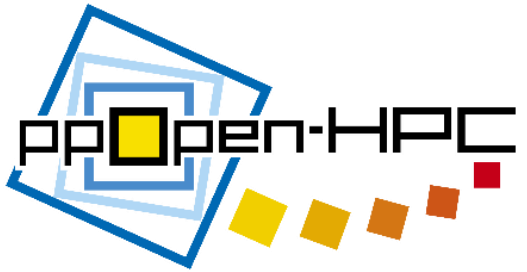
東京大学情報基盤センター
INFORMATION TECHNOLOGY CENTER, THE UNIVERSITY OF TOKYO

東京大学
THE UNIVERSITY OF TOKYO



ppOpen-HPCの概要とシミュレーション基本的流れ体験

松本正晴

東京大学情報基盤センター

第62回お試しアカウント付き並列プログラミング講習会

「ライブラリ利用：科学技術計算の効率化入門」

2016年9月6日（火）～7日（水）

本日の内容

1. 近年のスーパーコンピュータのトレンドとppOpen-HPCの概要（座学）

2. 3D熱伝導解析による並列化シミュレーションの基本的流れ（座学）

3. Oakleaf-FXでのサンプルコードを用いた演習

本日の内容

1. 近年のスーパーコンピュータのトレンドとppOpen-HPCの概要（座学）

2. 3D熱伝導解析による並列化シミュレーションの基本的流れ（座学）

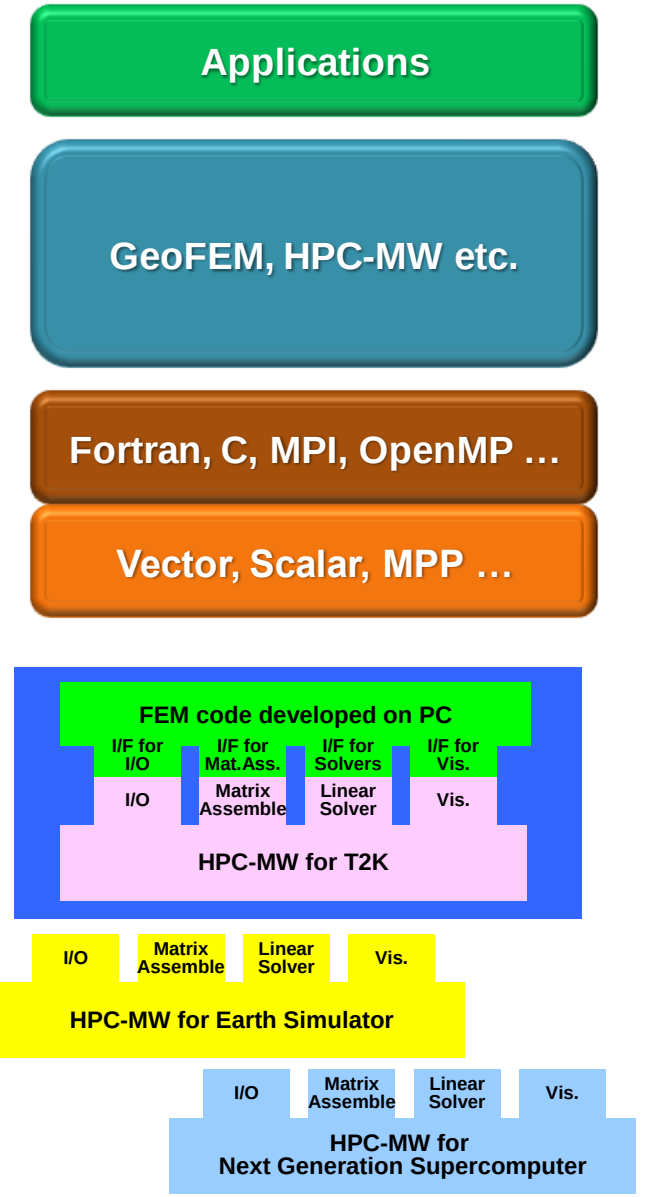
3. Oakleaf-FXでのサンプルコードを用いた演習

背景(1/2)

- 大規模化、複雑化、多様化するハイエンド計算機環境の能力を充分に引き出し、効率的なアプリケーションプログラムを開発することは困難
- 有限要素法等の科学技術計算手法:
  - ✓ プリ・ポスト処理，行列生成，線形方程式求解等の一連の共通プロセスから構成される。
  - ✓ これら共通プロセスを抽出し，ハードウェアに応じた最適化を施したライブラリとして整備することで，アプリケーション開発者から共通プロセスに関わるプログラミング作業，並列化も含むチューニング作業を隠蔽できる。
  - ✓ アプリケーションMW，HPC-MW，フレームワーク

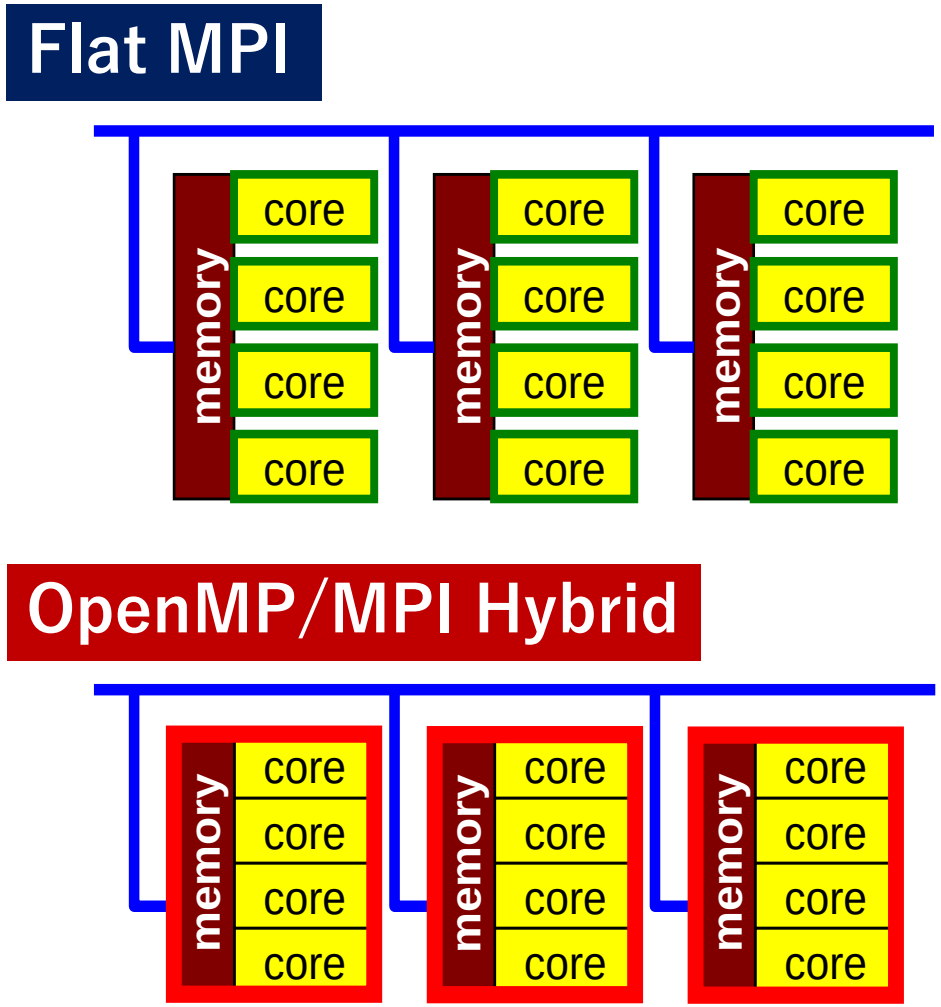
背景(2/2)

- A.D.2000年前後
  - ✓ GeoFEM，HPC-MW
  - ✓ 地球シミュレータ，Flat MPI，FEM
- 現在：より多様，複雑な環境
  - ✓ マルチコア，GPU
  - ✓ ハイブリッド並列
    - MPIまでは何とかたどり着いたが・・・
    - 「京」でも重要
  - ✓ CUDA，OpenCL，OpenACC
  - ✓ ポストペタスケールからエクサスケールへ
    - より一層の複雑化



Hybrid並列プログラミングモデル
MPI + “X”

- Message Passing
  - ✓ MPI
- Multi Threading
  - ✓ OpenMP
  - ✓ CUDA, OpenCL



Top500に見るスーパーコンピュータのトレンド（1/2）

http://www.top500.org/ June 2016 List R_{max}：実効性能（TFLOPS），R_{peak}：ピーク性能（TFLOPS），Power: kW

	Site	Computer/Year Vendor	Cores	R_{max}	R_{peak}	Power
1	National Supercomputer Center in Wuxi, China	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway	10,649,600	93,014 (=93.0PF)	125,435	15,371
2	National Supercomputer Center in Guangzhou, China	**Tianhe-2 (MilkyWay-2)** Xeon E5-2692, Xeon Phi 31S1P, 2013 NUDT	3,120,000	33,863 (=33.9PF)	54,902	17,808
3	DOE/SC/Oak Ridge National Laboratory, USA	**Titan** Cray XK7/NVIDIA K20x, 2012 Cray	560,640	17,590	27,113	8,209
4	DOE/NNSA/LLNL, USA	**Sequoia** BlueGene/Q, 2011 IBM	1,572,864	17,173	20,133	7,890
5	RIKEN AICS, Japan	**K computer** SPARC64 VIIIfx , 2011 Fujitsu	705,024	10,510	11,280	12,660
6	DOE/SE/Argonne National Laboratory, USA	**Mira** BlueGene/Q, 2012 IBM	786,432	8,586	10,066	3,945
7	DOE/NNSA/LANL/SNL, USA	**Trinity** Cray XC40, Xeon E5-2698v3, 2015 Cray	301,056	8,101	11,079	
8	Swiss National Supercomputing Ctr. (CSCS), Switzerland	**Piz Daint** Cray XC30, Xeon E5-2670 8C, NVIDIA K20x, 2013 Cray	115,984	6,271	7,789	2,325
9	Höchstleistungsrechenzentrum Stuttgart (HLRS), Germany	**Hazel Hen** Cray XC40, Xeon E5-2680v3, 2015 Cray	185,088	5,640	7,404	
10	KAUST, Saudi Arabia	**Shaheen II** Cray XC40, Xeon E5-2698v3 2015 Cray	196,608	5,537	7,235	2,834

Top500に見るスーパーコンピュータのトレンド（2/2）

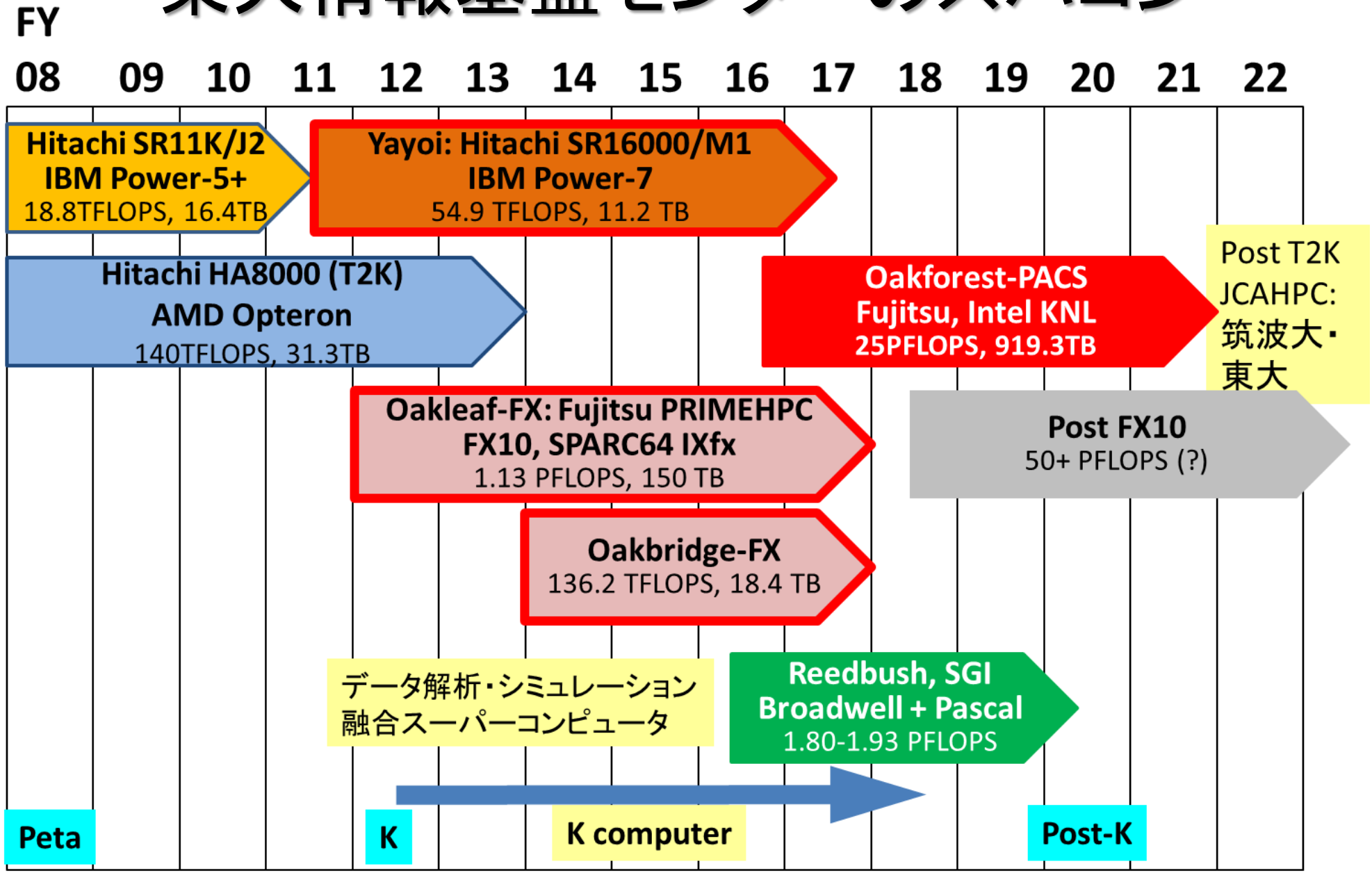
- ✓2016年6月現在，Top500に掲載されたシステムのうち，93システムはGPUかMICを搭載（67：NVIDIA，3：ATI-Radeon，26：Intel Xeon Phi，2: PEZY, 4: Intel/NVIDIA combined）
- ✓93システムのアクセラレータが持つ平均のコア数は76,000core/system
- ✓1位にランクインした中国のSunway TaihuLight（93 petaFLOPS）は40,960ノード、計10,649,600コア。SW26010プロセッサは4MPEs, 4CPEs (260コア/ノード)。
- ✓2位のTianhe-2はintel Xeon Phi、3位のTitanと8位のPiz DaintはNVIDIA GPUを搭載。

すでにGPUやMIC（accelerator/co-processor）を搭載したいわゆる“ヘテロジニアス”なアーキテクチャが主流になりつつあり、今後もこの傾向は続くと予想されている。

→ CUDA, OpenCL, OpenACC, OpenMP4.0….

→ 従来に比べ、（高性能）コーディングが難しい。

東大情報基盤センターのスパコン



データ解析・シミュレーション融合スーパーコンピュータシステム：Reedbush（2016年7月～）

- システム構成・運用：SGI
- Reedbush-U (CPU only)
  - Intel Xeon E5-2695v4 (Broadwell-EP, 2.1GHz 18core,) x 2ソケット (1.210 TF), 256 GiB (153.6GB/sec)
  - InfiniBand EDR, Full bisection BW Fat-tree
  - システム全系: 420 ノード, 508.0 TF
- Reedbush-H (with GPU) (2016年度末より試験運用開始予定)
  - CPU・メモリ：Reedbush-U と同様
  - NVIDIA Tesla P100 (Pascal世代 GPU)
    - (4.8-5.3TF, 720GB/sec, 16GiB) x 2 / ノード
  - InfiniBand FDR x 2ch, Full bisection BW Fat-tree
  - 120 ノード, 145.2 TF(CPU)+ 1.15~1.27 PF(GPU)= 1.30~1.42 PF

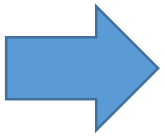
Post T2K: Oakforest-PACS

- 2016年12月1日稼働開始
- 8,208 Intel Xeon/Phi (KNL), ピーク性能25PFLOPS
  - 富士通が構築
- **最先端共同HPC 基盤施設(JCAHPC: Joint Center for Advanced High Performance Computing)**
  - 筑波大学計算科学研究センター
  - 東京大学情報基盤センター
    - 東京大学柏キャンパスの東京大学情報基盤センター内に、両機関の教職員が中心となって設計するスーパーコンピュータシステムを設置し，最先端の大規模高性能計算基盤を構築・運営するための組織
  - http://jcahpc.jp

Key-Issues for Appl's/Algorithms towards Post-Peta & Exa Computing

Jack Dongarra (ORNL/U. Tennessee) at ISC 2013

- Heterogeneous/Hybrid Architecture
- Communication/Synchronization Reducing Algorithms
- Mixed Precision Computation
- Auto-Tuning/Self-Adapting
- Fault Resilient Algorithms
- Reproducibility of Results



現在より複雑なチューニングが必須となるため，ユーザーを支援するためのフレームワークの重要性が高まる。

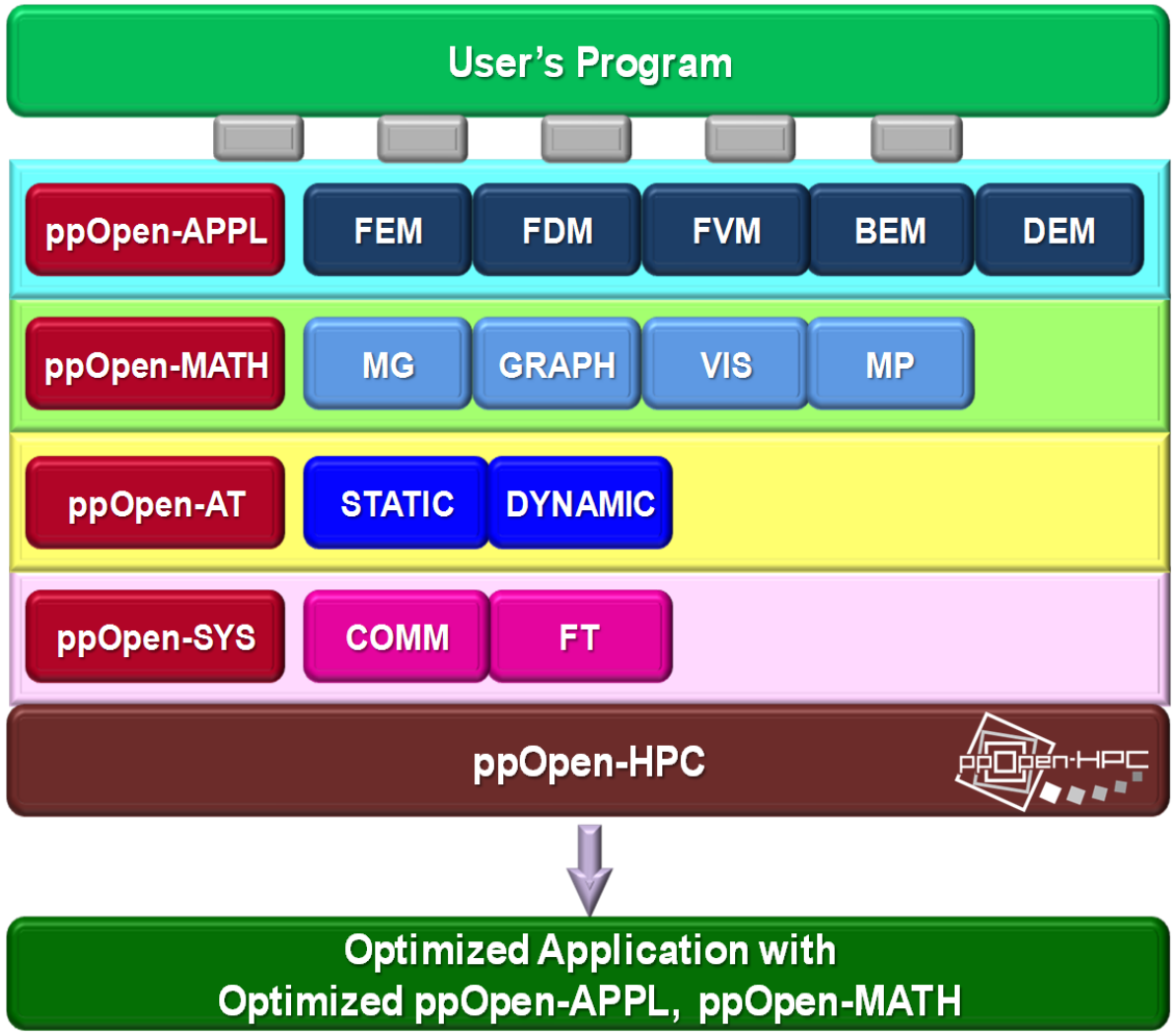
ppOpen-HPC

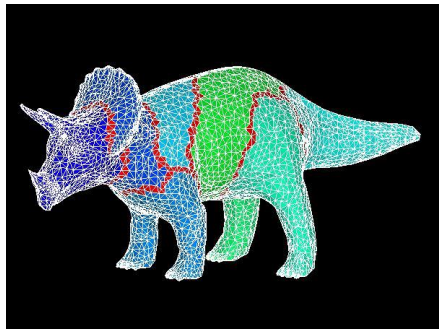
- 東京大学情報基盤センターでは，メニィコアに基づく計算ノードを有するポストペタスケールシステムの処理能力を充分に引き出す科学技術アプリケーションの効率的な開発，安定な実行に資する「自動チューニング機構を有するアプリケーション開発・実行環境：ppOpen-HPC」を開発中。
  - 科学技術振興機構戦略的創造研究推進事業（CREST）研究領域「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出（Post-Peta CREST）」（2011～2015年度）（領域統括：佐藤三久（理化学研究所計算科学研究機構））
  - PI：中島研吾（東京大学情報基盤センター）
  - 東大（情報基盤センター，大気海洋研究所，地震研究所，大学院新領域創成科学研究科），京都大学術情報メディアセンター，北海道大学情報基盤センター，海洋研究開発機構
  - 様々な分野の専門家によるCo-Design

概要

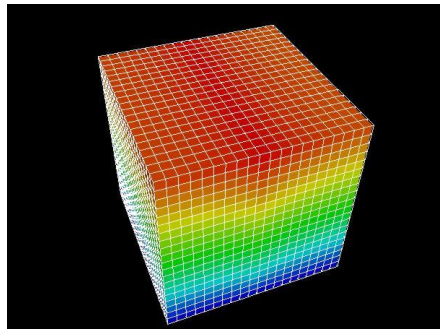
- 大規模シミュレーションに適した5種の離散化手法に限定し，各手法の特性に基づいたアプリケーション開発用ライブラリ群，耐故障機能を含む実行環境を実現する。
  - **ppOpen-APPL**：各手法に対応した並列プログラム開発のためのライブラリ群
  - **ppOpen-MATH**：各離散化手法に共通の数値演算ライブラリ群
  - **ppOpen-AT**：科学技術計算のための自動チューニング（AT）機構
  - **ppOpen-SYS**：ノード間通信，耐故障機能に関連するライブラリ群
- 平成24年11月にマルチコアクラスタ向けに各グループの開発したppOpen-APPL，ppOpen-AT，ppOpen-MATHの各機能を公開（Ver.0.1.0）
  - http://ppopenhpc.cc.u-tokyo.ac.jp/
  - 平成25年11月にVer.0.2.0公開
  - 平成26年11月にVer.0.3.0公開
  - 平成27年11月にVer.1.0.0公開

ライブラリ群と対象とする離散化手法

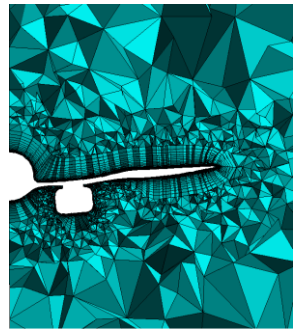




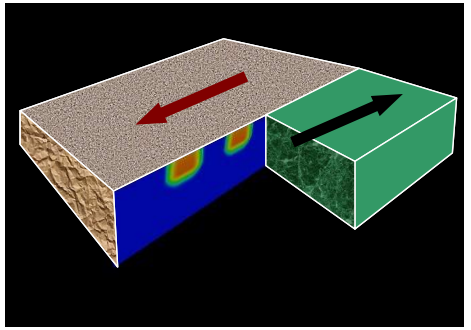
有限要素法
Finite Element Method
FEM



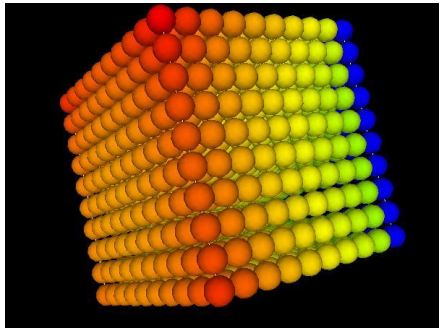
差分法
Finite Difference Method
FDM



有限体積法
Finite Volume Method
FVM

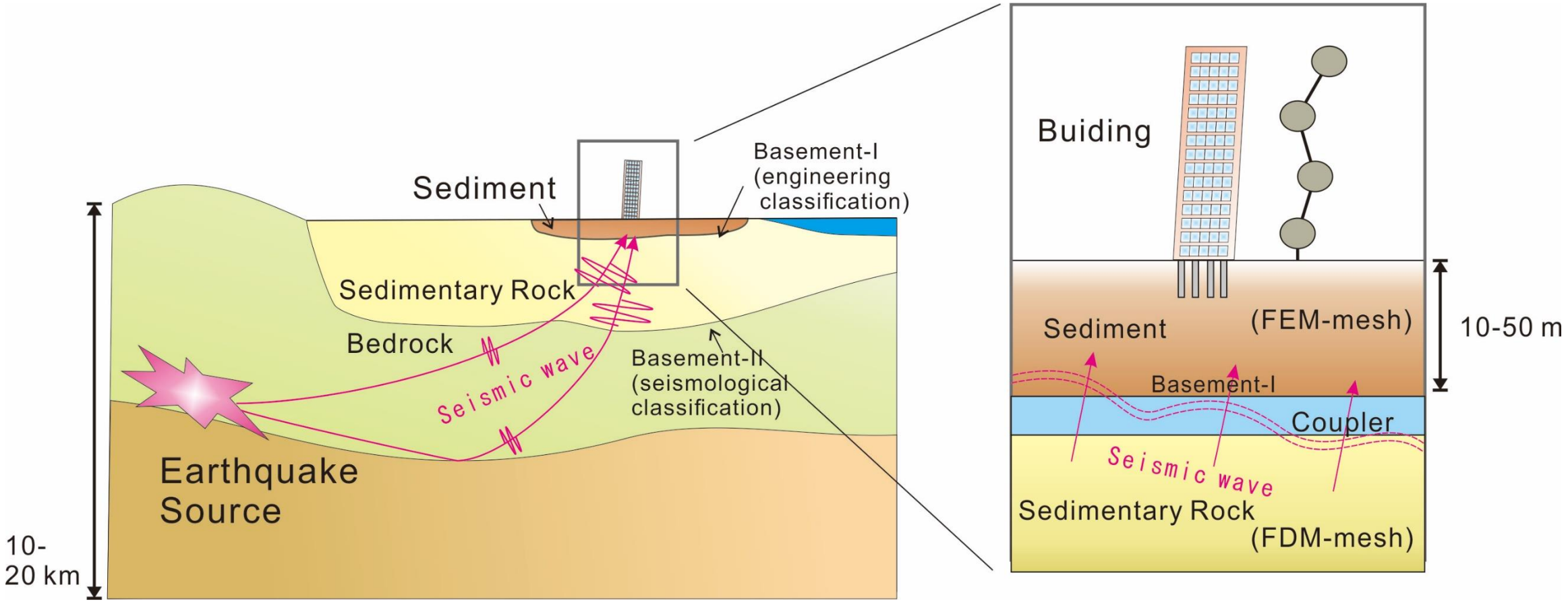


境界要素法
Boundary Element Method
BEM



個別要素法
Discrete Element Method
DEM

ppOpen-HPCを利用した地震波動－建築物振動連成シミュレーション



地震波動の伝播（FDM）と、それに伴う建築物の振動（FEM）について、
FDM→FEMの片側連成で解析を行う。

利用するアプリケーションについて

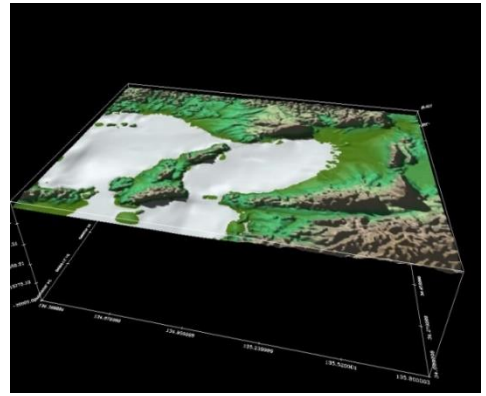
Seism3D+ (composed by ppOpen-APPL/FDM)

広域の地震波動伝播を解析するための陽解法FDMアプリ

$$\rho \frac{\partial v_p}{\partial t} = \left(\frac{\partial \sigma_{xp}}{\partial x} + \frac{\partial \sigma_{yp}}{\partial y} + \frac{\partial \sigma_{zp}}{\partial z} + f_p \right), \quad (p = x, y, z)$$

$$\frac{\partial \sigma_{pq}}{\partial t} = \lambda \left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} \right) \delta_{pq} + \mu \left(\frac{\partial v_p}{\partial q} + \frac{\partial v_q}{\partial p} \right), \quad (p, q = x, y, z)$$

v: 速度
σ: 応力
f: 外力
λ, μ: Lame定数

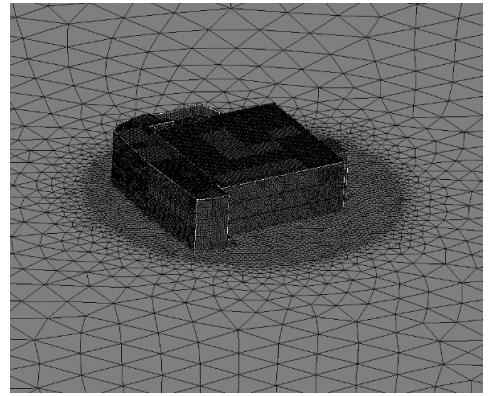


FrontISTR++ (composed by ppOpen-APPL/FEM)

複雑な形状にも対応できる構造解析用陰解法FEMアプリ

$$\mathbf{M}\ddot{\mathbf{d}} + \mathbf{C}\dot{\mathbf{d}} + \mathbf{K}\mathbf{d} = \mathbf{F}$$

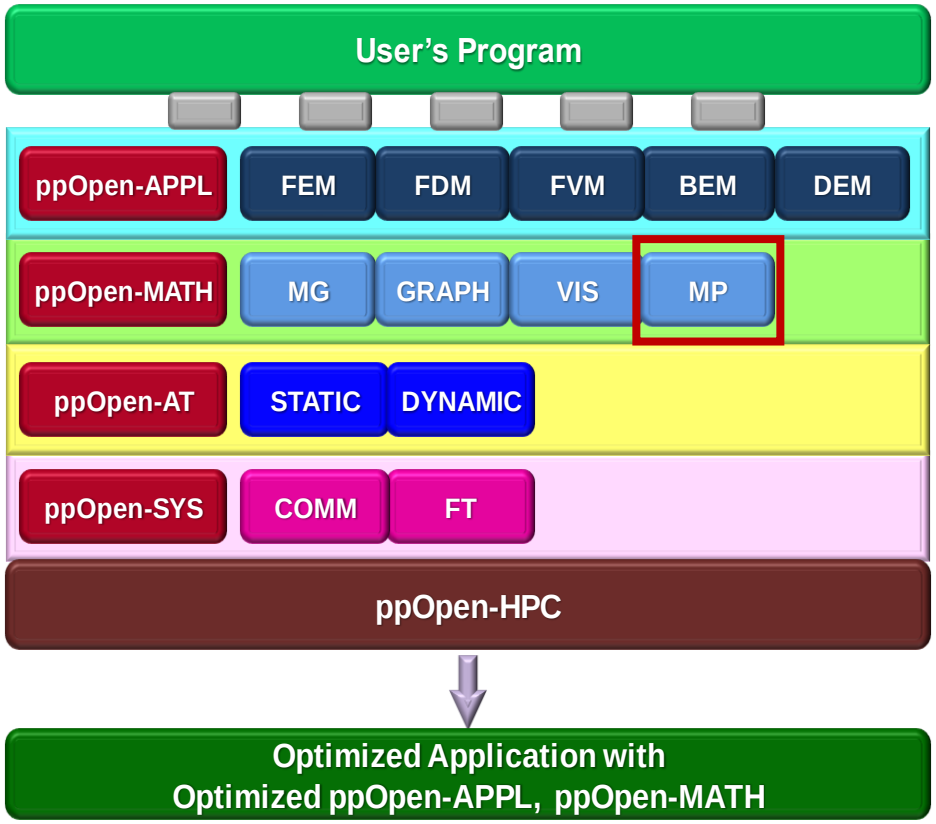
M: 質量行列
C: 減衰行列
K: 剛性行列
F: 節点荷重ベクトル
d: 節点変位ベクトル

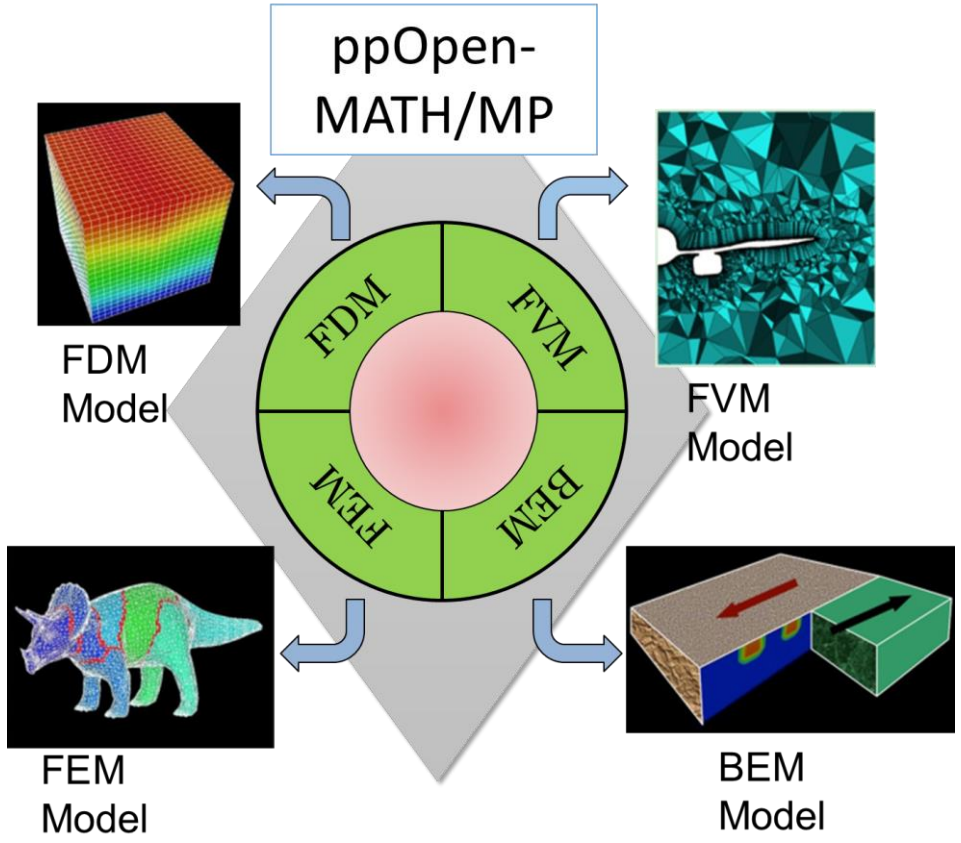


両者の計算負荷はSeism3D+ ＜ FrontISTR++

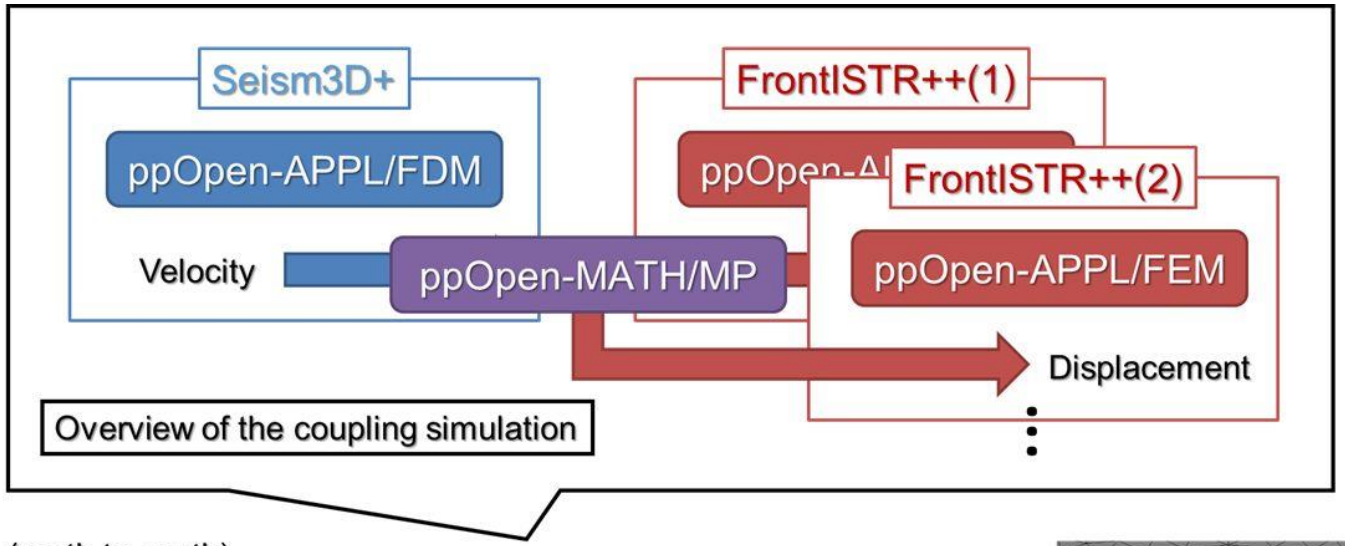
カップラライブラリppOpen-MATH/MP

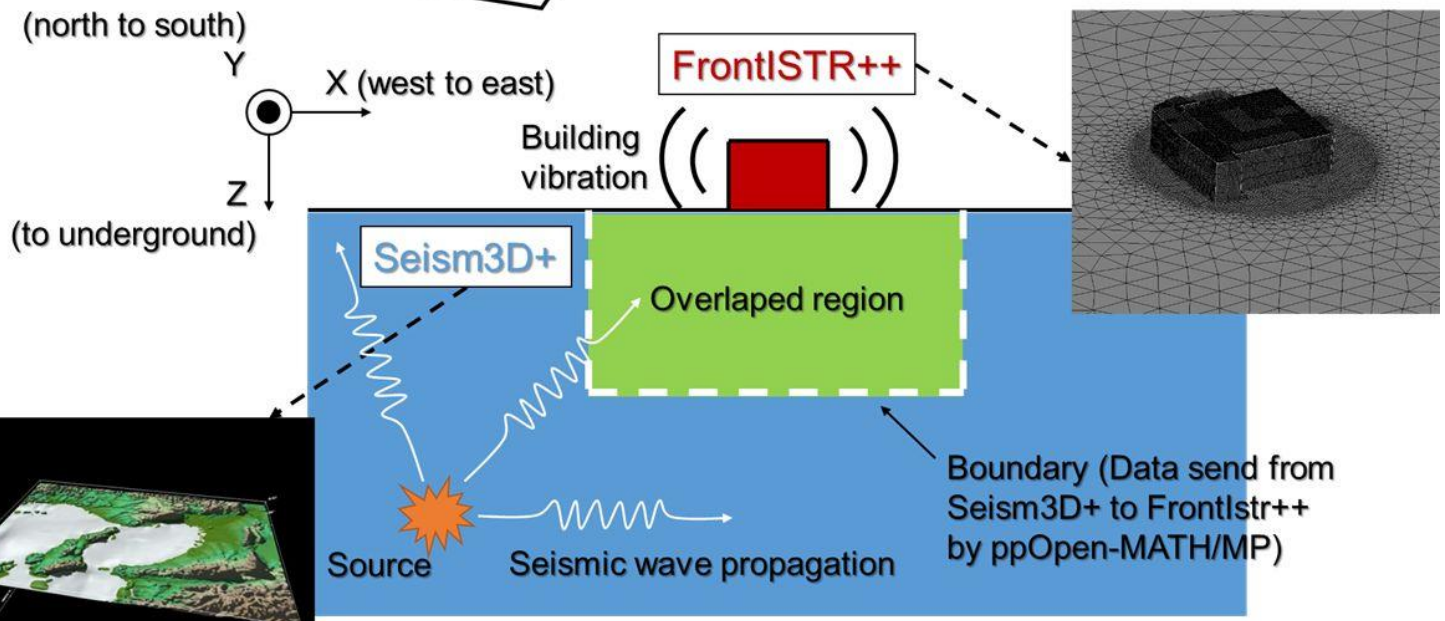
ppOpen-APPLライブラリ群がサポートする離散化手法（FDM，FEM，FVM，BEM，DEM）を基に構築される複数のアプリ間のモデル結合，データ送受信，データ変換のための弱連成用カップラー

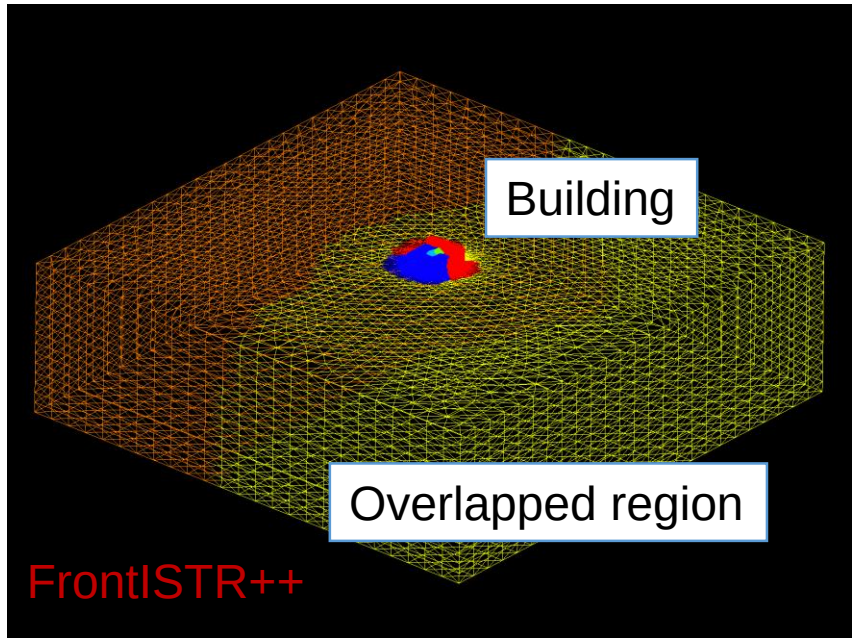




連成シミュレーションの実装



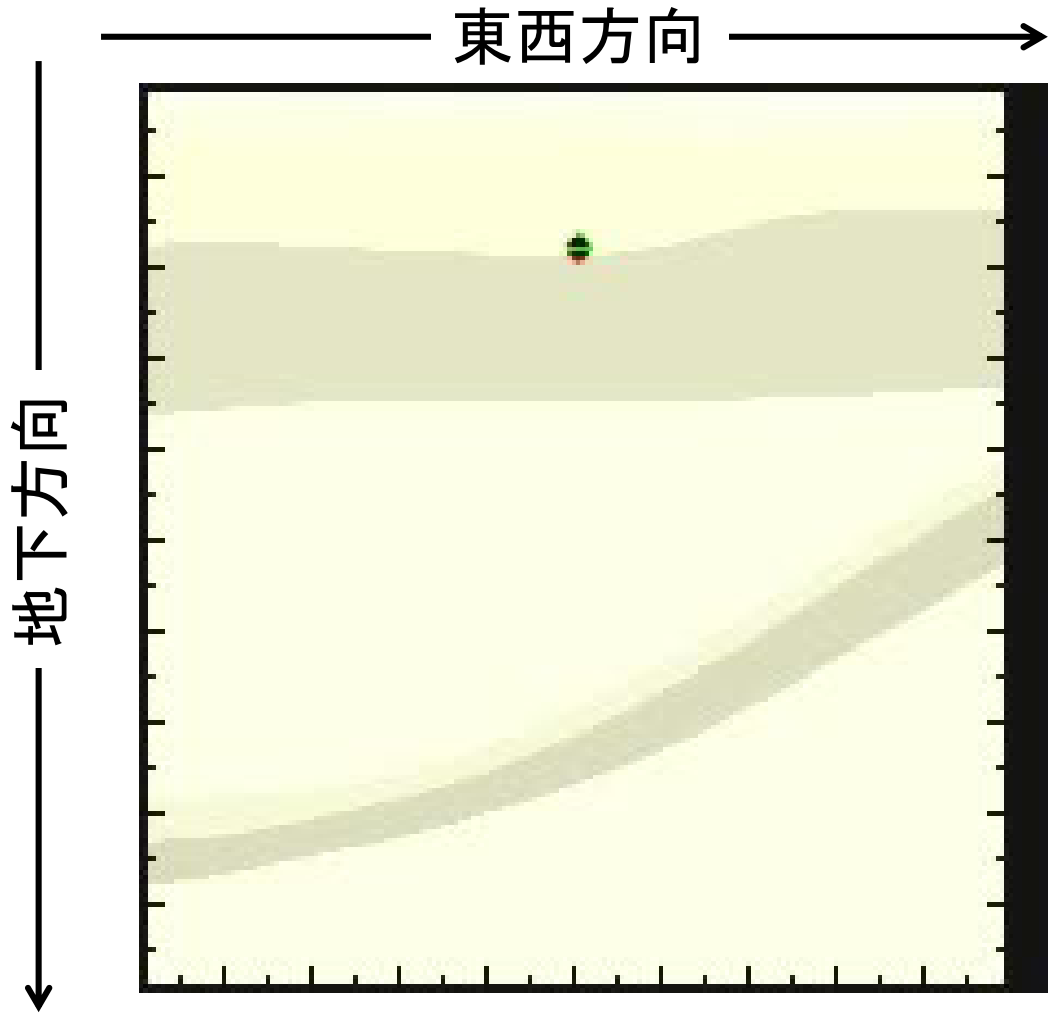




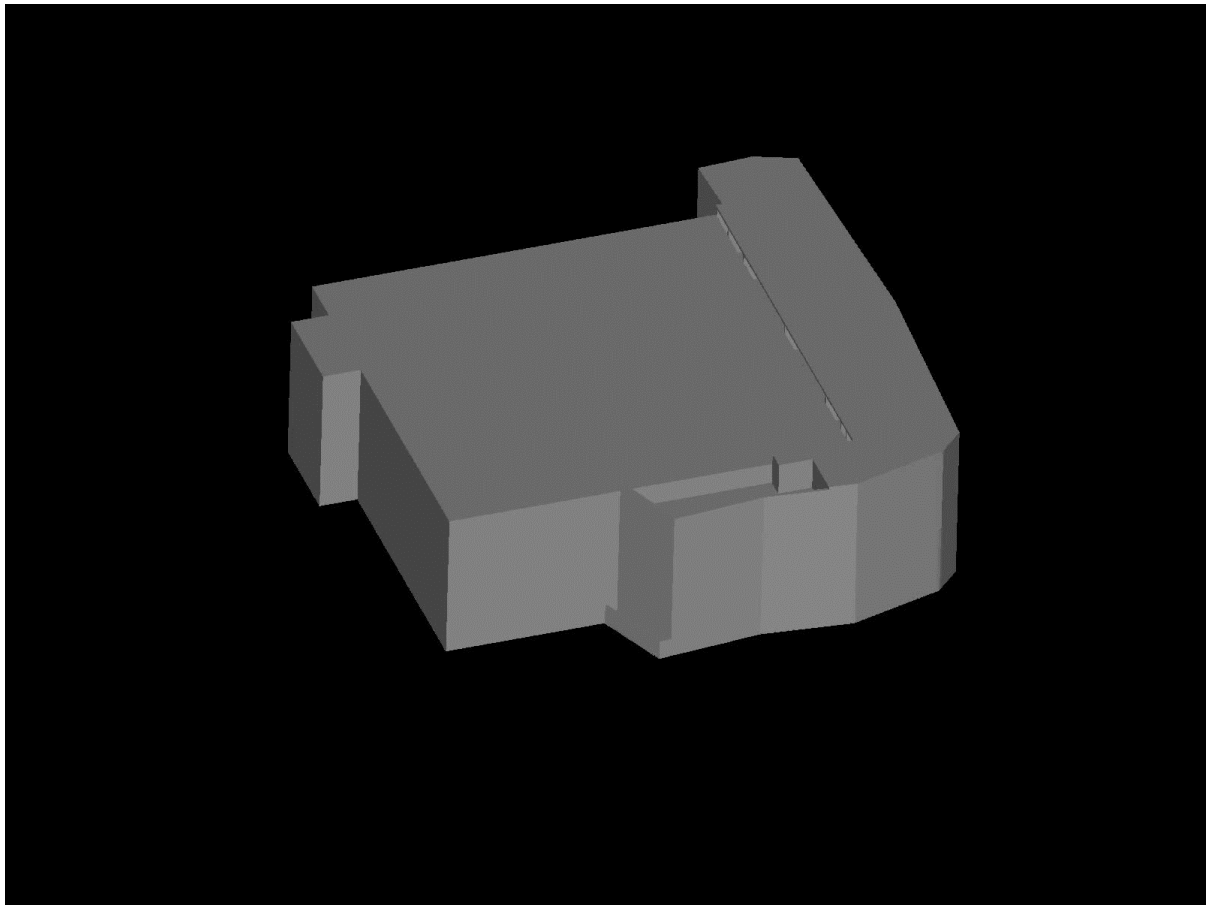
FrontISTR++の格子
色はプロセス（64プロセスの場合）

Oakleaf-FXによる大規模実行の結果

計4560ノード@Oakleaf-FX（Seism3D+: 2560ノード、FrontISTR++: 2000ノード）



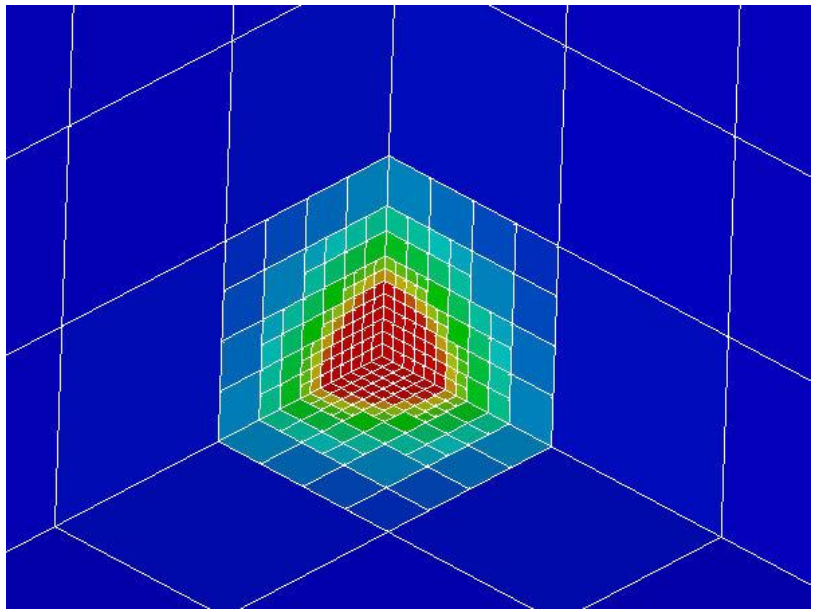
Seismic wave propagation by Seism3D+
（Red：P-wave、Green：S-wave）



Vibration of K computer bldg.
by FrontISTR++

可視化用ライブラリppOpen-MATH/VIS

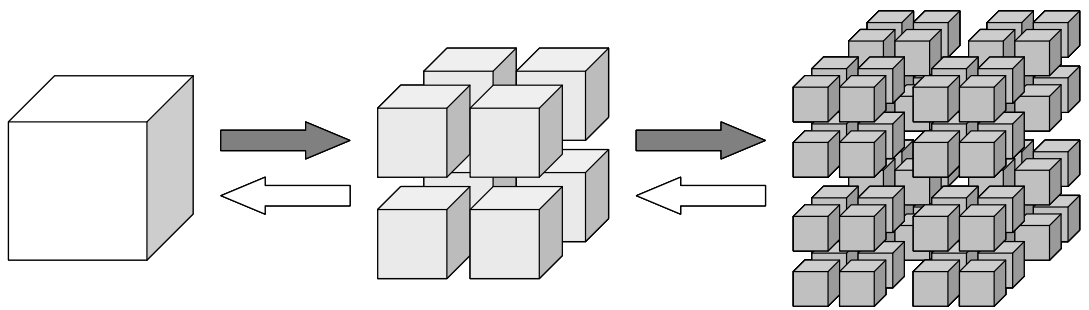
- ボクセル型背景格子を使用した大規模並列可視化手法〔Nakajima & Chen 2006〕に基づく
  - 差分格子用バージョン公開：ppOpen-MATH/VIS-FDM3D
- UCD single file
- プラットフォーム
  - T2K，Cray
  - FX10
  - Flat MPI

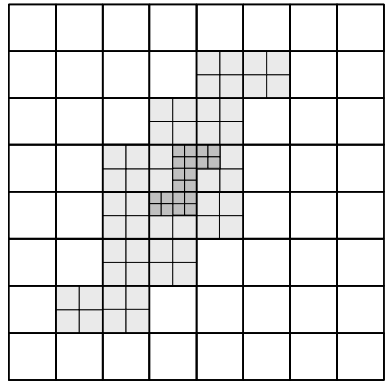


Simplified Parallel Visualization using Background Voxels

[KN, Chen 2006]

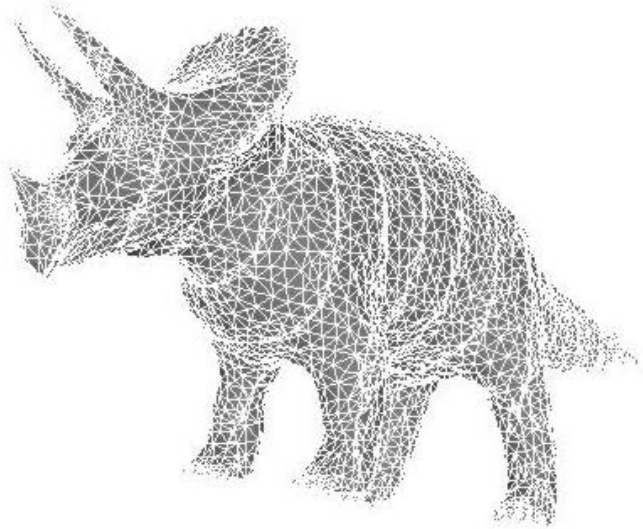
- Octree-based AMR
- AMR applied to the region where gradient of field values are large
  - stress concentration, shock wave, separation etc.
- If the number of voxels are controled, a single file with 10^5 meshes is possible, even though entire problem size is 10^9 with distributed data sets.



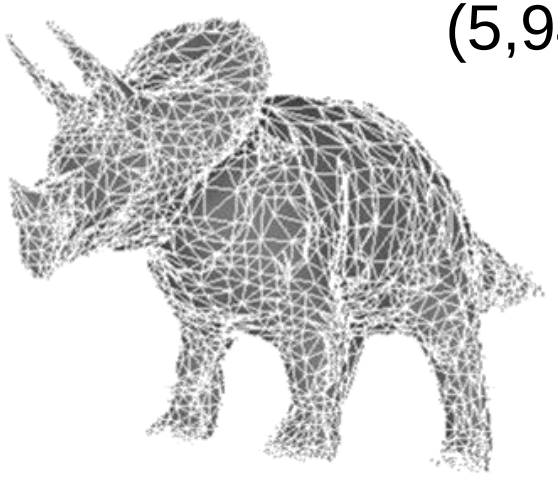


Example of Surface Simplification

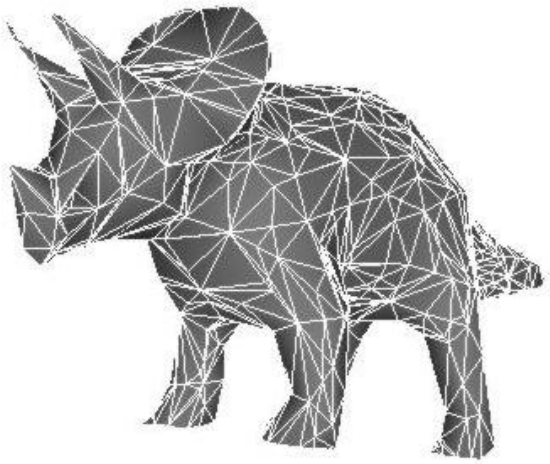
Initial
(11,884 tri's)



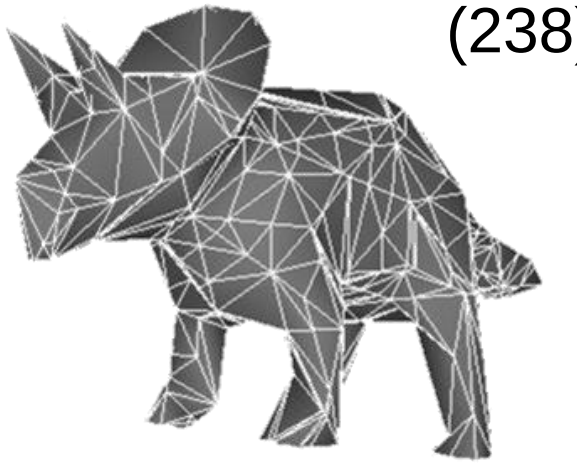
50% reduction
(5,942)



95% reduction
(594)

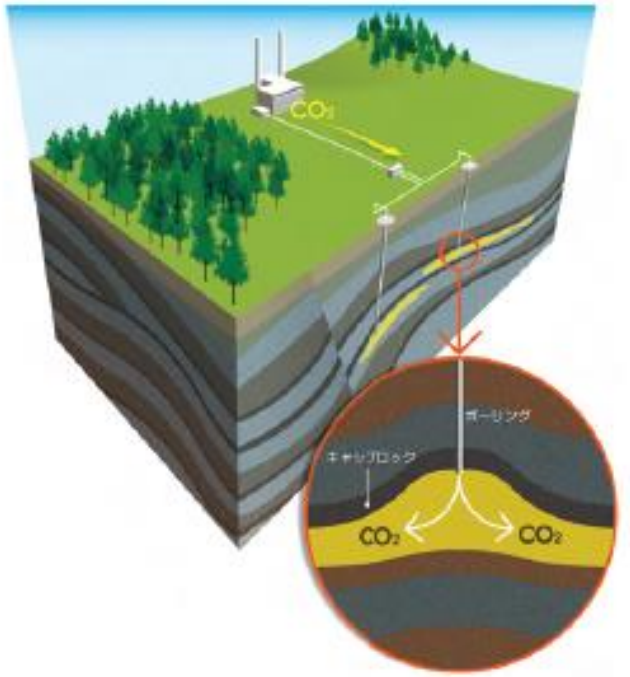


98% reduction
(238)



研究協力・普及

- 国際的共同研究
  - Lawrence Berkeley National Lab.
  - 国立台湾大学
  - **ESSEX/SPPEXA/DFG, Germany**
  - IPCC（Intel Parallel Computing Ctr.）
- 普及
  - 大規模シミュレーションへの適用
    - CO_2 地下貯留，物性物理
    - 宇宙物理，地震シミュレーション
    - ppOpen-AT, ppOpen-MATH/VIS, ppOpen-MATH/MP, 線形ソルバー群
    - H行列ライブラリ
  - 国際WS（2012,13,15）
  - 講習会（東大センター），講義



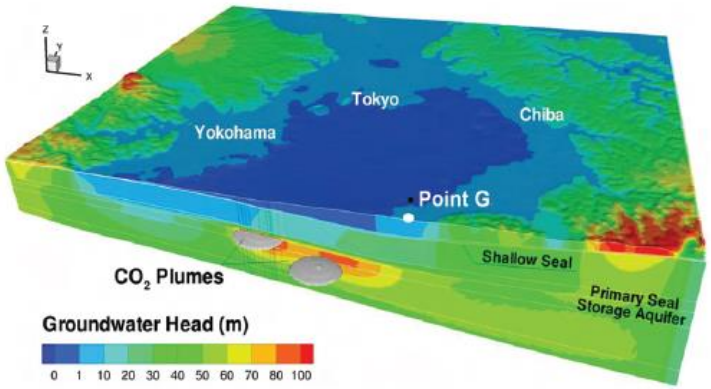


図 -4　CO_2 圧入後の地下水圧（全水頭換算）の分布（100 年後）

本日の内容

1. 近年のスーパーコンピュータのトレンドとppOpen-HPCの概要（座学）

2. 3D熱伝導解析による並列化シミュレーションの基本的流れ（座学）

3. Oakleaf-FXでのサンプルコードを用いた演習

3次元並列化熱伝導シミュレーションの演習

- 計算機シミュレーションは，支配方程式の離散化→並列化プログラムの実装→計算の実行→計算結果の可視化→最適化，が一連の流れ。
- 陽的差分法による離散化を施した3次元熱伝導方程式の並列シミュレーション（MPI + OpenMP）の基本的な流れを体験。

  - Oakleaf-FXの利用
  - Fortranによる実装例
  - ppOpen-MATH/VISを利用する可視化（Paraview使用）
  - 富士通プロファイラの利用（テキスト and Excel）

熱伝導方程式の離散化(1/3)

物質の内部エネルギー（$E=\rho CT$）の時間変化 ＋ 熱流束qの出入り ＝ 熱源S

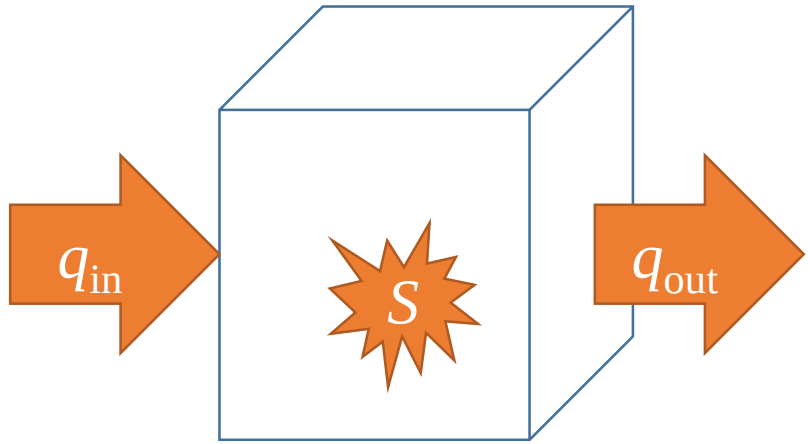
$$\rho C\frac{\partial T}{\partial t}+\nabla\cdot\mathbf{q}=S \rightarrow \rho C\frac{\partial T}{\partial t}+\frac{\partial q_x}{\partial x}+\frac{\partial q_y}{\partial y}+\frac{\partial q_z}{\partial z}=S$$

$$\rightarrow \frac{\partial T}{\partial t}=\frac{1}{\rho C}\left\{\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)+\frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right)+S\right\}$$

ただし，$q_x=-k\frac{\partial T}{\partial x}$，　$q_y=-k\frac{\partial T}{\partial y}$，　$q_z=-k\frac{\partial T}{\partial z}$

密度，比熱，熱伝導率が空間に対して一様なら，

$$\rightarrow \frac{\partial T}{\partial t}=\alpha\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}\right)+\frac{S}{\rho C}$$　ただし，$\alpha=\frac{k}{\rho C}$



T：物質の温度 [K]
ρ：物質の密度 [kg/m^3]
C：物質の比熱 [J/(kgK)]
$\mathbf{q}$：熱流束 [W/m^2]
S：熱源 [W/m^3]
k：物質の熱伝導率 [W/(mK)]

熱伝導方程式の離散化(2/3)

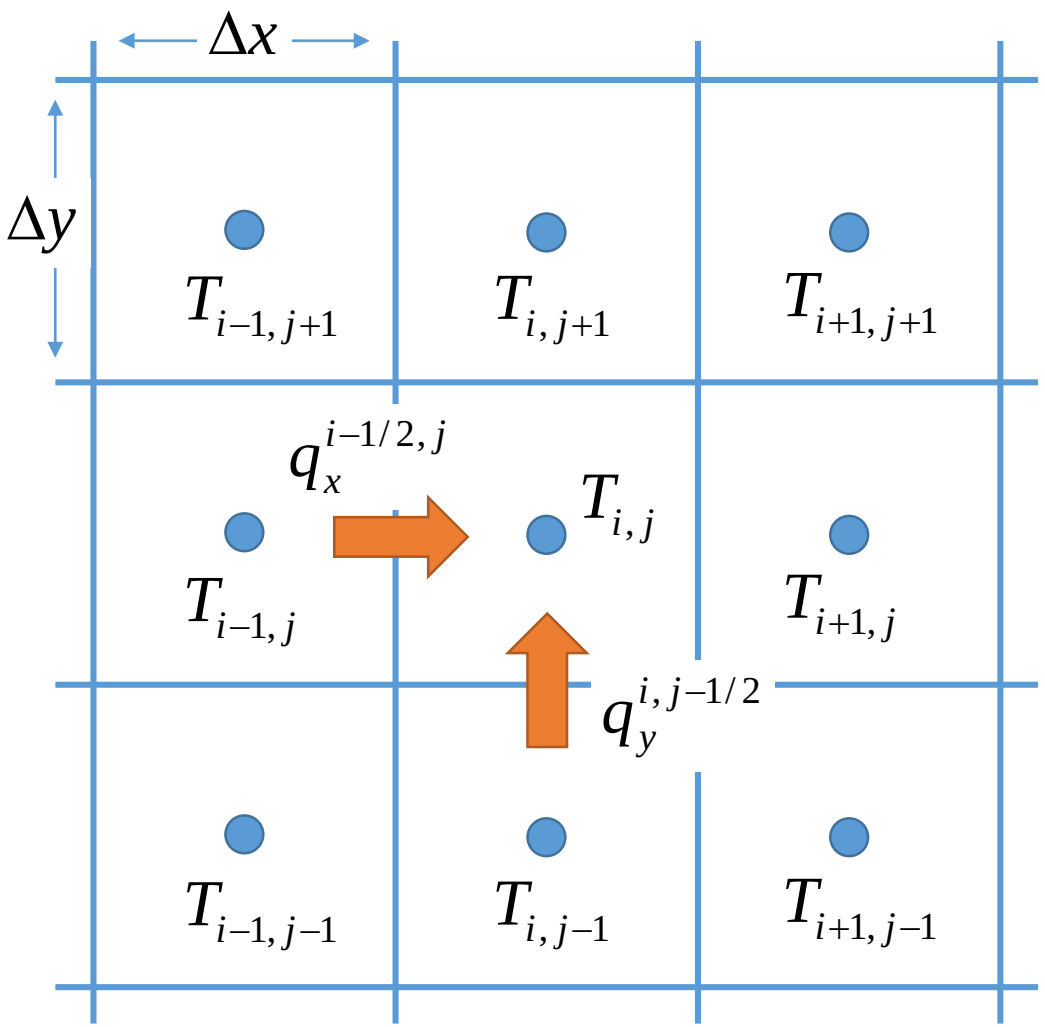
計算格子は3次元等間隔直交座標系(図は2次元)。

密度，比熱，熱伝導率が空間に対して**非一様**の場合は，右図のように有限体積的な格子系を用いると離散化しやすい。物理量は整数位置(i,j)に定義。

まず，熱流束を半整数位置で評価。

$$q_x^{i-1/2,j} = -k_{i-1/2,j}\frac{T_{i,j}-T_{i-1,j}}{\Delta x} = -\frac{k_{i,j}+k_{i-1,j}}{2}\frac{T_{i,j}-T_{i-1,j}}{\Delta x}$$

$$q_y^{i,j-1/2} = -k_{i,j-1/2}\frac{T_{i,j}-T_{i,j-1}}{\Delta x} = -\frac{k_{i,j}+k_{i,j-1}}{2}\frac{T_{i,j}-T_{i,j-1}}{\Delta x}$$



熱伝導方程式の離散化(3/3)

時間微分はEuler陽解法で離散化すると，次の時間ステップの温度は以下のように求められる。

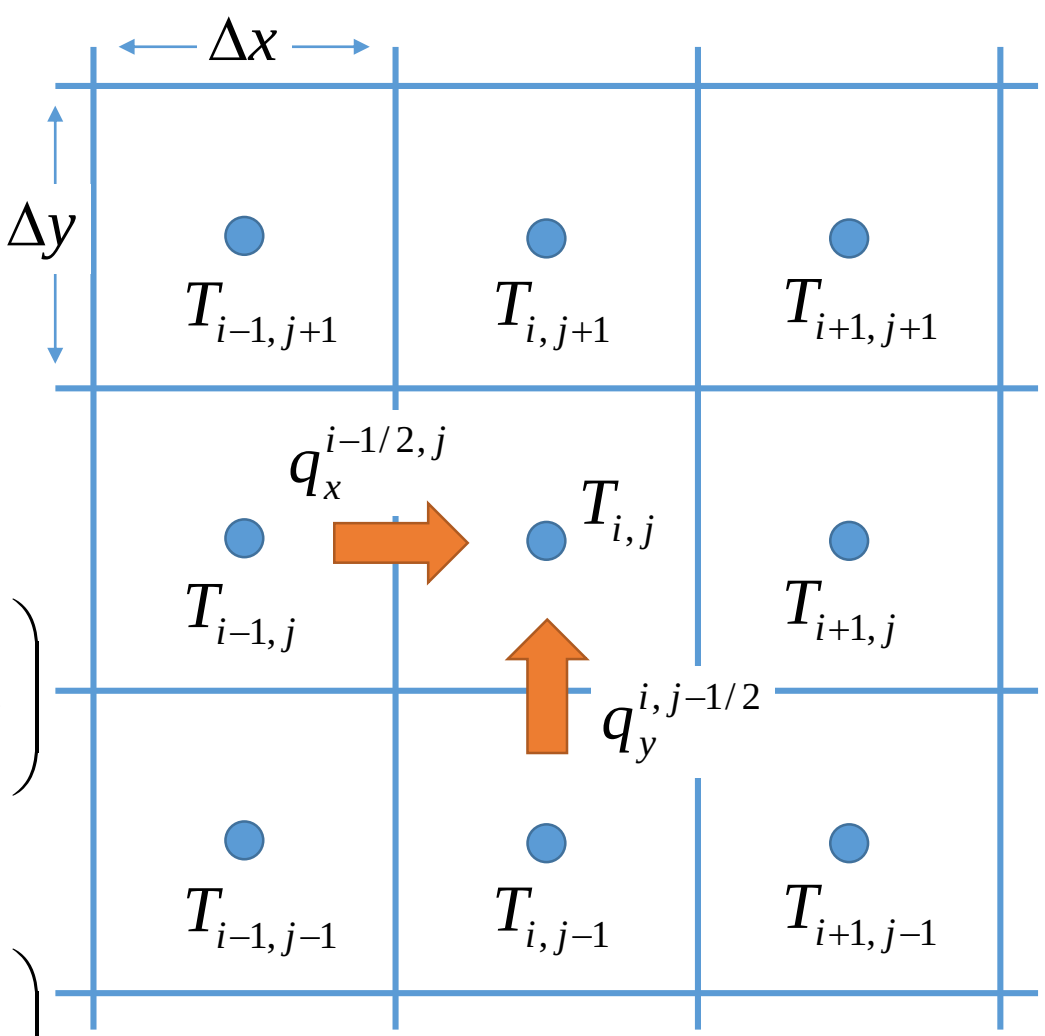
$$\frac{\partial T}{\partial t} = -\frac{1}{\rho C}\left(\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} - S\right)$$

$$\rightarrow \frac{T_{i,j}^{next} - T_{i,j}}{\Delta t} =$$

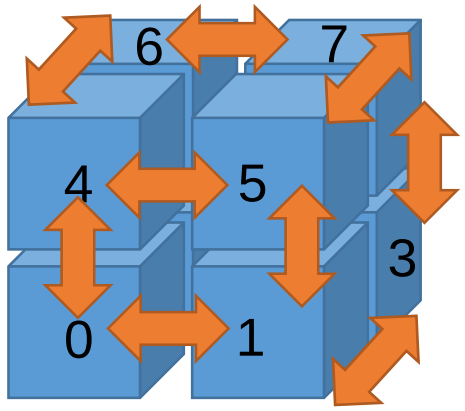
$$-\frac{1}{\rho_{i,j} C_{i,j}}\left(\frac{q_x^{i+1/2,j} - q_x^{i-1/2,j}}{\Delta x} + \frac{q_y^{i,j+1/2} - q_y^{i,j-1/2}}{\Delta y} - S_{i,j}\right)$$

$$\rightarrow \therefore T_{i,j}^{next} = T_{i,j}$$

$$-\frac{\Delta t}{\rho_{i,j} C_{i,j}}\left(\frac{q_x^{i+1/2,j} - q_x^{i-1/2,j}}{\Delta x} + \frac{q_y^{i,j+1/2} - q_y^{i,j-1/2}}{\Delta y} - S_{i,j}\right)$$

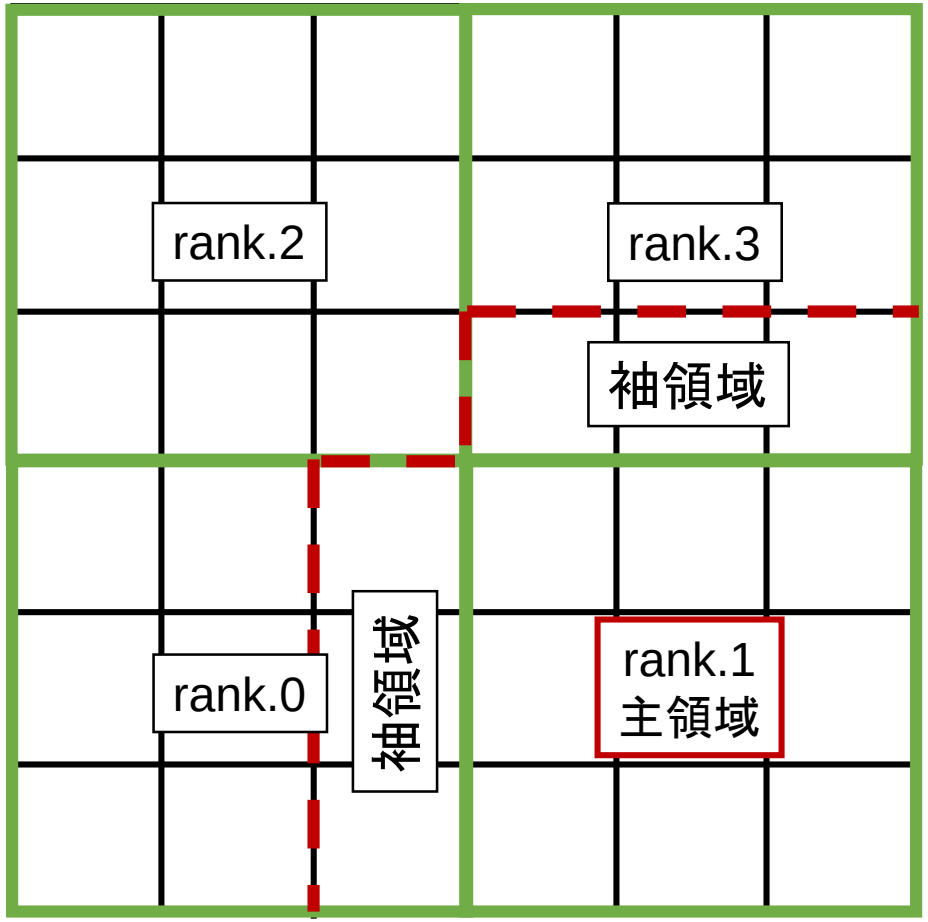


計算領域とMPI通信



3次元領域分割（数字はランク）

隣接する他のランクの計算領域と数グリッド分だけ袖領域を重なり合うように持っており、そこに隣接ランクの物理量を通信して持ってくる。



計算領域（2次元例）

本日の内容

1. 近年のスーパーコンピュータのトレンドとppOpen-HPCの概要（座学）

2. 3D熱伝導解析による並列化シミュレーションの基本的流れ（座学）

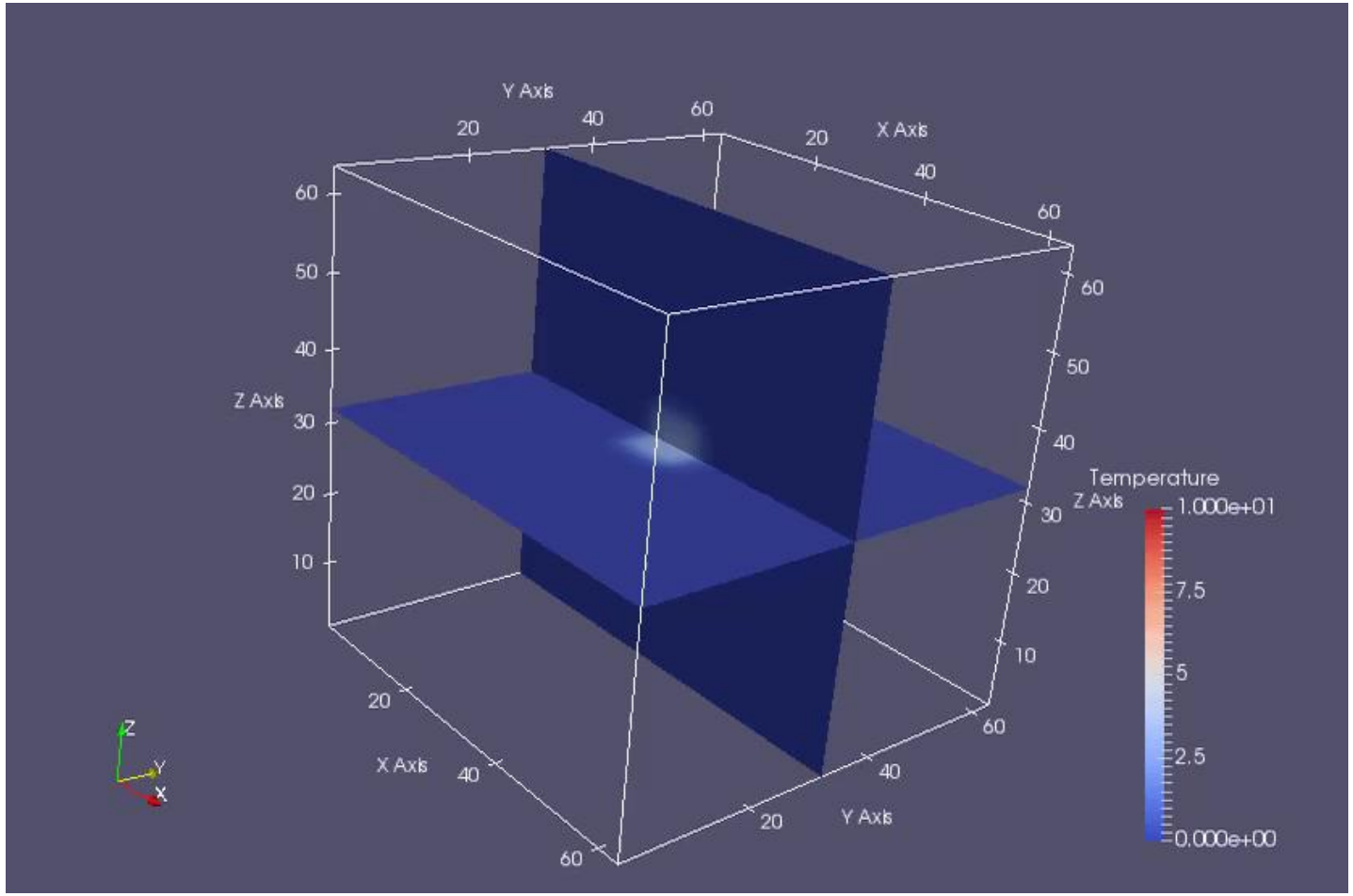
3. Oakleaf-FXでのサンプルコードを用いた演習

サンプルコードで計算する問題

3次元の計算領域（物体）の中央に熱源を置いた際の温度上昇と温度の伝わり方を見る。

計算領域（格子点数:64×64×64）
Δx=1.0, Δy=1.0, Δz=1.0
X=0.5～63.5
Y=0.5～63.5
Z=0.5～63.5

熱源の範囲
X=28～36, Y=28～36, Z=28～36



サンプルコード（main.f90）のメイン関数部分

```
program thermal_conduction_3D
  use global_var
  implicit none
  integer::nt,nstep
  double precision::mtime
  call start_parallel
  call load_param
  call allocate_array
  call initial_conditions
  mtime=0.0d0; nstep=0
  do nt=1,ntmax
     mtime=mtime+dt
     call kernel
     if(mod(nt,nfreq)==0) then
        if(rank==0) print *,'step & time:',nt,mtime
        nstep=nstep+1
        call output(nstep,T)
     endif
  enddo
  call end_parallel
contains
```

グローバル変数はモジュールで定義。各サブルーチンは
このモジュールをuseすることで、グローバル変数を共有。

mpi_init等
外部ファイルから計算パラメータを読み込み（格子点数，プロセス数等）
読み込んだパラメータを基に配列を用意

物理量の初期条件の設定

時間発展のメインループ

通信を含む熱伝導の計算部分

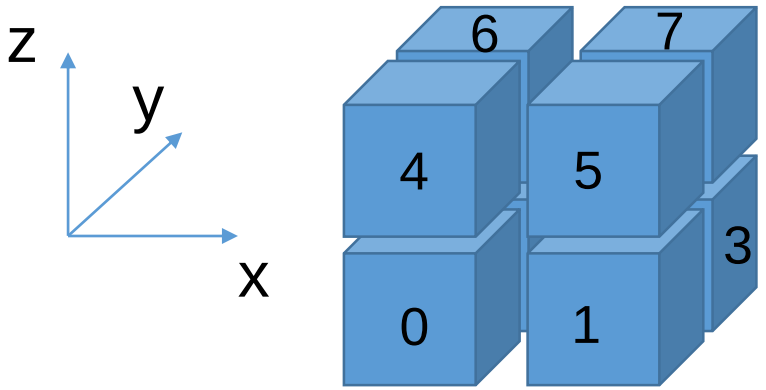
計算結果のファイル出力

mpi_finalize

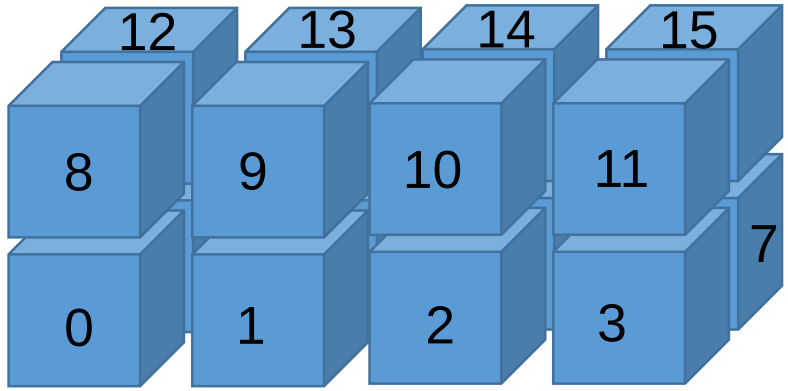
計算パラメータ設定用ファイル（input.dat）

```
32  32  32     !nxmax  nymax  nzmax
 1             !                nbuff
 2   2   2     !pxmax  pymax  pzmax
1000 100       !        ntmax  nfreq
0.1d0          !                 CFL
```

1プロセス当たりの格子点数（x,y,z）
袖領域の格子点数（>=1）
MPIプロセス数（x,y,z）
全タイムステップ数，出力タイミング
クーラン数（< 0.5）



例えば上図だと
pxmax=2, pymax=2, pzmax=2
（全8プロセス）



例えば上図だと
pxmax=4, pymax=2, pzmax=2
（全16プロセス）

ジョブスクリプトファイル（job.sh）

```
#!/bin/sh

#PJM -L "rscgrp=tutorial"
#PJM -L "node=2"
#PJM -L "elapse=00:15:00"
#PJM -g gt00
#PJM --mpi "proc=8"

export OMP_NUM_THREADS=4
export PARALLEL=4

rm ./results/*
mpiexec ./original
```

ノード数

MPIプロセス数
（input.datで指定したpxmax*pymax*pzmax
の値と合わせること！）

OpenMPスレッド数（<=16）

ノード数，MPIプロセス数，OpenMP
スレッド数は任意に設定できる。

左の例では，8プロセス×4スレッド
＝32コア使用する場合，16コア/
ノードなので，ノード数は2となる。

ソースファイルの準備

Oakleaf-FXの/home/c26050にLecture_20160907.tar.gzというファイルがありますので、各自コピーして展開してください。

```
% cp /home/c26050/Lecture_20160907.tar.gz ./
% tar xvzf Lecture_20160907.tar.gz
% cd Lecture_20160907
% ls
1.original 2.with_vis 3.with_prof_txt 4.with_prof_excel
```

Lecture_20160907/の中身

1.original/	オリジナルのサンプルコード
2.with_vis/	ppOpen-MATH/VISを実装したコード
3.with_prof_txt/	プロファイルルーチンを実装したコード（txt出力）
4.with_prof_excel/	プロファイルルーチンを実装したコード（excel出力）

1. サンプルコードの実行

1. 1.original/の中のcompileファイルを実行

```
% ./compile
```

2. すると，run/の中に実行ファイルoriginalが生成。runに移動。

```
% cd run
% ls
input.dat   job.sh   original   results/
```

パラメータ設定用入力ファイル　ジョブスクリプト　実行ファイル　計算結果出力用ディレクトリ

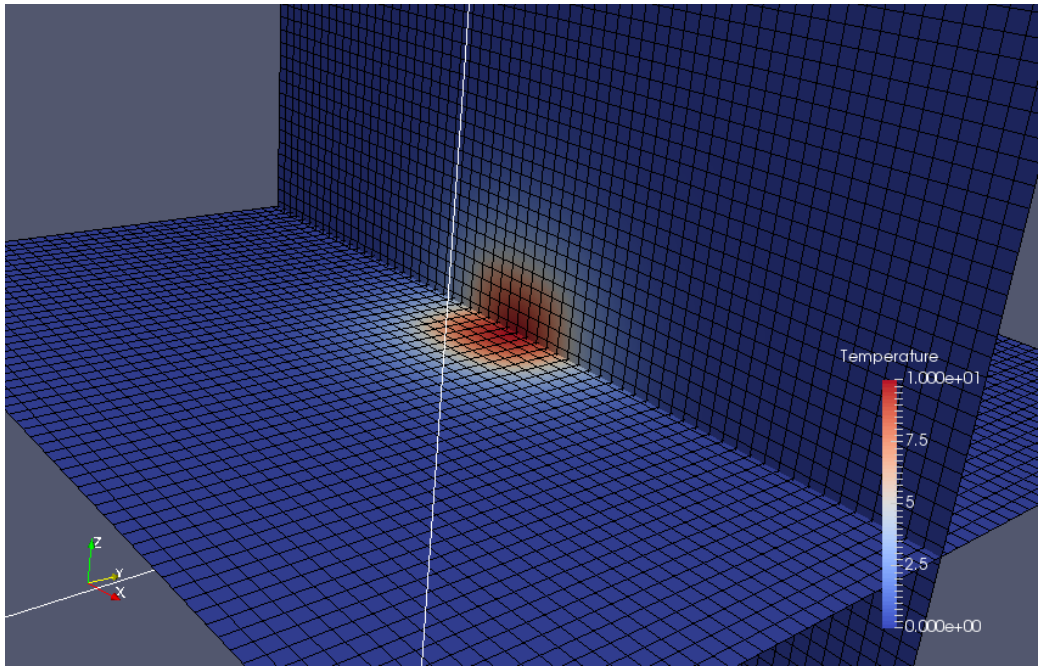
3. バッチジョブ実行で計算（ジョブスクリプトで指定するプロセス数はinput.datで指定したプロセス数pxmax*pymax*pzmaxと一致させること！）

```
% pjsub job.sh
```

4. 実行が終了したらresults/の中にvtkファイルが出力されていることを確認

演習1

1. 計算結果の出力ファイル（out_temperature.1.vtk～out_temperature.10.vtk）を自分のPCにダウンロードして，Paraviewで可視化（スナップショット，アニメーション等）しなさい。
2. 入力ファイル`input.dat`ならびにジョブスクリプト`job.sh`を自由に書き換えて計算しなさい。（格子点数，プロセス数，出力ファイル数等を増やしすぎると計算が重くなりすぎるので注意！）



ppOpen-MATH/VIS可視化ライブラリの利用

計算が大規模化すると，計算に使用する格子点数が膨大に増加してしまうため，可視化を行うことが難しくなってくる。ppOpen-MATH/VISライブラリを使うことによって，物理量の勾配がきつい領域は細かい，勾配が緩やかな領域は粗い格子点数で出力できる。

公開パッケージの配布場所：http://ppopenhpc.cc.u-tokyo.ac.jp/
使用パッケージ：ppOpen-MATH/VIS ver.0.2.0
ドキュメントや詳しいインストール方法等は公開パッケージをダウンロード。

本講習会では，`2.with_vis/ppohVIS/`ディレクトリにインストール済みで，ソースコードにも実装済み。

ppOpen-MATH/VIS用入力ファイル：`control.dat`

```
[Refine]                    細分化制御情報セクション
AvailableMemory = 2.0       利用可能メモリ容量（GB）not in use
MaxVoxelCount = 10000       Max Voxel #
MaxRefineLevel = 20         Max Voxel Refinement Level
[Simple]                    簡素化制御情報セクション
ReductionRate = 0.0         表面パッチ削減率
```

ppOpen-MATH/VISライブラリの実装例

1.original/main.f90

```
program thermal_conduction_3D
  use global_var
  implicit none
  integer::nt,nstep
  double precision::mtime
  call start_parallel
  call load_param
  call allocate_array
  call initial_conditions
```

⋮

2.with_vis/main.f90

```
program thermal_conduction_3D
  use global_var

  !C-PPOHVIS-s
  USE PPOHVIS_FDM3D_UTIL
  !C-PPOHVIS-e

  implicit none
  integer::nt,nstep
  double precision::mtime

  !C-PPOHVIS-s
  integer::nx,ny,nz,ierr,L
  TYPE(PPOHVIS_BASE_STCONTROL) PCONTROL
  TYPE(PPOHVIS_FDM3D_STSTRGRID) PSTRGRID
  TYPE(PPOHVIS_BASE_STRESULTCOLLECTION) PRESELM,PRESNOD
  CHARACTER(LEN=PPOHVIS_BASE_FILE_NAME_LEN) CTLNAME,UCDHEAD
  !C-PPOHVIS-e

  call start_parallel
  call load_param
  call allocate_array
  call initial_conditions
```

⋮

オリジナルのサンプルコード（左）と比較すると，ppOpen-MATH/VISライブラリ用の宣言やサブルーチンコール等が増えていることが分かる。

2. サンプルコード with ppOpen-MATH/VISの実行

1. 2.with_vis/の中のcompileファイルを実行

```
% ./compile
```

2. すると，run/の中に実行ファイルwith_visが生成。runに移動。

```
% cd run
% ls
control.dat  input.dat  job.sh  results/  with_vis
```

ppOpen-MATH/VIS用
入力ファイル

実行ファイル

3. バッチジョブ実行で計算

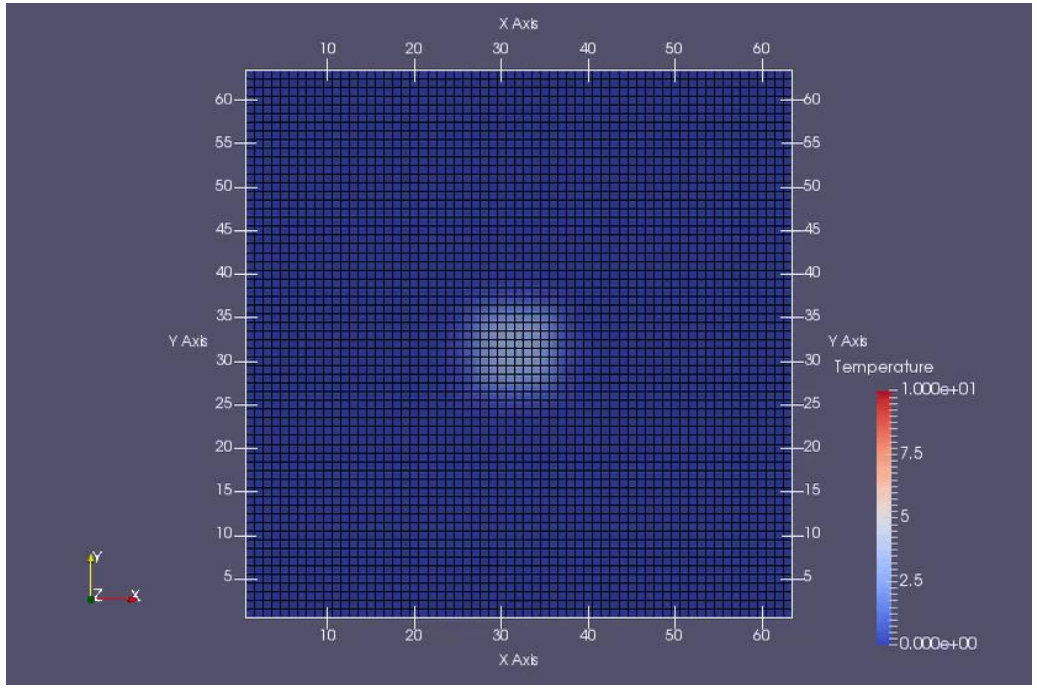
```
% pjsub job.sh
```

4. 実行が終了したらresults/の中にinpファイルが出力されていることを確認

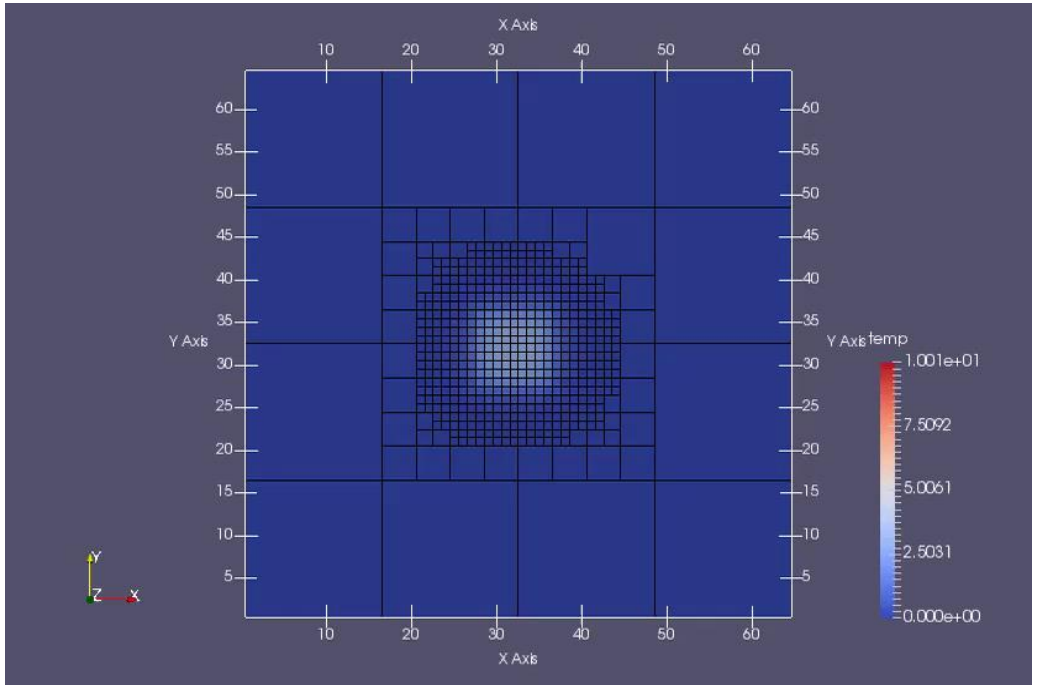
演習2

1. 計算結果の出力ファイル（out_temperature.1.inp～out_temperature.10.inp）を自分のPCにダウンロードして，Paraviewで可視化（スナップショット，アニメーション等）しなさい。
2. ppOpen-MATH/VISの入力ファイル`control.dat`を自由に書き換えて計算しなさい。

オリジナル



with ppOpen-MATH/VIS



プロファイラの利用

自分でプログラムを実装し，実行，可視化ができるようになってくると，より計算効率のよいコードとなるように，コードの最適化を行いたい場合がある。

プロファイラは性能解析ツールであり，関数呼び出しの頻度や，それにかかる時間，ハードウェアの情報等，プログラム実行時の各種情報を収集する。つまり，プロファイラを利用することで，コードの中で実行時間のボトルネックとなっている部分を洗い出すことが可能となる。

Oakleaf-FXでは予め富士通製プロファイラがインストールされており，本講習会ではそれを利用する。他のプロファイラとは，プログラムの実装方法などが異なるが，プロファイラの使い方という意味では，概ね変わらない。

Oakleaf-FXで利用できるプロファイラ(1/2)

＊詳しい利用方法についてはポータルにあるドキュメント参照
https://oakleaf-www.cc.u-tokyo.ac.jp/
(Oakleaf-FX利用者支援ポータル→ドキュメント閲覧→プロファイラ利用手引書)

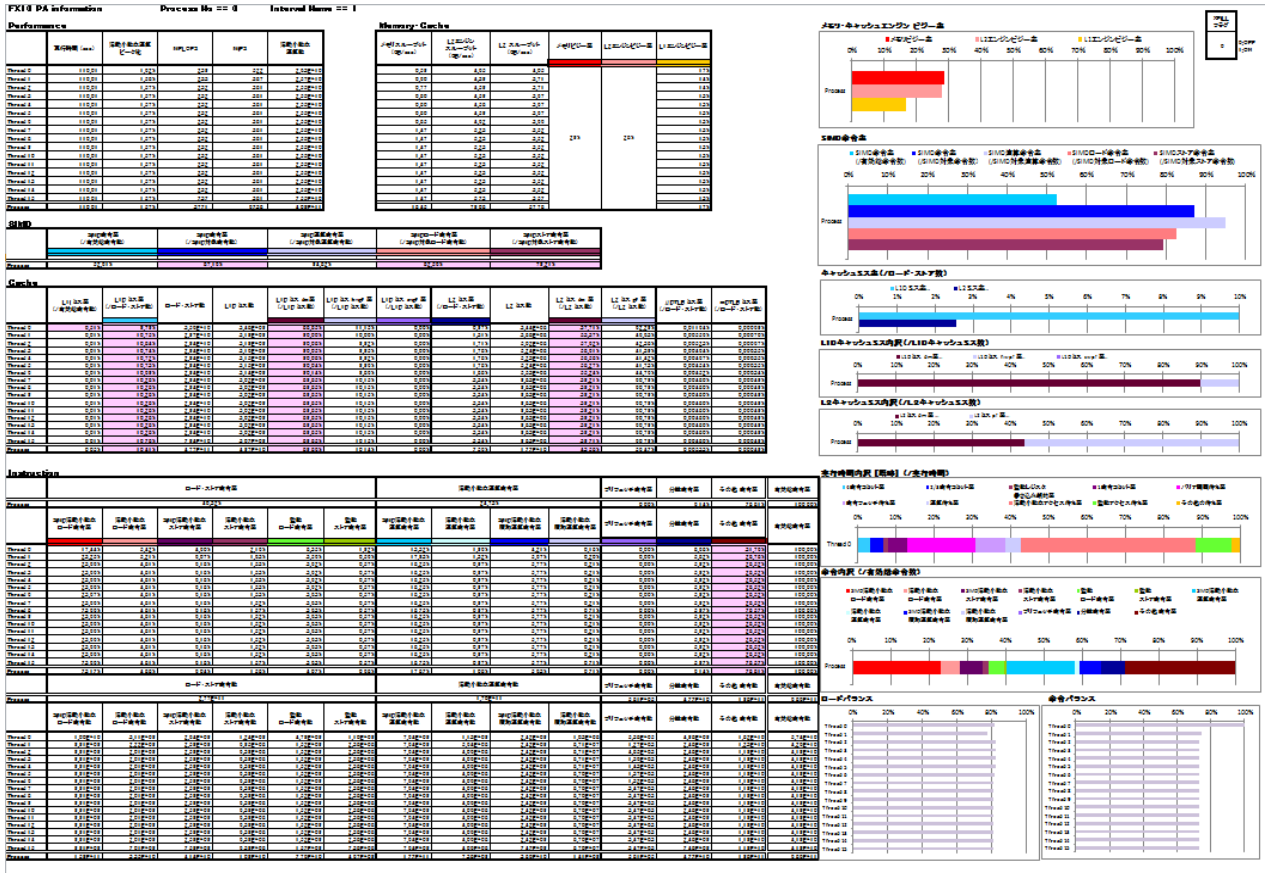
- 基本プロファイラ(GUI(Windows/Mac) or テキスト/CSV出力)
- 詳細プロファイラ(GUI(Windows/Mac) or テキスト/CSV出力)
  - 1回の実行で，測定区間の基本情報(呼び出し回数，経過時間，CPU時間)，MPI情報，ハードウェアモニタ情報を収集。
- 精密PA可視化(Excel使用(Windows/Mac))
  - 7回の実行で，パフォーマンス情報，メモリ情報，SIMD情報，キャッシュ情報，CPU時間情報，命令情報，バランス情報，時間情報を，プロセス毎かつ計測区間毎に収集。

Oakleaf-FXで利用できるプロファイラ(2/2)

詳細プロファイラ(テキスト出力)

```
Fujitsu Advanced Performance Profiler Version 1.2.0
   Measured time                 : Mon Sep  5 02:37:45 2016
   CPU frequency                 : Process      0 -      7  1848 (MHz)
   Type of program               : MPI & Thread (OpenMP)  16
   Virtual coordinate            : (8, 0, 0)

Basic profile

    ********************************************************************************
        Application
    ********************************************************************************

      Kind    Elapsed(s)      User(s)    System(s)        Call
    ------------------------------------------------------------
       AVG       11.8680     176.2937        1.0675      1.0000    all 0
       MAX       11.8736     176.9500        1.2700           1
       MIN       11.8547     175.9200        0.6100           1

      Kind    Elapsed(s)      User(s)    System(s)        Call
    ------------------------------------------------------------
       AVG        8.4638     135.4125        0.0150   1000.0000    kernel 2
       MAX        9.6835     155.0100        0.1100        1000
       MIN        0.1992       2.9300        0.0000        1000

      Kind    Elapsed(s)      User(s)    System(s)        Call
    ------------------------------------------------------------
       AVG        0.1221       1.8350        0.0550      1.0000    pre 1
       MAX        0.1342       1.9900        0.0900           1
       MIN        0.1069       1.6400        0.0300           1

      Kind    Elapsed(s)      User(s)    System(s)        Call
    ------------------------------------------------------------
       AVG        1.3966      22.3075        0.0288   1000.0000    output 3
       MAX       10.6202     170.0100        0.2200        1000
       MIN        0.0347       0.3700        0.0000        1000

    ********************************************************************************
```

精密PA可視化(Excel使用(Windows/Mac))



プロファイラの計測範囲の設定例（詳細プロファイラ）

1.original/main.f90

```
program thermal_conduction_3D
  use global_var
  implicit none
  integer::nt,nstep
  double precision::mtime
  call start_parallel
  call load_param
  call allocate_array
  call initial_conditions
  mtime=0.0d0; nstep=0
  do nt=1,ntmax
     mtime=mtime+dt
     call kernel
     if(mod(nt,nfreq)==0) then
```

⋮

精密PA可視化（Excel使用）の場合（4.with_prof_excel/main.f90）もサブルーチン名と引数（文字列のみ）は異なるが同様に実装。

3.with_prof_txt/main.f90

```
program thermal_conduction_3D
  use global_var
  implicit none
  integer::nt,nstep
  double precision::mtime

  call fapp_start("pre",1,0)
  call start_parallel
  call load_param
  call allocate_array
  call initial_conditions
  call fapp_stop("pre",1,0)

  mtime=0.0d0; nstep=0
  do nt=1,ntmax
     mtime=mtime+dt

     call fapp_start("kernel",2,0)
     call kernel
     call fapp_stop("kernel",2,0)

     if(mod(nt,nfreq)==0) then
```

⋮

計測したい範囲（計測区間）を専用サブルーチンで囲う。

引数の1つ目の文字列（任意）と2つめの数字は囲う範囲で共通，3つ目の数字はゼロにしてください。

3. サンプルコード with 詳細プロファイラ(テキスト出力)の実行

1. 3.with_prof_txt/の中のcompileファイルを実行

```
% ./compile
```

2. すると, run/の中に実行ファイルwith_prof_txtが生成。run/に移動。

```
% cd run
% ls
conv input.dat job.sh prof_data/ results/ with_prof_txt
```

プロファイルデータ変換用実行ファイル

プロファイルデータ出力ディレクトリ

実行ファイル

3. バッチジョブ実行で計算

```
% pjsub job.sh
```

4. 実行が終了したら, prof_data/のプロファイルデータをテキストへ変換するために, convファイルを実行, するとoutput_prof.txtが生成される。

```
% ./conv
```

4. サンプルコード with 精密PA可視化（Excel使用）の実行（1/2）

1. 4.with_prof_excel/の中のcompileファイルを実行

```
% ./compile
```

2. すると，run/の中に実行ファイルwith_prof_excelが生成。run/に移動。

```
% cd run
% ls
conv  FSDT_CPUPA.xlsm  input.dat  job.sh  pa1/  pa2/  pa3/
pa4/  pa5/  pa6/  pa7/  results/  with_prof_excel
```

3. バッチジョブ実行で計算

```
% pjsub job.sh
```

4. 実行が終了したらconvファイルを実行。するとoutput_prof_1.csv～output_prof_7.csvが生成される。

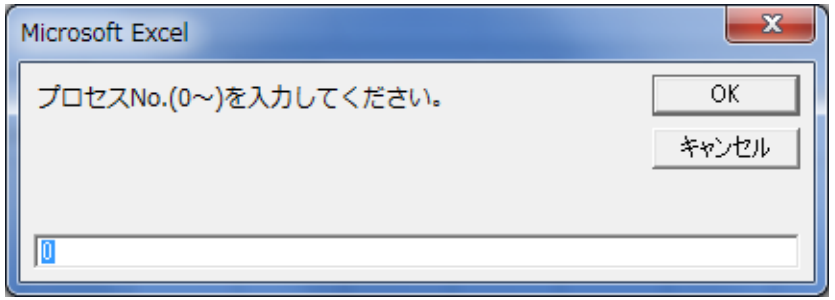
```
% ./conv
```

4. サンプルコード with 精密PA可視化（Excel使用）の実行（2/2）

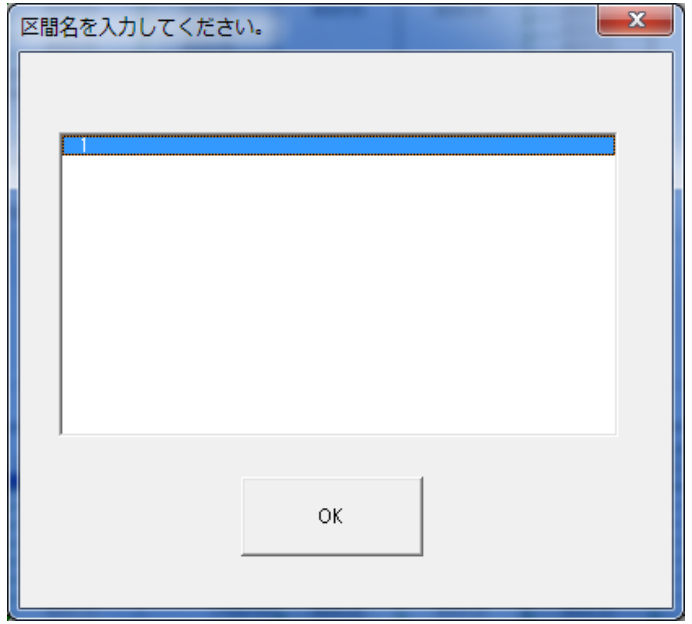
5. output_prof_1(～7).csvとFSDT_CPUPA.xlsmファイルを自分のPCの同じフォルダダウンロード。

6. FSDT_CPUPA.xlsmをダブルクリックしてExcelを開くと，以下の図のように，プロセス番号，計測区間の順にそれらの指定についてのダイアログが開くので，それぞれ見たいプロセスと計測区間を指定する。

プロセス番号指定ダイアログ



計測区間指定ダイアログ



演習3

1. 詳細プロファイラ(テキスト出力)，または精密PA可視化(Excel使用)により，プロファイルを収集・観測しなさい。(Macの最新のoffice，またはLinuxでは精密PA可視化は使えません。)

2. ソースファイルmain.f90を書き換え，計測区間を自由に変更してプロファイルを収集しなさい。

```
Fujitsu Advanced Performance Profiler Version 1.2.0
   Measured time              : Mon Sep  5 02:37:45 2016
   CPU frequency              : Process      0 -      7  1848 (MHz)
   Type of program            : MPI & Thread (OpenMP)  16
   Virtual coordinate         : (8, 0, 0)

Basic profile

    ********************************************************************************************
        Application
    ********************************************************************************************

      Kind    Elapsed(s)      User(s)    System(s)        Call

       AVG       11.8680     176.2937        1.0675      1.0000    all 0
       MAX       11.8736     176.9500        1.2700           1
       MIN       11.8547     175.9200        0.6100           1

      Kind    Elapsed(s)      User(s)    System(s)        Call

       AVG        8.4638     135.4125        0.0150   1000.0000    kernel 2
       MAX        9.6835     155.0100        0.1100        1000
       MIN        0.1992       2.9300        0.0000        1000

      Kind    Elapsed(s)      User(s)    System(s)        Call

       AVG        0.1221       1.8350        0.0550      1.0000    pre 1
       MAX        0.1342       1.9900        0.0900           1
       MIN        0.1069       1.6400        0.0300           1

      Kind    Elapsed(s)      User(s)    System(s)        Call

       AVG        1.3966      22.3075        0.0288   1000.0000    output 3
       MAX       10.6202     170.0100        0.2200        1000
       MIN        0.0347       0.3700        0.0000        1000

    ********************************************************************************************
```