
    - Several orders of magnitude fewer network nodes than CPU cores
    - Network node much simpler than CPU core
    - But usually driven by either purely random or purely deterministic and **non-reactive** traffic patterns
- **Need to adopt a holistic approach, taking into account application, node architecture, and network architecture**
  - Given the scale of current and future systems, parallel simulation is the only way to manage simulation run time and memory footprint

© 2013 IBM Corporation

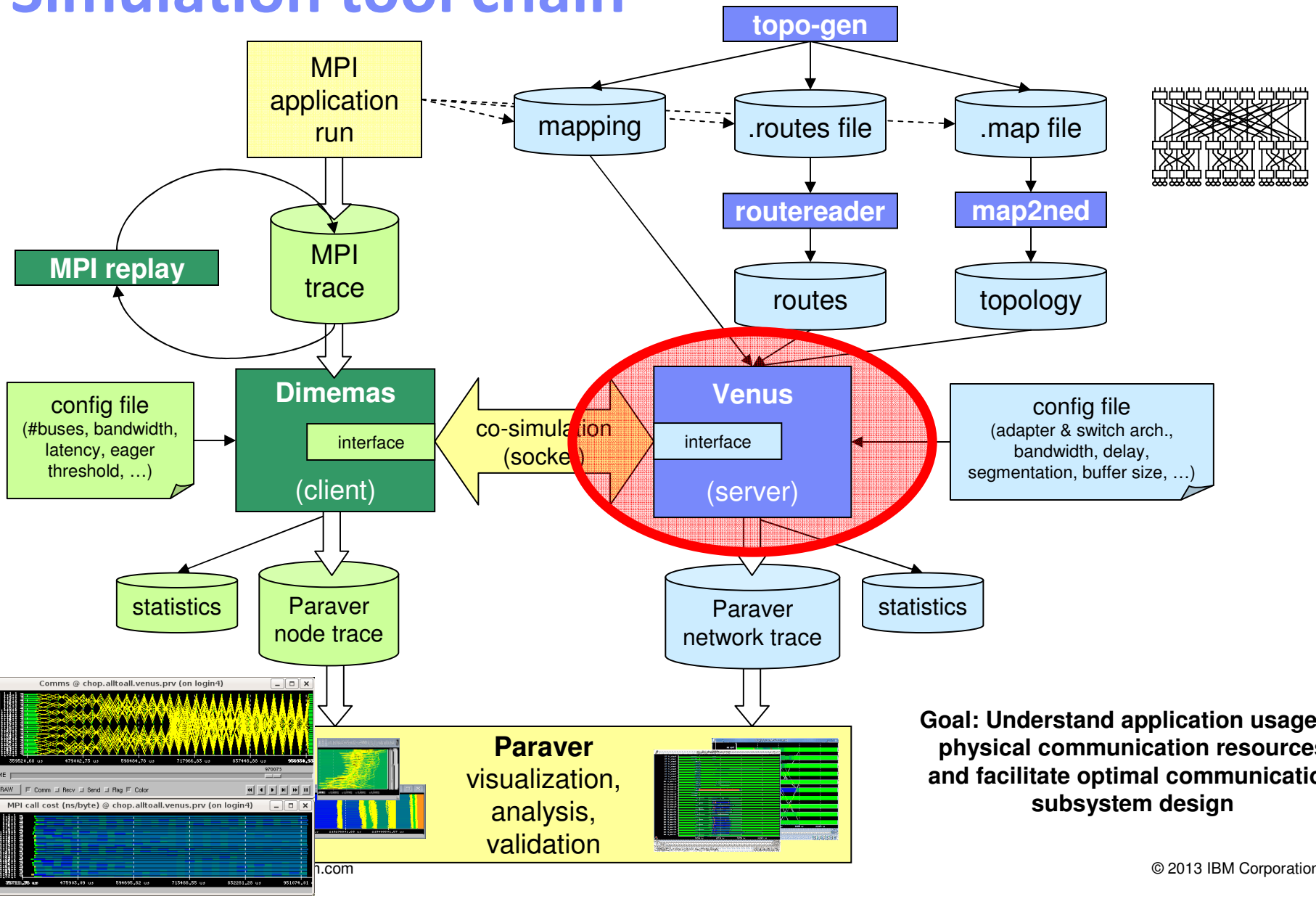
Application-level performance prediction

1. **Instrument applications to collect computation, communication and their inter-dependencies**
   - For apps or benchmarks of interest
2. **Collect traces on a production system**
   - e.g., BG/P, MareNostrum
3. **Perform full-system trace-driven simulations with Dimemas+Venus**
   - Tune model parameters to match reality
   - Perform parameter sensitivity studies
     - Network technology
     - Network topology
     - Routing, etc…
4. **Optimize**
   - Interconnect: e.g. performance/$
   - Application: e.g. communication scheduling



© 2013 IBM Corporation

Simulation tool chain



Goal: Understand application usage of physical communication resources and facilitate optimal communication subsystem design

© 2013 IBM Corporation

Outlook:
Exascale interconnects

© 2013 IBM Corporation

Exascale interconnect challenges

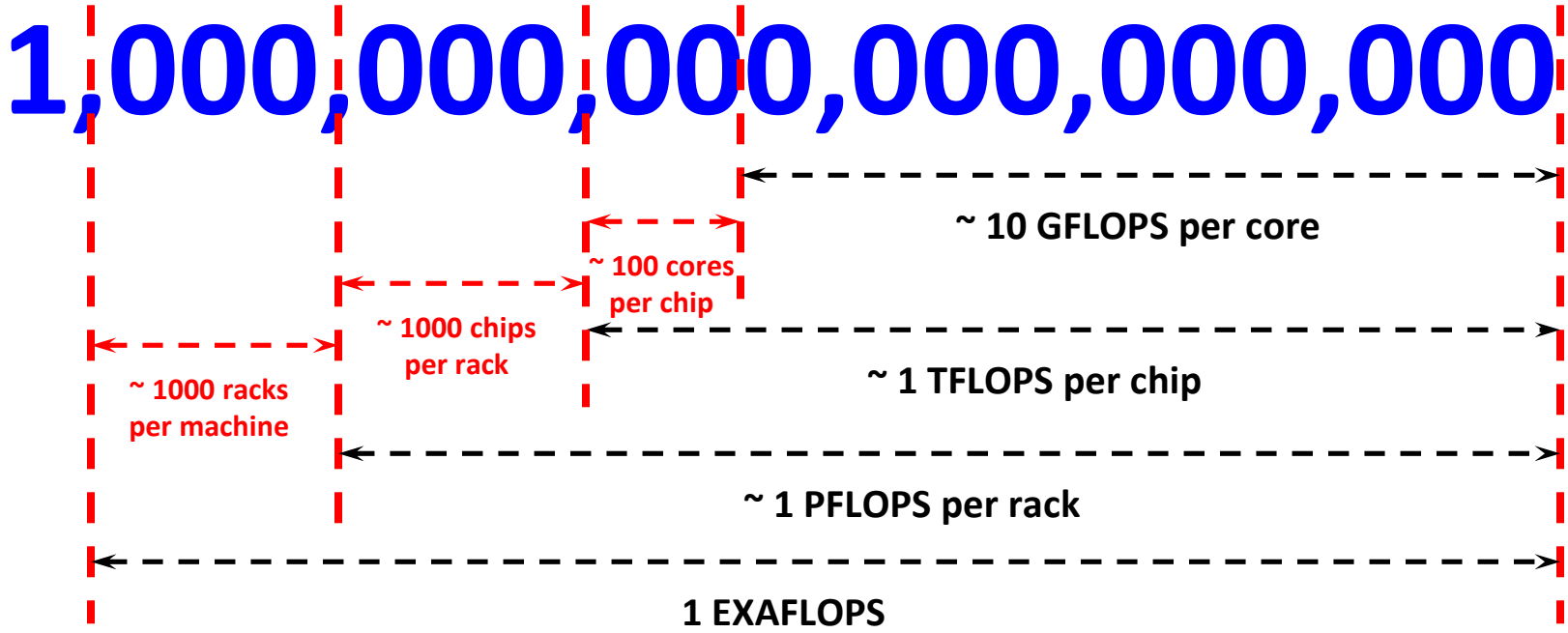
- Proprietary network technologies are not commercially viable
- Indirect networks are unaffordable for extreme-scale HPC
- Integrated networks (NIC/switch on CPU) are becoming a reality

© 2013 IBM Corporation

Towards exascale computing

- Current #1 on Top 500: 17.59 PetaFLOPS @ 8.2 MW
- 1 Exaflop = 10^{18} floating point operations per second @ 20 MW
- Timeframe: 2018/19/20

How do we increase FLOPs by two orders of magnitude while keeping power and space within the current order of magnitude?



© 2013 IBM Corporation

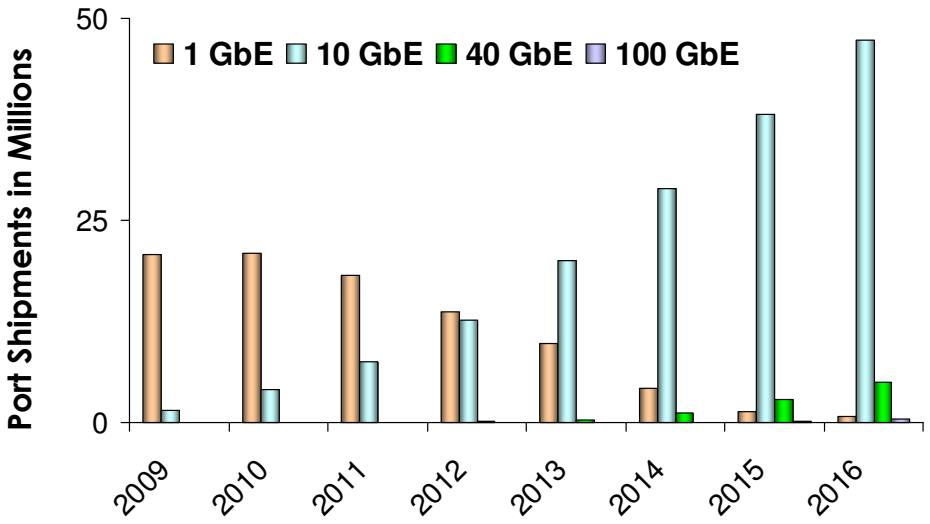
Exascale network implications

- **Assume improve FLOPS/node improve 10x**
  - General purpose CPUs
  - 2 TFLOP/node
  - i.e. about 500'000 nodes
- **Byte/FLOP ratio**
  - Typically between 0.1 and 1
  - Let's assume "cheap" 0.1 scenario
  - 10^{18} FLOPS ➔ 10^{17} B/s aggregate network injection bandwidth
  - i.e. about 200 GB/s/node
- **Network speed**
  - Speed will be driven by link technology: transceivers, connectors, cables/fibers; physical link probably ~50 Gb/s
  - By 2018/20 100 Gb/s will be ramping up, 400 Gb/s emerging (standard ratification expected 2017)
  - $(10^{17}) / (12.5*10^{9})$ = 8 million 100G ports!!
- **Network power**
  - Let's be generous and allocate 25% of power budget to the network, i.e. 5 MW
  - Assume mean #hops per communication is 5 (which is VERY low) and links consume ZERO power when idle
  - Link efficiency: $(5*10^{6}$ W) / $(5*8*10^{17}$ b/s) = $1.25*10^{-12}$ = 1.25 pJ/bit

	Power 775	BG/Q
Cores/chip	8	16
Chips/node	4	1
Cores/node	32	16
GFLOPS/core	31.2	12.8
GFLOPS/chip	249.6	204.8
GFLOPS/node	998.4	204.8
Node escape bw [GB/s]	448	20
Memory/node [GB]	256	16
Memory/core [GB]	8	1
Topology	2D Dragonfly	5D Torus
Bytes/FLOP	**0.45**	**0.10**

© 2013 IBM Corporation

Data Center Port Shipments – Ethernet Switching



Source: Dell'Oro, Ethernet Alliance panel @ OFC'13

Source: networkworld.com

© 2013 IBM Corporation

Silly example

- **How much would a 128K-node fat tree network with 200 GB/s bandwidth per node cost if we tried to build it using commercial Ethernet switches *today*?**
  - –64 port switches (
  - –4 layers: 32*32*3
  - –#switches = (128K
  - –To achieve 200 GI networks
  - –Total number of s
  - –Such a switch cos

US$ 26B

(and then you haven't bought a single cable yet)

© 2013 IBM Corporation

Topology options

- **Fat tree**
  - Switch radix 64: tree arity = 32
  - Requires at least 4 layers (32^3 = 32K) to scale to 100-500K nodes
  - Diameter = 7
  - Bisection: full
  - Total of 64K switches for a 512K node full bisection bandwidth network!
  - Indirect topology with high-radix switches and many long links
    - For small radix (k<n), the number of switches may exceed number of nodes
  - Switch radix too large for integration on CPU: Discrete switch chips & boxes (e.g. Ethernet or IB)
  - Best performance, but completely unaffordable

- **Torus**
  - Switch radix 10: 5 dimensions
  - Requires 10-14 nodes along each dimension to scale to 100-500K nodes
  - Diameter = 50-70
  - Bisection: low
  - Direct topology with low-radix switches and mostly short links
  - Switches can be integrated: vastly reduced cost, but low bisection bandwidth and huge diameter

- **Dragonfly**
  - Number of nodes = a(ah+1)
  - Requires switch radix 90-150: a = 60-100, h = 30-50 to scale to 100-500K nodes
  - Diameter = 3
  - Bisection: high (but requires non-minimal routing)
  - Direct topology with very-high-radix switches and many long links
  - Switch radix too large for integration on CPU: Discrete switch chips (e.g. p775 hub chip); 1 switch per node

"Die eierlegende Wollmilchsau"

- **What we need is a direct topology with low-radix switches, small diameter, high bisection bandwidth and not too many long links**
  - Is it too much to ask for?

© 2013 IBM Corporation

Outlook:
Trends in commercial servers

© 2013 IBM Corporation

Three key trends in commercial servers

- Move towards “density optimized” servers
- ARM-based SoCs moving into server domain
- Off-chip network moving on-chip

© 2013 IBM Corporation

HPC vs overall server market

- **2012 server shipments: US$ 51.3B (IDC)**
- **HPC market ~ US$ 10B**
  - Supercomputer (> 500 K$): ~45%
  - Divisional (250-500 K$): ~13%
  - Departmental (100-250 K$): ~31%
  - Workgroup (< 100 K$): ~11%
- **Modular form factors**
  - Blade servers revenue grew 3.3% year over year to $2.4 billion (16.3% of total server revenue)
  - Density Optimized servers revenue grew 66.4% year over year in 4Q12 to $705 million (4.8% of all server revenue and 9.2% of all server shipments).

Source: IDC

© 2013 IBM Corporation

High-density server platforms based on embedded CPUs

- **AMD SeaMicro™**
  - AMD Opteron™ + SeaMicro™ "Freedom" 3D torus network
- **HP Moonshot**
  - Intel® Atom™ /Calxeda® + Ethernet & backplane Torus
- **IBM DOME/SKA**
  - Freescale™ + Ethernet
- **Intel®'s acquisition of Fulcrum/Ethernet, QLogic®/InfiniBand, Cray®/Aries**
  - Intel® Atom™ "Avoton" to have integrated Ethernet controller
- **How long before a small Ethernet or InfiniBand switch moves onto CPU?**
- **Mid-range commodity server business getting squeezed between high-end and embedded?**

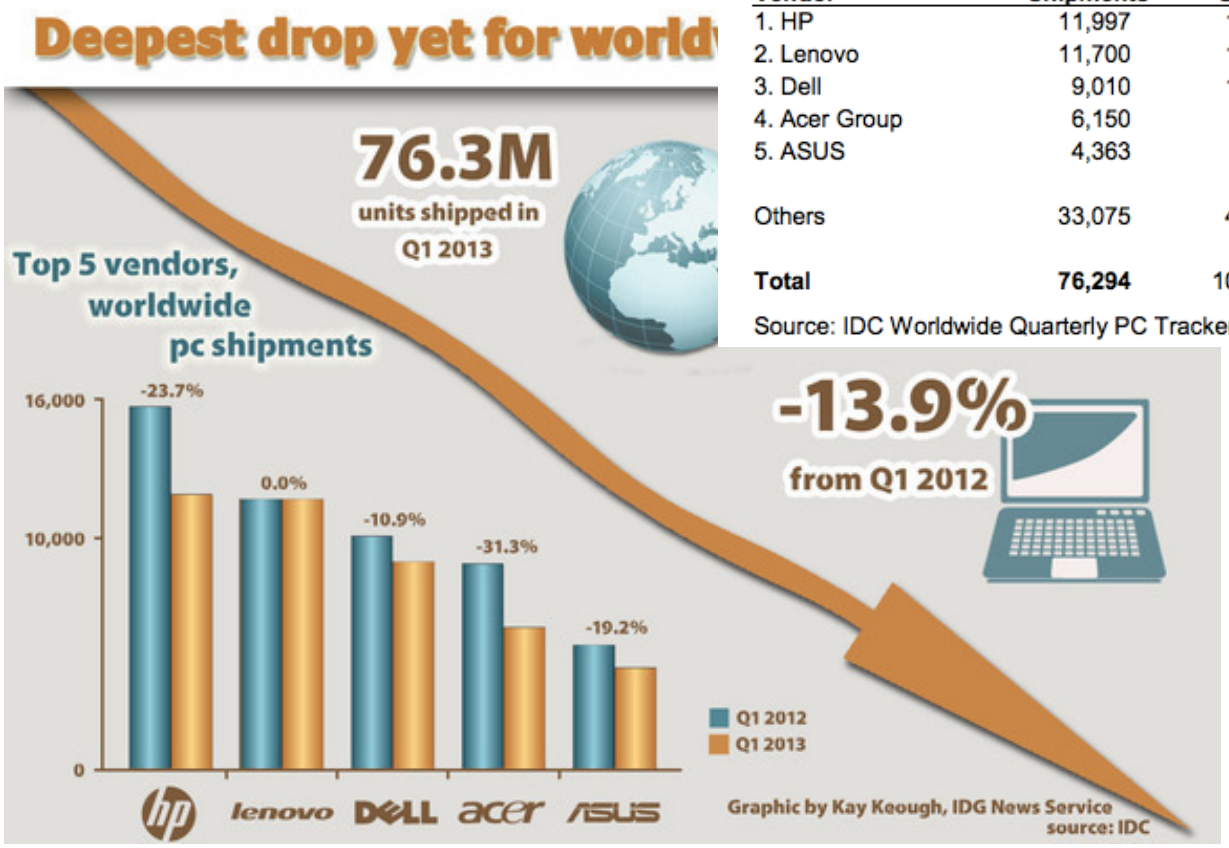
© 2013 IBM Corporation

PC market collapse erodes x86 volume base

Top 5 Vendors, Worldwide PC Shipments, First Quarter 2013 (Preliminary)
(Units Shipments are in thousands)

Vendor	1Q13 Shipments	1Q13 Market Share	1Q12 Shipments	1Q12 Market Share	1Q13/1Q12 Growth
1. HP	11,997	15.7%	15,726	17.7%	-23.7%
2. Lenovo	11,700	15.3%	11,705	13.2%	0.0%
3. Dell	9,010	11.8%	10,110	11.4%	-10.9%
4. Acer Group	6,150	8.1%	8,952	10.1%	-31.3%
5. ASUS	4,363	5.7%	5,401	6.1%	-19.2%
Others	33,075	43.4%	36,739	41.5%	-10.0%
Total	**76,294**	100.0%	88,635	100.0%	**-13.9%**

Source: IDC Worldwide Quarterly PC Tracker, April 10, 2013



Source: IDC

© 2013 IBM Corporation

Network moving onto processor

- **Driven by considerations of cost and power**
- **Convergence of computation and communication in silicon**
  - Integrated L1/L2/L3 caches; memory controller; PCIe controller; multi-chip interconnect (e.g. HT, QPI replacing FSB); network-on-chip
  - Integrated SMP fabric, I/O hub; NIC/CNA/HCA; special-purpose accelerators
- **High-end and "low"-end are at the vanguard of this trend**
  - Extreme scale HPC: BlueGene/Q
  - Low-end: embedded SoCs for mobile & server applications
  - In both cases, cost and power are critical
- **Commodity desktop & server processors are lagging somewhat**
  - But with Intel®'s recent networking acquisitions in Ethernet, IB & HPC we can expect this to change soon
  - HP Moonshot: example for HPC topology in commercial server platform

© 2013 IBM Corporation

Additional reading

- **William Dally and Brain Towles, "Principles and practices of interconnection networks," Morgan Kaufmann, 2004.**
  - Online lectures, EE382C: Interconnection Networks (Bill Dally, Stanford): http://classx.stanford.edu/ClassX/system/users/web/pg/view_subject.php?subject=EE382C_SPRING_2010_2011

- **Jose Duato, Sudhakar Yalamanchili, and Lionel Ni, "Interconnection networks: an engineering approach," Morgan Kaufmann, 2003.**

© 2013 IBM Corporation


Thank you!
sil@zurich.ibm.com
IBM