

""TSUBAME2.0, or the long road from
tiny clusters to Petascale, and its
Possible Contributions to High-
Resolution Natural Disaster
Simulations"

Satoshi Matsuoka

Global Scientific Information and Computing
Center (GSIC)

Tokyo Institute of Technology (Tokyo Tech.)

Charm++ Workshop, UIUC, 2010/04/19

"Accelerator"



- Special-purpose HW for accelerated computation
 - ▶ Small production units, special market, expensive
 - ▶ Lack of software inheritance over HW generations
- Restriction of application area
 - ▶ Only accelerates special type/portion of computation
 - ▶ Computation that "does not fit" impossible or slow
- Restriction of programming model
 - ▶ Specialized programming model, language
 - ▶ Restrictions on pointers, recursion, structures, etc.
 - ▶ No OS or other system software

GPUs as Modern-Day Vector Engines

Two types of HPC Workloads

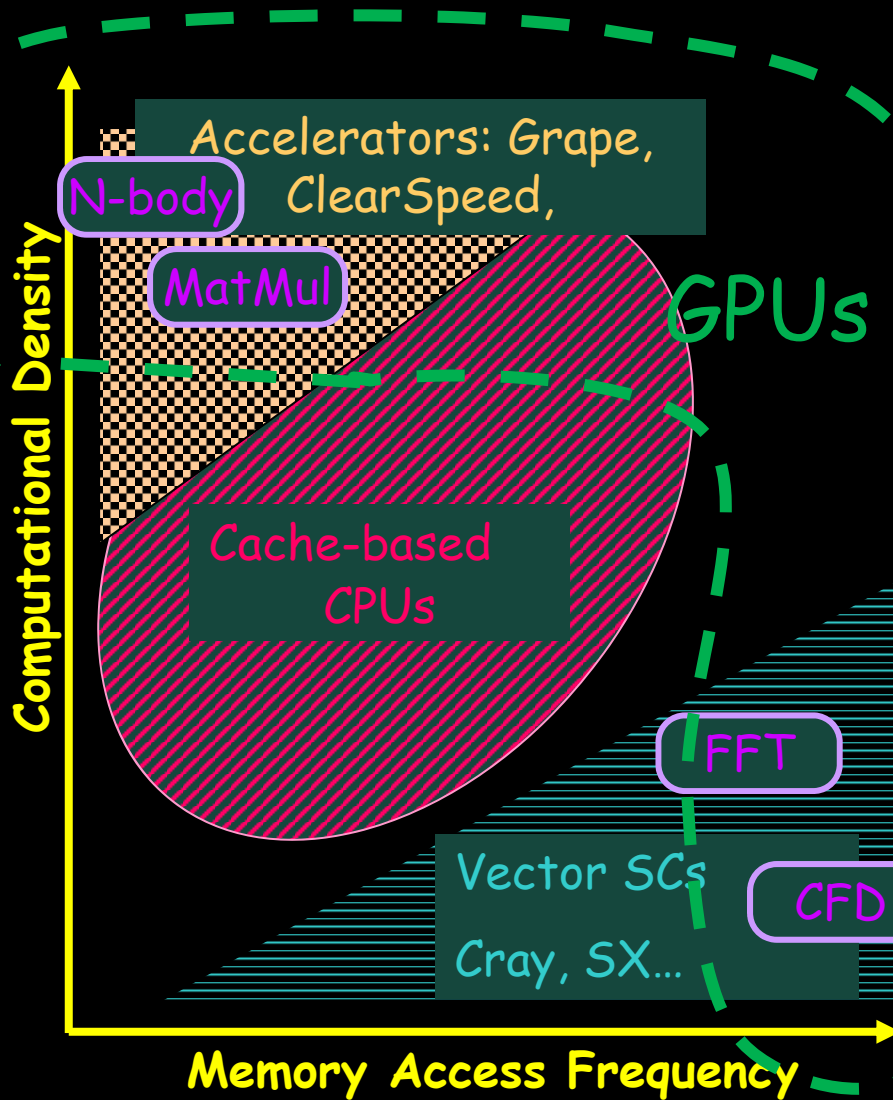
- High Computational Density
=> Traditional Accelerators
- High Memory Access Frequency
=> Traditional Vector SCs

Scalar CPUs are so-so at both
=> Massive system for speedup

GPUs are both modern-day vector engines and high compute density accelerators

=> Efficient Element of Next-Gen Supercomputers

Small memory, limited CPU-GPU BW, high-overhead communication with CPUs and other GPUs, lack of system-level SW support incl. fault tolerance, programming, ...

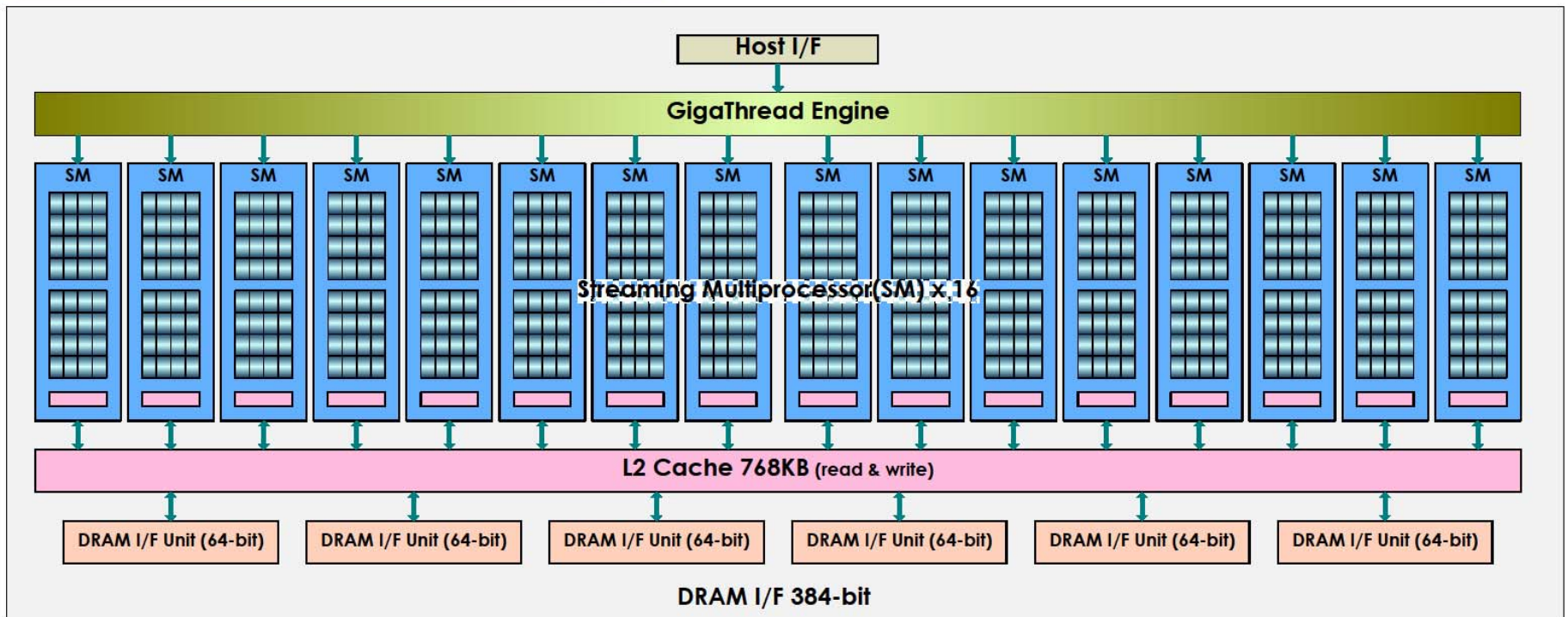


Our Research@
Tokyo Tech

NVIDIA Fermi

Many Core, Multithreaded, SIMD-Vector, MIMD Parallel Architecture

Fermi Overview

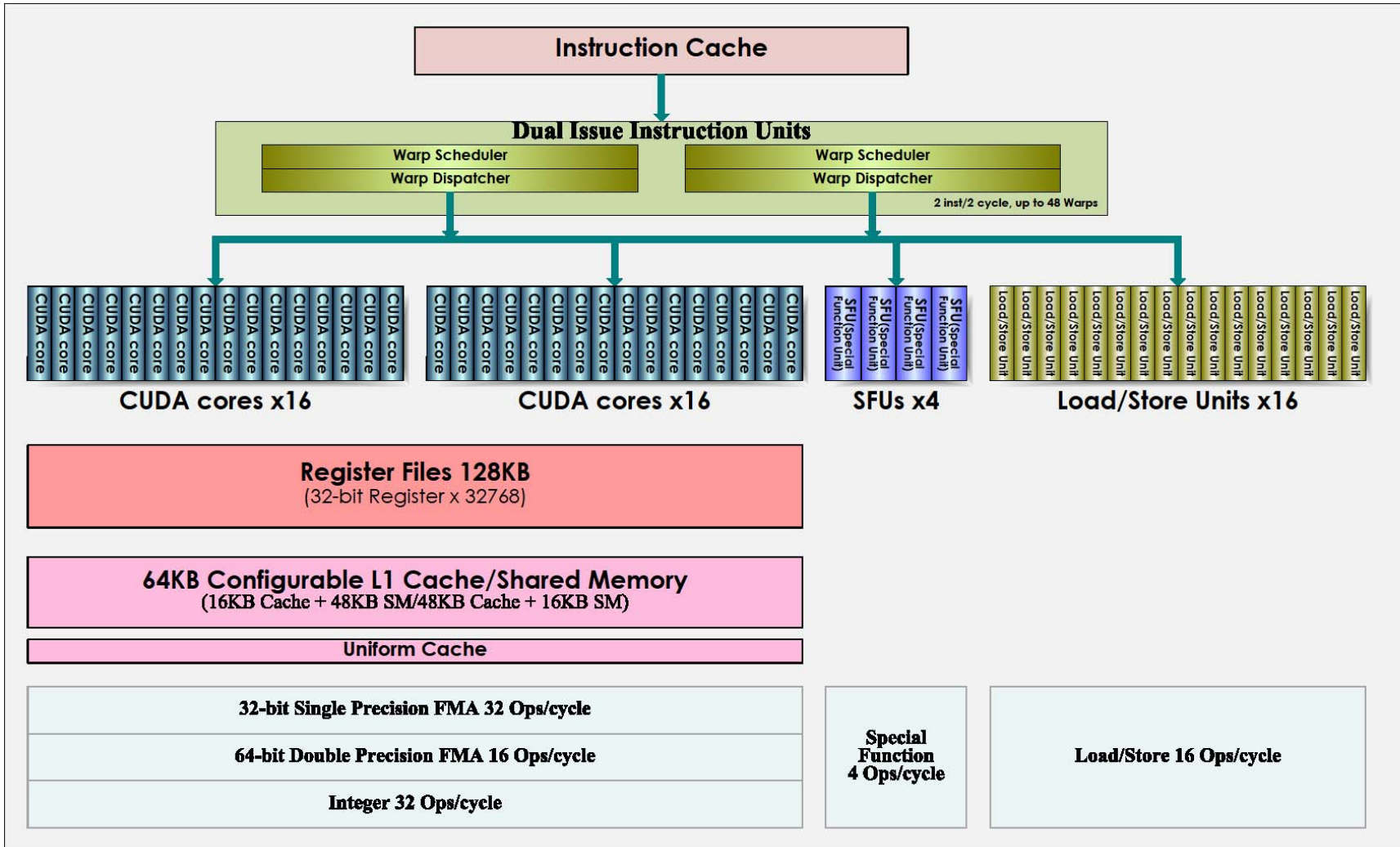


Copyright (c) 2009 Hiroshige Goto All rights reserved.

(Figure by Kazushige Goto)

NVIDIA Fermi SM "Core"

Fermi Streaming Multiprocessor(SM)



Parallelism in CUDA GPUs

- *SIMD-Vector Parallelism (WARP)*
 - 32 (16) way vector parallelism
- *SPMD Thread Parallelism*
 - Thousands of threads possible
 - Cyclic pipeline, hides memory latency, like vector processor but allows out-of-order execution
- *MIMD Parallelism (Kernel)*
 - Up to #SM (16 in Fermi) Kernels with independent instruction streams
- *GPU-CPU Hybrid Parallelism*
 - Streaming Data Transfer Engine via PCI-e
 - Massive Parallelism: GPU, Short Latency: CPU
 - In single chip sharing memory in Denver generation

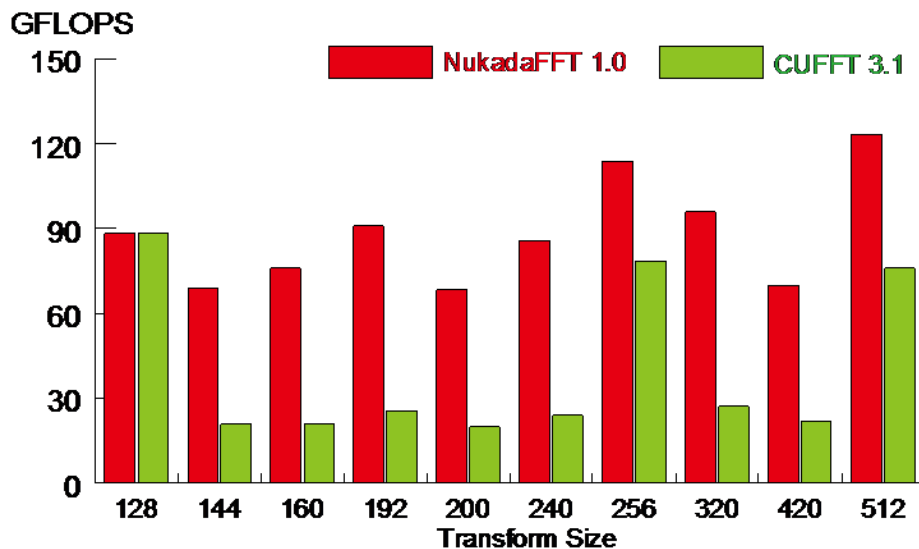


NukadaFFT 1.0 release !? [SC08,SC09]

NukadaFFT library is a very fast auto-tuning FFT library for CUDA GPUs.

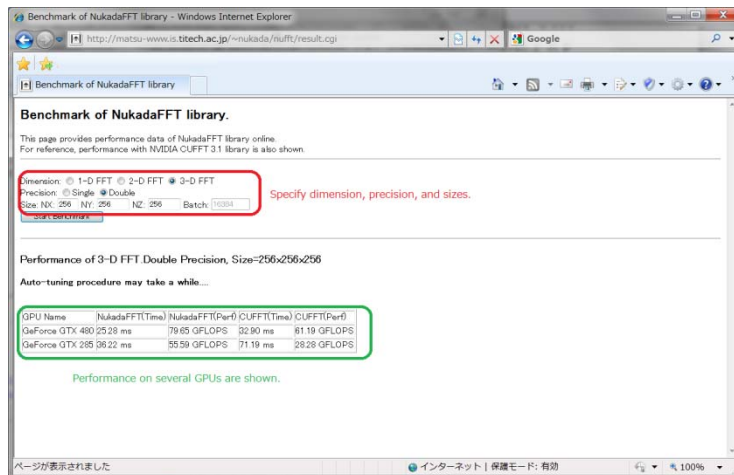
Tuning Parameters:

- (1) Factorization of transform size.
 - (2) # of threads launched per SM.
 - (3) Padding insertion patters for shared memory
- The library generates many kinds of FFT kernels and selects the fastest one. (exhaustive)



Performance of 1-D FFT.
(Double Precision, batch=32,768, GPU=GeForce GTX 480.)

For more details, you can see GTC 2010 Research Poster, or catch the author.



You can try online benchmarking as for the size you are interested in.

<http://matsu-www.is.titech.ac.jp/~nukada/nufft/>

4~5 times more power efficient than CPUs per socket

Compute Intensive or Memory Bound ?

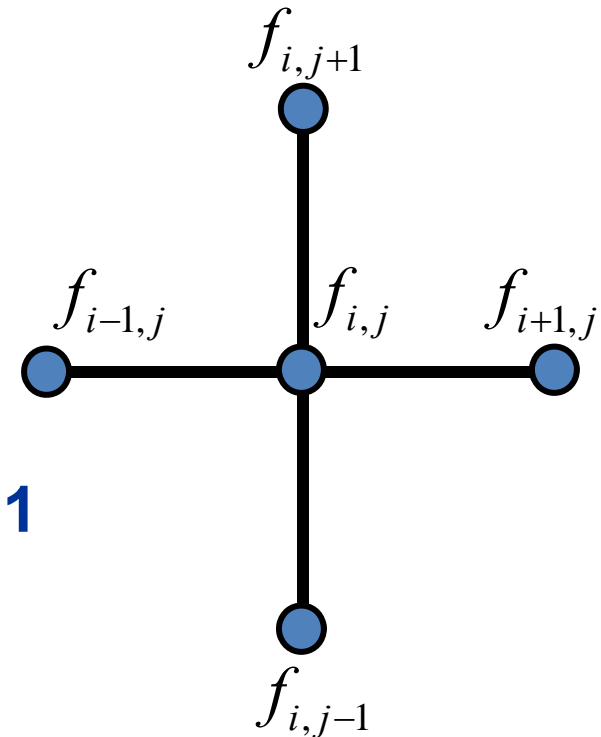


2-dimensional diffusion Equation

$$\frac{f_{i,j}^{n+1} - f_{i,j}^n}{\Delta t} = \kappa \left(\frac{f_{i+1,j}^n - 2f_{i,j}^n + f_{i-1,j}^n}{\Delta x^2} + \frac{f_{i,j+1}^n - 2f_{i,j}^n + f_{i,j-1}^n}{\Delta y^2} \right)$$



$$f_{i,j}^{n+1} = c_0 f_{i,j}^n + c_1 f_{i+1,j}^n + c_2 f_{i-1,j}^n + c_3 f_{i,j+1}^n + c_4 f_{i,j-1}^n$$



FLOP = 9

Byte = 4*6 = 24 byte : read 5, write 1

FLOP/Byte = 9/24 = 0.375

Arithmetic INTENSITY: FLOP/Byte



FLOP = number of FP operation for applications

Byte = Byte number of memory access for applications

F = Peak Performance of floating point operation

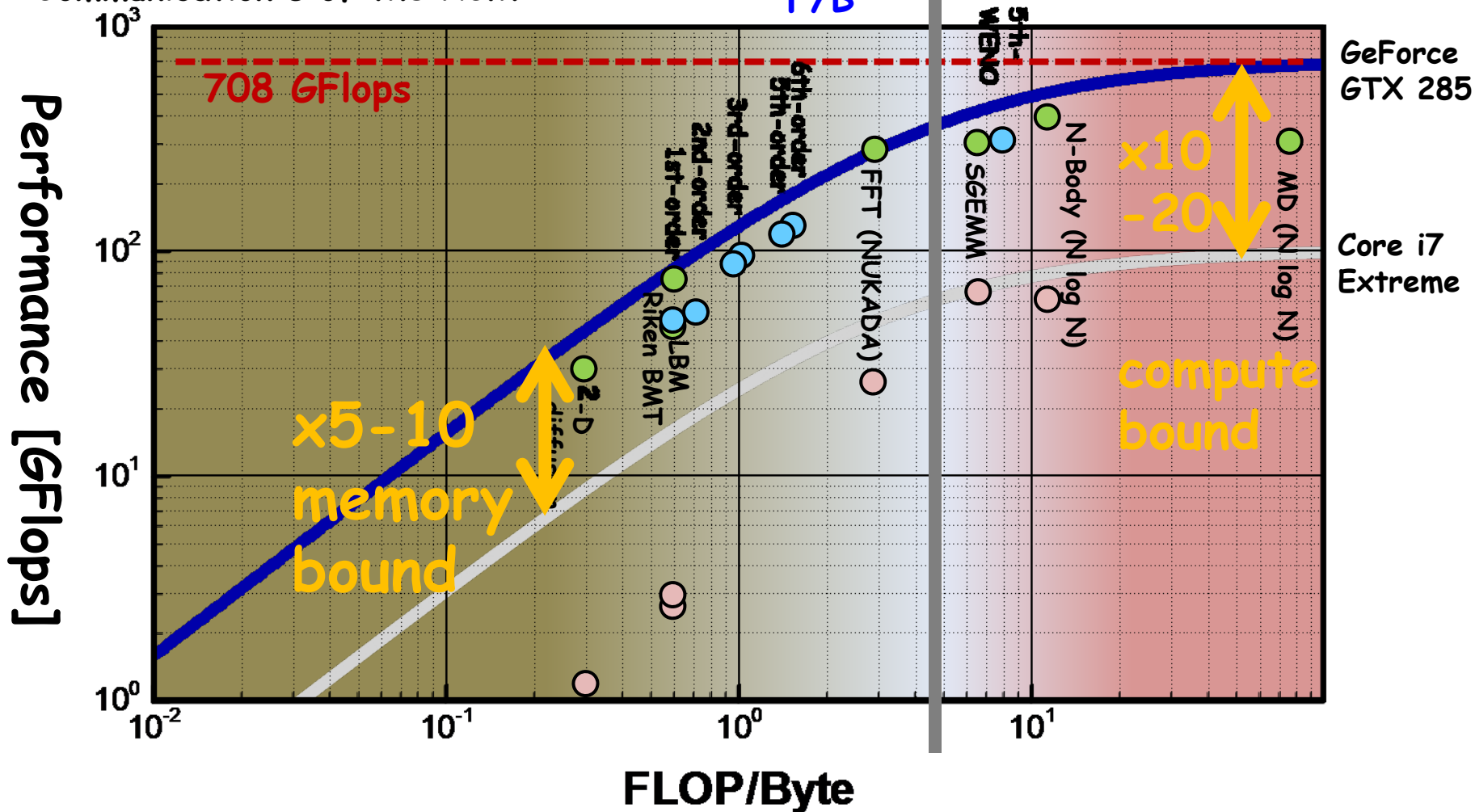
B = Peak Memory Bandwidth

$$\begin{aligned} \text{Performance} &= \frac{\text{FLOP}}{\text{FLOP}/F + \text{Byte}/B + \alpha} \\ &= \frac{\text{FLOP/Byte}}{\text{FLOP/Byte} + F/B + \alpha} F \end{aligned}$$

GPU vs. CPU Performance

Roofline model: Williams, Patterson 2008
 Communications of the ACM

$$\text{FLOP/Byte} = \frac{F}{B}$$



The TSUBAME 1.0 "Supercomputing Grid Cluster" Spring 2006

Unified IB network

Voltaire ISR9288 Infiniband 10Gbps
x2 (DDR next ver.)
~1310+50 Ports
~13.5Terabits/s (3Tbits bisection)



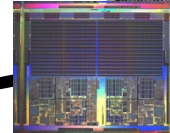
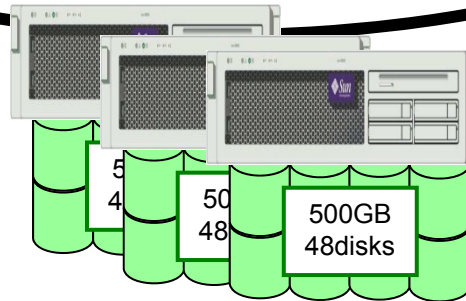
*"Fastest
Supercomputer in
Asia" 7th on the 27th
Top500 @38.18TF*

Sun Galaxy 4 (Opteron Dual
core 8-socket)
10480core/655Nodes
21.4Terabytes
50.4TeraFlops
OS Linux (SuSE 9, 10)
NAREGI Grid MW

10Gbps External
Network



NEC SX-8i
(for porting)



Storage

1.0 Petabyte (Sun "Thumper")

0.1Petabyte (NEC iStore)

Lustre FS, NFS, WebDAV (over IP)

50GB/s aggregate I/O BW

ClearSpeed CSX600

SIMD accelerator

360 boards,

35TeraFlops(Current))

TSUBAME 1.2 Experimental Evolution (Oct. 2008)

The first "Petascale" SC in Japan



Voltaire ISR9288 Infiniband x8
10Gbps x2 ~1310+50 Ports
~13.5Terabits/s
(3Tbits bisection)



NEC SX-8i



Storage

1.5 Petabyte (Sun x4500 x 60)

0.1Petabyte (NEC iStore)

Lustre FS, NFS, CIF, WebDAV (over IP)

60GB/s aggregate I/O BW

Sun x4600 (16 Opteron Cores)

32~128 GBytes/Node

10480core/655Nodes

21.4TeraBytes

50.4TeraFlops

OS Linux (SuSE 9, 10)

NAREGI Grid MW



PCI-e

ClearSpeed CSX600

SIMD accelerator

648 boards,

52.2TeraFlops

SFP/DFP

**87.1TF Linpack (The First GPU
Supercomputer on Top500)
[IPDPS10]**

**NEW Deploy:
GCOE TSUBASA
Harpertown-Xeon
90Node 720CPU
8.2TeraFlops**



**170 Nvidia Tesla 1070, ~680 Tesla cards
High Performance in Many BW-Intensive Apps
10% power increase over TSUBAME 1.0**

10Gbps+External NW

Unified Infiniband
network



680 Unit Tesla Installation...
While TSUBAME in Production Service (!)



Case for Hybrid Multi/Many-Core Architectures

- Apps scaling governed by two components
 - Weak Scaling part, $O(N)$ concurrency
 - Strong (Finite) Scaling Part, $O(K)$ concurrency

$N \gg K$ for
peta-exa

- S : speedup, H : fraction of $O(N)$ execution, h : #weak-scale cores, f : #strong scale cores, c : speedup of strong scaling cores over weak scaling cores

$$S = \frac{1}{\frac{1-H}{fc} + \frac{H}{h}}$$

- 10PF machine analysis
 - Hybrid (TSUBAME2+): $h=500,000$, $f=10,000$, $c=4$, $H=95\%$
 $\Rightarrow S=317,460$ (Efficiency 63.4%)
 - Homo (BG/Q): $h=800,000$, $f=10,000$, $c=1$, $H=95\%$
 $\Rightarrow S=161,616$ (Efficiency 20.2%)
 - Gap will widen with Exascale, 1-10 billion cores

DOE SC Applications Overview

(following slides courtesy John Shalf @ LBL NERSC)

NAME	Discipline	Problem/Method	Structure
MADCAP	Cosmology	CMB Analysis	Dense Matrix
FVCAM	Climate Modeling	AGCM	3D Grid
CACTUS	Astrophysics	General Relativity	3D Grid
LBMHD	Plasma Physics	MHD	2D/3D Lattice
GTC	Magnetic Fusion	Vlasov-Poisson	Particle in Cell
PARATEC	Material Science	DFT	Fourier/Grid
SuperLU	Multi-Discipline	LU Factorization	Sparse Matrix
PMEMD	Life Sciences	Molecular Dynamics	Particle

Latency Bound vs. Bandwidth Bound?

- How large does a message have to be in order to saturate a dedicated circuit on the interconnect?
 - $N^{1/2}$ from the early days of vector computing
 - Bandwidth Delay Product in TCP

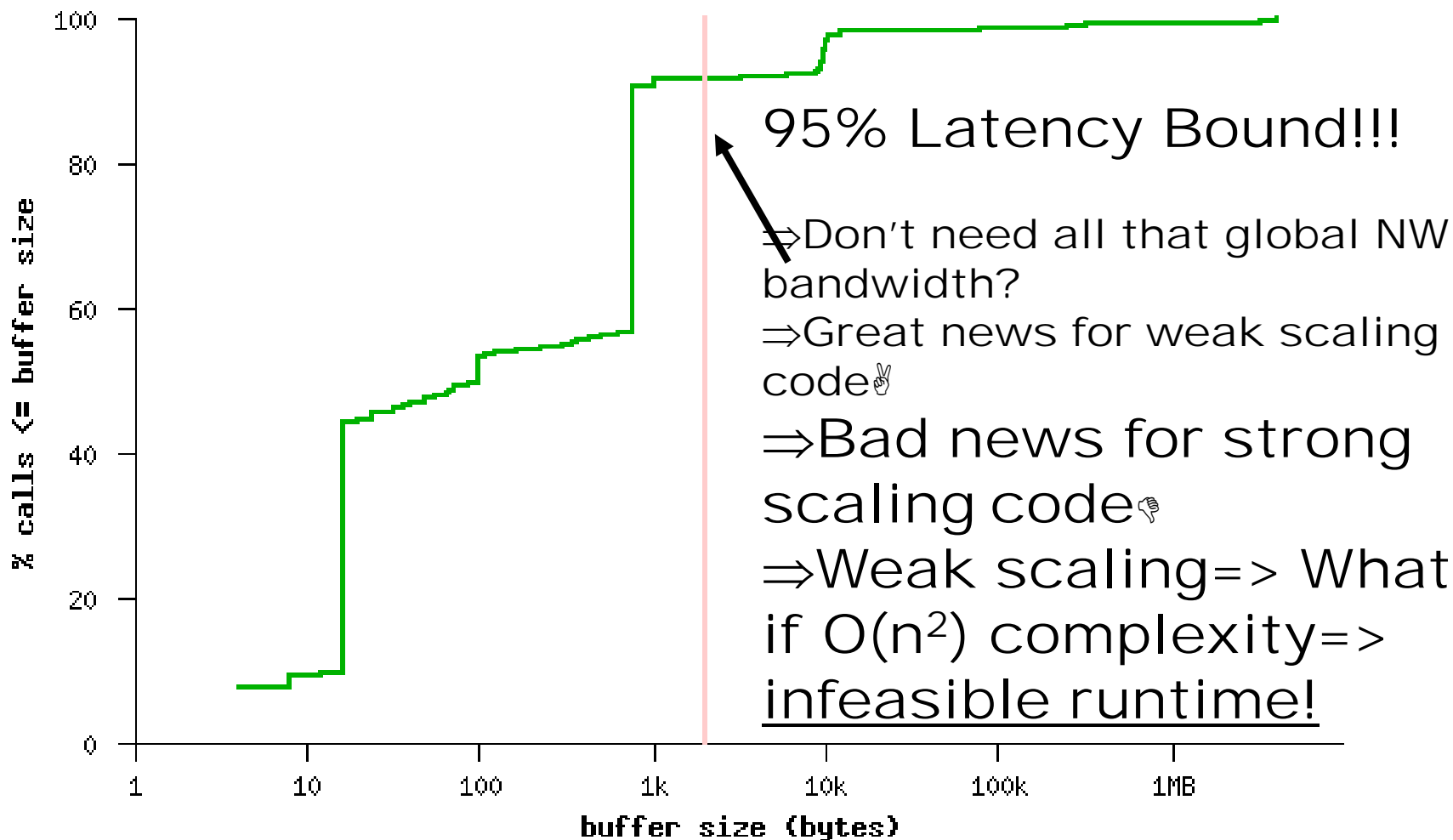
System	Technology	MPI Latency	Peak Bandwidth	Bandwidth Delay Product
SGI Altix	Numalink-4	1.1us	1.9GB/s	2KB
Cray X1	Cray Custom	7.3us	6.3GB/s	46KB
NEC ES	NEC Custom	5.6us	1.5GB/s	8.4KB
Myrinet Cluster	Myrinet 2000	5.7us	500MB/s	2.8KB
Cray XD1	RapidArray/IB4x	1.7us	2GB/s	3.4KB

- Bandwidth Bound if msg size $>$ Bandwidth*Delay
- Latency Bound if msg size $<$ Bandwidth*Delay
 - Except if pipelined (*unlikely with MPI due to overhead*)
 - W/HW DMA a few 100ns but not much more

Collective Buffer Sizes are Small(!)

- demise of message passing in strong scaling -

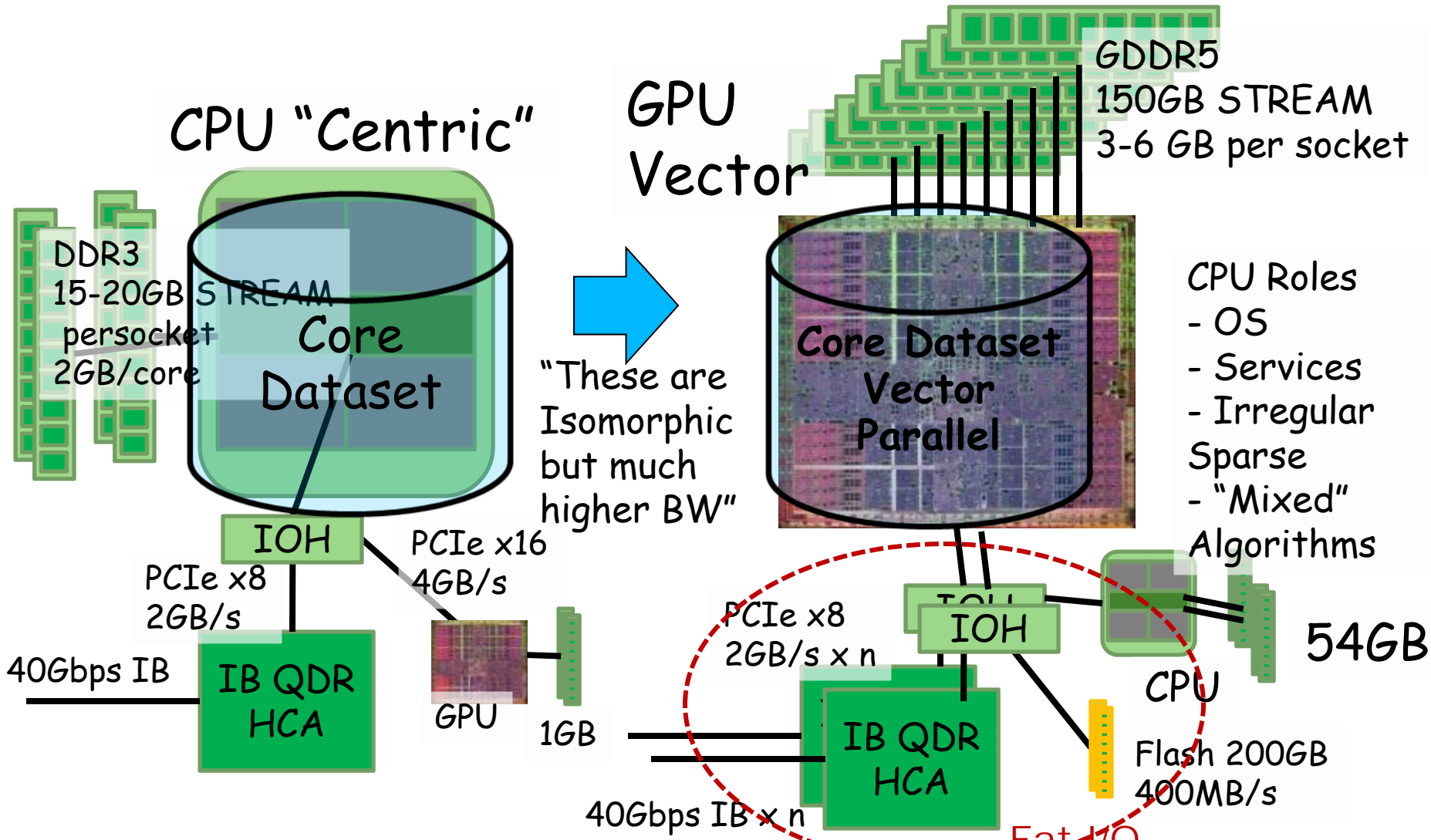
Collective Buffer Sizes for All Codes



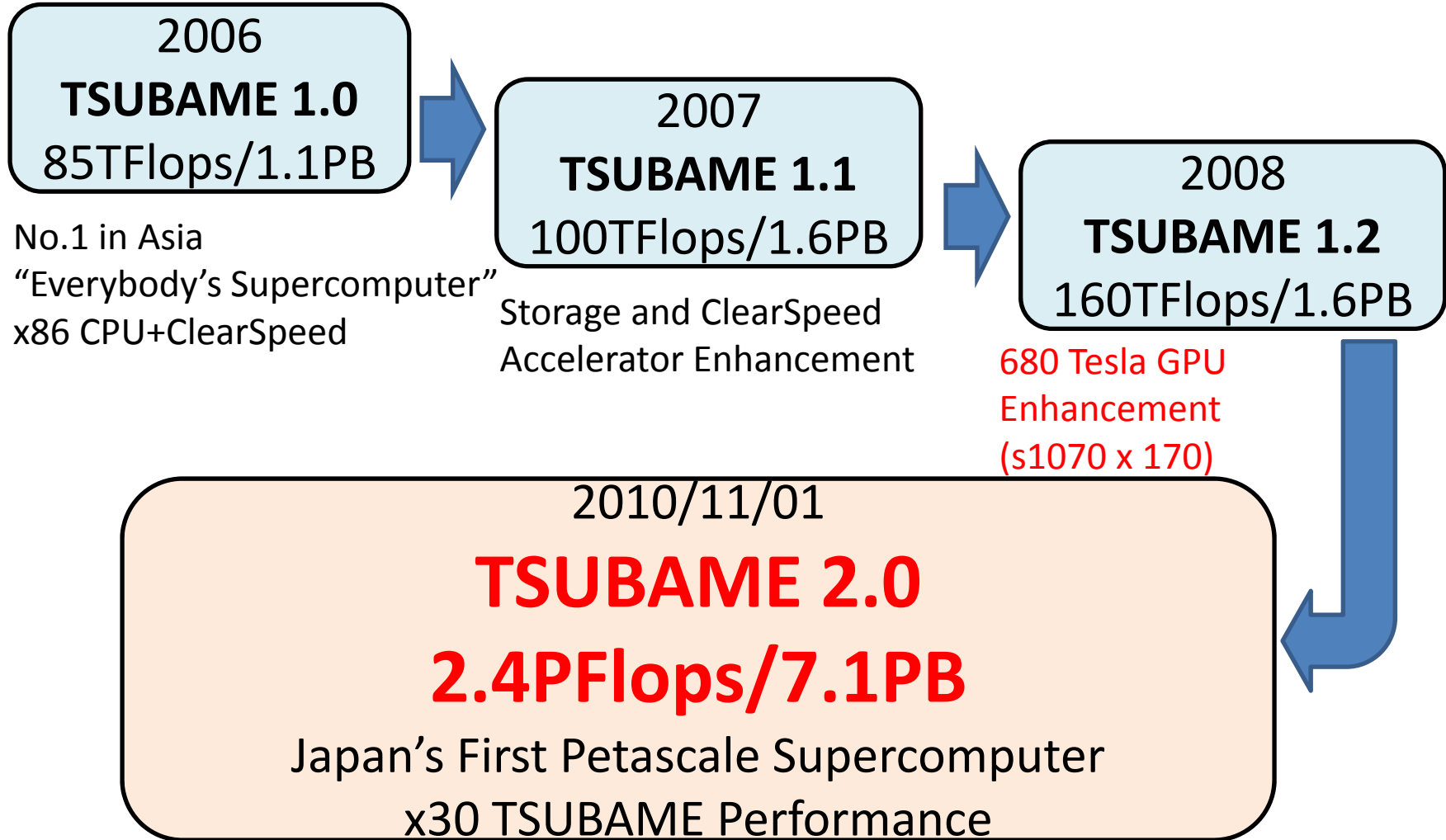
Scaling Peta to Exa Design?

- Shorten latency as much as possible
 - Extreme multi-core incl. vectors
 - "Fat" nodes, exploit short-distance interconnection
 - Direct cross-node DMA (e.g., put/get for PGAS)
- Hide latency if cannot be shortened
 - Dynamic multithreading (Old: dataflow, New: GPUs)
 - Trade Bandwidth for Latency (so we do need BW...)
 - Departure from simple mesh system scaling
- Change Latency-Starved Algorithms
 - From implicit Methods to direct/hybrid methods
 - Structural locality, extrapolation, stochastics (MC)
 - Need good programming model/lang/system for this...

From TSUBAME 1.2 to 2.0: From CPU Centric to GPU Centric Nodes for Scaling High Bandwidth in Network & I/O!!



History of TSUBAME



TSUBAME2.0 Nov. 1, 2011

See Live @ Tokyo Tech Booth #1127



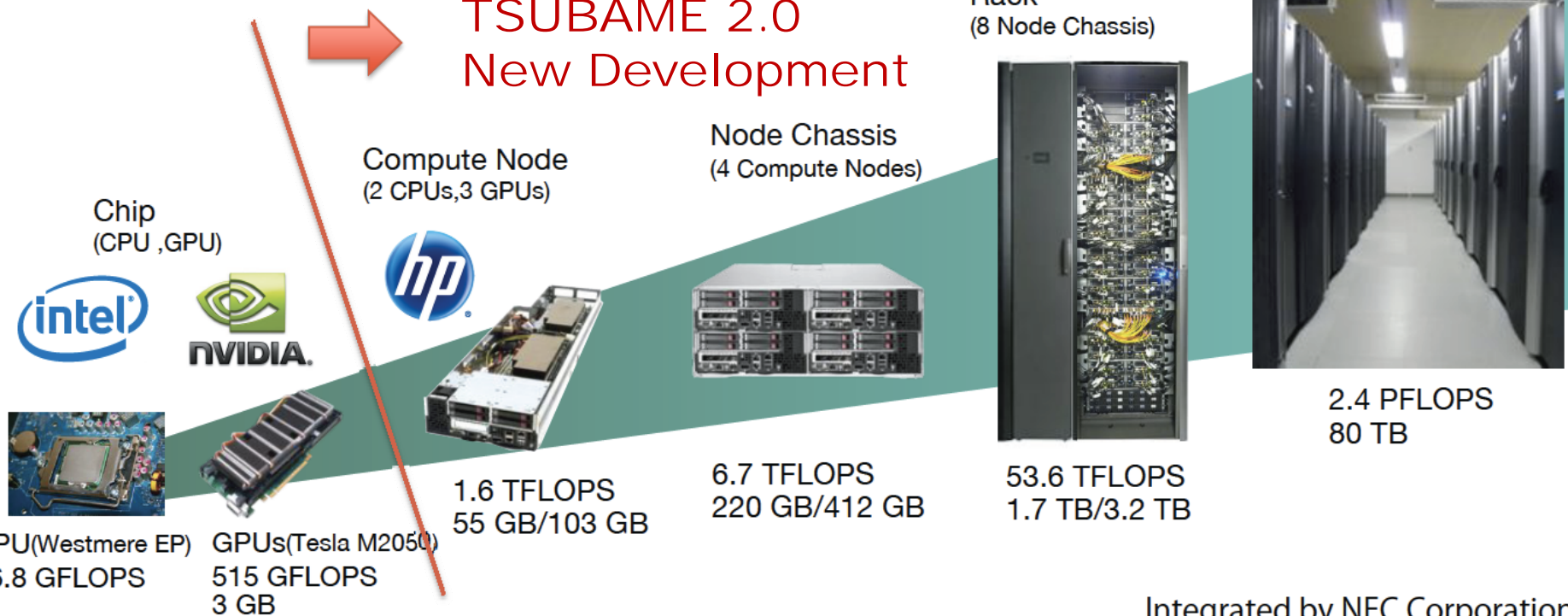
TSUBAME2.0: A GPU-centric Green 2.4 Petaflops Supercomputer

Tsubame 2.0: "Tiny" footprint, very power efficient

- Floorspace less than 200m² (2,100 ft²)
- Top-class power efficient machine on the Green 500

System
 (42 Racks)
 1408 GPU Compute Nodes,
 34 Nehalem "Fat Memory" Nodes

TSUBAME 2.0 New Development



Integrated by NEC Corporation

Highlights of TSUBAME 2.0 Design (Oct. 2010) w/NEC-HP

2.4 PF multi-core x86 + Fermi GPGPU

- ▶ 1432 nodes, Intel Westmere/Nehalem EX
- ▶ 4224 NVIDIA Tesla (Fermi) M2050 GPUs
- ▶ >100,000 total CPU and GPU "cores", High Bandwidth
- ▶ **1.9 million "CUDA cores", 32K x 4K = 130 million CUDA threads(!)**



0.72 Petabyte/s aggregate mem BW,

- ▶ Effective 0.3-0.5 Bytes/Flop, restrained memory capacity (100TB)

Optical Dual-Rail IB-QDR BW, full bisection BW(Fat Tree)

- ▶ **200Tbits/s**, Likely fastest in the world, still scalable

Flash/node, ~200TB (1PB in future), 660GB/s I/O BW

- ▶ >7 PB IB attached HDDs, 15PB Total HFS incl. LTO tape

Low power & efficient cooling, comparable to TSUBAME 1.0 (~1MW); **PUE = 1.28** (60% better c.f. TSUBAME1)

Virtualization and Dynamic Provisioning of **Windows HPC** + Linux job migration etc

TSUBAME2.0 Global Partnership

NEC: Main Contractor, overall system integration, Cloud-SC mgmt.

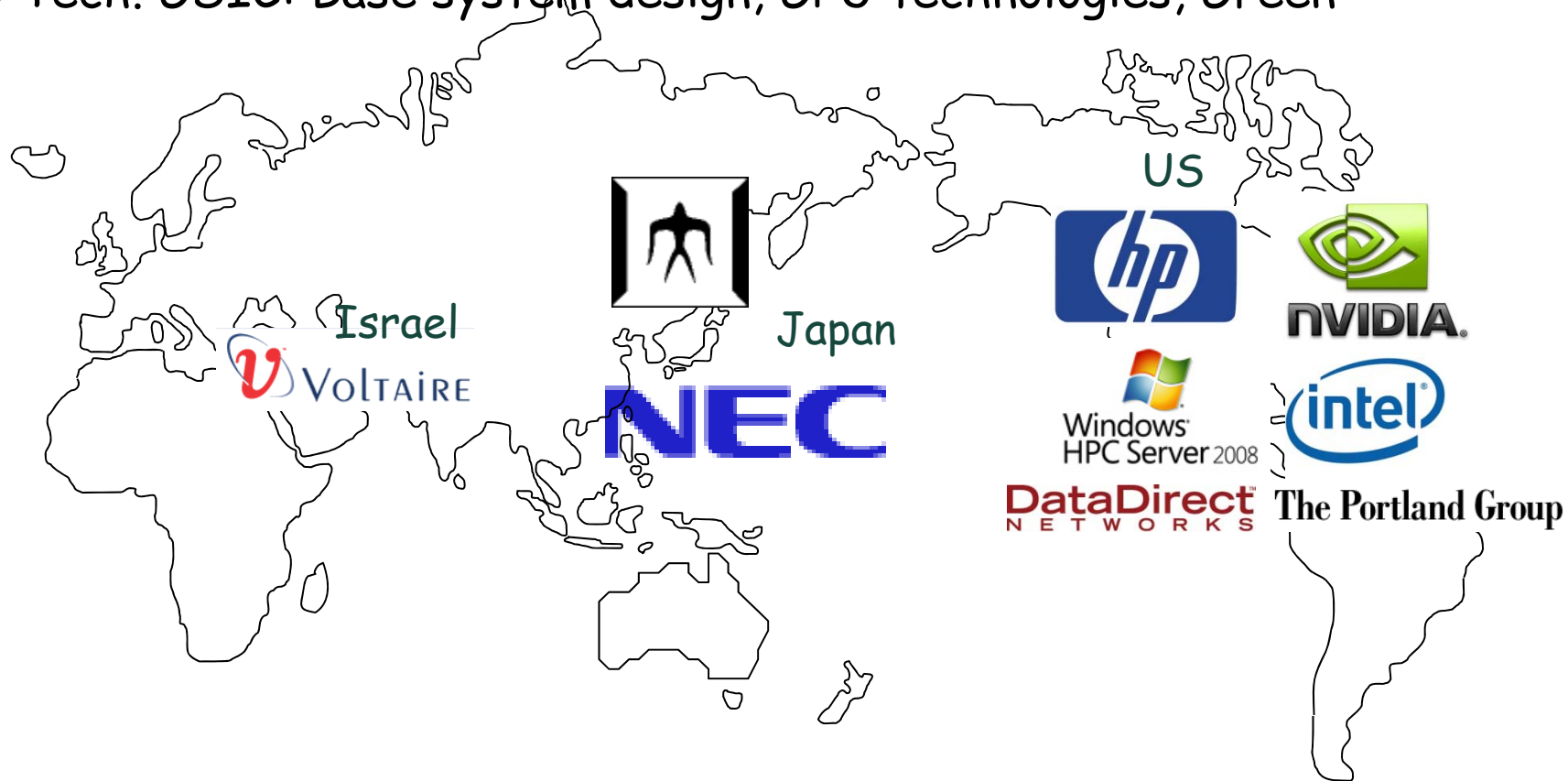
HP: Node R&D, Green; Microsoft: WindowsHPC, Cloud & VM

NVIDIA: Fermi GPUs, CUDA; Voltaire: QDR Infiniband Network

DDN: Large-Scale Storage; Intel: Westmere & Nehalem-EX CPUs

PGI: GPU Vectorizing Compiler

Tokyo Tech. GSIC: Base system design, GPU technologies, Green

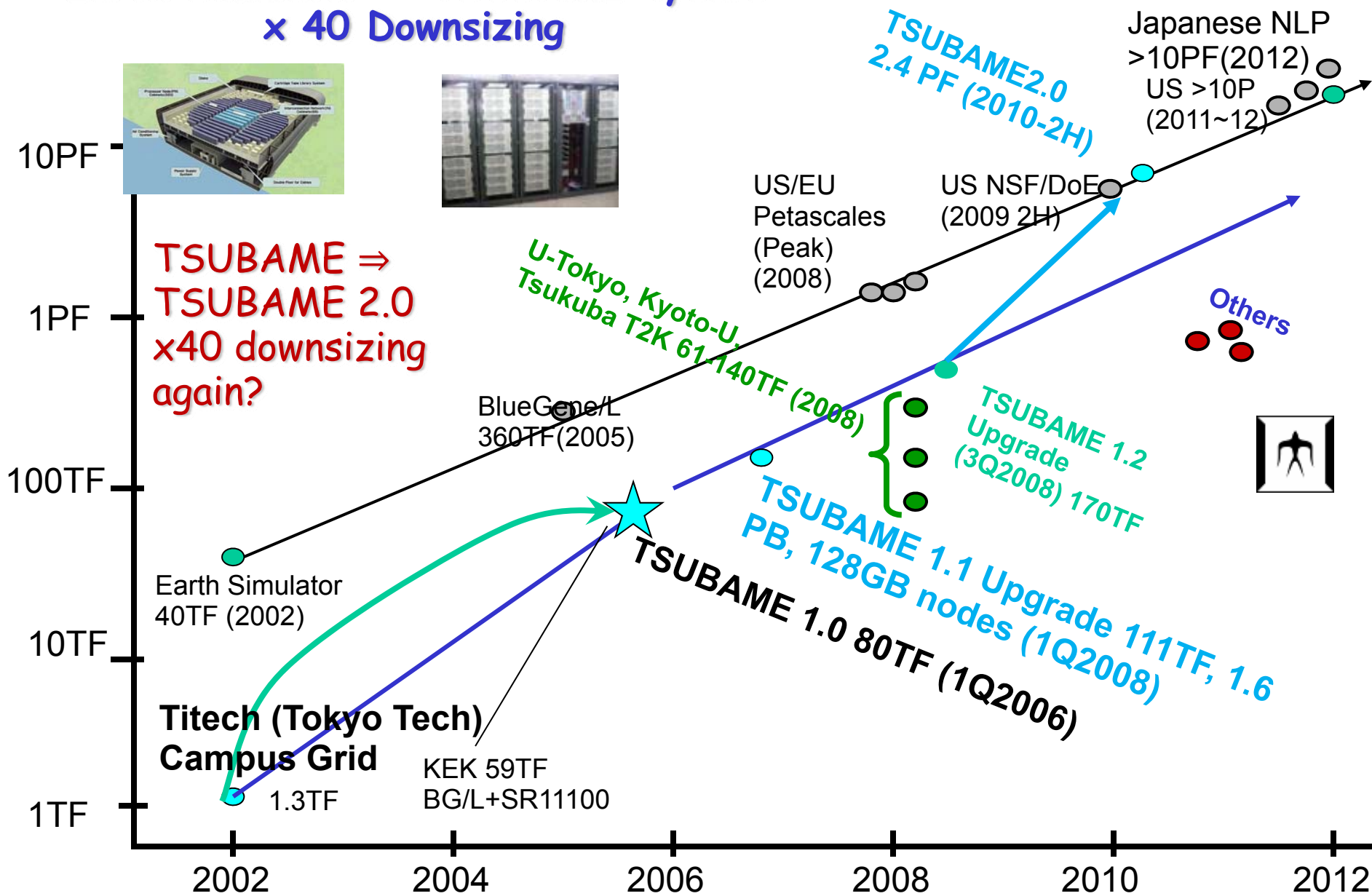


TSUBAME 2.0 Performance

Earth Simulator \Rightarrow TSUBAME 4years
 \times 40 Downsizing



TSUBAME \Rightarrow
 TSUBAME 2.0
 \times 40 downsizing
 again?

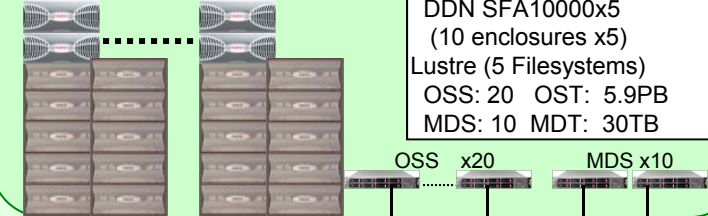


TSUBAME2.0 System Overview (2.4 Pflops/15PB)

Petascale Storage : Total **7.13PB**(Lustre + Accelerated NFS Home)

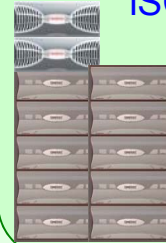
Lustre Partition 5.93PB

x5



MDS,OSS
HP DL360 G6 30nodes
Storage
DDN SFA10000x5
(10 enclosures x5)
Lustre (5 Filesystems)
OSS: 20 OST: 5.9PB
MDS: 10 MDT: 30TB

Home NFS/
iSCSI



Storage Server
HP DL380 G6 4nodes
BlueArc Mercury 100 x2
Storage
DDN SFA10000 x1
(10 enclosures x1)

Tape System
Sun SL8500 8PB

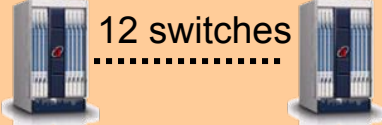
SuperTitenet

SuperSinet3

Node Interconnect: **Optical, Full Bisection, Non-Blocking, Dual-Rail QDR**

Core Switch

12 switches



Voltaire Grid Director 4700
IB QDR: 324ports

Edge Switch

179 switches



Voltaire Grid Director 4036
IB QDR : 36 ports

Edge Switch(w/10GbE)

6 switches



Voltaire Grid Director 4036E
IB QDR:34ports
10GbE: 2ports

Mgmt Servers

Compute Nodes : **2.4PFlops(CPU+GPU)**

1.69TFlops(CPU)

"Thin" Nodes



1408nodes
(32node x44 Racks)

NEW DESIGN Hewlett Packard CPU+GPU
High BW Compute Nodes x 1408
Intel Westmere-EP 2.93GHz
(TB 3.196GHz) 12Cores/node
Mem:55.8GB(=52GiB)or 103GB(=96GiB)
GPU NVIDIA M2050 515GFlops,
3GPUs/node
SSD 60GB x 2 120GB *55.8GB node
120GB x 2 240GB *103GB node
OS: Suse Linux Enterprise + Windows

HPC
4224 NVIDIA "Fermi" GPUs
Memory Total : 80.55TB
SSD Total : 173.88TB

"Medium" Nodes



24 nodes 6.14TFLOPS

HP 4Socket Server 10nodes
CPU Nehalem-EX 2.0GHz
32Cores/node
Mem:137GB(=128GiB)
SSD 120GB x 4 480GB
OS: Suse Linux Enterprise

PCI-E gen2 x16 x2slot/node

GSIC:NVIDIA Tesla S1070GPU (34 units)

"Fat" Nodes

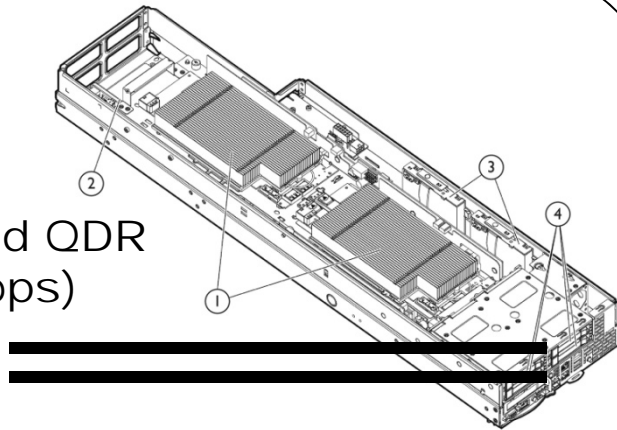


10 nodes 2.56TFLOPS

HP 4Socket Server 10nodes
CPU Nehalem-EX 2.0GHz
32Core/node
Mem:274GB(=256GiB)x8
549GB(=512GiB) x2
SSD 120GB x 4 480GB
OS: Suse Linux Enterprise

TSUBAME2.0 Compute Nodes

Thin Node



Infiniband QDR
x2 (80Gbps)

HP SL390G7 (Developed for
TSUBAME 2.0)

GPU: NVIDIA Fermi M2050 x 3
515GFlops, 3GByte memory /GPU
CPU: Intel Westmere-EP 2.93GHz x2
(12cores/node)
Memory: 54, 96 GB DDR3-1333
SSD: 60GBx2, 120GBx2



1408nodes:

4224GPUs: 59,136 SIMD Vector
Cores, 2175.36TFlops (Double FP)

2816CPUs, 16,896 Scalar Cores:
215.99TFlops

Total: 2391.35TFLOPS

Memory: 80.6TB (CPU) + 12.7TB
(GPU)

SSD: 173.9TB

IB QDR



PCI-e Gen2x16 x2
NVIDIA Tesla
S1070 GPU

HP 4 Socket Server
CPU: Intel Nehalem-EX 2.0GHz x4
(32cores/node)
Memory: 128, 256, 512GB DDR3-1066
SSD: 120GB x4 (480GB/node)

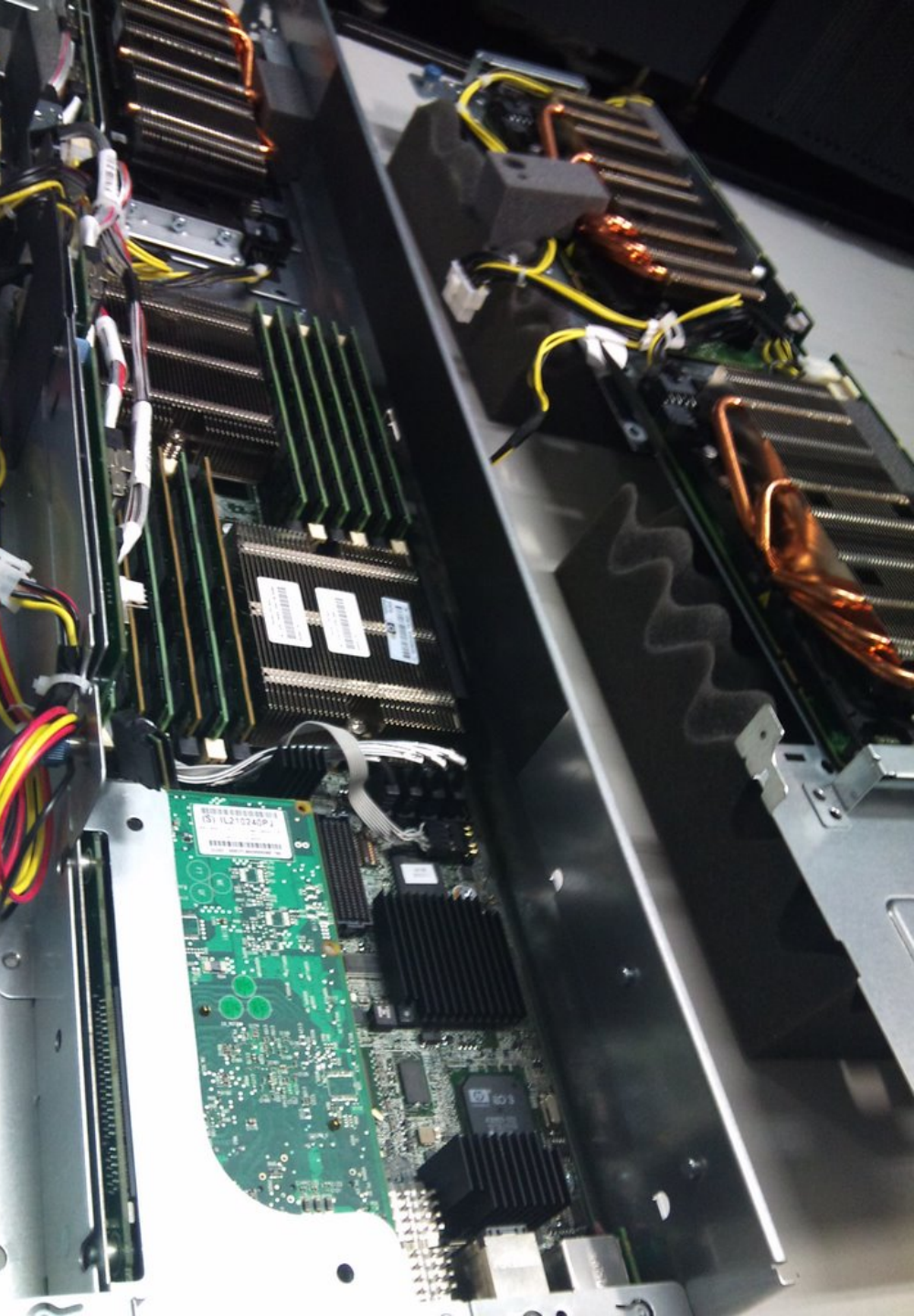


34 nodes:
8.7TFlops

Memory:
6.0TB+GPU

SSD: 16TB+

Total Perf
2.4PFlops
Mem: ~100TB
SSD: ~200TB



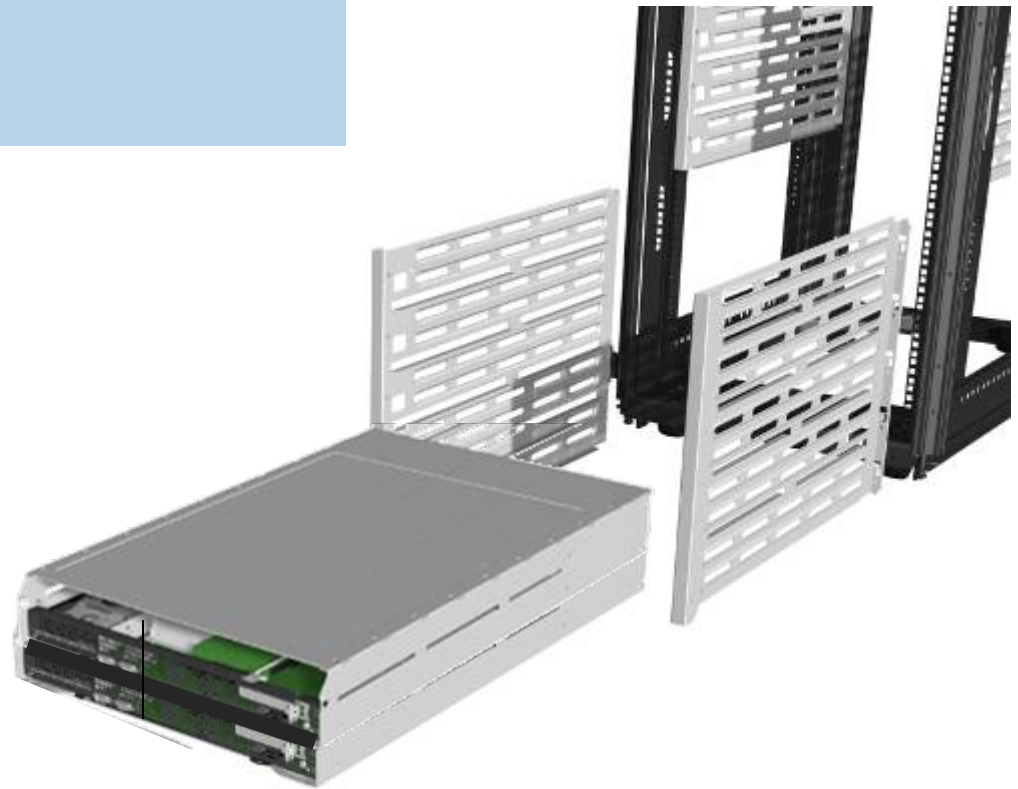


GPU-optimized node in HP Skinless Packaging

Typical Tsubame2 node:

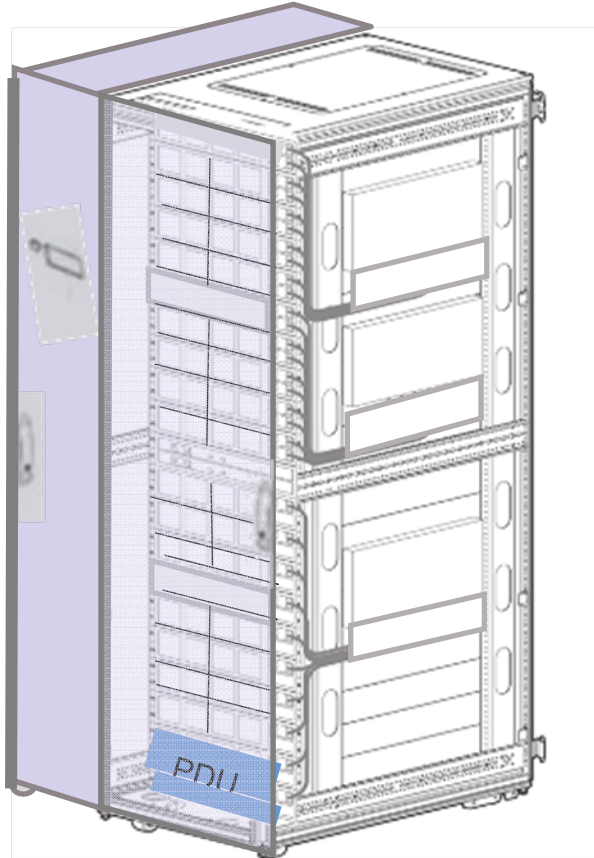
- 2 Intel® Xeon® Processor X5670 (six-core 2.93Ghz)
- 54 GB or 96 GB memory
- 2 SSDs per node
- Dual Rail IB
- 3 NVIDIA M2050 GPU board

- Nodes go into chassis with shared fans and power
- Chassis mounted into lightweight racks
- Easy assembly
- Lower weight
- Reduced heat retention
- Modular and flexible
- Standard 19" racks



Compute Rack Building Block for Tsubame2

HP Modular Cooling System
G2 rack



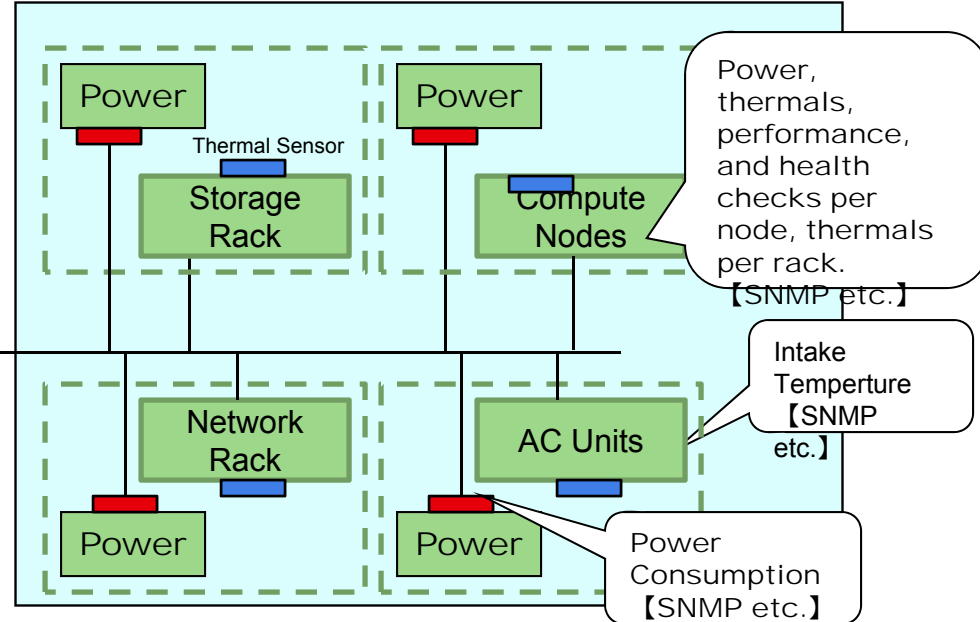
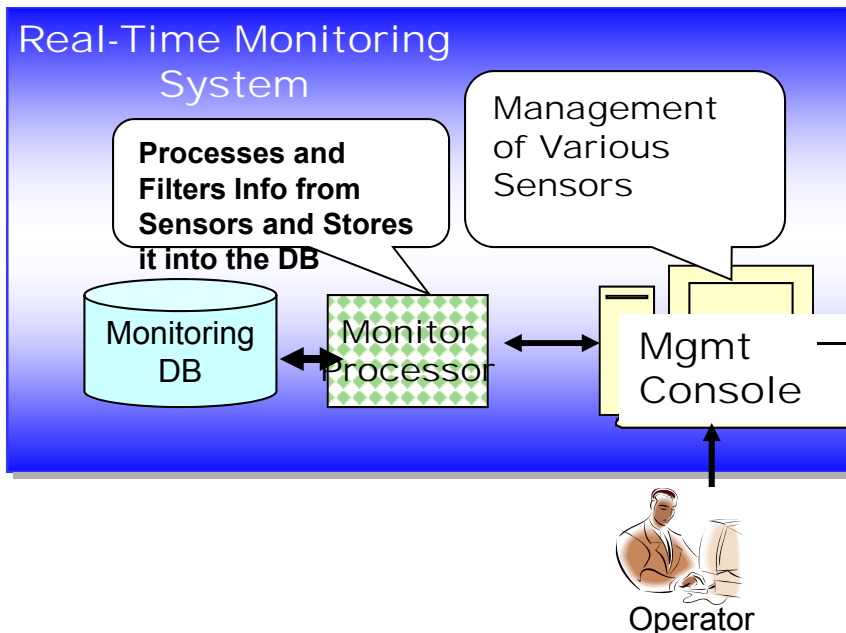
- 42U HP Modular Cooling System G2 rack
 - 30 nodes per rack – 90 GPUs
 - 8 chassis with Advanced Power Management
 - 1 HP Network Switch for shared console and admin local area network
 - 2 Airflow Dam (Liquid Cooled)
 - 4 Voltaire 4036 Leaf Switch
 - Power distribution units
- Power per rack approximately

35KW

Green and Reliable SC

- ▶ Monitoring Sensors (incl. GPUs)
 - ▶ Thermals
 - ▶ Power Consumption
 - ▶ Utilization
 - ▶ HW and SW Health checks

- Advanced Power Management
 - Dynamic Power Capping
 - Power monitoring
 - Node level power off/on
- Thermal logic technologies
 - Shared power and fans
 - Energy-efficient fans
 - Three-phase load balancing
- 94% Platinum Common Slot Power Supplies



Cooling: Enhanced HP Modular Cooling System G2

HP's **water-cooled** rack

Completely closed racks with their own heat exchanger.

1.5 x width of normal rack+rear ext.

Cooling for high density deployments

35kW of cooling capacity single rack

- **Highest Rack Heat Density ever**
- **3000CFM Intake airflow with 7C chiller water**

up to 2000 lbs of IT equipment

Uniform air flow across the front of the servers

Automatic door opening mechanism controlling both racks

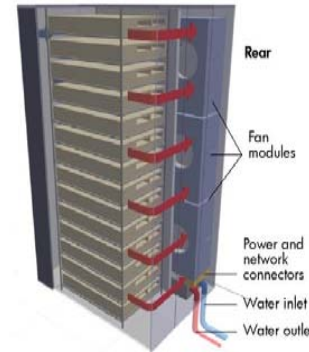
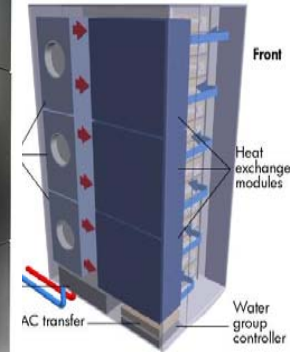
Adjustable temperature set point

Removes 95% to 97% of heat inside racks

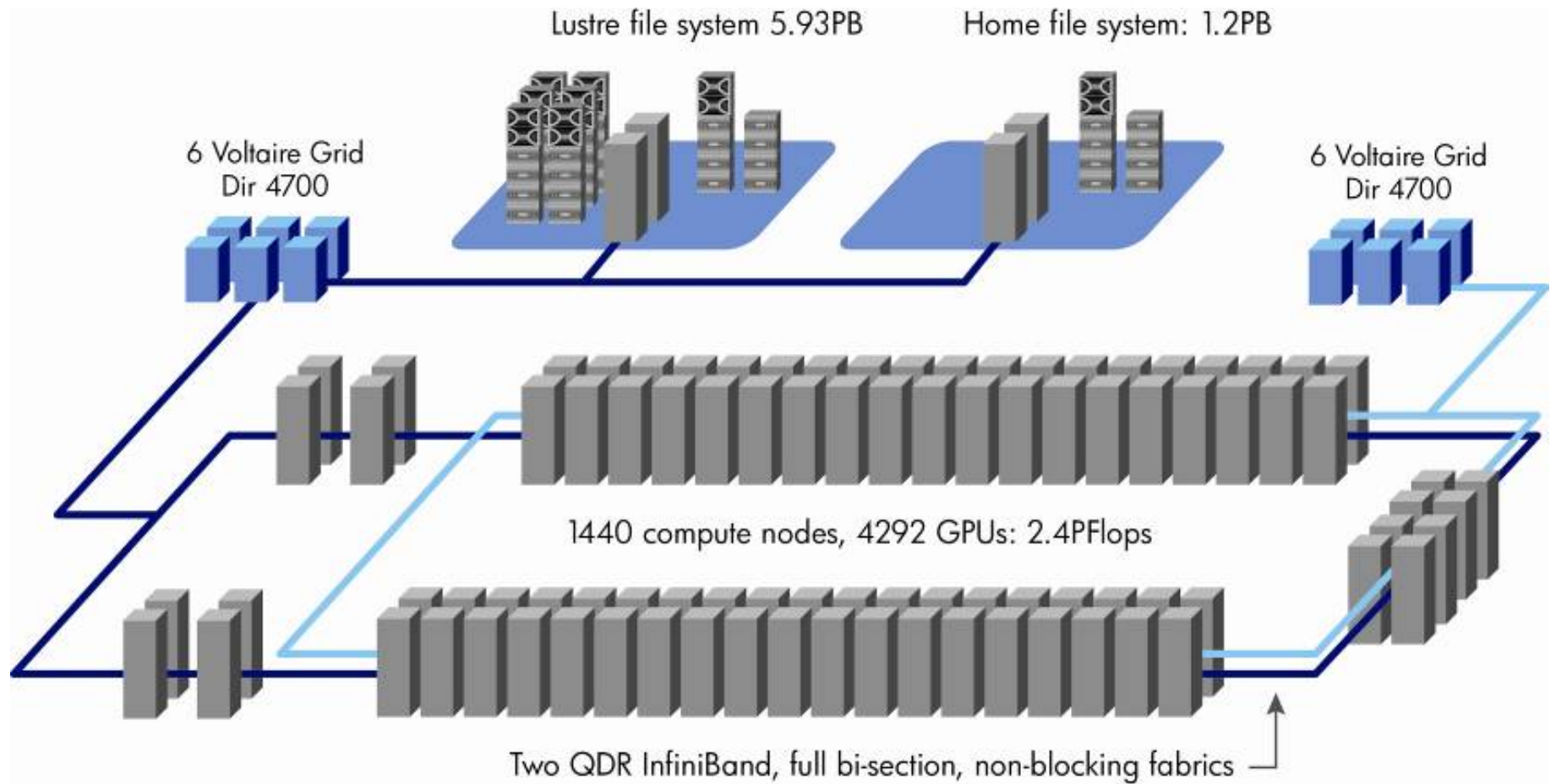
Polycarbonate front door reduces ambient noise considerably



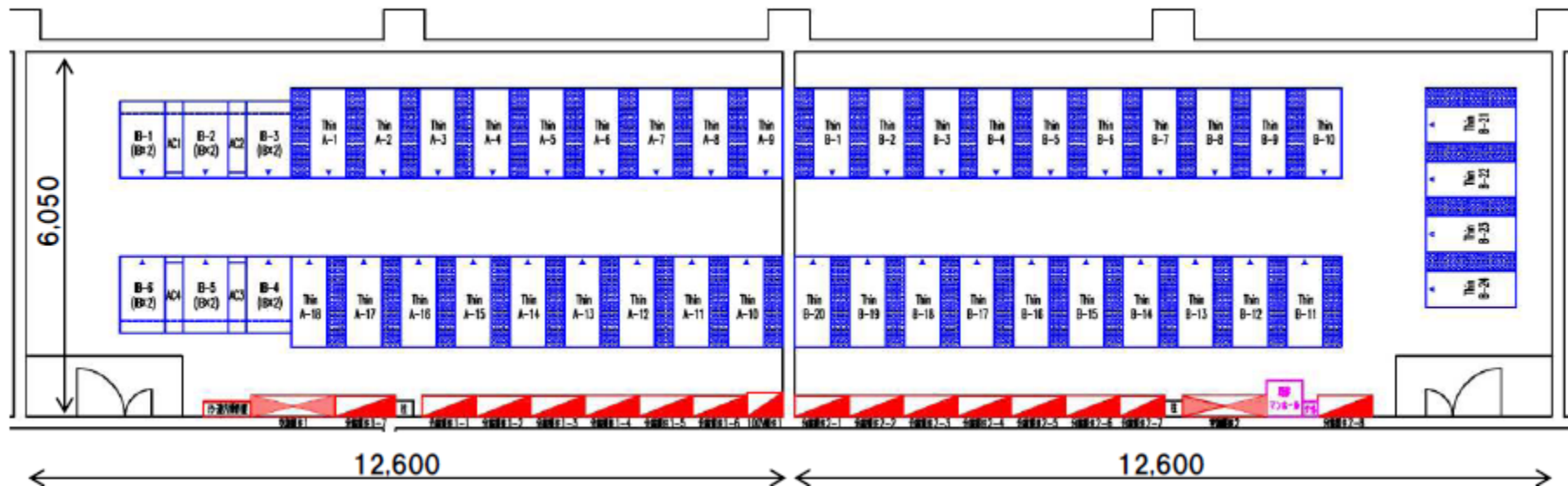
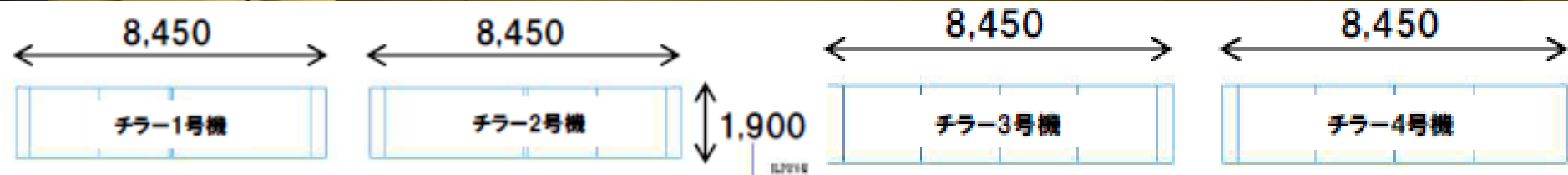
~= Entire Earth Simulator
(rack = 50TF)



Pulling It All Together



TSUBAME2.0 Layout (200m² for main compute nodes)







**2.4 Petaflops, 1408 nodes
~50 compute racks + 6 switch racks**

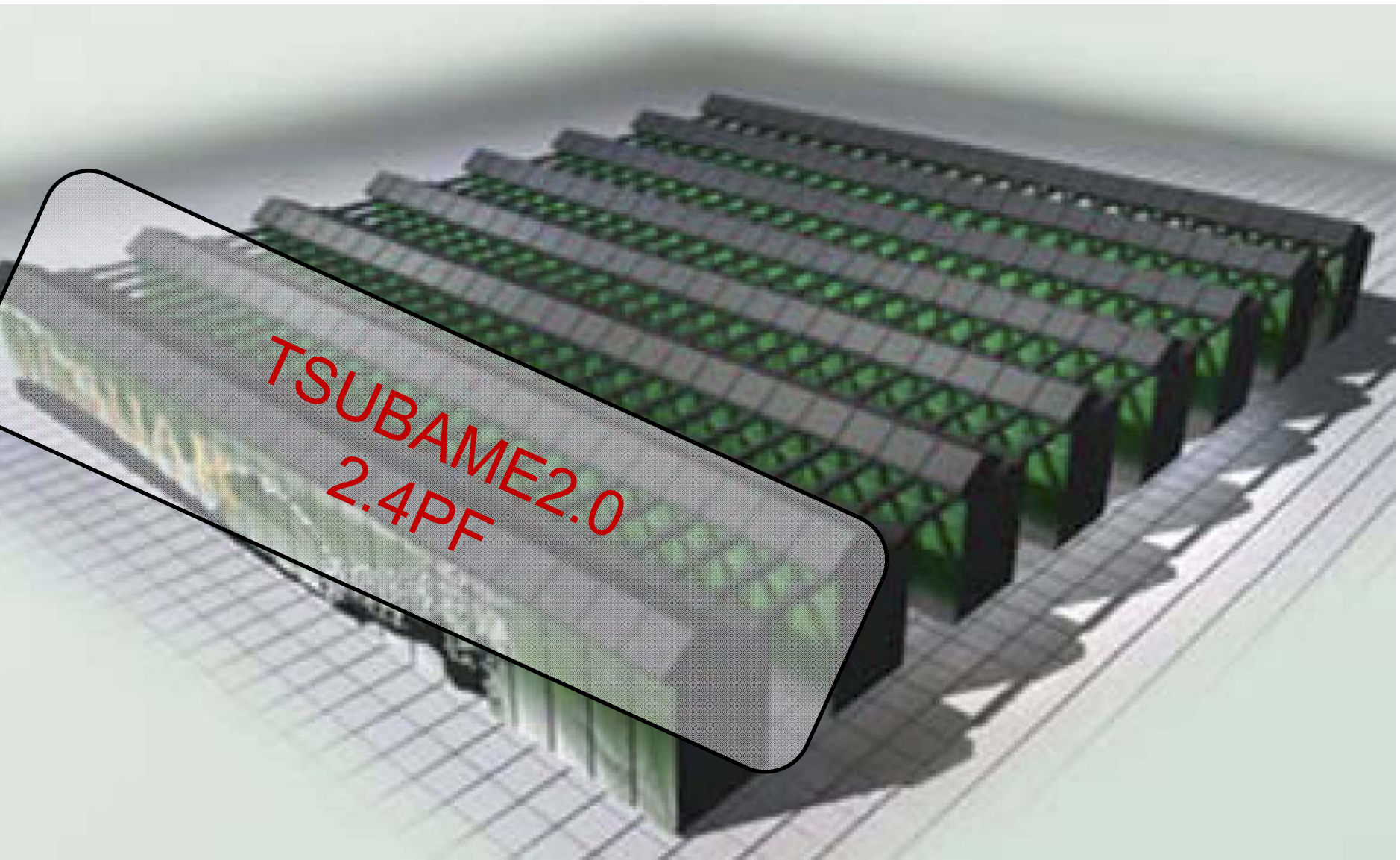
Two Rooms, Total 160m²

1.4MW (Max, Linpack), 0.48MW (Idle)

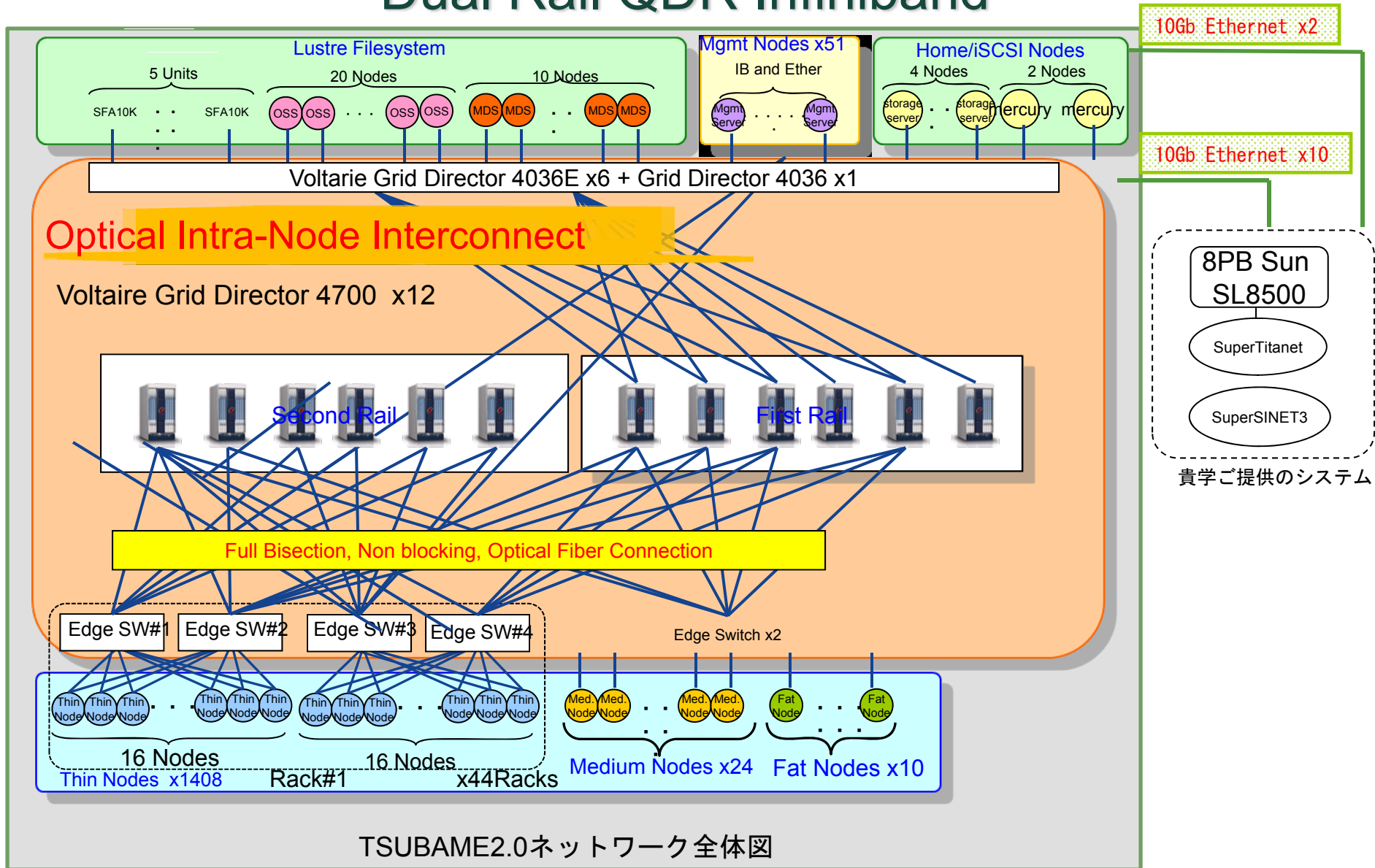


ORNL Jaguar and Tsubame 2.0

Similar Peak Performance, 1/4 the Size and Power



TSUBAME 2.0 Full Bisection Fat Tree, Optical, Dual Rail QDR Infiniband



TSUBAME2.0ネットワーク全体図



3500 Fiber Cables > 100Km



TSUBAME2.0 Storage

Multi-Petabyte storage consisting of Luster Parallel Filesystem Partition and NFS/CIFS/iSCSI Home Partition + Node SSD Acceleration

Lustre Parallel Filesystem Partition,

MDS : HP DL360 G6 **5.96PB**

- CPU: Intel Westmere-EP x2 socket (12 Cores)
- Memory : 51GB (=48GiB)
- IB HCA: IB 4X QDR PCI-e G2 x1 port

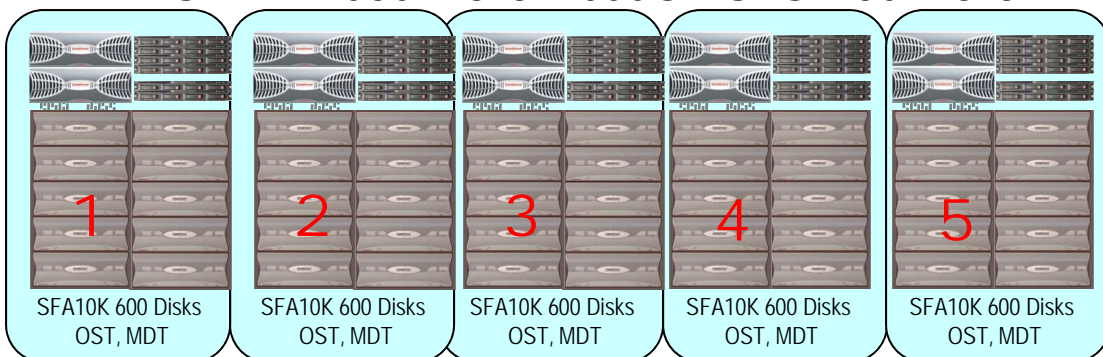
OSS : HP DL360 G6 x20

- CPU: Intel Westmere-EP x2 socket (12 Cores)
- Memory : 25GB (=24GiB)
- IB HCA: IB 4X QDR PCI-e G2 x2 port

Storage : DDN SFA10000 x5

- Total Capacity : 5.93PB

2TB SATA x 2950 Disks + 600GB SAS x 50 Disks



Home Partition 1.2PB

NFS/CIFS : HP DL380 G6 x4

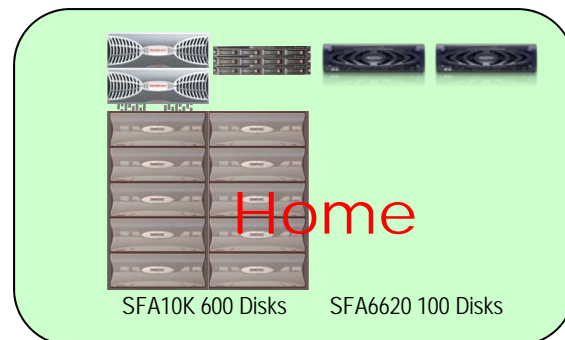
- CPU: Intel Westmere-EP x2 socket (12 Cores)
- Memory : 51GB (=48GiB)
- IB HCA: IB 4X QDR PCI-e G2 x2 port

NFS/CIFS/iSCSI : BlueArc Mercury100 x2

- 10GbE x2

ストレージ : DDN SFA10000 x1

- Total Capacity : 1.2PB
- 2TB SATA x 600 Disks



7.13PB HDD + 200TB SSD + 8PB Tape



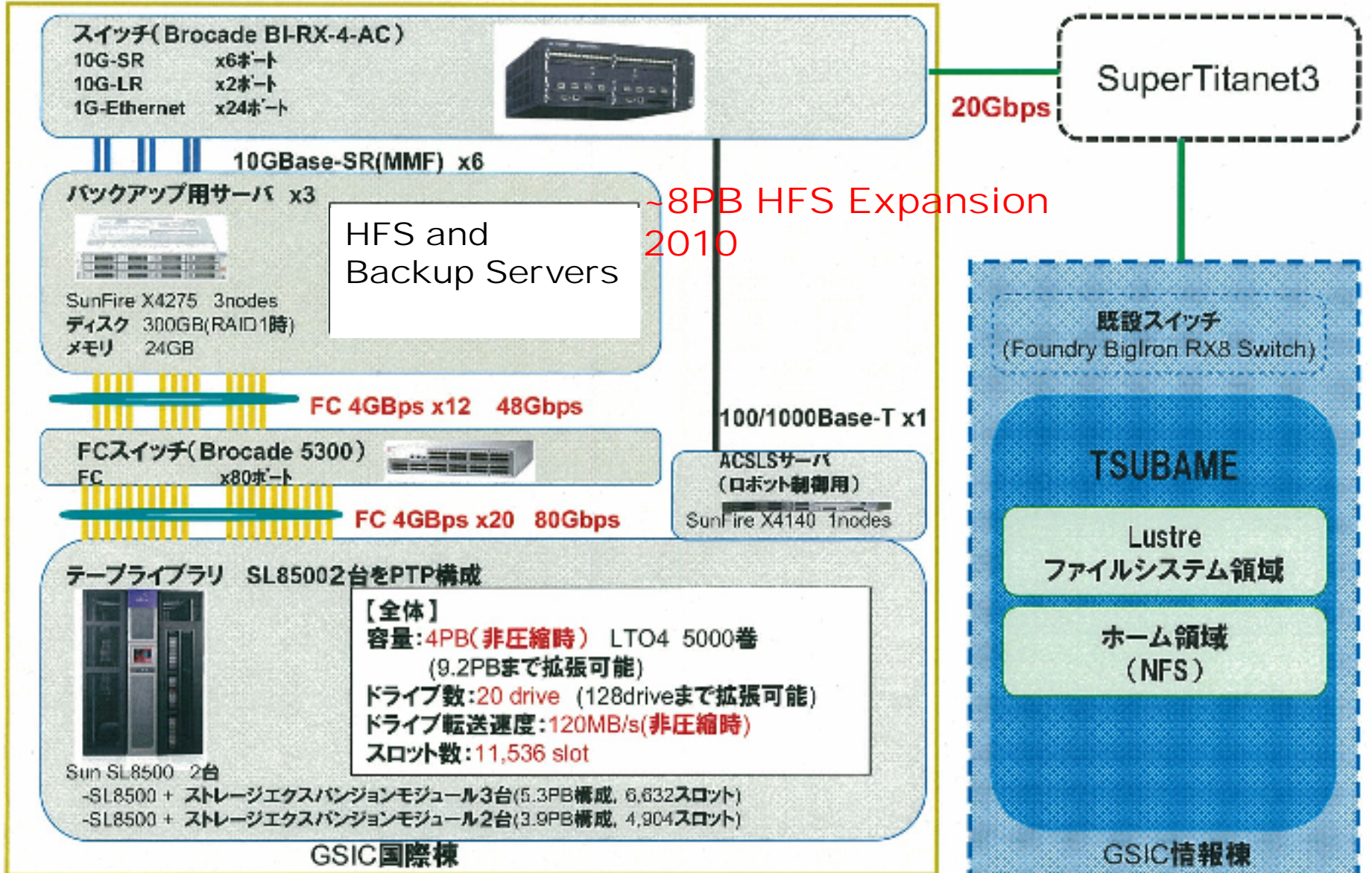
200 TB SSD (SLC, 1PB MLC future)
7.1 PB HDD (Highly redundant)
4-8 PB Tape (HFS+Backup)
All Infiniband+10GbE Connected



Lustre + GPFS
NFS, CIFS, WebDAV, ...
Tivoli HFS
GridScaler + BlueArc

TSUBAME2.0 Tape System

HFS of over15PB and Scalable



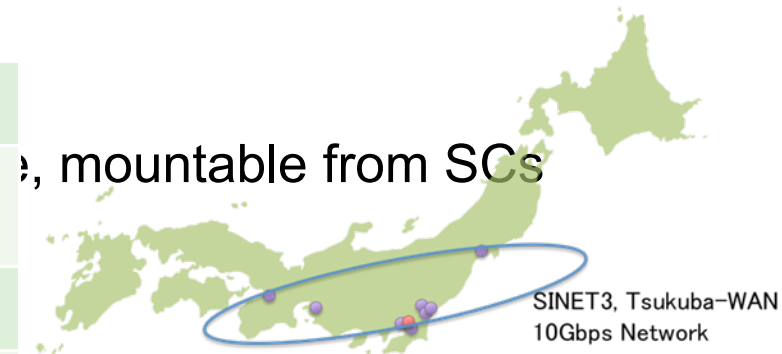
RENKEI-POP Distributed Storage

- Objective: SC Centers-Wide Distributed Storage on SINET National NW
 - ▶ RENKEI (MEXT e-Science) Project and R&D and nationwide deployment of RENKEI-PoPs (Point of Presence)
 - ▶ Data transfer server engine with large capacity and fast I/O
 - ▶ “RENKEI-Cloud” on SINET3 with various Grid/Cloud Services including Gfarm distributed filesystem



CPU	Core i7 975 Extreme (3.33 GHz)
Memory	12GB (DDR3 PC3-10600 , 2GB*6)
NIC	10GbE (without TCP/IP Offload Engine)
System Disk	500GB HDD
SSD RAID	30TB (RAID 5, 2TB HDD x 16)

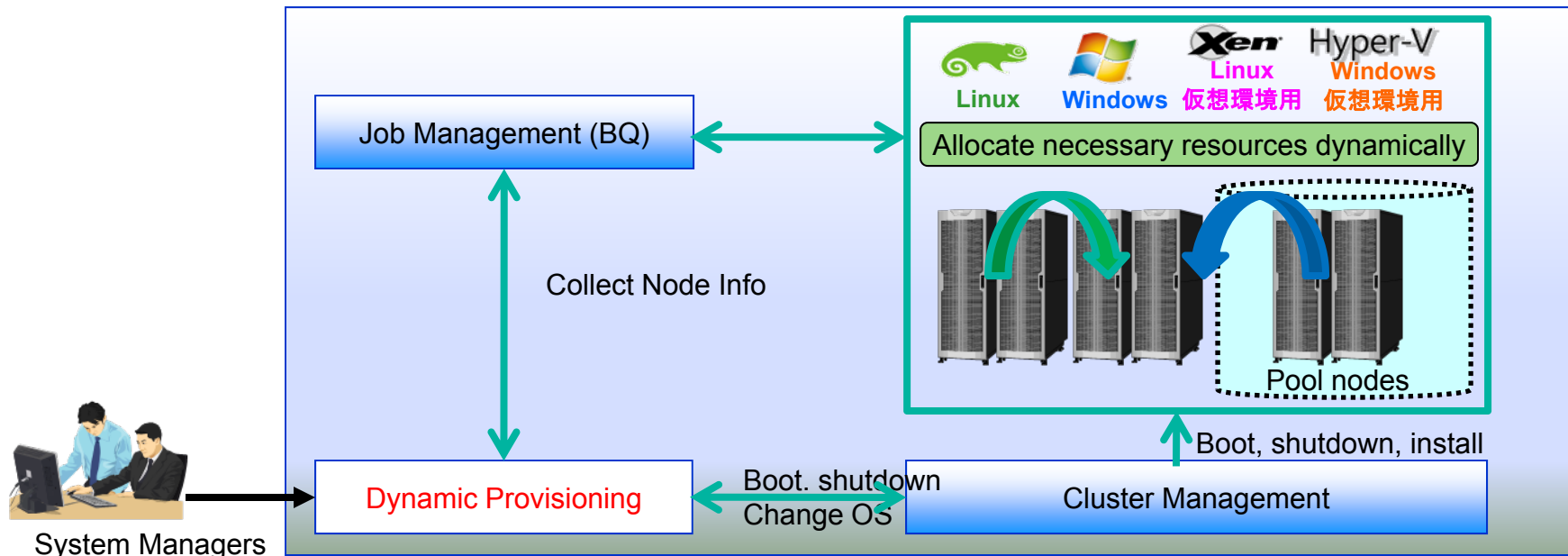
Tokyo Inst. Tech.	Osaka Univ.
National Inst. Inform.	KEK
Nagoya Univ.	Univ. Tsukuba
AIST	Tohoku Univ.



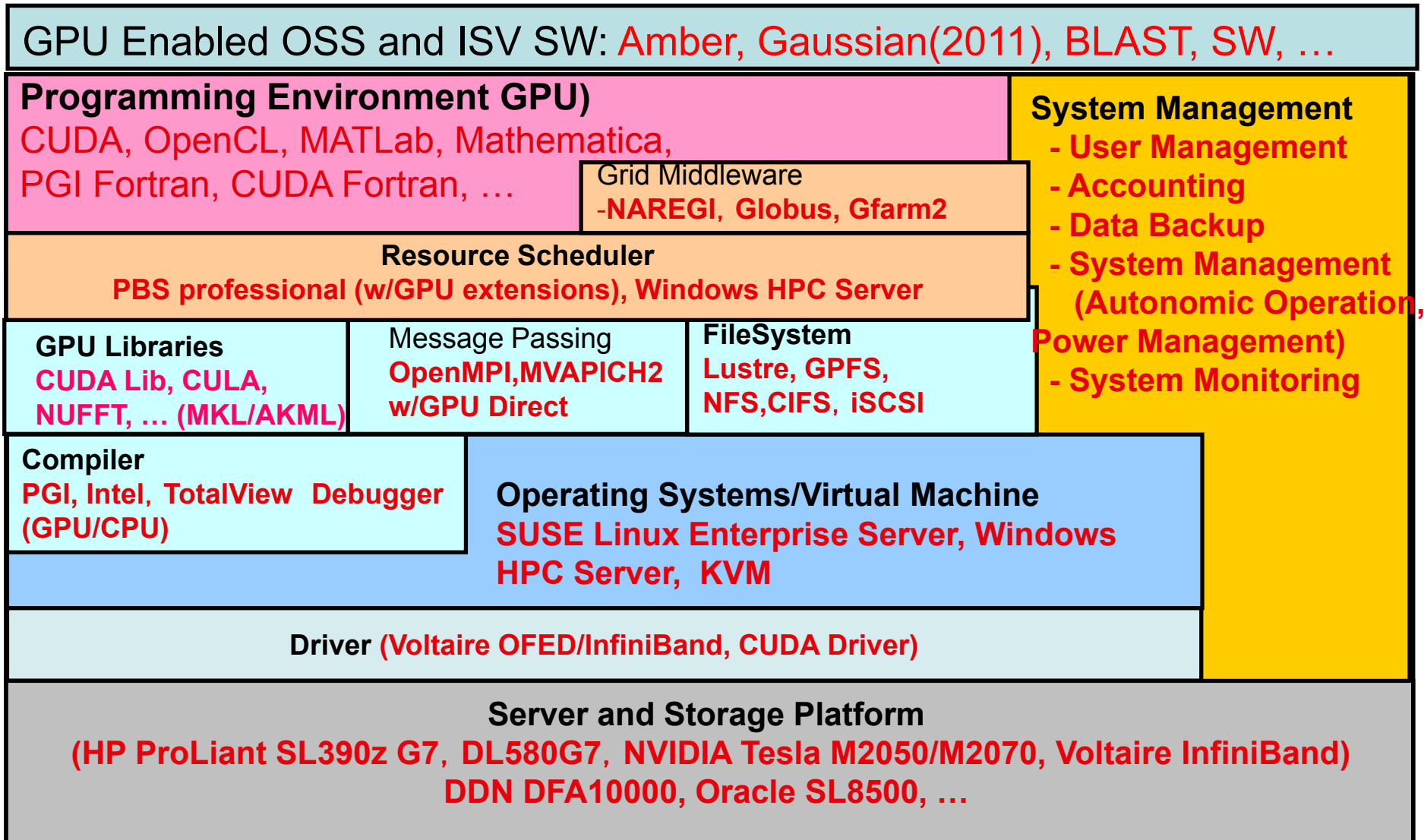
Cloud Provisioning Supercomputing Nodes

Tsubame 2.0 will have a dynamic provisioning system that works with batch queue systems and cluster management systems to dynamically provision compute resources

- ▶ Pool nodes can be dynamically allocated to various supercomputing services
- ▶ Linux and Windows batch queue systems coordinate to manage the nodes
- ▶ Dynamic increase/decrease of nodes allocated to particular services
 - ※ Increase / Decrease of Windows HPC and Linux nodes of different configurations
- ▶ Virtual Nodes on VMs on respective OSes can be subject to dynamically allocated and scheduled.



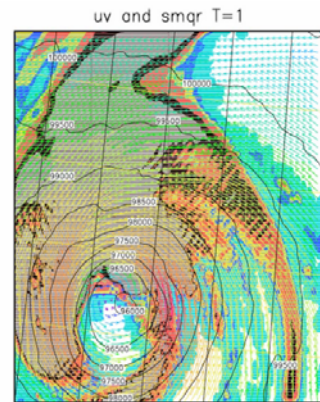
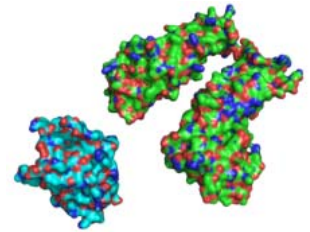
TSUBAME2.0 Software Stack



TSUBAME2.0 Estimated Performances

- 1.192 TFlops Linpack [IEEE IPDPS 2010]
 - Top ranks Green 500?
- ~0.5 PF 3D Protein Rigid Docking (Node 3-D FFT) [SC08, SC09]
- 145Tlops ASUCA Weather Forecast [SC10 Best Student Paper Finalist]
- Multiscale Simulation of Cardiovascular flows [SC10 Gordon Bell Finalist]
- Various FEM, Genomics, MD/MO (w/IMS), CS-Apps: search, optimization,

THE GREEN
500™



TSUBAME2.0 World Rankings (Nov. 2010 Announcement Green500!!!)

The Top 500 (Absolute Performance)

- #1: ~2.5 PetaFlops: China Defense Univ. Dawning Tianhe 1-A
- #2: 1.76 Petaflops: US ORNL Cray XT5 Jaguar
- #3: 1.27 PetaFlops: China Shenzen SC Nebulae
- #4: 1.19 PetaFlops: Japan Tokyo Tech. HP/NEC TSUBAME2.0**
- #5: 1.054 PetaFlops: US LLBL Cray XE6 Hopper
- #~33 (#2 Japan): 0.191 Petaflops: JAEA Fujitsu



Top500 BoF
Tue 16th
17:30~

The Green 500 (Performance/Power Efficiency)

- #1: 1684.20 : US IBM Research BG/Q Prototype (116)
- #2: 958.35: Japan Tokyo Tech/HP/NEC Tsubame 2.0 (4)**
- #3: 933.06 : US NCSA Hybrid Cluster Prototype (403)
- #4: 828.67: Japan Riken “K” Supercomputer Prototype (170)
- #5-7: 773.38: Germany Julich etc.IBM QPACE SFB TR (207-209)
- (#2+ 1448.03: Japan NAO Grape-DR Prototype) (383) (Added in Dec.)



Green 500
BoF
Thu 18th
12:15~

“Little Green 500” collaboration w/Microsoft
Achieved 1.037 Gigaflops/W in power efficient experimental config.

THE **GREEN**
500TM

sponsored by

SUPERMICRO[®]

This certificate is in recognition of your organization's achievements in reducing the environmental impact of high-performance computing.

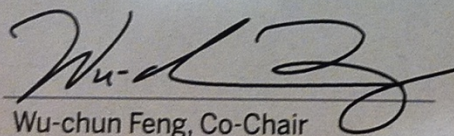
GSIC Center, Tokyo Institute of Technology

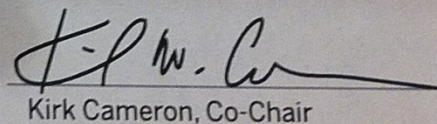
Is recognized as the

Greenest Production Supercomputer in the World

on the world's Green500 List of computer systems as of

November 2010


Wu-chun Feng, Co-Chair


Kirk Cameron, Co-Chair

Petaflops? Gigaflops/W?



x66,000
faster
x3 power
efficient

<<

x44,000 Data



Laptop: SONY Vaio type Z (VPCZ1)
CPU: Intel Core i7 620M (2.66GHz)
MEMORY: DDR3-1066 4GBx2
OS: Microsoft Windows 7 Ultimate 64bit
HPL: Intel(R) Optimized LINPACK Benchmark for
Windows (10.2.6.015)
256GB HDD

18.1 GigaFlops Linpack
369 MegaFlops/W

Supercomputer: TSUBAME 2.0
CPU: 2714 Intel Westmere 2.93 Ghz
GPU: 4071 nVidia Fermi M2050
MEMORY: DDR3-1333 80TB + GDDR5 12TB
OS: SuSE Linux 11 + Windows HPC Server R2
HPL: Tokyo Tech Heterogeneous HPL
11PB Hierarchical Storage

1.192 PetaFlops Linpack
1043 MegaFlops/W

TSUBAME2.0 (2010) vs. Earth Simulator1 (ES) (2002) vs. Japanese 10PF NexGen "K" @Kobe (2012)



200m²

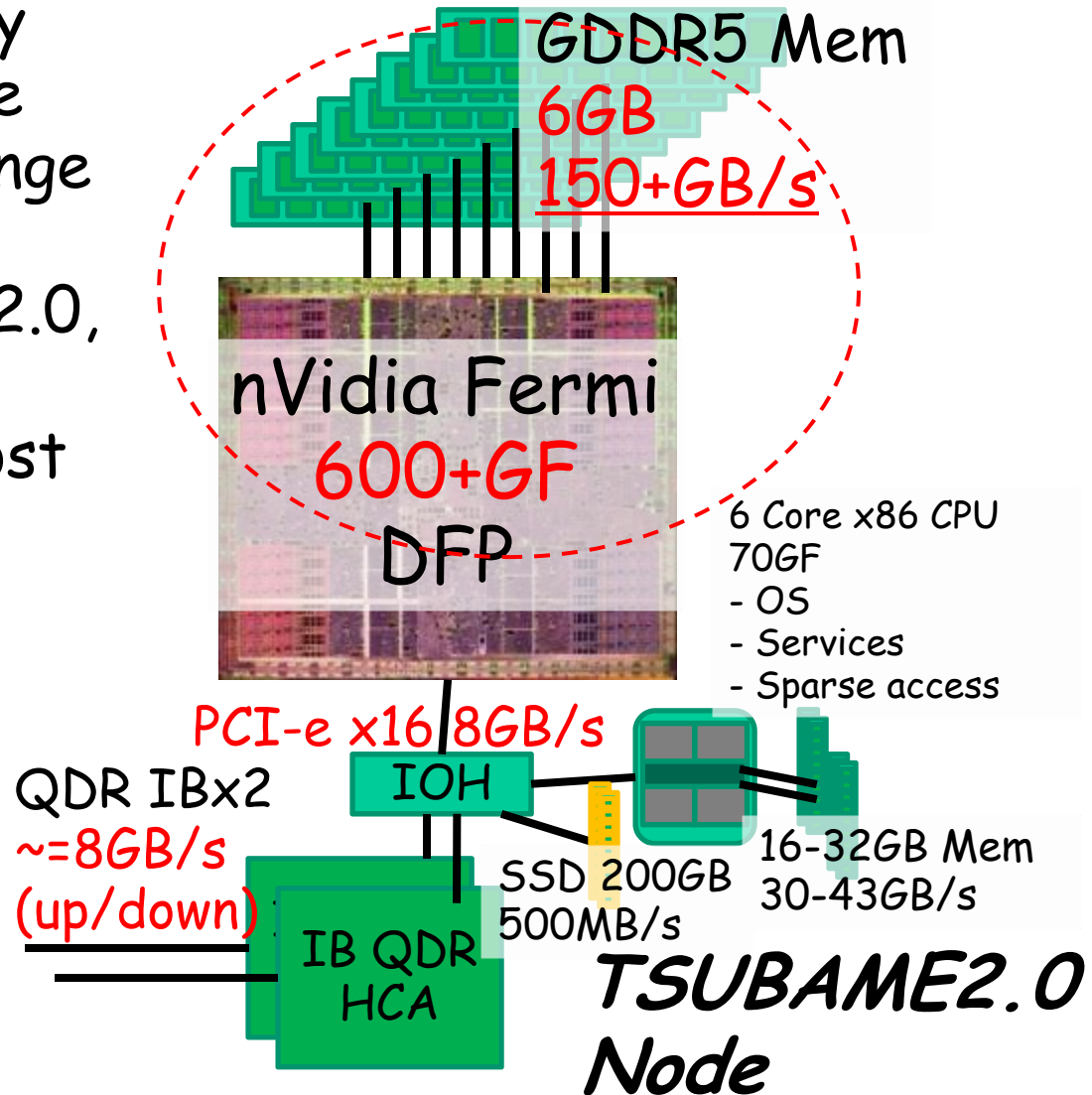


The "IDEAL TSUBAME2.0"

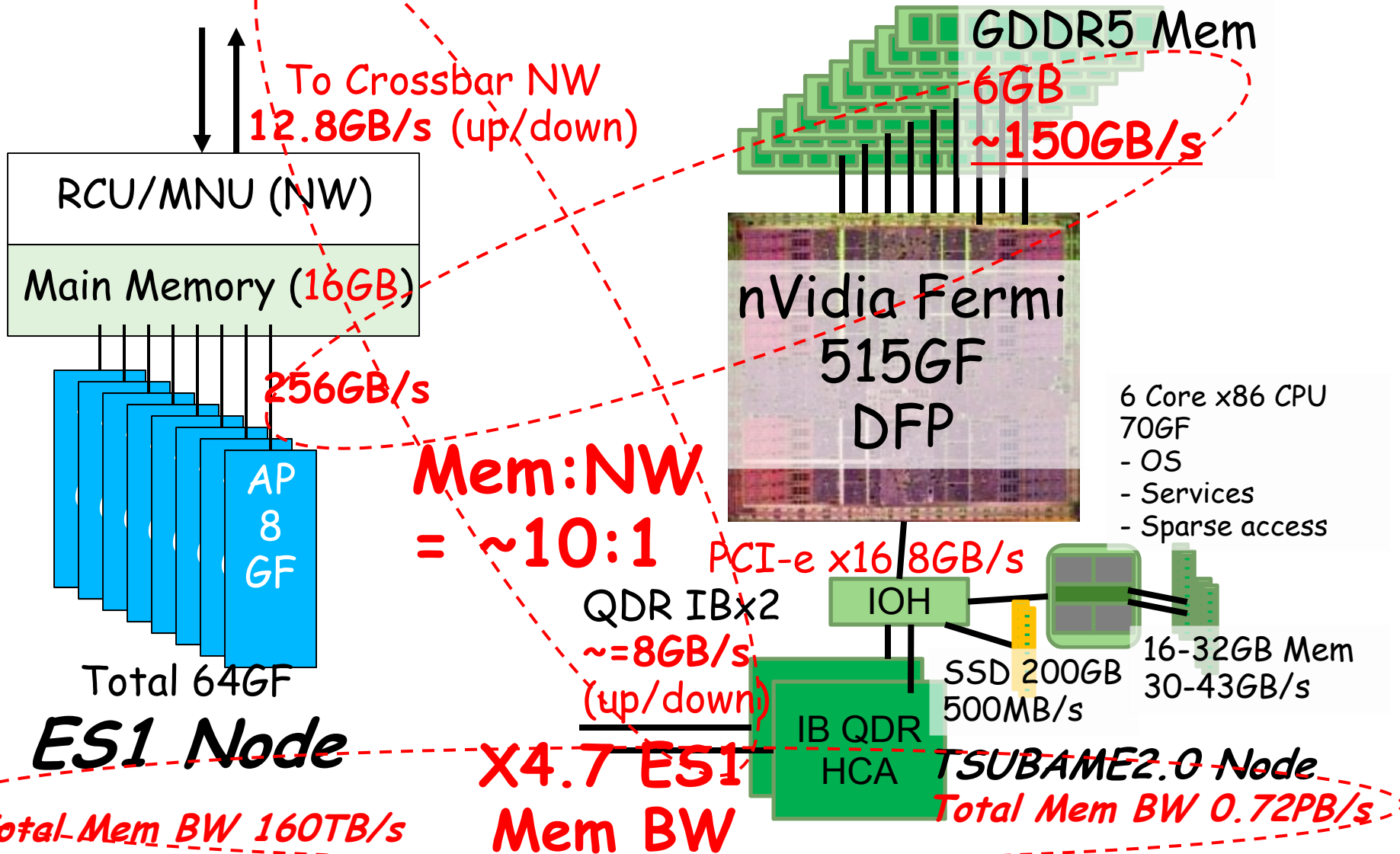
- What are architecturally possible without excessive design, power, or SW change

- In the REAL TSUBAME2.0, will have to compromise various parameters for cost and other reasons

- Almost Equal to "High Bandwidth" TSUBAME2/SL390 Config



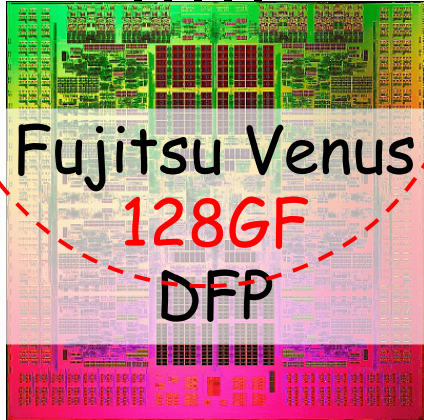
The "IDEAL TSUBAME2.0" (w/o cost constraints) node vs. ES1 node



The "IDEAL TSUBAME2.0" node vs. 10PF NLP Node (2012)

DDR3-1066 Mem

16GB 64GB/s



Bytes/Flop
= 0.3~0.5

GDDR5 Mem

6GB
~150+GB/s

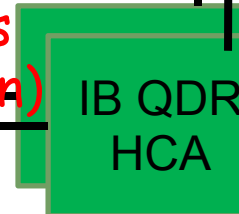


- 6 Core x86 CPU
- 70GF
- OS
- Services
- Sparse access

6-D Torus
5GB/s / link
up to 4
simultaneous
transfers

PCI-e x16 8GB/s

QDR IBx2
~8GB/s
(up/down)

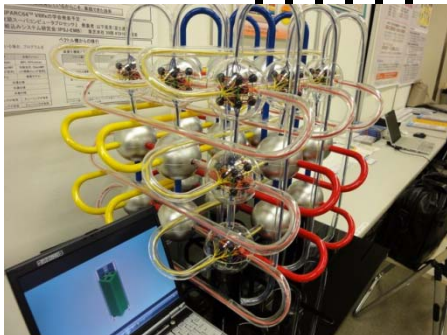


SSD 200GB
500MB/s

16-32GB Mem
30-43GB/s

**TSUBAME2.0
Node**

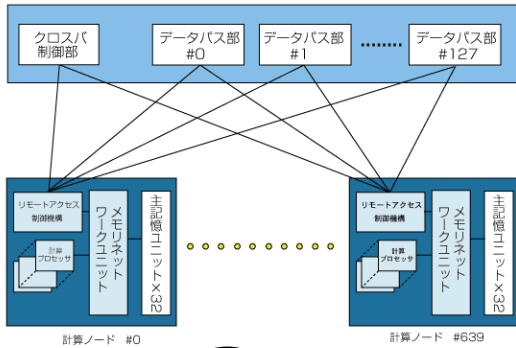
NLP Node



Comparing the Networks

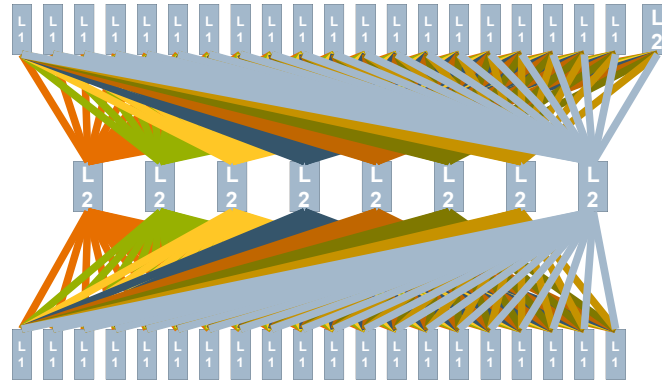


結合ネットワーク(IN)部



ES1

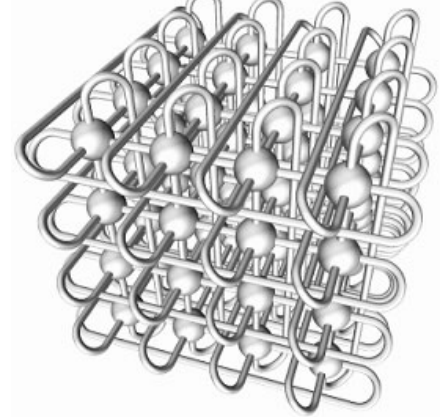
12.8GB/s Link
5us latency
Full Crossbar
~8TB/s
Bisection BW



Ideal

TSUBAME2.0

(4+4)GB/s Link
2us latency
Full Bisection Fat Tree
~60TB/s Bisection BW



10PF NLP

5GB/s Link
?us latency
6-D Torus
~30TB/s?
Bisection BW

Summary of Comparisons

- (1) ES1 vs. Ideal TSUBAME2.0
 - ▶ Similar (Mem BW : Network BW), full bisection NW
 - ▶ $ES1 \sum BW : TSUBAME2 \sum BW = 1 : 6$
 - ⇒ **BW-bound apps (e.g. CFD) should scale equally on both w.r.t. $\sum BW$ (TSUBAME2.0 6 times faster), Other apps *drastically* faster on TSUBAME2.0**
- (2) 10PF NLP vs. Ideal TSUBAME2.0
 - ▶ Similar Memory Bytes/Flop (0.3~0.5)
 - ▶ NLP x2 superior on Mem BW : Network BW
 - ▶ TSUBAME2.0 x2 better on Bisection BW?
 - ⇒ **Most apps similar efficiency and (strong) scalability
NLP ~4 times faster on full machine (weak scaling)**

(Faster Than) Real-time

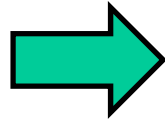
Tsunami Simulation

(Prof. Takayuki Aoki, Tokyo Tech.)

ADPC : Asian Disaster Preparedness Center

Early Warning System:

Data Based
Extrapolation



high accuracy

(Faster than)
Real-time CFD

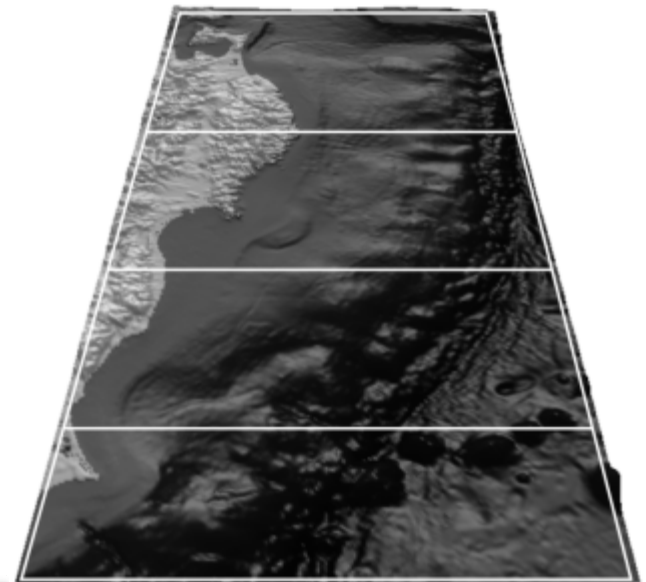
Shallow-Water Equation

Conservative Form:

Assuming hydrostatic
balance in the vertical
direction,

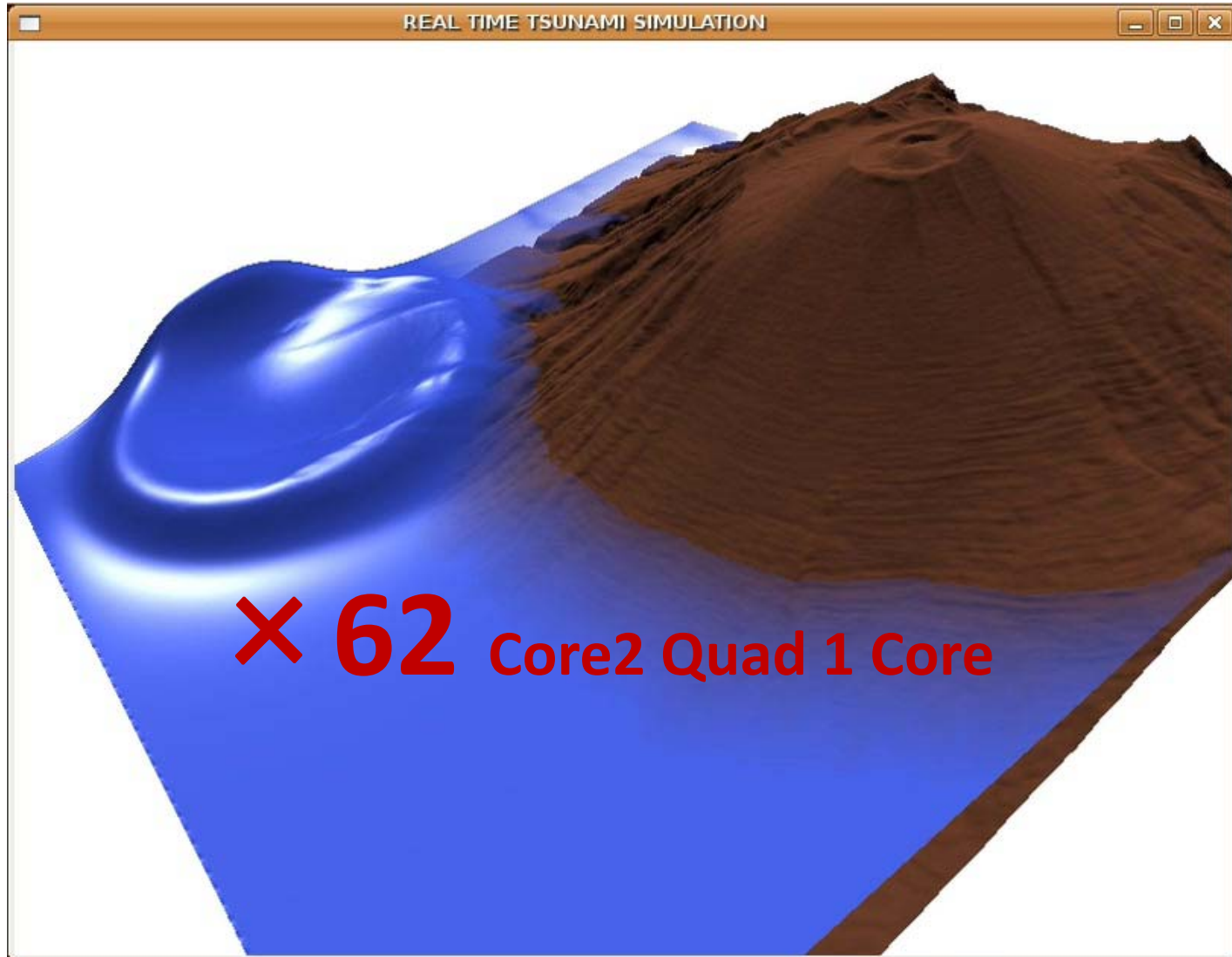


3D → 2D equation



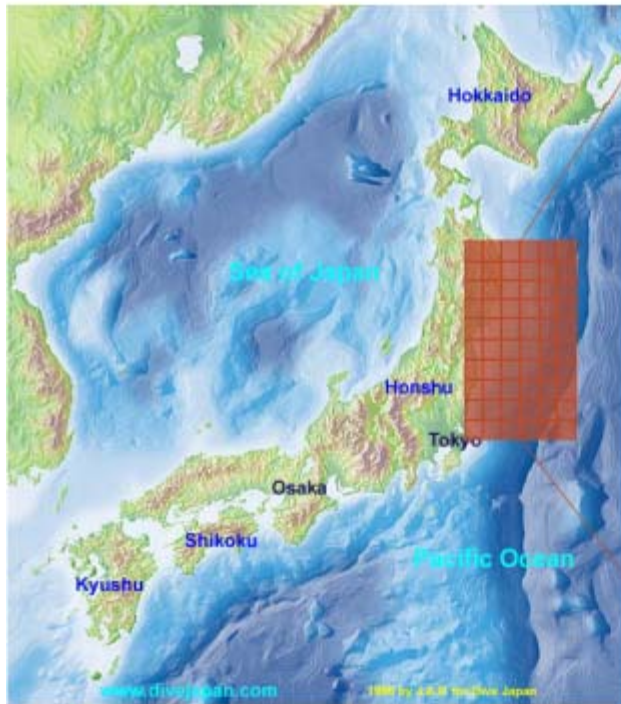
8 GPU 400km × 800km
(100m mesh)

SCREEN Capture

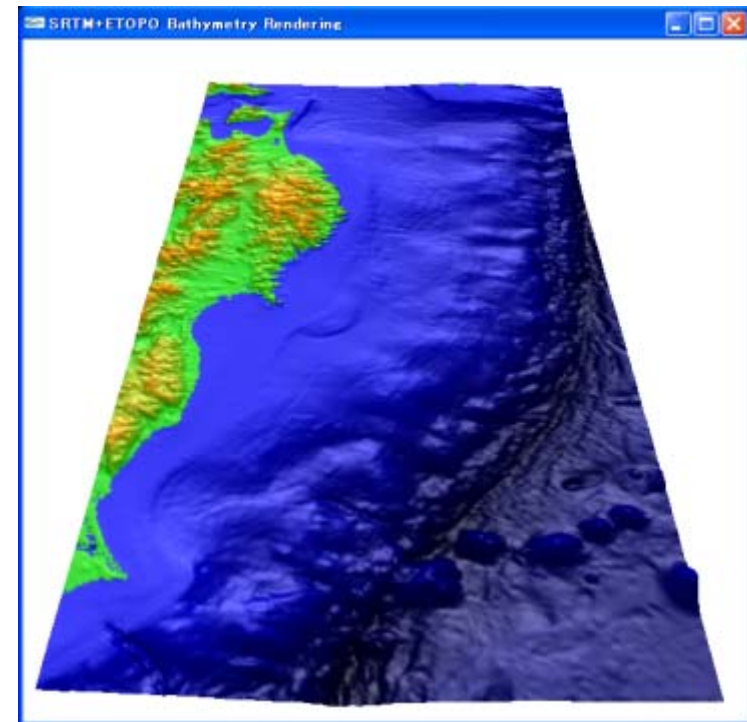


Tsunami Prediction of Northern Japan Pacific Coast

- Bathymetry



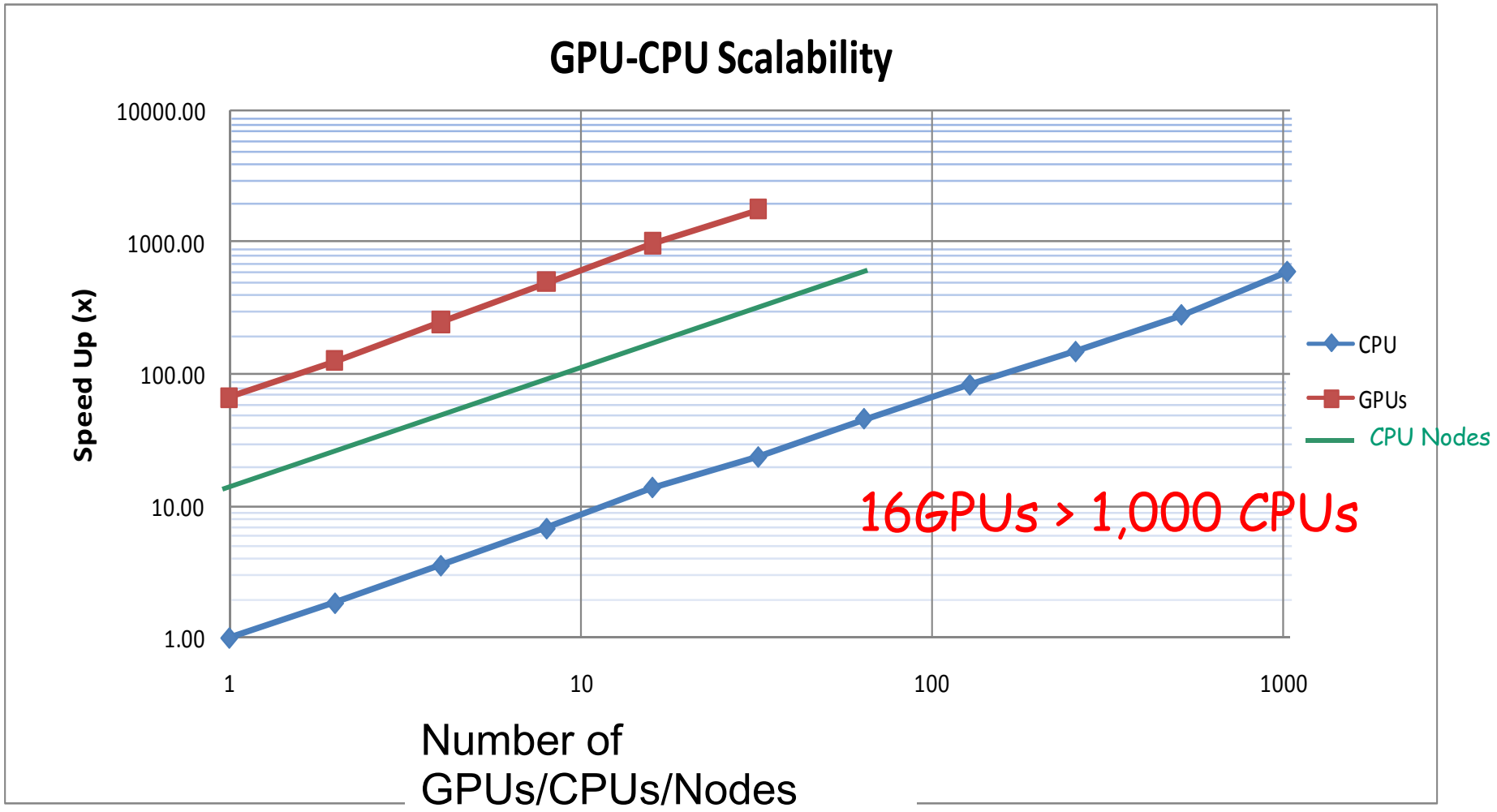
- ▣ Grid Size
4096x8192
- ▣ Latitude
 $N35^{\circ} - N42^{\circ}$
- ▣ Longitude
 $E140^{\circ}30' - E144^{\circ}18'$
- ▣ Length
370km x 740km



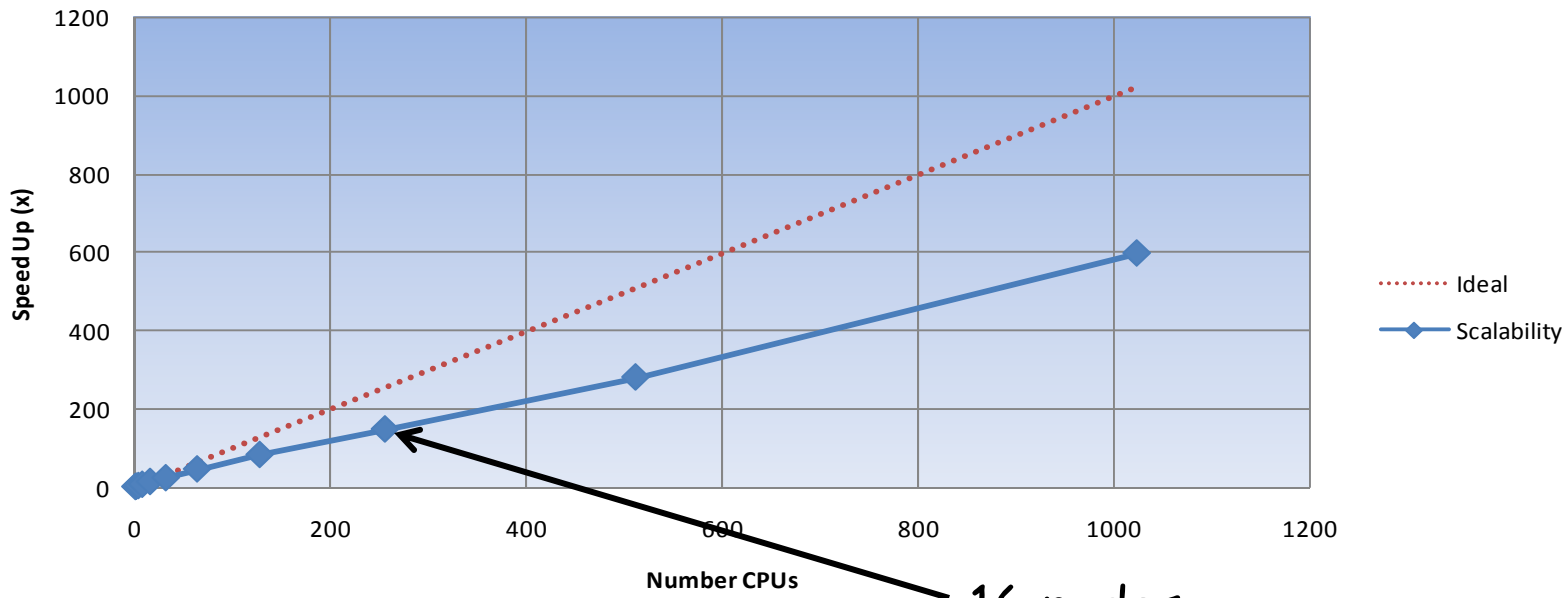
Multi-node CPU/GPU Comparison

***** Strong Scaling *****

- Results on TSUBAME1.2

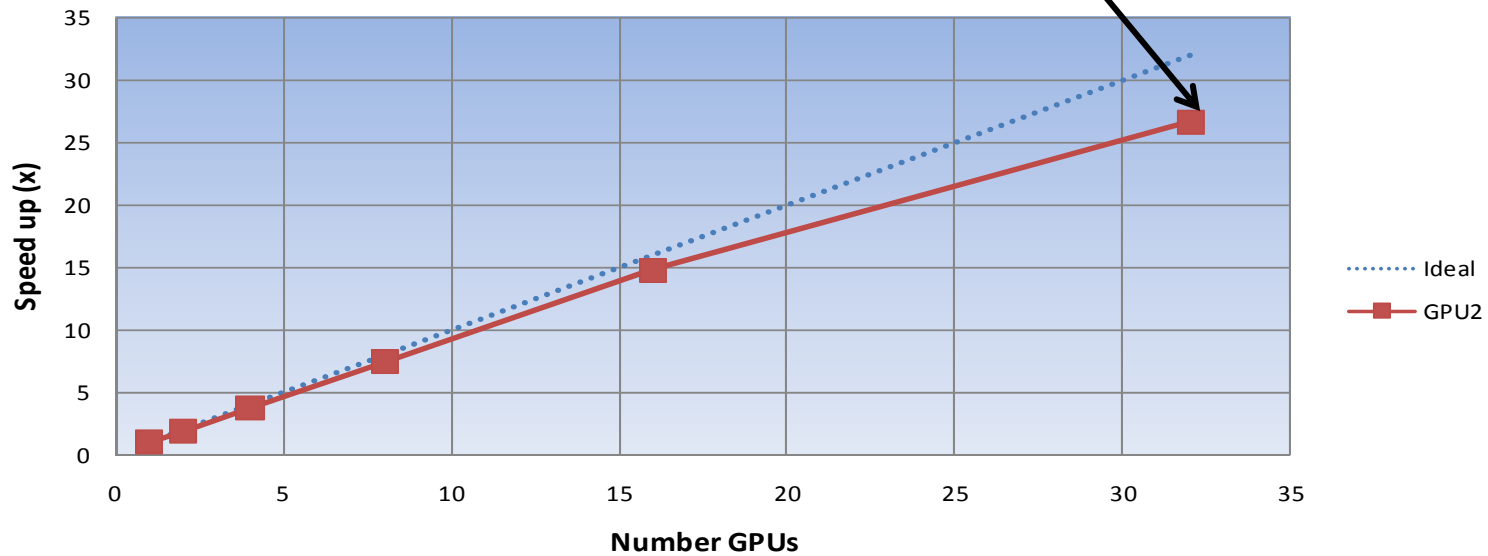


CPU Scalability



16 nodes

GPU Scalability



Next Generation Numerical Weather Prediction[SC10]

Collaboration: Japan Meteorological Agency

Meso-scale Atmosphere Model:
Cloud Resolving Non-hydrostatic model
 [Shimokawabe et. al. SC10 BSP Finalist]



Typhoon ~ 1000km
 1~ 10km
 Tornado,
 Down burst,
 Heavy Rain

ex. **WRF**(Weather Research and Forecast)

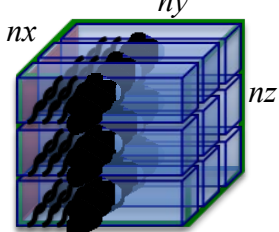
WSM5 (WRF Single Moment 5-tracer) Microphysics*
 Represents condensation, precipitation and thermodynamic effects of latent heat release
 1 % of lines of code, 25 % of elapsed time
 ⇒ 20 x boost in microphysics (1.2 - 1.3 x overall improvement)

ASUCA : full GPU Implementation

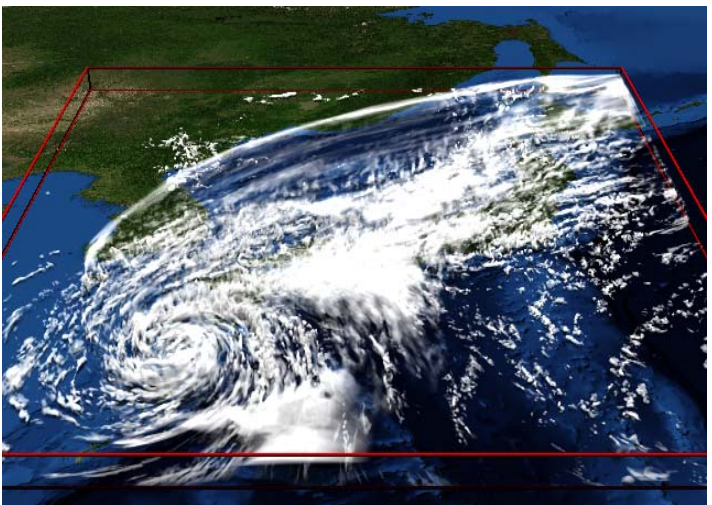
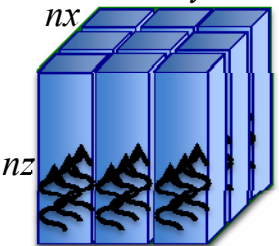
developed by Japan Meteorological Agency

TSUBAME 2.0 : 145 Tflops
World Record !!!

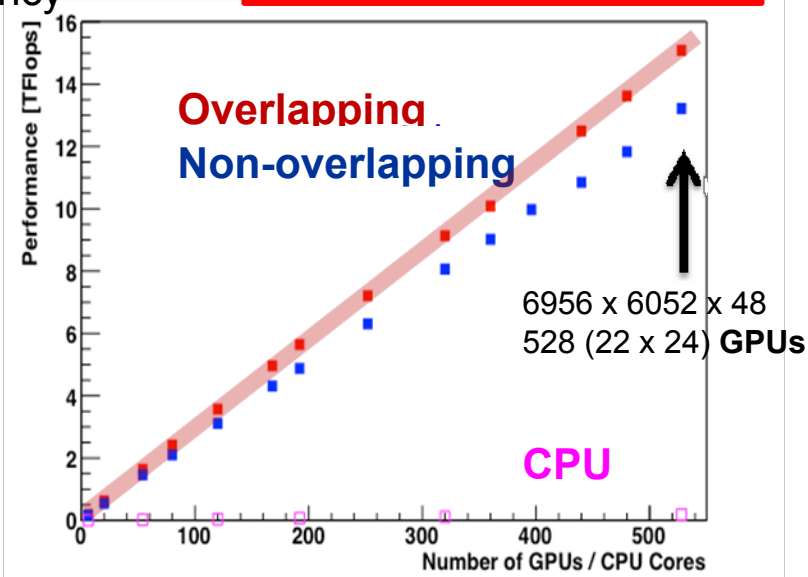
Block Division for Advection



for 1-D Helmholtz eq. ny



Typhoon



WRF GPU Computing

■ WRF (Weather Research and Forecast)

Open Source Community Code (NCAR, NCEP, OU, NOAA/FSL, AFWA)

WSM5 (WRF Single Moment 5-tracer) Microphysics*

Represents condensation, precipitation and thermodynamic effects of latent heat release

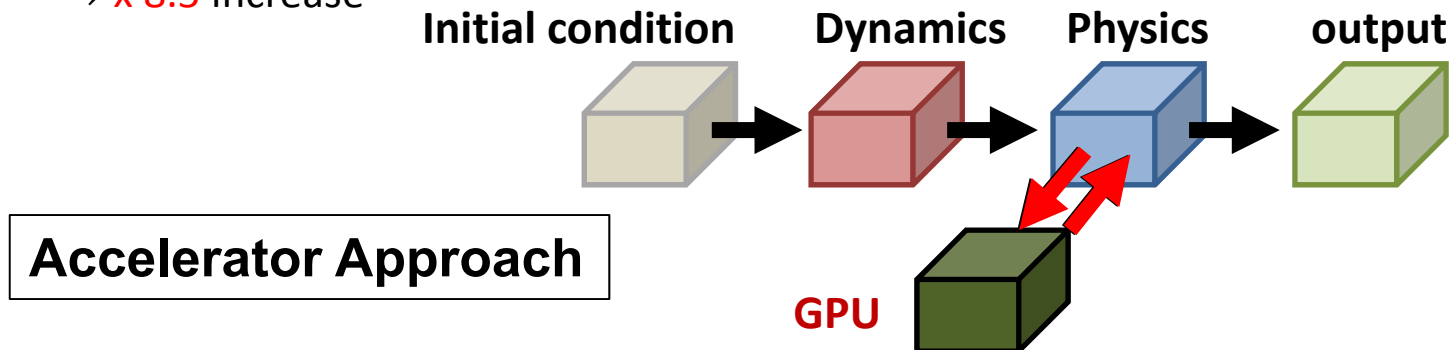
1 % of lines of code, 25 % of elapsed time

⇒ 20 x boost in microphysics (1.2 - 1.3 x overall improvement)

WRF-Chem**

provides the capability to simulate chemistry and aerosols from cloud scales to regional

⇒ x 8.5 increase



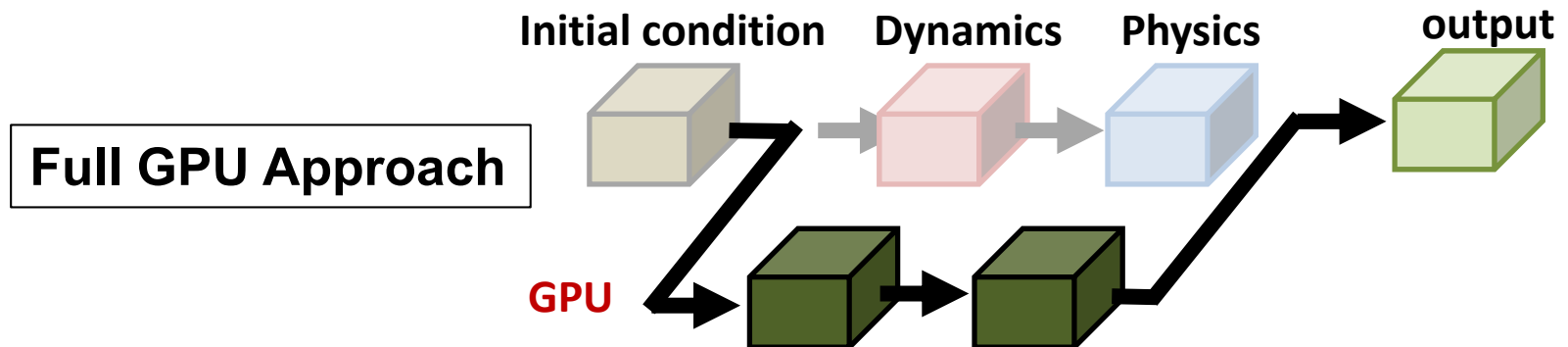
ASUCA: Next Gen Weather Simulation Code

■ **ASUCA Production Code**

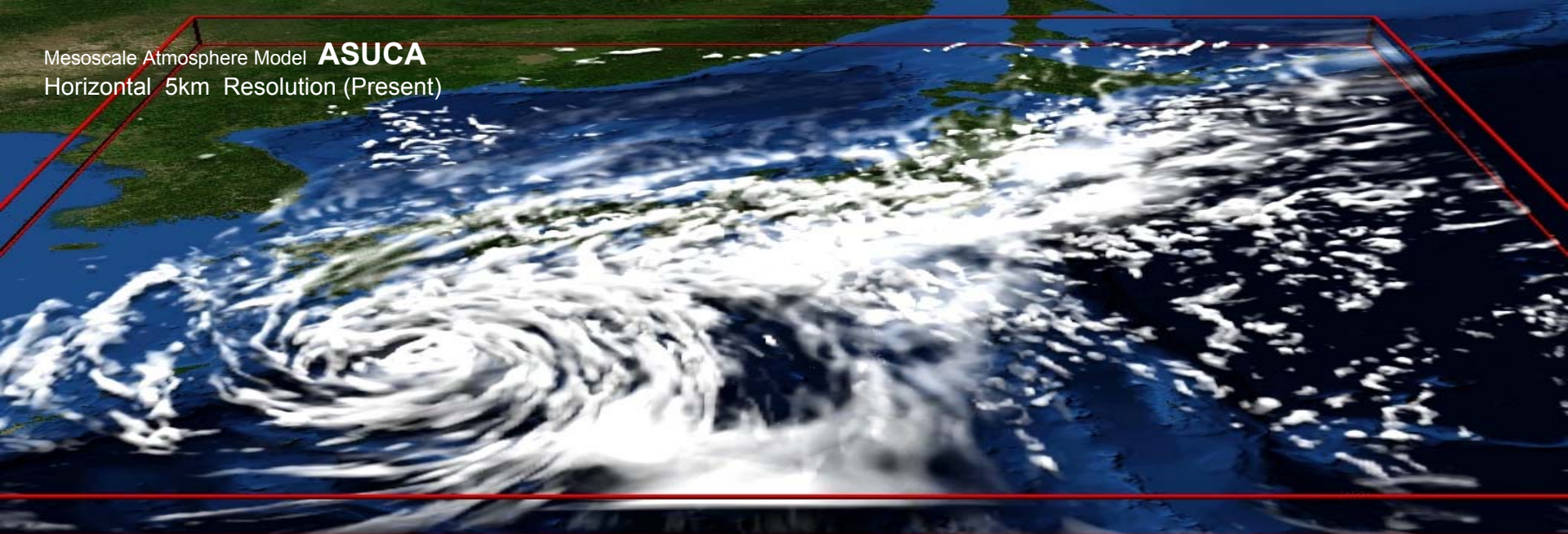
- ✓ *A next-generation high resolution weather simulation code that is being developed by Japan Meteorological Agency (JMA)*
- ✓ *ASUCA succeeds the JMA-NHM as an operational non-hydrostatic regional model at JMA*

■ **Similar Structure as WRF**

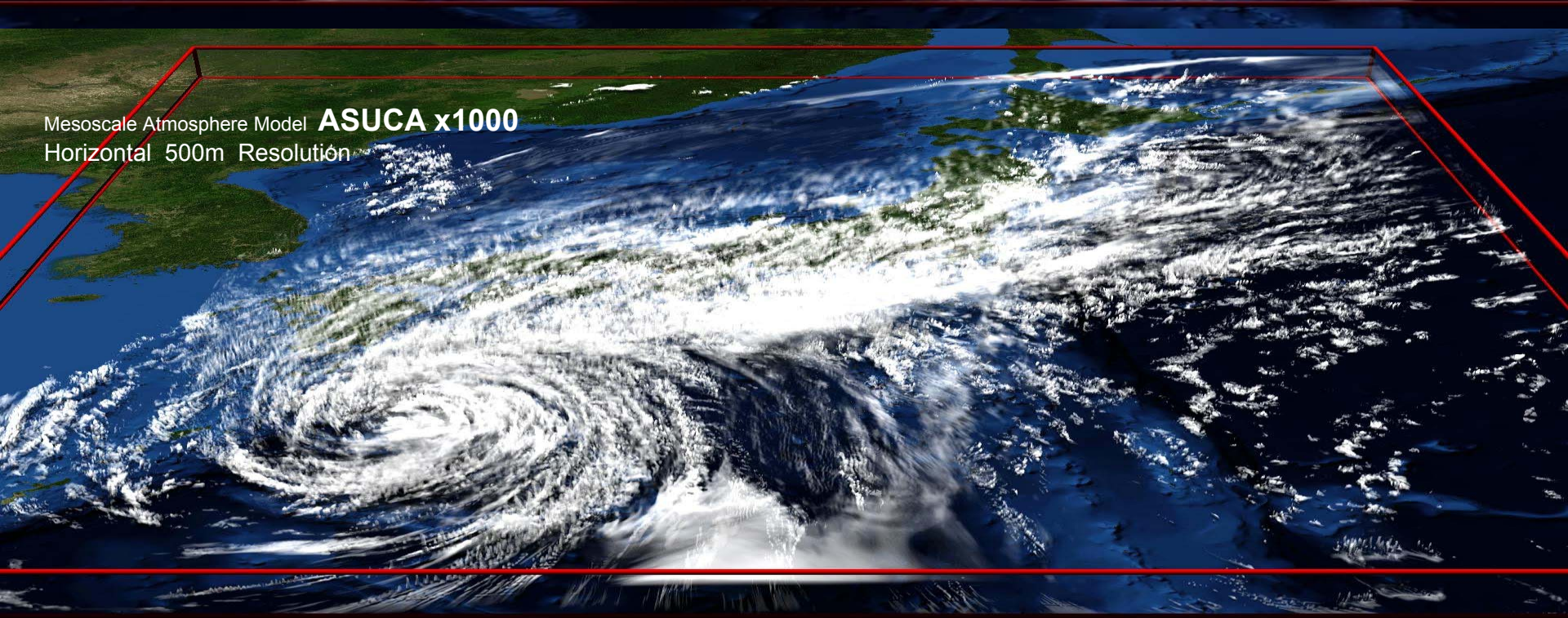
- ✓ *HEVI (Horizontally explicit Vertical implicit) scheme*
- ✓ *Dynamical Core uses a numerical scheme with 3rd-order accuracy in time and space*
Flux-form non-hydrostatic compressible equation
Generalized coordinate



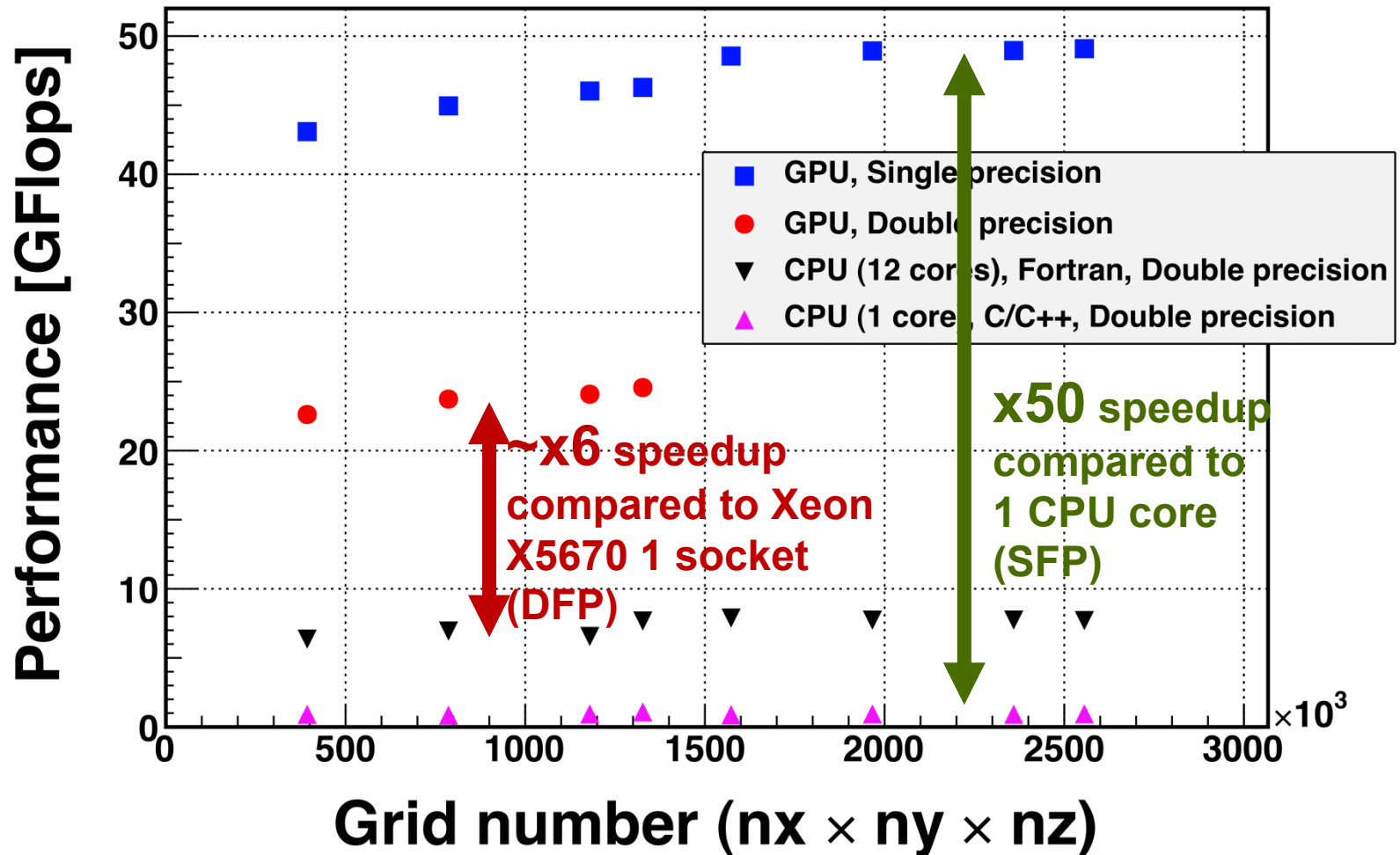
Mesoscale Atmosphere Model **ASUCA**
Horizontal 5km Resolution (Present)



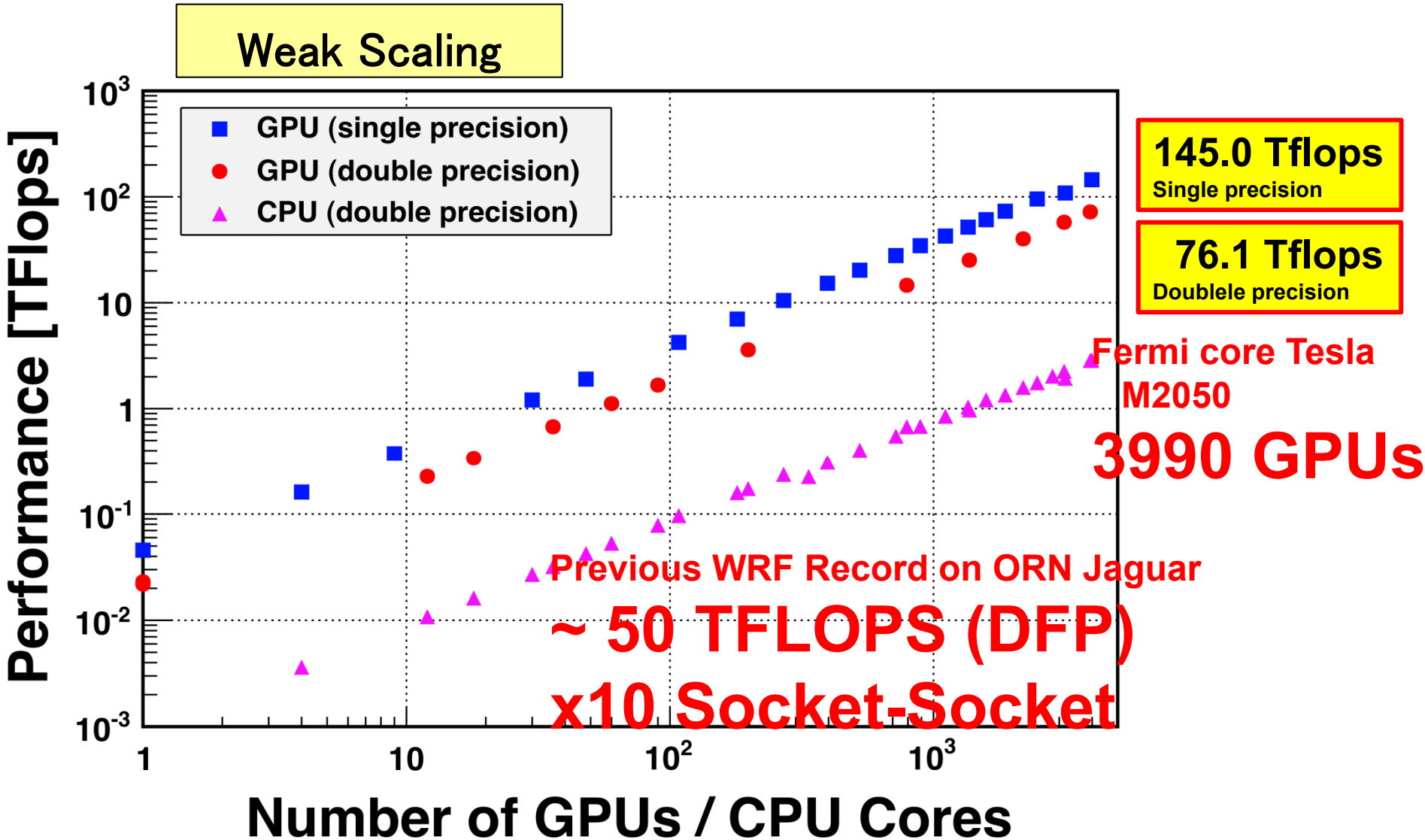
Mesoscale Atmosphere Model **ASUCA x1000**
Horizontal 500m Resolution



TSUBAME 2.0 (1 GPU)



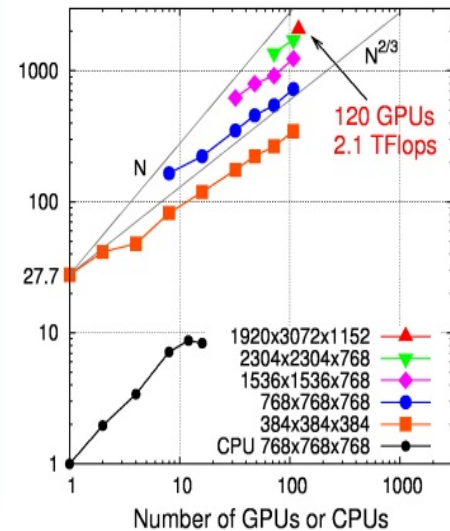
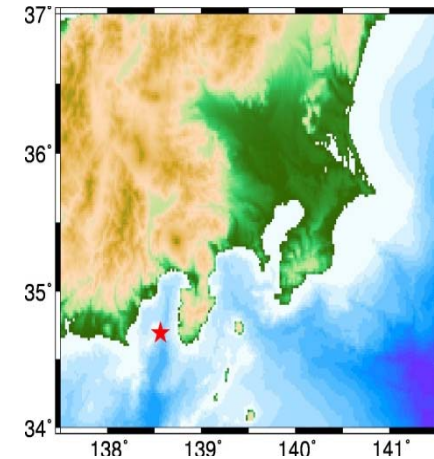
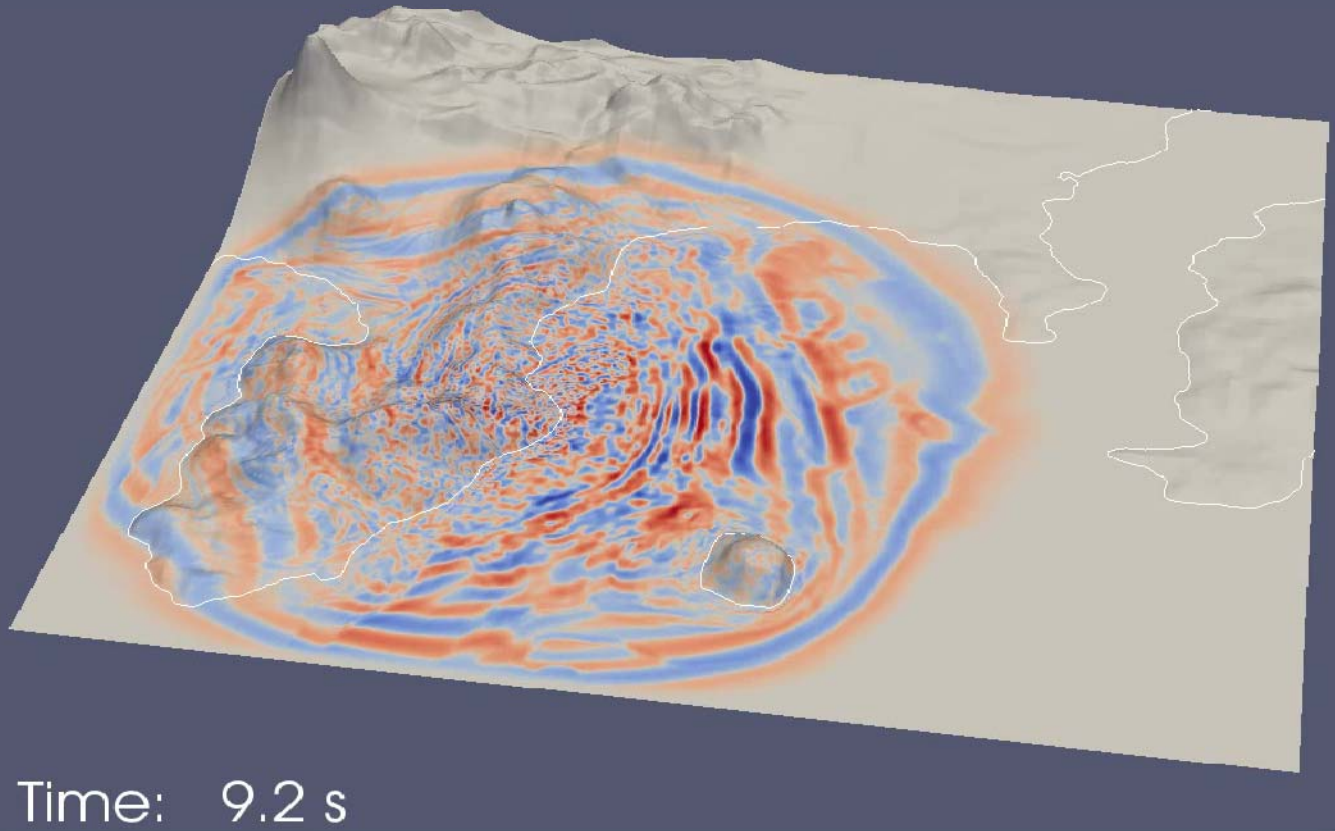
TSUBAME 2.0 Performance



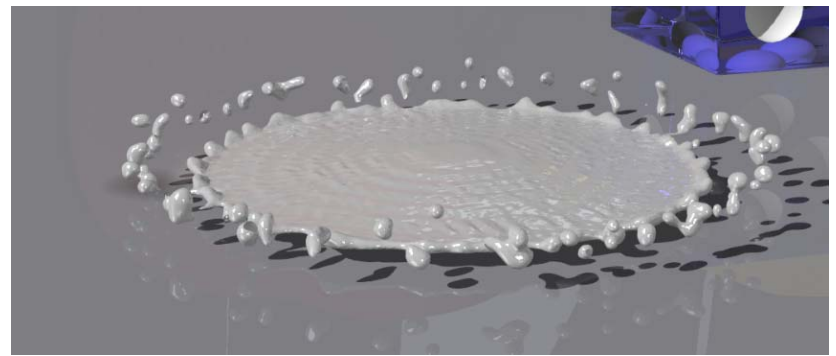
Earthquake Simulation

Prof. Taro Okamoto @ Tokyo Inst. Tech.

TSUBAME 1.2 120 GPUs
for **1920x3072x1152** 2.1 TFlops



Gas-Liquid Two-Phase Flow



Fluid Equations: Dam Breakage

- Navier-Stokes solver: Fractional Step
- Time integration: 3rd TVD Runge-Kutta
- Advection term: 5th WENO
- Diffusion term: 4th FD
- Poisson: AMG-BiCGstab
- Surface tension: CSF model
- Surface capture: CLSVOF (THINC + Level-Set)

Continuum eq.

$$\nabla \cdot \mathbf{u} = 0$$

Momentum eq.

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u} + \frac{1}{\rho} \mathbf{F}$$

Level-Set advection

$$\frac{\partial \phi}{\partial t} + (\mathbf{u} \cdot \nabla) \phi = 0$$

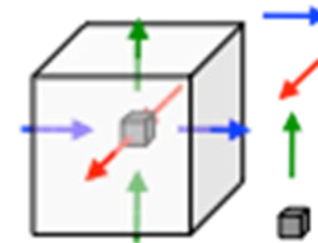
VOF continuum eq.

$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\mathbf{u} \psi) = 0$$

Level-Set re-initialization

$$\frac{\partial \phi}{\partial \tau} = \text{sgn}(\phi) (1 - |\nabla \phi|)$$

Staggered variable position



\mathbf{u} : velocity on x-direction

\mathbf{v} : velocity on y-direction

\mathbf{w} : velocity on z-direction

p, ϕ, ψ, ρ : scalar variables



Pulmonary Airflow Study

Prof. Takayuki Aoki
Collaboration with Tohoku University

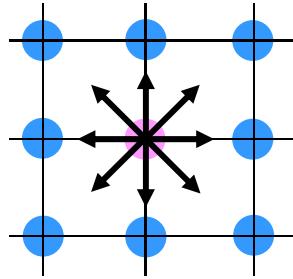
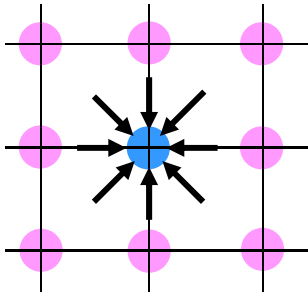
Lattice Boltzmann Method

$$\frac{\partial f_i}{\partial t} + \mathbf{e}_i \cdot \nabla f_i = -\frac{1}{\lambda} (f_i - f_i^{eq}) \quad f_i^{eq} = \rho w_i \left[1 + \frac{3}{c^2} (\mathbf{e}_i \cdot \mathbf{u}) + \frac{9}{2c^4} (\mathbf{e}_i \cdot \mathbf{u})^2 - \frac{3}{2c^2} (\mathbf{u} \cdot \mathbf{u}) \right]$$

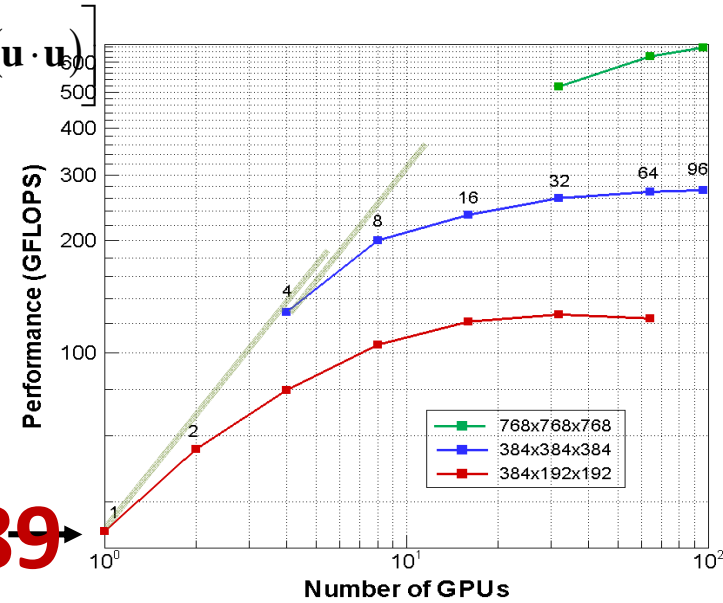
Collision step:

$$f_i(\mathbf{x} + \mathbf{e}_i \Delta t, t + \Delta t) = \bar{f}_i(\mathbf{x}, t) \quad \bar{f}_i(\mathbf{x}, t) = f_i(\mathbf{x}, t) - \frac{1}{\tau} (f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t))$$

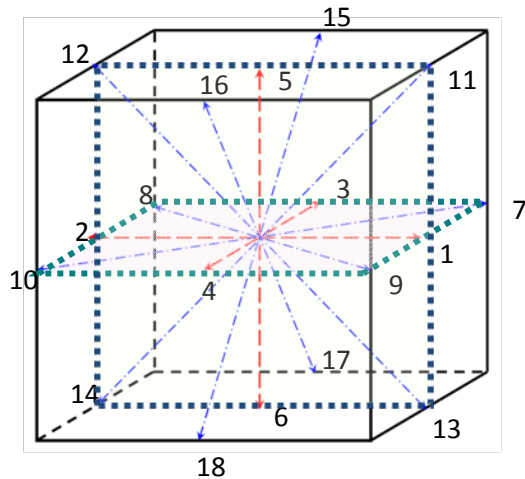
Streaming step:



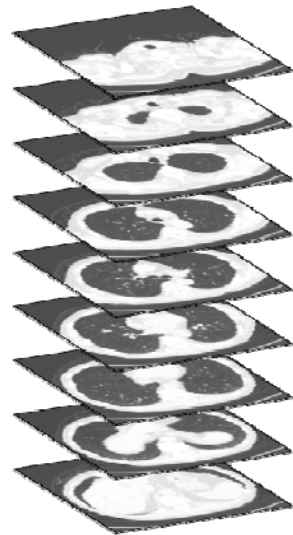
× 89



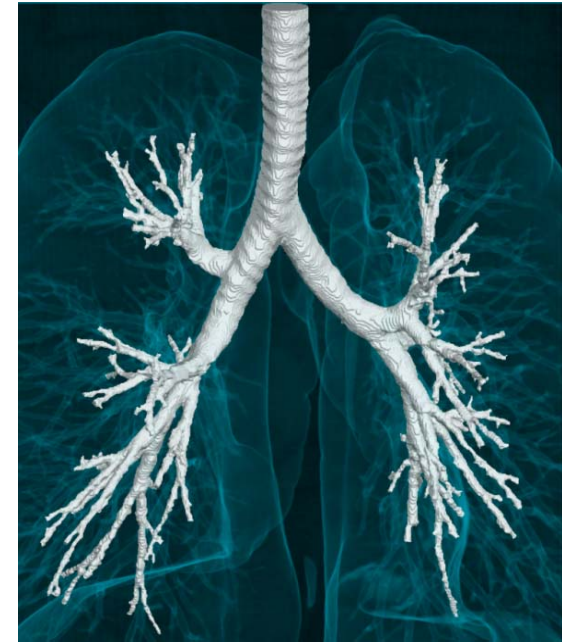
D3Q19



X-Ray CT images Airway structure Extraction

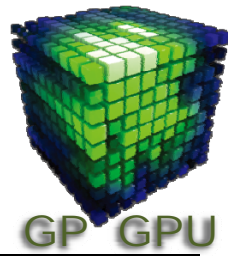


Lattice Boltzmann GPU computing

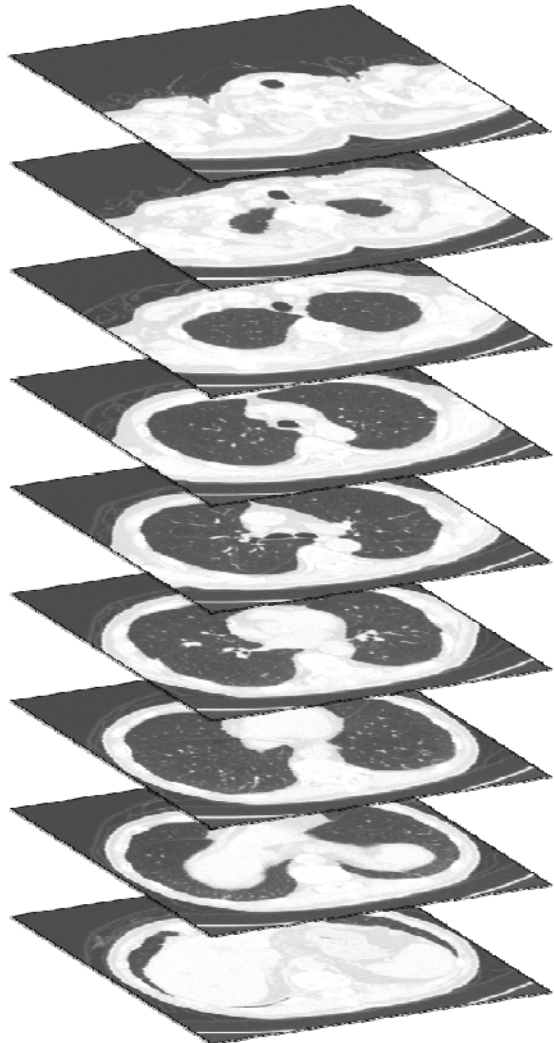


Pulmonary Airflow Study

Collaboration with Tohoku University



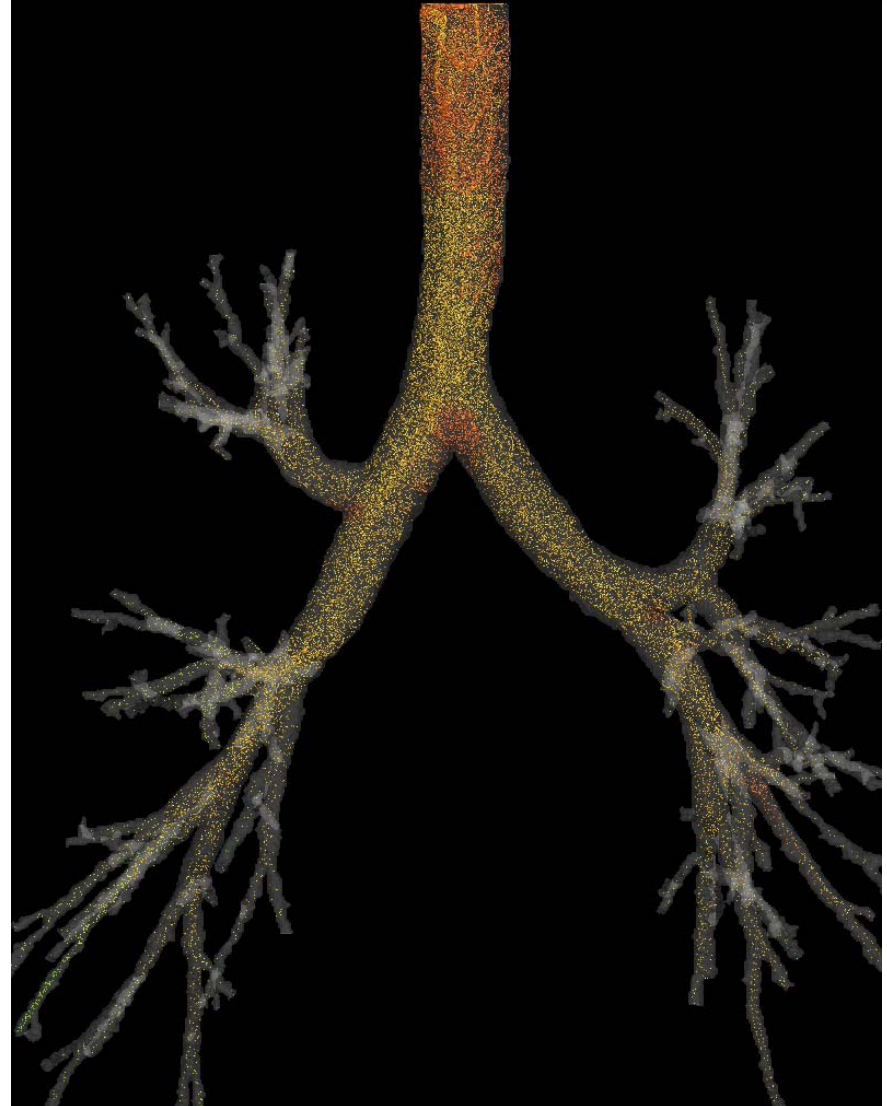
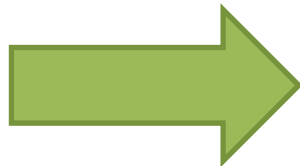
X-Ray CT images



Airway structure
Extraction

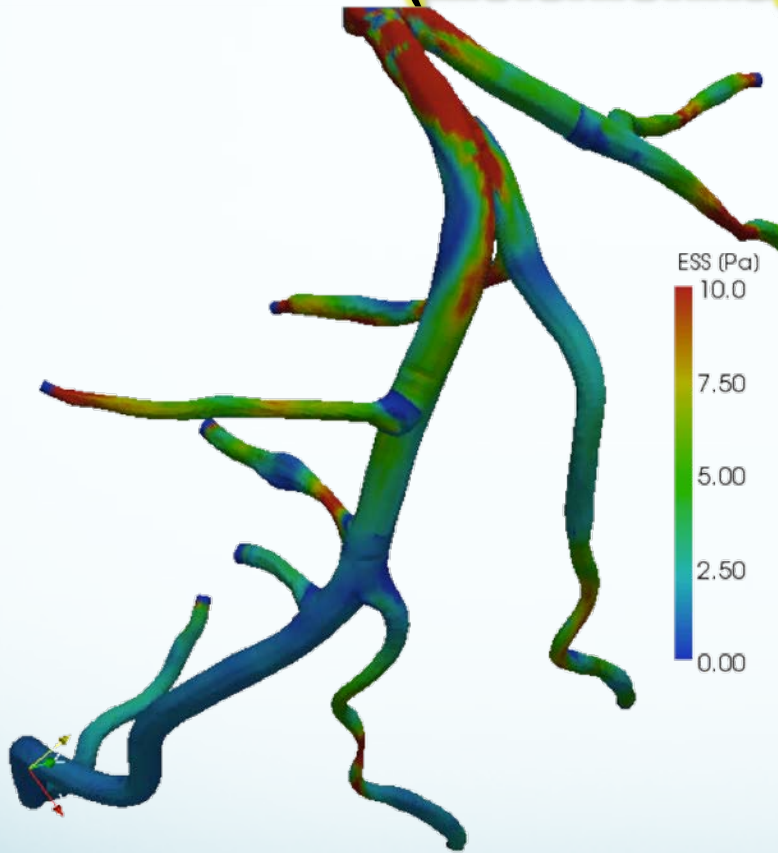


Lattice Boltzmann
GPU computing



MUPHY: Multiphysics simulation of blood flow

(Melchionna, Bernaschi et al.)



Combined Lattice-Boltzmann (LB) simulation for plasma and Molecular Dynamics (MD) for Red Blood Cells

Realistic geometry (from CAT scan)

Two-levels of parallelism: CUDA (on GPU) + MPI

- 1 Billion mesh node for LB component
- 100 Million RBCs

Results on Tsubame2 Supercomputer

Cluster of Nvidia M2050 GPUs connected by QDR Infiniband.
Scaling study up to 512 nodes (each node has 3 GPUs).
Very fast parallel I/O (read 100 GB in ~10 sec)

1 billion mesh nodes

GPUs	Time (s)	Efficiency
256	0.07616	N.A.
512	0.03852	98.86 %
1,024	0.01995	95.37 %
1,536	0.01343	94.43 %

Lattice Boltzmann Scaling
(time per step)

LB kernel: 1 GPU ~200 BG/P cores
1536 GPUs equivalent to full BlueGene/P

New run on FULL TSUBAME2.0 (4000 GPUs) just completed with an improved algorithm, exhibiting petascale performance(!)

1 billion mesh nodes + 100 million RBC

GPUs	Time (s)	Efficiency
256	0.44453	N.A.
512	0.25601	86.82%
1,024	0.14062	79.03%

Lattice Boltzmann +
Cell Dynamics Scaling
(time per step)

Time to completion on stationary flow:
23 minutes

Dendrite Solidification

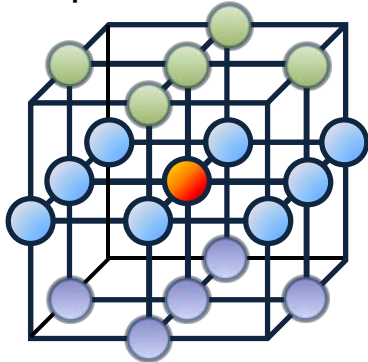
Allen-Cahn Equation based on Phase Field Model

$$\frac{\partial \phi}{\partial t} = M \left[\frac{\partial}{\partial x} \left(\varepsilon^2 \frac{\partial \phi}{\partial x} + \varepsilon \frac{\partial \varepsilon}{\partial \phi_x} |\nabla \phi|^2 \right) + \frac{\partial}{\partial y} \left(\varepsilon^2 \frac{\partial \phi}{\partial y} + \varepsilon \frac{\partial \varepsilon}{\partial \phi_y} |\nabla \phi|^2 \right) + \frac{\partial}{\partial z} \left(\varepsilon^2 \frac{\partial \phi}{\partial z} + \varepsilon \frac{\partial \varepsilon}{\partial \phi_z} |\nabla \phi|^2 \right) + 4W\phi(1-\phi) \left\{ \phi - \frac{1}{2} + \beta + a\chi \right\} \right]$$

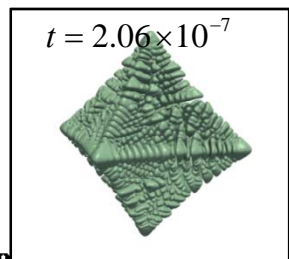
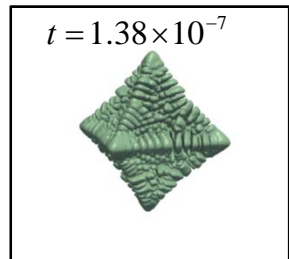
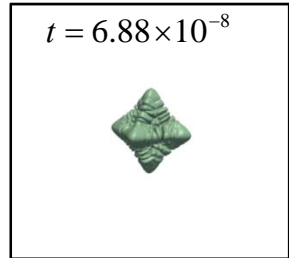
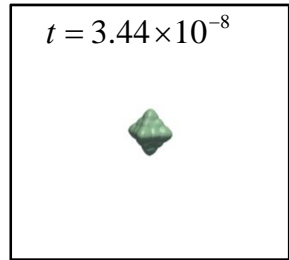
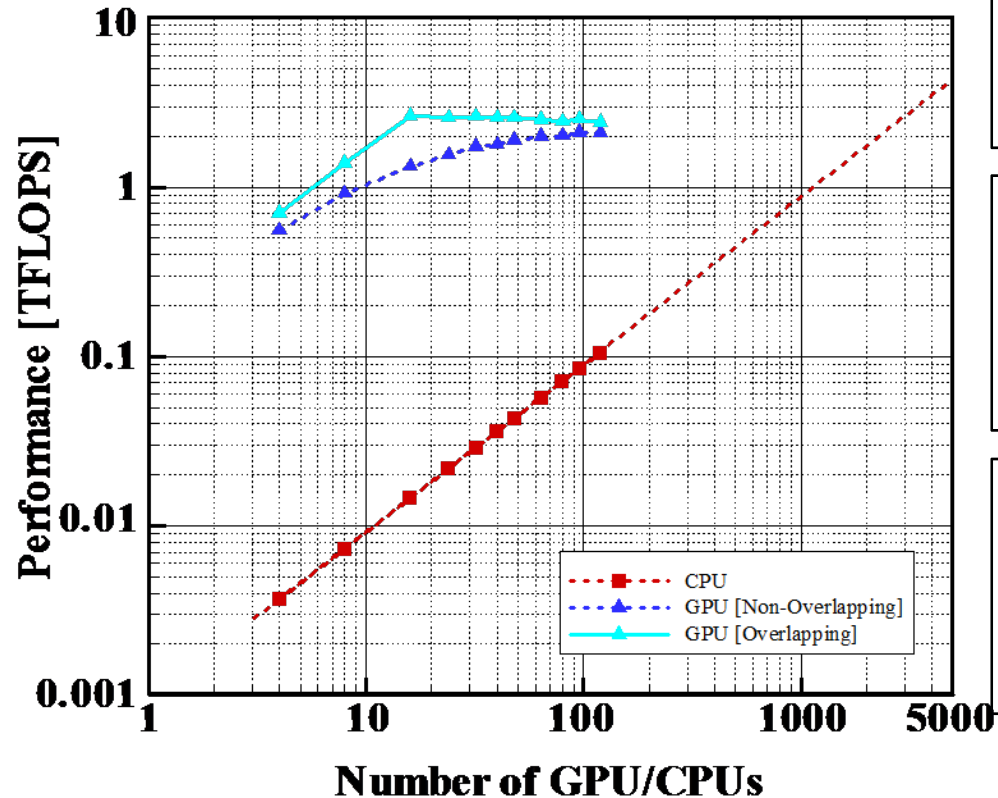
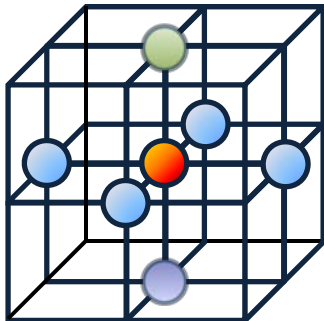
Thermal Conduction

$$\frac{\partial T}{\partial t} = k\nabla^2 T + \frac{L}{C} 30\phi^2(1-\phi) \frac{\partial \phi}{\partial t}$$

ϕ : 19 points to solve



T : 7 points to solve



The background of the slide is a grayscale micrograph of a metal surface. It displays a complex, textured pattern of dendrite solidification. The surface is covered with a dense network of fine, interconnected lines and larger, more irregular structures that resemble branching or 'flop' patterns. The overall appearance is rough and granular, with varying shades of gray indicating different levels of surface relief and texture. The text is overlaid on the upper portion of this image.

Dendrite Solidification for Pure Metal

Now Scales to flop

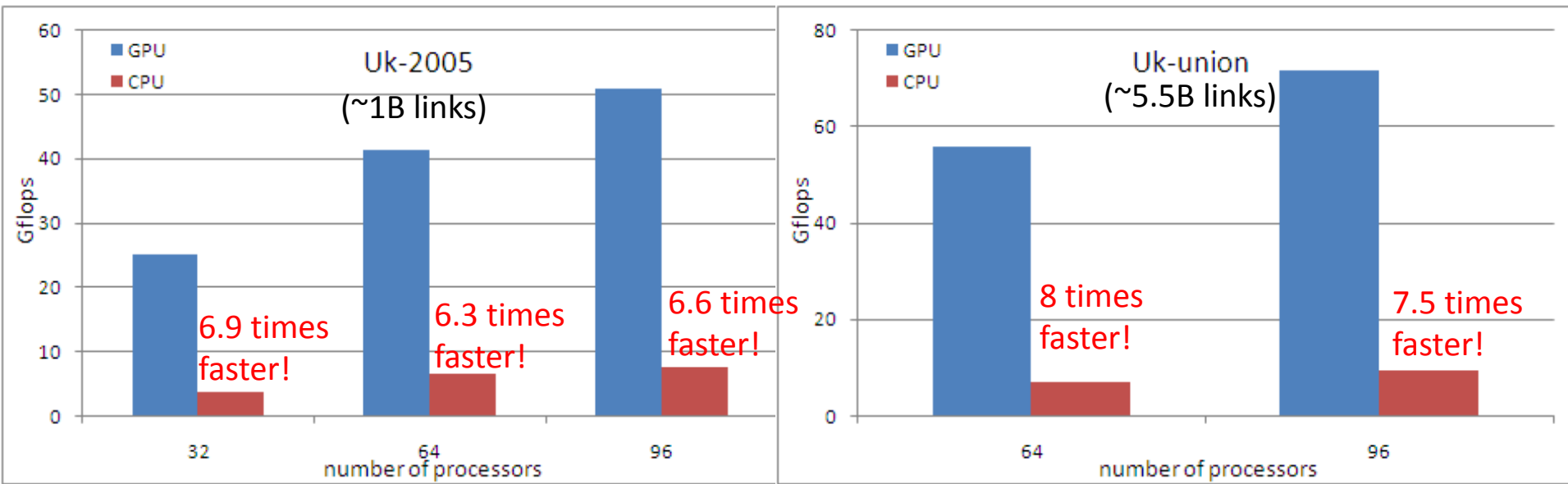
PageRank Results

(A. Cevahir, A. Nukada, S. Matsuoka)

- Huge data is processed -- bigger data scales better
- A state-of-the-art PageRank method is implemented

(based on *Cevahir, Aykanat, Turk, Cambazoglu: IEEE TPDS June 2010*)

(Problem bandwidth restricted – socket - to - socket comparison)



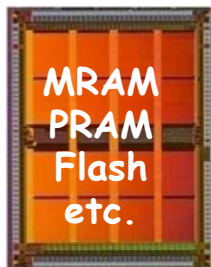
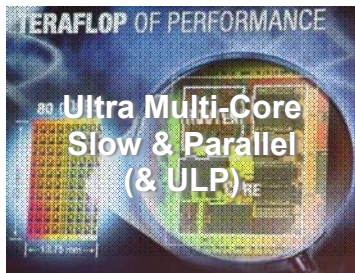
But it is not just HW Design...SOFTWARE(!)

- The *entire software stack* must preserve bandwidth, hide latency, lower power, heighten reliability for Petascaling
- Example: TSUBAME1.2, Inter-node GPU ↔ GPU achieves only 100-200MB/s in real apps
 - c.f. 1-2GB/s HW dual rail IB capability
 - Overhead due to unpinnable buffer memory?
 - New Mellanox driver will partially resolve this?
 - Still need programming model for GPU ↔ GPU
- Our SW research as CUDA CoE (and other projects such as Ultra Low Power HPC)

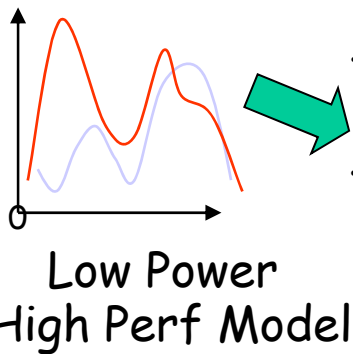
Tokyo Tech
GPU Research
JST-CREST ULP-HPC

(Ultra Low-Power HPC Project)
GPU Power Performance
Modeling and Heterogeneous
Scheduling

JST CREST ULP-HPC Scheme



Power Optimize using Novel Components in HPC



Auto-Tuning for Perf. & Power

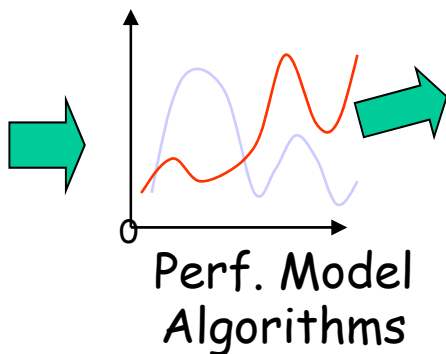
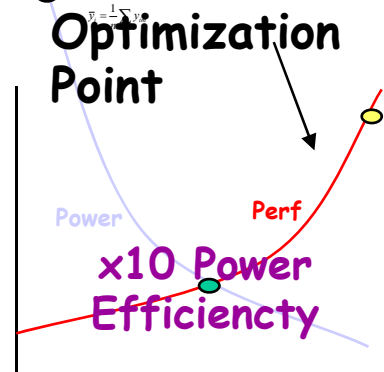
モデルと実測の Bayes 的融合

- Bayes モデルと事前分布
 - $y_i \sim N(\mu_i, \sigma_i^2)$
 - $\mu_i | \beta, \sigma_i^2 \sim N(\kappa_0^T \beta, \sigma_i^2 / \kappa_0)$
 - $\sigma_i^2 \sim \text{Inv-}\chi^2(v_0, \sigma_0^2)$
- n 回実測後の事後予測分布
 - $y_i | (y_{i1}, y_{i2}, \dots, y_{in}) \sim t_{v_n}(\mu_n, \sigma_n^2 \kappa_{n+1} / \kappa_n)$
 - $v_n = v_0 + n, \quad \kappa_n = \kappa_0 + n, \quad \mu_n = (\kappa_0 x_0^T \beta + n \bar{y}_i) / \kappa_n$
 - $v_n \sigma_n^2 = v_0 \sigma_0^2 + \sum (y_m - \bar{y}_i)^2 + \kappa_0 n (\bar{y}_i - x_0^T \beta / \kappa_n)^2$

ABCLibScript: アルゴリズム選択

```

IABCLibS static select region start
IABCLibS parameter (in CacheS, in NB, in NPrc)
IABCLibS select sub region start
IABCLibS according estimated
IABCLibS (2.0d0*CacheS*NB)/(3.0d0*NPrc)
IABCLibS select sub region end
IABCLibS select sub region start
IABCLibS according estimated
IABCLibS (4.0d0*CacheS*log(NB))/(2.0d0*NPrc)
IABCLibS select sub region end
IABCLibS static select region end
    
```



Power-Aware and Optimizable Applications

x1000 Improvement in 10 years

Tokyo Tech GPU Research

1. Auto-Tuning GPU

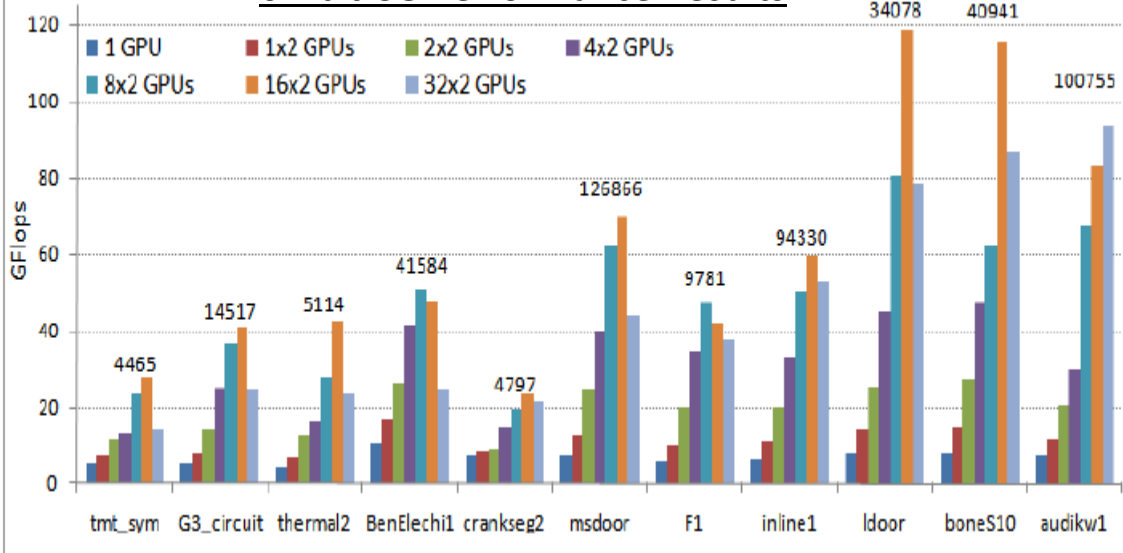
Numerical Libraries

“Auto-Tuning” Sparse Iterative Solvers on a Multi-GPU Clusters

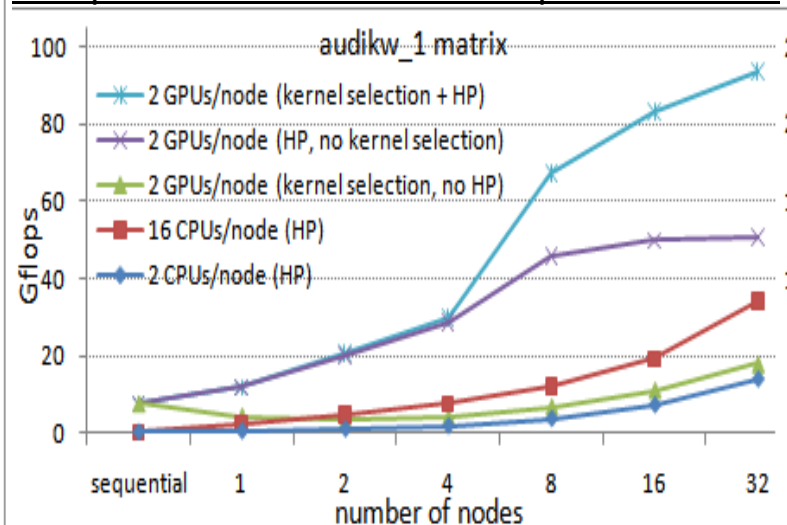
(A. Cevahir, A. Nukada, S. Matsuoka)

- High communication due to irregular sparsity
 - GPU computing is fast, communication bottleneck is more severe
- We propose techniques to achieve scalable parallelization
 - A new sparse matrix-vector multiplication (MxV) kernel [ICCS'09]
 - **Auto-selection of optimal running MxV kernel** [ISC'10]
 - Minimization of communication between GPUs [ISC'10]
- Two methods: **CG and PageRank (on 100CPUs)**

64-bit CG Performance Results

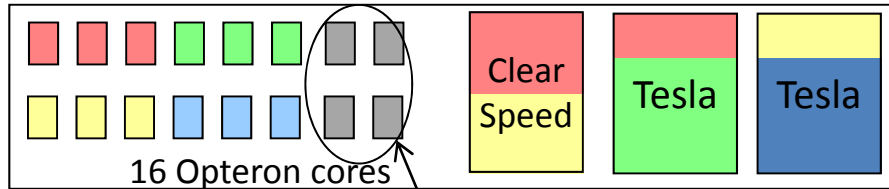


Comparisons of different CG implementations



High performance Linpack on heterogeneous Supercomputers [IPDPS 08,10]

- Dense linear computation on large heterogeneous systems
 - All types of processors are used for kernel (DGEMM)
 - Even supporting “heterogeneous nodes”
 - Load balancing techniques, tuning for reducing PCIe communication

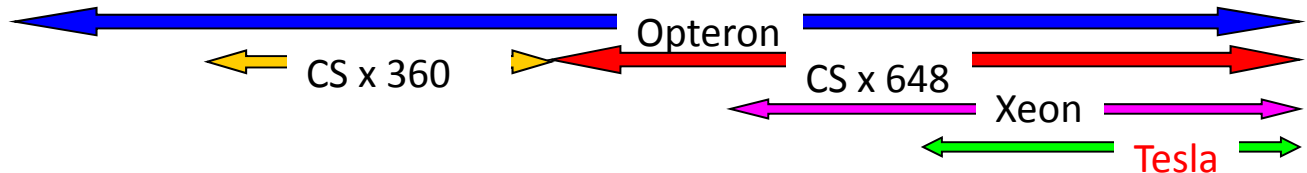


Mapping between MPI processes and heterogeneous processors

Some cores are dedicated for PCI comm

- >10,000 CPU cores, >600 Teslas, >600 ClearSpeed are cooperatively used
 - **87.01TFlops**, 53% efficiency → 7 times improvements on Top 500 ranking

	Jun06	Nov06	Jun07	Nov07	Jun08	Nov08	Jun09
Linpack speed	38.18TF	47.38	48.88	56.43	67.70	77.48	87.01
Rank	7	9	14	16	24	29	41



Measuring GPU Power Consumption

- Two power sources
 - Via PCI Express: $< 75\text{ W}$
 - Direct inputs from PSU: $< 240\text{ W}$
- Uses current clamp sensors around the power inputs

Precise and Accurate Measurement with Current Probes

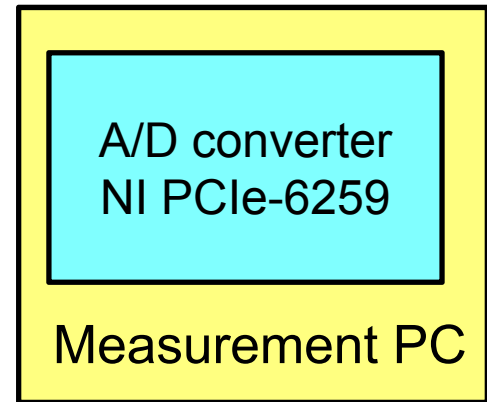


Attaches current sensors to two power lines in PCIe

+



Direct power inputs from PSU



Reads currents at 100 us interval

Statistical Modeling of GPU Power Consumption [IEEE HPPAC10]

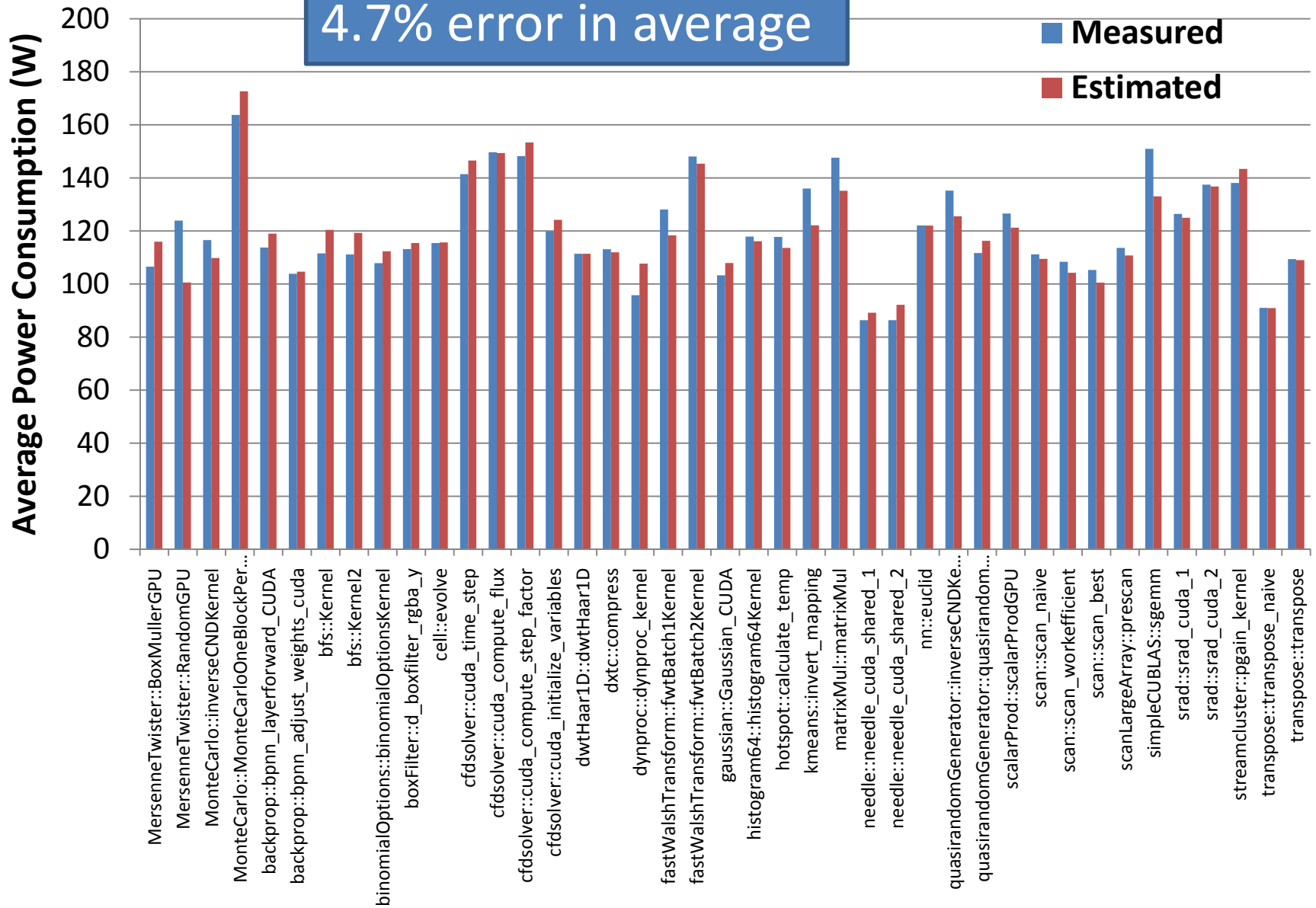
- Regularized linear regression
 - Finds linear correlation between per-second PMC values and average power of a kernel execution
 - Aggregates 3 kinds of global memory loads/stores
 - gld: $\text{gld}_{32b} + \text{gld}_{64b} * 2 + \text{gld}_{128b} * 4$
 - Regularization for avoiding overfitting to training data (Ridge Regression [10])

Average power of a kernel (Watts)

$$p = \sum_{i=1}^n \alpha_i c_i + \beta$$

Per-second PMC values

Actual vs Estimated Power



Low Power Scheduling in GPU Clusters

[IEEE IPDPS-HPPAC 09]

Objective: Optimize CPU/GPU

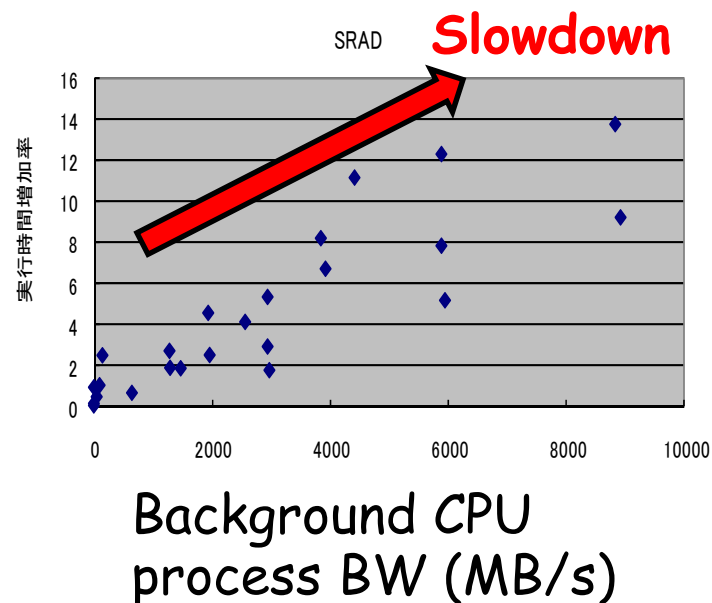
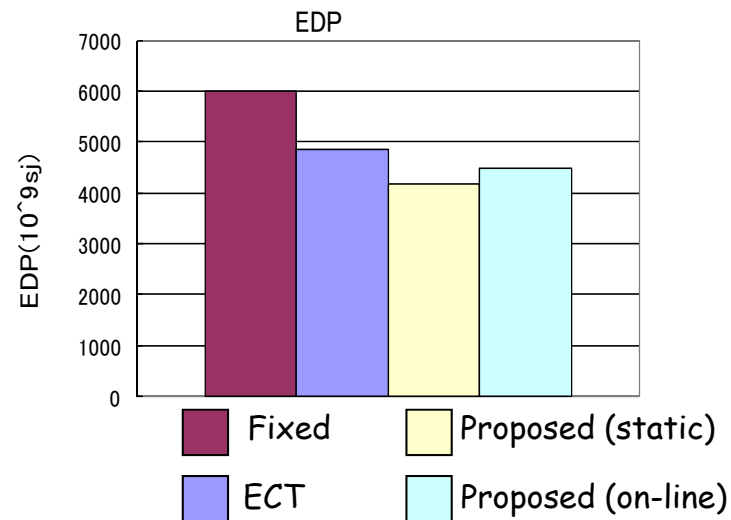
Heterogeneous

- Optimally schedule mixed sets of jobs executable on either CPU or GPU but w/different performance & power
- Assume GPU accel. factor (%) known

30% Improvement Energy-Delay Product

TODO: More realistic environment

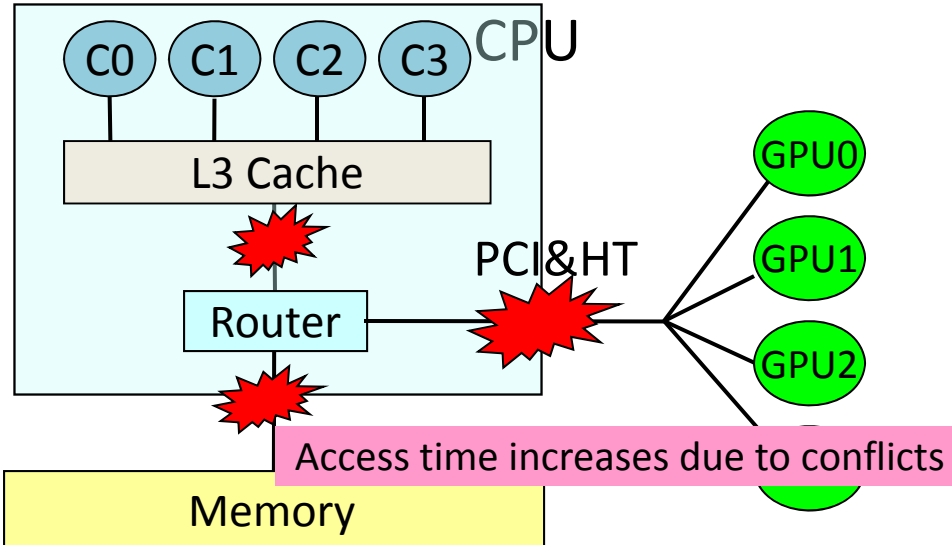
- Different app. power profile
- PCI bus vs. memory conflict
 - GPU applications slow down by 10 % or more when co-scheduled with memory-intensive CPU app.



Intelligent Task Scheduling for GPU clusters

- considering conflicts on memory bus and PCI-E bus -

4 processes share a node.



Assuming the following information of each job is known

- Execution time (at stand alone case)
- Total memory access (from performance counter)
- Total PCI-E transfer (from CUDA profiler)

Actual execution time is estimated by our model.

T. Hamano, et al. IPSJ SIGHPC-124 (domestic)

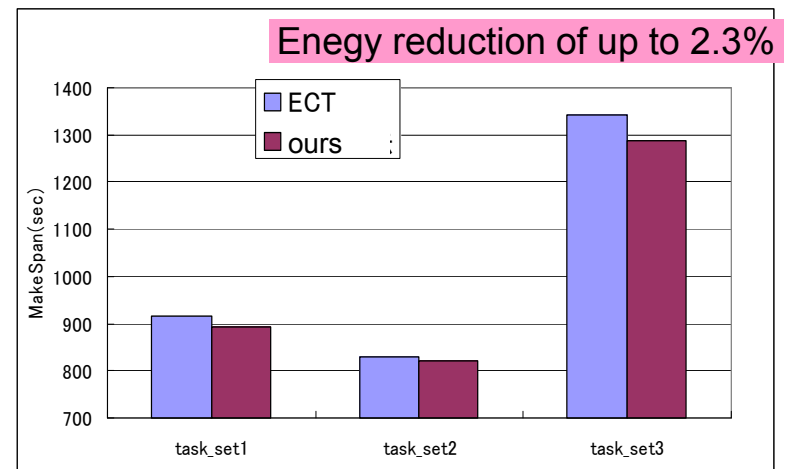
Status of each process is one of the following:

- (1) Memory access
- (2) PCI transfer (assume blocking cuMemcpy)
- (3) No memory access

Speed-down ratio of a process = r (percentage of each status) $\times \alpha$ (conflict coefficient)

$$\sum_{s_0} \sum_{s_1} \sum_{s_2} \sum_{s_3} \alpha(s_0, \{s_1, s_2, s_3\}) r_0(s_0) r_1(s_1) r_2(s_2) r_3(s_3)$$

We integrated this model into ECT(Earliest Complete Time) scheduling.



Tokyo Tech GPU Research
CUDA Center of Excellence
GPU Fault Tolerance

NVCR : A Transparent Checkpoint/Restart for CUDA

CUDA Checkpoint using BLCR (Takizawa, 2009)

- Save all data on CUDA resource.
- Destroy all CUDA context.
- BLCR
- Reallocate CUDA resource.
- Restore data.

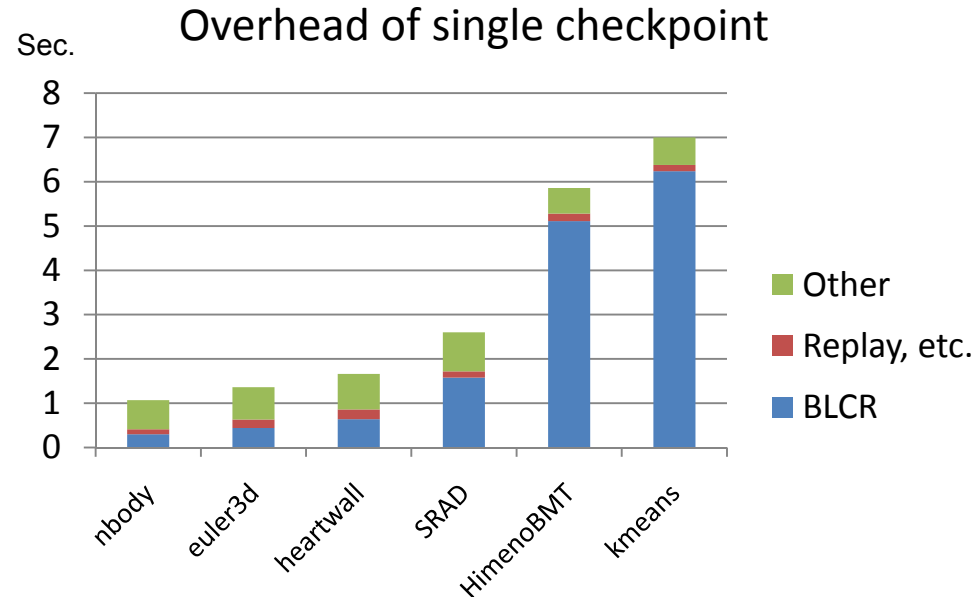
After reallocation, the address or handle may differ from previous one.

Handles are Opaque pointers : can be virtualized.
Addresses must be visible for user application.

Our answer is `REPLAY`.

NVCR records calls of all APIs related to device memory address such as;
`cuMem*()`, `cuArray*()`, `cuModule*()`.

We added some optimization of reducing the API records. For example, locally resolved alloc-free pairs are removed from the record.



The time for Replaying is quite negligible compared with the main BLCR procedure to write image to HDD.

NVCR supports

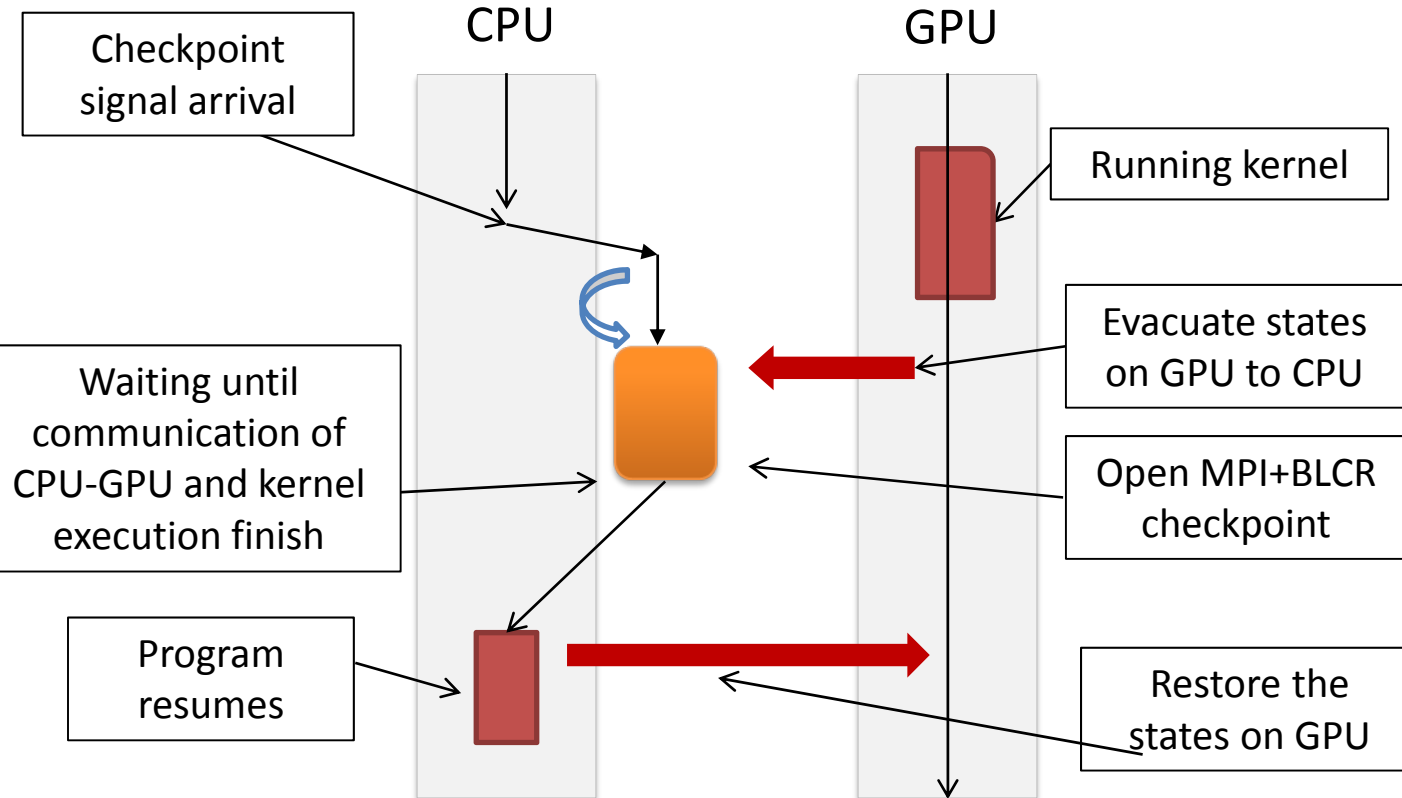
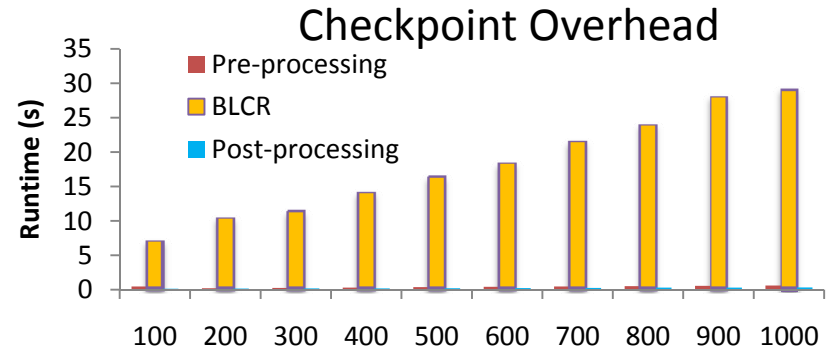
- All CUDA 3.0 APIs
- OpenMPI
- Both CUDA runtime and CUDA driver API.

CUDA pinned memory is supported partially ... Full support requires NVIDIA's official support.

MPI CUDA Checkpointing

Checkpointing for MPI+CUDA applications

- Very important in large-scale GPU systems
- Supports a majority of the CUDA RT API
- Based on BLCR and OpenMPI



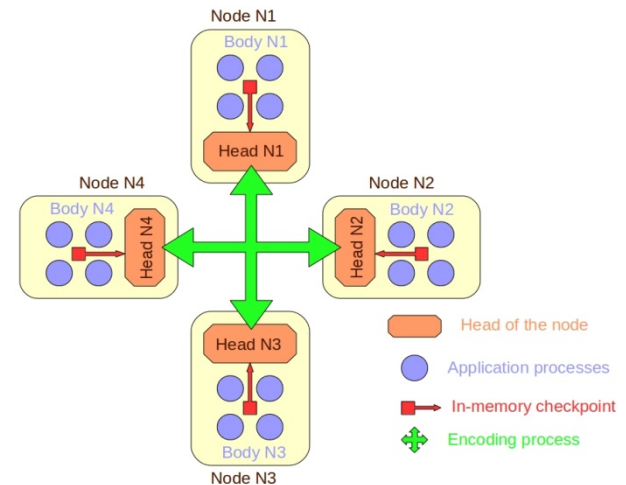
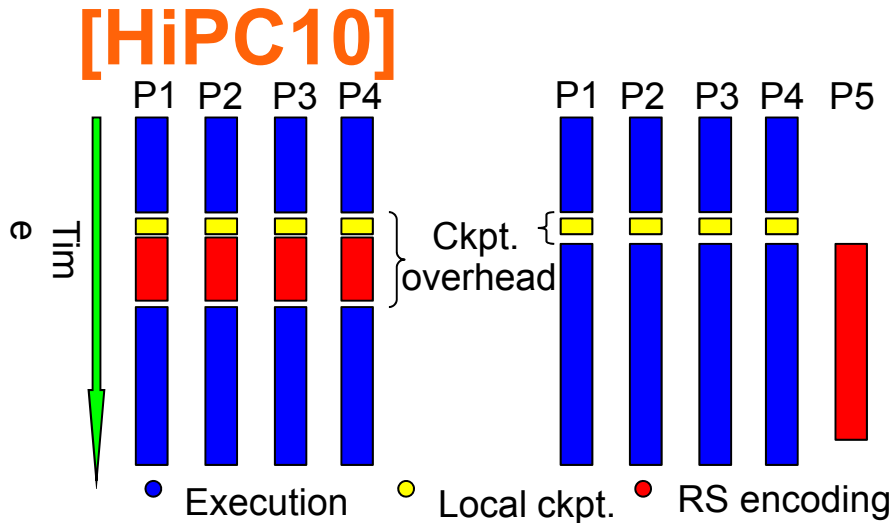
Data size (MB)

- GPU state saving and restoring has little overall performance impact.
- Can be significantly faster with our scalable checkpointing algorithms

**More details at
the Research
Poster Session**

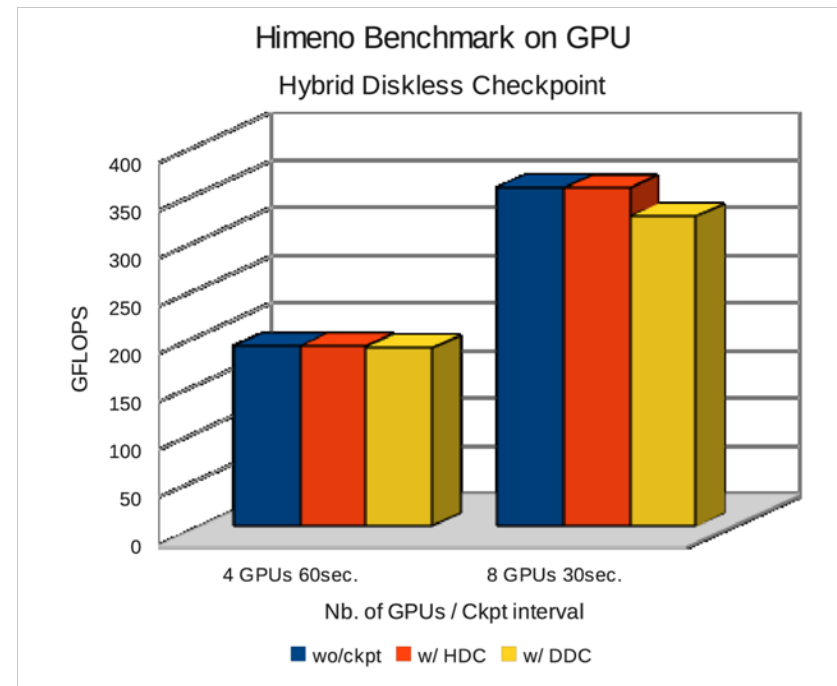
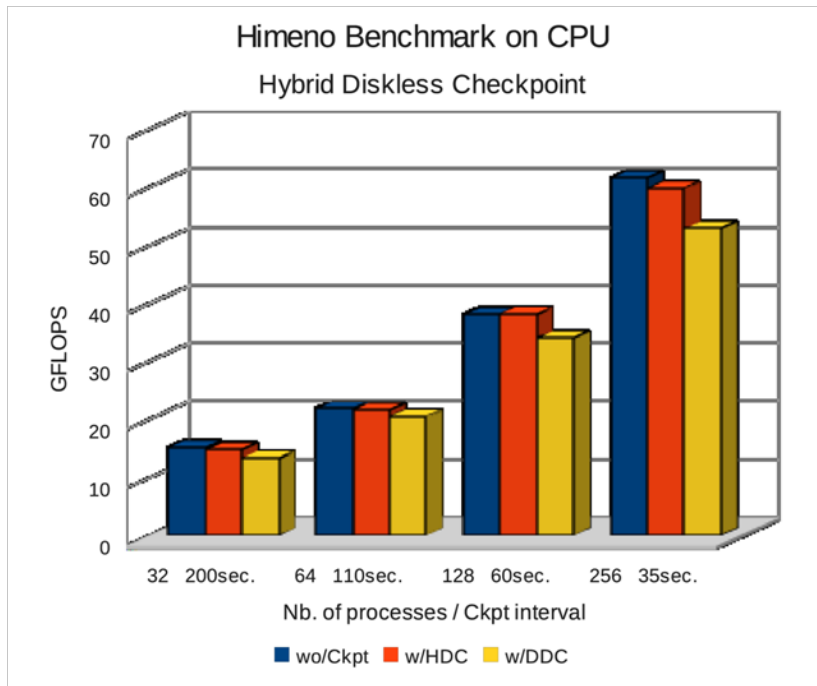
Hybrid Diskless Checkpoint

- **Problem:** Decrease the ckpt. overhead with erasure codes on large-scale GPU systems.
- Scalable encoding algorithm and efficient group mapping [CCGrid10]
- Fast encoding work on idle resource (CPU/GPU) [HiPC10]



Hybrid Diskless Checkpoint

- Less than 3% of ckpt. overhead using idle resources



- We are currently combining this technique with our MPI CUDA Checkpoint.

Software Framework for GPGPU Memory FT

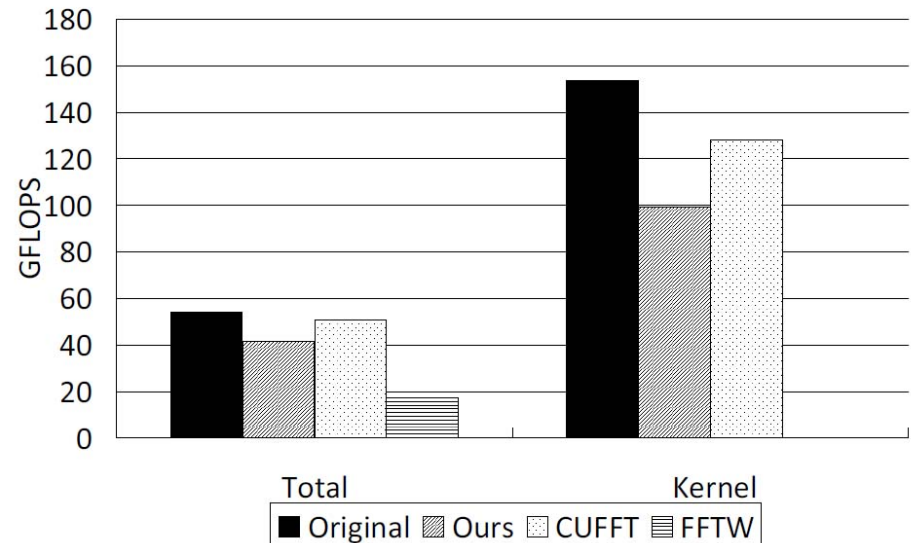
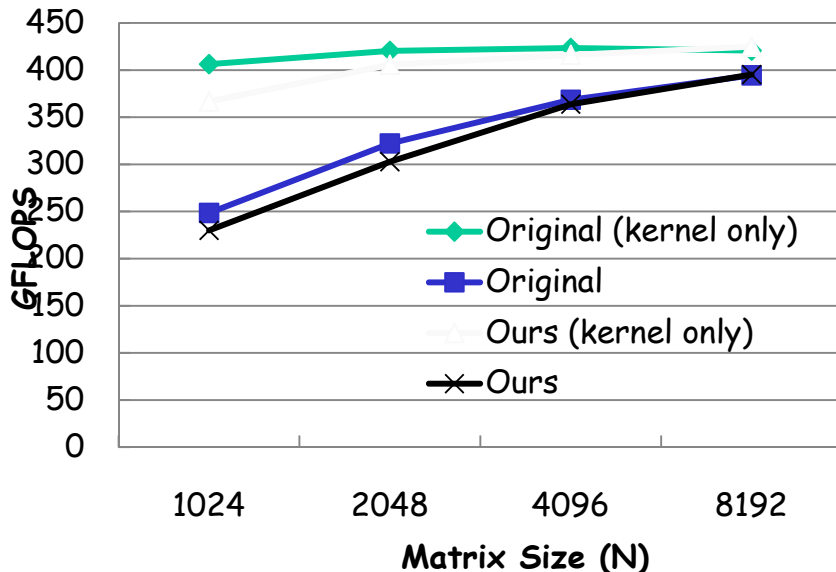
[IEEE IPDPS 2010]

- Error detection in CUDA global memory + Checkpoint/Restart
- Works with existing NVIDIA CUDA GPUs

Lightweight Error Detection

- *Cross Parity* for 128B blocks of data
- Detects a single-bit error in a 4B word
- Detects a two-bit error in a 128B block
- No on-the-fly correction → Rollback upon errors

Exploit the latency hiding capability of GPUs for data correctness



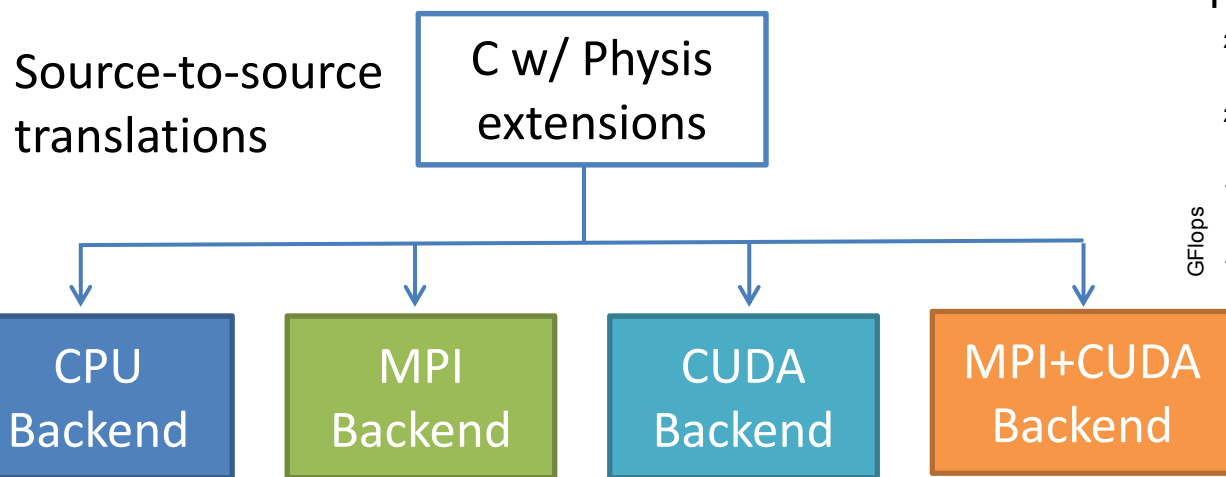
Tokyo Tech GPU Research

JST-ANR

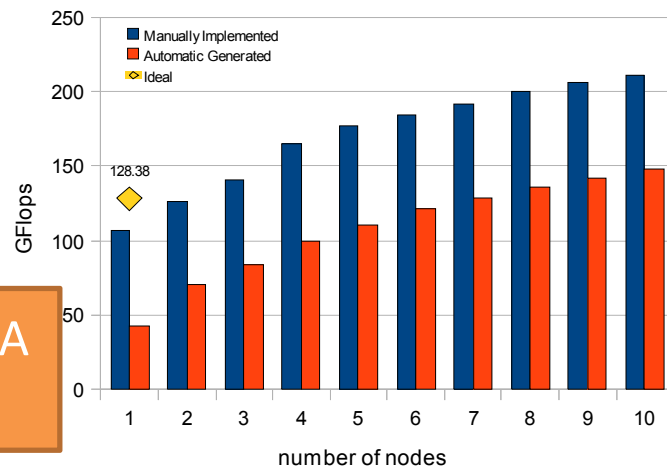
GPU Higher-Level Programming
Model

Physis: A Domain Specific Framework for Large-Scale GPU Clusters

- Portable programming environment for stencil computing on structured grids
 - Implicitly parallel
 - Distributed shared memory
- Provides concise, declarative abstractions for stencil computing



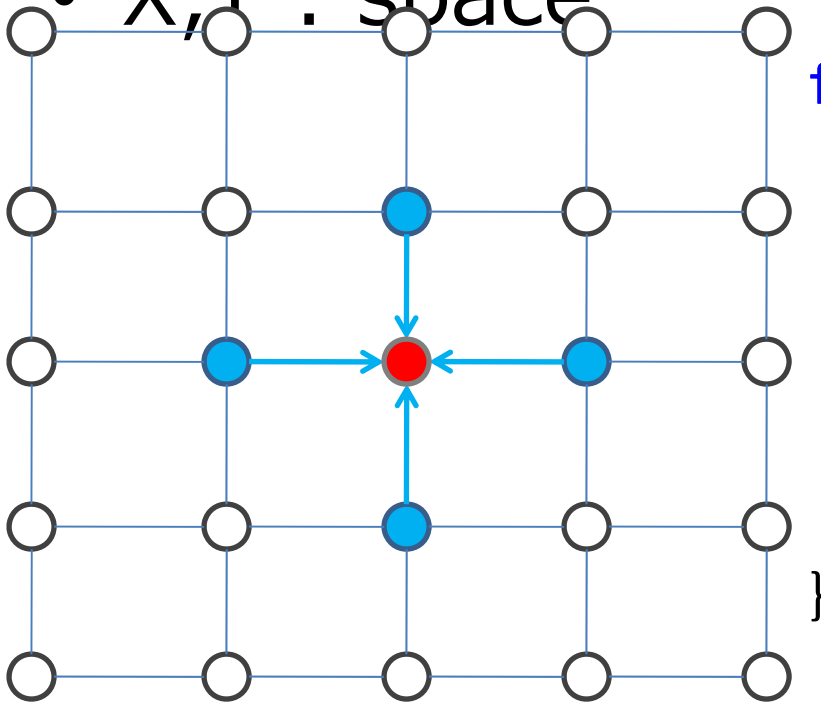
Preliminary Performance Results



A five-point stencil

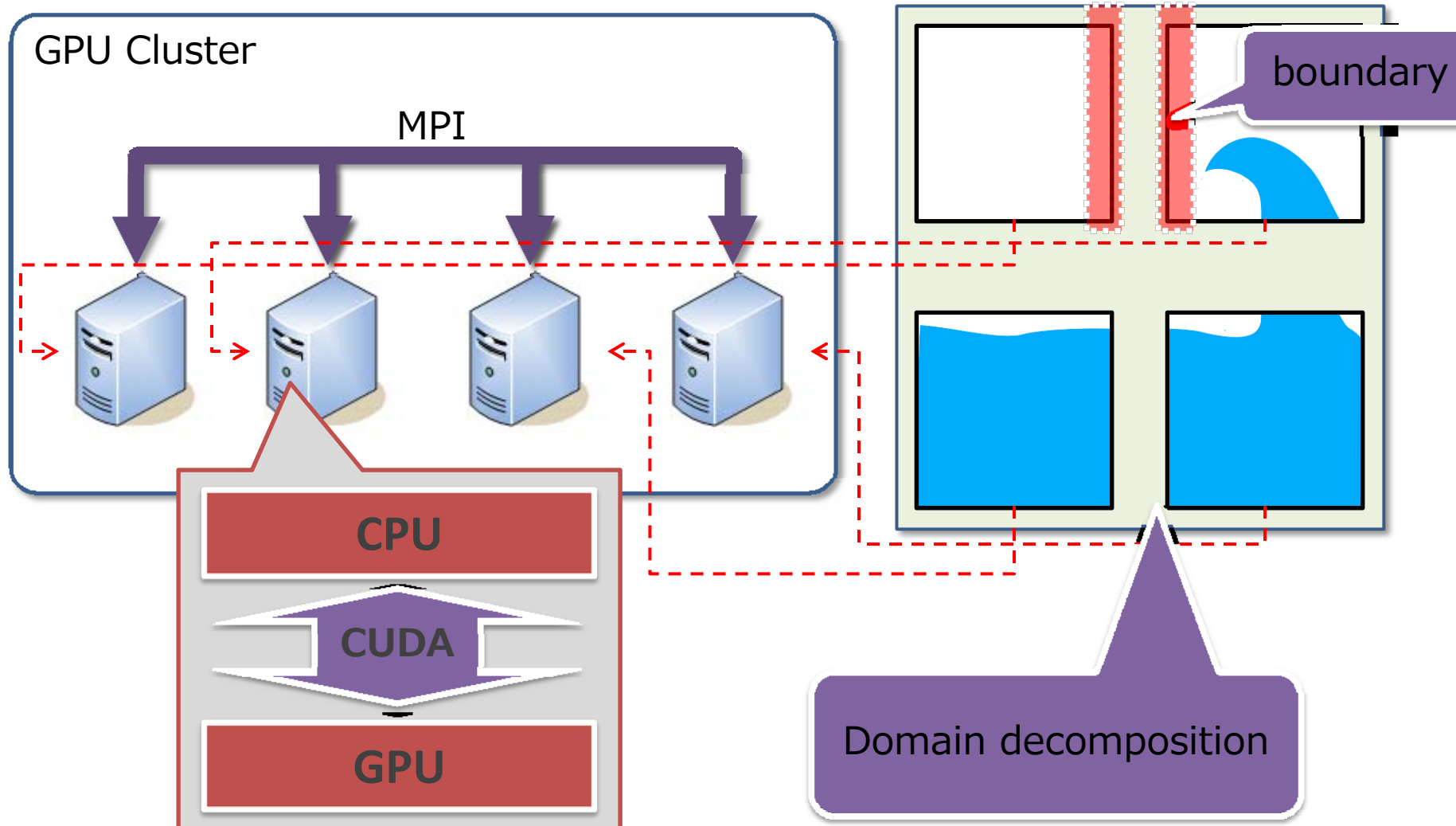
- T : time

- X, Y : space



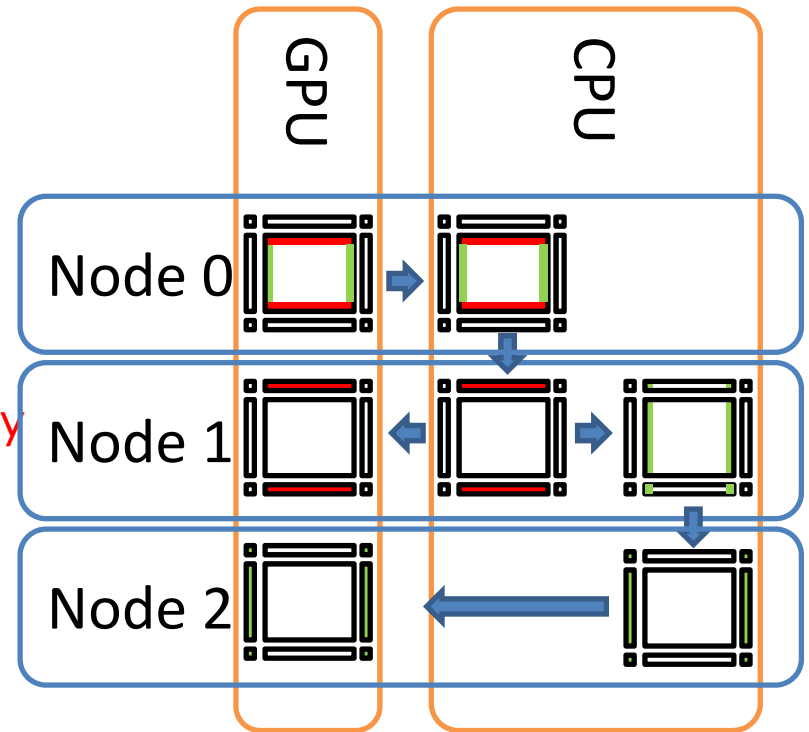
```
for (t = 0.0; t < T; t += dt) {  
  for (y = 0; y < Y; y++) {  
    for (x = 0; x < X; x++) {  
      new_f(x, y) = e1*old(y, x)  
        + e2*old(y, x-1) + e3*old(y, x+1)  
        + e4*old(y-1, x) + e5*old(y+1, x);  
    }  
  }  
}
```

Implementation on GPU clusters



Complexity of implementation

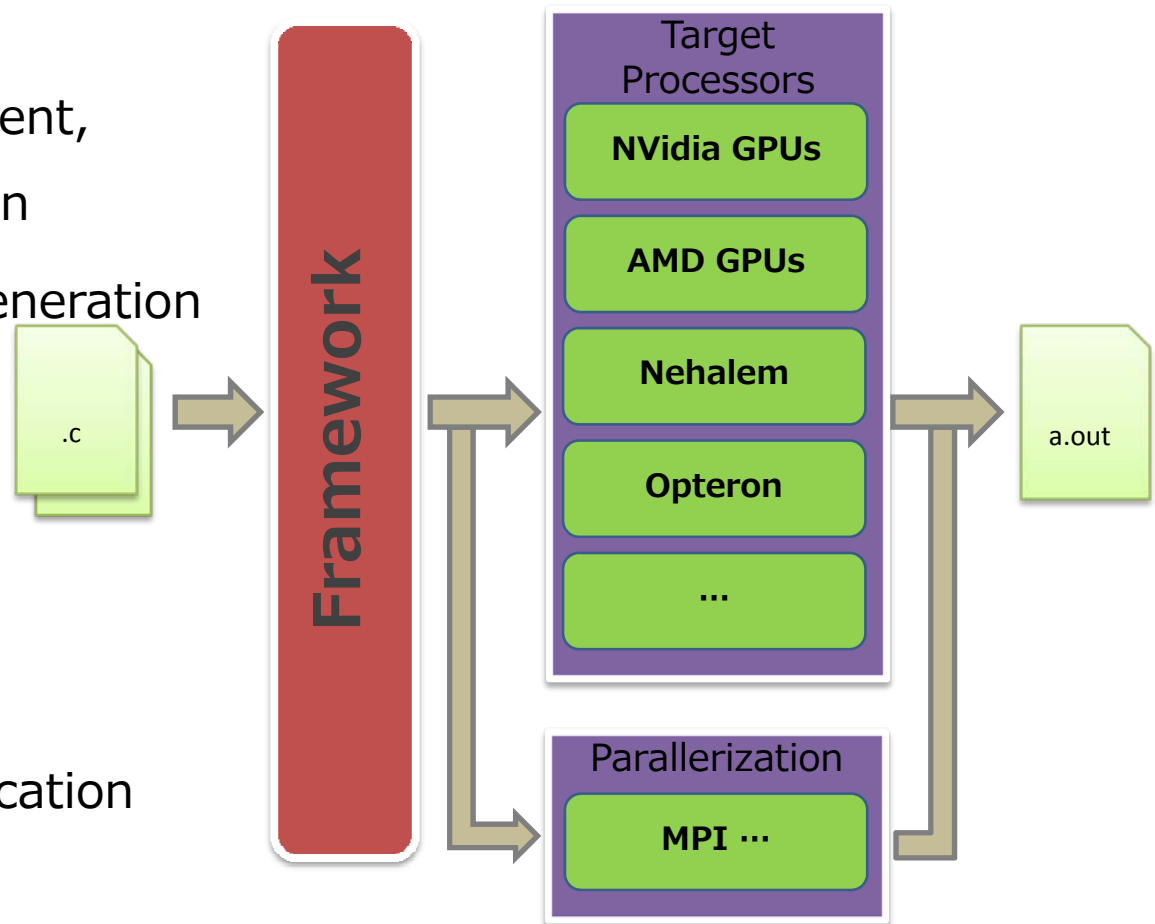
- CPU, GPU, MPI
- Code for computation is concise, code for parallelization isn't.
 - Problem decomposition
 - Boundary exchange
 - GPUs cannot communicate directly
 - GPU->CPU -> CPU->GPU
- Code for optimization brings more complexity
- Most difficult parts are non-essential for stencil computation



Procedure of boundary exchange

Overview of the framework

- Architecture independent, global view description
- CPU,GPU,MPI code generation
- Code 2 Code
- Optimization
- Auto-tuning
- Checkpointing
- Optimal resource allocation



An example of a 7-point stencil

```
__stencil__ void average(int x, int y, int z, grid3d_real g) {  
    float ret = psGridGet(g, 0, 0, 0)  
        + psGridGet(g, -1, 0, 0) + psGridGet(g, 1, 0, 0)  
        + psGridGet(g, 0, -1, 0) + psGridGet(g, 0, 1, 0)  
        + psGridGet(g, 0, 0, -1) + psGridGet(g, 0, 0, 1);  
    psGridEmit(g, ret / 7.0);  
}
```

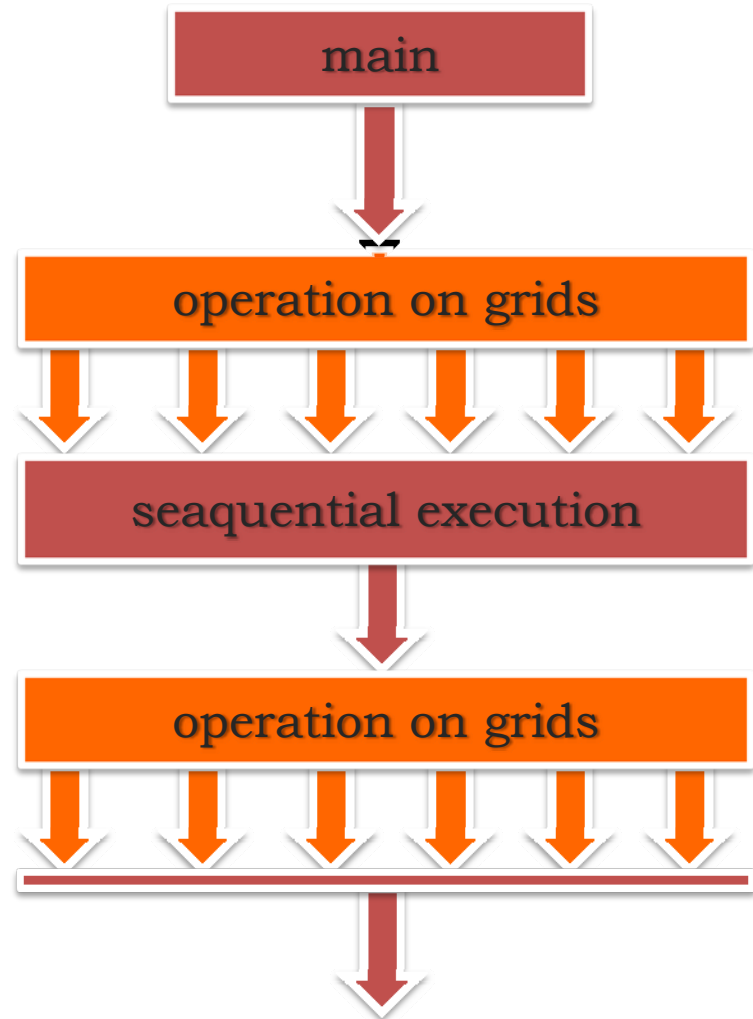
**Parallel
execution**

```
void computation(float *inbuff, float *outbuff) {  
    PS_Grid g = psGridNew(float, N, N, N);  
    psGridCopyIn(g, inbuff);  
    for (int t = 0; t < T; t += dt) psStencilMap(average, g);  
    psGridCopyOut(g, outbuff);  
}
```

Apply stencil kernel to a grid

Execution Model

```
__stencil__
void kernel(int x, int y, int z, grid g) {
    float val = (psGridGet(-1,0,0)
        + psGridGet(1,0,0)) / 2.0;
    psGridEmit(g, val);
}
int main(int argc, char *argv[]) {
    psInit(&argc, &argv);
    int i;
    PS_Grid g = psGridNew(float, 256, 256, 256);
    printf("start\n");
    int *buff = (int *)malloc(256*256*256*4);
    for (i = 0; i < 256; i++) {
        buff[i] = rand();
    }
    psGridCopyIn(g, buff);
    PS_Dom dom = psDom(0,255,0,255,0,255);
    for (i = 0; i < 100000; i++) {
        psStencilMap(kernel, dom, g);
    }
    psGridCopyOut(g, buff);
    printf("end\n");
    psFinalize();
    return 0;
}
```

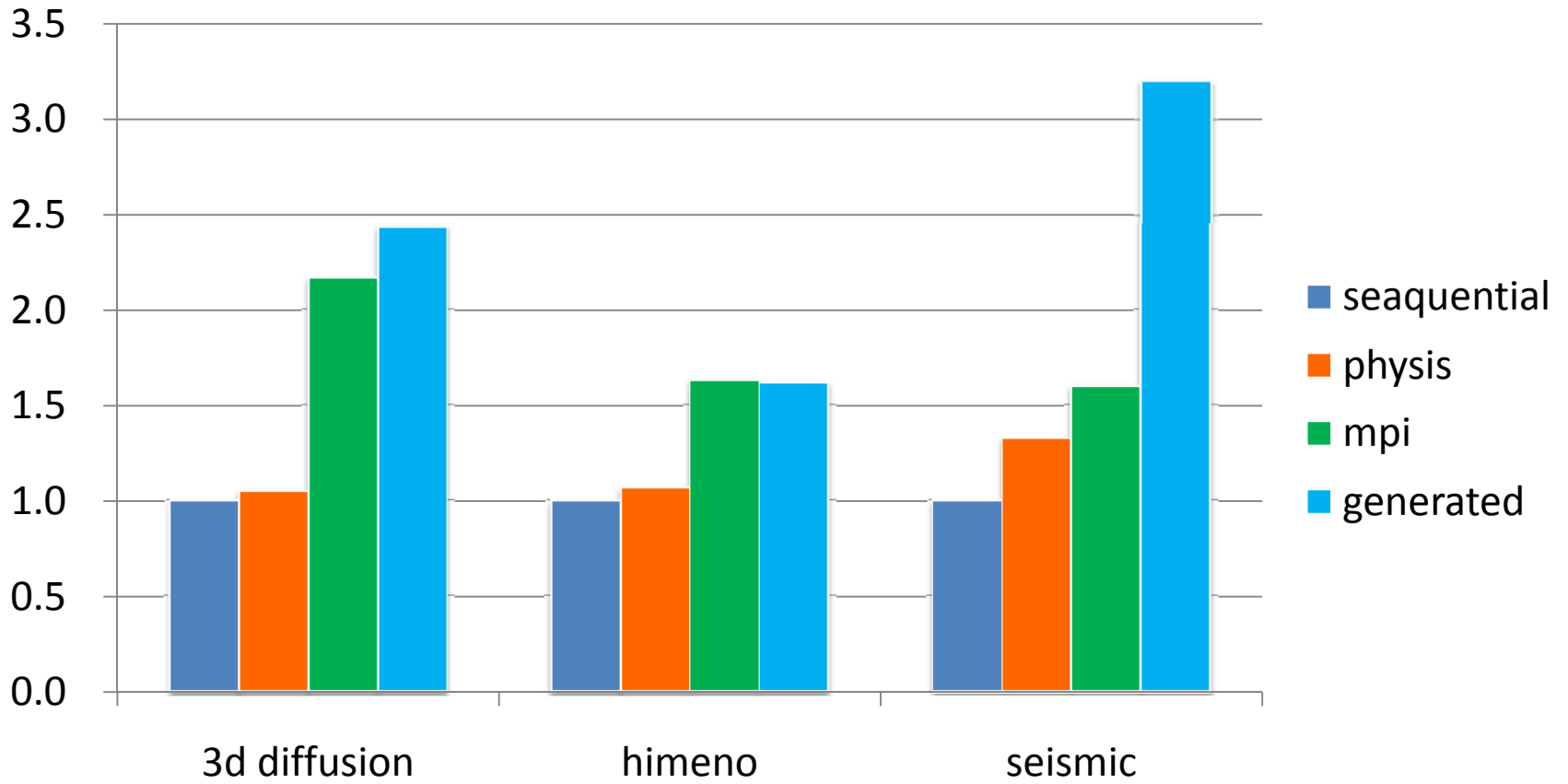


Evaluation

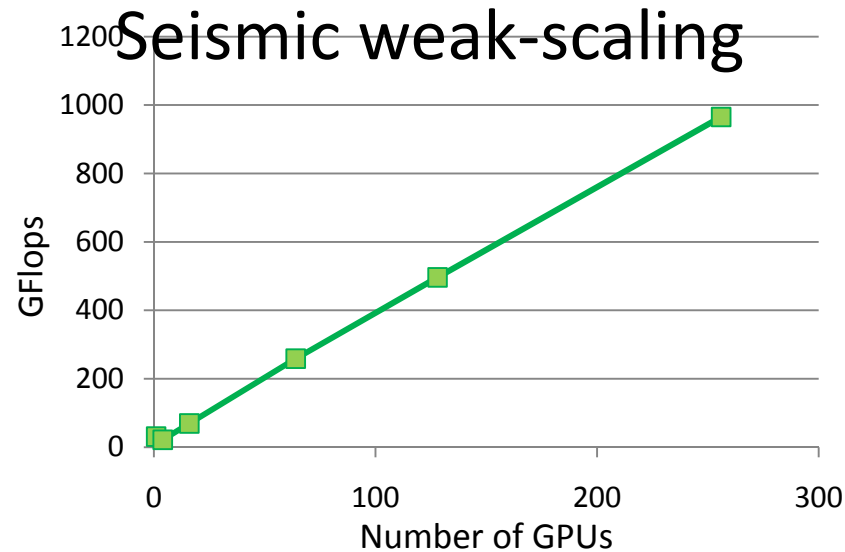
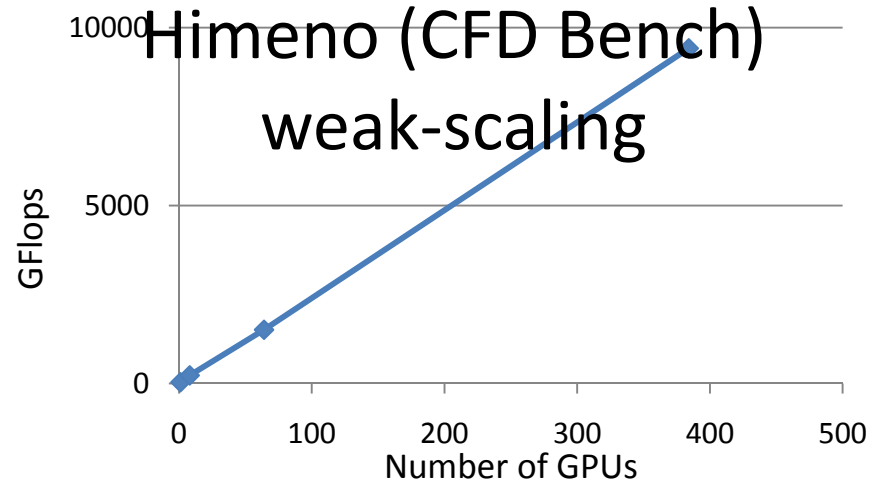
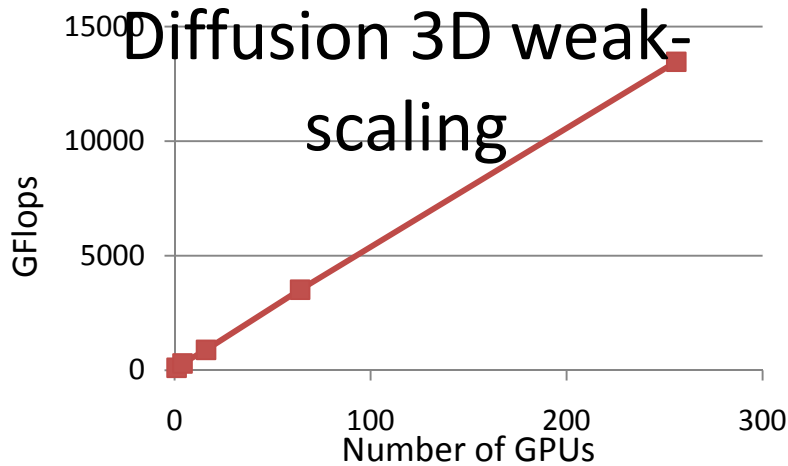
- Evaluated the performance and scalability of Physis on TSUBAME 2.0
 - 1D/2D decomposition
 - Overlapping of boundary exchange and computation
- Target applications
 - 3D diffusion equation
 - Himeno benchmark
 - Seismic wave simulation code
 - Free surface boundary condition is not supported

Productivity

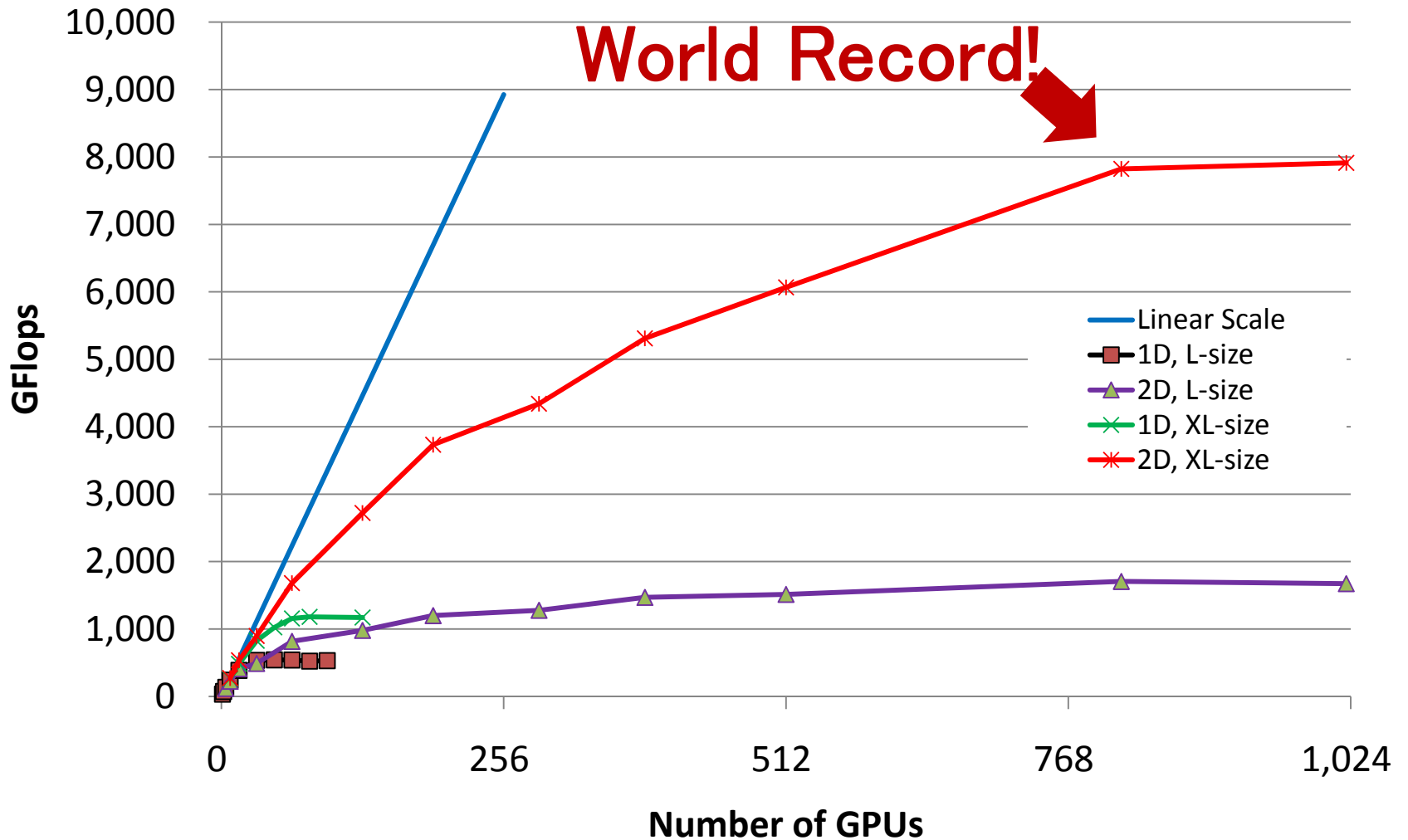
Lines of code



Weak Scalability

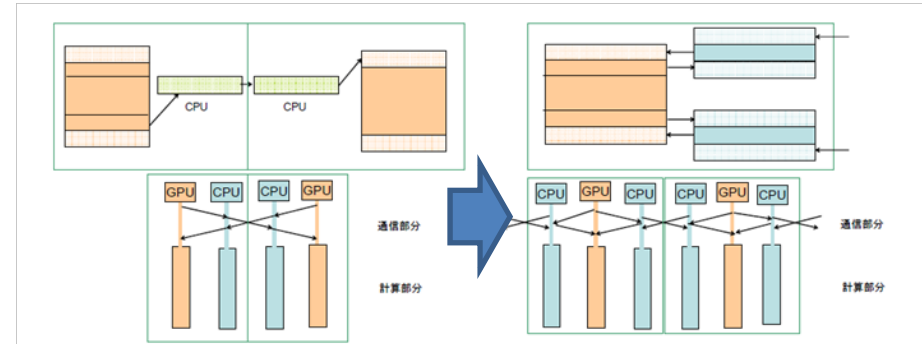


Strong scalability – Himeno bench



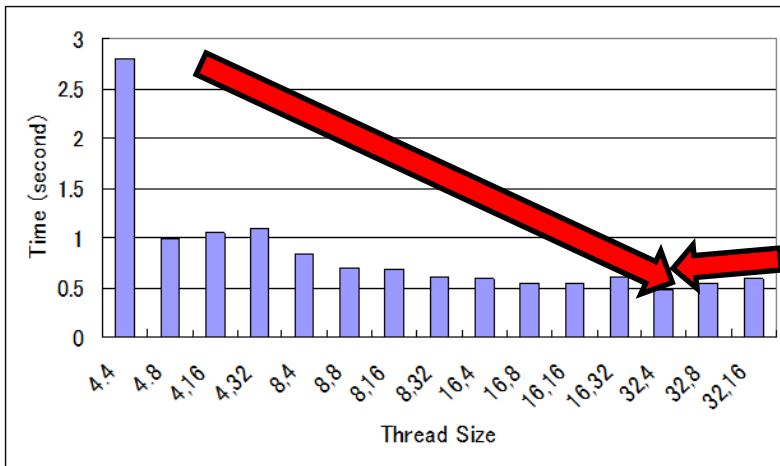
Towards Auto-Tuning of Stencil Computation on GPU-accelerated Clusters

- Programming on GPU clusters gets harder because of
 - Hybrid programming with CUDA/MPI/threads
 - # of tuning parameters is increasing.
 - Hiding latency
- Auto-tuning is more important



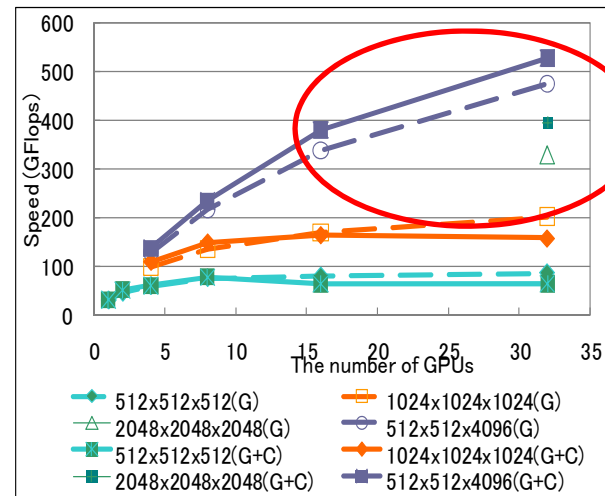
3-HOP communication has been required, for GPU<->CPU<->CPU<->GPU

→ 1-HOP method is proposed, and auto-tune between communication methods



We find the optimal parameters, such as block size, by auto-tuning

→ x60 faster than 1 CPU core



Up to 20% speed-up with 1-HOP method

526GFlops is achieved with 32GPUs on TSUBAME 1.2

→ x1170 faster than 1 CPU core

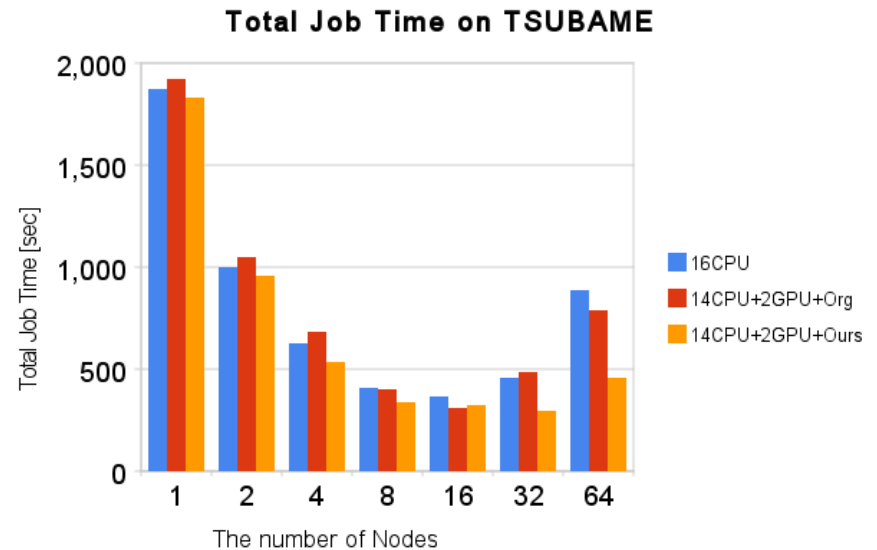
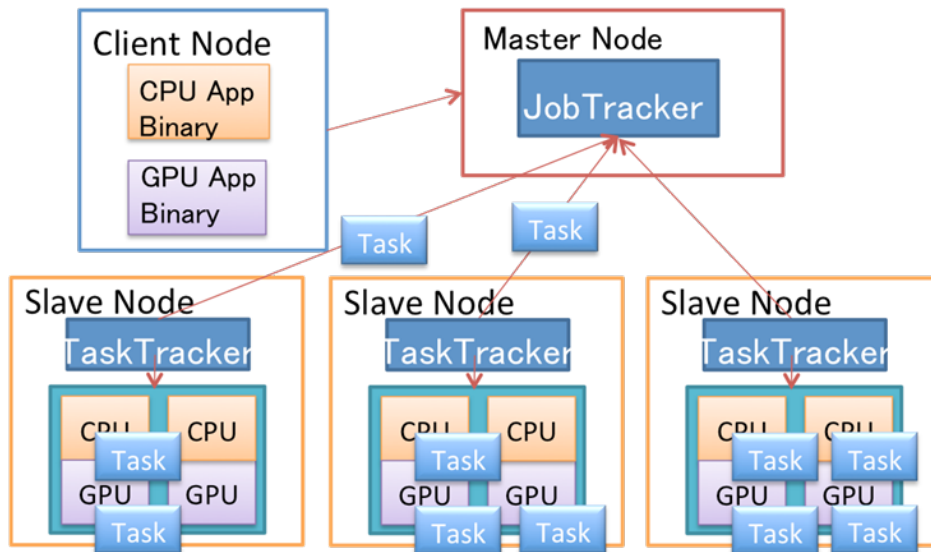
MapReduce

for GPU-based Heterogeneous Clusters

[MAPRED'2010]

Problem :

- Map Task Scheduling for efficient execution
 - Depends on **running task characteristics** and **underlying environments**
- Hybrid Map Task Scheduling
 - Automatically detects map task characteristics by monitoring
 - Scheduling map tasks to minimize overall MapReduce job execution time



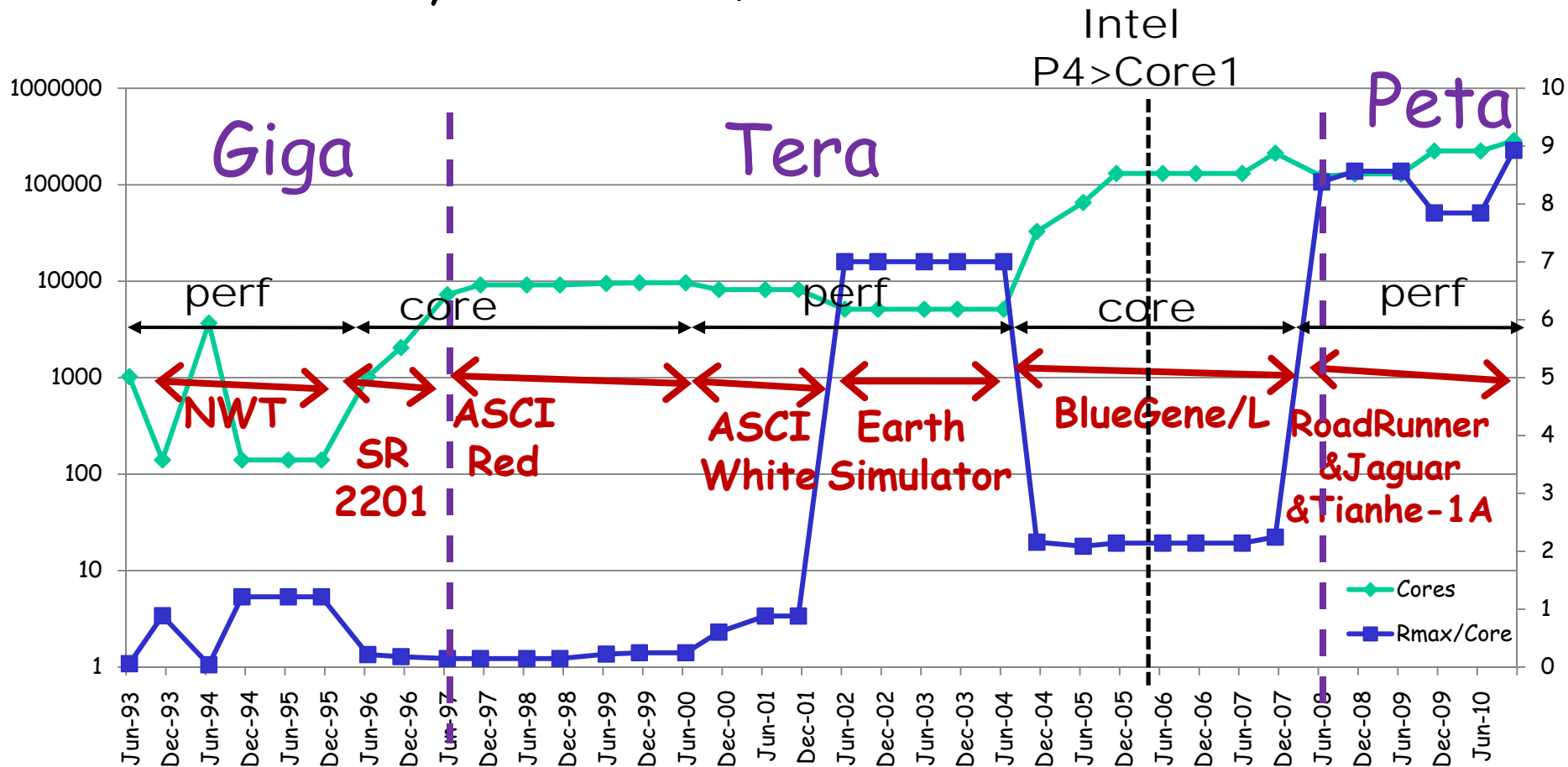
1.93 times faster than the Hadoop original scheduling

DoE Exascale Targets

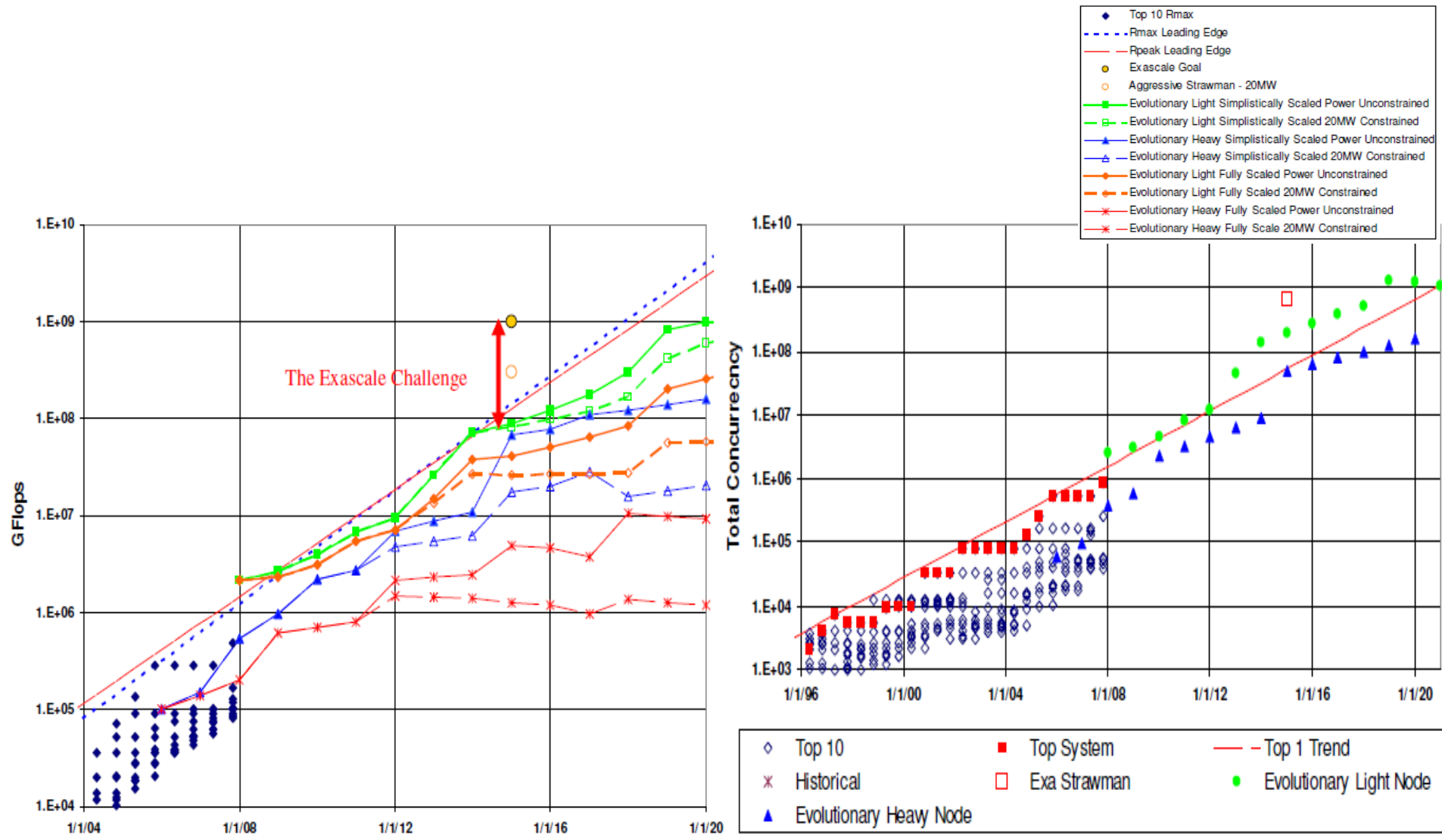
System attributes	“2010”		“2015”		“2018-20”	
System peak	2 PetaFlops		100-200 PetaFlops		1 ExaFlop	
Power	6 MW	1.3 MW	15 MW		20 MW	
System Memory	0.3PB	0.1PB	5 PB		32-64PB	
Node Perf	125GF	1.6TF	0.5TF	7TF	1TF	10TF
Node Mem BW	25GB/s	0.5TB/s	0.1TB/s	1TB/s	0.4TB/s	4TB/s
Node Concurrency	12	O(1000)	O(100)	O(1000)	O(1000)	O(10000)
#Nodes	18,700	1442	50,000	5,000	1 million	100,000
Total Node Interconnect BW	1.5GB/s	8GB/s	20GB/s		200GB/s	
MTTI	O(days)		O(1 day)		O(1 day)	

#Cores & Rmax/Core on #1 Top500

- Alternating core increase vs. perf/core increase
- Next generation (10PF) will mainly be # core increase, and thereafter...



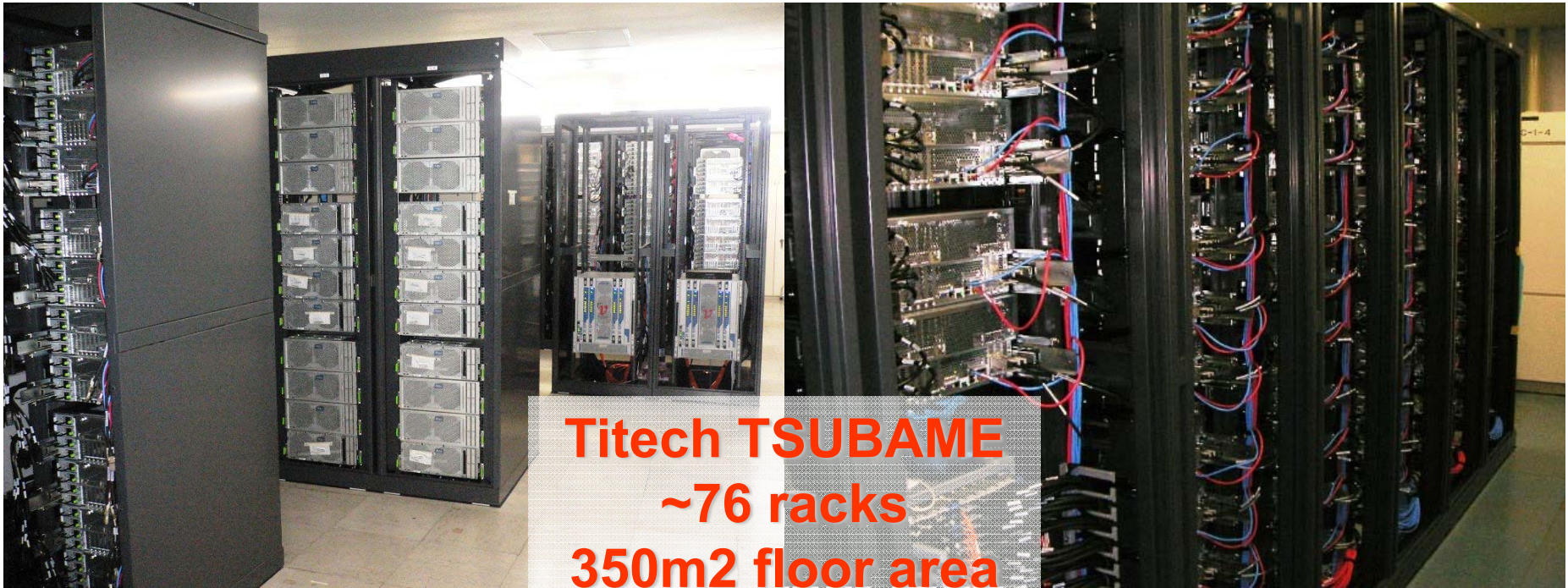
Extreme Many Core, Slow&Parallel is the Key to Low Power [Kogge08]



"Accelerator"



- Special-purpose HW for accelerated computation
 - ▶ Small production, special market, expensive No
 - ▶ Lack of software inheritance over HW generation No
- Restriction of application area
 - ▶ Only accelerates special type/portion of computation No
 - ▶ Computation that "does not fit" impossible or slow No w/ hybrid
- Restriction of programming model
 - ▶ Specialized programming model, language Improving
 - ▶ Restrictions on pointers, recursion, structures, etc. Improving
 - ▶ No OS or other system software No with hybrid



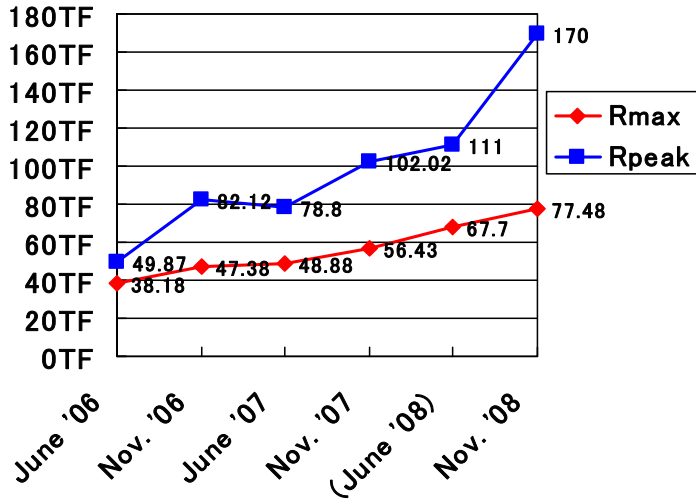
Titech TSUBAME
~76 racks
350m2 floor area
1.2 MW (peak),
PUE=1.44



TSUBAME's Lifetime Achievements

* The Top500 *

Six Consecutive No.1 in Japan and performance improvements
1st hetero arch. to rank in Top 10



* Industry Collab. Programs and Grants *



Ministry of Education
Innovative Industrial
Usage Program



Ministry of Education
Global Center of Excellence
HPC Ph.D. Education Program



Microsoft HPC Institute and
other vendor research collab.



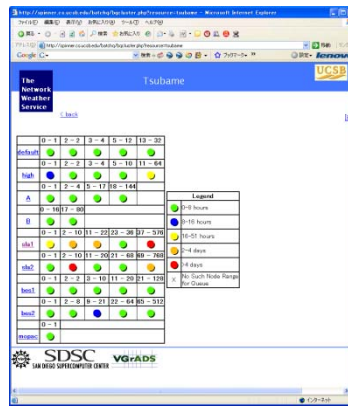
NAREGI/CyberScience
Infrastructure
Core Center

TSUBAME in Nature and on TV

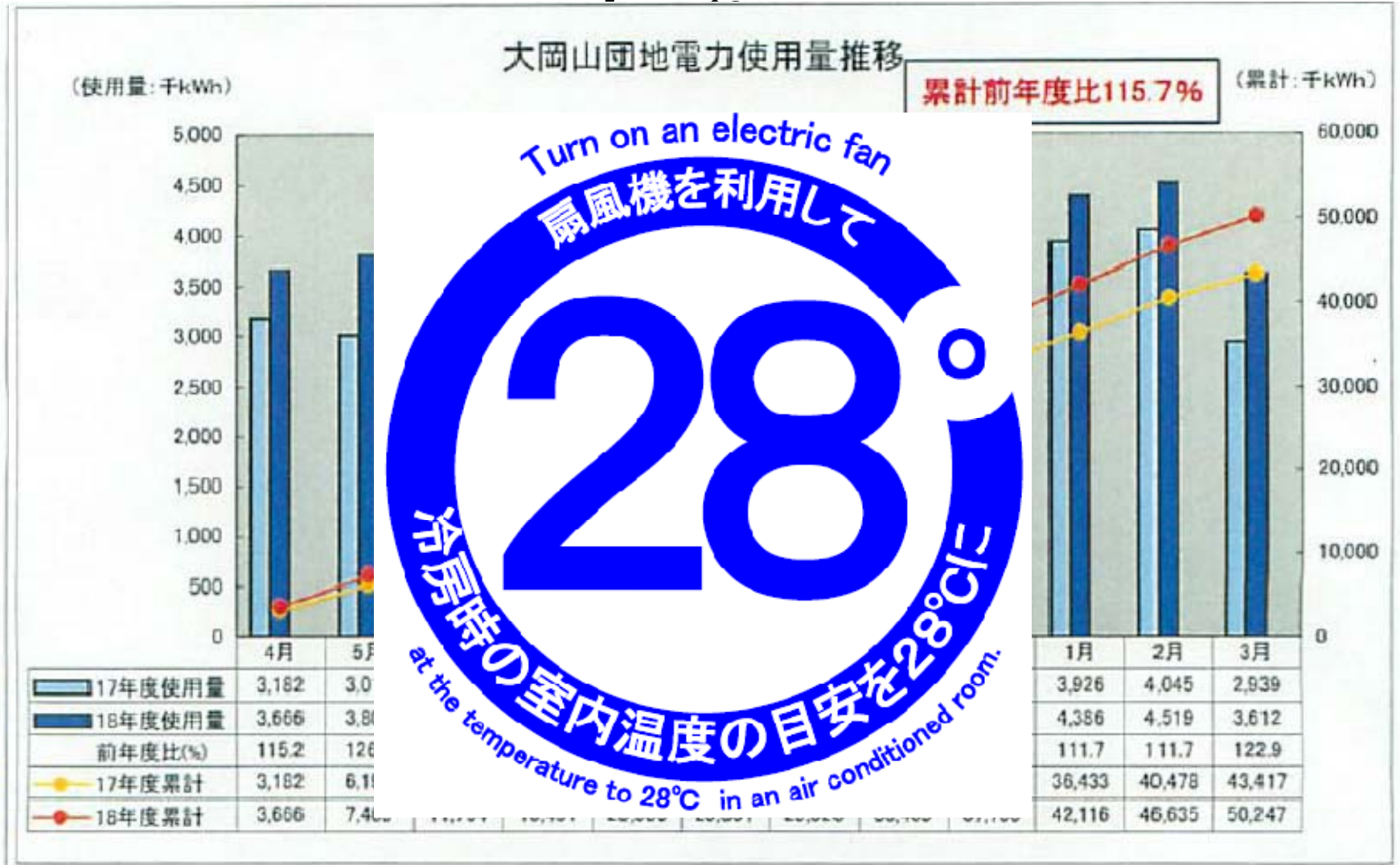


* Supercomputing for Everyone *

- Institution-wide usage: > 1000 users, accounts for undergrads
- New QoS and Market-based Scheduling
- New HPC Services (Grid Services / ASP)
- General Datacenter Server Hosting via VM
- Use very little resources for institutional IT Consolidation



You know you have a problem



Biggest Problem is Power...

Machine	CPU Cores	Watts	Peak GFLOPS	Peak MFLOPS/ Watt	Watts/ CPU Core	Ratio c.f. TSUBAME
TSUBAME(Opteron)	10480	800,000	50,400	63.00	76.34	
TSUBAME2006 (w/360CSs)	11,200	810,000	79,430	98.06	72.32	
TSUBAME2007 (w/648CSs)	11,776	820,000	102,200	124.63	69.63	1.00
Earth Simulator	5120	6,000,000	40,000	6.67	1171.88	0.05
ASCI Purple (LLNL)	12240	6,000,000	77,824	12.97	490.20	0.10
AIST Supercluster (Opteron)	3188	522,240	14400	27.57	163.81	0.22
LLNL BG/L (rack)	2048	25,000	5734.4	229.38	12.21	1.84
Next Gen BG/P (rack)	4096	30,000	16384	546.13	7.32	4.38
TSUBAME 2.0 (2010Q3/4)	160,000	810,000	1,024,000	1264.20	5.06	10.14

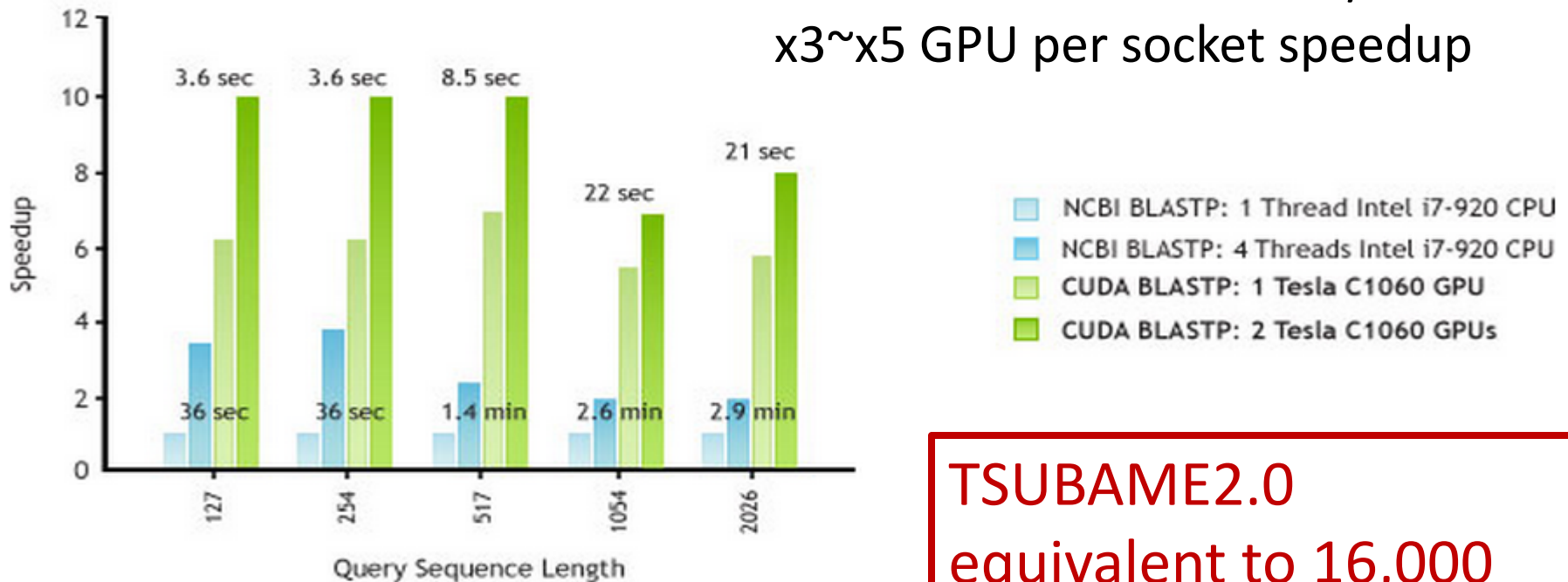
TSUBAME 2.0 x24 improvement in 4.5 years...? → ~ x1000 over 10 years

Bioinformatics: BLAST on GPUs

- CUDA-BLASTP (NTU)

http://www.nvidia.com/object/blastp_on_tesla.html

CUDA-BLASTP vs NCBI BLASTP Speedups



Data Courtesy of Nanyang Technological University, Singapore

Previous-Generation CPU/GPU
x3~x5 GPU per socket speedup

TSUBAME2.0
equivalent to 16,000
sockets (100,000 cores)

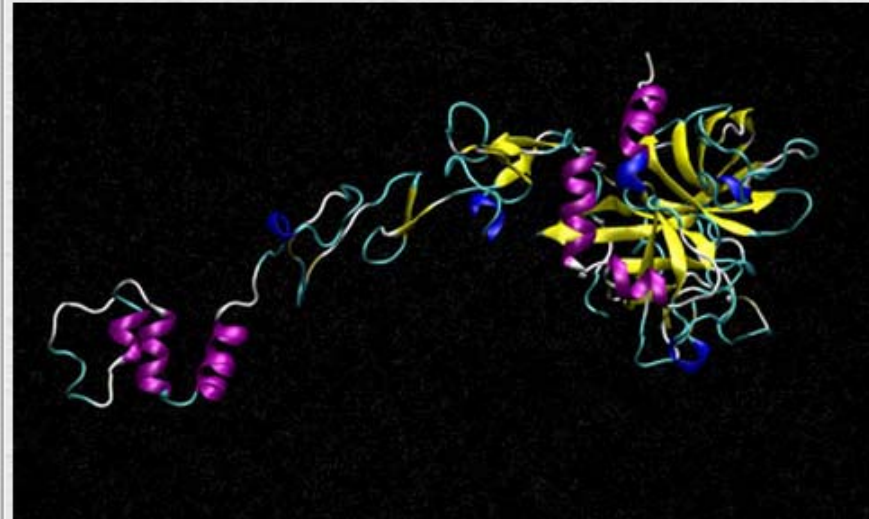
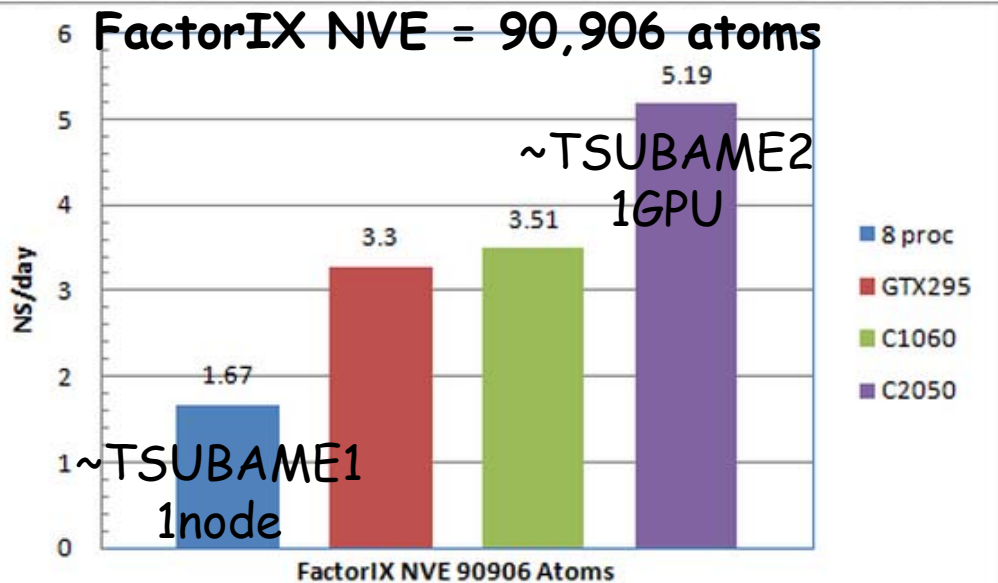
- GPU-BLAST (CMU)

<http://eudoxus.cheme.cmu.edu/gpublast/gpublast.html>

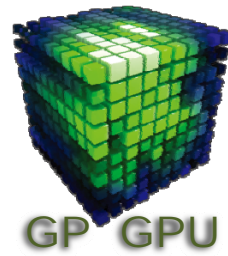
“4 times speedup on Fermi GPUs”

GPU Enabled Amber 11

- 1GPU \approx 8 Core CPU node \times 4~20
 - Already in service on TSUBAME2.0
 - Waiting for Multi-GPU version presented at GTC 2010...
- Equivalent to 300,000 CPU cores on TSUBAME2.0
 - Assuming $\times 10$ speedup per socket



Electric Power Consumption for Multiple-GPU Applications



CFD Applications

Comparing to TSUBAME Opteron CPU 1 core,



MAX 200W/GPU

Weather Prediction: x80 (10 GPU = 1000 CPU)

Lattice Boltzmann: ~ x100

Phase Field Model: x170

When we assume the acceleration is typically x100,
1 GPU < 200W for our applications



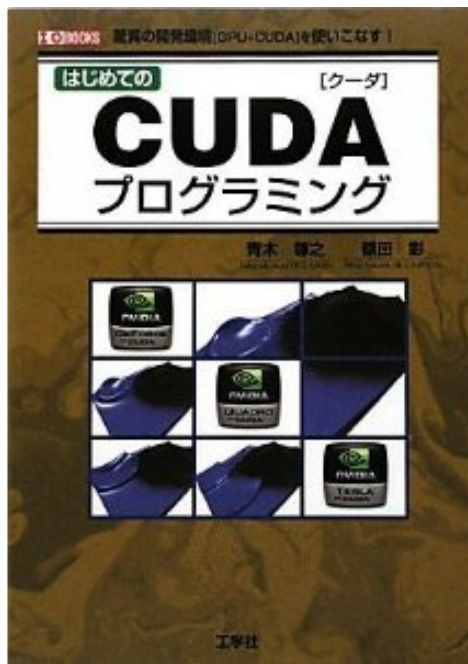
TSUBAME ~ 1MW/10000CPU

100 GPU: 20 kW

10000 CPU core of **TSUBAME 1.2**: 1MW (1000kW)

1/50 Ultra low Power





Aoki and Nukada
 "The First CUDA
 Programming"
 Textbook

情報処理 2

[2009] Vol.50 No.2 通巻528号

特集 アクセラレータ, 再び
 —スパコン化の切り札—

解説 アウトソーシングと情報セキュリティ問題
 —プリント業務のマネージド・サービスを題材として—
 報告 Xen Summit Tokyo(Aisa) 2008レポート
 コラム わが支部の勢力はここにあり 関西支部:関西支部大会1.5倍の研究発表で支部活動の活性化



Matsuoka, Endo,
 Nukada, Aoki et. al.
 Special Issue IPSJ
 Magazine 2009,
 "Accelerators,
 Reborn"



TSUBAME E-Science
 Journal
 (Vol.2 at SC10)

GPU Computing Consortium

- Established Sep. 1, 2009 in GSIC
- 607 Members as of Sep. 2010
- 8 CUDA Lectures (Hands-on)
- 1 International Workshop
- 1 Symposium (Oct., 2010)
- 3 Seminars by World-famous



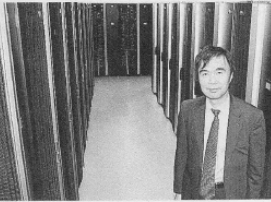
GPUで最速スパコン

Graphics Processing Unit (GPU)は画像処理技術の略。コンピュータの画面に写真や動画を表示する装置。ゲームの3D表示やハイビジョン放送が当たり前になった2000年代後半から急速に高性能が進んだ。最新の機種では切手大の中心に10個



世界の「天河」号は、中国国防科学技術大は、2位の「シャカガ」(米オクラージ)の約1.5倍の計算速度をたたき出した。

天河の部品はほとんどは米国製のTOP500の上位100台のうち4台、米インテル製のCPUと4台、米エヌビディア製のGPUで千個からなる。これまで



GPU

本職は画像処理

3位の星雲(中国深圳スパコンセンター)や4位のツバメ・0(東工大)も構成や部品の

省エネスパコン 日本2位

中国のスーパーコンピュータが計算速度で初めて世界一になったスパコンキング「Oodao」は、中央演算処理装置(CPU)に加え、「GPU」を駆使する画像処理装置を多く積んだスパコンが、上位5台のうち3台を占めた。高速の部材で消費電力も少ないのが特徴だ。大規模化が限界に来ており、GPUや専用演算素子駆使した新世代へ、急速に多様化が進んでいる。(東山正宜)

スパコンは消費電力の問題で大型化が限界に近づいている。ブルージーンやツバメは従来の演算装置だけに頼らない新世代のスパコンだ。(東山正宜、ニューオーリンズ小宮山亮磨)

ランキングは「グリーン500」で消費電力当たりの計算速度を競う。1号当たりの計算速度は、ブルージーンが毎秒16億8400万回、ツバメが9億5800万回。ツバメは、計算速度を競う世界ランキング「TOP500」では4位だった。京は現在、全体の0.5%しかできていないが、高性能が示された。

スパコン省エネ性能

東工大、世界2位

米バージニア工科大学が「SUBAME2.0」は米IBM製のスパコンだが世界2位となった。日本が、同機種は試作機の扱いが4位、国立環境研究所のスパコンは10位内に3つで、実際に運用しているスパコンが10位に入った。

東京工業大学のスパコン「SUBAME2.0」は米IBM製のスパコンだが世界2位となった。日本が、同機種は試作機の扱いが4位、国立環境研究所のスパコンは10位内に3つで、実際に運用しているスパコンが10位に入った。

朝日20101119(夕刊)

日経20101121

で高速計算

開発費も安く

ほとんど同じ。3台の計算速度の違いはCPUとGPUの数の違いを反映している。東工大の松岡聡教授は「性能を増やしても、使いこなし性を増やすのは簡単ではない。中国もなかなかやるようになってきた」と話す。

市販品を使っているのが開発費を安くしている。天河は約80億円、ツバメは30億円だった。2002年には世界一をとった地球シミュレータが600億円、理化学研究所が建設している「京」が、建屋も含めて1100億円かかるのに対して、倍以上安い。

クラウドで普及、省エネに貢献

GPUの本来の仕事である画像処理では、簡単な計算を数多くこなす必要がある。このため、GPUには単純な計算を担う小さな演算素子がたくさん詰め込まれている。一方、CPUは複雑な計算をこなすため、そのために対応した大規模な素子を積み重ねる。二つを使い分け、計算をうまく割り振れば効果は絶大だ。

計算速度を競う「TOP500」を並べた注目されるランキングが18日に発表される。Oodaoは5000台、TOP500には選ばれたスパコン5000台が消費電力あたりの計算速度を競う。2007年に始まった、松岡聡が中心で、世界の電力の1割程度がIT、情報技術の消費で使われ、そのうちパソコンの消費電力は4割程度に達している。クラウド化が進めばITの電力消費が減ると、松岡教授は言う。

世界のスパコンの「計算力総計」

順位	名称	計算速度
1	天河1号	2566
2	シャカガ	1759
3	星雲	1271
4	ツバメ2.0	1192
5	ホバ	1054
6	X900	191
7	地球シミュレータ	122
8	京(開発中)	48
169	TOP500コンテスト	

2010年11月現在
2010年11月現在

中国のスパコン

初め世界一になった。上位を独占してきた米国の地位は、逆のくぼかした。

計算速度世界一

米、技術力を注目

世界のスパコンの性能を測る研究者組織「TOP500」が16日、米ニューオーリンズで開いた発表式で、頂点に立った「天河1号」を表彰した。中国国防科学技術大の劉光明教授が登壇し、天河1号は中国の科学、技術、経済の発展を加速する一助となった。

計算速度を合計した計算力総計では、中国は世界全体の13%を占めて1位になった。

米国防省は16日、天河1号の性能を高く評価した。米国防省は、天河1号の性能を高く評価した。米国防省は、天河1号の性能を高く評価した。

日本、ハ

世界一を奪われた。米国防省は、天河1号の性能を高く評価した。米国防省は、天河1号の性能を高く評価した。



Supercomputing
2010
@ New Orleans
東工大ブース

