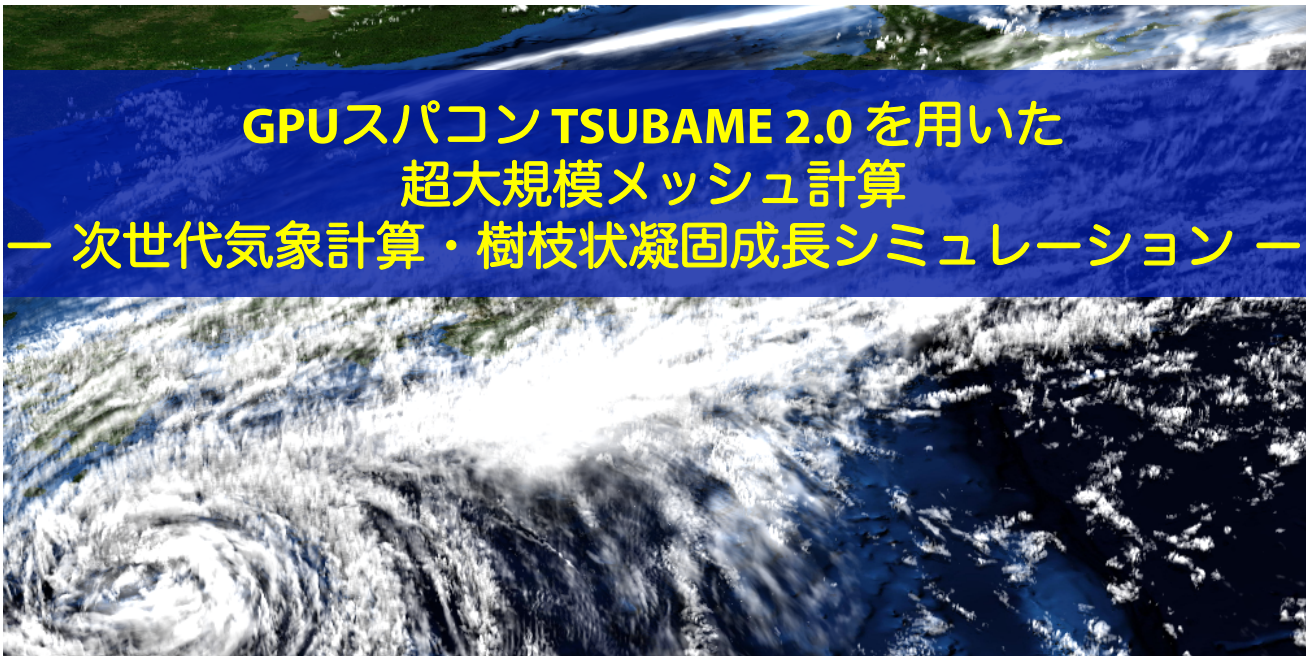


GPUスパコン TSUBAME 2.0 を用いた 超大規模メッシュ計算 ー 次世代気象計算・樹枝状凝固成長シミュレーション ー

下川辺　隆史

東京工業大学
学術国際情報センター

2012/12/3　第18回 ビジュアリゼーションカンファレンス（東京）

目次

- GPUを用いた汎用計算とGPUスパコンTSUBAME2.0
- 大規模気象計算
  - ✓ 気象計算のフルGPU化
  - ✓ オーバラップ手法
  - ✓ TSUBAME2.0 によるマルチGPU計算の実行性能
- フェーズフィールド法を用いた樹枝状凝固成長
  - ✓ GPU-CPU ハイブリッド計算
  - ✓ TSUBAME2.0 によるマルチGPU計算の実行性能
- まとめ

What's GPU ?

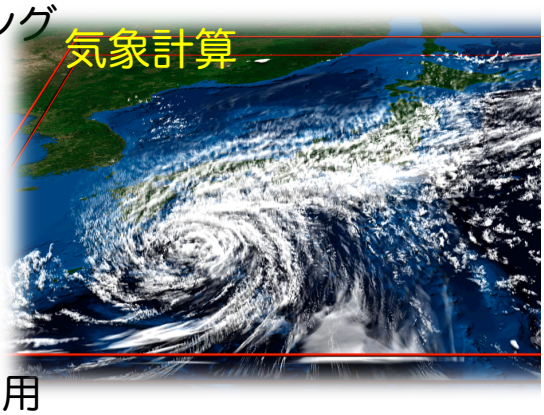
- Graphics Processing Unit
- もともと PC の3D描画専用の装置
- パソコンの部品として量産されてる。= 非常に安価

http://www.nvidia.co.jp

GPGPU

http://gpgpu.org/

- General Purpose computation on GPU
  汎用GPU計算、GPUコンピューティング
  - ✓ 数値流体力学 (CFD)
  - ✓ N体問題
  - ✓ 気象計算
  - ✓ 高速フーリエ変換 (FFT)
  - ✓ 分子動力学



- GPGPU向けプログラミング言語の利用
  →グラフィックス向けAPIを使わず、科学計算が可能
  - ✓ **CUDA** (NVIDIA GPUsに特化したC/C++拡張言語)
  - ✓ OpenCL (複数社のCPUとGPUに対応した汎用言語)
  - ✓ OpenACC (高レベルのディレクティブ・ベースのフレームワーク)

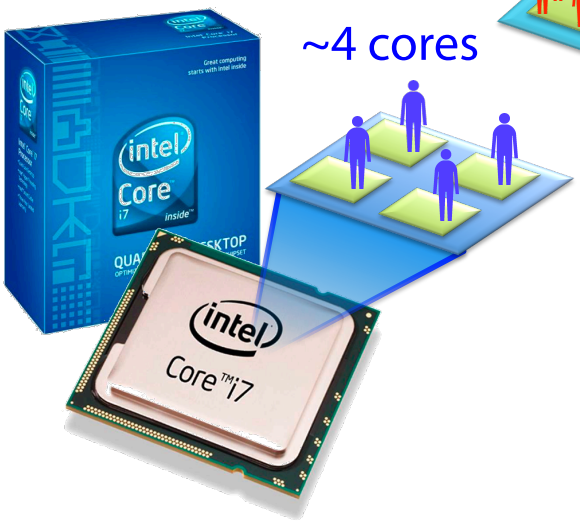
CPU and GPU

CPU

~4 cores

~240 cores

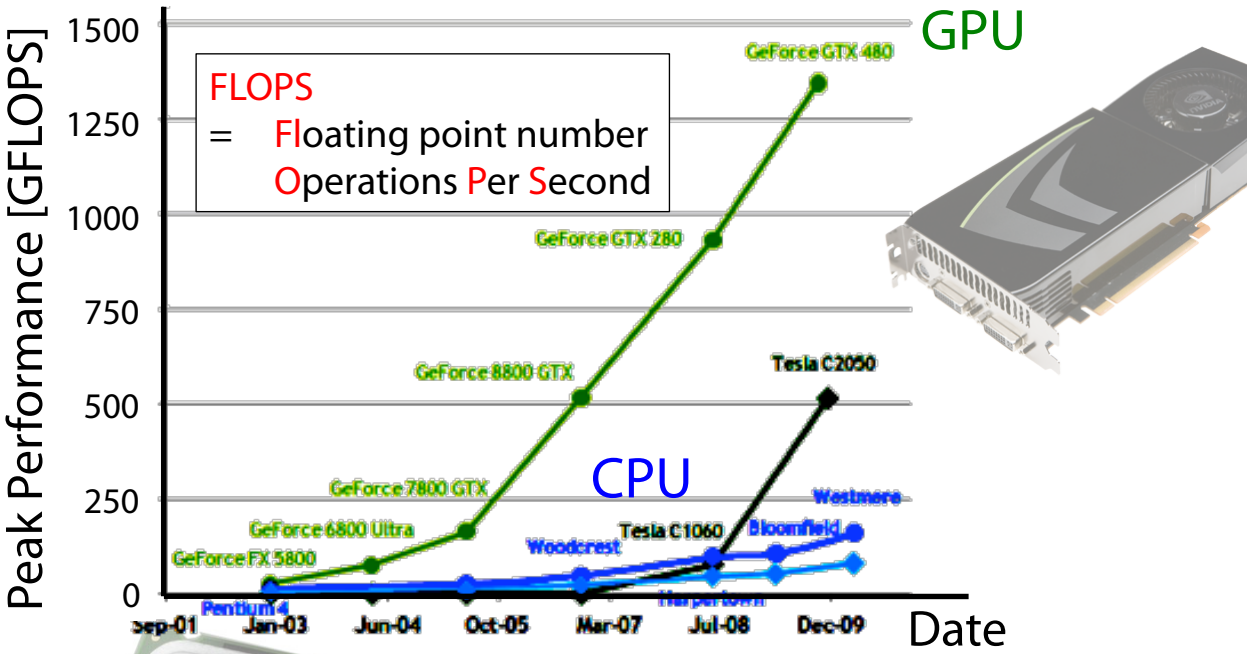
GPU

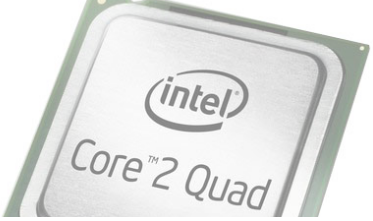


異なるデータに対して異なる命令を実行

複数のデータに対して同じ命令を同時に実行（SPMD, SIMD）

CPU and GPUの演算性能



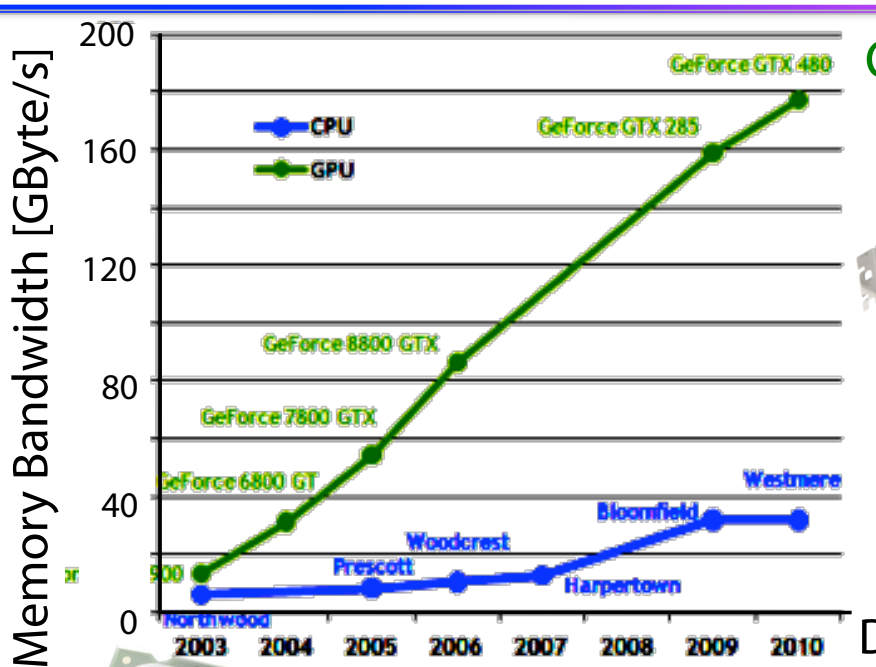



```
d = a + b * c
```

2 FLOP

http://developer.nvidia.com/object/cuda_3_2_downloads.html

CPU and GPUのメモリバンド幅



d = a + b * c

1 write (4 Byte), 3 reads (12 Byte)

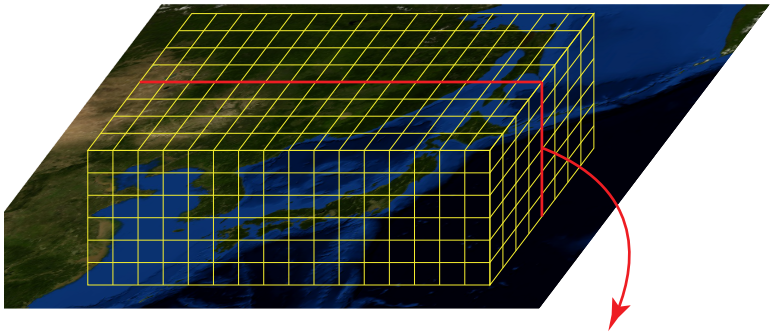
格子計算のアプリケーションでは多くの場合、演算性能よりもメモリバンド幅が重要

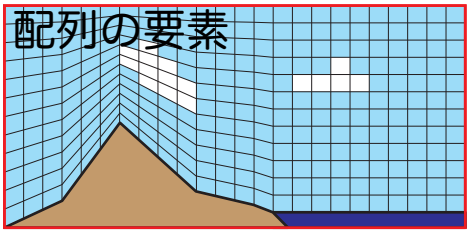
GPUによるメッシュ計算

- メッシュ計算：計算領域がメッシュに分割
- メッシュ上で偏微分方程式を離散化し解く

GPUによる気象計算

3次元計算領域





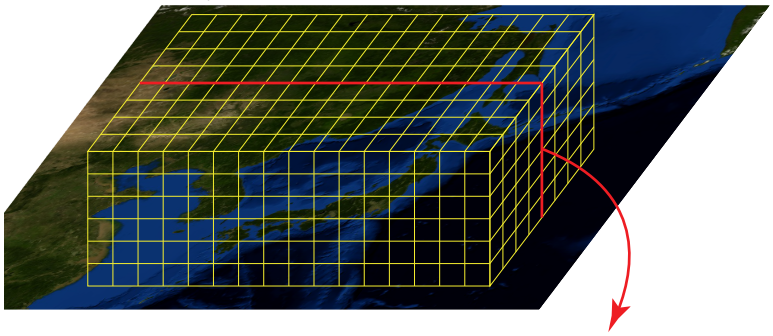
GPU

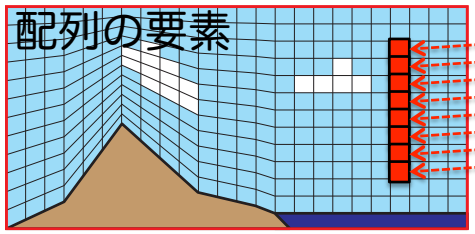
GPUによるメッシュ計算

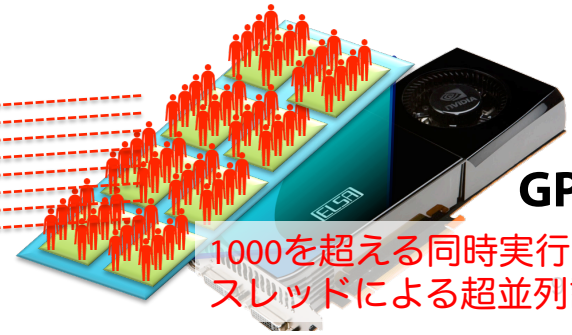
- メッシュ計算：計算領域がメッシュに分割
- メッシュ上で偏微分方程式を離散化し解く

GPUによる気象計算

3次元計算領域







GPU

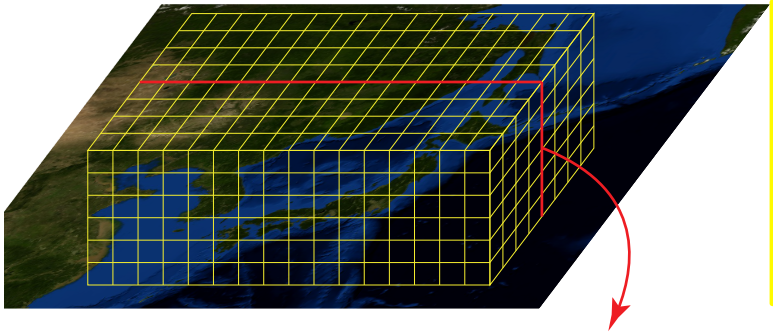
1000を超える同時実行
スレッドによる超並列計算

GPUによるメッシュ計算

- メッシュ計算：計算領域がメッシュに分割
- メッシュ上で偏微分方程式を離散化し解く

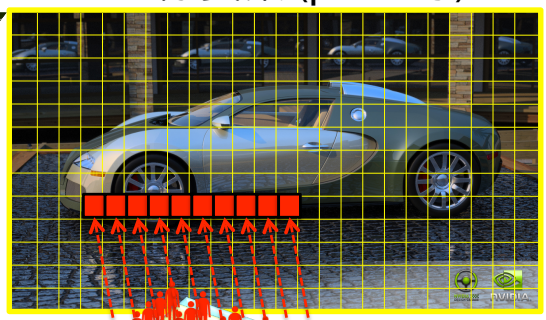
GPUによる気象計算

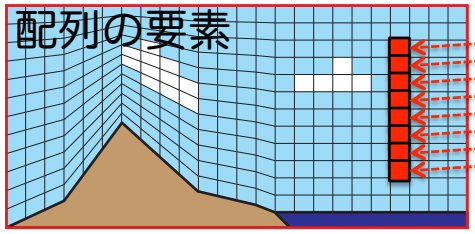
3次元計算領域

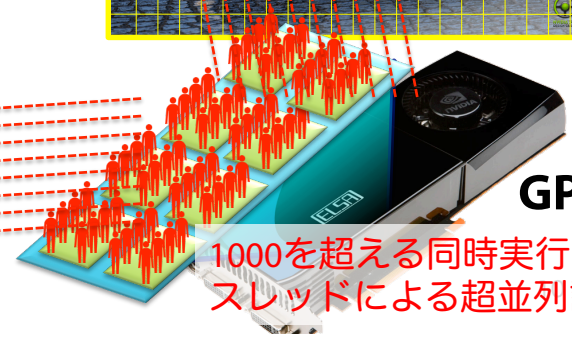


GPUによる画像処理

画像要素 (pixels等)







GPU

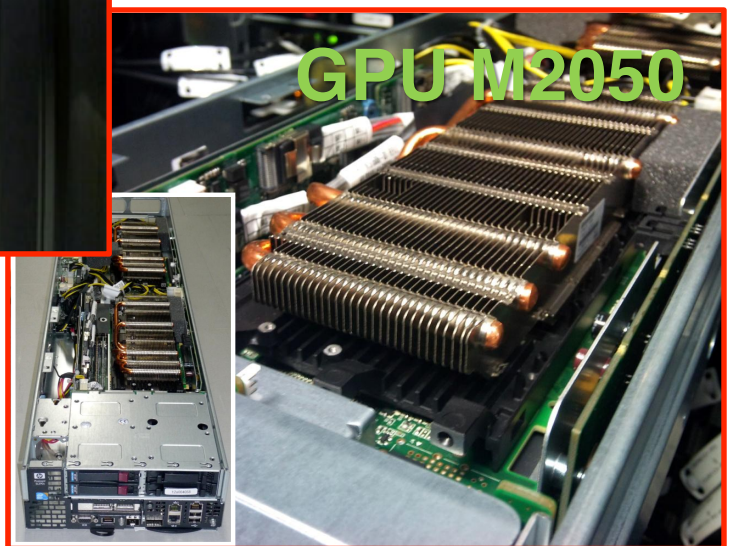
1000を超える同時実行
スレッドによる超並列計算

Tokyo Tech **TSUBAME 2.0** Supercomputer

- TSUBAME 2.0 started operating in Nov. 2010

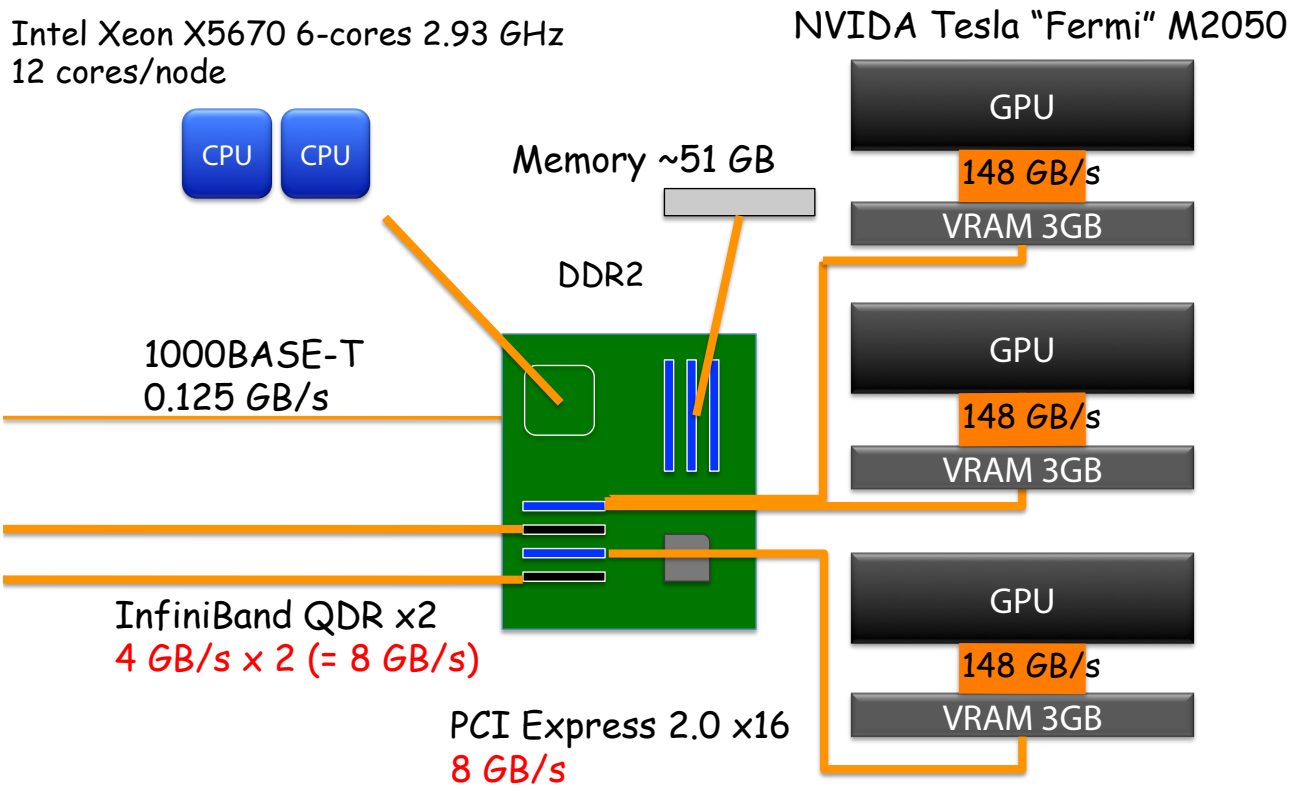
TSUBAME 2.0

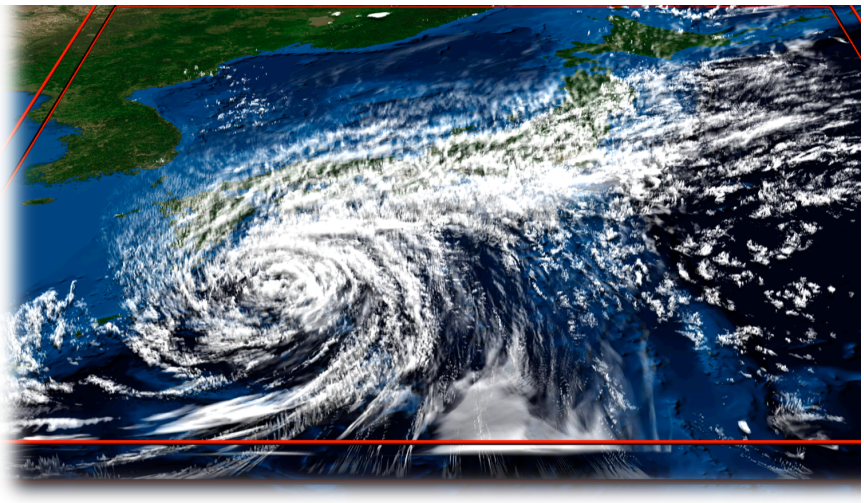
NVIDIA Tesla "Fermi" M2050
515 GFlops (DP), 1030 GFlops (SP)
Memory BW 148 GB/s
4224 GPUs on TSUBAME 2.0



Node of TSUBAME 2.0

TSUBAME 2.0 Node detail





Numerical Weather Prediction
ASUCA

高解像度の気象計算

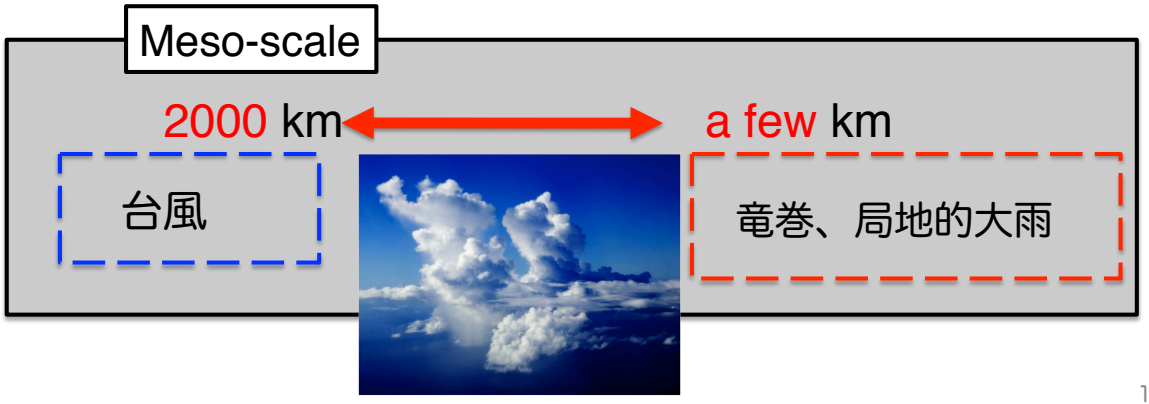
- ゲリラ豪雨
  - ✓ 突発的で局地的な豪雨（~km - ~10km）
  - ✓ 参考：梅雨前線などによる集中豪雨（~100km）

  → 現業の5km 以下の格子で計算することが必須

http://www.nikkeibp.co.jp/sj/2/column/z/33/
http://trendy.nikkeibp.co.jp/lc/photorepo/080916_photo/

気象シミュレーションコードASUCA

- ASUCA Production Code
  - ✓ 気象庁によって開発が進められる次世代高解像度気象シミュレーションコード
  - ✓ 完全圧縮非静力学方程式系
    フラックス形式、一般座標系
- なぜ GPU コンピューティング？
  - ✓ 速い, 安い, 低消費電力



http://wallpaper.free-photograph.net/

気象計算におけるGPUによる高速化

- The Weather Research and Forecast (WRF) の GPU 利用
  = 計算律速の物理モジュールをGPUで部分的に加速
  - 雲微物理モジュール WSM5 (WRF Single Moment 5-tracer) *
    水物質(水蒸気、 雲水、雨水、雲氷、雪)の混合比を予報
    WRF の 1 % のコード、25%の実行時間
    ⇒ 20 x 高速化
    (アプリケーション全体で 1.2-1.3倍の高速化)
  - 領域化学輸送モデル WRF-Chem**
    特定の領域における汚染気体の輸送や化学反応
    ⇒ 8.5 x 高速化

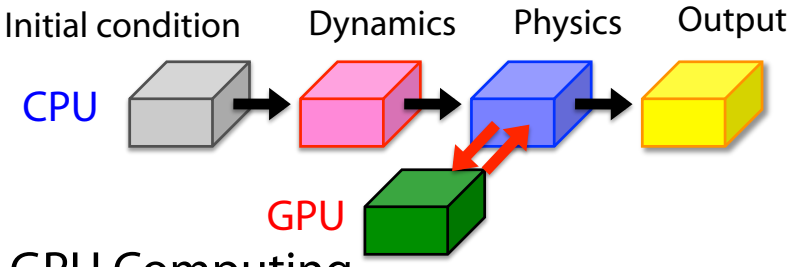
* Michalakes, J. and M. Vachharajani: GPU Acceleration of Numerical Weather Prediction. *Parallel Processing Letters Vol. 18 No. 4. World Scientific. Dec. 2008. pp. 531—548*

**John C. Linford, John Michalakes, Manish Vachharijani, and Adrian Sandu. Multi-core acceleration of chemical kinetics for simulation and prediction, *proceedings of the 2009 ACM/IEEE conference on supercomputing (SC'09), ACM, 2009.*

WRFとASUCAの高速化のアプローチの違い

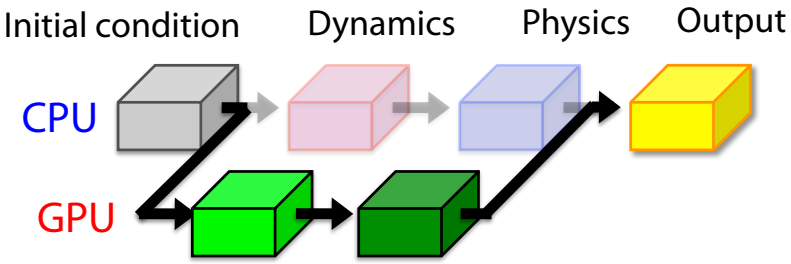
✓ WRF GPU Acceleration

Accelerator Approach



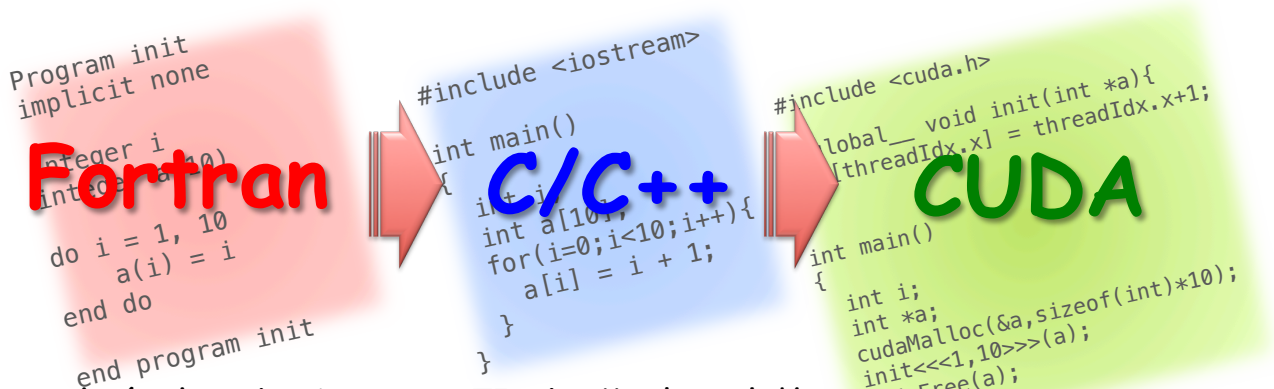
✓ ASUCA GPU Computing

Full GPU Approach ➡ 数十倍の高速化の実現



ASUCA: Fortran から CUDA へ

- フル GPU アプリケーション
- ゼロから書き換え



✓気象庁における オリジナルコード

✓配列の順序の交換

✓GPU コード

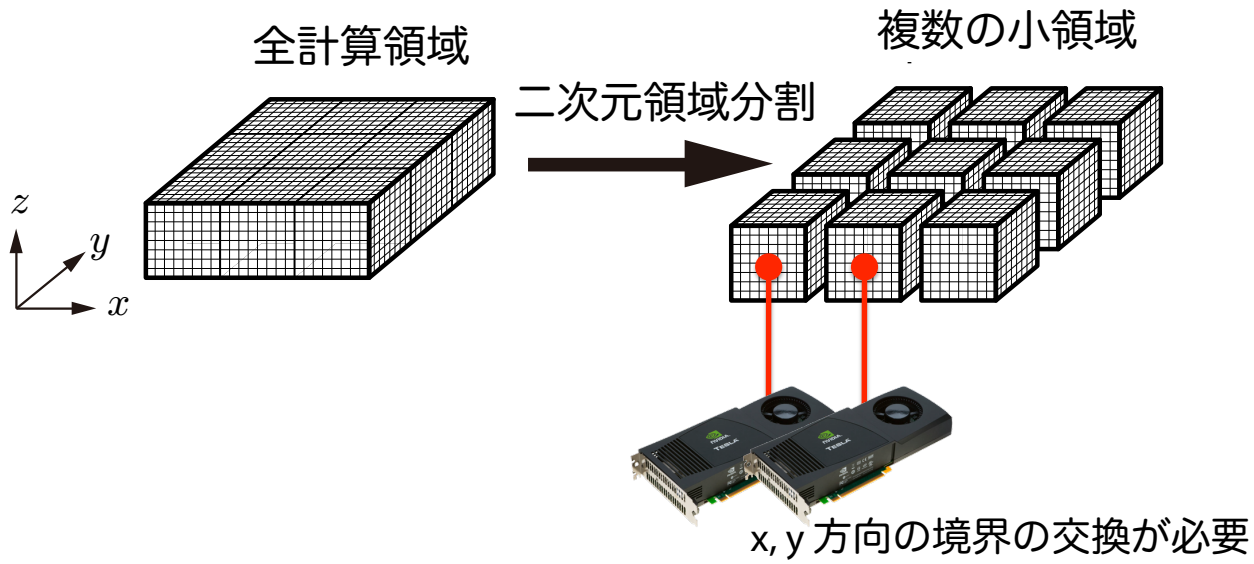
[3次元配列の要素順序]

z,x,y (k,i,j)-ordering	*x,z,y* (i,k,j)-ordering	*x,z,y* (i,k,j)-ordering

➡ GPUコードでのメモリアクセスパフォーマンスを向上

マルチGPU計算：領域分割

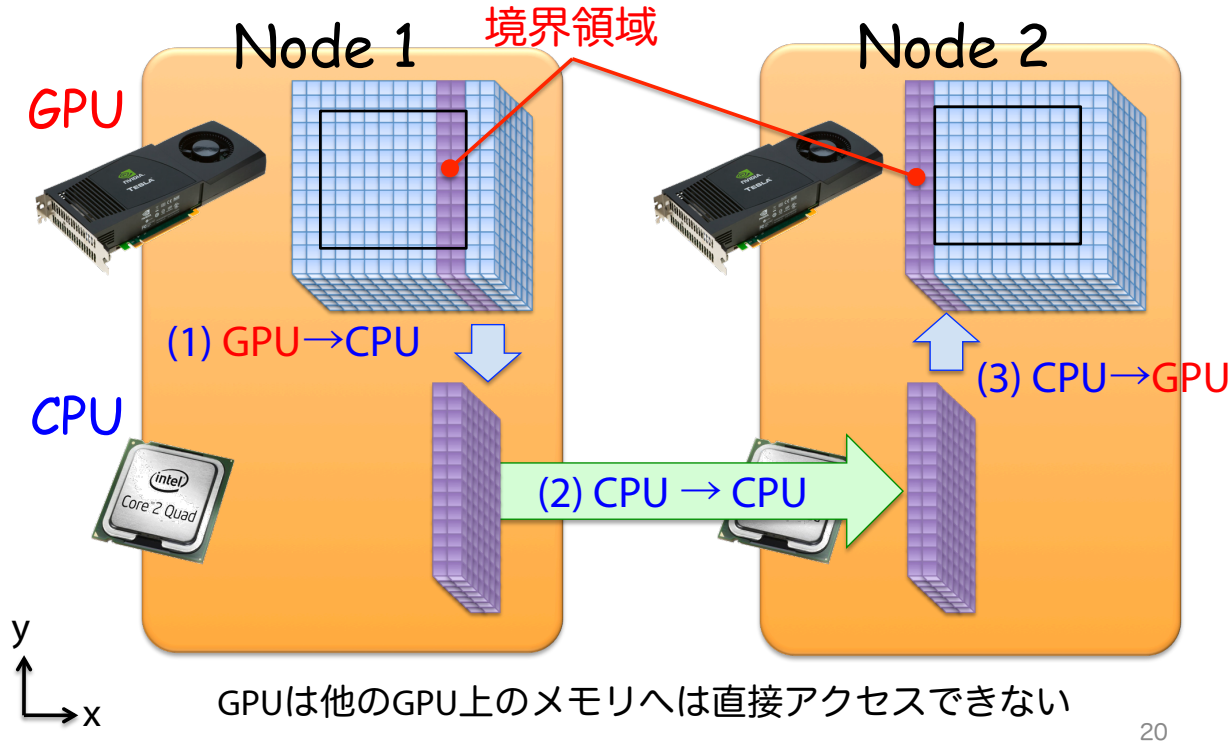
- 二次元領域分割 （x, y 方向）
  - ✓ 水平方向の格子数 ~ 1000 > 鉛直方向の格子数 ~ 100
  - ✓ 256 x 208 x 48 Sub-domain / GPU



x, y 方向の境界の交換が必要

マルチGPU計算：境界領域のデータ交換

- MPIを用いたGPUとCPUによるデータ交換

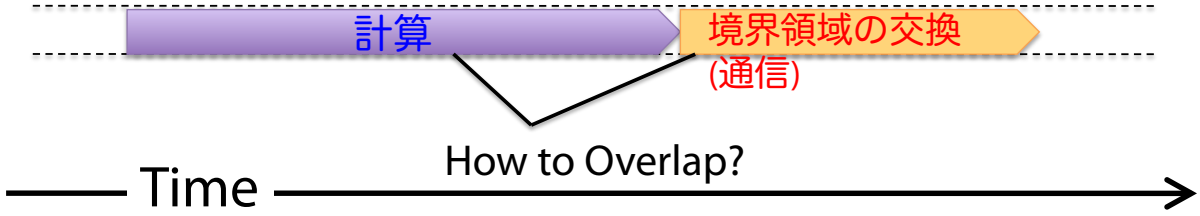


GPUは他のGPU上のメモリへは直接アクセスできない

マルチGPU計算における最適化手法

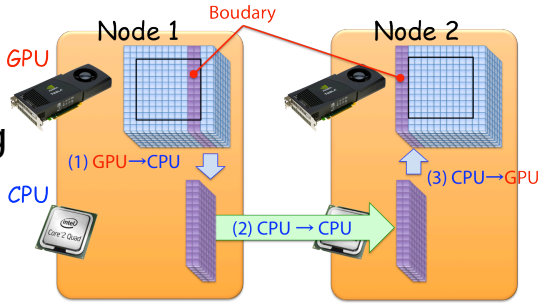
■ 計算と通信のオーバラップ

ある変数に対する計算の典型例



■ 最適化手法

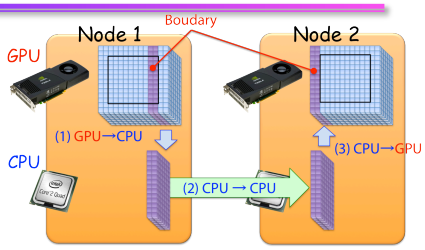
- ✓ (1) Inter-variable Overlapping
- ✓ (2) Kernel division Overlapping
- ✓ (3) Fused Kernel Overlapping

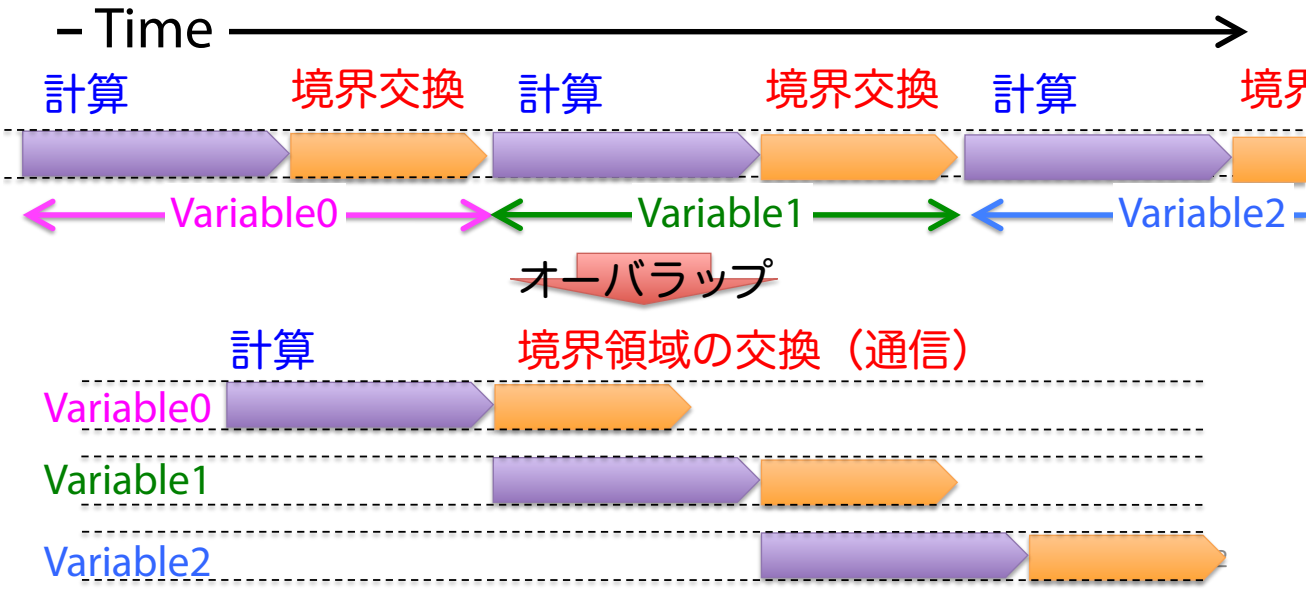


最適化手法 1 : Inter-variable Overlapping

■ 最適化手法 1

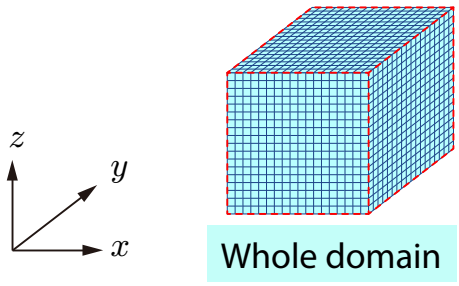
- ✓ 変数間の非依存関係の利用
- ✓ 水物質の移流計算へ適用





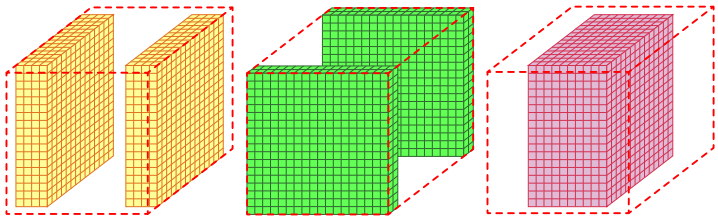
最適化手法２: Kernel division Overlapping

Non-Overlapping (Single Kernel)

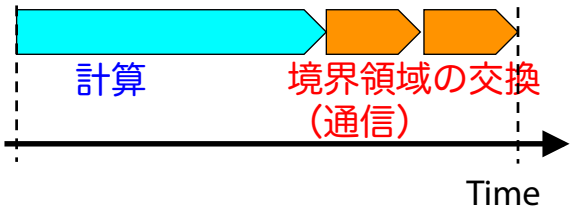


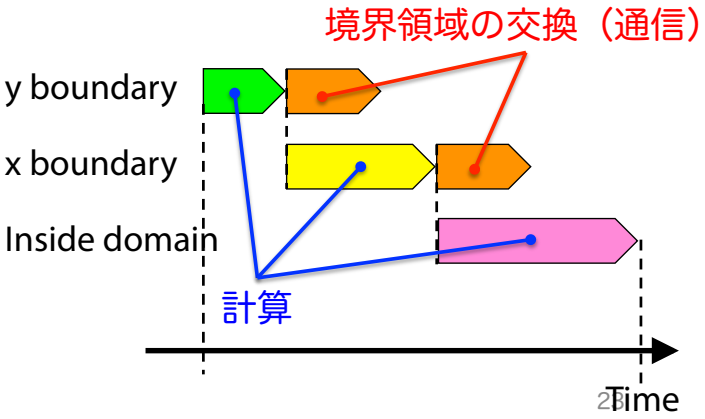
Whole domain

Overlapping (Three divided kernels)



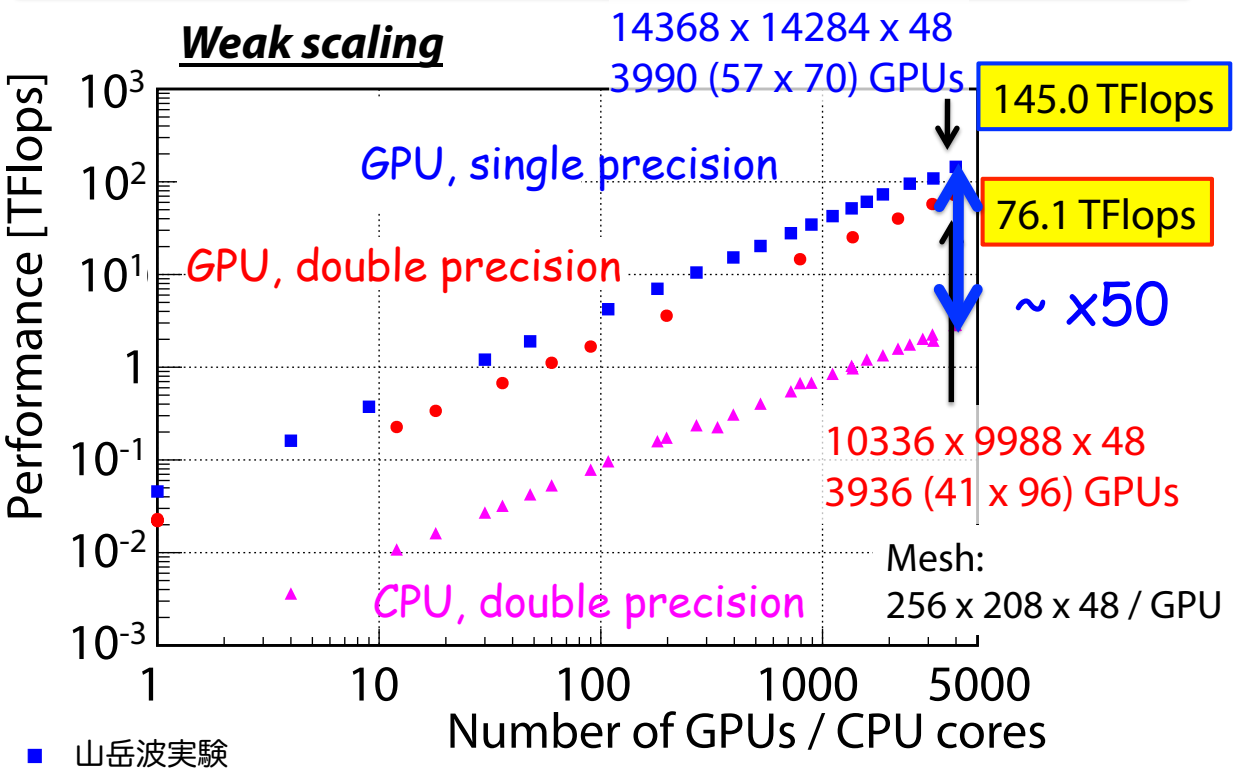
x boundary　y boundary　Inside domain





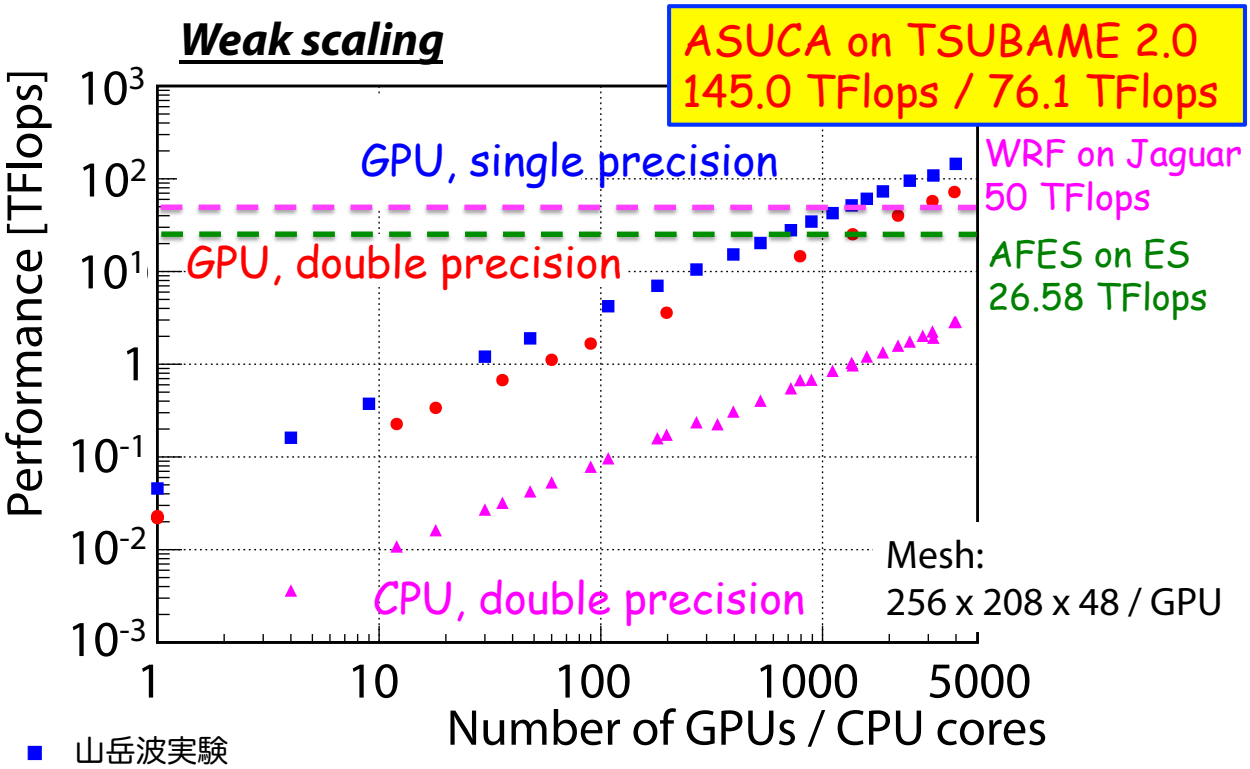
✓ひとつの変数内でのデータの非依存関係を利用

TSUBAME 2.0によるASUCAの計算性能



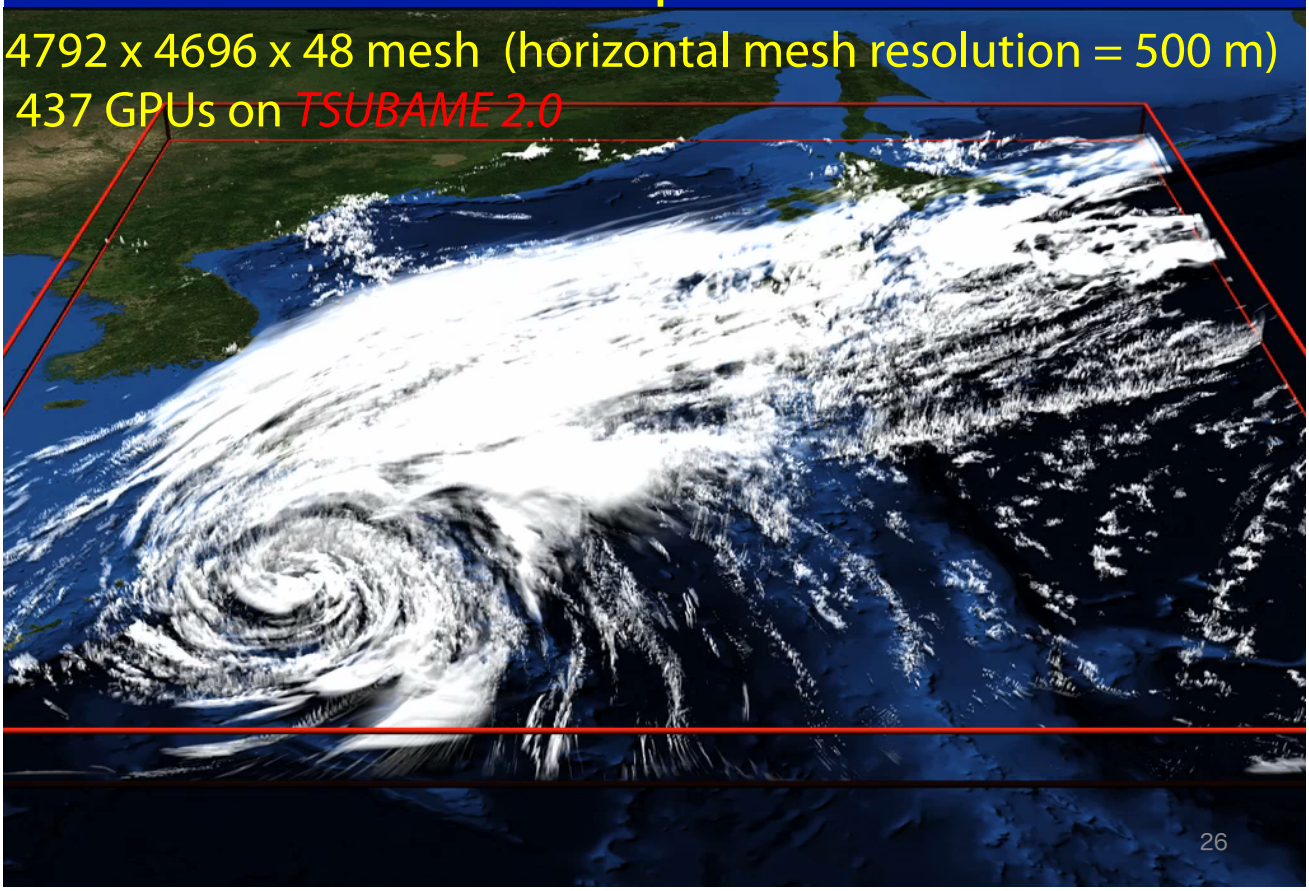
- 山岳波実験
- NVIDIA Tesla M2050 card / Intel Xeon X5670 2.93 GHz on TSUBAME 2.0
- Weak Scaling, each GPU handles 256 x 208 x 48(SP) , 256 x 108 x 48 (DP)

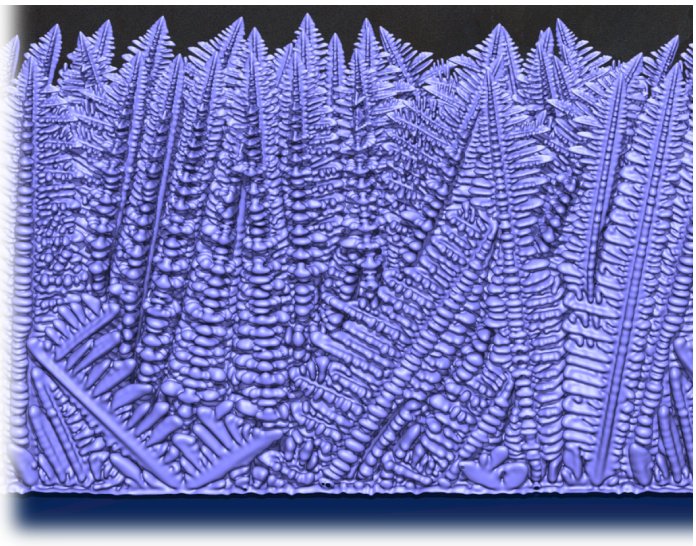
TSUBAME 2.0によるASUCAの計算性能



- 山岳波実験
- NVIDIA Tesla M2050 card / Intel Xeon X5670 2.93 GHz on TSUBAME 2.0
- Weak Scaling, each GPU handles 256 x 208 x 48(SP) , 256 x 108 x 48 (DP)

Real case of ASUCA operation



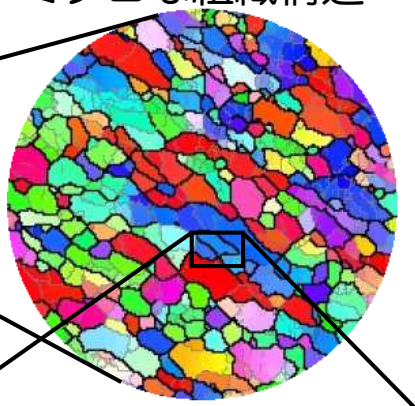


Phase-field Simulation for Dendritic Solidification

軽量・高強度の金属材料の開発

金属構造物

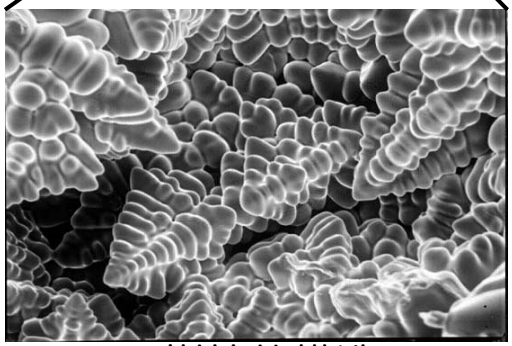
ミクロな組織構造



低炭素消費社会の実現

自動車や電車を軽量化し、
燃費の向上

ミクロな構造を制御することにより、
軽量で機械的強度の高い材料の開発

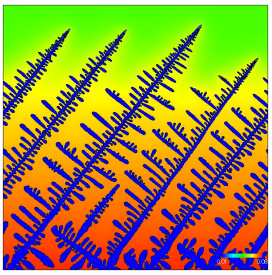


樹枝状構造

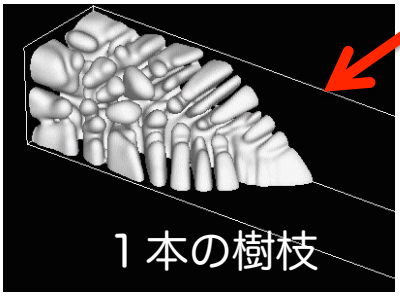
材料科学のペタスケールのシミュレーション

先行研究

2次元計算

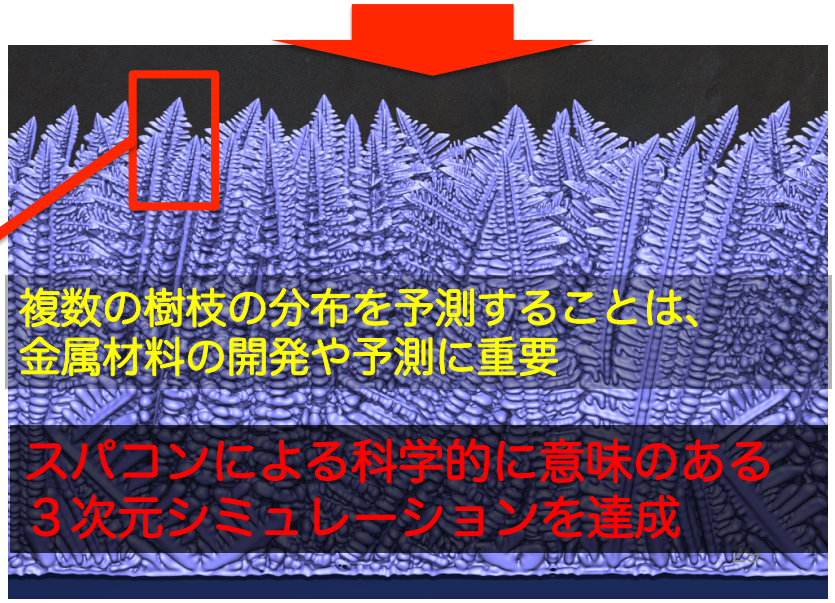


単純形状の3次元計算



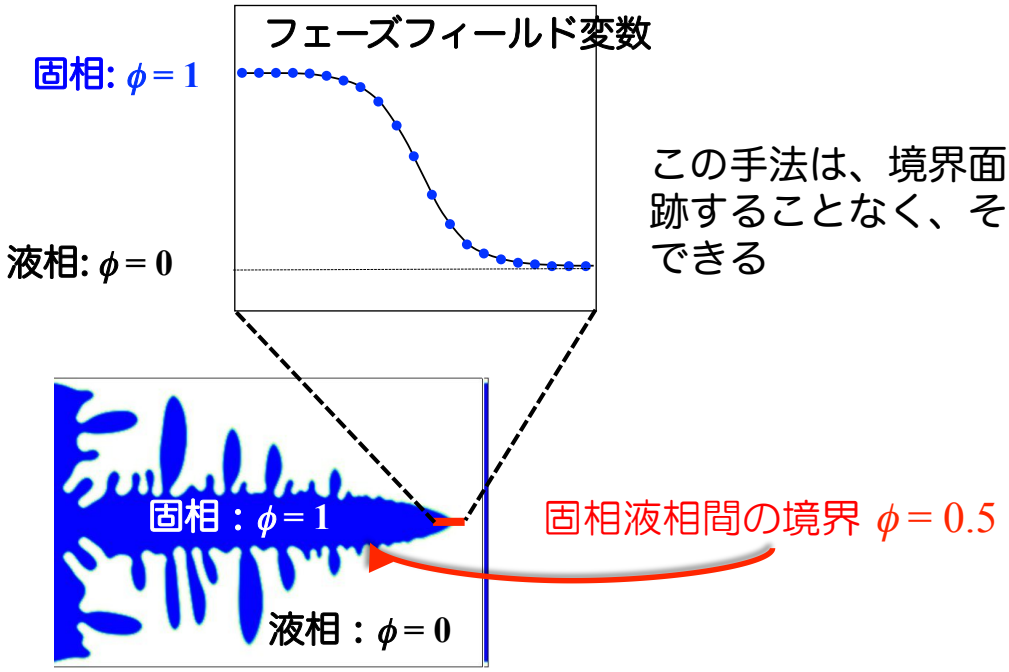
ペタスケールのフェーズフィールド計算

- ✓ GPUスーパーコンピュータ（スパコン）
- ✓ ペタスケール計算に向けた最適化手法
  - ➢ GPU-CPU を用いたハイブリッド計算



フェーズフィールドモデル

- フェーズフィールドモデルは非平衡統計力学から導出
- 複雑なミクロな構造をシミュレーション



この手法は、境界面を明示的に追跡することなく、その場所を記述できる

フェーズフィールドモデル

- Phase-field ϕ の時間発展（Allen-Cahn方程式）

$$\frac{\partial \phi}{\partial t} = M_\phi \left[\nabla \cdot (a^2 \nabla \phi) + \frac{\partial}{\partial x}\left(a \frac{\partial a}{\partial \phi_x} |\nabla \phi|^2 \right) + \frac{\partial}{\partial y}\left(a \frac{\partial a}{\partial \phi_y} |\nabla \phi|^2 \right) \right.$$

拡散項　　異方性

$$\left. + \frac{\partial}{\partial z}\left(a \frac{\partial a}{\partial \phi_z} |\nabla \phi|^2 \right) - \Delta S \Delta T \frac{dp(\phi)}{d\phi} - W \frac{dq(\phi)}{d\phi} \right]$$

化学的駆動力　　相変化のエネルギー

- 濃度 c の時間発展

$$\frac{\partial c}{\partial t} = \nabla \cdot [D_S \phi \nabla c_S + D_L (1-\phi) \nabla c_L]$$

$$c_S = \frac{kc}{1-\phi+k\phi},\ c_L = \frac{c}{1-\phi+k\phi},\quad k = c_S / c_L$$

M_ϕ モビリティー
a 異方性
ΔS 融解エントロピー
ΔT 過冷却

D_S 固相の拡散係数
D_L 液相の拡散係数

フェーズフィールドのデータアクセス

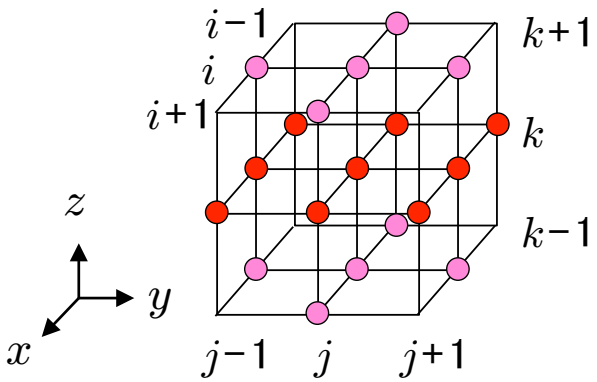
- 空間を格子に分割して計算（気象計算と同じ）
- 1タイムステップ更新するために、

Phase-field変数は19点、濃度変数は7点

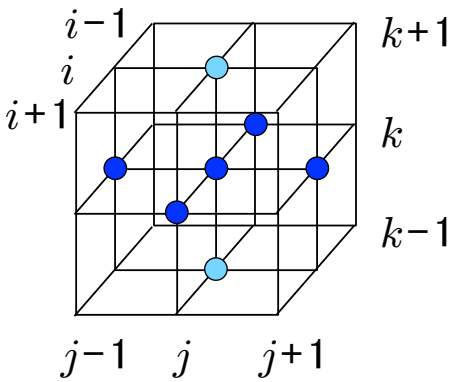
のデータ読み込みが必要。計算量も比較的大。

（気象計算よりもデータアクセスと計算量が多い）

Phase field

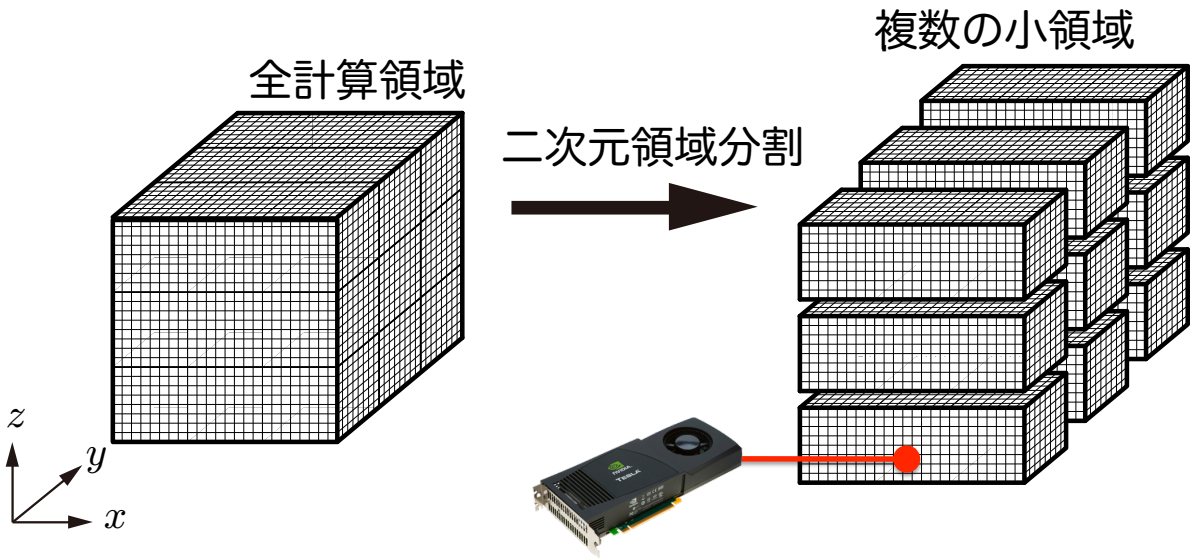


Concentration

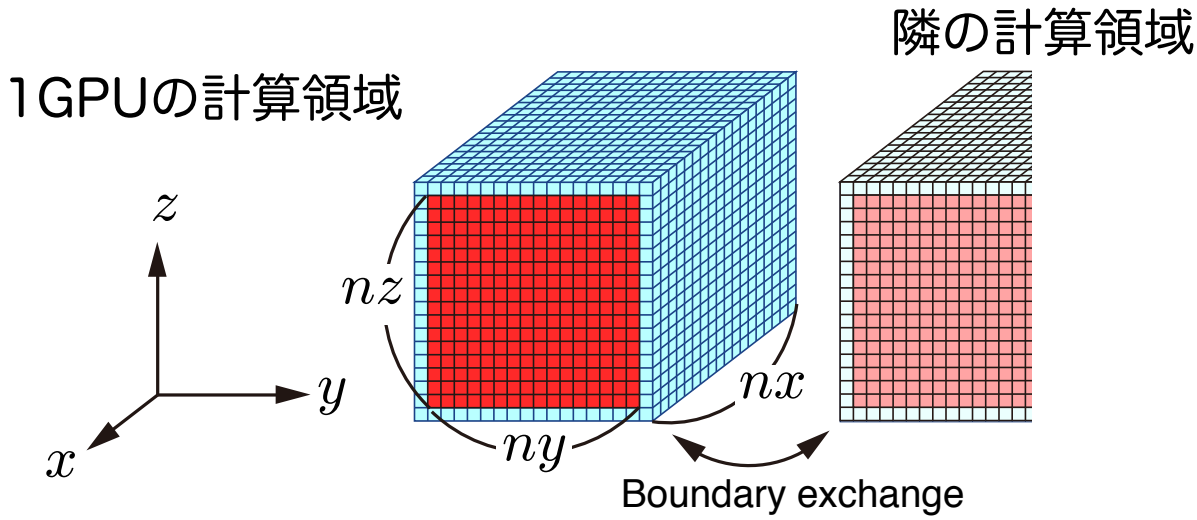


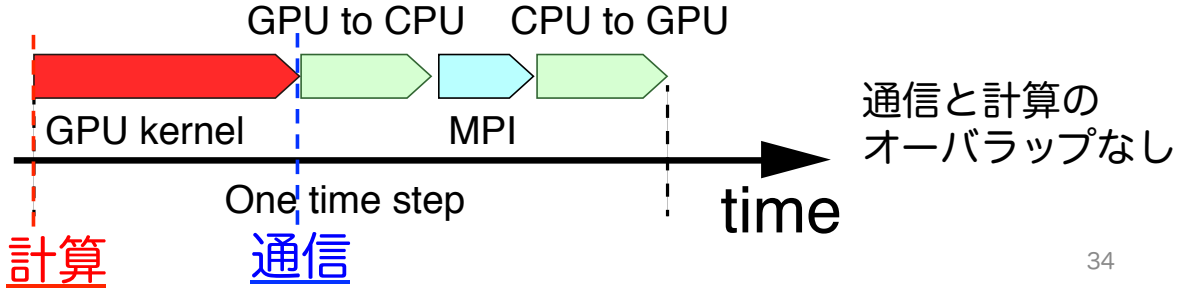
マルチGPU計算：領域分割

- 二次元領域分割 （y, z 方向）
  - ✓ 三次元分割はデータアクセスパターンが比較的複雑
    性能低下を起こしやすい

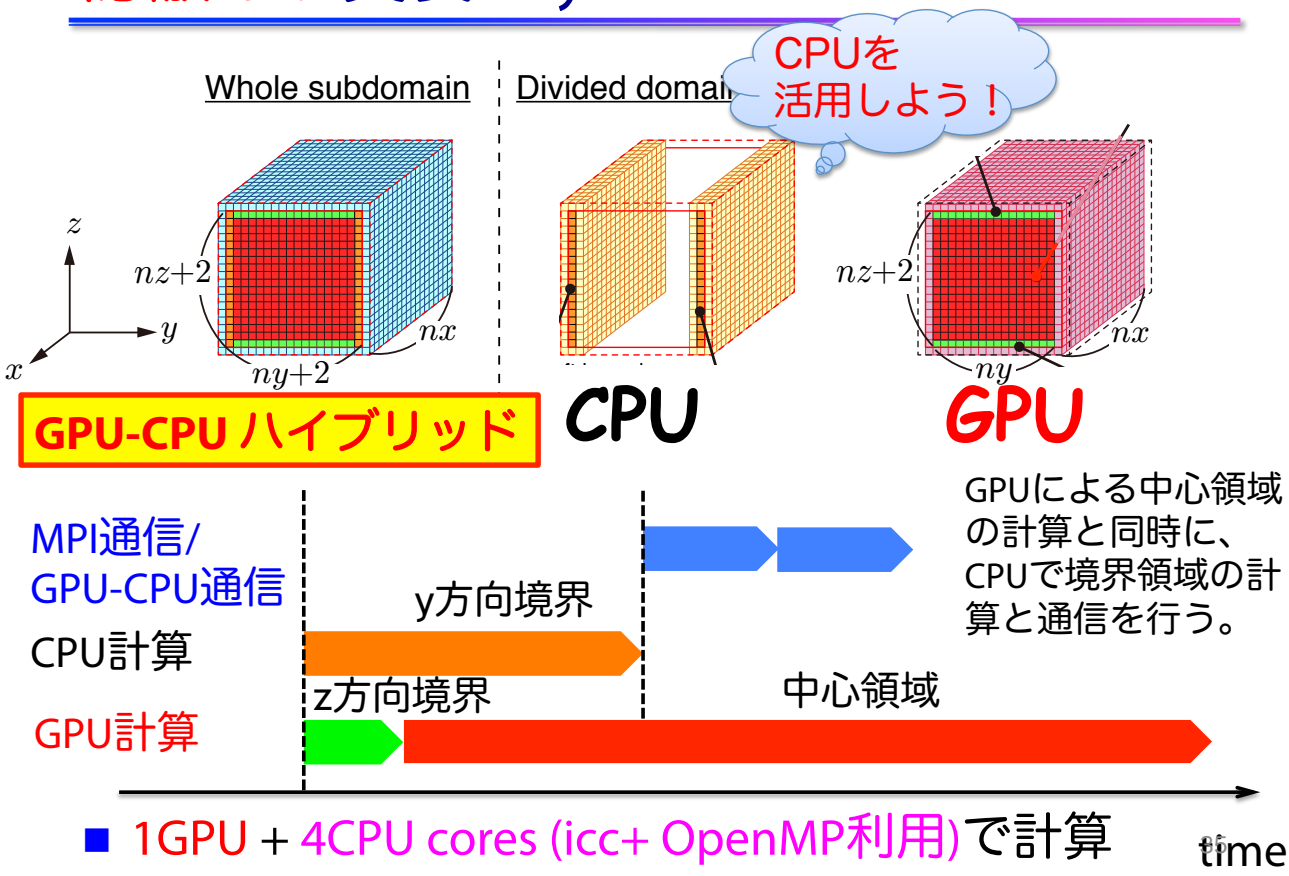


隠蔽なし実装 : GPU-only method



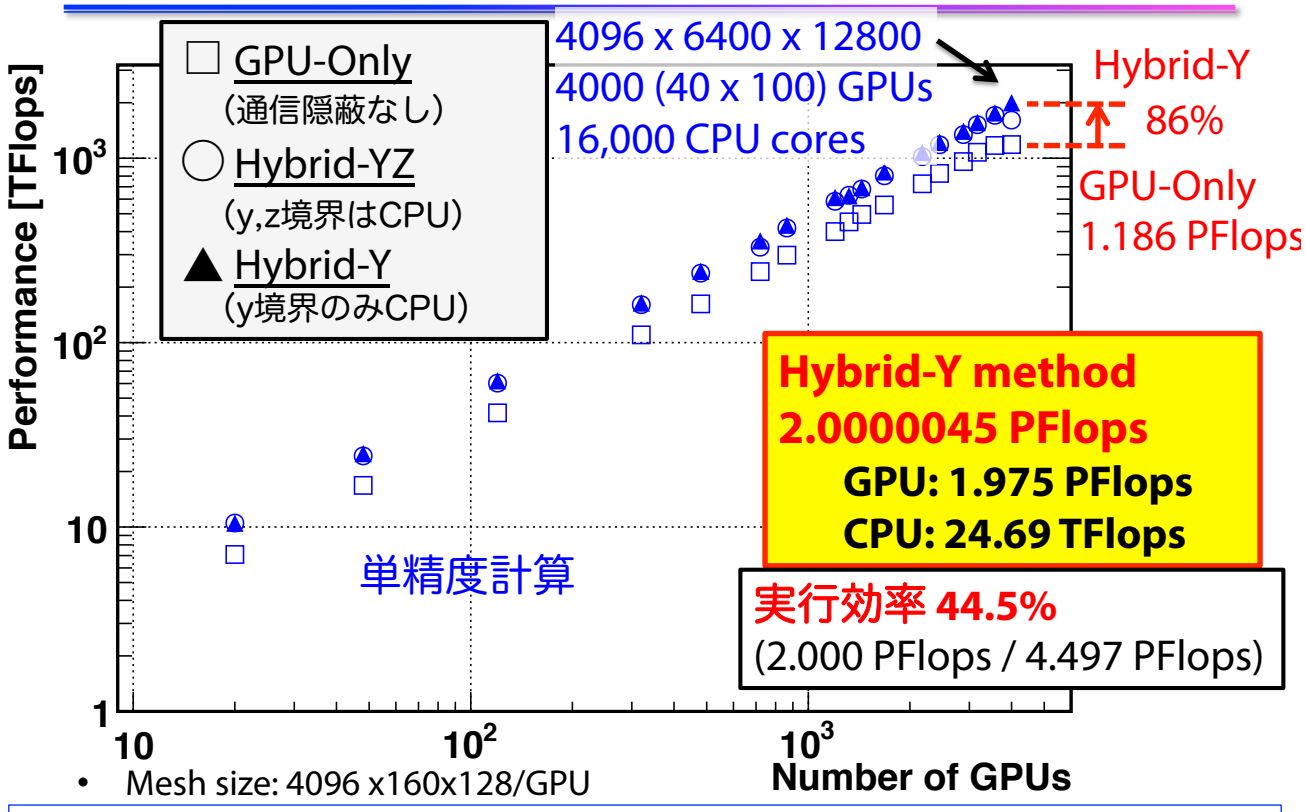


隠蔽あり実装 : Hybrid-Y method



■ 1GPU + 4CPU cores (icc+ OpenMP利用)で計算

TSUBAME 2.0での計算性能（Weak scaling）

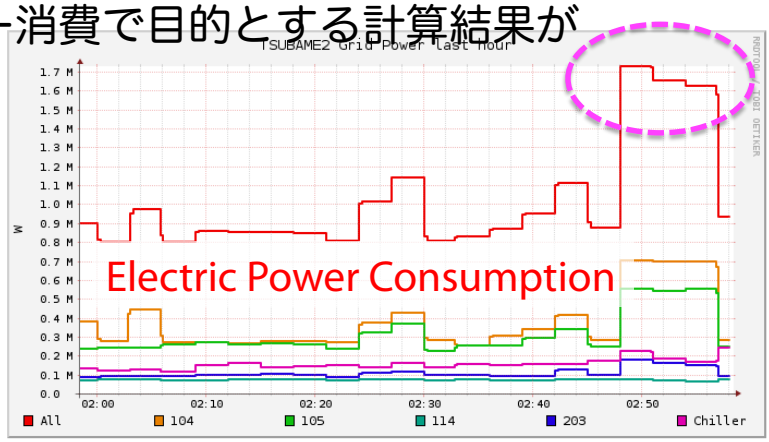


- Mesh size: 4096 x160x128/GPU

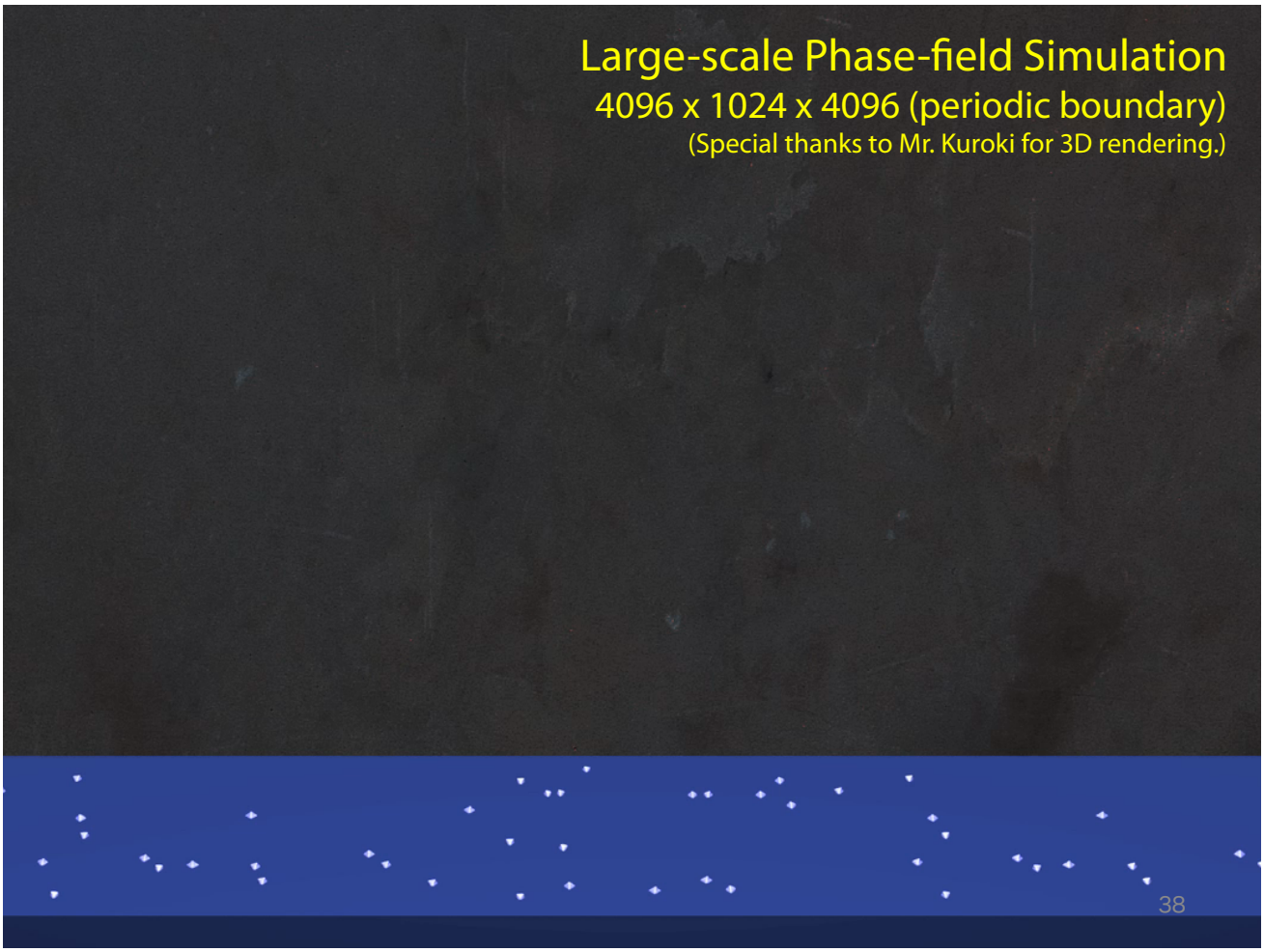
Weak Scaling: GPUあたりの問題サイズ一定で全体サイズを変える

グリーンコンピューティング

- TSUBAME 2.0 では消費電力を詳細にモニタリングしている。
- フェーズフィールド計算 (実アプリケーション)
  - ✓2.000 PFlops (単精度計算)
  - ✓実行効率: **ピーク性能の44.5%**　　2PFlopsシミュレーショ
  - ✓**低エネルギー消費: 1468 MFlops/W**　　**~1.36 MW**
  - → 少ないエネルギー消費で目的とする計算結果が得られた。

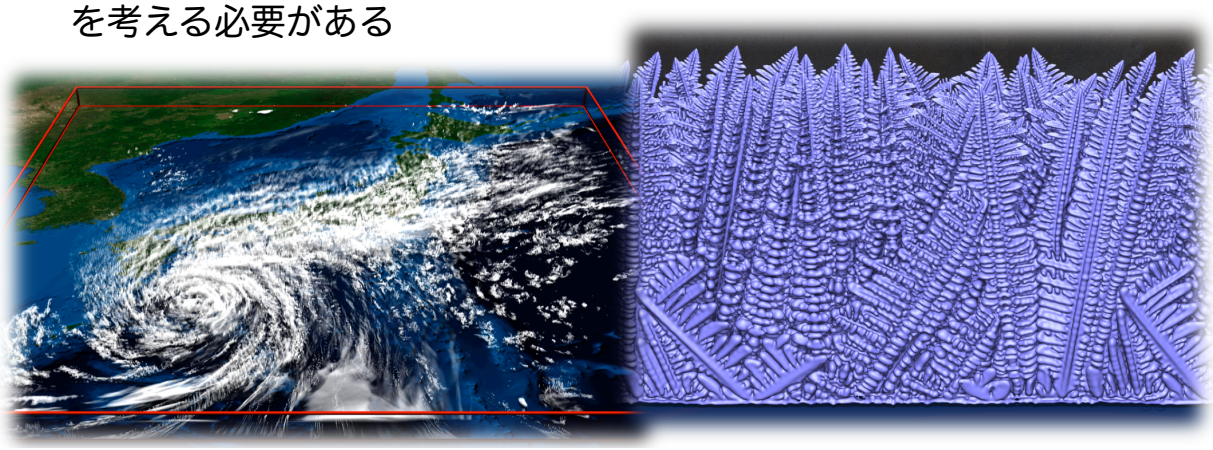


参考
Linpack ベンチマーク
- ✓ 1.192 PFlops (倍精度計算)
- ✓ 実行効率52.1%
- ✓ 827.8 MFlops/W



スパコンによる計算結果の可視化

- 計算全体を把握するため、全領域の可視化が有効
- スパコン計算ではポスト処理による可視化
- スパコンによるメッシュ計算では多くのデータが出力
  - ✓ 計算に使用されるメモリサイズ程度のデータがそれぞれのステップで出力される（~ 1000GPU x 1 GByte x タイムステップ数）
- スパコンによる計算は可能であるが、データサイズが大きく、ポスト処理でのデータ移動は困難になってきている
- スパコンで計算と可視化を同時に行う等、データ移動しない可視化を考える必要がある



まとめ

GPUスパコンを用い、高い実行性能を達成した超大規模メッシュ計算を行った

- **メッシュ計算の大規模GPU計算**
  - ✓ フルGPUアプリケーション
  - ✓ GPUスパコンによる1000 GPU 以上を用いた大規模計算
- **マルチGPU計算のための最適化手法**
  - ✓ 計算による通信コストの隠蔽
  - ✓ GPU-CPU ハイブリッド計算
- **マルチGPU計算による実行性能**
  - ✓ **実アプリケーションによるペタスケールの実行性能**
  - ✓ 気象計算 ASUCA
    14368 x 14284 x 48 計算格子で**3990 GPUsを用い145.0 TFlops**
  - ✓ フェーズフィールド法による樹枝状凝固成長計算
    4,000 GPUs と16,000 CPU coresを用い **2.000 PFlops, 実行効率 44%**
    グリーンコンピューティング: 1468 MFlops/W