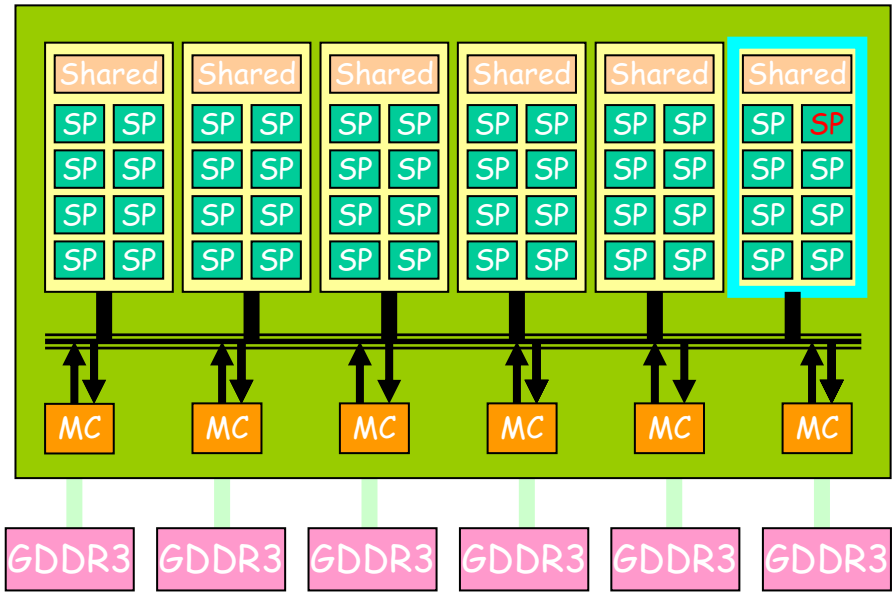
GPU Acceleration: A Fad or the Yellow Brick Road onto Exascale?

Satoshi Matsuoka, Professor/Dr.Sci.

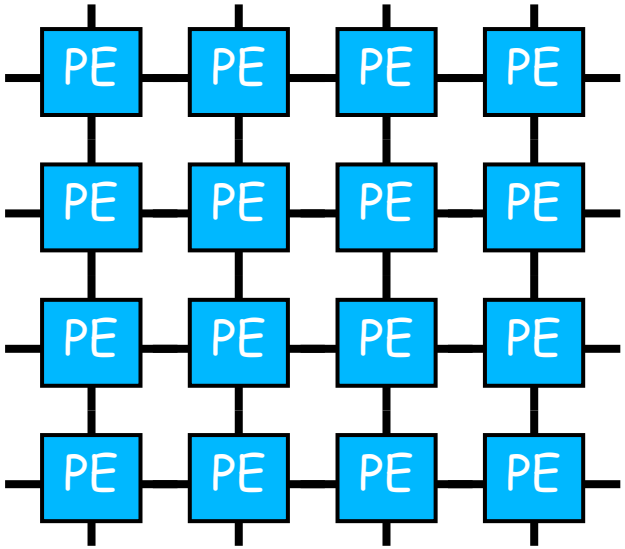
Global Scientific Information and Computing Center (GSIC)
Tokyo Inst. Technology
/ National Institute of Informatics, Japan

CNRS ORAP Forum, 2010 Mar 31, Paris, France

GPU vs. Standard Many Cores?

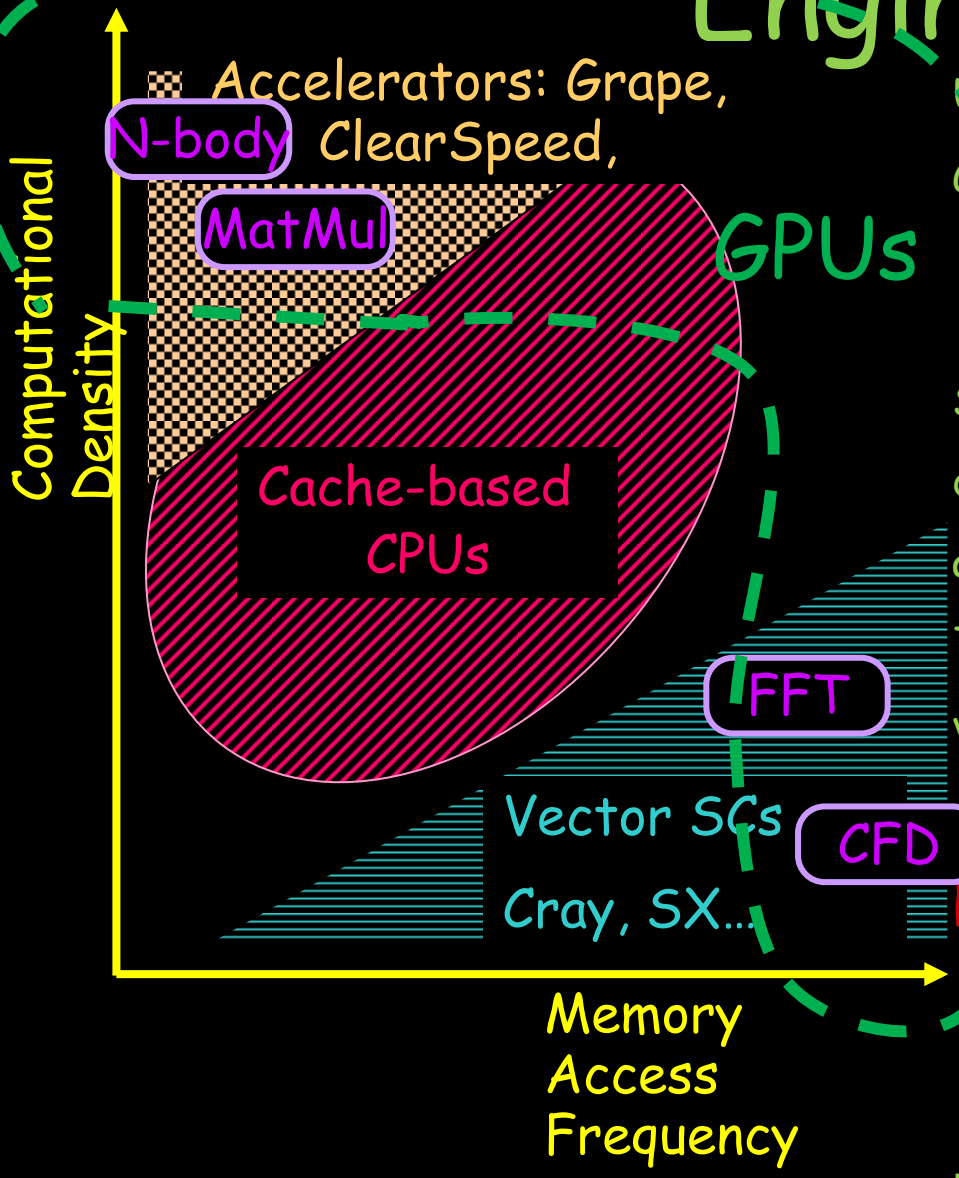


vs.



- Are these just legacy programming compatibility differences?

GPUs as Commodity Vector Engines



Unlike the conventional accelerators, GPUs have high memory bandwidth and dense

Since latest high-end GPUs support double precision, GPUs also work as commodity vector processors.
The target application area for GPUs is very wide.

Restrictions: Limited non-stream memory access, PCI-express overhead, etc.

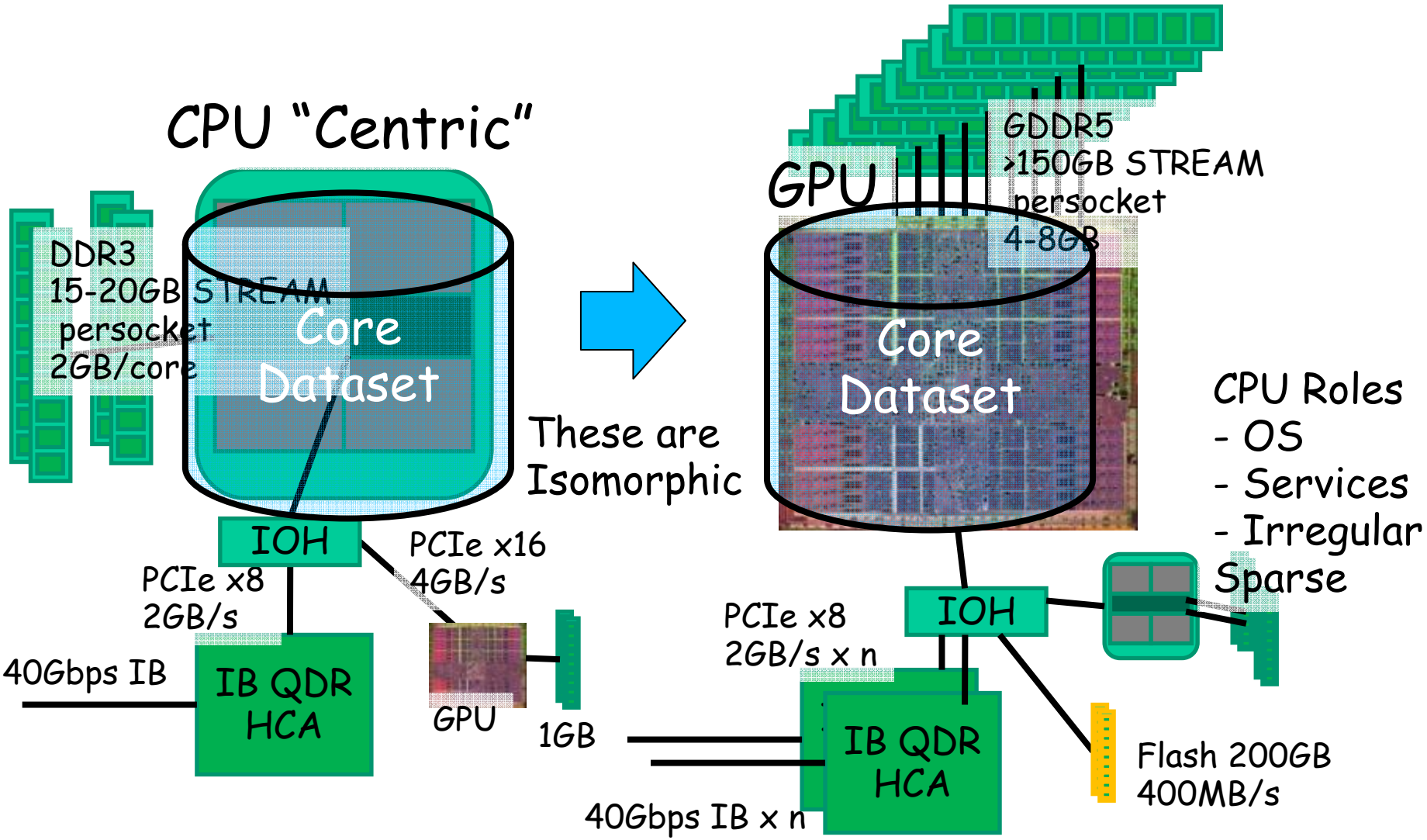
➔ How do we utilize them easily?

GPUs as Commodity Massively Parallel Vector Processors

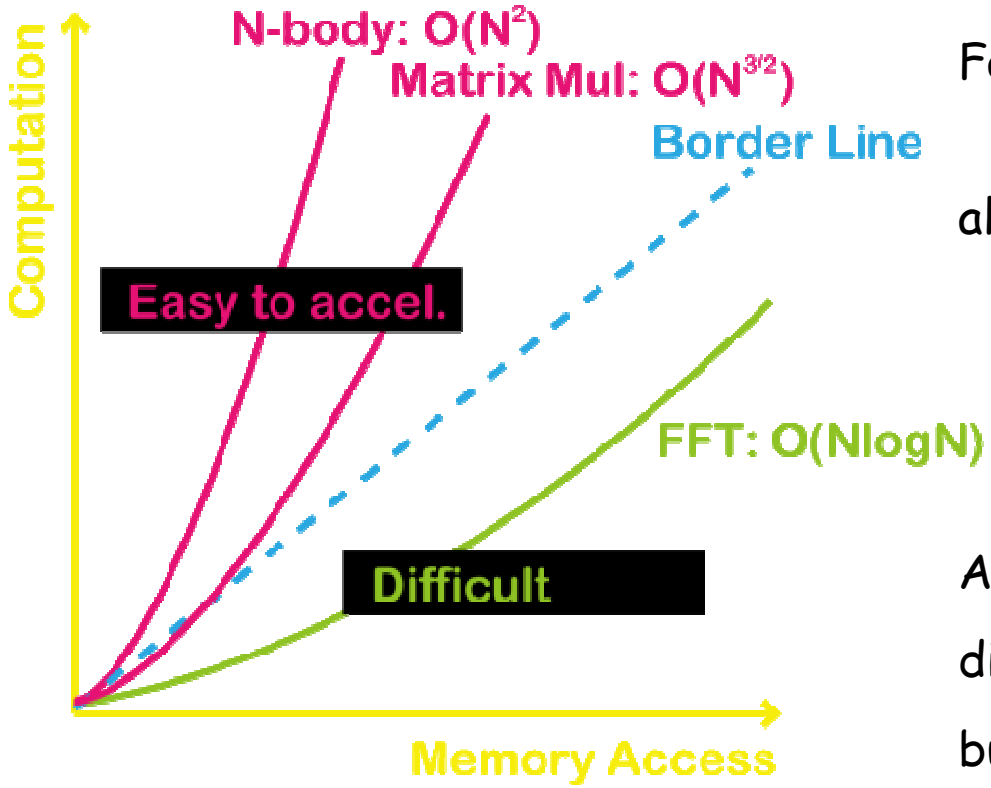
- E.g., NVIDIA Tesla, AMD Firestream
  - High Peak Performance > 1TFlops
    - Good for tightly coupled code e.g. Nbody
  - **High Memory bandwidth (>100GB/s)**
    - **Good for sparse codes e.g. CFD**
  - **Low latency over shared memory**
    - **Thousands threads hide latency w/zero overhead**
  - **Slow and Parallel and Efficient vector engines for HPC**
  - Restrictions: Limited non-stream memory access, PCI-express overhead, programming model etc.

How do we exploit them given vector computing experiences?

From CPU Centric to GPU Centric Nodes for Scaling



High Performance 3-D FFT on NVIDIA CUDA GPUs [Nukada et. al. SC08]

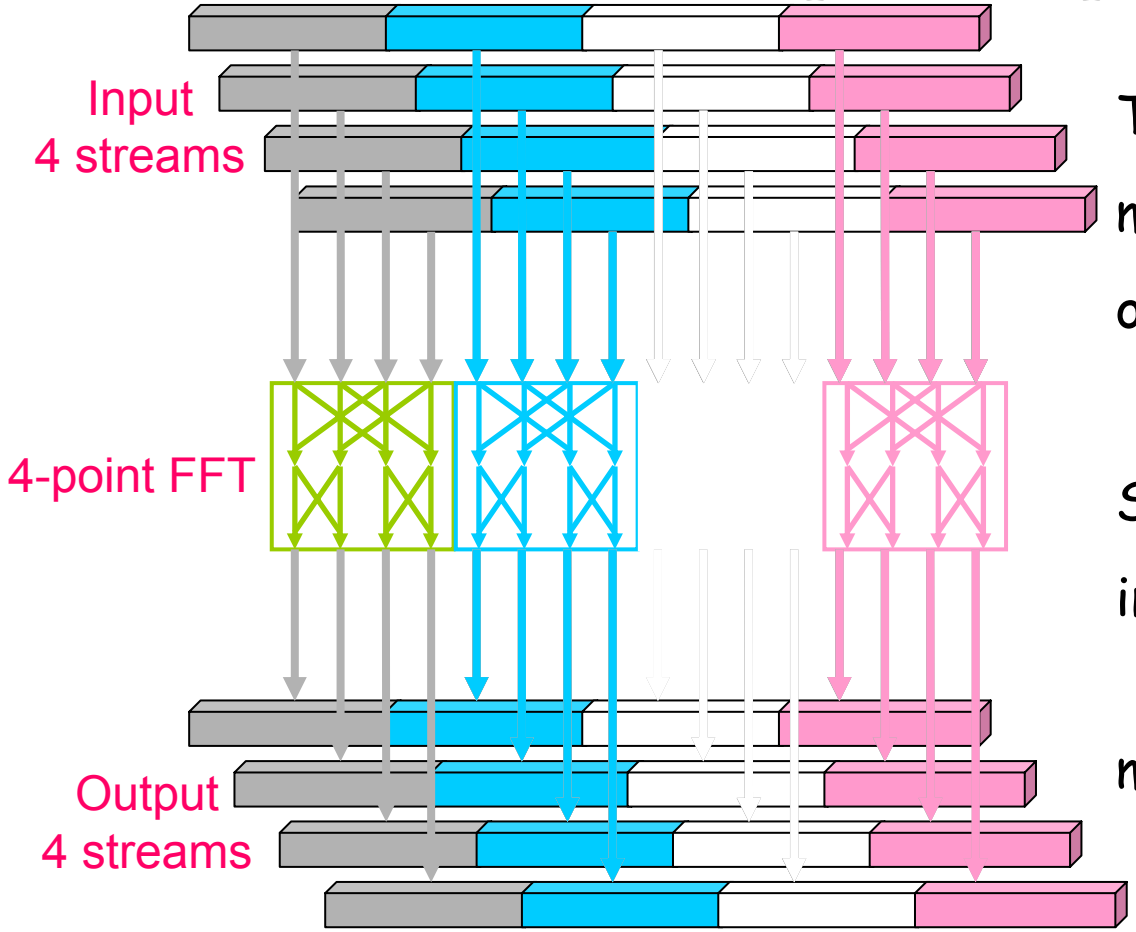


Fast Fourier Transform is an O(NlogN) algorithm with O(N) memory access.

Acceleration of FFT is much more difficult than N-body, MatMul, etc., but possible!

At least, GPUs' theoretical memory bandwidth is much higher than CPUs.

multi-row FFT algorithm for GPU [SC08]

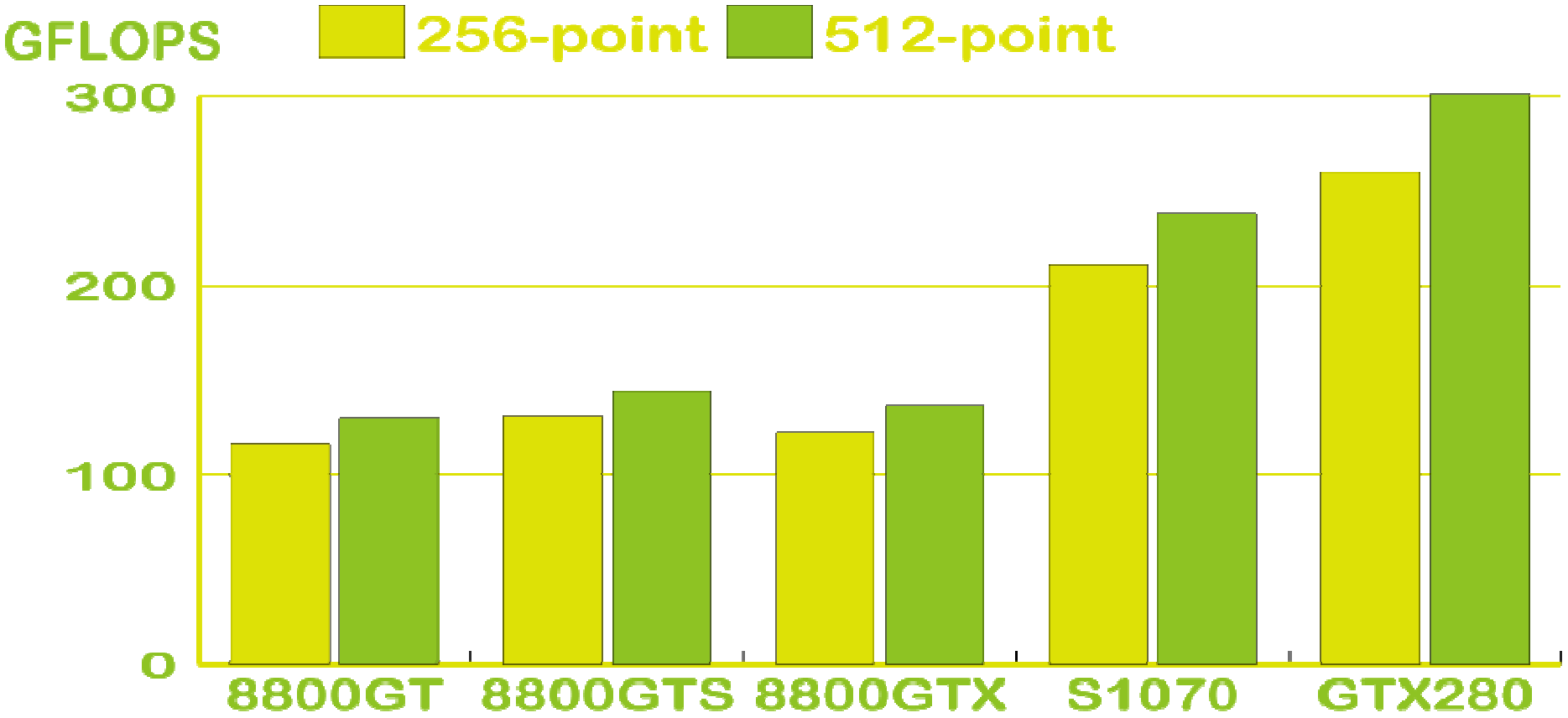


This algorithm accesses multiple streams, but each of them is successive.

Since each thread compute independent set of small FFT,
many registers are required.

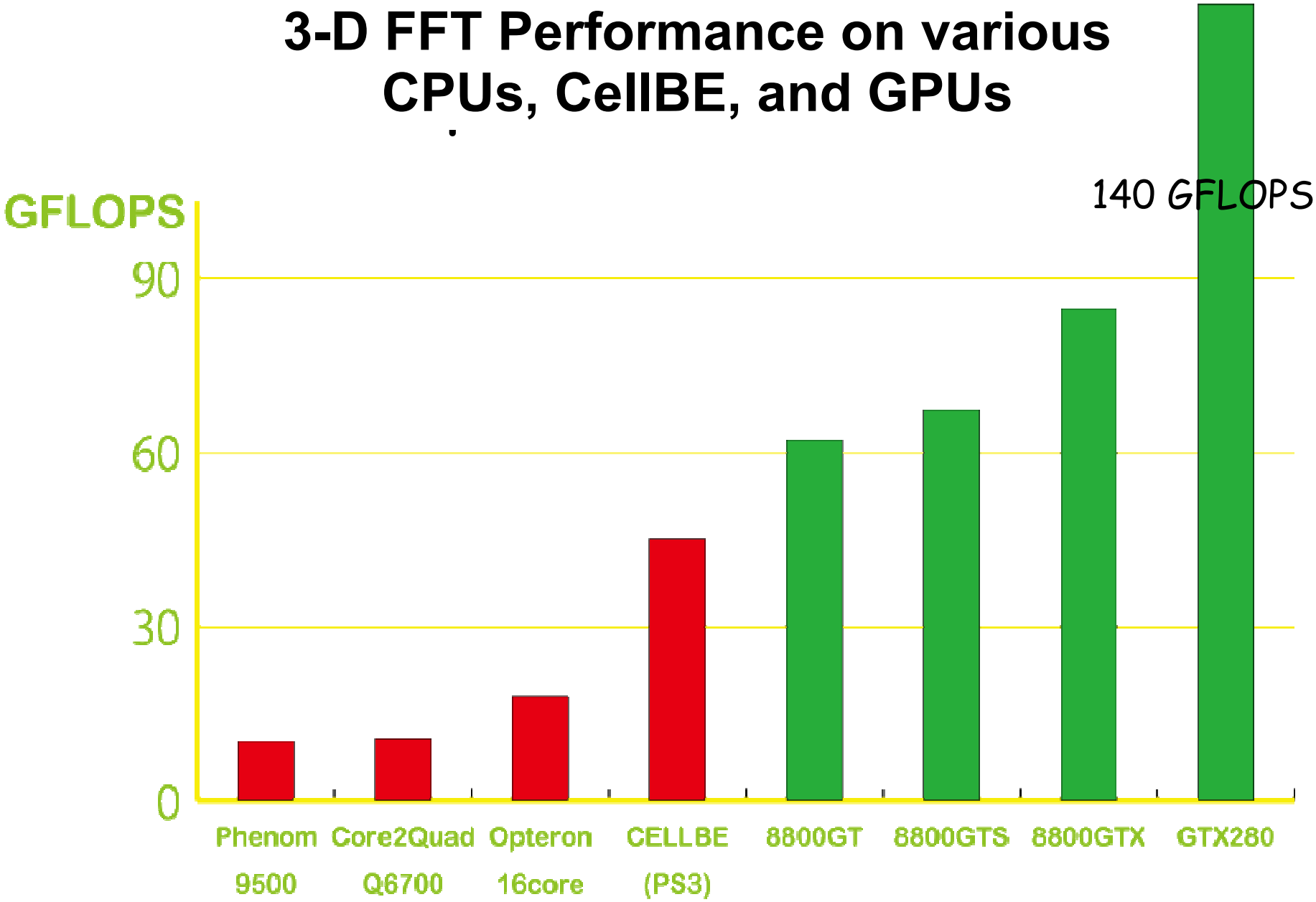
For 256-point FFT, use two-pass 16-point FFT kernels.

Performance of 1-D FFT



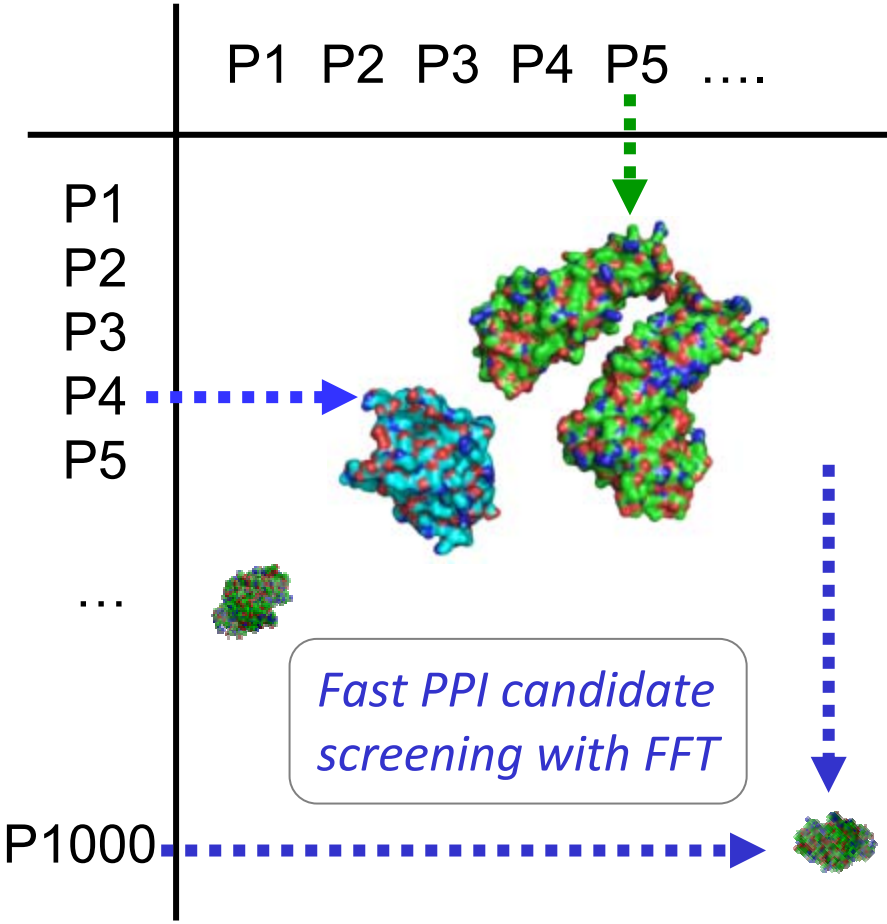
Note: An earlier sample with 1.3GHz is used for Tesla S1070.

3-D FFT Performance on various CPUs, CellBE, and GPUs



All-to-all 3-D Protein Docking Challenge

(Collaboration with Yutaka Akiyama, Tokyo Tech.)



1,000 x1,000 all-to-all docking fitness evaluation will take only

1-2 months (15 deg. pitch)

with a 32-node HPC-GPGPU cluster (128 GPGPUs).

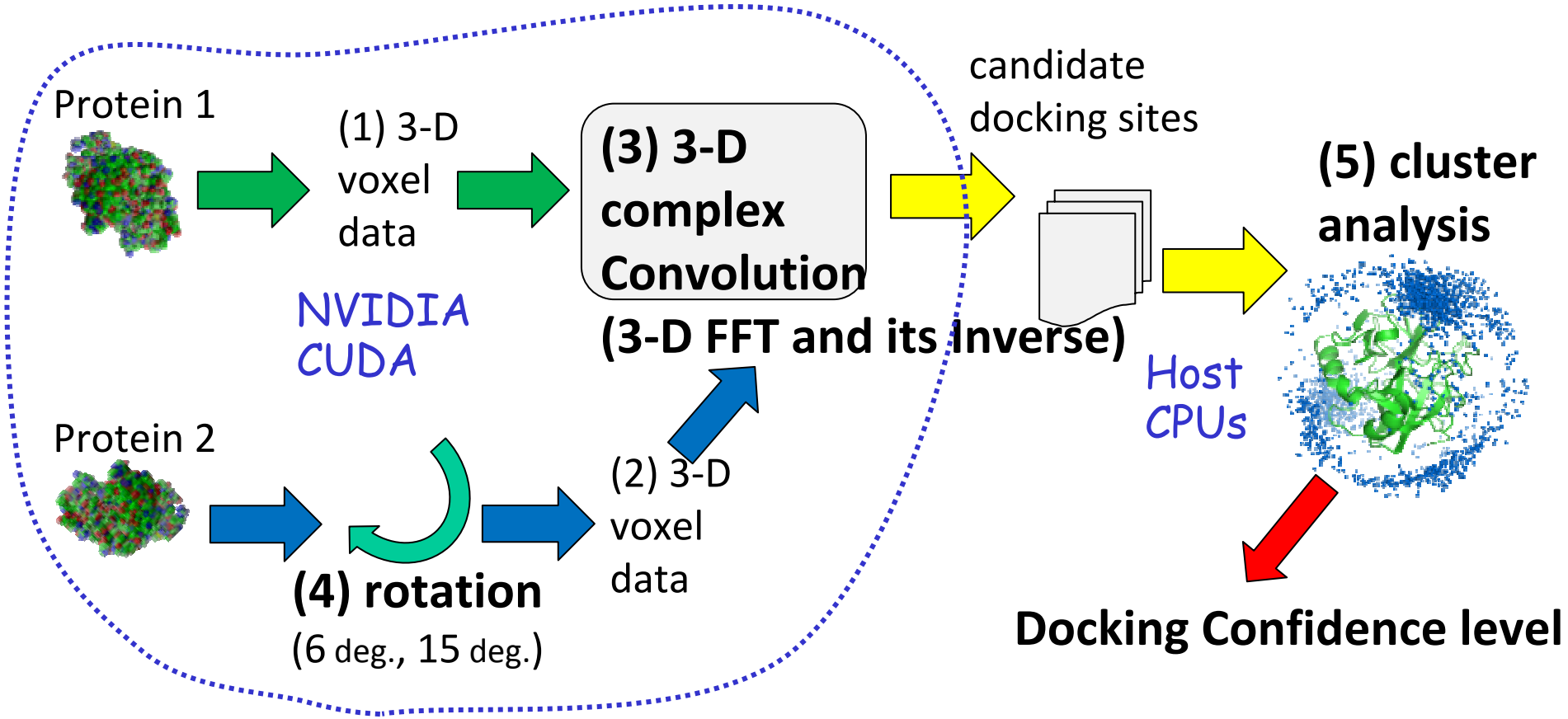
cf.

~ 500 years with single CPU (sus. 1+GF)

~ 2 years with 1-rack BlueGene/L

Blue Protein system
CBRC, AIST
(4 rack, 8192 nodes)

Algorithm for 3-D All-to-All Protein Docking



Calculation for **a single protein-protein pair**: **~= 200 Tera ops.**

3-D complex convolution $O(N^3 \log N)$, *typically* $N = 256$

x

Possible rotations $R = 54{,}000$ *(6 deg. pitch)*

200 Exa Ops for 1000 x 1000

Heavily GPU Acclerated Windows HPC Prototype Cluster

- 32 compute nodes
- 128 NVIDIA 8800GTS
- one head node.
- Gigabit Ethernet network
- Three 40U rack cabinets.
- Windows Compute Cluster Server 2008
- Visual Studio 2005 SP1
- nVidia CUDA 2.x

Performance Estimation of 3D PPD

Single Node

	Power (W)	Peak (GFLOPS)	3D-FFT (GFLOPS)	Docking (GFLOPS)	Nodes per 40 U rack
Blue Gene/L	20	5.6	-	1.8	1024
TSUBAME	1000 (est.)	76.8 (DP)	18.8 (DP)	26.7 (DP)	10
8800 GTS *4	570	1664	256	207	8~13

System Total

! Only CPUs for TSUBAME. DP=double precision.

	# of nodes	Power (kW)	Peak (TFLOPS)	Docking (TFLOPS)	MFLOPS/W
Blue Gene/L (Blue Protein@AIST,Japan)	4096 (4racks)	80	22.9	7.0	87.5
TSUBAME. (Opteron Only)	655 (~70 racks)	~700	50.3 (DP)	17.5 (DP)	25
GPU Accel .WinHPC	32 (4racks)	18	53.2	6.5	361

Can compute 1000x1000 in 1 month (15 deg.) or 1 year (6 deg.)

On full TSUBAME 1.2, ~100TFlops (1MW)-> ~52 rack BG/L

Tokyo Tech. TSUBAME 1.0 Production SC Cluster
Spring 2006

Unified IB network

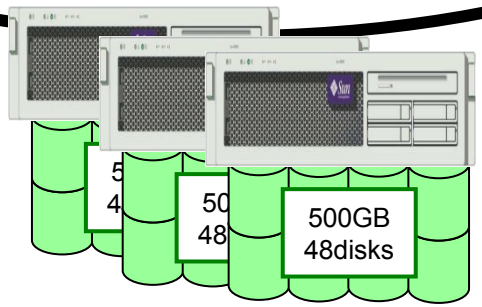
Voltaire ISR9288 Infiniband 10Gbps
x2 (DDR next ver.)
~1310+50 Ports
~13.5Terabits/s (3Tbits bisection)

10Gbps+External Network

7th on the 27th
Top500@38.18TF
#1 SC in Asia 27-29th

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

NEC SX-8i
(for porting)



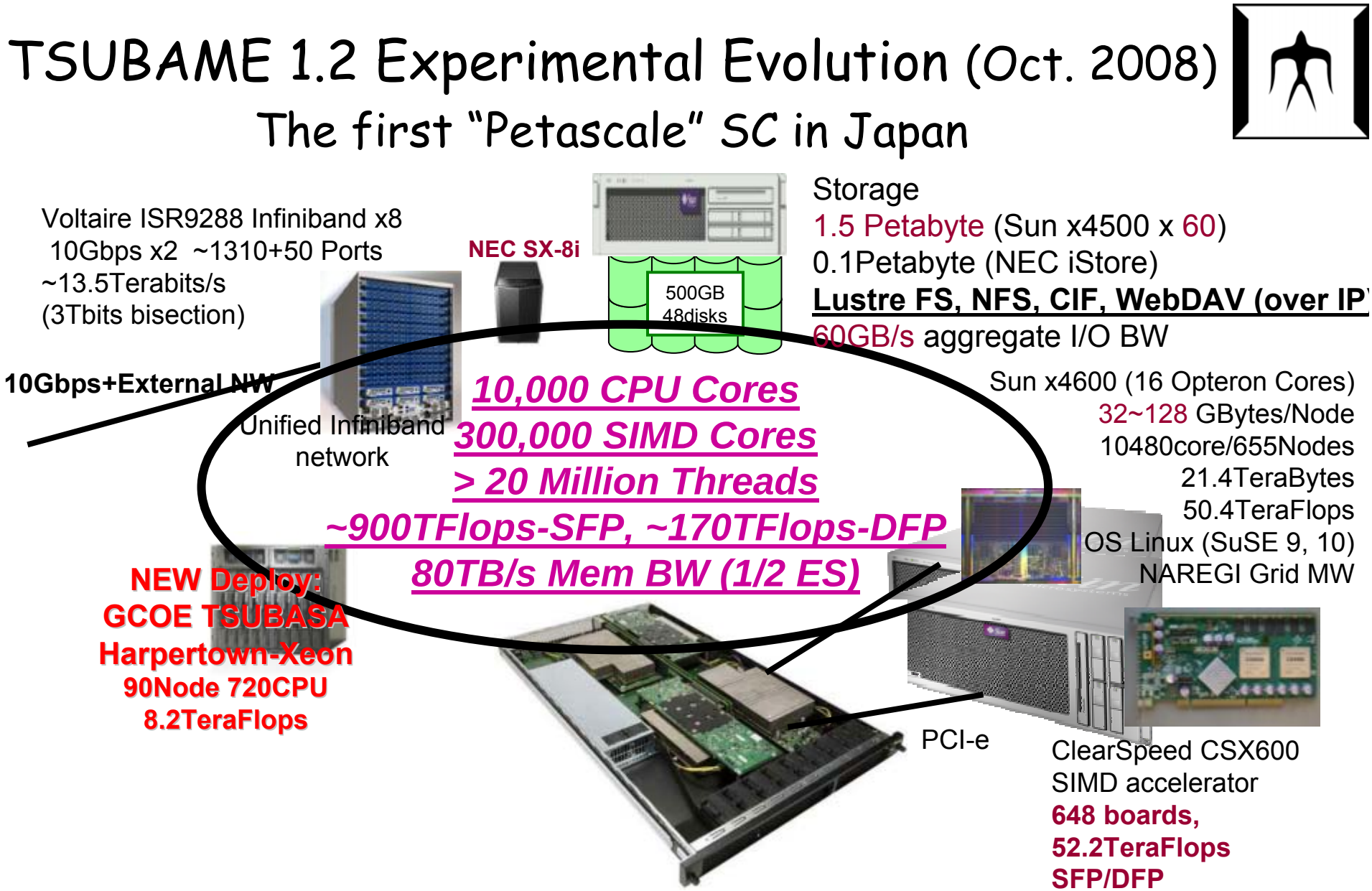
~2000 Users

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)

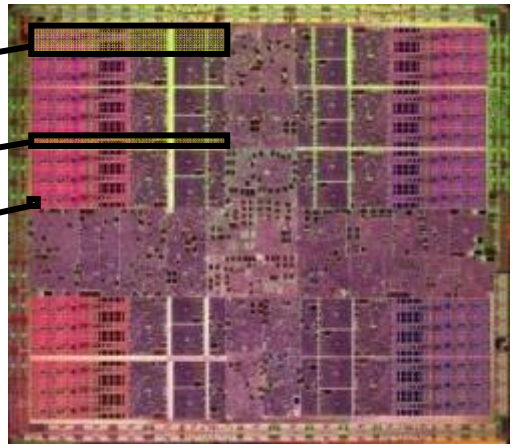
The first "Petascale" SC in Japan



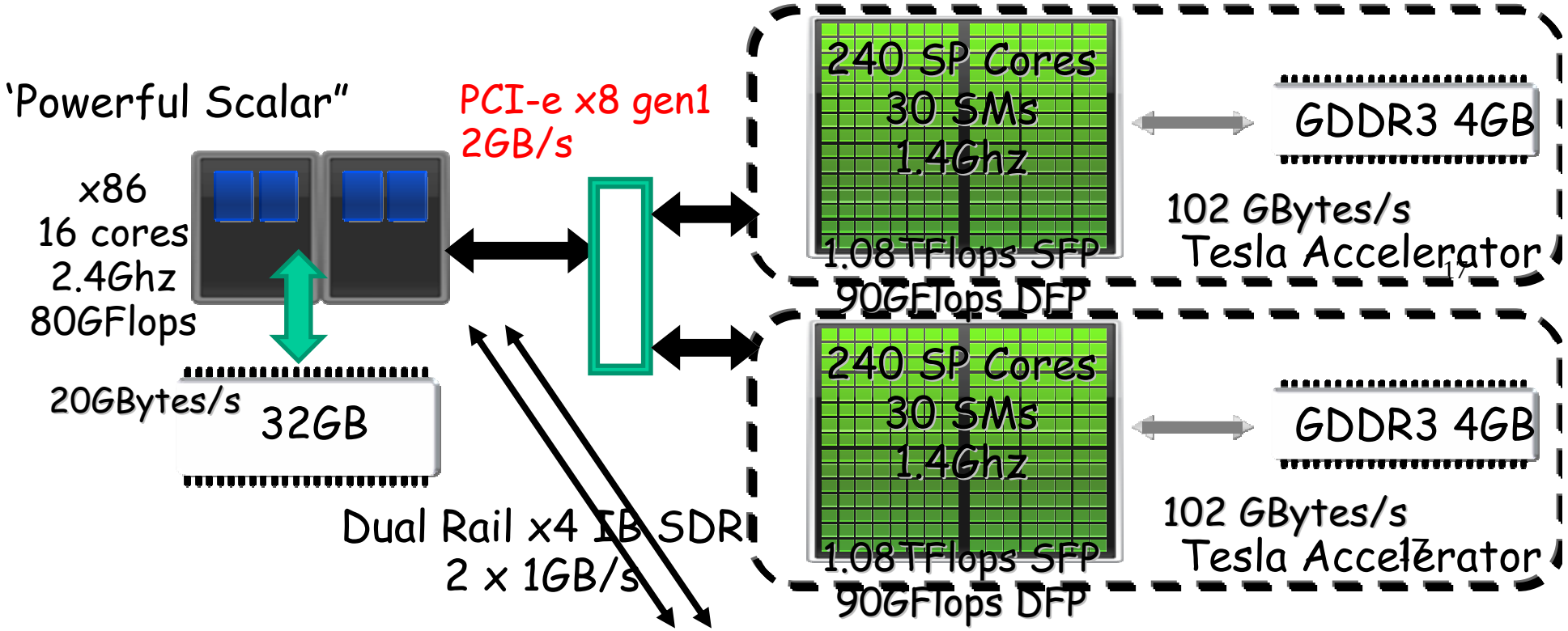
170 Nvidia Tesla 1070, ~680 Tesla cards
High Performance in Many BW-Intensive Apps
10% power increase over TSUBAME 1.0


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

TSUBAME 1.2 Node Configuration

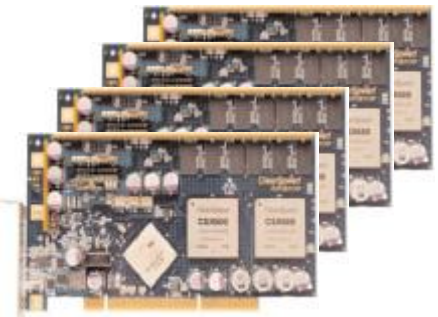


nVidia Tesla T10: 55nm, 470m2,
1.4billion transistors
>1TF SFP
90GF DFP



Linpack on TSUBAME---Heterogeneity at Work [Endo et.al. IPDPS08, IPDPS10.]

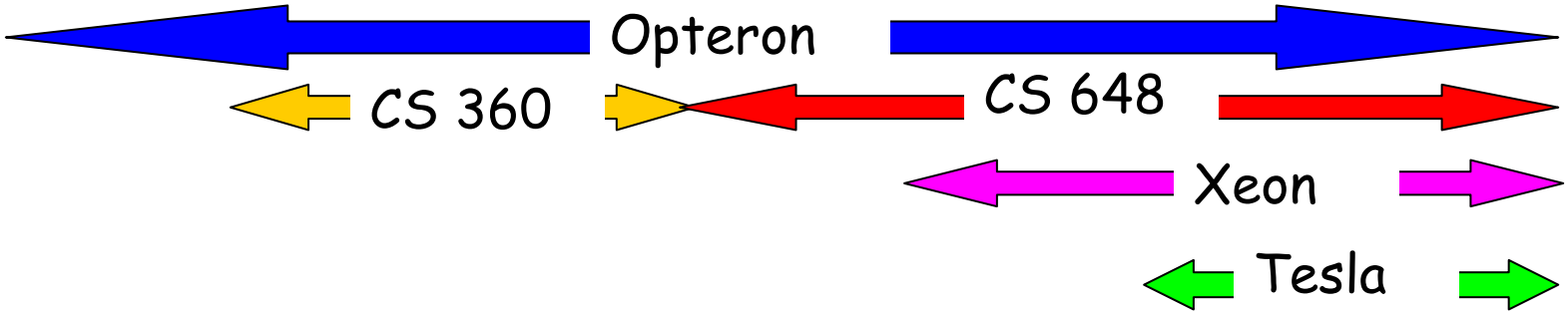
- Linpack implementation that uses cooperatively:
  - 10,368 Opteron cores
  - 612 Tesla GPUs
  - 648 ClearSpeed accelerators
  - 640 Xeon cores (another cluster named "tsubasa")
- Different strategy than on Roadrunner is required
- 87.01TFlops: The world's highest Linpack performance on GPU-accelerated clusters



TSUBAME in Top500 Ranking

[Endo et.al. IEEE IPDPS08, IPDPS10 etc.]

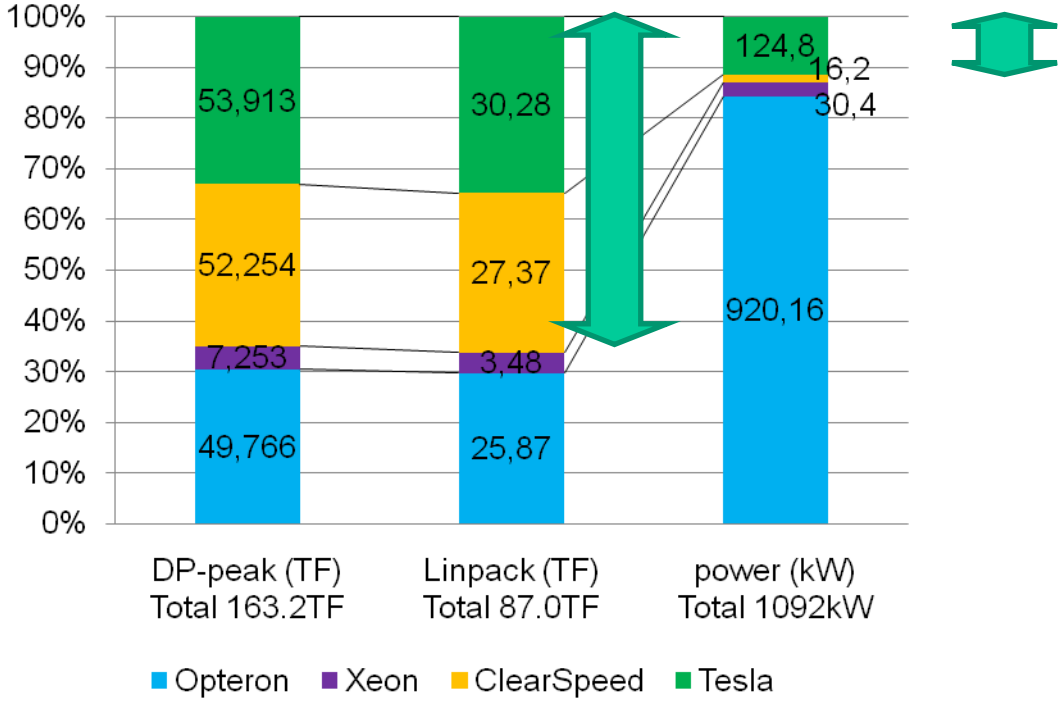
	Jun 06	Nov 06	Jun 07	Nov 07	Jun 08	Nov 08	Jun09
Rmax (Tflops)	38.18	47.38	48.88 [IPDPS 2008]	56.43	67.70	77.48	87.01 [IPDPS2 010]
Rank	7	9	14	16	24	29	41



- Continuous improvement for 6 times
- The 2nd fastest heterogeneous supercomputer in the world (No.1 is RoadRunner)

Electric Power Consumption

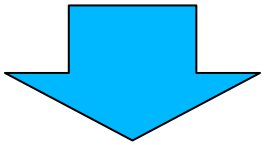
- About 1090kW during Linpack
  - An estimated value from sampling
  - Cooling, network are excluded



Even for dense problems GPUs are effective low power vector-like engines

Higher performance with much lower memory footprint

OK, so we put GPUs into commodity servers and slap them together with cheap networks and we are "supercomputing"



No, since (strong) scaling becomes THE problem (!)

Multi-GPU in CFD: Riken Himeno Benchmark

(Joint Work w/NEC)

RIKEN Himeno CFD Benchmark

Poisson Equation: (Generalized coordinate)

$$\nabla \cdot (\nabla p) = \rho$$

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2} + \alpha \frac{\partial^2 p}{\partial xy} + \beta \frac{\partial^2 p}{\partial xz} + \gamma \frac{\partial^2 p}{\partial yz} = \rho$$

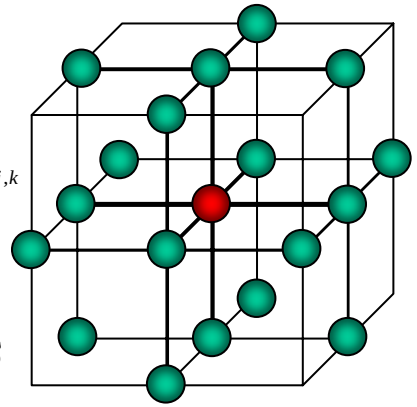
Discretized Form:

$$\frac{p_{i+1,j,k} - 2p_{i,j,k} + p_{i-1,j,k}}{\Delta x^2} + \frac{p_{i,j+1,k} - 2p_{i,j,k} + p_{i,j-1,k}}{\Delta y^2} + \frac{p_{i,j,k+1} - 2p_{i,j,k} + p_{i,j,k-1}}{\Delta y^2}$$

$$+ \alpha \frac{p_{i+1,j+1,k} - p_{i-1,j+1,k} - p_{i+1,j-1,k} + p_{i-1,j-1,k}}{4\Delta x \Delta y}$$

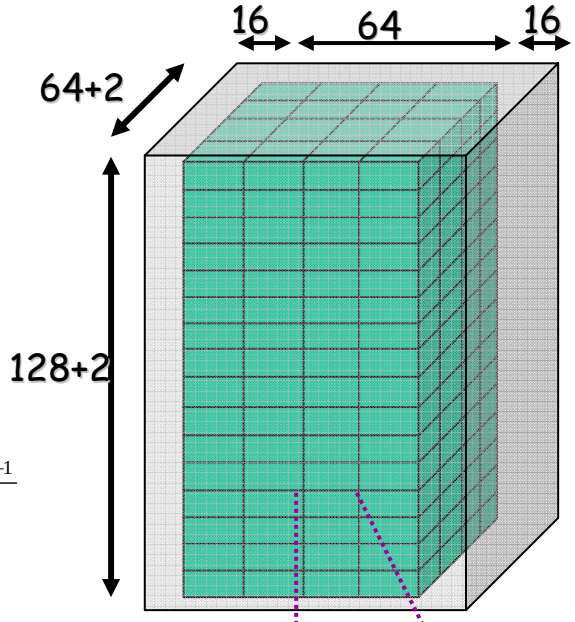
$$+ \beta \frac{p_{i+1,j,k+1} - p_{i-1,j,k+1} - p_{i+1,j-1,k} + p_{i-1,j,k-1}}{4\Delta x \Delta z}$$

$$+ \gamma \frac{p_{i,j+1,k+1} - p_{i,j+1,k-1} - p_{i,j-1,k-1} + p_{i,j-1,k-1}}{4\Delta y \Delta z} = \rho_{i,j,k}$$



18 neighbor point access

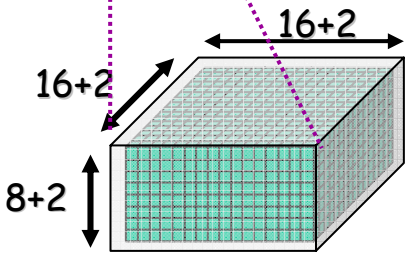
Himeno for CUDA



1 block = 16x16x8 compute region

Block has 256 thread

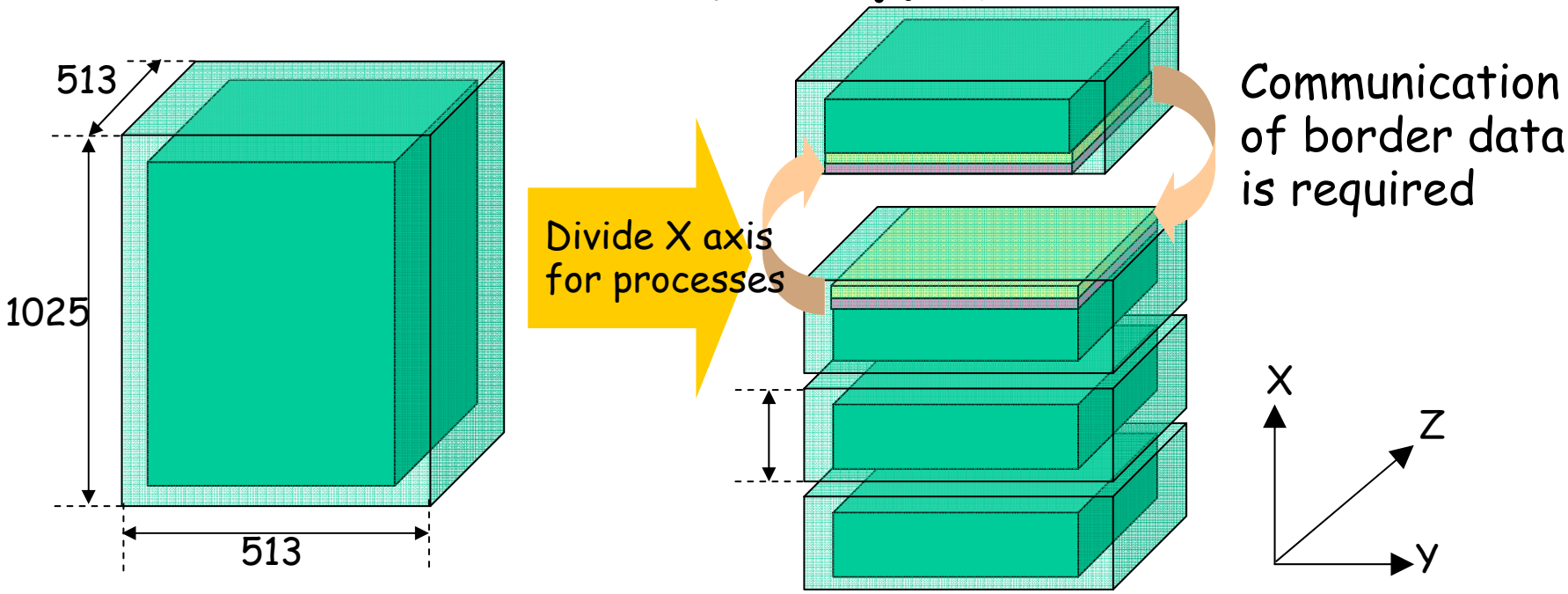
Total 256 blocks = 65536 threads



Block shared mem =16kB

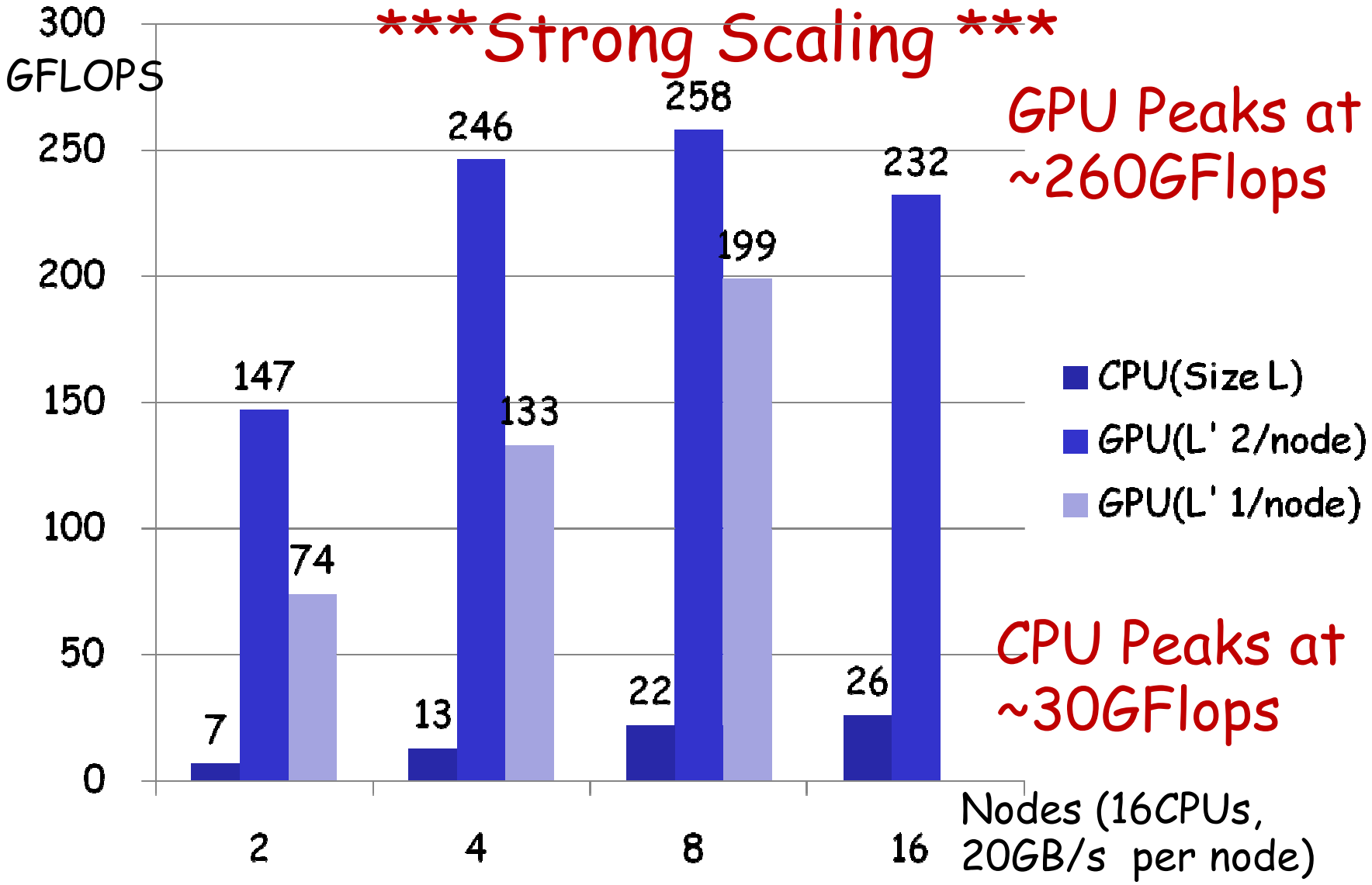
Boundary region used for transfer

Parallelization of Himeno Benchmark



- Although original Himeno supports 3D division, our GPU version currently supports only 1D division

Himeno Size L/L' (257x257x513)
CPU vs. GPU Scaling on TSUBAME 1.2



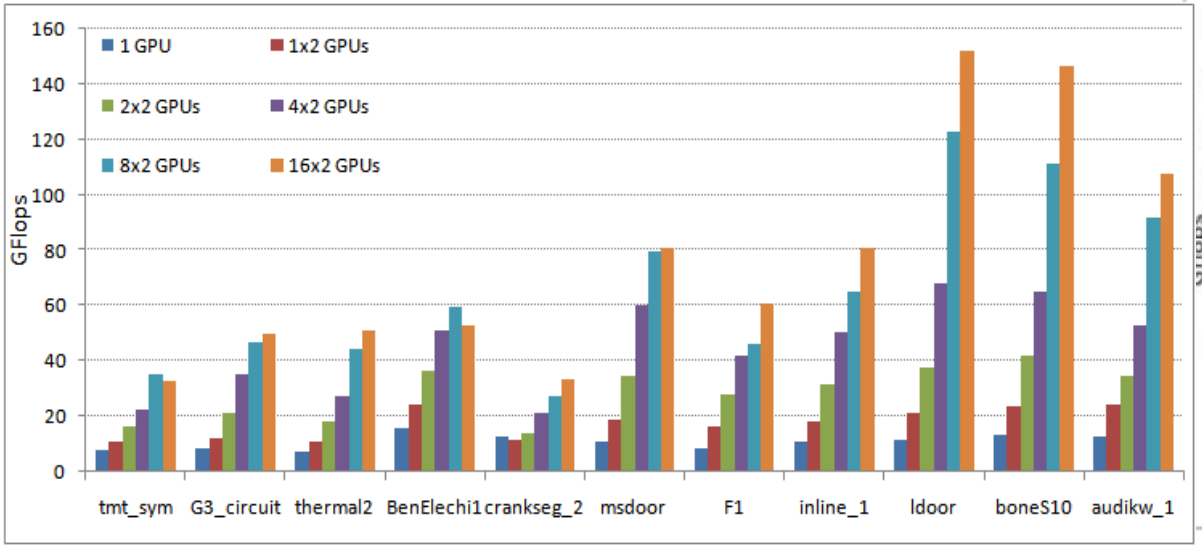
Conjugate Gradient Solver on a Multi-GPU Cluster

(A. Cevahir, A. Nukada, S. Matsuoka) [ICCS09 and follow on]

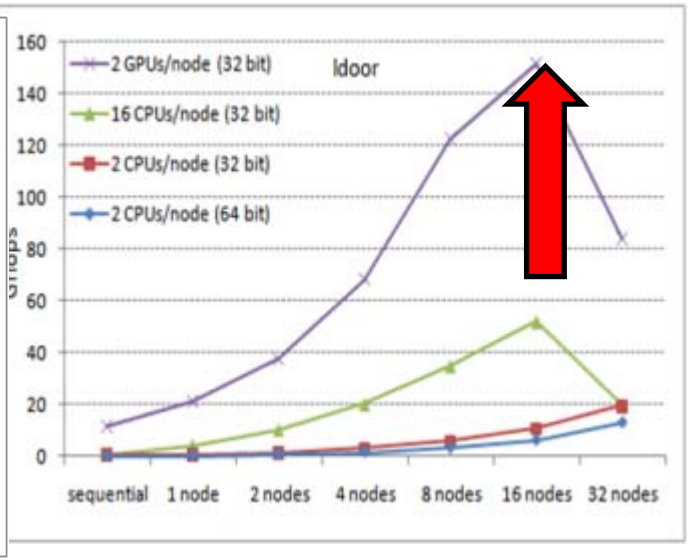
- Hypergraph partitioning for reduction of communication
  - GPU computing is fast, communication bottleneck is more severe
- All matrix and vector operations are implemented on GPUs
- Auto-selection of MxV kernel. GPUs may run different kernel

152GFlops on 32 NVIDIA GPUs on TSUBAME
(c.f. NPB CG ~3 GF on 32 node TSUBAME)

Performance comparisons over well-known matrices



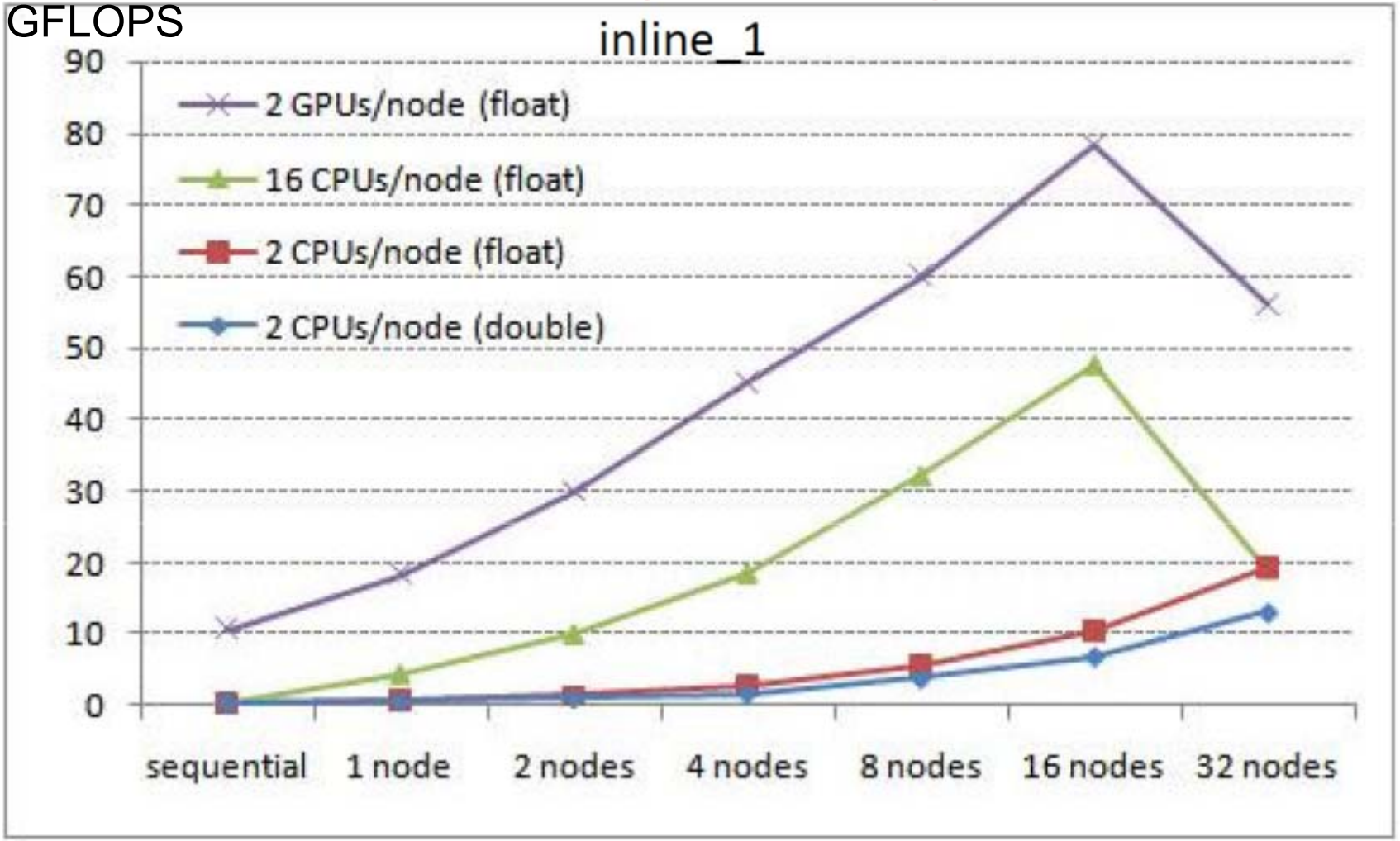
GPUs vs CPUs on TSUBAME
(Strong scaling)



Approximately x50-100 power efficient due to GPU + algorithmic improvement

Fast CG Result (2)

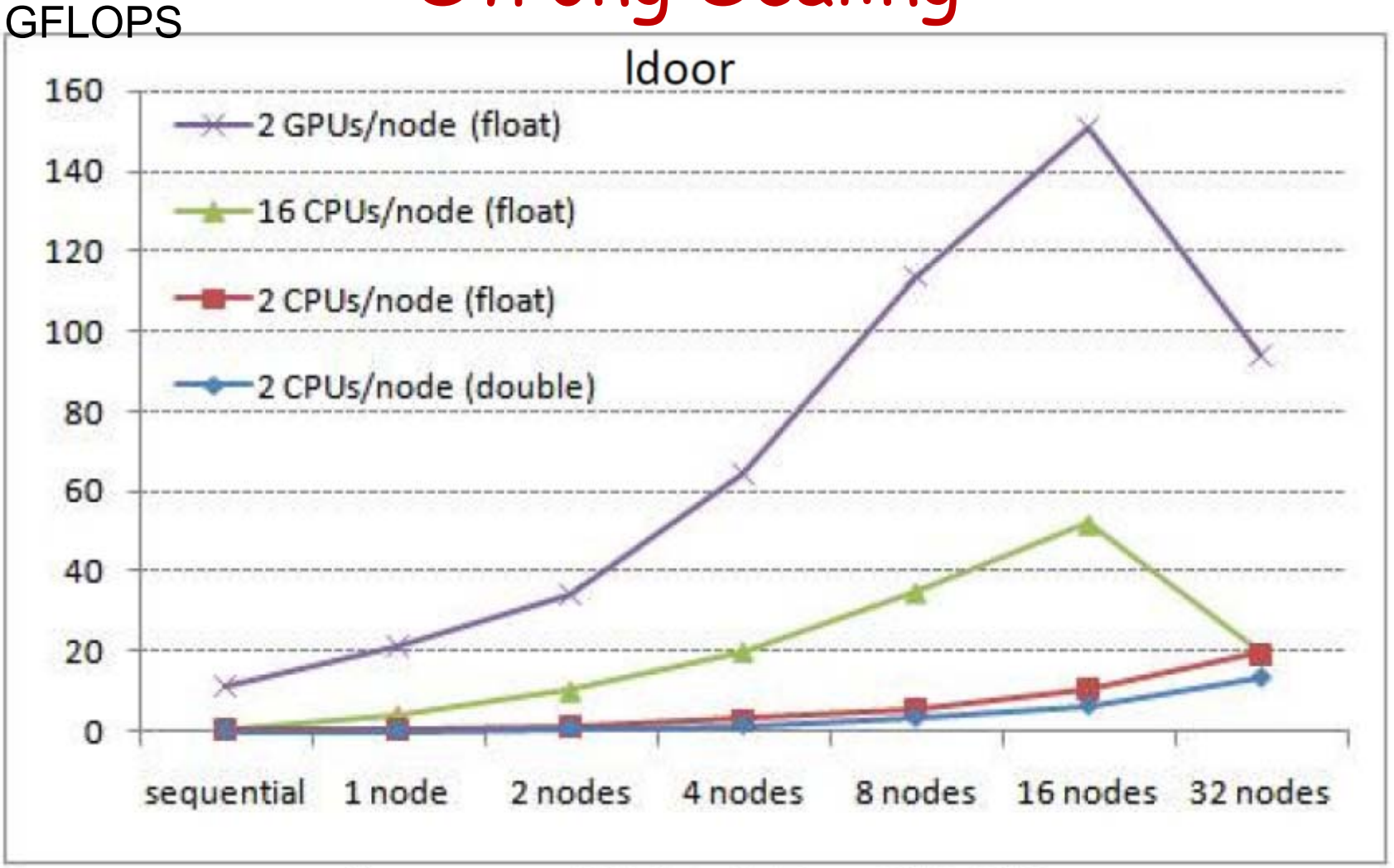
***Strong Scaling ***



Nonzeros: 36,816,342 n: 503,712

Fast CG Result (3)

***Strong Scaling ***



Nonzeros: 46,522,475 n: 952,253

(Faster Than) Real-time Tsunami Simulation

(Prof. Takayuki Aoki, Tokyo Tech.)

ADPC : Asian Disaster Preparedness Center

Early Warning System:

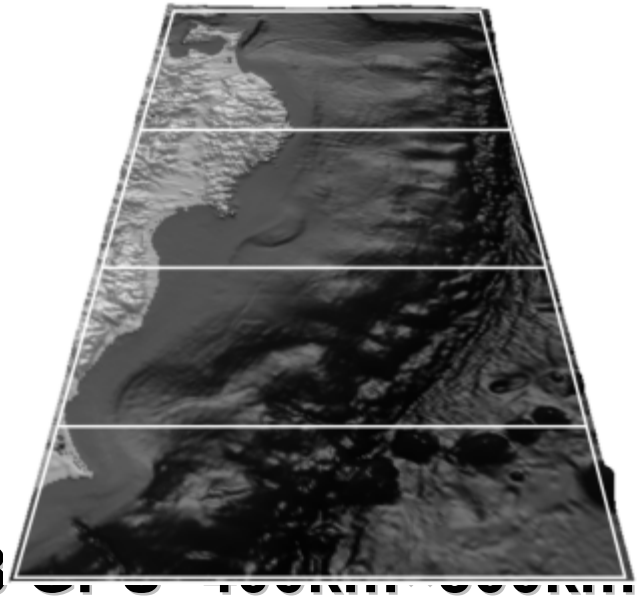
Data Based Extrapolation → high accuracy → **(Faster than) Real-time CFD**

Shallow-Water Equation

Conservative Form:

Assuming hydrostatic balance in the vertical direction,

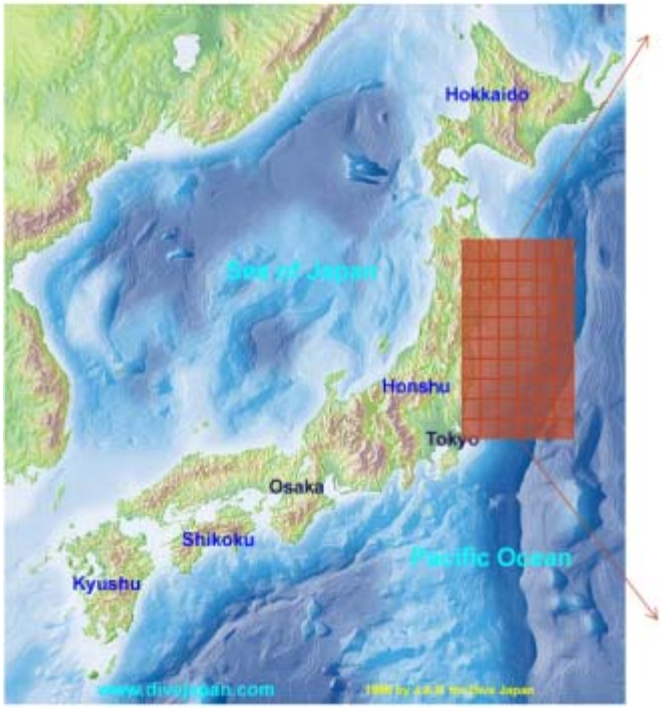
3D → 2D equation



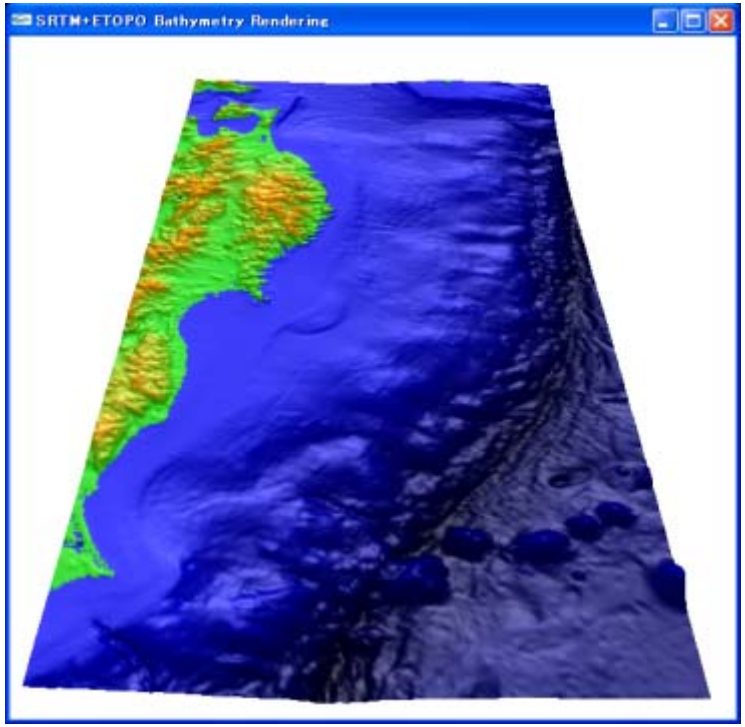
8 (100m mesh)

Tsunami Prediction of Northern Japan Pacific Coast

- Bathymetry



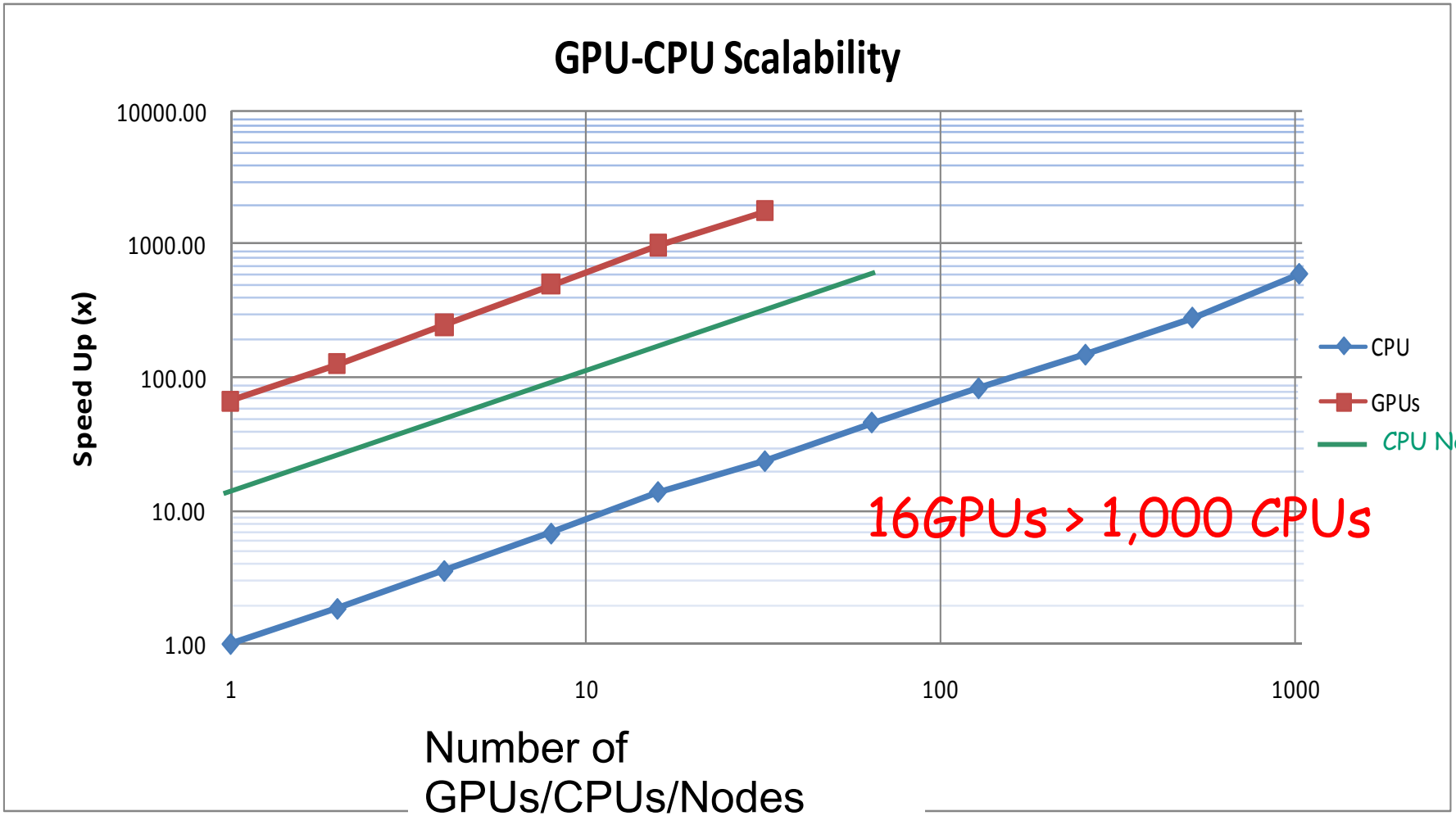
- Grid Size
  4096x8192
- Latitude
  $N35° - N42°$
- Longitude
  $E140°30' - E144°18'$
- Length
  370km x 740km

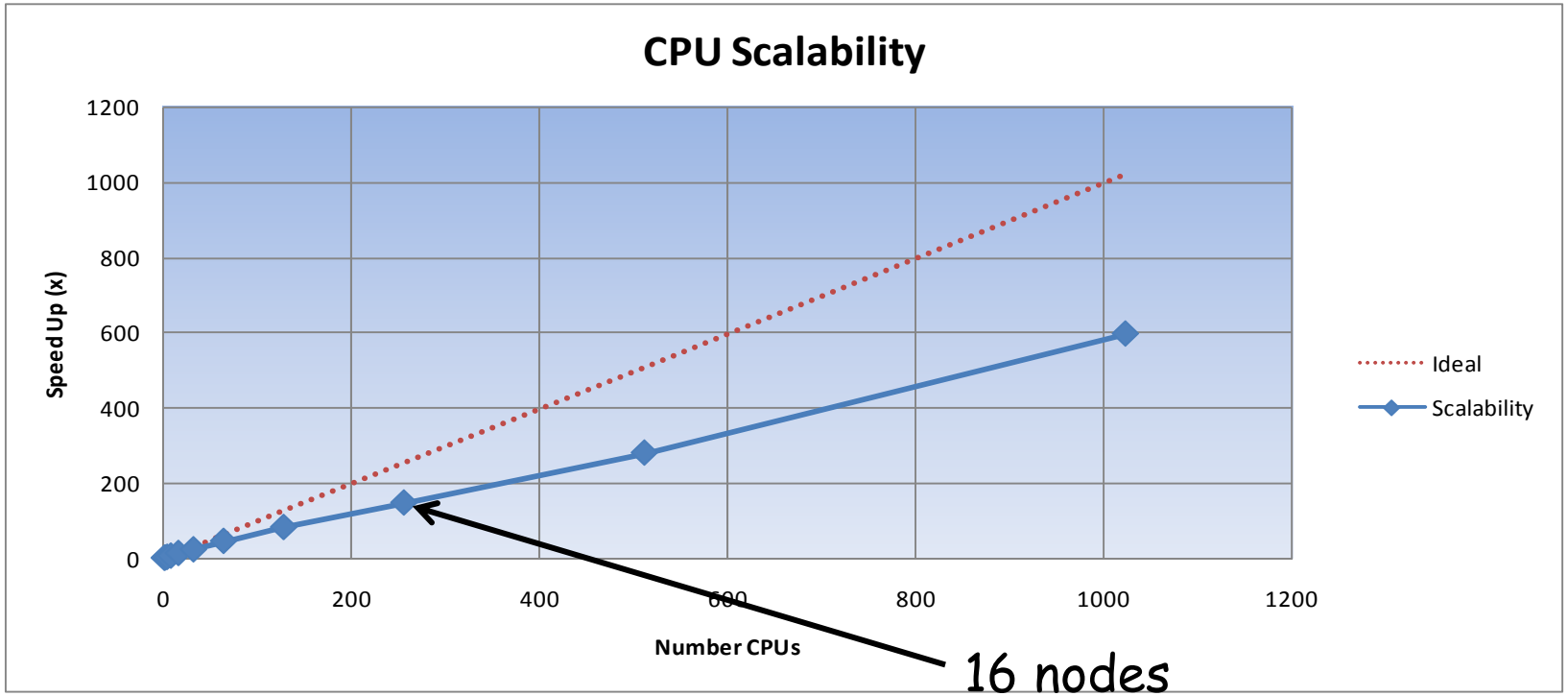


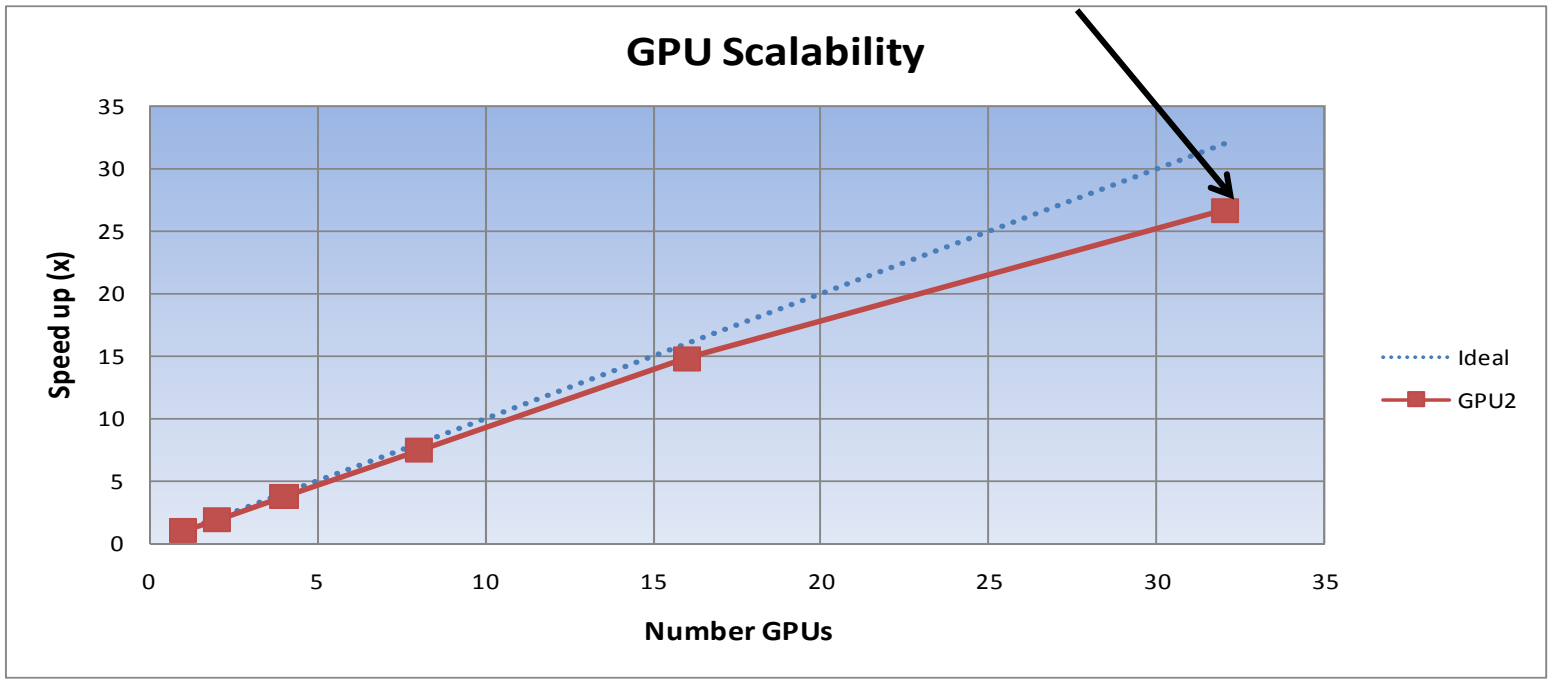
Multi-node CPU/GPU Comparison

*** Strong Scaling ***

- Results on TSUBAME1.2




CPU Scalability
Speed Up (x)
1200
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
Number CPUs
Ideal
Scalability
16 nodes

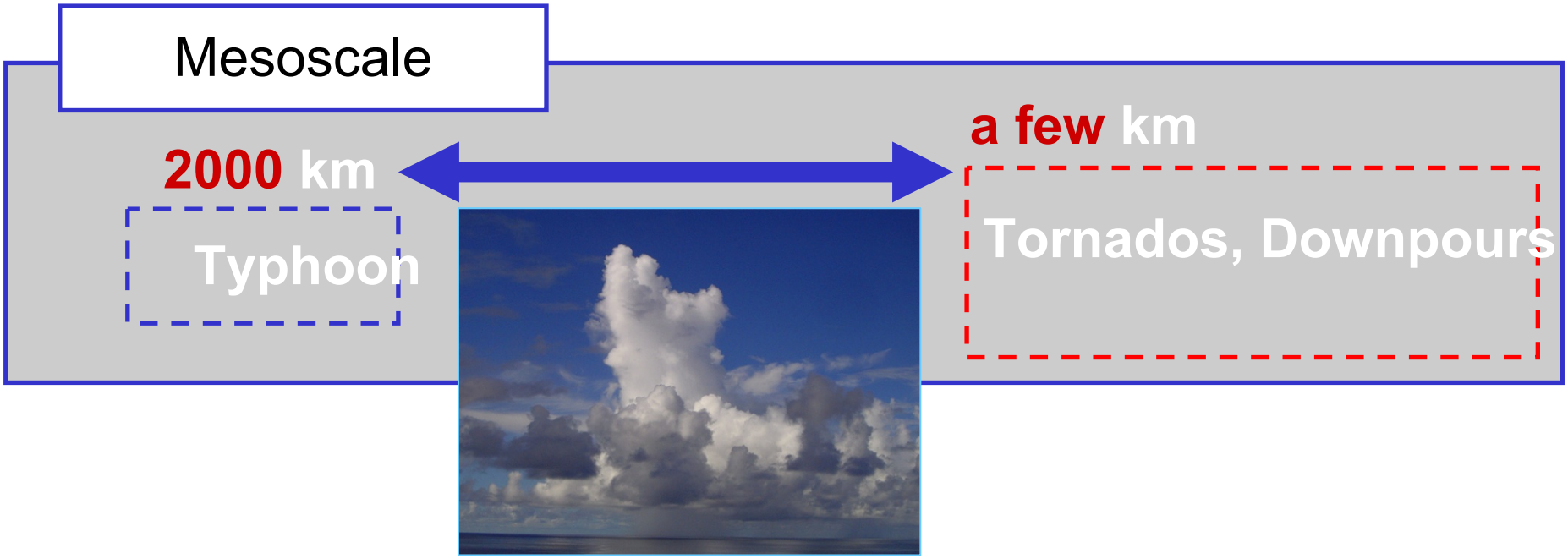

GPU Scalability
Speed up (x)
35
30
25
20
15
10
5
0
0
5
10
15
20
25
30
35
Number GPUs
Ideal
GPU2

Next Gen Weather Forecast

Mesoscale Atmospheric Model:

Cloud Resolution: 3-D non-static

Compressible equation taking consideration of sound waves.



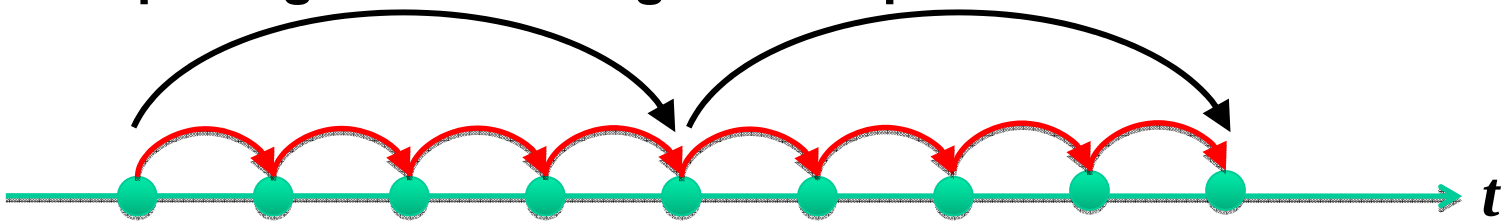
GPU enabling ASUCA

- **ASUCA : Next Generation Production Weather Forcast Code (by Japan's National Meteorlogical Agency)**

Mesoscale production code for real weather forcast

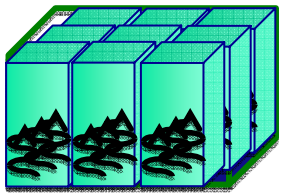
Very similar to NCAR's WRF

Time-splitting method: long time step for flow



u, v (~ 100 m/s), w (~ 10 m/s) **<<** sound velocity (~300/ms)

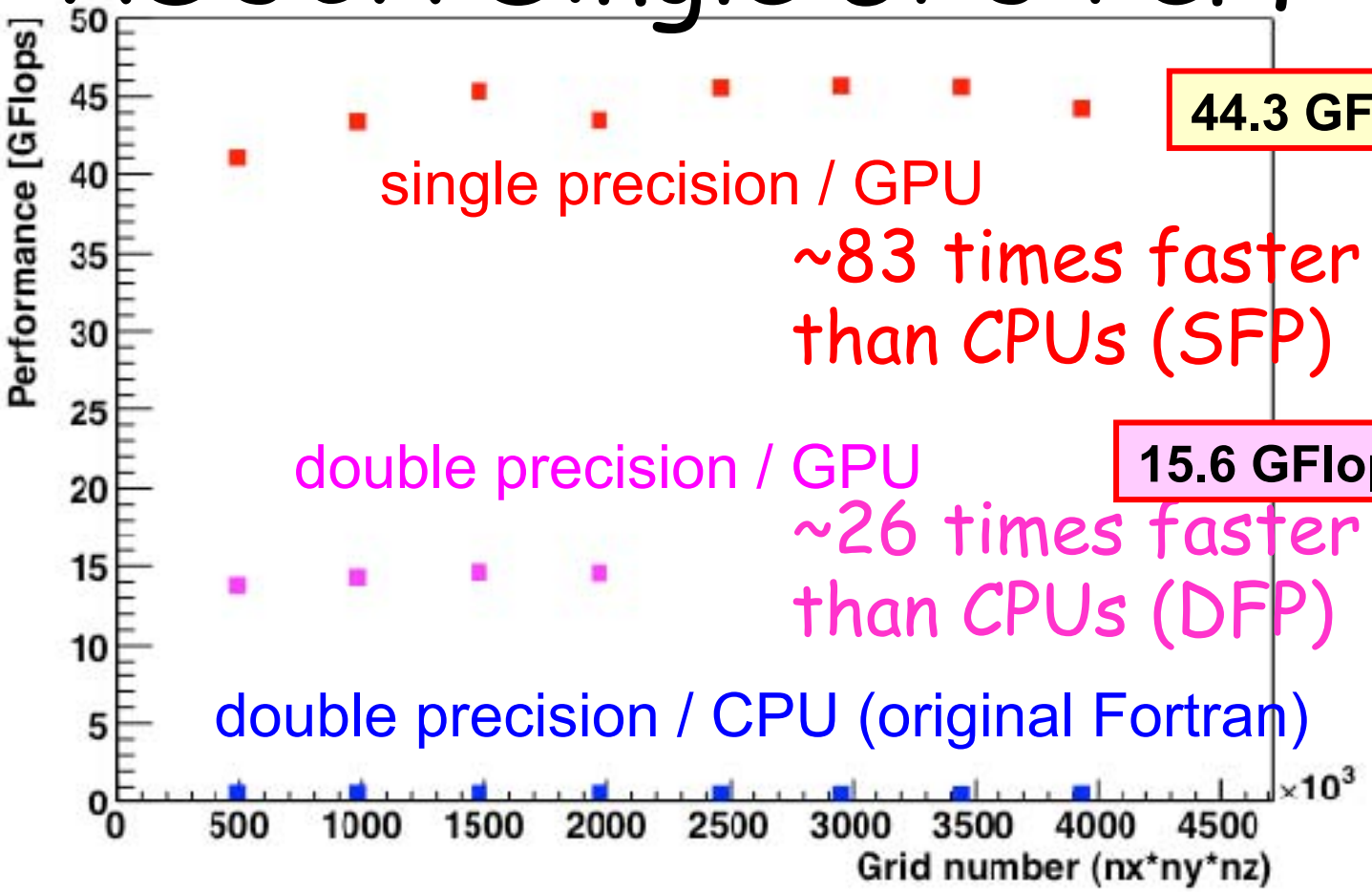
HEVI (Horizontally explicit Vertical implicit) scheme
Horizontal resolution ~ 1 km
Vertical resolution ~ 100 m



1-D Helmholtz equation (like Poisson eq.) ⇨ sequential process

Entire "Core" of ASUCA now ported to GPU (~30,000 lines)
By Prof. Aoki Takayuki's team at Tokyo Tech.

ASUCA Single GPU Perf



44.3 GFlops

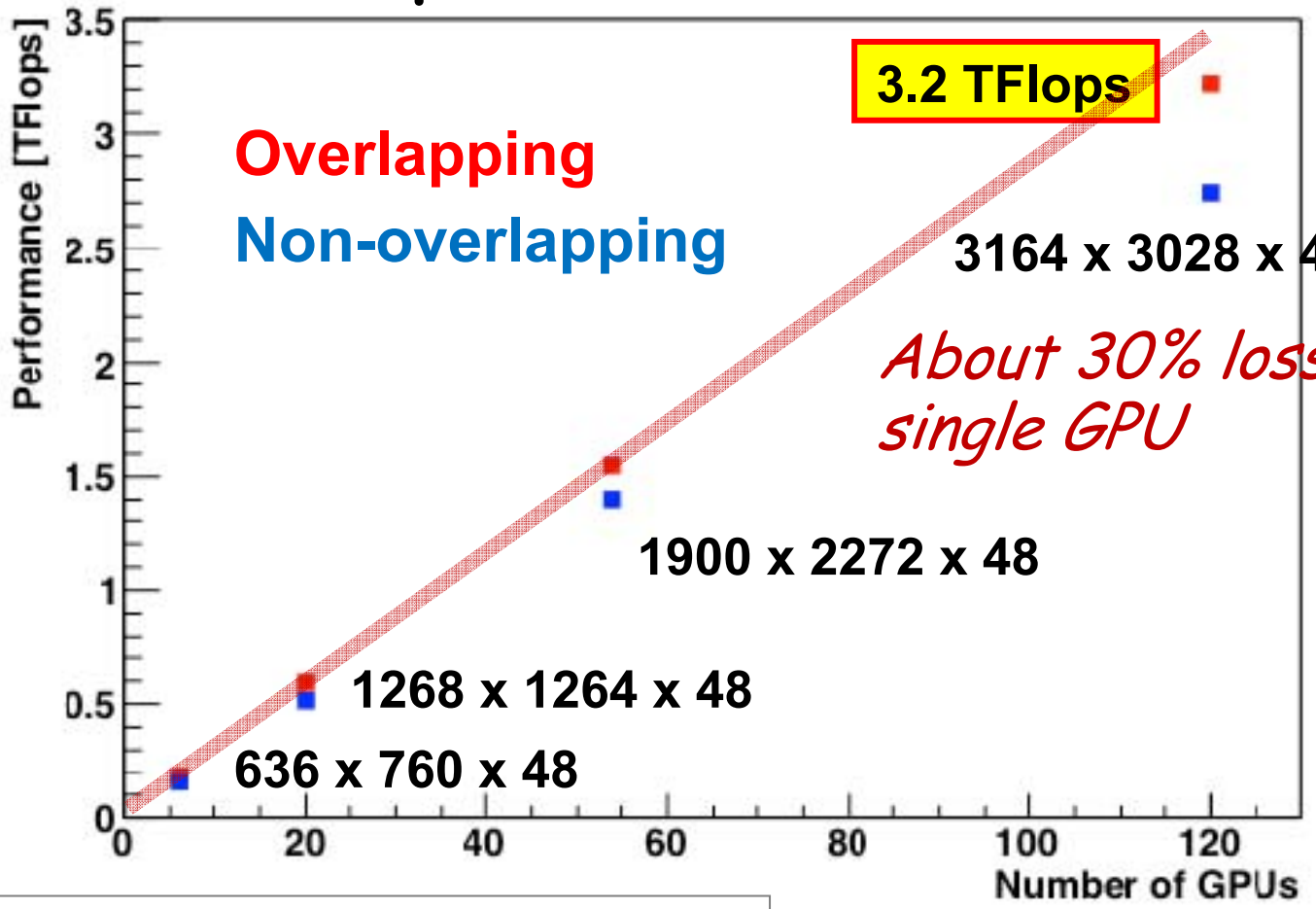
15.6 GFlops

320 x 256 x 64

Mountain Wave Test

NVIDIA Tesla S1070 card

ASUCA Multi GPU Performance (up to 120 GPUs)



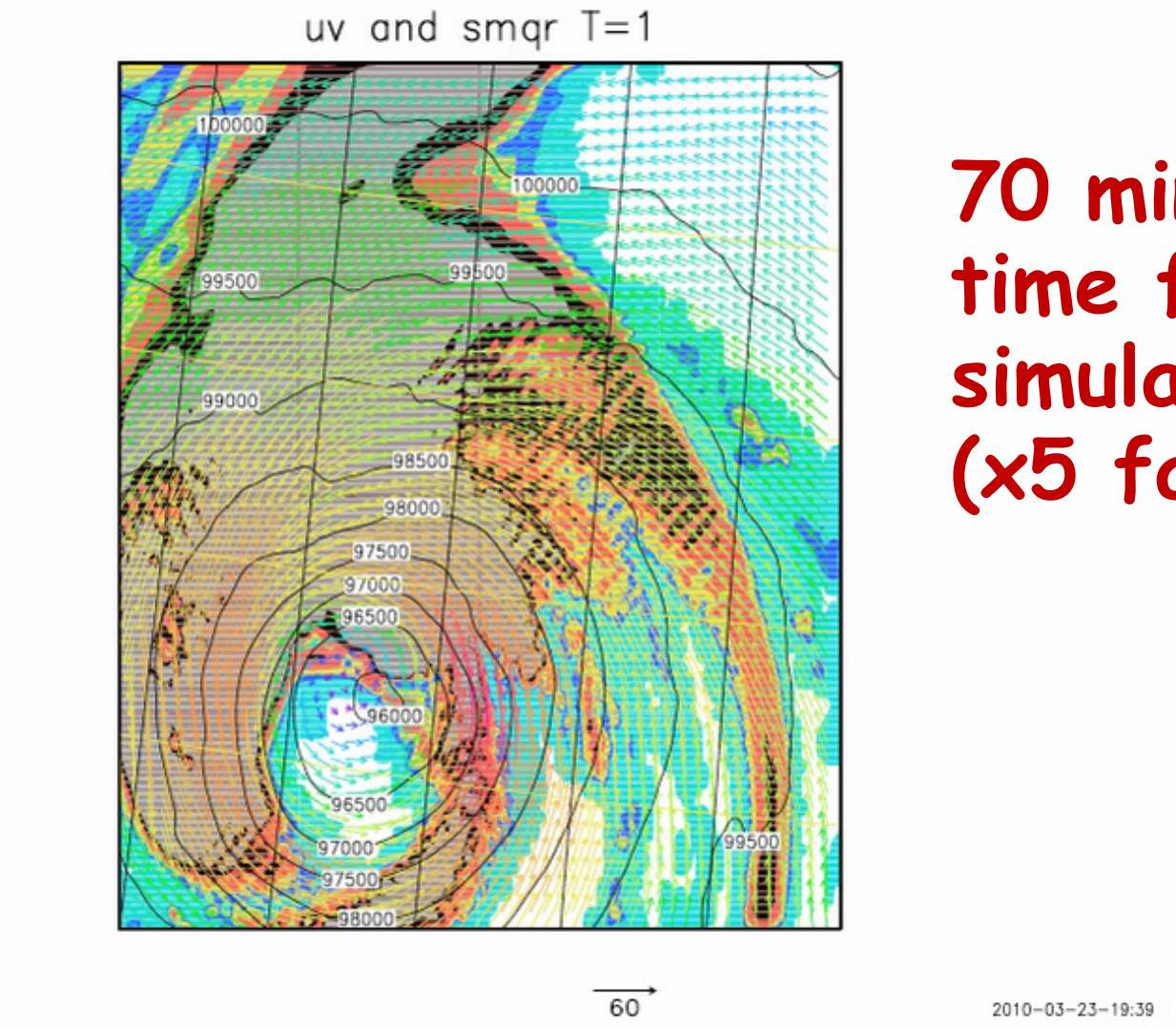
Mountain Wave Test in single precision

NVIDIA Tesla S1070 on TSUBAME

620 GPU Test ongoing

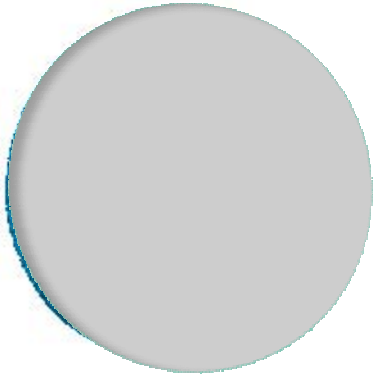
15-20 Teraflops

ASUCA Typhoon Simulation 2km mesh 3164×3028×48

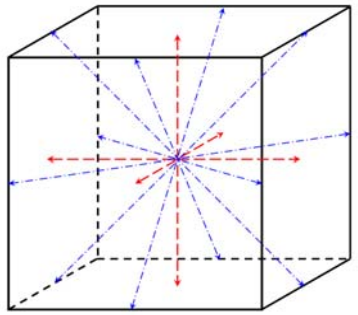


70 minutes wallclock time for 6 hour simulation time (x5 faster)

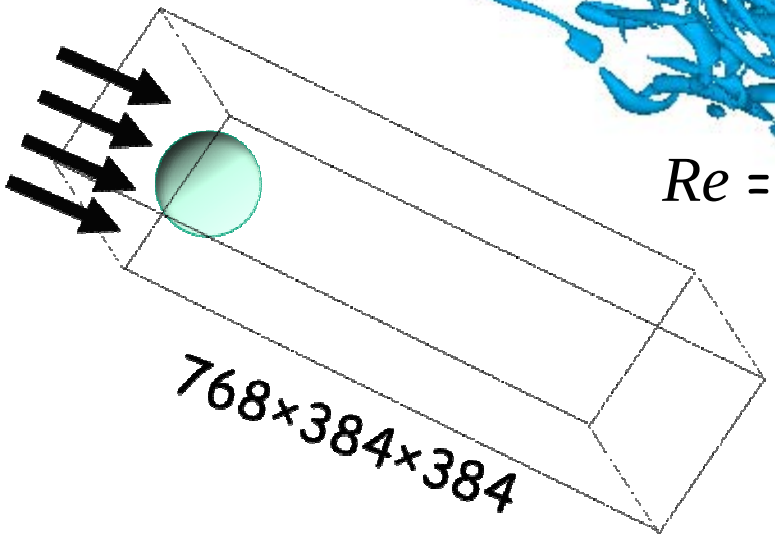
Lattice Boltzman Method on Multi-GPUs [Aoki etl al.]



$$\frac{\partial f_i}{\partial t} + \mathbf{e}_i \cdot \nabla f_i = -\frac{1}{\lambda}\left(f_i - f_i^{eq}\right)$$

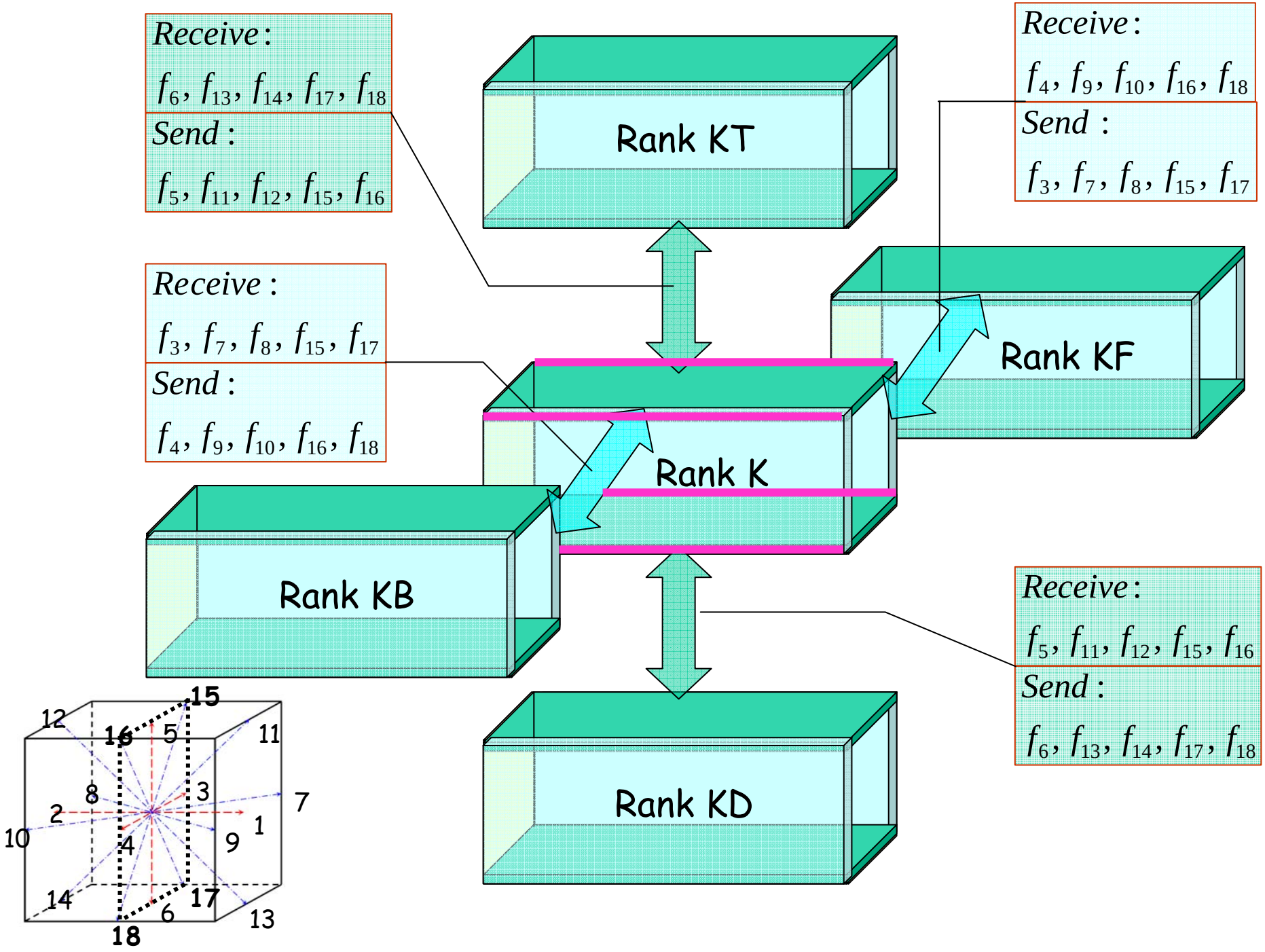


Uo

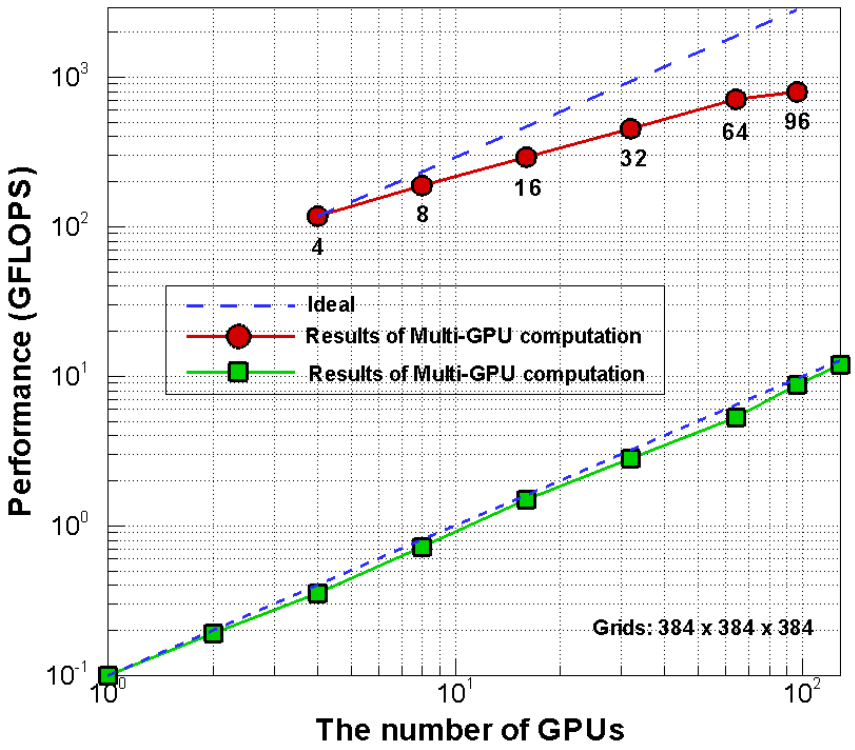


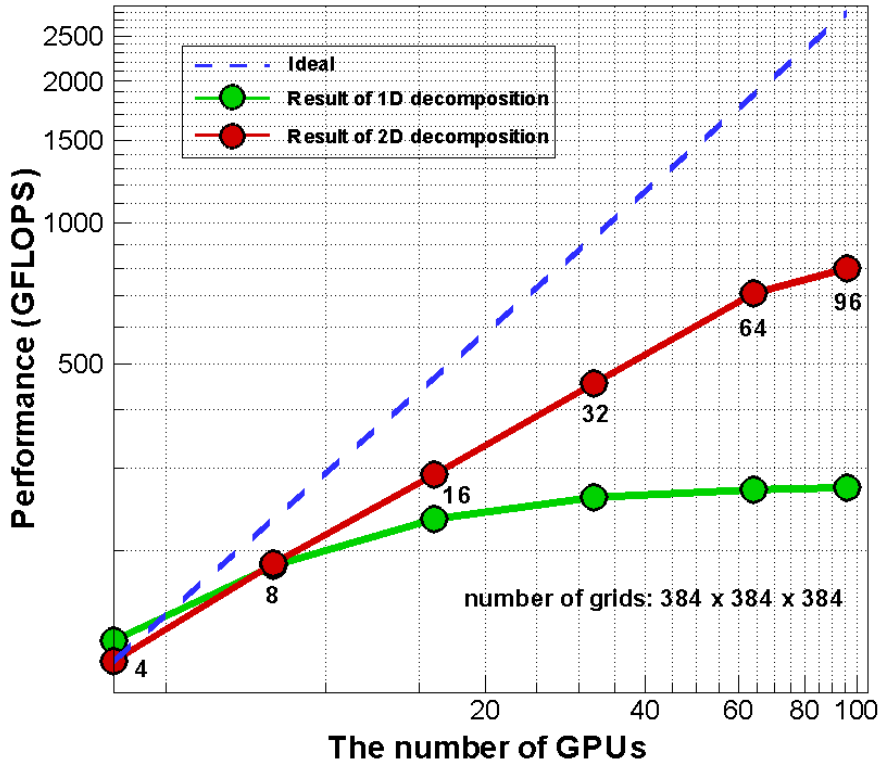
Re = 3000

Tesla S1070 **64 GPU**


Receive :
$f_6, f_{13}, f_{14}, f_{17}, f_{18}$
Send :
$f_5, f_{11}, f_{12}, f_{15}, f_{16}$
Rank KT
Receive :
$f_4, f_9, f_{10}, f_{16}, f_{18}$
Send :
$f_3, f_7, f_8, f_{15}, f_{17}$
Rank KF
Receive :
$f_3, f_7, f_8, f_{15}, f_{17}$
Send :
$f_4, f_9, f_{10}, f_{16}, f_{18}$
Rank K
Rank KB
Receive :
$f_5, f_{11}, f_{12}, f_{15}, f_{16}$
Send :
$f_6, f_{13}, f_{14}, f_{17}, f_{18}$
Rank KD
15
12
16
5
11
8
3
7
2
1
10
4
9
14
17
6
13
18

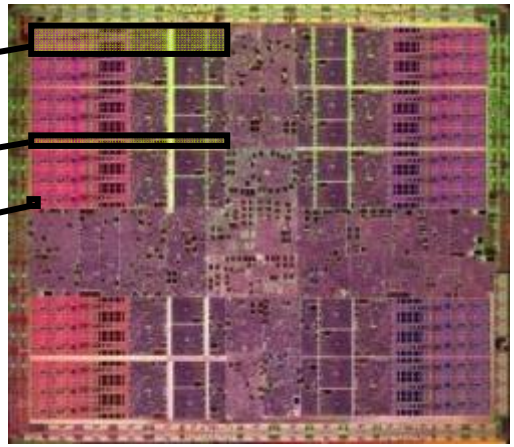
384 x 384 x 384 Lattice Boltzmann Scaling on Multi-GPUs on TSUBAME1.2



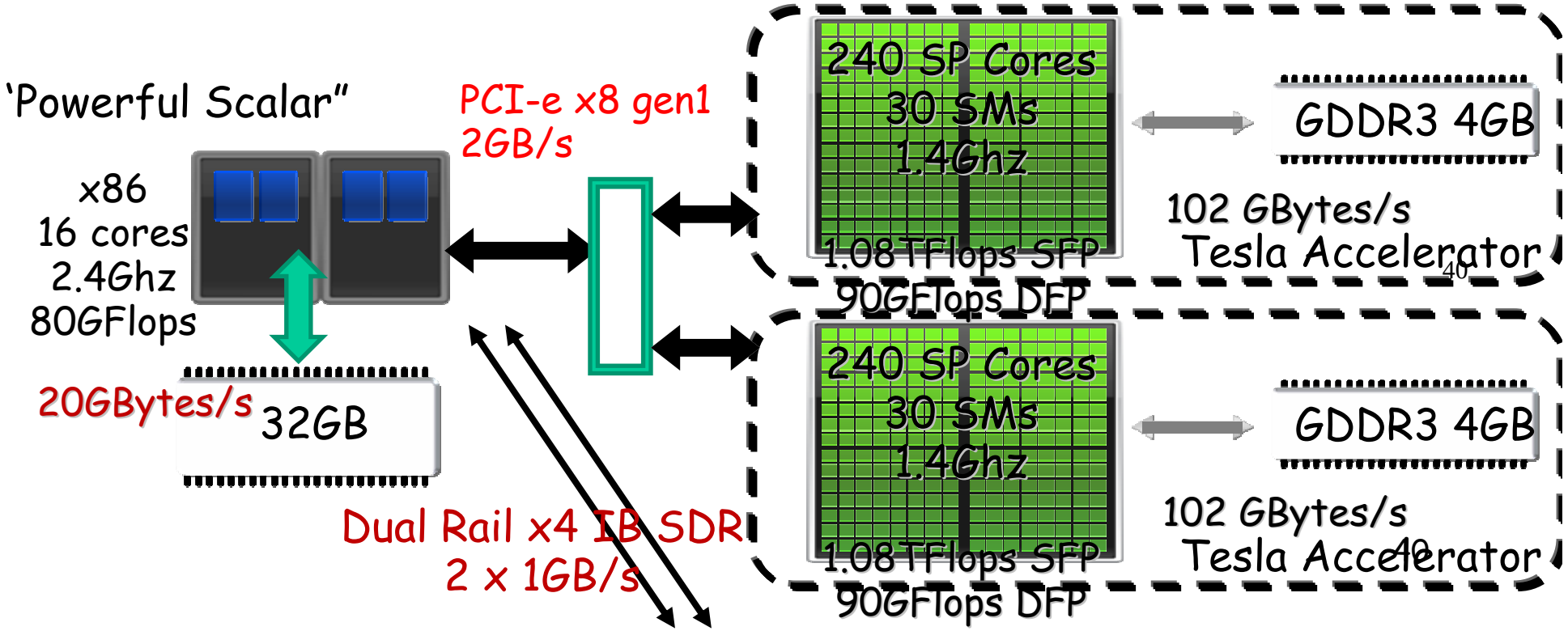


3-D Domain Decomposition + Latency Hiding
However, loss of scalability largely due to
lack of bandwidth in TSUBAME1.2(!)

TSUBAME 1.2 Node Configuration



nVidia Tesla T10: 55nm, 470m2,
1.4billion transistors
>1TF SFP
90GF DFP



Game Changing Cluster Design for 2010 and Beyond

- Multi-petascale clusters should be built with:
  - Fat, teraflop-class, compute dense, high memory BW *vector processors* for single node scalability
  - Multithreaded shared memory processors to hide latency and limit memory capacity per node GPU
  - High bandwidth, low latency, full bisection network for intra-node scalability
  - High bandwidth node memory and I/O channels to accommodate all of above
  - Node-wise non-volatile, high bandwidth silicone storage for scalable storage I/O
  - Software layers for attaining bandwidth, fault tolerance, programmability, and low power
  - Such an architecture is the basis towards Exascale

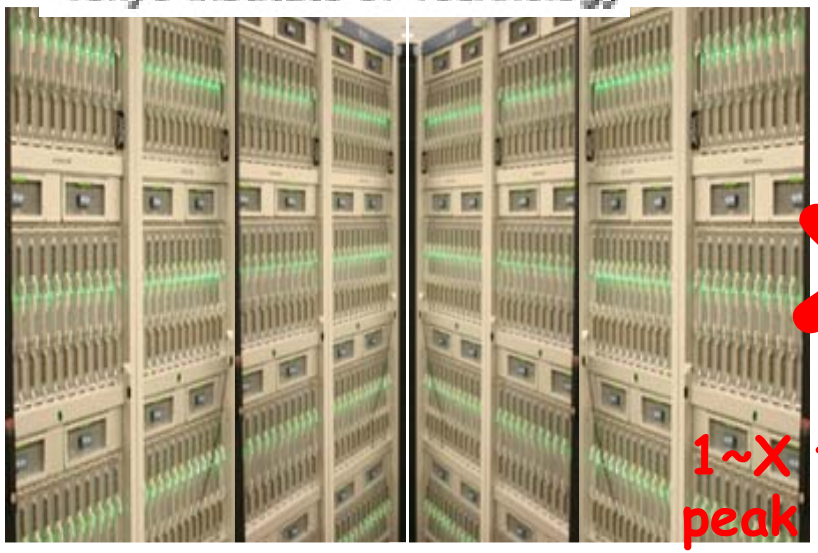
Highlights of TSUBAME 2.0 Design (Oct. 2010)

- Next gen multi-core x86 + next gen GPU
  - 1 CPU + 1 GPU ratio (or more), several thousands nodes
  - Massively Parallel Vector multithreading for **high bandwidth**
- ~Petabyte/s aggregate mem BW,
  - Effective 0.3-0.5 Bytes/Flop, restrained memory capacity
- Multi-Rail IB-QDR BW, full bisection BW (Fat Tree)
  - Likely fastest in the world, esp. c.f. mesh topology SCs
- Flash memory/node, ~Petabyte, ~Terabyte/s I/O BW
  - 6-7 PB IB attached HDDs, 15PB Total HFS incl. LTO5 tape
- Low power & efficient cooling, comparable to TSUBAME 1.0 (despite ~x40 speedup)
- Virtualization and Dynamic Provisioning of Windows HPC + Linux, job migration, etc.

Tokyo Tech. TSUBAME2.0 Oct 2010

東京工業大学
Tokyo Institute of Technology

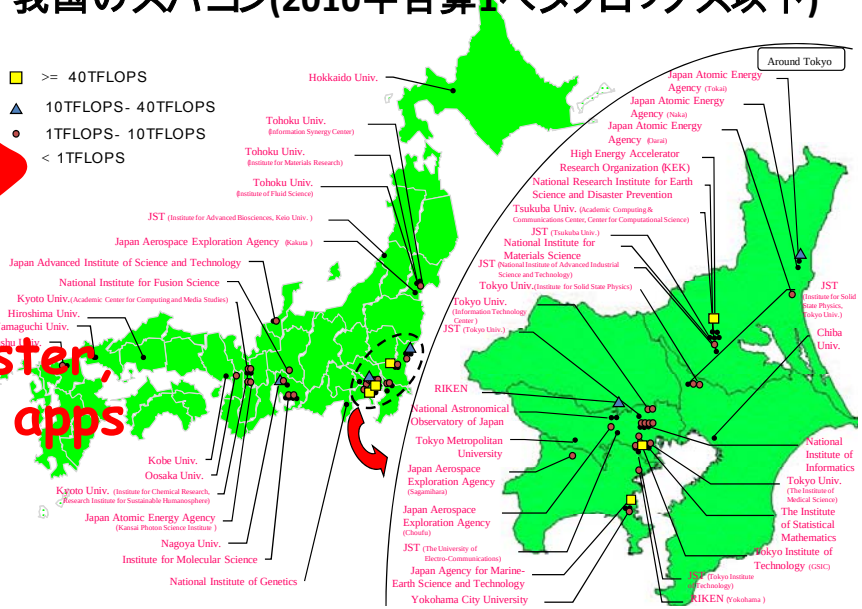
GSIC
Global Scientific Information and Computing Center



1~X times faster, peak and real apps

Entire Japan

我国のスパコン(2010年合算1ペタフロップス以下)

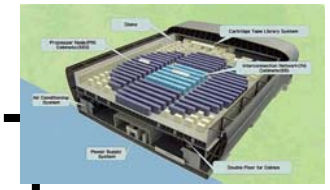


2~3 Pflop CPU+GPU (DFP), >4000GPUs
2/3~1 PB/s Memory BW
0.5~1PB SSD, 0.5~1TB/s I/O BW
Several hundred Tb/s NW Bisection
60~70 racks, incl. Storage (200m2)
6-7PB HDD, ~15PB Total Storage
~1MW Operational Power
Windows HPC + Linux, Cloud VM

2010 Total < 1PF
60 sites
> $300 million

TSUBAME 2.0 Performance

Earth Simulator → TSUBAME 4years x 40 Downsizing



TSUBAME2.0 >3 PF (2010-2H)

Japanese NLP >10PF(2012)
US >10P (2011~12)

US/EU Petascales (Peak) (2008)

US NSF/DoE (2009 2H)

U-Tokyo, Kyoto-U, Tsukuba T2K 61-140TF (2008)

Others

TSUBAME 1.2 Upgrade (3Q2008) 170TF

TSUBAME → TSUBAME 2.0 x40 downsizing again?

BlueGene/L 360TF(2005)

TSUBAME 1.1 Upgrade 111TF, 1.6 PB, 128GB nodes (1Q2008)

TSUBAME 1.0 80TF (1Q2006)

Earth Simulator 40TF (2002)

Titech Campus Grid 1.3TF

KEK 59TF BG/L+SR11100

10PF
1PF
100TF
10TF
1TF

2002 2004 2006 2008 2010 2012

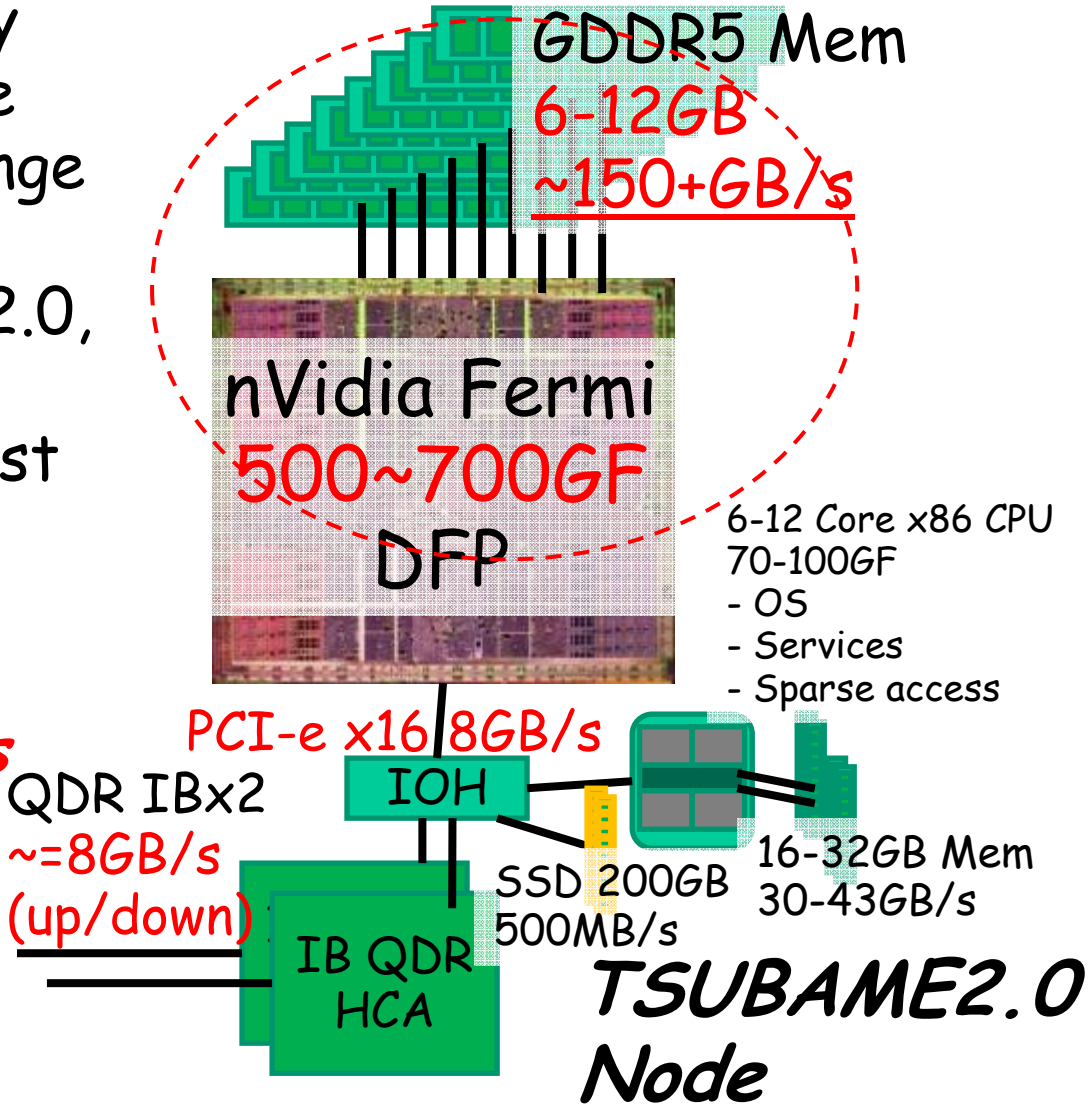
The "IDEAL TSUBAME2.0"

- 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

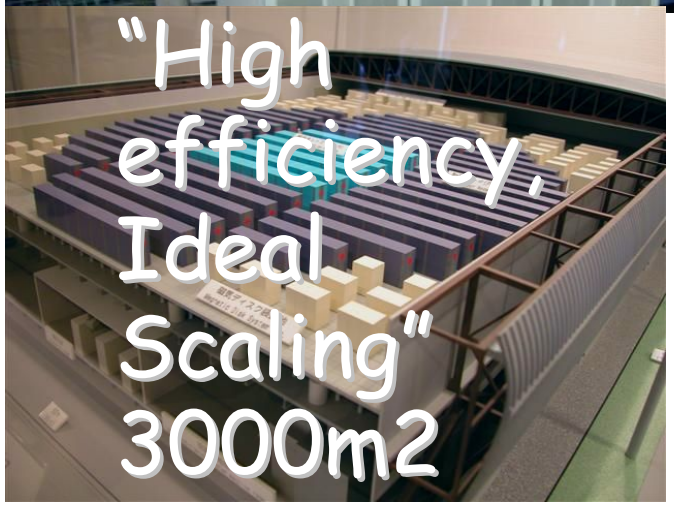
-***No current servers on market satisfy the specs***

- May be retrofitted later to come closer to "ideal"

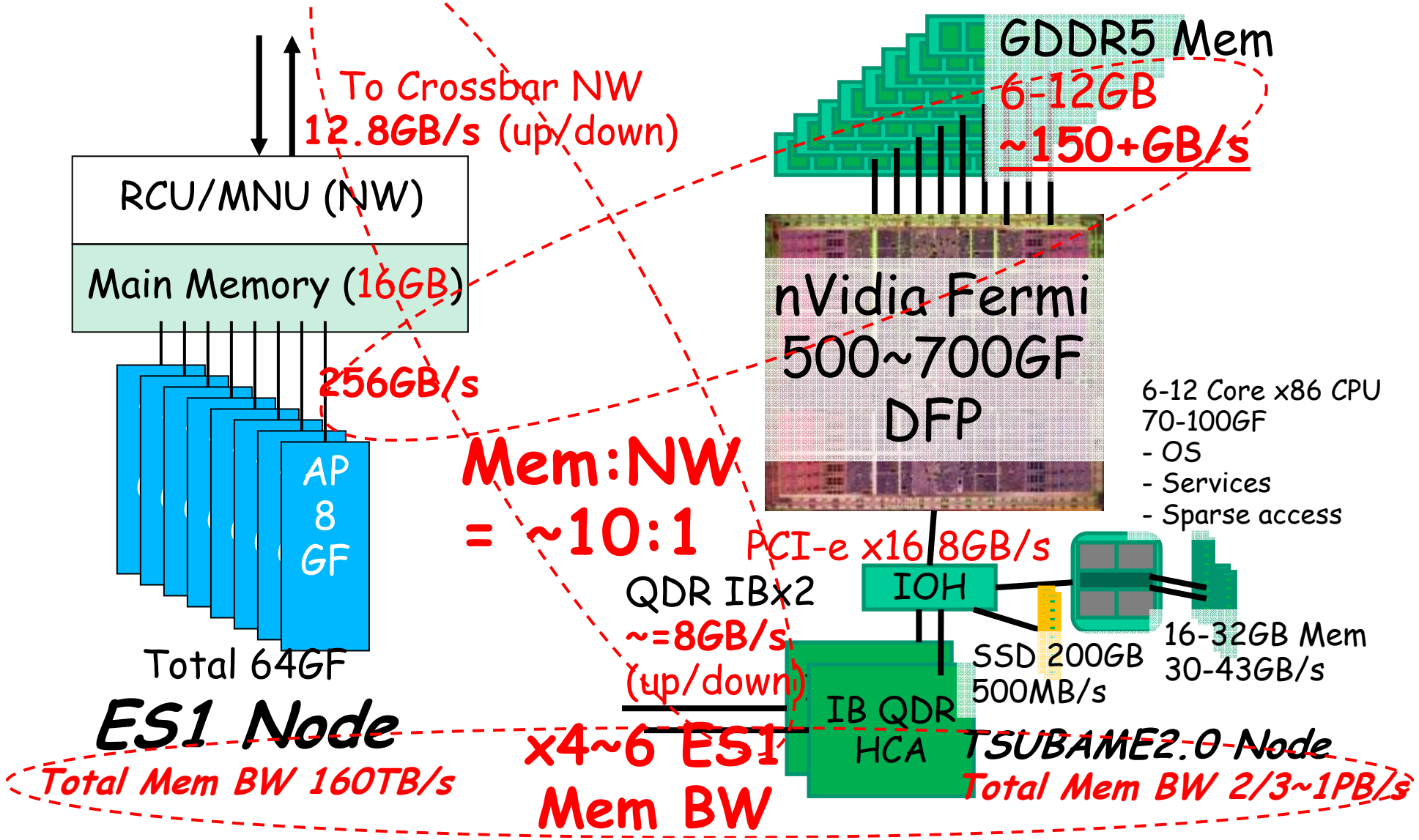


TSUBAME2.0 (2010) vs. Earth Simulator1 (ES) (2002) vs. Japanese 10PF NLP @Kobe (2012)

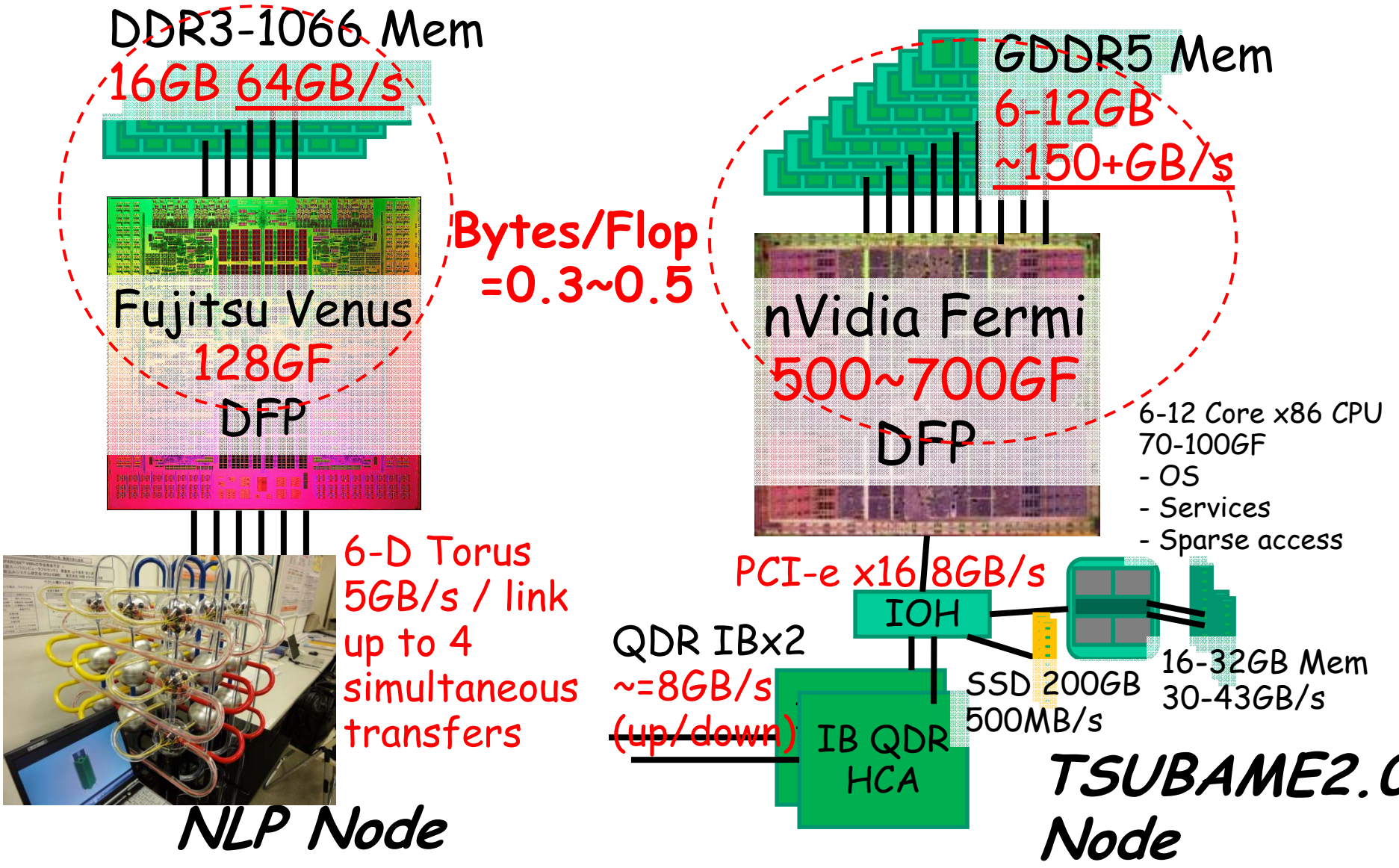
200m2



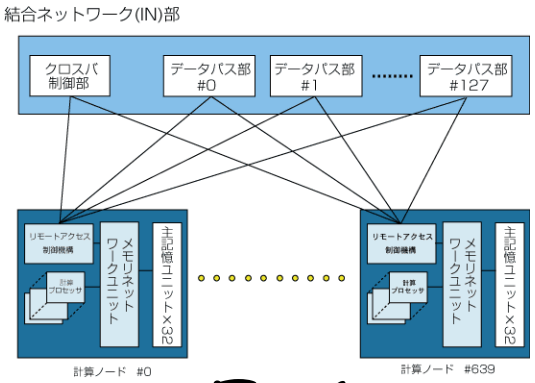
The "IDEAL TSUBAME2.0" (w/o cost constraints) node vs. ES1 node



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

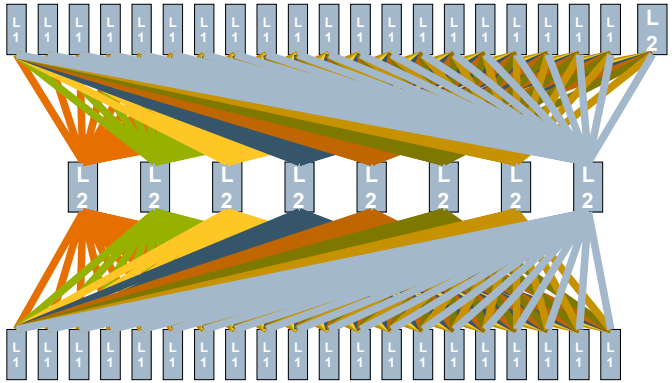


Comparing the Networks



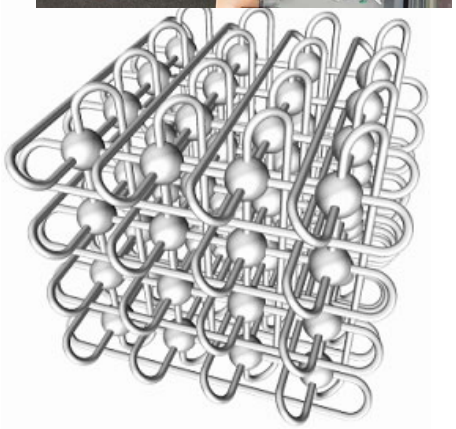
ES1

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 ∑BW : TSUBAME2 ∑BW = 1 : 6

  ⇒ **BW-bound apps (e.g. CFD) should scale equally on both w.r.t. ∑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 3 times faster on full machine (weak scaling)**

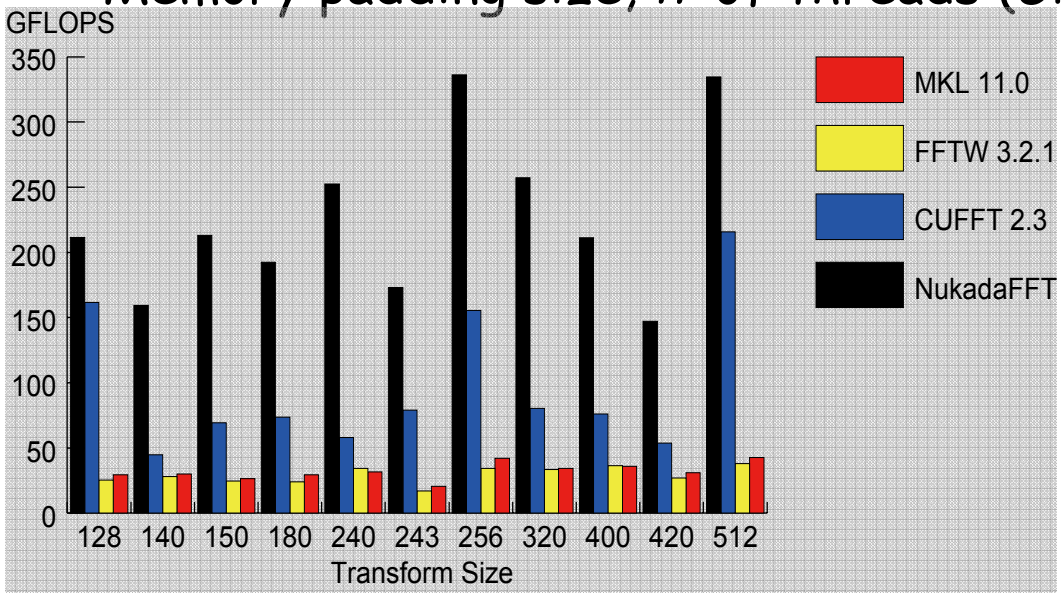
But it is not just HW Design…SOFTWARE(!)

- 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)

Auto-Tuning FFT for CUDA GPUs [ACM/IEEE SC09]

HPC libraries should support GPU variations:

- # of SPs/SMs, Core/memory clock speed,
- Device/shared memory size, Property of memory bank, etc…
- Auto-tuning of various parameters!
- Selection of kernel radices (FFT-specific)
- Memory padding size, # of threads (GPU-specific)



Our auto-tuned library is

- x4 power efficient than libraries on CPUs
- x1.2 -- 4 faster than vendor's library
- RELEASE RSN!

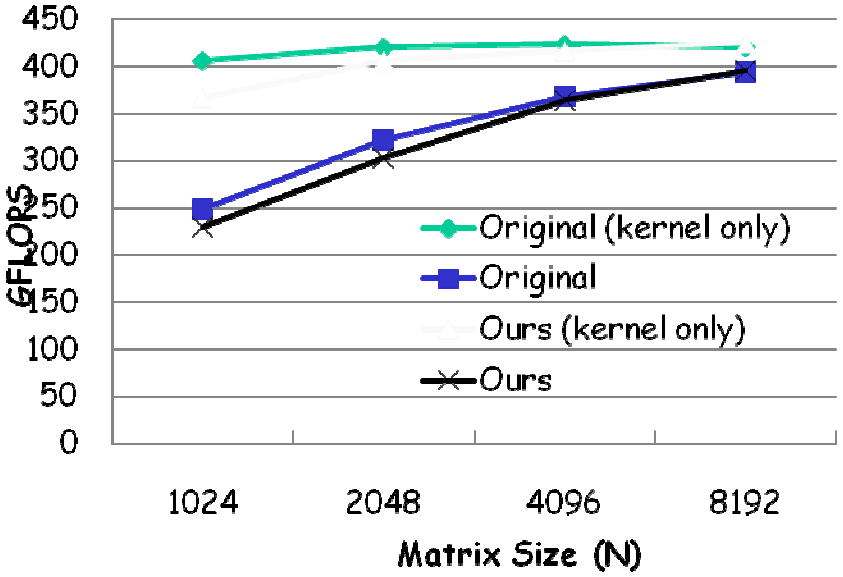
Software Framework for GPGPU Memory FT [IEEE IPDPS 2010]

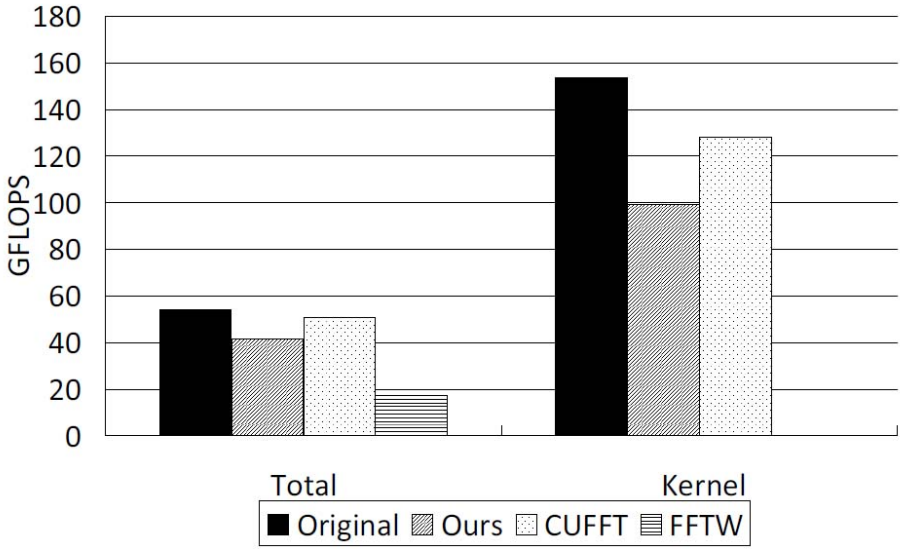
- Error detection in CUDA global memory + Checkpoint/Restart
- Works with existing NVIDIA CUDA GPUs

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 error

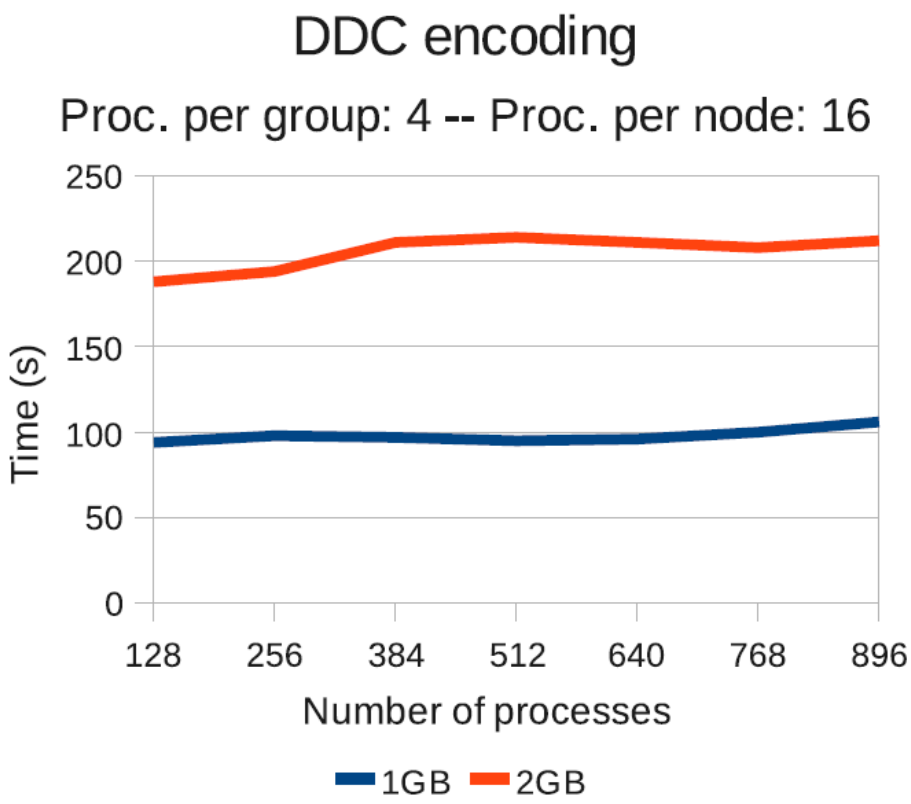
Exploit the latency hiding capability of GPUs for data correctness





Distributed Diskless Checkpoint for Large Scale Systems (to 100,00s of nodes + GPU) [IEEE CCGrid 2010]

- *Global I/O for FT will not scale and major Energy/BW waste > TBs for peta~exascale syste*
- Exploit fast *local* I/O w/NVMs (aggr. TB/s) for checkpoints
- Group-based redunduncy + RS encoding technique for *O(1) scalability*
- Combine with our ***ongoing work on checkpointing on GPUs***



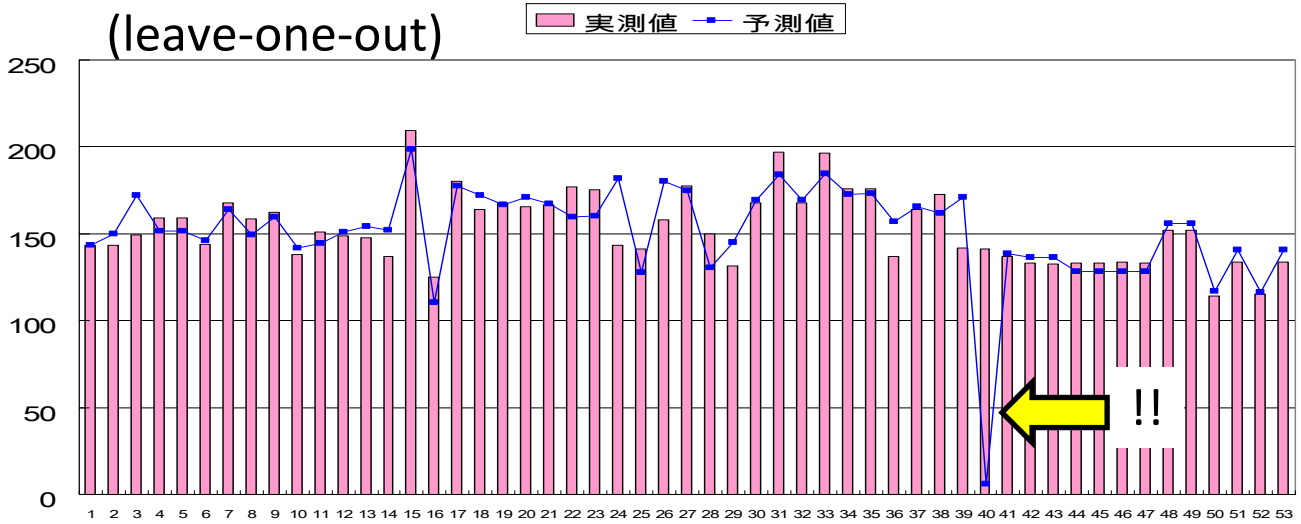
GPU Power Consumption Modeling

Employ learning theory to predict GPU power from performance counters

GPU power = $\Sigma c_i \times p_i$

p_i: GPU perf. Counter (12 types)、 c_i: learning constant

54 GPU kernels: prediction vs. measurement (leave-one-out)



Average error 7%

- Do need to compensate for extraneously erroneous kernel (!!)

TODO: Fewer couters for real-time prediction higher-precition, non-linear modeling

Low Power Scheduling in GPU Clusters

[IEEE IPDPS-HPPAC 091

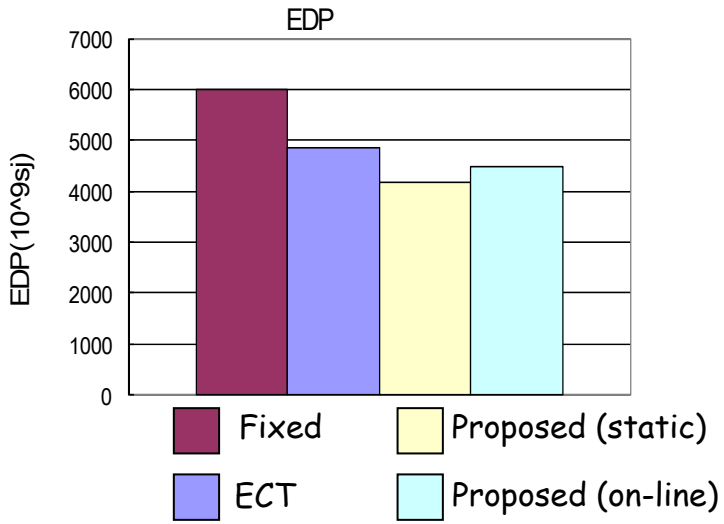
Objective : Optimize CPU/GPU Heterogeneous

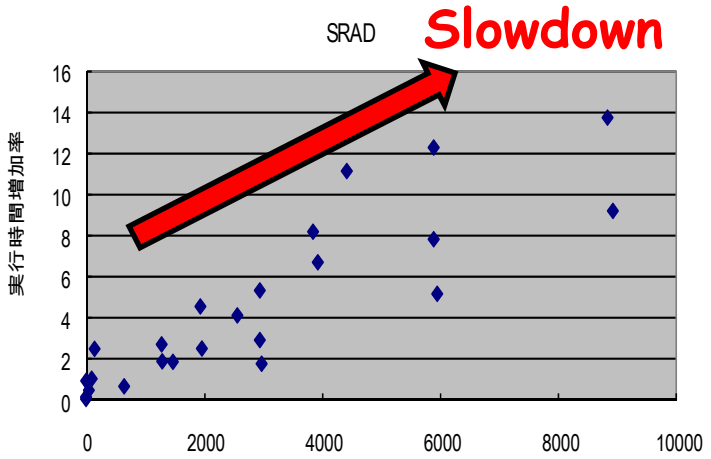
- Optimally schedule mixed sets of jobs executable on either CPU or GPU but w/different performance & power
- Assume GPU accel. factor (%) known

  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.



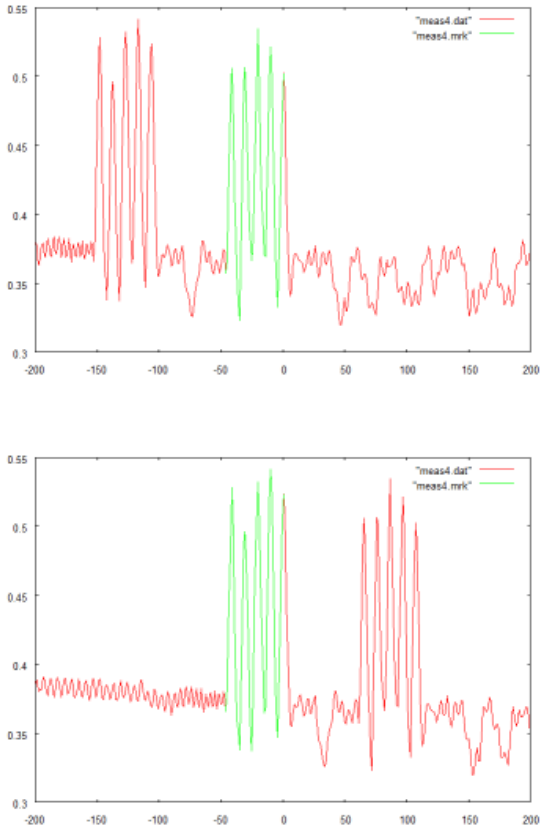


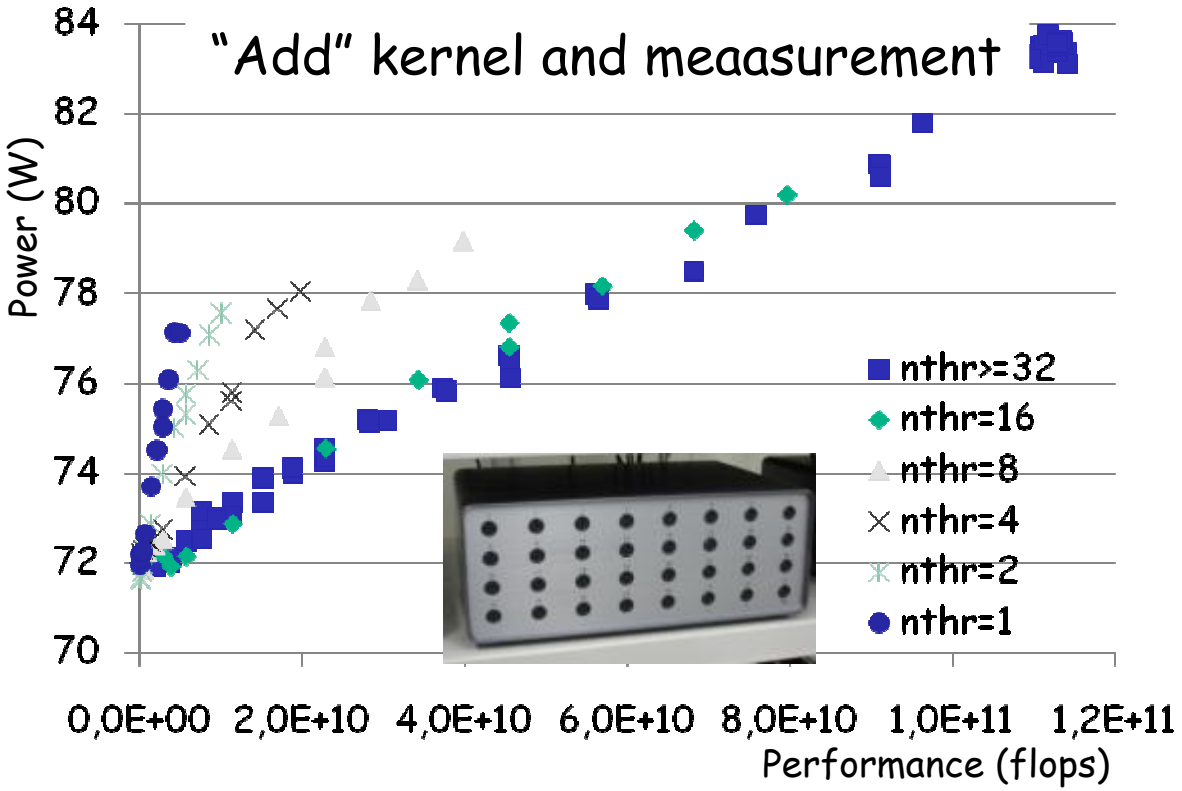
Background CPU process BW (MB/s)

High Precision GPU Power Measurement (Reiji Suda, U-Tokyo, ULPHPC)

- "Marker" = Compute + Data transfer
- Automatically detect Markers
- Sample 1000s execution in 10s seconds

マーカーの自動検出例

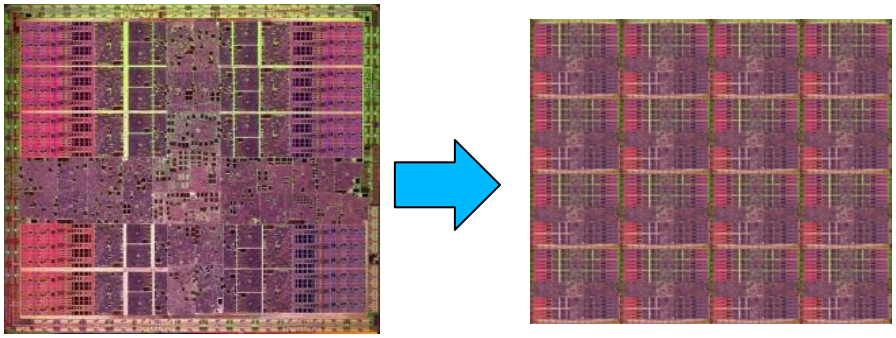




Scaling TSUBAME2.0 to Exaflop

- TSUBAME2.0: 40-45nm, ~3PF, ~50 racks, 1.5MW = x320 scaling?
- x20 physical scaling now (1000 racks, 30MW)
  - >3000-4000m^2, 1000 tons
- x16 semiconductor feature scaling 2016~2017

2008	2009	2010	2011	2012	2014	2016-17
45nm	40nm	32nm	28nm	22nm	15nm	11nm



45nm

11nm, x16 transistors & FLOPs

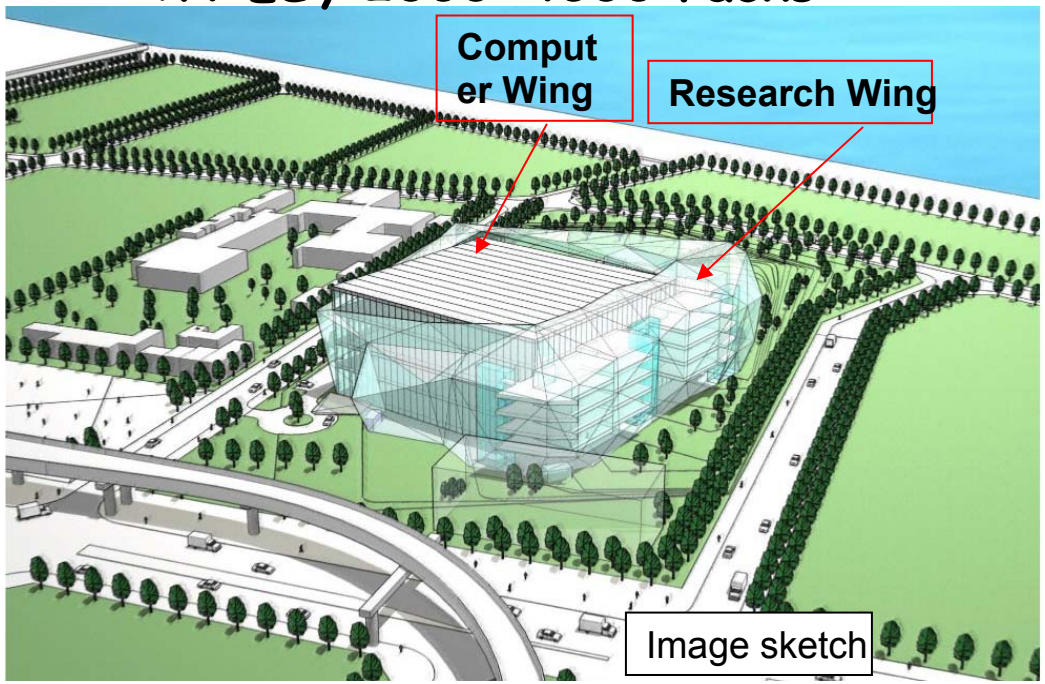
Other innovations such as 3-D memory / flash packaging, optical chip-chip connect, multi-rail optical interconnect etc.

Japanese 10 PF Facility @ Kobe, Japan

Construction: started in March, 2008 and will complete in May, 2010, Machine operation late 2011 ~ early 2012

Computer Wing

Total Floor Area:17,500m^2
2 Computer rooms: 6,300m^2 each
4 Floors (1 underground floor)
=> x4 ES, 2000~4000 racks



Research Wing

Total floor area: 8,500m^2
Area of 1floor: 1,900m^2
7 floors (1 underground floor)
(cafeteria is planned on the 6th floor)

Other Facilities

Co-generation System
Water chiller system
Electric Subsystem

Initially 30MW capability@2011