# HPC-GPU: Large-Scale GPU Accelerated Windows HPC Clusters and its Application to Advanced Bioinformatics and Structural Proteomics (and Climate/Environment)

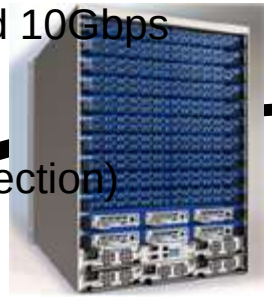
Satoshi Matsuoka, Professor/Dr.Sci.
&
Yutaka Akiyama, Professor
with
Toshio Endo, Fumikazu Konishi, Akira Nukada, Naoya Maruyama…

Tokyo Inst. Technology

# The TSUBAME 1.0 "Supercomputing Grid Cluster" April 2006 at Tokyo Tech 80 TFlops, 1400 Users, 200 Industry Users

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

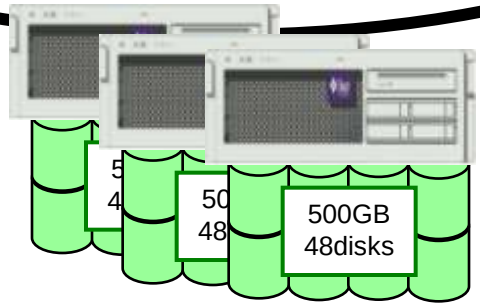
10Gbps+External Network



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

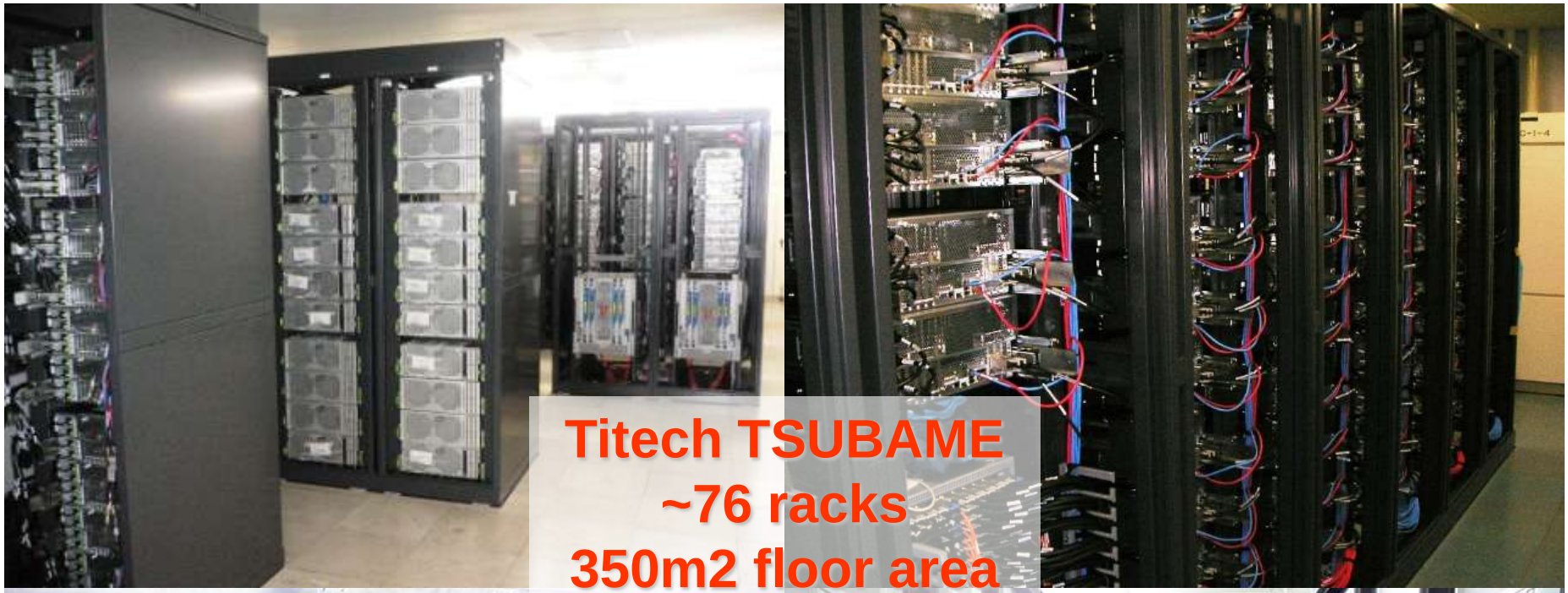
Sun Galaxy 4 (Opteron Dual core 8-socket)
10480core/655Nodes
21.4Terabytes
50.4TeraFlops
OS Linux (SuSE 9, 10)
NAREGI Grid MW


Unified IB network



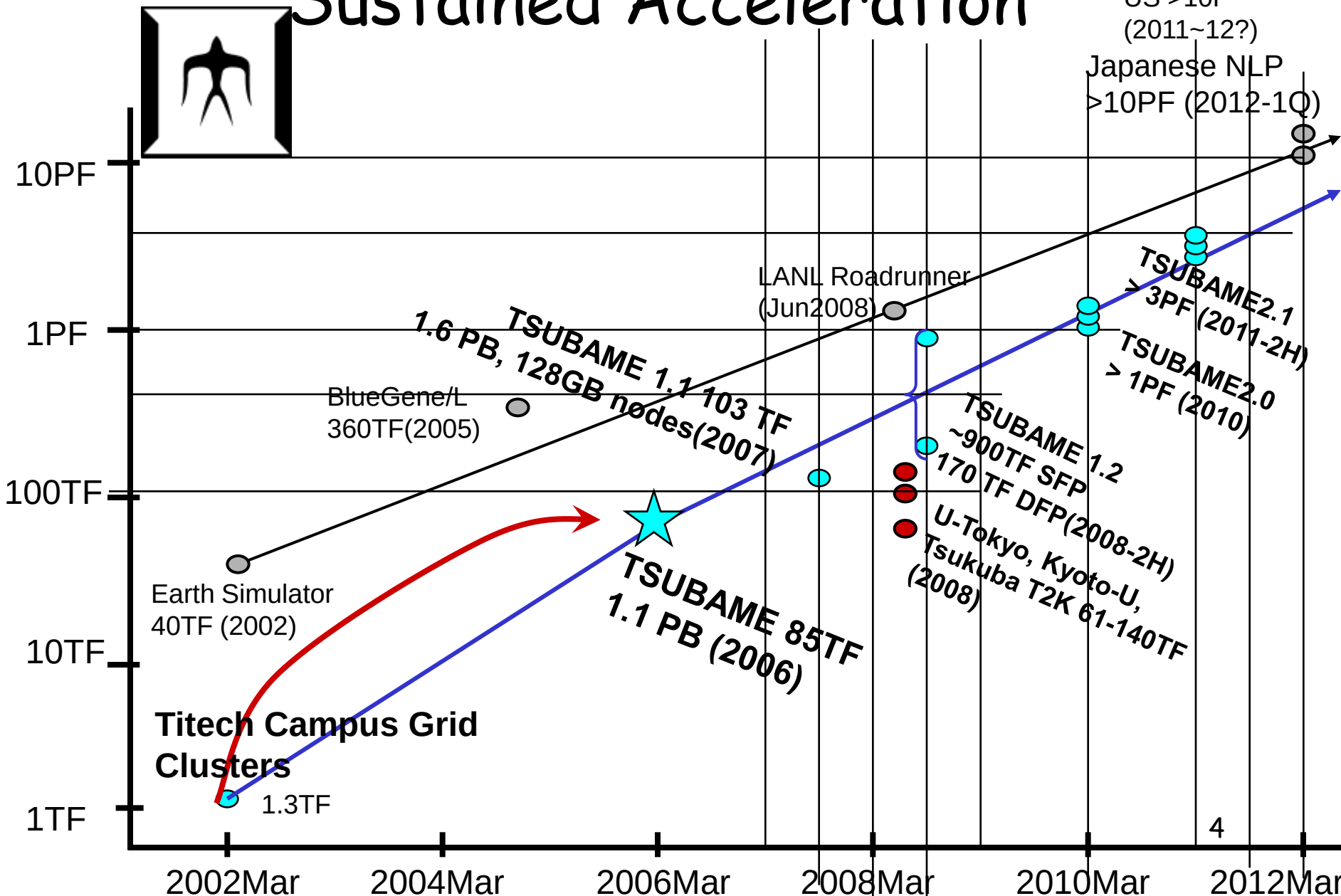

Storage
1.0 Petabyte (Sun "Thumper")
0.1Petabyte (NEC iStore)
**Lustre FS, NFS, WebDAV (over IP)**
50GB/s aggregate I/O BW

ClearSpeed CSX600
SIMD accelerator
360 boards,
35TeraFlops(Current))



**Titech TSUBAME**
**~76 racks**
**350m2 floor area**
**1.2 MW (peak)**

# TSUBAME Upgrades Towards Petaflops Sustained Acceleration

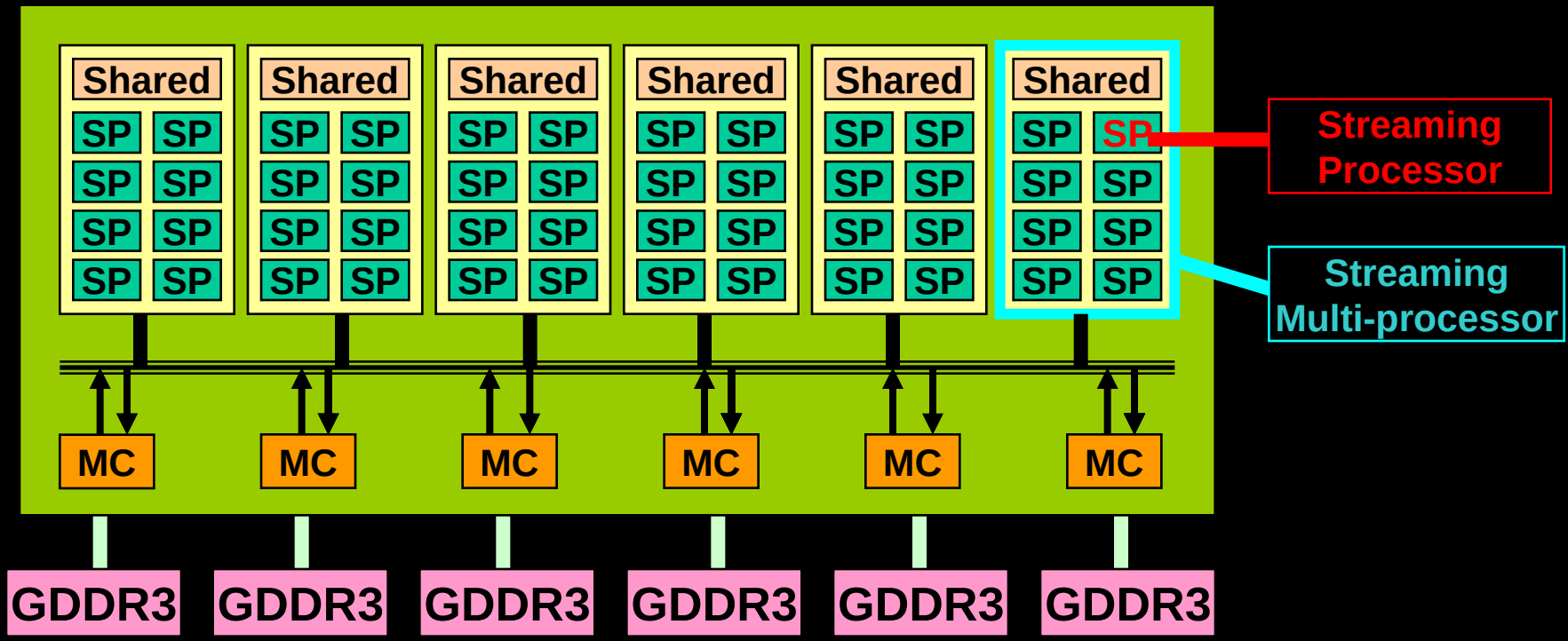


# Biggest Problem is Power…

| 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 |

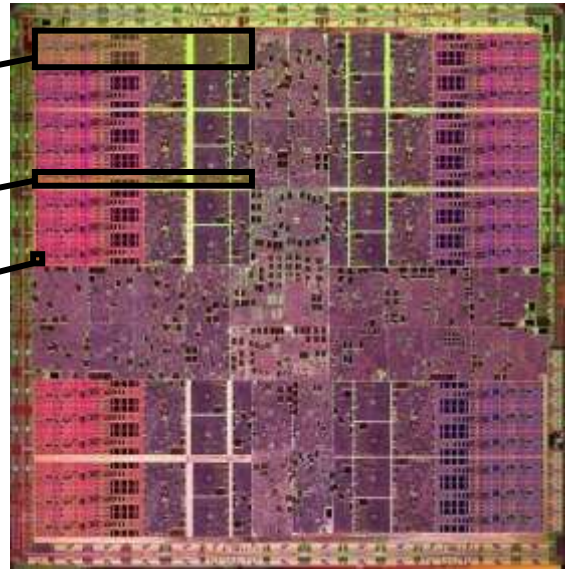
TSUBAME 2.0 x24 improvement in 4.5 years…? ➔ ~ x1000 over 10 years

# NVIDIA CUDA Architecture




240 SPs / Chip (Tesla G200 GPU)
Great Relative Power/Performance
Possibilities for Scientific Codes

# GPUs (Tesla, FireStream, Larrabee, ClearSpeed) as Extreme Many-Core and Solution to This Problem

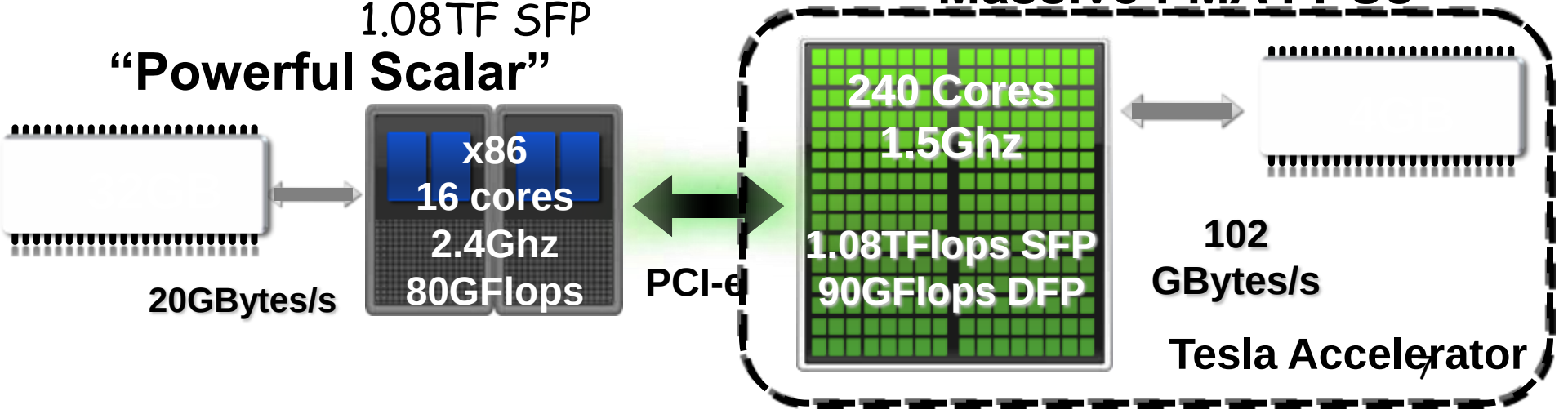



**65~55nm(2008)**
**=> 15 nm (2016)**
**x20 transitors (30 bil)**
**5000 Cores**
**20TF FMA SFP**
**10TF FMA DFP**

nVidia Tesla T10: 65nm, 600m2, 1.4 bil Tr
1.08TF SFP

**"Powerful Scalar"**

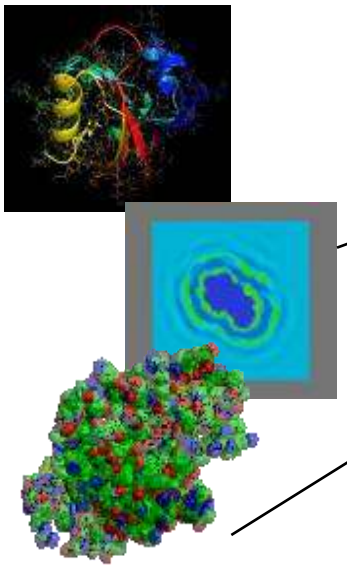
**"Massive FMA FPUs"**



# Overview of HPC-GPGPU Project

## (sponsored by Microsoft Research)

***Advanced Bioinformatics/ Proteomics***

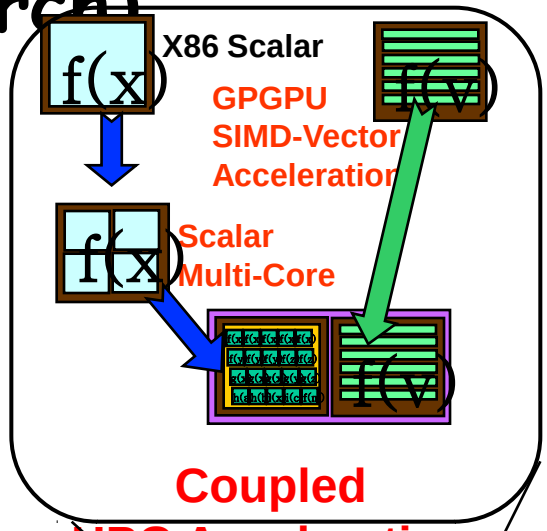


**Need x1000 acceleration over standard PCs**

**Bioinformatics Acceleration e.g., 3-D All-to-All Protein Docking**

**GPGPU-CPU Hybrid Massively Parallel "Adaptive" Solvers + GPGPU FFT and other Acceleration Kernels**

**- Improving GPGPU Programmability w/ Library/Languages e.g. MS Accelerator**
**- High Dependability w/ large-scale GPGPU Cluster**
**- CPU-GPU Heterogeneity**
**- Low Energy Computing w/GPUs**

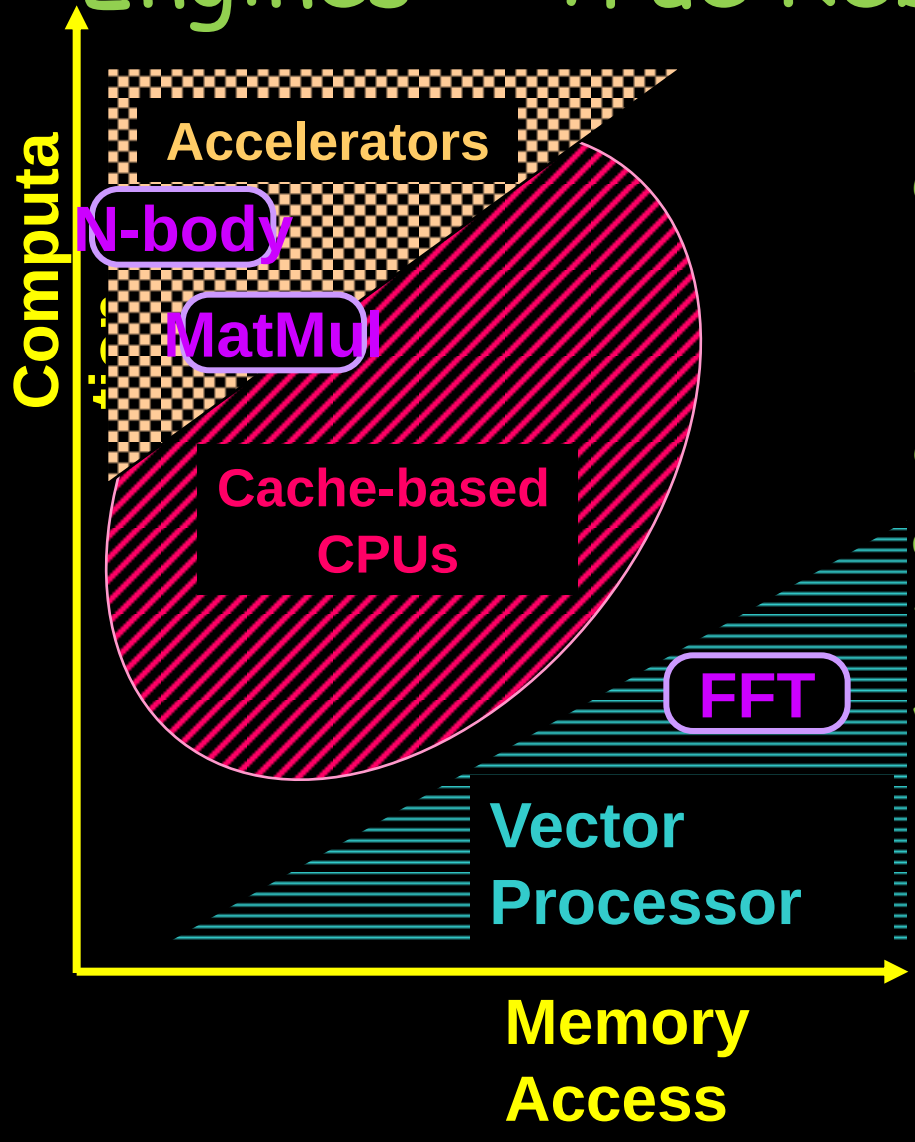



**Coupled HPC Acceleration**

**Prototype Cluster 32-64 nodes, 50-100TFlops**

**Towards Next Gen Petascale Personal Clusters and Desksides**

# GPUs as Commodity Vector Engines---True Rebirth of Vectors




Unlike the conventional accelerators, GPUs have high memory bandwidth.

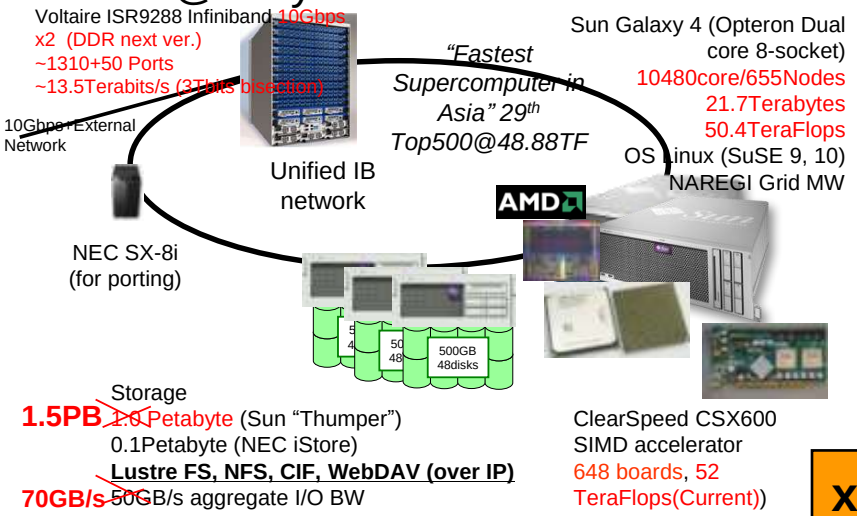
Since latest high-end GPUs support double precision, GPUs also work as commodity vector processors. The target application area for GPUs is very wide.

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

➔ How do we utilize them easily?

# Prototype HPC-GPGPU Cluster

*Current*: TSUBAME Production "Supercomputing Grid Cluster" @Tokyo Tech 2006-2010




**~100 TFlops**
**80+ racks**
**350m2 floor**
**1.2 MW (peak)**
**~50 tons**
**$70~80 mil MSRP**
**Fastest SC in Japan**

**x500 C/P improve-ment**

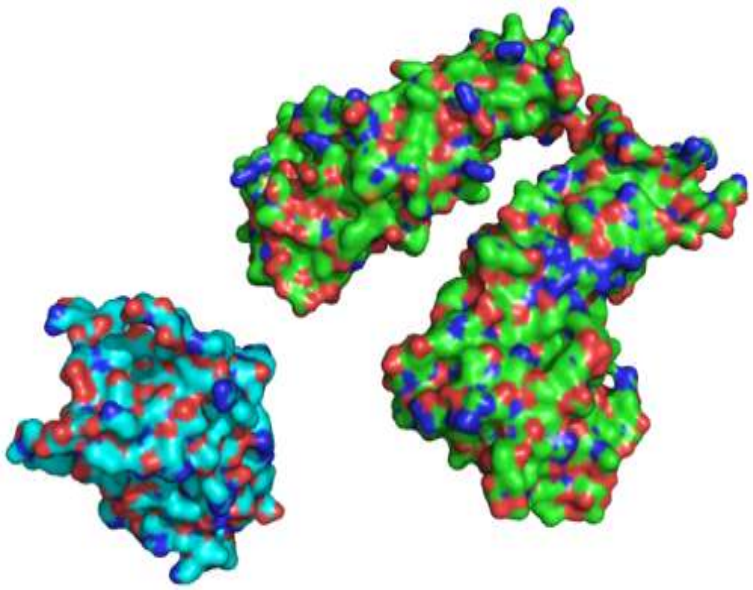
**Windows HPC-GPU Cluster**

**50~100 TFlops Peak, fast FFT**

Node Architecture and Specs
- Single Socket, Quad Core x64
- Latest GPGPUs from NVidia/AMD
- Small Memory (4GB DDR2)
- 600W Power
- 1~2 TFlops, $2500/node

- Whole Cluster Specs (~50 nodes)
  - **50-100 Teraflops**
  - 20K-30KW Power
  - Massive FFT Engine
  - $100,000-$200,000 MSRP

# Protein-Protein Interaction



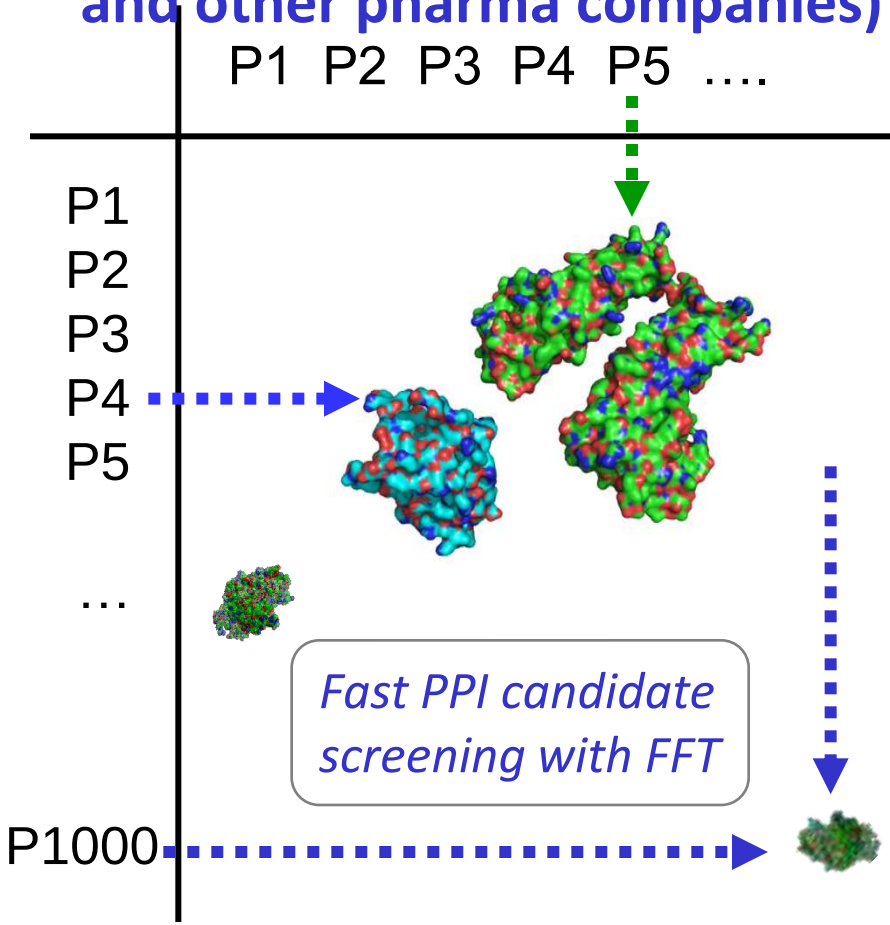
Protein docking is a central issue both in computational biology and current drug design technique in pharmaceutical industry.

It is becoming a new trend to design an "inhibitor" drug compound which controls specified protein-protein interaction as a target.

**Protein Docking is important for:**

1) **Enzymatic reaction**
2) **Signal Transduction**
3) **Formulating protein scaffold, etc.**

# All-to-all 3-D Protein Docking Challenge (by Y. Akiyama, in collaboration w/AIST CBRC and other pharma companies)




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).

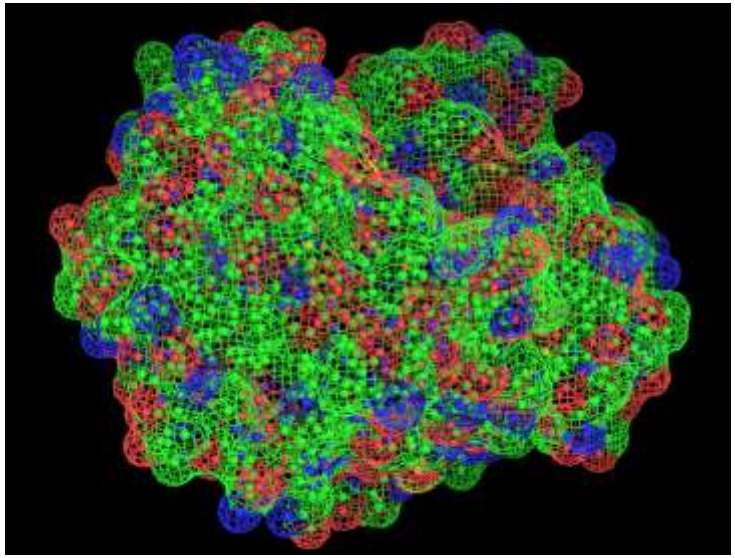
cf.

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

> 2 years with 1-rack BlueGene/L

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

# Rigid Docking vs. Flexible Docking



Porcine Pancreatic Trypsin (PDB:1AVX).

Rigid Docking

- Protein is regarded as a rigid body
- Shape complementarity and only simple physicochemical potentials.

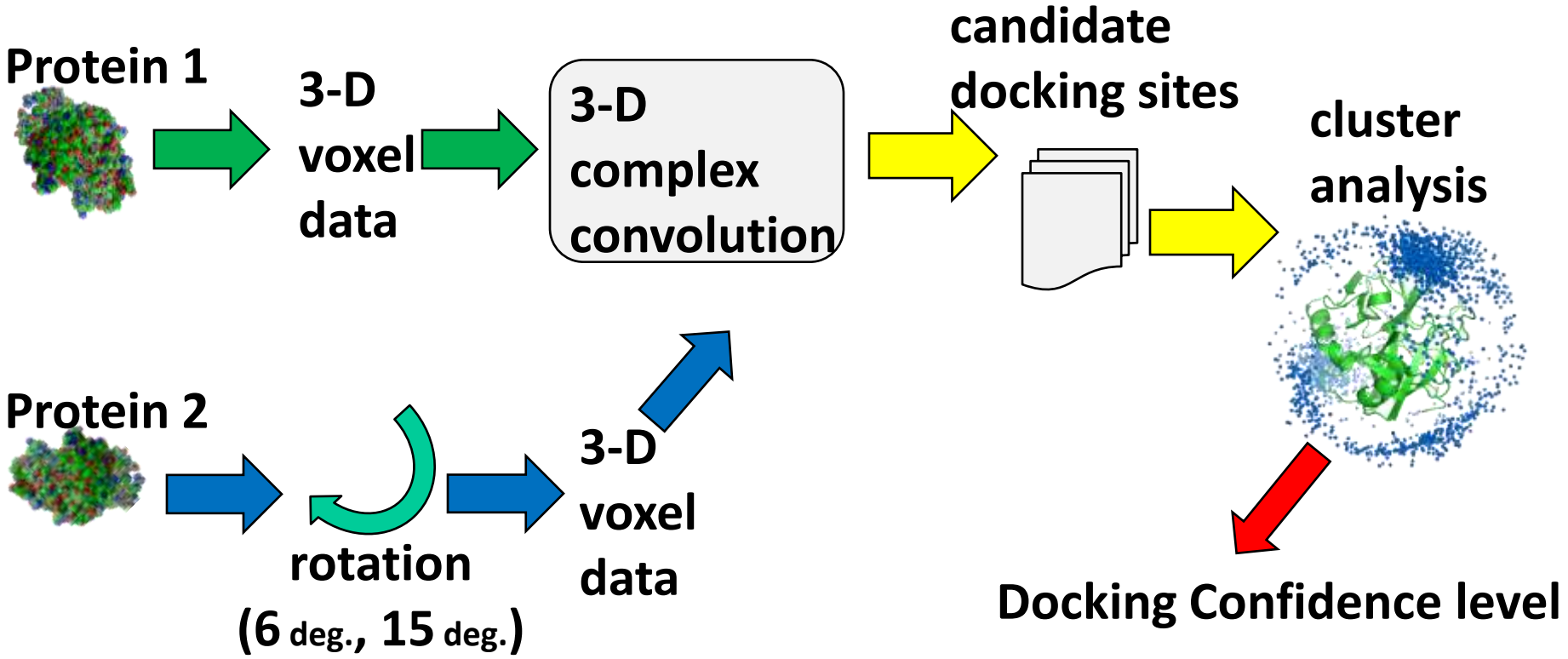
*No biologist* think proteins are rigid.

However,

- Flexible docking is prohibitively expensive due to energy local minima.
- Flexible docking usually needs good initial docking structure.
- Some proteins are almost rigid.

**Single Rigid docking has only limited validity.**
**But all-to-all Rigid Docking screening can be a good basis for quick survey of potential pairs, and for flexible docking study.**

# Calculation flow and Complexity




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

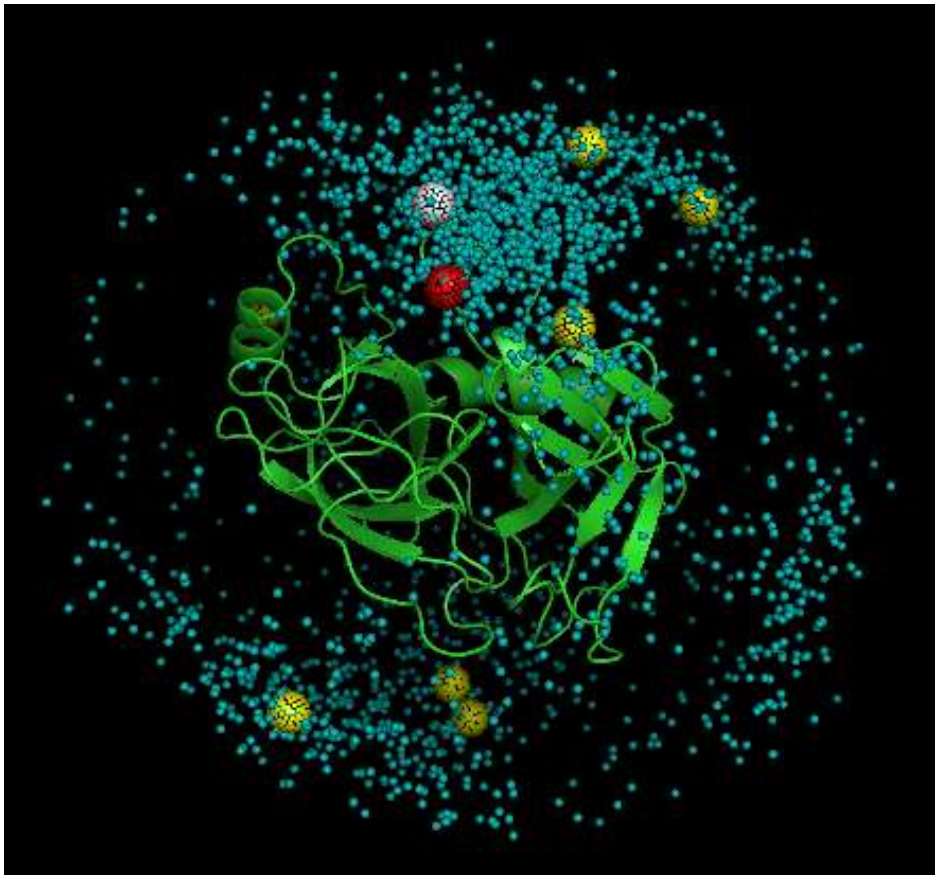
3-D complex convolution $O(N^3 \log N)$, typically $N = 256$

x

Possible rotations $R = 54{,}000$ (6 deg. pitch) ~= 200 Exa ($10^{20}$) ops

# Rigid Docking Example

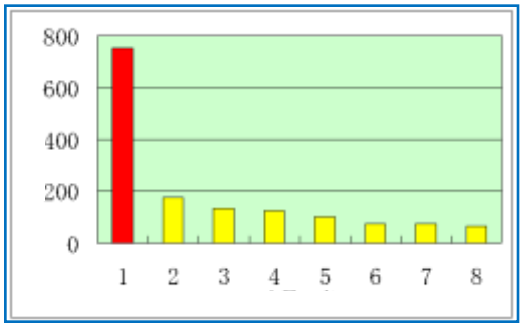
- Implemented 6 clustering methods for Post-Docking analysis
- Developed a confidence level evaluation procedure



Trypsin and inhibitor (PDB:1AVX)

prediction other clusters real

Receptor protein (Trypsin) is shown in center as a ribbon diagram. Top 2,000 candidate ligand docking sites are shown by small dots. Final prediction site is shown by a red sphere.




clusters vs. significance

If significant candidate is found, the system outputs predicted docking structure with a confidence level, otherwise reports as "No docking".

# Bandwidth Intensive 3-D FFT on NVIDIA CUDA GPUs [SC08]

By Akira Nukada, Tokyo Tech.

Our 3-D FFT algorithm consists of the following two algorithms

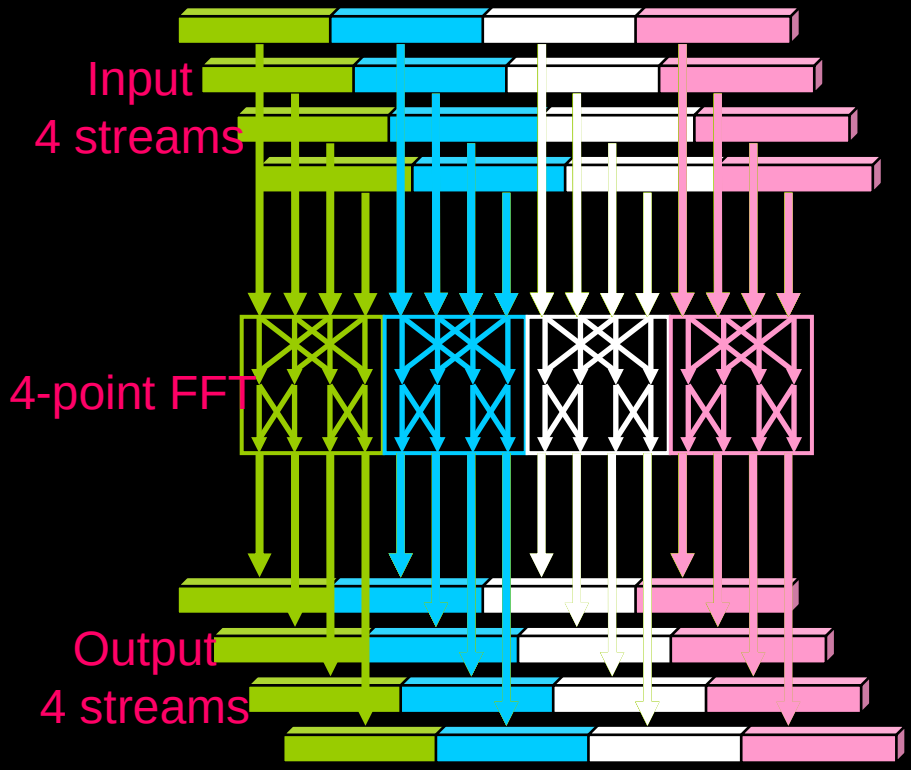
to maximize the memory bandwidth.

(1) optimized 1-D FFTs for dimension X,

(2) *multi-row FFT* for dimension Y & Z.

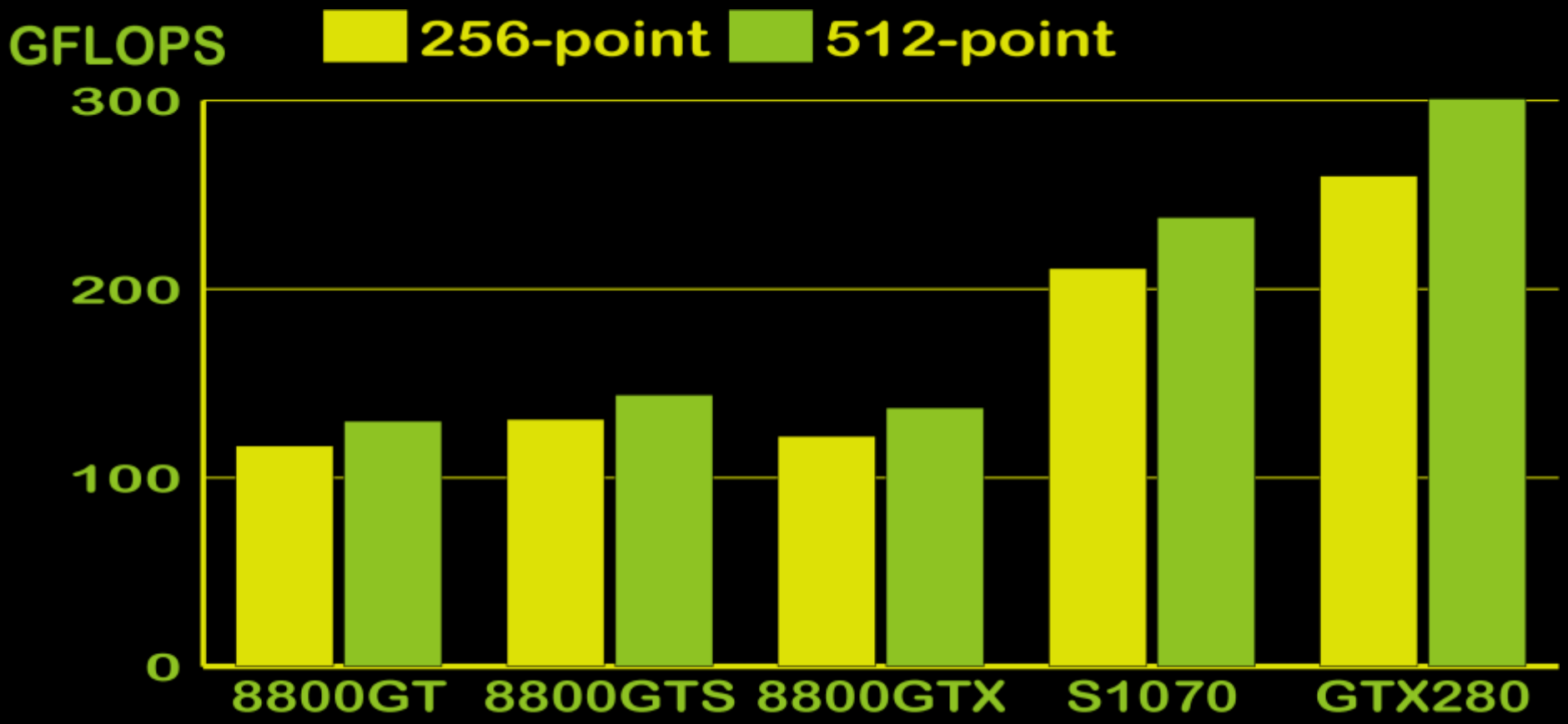
The multi-row FFT computes multiple 1-D FFTs simultaneously.

Adapted from vector algorithms, assuming high memory bandwidth.



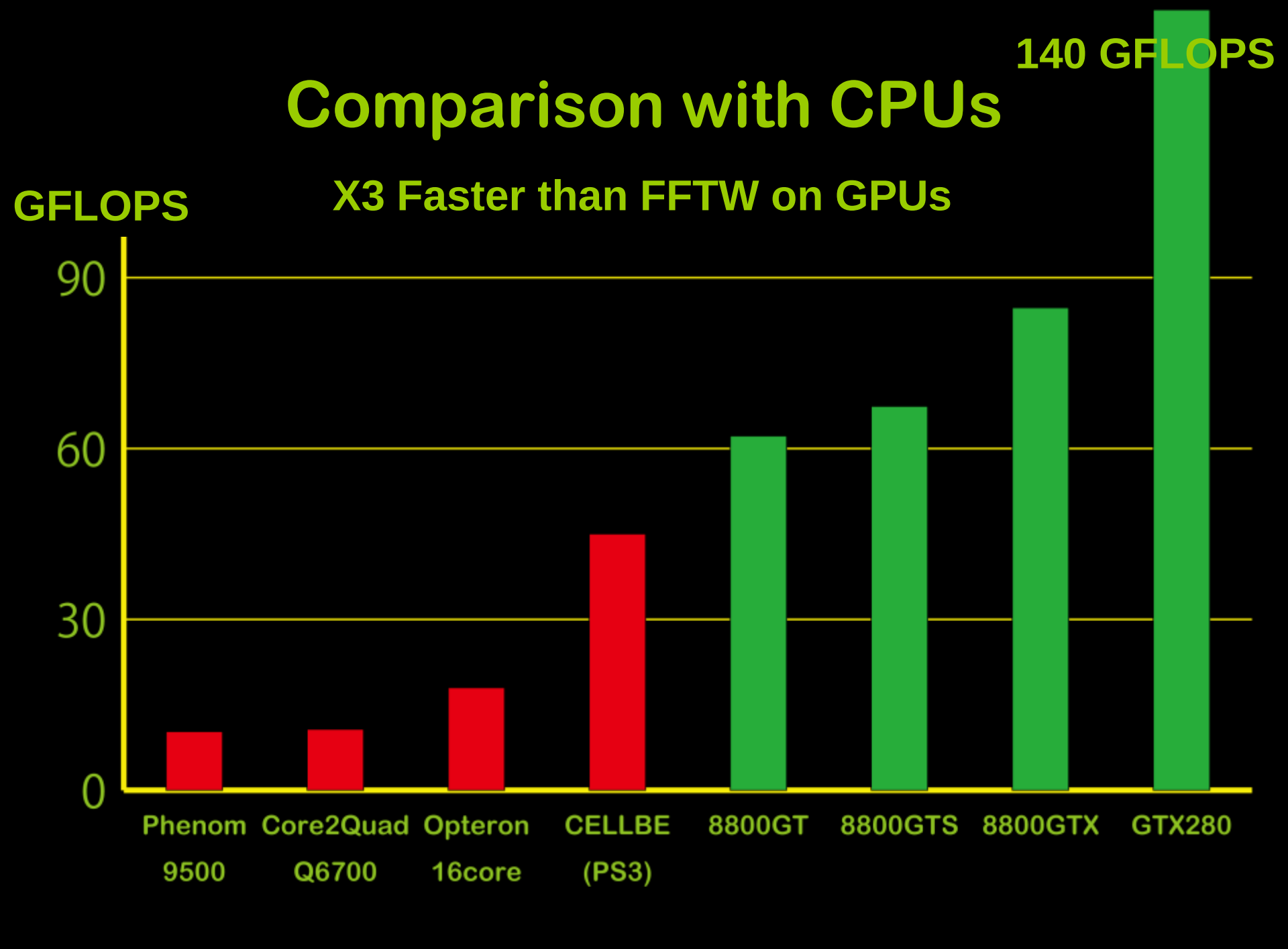

This algorithm accesses multiple streams, but each of them is successive.
Since each thread compute independent set of small FFT, thousands of registers are required
Solution: for 256-point FFT, use two- pass 16-point FFT kernels.

# Performance of 1-D FFT

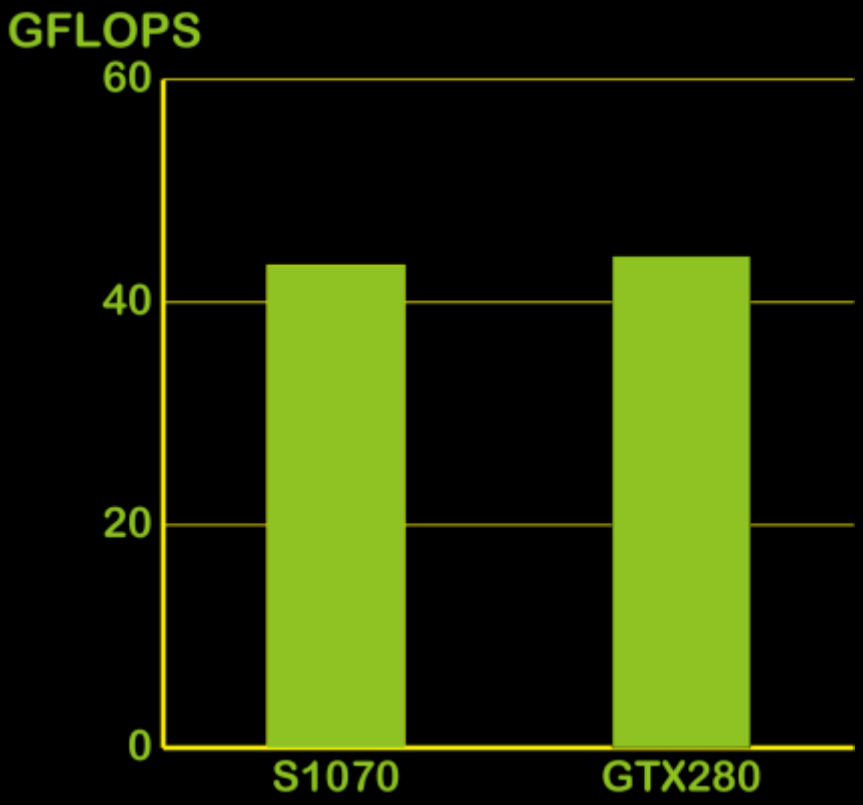



Note: An earlier sample with 1.3GHz is used for Tesla S1070.

# Comparison with CPUs



# Performance in Double Precision



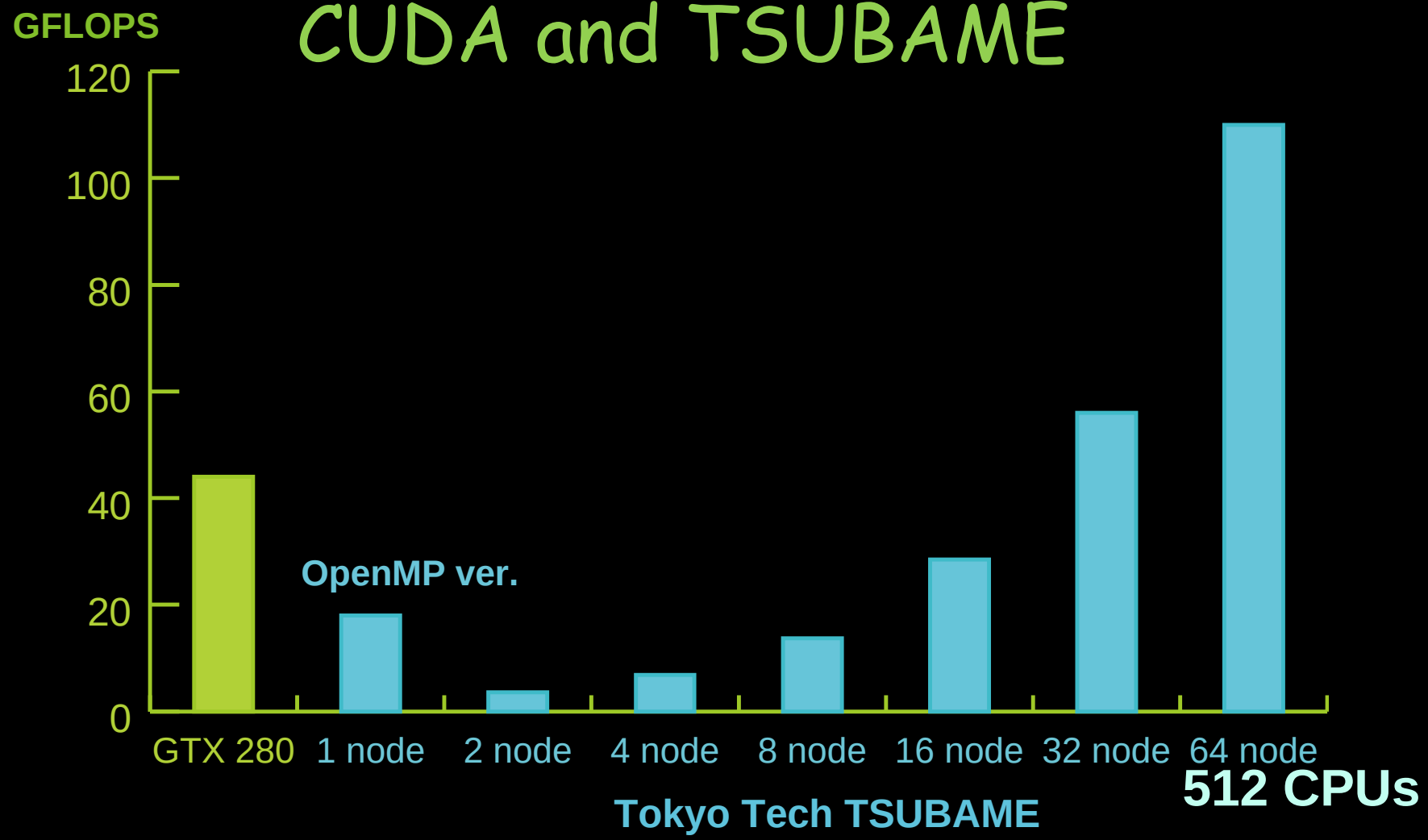

**3-D FFT of size $256^3$**

The bottleneck is floating-point operation in double precision.

Both GPUs are running at 1.3GHz, but product version of S1070 will come with higher clock improve the performance.

Performance is competitive with that of a single-node vector supercomputer NEC SX-6 (64 GFLOPS peak).

# Performance of DP 3-D FFT on CUDA and TSUBAME




**MPI version is used for computation with multiple nodes**

# Performance including Data Transfer

**The Worst Case:**

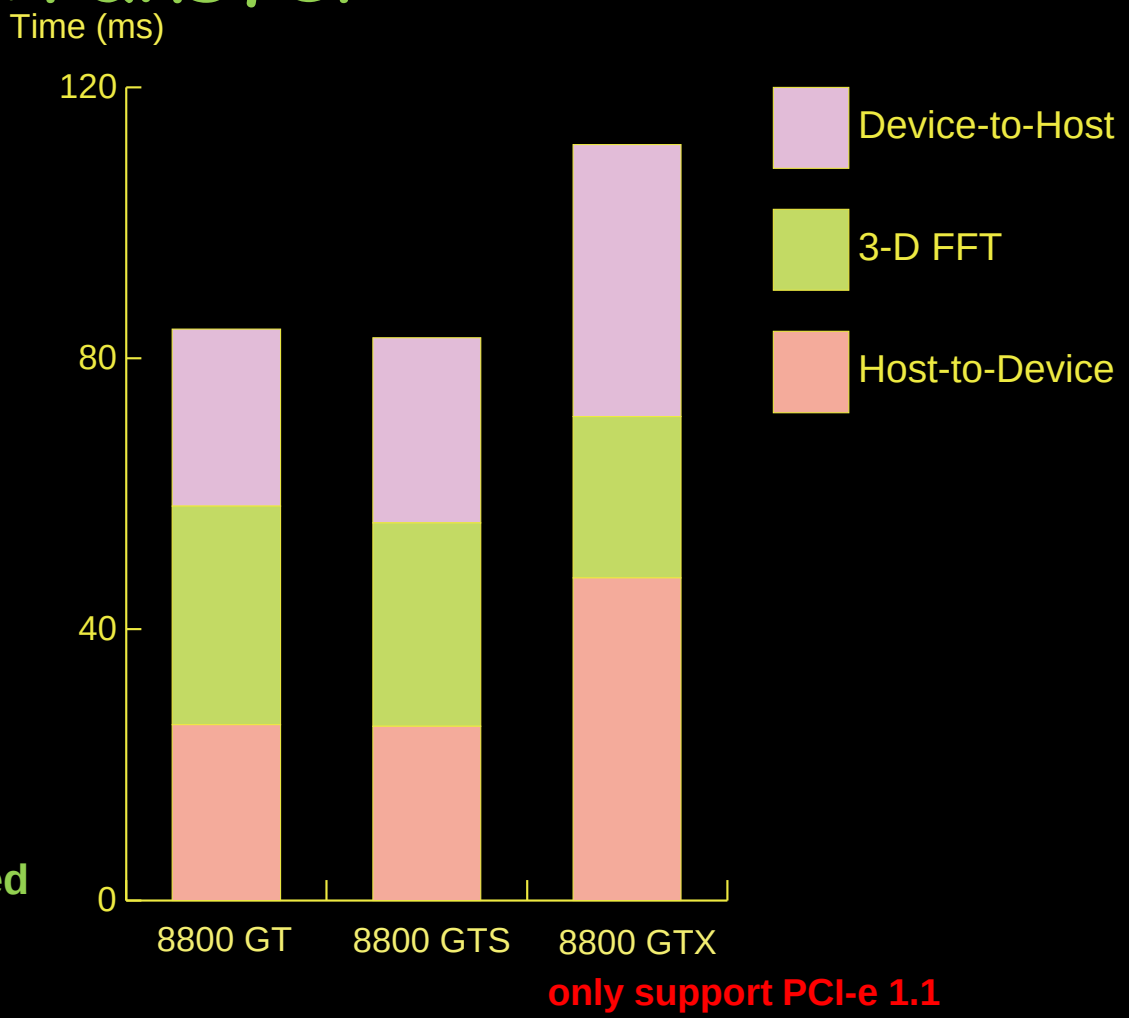
**GPU computes only FFT, and CPU computes all the others.**

**Ex) simply replacing CPU library by GPU**

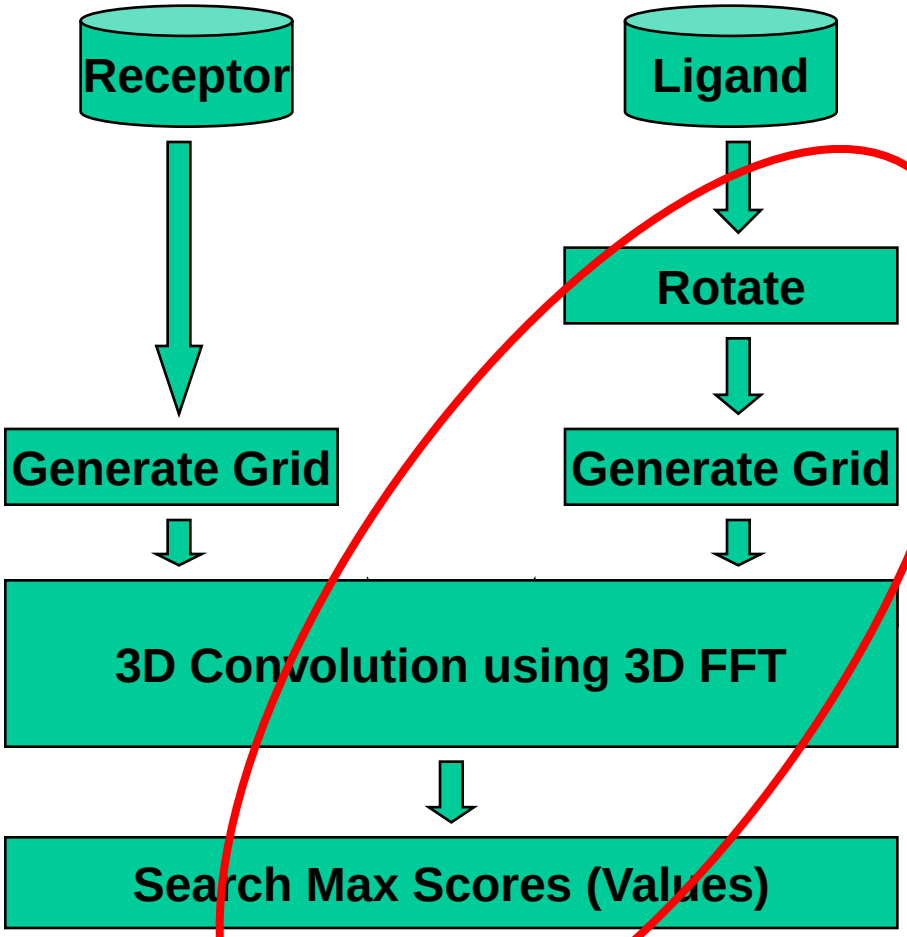
**Ex) data come from I/O devices**

**We have to transfer data between host and device using PCI-Express bus.**

**In the best case, the host CPU is used only to control GPUs.**



# Eliminating the Bandwidth Bottleneck---Entire docking in GPU




**Read geometry data of atoms of Proteins from PDB, preprocess, and transfer both to the GPGPU card.**

**Generate 3D Grid data on the card. For ligands rotate by 6 or 15 degrees increments**

**Conduct two forward transforms, one backward transform, and an element-wise multiplication.**

**Find the best docking position with statistical clustering**

**MAIN LOOP on GPGPU Card**

**=> almost no data transfer between CPU and GPU**

# Raccoon: Acclerated WinHPC GPU Prototype Cluster

- 32 compute nodes
- 128 8800GTS GPGPUs
- one head node.
- Gigabit Ethernet network
- Three 40U rack cabinets.
- Linux or Windows HPC 2008
- Visual Studio 2005 SP1
- nVidia CUDA 2.x

- New nodes for GTX280, Infiniband, …

# Performance Estimation for 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 | 10~13 |

## System Total

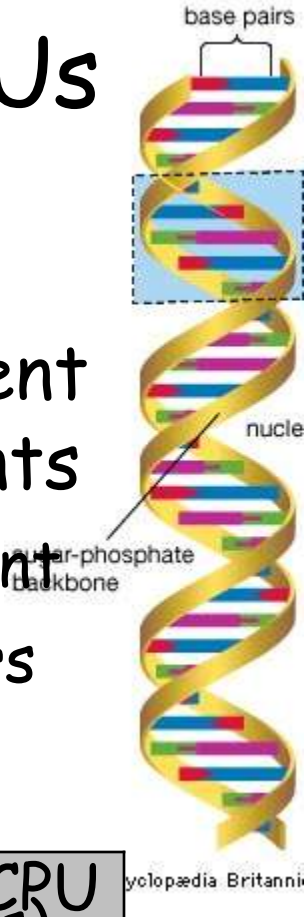
! Only CPUs for TSUBAME. DP=double precision.

| | # of nodes | Power (kW) | Peak (TFLOPS) | Docking (TFLOPS) | MFLOPS/W |
|---|---|---|---|---|---|
| Blue Gene/L (Blue Protein @ AIST) | 4096 (4racks) | 80 | 22.9 | 7.0 | 87.5 |
| TSUBAME | 655 (~70 racks) | ~700 | 50.3 (DP) | 17.5 (DP) | 25 |
| 8800 GTS | 32 (3racks) | 18 | 53.2 | 6.5 | 361 (x4 BG!) |

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

# DNA Giga Sequencing Using GPUs

- Personalized DNA-based Diagnostics and Medicine => Giga Sequencers
- Proposal: GPUs for giga-size short fragment DNA sequencing, for 50,000,000 fragments
  - Rigourous Smith-Watherman gapped alignment
  - Months on a conventional cluster, just 17 hours on our test cluster

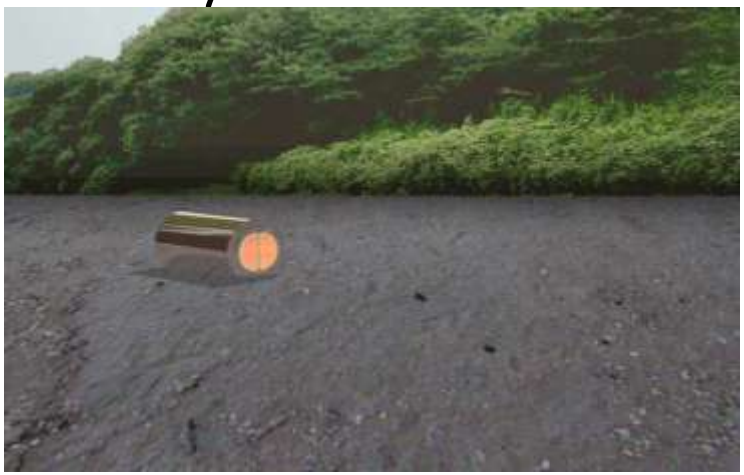



| Type of GPU | # of GPUs | Pruning mode | # of Probes | total CPU (hours) |
|---|---|---|---|---|
| 8800GTS | 128 | off | 10,000,000 | 22.6 |
| Tesla10 | 128 | off | 10,000,000 | 12.2 |
| 8800GTS | 128 | on | 50,000,000 | 16.7 |
| Tesla 10 | 128 | on | 50,000,000 | 8.9 |

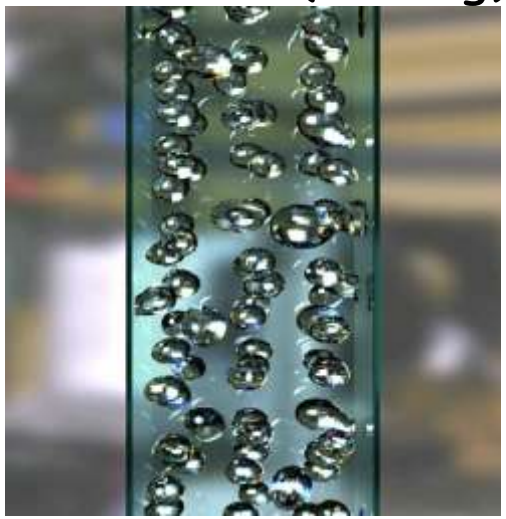
# CFD, esp. Climate and Disaster Prevention on GPUs

(Material from Prof. Takayuki Aoki, Tokyo Tech.)

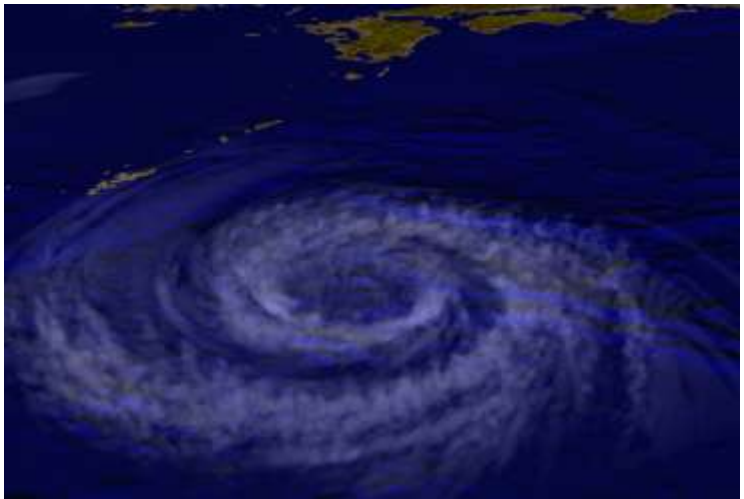
Safety



Nuclear (Cooling)



Weather/Environmental



Civil Engineering

# Rayleigh-Taylor Instability

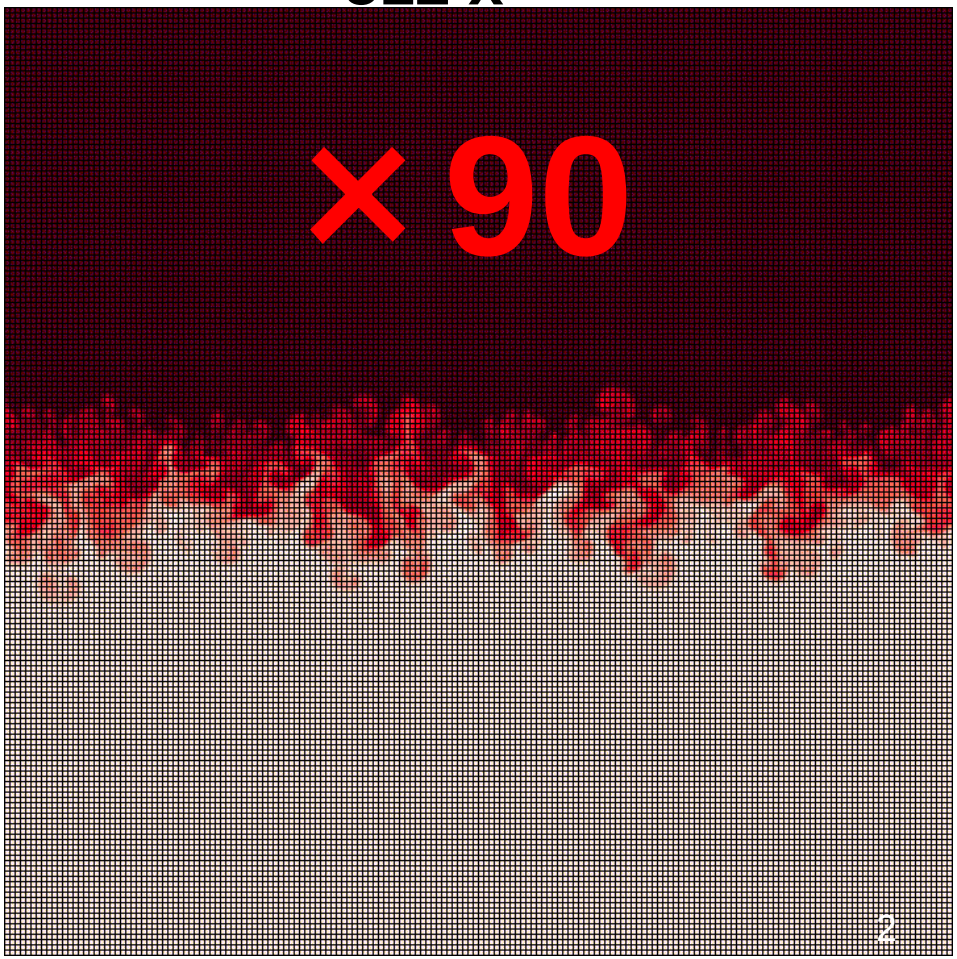
**Heavy fluid lays on light fluid and unstable.**

**Euler equation:**

$$\frac{\partial \boldsymbol{Q}}{\partial t} + \frac{\partial \boldsymbol{E}}{\partial x} + \frac{\partial \boldsymbol{F}}{\partial y} = 0$$

$$\boldsymbol{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix} \quad \boldsymbol{E} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ eu + pu \end{bmatrix} \quad \boldsymbol{F} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ ev + pv \end{bmatrix}$$

**512 x**




**42 GFLOPS** **using GTX280**

# Two-Stream Instability

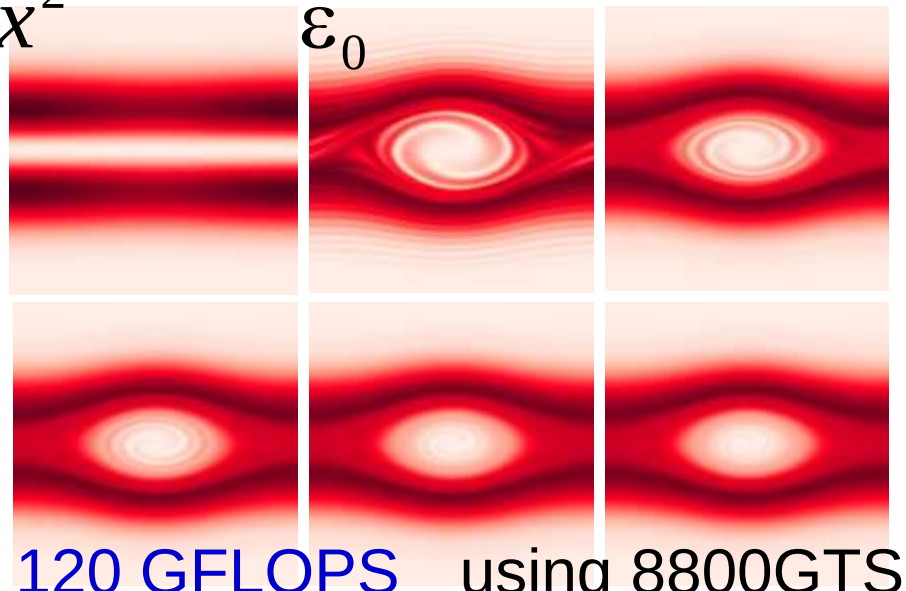
## in Plasma Physics

**Vlasov-Poisson Equation:**

**×130**

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - \frac{eE}{m_e}\frac{\partial f}{\partial v} = 0 \qquad \frac{\partial^2 \phi}{\partial x^2} = \frac{e(n_e - n_i)}{\varepsilon_0}$$

$$\left( E = -\frac{\partial \phi}{\partial x}, \quad n_e = \int f dv \right)$$



120 GFLOPS using 8800GTS

$f$ : **electron distribution function**

$n$ : **electron number density**

# 3-D Computation of Phase Separation

Nanoscience Structure:

158 GFLOPS using GTX280

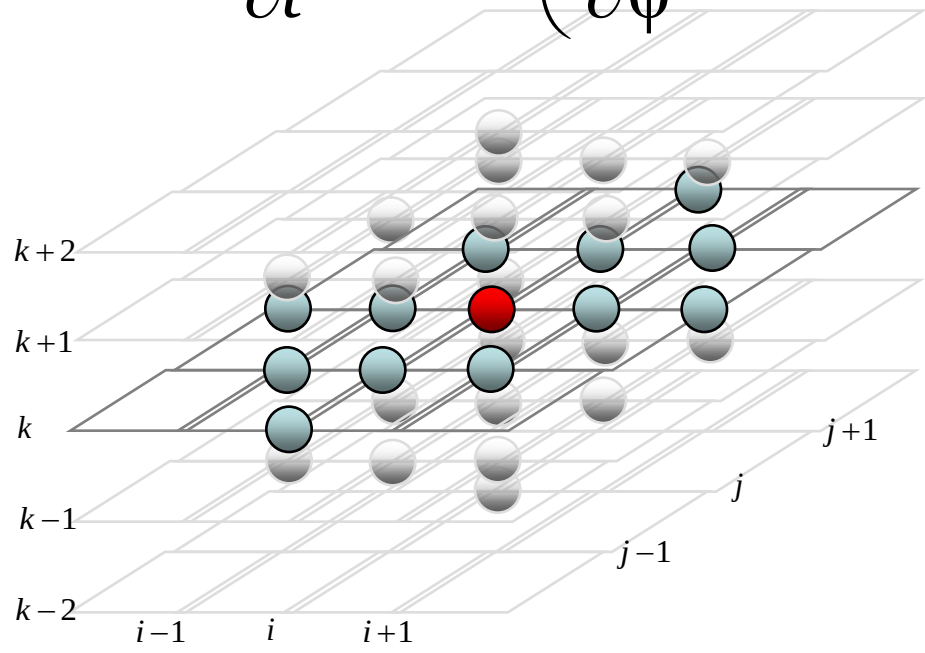
×160

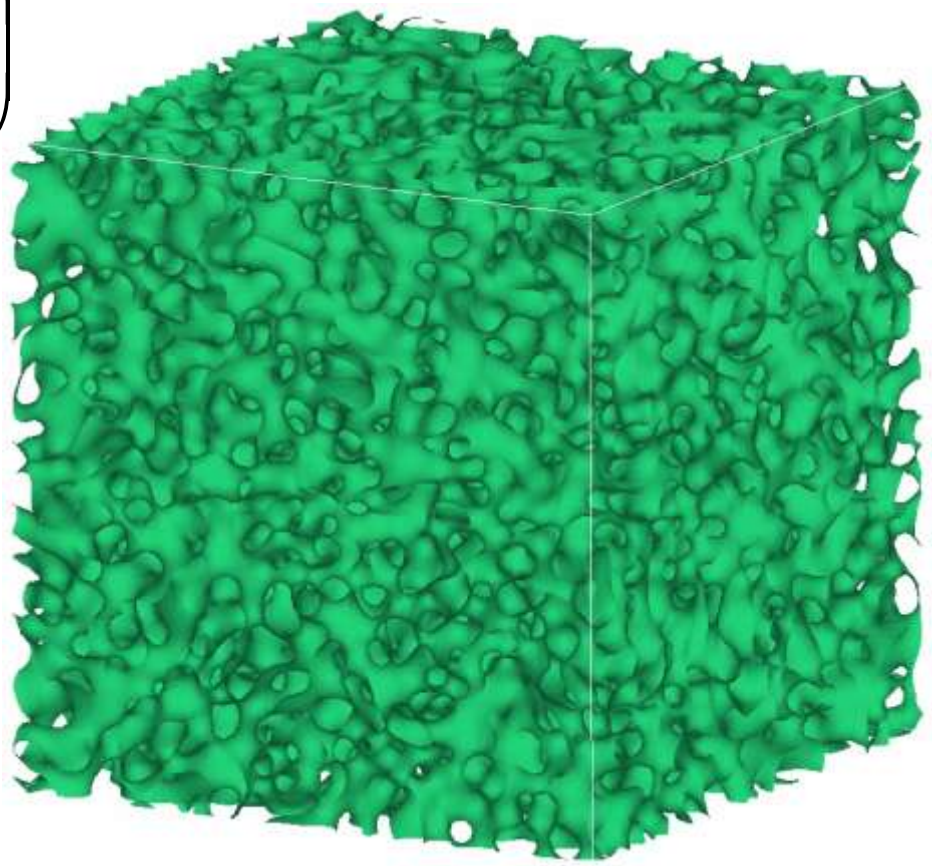
256 x 256 x 256

**Cahn-Hilliard equation:**

$$\frac{\partial \psi}{\partial t} = L\nabla^2\left(\frac{\partial H}{\partial \psi} - C\nabla^2\psi\right)$$

$H$ : free energy



# Real-time Tsunami Simulation

**Collaboration with ADPC (Asian Disaster Preparedness Center) and Japan Meteorological Agency**

## Early Warning System:

**Data Base** ⇒ **Real-time CFD**

**high accuracy**

## Shallow-Water Eq.

**Conservative Form:**

$$\frac{\partial h}{\partial t}+\frac{\partial hu}{\partial x}+\frac{\partial hv}{\partial y}=0$$

**Assuming hydrostatic balance in the vertical direction,**

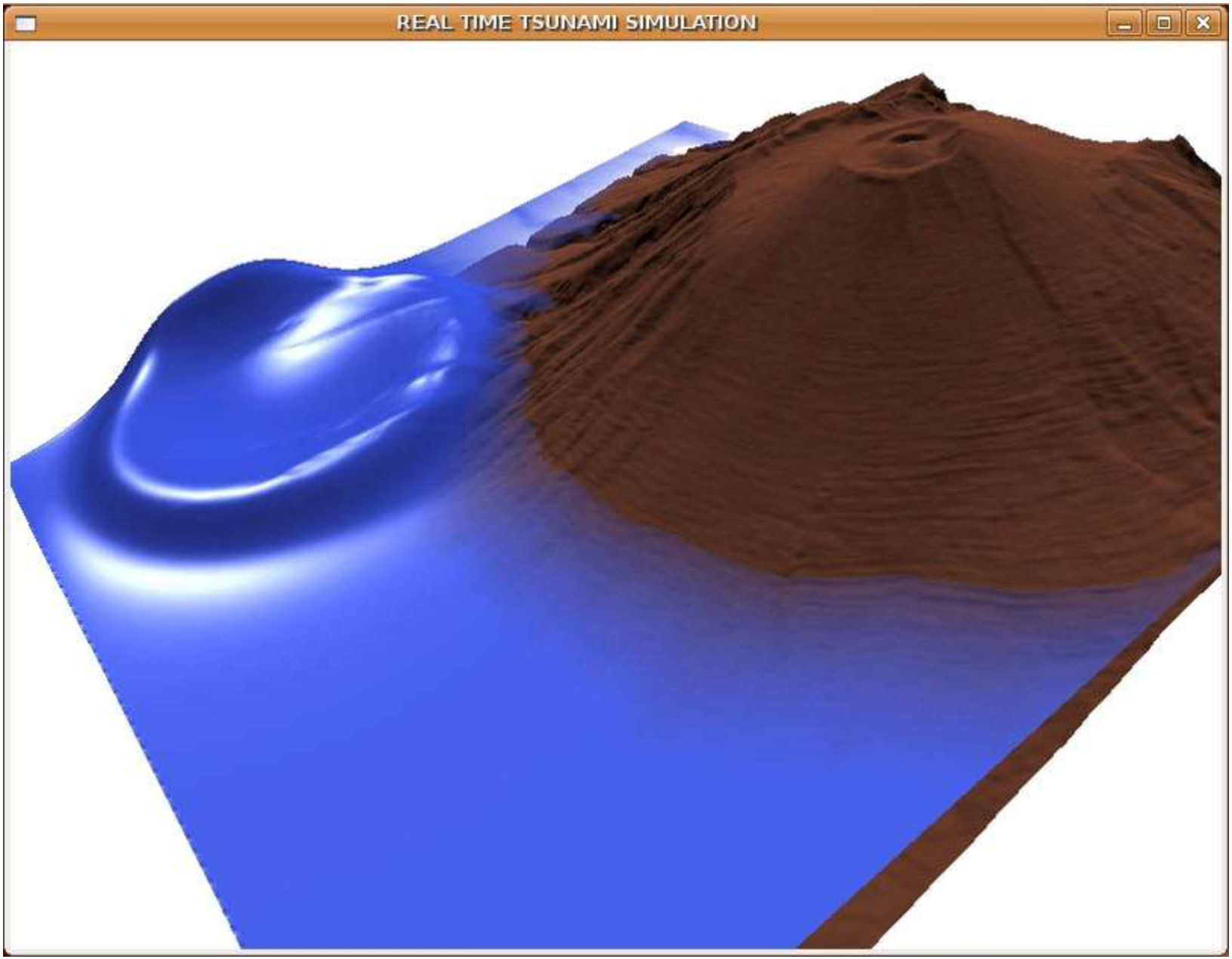
$$\frac{\partial hu}{\partial t}+\frac{\partial}{\partial x}\left(hu^2+\frac{1}{2}gh^2\right)+\frac{\partial huv}{\partial y}=-gh\frac{\partial z}{\partial x}$$

**3D → 2D equation**

$$\frac{\partial hv}{\partial t}+\frac{\partial huv}{\partial x}+\frac{\partial}{\partial y}\left(hv^2+\frac{1}{2}gh^2\right)=-gh\frac{\partial z}{\partial y}$$

# Numerical Methods of Tsunami Simulation

- 2-dimensional Problem : Directional-Splitting Fractional Method
- Point Value Comp. : Characteristic-based Method using Multi-moment Interpolation
- Integral Value Comp. : Conservative Semi-Lagrangian CIP + IDO
- Run-up to dry area: thin water layer and artificial viscosities


REAL TIME TSUNAMI SIMULATION

# GPU Performance

**Speed Comparison**

**x-direction : y-direction = 10 : 7**

**Current Speed-up**

**GPU : CPU = 62 : 1**

**GPU – GeForce GTX280 (sp = 240, clock 1.3Ghz)**

**CPU – Xeon 2.4GHz 6MB Cache Memory**

# Multi-GPU Estimation

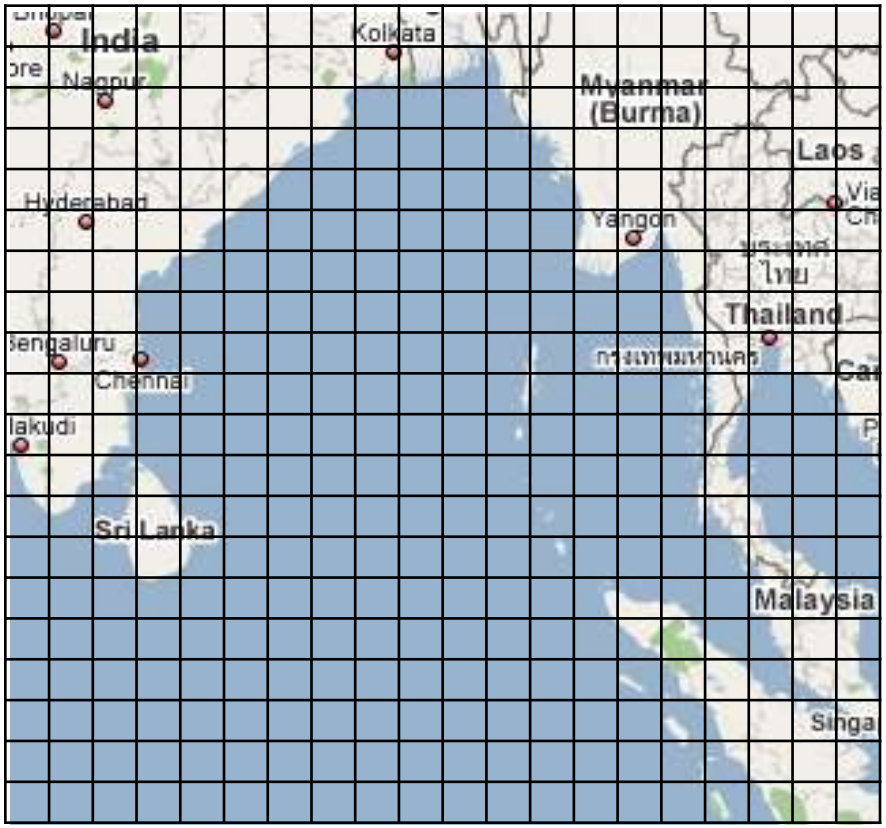
**3000km x 3000km (500m mesh)**

**17280 sec (1 GPU)**

**32 GPU**

**540 sec < 10 min**

**covering Indian Ocean**




**Colioris Force**
**Tidal Potential**
**Wind effect**

# TSUBAME 1.2 Evolution (Oct. 2008)

## The world's first GPU-based SC in the World

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

NEC SX-8i

500GB
48disks

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

**10Gbps+External NW**

Unified Infiniband network

***10,000 CPU Cores***
***300,000 SIMD Cores***
***> 3 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**

**NEW: co-TSUBAME**
**90Node 720CPU (Low Power)**
**~7.2TeraFlops**

PCI-e

ClearSpeed CSX600
SIMD accelerator
**360 648 boards,**
**35 52.2TeraFlops**

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


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

# TSUBAME 1.2. The most Heterogeneous Supercomputer in the world

- Three node configurations with four different processors → >30,000 cores, ~170TFlops system

**SunFire X4600+ 2 TESLAs + ClearSpeed**
- Opteron 2.4GHz 16 cores
- TESLA S1070 (30cores) 2boards
- ClearSpeed X620 (2cores) 1board

→ 78 cores, 330 Gflops peak

x 318nodes

**SunFire X4600+ClearSpeed**
- Opteron 2.4GHz 16 cores
- ClearSpeed X620 (2cores) 1board

→ 18 cores, 157 Gflops peak

x 330nodes

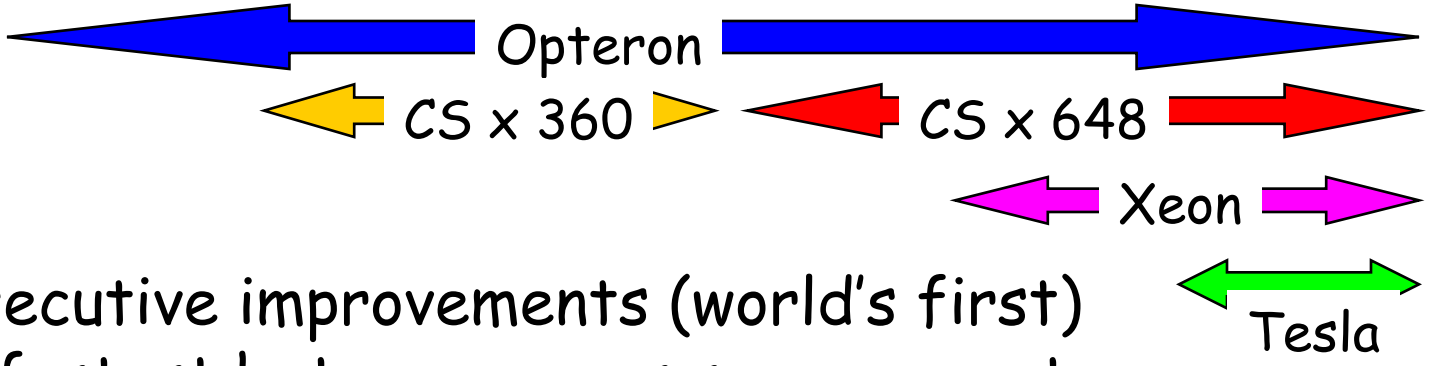
**SunBlade X6250** (TSUBASA cluster)
- Xeon 2.83GHz 8 cores

→ 8 cores, 90.7 Gflops peak

x 90nodes

# TSUBAME in Top500 Ranking w/our Hetero-Linpack

| | Jun 06 | Nov 06 | Jun 07 | Nov 07 | Jun 08 | Nov 08 |
|---|---|---|---|---|---|---|
| Speed(TF) | 38.18 | 47.38 | 48.88 [HPDC 2008] | 56.43 | 67.70 | 77.48 |
| Rank | 7 | 9 | 14 | 16 | 24 | 29 |




- Six consecutive improvements (world's first)
- The 2nd fastest heterogeneous supercomputer in the world (No.1 is RoadRunner)
- Achieved through extremely intricate hetero-algorithmic advances and tuning

# Portfolio of our GPU Computing Base Technologies for HPC & eScience

- TSUBAME 1.2 (680 Teslas) & 2.0
- Kernels(FFT, Dense/Sparse Matrix)
- Parallel Algorithms(Large FFT, LINPACK, CG)
- Task & Resource Mgmt (Heterogeneity, Scheduling, BQ Scheduling, etc.)
- Fault Tolerance(ECC, redundant computation, GPU checkpointing)
- Languages(OpenMP on GPU, Accelerator, MP)
- GPU Low Power computing (power modeling, measurement, optimization)

GPU Leadership from research to deployment(!)

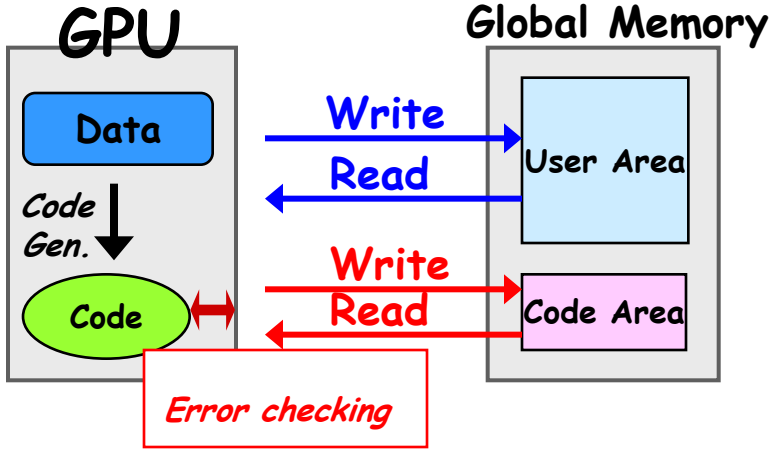
# Software-Based ECC for GPUs (N. Maruyama)

## Possible Collab. w/MSR Vivian Sewelsen and HPC Cluster

**GPU computing reliabiliy**

No ECCs on GPUs yet => Bit flip errors?

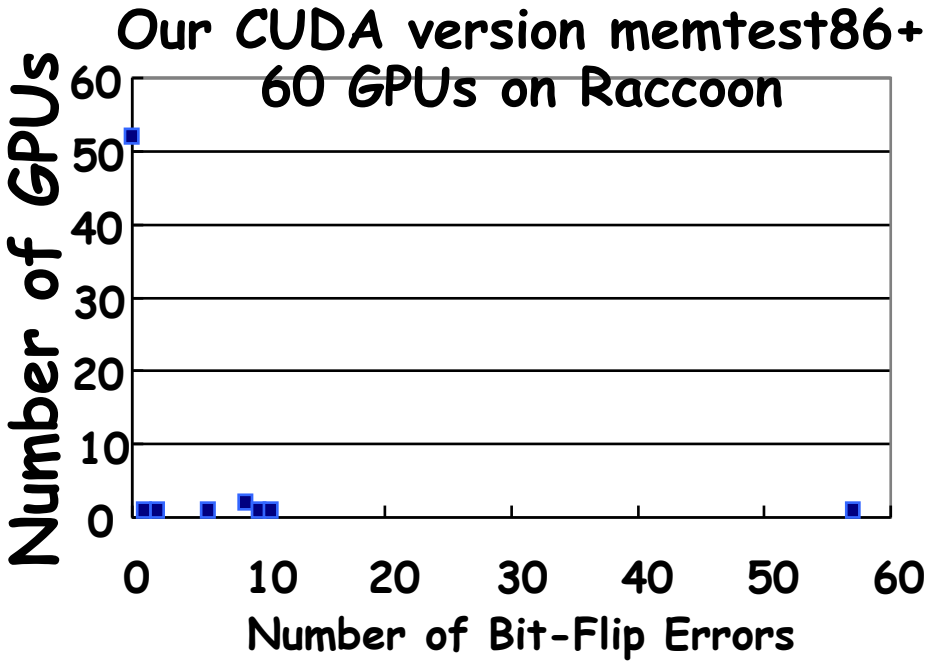
Large scale clusters will be quite problematic

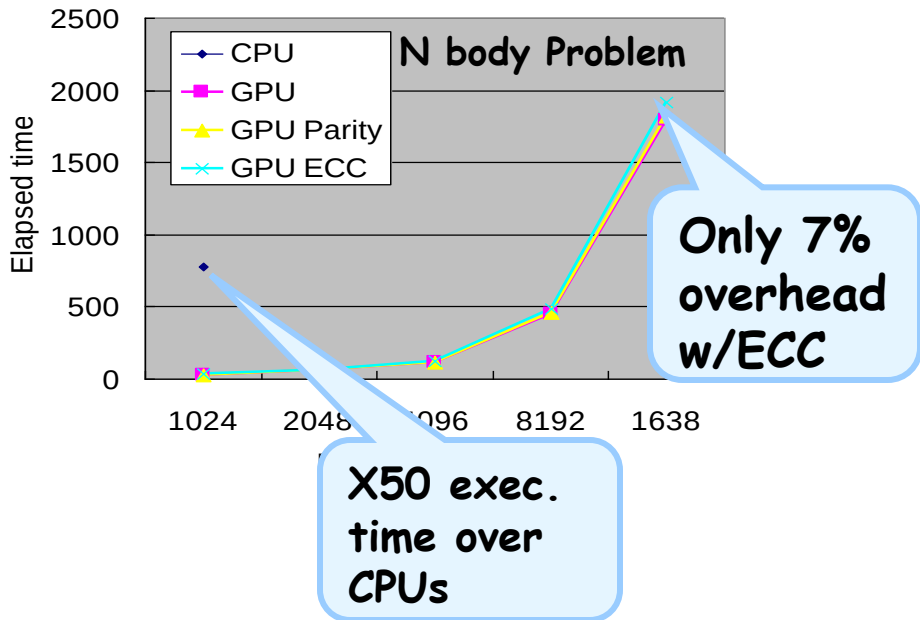



**Software-based ECC on GPUs**

Read : Read ECC data and check

Write : Generate ECC and store alongside data



# Fast Conjugate Gradient Solver on Multi-GPU Systems (Ali Cevahir)

Iterative CG on multi-CUDA GPUs w/mixed precision

Level 1 BLAS => OK for GPUs

Sparse Matrix*Vector => incurs random access NG for GPUs

Utilize various memory on CUDA GPUs

1) Matrix element and index (accessed only once) => global memory

2) JDS method offset table (accessed several times) => constant memory (cached)

3) Vector element (accessed many times) => texture memory (cached)

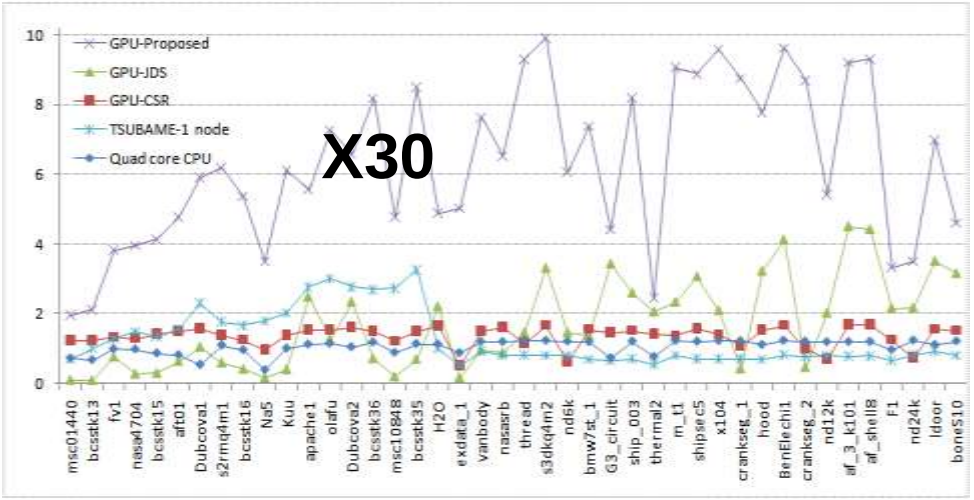
Previous Methods

CRS Method (Cache-based CPUs)

Small # of memory accesses

JDS Method (for vector CPUs)

Continueous memory accesses




**14.5GFlops 4 GPUs (nVidia 8800 GTS) vs. 0.54GFlops 4 Core CPU (Phenom 2.5Ghz) (Sparse Matrix collection from UFlorida)**

GPU-friendly Implementation

1) JDS-like Storage Format

- Efficient memory access
- Adjust alignment for coalescing

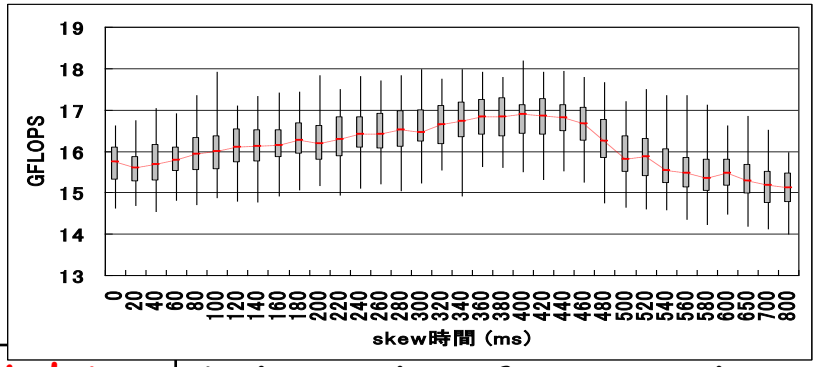
2) CRS-like computing order

- Reduce # of memory accesses

# Large Scale Hybrid CPUs-GPUs 3-D FFT (Y.Ogata)

Intel Skulltrail
-FFTW on Dual QuadCore Xeon E5420
-CUFFT library on Dual GeForce GTX 280

Multi-CPU,multi-GPU => Distribute work based on respective performance models

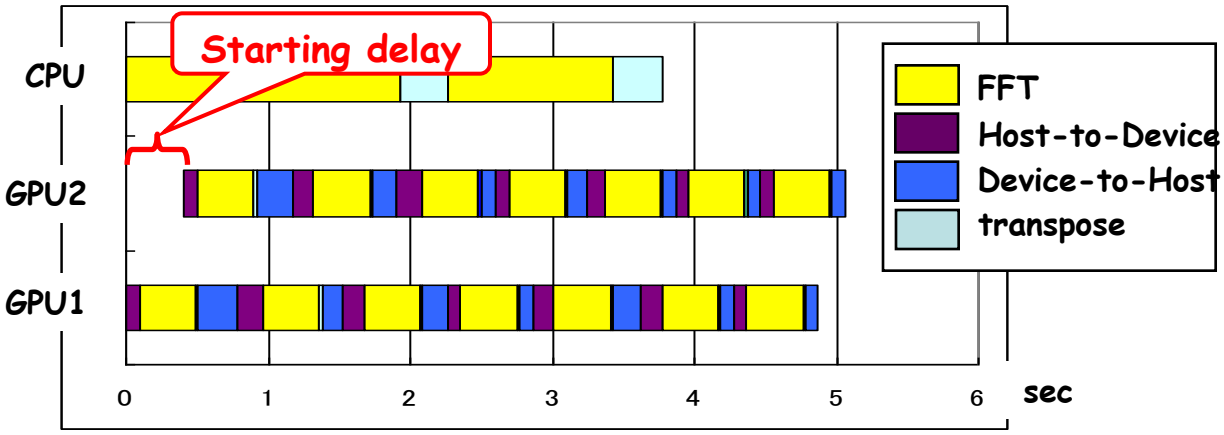


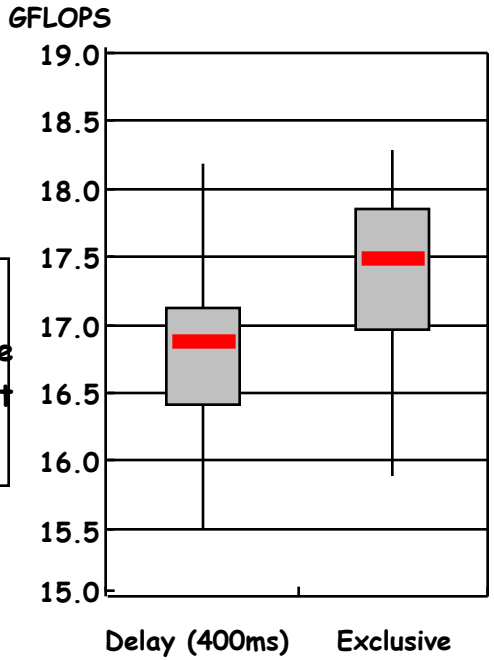

Delay and performance dist.

Problem: perf degredation due to multi-GPU data transfer conflict

Meth1:[Exclusive] Mutually exclude host-device transfer

Meth2: [Delay] delay GPU execution for overlaps

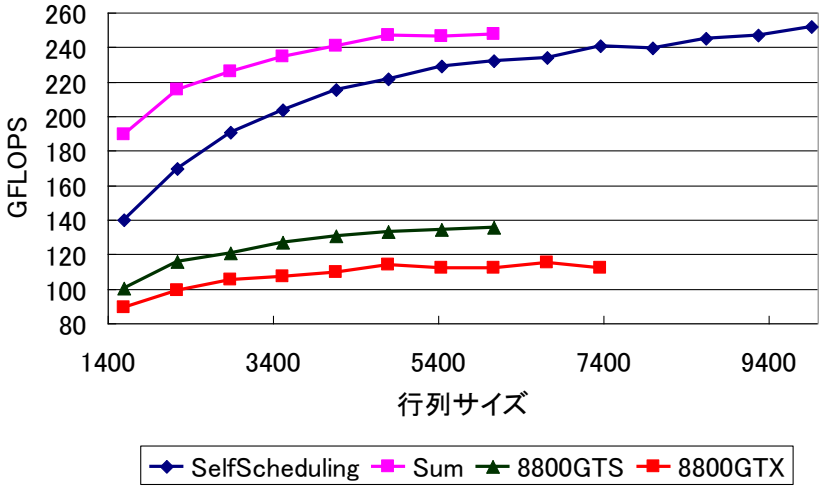


# Dynamic Scheduling of Matrix Kernels on Heterogeneous GPUs (Y. Watanabe)

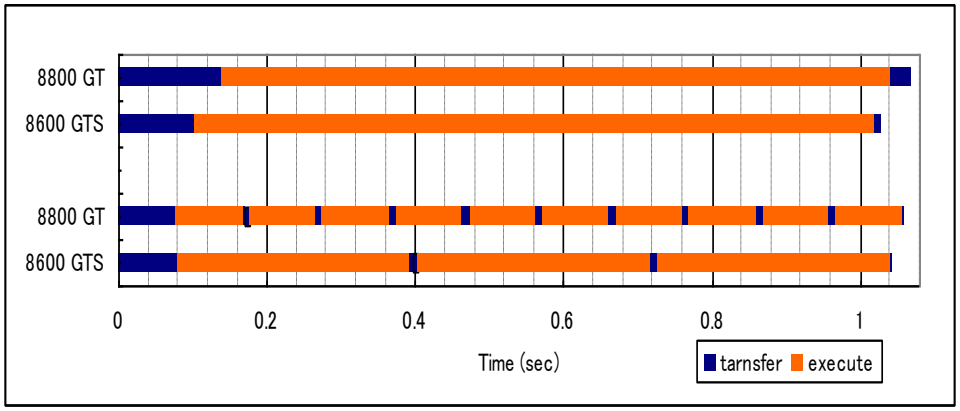
**Rapid GPU Performance Progress and Updates**
**➔ New GPU add-ons will differ in performance**

## Dynamic Load distribution to heterogeneous GPUs

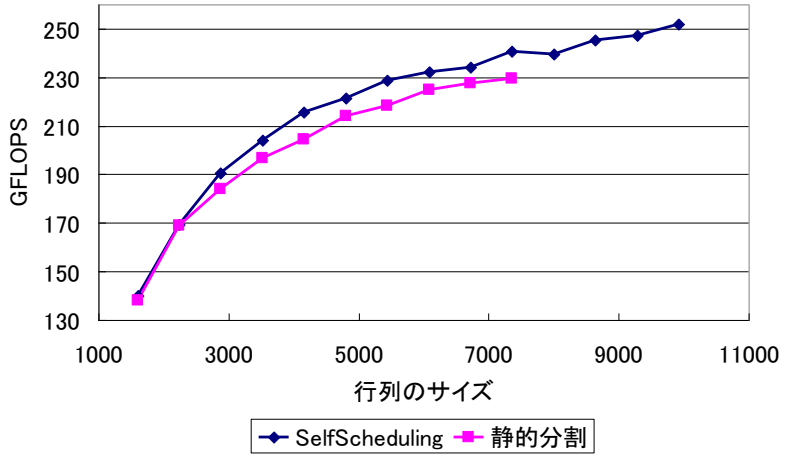



**Thesis: Dynamic task allocation works effectively for load distribution**

**Characteristics**
**Good: No a-priori info necessary**
**Bad: GPU control API Overhead**
**Good: natural distribution of Host-GPU transfers, less collision**




**94% efficiency c.f. summation of single GPU performances**




**Better performance c.f. static allocation**

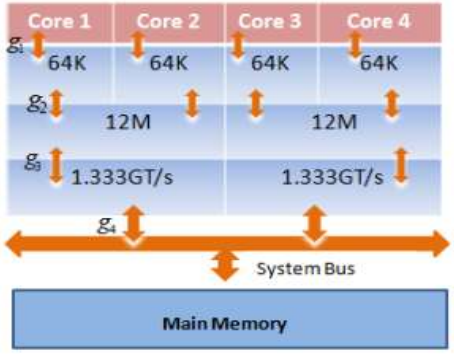
# GPU Computing: Power modeling

(R. Suda@U-Tokyo)

**Problem: To our knowledge there are no tools to predict the power consumption of large-scale parallel program in the algorithm design stage.**

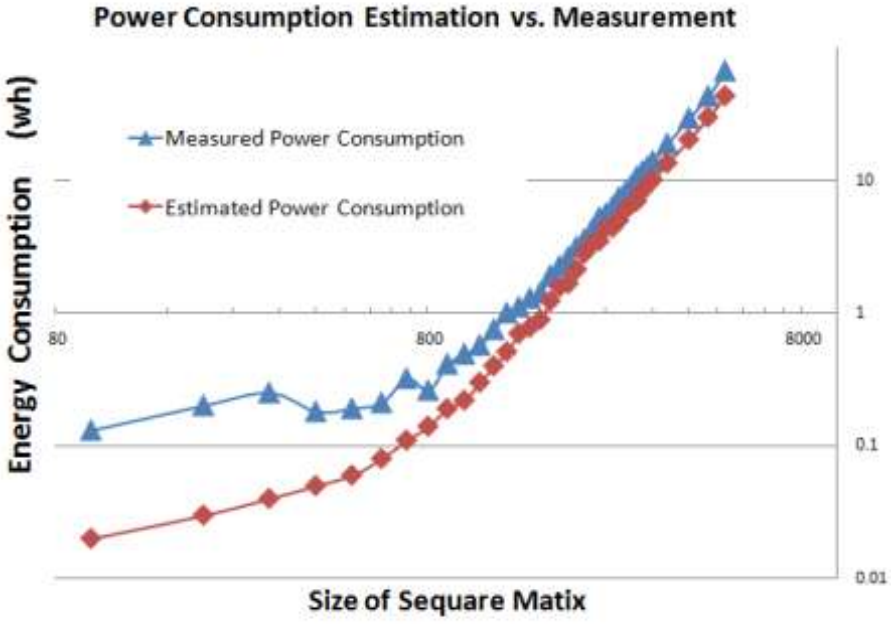
**Approach: BSP model provides parallel framework; BM Model incorporates physical constraints. Extending them with power-relevant characters, and , power model is built, and performance is examined.**

**Results: The power measurement results of *m*atrix computation agrees with the estimated result. Further refinement to the model is needed.**



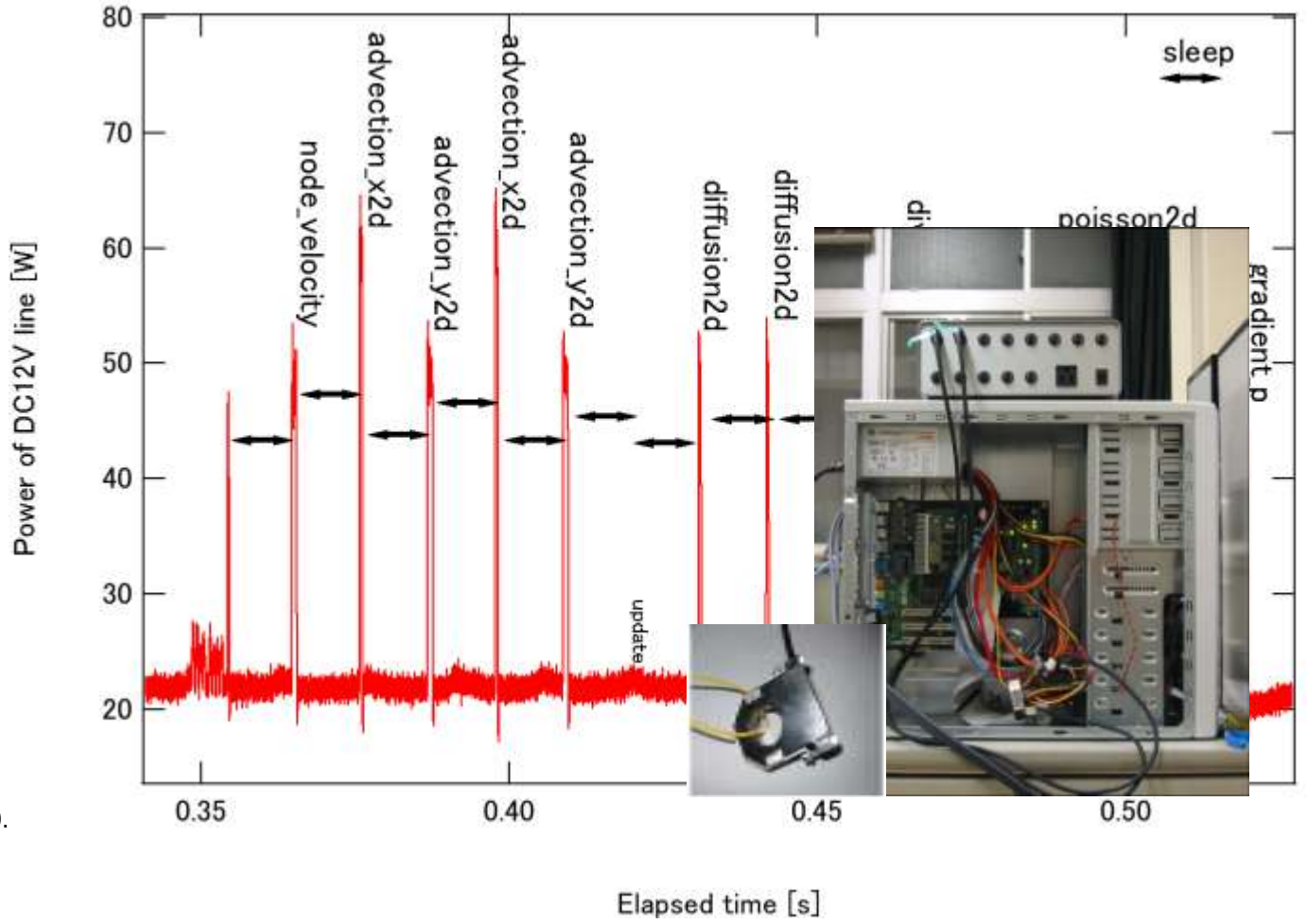

Intel Core 2 Extreme Quad-core Processor

| Level | $p_i$ | $e_i^p$ (W/I) | $g_i$ (GT/s) | $e_i^m$ (W/word) | $m_i$ KB |
|---|---|---|---|---|---|
| 1 | 4 | 4.1E-8 | 300 | 1.5E-5 | 64 |
| 2 | Threads | 2.8E-8 | 300 | 2.1E-5 | 256 |
| 3 | Threads | 2.8E-8 | 75 | 3.0E-5 | 12K |



# Incompressible Fluid Application Power Measurements thru Sleep Insertions (T. Aoki)

```
INT  MAIN(INT ARGC, CHAR *ARGV[]){
    CUDAGETDEVICECOUNT();
＜初期化や初期条件導入＞
＜"計算"開始＞
    DO {
        FOR(JM = 0; JM < 4; JM++) {
          NODE_VELOCITY();
          ADVECTION_X2D(U,U);
          ADVECTION_Y2D(U,V);
          ADVECTION_X2D(V,U);
          ADVECTION_Y2D(V,V);
          UPDATE_VARIABLE(U,UN);
          UPDATE_VARIABLE(V,VN);
           DIFFUSION2D(U);
           DIFFUSION2D(V);
           UPDATE_VARIABLE(U,UN);
             UPDATE_VARIABLE(V,VN);
        TIME += DT;
         }
        DIVERGENCE();
        POISSON2D_MG();
        GRAD_P();
   } WHILE(ICNT++ < 99999 && TIME + 0.
       0.7);
＜ここまで＞
       終了処理
}
```



# Power Efficiency in 3-D FFD (A. Nukada)

| GPU | Computation | Idle | Power | GFLOPS | GFLOPS/W |
|---|---|---|---|---|---|
| RIVA128 | On CPU | 126 W | 140 W | 10.3 | 0.074 |
| 8800 GT | On GPU | 180 W | 215 W | 62.2 | 0.289 |
| 8800 GTS | On GPU | 196 W | 238 W | 67.2 | 0.282 |
| 8800 GTX | On GPU | 224 W | 290 W | 84.4 | 0.291 |

CUDA GPUs have four times
higher power efficiency than CPU
in high-performance FFT.

me

RIVA128 is an old, low-power GPU,
to measure pure power consumption
of host system (CPU, chipset, memory).
The interface is legacy PCI.

# Onto TSUBAME 2.0 & 2.1

- TSUBAME 2.0 will be
  - Deployed 1H 2010
  - A Petascale machine w/GPU acceleration
  - Will be VERY GREEN (same energy envelope)
  - Will be a Cloudy supercomputer, virtualization/multi-OS dynamic deployment
  - Will be a Supercomputer for Everyone
- TSUBAME will run Windows HPC (!)
  - Users will have a choice along w/Linux
  - The first Petascale Windows HPC Cluster?

# Road to Exascale and Personal Petascale