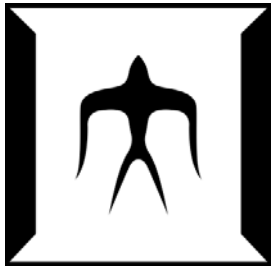
“Rise of the Commodity Vectors”
or
Democratization of Supercomputing

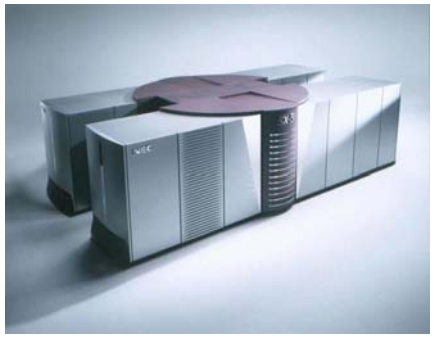
Satoshi Matsuoka, Professor
Takayuki Aoki, Professor
Akira Nukada, PhD Researcher

Tokyo Institute of Technology

NVISION2008, Aug. 26, 2008



Vector Machines- NEC SX



- ACOS/SX-1
- SX-2 (1983): Bipolar, 4-wide, 1.3GFlops, single CPU (c.f. Intel 80286/80287)
- SX-3: (1989): Bipolar, 8-wide, 22GFLOPS(2x4CPUs) (Intel 486)
- SX-4 (1994): CMOS 8-wide 64GF/node (2GF x 32CPUs) (Pentium)
- SX-5 (1998): 16wide, 128GF/node (8GF x 16 CPUs) (Pentium III)
- SX-6 (2001): 8-wide x 2 clock, 64GF/node (8x8CPUs), core of ES
- SX-7 (2002): 8-wide x 2 clock, 282GF/node (9GF x 32 CPUs)
- SX-8 (2004): 8-wide 2Ghz vector, 128GF/node (16GF x 8CPUs)
- SX-9 (2007): 8-wide x 4 3.2 Ghz 102GF/CPU, 1.6TF/node, 128GB/s inter-node BW

Glory Days of Vectors…just 12 years ago

Cray	71	
NEC	40	
Fujitsu	33	(#2)
Hitachi	9	
CM-2	7	
Total	160	
x86 (Meiko)	3	

"Cretaceous"

2008/06/24 16:24:27 ics ▸ Sublist Generator

TOP500 Sublist Generator

R_{max} and R_{peak} values are in GFlops. For more details about other fields, check the TOP500 description.

40 entries found.

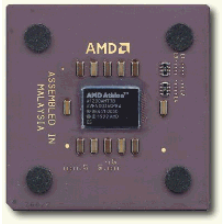
Top500 June 1996, 40 NEC SXs

Rank	Site	System	Cores	R_{max}	R_{peak}
10	HWW/Universitaet Stuttgart Germany	SX-4/32 NEC	32	66.53	64
11	NEC Fuchu Plant Japan	SX-4/32 NEC	32	66.53	64
24	Japan Marine Science and Technology Japan	SX-4/20 NEC	20	42.4	40
25	National Research Institute for Metals Japan	SX-4/20 NEC	20	42.4	40
26	Toyota Central Research & Development Japan	SX-4/20 NEC	20	42.4	40
30	National Aerospace Laboratory (NLR) Netherlands	SX-4/16 NEC	16	34.42	32
31	National Cardiovascular Center Japan	SX-4/16 NEC	16	34.42	32
41	Swiss Scientific Computing Center (CSCS) Switzerland	SX-4/12 NEC	12	25.8	24
49	Atmospheric Environment Service (AES) Canada	SX-3/44R NEC	4	23.2	25.6
50	Tohoku University Japan	SX-3/44R NEC	4	23.2	25.6
55	Atmospheric Environment Service (AES) Canada	SX-3/44 NEC	4	20	22
59	Institute for Molecular Science Japan	SX-3/34R NEC	3	17.4	19.2
60	ATR Optical Communication Lab Japan	SX-4/8 NEC	8	17.2	16
61	Atmospheric Environment Service (AES) Canada	SX-4/8 NEC	8	17.2	16
62	Danish Meteorological Institute Denmark	SX-4/8 NEC	8	17.2	16
63	National Geographic Agency Japan	SX-4/8 NEC	8	17.2	16
111	Veritas DGC United States	SX-4/6 NEC	6	12.9	12
136	German Aerospace Laboratory (DLR) Germany	SX-3/24R NEC	2	11.6	12.8
137	National Institute for Fusion Science Japan	SX-3/24R NEC	2	11.6	12.8

Japan had ~30% performance share as a country

Rise of the Commodity Clusters

High Performance Commodity Computing
-High Performance x86 CPUs
-Fast Commodity Interconnect
-Cluster Software

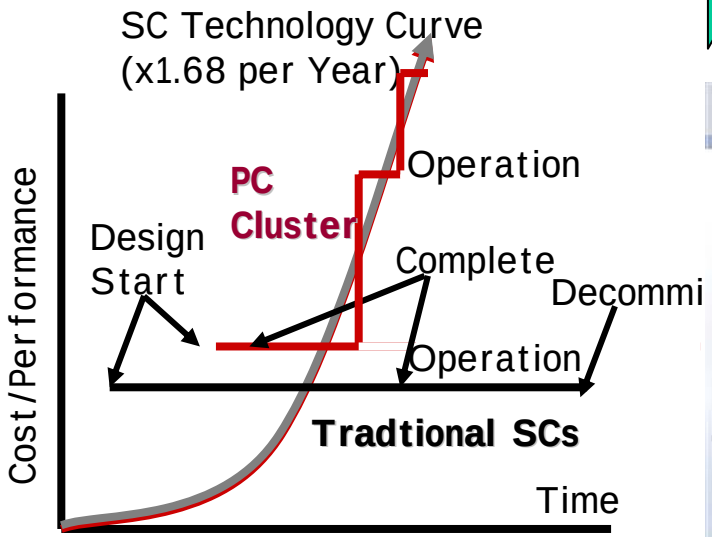


High-Performance x86 CPUs

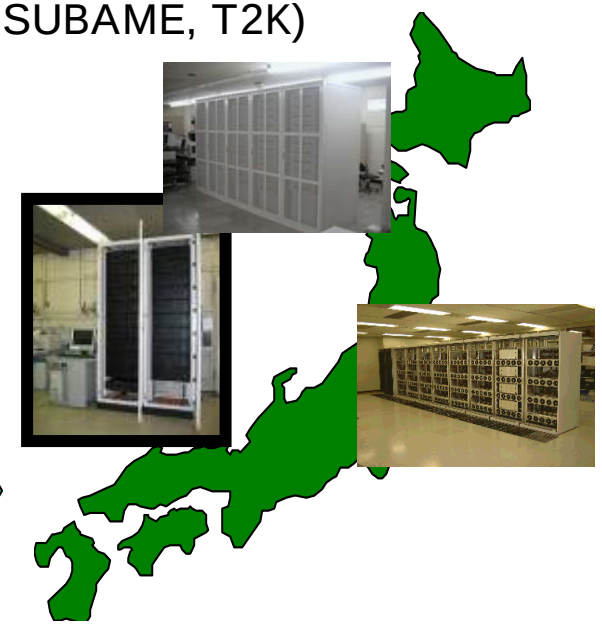
Myrinet, Infiniband, etc.

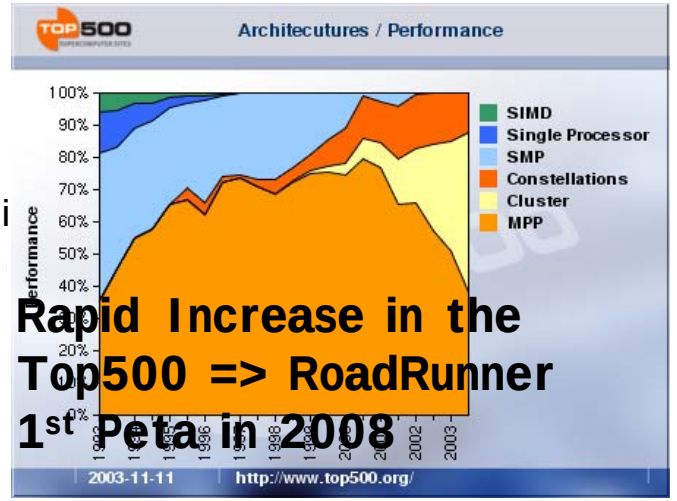
Rise and spread of Commodity Clusters and increase in their size

Real-time tracking of the technology curve



Widespread Use of Clusters: Small to very large (e.g. TSUBAME, T2K)

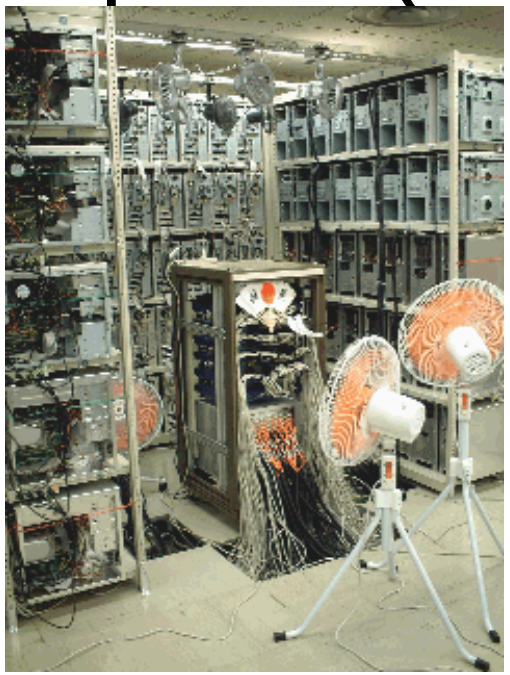




Rapid Increase in the Top500 => RoadRunner 1st Peta in 2008

Presto III: The 2nd Fastest PC Cluster on the Top 500 (2002)

- Collaboration with AMD, Bestsystems Co., Tyan, Appro, Myricom
- 256 node/512 CPU AthlonMP 1900+ Rpeak 1.6 TeraFlops (< 1/50 TSUBAME)
- Full Myrinet 2K network
- June 2002 47th on Top 500, 716GFlops
  - 2nd Fastest PC cluster
  - (lost to a German machine)



TSUBAME-Bridging the Cluster and Vector Acceleration

Commodity Democratization of Supercomputing Now

The TSUBAME 1.0 “Supercomputing Grid Cluster” Spring 2006 at Tokyo Tech

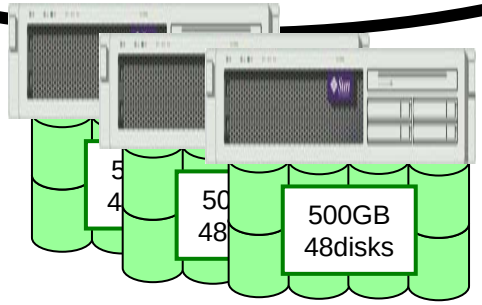
Unified IB network
Voltaire ISR9288 Infiniband 10Gbps
x2 (DDR next ver.)
~1310+50 Ports
~13.5Terabits/s (3Tbits bisection)

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

NEC SX-8i
(for porting)



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

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


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

TSUBAME Architecture =

Commodity PC Cluster

+

Traditional FAT node Supercomputer

+

The Internet & Grid Services

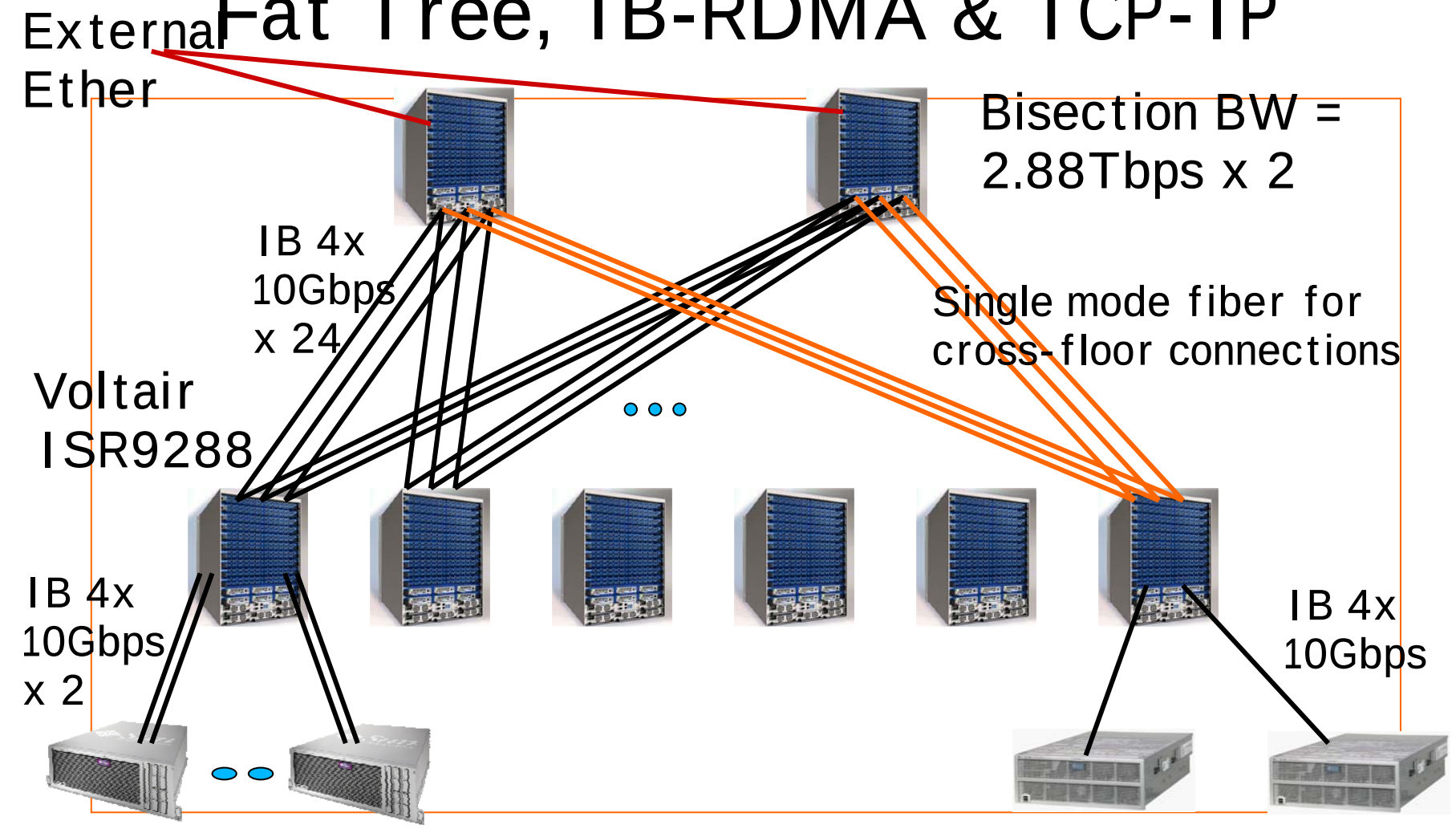
+

(Modern) Commodity SIMD-Vector Acceleration

+

iPod (HW integration & PC Ecosystem)

TSUBAME Network: (Restricted) Fat Tree, IB-RDMA & TCP-IP



X4600 x 120nodes (240 ports) per switch
=> 600 + 55 nodes, 1310 ports, 13.5Tbps

X4500 x 42nodes (42 ports)
=> 42ports 420Gbps

TSUBAME as No.1 in Japan

TITECH GRID

東京工業大学
Tokyo Institute of Technology

GSIC
Global Scientific Information and Computing Center

>>

北海道大学 HOKKAIDO UNIVERSITY
東京大学 THE UNIVERSITY OF TOKYO
東北大学 TOHOKU UNIVERSITY
京都大学
国立大学法人 名古屋大学
大阪大學
九州大学 KYUSHU UNIVERSITY

All University National Centers COMBINED

>85 TeraFlops
1.1Petabyte
4 year procurement cycle

Has beaten the Earth Simulator

Has beaten all the other Univ. centers combined

Total 45 TeraFlops,
350 Terabytes

TSUBAME HPL Results as of May 8, 2006 (Opteron Only)

```
T/V                N    NB     P     Q               Time          Gflops
--------------------------------------------------------------------------------
WR10R2R4     1334160   240    36   144           41462.22      3.818e+04
--------------------------------------------------------------------------------
```

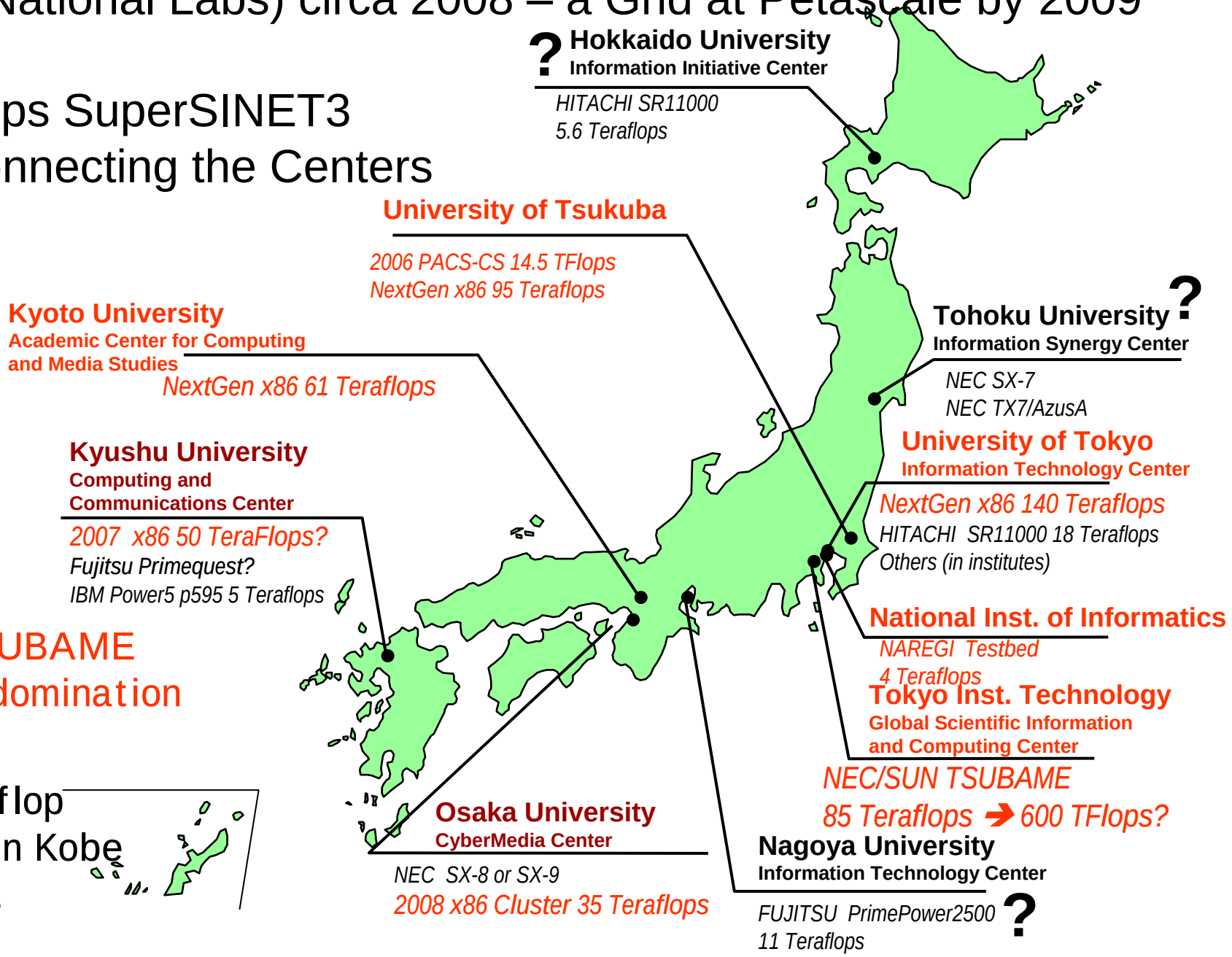
Rank	Site	Computer	Processors	Year	R_{max}	R_{peak}
1	DOE/NNSA/LLNL United States	BlueGene/L - eServer Blue Gene Solution IBM	131072	2005	280600	367000
2	IBM Thomas J. Watson Research Center United States	BGW - eServer Blue Gene Solution IBM	40960	2005	91290	114688
3	DOE/NNSA/LLNL United States	ASC Purple - eServer pSeries p5 575 1.9 GHz IBM	10240	2005	63390	77824
4	NASA/Ames Research Center/NAS United States	Columbia - SGI Altix 1.5 GHz, Voltaire Infiniband SGI	10160	2004	51870	60960
5	Sandia National Laboratories United States	Thunderbird - PowerEdge 1850, 3.6 GHz, Infiniband Dell	8000	2005	38270	64512
6	Sandia National Laboratories United States	Red Storm Cray XT3, 2.0 GHz Cray Inc.	10880	2005	36190	43520
7	The Earth Simulator Center Japan	Earth-Simulator NEC	5120	2002	35860	40960
8	Barcelona Supercomputer Center Spain	MareNostrum - JS20 Cluster, PPC 970, 2.2 GHz, Myrinet IBM	4800	2005	27910	42144
9	ASTRON/University Groningen Netherlands	Stella - eServer Blue Gene Solution IBM	12288	2005	27450	34406.4
10	Oak Ridge National Laboratory United States	Jaguar - Cray XT3, 2.4 GHz Cray Inc.	5200	2005	20527	24960

- N : 1334160
- NB : 240
- PMAP : Row-major process map
- P : 36
- Q : 144
- PFACT : Right
- NBMIN : 4
- NDIV : 2
- RFACT : Right
- BCAST : 1ring
- DEPTH : 1
- SWAP : Mix (threshold = 240)
- L1 : transposed form
- U : transposed form
- EQUIL : yes
- ALIGN : 8 DP words

- **38.18 TeraFlops@648 node**
- **~14 Terabytes memory**

Japan's 9 Major University Computer Centers (excl. National Labs) circa 2008 – a Grid at Petascale by 2009

>40Gbps SuperSINET3
Interconnecting the Centers



x86 TSUBAME
sibling domination

10 Petaflop
center in Kobe
by 2012

From Glory Days to Near Extinction...in 10+ years

Cray	**71**	
NEC	**40**	
Fujitsu	**33**	**(#2)**
Hitachi	**9**	
CM-2	**7**	
Total	**160**	
x86 (Meiko)	**3**	

"Cretaceous"

2008/06/24 16:24:27 ics ▸ Sublist Generator

TOP500 Sublist Generator

R_{max} and R_{peak} values are in GFlops. For more details about other fields, check the TOP500 description.

40 entries found.

Top500 June 1996, 40 NEC SXs

Rank	Site	System	Cores	R_{max}	R_{peak}
10	HWW/Universitaet Stuttgart Germany	SX-4/32 NEC	32	66.53	64
11	NEC Fuchu Plant Japan	SX-4/32 NEC	32	66.53	64
24	Japan Marine Science and Technology Japan	SX-4/20 NEC	20	42.4	40
25	National Research Institute for Metals Japan	SX-4/20 NEC	20	42.4	40
26	Toyota Central Research & Development Japan	SX-4/20 NEC	20	42.4	40
30	National Aerospace Laboratory (NLR) Netherlands	SX-4/16 NEC	16	34.42	32
31	National Cardiovascular Center Japan	SX-4/16 NEC	16	34.42	32
41	Swiss Scientific Computing Center (CSCS) Switzerland	SX-4/12 NEC	12	25.8	24
49	Atmospheric Environment Service (AES) Canada	SX-3/44R NEC	4	23.2	25.6
50	Tohoku University Japan	SX-3/44R NEC	4	23.2	25.6
55	Atmospheric Environment Service (AES) Canada	SX-3/44 NEC	4	20	22
59	Institute for Molecular Science Japan	SX-3/34R NEC	3	17.4	19.2
60	ATR Optical Communication Lab Japan	SX-4/8 NEC	8	17.2	16
61	Atmospheric Environment Service (AES) Canada	SX-4/8 NEC	8	17.2	16
62	Danish Meteorological Institute Denmark	SX-4/8 NEC	8	17.2	16
63	National Geographic Agency Japan	SX-4/8 NEC	8	17.2	16
111	Veritas DGC United States	SX-4/6 NEC	6	12.9	12
136	German Aerospace Laboratory (DLR) Germany	SX-3/24R NEC	2	11.6	12.8
137	National Institute for Fusion Science Japan	SX-3/24R NEC	2	11.6	12.8

"Paleogene"

Japan as a country now has 4% share --- now beaten by France

(Top500 Jun 2008 --- just 1 SX)

2008/06/24 16:29:29 ics ▸ Sublist Generator

TOP500 Sublist Generator

Top500 Nov 2007, 2 NEC SXs

R_{max} and R_{peak} values are in TFlops. For more details about other fields, check the TOP500 description.

2 entries found.

Rank	Site	System	Cores	R_{max}	R_{peak}
30	The Earth Simulator Center Japan	Earth-Simulator NEC	5120	35.86	40.96
200	HWW/Universitaet Stuttgart Germany	SX8/576M72 NEC	576	8.92	9.22

So is vector computing extinct?

NO(!)

Rise of the Commodity Vectors

New Era of Commodity Vector Accelerated Supercomputing

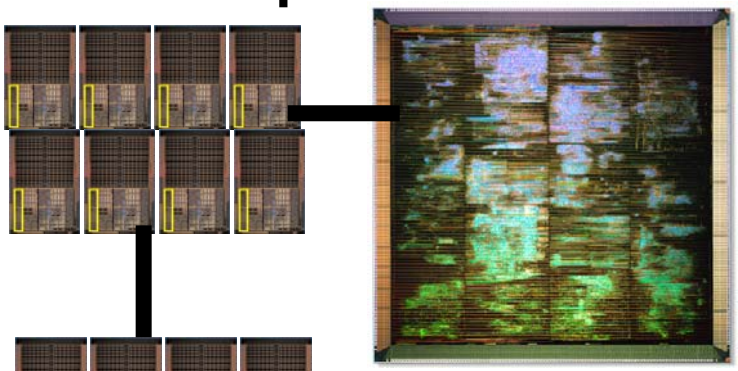
Circa 2004 My Prediction for a Petaflops Machine in 2004 (as TSUBAME was being designed)

"Future AV Vector-Parallel"

IBM/Toshiba/SONY Cell (Chip Vector) + SMT/CMT
256GF-1TFlops (@0.065-0.03)

OR

"Future PC Vector-Graphics"



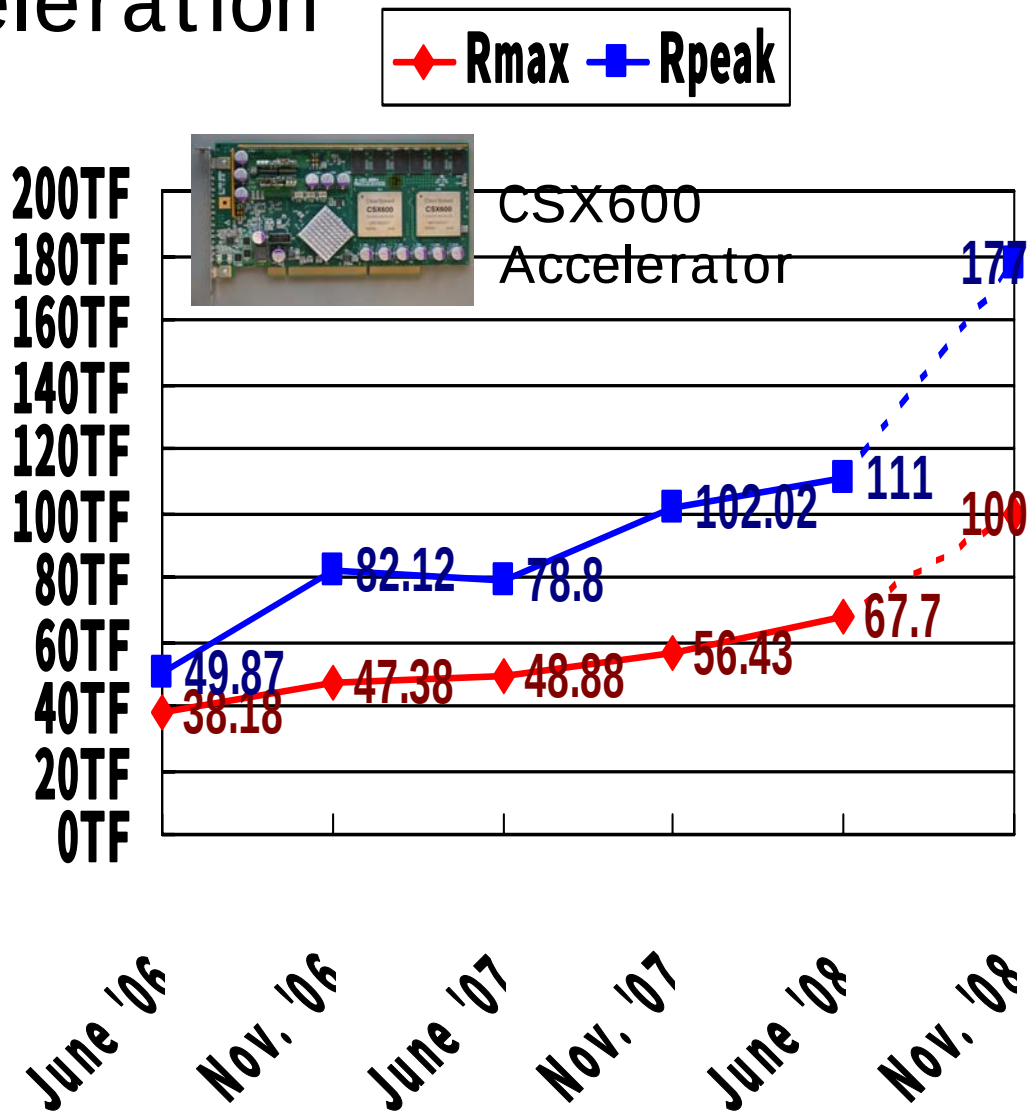
GPU or DSP, ~1TFlops @ 0.045

Multicore SMT/CMT, ~50Gflops@0.065 µ

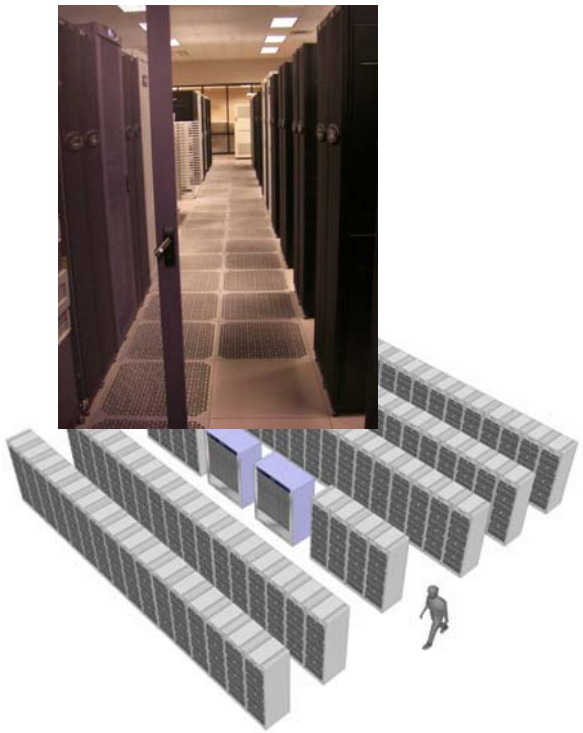
100CPUs/Rack => A 100 Teraflops per Rack

TSUBAME: 5 consecutive Top500 Performance Increase via SIMD-Vector Acceleration

- First ever "Heterogeneous" Architecture on Top500 w/47.38TF (#9 Nov. 2006 28th Top 500)
  - 648 nodes 360 Accelerators
- "Heterogeneous HPL" algorithm published @ IEEE IPDPS 2008
- Continued improvements via
  - Adding more Clearspeed boards (648)
  - Algorithmic Improvements
  - Various Tuning
- Now at 67.7 TF for the June 2008 (31st) Top500



In the Large-Scale Supercomputing Landscape, The 1st Petaflops achieved by commodity vector hybrid cluster



2008Q1 TACC/Sun "Ranger"
~52,600 "Barcelona" Opteron CPU Cores, ~500TFlops
~100 racks, ~300m2 floorspace
2.4MW Power, 1.4km IB cx4 *copper* cabling
2 Petabytes HDD

2008 IBM Roadrunner

~12000 Cell Processors
The first Petaflops Sustained Machine on the Top500
(June 2008, 1.03 PetaFlops)

2008 LLNL/IBM "BlueGene/P"
~300,000 PPC Cores, ~1PFlops
~72 racks, ~400m2 floorspace
~3MW Power, *copper* cabling

> 10 Petaflops
> million cores
> 10s Petabytes
for 2011-2012 in the US, Japan, (EU), …

Significance of Roadrunner

- The first “commodity” cluster to become #1
  - The first #1 machine to use Infiniband
  - The first #1 Linux machine
  - C.f. x86 ASCI Red
- The first “heterogeneous” machine to become #1, Cell and Opteron
- The first many “commodity vector” cores CPU to become #1
  - 8 SPE vector cores total/chip

GPGPUs as Commodity Vector Engines---True Rebirth of Vectors

- E.g., Nvidia 8800GTX/8800GTS/280GTX
  - High Peak Performance 1TFlops
    - Good for tightly coupled code e.g. Nbody
  - High Memory bandwidth (>100GB/s)
    - Good for sparse codes e.g. CFD
    - High 3DFFT performance (>100 GFlops) due primarily to memory bandwidth
  - **Looks very much like classic vector machines**
    - Many many registers, small cache, abundunt multithread ~= long vectors
  - Restrictions: Limited non-stream memory access, PCI-express overhead, etc.

How do we exploit them given vector computing experiences?

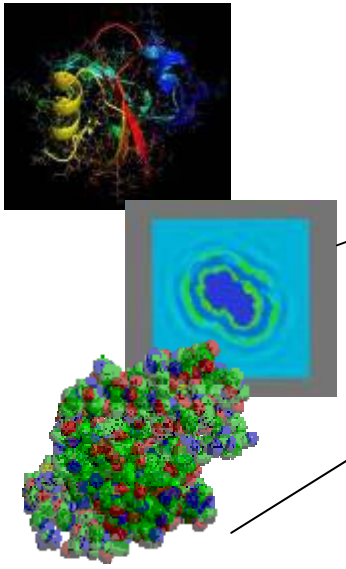
Microsoft Science All Hands Meeting, Redmond, 7 March 2008

3-D Protein Docking Challenge Powered by Accelerator-based HPC-GPGPU Technologies

Overview of HPC-GPGPU Project

(work w/MS Research)

Advanced Bioinformatics/ Proteomics



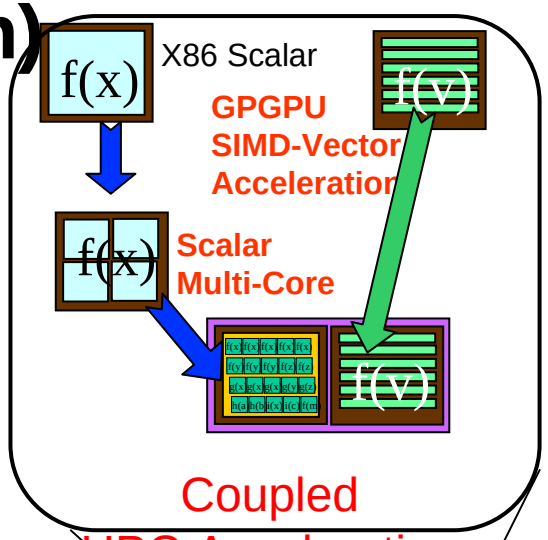
Need x1000 acceleration over standard PCs

Research Focus

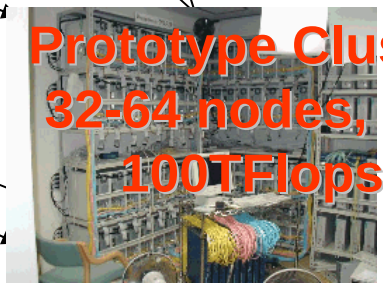
Bioinformatics Acceleration
e.g., 3-D All-to-All Protein Docking

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
- Model-based GPGPU-CPU Load Balancing



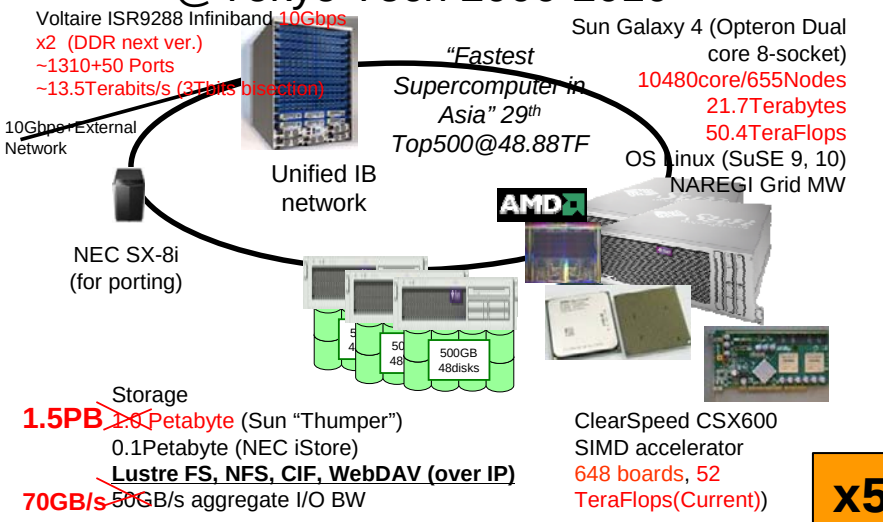
Coupled HPC Acceleration



Towards Next Gen Petascale Personal Clusters and Desksides

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

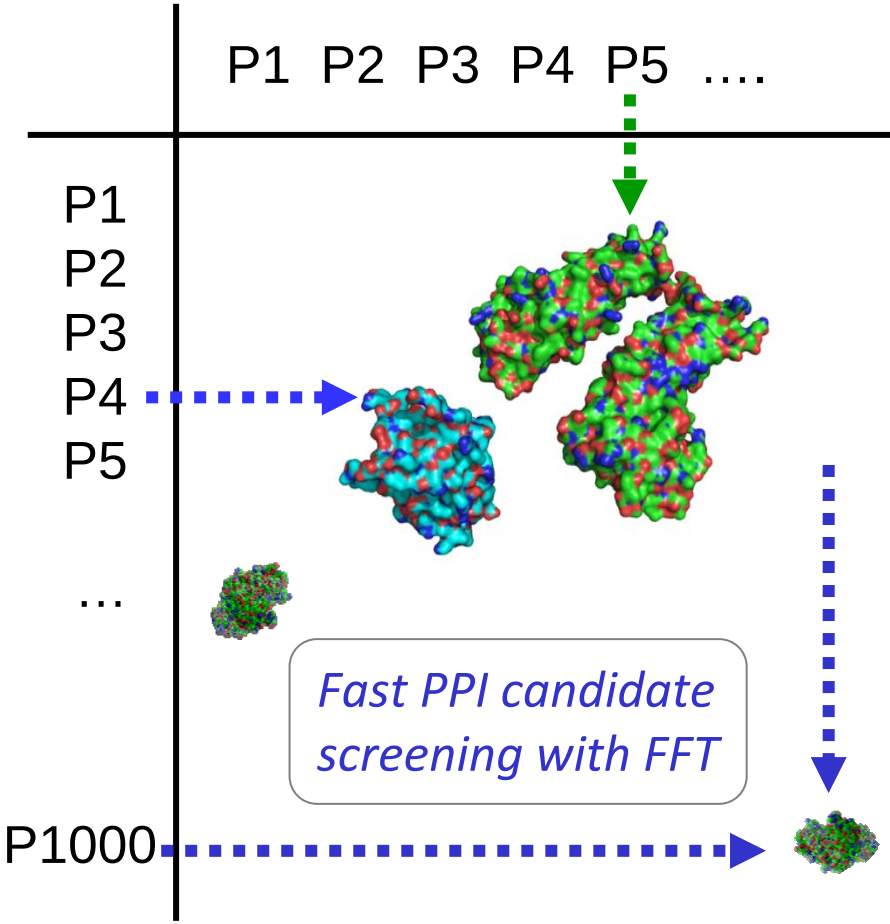
WinCCS HPC-GPGPU Cluster

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 for 3D Proteomics
  - $100,000-$200,000 MSRP

All-to-all 3-D Protein Docking Challenge



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).

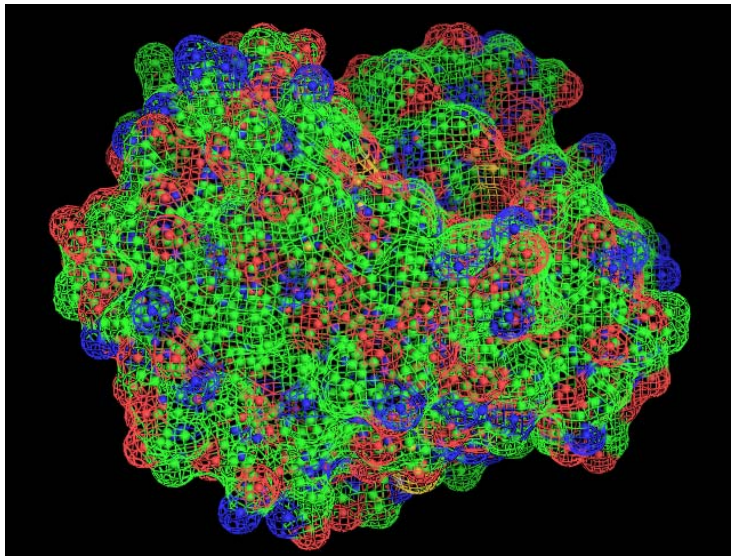
cf.

~ 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.

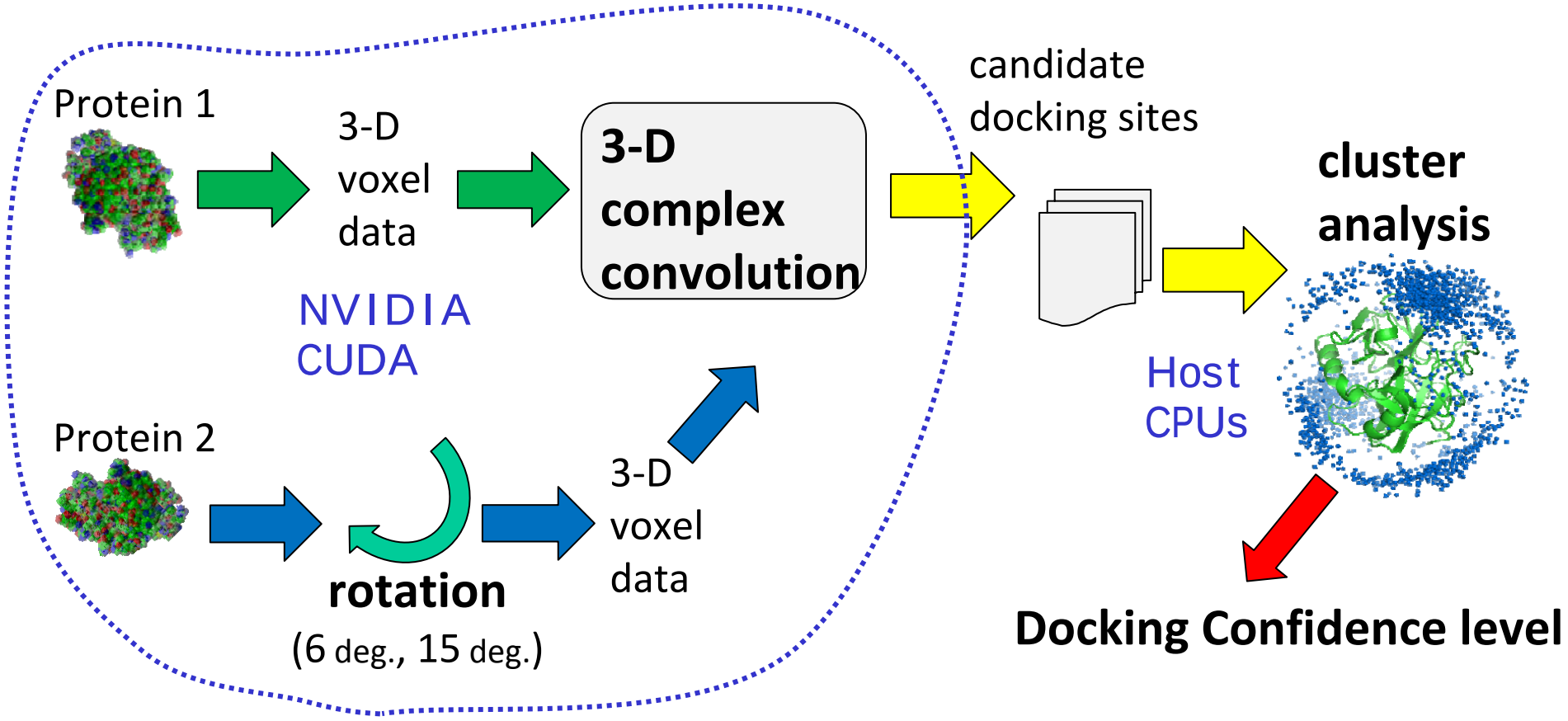
No biologist think proteins are rigid.

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 for 3-D AA docking



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

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

x

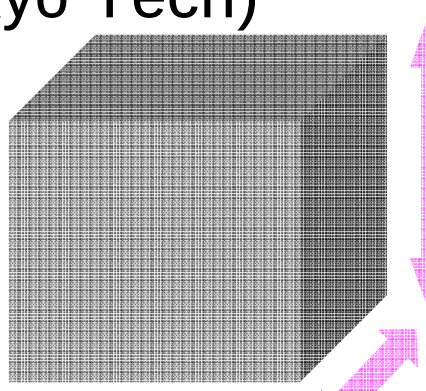
Possible rotations $R = 54{,}000$ *(6 deg. pitch)*

200 Exa Ops for 1000 x 1000

High-Performance 3D FFT in CUDA [SC08]

(Akira Nukada @ Tokyo Tech)

Easty to do 1-D FFT on GPUs but hard to do 3-D

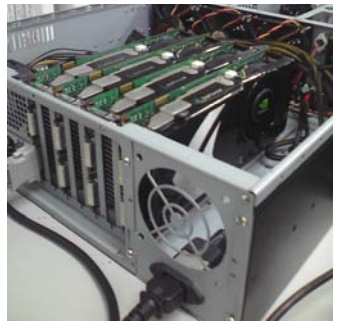


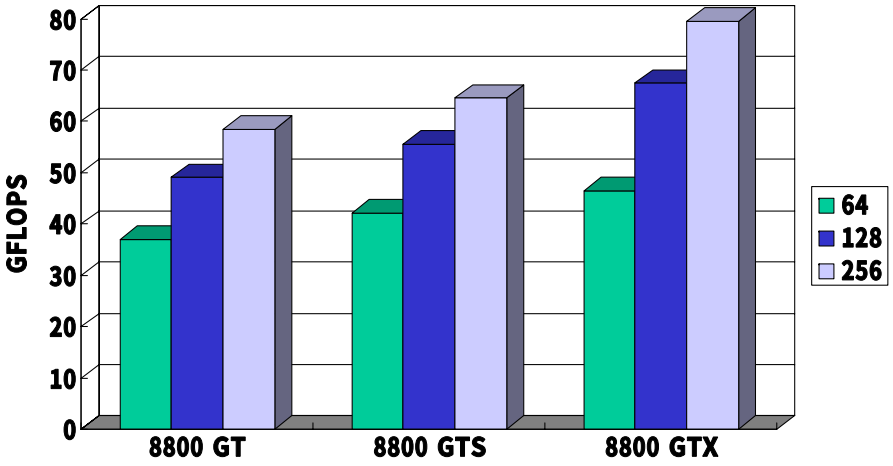
A Novel algorithm using shared memory in CUDA to Overcome this restrictions


GeForce 8800 GTS

Optimizing Memory Bandwidth


GeForce 8800M GTX





x3 CUFFD, x2 FFTW

	Power(W)	GFLOPS	GF/W
Blue Gene/L	20	1.8	0.09
GTS×4	560	220	0.39
GX2×8	960	439	0.46
M-GTX×2	160	80	0.50

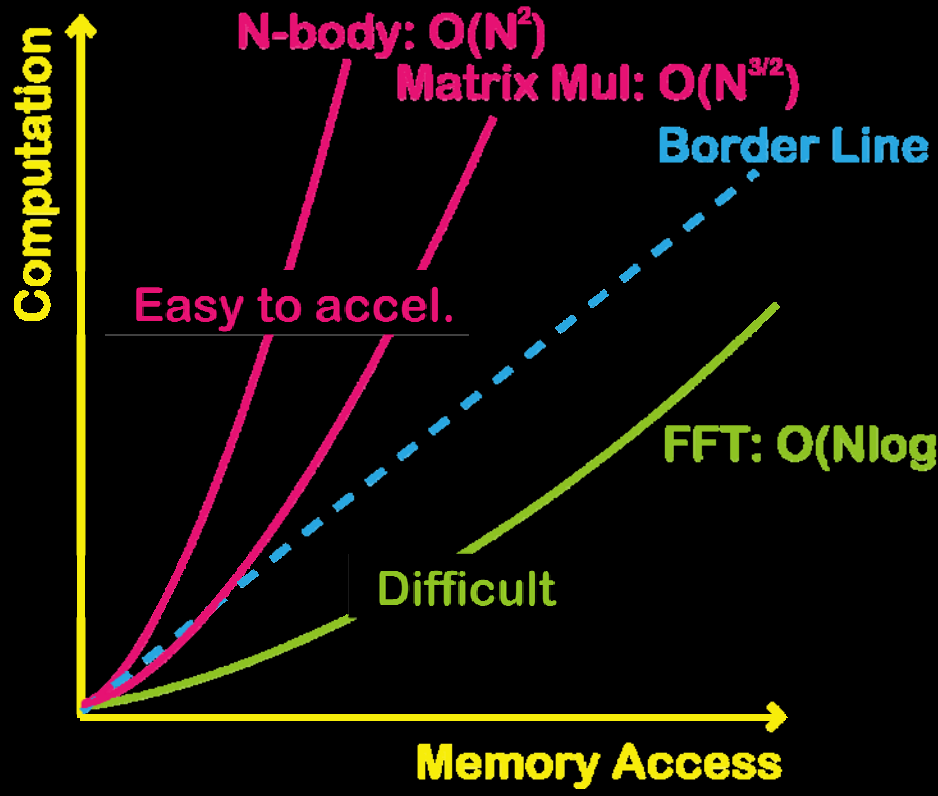
x6 BG/L, x20 TSUBAME Performance/Watt

High Performance 3-D FFT on NVIDIA CUDA GPUs [SC08 paper preview]

Akira Nukada

Tokyo Institute of Technology, GSIC.

Acceleration of FFT

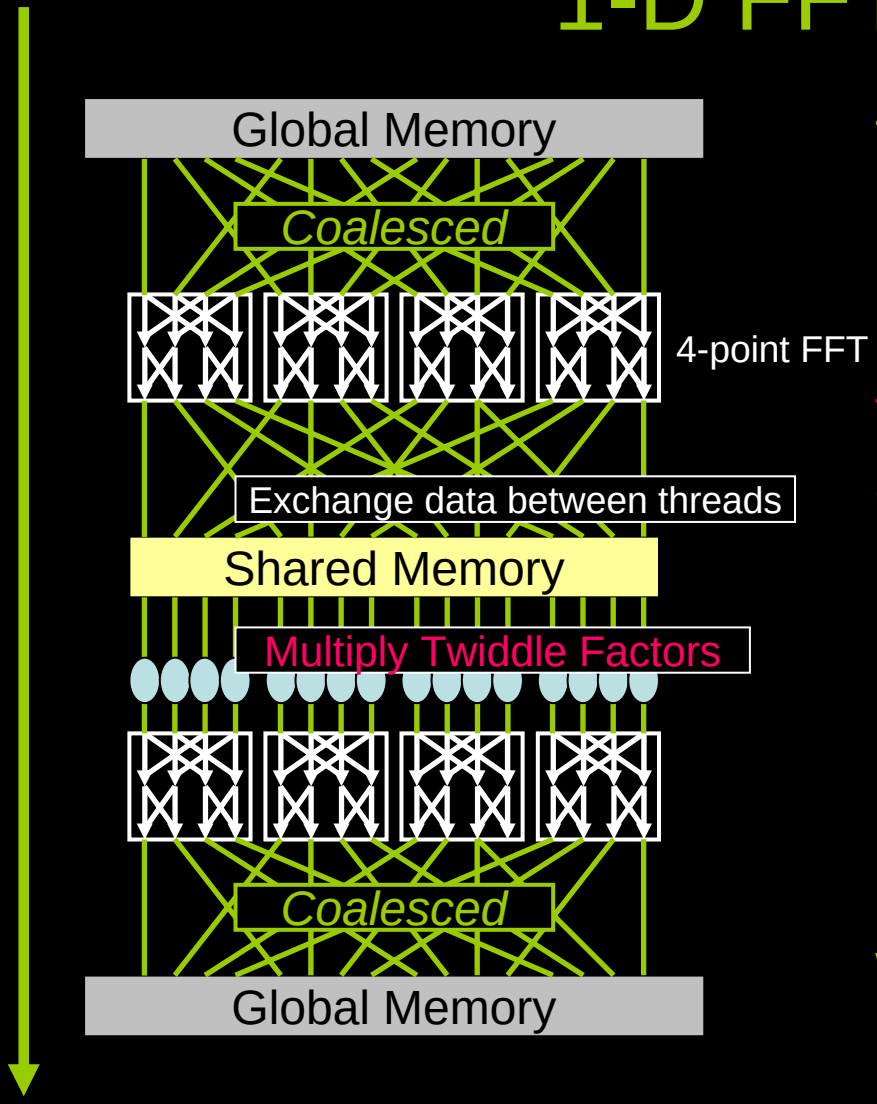


Fast Fourier Transform is an O(NlogN) algorithm with O(N) memory access.

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.

1-D FFT on GPU



This is a simple example to compute 16-point FFT with 4 threads.
(Actually 256-point@64 threads, etc.)

Twiddle Factors are triangular functions, and thread-dependent value.
In CUDA, they should come from one of
(1) registers.
(2) table on constant (cache) memory.
(3) table on texture (cache) memory.
(4) table on shared memory.
(5) calculate using SFU each time.
We selected ‘texture plan’ to reduce the number of instructions and registers.

Available methods for twiddle factors

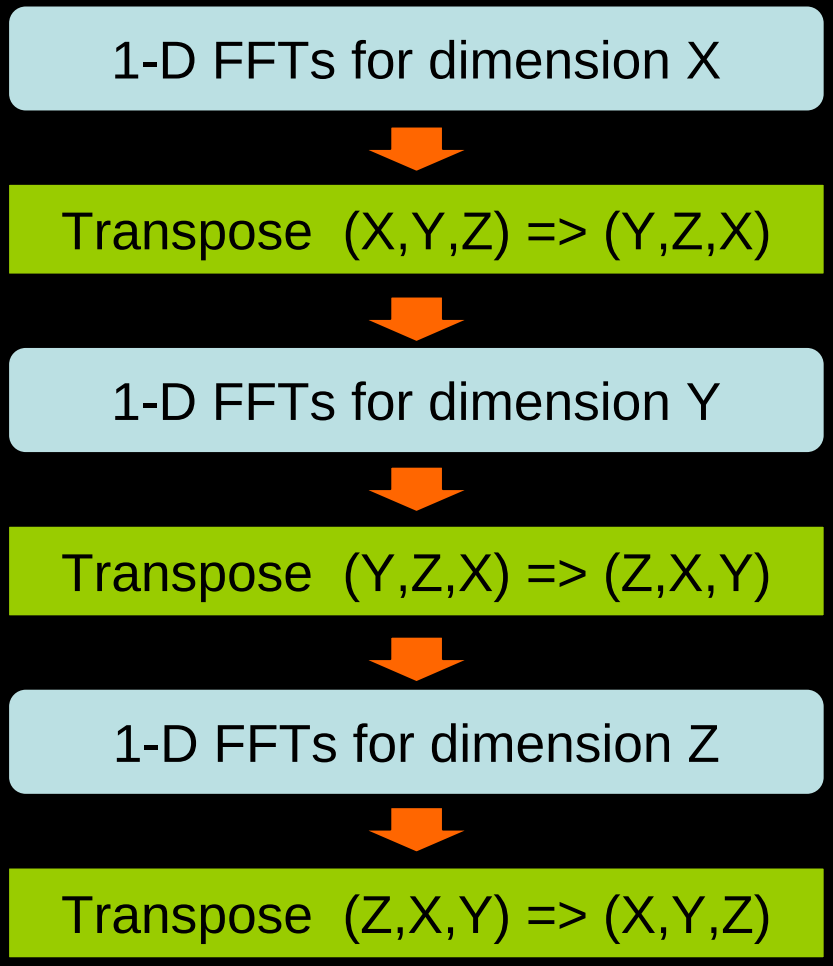
	Speed	Requirements
register	fastest	More registers.
constant	fast	More processor cycles. (Only 1-way)
texture	fast	
shared	fastest	More shared memory.
calculate	slow	Much more processor cycles.

To improve achieved memory bandwidth of 1-D FFT kernels,

we have to execute many active thread blocks on each Streaming Multiprocessor.

For this reason, the amount of resource usage of the kernel should be saved.

Conventional Algorithm with Transposes



Three transposes to optimize the performance of 1-D FFTs.

Their memory accesses can be *coalesced*, but achieved bandwidth is much lower.

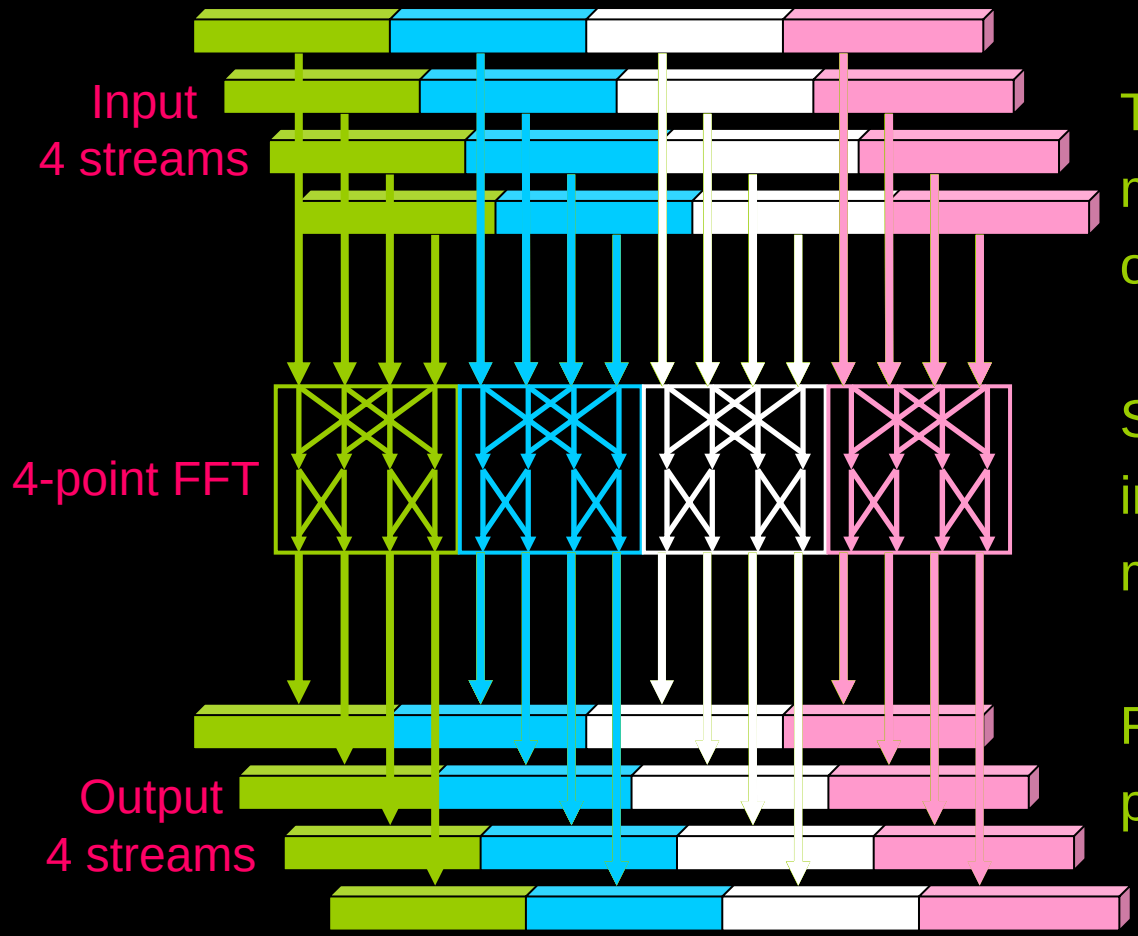
With optimized 1-D FFTs, transposes occupy about 70%.

Bandwidth Intensive Approach

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.
Used for vector machines which provide high memory bandwidth.

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.

The idea / decision comes from

- Experiences of implementing FFTs on
  cache-based systems: Sun E10000, SGI Origin 2000, HITACHI SR2201, IBM Regatta, Apple PowerMac G5, SGI Altix 3700, Intel x86/Intel 64, AMD64, BlueGene/L, CRAY XT3, SECI PlayStation3,
  and vector systems: HITACHI SR2201, NEC SX-6i, The Earth Simulator
- Measurement of memory bandwidth with various
  - pitch and block size of stride access.
  - number of streams

Evaluation System

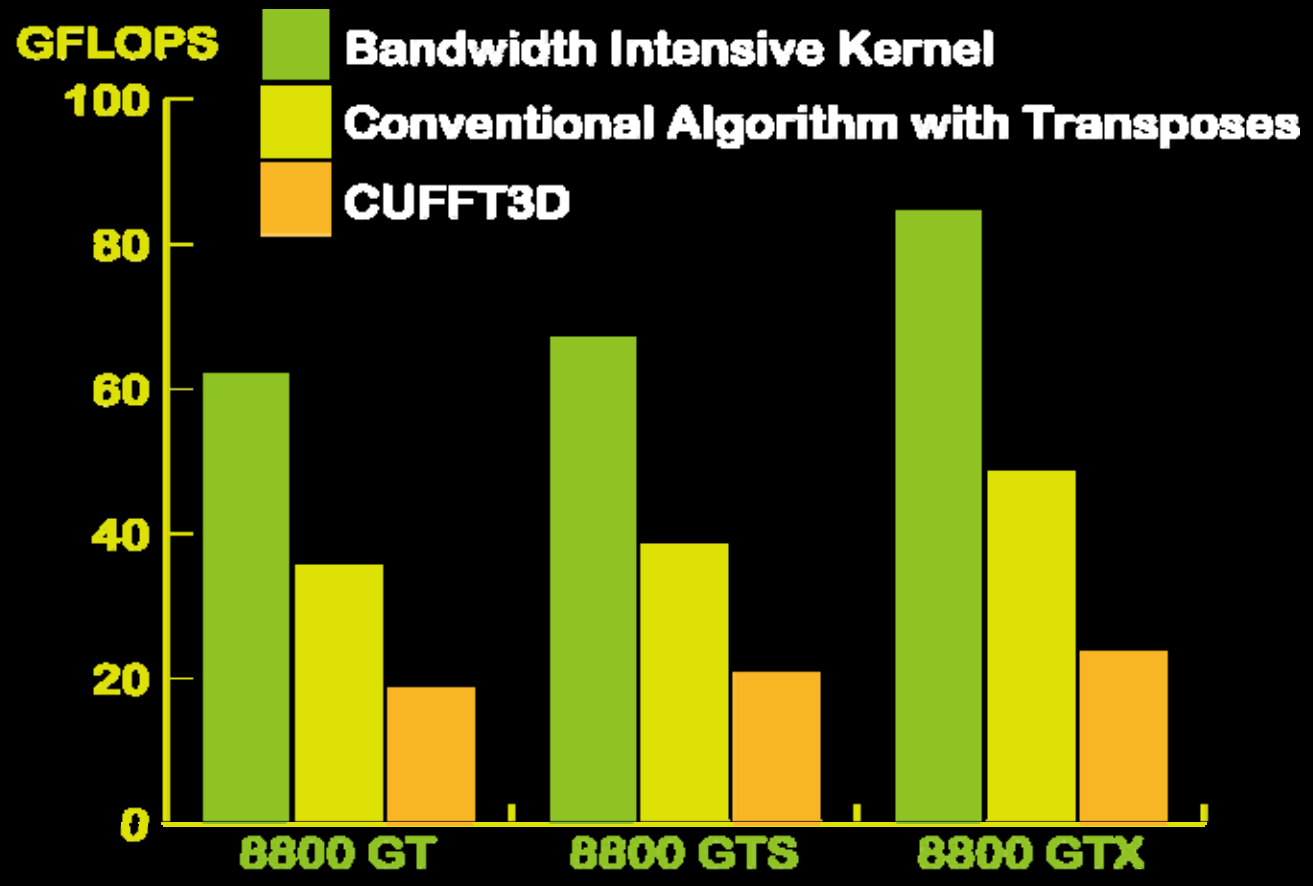
Hardware

- AMD Phenom 9500 (Quad-Core, 2.2GHz)
- AMD 790FX Chipset (PCI-Express Gen2 support)
- DDR2-6400 SDRAM 1GB *4
- NVIDIA GPU Card

Software

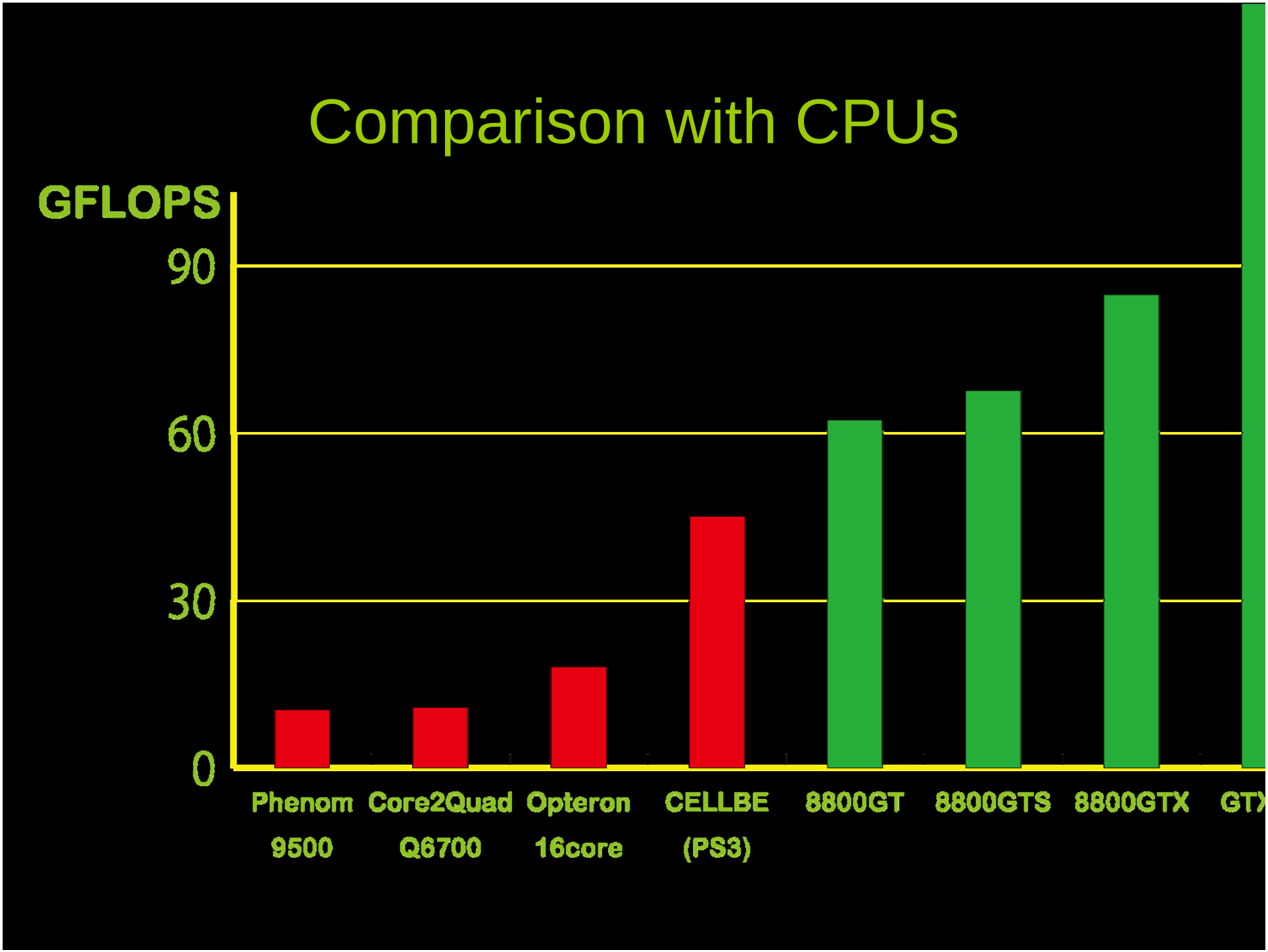
- Fedora Core 8 Linux, 64-bit,
- NVIDIA Driver version 169.09
- CUDA Toolkit 2.0beta2, gcc 4.1.2.

Performance on GeForce 8 series

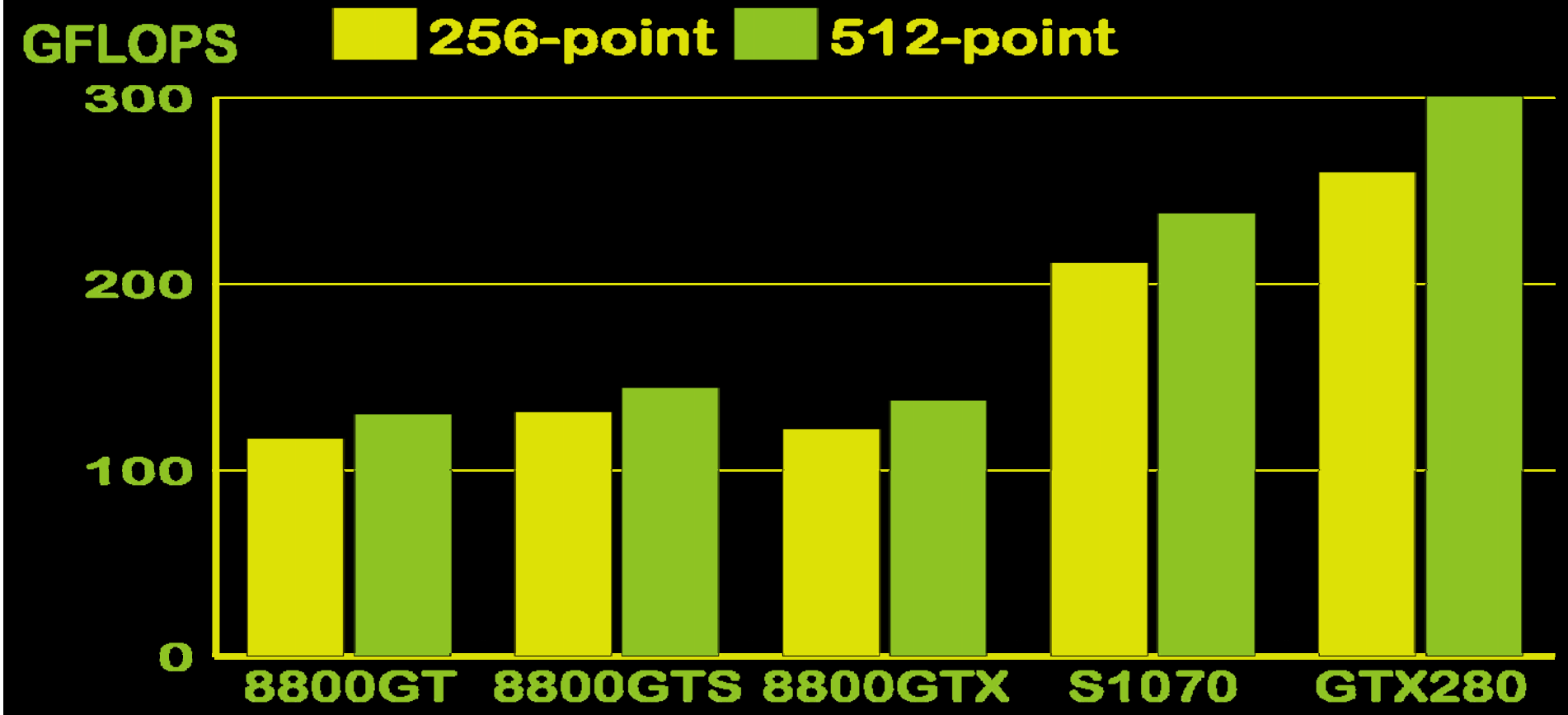


3-D FFT of size 256^3 in single precision.

Comparison with CPUs

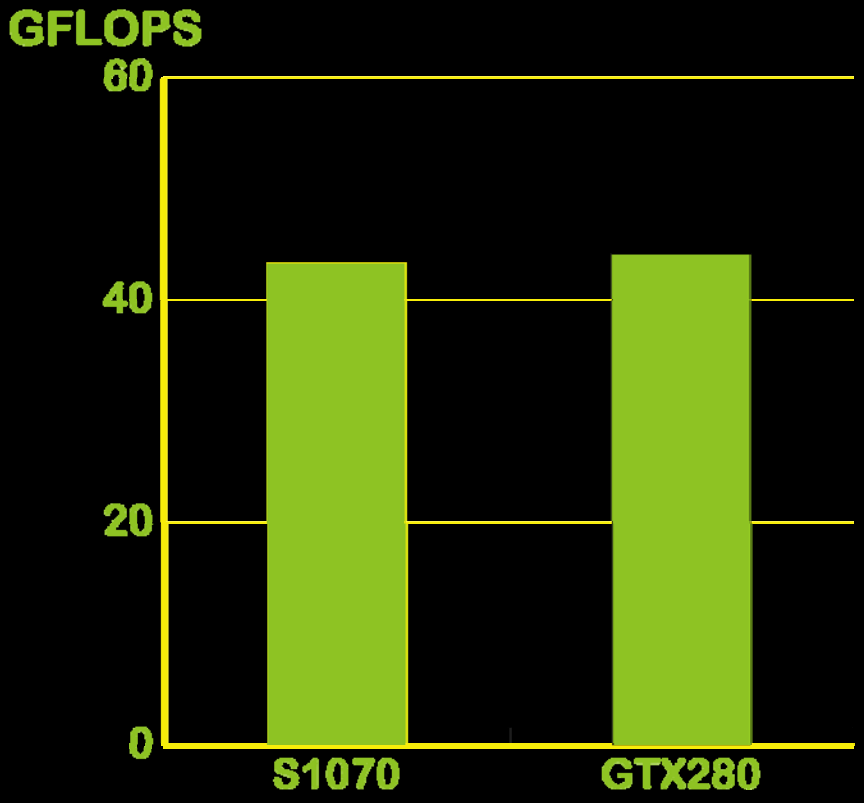


Performance of 1-D FFT



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

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).

Power Efficiency

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.

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

Summary

Our high performance 3-D FFT on CUDA achieved

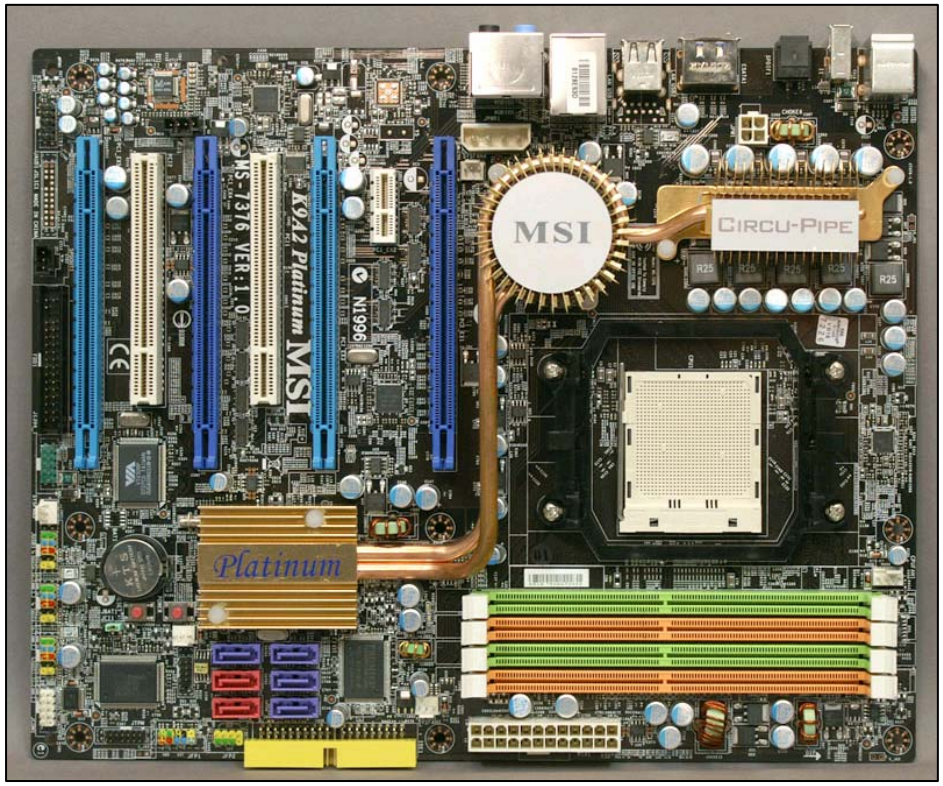
- three times higher performance than CUFFT
- four times higher power efficiency (GFLOPS/W)

Guidelines for developing CUDA applications

- Try all memory access patterns.
- Try algorithms for vector processors, or mix it.
- General guidelines by NVIDIA.

Main board Selection

- Enabling 4 GPGPUs
  - Each High-end GPGPU occupies 2 PCI-e slots for cooling
  - So we need four 16x PCI-e with 2 slot spacing



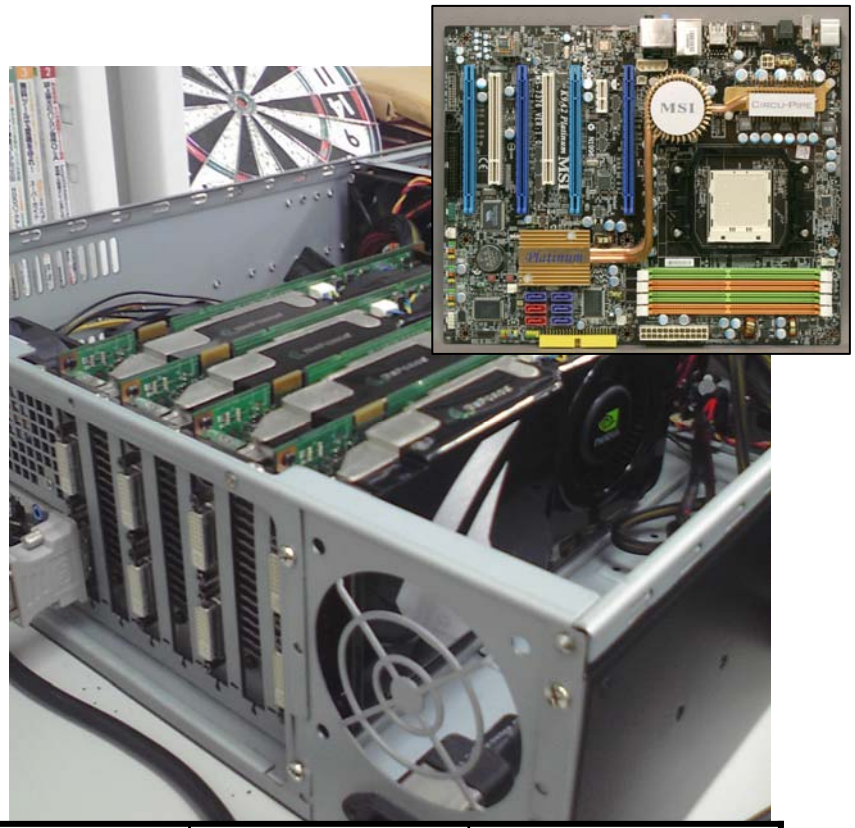
MSI K9A2 Platinum

- "MSI K9A2 Platinum" is the only such ATX M/B
  - AMD 790FX & AMD Phenom

3U Rack-mount Prototype Node

Configuration

- AMD Phenom 9500 (2.2Ghz QC)
- MSI K9A2 Platinum (AMD 790FX Chipset)
- DDR2-800 SDRAM 2GB*4
- Onboard Gigabit Ethernet
- NVIDIA GeForce 8800 GTS*4 (G92, 512MB)
- Efficient Front-to-back cooling
- Power Consumption: 570W.

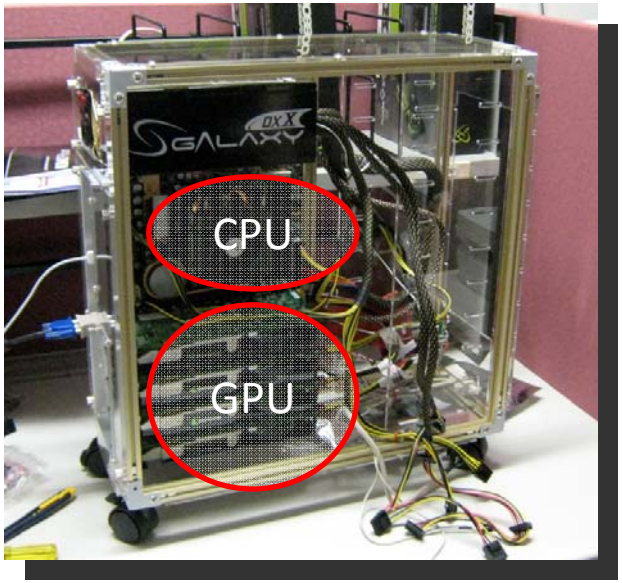


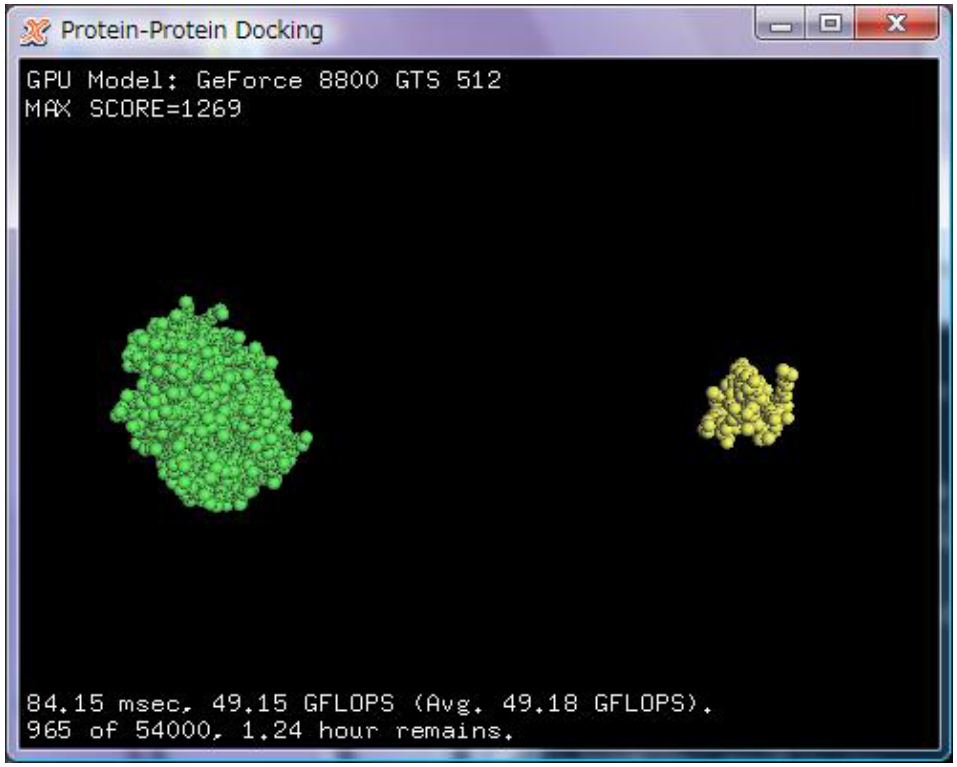
	GFLOPS	Memory (GB/s)	Proc.	Power (W)	Interface
8800 GTX	345	86.4	90nm	>150	PCI-E 1.0
8800 GT	336	57.6	65nm	90	PCI-E 2.0
8800 GTS	416	62.0	65nm	100	PCI-E 2.0

Heavily Acclerated Cluster System Configuration

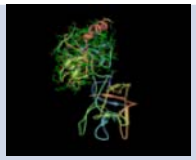
- 32 compute nodes
- 128 8800GTS GPGPUs
- one head node.
- Gigabit Ethernet network
- Three 40U rack cabinets.
- Windows Compute Cluster Server 2003 SP1, planned 2008 migration
- Visual Studio 2005 SP1
- nVidia CUDA 2.x

Prototype Performance





Measured docking performance on GPGPU:

Structure	Size of FFT space	Rotation pitch	Elapsed time per rotation	Rotation count	Estimated time per structure	Docking Performance
 1AVX	256^3	6 deg.	85ms	60*15*60 =54,000	1.28 hour	48.4 GFlops /GPU

Porcine Pancreatic Trypsin and inhibitor (PDB:1AVX). Other 11 proteins showed 48.0-48.5 GFlops.

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

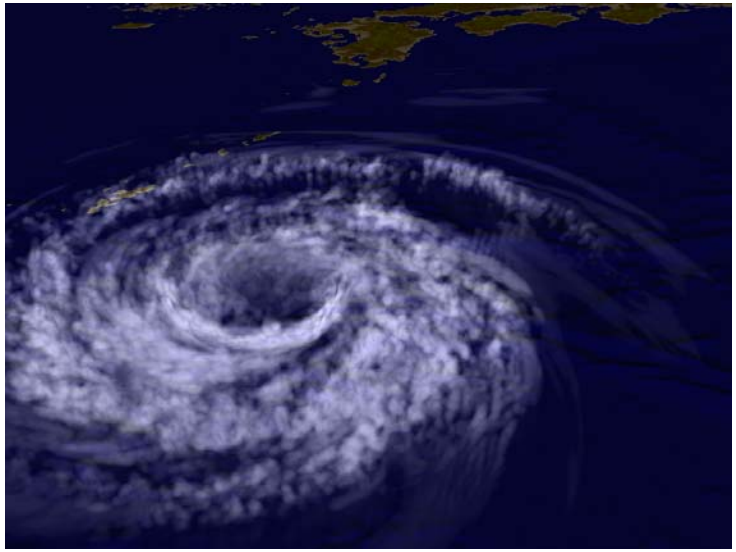
! Only CPUs for TSUBAME. DP=double precision.

	# 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

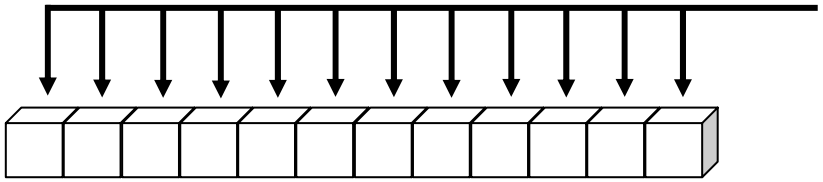
Can compute 1000x1000 in 1 month (15 deg.) or 1 year (6 deg.)

GPU Computing for CFD

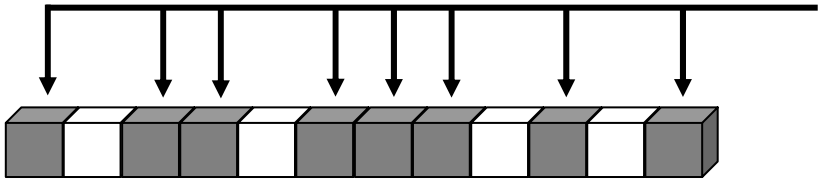
- One of major applications of HPC.
- An inevitable process of new product design (Car aerodynamics, . .)
- It requires a large amount of memory and sparse memory access.
- Most of CFD application use calculation grid.
- Many numerical methods, FDM, FVM, FEM, . . .



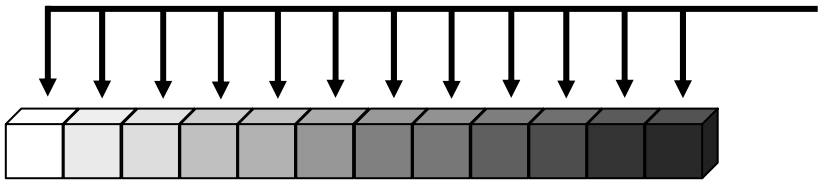
Types of Memory Access



Continuous Access
FDM (Finite Difference)



Random (Indirect) Access
FEM (Finite Element)
A[i] = A[IP[i]] + A[IP[i-1]];



Data Dependency
A[i] = A[i-1] + A[i-2]*C;

Classification of CFD

Compressible fluid analysis

Supersonic flow, Acoustic wave, Explosion, Shock wave, . . .

High-accurate numerical methods:

T. Aoki, Comp. Phys. Comm., Vol.102, No.1-3, 132-146 (1997)

Y. Imai, T. Aoki and K. Takizawa, J. Comp. Phys., Vol. 227, Issue 4, 2263-2285 (2008)

K. Kato, T. Aoki, M. Yoshida, et. al., Int. J. Numerical Methods in Fluids, Vol.51, 1335-1353 (2006)

Y. Imai, T. Aoki, J. Comp. Phys., Vol.217, 453-472 (2006)

Y. Imai, T. Aoki, J. Comp. Phys., Vol.215, 81-97 (2006)

Incompressible fluid analysis

- Most of flow phenomena in our daily life,
- Turbulent flow, Multi-phase flow, Reacting flow, . .
- Semi-implicit Time Integration → Poisson Solver

Incompressible CFD Application

Incompressible Navior-Stokes Equation

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

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\frac{1}{\rho} \nabla p + \nu \Delta \boldsymbol{u}$$

Poisson equation

$$\Delta p^{n+1} = \frac{\nabla \cdot \boldsymbol{u}^{n+1}}{\Delta t}$$

- Advection Term： High-accurate FDM (Suitable for GPU)
- Diffusion Term： 2nd order Center FDM (easy)
- Velocity Divergence： Staggered FDM (easy)
- Poisson equation： Red & Black MG (hard)
- Pressure Gradient： Staggered FDM (easy)

RIKEN Benchmark Problem

Poisson Equation:
(Generalized coordinate)

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

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

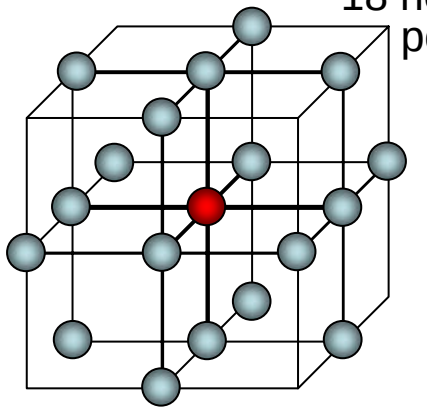
Discretized Form:

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

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

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

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



18 neighbor
point access

Detail of RIKEN Benchmark Problem

Read only 12 arrays : a, b, c, bnd, . . .

Read-write 2 arrays : p, wrk2

```
#define MIMAX             65
#define MJMAX             65
#define MKMAX            129

static float  p[MIMAX][MJMAX][MKMAX];
static float  a[4][MIMAX][MJMAX][MKMAX],
              b[3][MIMAX][MJMAX][MKMAX],
              c[3][MIMAX][MJMAX][MKMAX];
static float  bnd[MIMAX][MJMAX][MKMAX];
static float  wrk1[MIMAX][MJMAX][MKMAX],
              wrk2[MIMAX][MJMAX][MKMAX];
```

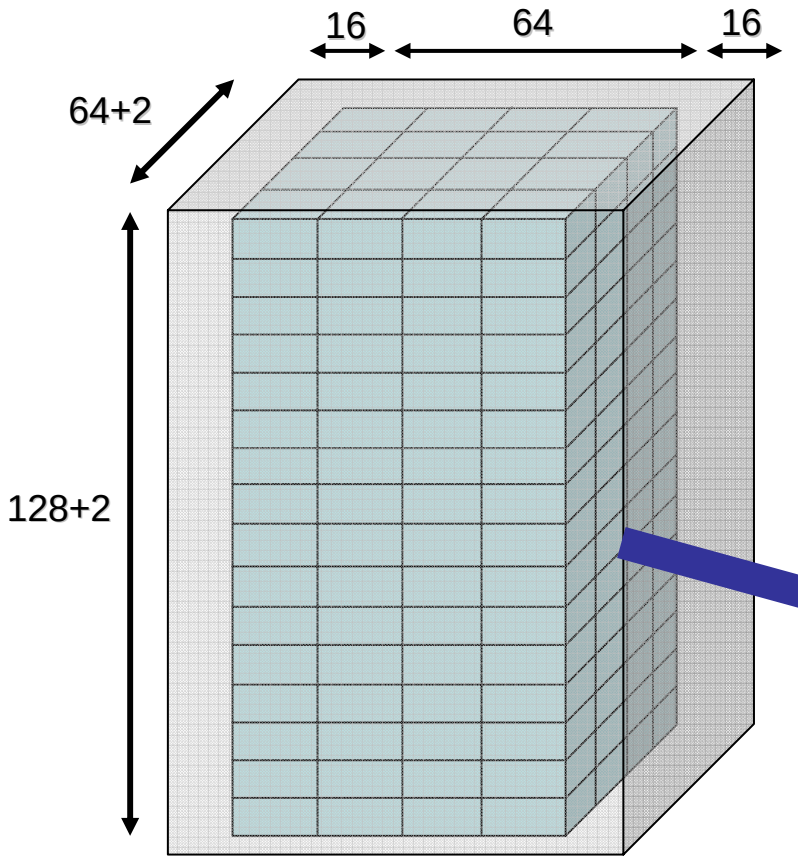
```
for(i=1 ; i<imax-1 ; i++)
      for(j=1 ; j<jmax-1 ; j++)
        for(k=1 ; k<kmax-1 ; k++){
          s0 = a[0][i][j][k] * p[i+1][j  ][k  ]
             + a[1][i][j][k] * p[i  ][j+1][k  ]
             + a[2][i][j][k] * p[i  ][j  ][k+1]
             + b[0][i][j][k] * ( p[i+1][j+1][k  ] - p[i+1][j-1][k  ]
                               - p[i-1][j+1][k  ] + p[i-1][j-1][k  ] )
             + b[1][i][j][k] * ( p[i  ][j+1][k+1] - p[i  ][j-1][k+1]
                               - p[i  ][j+1][k-1] + p[i  ][j-1][k-1] )
             + b[2][i][j][k] * ( p[i+1][j  ][k+1] - p[i-1][j  ][k+1]
                               - p[i+1][j  ][k-1] + p[i-1][j  ][k-1] )
             + c[0][i][j][k] * p[i-1][j  ][k  ]
             + c[1][i][j][k] * p[i  ][j-1][k  ]
             + c[2][i][j][k] * p[i  ][j  ][k-1]
             + wrk1[i][j][k];

          ss = ( s0 * a[3][i][j][k] - p[i][j][k] ) * bnd[i][j][k];
          wrk2[i][j][k] = p[i][j][k] + omega * ss;
        }

} /* end n loop */
```

Shared-memory Use

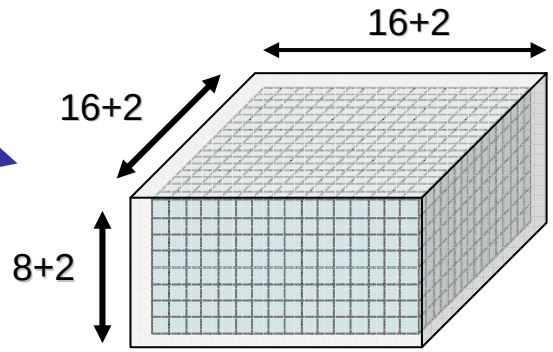
1 block = 16x16x8 : A part of the Computation Domain



for 1 block, 256 thread

Total 256 block = 65536 thread

Maximum Use of
the shared memory = 16kB



Boundary : Non-coal eased data access

Memory Bounded Problem

A[129][65][65]：2.18 MB✕14 variables

12：read only
1：read-write
1：write

```
#define MIMAX             65
#define MJMAX             65
#define MKMAX            129

static float  p[MIMAX][MJMAX][MKMAX];
static float  a[4][MIMAX][MJMAX][MKMAX],
              b[3][MIMAX][MJMAX][MKMAX],
              c[3][MIMAX][MJMAX][MKMAX];
static float  bnd[MIMAX][MJMAX][MKMAX];
static float  wrk1[MIMAX][MJMAX][MKMAX],
              wrk2[MIMAX][MJMAX][MKMAX];
```

per 1 grid = 34 floating point calculations

per 1 word Data transfer = 34/14 = 2.4

If GPU data transfer rate is

60 GB/sec（15 GWord/sec）,

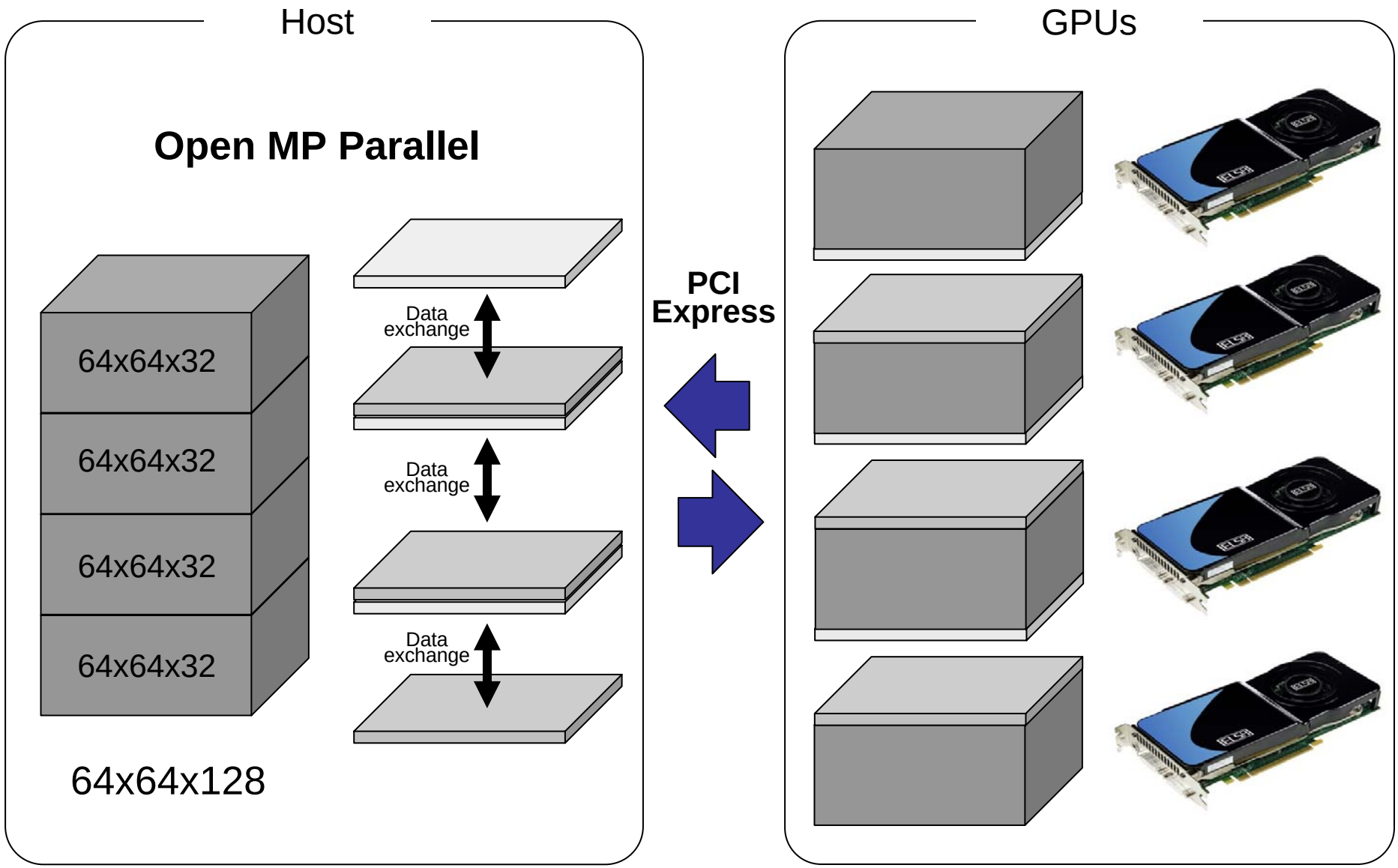
Even if GPU is very fast

15✕2.4 = 36.4 GFLOPS

Without shared memory : 34/14 → 34/(14+18) = 1.06

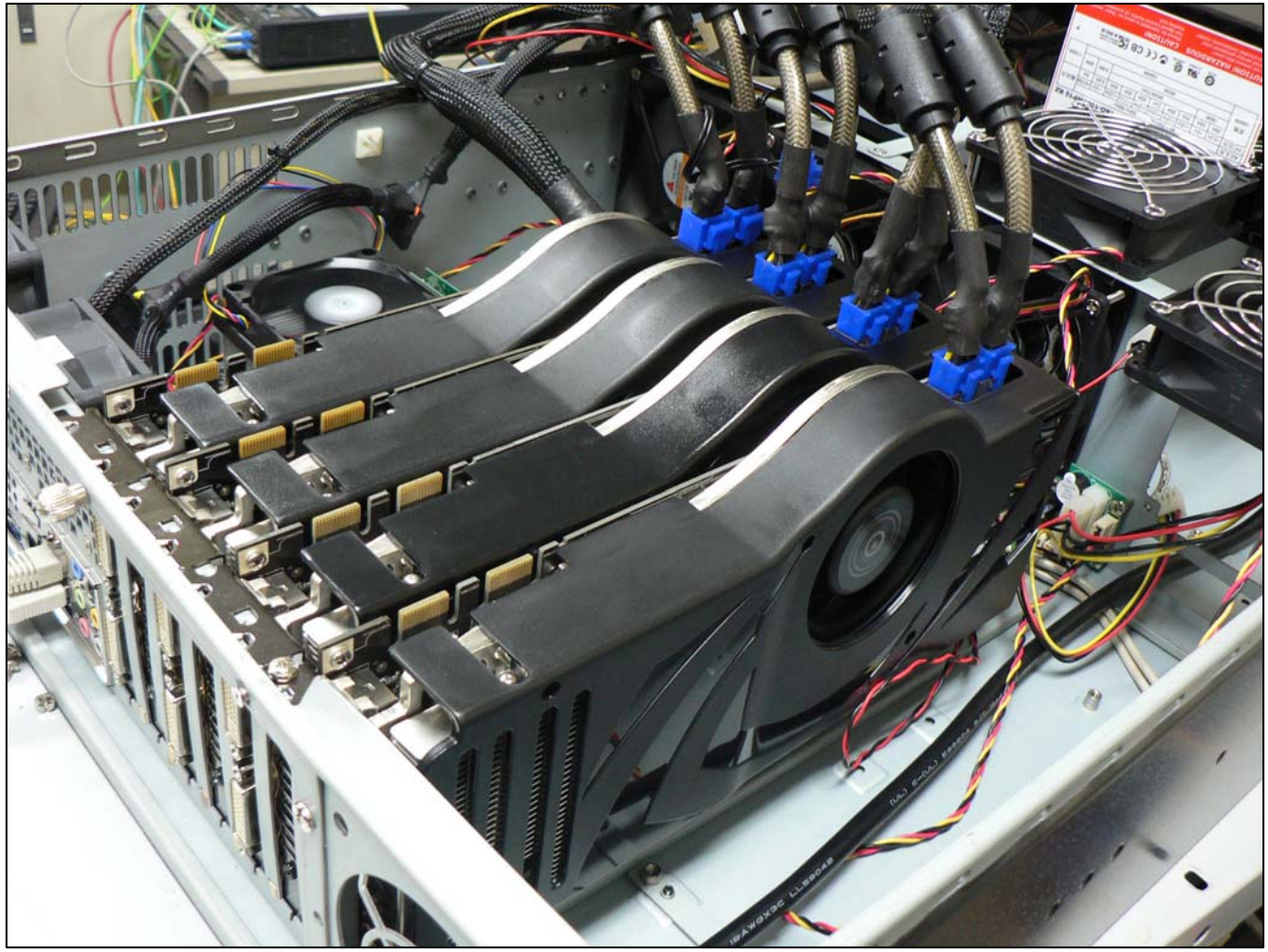
15✕2.4 = 15.9 GFLOPS

Parallelization using 4 GPU



Machine (Phenom Desktop PC + 4 GPU)

GeForce 8800 Ultra x 4



Specifications of GPU and Motherboard

		GeForce 8800Ultra(G80) (MSI)	GeForce 8800Ultra(G80) (ELSA)
GPU	Peak Performance [GFlops]*	414.2	384
	# of SP	128	128
	SP Clock(CoreCock)[MHz]	1618(660)	1500(612)
Video Memory	Transfer Rate[GB/s]	110.4	103.68
	Memory Bus width[bit]	384	384
	Data Rate[GHz]	2.3(GDDR3)	2.16(GDDR3)
	Capacity [MB]	768	768

* 2 instruction issue

MSI K9A2 Platinum

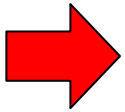
AMD 790FX + AMD SB600

4 PCI-Express x16-type slots

- 2 slot Support up to PCI-Express 2.0 x 16
- 2 slot Support up to PCI-Express 2.0 x 8

When using 4 GPU card, it works as PCI-Express x 8 for each GPU

RESULT

0.976 GFLOPS (8.431sec)  51.91 GFLOPS (0.158sec)

×53.1

- Before

```
mimax = 65 mjmax = 65 mkmax = 129
imax = 64 jmax = 64 kmax =128
 Start rehearsal measurement process.
 Measure the performance in 3 times.

 MFLOPS: 941.082902 time(s): 0.052496 3.288628e-03

 Now, start the actual measurement process.
 The loop will be excuted in 500 times
 This will take about one minute.
 Wait for a while

 Loop executed for 500 times
 Gosa : 9.673350e-04
 MFLOPS measured : 976.566479       cpu : 8.431426
 Score based on Pentium III 600MHz : 11.909347
```

- After

```
[INFO] Number of host available CPU : 4
[INFO] Number of CUDA devices       : 4

 4-GPU OpenMP execution . . . . .

mimax = 65 mjmax = 65 mkmax = 129
imax = 64 jmax = 64 kmax =128
 Start rehearsal measurement process.
 Measure the performance in 3 times.

 MFLOPS: 34717.560084 time(s): 0.001423 3.295089e-03

 Now, start the actual measurement process.
 The loop will be excuted in 500 times
 This will take about one minute.
 Wait for a while

 Loop executed for 500 times
 Gosa : 9.672065e-04
 MFLOPS measured : 51909.594689     cpu : 0.158619
 Score based on Pentium III 600MHz : 633.043838
```

Parallel Performance

Sモデル [65x65x129]

1 GPU (no data transfer)	**30.6 GFLOPS**	**(0.269sec)**
2 GPU (16kB transfer)	**42.5 GFLOPS**	**(0.193sec)**
4 GPU (32kB transfer)	**51.9 GFLOPS**	**(0.158sec)**

..... Reference .....

Mモデル [129x129x257]

1 GPU (no data transfer)	29.4 GFLOPS	(2.328sec)
2 GPU (66kB transfer)	53.7 GFLOPS	(1.275sec)
4 GPU (131kB transfer)	83.6 GFLOPS	(0.819sec)

Lモデル [257x257x512]

1 GPU (no data transfer)	……	
2 GPU (262kB transfer)	……	
4 GPU (524kB transfer)	**93.6 GFLOPS**	(5.974sec)

Poisson Equation solved by
MG(Multi Grid), Red & Black method

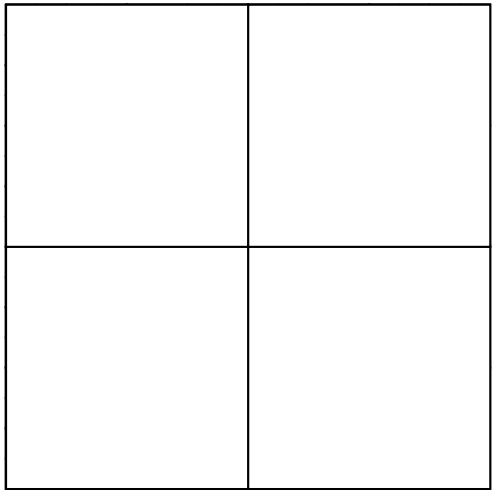
■ **Algorithm Acceleration**

Point Jacobi $\xrightarrow{\times 4 \sim 5}$ SOR $\xrightarrow{\times 100}$ MG-SOR

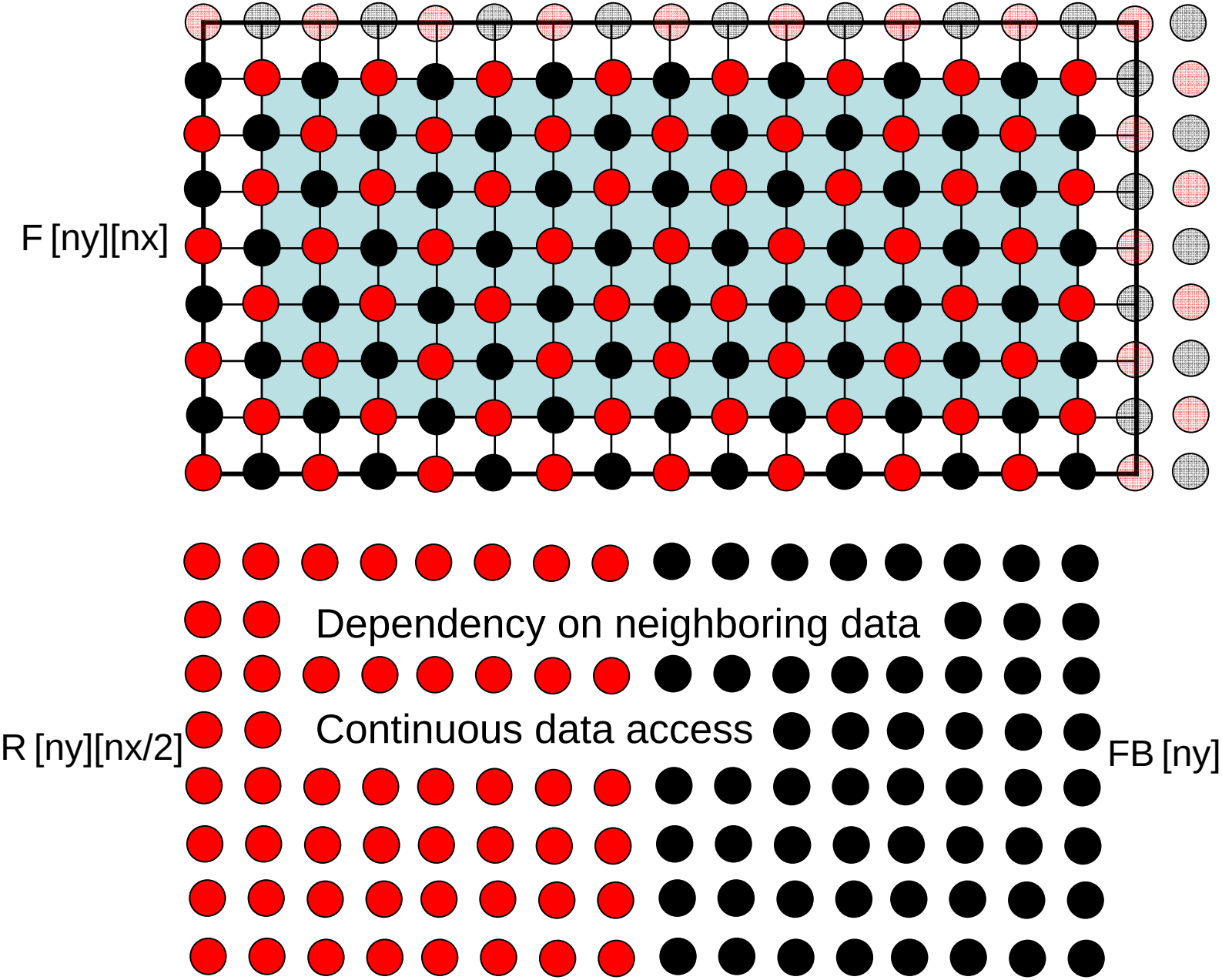
■ **Hardware Acceleration :**

GPU (CUDA) ×50?

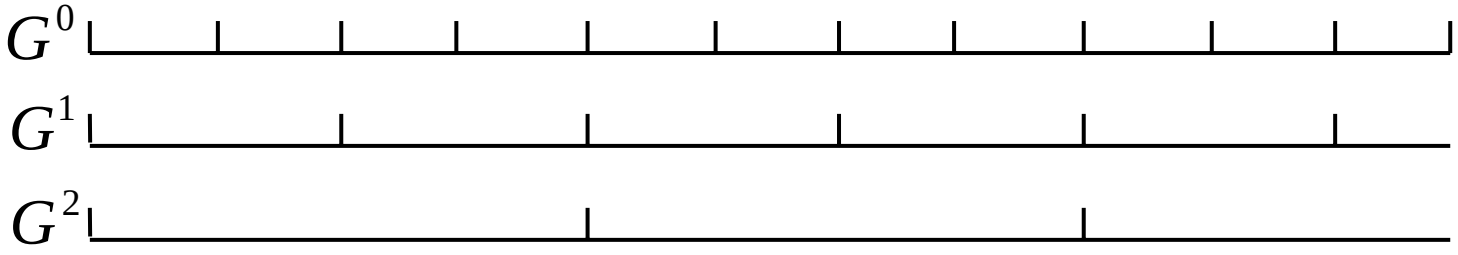
$$\frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{\Delta x^2} + \frac{f_{i,j+1} - 2f_{i,j} + f_{i,j-1}}{\Delta y^2} = \rho_{i,j}$$



Red & Black method



Multi-Grid Method



Δx^k Grid spacing of the k-th Grid G^k $\Delta x^k = 2\Delta x^{k-1}$

Discretized Poisson Equation on G^k

$$L^k F^k = S^k$$

L^k : operator F^k: exact solution

S^k: source term

$$f_1^k = SOR_{RED\&BLACK}\left(L^k, S^k, f_0^k, n\right)$$

n-times iteration result, starting from f_0^k

Residual $R^k = S^k - L^k f_1^k$

Correction Equation

Correction $v^k = F^k - f_1^k$

Residual $R^k = S^k - L^k f_1^k$

$$= L^k F^k - L^k f_1^k = L^k \left(F^k - f_1^k \right)$$

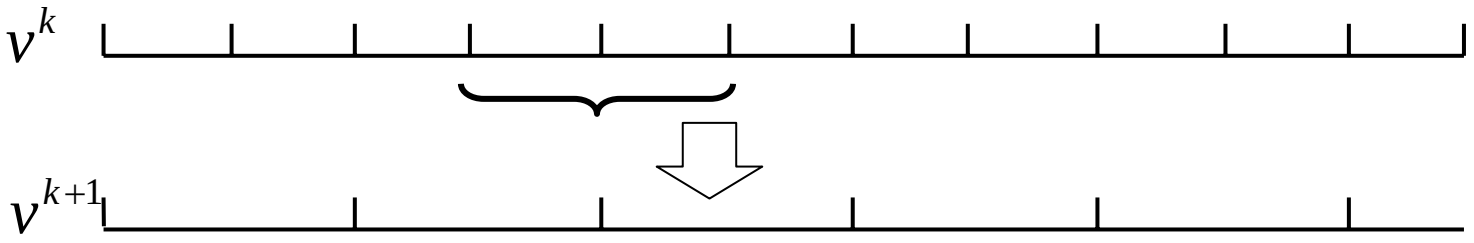
$$\therefore \quad L^k v^k = R^k$$

The k-th grid correction equation has the same form as Poisson equation.

$$L^{k+1} v^{k+1} = R^{k+1} \qquad \left(\Delta x^{k+1} = 2 \Delta x^k \right)$$

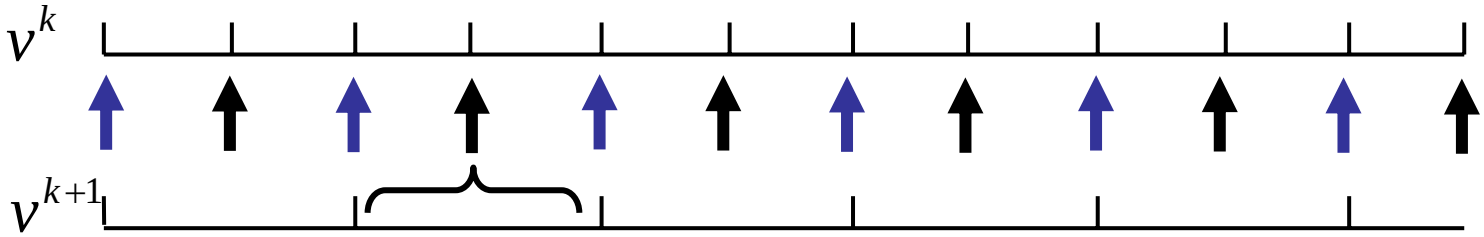
The $k+1$-th grid correction equation

Restriction and Prolongation



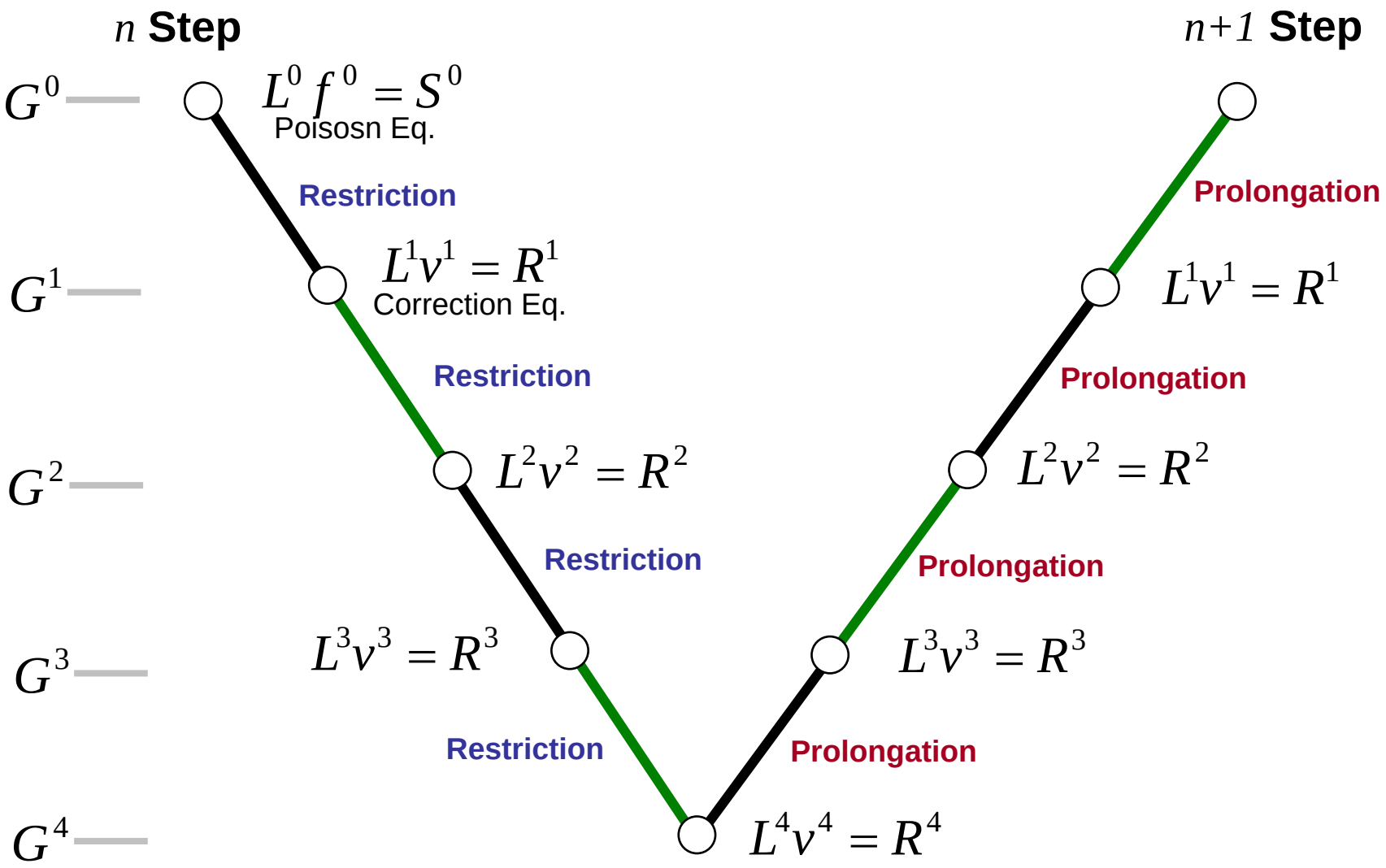
$$v_i^{k+1} = \frac{1}{4}\left(v_{i+1}^k + 2v_i^k + v_{i-1}^k\right)$$

Solving $k+1$-th correction equation $L^{k+1}v^{k+1} = R^{k+1}$



$$v_i^k = \frac{1}{2}\left(v_{2i+1}^{k+1} + v_{2i}^{k+1}\right)$$

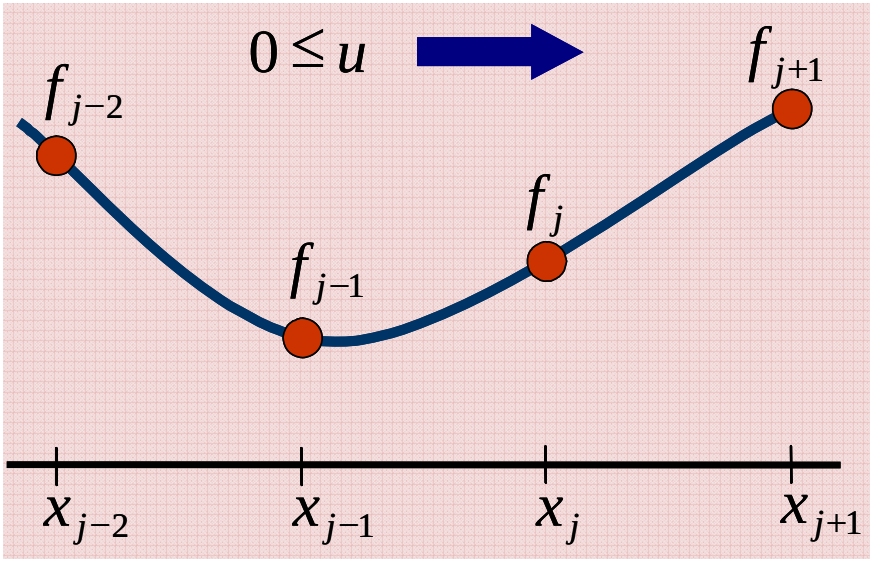
V-Cycle MG


n Step
n+1 Step
G^0
G^1
G^2
G^3
G^4
$L^0 f^0 = S^0$
Poisosn Eq.
$L^1 v^1 = R^1$
Correction Eq.
$L^2 v^2 = R^2$
$L^3 v^3 = R^3$
$L^4 v^4 = R^4$
$L^3 v^3 = R^3$
$L^2 v^2 = R^2$
$L^1 v^1 = R^1$
Restriction
Restriction
Restriction
Restriction
Prolongation
Prolongation
Prolongation
Prolongation

Two-dimensional Advection Equation

Most basic hyperbolic equation:

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + v\frac{\partial f}{\partial y} = 0$$



Cubic Semi-Lagrangian Scheme (3rd-order accuracy in time and space)

$$f_j^{n+1} = F_c^n(x_j - u\Delta t) = a(-u\Delta t)^3 + b(-u\Delta t)^2 + c(-u\Delta t) + f_j^n$$

$$a = \frac{f_{j+1}^n - 3f_j^n + 3f_{j-1}^n - f_{j-2}^n}{6\Delta x^3} \quad b = \frac{f_{j+1}^n - 2f_j^n + f_{j-1}^n}{2\Delta x^2} \quad c = \frac{2f_{j+1}^n + 3f_j^n - 6f_{j-1}^n + f_{j-2}^n}{6\Delta x}$$

Two-dimensional Advection Equation

Frontgenesis velocity profile:

GeForce 8800 GTS

56 GFLOPS

×60

1024×1024

Two-Stream Instability
in Plasma Physics

Vlasov-Poisson Equation:

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

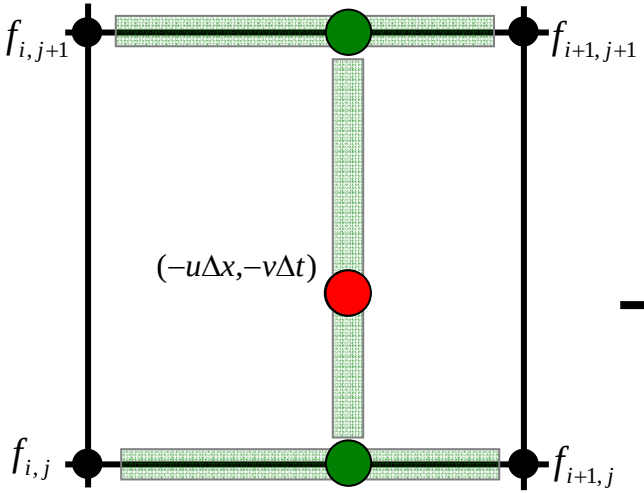
$$\left(E = -\frac{\partial \phi}{\partial x}, \quad n_e = \int f dv \right)$$

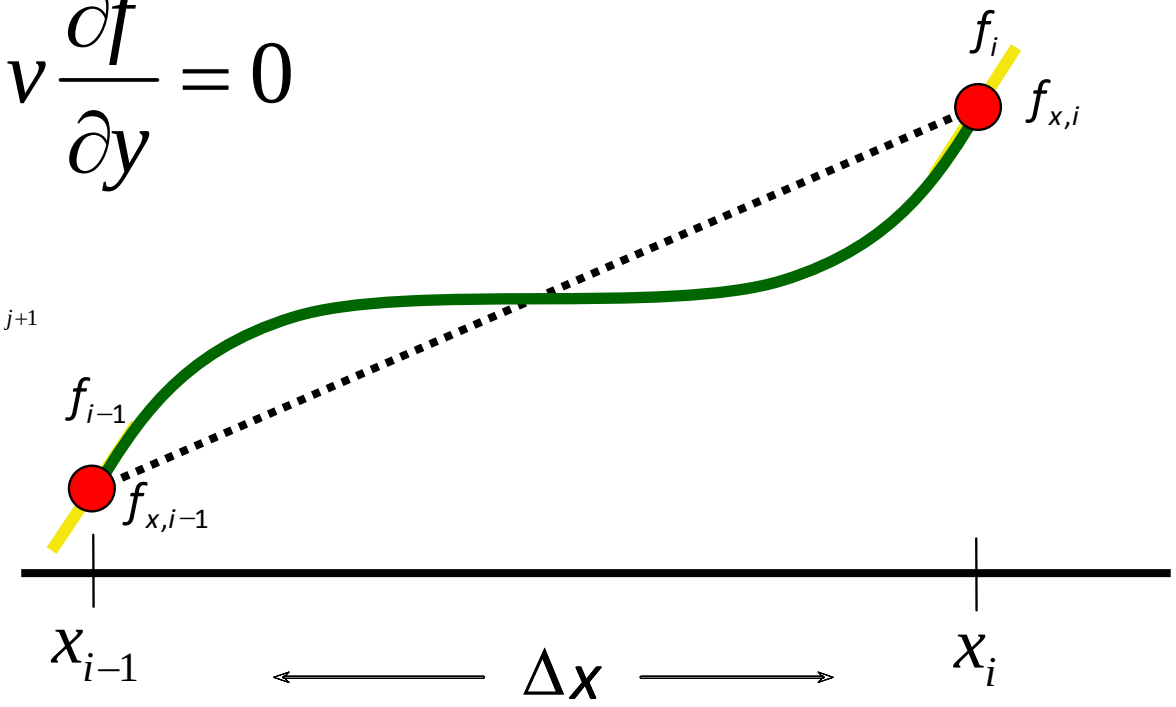
f **: electron distribution function**

n **: electron number density**

CIP Method for 2-dimensional Advection Equation

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + v\frac{\partial f}{\partial y} = 0$$

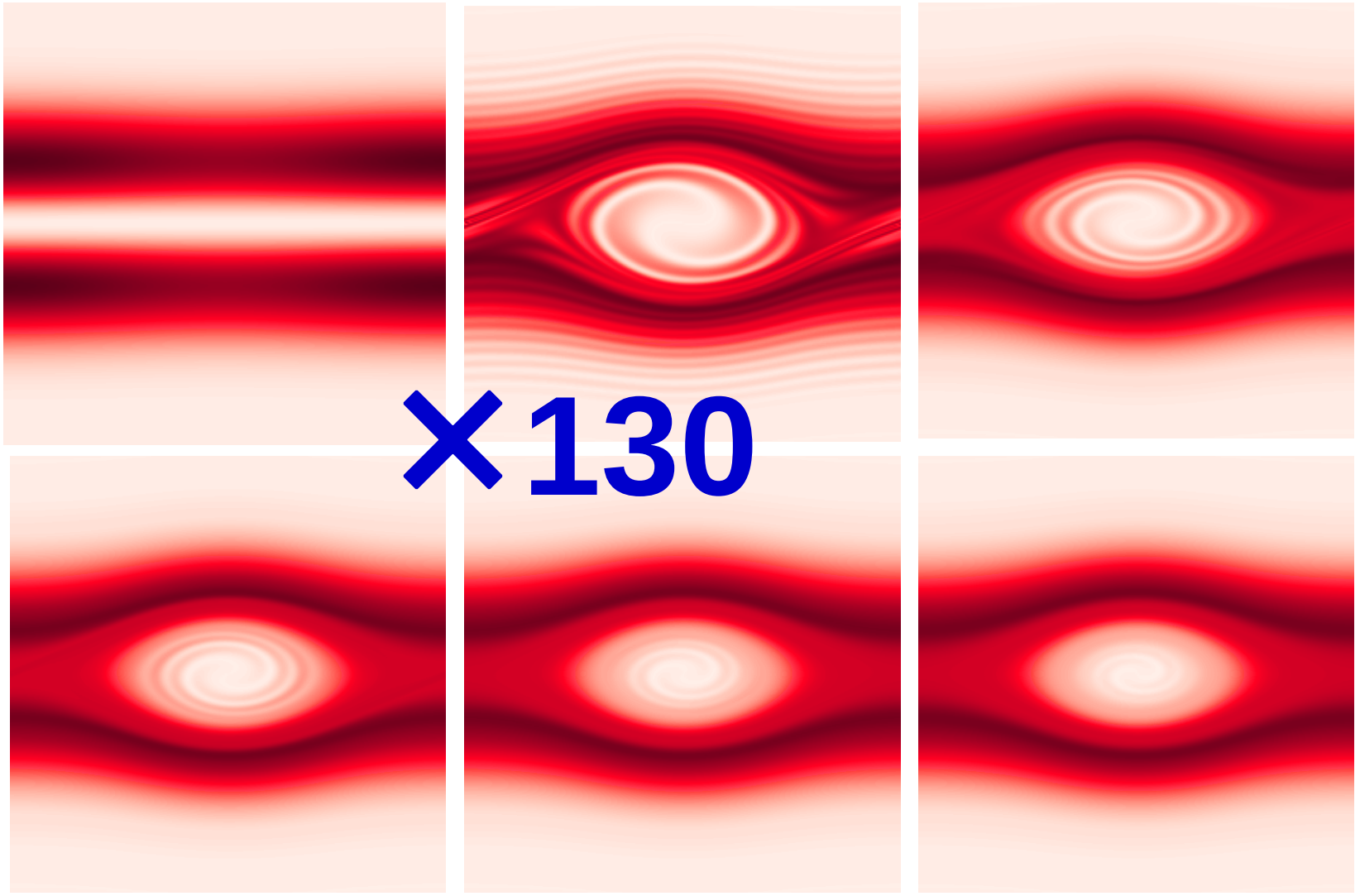




$$f_i^{n+1} = F_{CIP}(-u\Delta x) = a\xi^3 + b\xi^2 + f_{x,i}\xi + f_i$$

$$a = \frac{1}{\Delta x^2}\left(f_{x,i} + f_{x,i-1}\right) - \frac{2}{\Delta x}\left(f_i - f_{i-1}\right), \quad b = \frac{1}{\Delta x}\left(2f_{x,i} + f_{x,i-1}\right) - \frac{3}{\Delta x^2}\left(f_i - f_{i-1}\right)$$

120 GFLOPS using 8800GTS



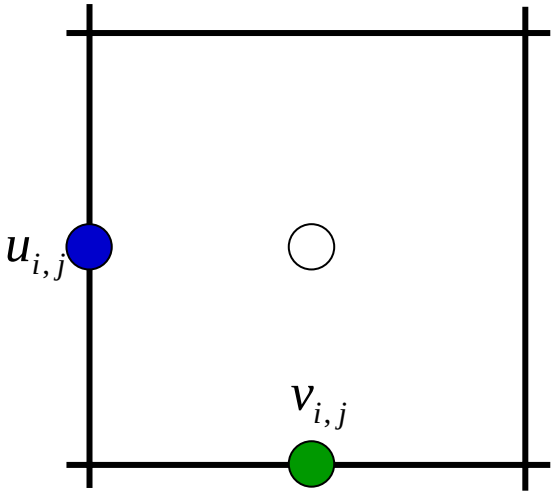
Two-dimensional Burgers Equation

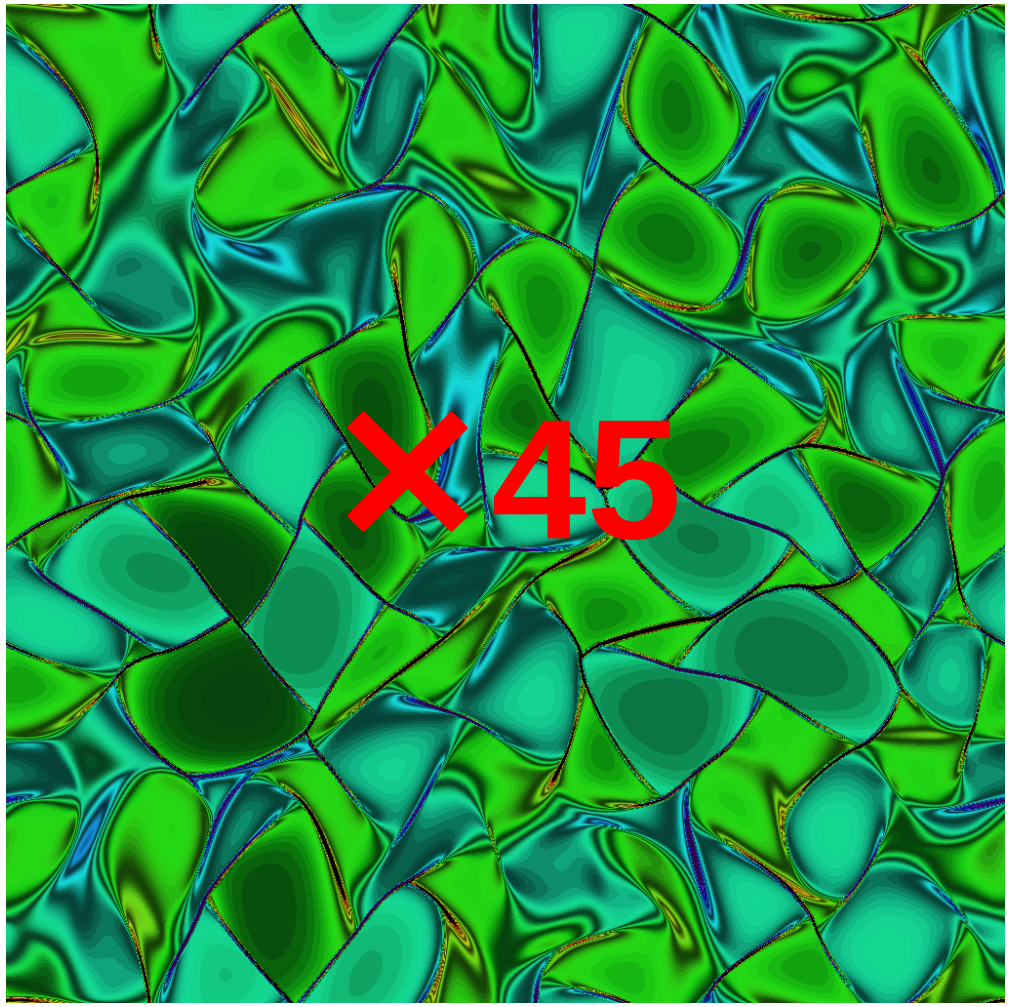
$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=\kappa\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}=\kappa\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)$$

GeForce 8800 GTS

40 GFLOPS





1024×1024

velocity u at the *v-point* $u_s=\dfrac{u_{i,j}+u_{i+1,j}+u_{i,j-1}+u_{i+1,j-1}}{4}$

Homogeneous Isotopic Turbulence

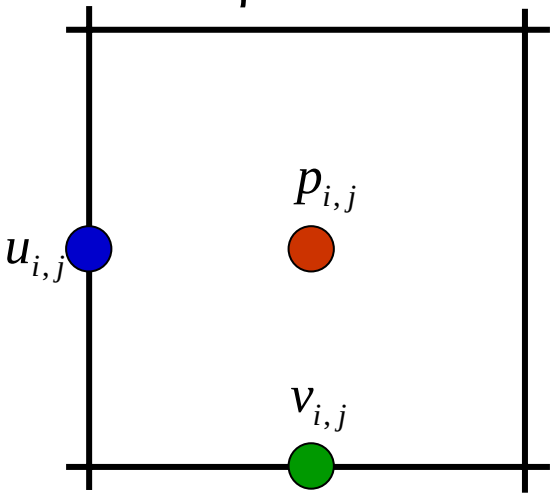
Burgers equation

Poisson equation

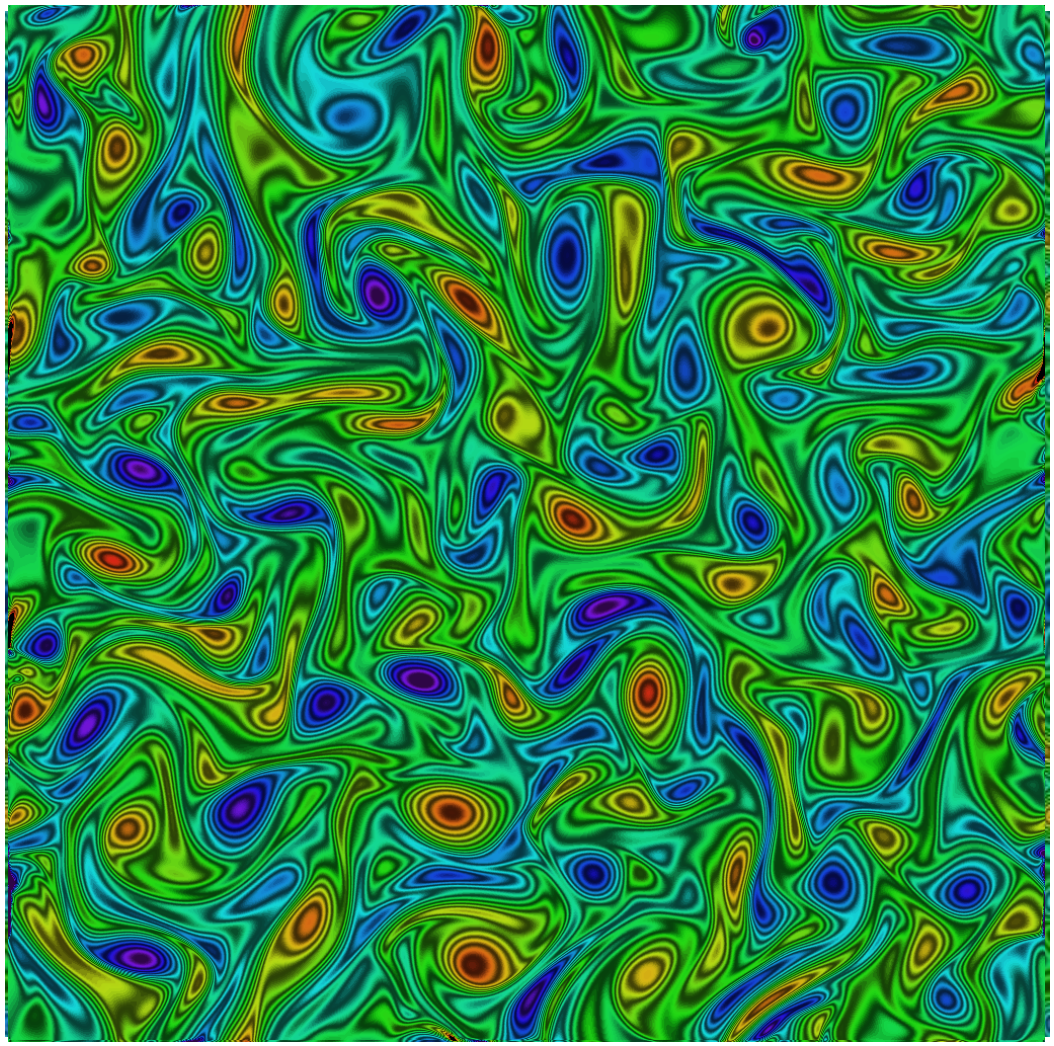
$$\frac{\partial^2 p}{\partial x^2}+\frac{\partial^2 p}{\partial y^2}=\frac{1}{\Delta t}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)$$

Correction

$$\frac{\partial u}{\partial t}=-\frac{1}{\rho}\frac{\partial p}{\partial x} \qquad \frac{\partial v}{\partial t}=-\frac{1}{\rho}\frac{\partial p}{\partial y}$$



1024×1024



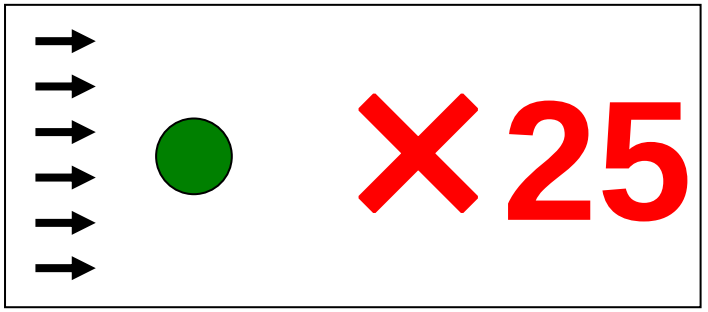
Incompressible Flow around circle

(Aerodynamics analysis)

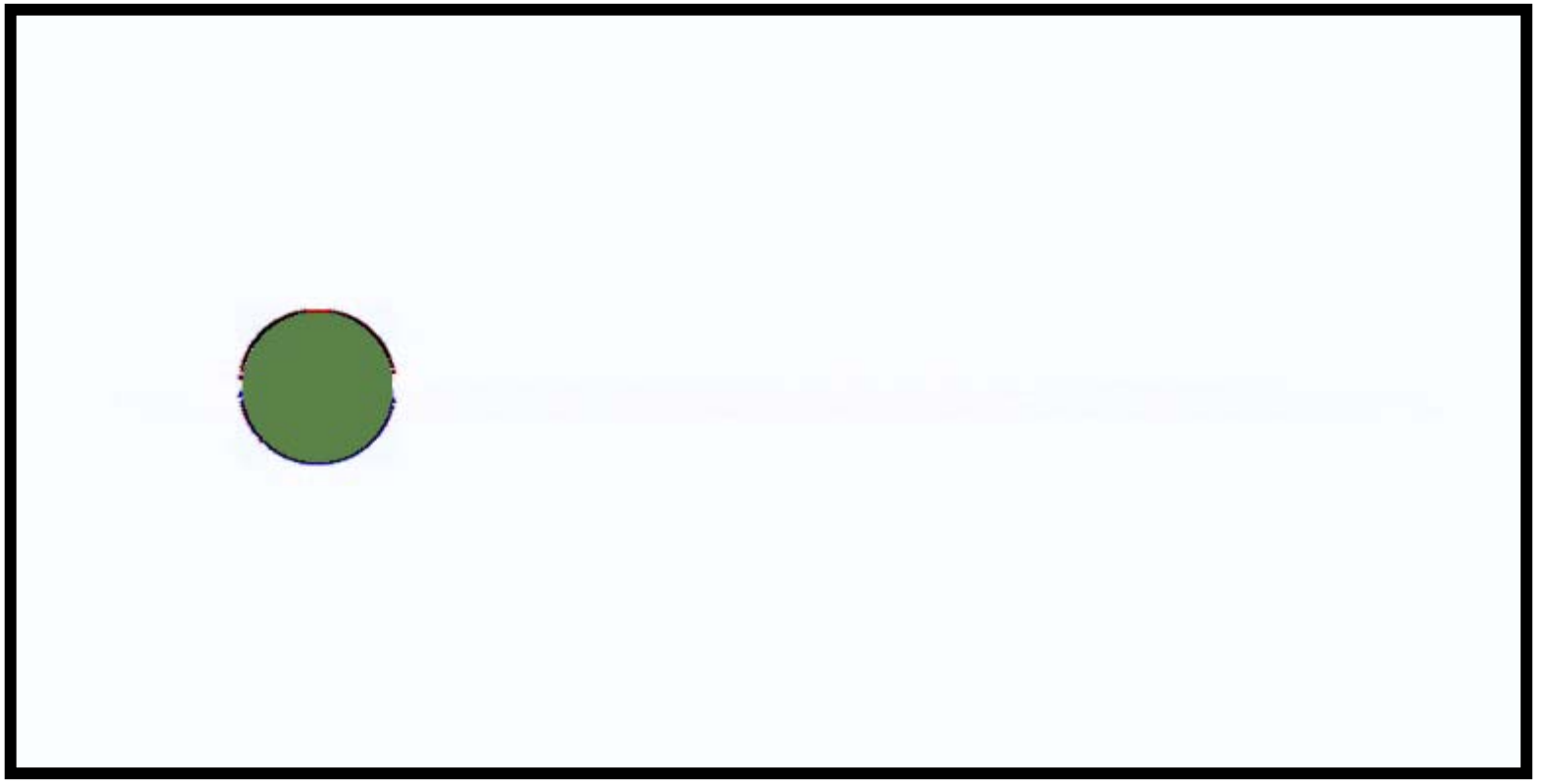
Incoming: Uniform Flow

Karman Vortex Street
behind the circle

Reynolds number: Re = 2000

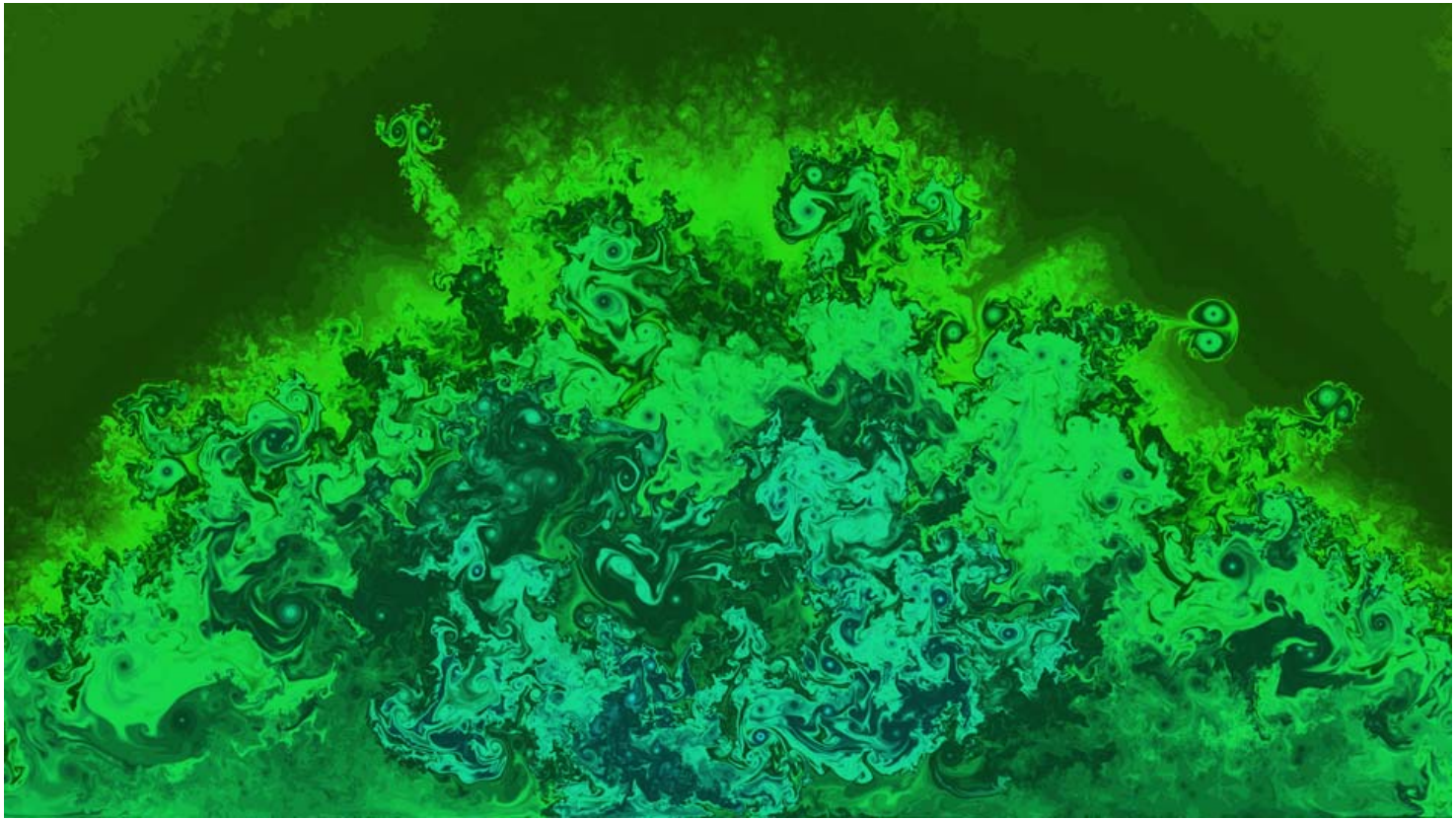


1024×512



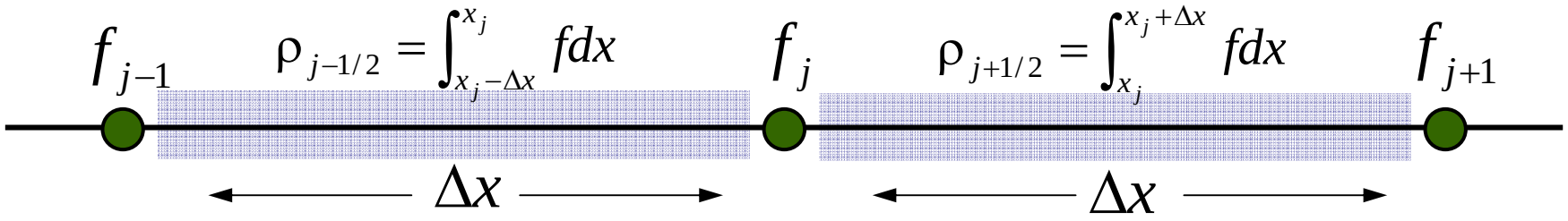
Compressible CFD Application

High-accurate numerical Scheme becomes very important.



Numerical Scheme IDO-CF

Y. Imai, T. Aoki and K. Takizawa, J. Comp. Phys., Vol. 227, Issue 4, 2263-2285 (2008)



$$F(x) = ax^4 + bx^3 + cx^2 + dx + f_j$$

$F(\Delta x) = f_{j+1}$,

Four matching conditions : $\int_{x_j-\Delta x}^{x_j} F(x)dx = \rho_{j-1/2}$ $F(-\Delta x) = f_{j-1}$ $\int_{x_j}^{x_j+\Delta x} F(x)dx = \rho_{j+1/2}$

Unknown coefficients : $c = \frac{5}{4}\frac{3\rho_{j+1/2} + 3\rho_{j-1/2} - 6f_j\Delta x}{\Delta x^3} - \frac{3}{4}\frac{f_{j+1} - 2f_j + f_{j-1}}{\Delta x^2}$ $d = 2\frac{\rho_{j+1/2} - \rho_{j-1/2}}{\Delta x^2} - \frac{f_{j+1} - f_{j-1}}{2\Delta x}$

$$\frac{\partial}{\partial x}F(0) = 2\frac{\rho_{j+1/2} - \rho_{j-1/2}}{\Delta x^2} - \frac{f_{j+1} - f_{j-1}}{2\Delta x}$$

$$\frac{\partial^2}{\partial x^2}F(0) = \frac{5}{2}\left(\frac{3\rho_{j+1/2} + 3\rho_{j-1/2} - 6f_j\Delta x}{\Delta x^3}\right) - \frac{3}{2}\left(\frac{f_{j+1} - 2f_j + f_{j-1}}{\Delta x^2}\right)$$

Rayleigh-Taylor Instability

Heavy fluid lays on light fluid and unstable.

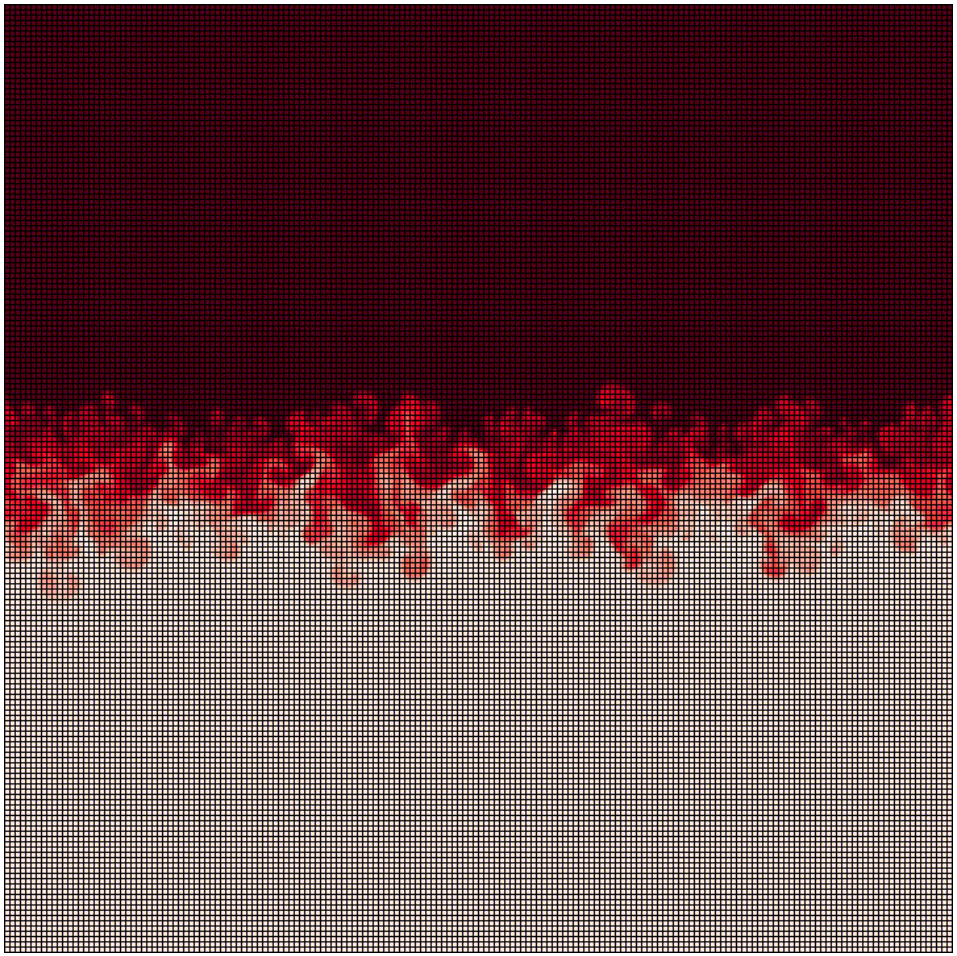
×90

512 x 512

Euler equation:

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

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



88 GFLOPS using GTX280

Phase Separation

Phase transition dynamics is described by the Phase Field Model.

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

$$\frac{\partial H}{\partial \psi} = \tau\psi - u\psi^3$$

Discretization:

$$\frac{\partial^4 \psi}{\partial x^4} = \frac{\psi_{i+2,j} - 4\psi_{i+1,j} + 6\psi_{i,j} - 4\psi_{i-1,j} + \psi_{i-2,j}}{\Delta x^4}$$

$$\frac{\partial^4 \psi}{\partial x^2 \partial y^2} = \left(\psi_{i+1,j+1} - 2\psi_{i,j+1} + \psi_{i-1,j+1} \right.$$

$$- 2\psi_{i+1,j} + 4\psi_{i,j} - 2\psi_{i-1,j}$$

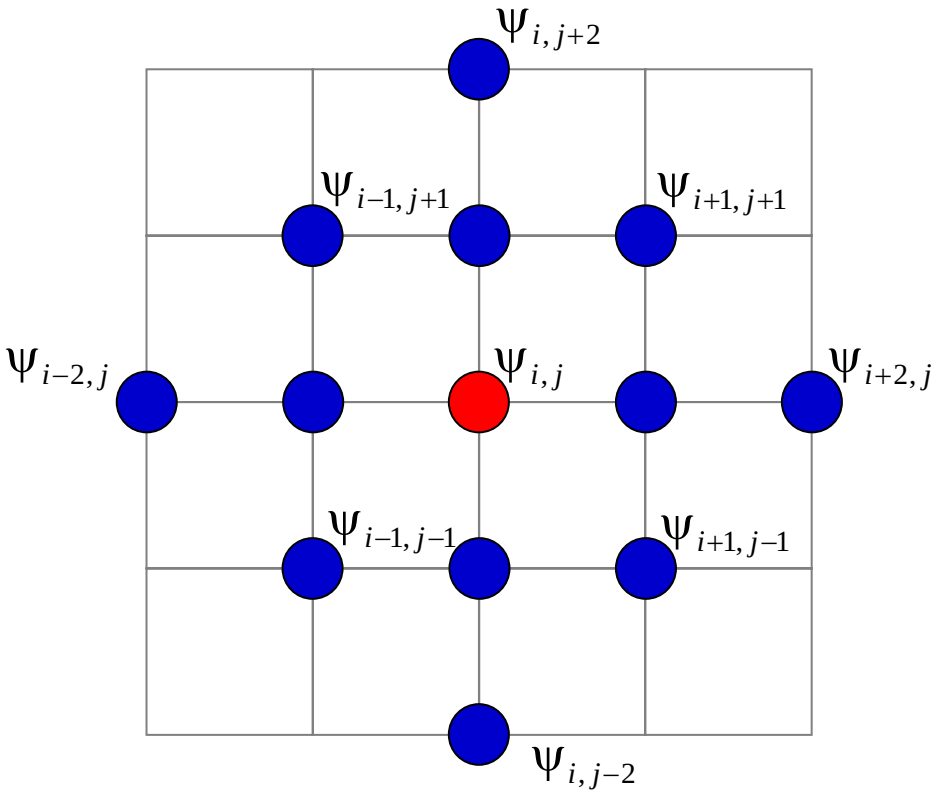
$$\left. + \psi_{i+1,j-1} - 2\psi_{i,j-1} + \psi_{i-1,j-1} \right)$$

2-D Computation of Phase Separation

Mixture of Oil and Water:

114 GFLOPS using GTX280

512 x 512



3-D Computation of Phase Separation

Mixture of Oil and Water:

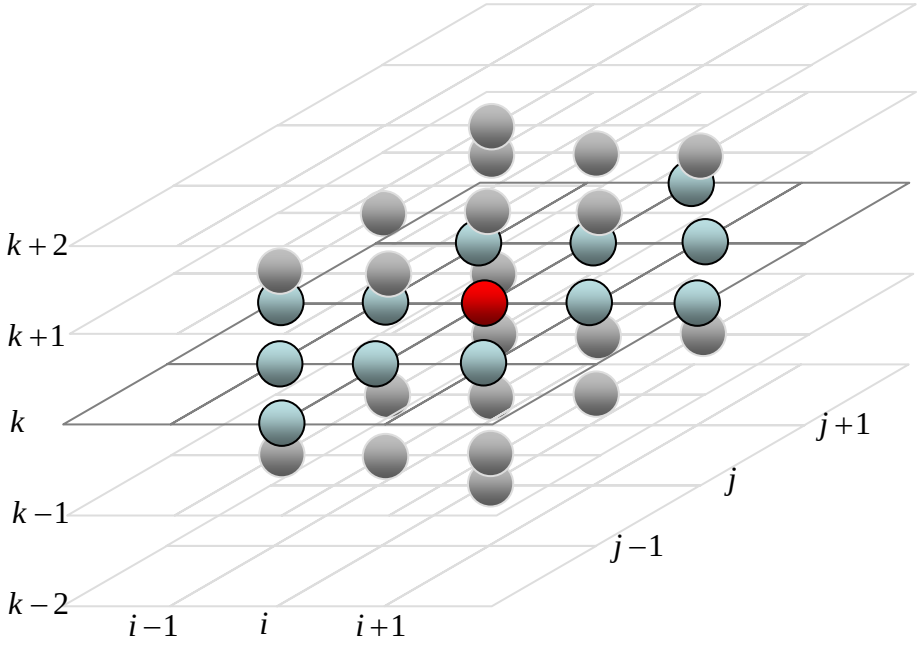
158 GFLOPS using GTX280

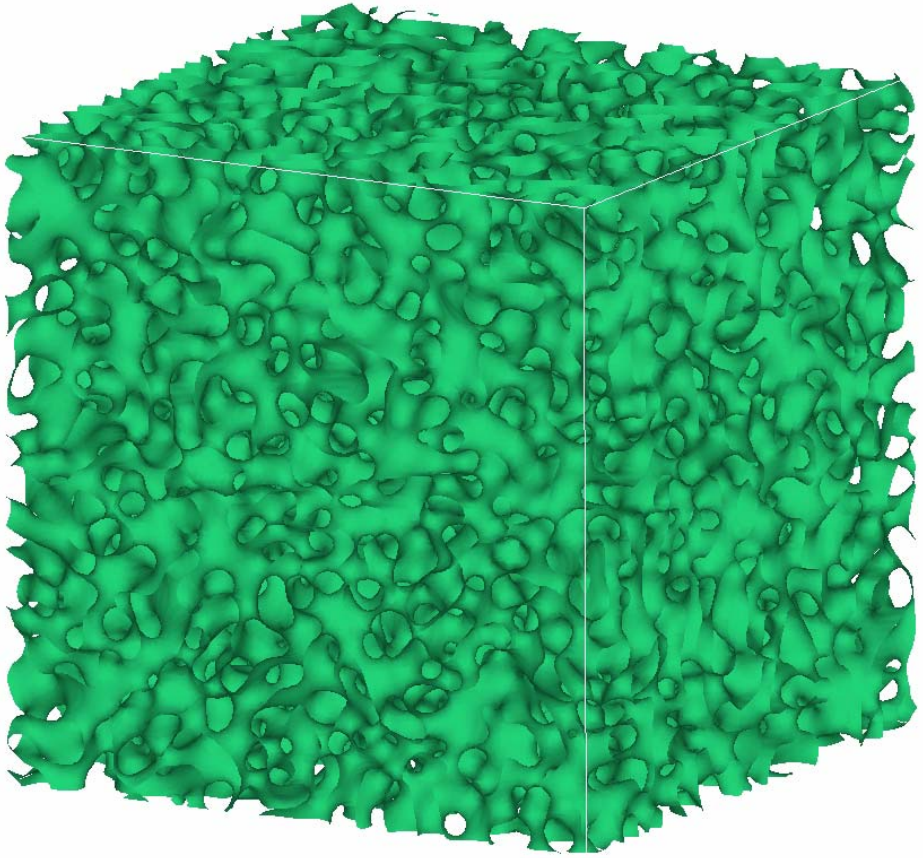
×160

256 x 256 x 256

Used register number = 46

※ nvcc option –maxrregcount 32
for G80, 92





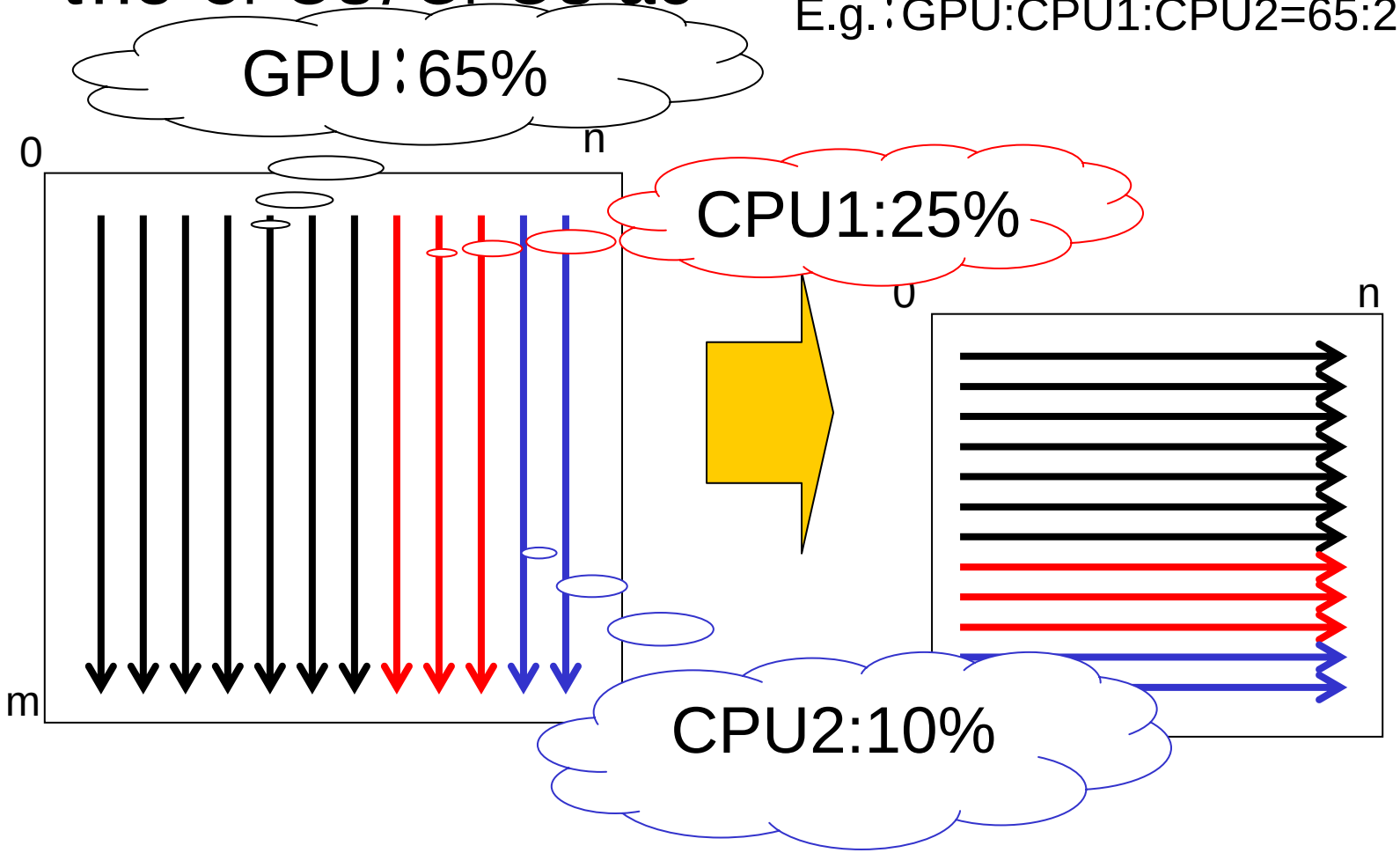
CFD Summary

- Mesh-type CFD calculations are suitable for GPU
- The shared-memory use is the key point
- 3-dimensiolal calculation and high-accurate numerical scheme can extract the GPU performance more.
- Card parallel and node parallel computings are the next issues for large-scale problems.

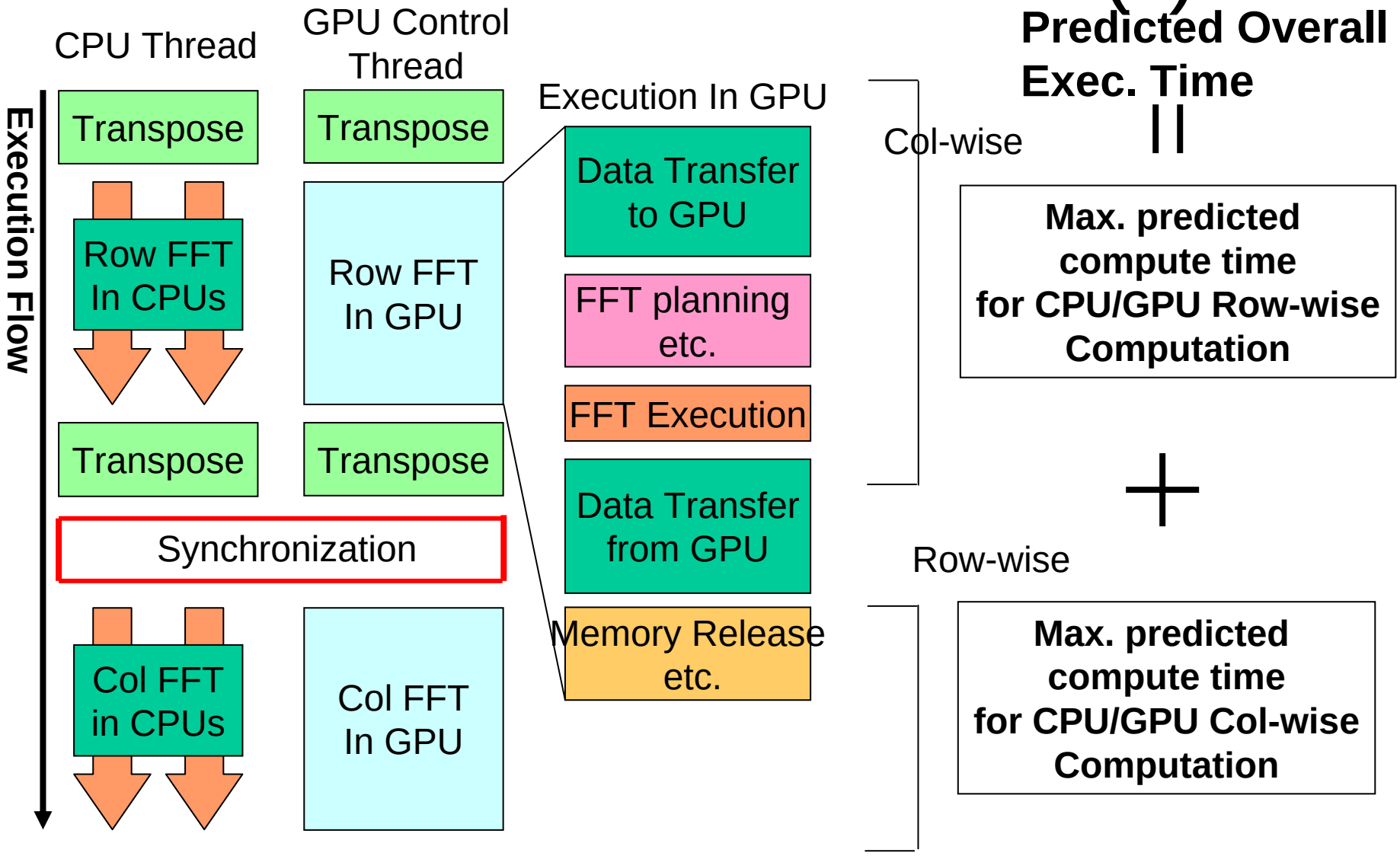
Dividing the Labor in 2D FFT ([IEEE IPDPS-HCW08])

- Divide the labor for each 1D FFT among the CPUs/GPUs ac

E.g.：GPU:CPU1:CPU2=65:25:10



Performance Model of Combined CPU/GPU FFT Execution (1)



Performance Model (2)

Total exec time	$T_{total} = \max(T_{gpu1},T_{cpu1})+\max(T_{gpu2},T_{cpu2})$
Row-wise phase on GPU	$T_{gpu1}= r\ n^2\log(n)K_{gFFT} + r\ n^2(K_{gMa}+K_{c2g}+K_{gP}+K_{gDP}+K_{g2c}) + 2rn^2*M_{tr}(r)*K_{tr}$
Col-wise phase on GPU	$T_{gpu2}= r\ n^2\log(n)K_{gFFT} + r\ n^2(K_{c2g}+K_{gP}+K_{gDP}+K_{g2c}+K_{gMf})$
Row-wise phase on CPU	$T_{cpu1}= (1-r)n^2\log(n)K_{cFFT} + 2(1-r)n^2*K_{tr}*M_{tr}(1-r)$
Col-wise phase on CPU	$T_{cpu2}= (1-r)n^2\log(n)K_{cFFT}$
Adjustment of mat. trans.	$M_{tr}(x) = (1-C_{tr})(x-0.5)/0.5+C_{tr}$ (if $x > 0.5$) C_{tr} (if $x \leqq 0.5$)

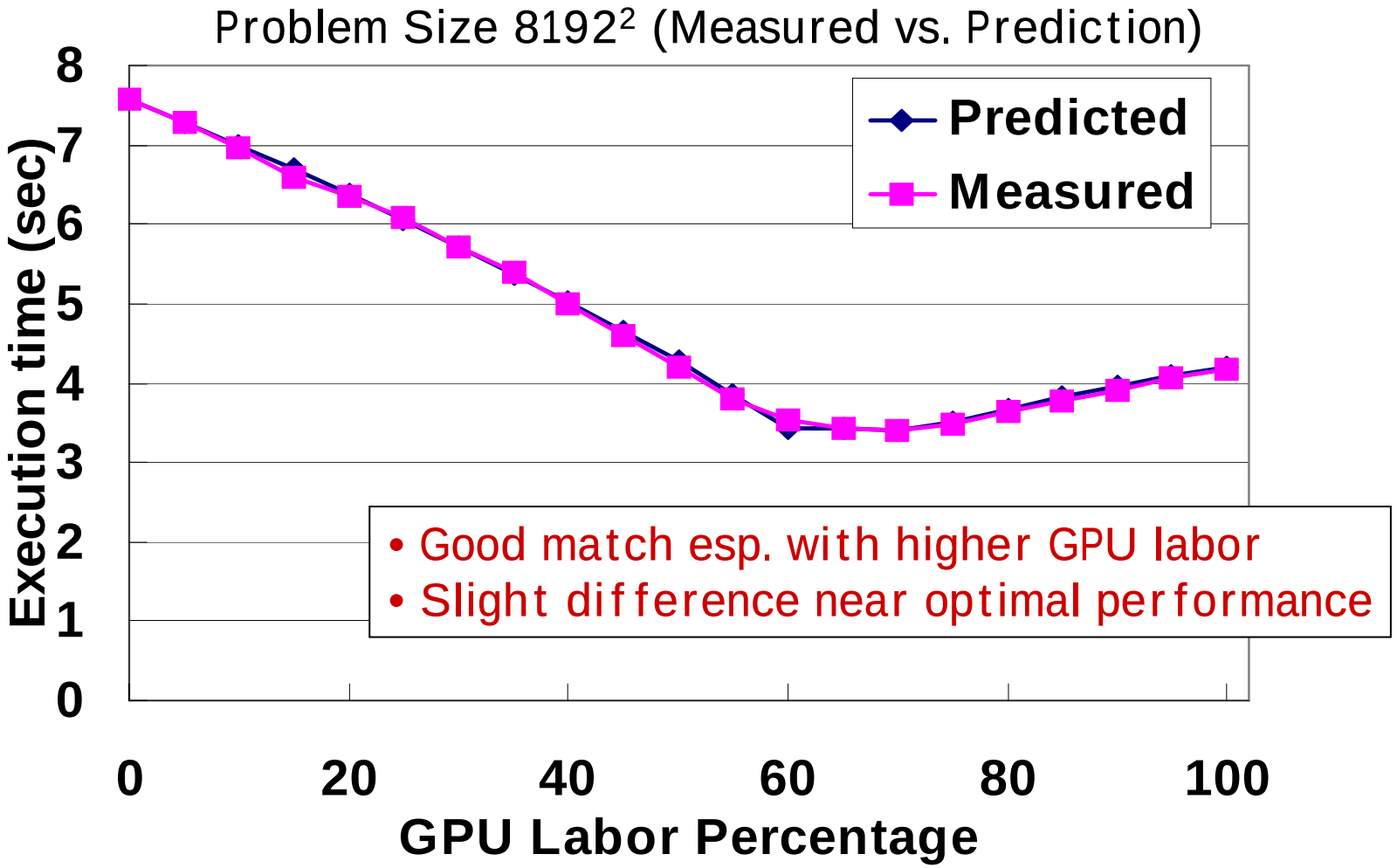
n: problem size, r: allocation ratio on GPU ($0 \leqq r \leqq 1$)

Parameters to be determined:

K_{cFFT}	FFT on CPU	K_{gP}	Initialize FFT on GPU
K_{gFFT}	FFT on GPU	K_{gDP}	Finalize FFT on GPU
K_{c2g}	Comm. from CPU to GPU	K_{gMa}	Memory alloc on GPU
K_{g2c}	Comm. from GPU to CPU	K_{gMf}	Memory free on GPU
K_{tr}	Matrix transposition	C_{tr}	Adjust mat. Trans.

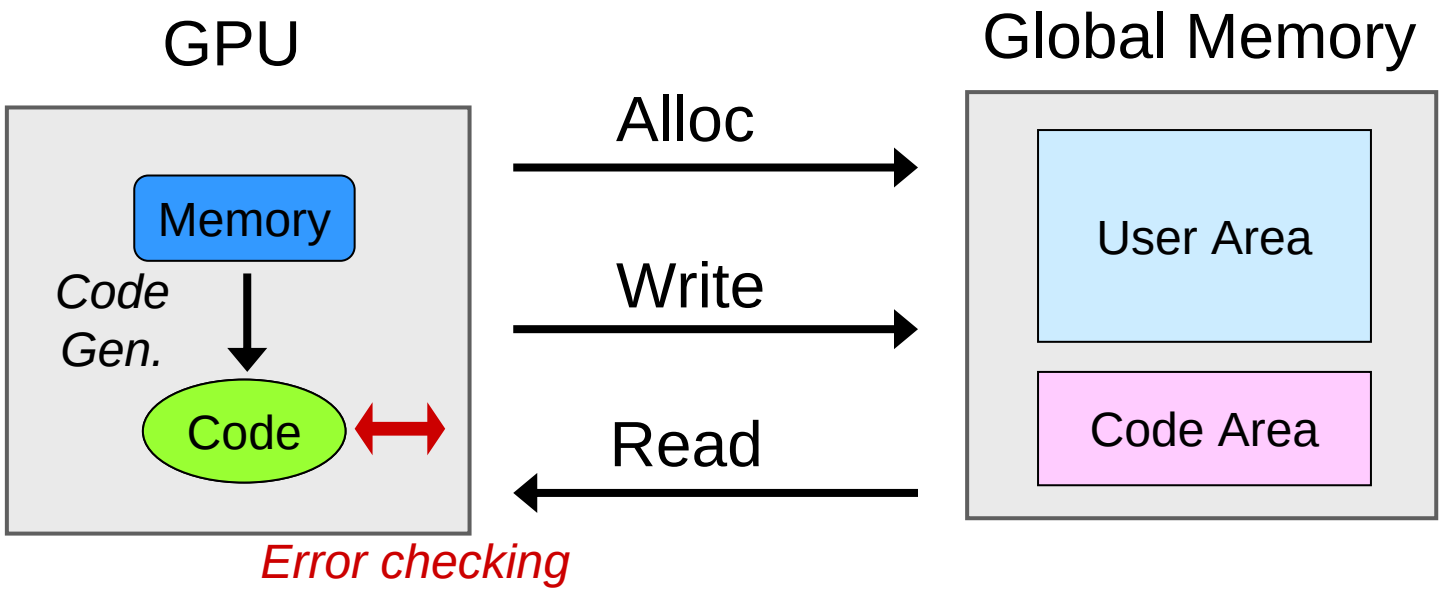
Performance Model Accuracy (1)

- Problem Size 8192^2, 1 CPU + GPU
- Predicted perf. of 8192^2 derived from our model



Software-Based Error Checking

- Protects applications from bit-flip errors in global memory using error-checking code (e.g., parity and ECC)
  - memory allocation → *allocates additional global memory for storing error check code*
  - write accesses → *calculates code for data written*
  - read accesses → *calculates and compares code for data read*



Prototype Implementation for Feasibility Studies

- Library-based approach
  - Implements necessary functions as a library for CUDA
  - Hides the complexity of dealing with error checking as much as possible
  - Easy to prototype
  - Can be tedious for users
- Automated approach
  - Automatically translates PTX
  - Might be possible if nvopencc is really “open”
  - Not easy to prototype
  - More transparent for users

Matrix Multiply with Error Checking

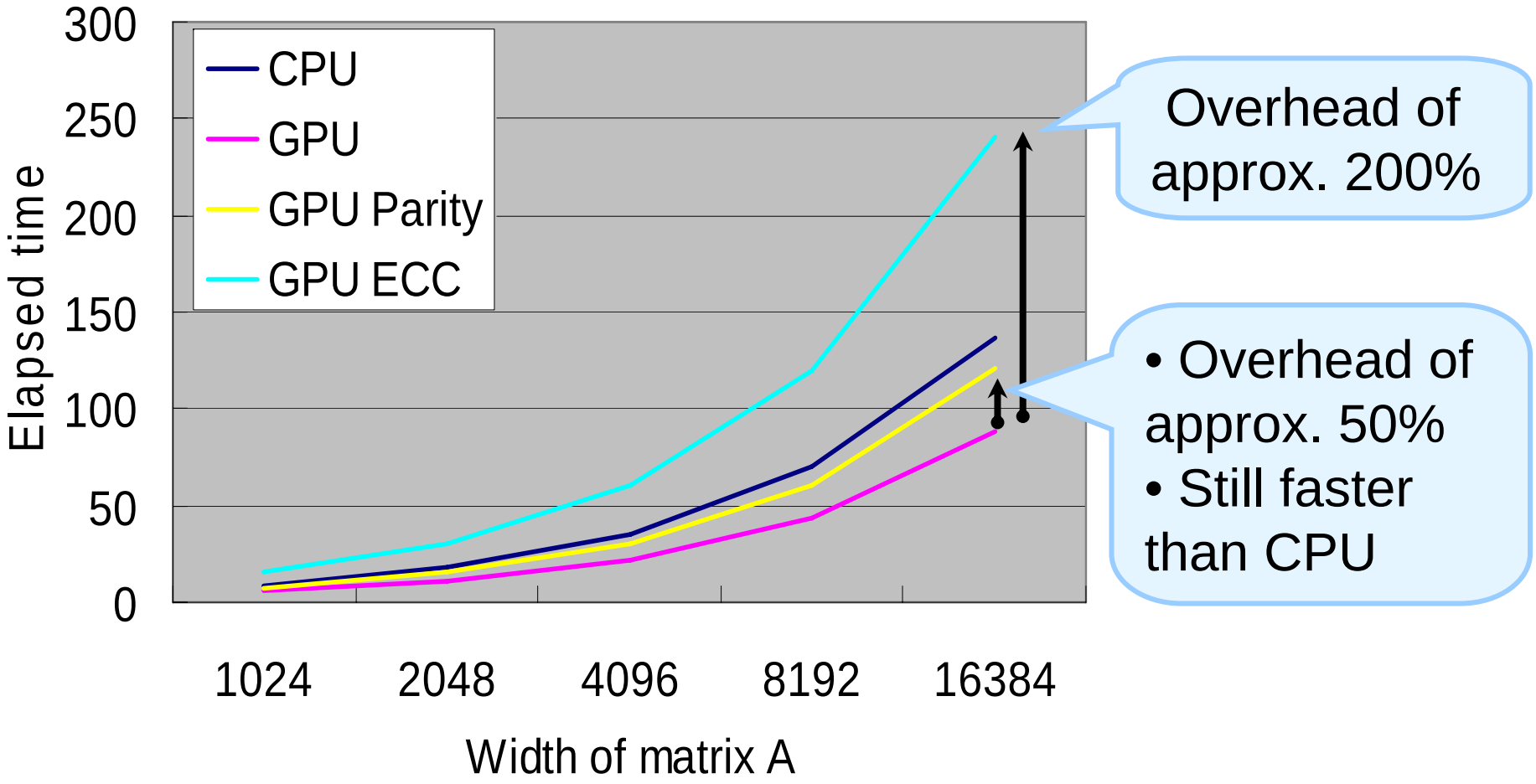
- The sample code shipped with the CUDA SDK
  - computes A x B=C by a simple blocking algorithm
- Error checking
  - read accesses to input matrices A and B
  - write accesses to output matrix C

```
matMul(float *A, unsigned *A_code,
       float *B, unsigned *B_code,
       float *C, unsigned *C_code){
  shared sA[][];
  shared sB[][];
  float sum = 0.0;
  for (i, j in BLOCK) {
    sA[i][j] = A[i][j];
    check_float32(sA[i][j]);
    sB[i][j] = B[i][j];
    check_float32(sB[i][j]);
    for (k, l in block) {
      sum += sA[k][l] * sB[k][l];
    }
  }
  C[i][j] = sum;
  write_check_code32(sum);
}
```

Results of Matrix Multiply

CPU (GotoBLAS using 4 cores) vs GPU vs GPU+Parity vs GPU+ECC

Phenom 9850 2.5 GHz, 4 GB of DRAM, Linux v2.6.23, gcc v4.1.2, GeForce 8800 GTS 512, CUDA v2.0 Beta 2



N-body Problem with Error Checking

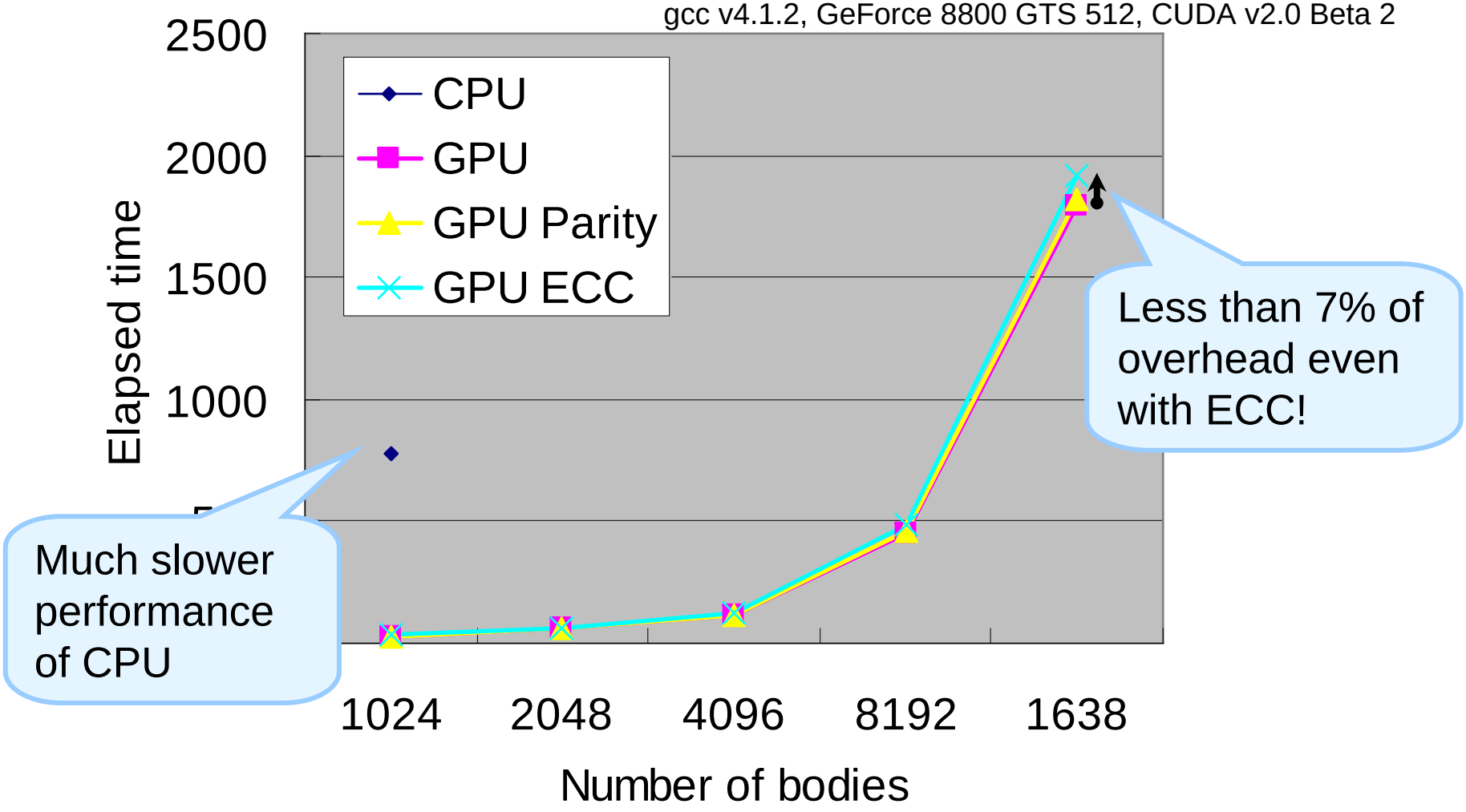
- The sample code shipped with the CUDA SDK
- Error checking
  - read accesses to position array
  - write accesses to position and velocity arrays

```
nbody (float *pos, unsigned *pos_code,
       float *vel, unsigned *vel_code) {
  float4 *mypos = pos[i];
  check_float4(mypos, i);
  for (j in BLOCK) {
    shared float4 block[];
    block[k] = pos[k];
    check_float4(block[k], k);
    for (pt in block_size) {
      acc+=compute_acc(mypos,block[pt]);
    }
  }
  update_pos(pos, vel);
  write_check_code_float4(pos);
  update_vel(vel, acc);
  write_check_code_float4(vel);
}
```

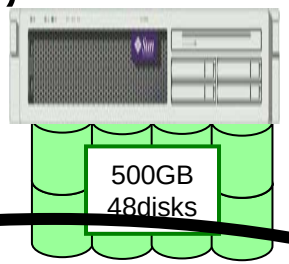
Results of N-body Problem

CPU (4 cores with OpenMP) vs GPU vs GPU+Parity vs GPU+ECC

Phenom 9850 2.5 GHz, 4 GB of DRAM, Linux v2.6.23, gcc v4.1.2, GeForce 8800 GTS 512, CUDA v2.0 Beta 2



Design Study: Interim TSUBAME 1.2 (Hypothetical, still under plans…)



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

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

NEC SX-8i

10Gbps+External NW

Unified Infiniband network

~800TFlops-SFP,
~180TFlops-DFP
Balanced Scalar-Vector Design

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

Nvidia Tesla T10P-one card per node, ~650 cards
High Performance in Many BW-Intensive Apps
10% Power consumption increase over TSUBAME 1.0 (130TF SFP / 80TF DFP)

TSUBAME Upgrades Towards Petaflops Sustained Acceleration