ポストペタスケールの計算機システム
～ ヘテロ，マルチコア，加速器，超並列，大規模ストレージ ～

東京工業大学　学術国際情報センター (GSIC)
松岡　聡

筑波大学

「先端学際計算科学共同研究拠点キックオフ・シンポジウム」

第一回「学際計算科学による新たな知の発見・統合・創出」シンポジウム

--ポストペタスケールコンピューティングへの学際計算科学の展開 --

今後のペタ級マシン

Inst/Agency/Country	Name	Machine	Perf
ORNL/DoE/US 2009	Jaguar Upgrade	Cray XT5	~2PF
Utennessee/NSF/US	Cracken	Cray XT5	1PF
LLNL/DoE/US	Sequoia Proto	IBM BG/P	~1PF
Tokyo tech./MEXT/JP	TSUBAME2.0	GPU Cluster/TBD	2-3PF
LBNL/DoE/US 2010	Franklin 6	Cray XT6	1.2PF
Pittsburgh SC/NSF/US	???	SGI UV	2PF?
LANL/DoE/US	???	???	???
欧州ペタ	???	IBM/Cray/Sun/Bull…	1-2PF?
ORNL/DoE/US	Jaguar Upgrade	Cray XT6 +GPU?	20PF
NCSA/NSF/US 2011-12	Blue Waters	IBM Power7 server	10+PF
LLNL/DoE/US	Sequoia	IBM BG/Q / PERCS	22PF
ArgonneNL/DoE/US	???	IBM BG/Q / PERCS	~20PF
神戸ペタ-Riken/MEXT/JP	???	富士通 Venus 専用設計	~10PF
欧州ペタコン郡/独仏スペインなど	???	IBM, Cray等	~xPF x 4~5
中国	6個所	???Dawning?	~1PF x 6

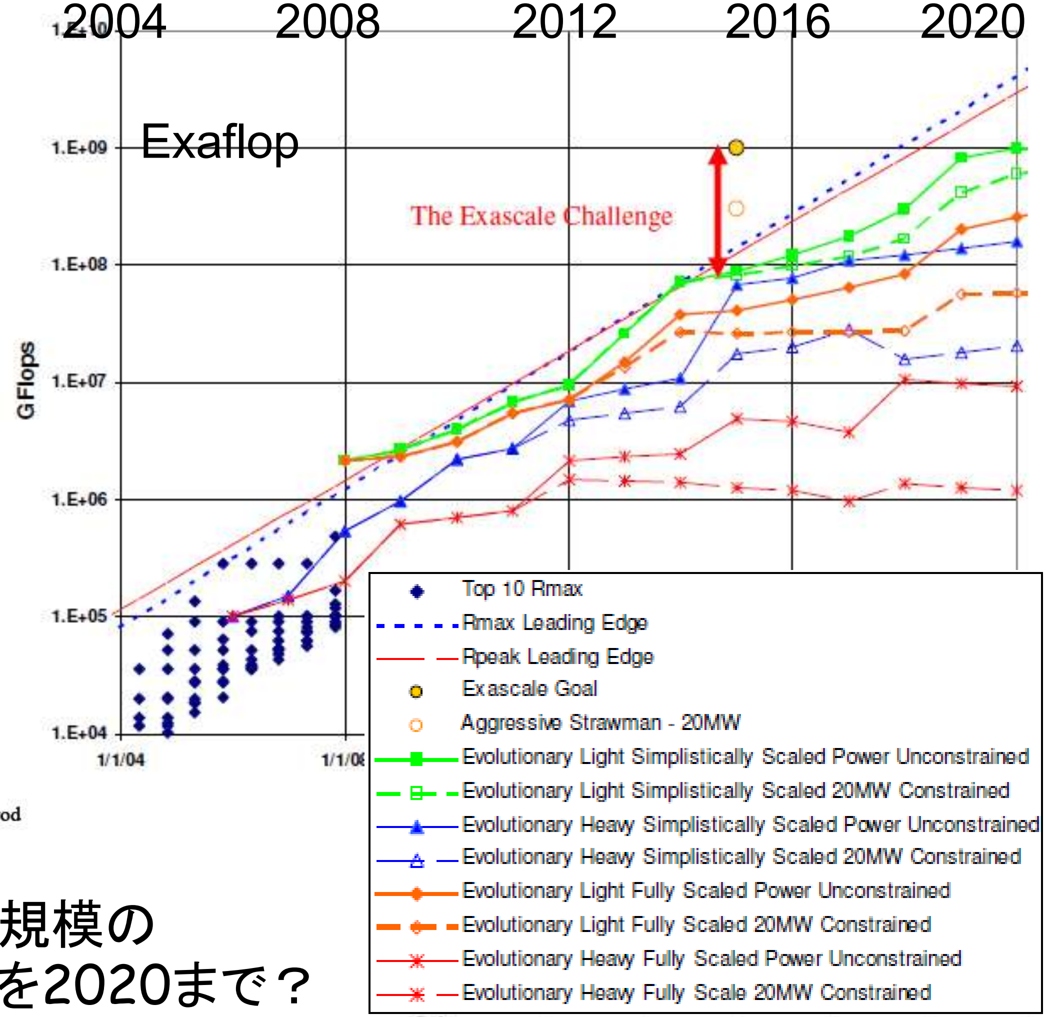
ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems

Peter Kogge, Editor & Study Lead
Keren Bergman
Shekhar Borkar
Dan Campbell
William Carlson
William Dally
Monty Denneau
Paul Franzon
William Harrod
Kerry Hill
Jon Hiller
Sherman Karp
Stephen Keckler
Dean Klein
Robert Lucas
Mark Richards
Al Scarpelli
Steven Scott
Allan Snavely
Thomas Sterling
R. Stanley Williams
Katherine Yelick

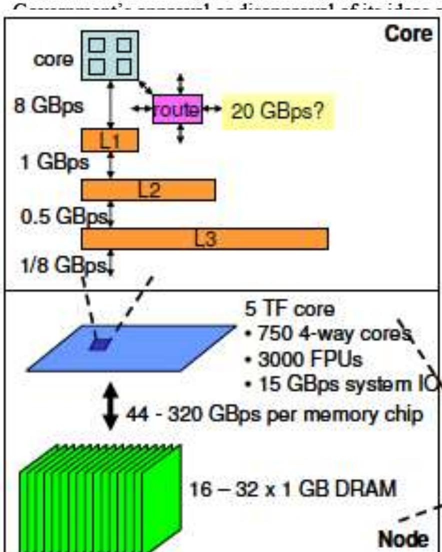
September 28, 2008

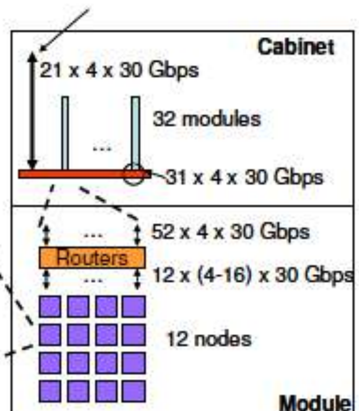
This work was sponsored by DARPA IPTO in the ExaScale Computing Study with Dr. William Harrod as Program Manager; AFRL contract number **FA8650-07-C-7724**. This report is published in the interest of scientific and technical information exchange and its publication does not constitute the

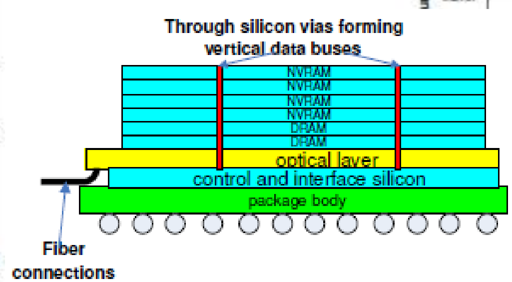
Peter Koggeらによる300ページのDoD Exascaleシステムのレポート



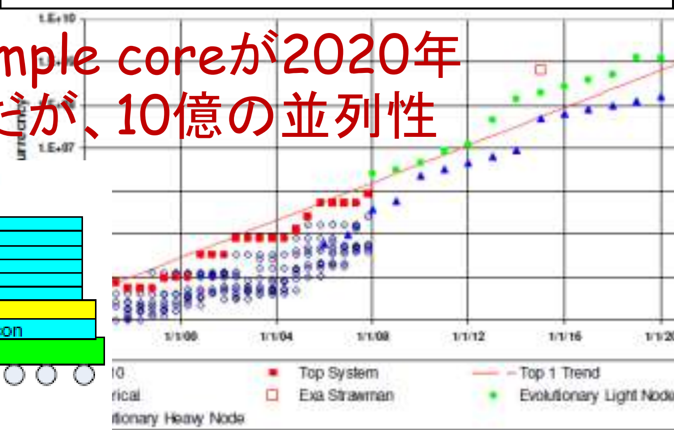
DoE は5000億規模のExaプロジェクトを2020まで？



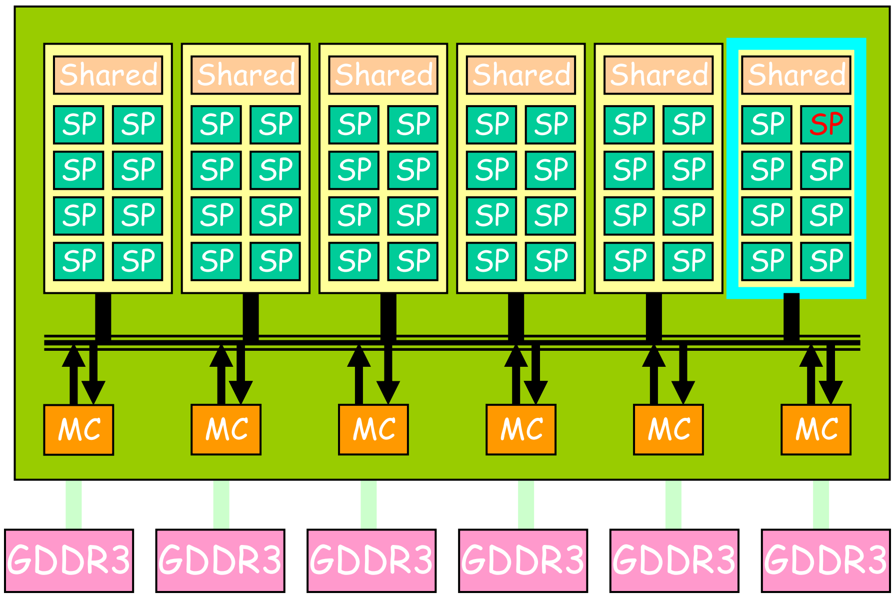




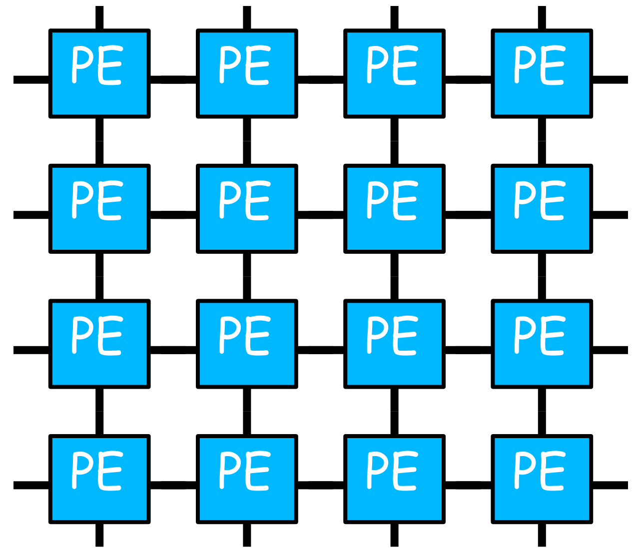
軽量なsimple coreが2020年頃有望だが、10億の並列性



GPU (Multithreaded Vector) vs. Standard Many Cores?



vs.



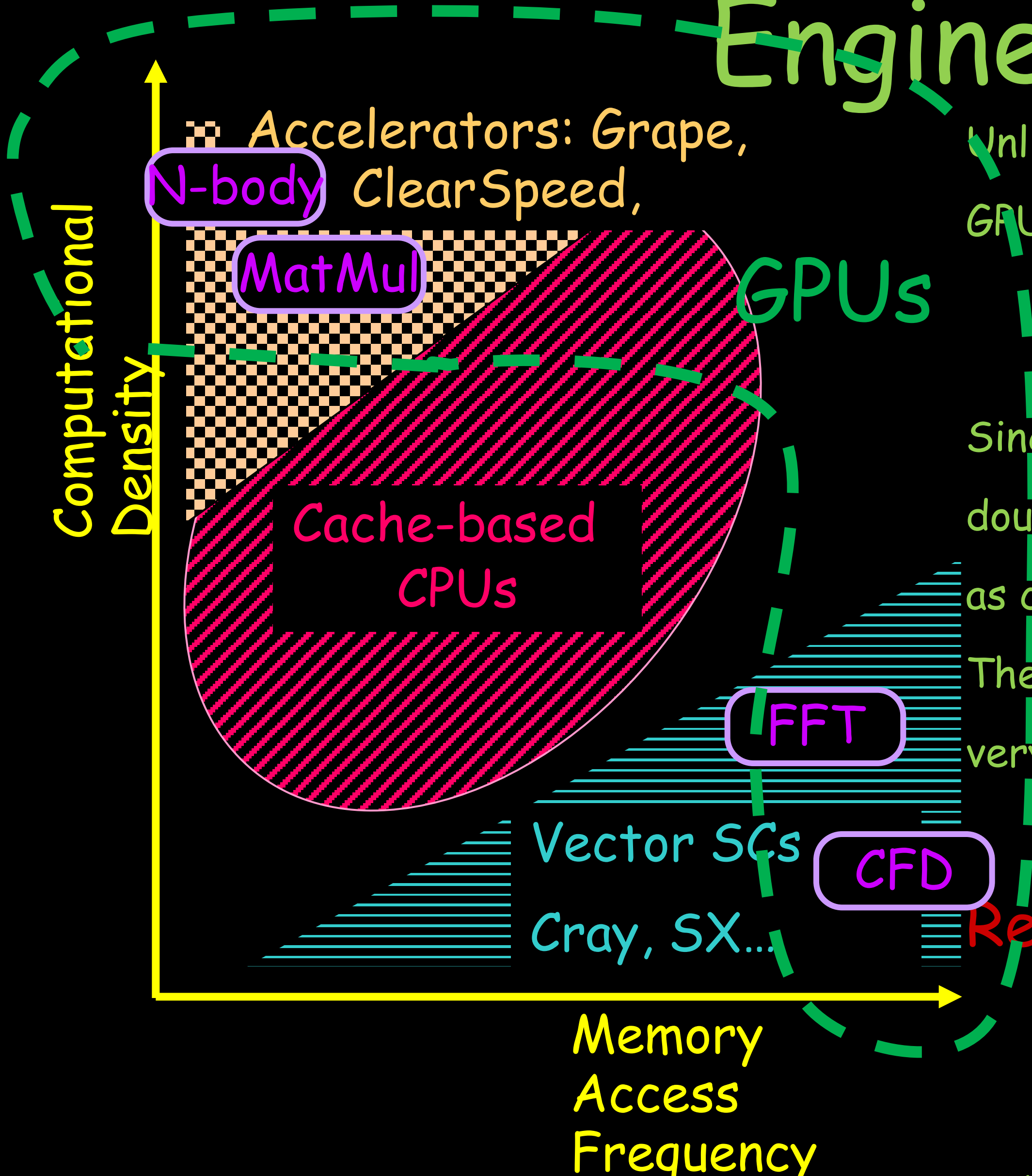
- Fine-Grain Multithreaded Vector Computing required for density, efficiency, and power

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?

GPUs as Commodity Vector Engines



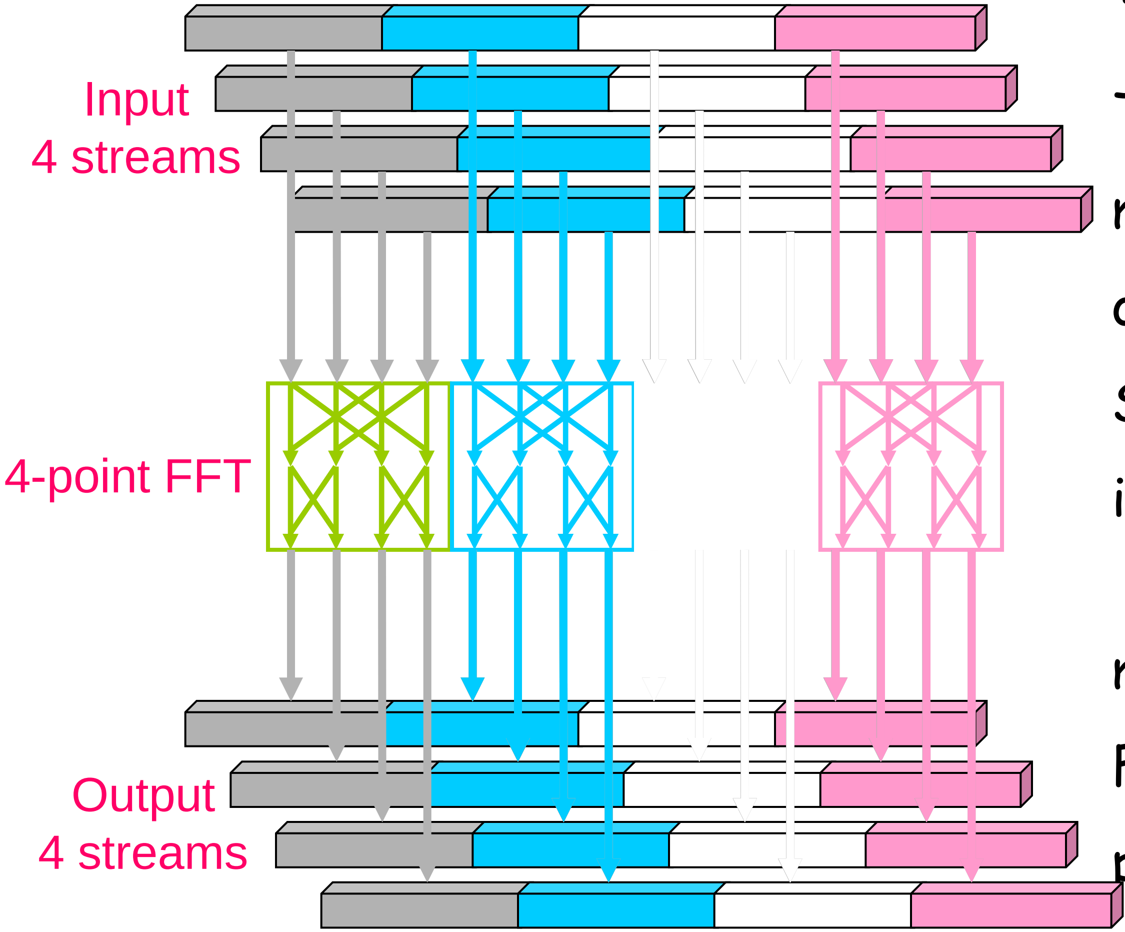
Unlike the conventional accelerators, GPUs have high memory bandwidth and dense

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?

High Performance 3-D FFT on NVIDIA CUDA GPUs [Nukada, Matsuoka et. al. SC08] multi-row FFT algorithm



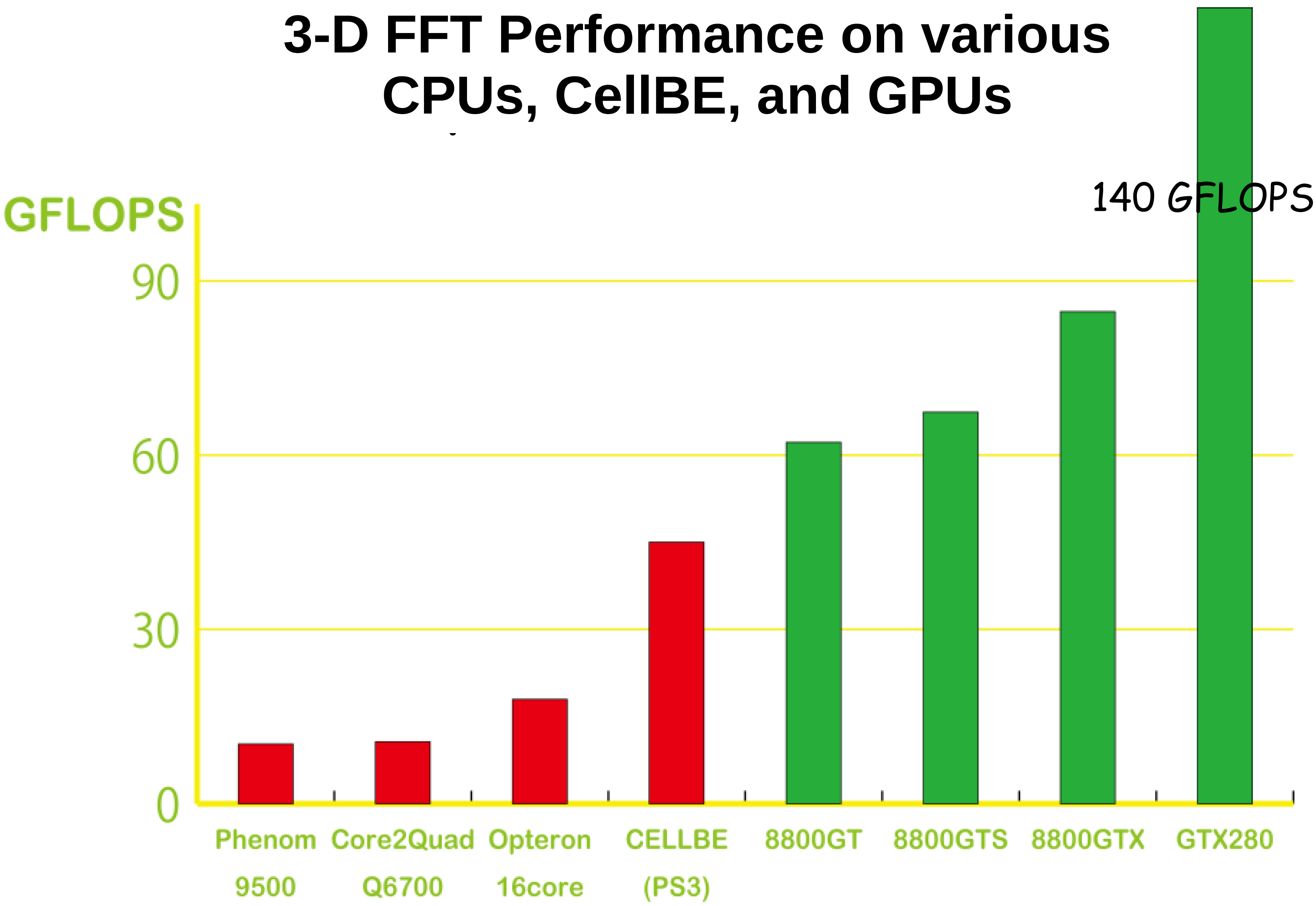
This algorithm accesses multiple streams, but each of them is successive.

Since each thread compute independent set of small FFT,

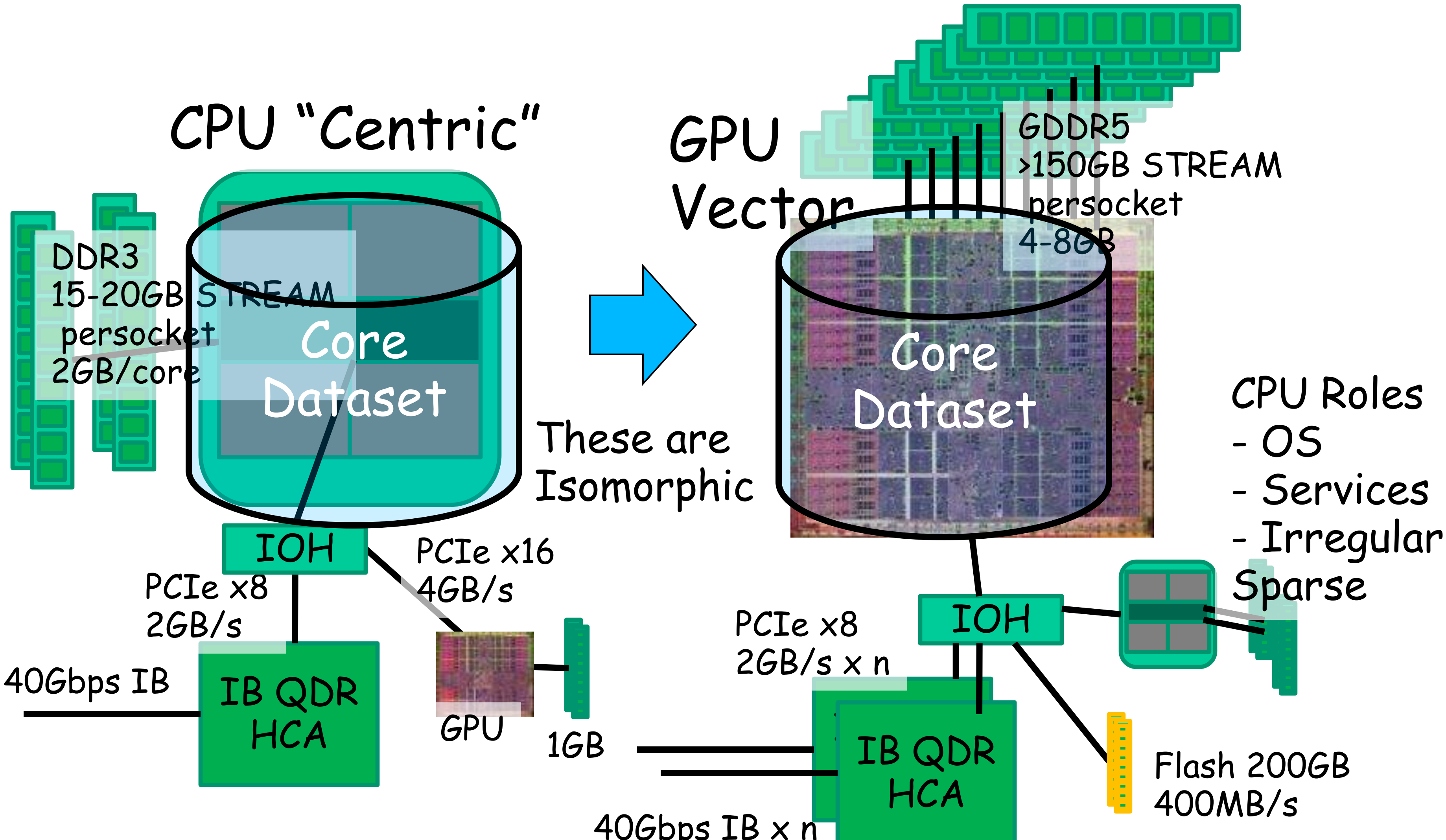
many registers are required.

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

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



From CPU Centric to GPU Centric Nodes for Scaling



TSUBAME 1.2 Experimental Evolution (Oct. 2008)

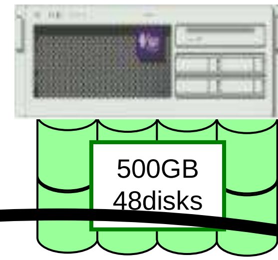
The first "Petascale" SC in Japan

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

10Gbps+External NW

Unified Infiniband network

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

10,000 CPU Cores
300,000 SIMD Cores
> 20 Million Threads
~900TFlops-SFP, ~170TFlops-DFP
80TB/s Mem BW (1/2 ES)

Sun x4600 (16 Opteron Cores)
32~128 GBytes/Node
10480core/655Nodes
21.4TeraBytes
50.4TeraFlops
OS Linux (SuSE 9, 10)
NAREGI Grid MW

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

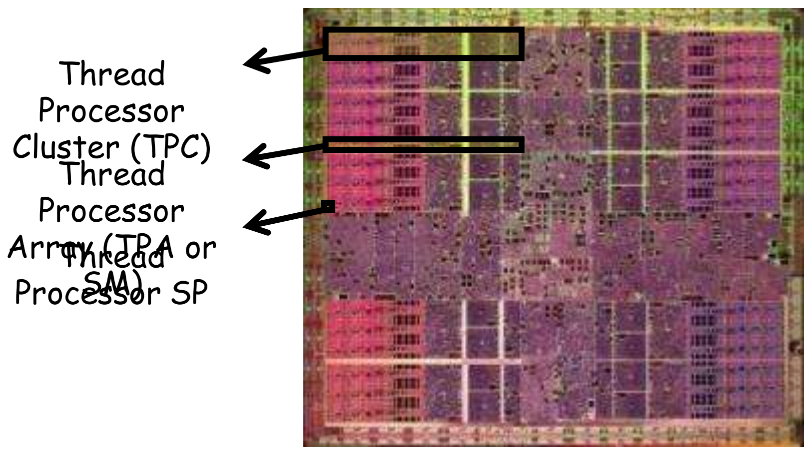
PCI-e

ClearSpeed CSX600
SIMD accelerator
648 boards,
52.2TeraFlops
SFP/DFP

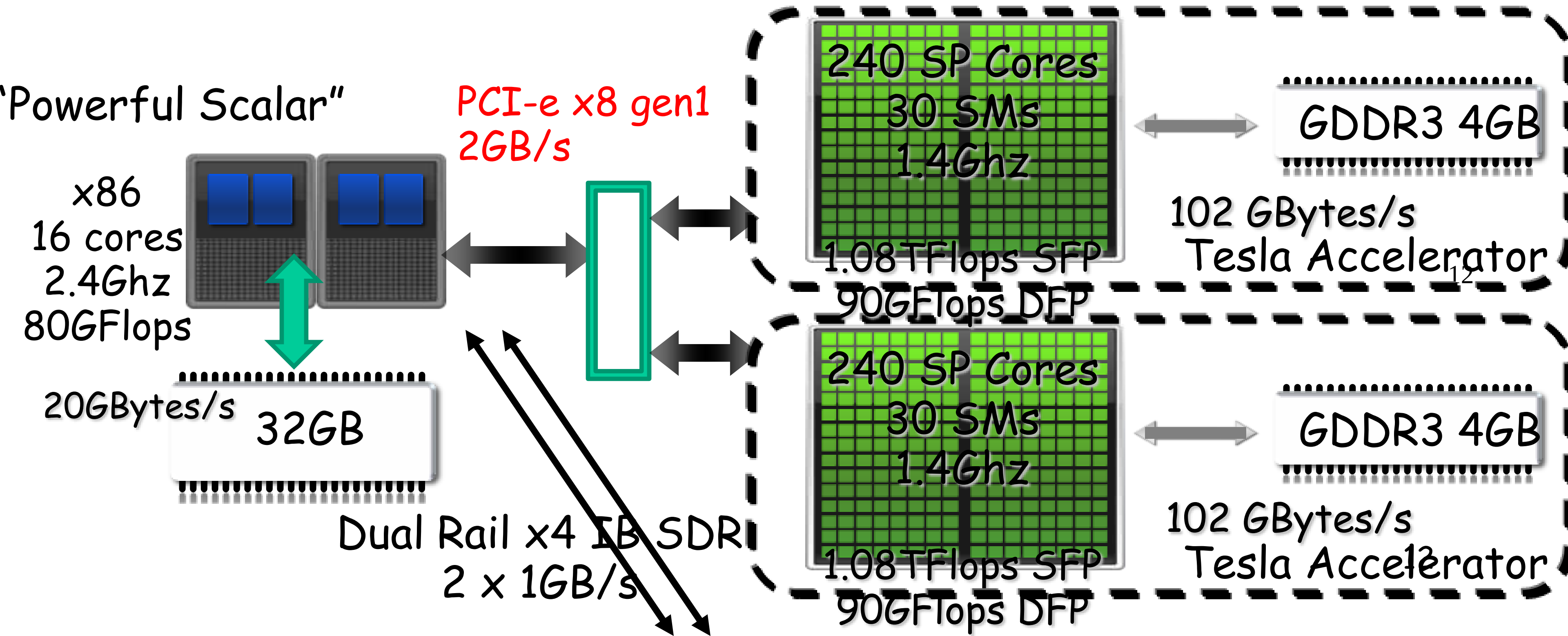
170 Nvidia Tesla 1070, ~680 Tesla cards
High Performance in Many BW-Intensive Apps
10% power increase over TSUBAME 1.0

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

TSUBAME 1.2 Node Configuration

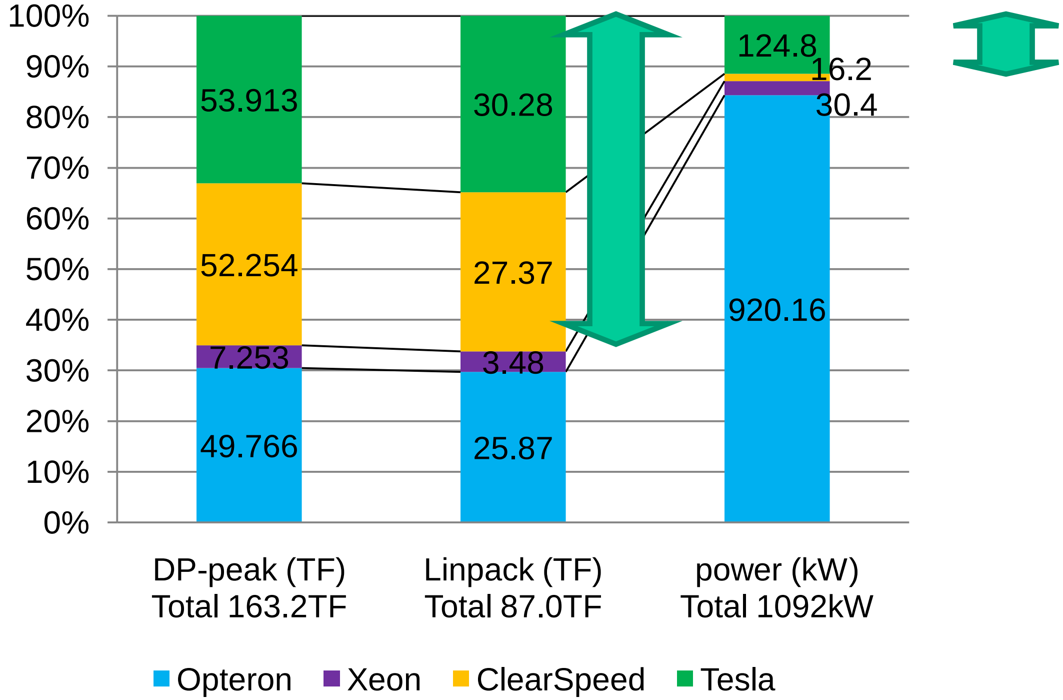


nVidia Tesla T10: 55nm, 470m2,
1.4billion transistors
>1TF SFP
90GF DFP



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 (!)

(Faster Than) Real-time Tsunami Simulation

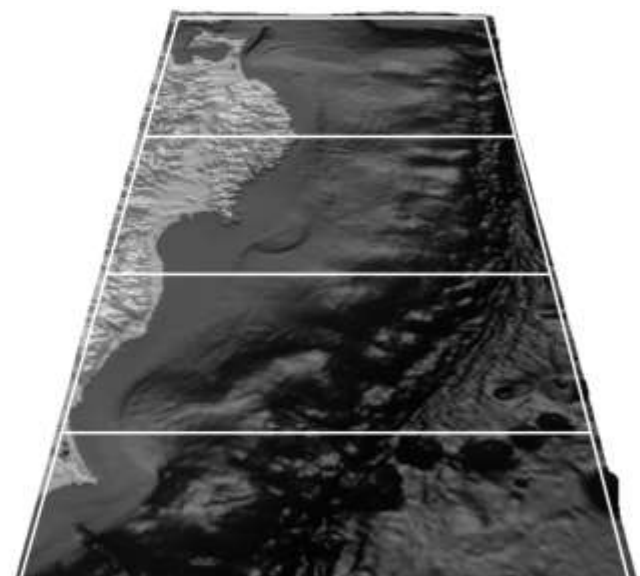
(Prof. Takayuki Aoki, Tokyo Tech.)

ADPC : Asian Disaster Preparedness Center

Early Warning System:

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

high accuracy



8 GPU 400km×800km
(100m mesh)

Shallow-Water Equation

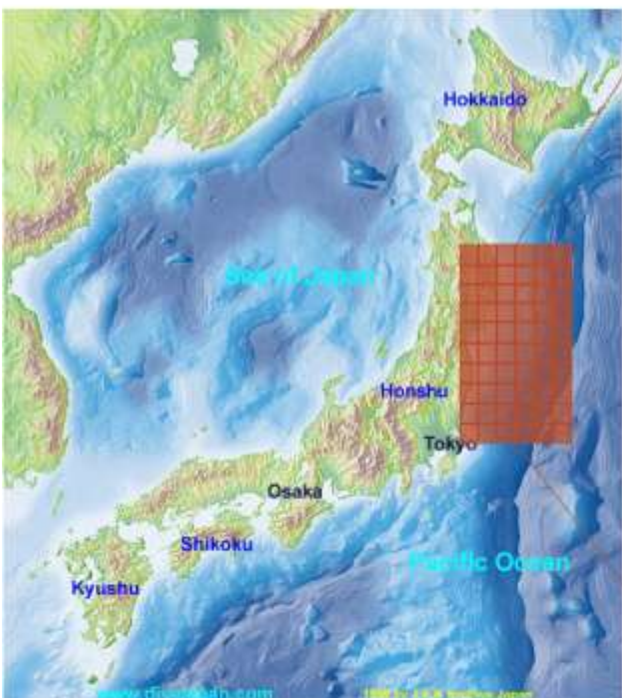
Conservative Form:

Assuming hydrostatic balance in the vertical direction,

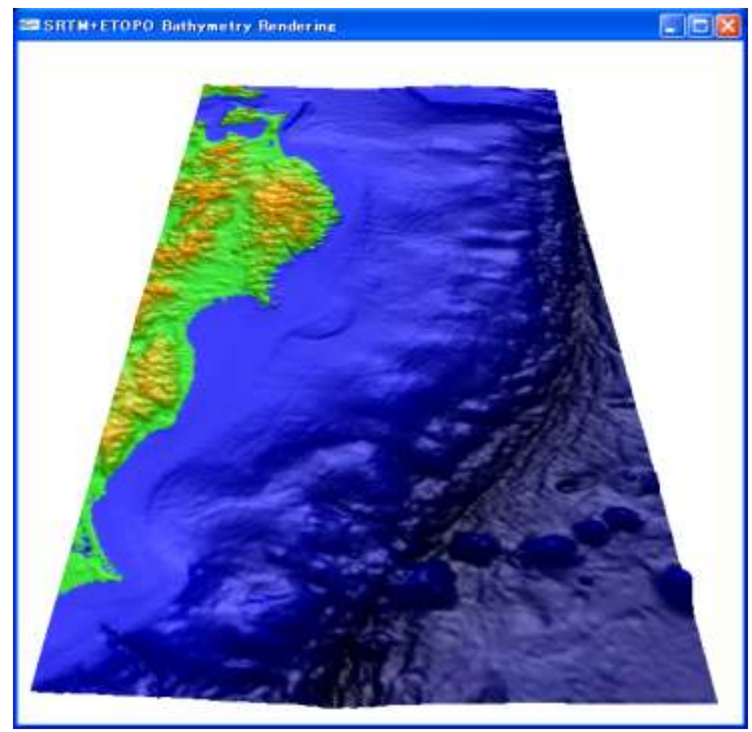
3D → 2D equation

Tsunami Prediction of Northern Japan Pacific Coast

- Bathymetry



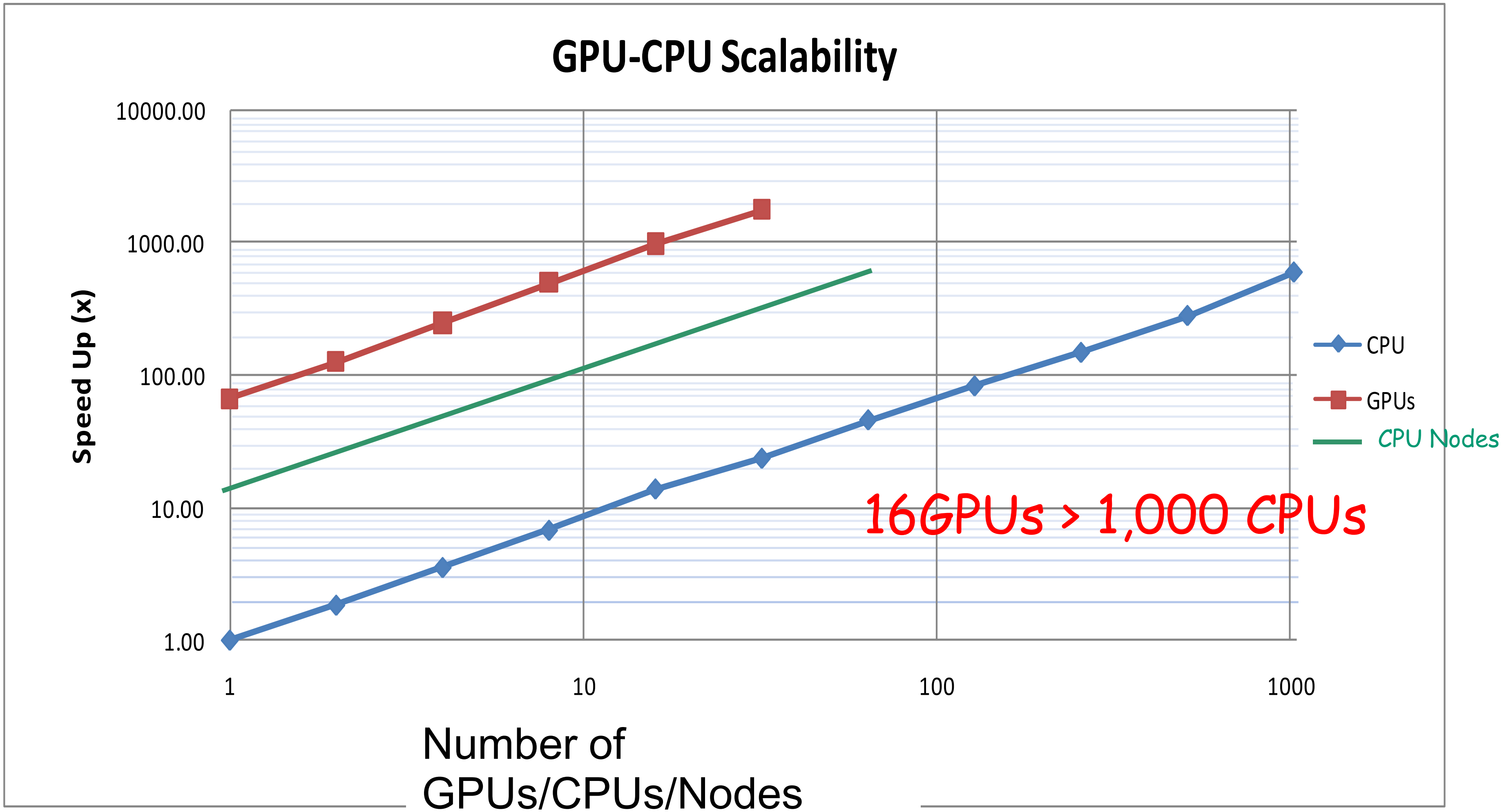
- Grid Size
  4096x8192
- Latitude
  $N35° - N42°$
- Longitude
  $E140°30' - E144°18'$
- Length
  370km x 740km

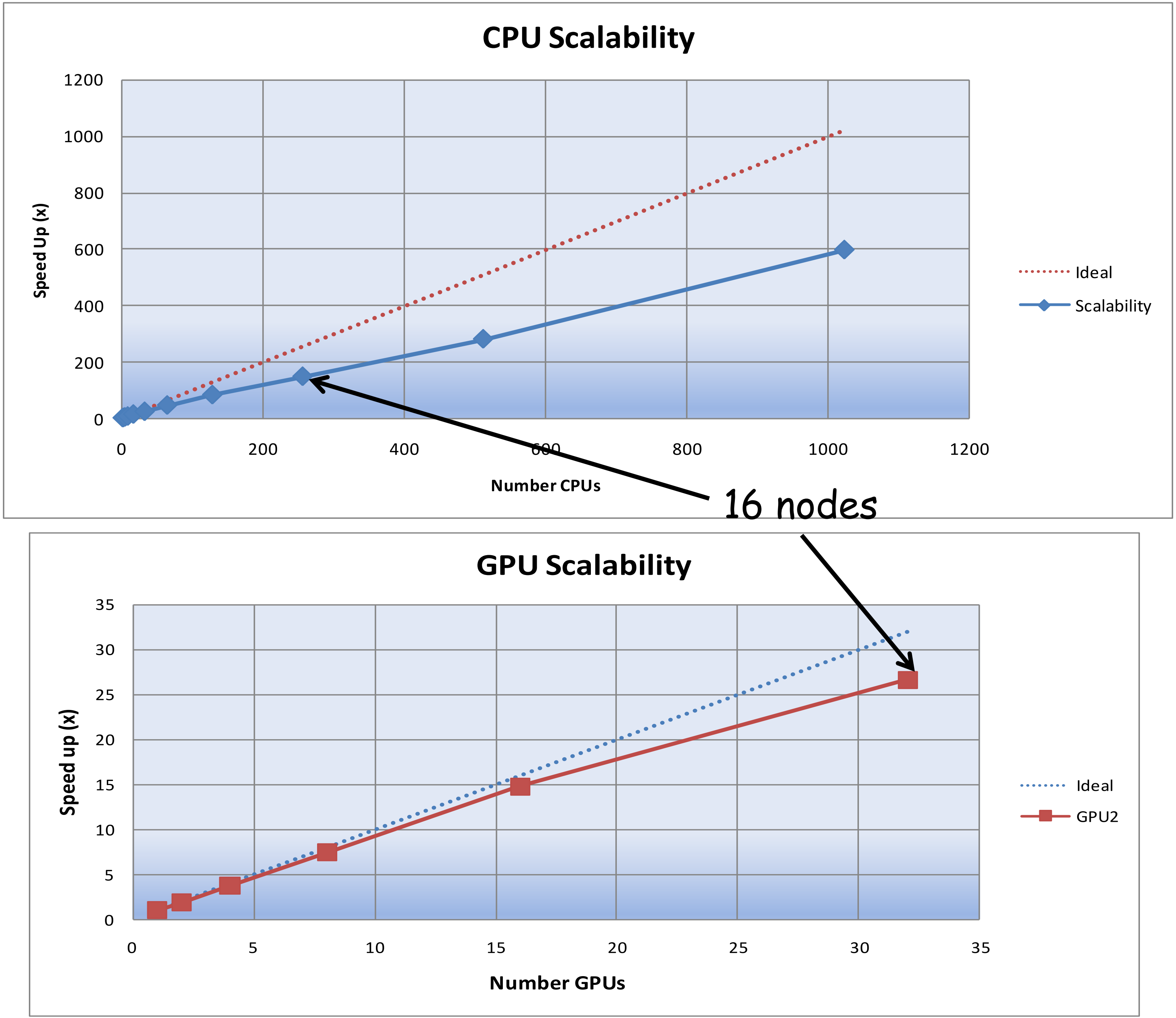


Multi-node CPU/GPU Comparison

*** Strong Scaling ***

- Results on TSUBAME1.2



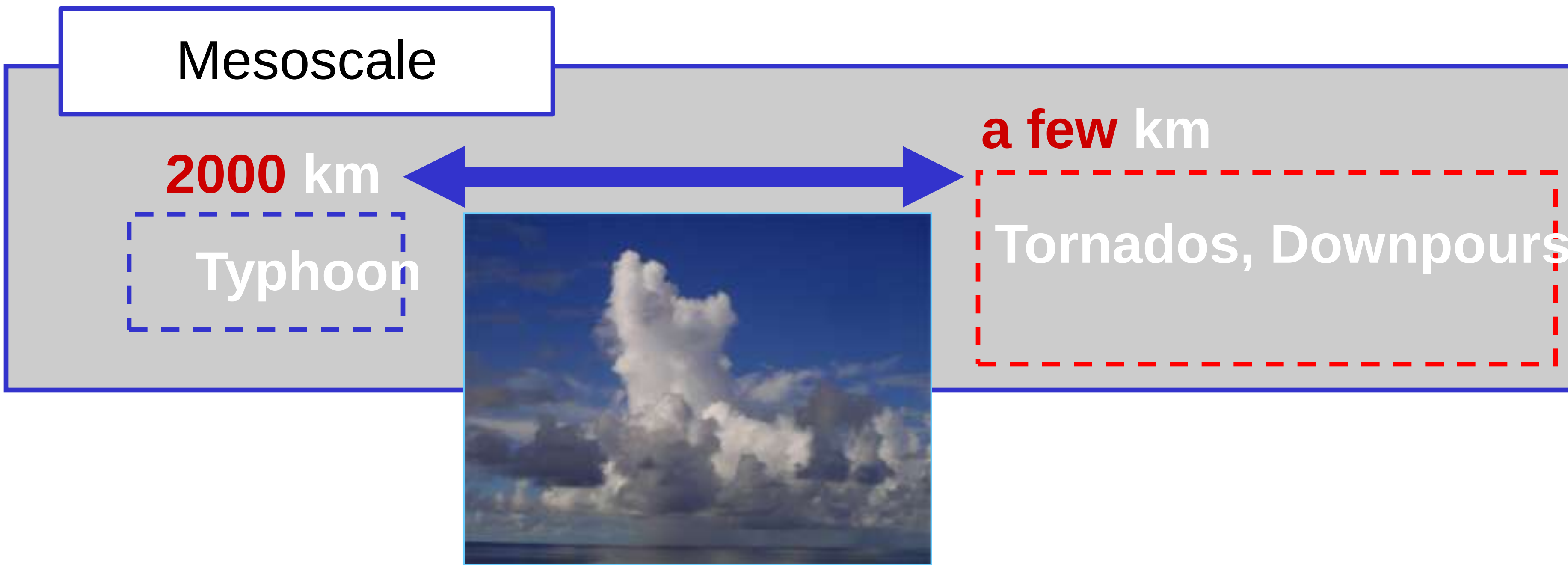

CPU Scalability
Speed Up (x)
1200
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
Number CPUs
Ideal
Scalability
16 nodes
GPU Scalability
Speed up (x)
35
30
25
20
15
10
5
0
0
5
10
15
20
25
30
35
Number GPUs
Ideal
GPU2

Next Gen Weather Forecast

Mesoscale Atmospheric Model:

Cloud Resolution: 3-D non-static

Compressible equation taking consideration of sound waves.



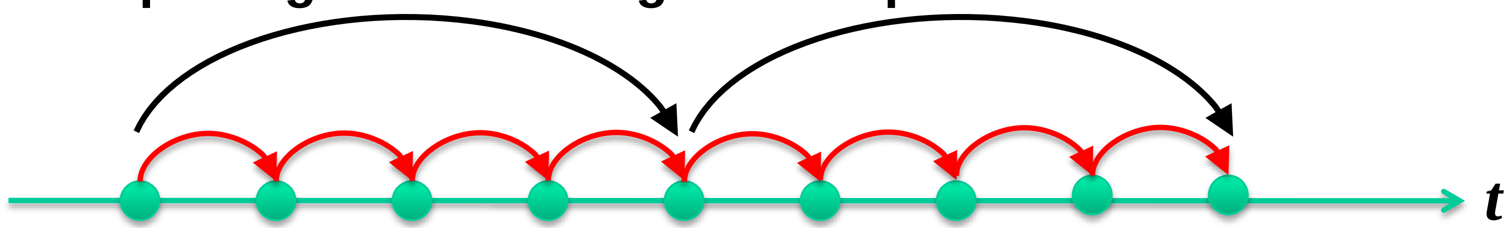
GPU enabling ASUCA

- **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

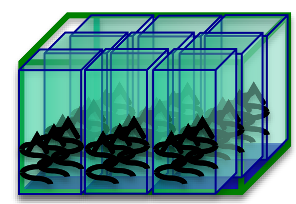


u, v (~ 100 m/s), w (~ 10 m/s) **<<** sound velocity (~300/ms)

HEVI **(Horizontally explicit Vertical implicit) scheme**

Horizontal resolution ~ 1 km

Vertical resolution ~ 100 m

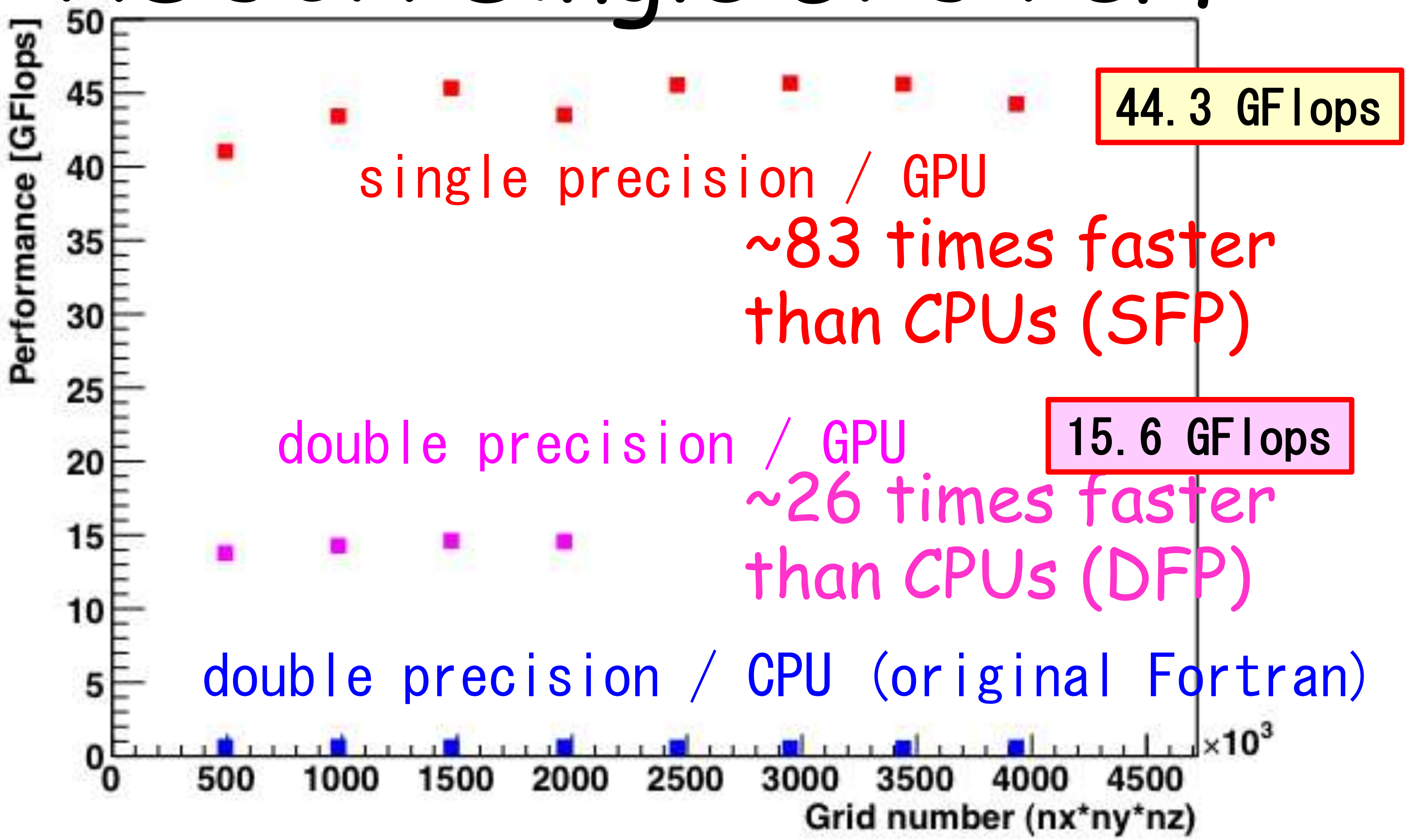


1-D Helmholtz equation (like Poisson eq.) ⇨ sequential process

Entire "Core" of ASUCA now ported to GPU (~30,000 lines)

By Prof. Aoki Takayuki's team at Tokyo Tech.

ASUCA Single GPU Perf

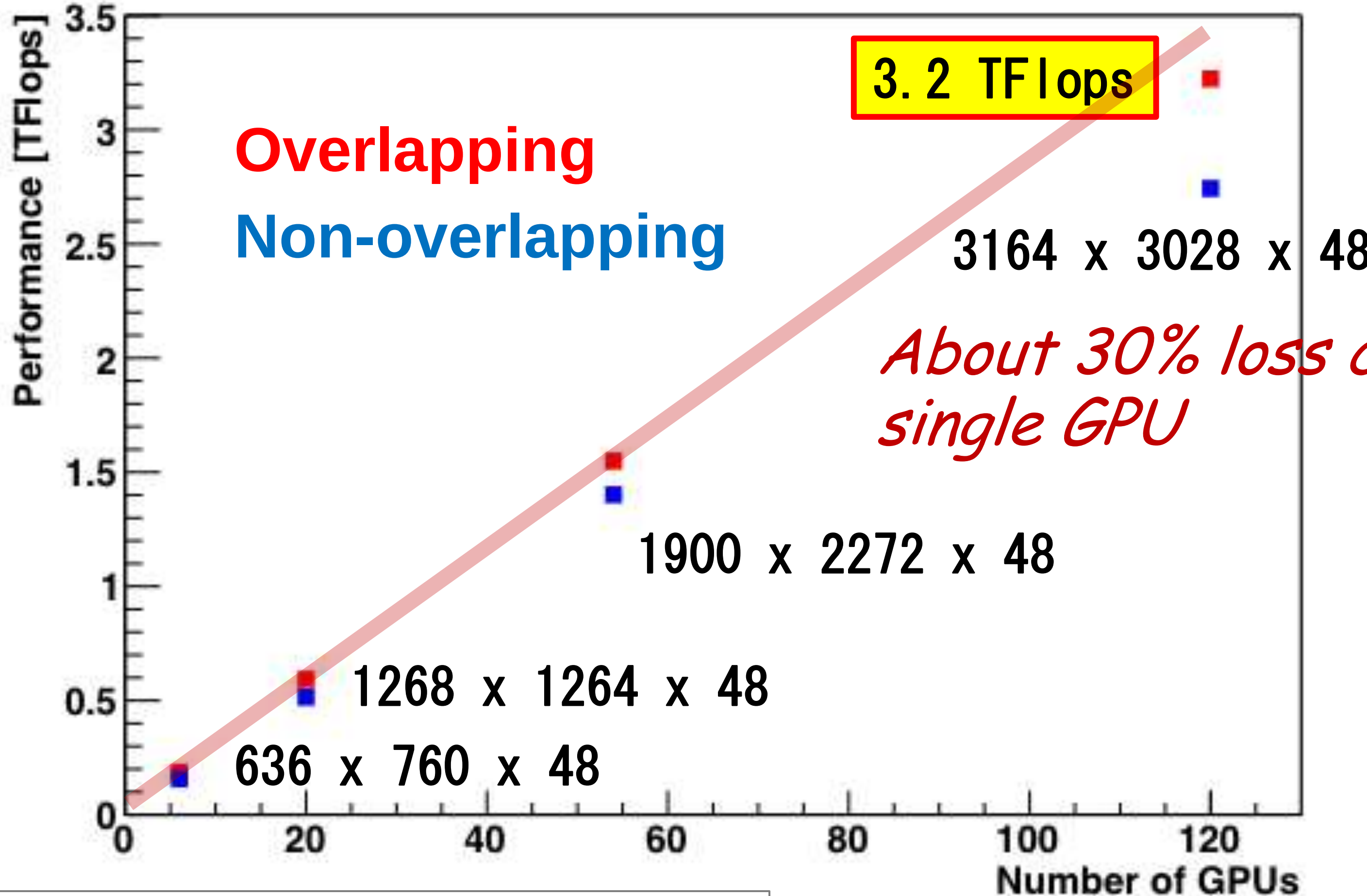


Mountain Wave Test

NVIDIA Tesla S1070 card

320 x 256 x 64

ASUCA Multi GPU Performance (up to 120 GPUs)



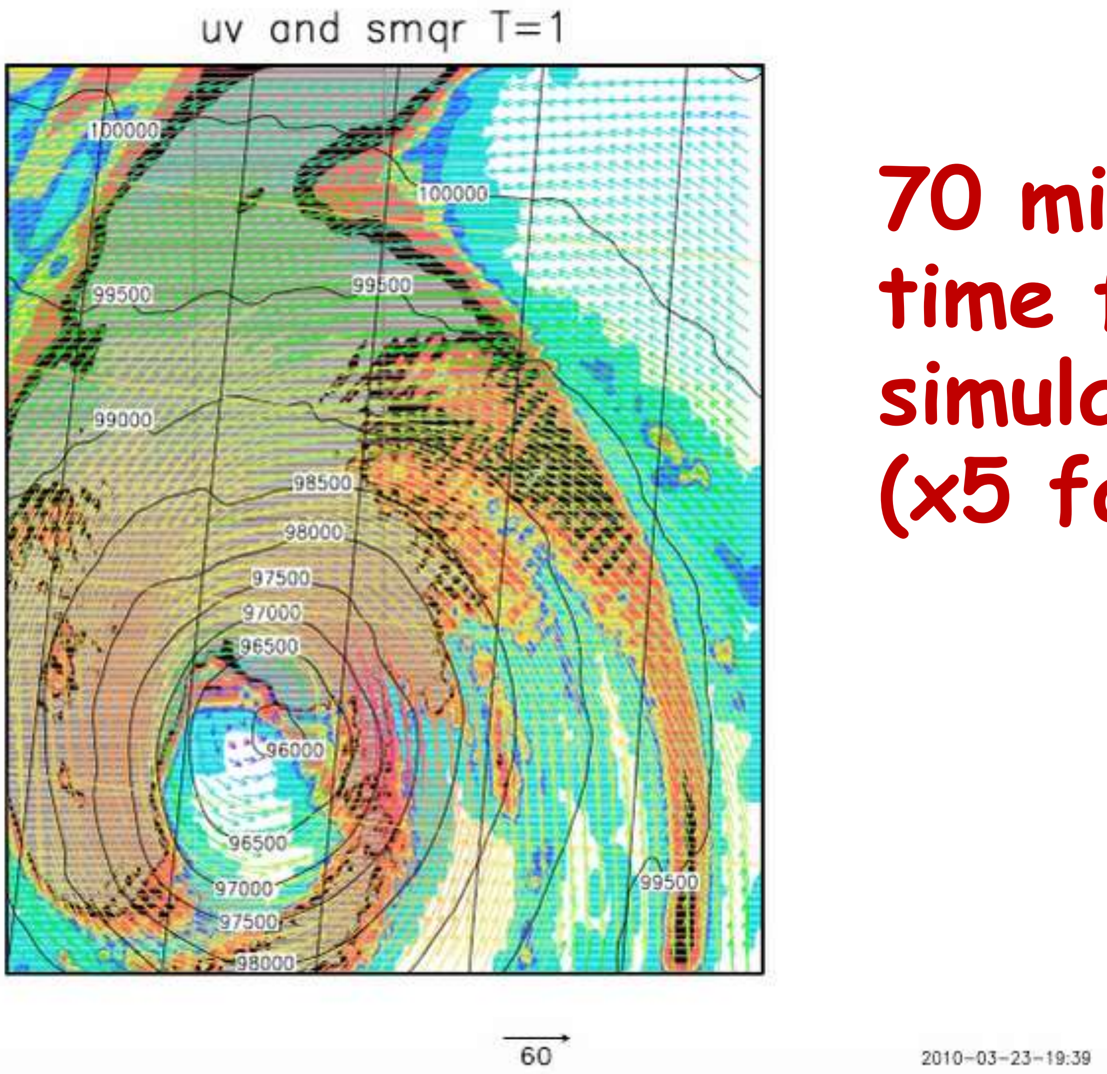
About 30% loss c.f. single GPU

Mountain Wave Test in single precision

NVIDIA Tesla S1070 on TSUBAME

600 GPU in April
15 Teraflops

ASUCA Typhoon Simulation
2km mesh 3164×3028×48



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

Game Changing Cluster Design for Petaflops 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

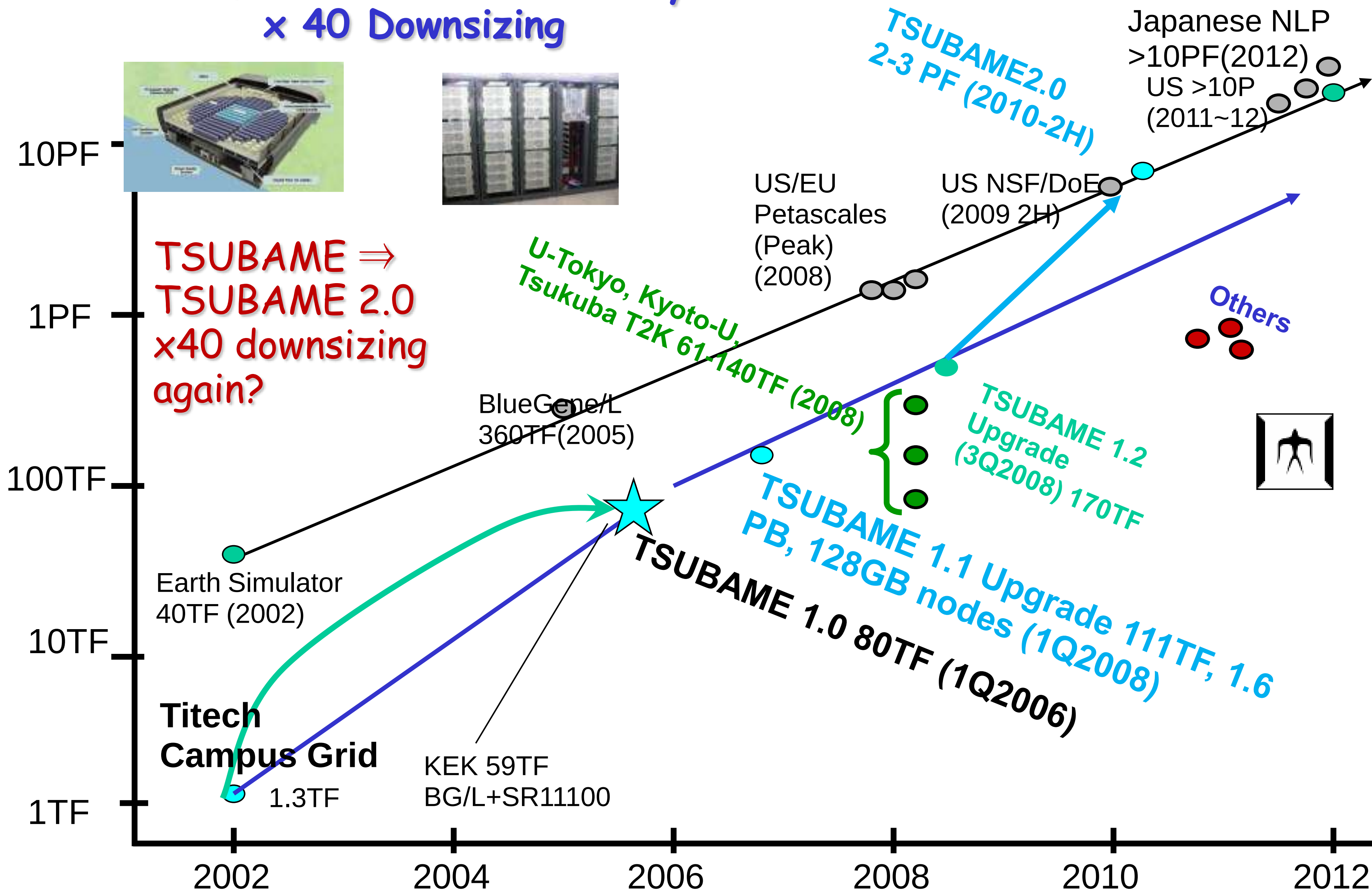
    GPU
  - High bandwidth, low latency, full bisection network for intra-node scalability
  - High bandwidth node memory and I/O channels to accommodate all of above
  - Node-wise non-volatile, high bandwidth silicone storage for scalable storage I/O
  - Software layers for attaining bandwidth, fault tolerance, programmability, and low power
  - Such an architecture is the basis towards Exascale

Highlights of TSUBAME 2.0 Design (Oct. 2010)

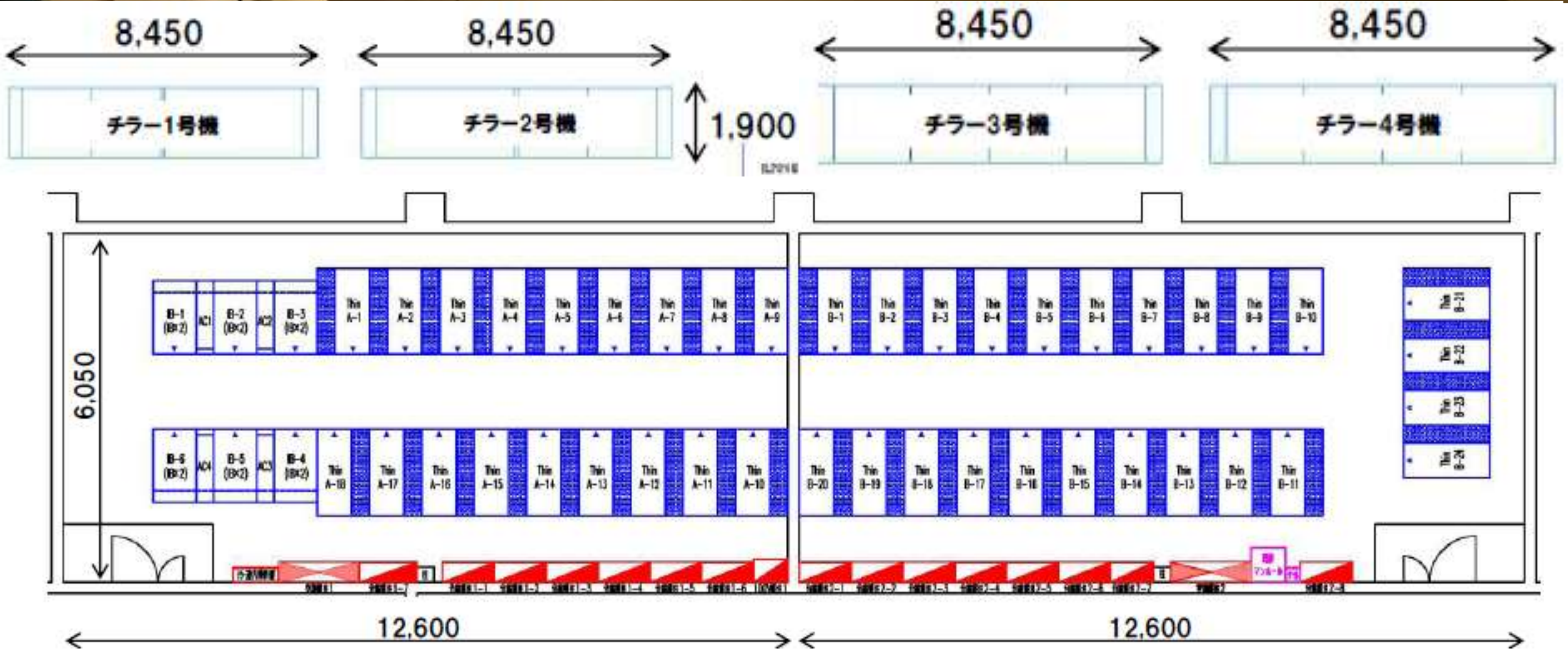
- 2-3 PF Next gen multi-core x86 + next gen GPU
  - ~100,000 total CPU and GPU "cores", 10-100 million "threads"
  - Massively Parallel Vector multithreading for **high bandwidth**
- 0.5~1 Petabyte/s aggregate mem BW,
  - Effective 0.3-0.5 Bytes/Flop, restrained memory capacity
- Multi-Rail IB-QDR BW, full bisection BW (Fat Tree)
  - 200Tbits/s, Likely fastest in the world, still scalable
- Flash/node, ~200TB (1PB in future), ½~1TB/s I/O BW
  - 6-7 PB IB attached HDDs, 15PB Total HFS incl. LTO tape
- Low power & efficient cooling, comparable to TSUBAME 1.0 (~1MW) – PUE < 1.3?
- Virtualization and Dynamic Provisioning of Windows HPC + Linux, job migration, etc.

TSUBAME 2.0 Performance

Earth Simulator ⇒ TSUBAME 4years x 40 Downsizing

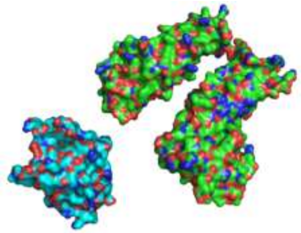


TSUBAME2.0 レイアウト（サンプル）



TSUBAME2.0 Estimated Performances

- > 1.4 PFlops Linpack [IEEE IPDPS 2010]
- ~1PF 3D Protein Docking (Node 3-D FFT)
  - > x2 ORNL Jaguar



- 100-150 TFlops ASUCA Forecast
  - C.f. 50 TFlops NCAR WRF on ORNL Jaguar
- Top-level HPC-Challenge Performances
- QCD? Lattice-Boltzmann? FEM? Genomics? MD/MO? Search?

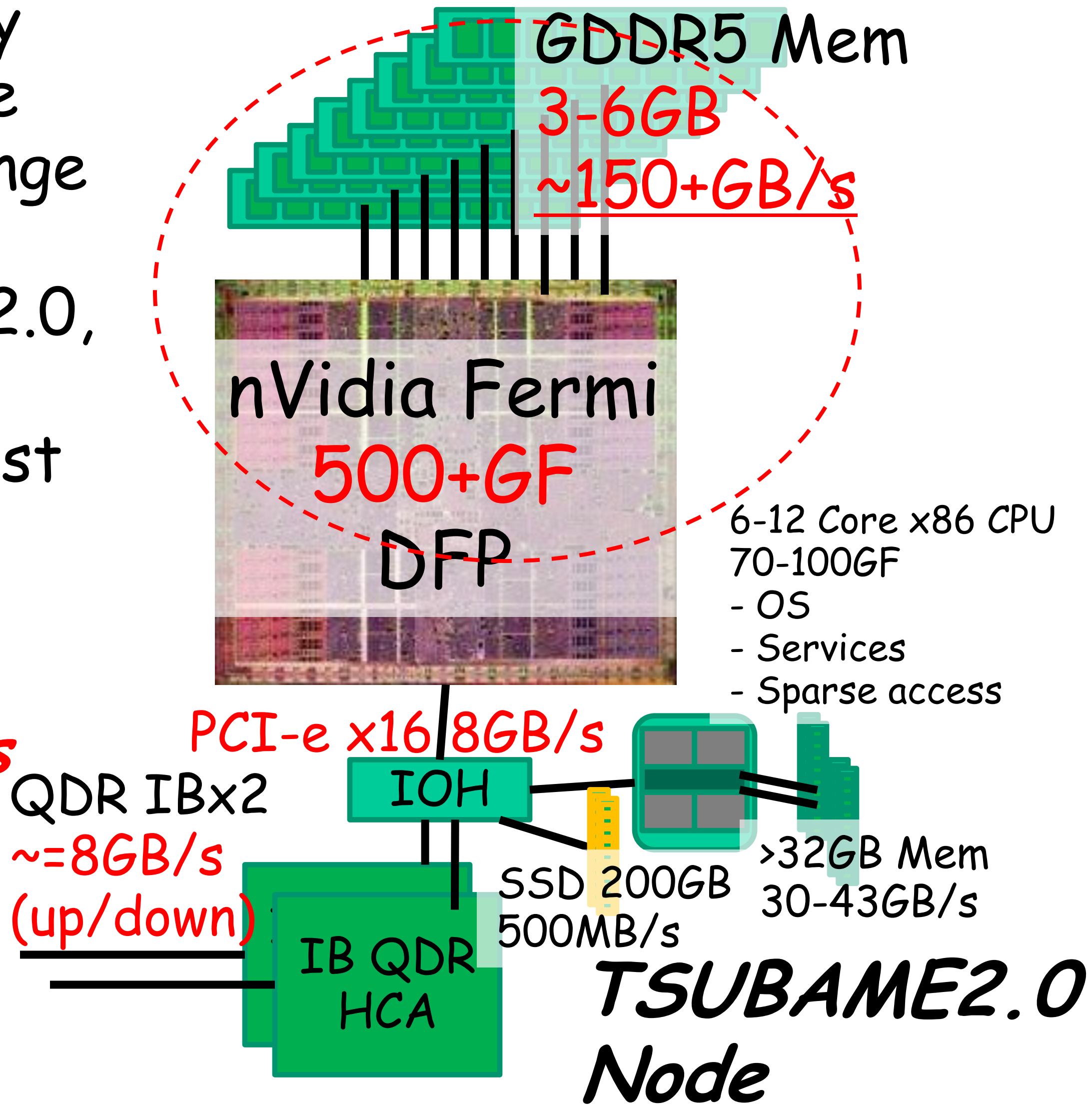
The "IDEAL TSUBAME2.0"

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

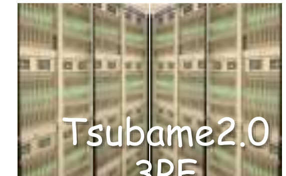
- In the REAL TSUBAME2.0, will have to compromise various parameters for cost and other reasons

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

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

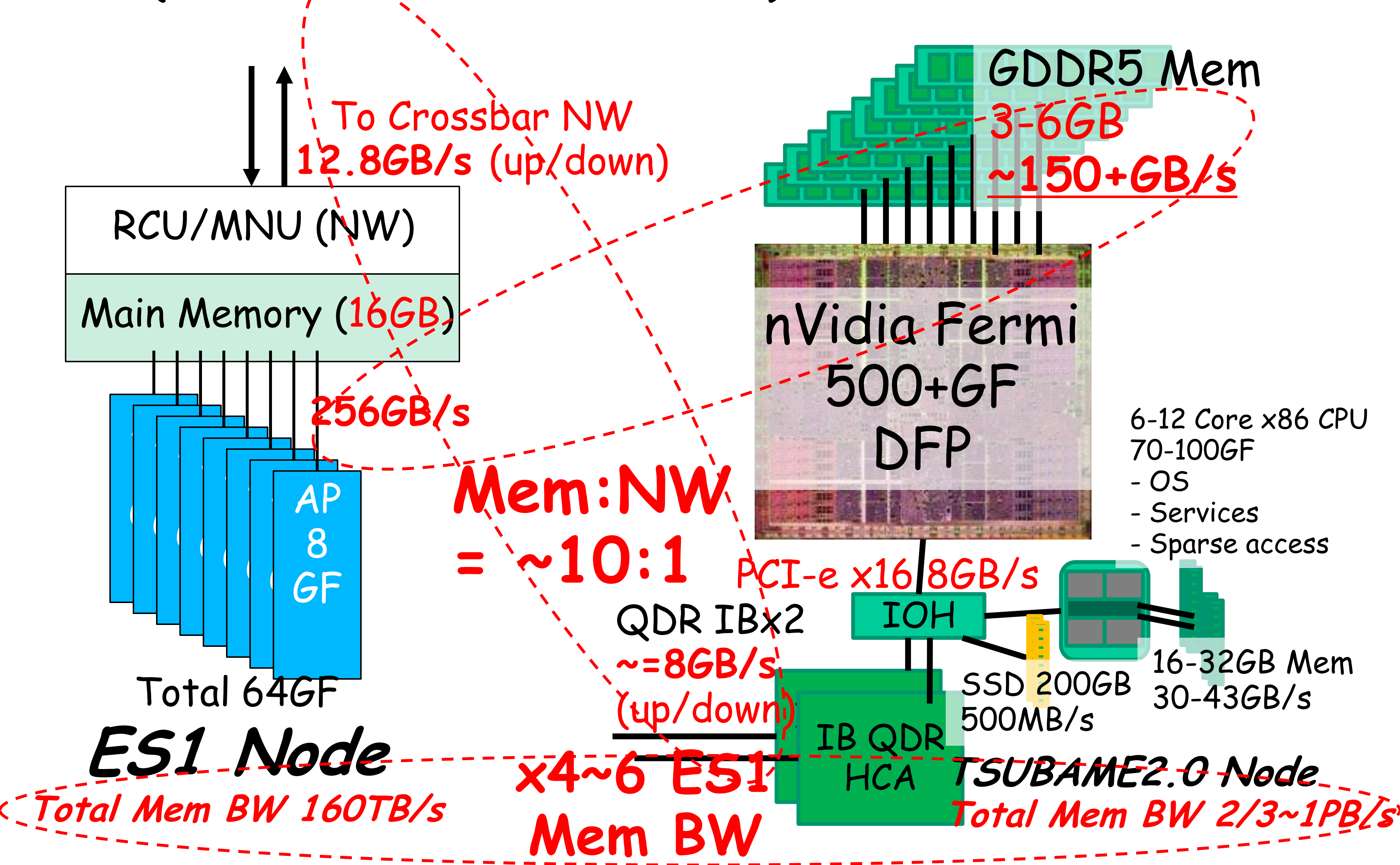


TSUBAME2.0 (2010) vs. Earth Simulator1 (ES) (2002) vs. Japanese 10PF NLP @Kobe (2012)

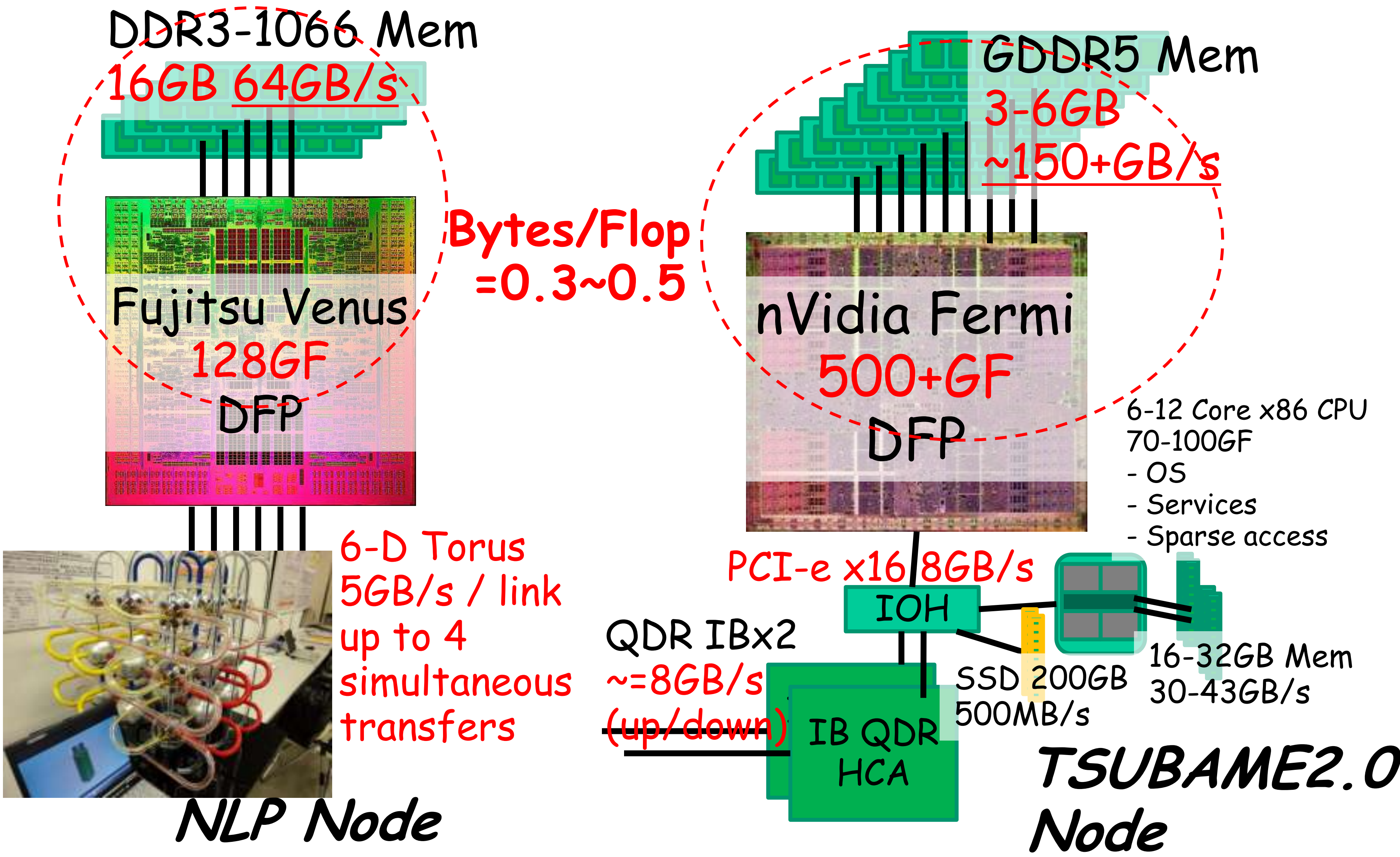


200m2

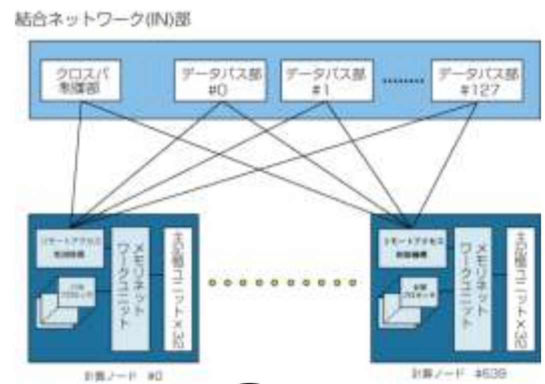
The "IDEAL TSUBAME2.0" (w/o cost constraints) node vs. ES1 node



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

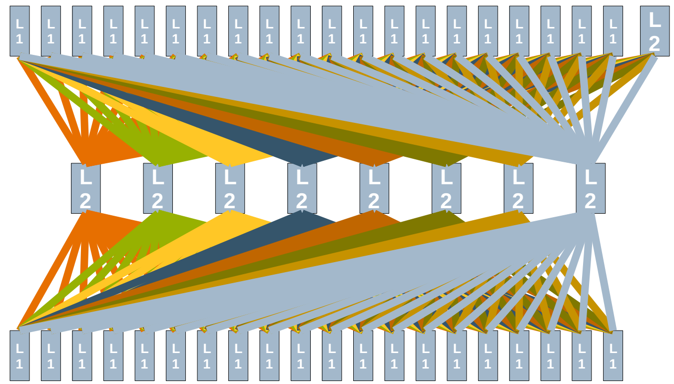


Comparing the Networks



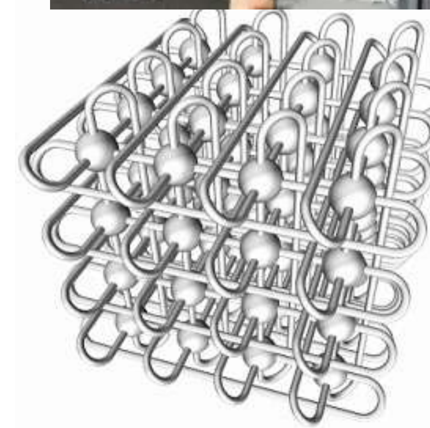
ES1

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 ~4 times faster on full machine (weak scaling)**

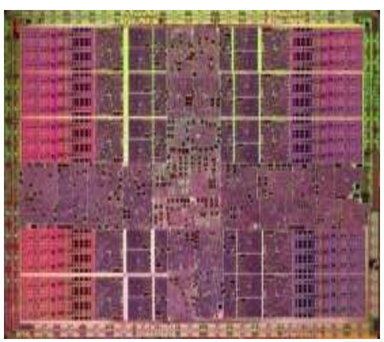
TSUBAME2.0 to 3.0 and beyond

- Straightforward scaling of TSUBAME2.0 Architecture in 2012Q4~2013Q1 by x10
  - Size: Full bisection NW to ~5000 nodes (x3~4)
  - "Moore" speeup: ~x3
    - Node FLOPS x3, Mem BW x2, I/O BW x2~3
    - Network BW x2.5 (40Gbps => 100Gbps)
    - Disk and Flash capacity > x3
- 20 PF Linpack, 600m2, 3-4MW w/PUE <= 1.1
- ASUCA and Climate code at Petaflop or more

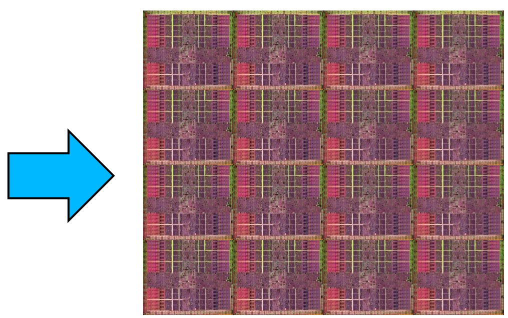
Scaling TSUBAME2.0 to Exaflop

- TSUBAME2.0: 40-45nm, 2~3PF, ~60 racks, 1.5MW = x320 scaling?
- x20 physical scaling now (1000 racks, 30MW)
  - >3000-4000m², 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.

But what about the network? 3-40,000 nodes?

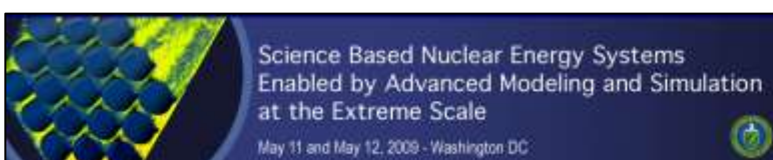
(From US DoE Exascale PPT by Rick Stevens@ANL)
Uncertainty quantification is critical and requires exascale resources.

EXASCALE RESOURCES

Large ensembles	***Embedded UQ***
Response surface	Adjoint enabled forward models
Posterior exploration	Data extraction from model
Finding least favorable priors	Local approximations, filtering
Bounds on functionals	Stochastic error estimation

"We need to be able to make quantitative statements about the predictability of regional climatic variables that are of use to society."

"computational techniques and needs complement the scientific areas that will be pursued with extreme scale computing. Examples include … verification and validation issues for extreme scale computations "

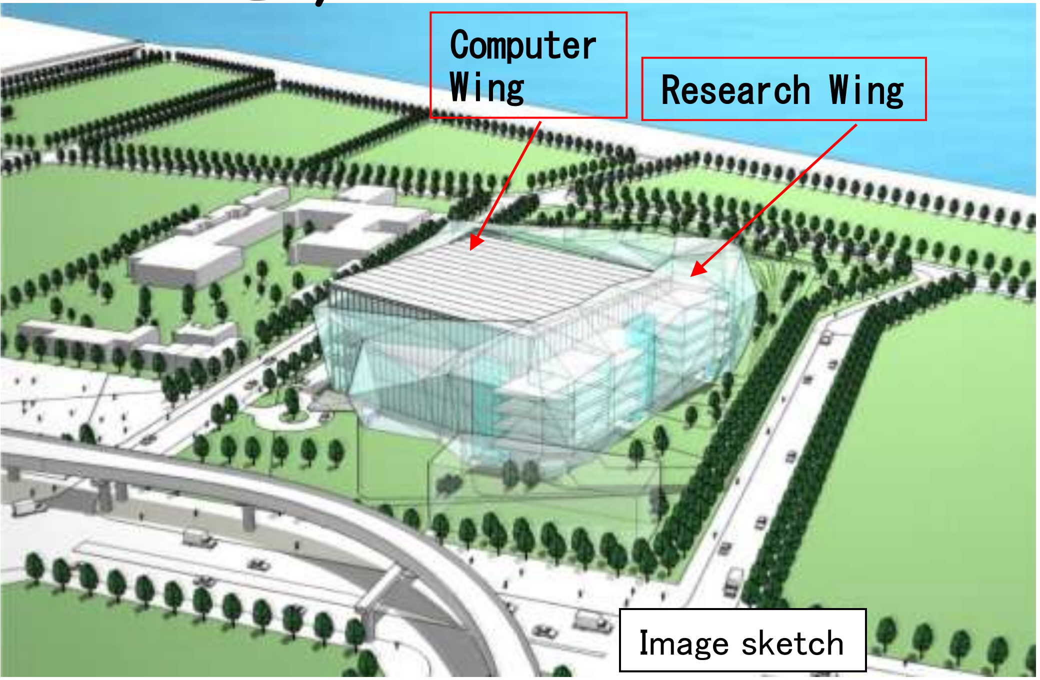


"scientists must create new suites of application codes, Integrated Performance and Safety Codes (IPSCs) that incorporate …integrated uncertainty quantification.."

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

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)

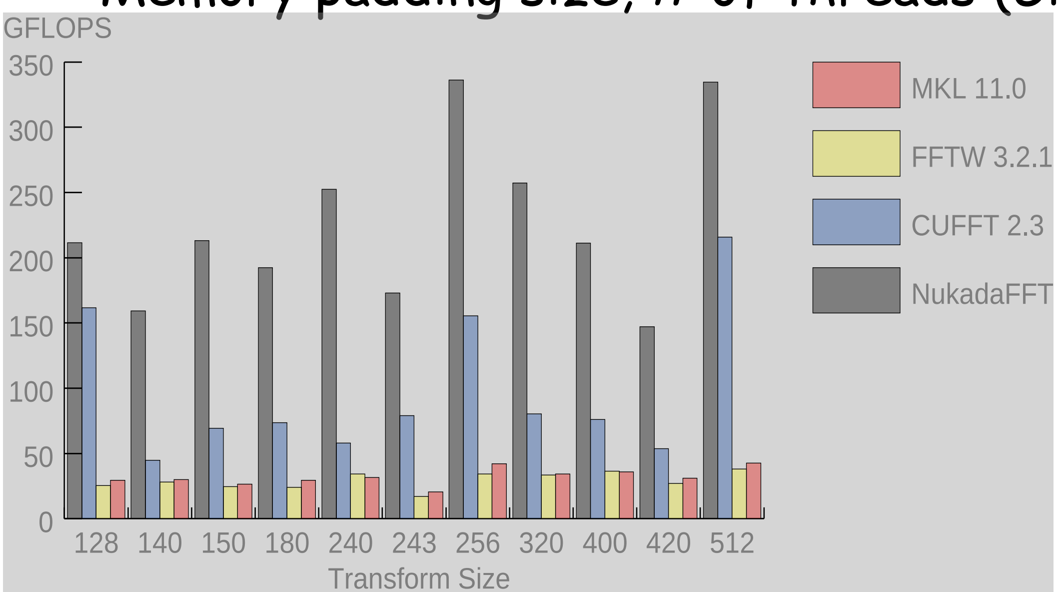
Auto-Tuning FFT for CUDA GPUs [ACM/IEEE SC09]

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)

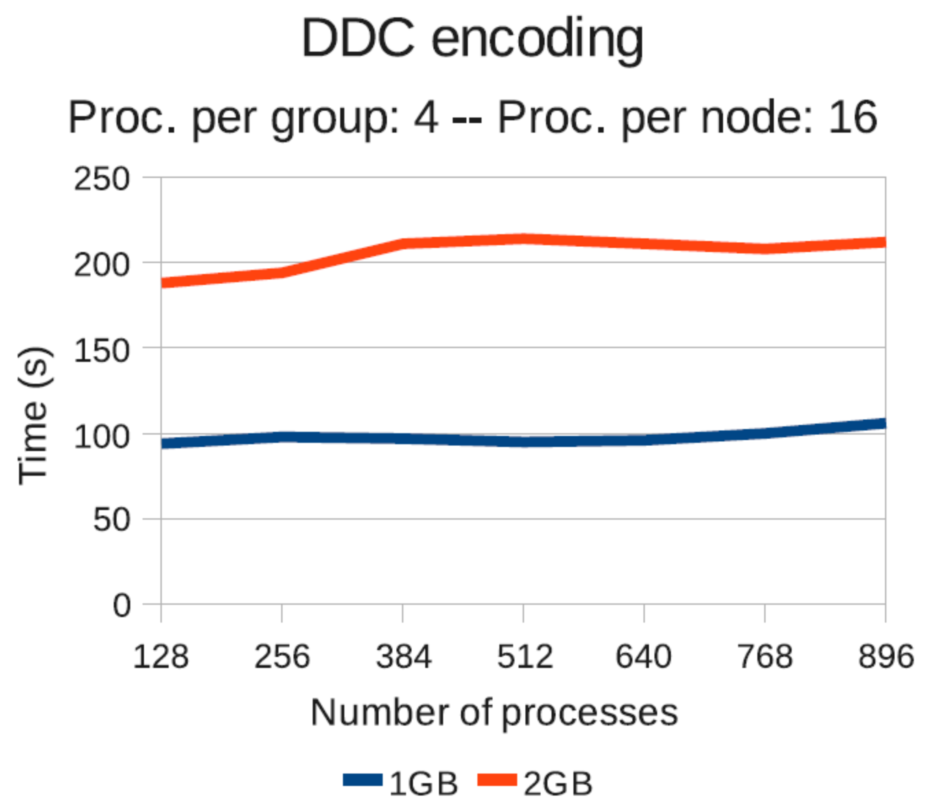


Our auto-tuned library is

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

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 syster*
- Exploit fast *local* I/O w/NVMs (aggr. TB/s) for checkpoints
- Group-based redundundy + RS encoding technique for *O(1) scalability*
- Combine with our ***ongoing work on checkpointing on GPUs***



Software (apps, algorithms, system) are the key to Exascale

- Many, many research issues, 一センター・一国では無理
  - GPU heterogeneity “overhead”
  - Memory and network BW “improvements”
    - Despite n^2 vs. n problem
  - Node-to-node latency
  - Fault Tolerance
  - Programming Models
  - Languages, Libraries, Tool Chains
  - Exascale algorithms

http://www.exascale.org/
筑波 10/19-21/2009
Oxford 4/12-13/2010
Maui in Sept. 2010

- 我が国も多くの計算機科学・計算科学の研究者が一体となって研究開発・国際貢献していく体制の確立が急務

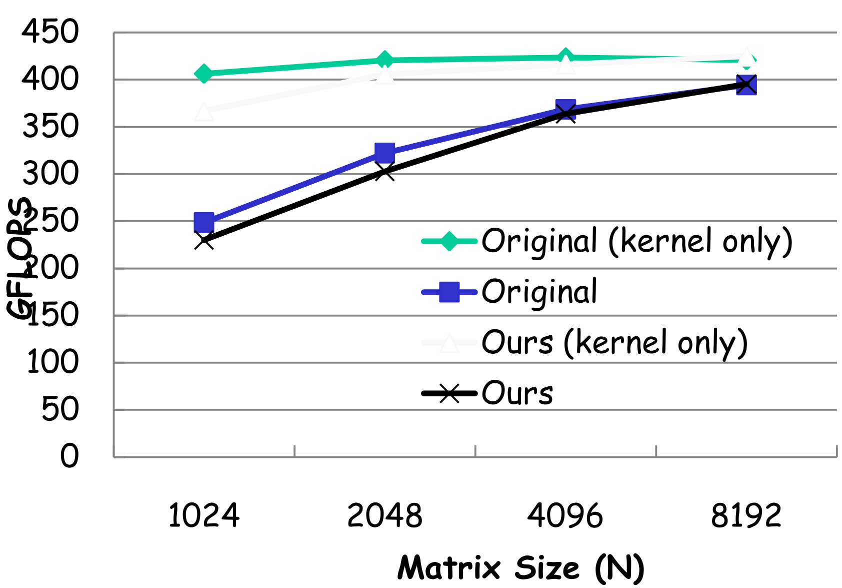
Software Framework for GPGPU Memory FT [IEEE IPDPS 2010]

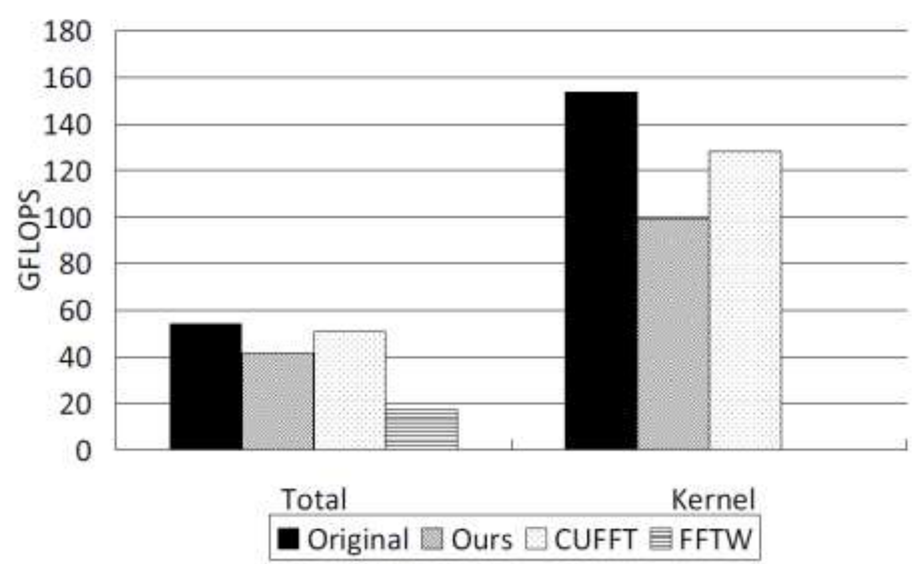
- Error detection in CUDA global memory + Checkpoint/Restart
- Works with existing NVIDIA CUDA GPUs

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





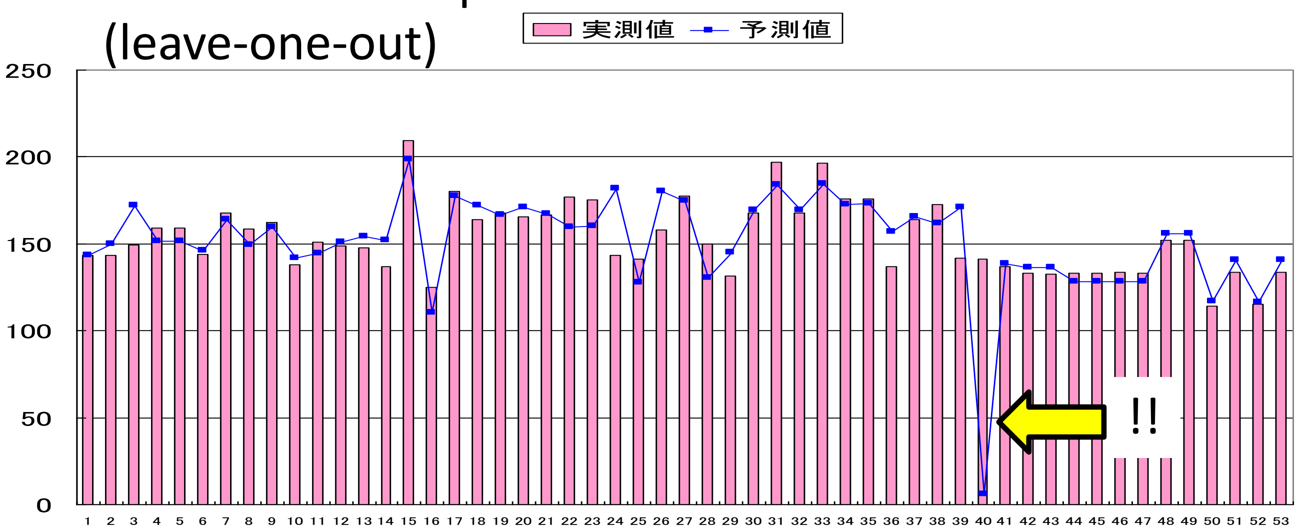
GPU Power Consumption Modeling

Employ learning theory to predict GPU power from performance counters

$$\text{GPU power} = \Sigma\, c_i \times p_i$$

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

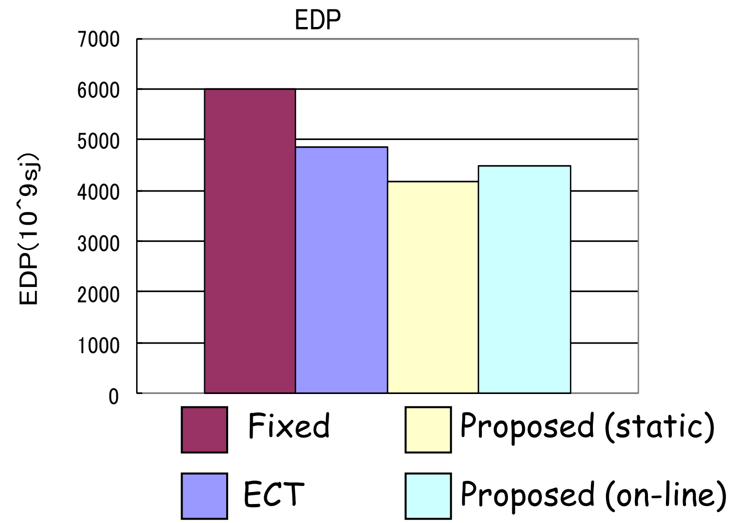
Low Power Scheduling in GPU Clusters

[IEEE IPDPS-HPPAC 09]

Objective: Optimize CPU/GPU Heterogeneous

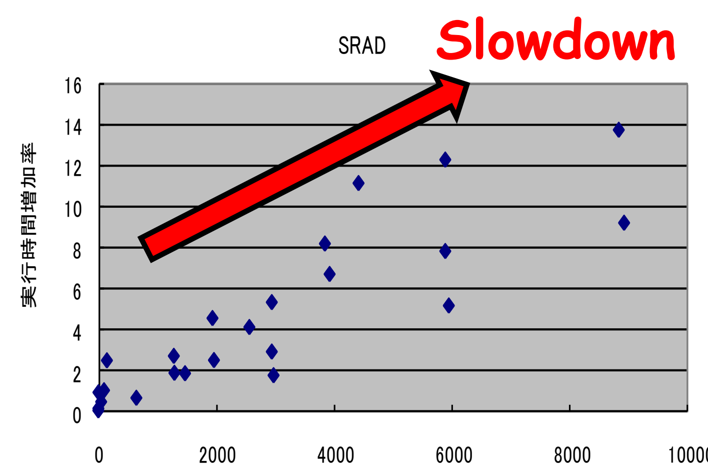
- Optimally schedule mixed sets of jobs executable on either CPU or GPU but w/different performance & power
- Assume GPU accel. factor (%) known

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.

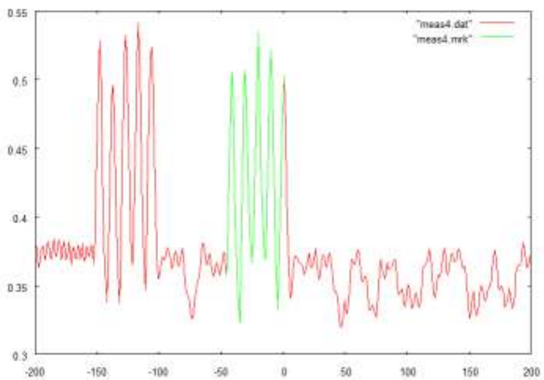


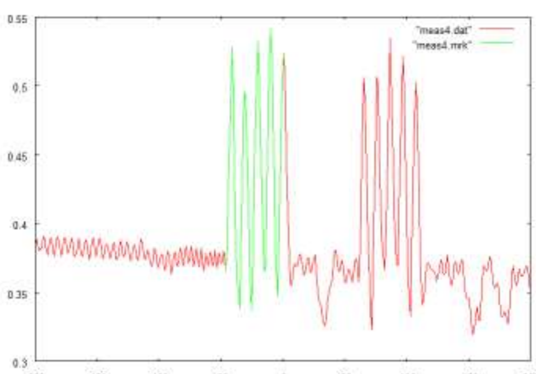
Background CPU process BW (MB/s)

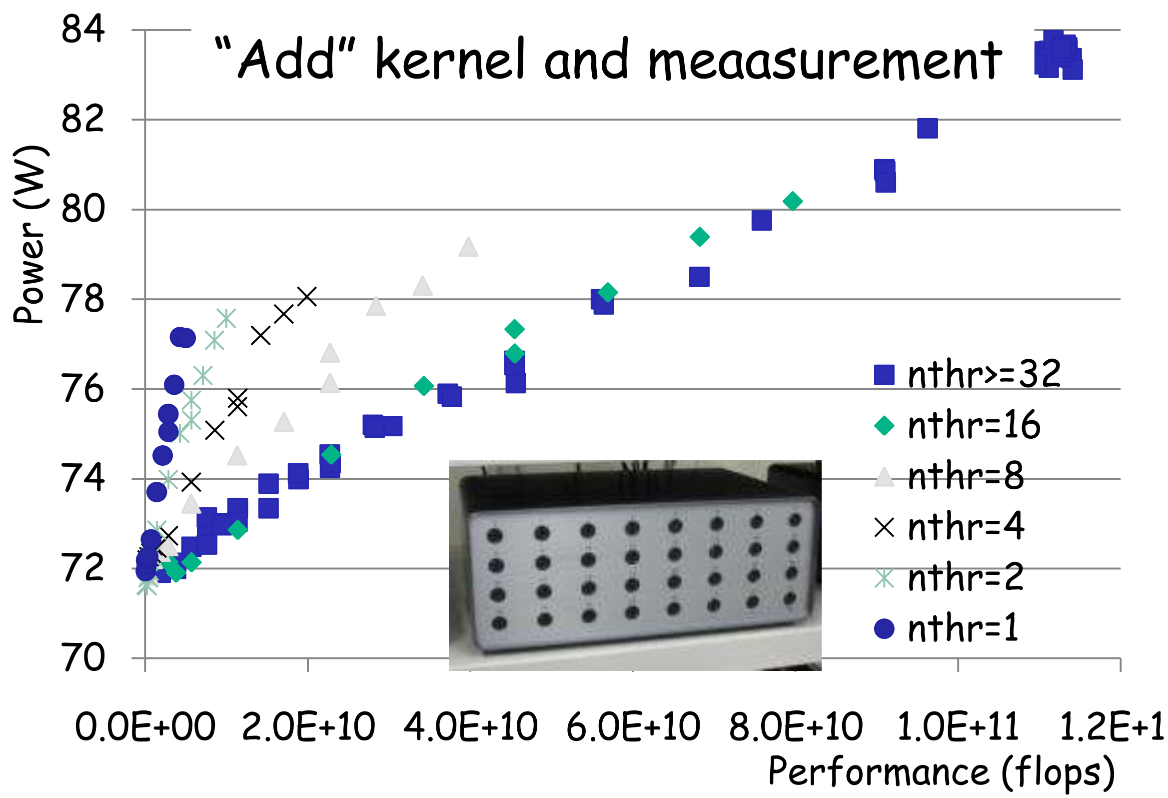
High Precision GPU Power Measurement (Reiji Suda, U-Tokyo, ULPHPC)

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

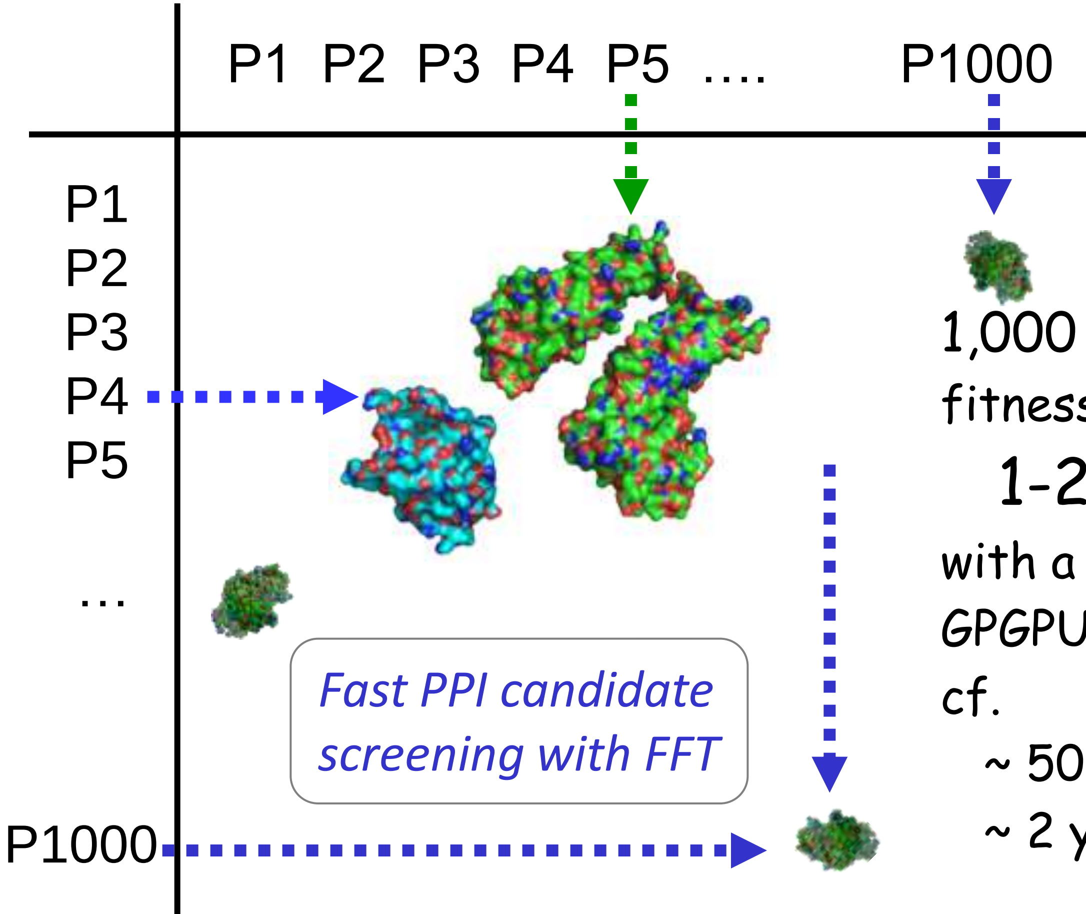
マーカーの自動検出例







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).

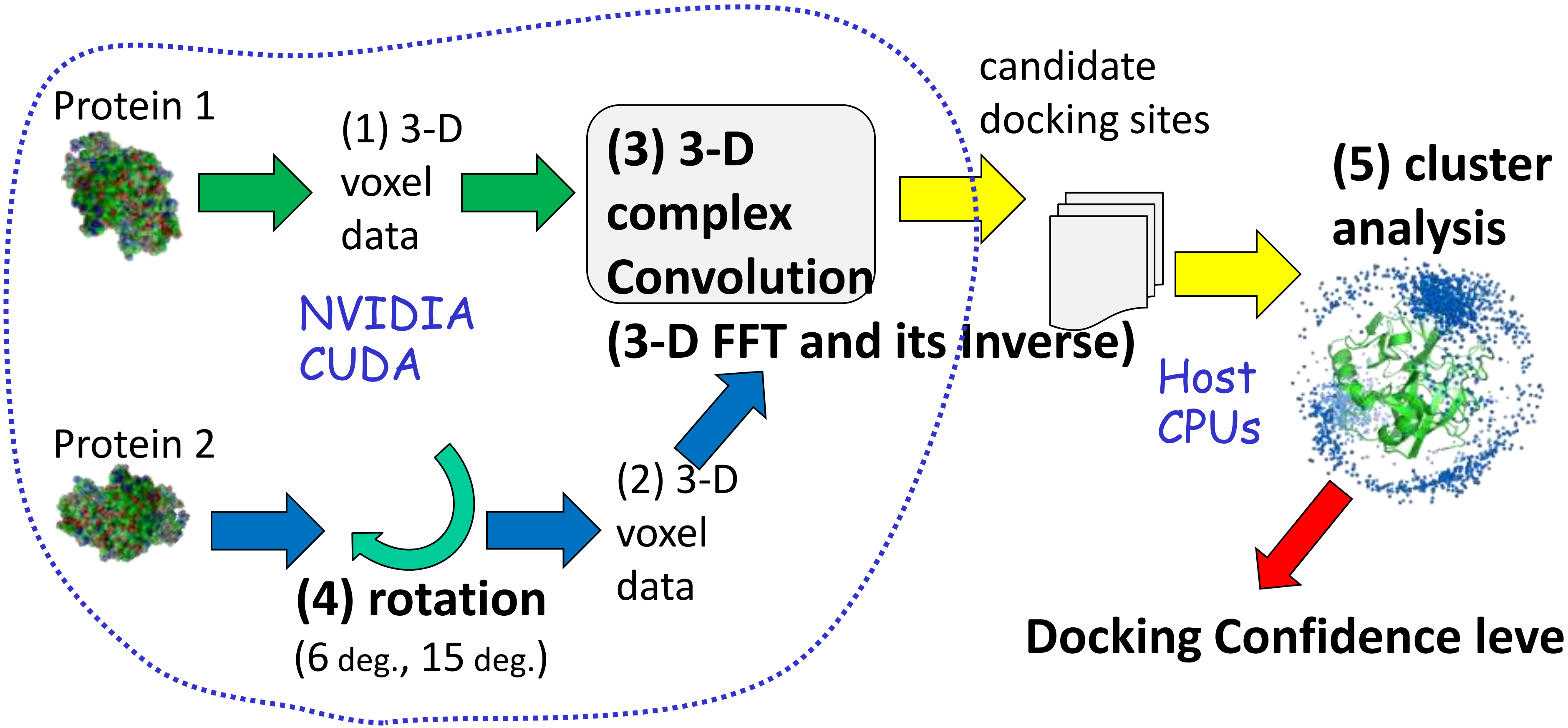
cf.

~ 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

Multi-GPU in CFD: Riken Himeno Benchmark

(Joint Work w/NEC)

RIKEN Himeno CFD Benchmark

Poisson Equation:
(Generalized coordinate)

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

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

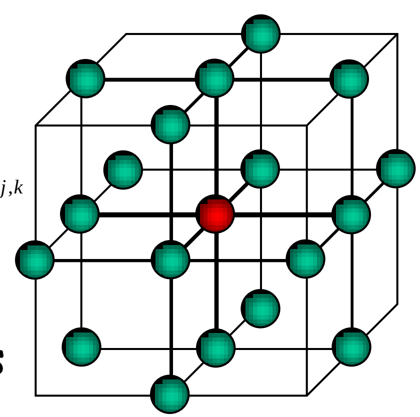
Discretized Form:

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

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

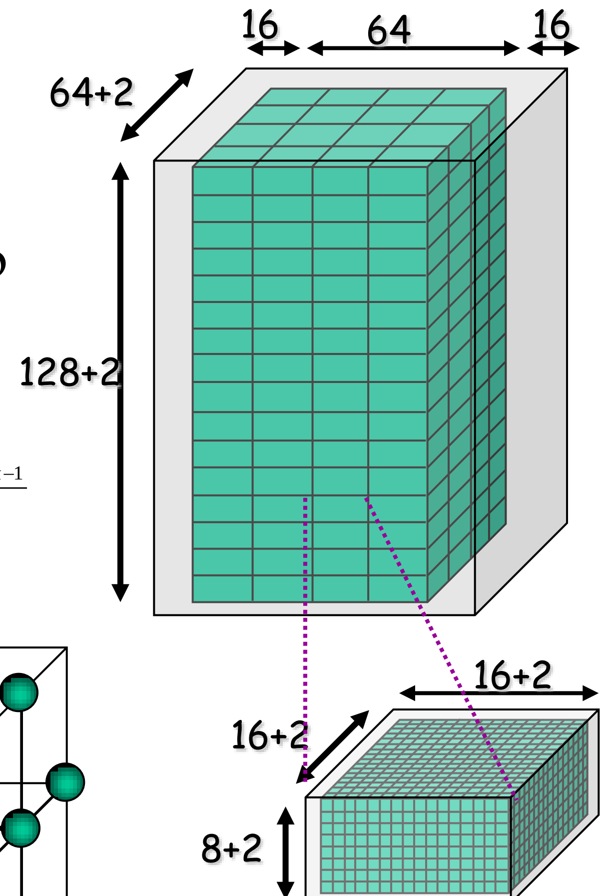
$$+ \beta \frac{p_{i+1,j,k+1} - p_{i-1,j,k+1} - p_{i+1,j-1,k} + p_{i-1,j,k-1}}{4\Delta x \Delta z}$$

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



18 neighbor point access

Himeno for CUDA



1 block = 16x16x8 compute region

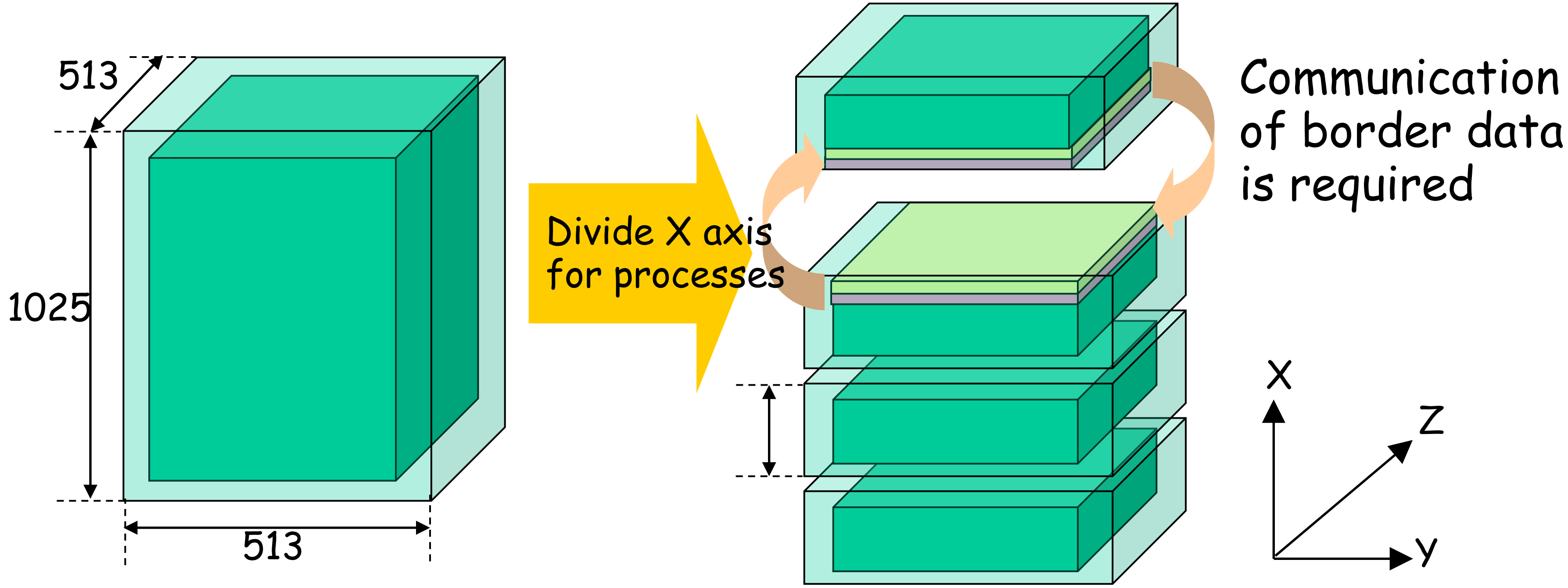
Block has 256 thread

Total 256 blocks = 65536 threads

Block shared mem =16kB

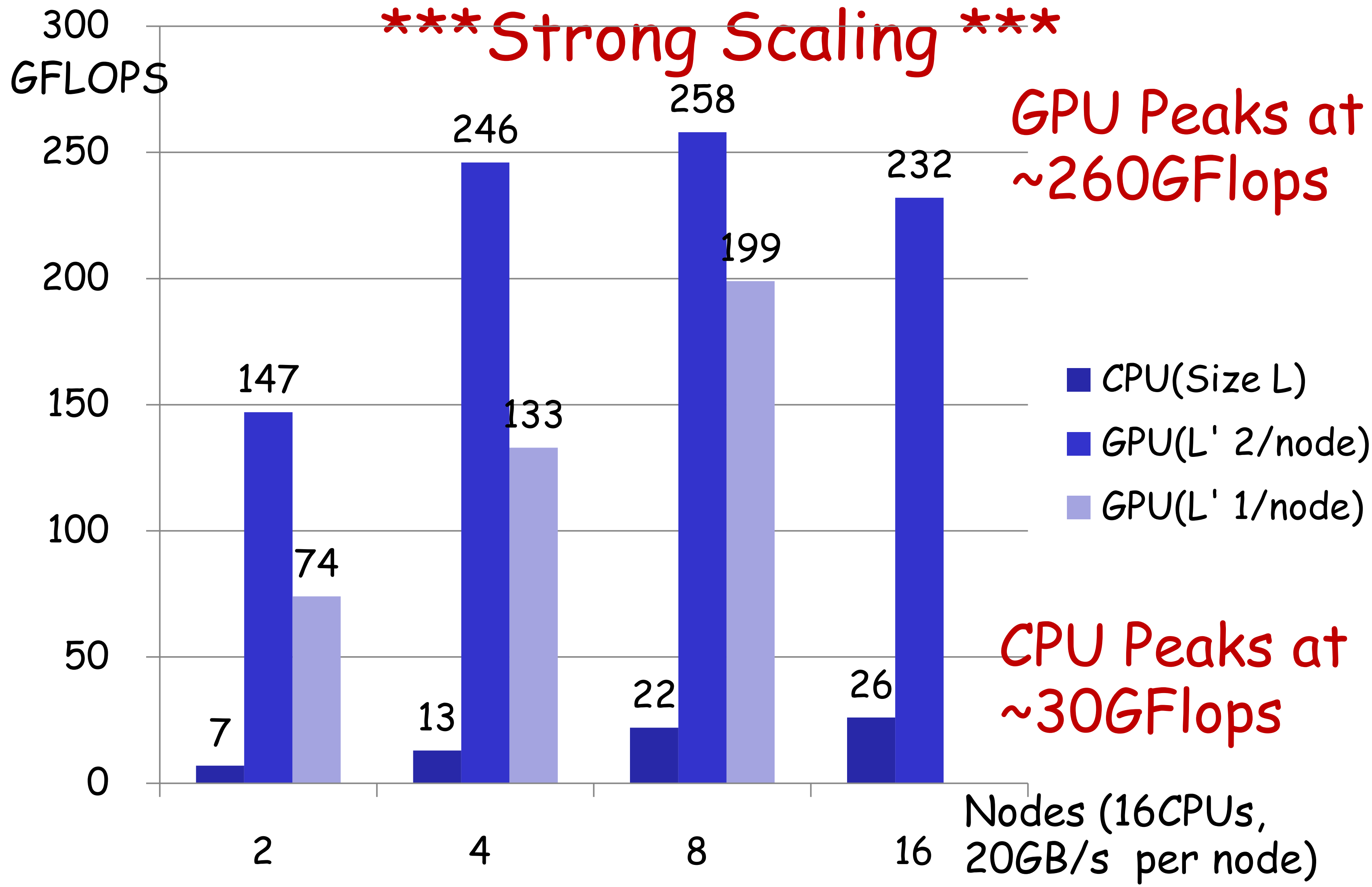
Boundary region used for transfer

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



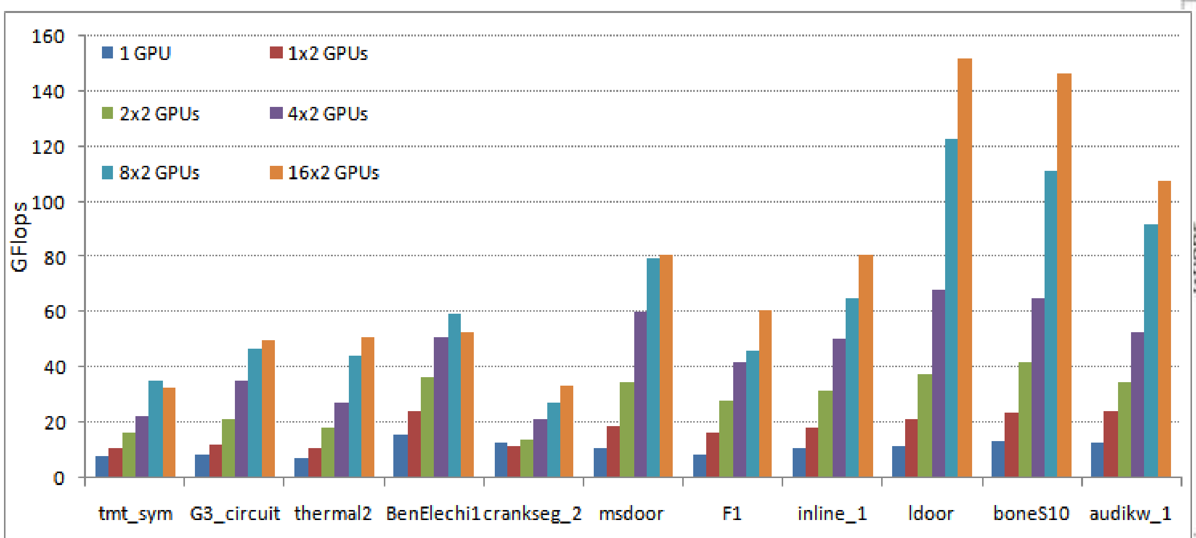
Conjugate Gradient Solver on a Multi-GPU Cluster

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

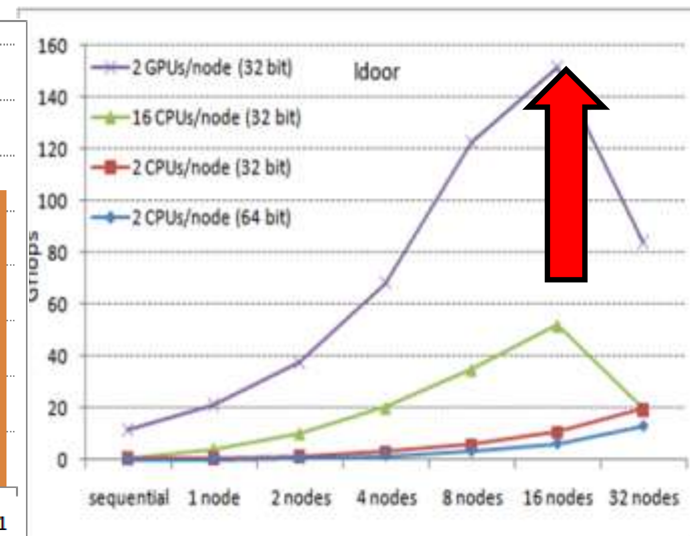
- Hypergraph partitioning for reduction of communication
  - GPU computing is fast, communication bottleneck is more severe
- All matrix and vector operations are implemented on GPUs
- Auto-selection of MxV kernel. GPUs may run different kernel

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

Performance comparisons over well-known matrices



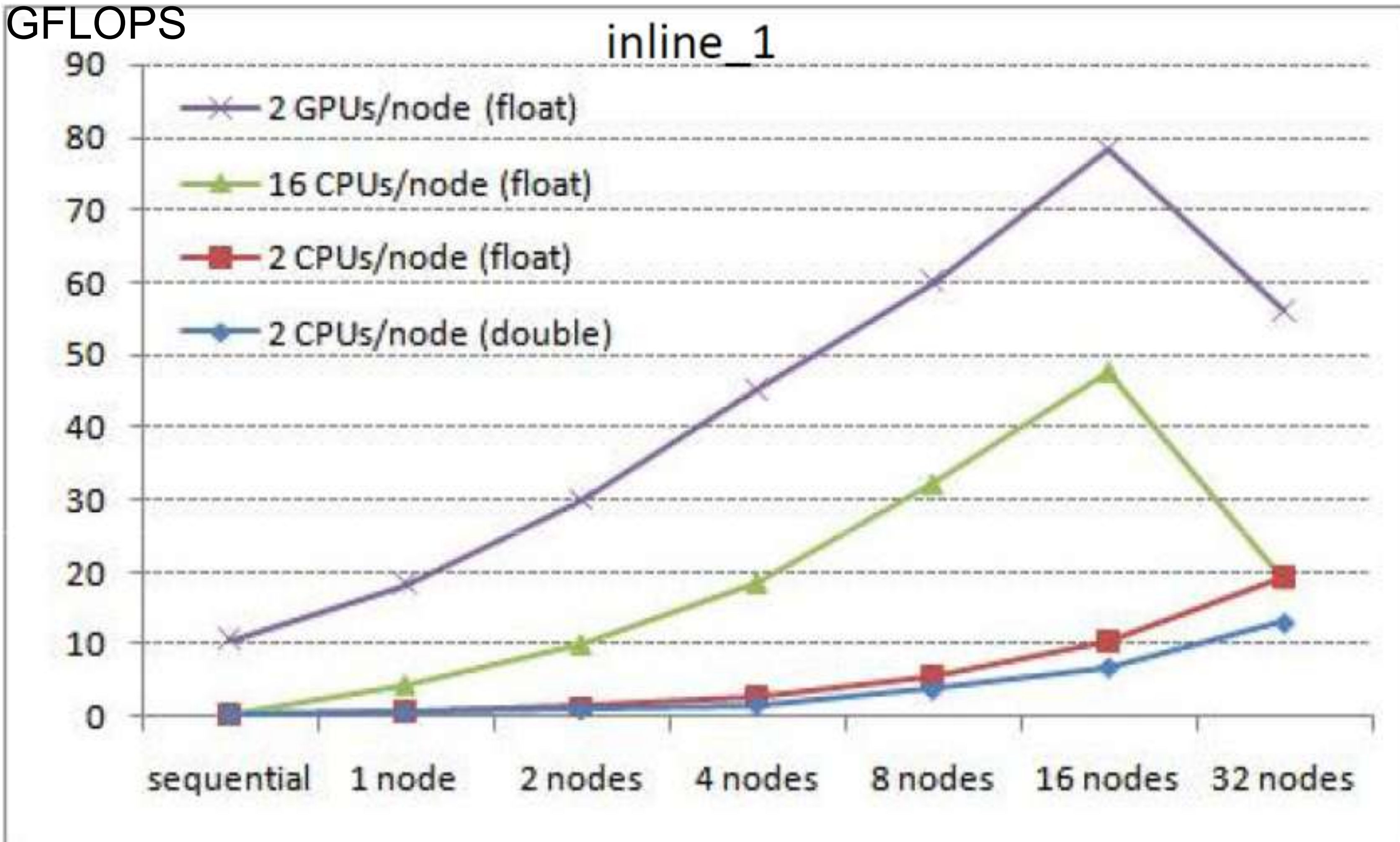
GPUs vs CPUs on TSUBAME
(Strong scaling)



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

Fast CG Result (2)

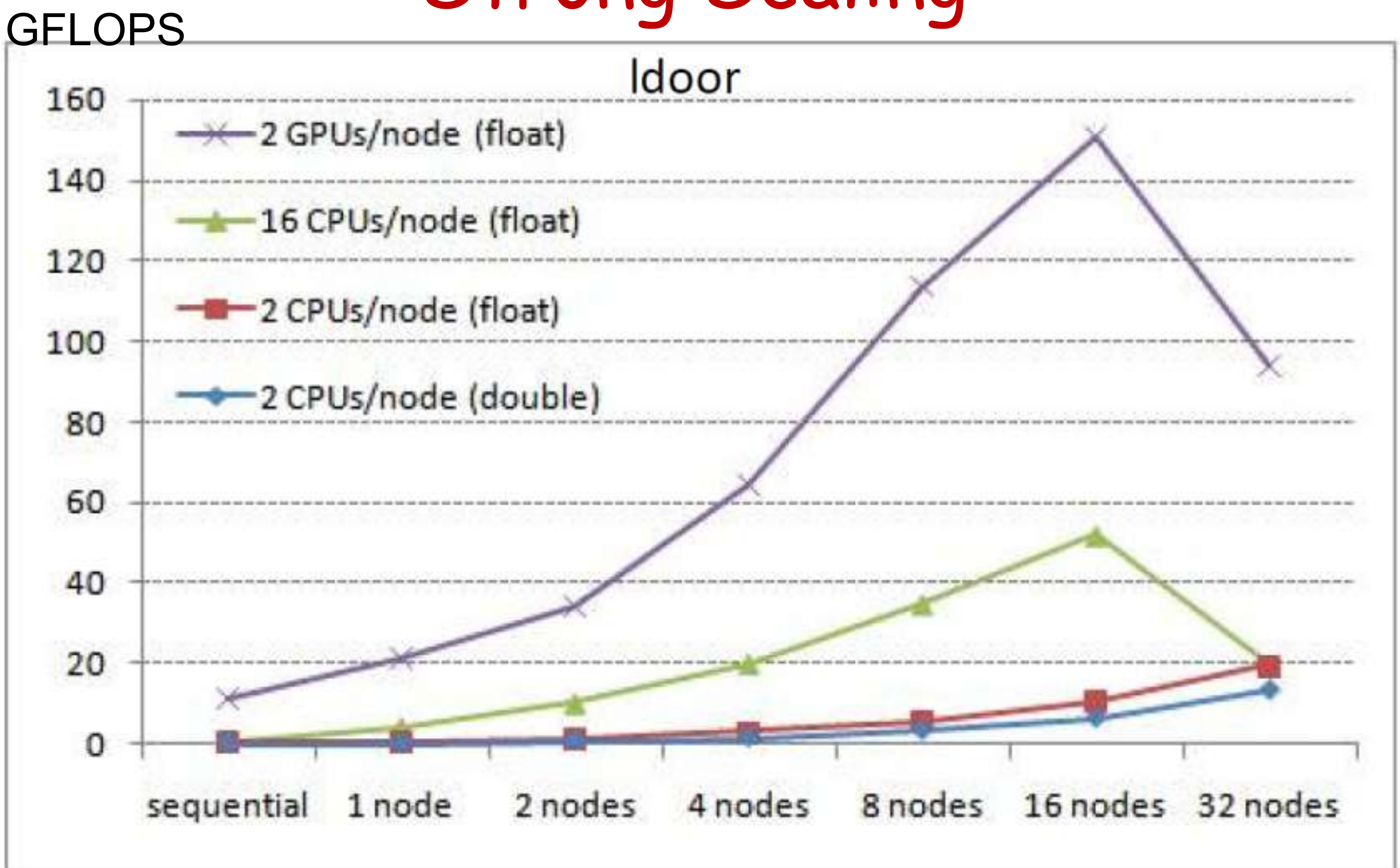
***Strong Scaling ***



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

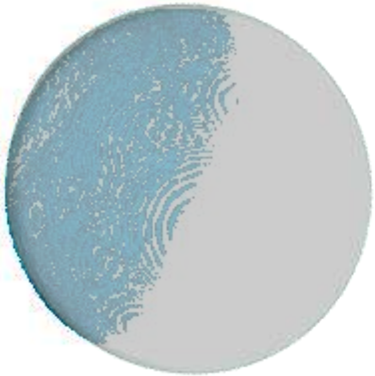
Fast CG Result (3)

***Strong Scaling ***

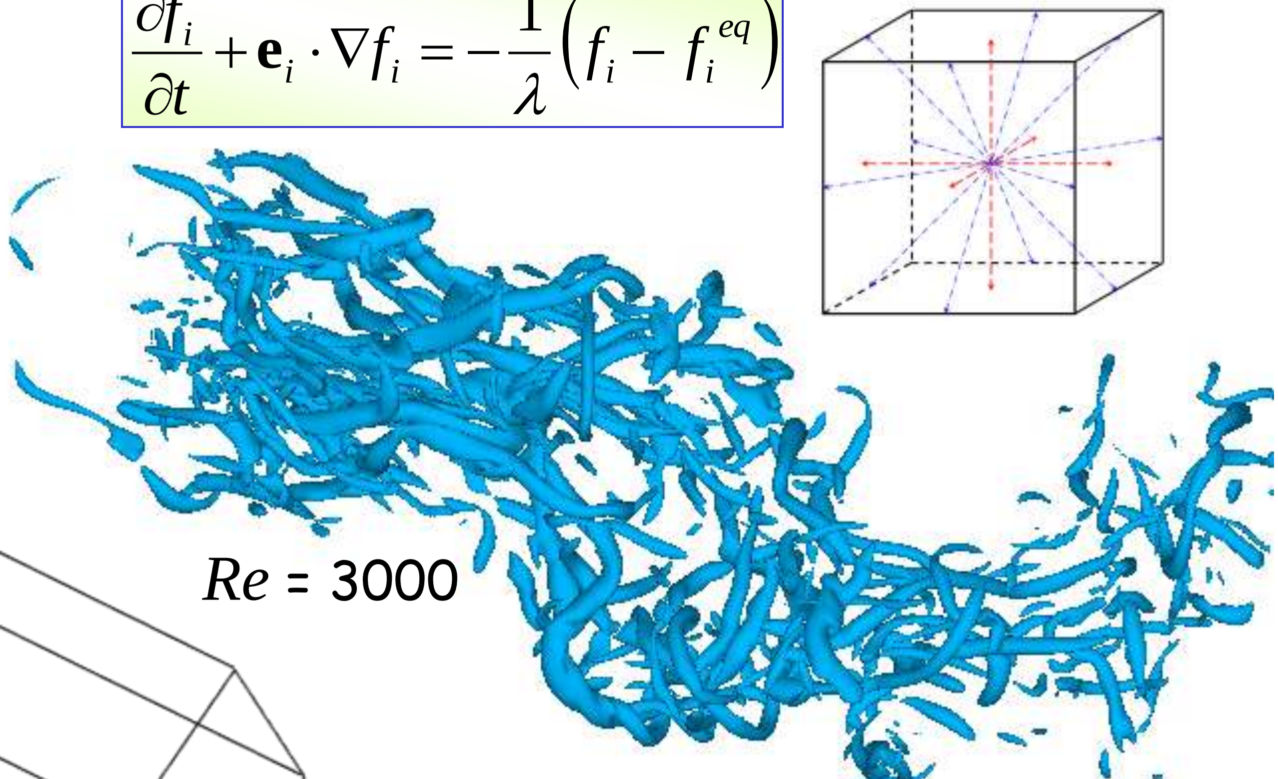


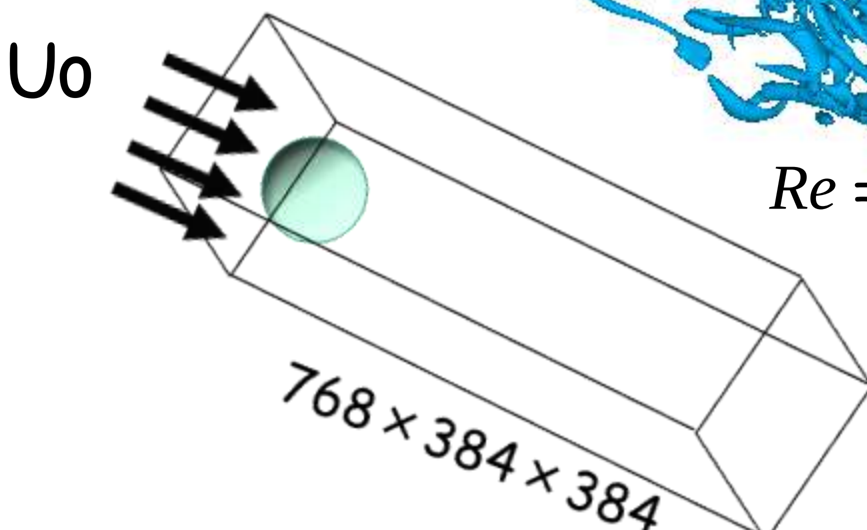
Nonzeros: 46,522,475 n: 952,253

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



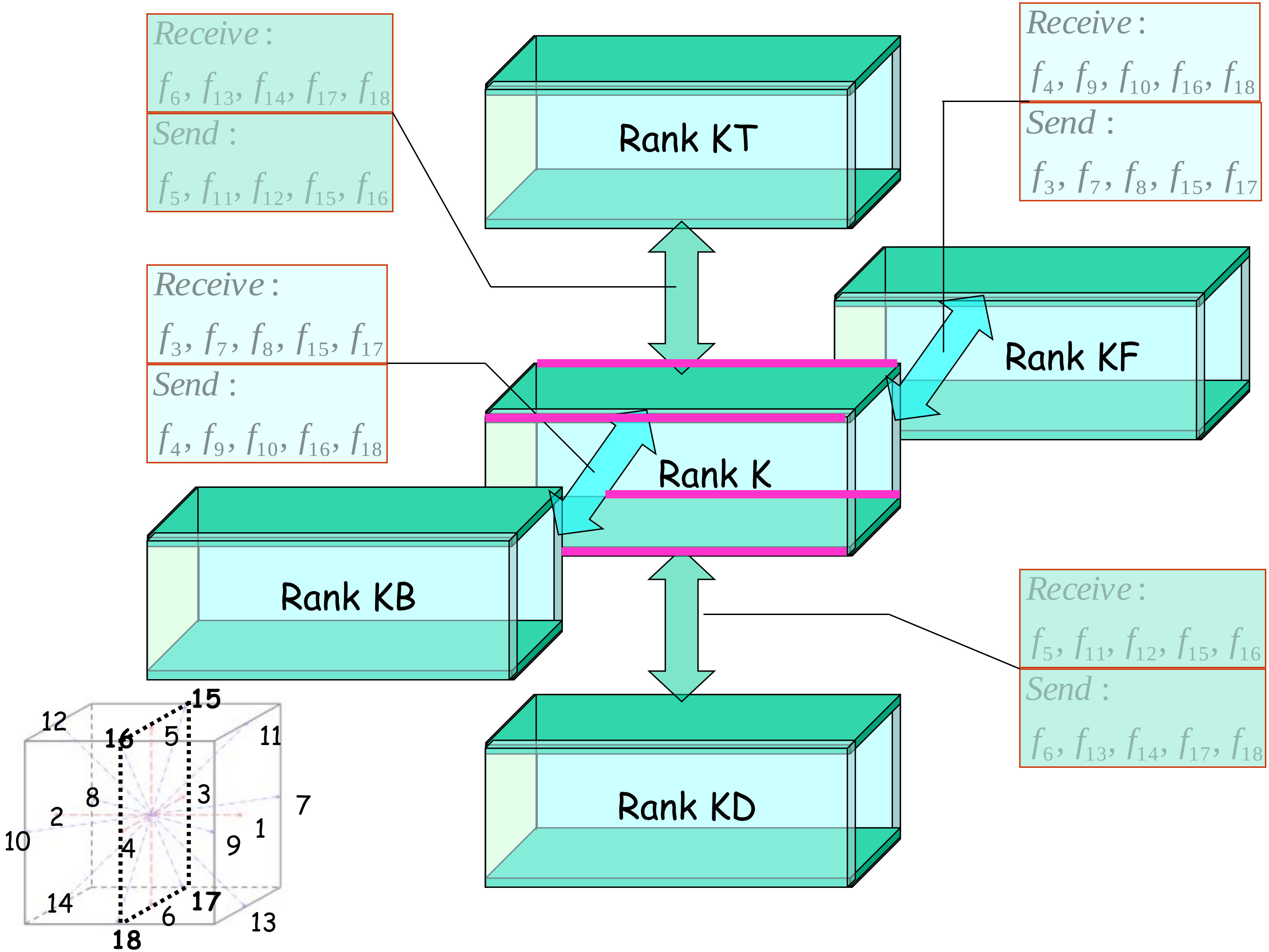
$$\frac{\partial f_i}{\partial t} + \mathbf{e}_i \cdot \nabla f_i = -\frac{1}{\lambda}\left(f_i - f_i^{eq}\right)$$





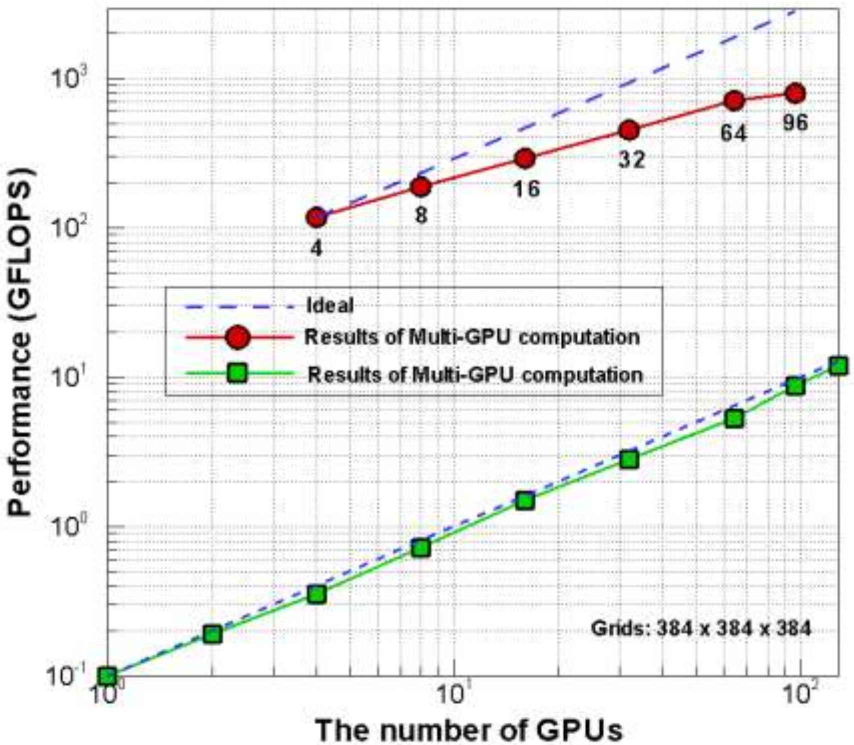
Re = 3000

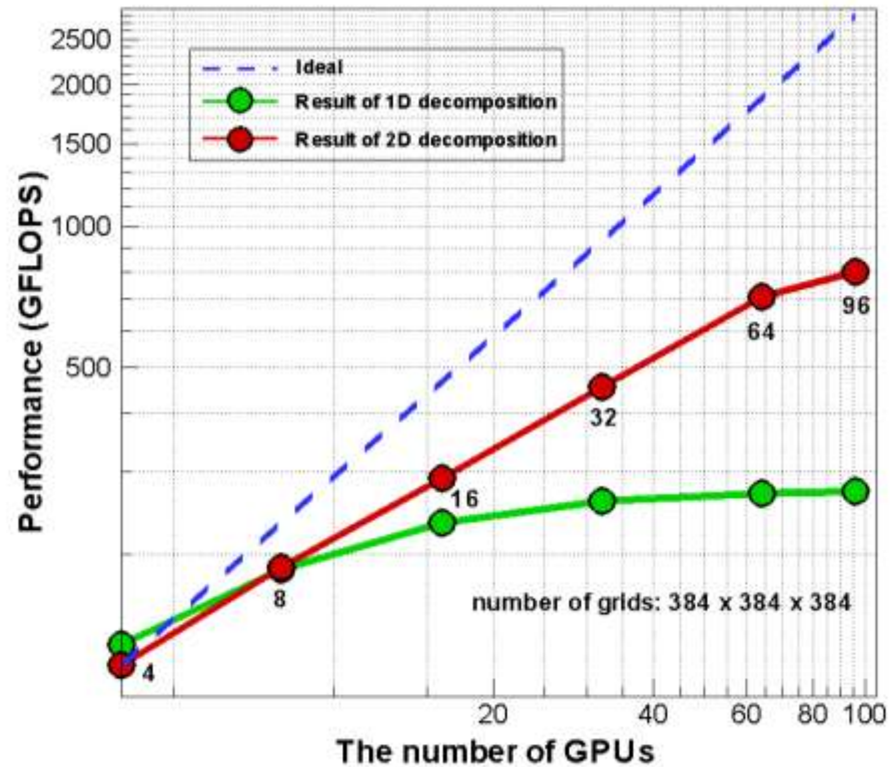
Tesla S1070 **64 GPU**



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 : $f_3, f_7, f_8, f_{15}, f_{17}$
Receive : $f_3, f_7, f_8, f_{15}, f_{17}$
Send : $f_4, f_9, f_{10}, f_{16}, f_{18}$
Rank KF
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
12 15 16 5 11 8 3 7 2 1 10 9 4 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(!)