3D Parallel FEM (III) Parallel Visualization and ppOpen-HPC

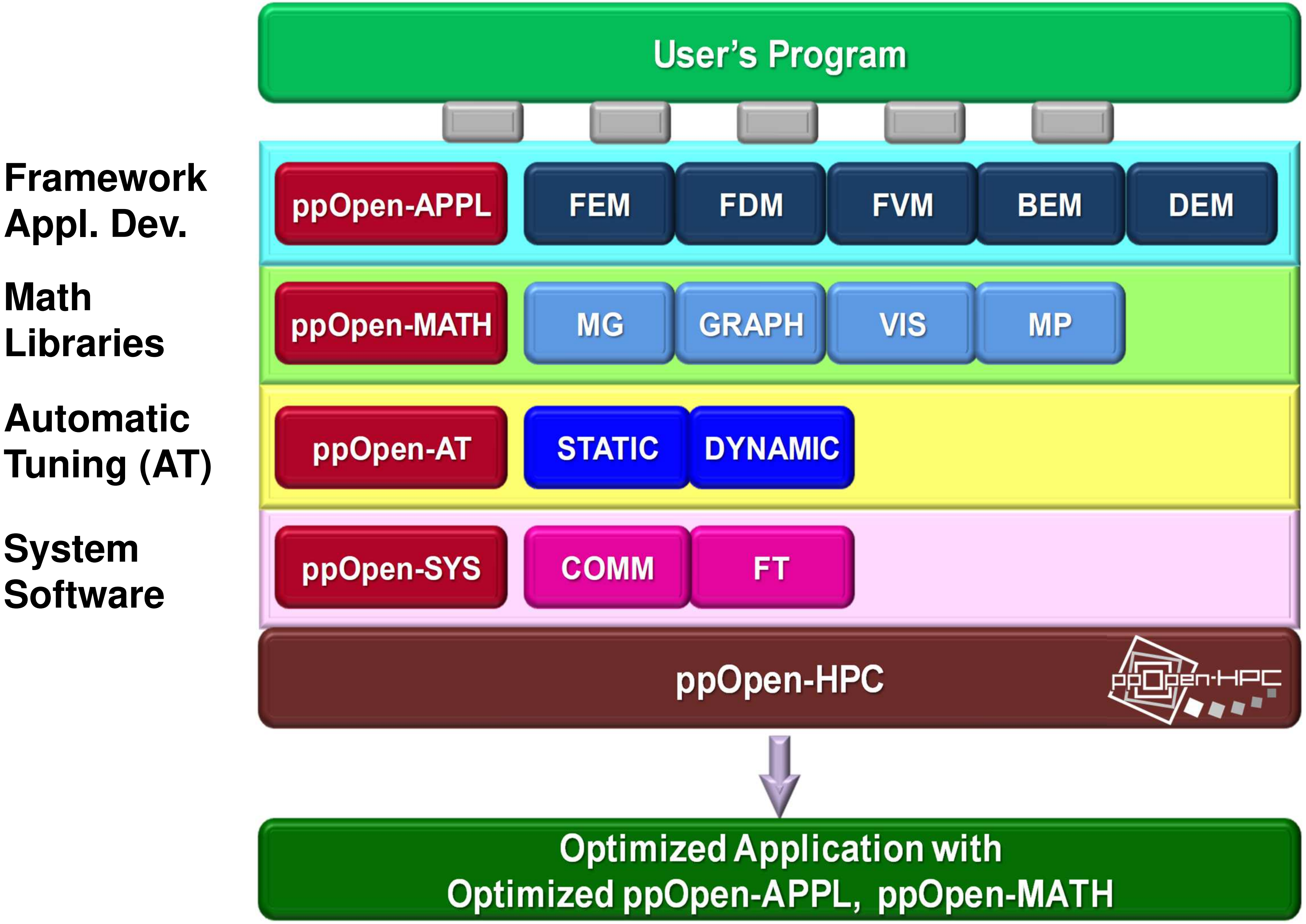
Kengo Nakajima
RIKEN R-CCS

ppOpen-HPC: Overview

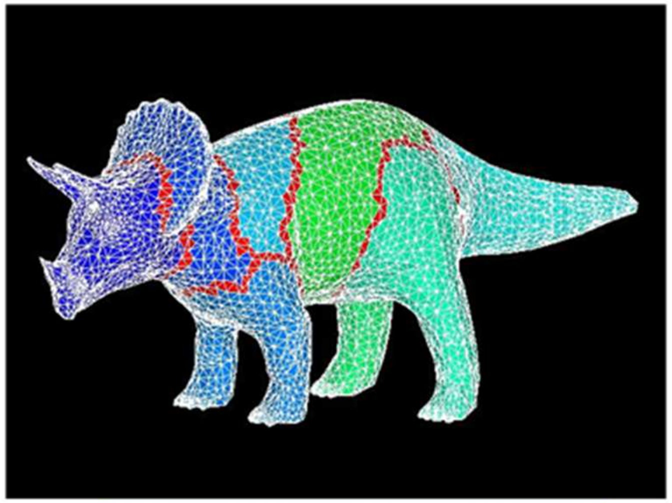
- Application framework with automatic tuning (AT)
  - ✓ "pp" : post-peta-scale
- Five-year project (FY.2011-2015) (since April 2011)
  - ✓ P.I.: Kengo Nakajima (ITC, The University of Tokyo)
  - ✓ Part of "Development of System Software Technologies for Post-Peta Scale High Performance Computing" funded by JST/CREST (Supervisor: Prof. M. Sato, RIKEN R-CCS)
  - ✓ Finally, it was extended to FY.2018 under collaborations with German Projects (SPPEXA/DFG)
- Team with 7 institutes, >50 people (5 PDs) from various fields: Co-Design
- Open Source Software
  - ✓ http://ppopenhpc.cc.u-tokyo.ac.jp/
  - ✓ https://github.com/Post-Peta-Crest/ppOpenHPC
  - ✓ English Documents, MIT License

独立行政法人
科学技術振興機構
Japan Science and Technology Agency

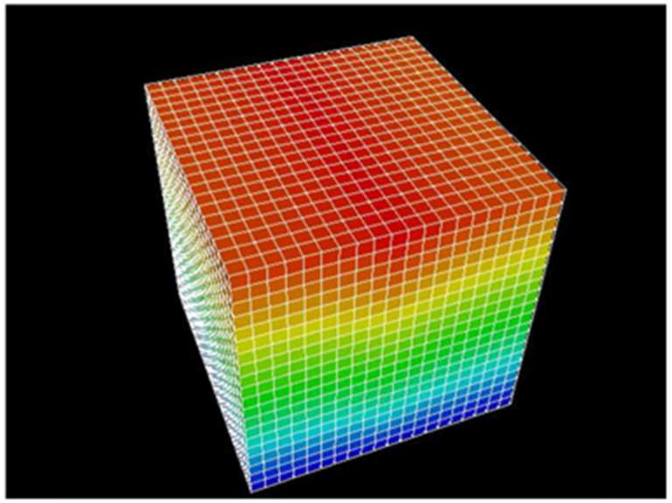
CREST
戦略的創造研究推進事業
Core Research for Evolutionary Science and Technology


User's Program
Framework
Appl. Dev.
ppOpen-APPL
FEM
FDM
FVM
BEM
DEM
Math
Libraries
ppOpen-MATH
MG
GRAPH
VIS
MP
Automatic
Tuning (AT)
ppOpen-AT
STATIC
DYNAMIC
System
Software
ppOpen-SYS
COMM
FT
ppOpen-HPC
Optimized Application with
Optimized ppOpen-APPL, ppOpen-MATH

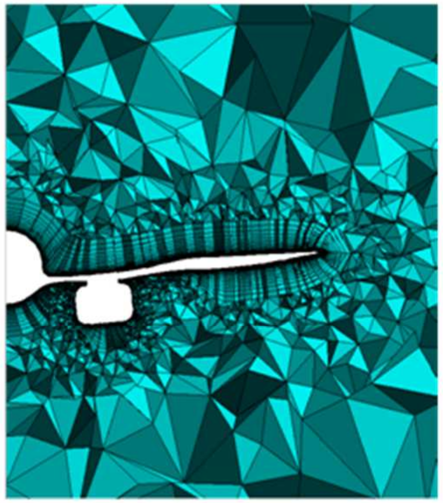
ppOpen-HPC covers …



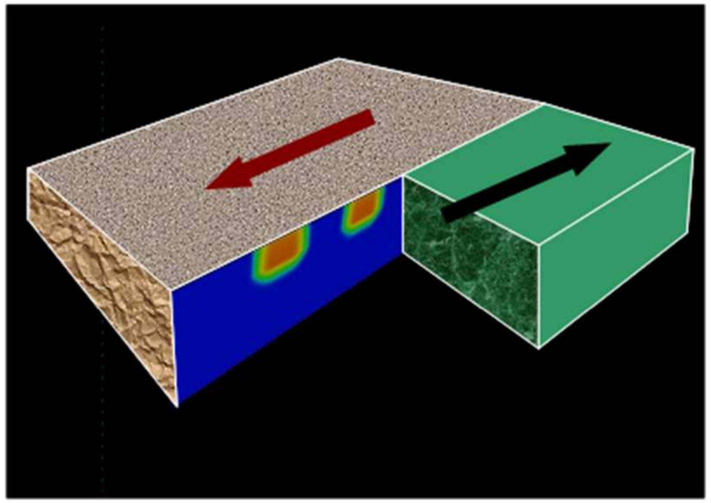
FEM
Finite Element Method



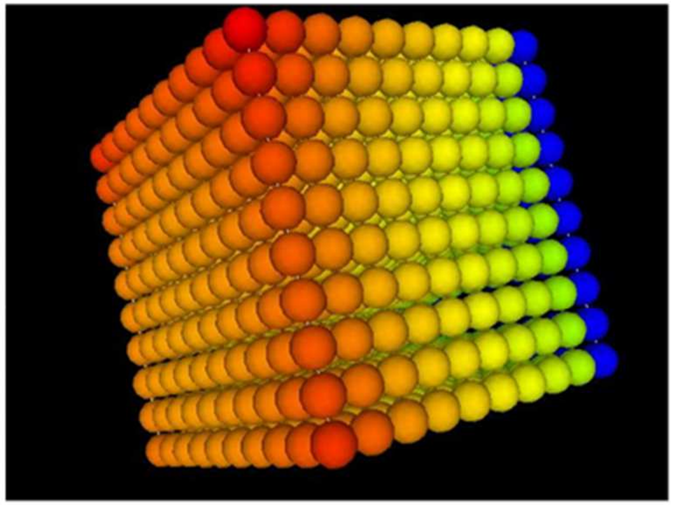
FDM
Finite Difference Method



FVM
Finite Volume Method



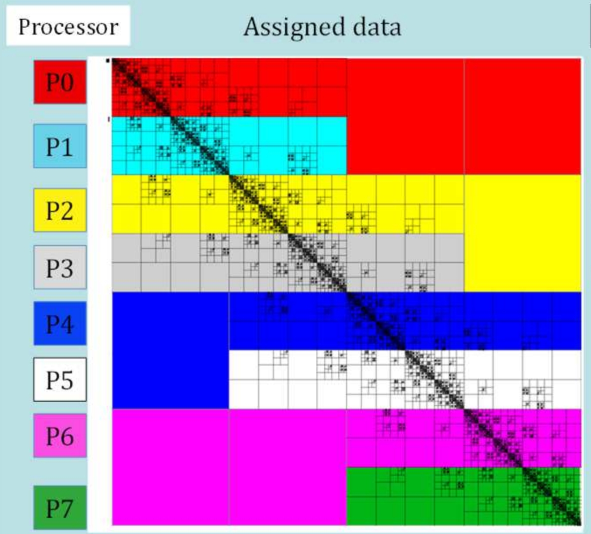
BEM
Boundary Element Method



DEM
Discrete Element Method

Featured Developments

- ppOpen-AT: AT Language for Loop Optimization
- 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)**
- **Sparse Linear Solvers**



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

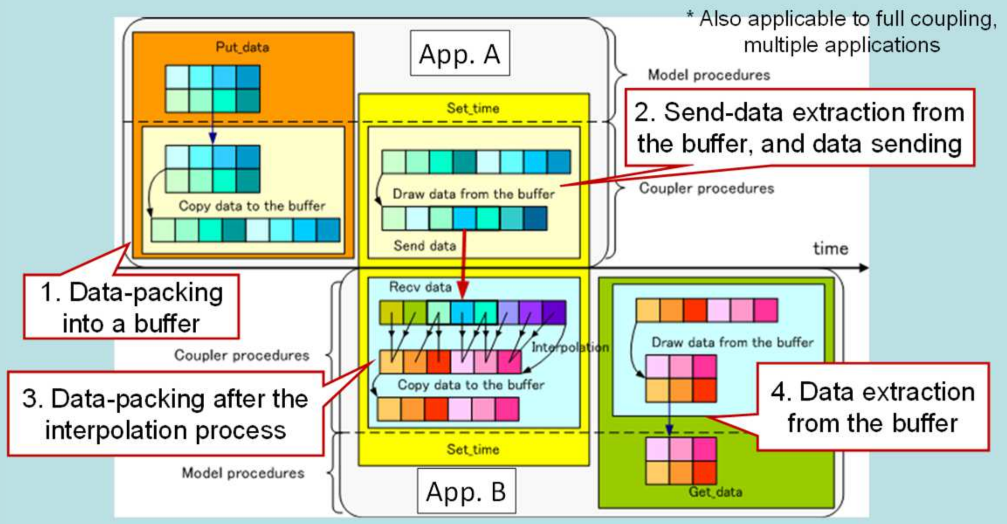
n : Number of processors

P_k : The k-th processor

$l_k : 1 = l_0 < \cdots < l_k < \cdots < l_{n+1} = N + 1$

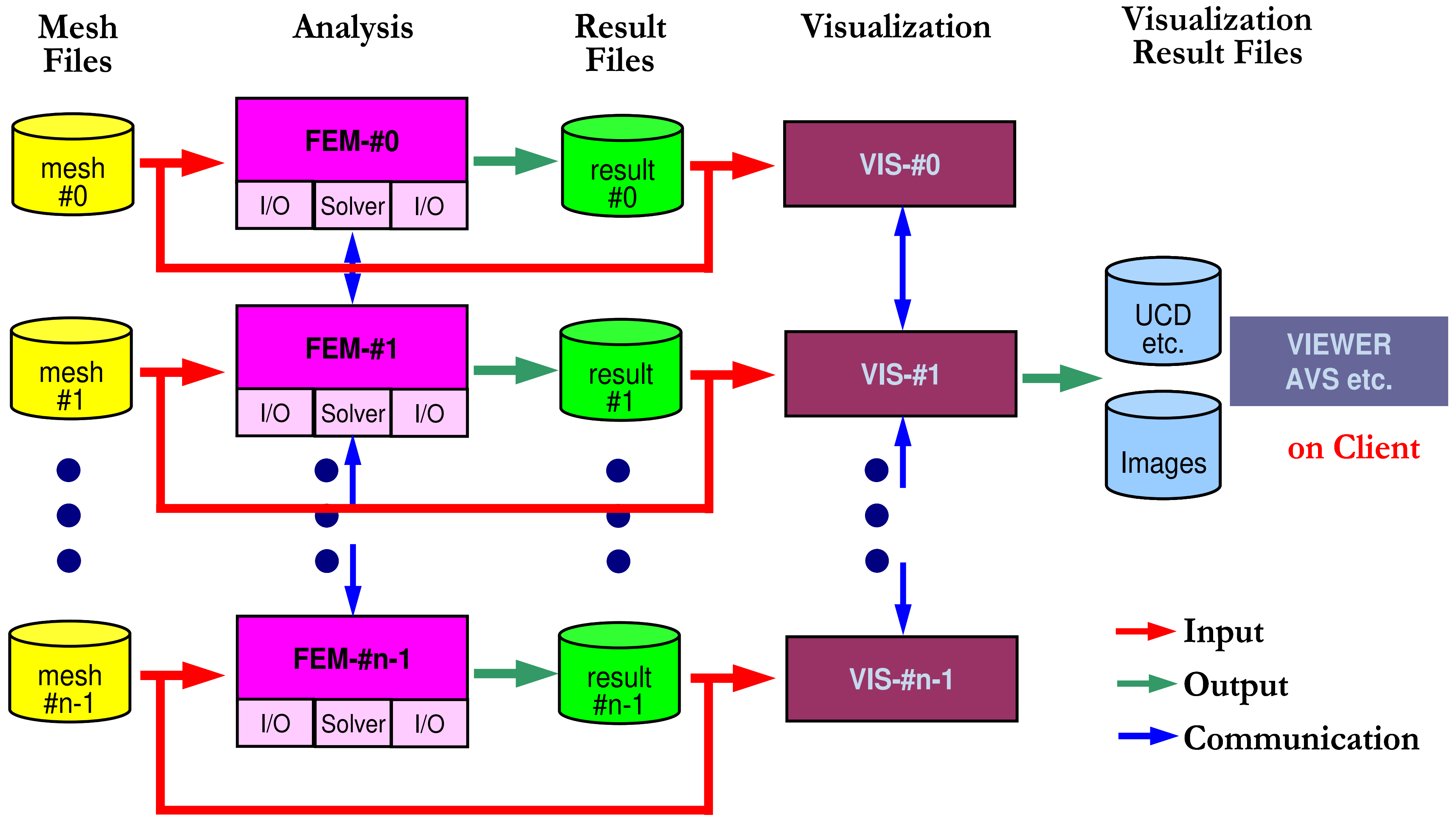
$R : R(P_k) = \{ i \mid l_k \le i < l_{k+1} \}$

$S : S(a^c{}_{ij}) = I$ when $a^c{}_{ij} = A_{IJ}$

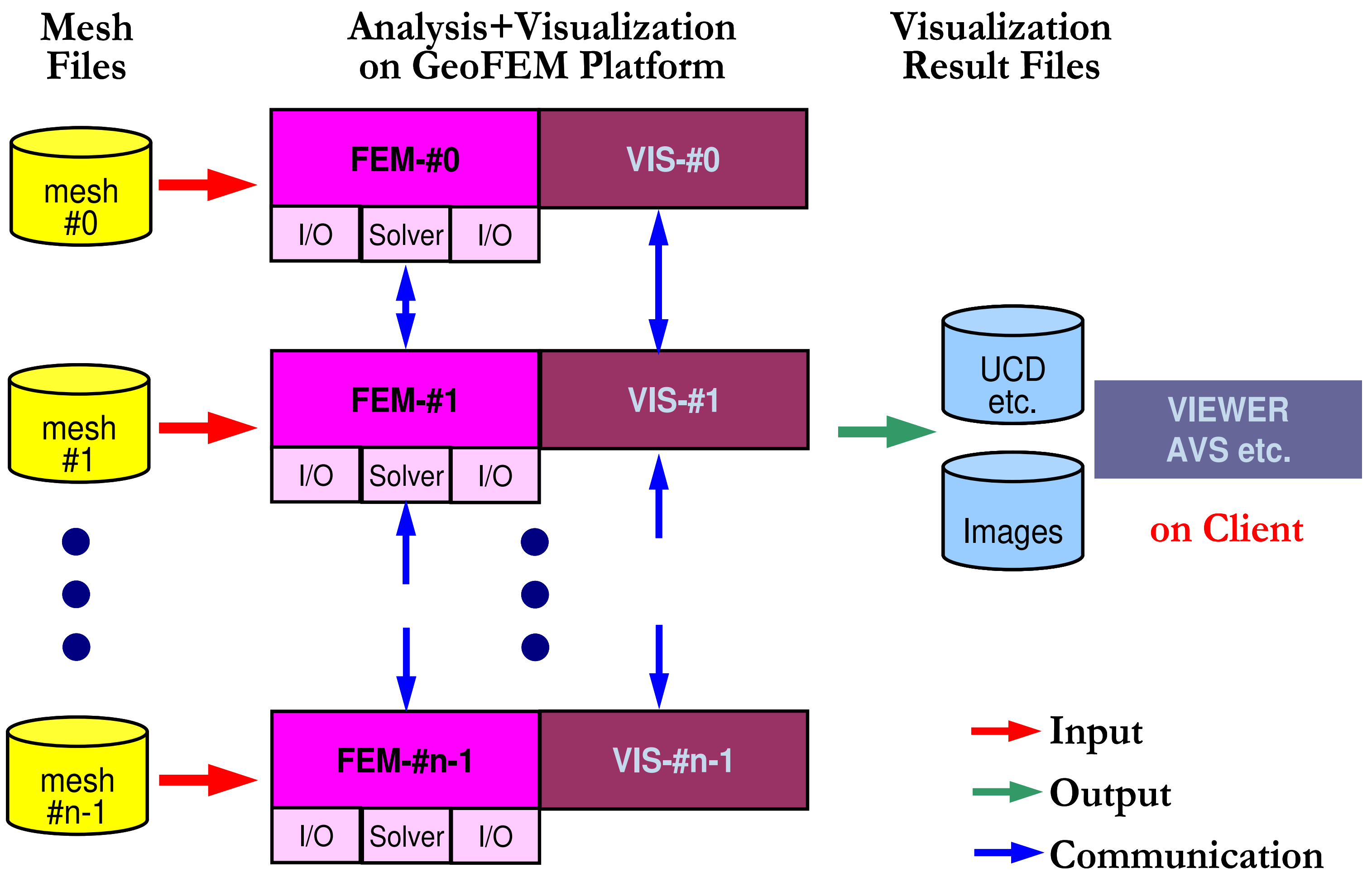


* Also applicable to full coupling, multiple applications
Put_data
App. A
Set_time
Model procedures
2. Send-data extraction from the buffer, and data sending
Copy data to the buffer
Draw data from the buffer
Coupler procedures
Send data
time
1. Data-packing into a buffer
Recv data
Interpolation
Draw data from the buffer
Coupler procedures
Copy data to the buffer
3. Data-packing after the interpolation process
4. Data extraction from the buffer
Set_time
Model procedures
App. B
Get_data

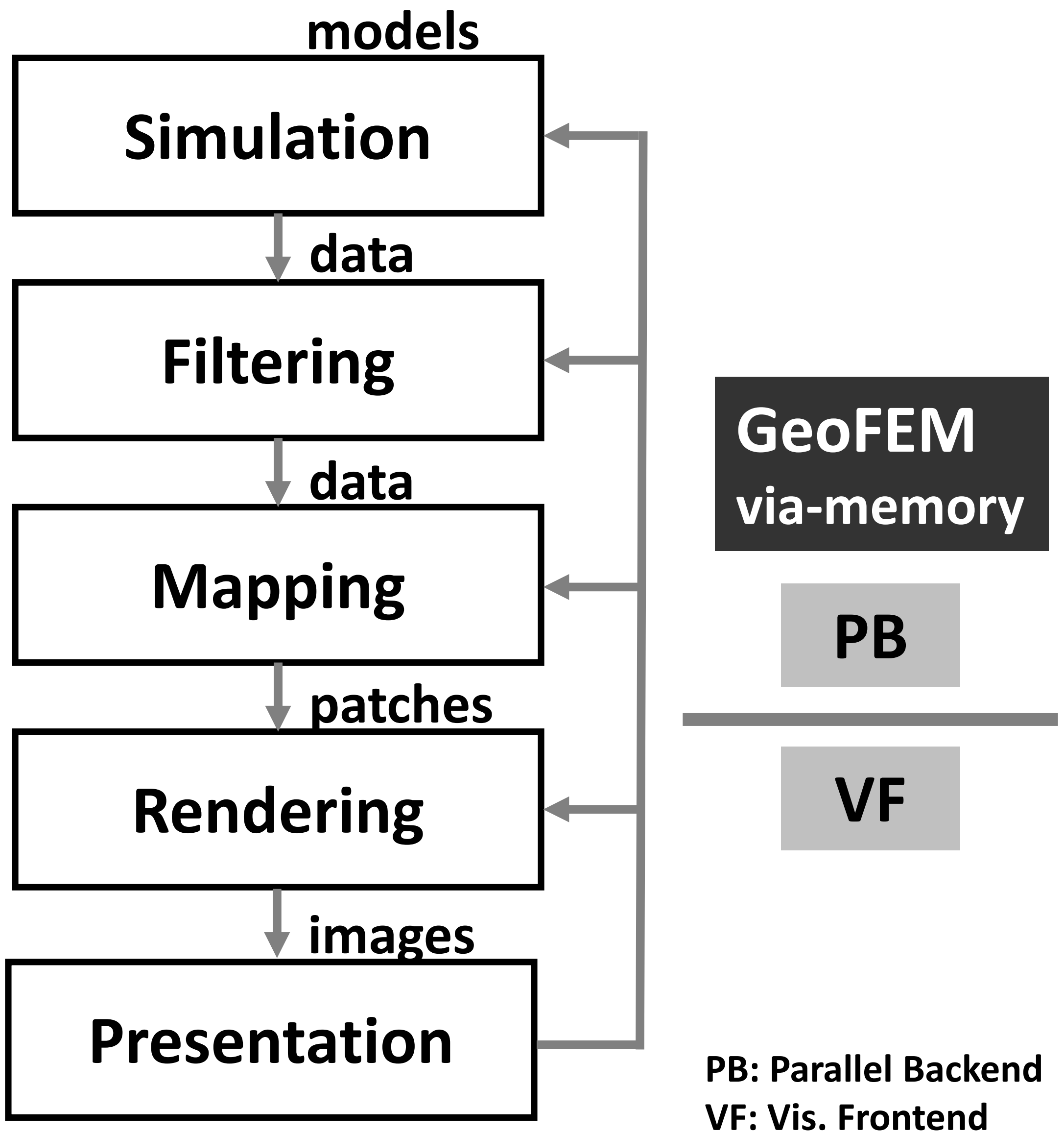
Framework for Parallel Visualization 1 Via-File



Framework for Parallel Visualization 2 Via-Memory (GeoFEM Project)



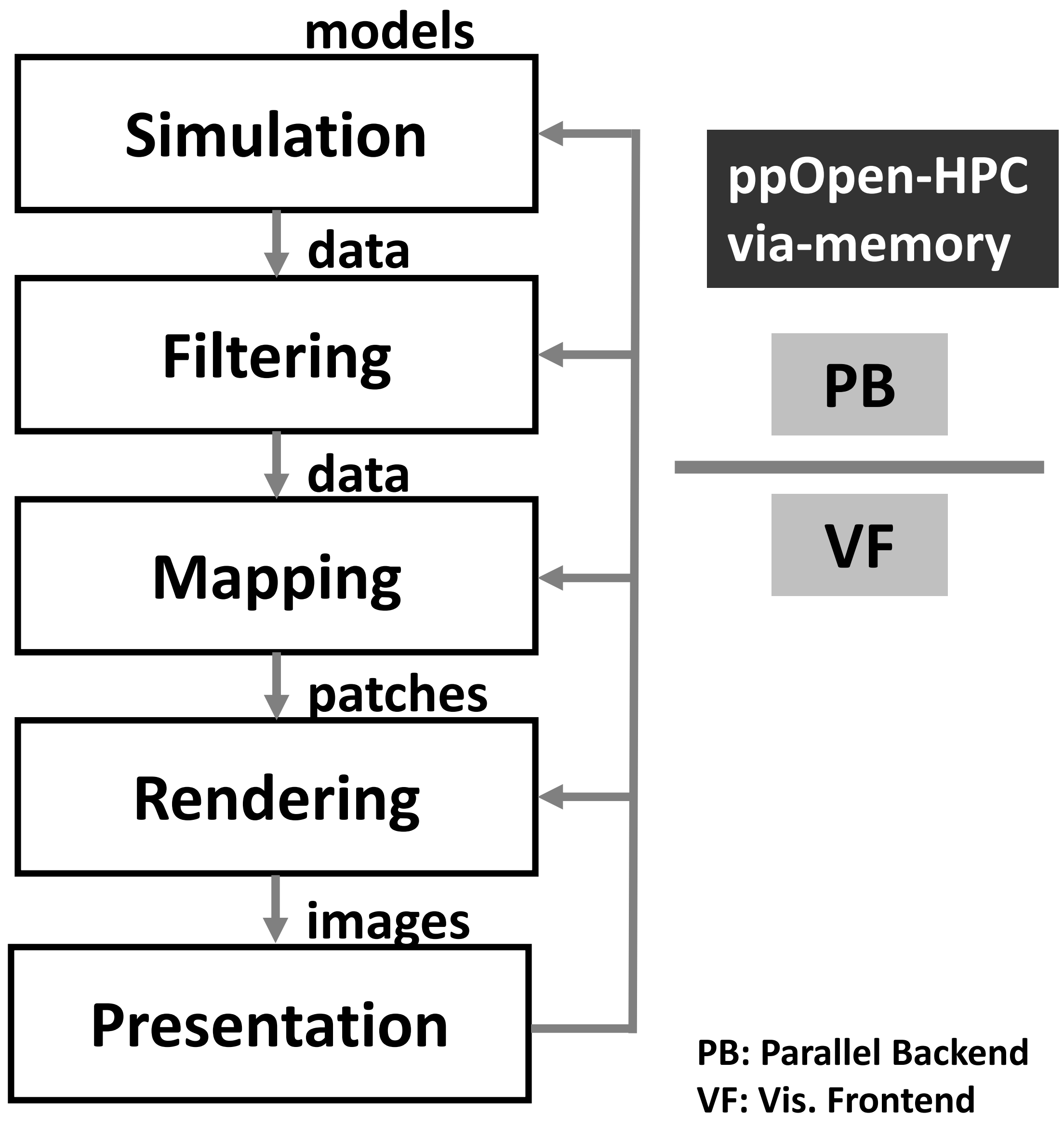
Visualization in GeoFEM (1997-2003)



PB: Parallel Backend
VF: Vis. Frontend

- Concurrent Visualization-Computation (Via Memory)
- In GeoFEM (previous project), only patch files were obtained.

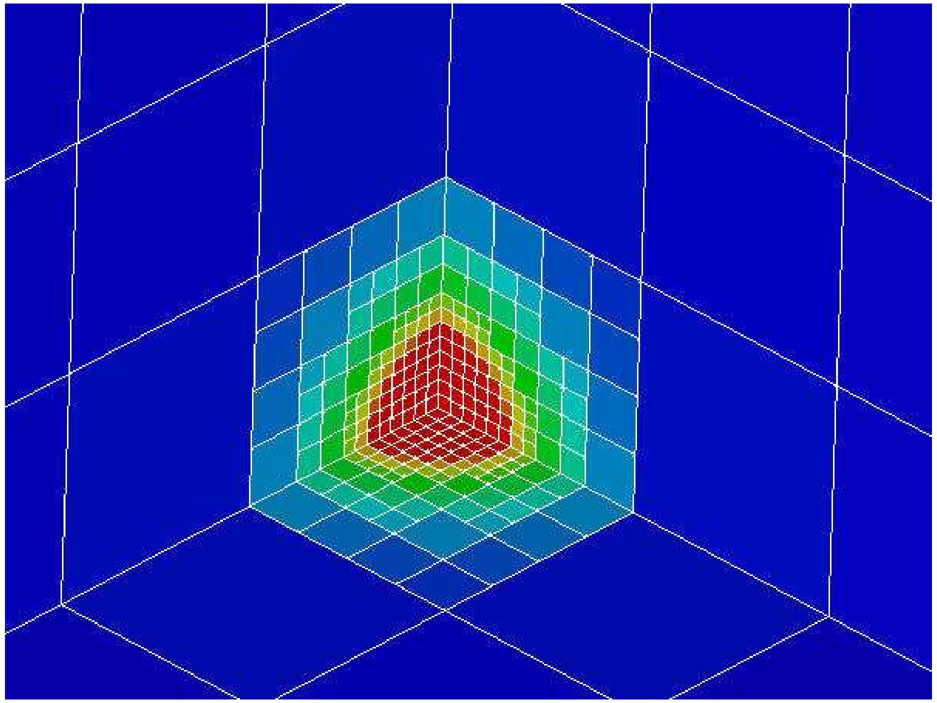
Visualization in ppOpen-HPC



- Concurrent Visualization-Computation (Via Memory)
- Output files (single “self-contained（自己完結）” file) are browsed by MicroAVS & Paraview on a PC
  - ✓ Different from GeoFEM (previous project), where only patch files were obtained.
- Not detailed visualization.
- Just for understanding MIN-MAX, and peaks
- Detailed geometry is preferable

ppOpen-MATH/VIS

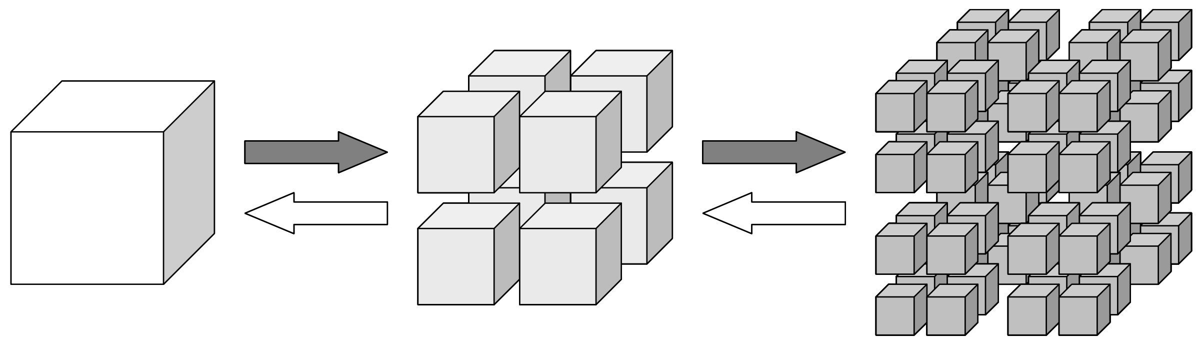
- Parallel Visualization using Information of Background Voxels [Nakajima & Chen 2006]
  - FDM version is released: ppOpen-MATH/VIS-FDM3D
- UCD single file

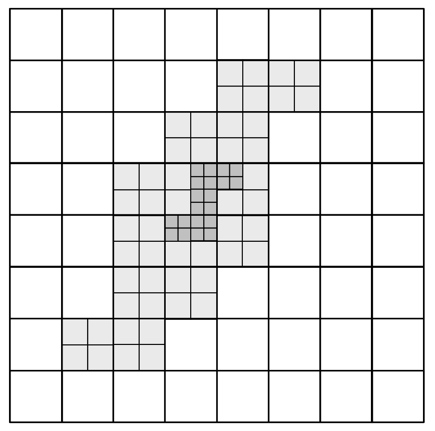


```
[Refine]
AvailableMemory = 2.0     Available memory size (GB), not available in this version.
MaxVoxelCount   = 500     Maximum number of voxels
MaxRefineLevel  = 20      Maximum number of refinement levels
```

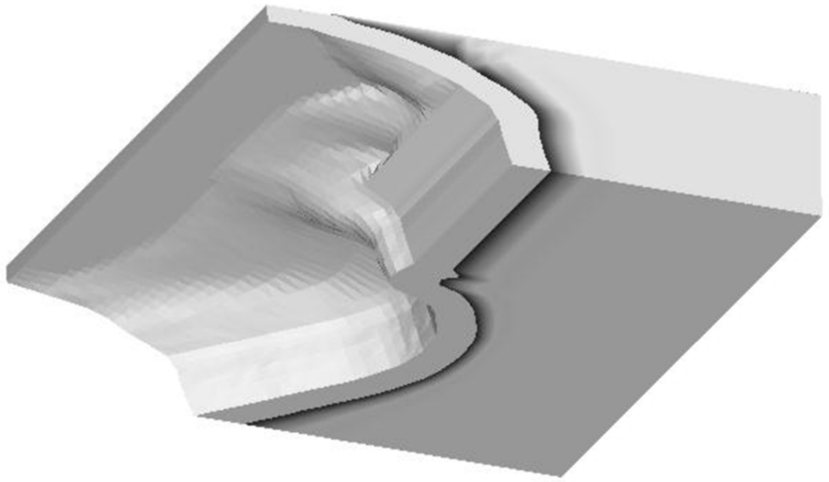
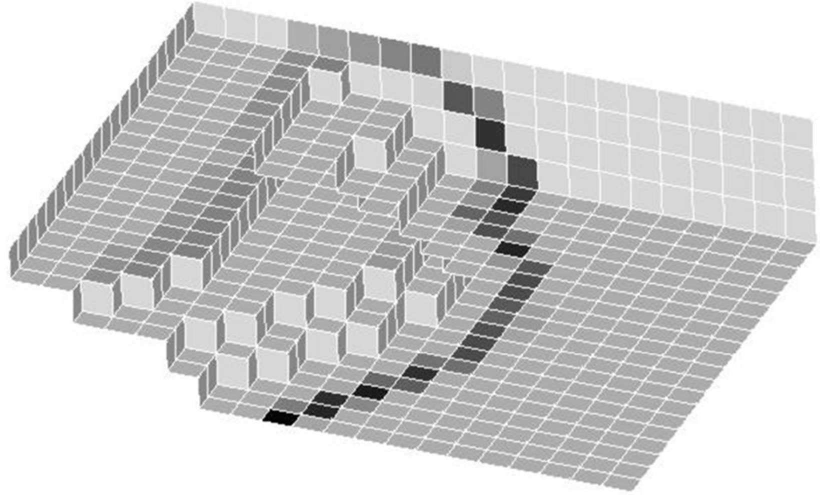
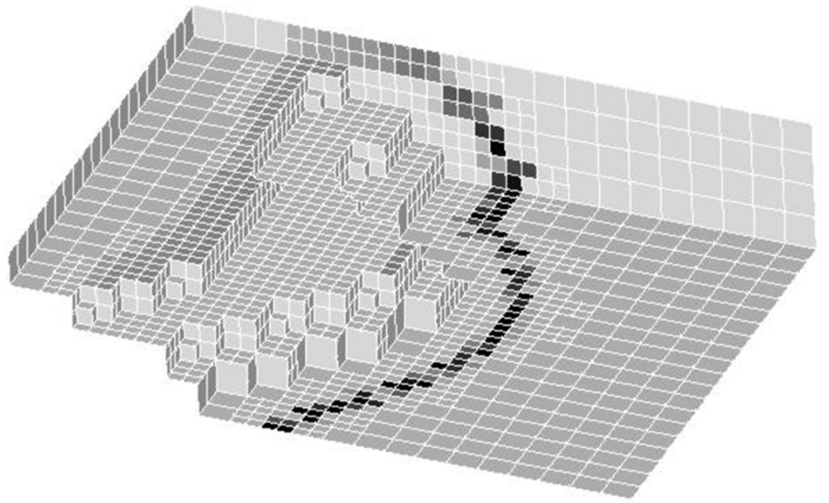
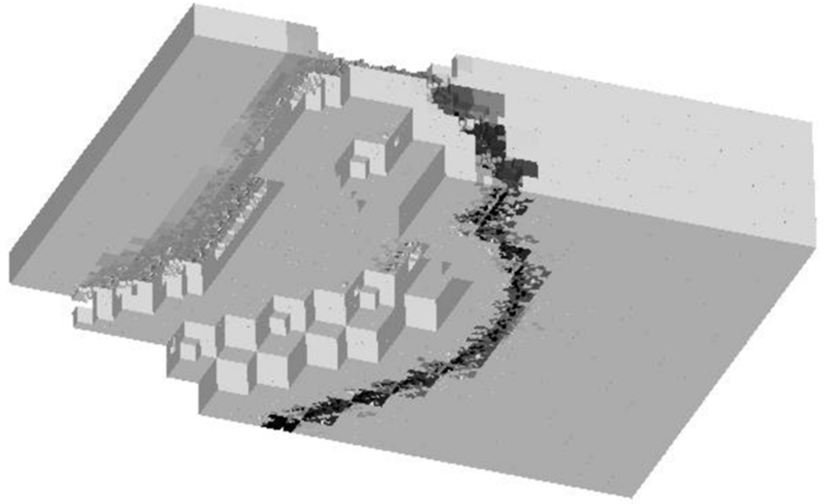
Simplified Parallel Visualization using Background Voxels

- 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 controlled, a single file with 10^5 meshes is possible, even though entire problem size is 10^9 with distributed data sets.

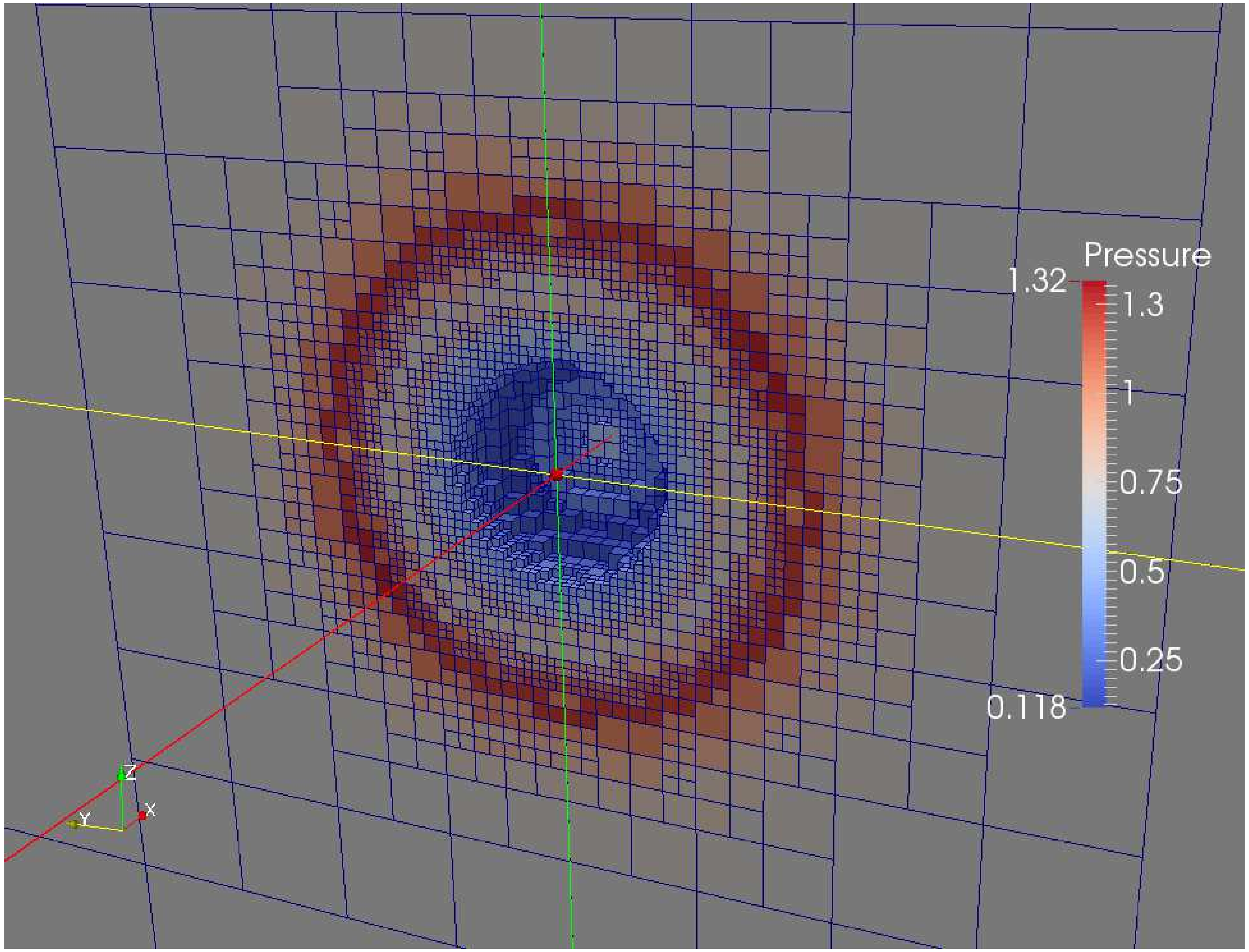




Voxel Mesh (adapted)

Flow around a sphere



pFEM3D + ppOpen-MATH/VIS

Fortran

```
>$ cd /home/ra020019/<Your-UID>/pFEM/pfem3d/srcV
>$ make
>$ cd ../runV
>$ ls solv
     solv
```

C

```
>$ cd /home/ra020019/<Your-UID>/pFEM/pfem3d/srcV
>$ make
>$ cd ../runV
>$ ls solv
     solv
```

pFEM/pfem3d/srcV/Makefile（F）

```
include Makefile.in

FFLAGSL    = -I/vol0001/ra020019/ppohVIS/include
FLDFLAGSL = -L/vol0001/ra020019/ppohVIS/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 = ¥
        pfem_util.o …

all: $(TARGET)

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

clean:
        rm -f *.o *.mod $(TARGET)
distclean:
        rm -f *.o *.mod $(TARGET)
```

pFEM/pfem3d/srcV/Makefile（C）

```
include Makefile.in

CFLAGSL   = -I/vol0001/ra020019/ppohVIS/include
LDFLAGSL = -L/vol0001/ra020019/ppohVIS/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)
clean:
	rm -f *.o *.mod $(TARGET)

distclean:
	rm -f *.o *.mod $(TARGET)
```

pFEM/pfem3d/srcV/Makefile.in (1/2)

```
# Install directory
PREFIX      = /vol0001/ra020019//ppohVISflat
BINDIR      = $(PREFIX)/bin
INCDIR      = $(PREFIX)/include
LIBDIR      = $(PREFIX)/lib

# TetGen directory
TETGENDIR   = $(HOME)/usr/local/tetgen1.4.3
TETINCDIR   = $(TETGENDIR)
TETLIBDIR   = $(TETGENDIR)

# C compiler settings
CC          = mpifccpx
CFLAGS      = $(CINCDIR) $(COPTFLAGS)
COPTFLAGS   = -Nclang -Kfast -Kopenmp
CINCDIR     = -I. -I$(TETINCDIR)

# C++ compiler settings
CXX         = mpiFCCpx
CXXFLAGS    = $(CXXINCDIR) $(CXXOPTFLAGS) -DTETGEN -DTETLIBRARY
CXXOPTFLAGS = -Nclang -Kfast -Kopenmp
CXXINCDIR   = -I. -I$(TETINCDIR)

# Fortran 77 compiler settings
FC          = mpifrtpx
FFLAGS      = $(FINCDIR) $(FOPTFLAGS)
FOPTFLAGS   = -Kfast -Kopenmp
FINCDIR     = -I.
```

pFEM/pfem3d/srcV/Makefile.in (2/2)

```
# Fortran 90 compiler settings
F90          = mpifrtpx
F90FLAGS     = $(F90INCDIR) $(F90OPTFLAGS)
F90OPTFLAGS = -Kfast -Kopenmp
F90INCDIR    = -I.

# Linker settings
LD           = $(CC)
LDFLAGS      = $(LIBS)
#LIBS          = -L$(TETLIBDIR) -ltet -lm
LIBS         = -L$(TETLIBDIR) -lm
LDLIBDIR     =

# Archiver settings
AR           = ar rv

# ETC
CP           = cp -f
RM           = rm -rf
MKDIR        = mkdir -p
```

https://www.dropbox.com/s/i3lvkvyta28xwn6/ppohVIS.tar?dl=0

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
```

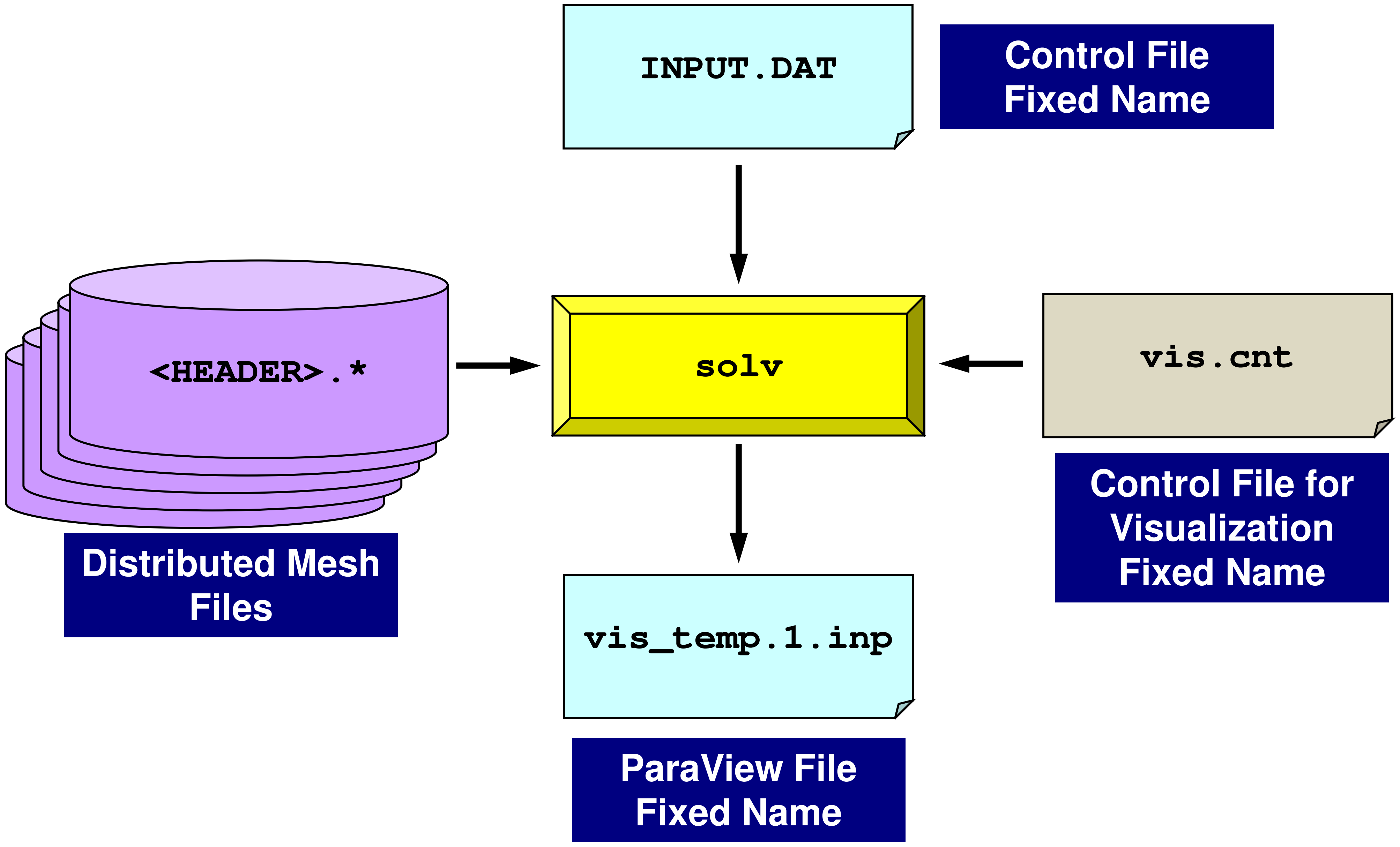
C/main (1/2)

```
#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



Preparing Distributed Mesh Files

```
>$ cd /home/ra020019/<Your-UID>/pFEM/pfem3d/runV
(mesh.inp, mg.sh)
>$ pjsub mg.sh
```

mesh.inp

```
256 192 192
 16   6   6
pcube
```

256×192×192 nodes into
16×6×6=576 partitions

9,437,184 nodes
9,302,655 elements

Each MPI process has
16x32x32 nodes

mg.sh

```
#!/bin/sh
#PJM -N "pmg"
#PJM -L "rscgrp=small"
#PJM -L "node=12"
#PJM --mpi "max-proc-per-node=48"
#PJM -L elapse=00:15:00
#PJM -g ra020019
#PJM -j
#PJM -e err
#PJM -o pmg.lst

setenv MP_STDINMODE all

mpiexec ./pmesh < INPUT.DAT

rm wk.*
```

Computation + Visualization

```
>$ cd /home/ra020019/<Your-UID>/pFEM/pfem3d/runV
(INPUT.DAT, gg.sh)
>$ pjsub gg.sh
```

INPUT.DAT

```
pcube
2000
1.0 1.0
1.0e-08
```

gg.sh

```
#!/bin/sh
#PJM -N "VIS"
#PJM -L rscgrp=small
#PJM -L "node=12:torus"
#PJM --mpi "max-proc-per-node=48"
#PJM -L elapse=00:15:00
#PJM -g ra020019
#PJM -j
#PJM -e err
#PJM -o testV.lst

setenv MP_STDINMODE all

mpiexec ./solv < INPUT.DAT
```

Preparing Distributed Mesh Files

```
>$ cd /home/ra020019/<Your-UID>/pFEM/pfem3d/runV
 (mesh.inp, mg.sh)
>$ pjsub mg.sh
```

mesh.inp

```
256 256 256
  8   8   4
pcube
```

256^3 nodes into 8×8×4=256 partitions
16,777,216 nodes
16,581,375 elements
Each MPI process has 32x32x64 nodes

mg.sh

```
#!/bin/bash
#PJM -N "pmg"
#PJM -L "rscgrp=small"
#PJM -L "node=8:torus"
#PJM --mpi "max-proc-per-node=32"
#PJM -L "elapse=00:15:00"
#PJM -g ra020019
#PJM -s
#PJM -e err
#PJM -o pmg.lst

mpiexec ./pmesh

rm wk.*
```

Computation + Visualization

```
>$ cd /home/ra020019/<Your-UID>/pFEM/pfem3d/runV
(INPUT.DAT, go.sh)
>$ pjsub go.sh
```

INPUT.DAT

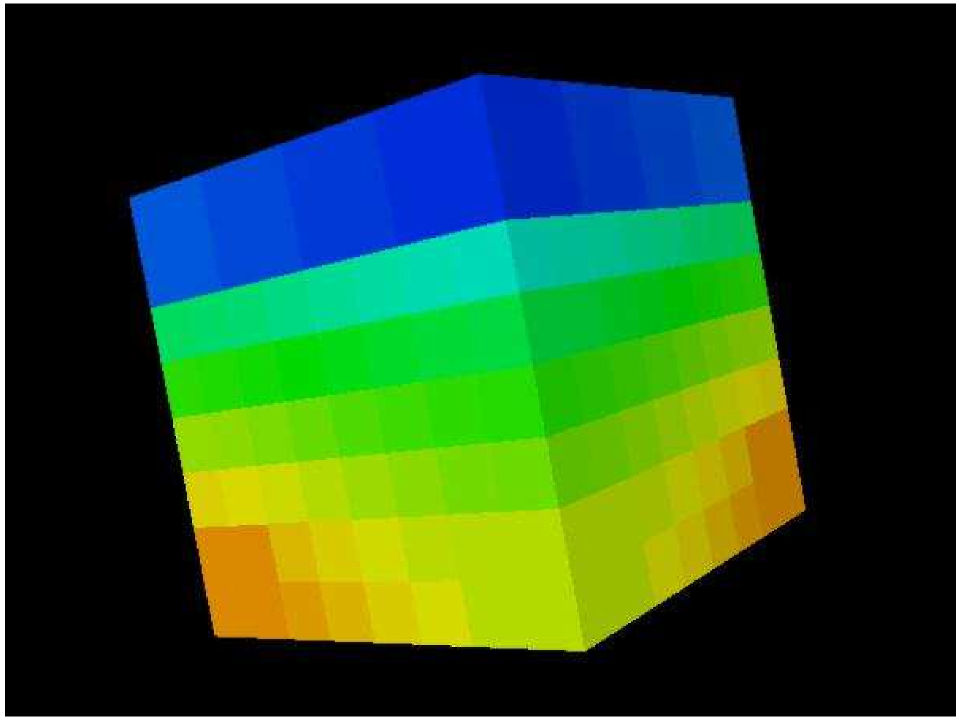
```
pcube
2000
1.0 1.0
1.0e-08
```

go.sh

```
#!/bin/sh
#PJM -N "VIS"
#PJM -L "rscgrp=small"
#PJM -L "node=8:torus"
#PJM --mpi "max-proc-per-node=32"
#PJM -L "elapse=00:15:00"
#PJM -g ra020019
#PJM -j
#PJM -e err
#PJM -o testV.lst

mpiexec ./solv
```

vis.cnt

```
[Refine]                    Control Info. for Refinement
AvailableMemory = 2.0       Available Memory (GB) not in use
MaxVoxelCount = 1000        Max Voxel #
MaxRefineLevel = 20         Max Voxel Refinement Level
[Simple]                    Control Info. for Simplification
ReductionRate = 0.0         Reduction Rate of Surface Patches
[Output]                    Output Format
FileFormat     = 2          =1:MicroAVS, =2:ParaView
```



Values at Cell Ctr.

16,777,216 nodes
16,581,375 elem's, 256 MPI proc's

vis_temp.1.inp
1,436 nodes
1,002 elements

COPY the File to your PC

from Fugaku

```
>$ scp <YourUID>@login.fugaku.r-ccs.riken.jp:/home/ra020019/<Your-UID>/pFEM/XXX .
```

to Fugaku

```
>$ scp YYY <YourUID>@login.fugaku.r-ccs.riken.jp:/home/ra020019/<Your-UID>/pFEM/.
```