並列有限要素法による
三次元定常熱伝導解析プログラム
並列可視化

中島 研吾
東京大学情報基盤センター

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

中島研吾
東京大学情報基盤センター

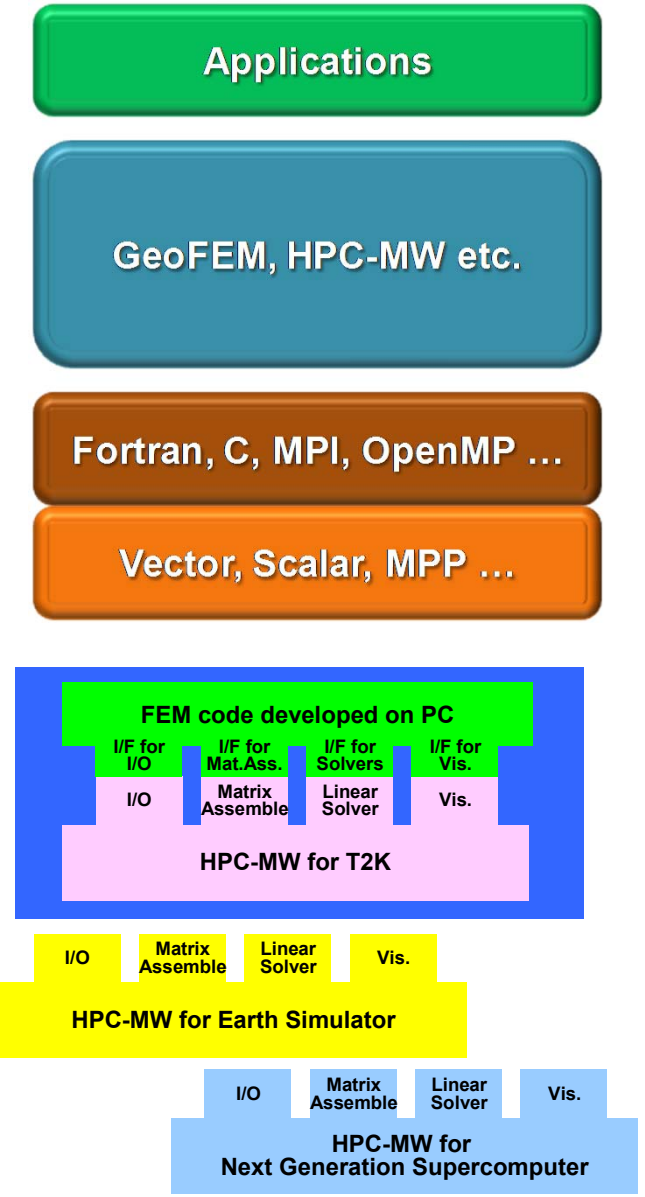
佐藤正樹（東大・大気海洋研究所），奥田洋司（東大・新領域創成科学研究科），
古村孝志（東大・情報学環/地震研），岩下武史（北大・情報基盤センター），
阪口秀（海洋研究開発機構）

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

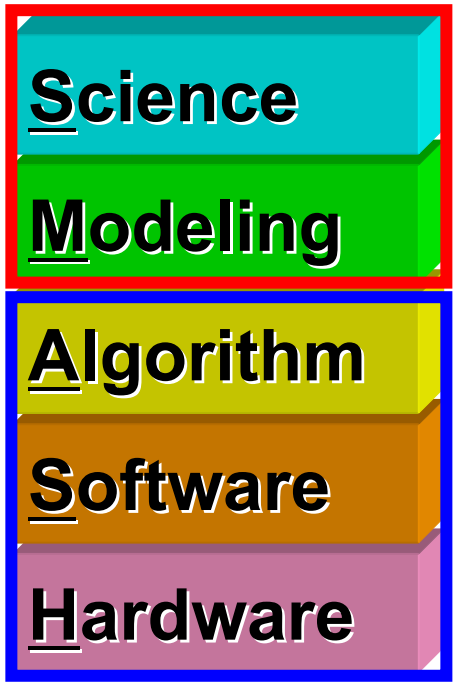
背　景(2/2)

- 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



HPCミドルウェア：何がうれしいか

- アプリケーション開発者のチューニング（並列，単体）からの解放
  - **SM**ASHの探求に専念
    - 一生SMASHと付き合うのはきつい
  - SM**ASH**をカバー
- コーディングの量が減る
- 教育にも適している
- 問題点
  - ハードウェア，環境が変わるたびに最適化が必要となる



ppOpen-HPC

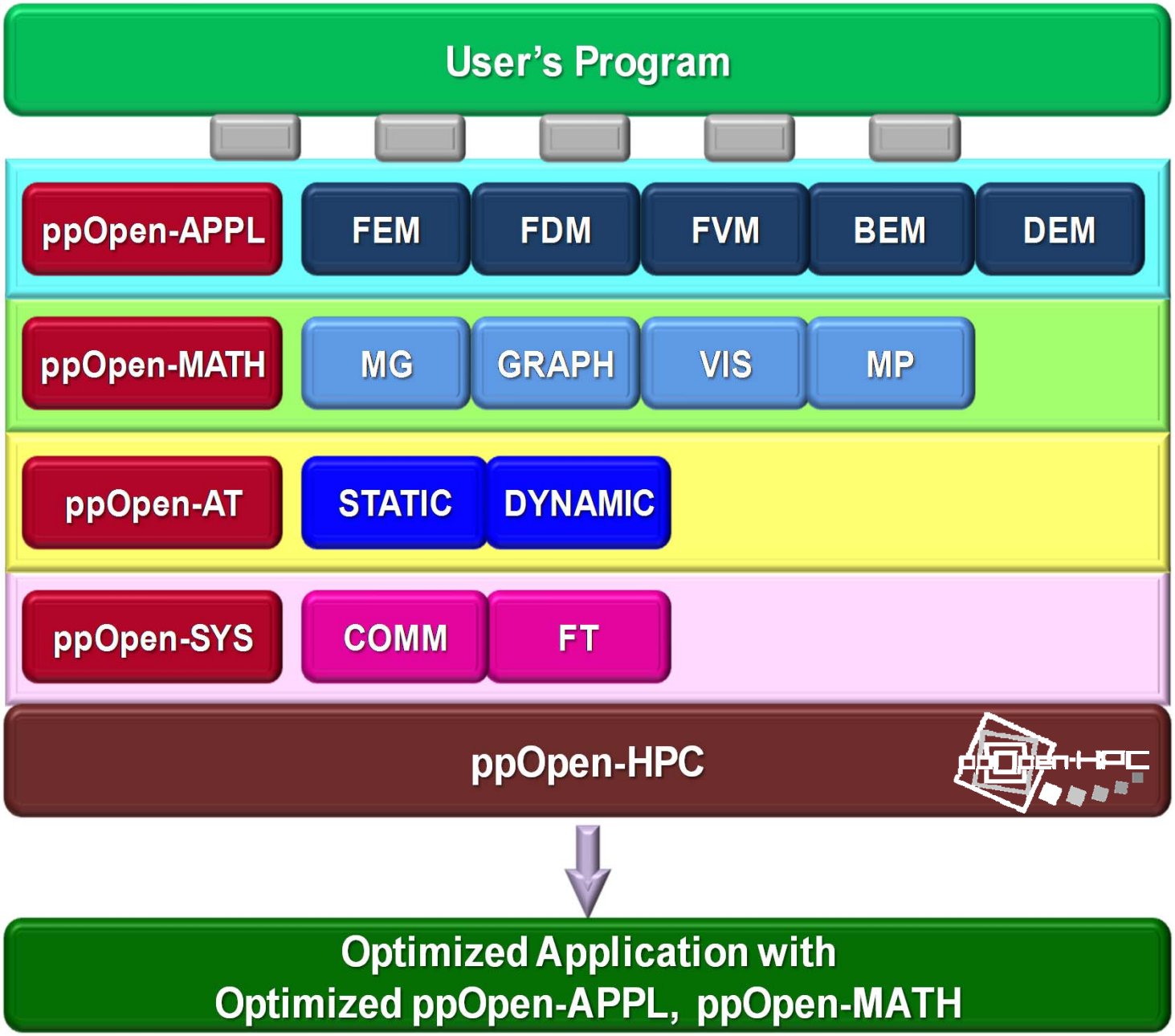
- ppOpen-HPCはJST戦略的創造研究推進事業CREST研究領域「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出（研究総括：佐藤三久（理研AICS））」の一環として「自動チューニング機構を有するアプリケーション開発・実行環境」において**2011年度から5年計画で開発されている**オープンソースフレームワーク（pp=post peta）であり，
- メニィコアアーキテクチャに基づくポストペタスケールシステムの処理能力を充分に引き出す科学技術アプリケーションの効率的な開発，安定な実行に資するものである。
- 対象離散化手法を有限要素法等の5手法に限定し，各手法の特性に基くアプリケーション開発用ライブラリ群，実行環境を提供する。
- 本研究は2016年度に筑波大学，東京大学共同で導入予定のポストT2Kシステム（ピーク性能30PFLOPS以上）をターゲットとし，東大情報基盤センタースパコンの2,000人以上の利用者の新システムへの円滑な移行を支援する。

Framework Appl. Dev.

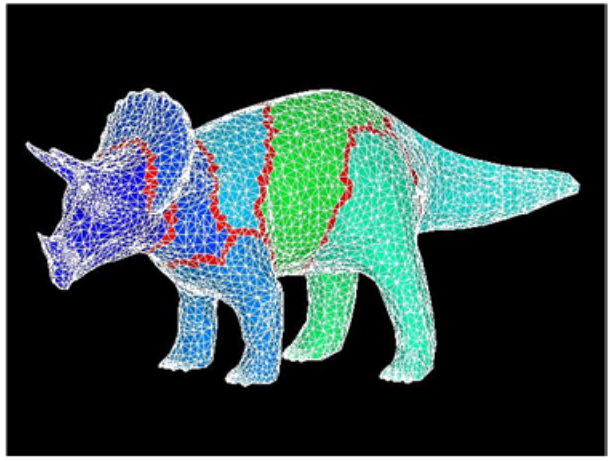
Math Libraries

Automatic Tuning (AT)

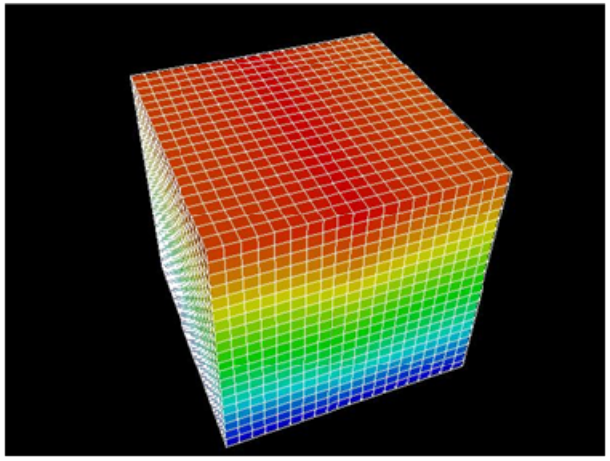
System Software



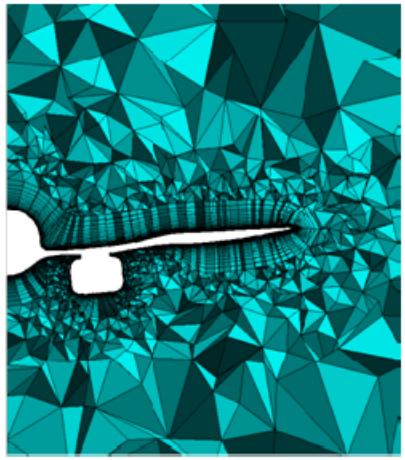
ppOpen-HPC covers …



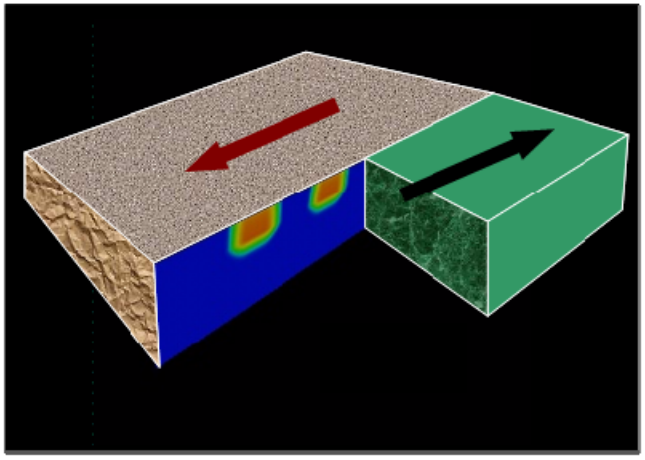
FEM
Finite Element Method



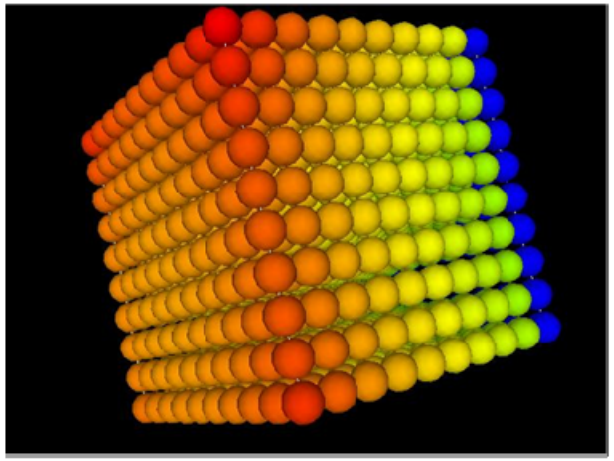
FDM
Finite Difference Method



FVM
Finite Volume Method



BEM
Boundary Element Method



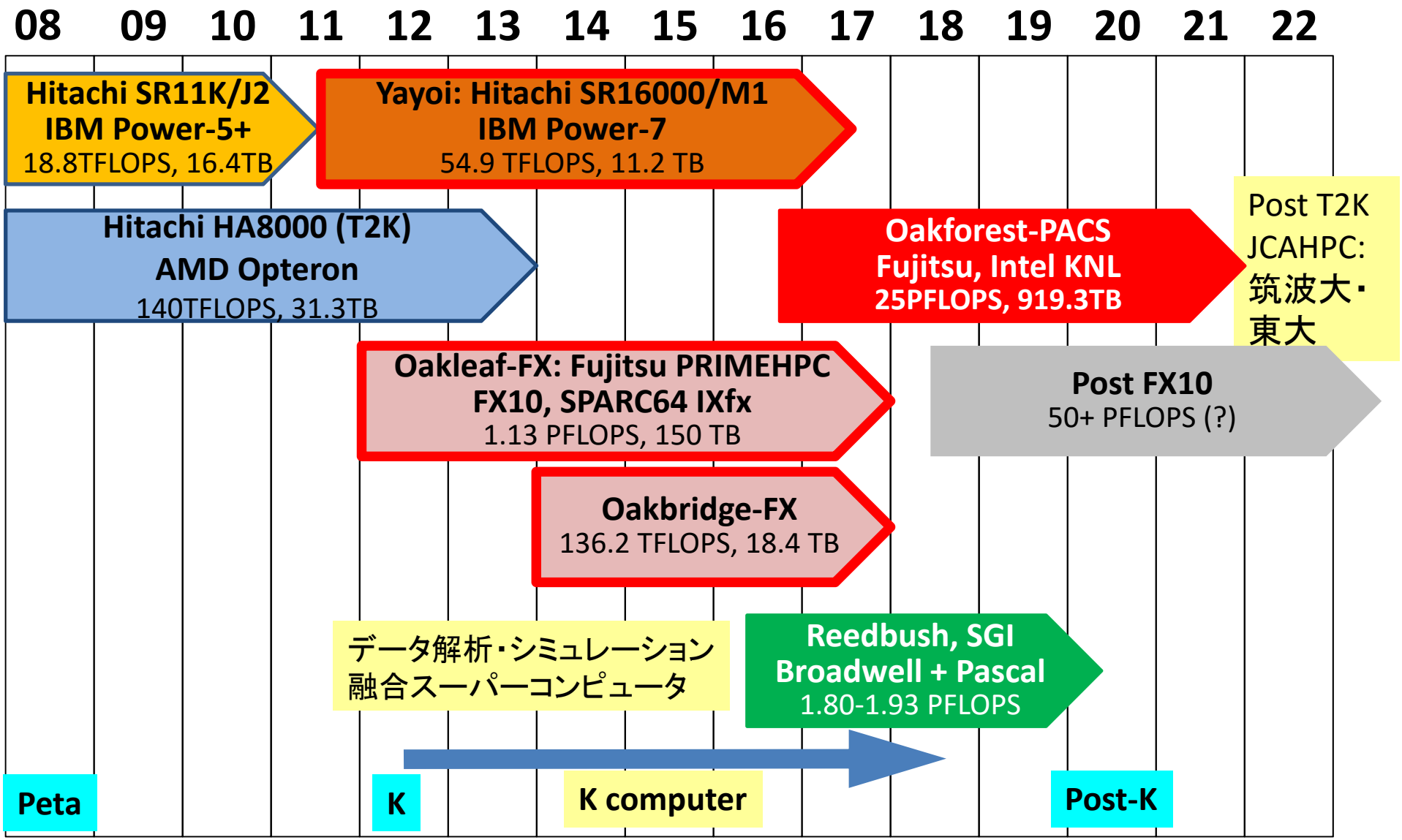
DEM
Discrete Element Method

ppOpen-HPC

- ppOpen-HPCはJST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（研究総括：佐藤三久（理研AICS））」の一環として「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」において2011年度から5年計画で開発されているオープンソースフレームワークであり，
- メニィコアアーキテクチャに基づく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ものである。
- 対象離散化手法を有限要素法等の5手法に限定し，各手法の特性に基くアプリケーション開発用ライブラリ群，実行環境を提供する。
- 本研究は2016年度に筑波大学，東京大学共同で導入予定のポストT2Kシステム（ピーク性能30PFLOPS以上）をターゲットとし，東大情報基盤センタースパコンの2,000人以上の利用者の新システムへの円滑な移行を支援する。

東大センターのスパコン

2基の大型システム，6年サイクル



ppOpen-HPCのターゲット ポストT2K

- ポストT2K
  - ピーク性能30 PFLOPS超，（多分）2016年秋稼働開始
    - ✓最先端共同HPC基盤施設（JCAHPC，Joint Center for Advanced High Performance Computing）：筑波大学，東京大学
    - ✓http://jcahpc.jp/
  - メニーコアベース（Intel MIC/Xeon Phi等）
    - ✓MPI + OpenMP + X
  - ppOpen-HPCは利用者（東大センター：2,000人）の新システムの円滑な移行を促進する
- 京/FX10, Cray, Xeonクラスターも視野

筑波大学
University of Tsukuba

JCAHPC

東京大学
THE UNIVERSITY OF TOKYO

ポストT2K：Oakforest-PACS

http://www.cc.u-tokyo.ac.jp/system/ofp/

- 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

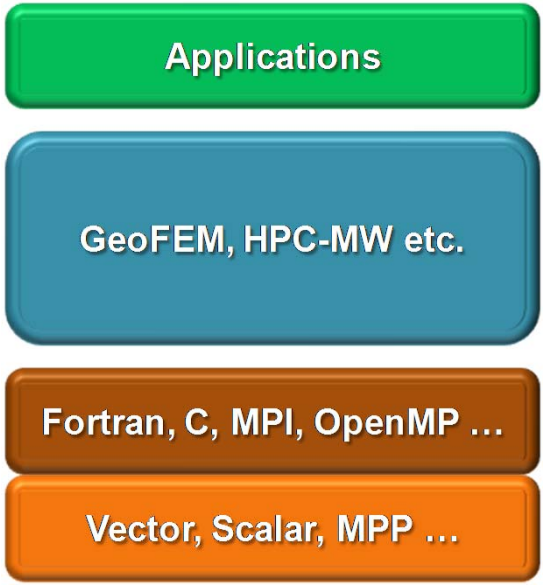
東京大学 THE UNIVERSITY OF TOKYO　JCAHPC　筑波大学 University of Tsukuba

ppOpen-HPCの精神

- ppOpen-HPCでは北大・東大・京大の各大型計算機センターと，地球科学（東大（大気海洋研，地震研），JAMSTEC），工学（東大（新領域））分野の密接な協力のもとで，実用的な大規模アプリケーション開発フレームワーク構築を目指している。
  - 様々な分野の専門家によるCo-Design
- 地球シミュレータ，京コンピュータ，T2K等で既に稼働実績のある大規模シミュレーションコードの共通機能を取り出し，次世代システム向けの最適化，自動チューニング機構の適用を実施している。
- 開発成果はソースレベルで公開すると共に，東大情報基盤センター，PCクラスタコンソーシアム（PCクラスタ実用アプリケーション部会）共催により，産業利用も目指した講習会を開催し普及を図っている。

背景：前世紀末における知見

- FEM等の科学技術計算手法はいくつかの典型的な計算プロセスを含む
  - 各プロセスの最適化は可能であり，アプリ開発において有用
- データ構造を適切に設計すれば，並列計算における通信の「隠蔽」が可能
  - 並列有限要素法におけるHalo
- GeoFEM，HPC-MW等のフレームワークによってアプリ開発者を最適化，並列化から解放することが可能
- チューニングの継承が問題



ppOpen-HPC開発方針

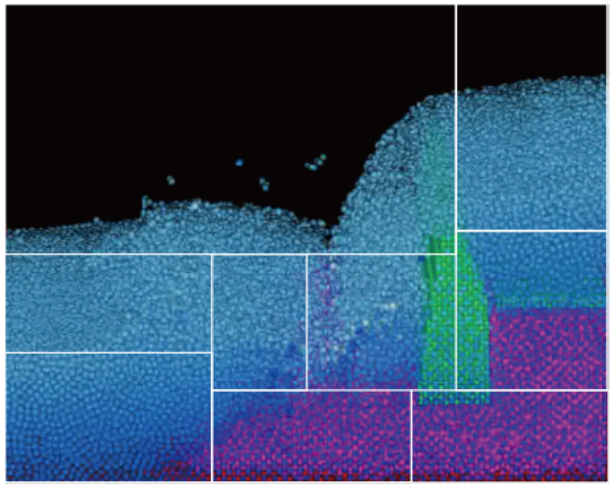
- 先行研究において各メンバーが開発した大規模アプリケーションに基づきppOpen-APPLの各機能を開発，実装
  - 各離散化手法の特性に基づき開発・最適化
    - 共通データ入出力インタフェース，領域間通信，係数マトリクス生成
    - 離散化手法の特性を考慮した前処理付き反復法
    - 適応格子，動的負荷分散
  - 実際に動いているアプリケーションから機能を切り出す
  - 各メンバー開発による既存ソフトウェア資産の効率的利用
    - GeoFEM，HEC-MW，HPC-MW，DEMIGLACE，ABCLibScript
- ppOpen-ATはppOpen-APPLの原型コードを対象として研究開発を実施し，その知見を各ppOpen-APPLの開発，最適化に適用
  - 自動チューニング技術により，様々な環境下における最適化ライブラリ・アプリケーション自動生成を目指す

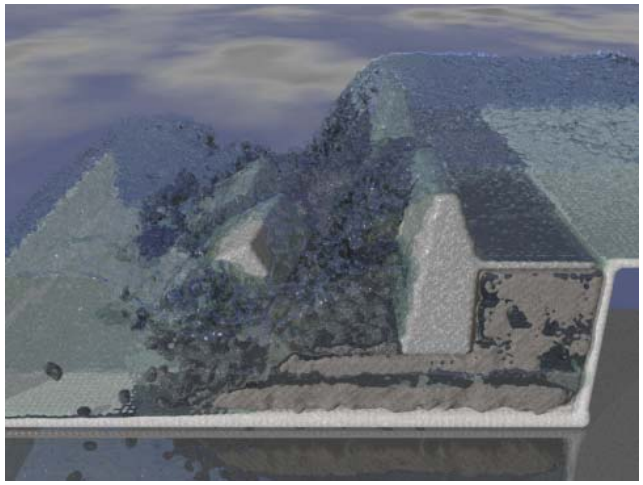
既存フレームワークとの相違

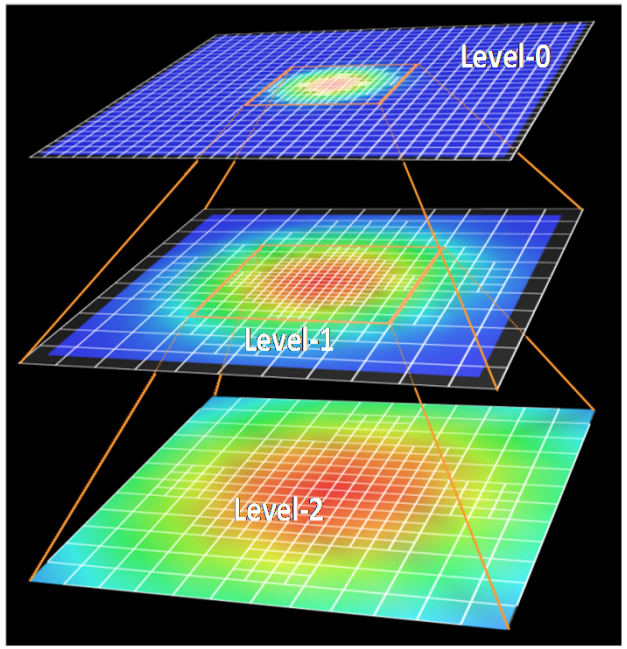
- 5種類の離散化手法に対応
  - 連成計算も可能
- 自動チューニング（AT）の採用
  - チューニング技術の継承
  - アーキテクチャの（ものすごく大きくない）変化に自動的に対応
  - 問題サイズ等の細かい設定に最適なコードを生成できる

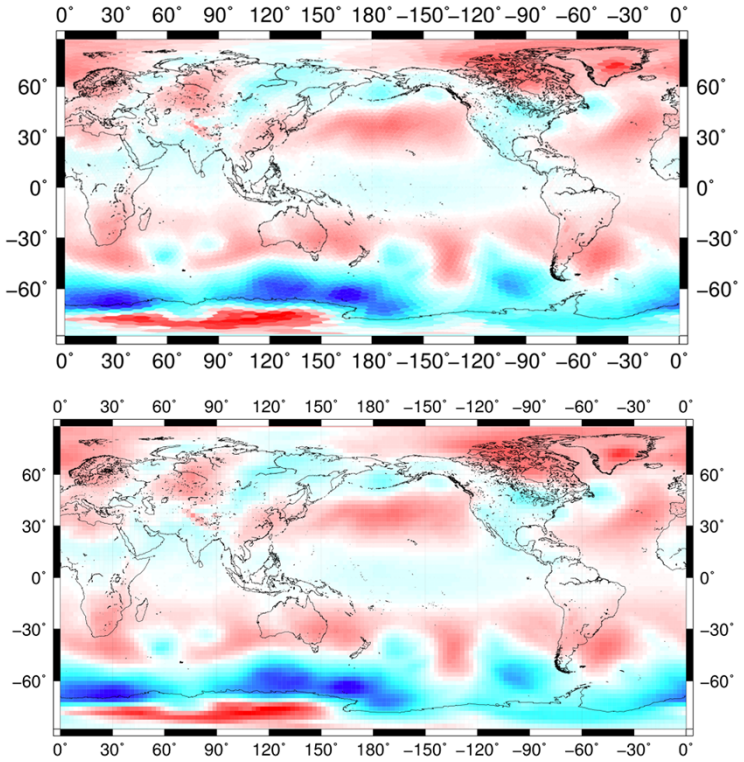
ターゲットアプリケーション

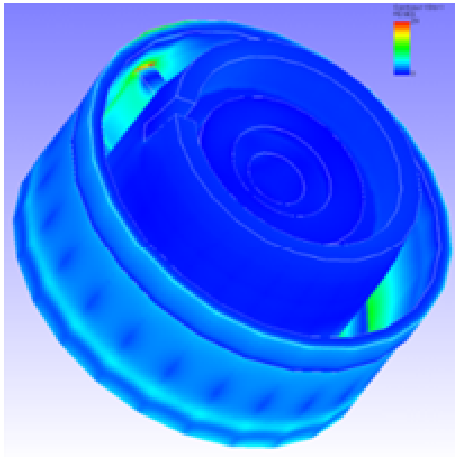
- 本プロジェクトの目的はアプリ開発ではないが，検証用，潜在ユーザー向けにターゲットアプリを設定
- ppOpen-APPL/FEM
  - 非圧縮性流体，熱伝導（定常，非定常），固体力学（動的，静的）
- ppOpen-APPL/FDM
  - 非圧縮性流体，熱伝導（非定常），固体力学（動的）
- ppOpen-APPL/FVM
  - 圧縮性流体，熱伝導（定常）
- ppOpen-APPL/BEM
  - 電磁気学，固体力学（準静的）：地震発生サイクル
- ppOpen-APPL/DEM
  - 非圧縮性流体，固体力学（動的）

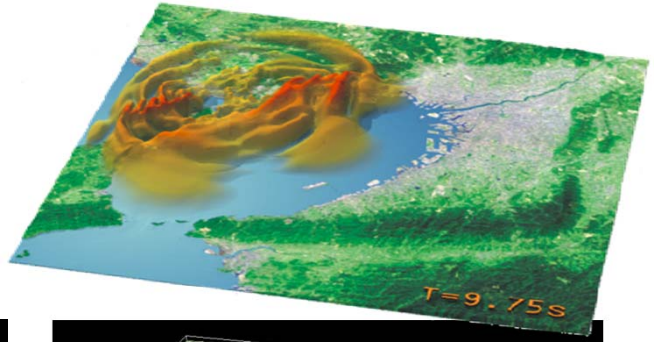


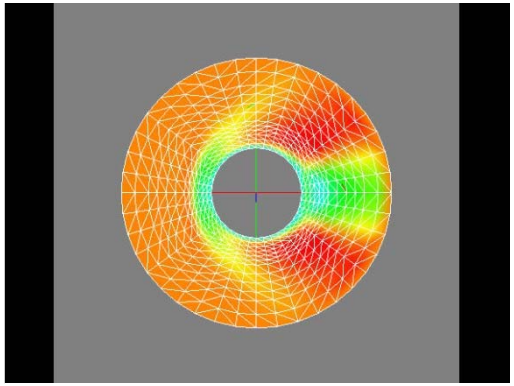












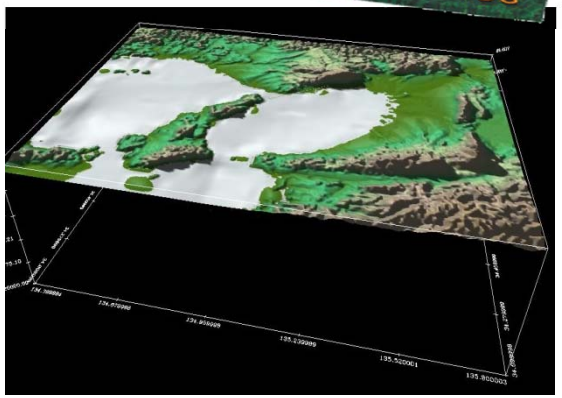
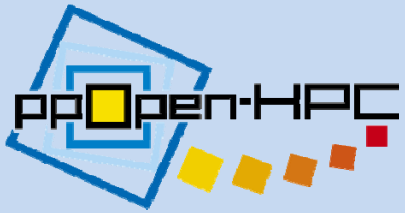


図 1　NICAM と IO モジュールの海面気圧



コード公開スケジュール

(英語ドキュメント付き，**MIT**ライセンス)

http://ppopenhpc.cc.u-tokyo.ac.jp/

- 毎年のSC-XYで更新，公開
- Flat MPI，OpenMP/MPIハイブリッド並列
- Multicore/Manycoreクラスタ向け→Xeon Phi最適化

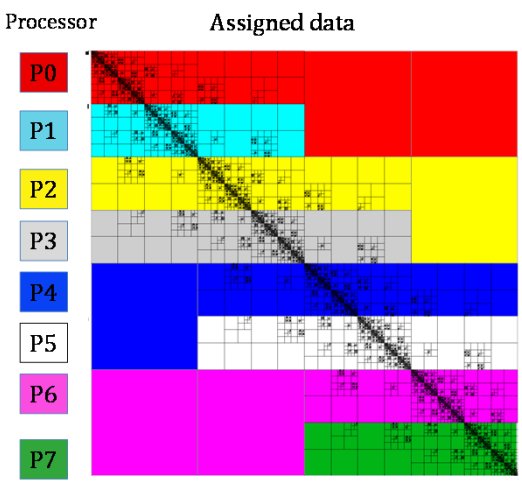
公開の履歴

- SC12, Nov 2012 (Ver.0.1.0)
- SC13, Nov 2013 (Ver.0.2.0)
- SC14, Nov 2014 (Ver.0.3.0)
- SC15, Nov 2015 (Ver.1.0.0)

New Features in Ver.1.0.0

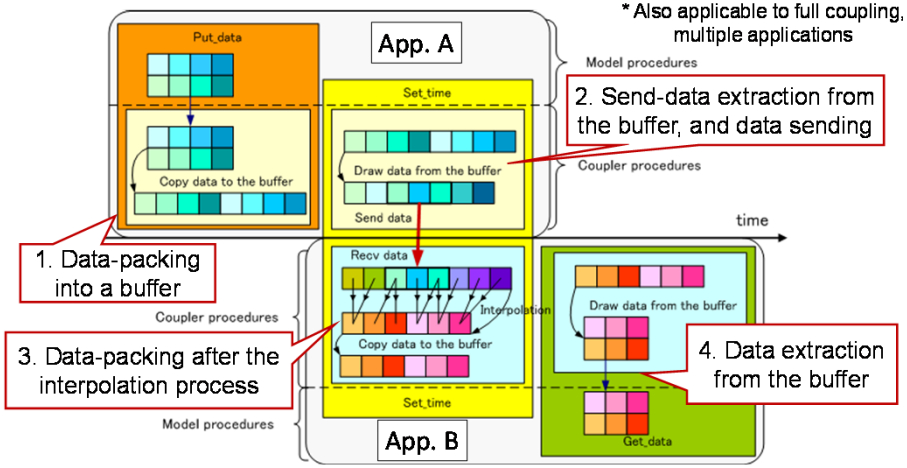
http://ppopenhpc.cc.u-tokyo.ac.jp/

- **HACApK library for H-matrix comp. in ppOpen-APPL/BEM (OpenMP/MPI Hybrid Version)**
  - **First Open Source Library by OpenMP/MPI Hybrid**
- ppOpen-MATH/MP (Coupler for Multiphysics Simulations, Loose Coupling of FEM & FDM)
- Matrix Assembly and Linear Solvers for ppOpen-APPL/FVM



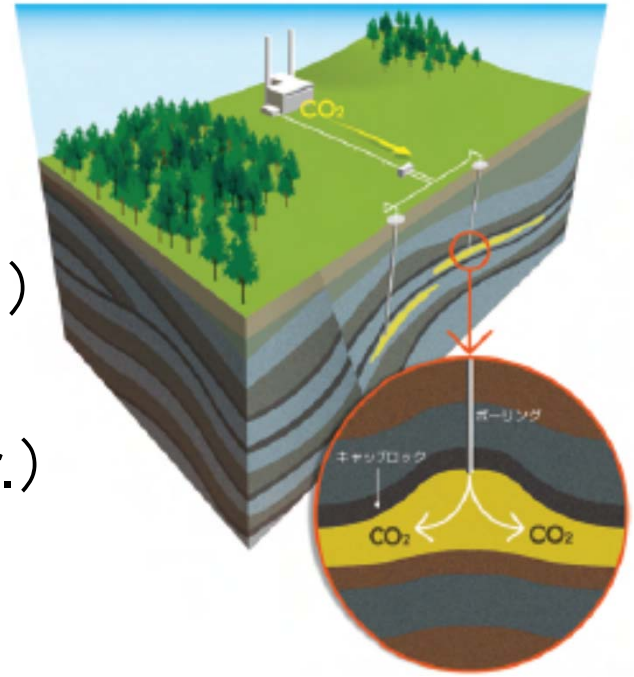
Assign a^c to P_k if $S(a^c{}_{ij})$ in $R(P_k)$

A : Whole matrix
A_{IJ} : The entry in the I-th row and the J-th column of A
a^c: A small submatrix of A
$a^c{}_{ij}$: The entry in the i-th row and the j-th column of a^c
N : Number of rows of A
n : Number of processors
P_k : The k-th processor
l_k : $1 = l_0 < \cdots < l_k < \cdots < l_{n+1} = N + 1$
R : $R(P_k) = \{ i \mid l_k \le i < l_{k+1} \}$
S : $S(a^c{}_{ij}) = I$ when $a^c{}_{ij} = A_{IJ}$



研究協力・普及

- 国際的共同研究
  - Lawrence Berkeley National Lab.
  - 国立台湾大学，国立中央大学（台湾）
  - **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



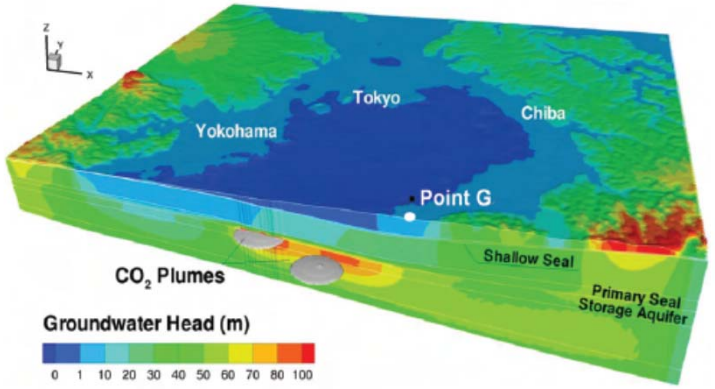


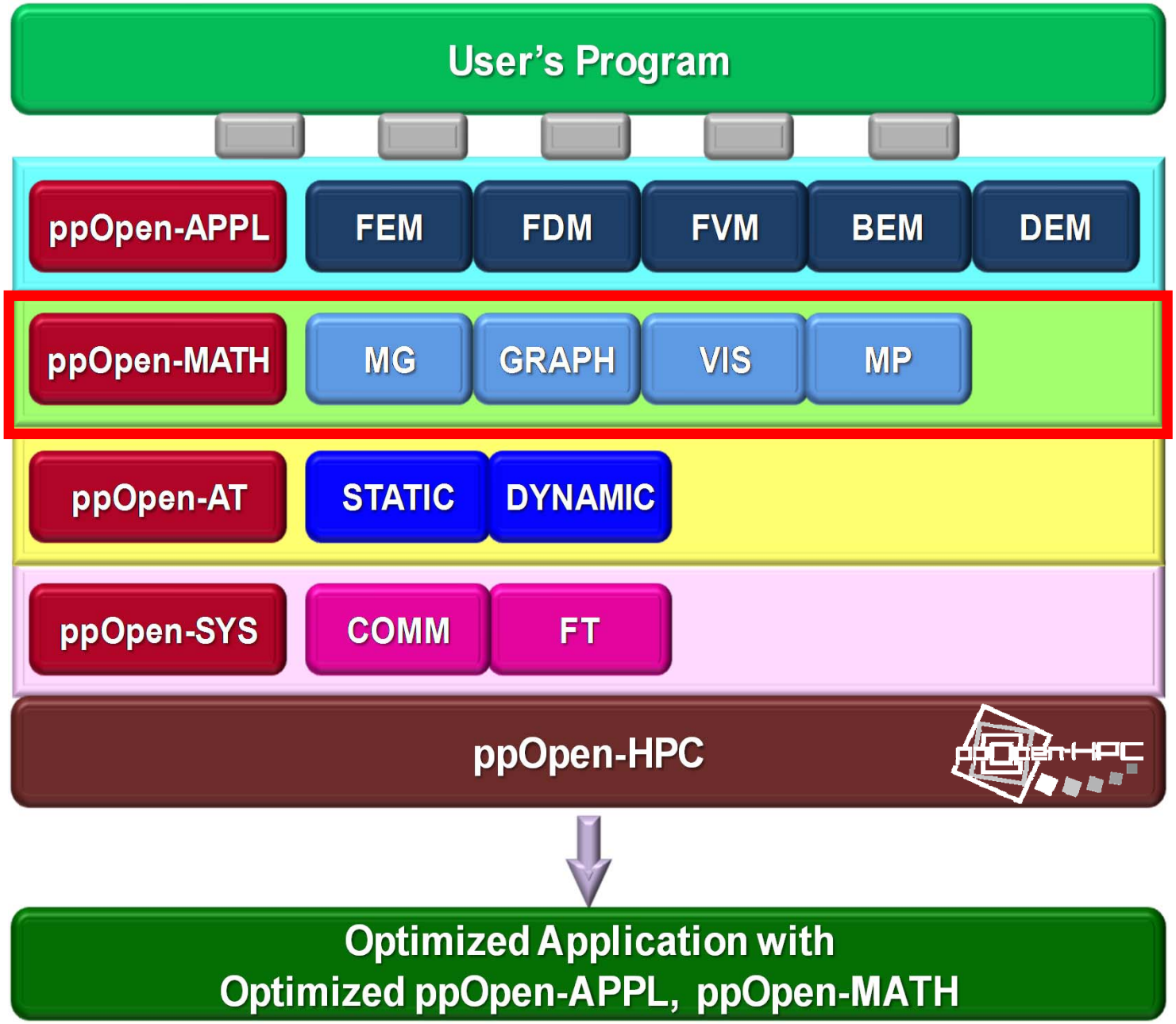
図-4 CO_2 圧入後の地下水圧（全水頭換算）の分布（100 年後）

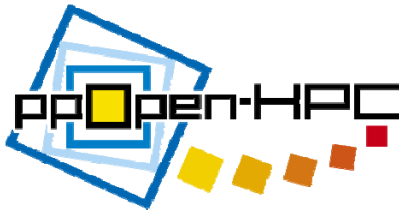
共同研究等事例(1/2)

- ppOpen-AT関連共同研究
  - 工学院大学　田中研究室
    - 田中研究室開発のAT方式(d-spline方式)の適用対象としてppOpen-ATのAT機能を拡張
  - 東京大学　須田研究室
    - 電力最適化のため，須田研究室で開発中のAT方式と電力測定の共通APIを利用し，ppOpen-ATを用いた電力最適化方式を提案〔Katagiri et al. IEEE/MCSoC 2013 Best Paper Award〕
- JHPCN共同研究課題
  - 高精度行列 - 行列積アルゴリズムにおける並列化手法の開発（東大，早稲田大）（H24年度）（研究としては継続）
    - 高精度行列-行列積演算における行列-行列積の実装方式選択に利用
  - 粉体解析アルゴリズムの並列化に関する研究（東大，法政大）（H25年度）
    - 粉体シミュレーションのための高速化手法で現れる性能パラメタのATで利用を検討

共同研究等事例(2/2)

- JHPCN共同研究課題(続き)
  - 巨大地震発生サイクルシミュレーションの高度化(京大，東大他)(H24･25年度)
    - Hマトリクス，領域細分化
  - ポストペタスケールシステムを目指した二酸化炭素地中貯留シミュレーション技術の研究開発(大成建設，東大)(H25年度)
    - 疎行列ソルバー，並列可視化
  - 降着円盤シミュレーション(千葉大，東大)(H22年度～)
    - 疎行列ソルバー，並列可視化
  - 太陽磁気活動の大規模シミュレーション」(東大(地球惑星，情報基盤センター))(H25年度～)
    - 疎行列ソルバー，並列可視化

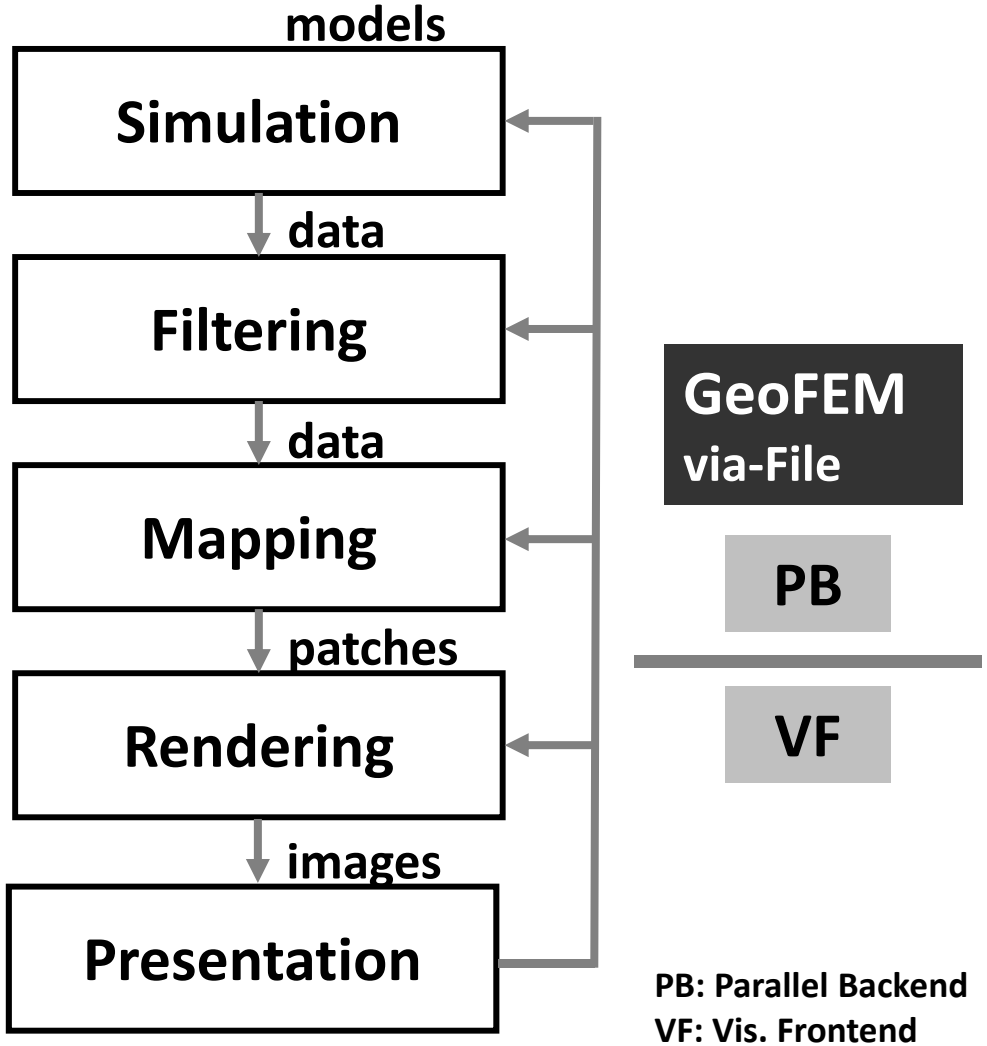

User's Program
ppOpen-APPL
FEM
FDM
FVM
BEM
DEM
ppOpen-MATH
MG
GRAPH
VIS
MP
ppOpen-AT
STATIC
DYNAMIC
ppOpen-SYS
COMM
FT
ppOpen-HPC
Optimized Application with
Optimized ppOpen-APPL, ppOpen-MATH



ppOpen-MATH

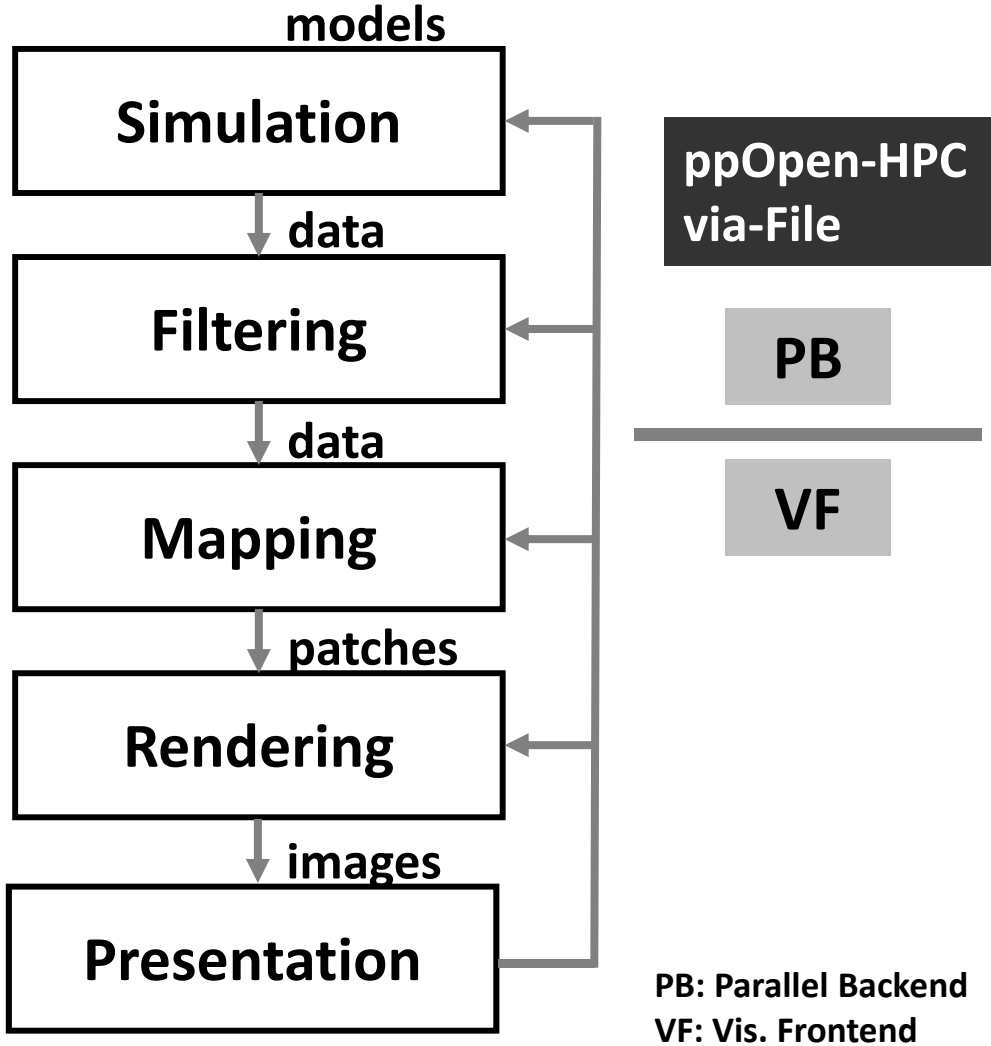
- ppOpen-HPCにおける共通ライブラリ
- ppOpen-MATH/MG
  - 多重格子法ソルバー
- ppOpen-MATH/GRAPH
  - 並列グラフライブラリ（マルチスレッド版）：RCM, MC（開発中）
- ppOpen-MATH/VIS
  - 並列可視化ライブラリ
- ppOpen-MATH/MP
  - 弱連成カプラー

ppOpen-HPCにおける並列可視化の考え方



- 並列計算・並列可視化を並行して同時に実行
- 自己完結的なファイルを生成してPCで見る（e.g. MicroAVS, ParaView）。
- GeoFEMはPatch抽出型で，例えば視点を変えることはできたが，可視化する変数，切り出す面等を変更することはできなかった。
- ピーク（最大，最小），分布を抑えることが大事，形状もある程度再現できていてほしい。

ppOpen-HPCにおける並列可視化の考え方



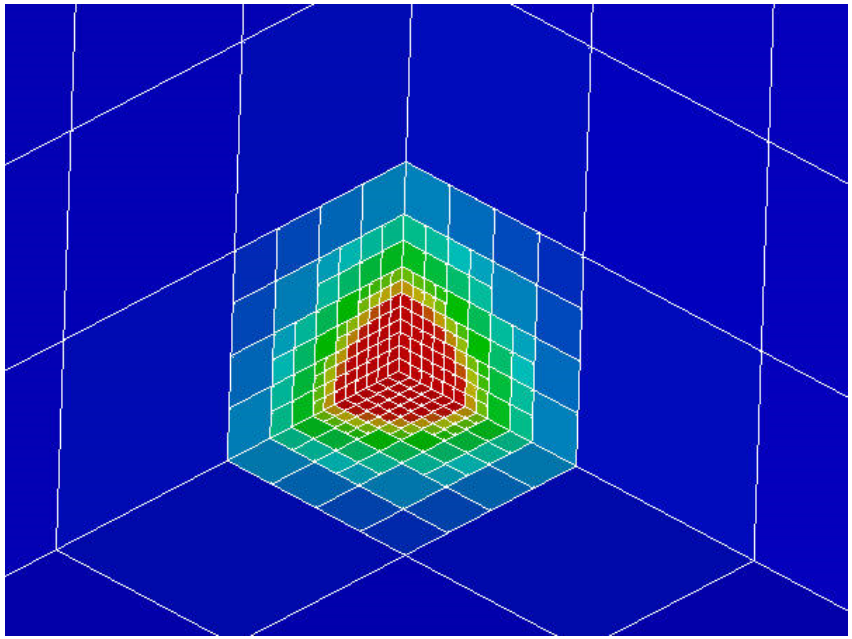
- 並列計算・並列可視化を並行して同時に実行
- 自己完結的なファイルを生成してPCで見る（e.g. MicroAVS, ParaView）。
- GeoFEMはPatch抽出型で，例えば視点を変えることはできたが，可視化する変数，切り出す面等を変更することはできなかった。
- ピーク（最大，最小），分布を抑えることが大事，形状もある程度再現できていてほしい。

並列可視化とスパコン

- 「見る」ためにスパコンは使わない
- 「絵を出すために計算をやり直す」という考え方も採らない
- 大型計算機センターとしては，つぎ込めるだけの予算を計算エンジンにつぎ込みたい

ppOpen-MATH/VIS

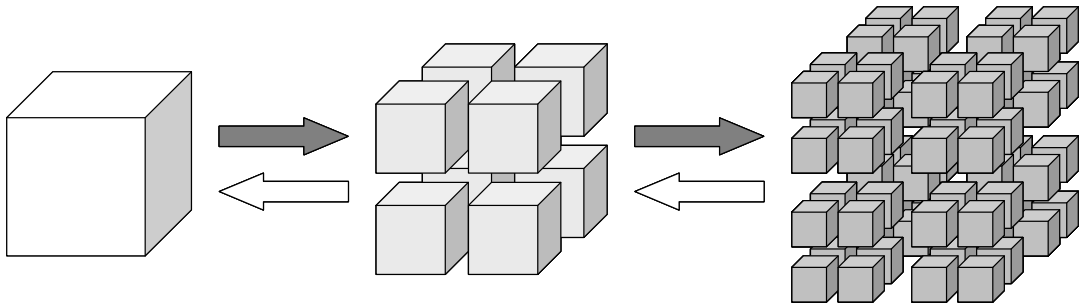
- ボクセル型背景格子を使用した大規模並列可視化手法〔Nakajima & Chen 2006〕に基づく
  - 差分格子用バージョン公開：ppOpen-MATH/VIS-FDM3D
- UCD single file
- プラットフォーム
  - T2K，Cray
  - FX10
  - Flat MPI
    - Hybrid，非構造格子：今年度実施

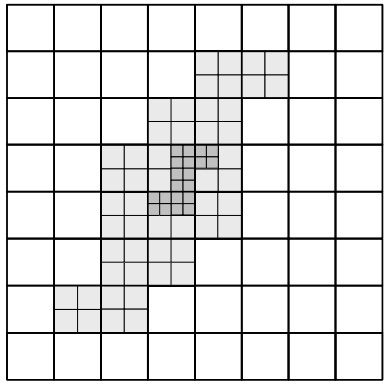


```
[Refine]
AvailableMemory = 2.0    Available memory size (GB), not available in this version.
MaxVoxelCount   = 500    Maximum number of voxels
MaxRefineLevel  = 20     Maximum number of refinement levels
```

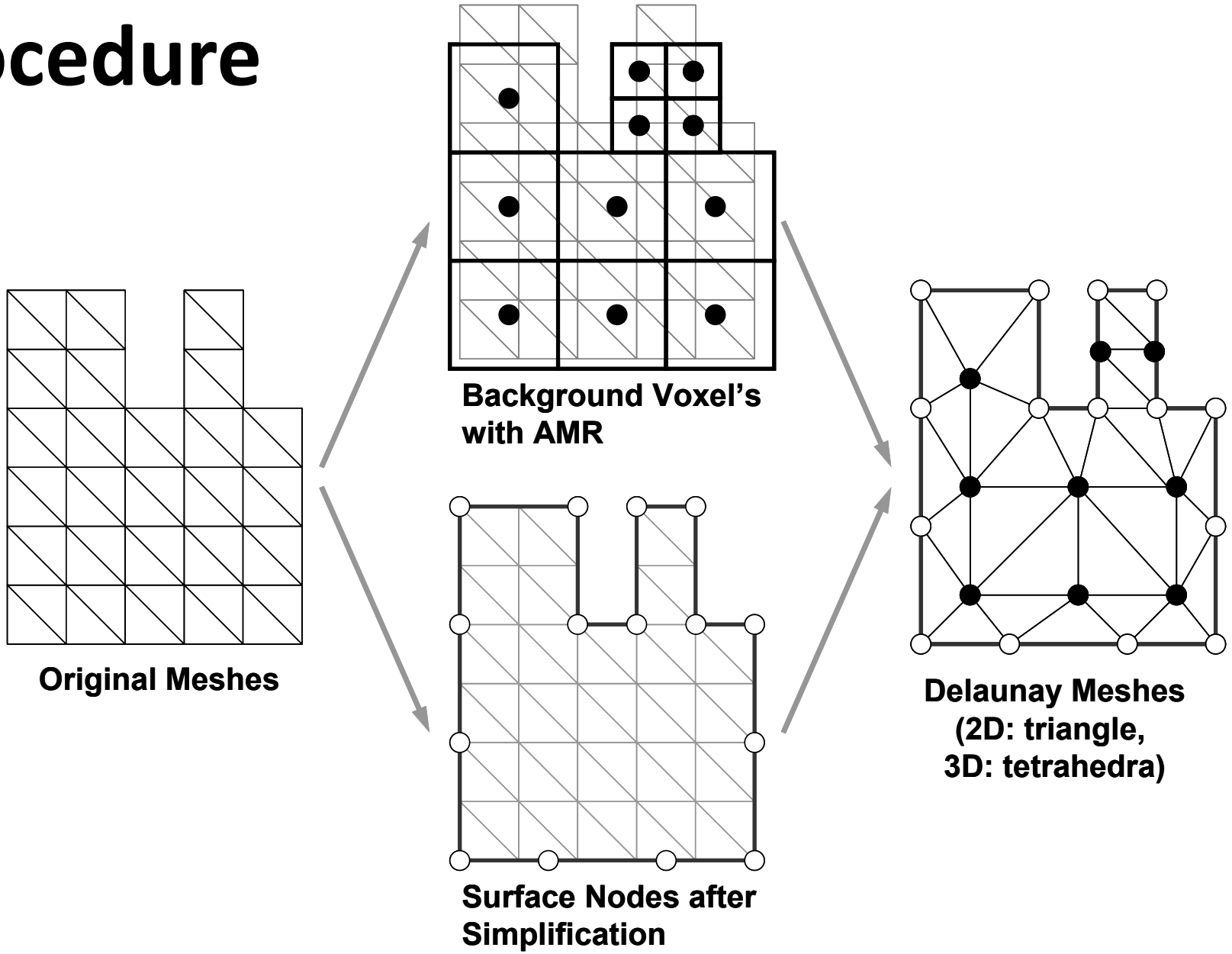
Simplified Parallel Visualization using Background Voxels

- 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.

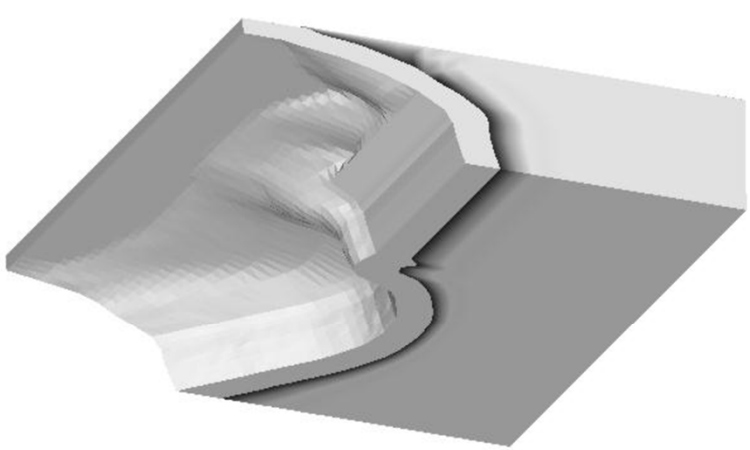


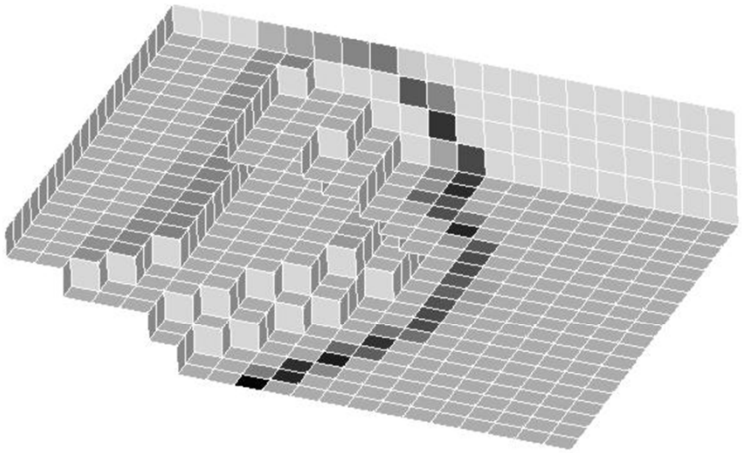


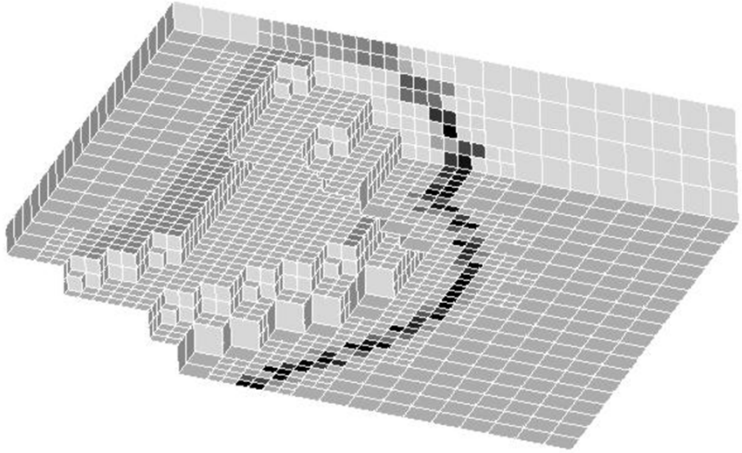
Procedure

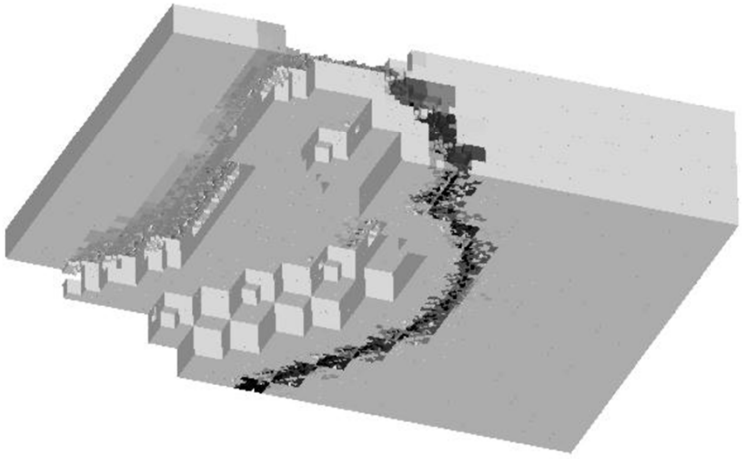

Original Meshes
Background Voxel's with AMR
Surface Nodes after Simplification
Delaunay Meshes (2D: triangle, 3D: tetrahedra)

Voxel Mesh (adapted)

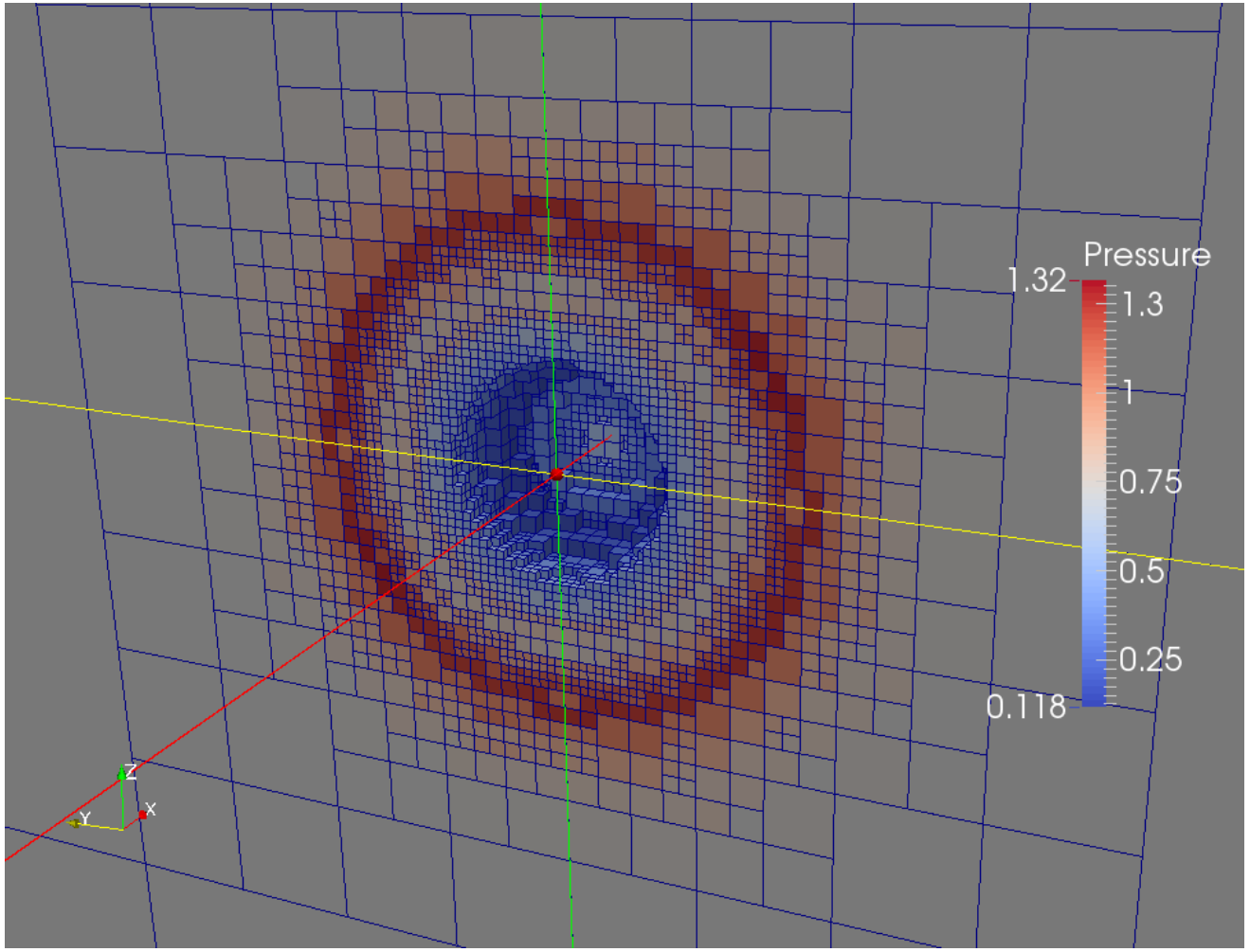






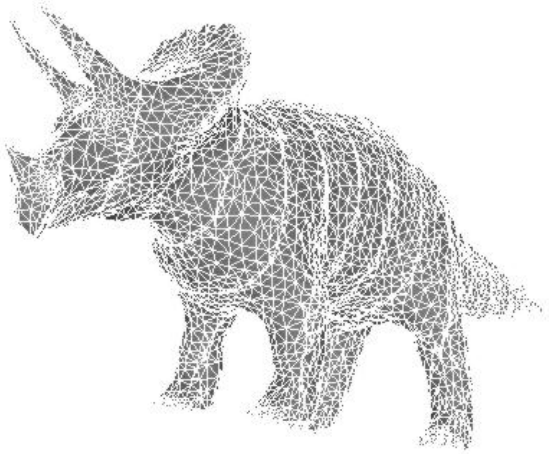


Flow around a sphere

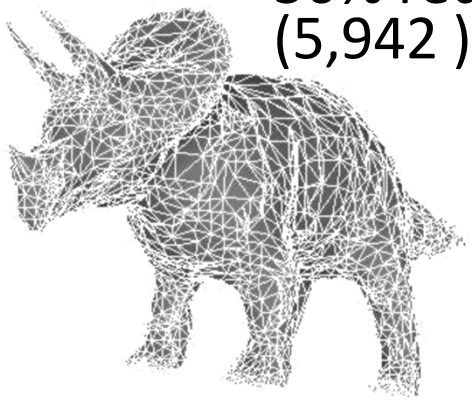


Example of Surface Simplification

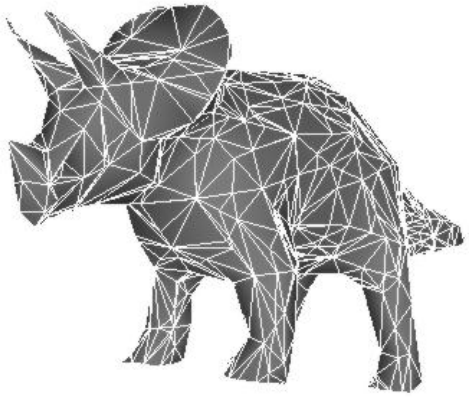
Initial
(11,884 tri's)



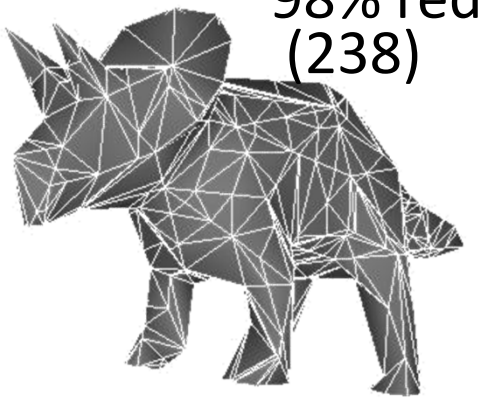
50% reduction
(5,942)



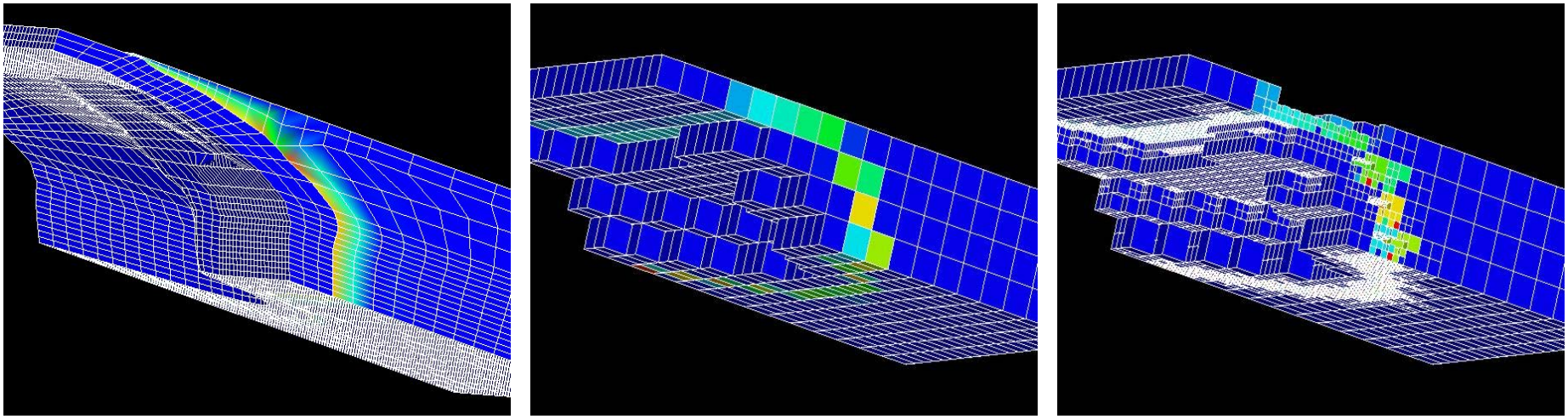
95% reduction
(594)



98% reduction
(238)



FEM Mesh (SW Japan Model)



pFEM3D + ppOpen-MATH/VIS

FORTRANユーザー

```
>$ cd ~/pFEM/srcV
>$ make
>$ cd ../run
>$ pjsub gv.sh
```

Cユーザー

```
>$ cd ~/pFEM/srcV
>$ make
>$ cd ../run
>$ pjsub gv.sh
```

Makefile（Fortran）

```
include Makefile.in

FFLAGSL   = -I/home/S11502/nakajima/pfem/include
FLDFLAGSL = -L/home/S11502/nakajima/pfem/lib
LIBSL     = -lfppohvispfem3d -lppohvispfem3d

.SUFFIXES:
.SUFFIXES: .o .f90 .f

.f.o:
        $(FC) -c $(FFLAGS) $(FFLAGSL) $< -o $@
.f90.o:
        $(F90) -c $(F90FLAGS) $(FFLAGSL) $< -o $@

TARGET = ../run/solv

OBJS = ¥
        test1.o ...

all: $(TARGET)

$(TARGET): $(OBJS)
        $(F90) -o $(TARGET) $(F90FLAGS) $(FFLAGSL) $(OBJS) $(LDFLAGSL)
$(LIBS) $(LIBSL) $(FLDFLAGSL)¥
```

Makefile（C）

```
include Makefile.in

CFLAGSL  = -I/home/S11502/nakajima/pfem/include
LDFLAGSL = -L/home/S11502/nakajima/pfem/lib
LIBSL    = -lppohvispfem3d

.SUFFIXES:
.SUFFIXES: .o .c

.c.o:
        $(CC) -c $(CFLAGS) $(CFLAGSL) $< -o $@

TARGET = ../run/solv

OBJS = ¥
        test1.o ...

all: $(TARGET)

$(TARGET): $(OBJS)
        $(CC) -o $(TARGET) $(CFLAGS) $(CFLAGSL) $(OBJS) $(LDFLAGSL)
$(LIBS) $(LIBSL)
        rm -f *.o *.mod
```

Fortran/main (1/2)

```
      use solver11
      use pfem_util
      use ppohvis_pfem3d_util

      implicit REAL*8(A-H,O-Z)
      type(ppohVIS_BASE_stControl)               :: pControl
      type(ppohVIS_BASE_stResultCollection)      :: pNodeResult
      type(ppohVIS_BASE_stResultCollection)      :: pElemResult
      character(len=PPOHVIS_BASE_FILE_NAME_LEN) :: CtrlName
      character(len=PPOHVIS_BASE_FILE_NAME_LEN) :: VisName
      character(len=PPOHVIS_BASE_LABEL_LEN)      :: ValLabel
      integer(kind=4)                             :: iErr

      CtrlName = ""
      CtrlName = "vis.cnt"

      VisName = ""
      VisName = "vis"

      ValLabel = ""
      ValLabel = "temp"

      call PFEM_INIT

      call ppohVIS_PFEM3D_Init(MPI_COMM_WORLD, iErr)
      call ppohVIS_PFEM3D_GetControl(CtrlName, pControl, iErr);
      call INPUT_CNTL
      call INPUT_GRID

      call ppohVIS_PFEM3D_SETMESHEX(                                    &
     &       NP,      N,           NODE_ID, XYZ,                      &
     &       ICELTOT, ICELTOT_INT, ELEM_ID, ICELNOD,                  &
     &       NEIBPETOT, NEIBPE, IMPORT_INDEX, IMPORT_ITEM,            &
     &                          EXPORT_INDEX, EXPORT_ITEM, iErr)
```

Fortran/main (2/2)

```
      call MAT_ASS_MAIN
      call MAT_ASS_BC

      call SOLVE11

      pNodeResult%ListCount = 1
      pElemResult%ListCount = 0
      allocate(pNodeResult%Results(1))

      call ppohVIS_PFEM3D_ConvResultNodeItem1N(                          &
     &       NP, ValLabel, X, pNodeResult%Results(1), iErr)

      call ppohVIS_PFEM3D_Visualize(pNodeResult, pElemResult, pControl, &
     &                              VisName, 1, iErr)

      call PFEM_FINALIZE

      end program heat3Dp
```

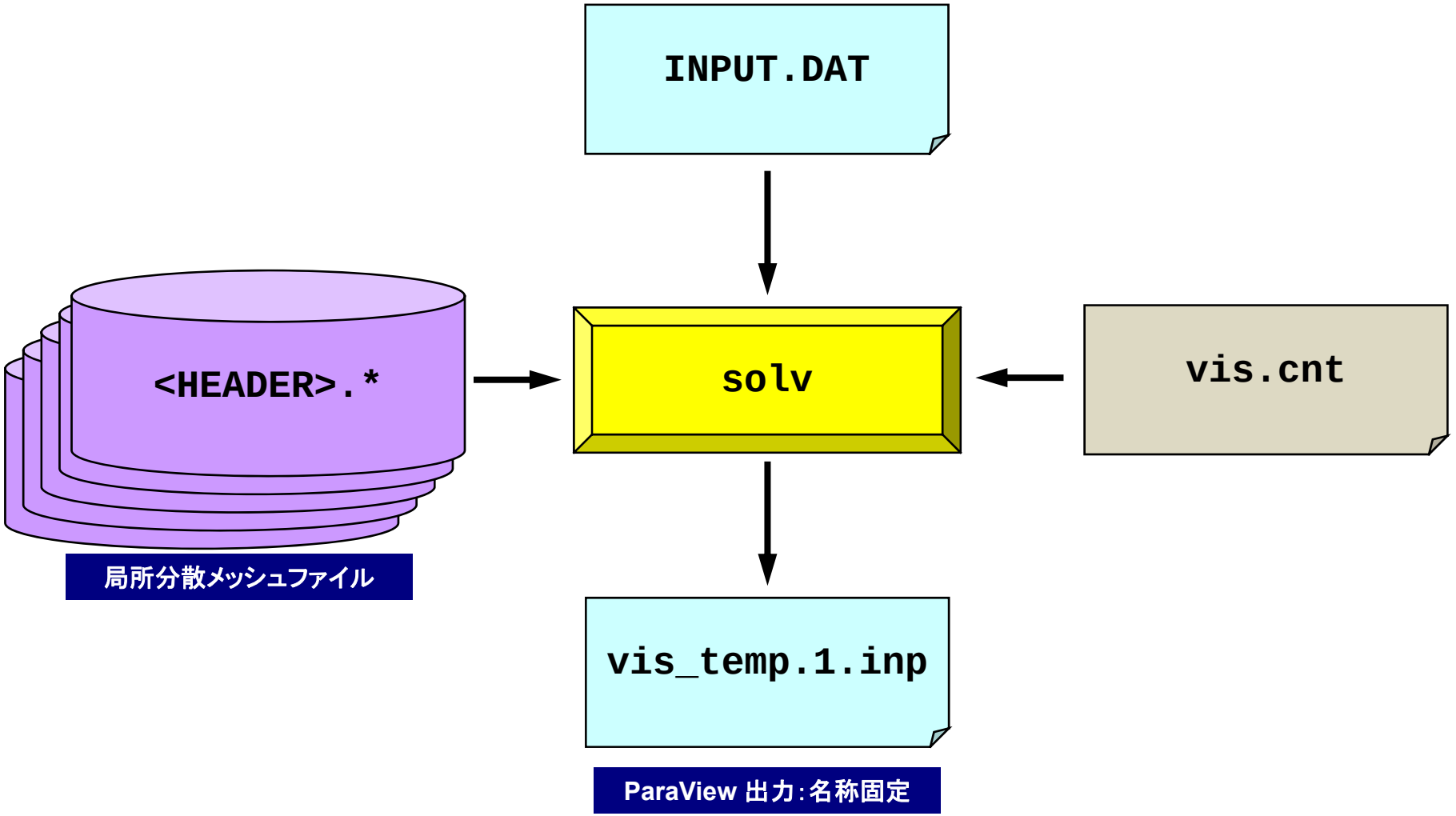
C/main (1/2)

```
#include <stdio.h>
#include <stdlib.h>
FILE* fp_log;
#define GLOBAL_VALUE_DEFINE
#include "pfem_util.h"
#include "ppohVIS_PFEM3D_Util.h"
extern void PFEM_INIT(int,char**);
extern void INPUT_CNTL();
extern void INPUT_GRID();
extern void MAT_CON0();
extern void MAT_CON1();
extern void MAT_ASS_MAIN();
extern void MAT_ASS_BC();
extern void SOLVE11();
extern void OUTPUT_UCD();
extern void PFEM_FINALIZE();
int main(int argc,char* argv[])
{
  double START_TIME,END_TIME;
  struct ppohVIS_FDM3D_stControl *pControl = NULL;
  struct ppohVIS_FDM3D_stResultCollection *pNodeResult = NULL;

  PFEM_INIT(argc,argv);
  ppohVIS_PFEM3D_Init(MPI_COMM_WORLD);
  pControl = ppohVIS_FDM3D_GetControl("vis.cnt");

  INPUT_CNTL();
  INPUT_GRID();

  if(ppohVIS_PFEM3D_SetMeshEx(
      NP,N,NODE_ID,XYZ,
      ICELTOT,ICELTOT_INT,ELEM_ID,ICELNOD,
      NEIBPETOT,NEIBPE,IMPORT_INDEX,IMPORT_ITEM,EXPORT_INDEX,EXPORT_ITEM)) {
                ppohVIS_BASE_PrintError(stderr);
                MPI_Abort(MPI_COMM_WORLD,errno);
  };
```

C/main (2/2)

```
  MAT_CON0();
  MAT_CON1();

  MAT_ASS_MAIN();
  MAT_ASS_BC()  ;

  SOLVE11();

  OUTPUT_UCD();

  pNodeResult=ppohVIS_BASE_AllocateResultCollection();
        if(pNodeResult == NULL) {
                ppohVIS_BASE_PrintError(stderr);
                MPI_Abort(MPI_COMM_WORLD,errno);
        };
        if(ppohVIS_BASE_InitResultCollection(pNodeResult, 1)) {
                ppohVIS_BASE_PrintError(stderr);
                MPI_Abort(MPI_COMM_WORLD,errno);
        };

        pNodeResult->Results[0] =

  ppohVIS_PFEM3D_ConvResultNodeItemPart(NP,1,0,"temp",X);

  START_TIME= MPI_Wtime();
        if(ppohVIS_PFEM3D_Visualize(pNodeResult,NULL,pControl,"vis",1)) {
                ppohVIS_BASE_PrintError(stderr);
                MPI_Abort(MPI_COMM_WORLD,errno);
        };

  ppohVIS_PFEM3D_Finalize();

  PFEM_FINALIZE() ;
}
```

pFEM3D + ppOpen-MATH/VIS



分散メッシュファイルの準備

```
>$ cd <$O-TOP>/pfem3d/pmesh
(mesh.inp, mg.sh)

>$ pjsub mg.sh
```

mesh.inp

```
256 256 256
  4   4   4
pcube
```

256^3のメッシュを4×4×4=64分割
各MPIプロセスは64^3

mg.sh

```
#!/bin/sh
#PJM -L "node=4"
#PJM -L "elapse=00:05:00"
#PJM -L "rscgrp=school"
#PJM -j
#PJM -o "mg.lst"
#PJM --mpi "proc=64"

mpiexec ./pmesh
rm wk.*
```

計算の実行＋可視化

```
>$ cd <$O-TOP>/pfem3d/run
(INPUT.DAT, gv.sh)

>$ pjsub gv.sh
```

INPUT.DAT

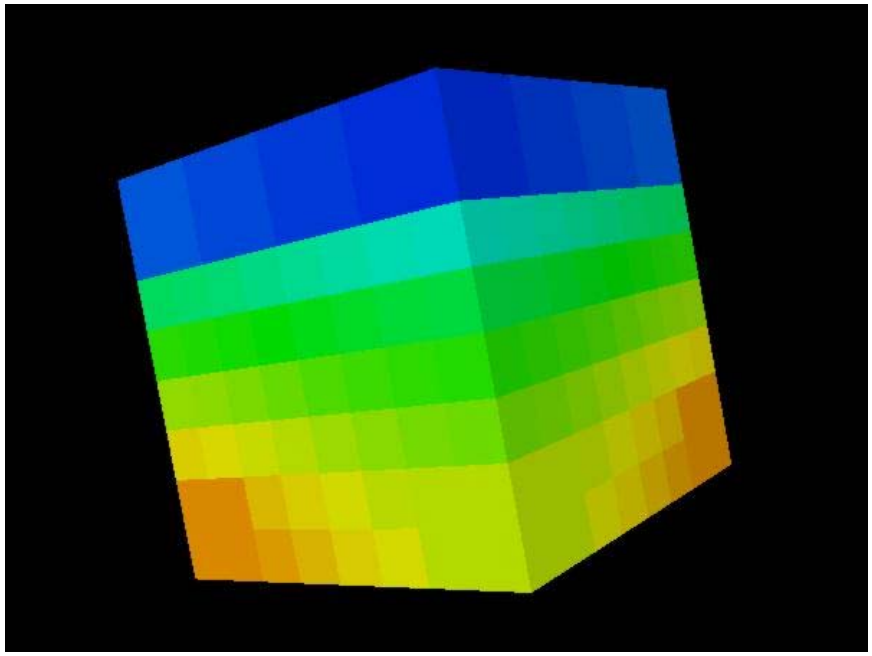
```
../pmesh/pcube
2000
1.0 1.0
1.0e-08
```

gv.sh

```
#!/bin/sh
#PJM -L "node=4"
#PJM -L "elapse=00:10:00"
#PJM -L "rscgrp=school"
#PJM -j
#PJM -o "aa.lst"
#PJM --mpi "proc=64"

mpiexec ./solv
```

vis.cnt

```
[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 = 1000        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
[Output]                    出力形式
FileFormat    = 2           =1:MicroAVS, =2:ParaView
```



Movie：要素中心の値

16,777,216節点
16,581,375要素，64 MPIプロセス

↓

vis_temp.1.inp
1,436節点
1,002要素