The “How” of Exascale

Satoshi Matsuoka
Tokyo Institute of Technology

June 28, 2011
MEXT Symposium Panel

DoE Exascale 性能指標

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

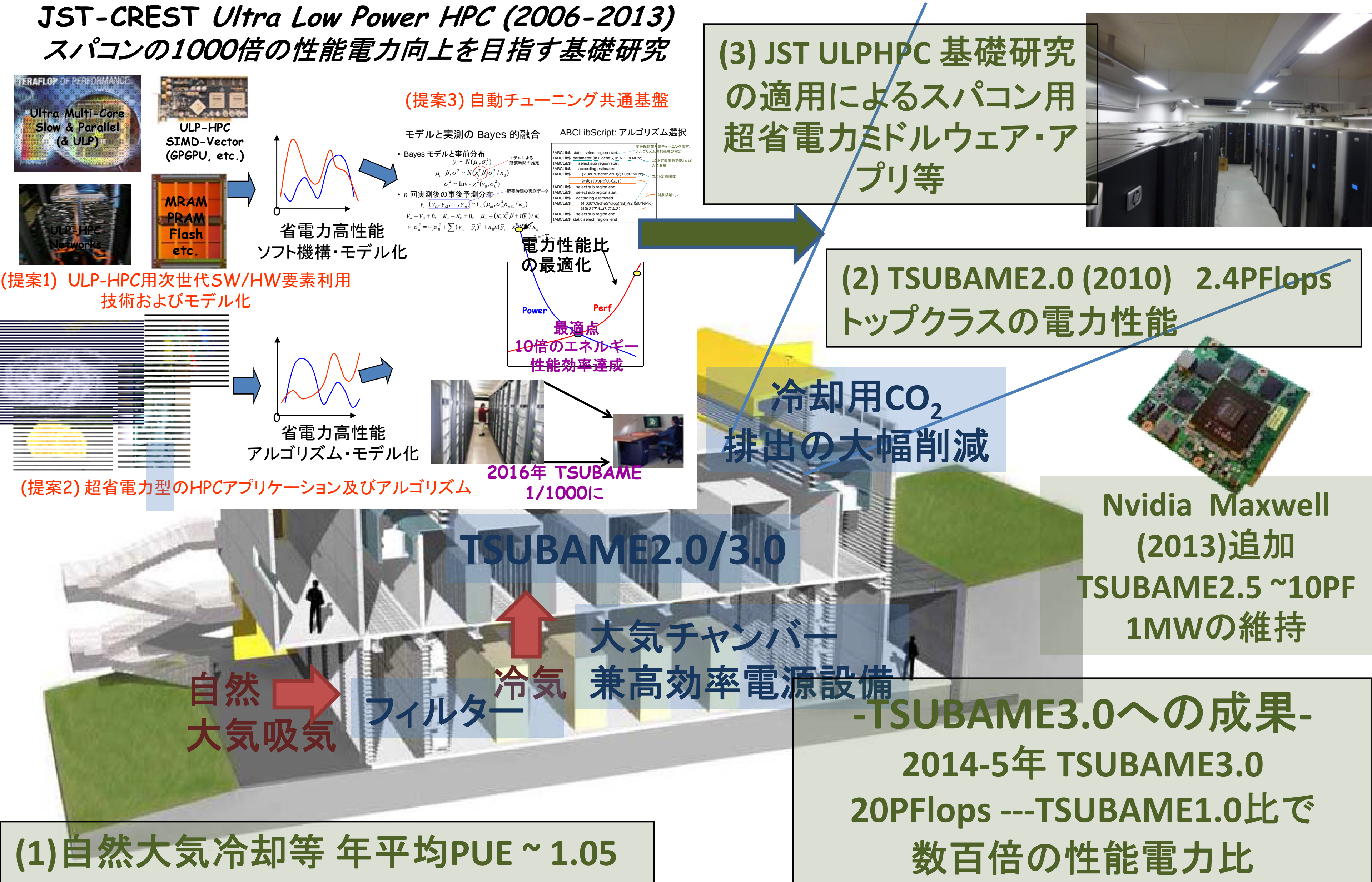
Important Strategy Towards Exascale

- ***Identify Key Application Areas (incl. Stakeholders)***
  - Disaster, Environment, Energy, Life/Medical, etc...
- ***Identify Technology Strength (of Japanese HPC)***
  - Device, Archiecture, Software, Applications
- ***Identify the Impediments and do Research***
  - Power, Resiliency, Programming, ...
- ***Learn from Existing Systems***
  - ES2, T2K, Tsubame2.0, K Computer ...
- ***Set Performance and other Metric Timelines***
  - 2010 1PetaFlop (done), 2012 10PetaFlops (to be done), 2015 100PetaFlops, 2018 ExaFlops, etc...

「エクサ10億並列へ」は勇ましいが。。。

- *電力・エネルギー (JST CREST ULPHPC + MEXT Green HPC)*
- (強)スケーリングの欠落 (Co-Design Center)
- *N^2 vs. N 問題により深まるメモリ階層*
  - *(ネットワークやI/O含む)*
  - *(レーテンシとバンド幅) (Device & Architecture)*
- 極端に低まる信頼性と実行不能性 (科研基盤S)
- *プログラミングや実行モデル (JST CREST Post Petascale)*

JST-CREST ULP-HPCの成果適用による スパコンのウルトラグリーン化(H23~概算要求)



ULP-HPC 遂行計画・マイルストーン

2007Q3

基礎開発

各種要素技術の研究開発
- マルチコア・アクセラレータ関係の種々の電力最適化アルゴリズム・プログラミング機構・コンパイラ等
- 新世代メモリ利用を含むプログラム
- VMベースのシステム全体の電力最適化
- 低消費電力・高性能ネットワークの設計活用
- 流体構造および分子科学における省電力高性能アプリケーションアルゴリズム
-上記の電力性能モデル化、自動チューニング適用
-電力性能自動チューニング基盤(理論・システム)

-各種テストベッドでの実際のソフトウェア機構開発・モデルの制定・自動チューニング
-モデルベースのシミュレーション遂行・大規模性能電力最適化シミュレーション
-一部HW小プロトタイプの開発

TSUBAME2.0の設計

2009Q1

第一次統合化

各種要素技術の連携・統合開始
- 各要素のモデルベースを統合化した自動チューニングシステムを開発
- 結果をTSUBAME2.0の設計に反映

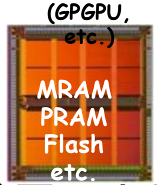
他の関連マシン

2010Q3

TSUBAME 2.0 完成・設置: 本研究の成果を用い<u>20倍以上の性能向上</u>を達成

新基盤上での更なる開発・最終統合化

ULP-HPC SIMD-Vector (GPGPU, etc.)

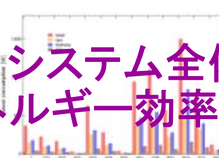


HWの要素活用によるポテンシャル x3~x30

アプリ最適化(x2)・
システム最適化 x2(平均)
冷却最適化 (x2)

2013Q4

自動チューニングで両ポテンシャル最大化・システム全体平均で10倍程度のエネルギー効率化



TSUBAME3.0の設計(2014年度完成)

Power Efficiency Compared

Machine	Power (incl. cooling)	Linpack Perf (PF)	Linpack Mflops/W	Factor	
Tsubame1.0 (2006Q1)	1.8MW	0.038	21	2368	
ORNL Jaguar (2009Q4)	~9MW	1.76	196	256	
Tsubame2.0 (2010Q4)	1.8MW	1.2	667	75	
K Computer (2011Q2)	~16MW	10	625	80	
BlueGene/Q (2012Q1)	~12MW?	17	1417	35.3	
Tsubame3.0 (2015Q1)	1.8MW	20	11000	4.6	
EXA (2018Q4)?	20MW	1000	50000	1	

Next Generation Numerical Weather Prediction[SC10]

Collaboration: Japan Meteorological Agency

Meso-scale Atmosphere Model:

Cloud Resolving Non-hydrostatic model [Shimokawabe et. al. SC10 BSP Finalist]

ex. WRF(Weather Research and Forecast)

Typhoon ~ 1000km

1~ 10km Tornado, Down burst, Heavy Rain

WSM5 (WRF Single Moment 5-tracer) Microphysics*

Represents condensation, precipitation and thermodynamic effects of latent heat release

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

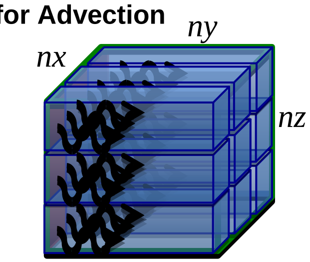
⇒ 20 x boost in microphysics (1.2 - 1.3 x overall improvement)

ASUCA : full GPU Implementation

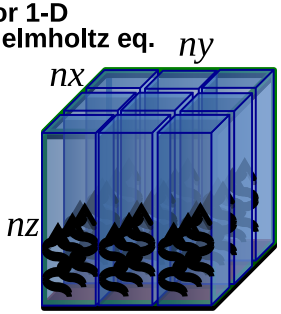
developed by Japan Meteorological Agency

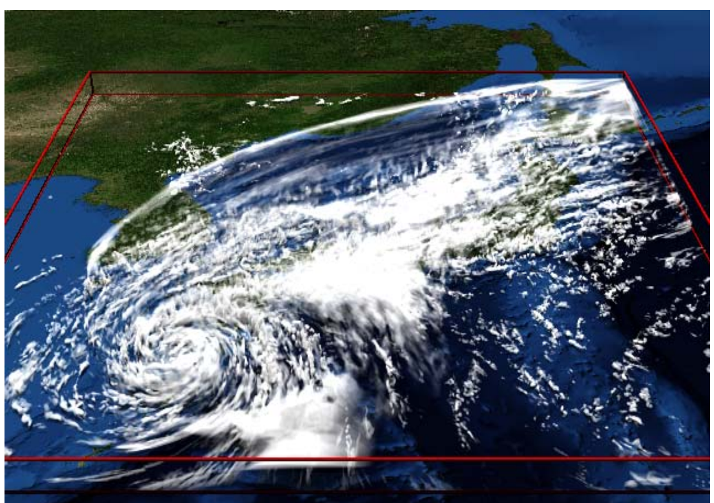
TSUBAME 2.0 : 145 Tflops World Record !!!

Block Division for Advection

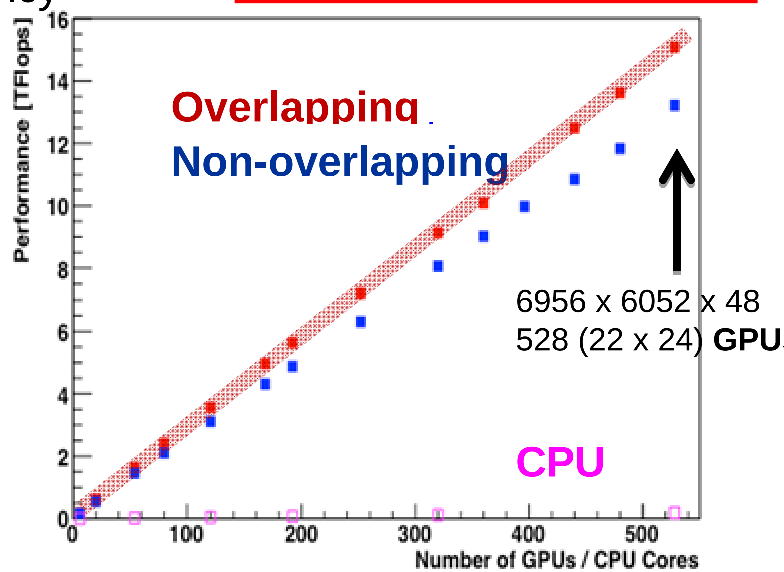


for 1-D Helmholtz eq.





Typhoon



ASUCA: Next Gen Weather Simulation Code

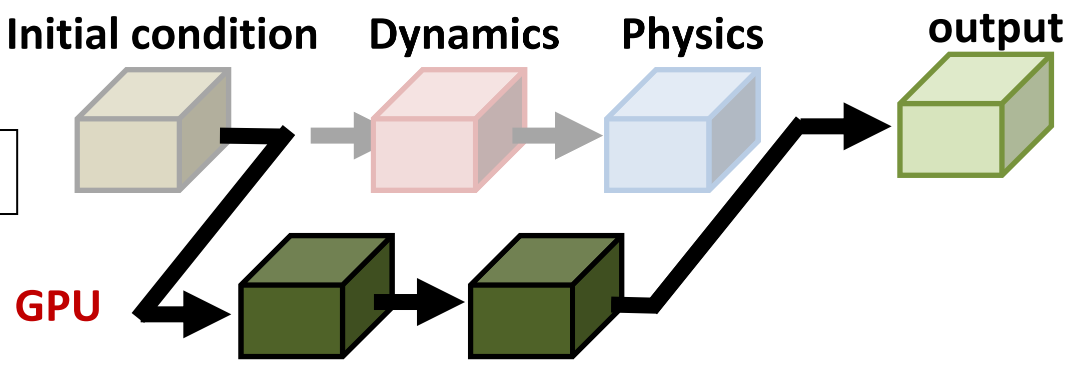
■ASUCA Production Code

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

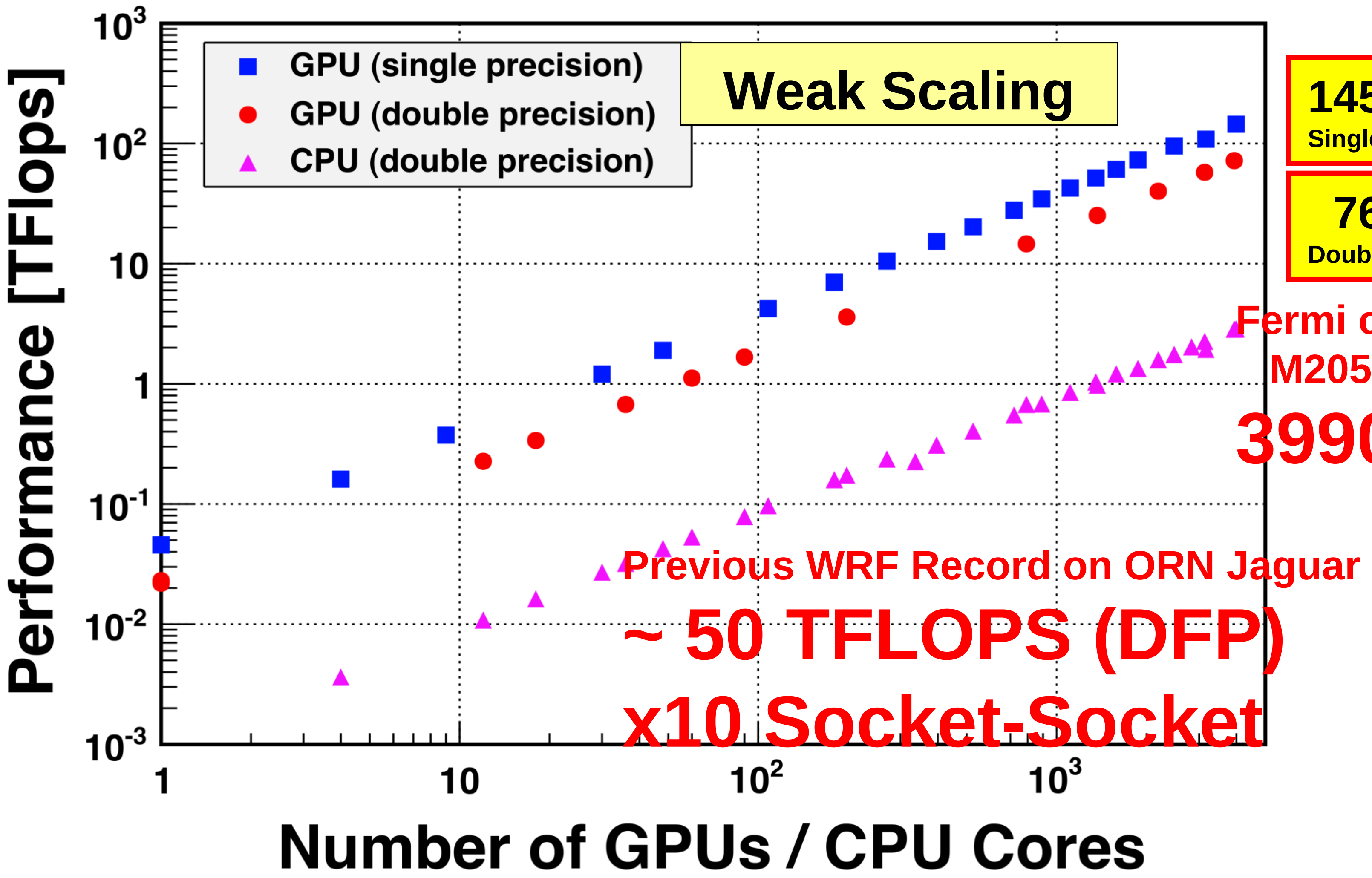
■Similar Structure as WRF

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

Full GPU Approach



TSUBAME 2.0 ASUCA Weather SImulationPerformance



145.0 Tflops
Single precision

76.1 Tflops
Doublele precision

Fermi core Tesla M2050

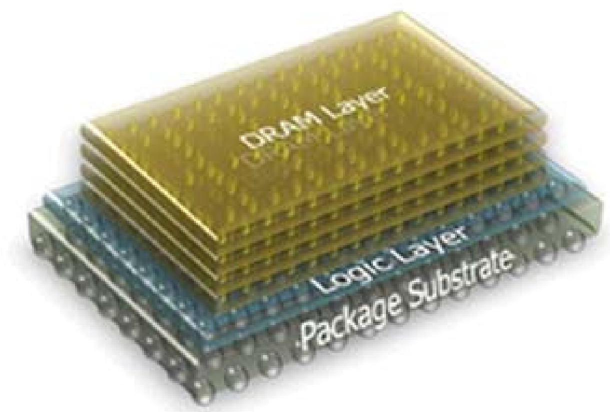
3990 GPUs

Memory Bandwidth Compared

Machine	Nodes	Stream BW/Node (GB/s)	GFLOPs /Node	Byte/Flop	Peak (PF)	Aggregate BW (TB/s)
Jaguar (2009)	18700	26	125	**0.2**	2.33	**486**
Tsubame 2.0 (2010)	4000 (GPUs)	130	515	**0.25**	2.06	**520**
K (2011)	80000	46	128	**0.35**	10	**3680**
BG/Q (2012)					20	**6-7000?**
Tsubame 3.0 (2015)	???	???	**???**	**???**	>25	**7000**

Many Core Drives Memory Innovation

- Disruptive DRAM memory packaging and interfaces beyond DDR3: TSV and Micron Hybrid Memory Cube



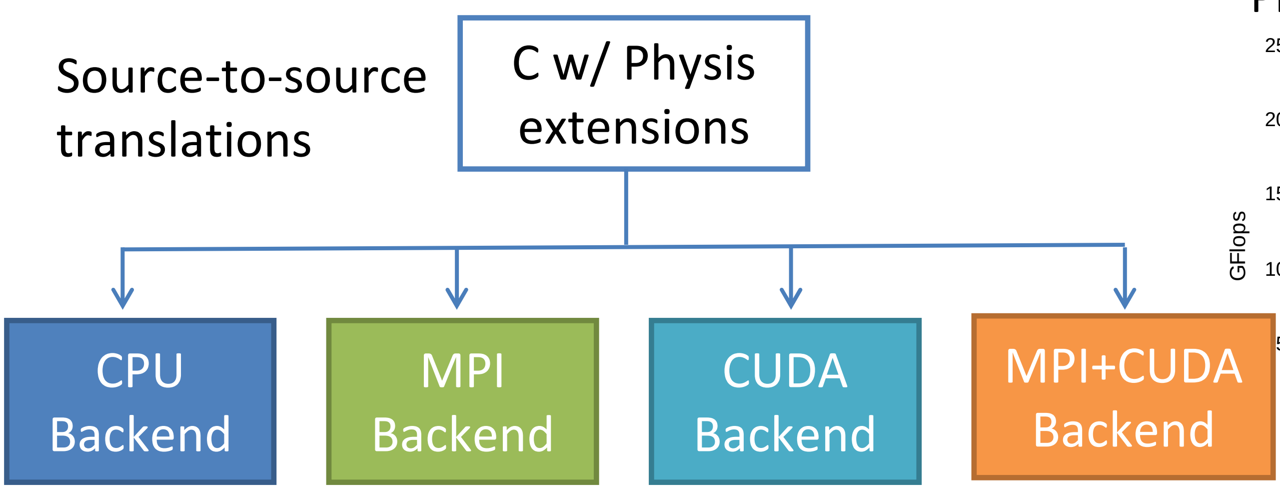
  - x20 bandwidth, low latency 1/10 power
  - Useful for Smarphones, too!

- Non-volatile memory
  - New flash, Phase Change Memory…

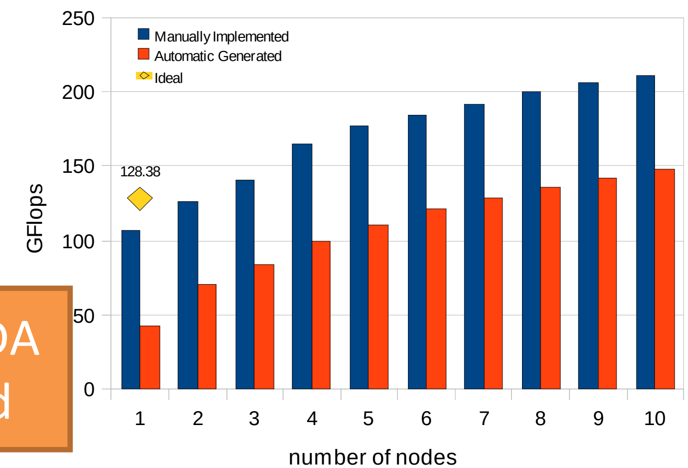
Physis: A Domain Specific Framework for Large-Scale GPU Clusters

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

Source-to-source translations

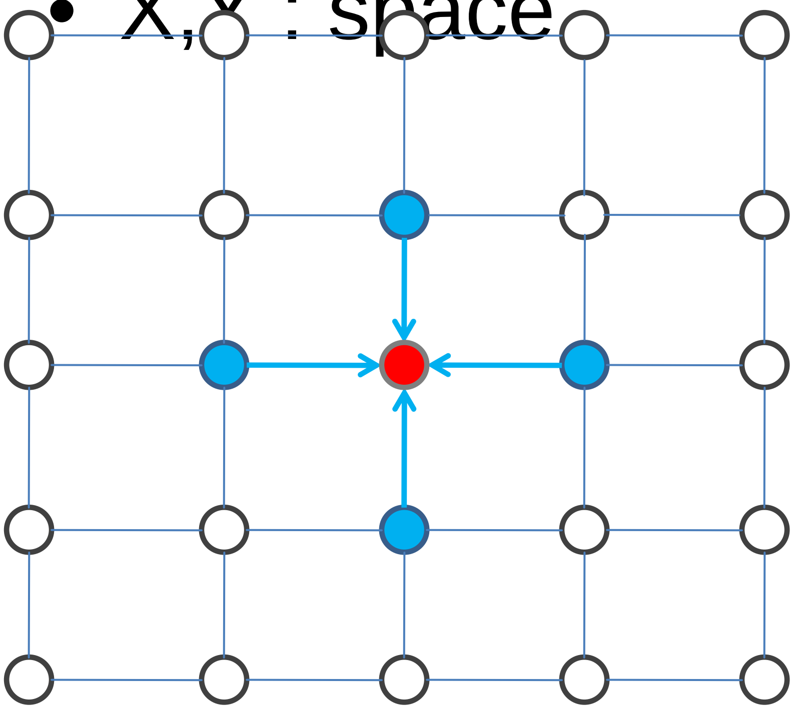


Preliminary Performance Results



A five-point stencil

- T : time
- X,Y : space



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

An example of a 7-point stencil

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

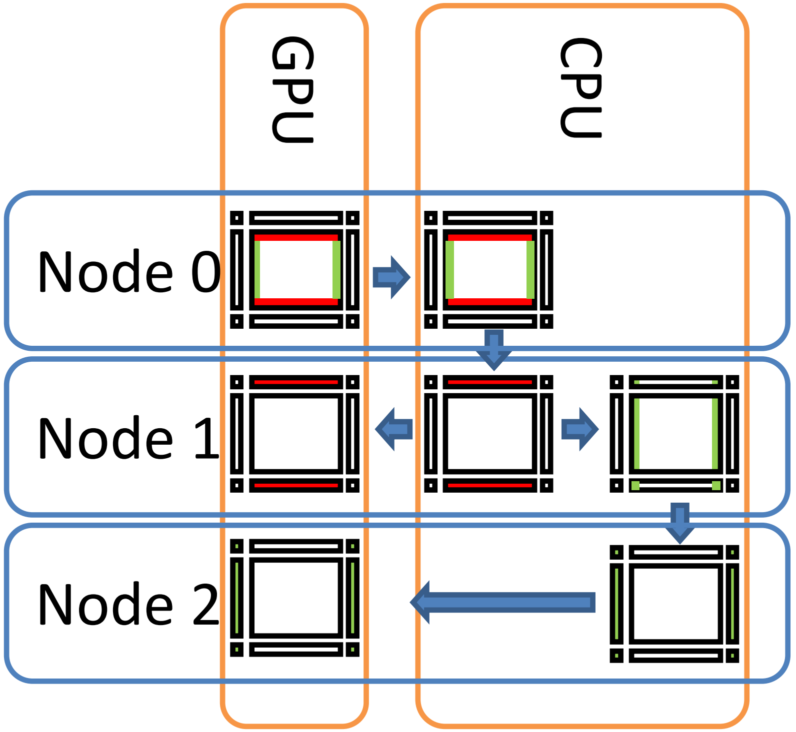
Parallel execution

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

Apply stencil kernel to a grid

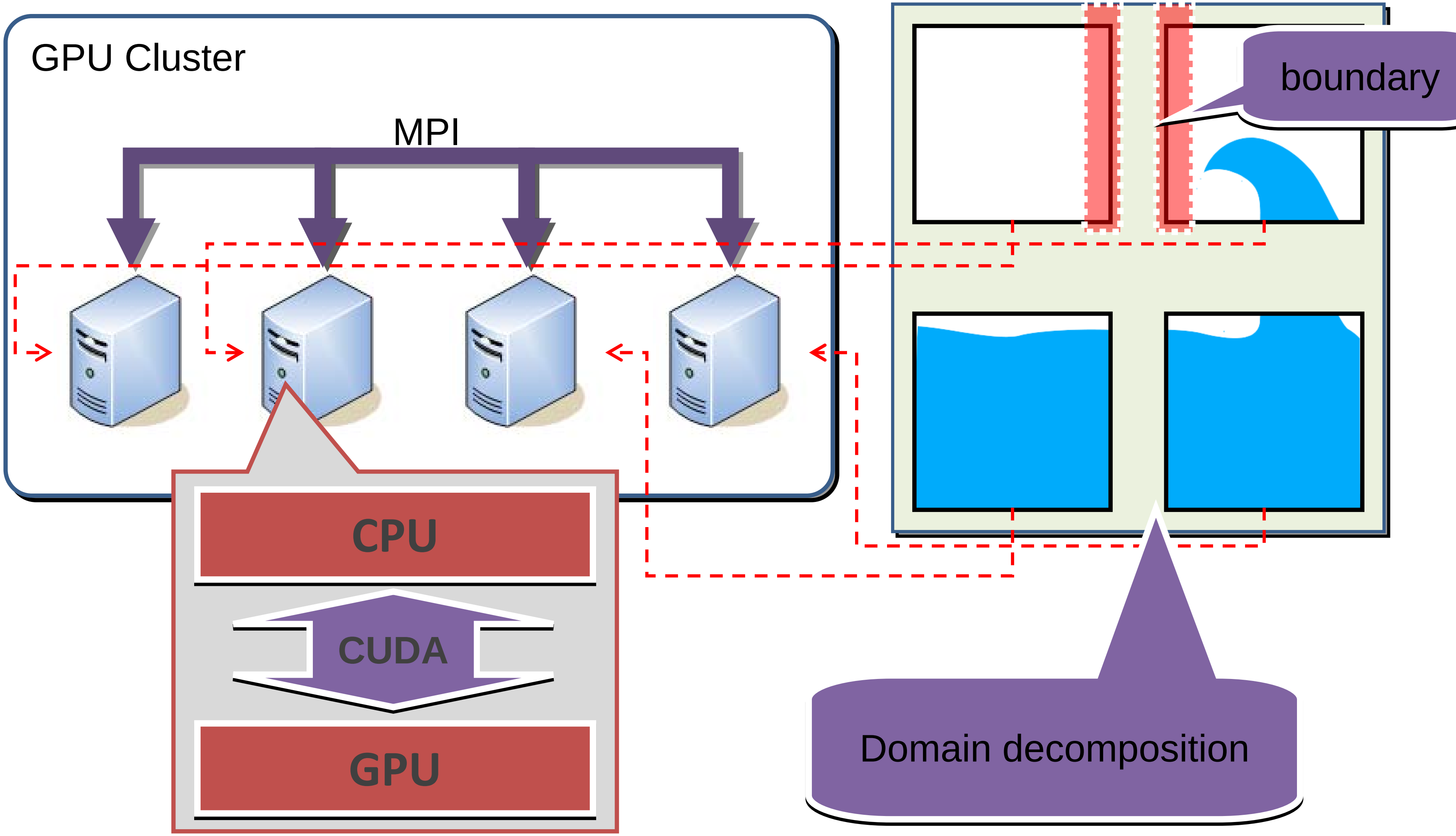
Complexity of implementation

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

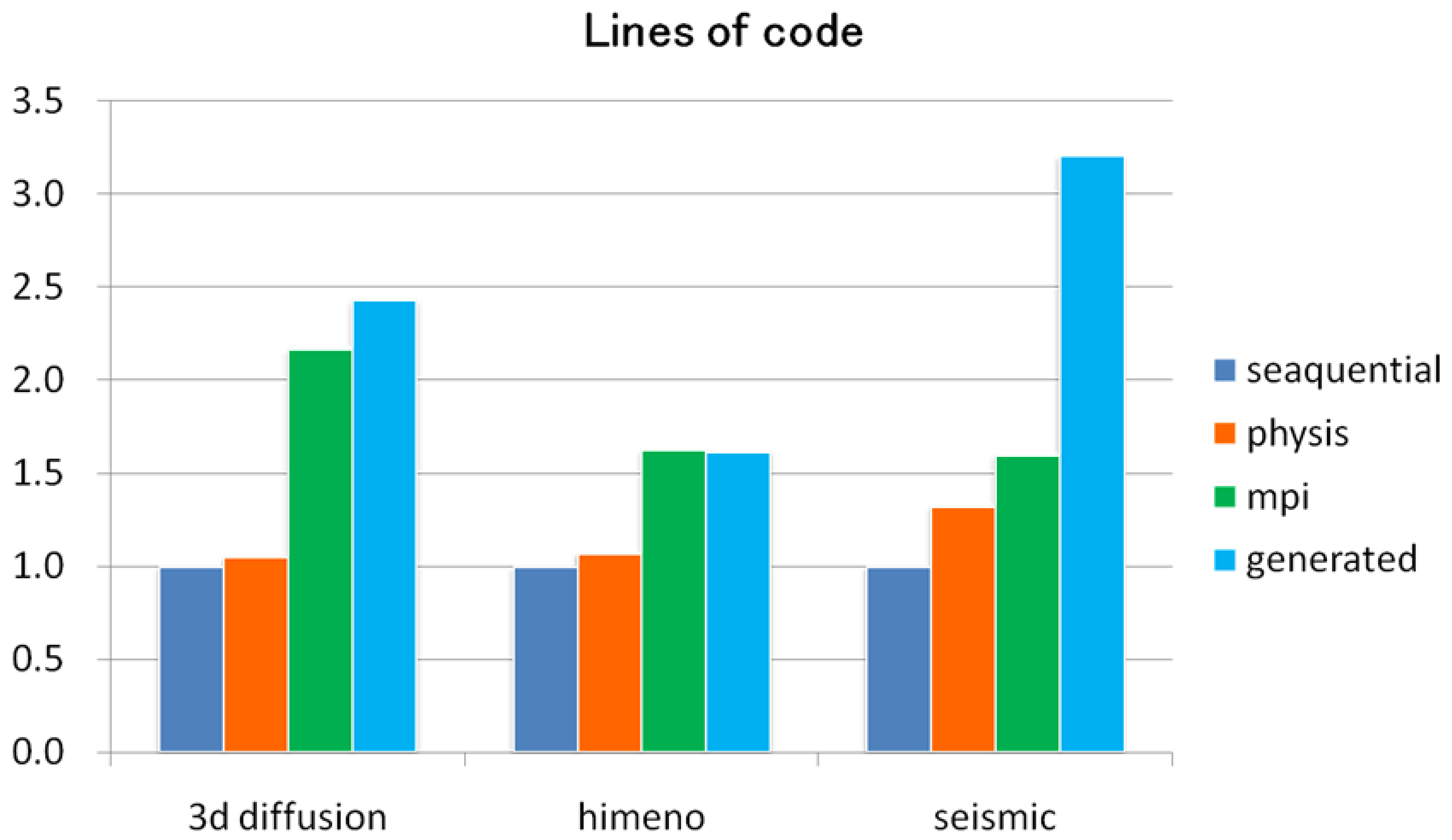


Procedure of boundary exchange

Implementation on GPU clusters

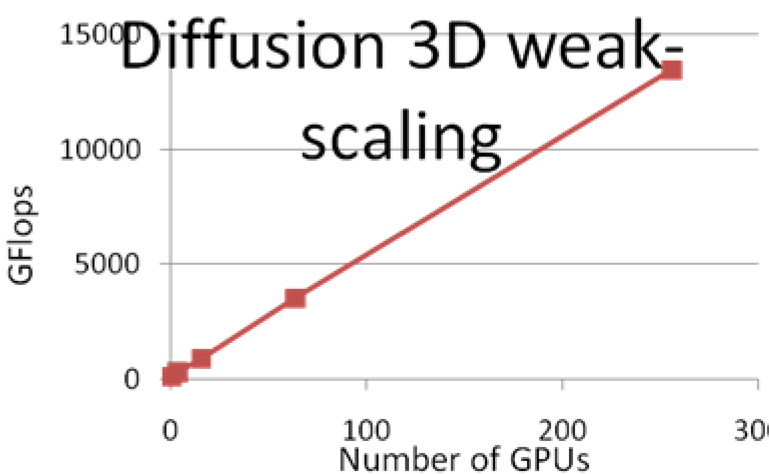


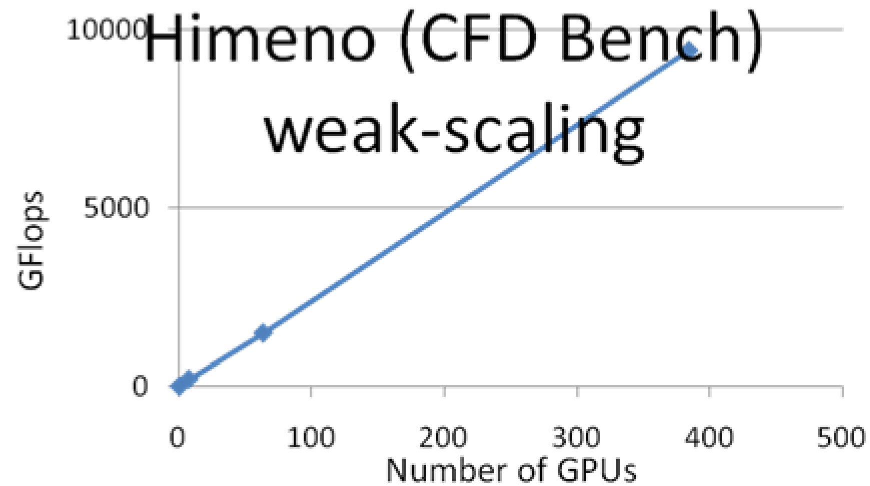
Productivity

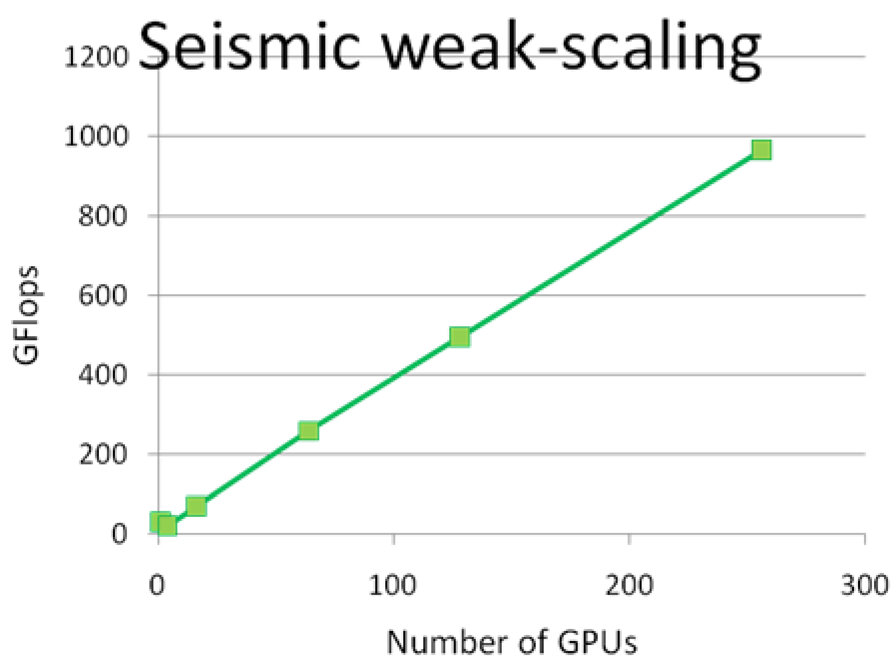


(Original FORTRAN code courtesy of Prof. Takashi Furumura)

Weak Scalability

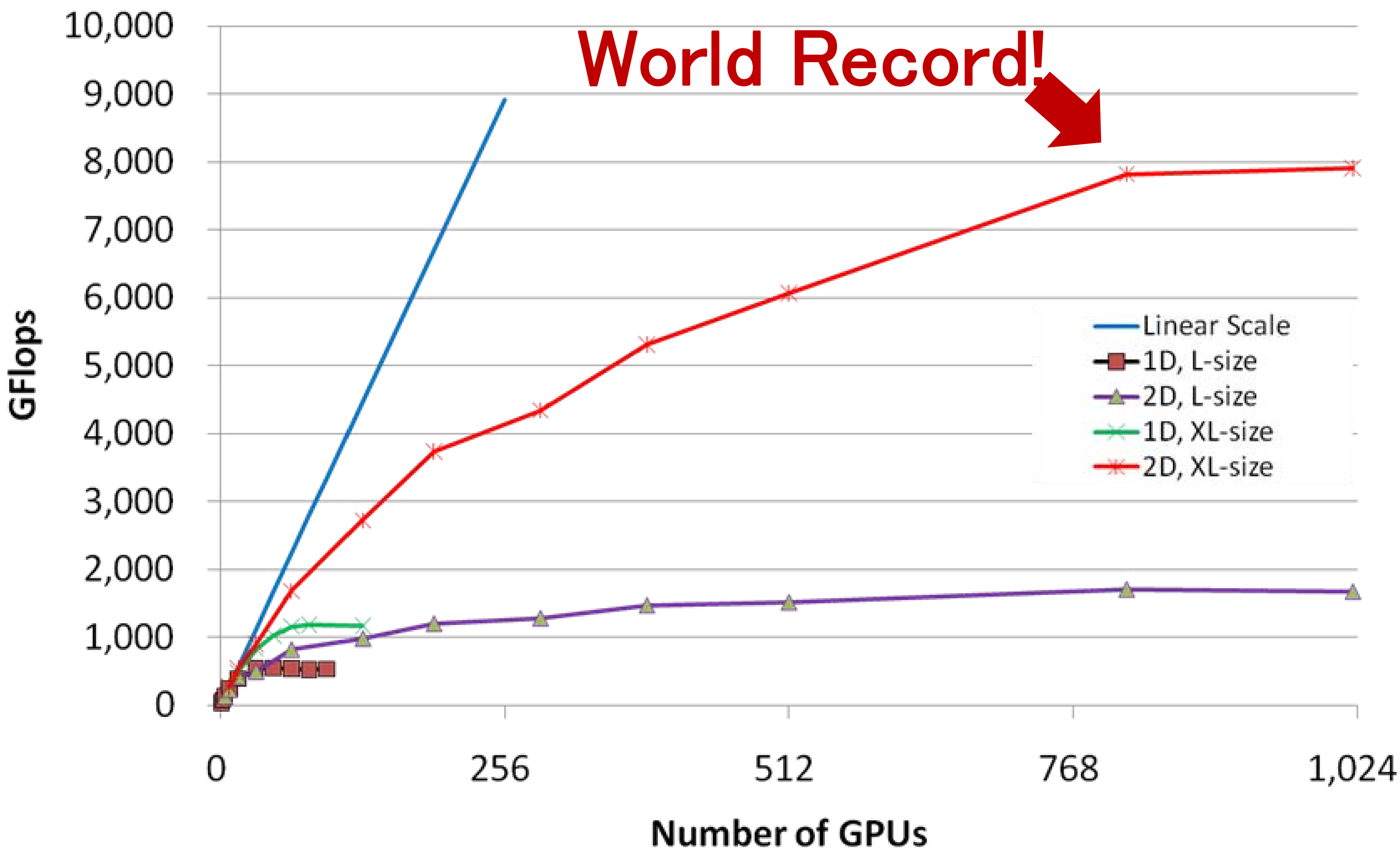






(Original FORTRAN code courtesy of Prof. Takashi Furumura)

Strong scalability – Himeno bench



Important Strategy Towards Exascale

- ***Identify Key Application Areas***
  - Disaster, Environment, Energy, Life/Medical, etc…
- ***Identify Technology Strength (of Japanese HPC)***
  - Device, Archiecture, Software, Applications
- ***Identify the Impediments and do Research***
  - Power, Resiliency, Programming, …
- ***Learn from Existing Systems***
  - K Computer, Tsubame2
- ***Set Performance and other Metric Timelines***
  - 2010 1PetaFlop (done), 2012 10PetaFlops (to be done), 2015 100PetaFlops, 2018 ExaFlops, etc…