

3D Parallel FEM (I)

C

Kengo Nakajima
RIKEN R-CCS

Target Application

- Parallel version of “heat3d”
- Using MPI

- Installation
- Execution
 - Procedures of Parallel FEM
 - Domain Decomposition/Partitioning
 - Real Execution
- Data Structure

Preparation (Odyssey)

FORTRAN

```
>$ cd /work/gt36/t36XYZ/pFEM  
>$ cp /work/gt00/z30088/pFEM/F/fem3d.tar .  
>$ tar xvf fem3d.tar
```

C

```
>$ cd /work/gt36/t36XYZ/pFEM  
>$ cp /work/gt00/z30088/pFEM/C/fem3d.tar .  
>$ tar xvf fem3d.tar
```

Confirmation

```
>$ ls  
    mpi    fem3d    pfem3d  
>$ cd pfem3d
```

Compilation (Odyssey)

Mesh Generator

```
>$ cd /work/gt36/t36XXX/pFEM/pfem3d/mesh  
>$ module load fj  
>$ frtpx -Kfast mgcube.f -o mgcube
```

Domain Partitioner

```
>$ cd ../part  
>$ module load metis/4.0.3  
>$ make  
>$ ls ../mesh/part  
part
```

Parallel FEM

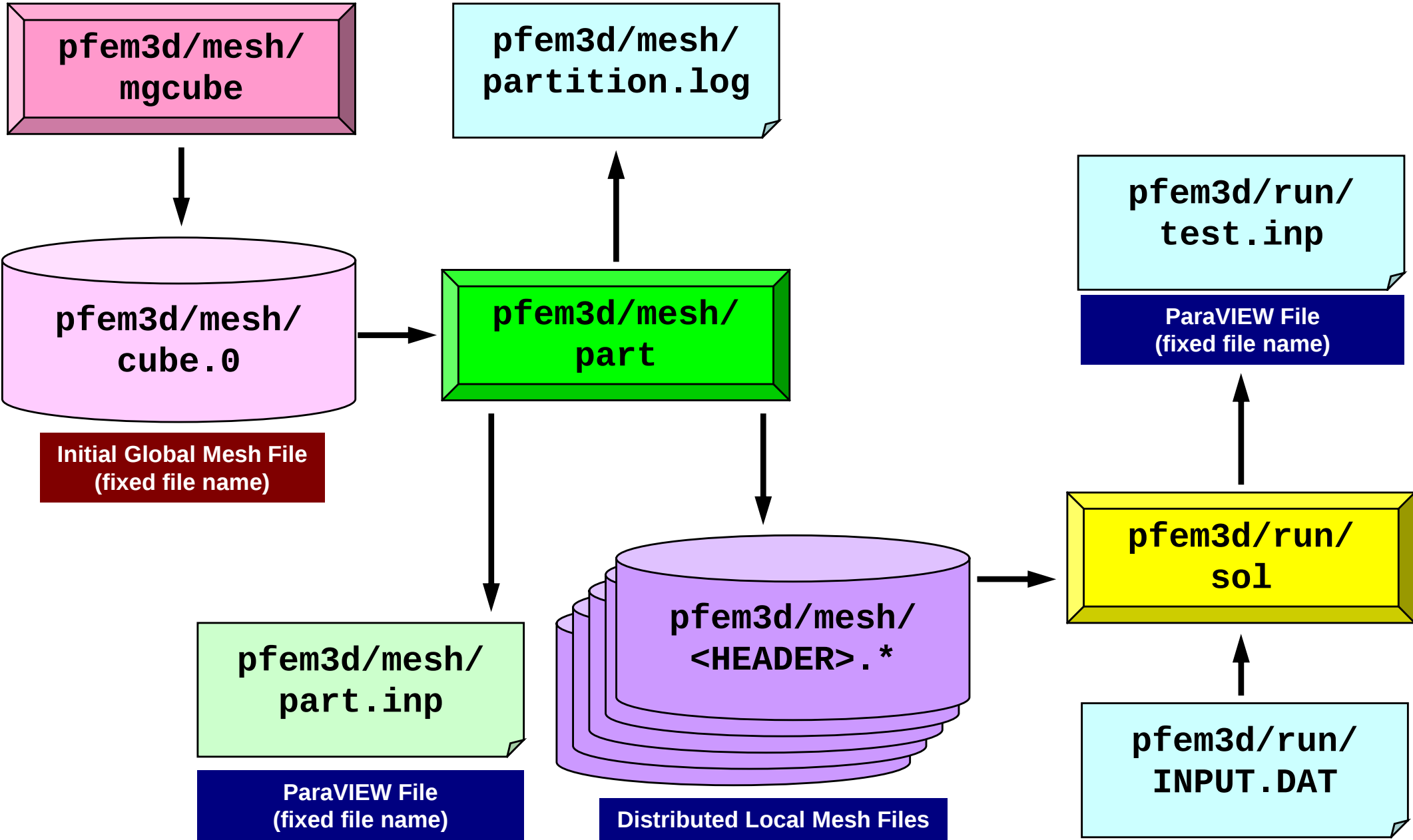
```
>$ cd ../src  
>$ make  
>$ ls ../run/sol  
sol
```

- Installation
- **Execution**
 - **Procedures of Parallel FEM**
 - **Domain Decomposition/Partitioning**
 - **Real Execution**
- Data Structure

Procedures for Parallel FEM

- Initial Global Mesh File
 - **/work/gt36/t36XYZ/pFEM/pfem3d/mesh/mg.sh**
- Distributed Local Mesh Files (Domain Partitioning)
 - **/work/gt36/t36XYZ/pFEM/pfem3d/mesh/part_XXX.sh**
- Parallel FEM Computation
 - **/work/gt36/t36XYZ/pFEM/pfem3d/run/XYZ.sh**

Procedures for Parallel FEM



- Installation
- Execution
 - Procedures of Parallel FEM
 - **Domain Decomposition/Partitioning**
 - Real Execution
- Data Structure

Partitioner

creates distributed local mesh files from
initial global mesh automatically

1D code: in parallel FEM program, 3D: too complicated

- Internal/External Points
 - Distributed Local Mesh Files
 - Numbering: Internal -> External pts.
- Communication Tables
 - Neighbors
 - Number of Neighbors
 - ID's of Neighbors
 - External Points
 - From where, how many, and which external points are received/imported ?
 - Boundary Points
 - To where, how many and which boundary points are sent/exported ?

What is Partitioning ?

- Graph/Graphic Partitioning
- Procedures/Operations of Domain Decomposition/Partitioning for Parallel Computing
- Creating Distributed Local Meshes from Huge Global Mesh which cannot be handled by a single PE

What is Graph/Graphic Partitioning

“Graph/Graphic Partitioning”: Application of “Graph Theory” for *graphs* (set of vertices and edges) to domain partitioning in parallel computing

- one-stroke sketch
- 4-color problem

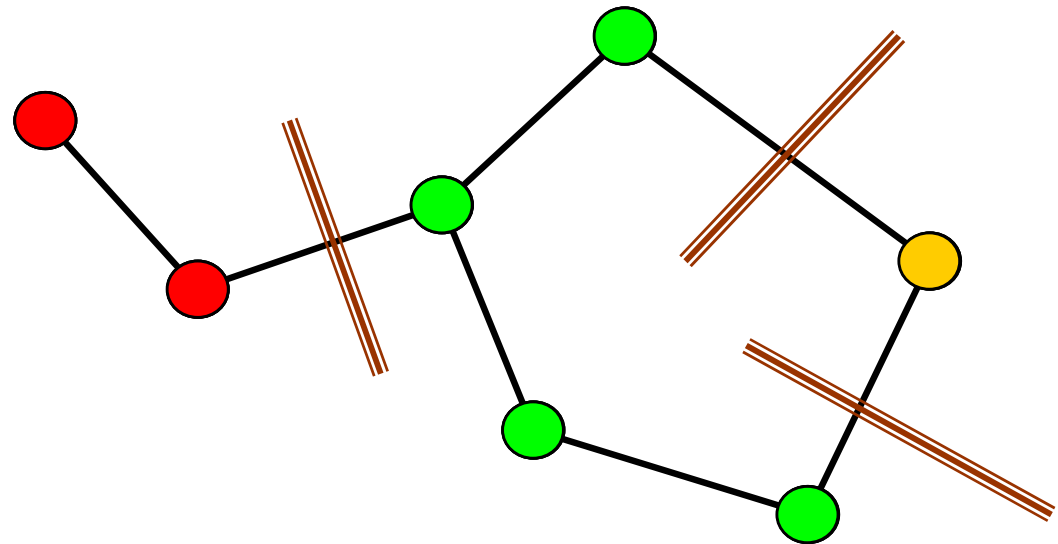
Good Partitioning

Load Balancing

Small Communications

Convergence of Preconditioned Iterative Solvers

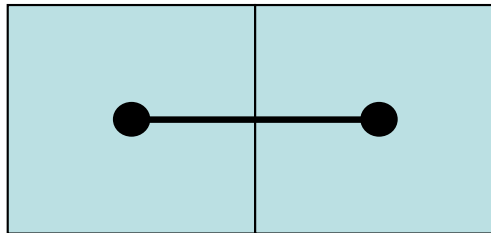
Minimum # of Neighbors



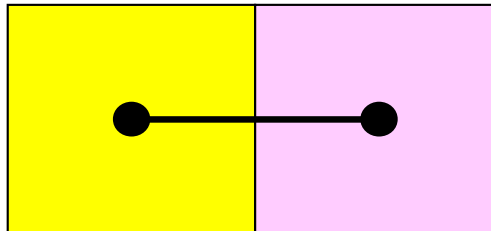
What is Edge-Cut ?

- If each of vertices of the edge belongs to different PE (domain, partition), “edge-cut” occurs
- Smaller number of edge-cut’s, smaller communications

No EDGE-CUT



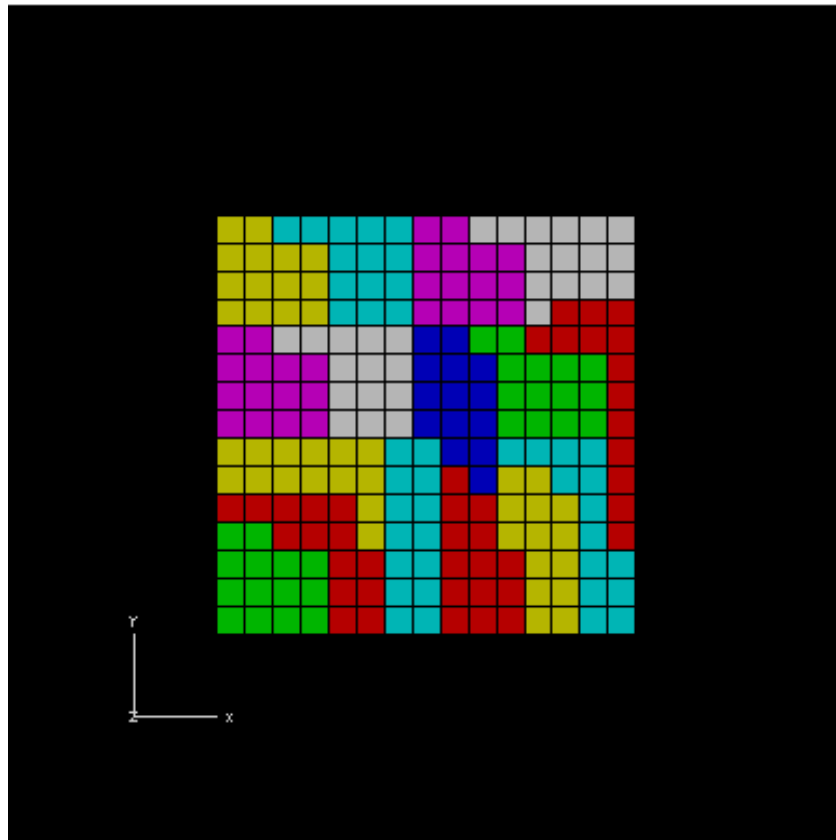
EDGE-CUT



Effect of Partitioning on Convergence

16 PE's for 2D (15×15) : Load Balanced

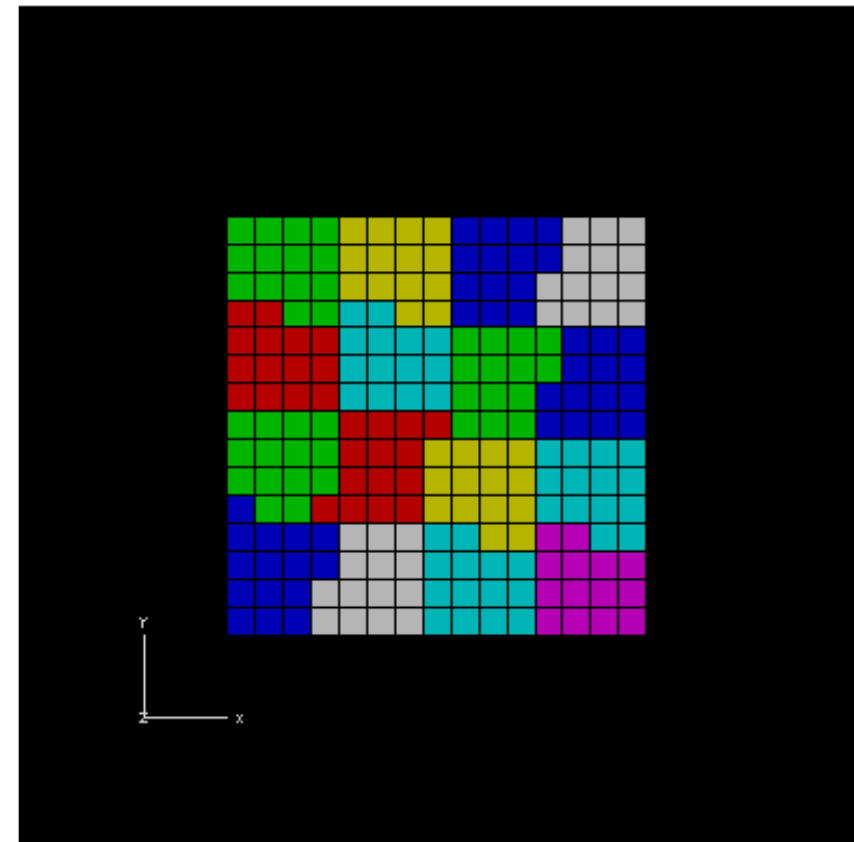
Many Edge-Cut's



RGB

Recursive Graph Bisection

Fewer Edge-Cut's



RSB

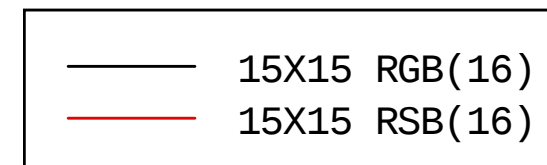
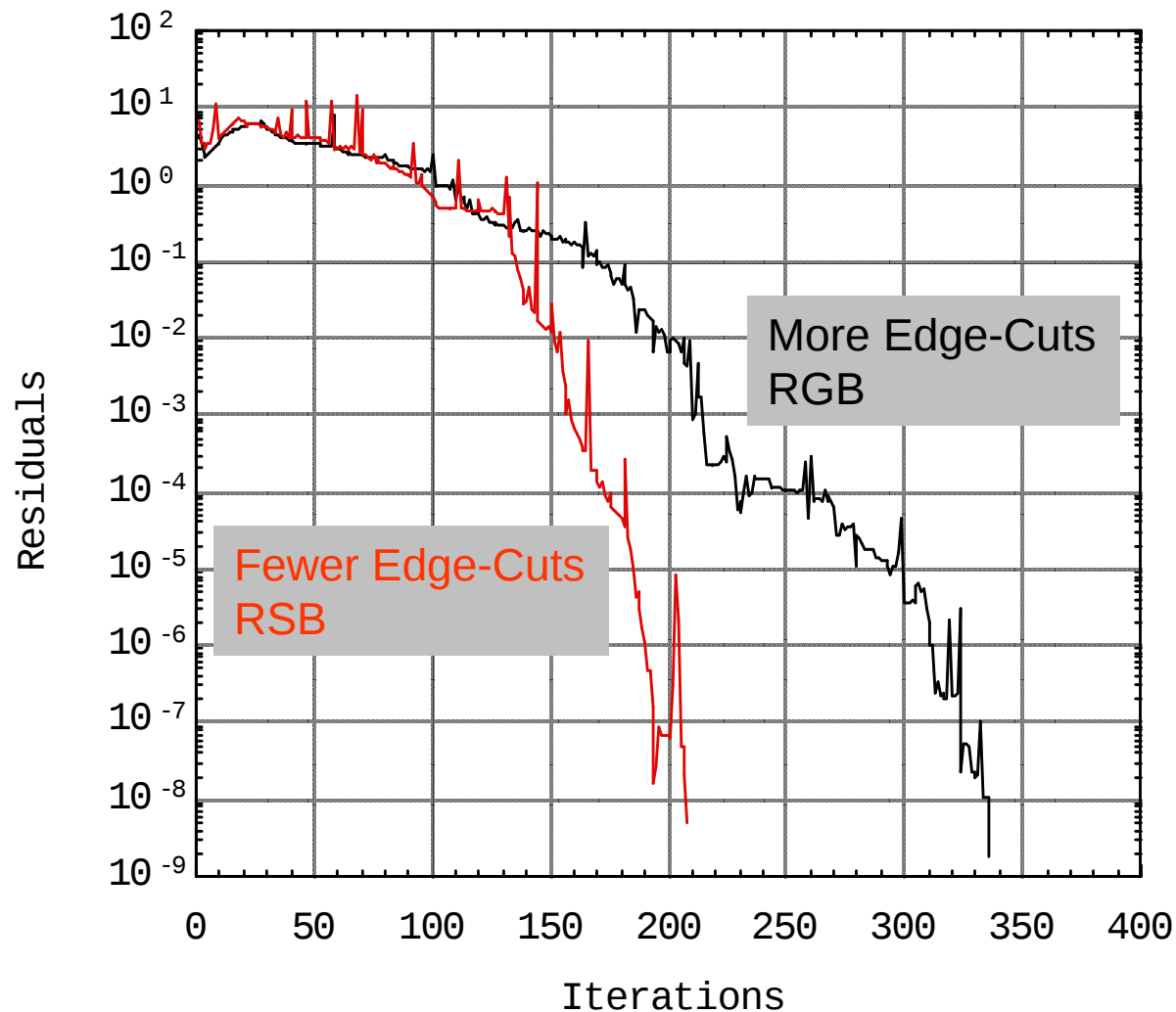
Recursive Spectral Bisection

Effect of Partitioning on Convergence

BiCGSTAB with Localized ILU(0) Preconditioning

15X15 region, RGB/RSB for 16 PE's , Poisson eqn's

Fewer "edge-cut's" (smaller comm.), faster convergence



	RGB	RSB
Neighboring PEs (Ave., max)	3.63, 7	3.63, 6
Boundary Edges (Ave, max)	15.1, 19	12.5, 18

Done in February 1996

Methods for Partitioning

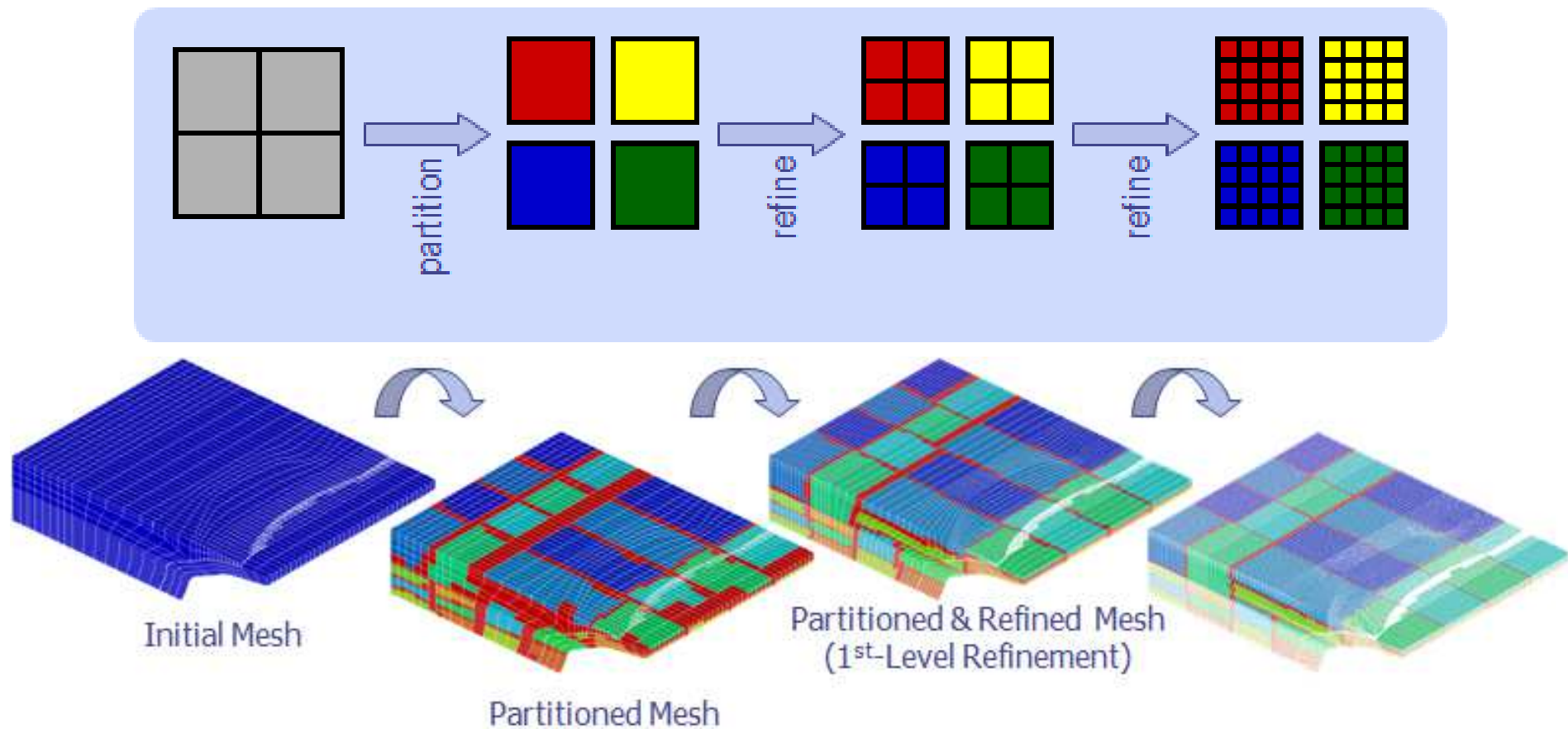
- Many research groups in late 1990's, but currently **MeTiS** and **Scotch/PT-Scotch** are two major tools.
- MeTiS : Univ.Minnesota
 - <http://glaros.dtc.umn.edu/gkhome/views/metis/>
- Scotch/PT-Scotch: developed recently
 - <http://www.labri.fr/perso/pelegrin/scotch/>
- JOSTLE : Univ.Greenwich
 - <http://staffweb.cms.gre.ac.uk/~c.walshaw/jostle/>

pFEM/pfem3d/mesh/part

- Tool which partitions initial global mesh file.
 - serial operation
- And creates distributed local mesh files with communication tables.
- Methods for Partitioning
 - RCB (Recursive Coordinate Bisection)
 - METIS
 - kmetis Minimum edge-cut's
 - pmetis Optimum load balancing

Actual Large-Scale Computations

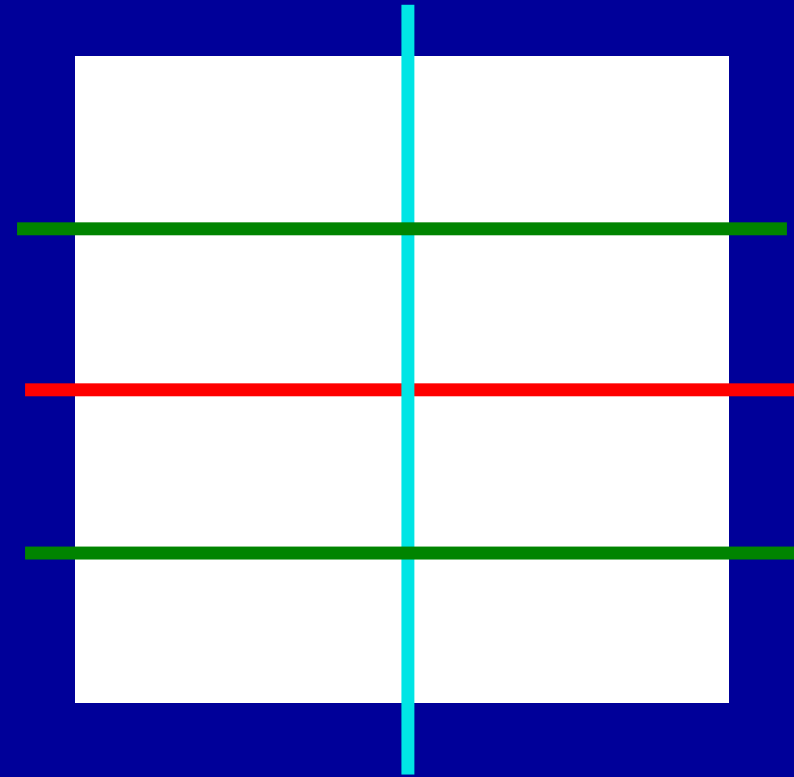
- Sometimes, it is difficult to prepare “initial global mesh”
- Starting from “coarse” initial mesh -> partitioning -> AMR (adaptive mesh refinement)



RCB: Recursive Coordinate Bisection

H.D.Simon "Partitioning of unstructured problems for parallel processing", Comp. Sys. in Eng., Vol.2, 1991.

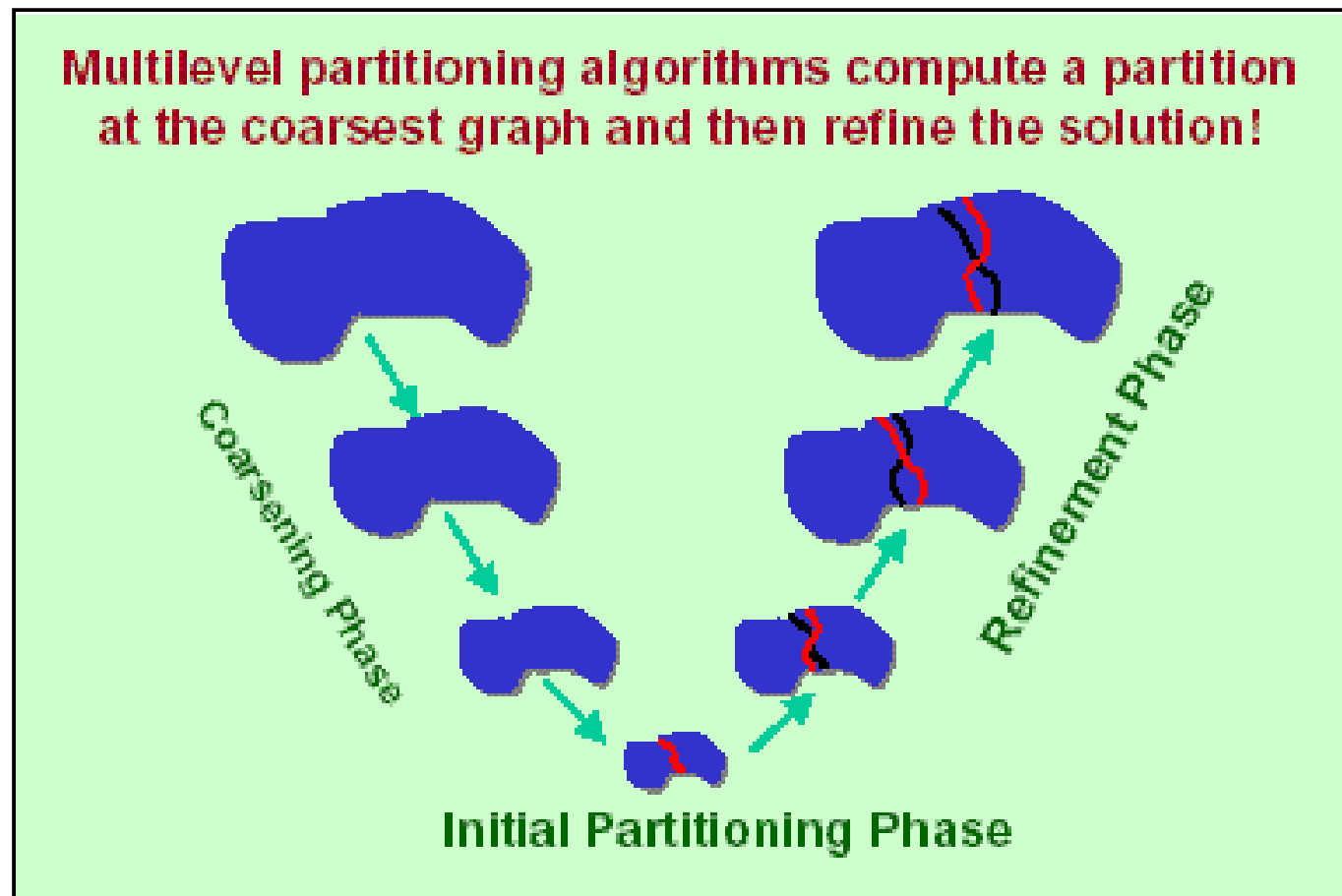
- Comparing X-Y-Z components
- Reference axis can be selected according to the geometry
- Continuous partitioning along X-axis for slender objects
- Only 2^n PE's
- Faster than **MeTiS** for simple geometry



MeTiS

<http://glaros.dtc.umn.edu/gkhome/views/metis/>

- based on Multi-Level Graph Theory



METIS

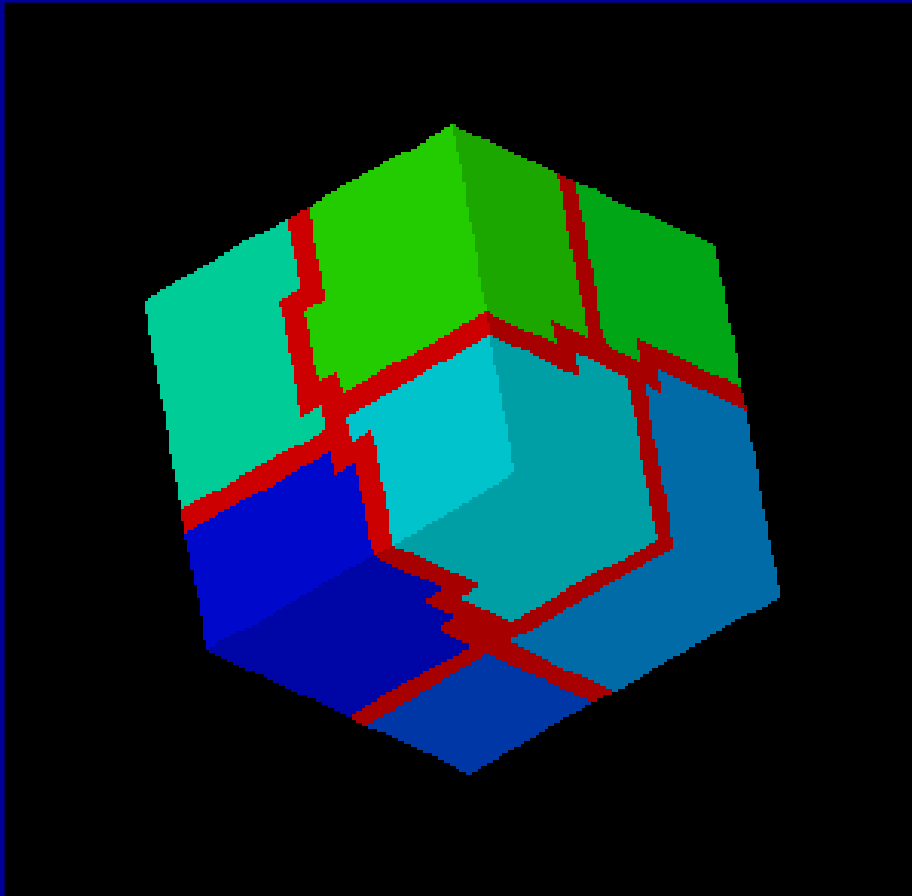
<http://glaros.dtc.umn.edu/gkhome/views/metis/>

- based on Multi-Level Graph Theory
 - minimize edge-cut's (communications)
 - stable, fast
 - free, both stand-alone and library versions
- Various Procedures
 - k-METIS Minimum Edge-Cut's
 - p-METIS Optimum Load Balancing
 - ParMETIS Parallel Version
 - applied to ordering, data-mining etc.
 - parallel contact search for crash problems

Example: Cubes: 8 PEs

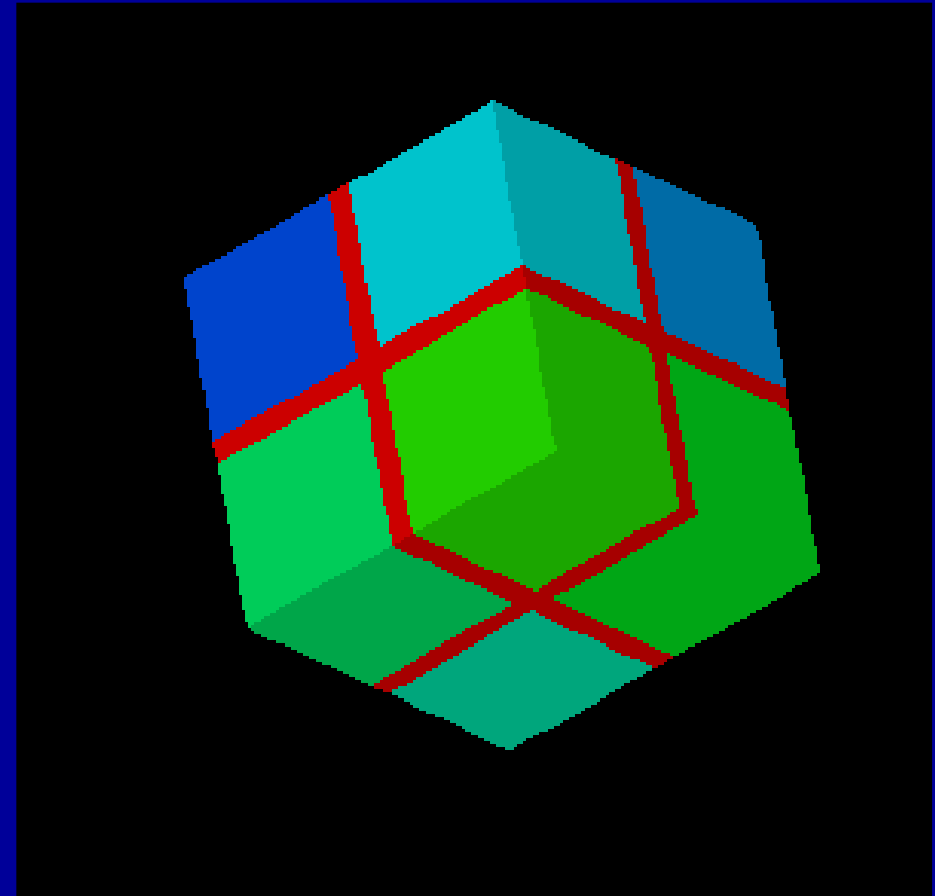
3,375 elements ($=15^3$), 4,096 nodes

RCB is good for simple geometries



k-METIS

edgecut = 882



RCB

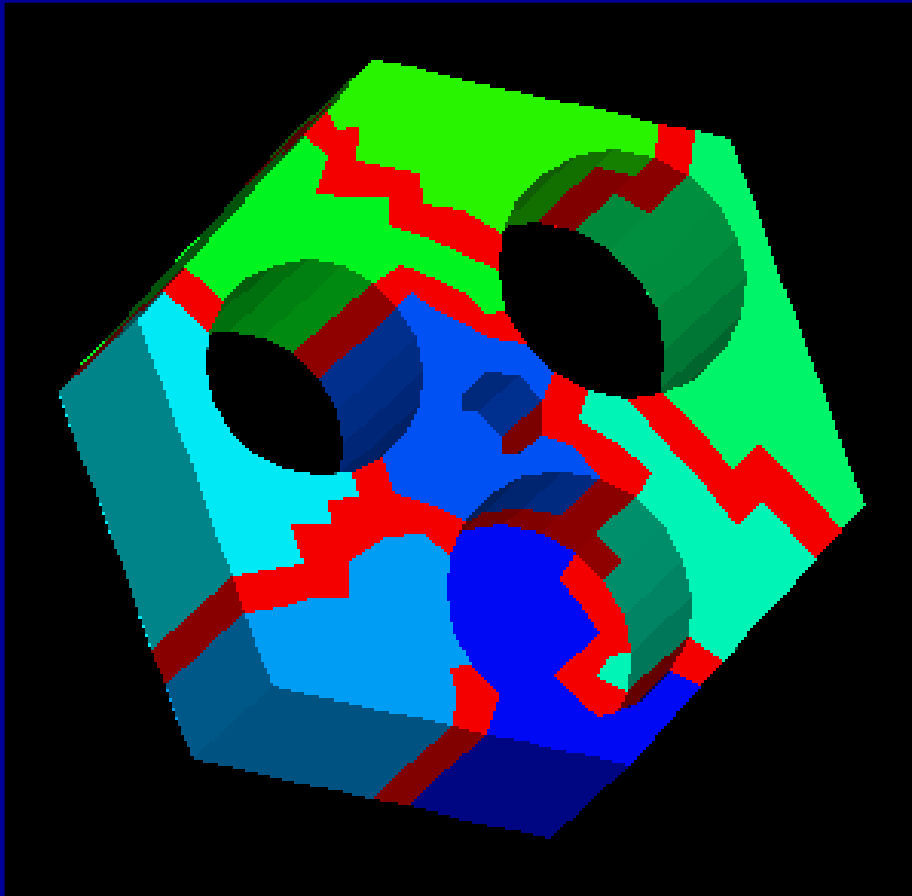
edgecut = 768

Example: Graphite Block: 8 PEs

795 elements, 1,308 nodes

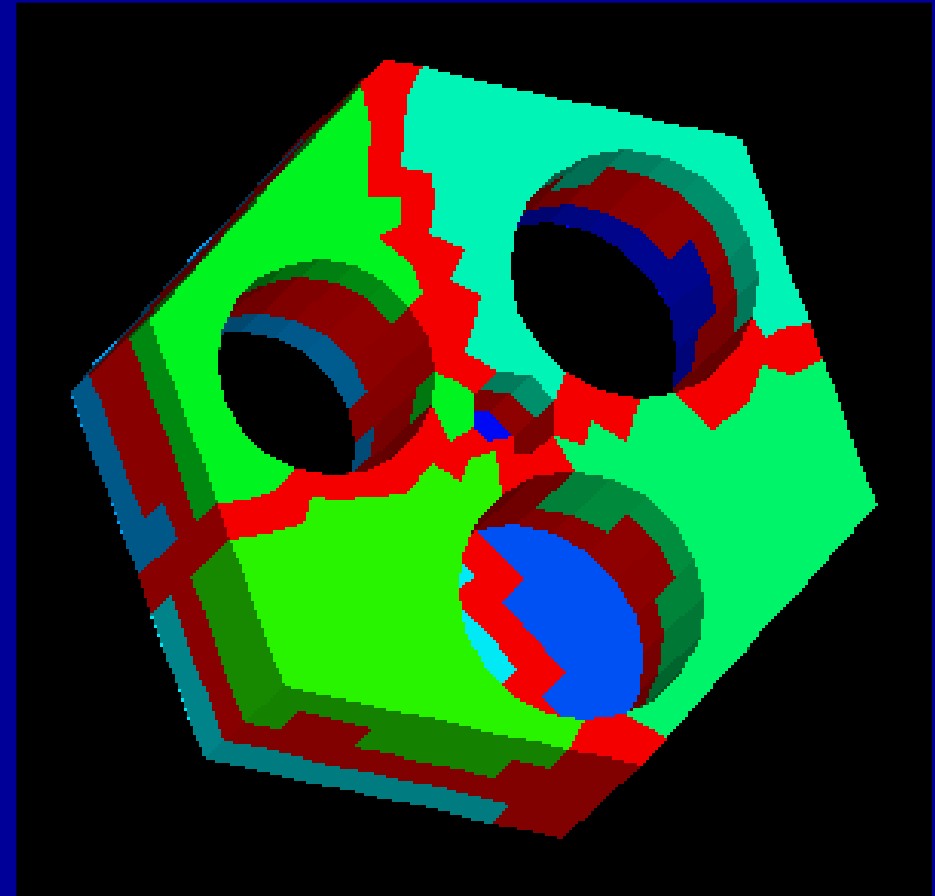
MeTiS is better for complicated geometries

Overlapping zones are thin



k-MeTiS

edgecut = 307



RCB

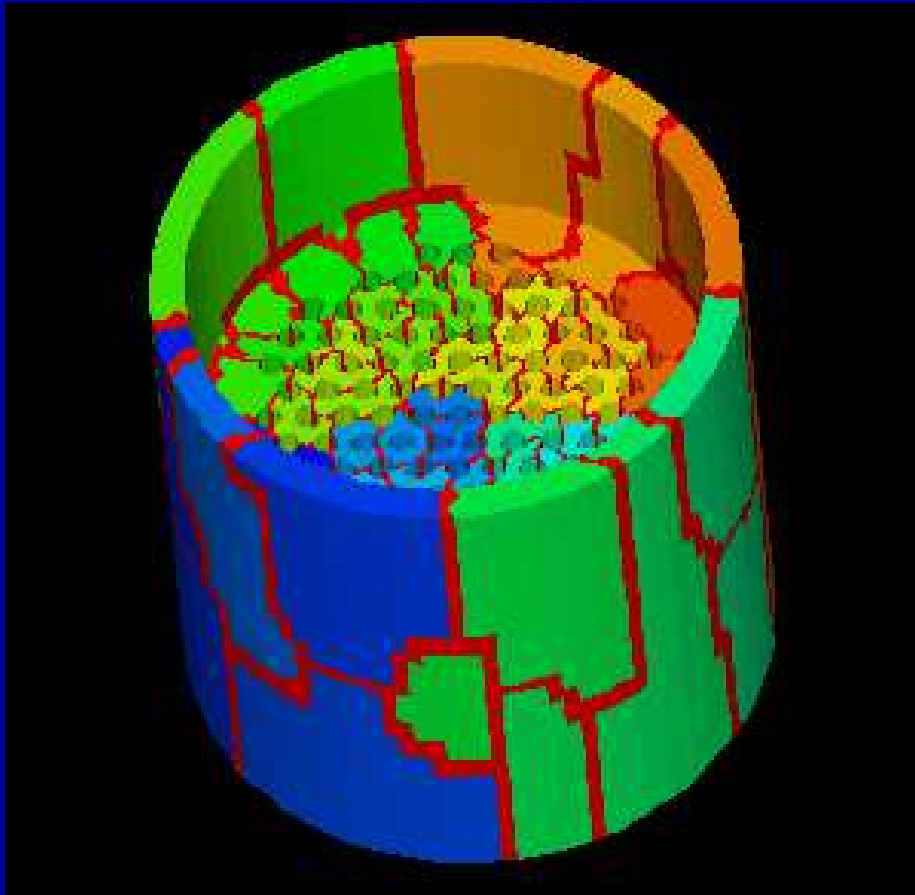
edgecut = 614

Example: Tube Sheet: 64 PEs

40,416 elements, 54,084 nodes

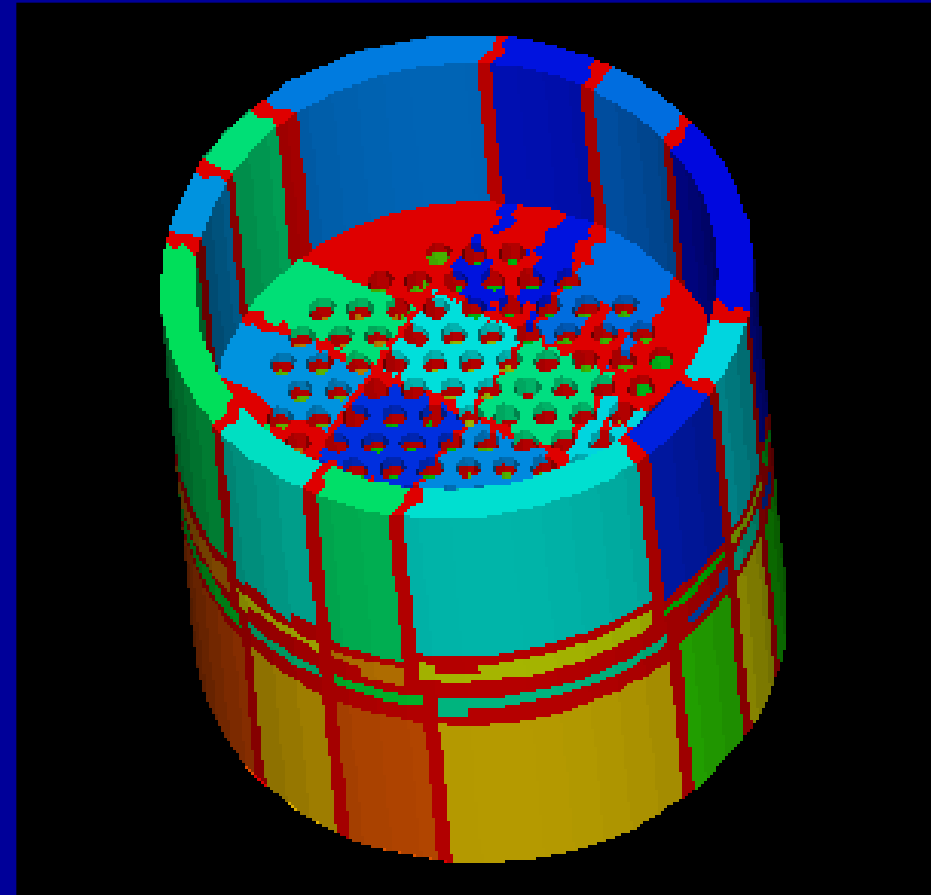
MeTiS is better for complicated geometries

Overlapping zones are thin



k-MeTiS

edgecut = 9,489



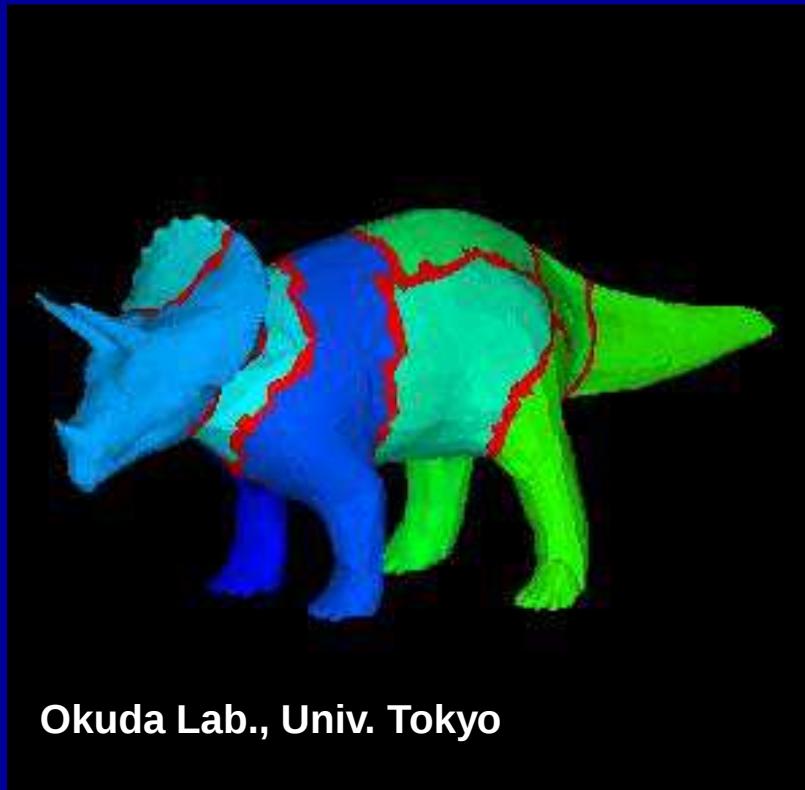
RCB

edgecut = 28,320

Strange Animal in 8 PEs

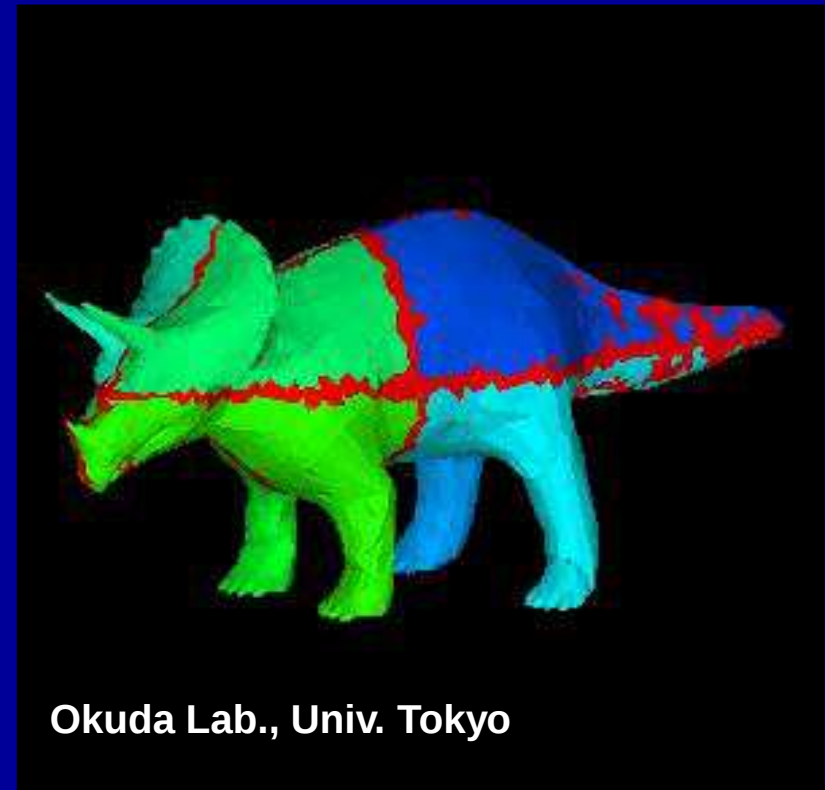
53,510 elements, 11,749 nodes.

METIS is better for complicated geometries.



Okuda Lab., Univ. Tokyo

k-METIS
edgecut = 4,573



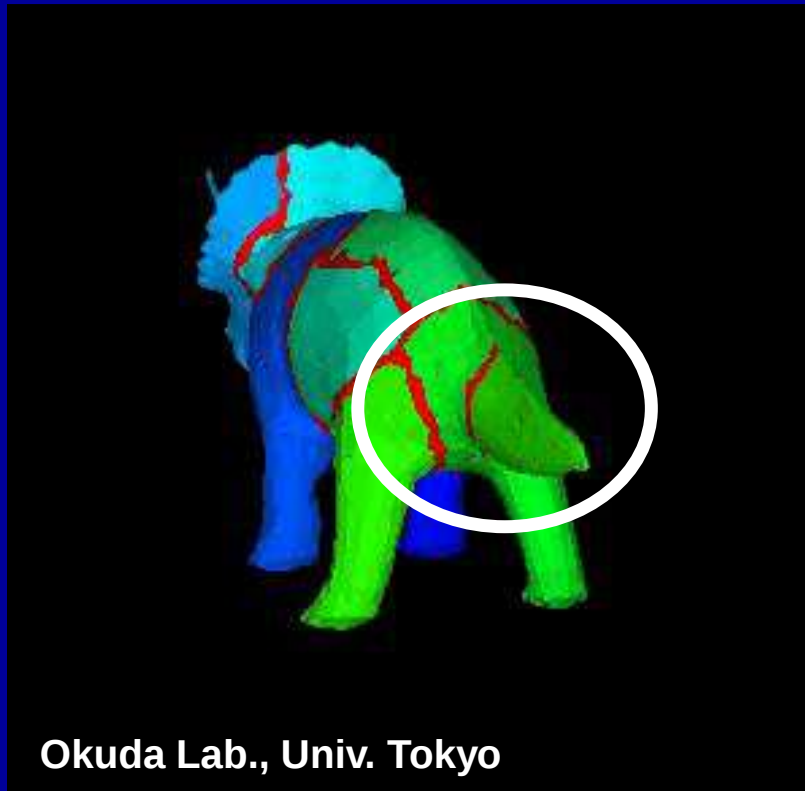
Okuda Lab., Univ. Tokyo

RCB
edgecut = 7,898

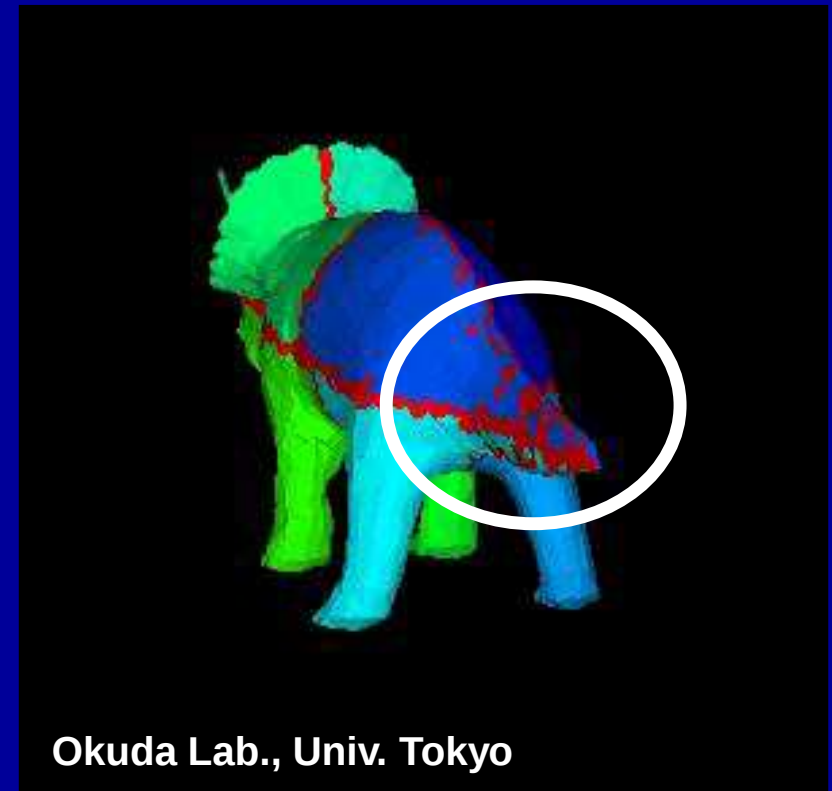
Strange Animal in 8 PEs

53,510 elements, 11,749 nodes.

MeTiS is better for complicated geometries



k-MeTiS
edgcut = 4,573



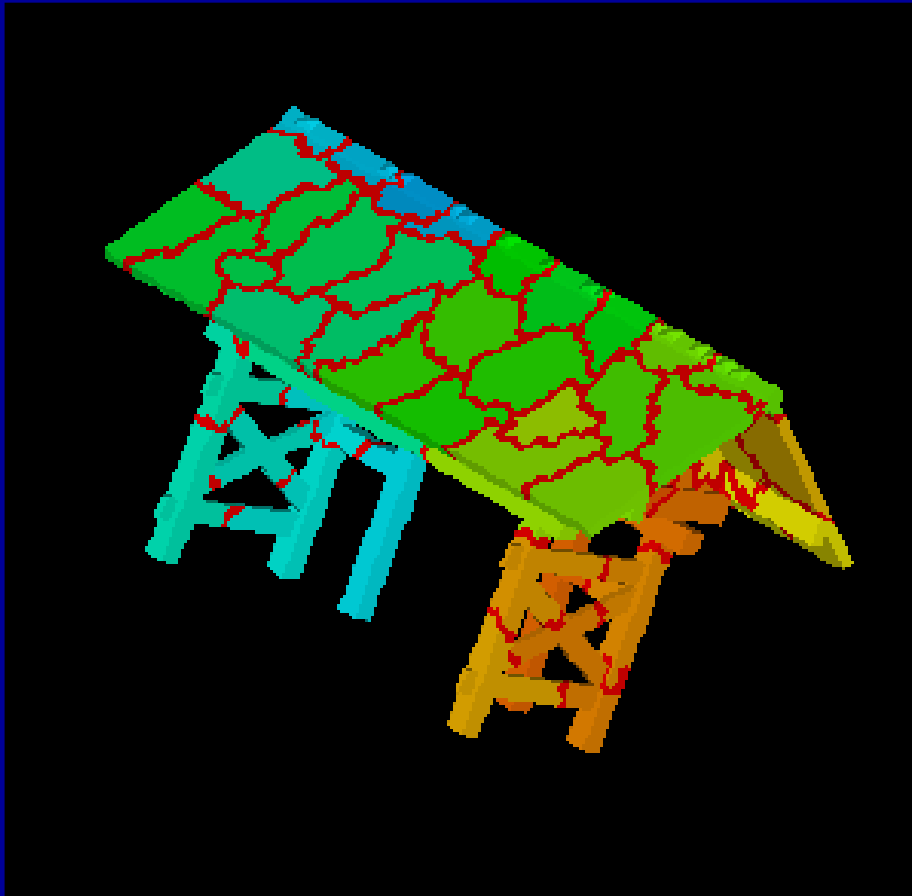
RCB
edgcut = 7,898

Red Lacquered Gate in 64 PEs

[movie](#)

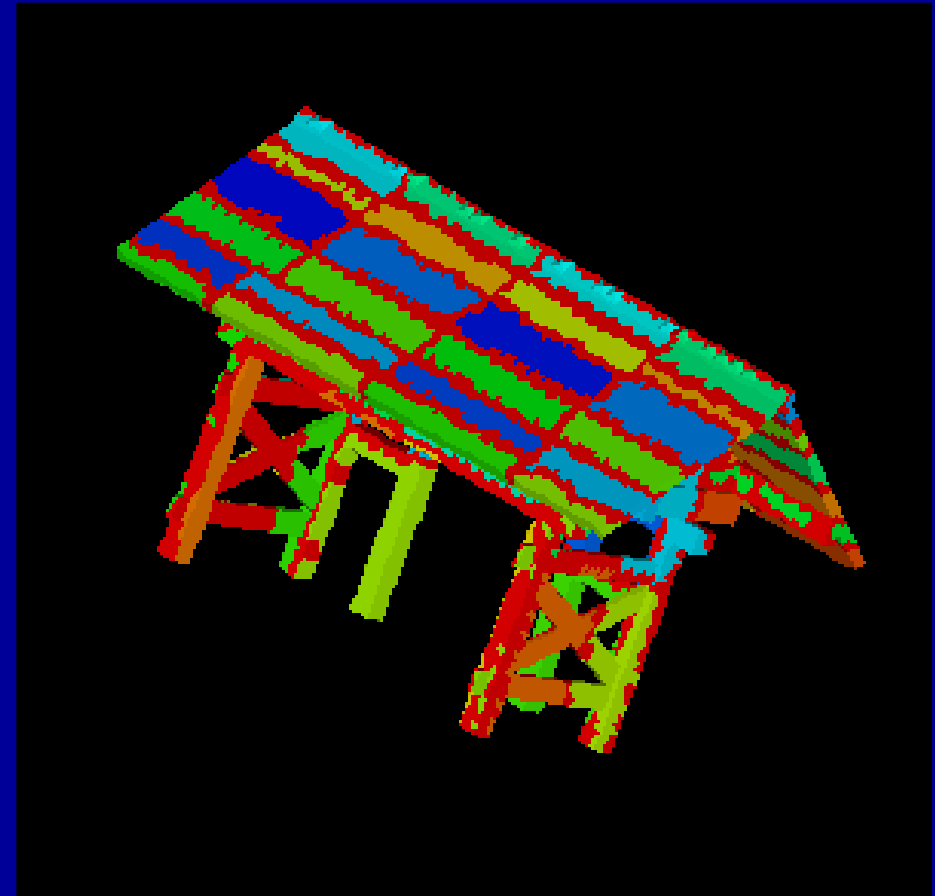
40,624 elements, 54,659 nodes

MeTiS is better for complicated geometries



k-MeTiS

edgecut = 7,563

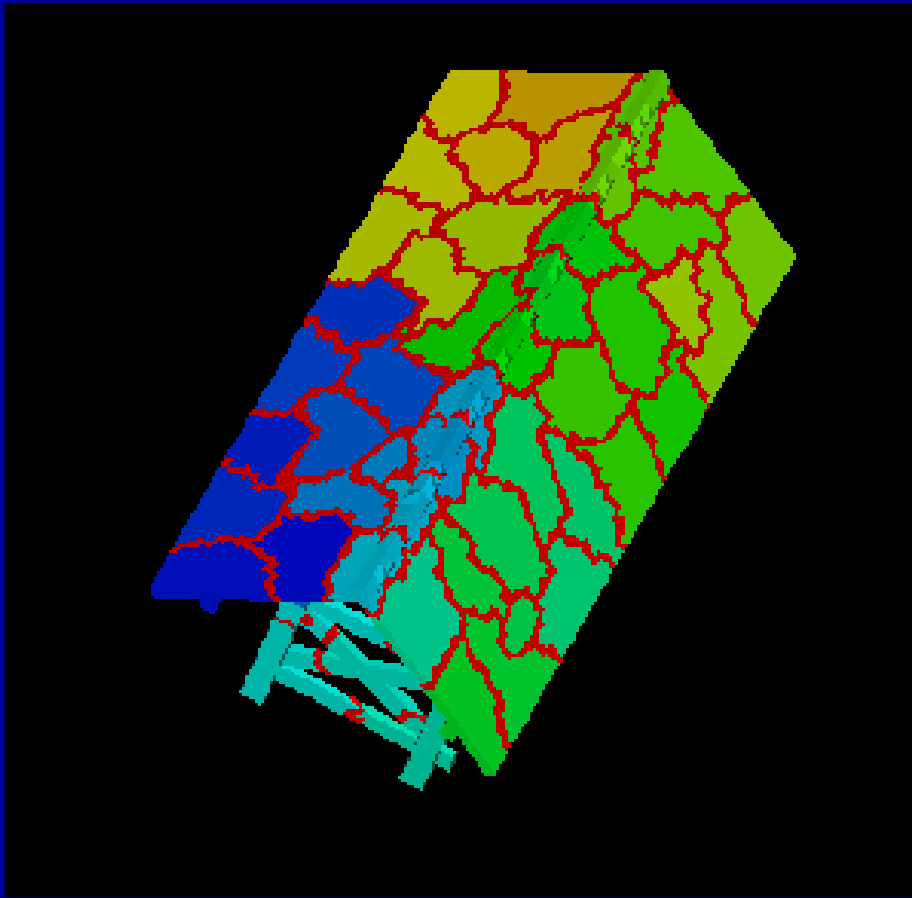


RCB

edgecut = 18,624

Red Lacquered Gate in 64 PEs

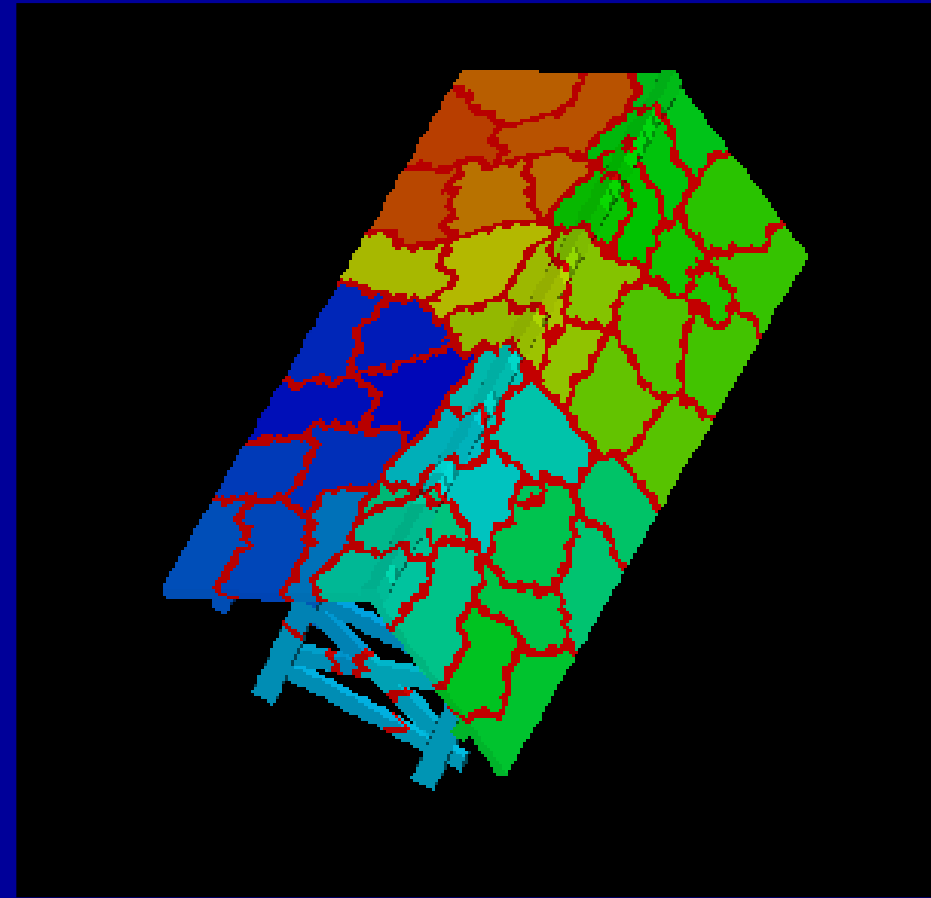
40,624 elements, 54,659 nodes



k-METIS

Load Balance= 1.03

edgecut = 7,563

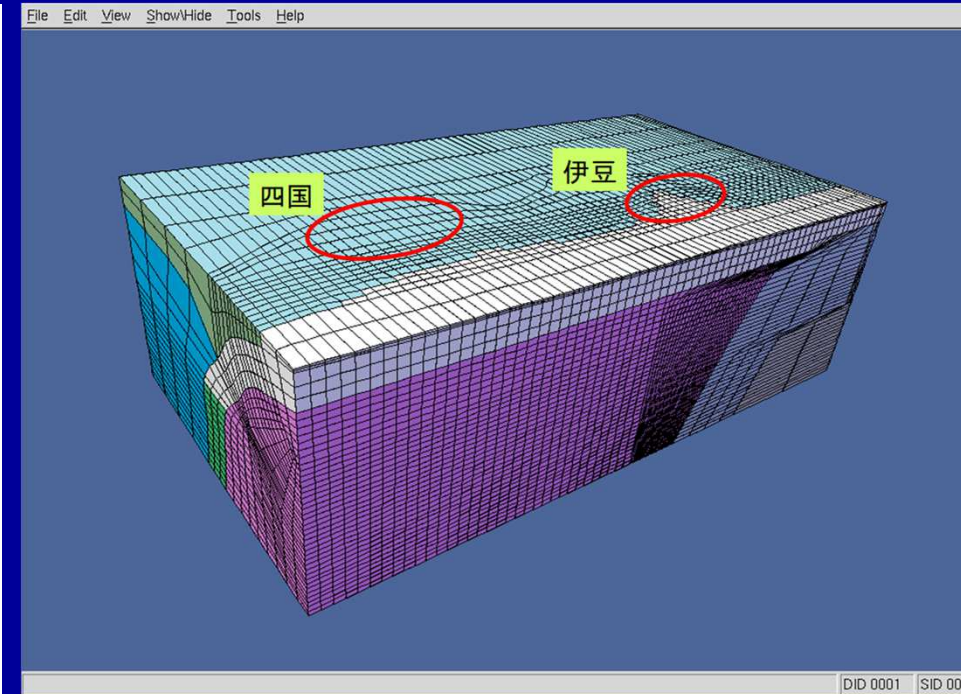
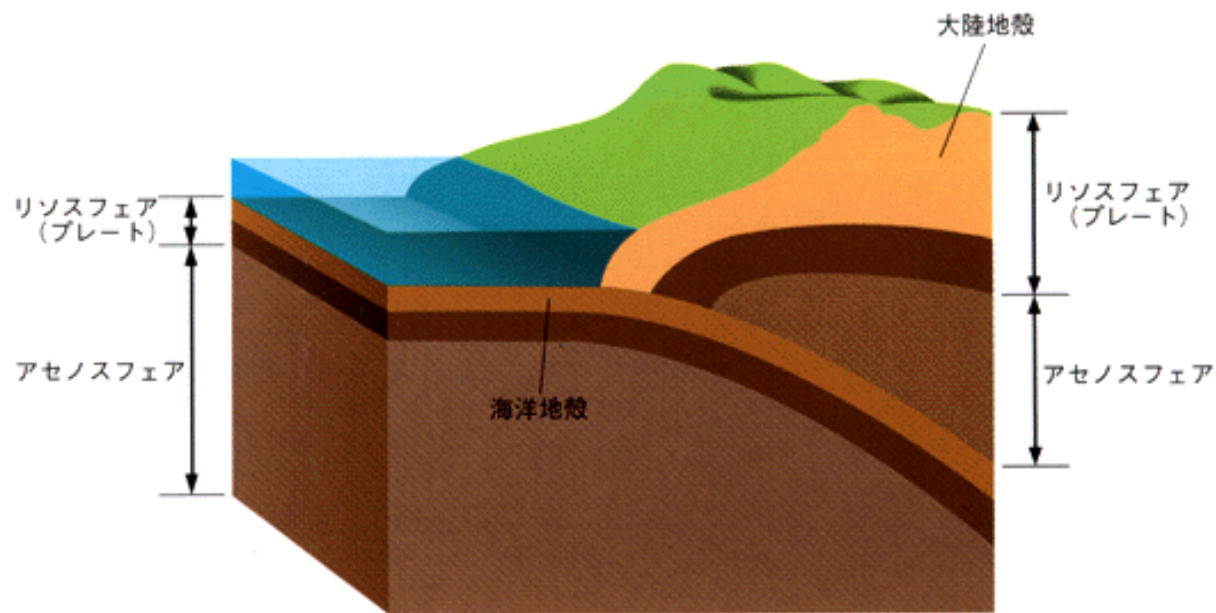


p-METIS

Load Balance= 1.00

edgecut = 7,738

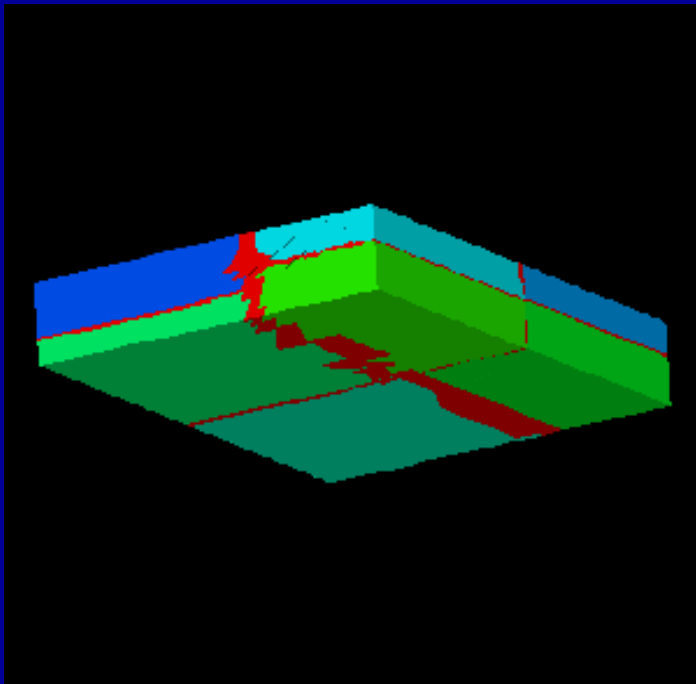
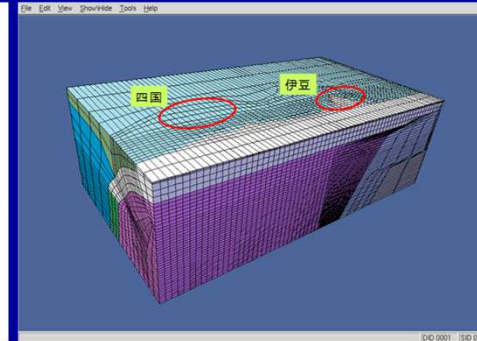
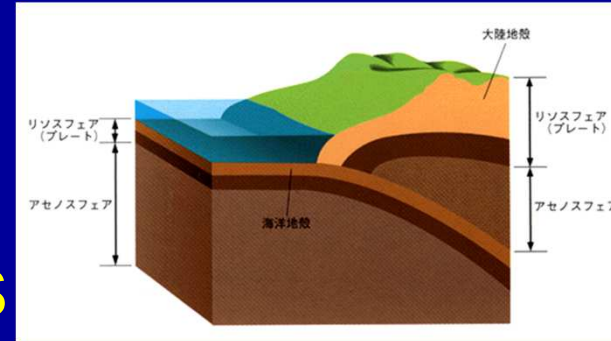
South-West Japan



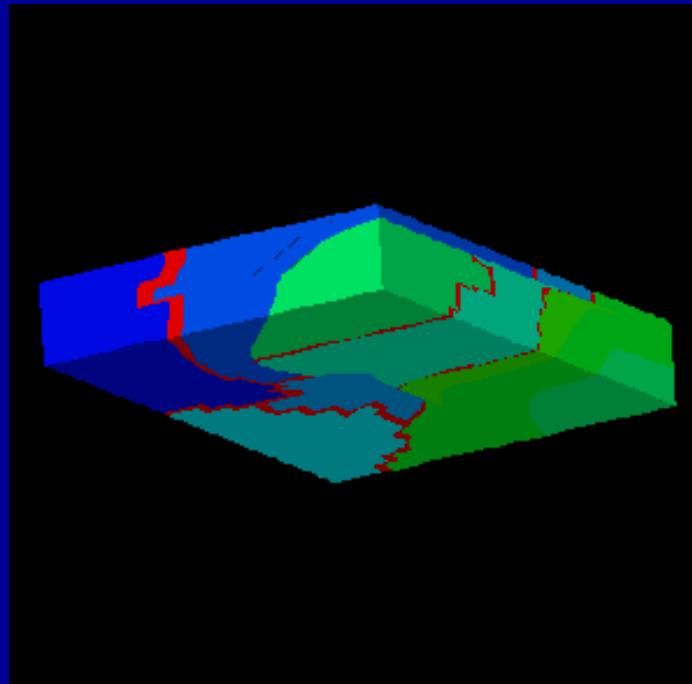
South-West Japan in 8 PEs

57,205 elem's, 58,544 nodes

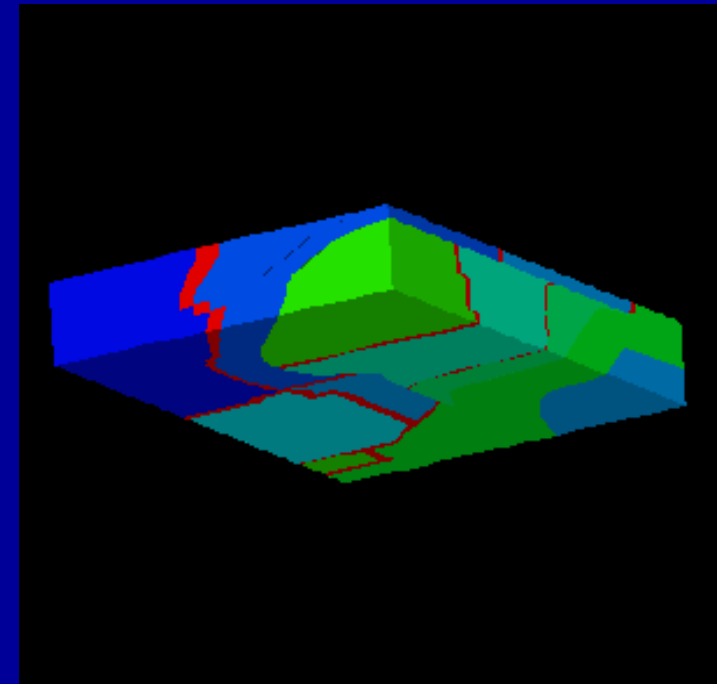
movie



RCB e.c.=7433



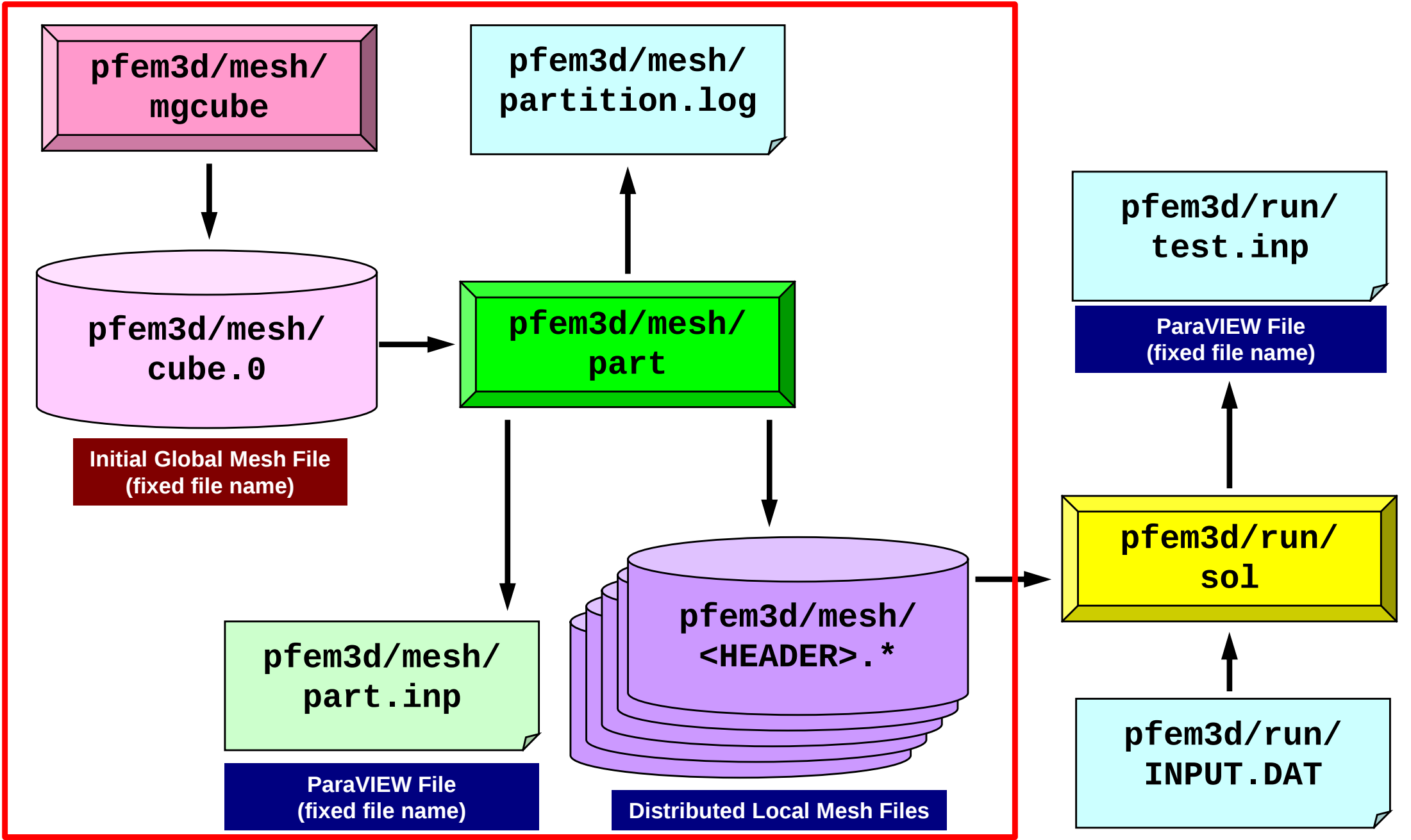
k-METIS :4,221



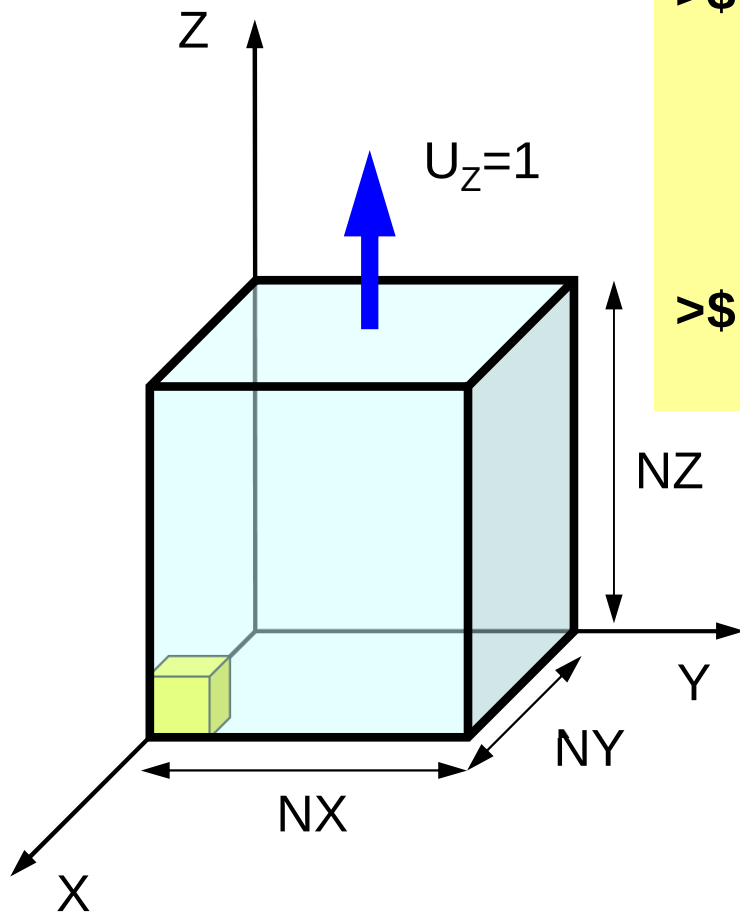
p-METIS :3,672

- Installation
- Execution
 - Procedures of Parallel FEM
 - Domain Decomposition/Partitioning
 - **Real Execution**
- Data Structure

Procedures for Parallel FEM



Initial Global Mesh



```
>$ cd /work/gt36/t36XYZ/pFEM/pfem3d/mesh  
>$ ./mgcube
```

NX, NY, NZ

← **Mesher in each
direction**

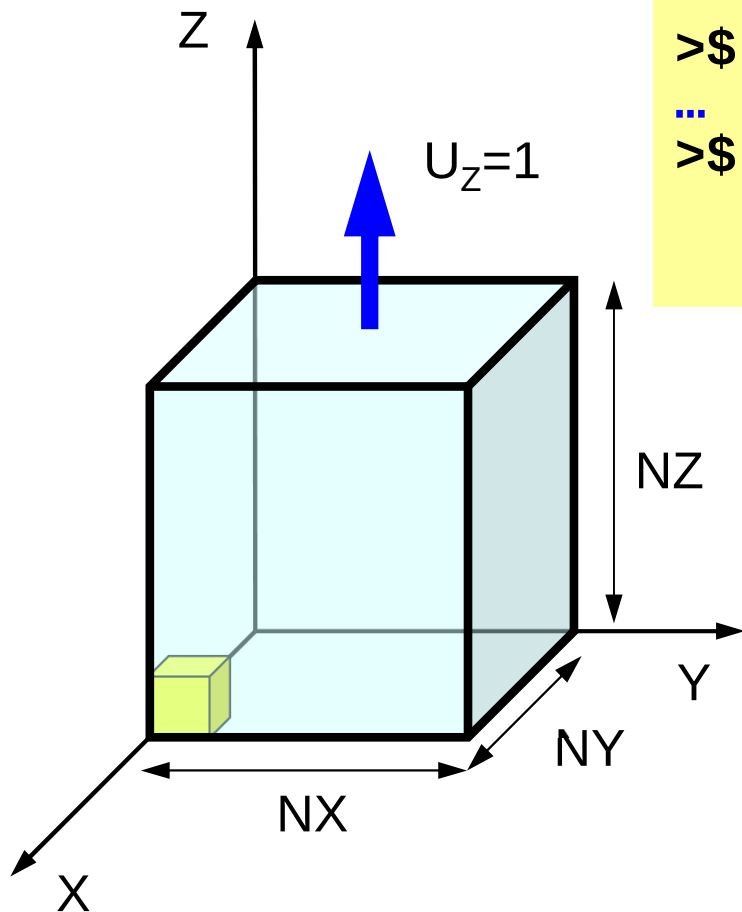
20 20 20

← **20x20x20 elem's**

```
>$ ls cube.0      confirmation  
    cube.0
```

This type of interactive execution is not allowed for “education” users on Odyssey.

Please submit batch-job's !



```
>$ cd /work/gt36/t36XYZ/pFEM/pfem3d/mesh
>$ pjsub mg.sh
...
>$ ls cube.0      confirmation
cube.0
```

mg.sh

```
#!/bin/sh
#PJM -L rscgrp=lecture6-o
#PJM -L node=1
#PJM --mpi proc=1
#PJM -L elapsed=00:15:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o mg.lst
```

```
module load fj
module load fjmpi
```

```
./mgcube < inp_mg
```

inp_mg

```
20 20 20
```

Domain Decomposition/Partitioning

- File name of initial global mesh is fixed (cube.0)
- RCB and METIS are supported
- Header of distributed local mesh files
 - “work” is not allowed as header name
- RCB
 - Number of PE's, Reference axes
- METIS
 - Number of PE's

pFEM/pfem3d/part/Makefile

```
F77      = frtpx
F90      = frtpx
FLINKER  = $(F77)
F90LINKER = $(F90)
FLIB_PATH =
INC_DIR  =
OPTFLAGS = -Kfast
FFLAGS   = $(OPTFLAGS)
FLIBS    = -lmetis

TARGET = ../mesh/part
default: $(TARGET)
OBJS =¥
geofem_util.o partitioner.o input_grid.o main.o ¥
calc_edgcut.o cre_local_data.o define_file_name.o ¥
interface_nodes.o metis.o¥
neib_pe.o paraset.o proc_local.o local_data.o¥
double_numbering.o output_ucd.o util.o

$(TARGET): $(OBJS)
            $(F90LINKER) $(OPTFLAGS) -o $(TARGET) $(OBJS) $(FLIBS)
clean:
    /bin/rm -f *.o $(TARGET) *~ *.mod
.f.o:
    $(F90) $(FFLAGS) $(INC_DIR) -c  $*.f
.SUFFIXES: .f
```

```

>$ cd /work/gt87/t87XYZ/pFEM/pfem3d/mesh
>$ ./part

Original GRID-FILE ?
cube.0
* INODTOT =      9261
* GRID
* IELMTOT =      8000
* ELM
* BOUNDARY : NODE group
Xmin
Ymin
Zmin
Zmax
* IEDGTOT =      26460      37044

# select PARTITIONING METHOD
RCB                (1)
K-METIS            (2)
P-METIS            (3)

Please TYPE 1 or 3 or 4 !!

>>>
1

*** RECURSIVE COORDINATE BISECTION (RCB)
How many partitions (2**n)?

>>>
3

***      8 REGIONS

```

```

# HEADER of the OUTPUT file ?
HEADER should not be <work>

>>>
aaa

##### 1-th BiSECTION #####

in which direction ? X:1, Y:2, Z:3

>>>
1
X-direction

##### 2-th BiSECTION #####

in which direction ? X:1, Y:2, Z:3

>>>
2
Y-direction

##### 3-th BiSECTION #####

in which direction ? X:1, Y:2, Z:3

>>>
3
Z-direction

RECURSIVE COORDINATE BISECTION

*** GRID file

      8 PEs

TOTAL EDGE      #      26460
TOTAL EDGE CUT #      1593

TOTAL NODE      #      9261
TOTAL CELL      #      8000

```

PE	NODE#	CELL#						
0	1158	1223						
1	1158	1188						
2	1158	1222						
3	1158	1176						
4	1158	1188						
5	1157	1179						
6	1157	1188						
7	1157	1175						
MAX.node/PE			1158					
MIN.node/PE			1157					
MAX.cell/PE			1223					
MIN.cell/PE			1175					
OVERLAPPED ELEMENTS			1373					
PE/NEIB-PE#		NEIB-PEs						
0	7	7	6	4	5	2	1	3
1	7	7	5	6	0	2	4	3
2	7	7	6	0	5	1	4	3
3	6	7	2	6	1	5	0	
4	6	6	7	5	0	2	1	
5	7	7	6	4	0	1	2	3
6	7	7	5	4	0	2	1	3
7	7	6	5	4	0	2	1	3
PE:	0	1626	1158	468	435			
PE:	1	1589	1158	431	411			
PE:	2	1620	1158	462	490			
PE:	3	1560	1158	402	409			
PE:	4	1574	1158	416	421			
PE:	5	1565	1157	408	397			
PE:	6	1580	1157	423	414			
PE:	7	1564	1157	407	440			
(Int.+Ext.) Internal External Boundary								
KCHF091R STOP * normal termination								

```
>$ ls -l aaa.*  
  
-rw-r--r-- 1 t36013 t36 268829 Jan 12 14:57 aaa.0  
-rw-r--r-- 1 t36013 t36 261490 Jan 12 14:57 aaa.1  
-rw-r--r-- 1 t36013 t36 268086 Jan 12 14:57 aaa.2  
-rw-r--r-- 1 t36013 t36 257631 Jan 12 14:57 aaa.3  
-rw-r--r-- 1 t36013 t36 258719 Jan 12 14:57 aaa.4  
-rw-r--r-- 1 t36013 t36 256853 Jan 12 14:57 aaa.5  
-rw-r--r-- 1 t36013 t36 259093 Jan 12 14:57 aaa.6  
-rw-r--r-- 1 t36013 t36 257161 Jan 12 14:57 aaa.7
```

- Distributed Local Files
 - <HEADER>.<ID of PEs>
 - ID of PEs starting from “0”

Again, this interactive operation is not allowed !

Please submit batch-job's !

RCB: part_rcb.sh

inp_rcb

part_rcb.sh

```
#!/bin/sh
#PJM -N "RCB"
#PJM -L rscgrp=lecture6-o
#PJM -L node=1
#PJM --mpi proc=1
#PJM -L elapse=00:15:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o test.lst
```

```
module load fj
module load fjmpi
module load metis/4.0.3
```

```
./part < inp_rcb
rm work.*
```

inp_rcb

```
cube.0    Initial Global File (fixed)
1          1:RCB, 2:KMETIS, 3:PMETIS
3          m: 2m PE's
aaa        Header of Distributed Local Files
1          Reference Axis (X:1, Y:2, Z:3)
2
3
```

inp_rcb: 1-PE

```
cube.0    Initial Global File (fixed)
1          1:RCB, 2:KMETIS, 3:PMETIS
0          m: 2m PE's
aaa        Header of Distributed Local Files
```

kmetis: part_kmetis.sh inp_kmetis

Minimum Edge-Cut

part_kmetis.sh

```
#!/bin/sh
#PJM -N "K-MeTiS"
#PJM -L rscgrp=lecture6-o
#PJM -L node=1
#PJM --mpi proc=1
#PJM -L elapse=00:15:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o test.lst
```

```
module load fj
module load fjmpi
module load metis/4.0.3
```

```
./part < inp_kmetis
rm work.*
```

inp_kmetis

cube.0	Initial Global File (fixed)
2	1:RCB, 2:KMETIS, 3:PMETIS
8	Number of PE's
aaa	Header of Distributed Local Files

pmetis: part_pmetis.sh inp_pmetis

Optimum Load-Balancing

part_pmetis.sh

```
#!/bin/sh
#PJM -N "P-MeTiS"
#PJM -L rscgrp=lecture6-o
#PJM -L node=1
#PJM --mpi proc=1
#PJM -L elapse=00:15:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o test.lst
```

```
module load fj
module load fjmpi
module load metis/4.0.3
```

```
./part < inp_pmetis
rm work.*
```

inp_pmetis

cube.0	Initial Global File (fixed)
3	1:RCB, 2:KMETIS, 3:PMETIS
8	Number of PE's
aaa	Header of Distributed Local Files

partition.log

```

RECURSIVE COORDINATE BISECTION

***  GRID  file

      8  PEs

TOTAL  EDGE      #      26460
TOTAL  EDGE CUT  #      1593

TOTAL  NODE      #      9261
TOTAL  CELL      #      8000

PE      NODE#    CELL#
 0      1158     1223
 1      1158     1188
 2      1158     1222
 3      1158     1176
 4      1158     1188
 5      1157     1179
 6      1157     1188
 7      1157     1175

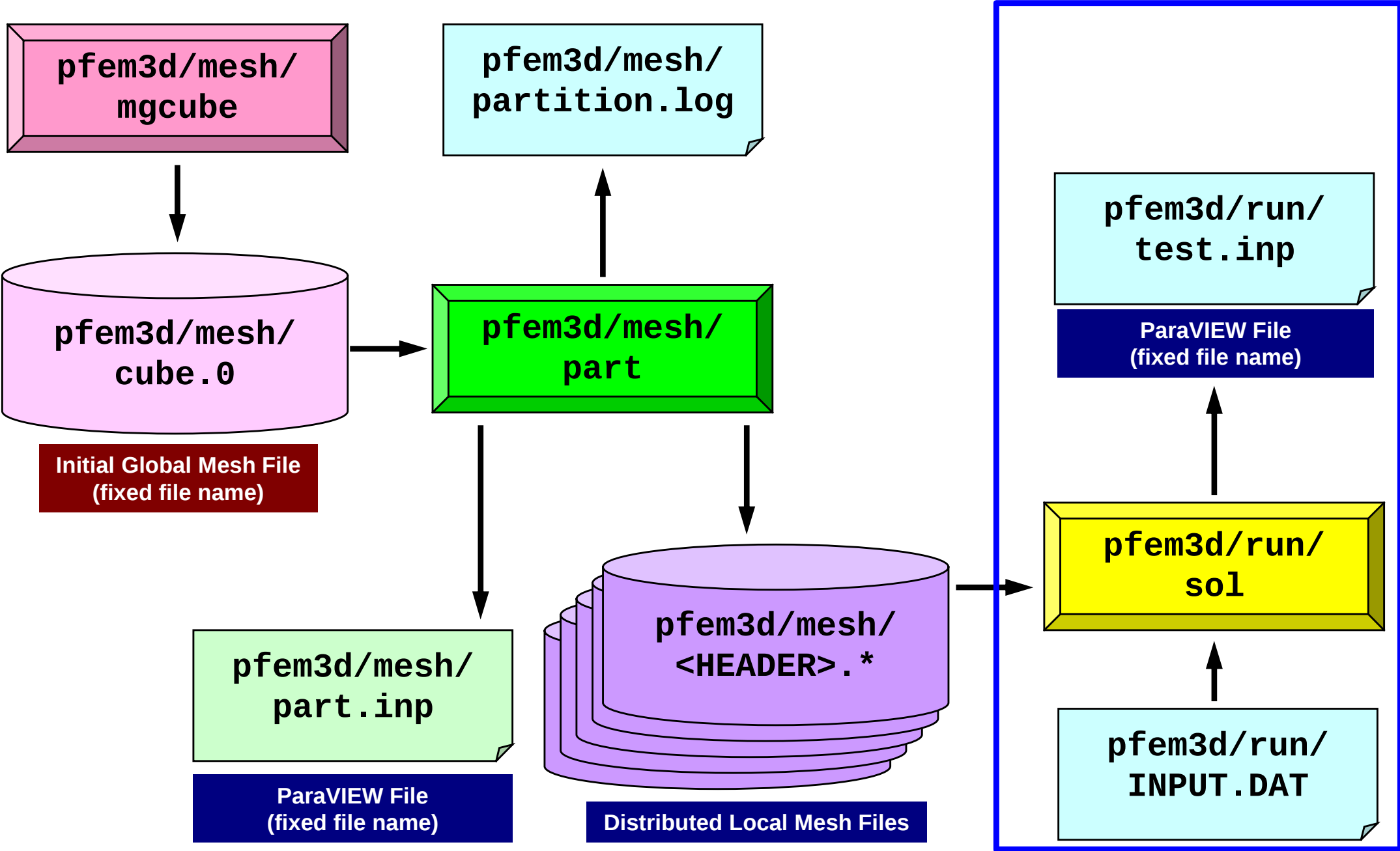
MAX.node/PE      1158
MIN.node/PE      1157
MAX.cell/PE      1223
MIN.cell/PE      1175

OVERLAPPED ELEMENTS      1373

PE/NEIB-PE#      NEIB-PEs
 0      7      7      6      4      5      2      1      3
 1      7      7      5      6      0      2      4      3
 2      7      7      6      0      5      1      4      3
 3      6      7      2      6      1      5      0
 4      6      6      7      5      0      2      1
 5      7      7      6      4      0      1      2      3
 6      7      7      5      4      0      2      1      3
 7      7      6      5      4      0      2      1      3

```

Procedures for Parallel FEM



INPUT.DAT (fixed name)

INPUT.DAT

../mesh/aaa	HEADER
2000	ITER
1.0 1.0	COND, QVOL
1.0e-08	RESID

- **HEADER:** Header of Distributed Local Files
- **ITER:** Max. Number of Iterations
- **COND:** Thermal Conductivity
- **QVOL:** Heat Generation Rate
- **RESID:** Convergence Criteria for CG Method

$$\frac{\partial}{\partial x} \left(\lambda \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(\lambda \frac{\partial T}{\partial y} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) + \dot{Q}(x, y, z) = 0$$

$$\dot{Q}(x, y, z) = QVOL |x_C + y_C|$$

pFEM/pfem3d/run/a08.sh

```
#!/bin/sh
#PJM -N "flat-08"
#PJM -L rscgrp=lecture6-o
#PJM -L node=8
#PJM -mpi proc=384
#PJM -L elapse=00:15:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o a08.lst

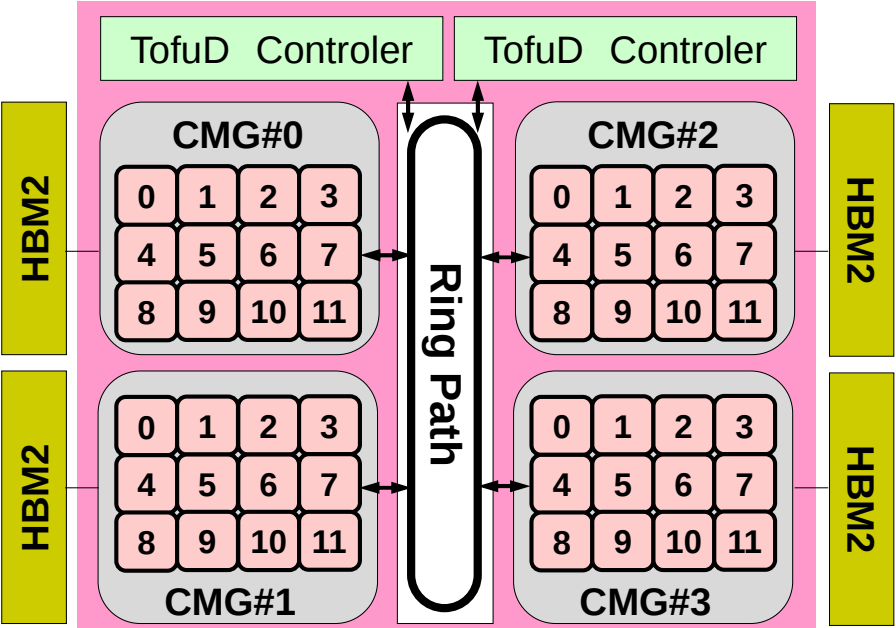
module load fj
module load fjmp

mpiexec ./sol
mpiexec numactl -l ./sol
```

Job Name
Name of "Queue/Resource Group"
Node #
Total MPI # (384/8= 48 per node)
Computation Time
Group Name (Wallet)
Standard Error
Standard Output

Number of Processes

#PJM	-L	node=1;	#PJM	--mpi	proc= 1	1-node, 1-proc, 1-proc/n
#PJM	-L	node=1;	#PJM	--mpi	proc= 4	1-node, 4-proc, 4-proc/n
#PJM	-L	node=1;	#PJM	--mpi	proc=12	1-node, 12-proc, 12-proc/n
#PJM	-L	node=1;	#PJM	--mpi	proc=24	1-node, 24-proc, 24-proc/n
#PJM	-L	node=1;	#PJM	--mpi	proc=48	1-node, 48-proc, 48-proc/n
#PJM	-L	node= 4;	#PJM	--mpi	proc=192	4-node, 192-proc, 48-proc/n
#PJM	-L	node= 8;	#PJM	--mpi	proc=384	8-node, 384-proc, 48-proc/n
#PJM	-L	node=12;	#PJM	--mpi	proc=576	12-node, 576-proc, 48-proc/n



Example: k-MeTis (1/2) (8-part's)

```
>$ cd /work/gt36/t36XYZ/pFEM/pfem3d/mesh
```

```
<modify inp_mg, mg.sh, inp_kmetis>  
<modify part_kmetis.sh>
```

```
>$ pjsub mg.sh  
>$ pjsub part_kmetis.sh
```

inp_mg

31 31 31

inp_kmetis

cube . 0

2

8

aaa

31^3 elements

32^3 nodes

$2^3 = 8$ partitions

Example: k-MeTis (2/2) (8-part's)

```
>$ cd ../run
<modify INPUT.DAT, test.sh>
```

```
>$ pjsub test.sh
```

INPUT.DAT

```
../mesh/aaa
2000
1.0 1.0
1.0e-08
```

test.sh

```
#!/bin/sh
#PJM -N "flat-02"
#PJM -L rscgrp=lecture6-o
#PJM -L node=1
#PJM -mpi proc=8
#PJM -L elapse=00:15:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o test.lst
```

```
module load fj
module load fjmpi
```

```
mpiexec ./sol
```

test.lst

```
*** matrix conn.      3.643613E-03 sec.
*** matrix ass.       4.854173E-03 sec.
```

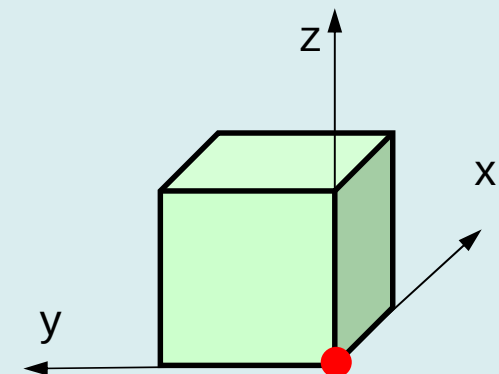
```
1 5.189549E+00
2 4.822416E+00
3 4.514909E+00
4 4.253956E+00
5 4.030138E+00
```

(...)

```
91 7.309762E-08
92 3.569387E-08
93 1.914244E-08
94 1.314290E-08
95 6.435628E-09
```

```
*** real COMP.      6.575127E-02 sec.
```

```
      3      1      1.262583E+04
* normal termination
```



Validation: Single CPU on PC

31³ elements
32³ nodes

INPUT.DAT

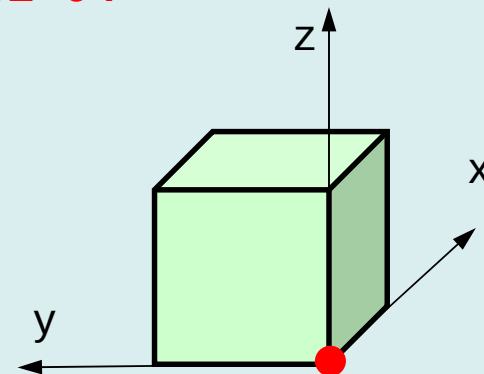
```
cube . 0  
2000  
1.0 1.0  
1.0e-08
```

```
1      5.189549E+00  
2      4.822416E+00  
3      4.514909E+00  
4      4.253956E+00  
5      4.030138E+00
```

(...)

```
91      7.309762E-08  
92      3.569387E-08  
93      1.914244E-08  
94      1.314290E-08  
95      6.435628E-09
```

```
1      1.262583E+04
```



- Installation
- Execution
 - Procedures of Parallel FEM
 - Domain Decomposition/Partitioning
 - Real Execution
- **Data Structure**

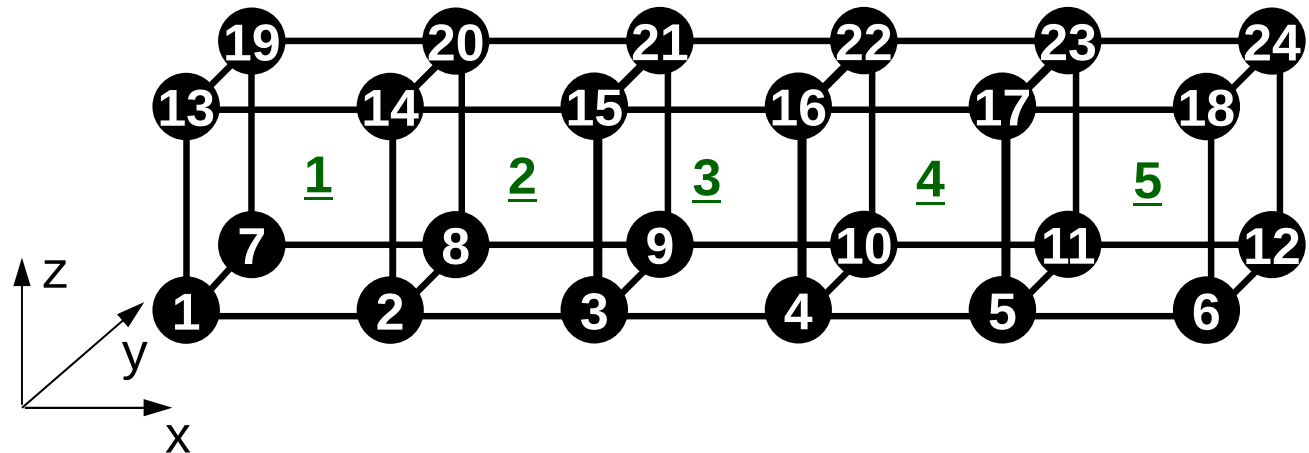
Attention !!

- If you cannot find “test.sh”, please copy the following:
 - `/work/gt36/share/test.sh`
- Processes of mesh generation & partitioning are not parallelized, therefore it is very expensive in the following cases:
 - larger meshes
 - more domains
- Parallel mesh generator is also available.
 - Generally, this procedure is used in this class
 - But partitioning using METIS is very flexible.

“mesh.inp”: parallel mesh generation

(values)	(variables)	(descriptions)
6 2 2	npx, npy, npz	Total number of nodes in X-, Y-, and Z-direction (Nx, Ny, Nz in the prev. page)
2 1 1	ndx, ndy, ndz	Partition # in each direction (X,Y,Z)
pcube	HEADER	Header of distributed local file

- Each of “npx, npy, npz” must be “divisible (割り切れる)” by each of “ndx, ndy, ndz”
- MPI process # = $ndx \times ndy \times ndz$
 - In this case,
6x2x2 nodes,
5x1x1elem’s,
2 partitions in X-direction



mg.sh: parallel mesh generation

"proc" must be equal to $(\text{ndx} \times \text{ndy} \times \text{ndz})$

Each MPI process generates each local mesh file

mg.sh

```
#!/bin/sh
#PJM -N "pmg"
#PJM -L rscgrp=lecture6-o
#PJM -L node=1
#PJM --mpi proc=2
#PJM -L elapse=00:10:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o pmg.lst

module load fj
module load fjmpi
mpiexec ./pmesh

rm wk.*
```

Example: pmesh (1/2) (8-part's)

```
>$ cd /work/gt36/t36XYZ/pFEM/pfem3d/pmesh  
<modify mesh.inp, mg.sh>
```

```
>$ pjsub mg.sh
```

mesh.inp

```
32 32 32  
2 2 2  
pcube
```

31^3 elements
 32^3 nodes
 $2^3 = 8$ partitions

mg.sh

```
#!/bin/sh  
#PJM -N "pmg"  
#PJM -L rscgrp=lecture6-o  
#PJM -L node=1  
#PJM --mpi proc=8  
#PJM -L elapse=00:10:00  
#PJM -g gt36  
#PJM -j  
#PJM -e err  
#PJM -o pmg.lst
```

```
module load fj  
module load fjmpi  
mpiexec ./pmesh
```

```
rm wk.*
```

Example: pmesh (2/2) (8-part's)

```
>$ cd ../run
<modify INPUT.DAT, test.sh>

>$ pjsub test.sh
```

INPUT.DAT

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

test.sh

```
#!/bin/sh
#PJM -N "flat-02"
#PJM -L rscgrp=lecture6-o
#PJM -L node=1
#PJM --mpi proc=8
#PJM -L elapse=00:15:00
#PJM -g gt36
#PJM -j
#PJM -e err
#PJM -o test.lst
```

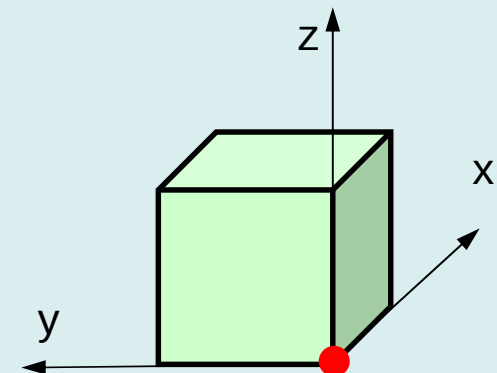
```
module load fj
module load fjmpi
mpiexec ./sol
```

test.lst

```
*** matrix conn.      3.643613E-03 sec.
*** matrix ass.      4.854173E-03 sec.

  1      5.189549E+00
  2      4.822416E+00
  3      4.514909E+00
  4      4.253956E+00
  5      4.030138E+00
(...)
 91      7.309762E-08
 92      3.569387E-08
 93      1.914244E-08
 94      1.314290E-08
 95      6.435628E-09
*** real    COMP.      6.575127E-02 sec.

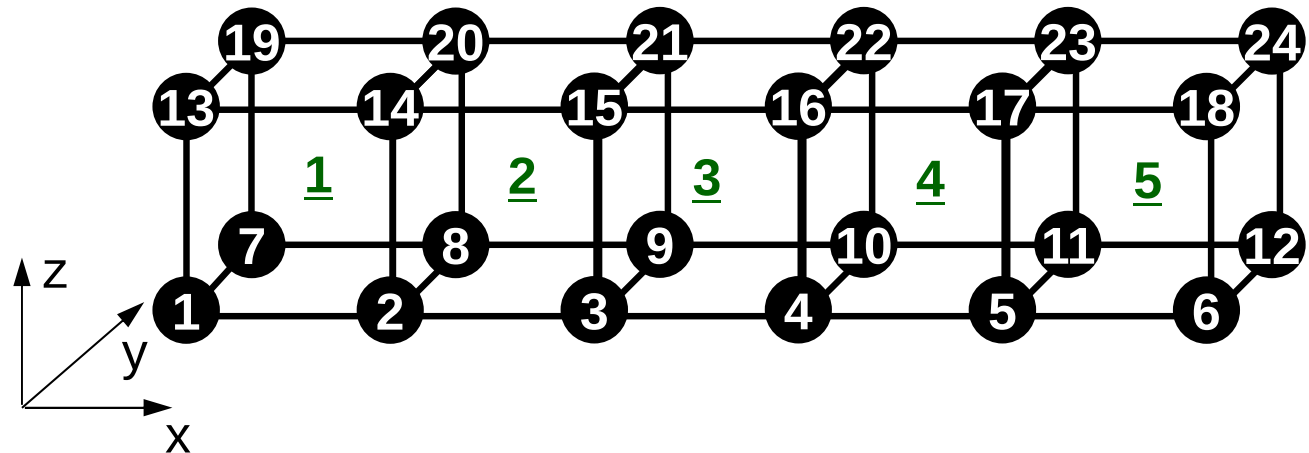
      0      1      1.262583E+04
* normal termination
```



“mesh.inp”: parallel mesh generation

(values)	(variables)	(descriptions)
6 2 2	npx, npy, npz	Total number of nodes in X-, Y-, and Z-direction (Nx, Ny, Nz in the prev. page)
2 1 1	ndx, ndy, ndz	Partition # in each direction (X,Y,Z)
pcube	HEADER	Header of distributed local file

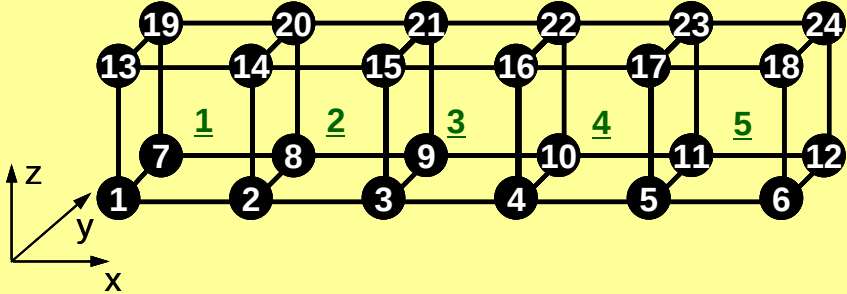
- Each of “npx, npy, npz” must be “divisible (割り切れる)” by each of “ndx, ndy, ndz”
- MPI process # = $ndx \times ndy \times ndz$
 - In this case,
6x2x2 nodes,
5x1x1 elem’s,
2 partitions in X-direction



Initial Global Mesh (1/2)

24

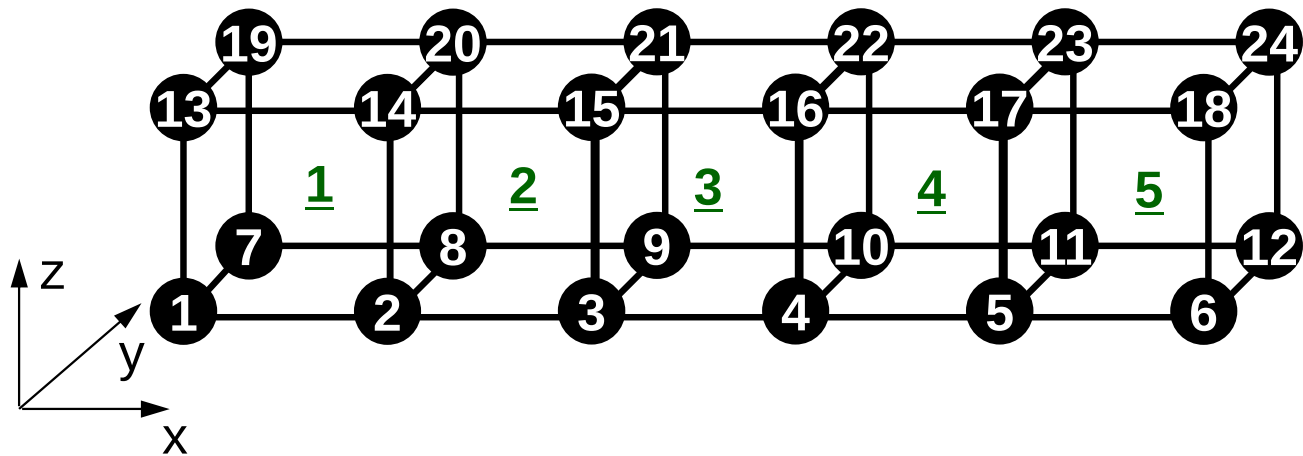
1	0.000000E+00	0.000000E+00	0.000000E+00
2	1.000000E+00	0.000000E+00	0.000000E+00
3	2.000000E+00	0.000000E+00	0.000000E+00
4	3.000000E+00	0.000000E+00	0.000000E+00
5	4.000000E+00	0.000000E+00	0.000000E+00
6	5.000000E+00	0.000000E+00	0.000000E+00
7	0.000000E+00	1.000000E+00	0.000000E+00
8	1.000000E+00	1.000000E+00	0.000000E+00
9	2.000000E+00	1.000000E+00	0.000000E+00
10	3.000000E+00	1.000000E+00	0.000000E+00
11	4.000000E+00	1.000000E+00	0.000000E+00
12	5.000000E+00	1.000000E+00	0.000000E+00
13	0.000000E+00	0.000000E+00	1.000000E+00
14	1.000000E+00	0.000000E+00	1.000000E+00
15	2.000000E+00	0.000000E+00	1.000000E+00
16	3.000000E+00	0.000000E+00	1.000000E+00
17	4.000000E+00	0.000000E+00	1.000000E+00
18	5.000000E+00	0.000000E+00	1.000000E+00
19	0.000000E+00	1.000000E+00	1.000000E+00
20	1.000000E+00	1.000000E+00	1.000000E+00
21	2.000000E+00	1.000000E+00	1.000000E+00
22	3.000000E+00	1.000000E+00	1.000000E+00
23	4.000000E+00	1.000000E+00	1.000000E+00
24	5.000000E+00	1.000000E+00	1.000000E+00



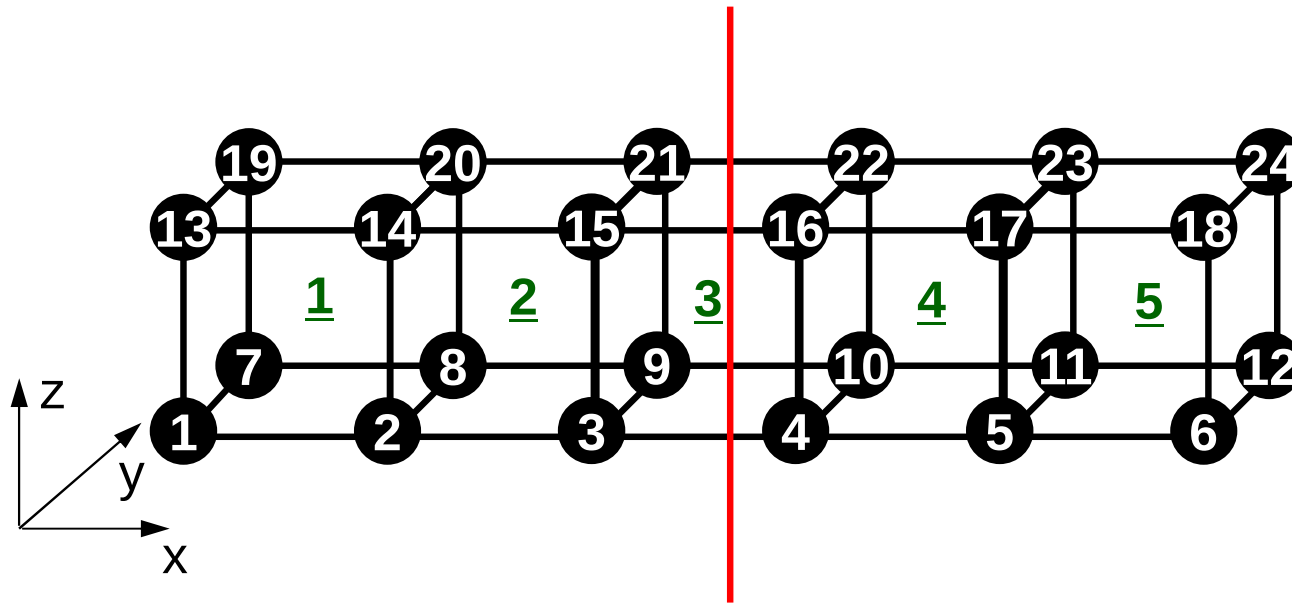
5									
361	361	361	361	361					
1	1	1	2	8	7	13	14	20	19
2	1	2	3	9	8	14	15	21	20
3	1	3	4	10	9	15	16	22	21
4	1	4	5	11	10	16	17	23	22
5	1	5	6	12	11	17	18	24	23

Initial Global Mesh (2/2)

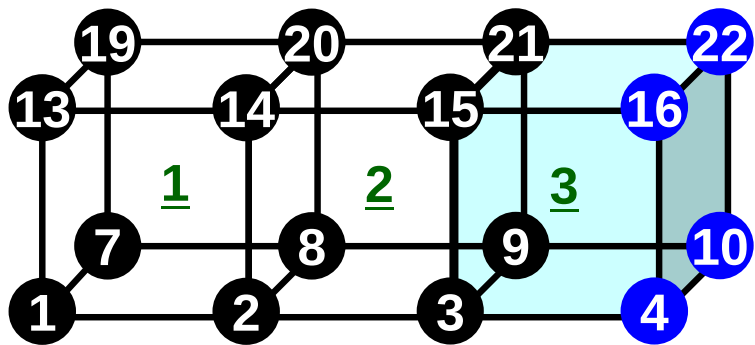
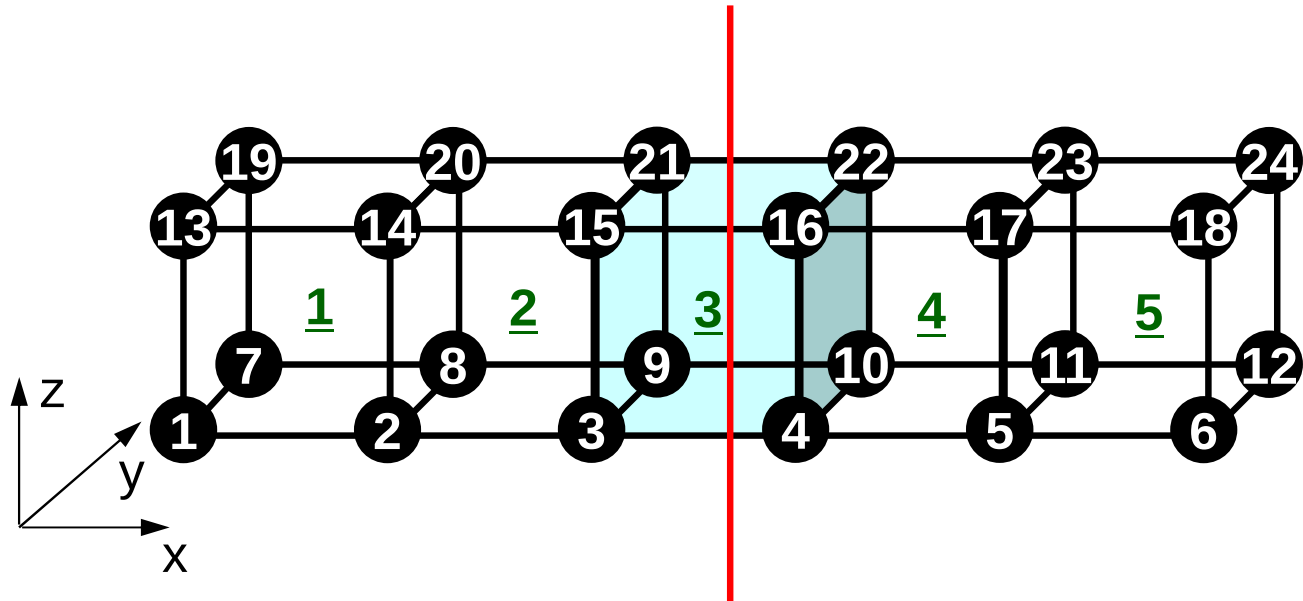
	4										
Xmin	4	16	28	40							
Ymin	1	7	13	19							
Zmin	1	2	3	4	5	6	13	14	15	16	
	17	18									
Zmax	1	2	3	4	5	6	7	8	9	10	
	11	12									
	13	14	15	16	17	18	19	20	21	22	
	23	24									



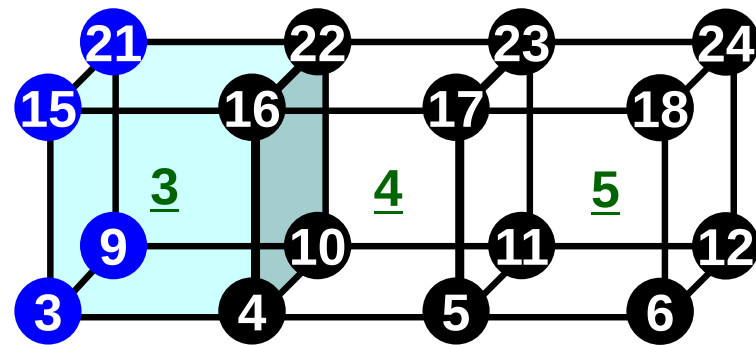
RCB: 2 PE's in X-direction



RCB: 2 PE's in X-direction



pcube.0



pcube.1

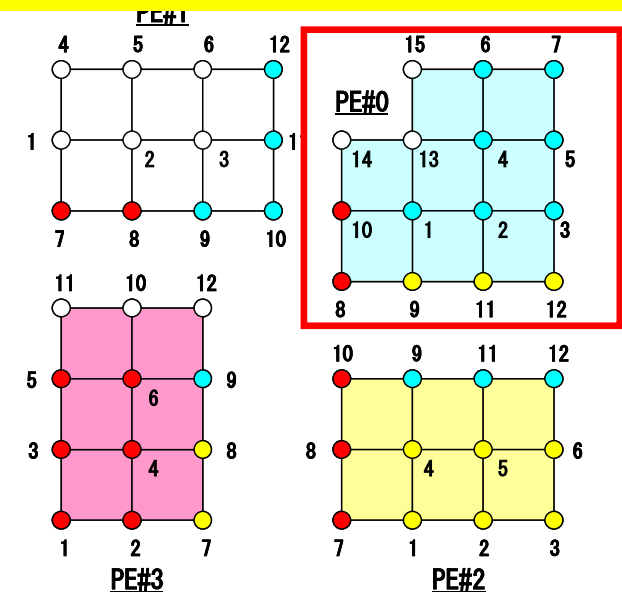
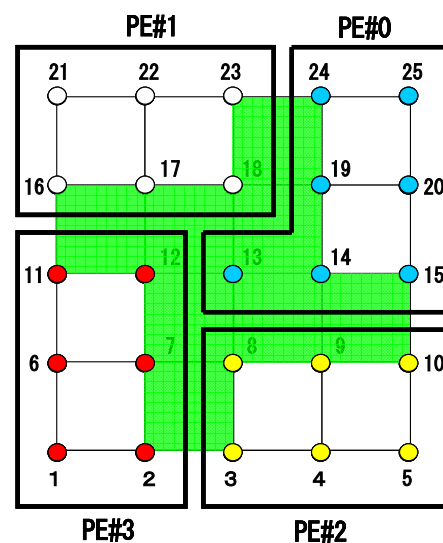
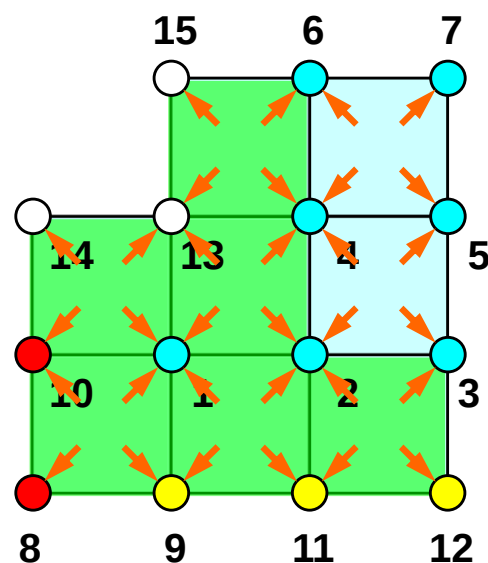
Distributed Local Mesh Files

- Neighbors
- Nodes
- Elements
- Communication Table (Import/Recv)
- Communication Table (Export/Send)
- Node Groups

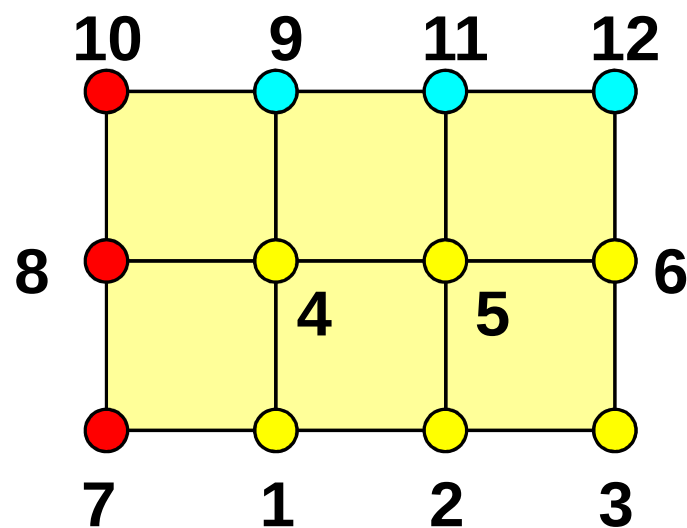
Node-based Partitioning

internal nodes - elements - external nodes

- Partitioned nodes themselves (Internal Nodes) 内点
- Elements which include Internal Nodes 内点を含む要素
- External Nodes included in the Elements 外点
in overlapped region among partitions.
- Info of External Nodes are required for completely local element-based operations on each processor.



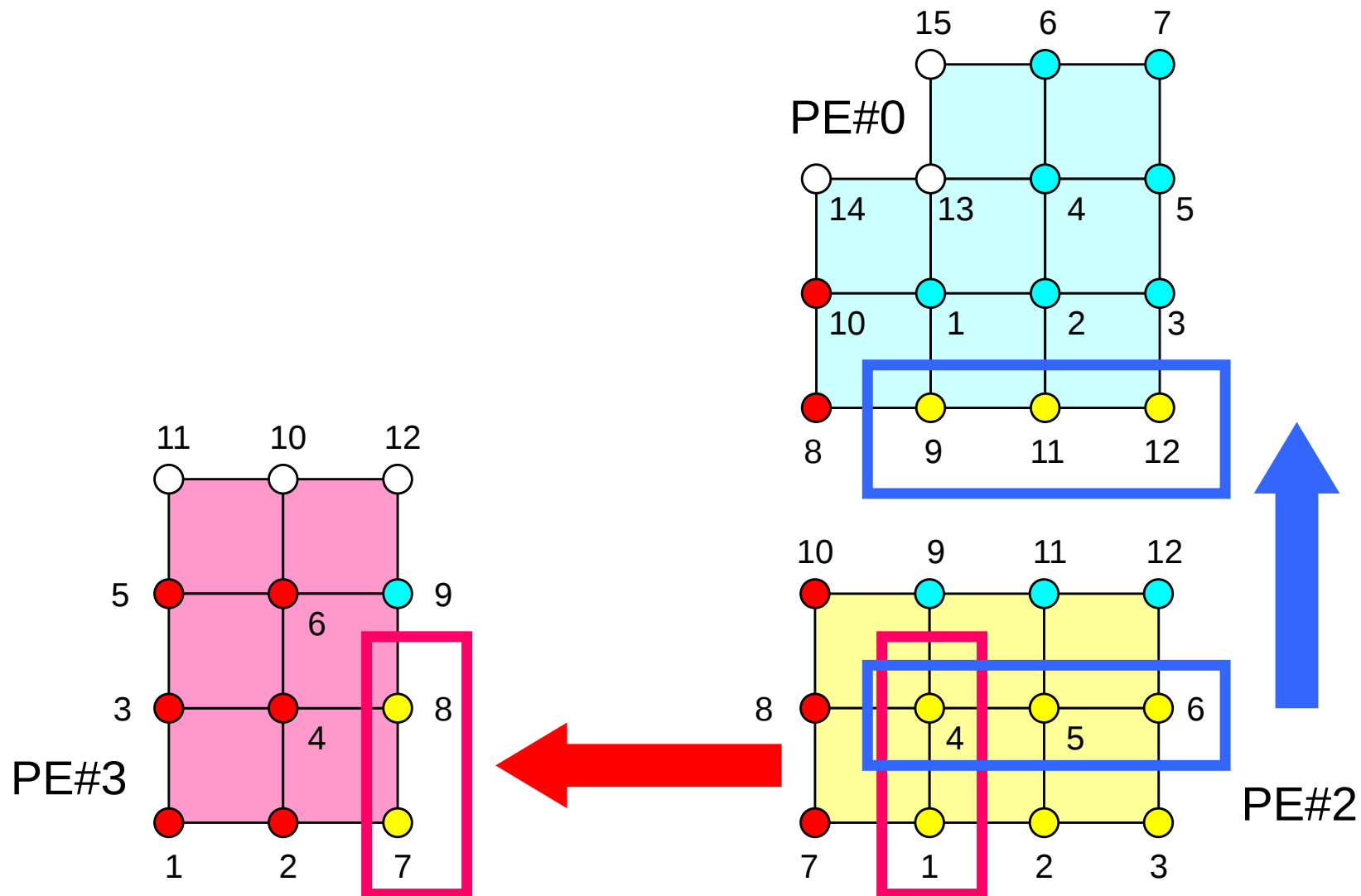
Description of Distributed Local Data



- Internal/External Points
 - Numbering: Starting from internal pts, then external pts after that
- Neighbors
 - Shares overlapped meshes
 - Number and ID of neighbors
- External Points
 - From where, how many, and which external points are received/imported ?
- Boundary Points
 - To where, how many and which boundary points are sent/exported ?

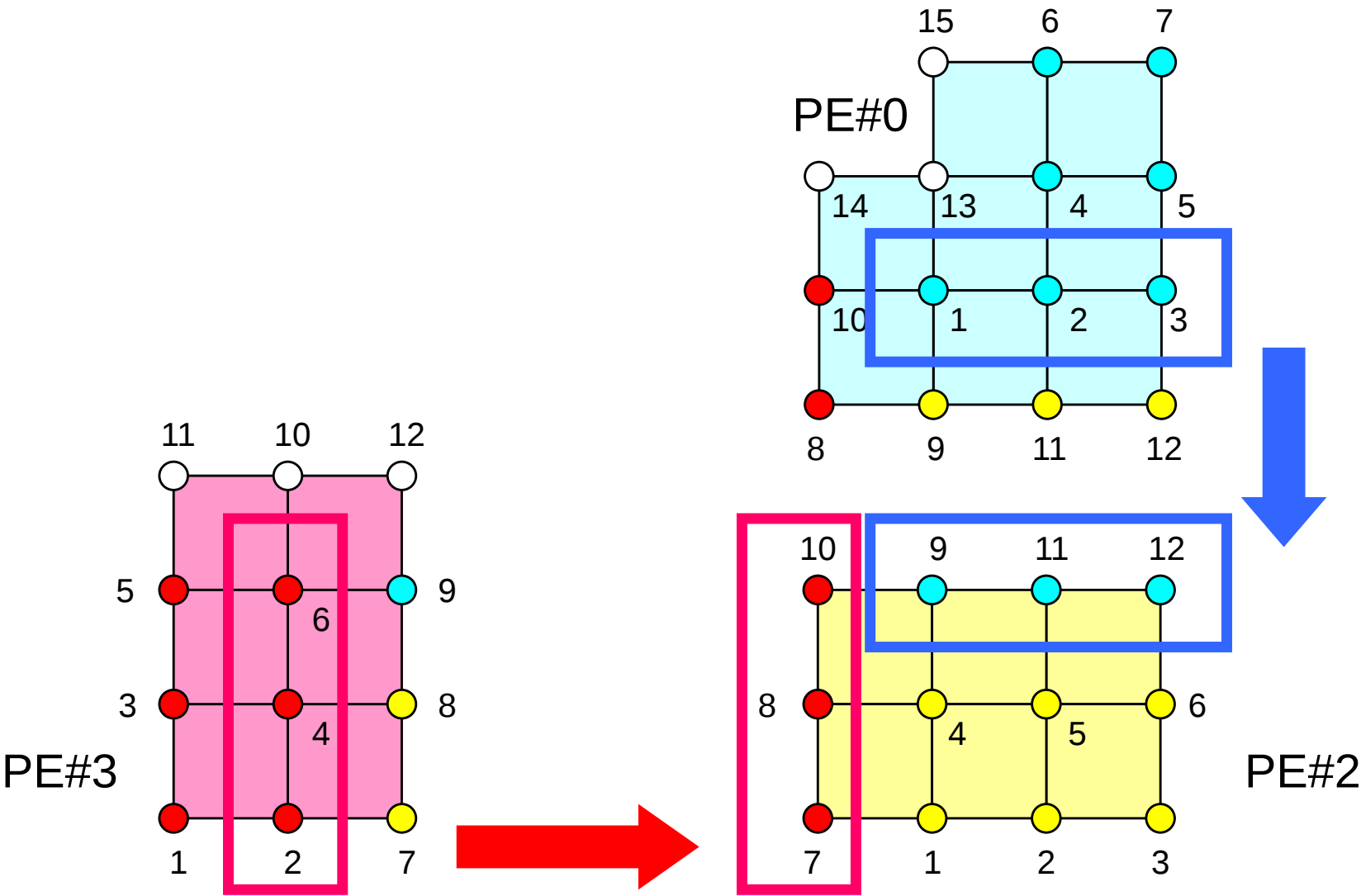
Boundary Nodes (境界点) : SEND

PE#2 : send information on “boundary nodes”



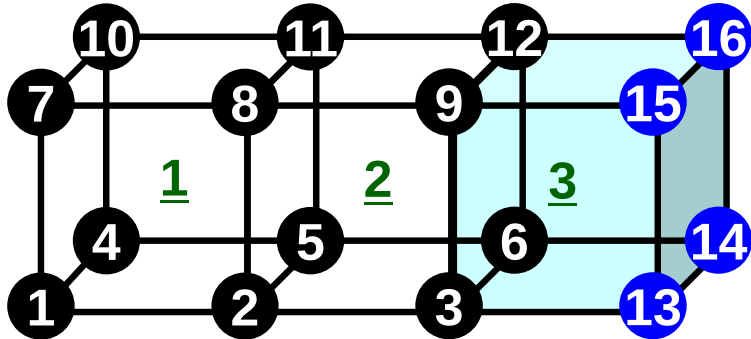
External Nodes (外点) : RECEIVE

PE#2 : receive information for “external nodes”

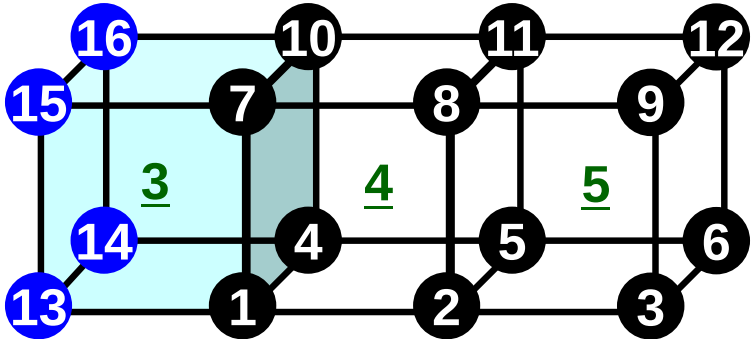


Neighbors

pc.0



pc.1



0				
1				
1				
16		12		
1	0	0.00	0.00	0.00
2	0	1.00	0.00	0.00
3	0	2.00	0.00	0.00
4	0	0.00	1.00	0.00
5	0	1.00	1.00	0.00
6	0	2.00	1.00	0.00
7	0	0.00	0.00	1.00
8	0	1.00	0.00	1.00
9	0	2.00	0.00	1.00
10	0	0.00	1.00	1.00
11	0	1.00	1.00	1.00
12	0	2.00	1.00	1.00
1	1	3.00	0.00	0.00
4	1	3.00	1.00	0.00
7	1	3.00	0.00	1.00
10	1	3.00	1.00	1.00

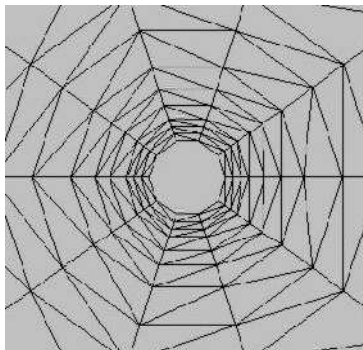
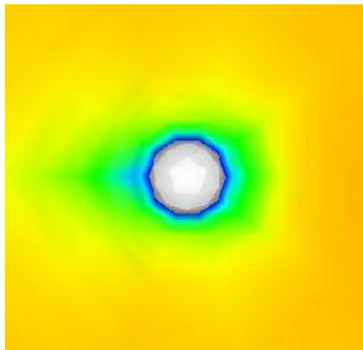
1				
1				
0				
16		12		
1	1	3.00	0.00	0.00
2	1	4.00	0.00	0.00
3	1	5.00	0.00	0.00
4	1	3.00	1.00	0.00
5	1	4.00	1.00	0.00
6	1	5.00	1.00	0.00
7	1	3.00	0.00	1.00
8	1	4.00	0.00	1.00
9	1	5.00	0.00	1.00
10	1	3.00	1.00	1.00
11	1	4.00	1.00	1.00
12	1	5.00	1.00	1.00
3	0	2.00	0.00	0.00
6	0	2.00	1.00	0.00
9	0	2.00	0.00	1.00
12	0	2.00	1.00	1.00

Local Numbering: Nodes

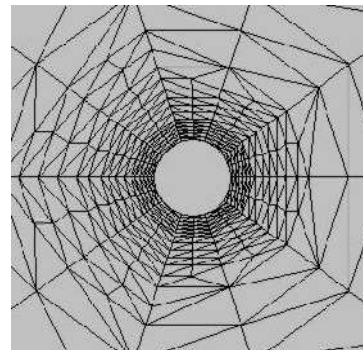
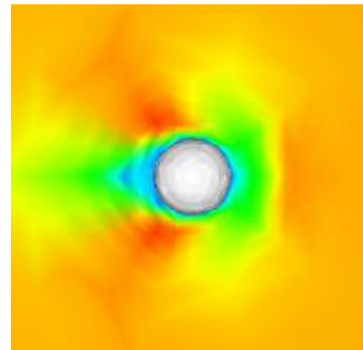
- Local node ID starts from “1” in each PE
 - Same program for 1-CPU can be used: SPMD
 - Local element ID also starts from “1”
- Numbering: Internal -> External Points
- Double Numbering
 - Local node ID at its “home” PE: **NODE_ID(*i*, 1)**
 - ID of “home” PE: **NODE_ID(*i*, 2)**
- **Suitable for Adaptive Mesh Refinement and Dynamic Load Balancing (next page)**

Supersonic Flow around a Sphere

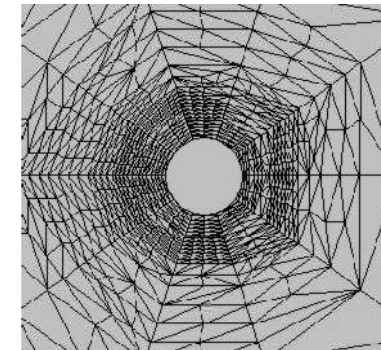
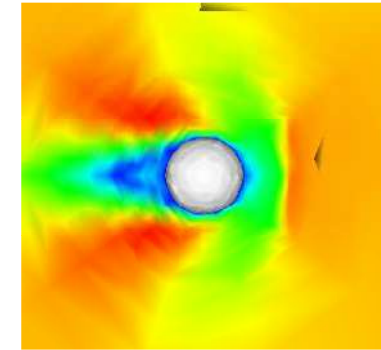
Ideal Gas, $M = 1.40$, Uniform Flow, $Re = 10^6$
before/after Dynamic Load Balancing



Initial Grid



1-Lev. Adapted

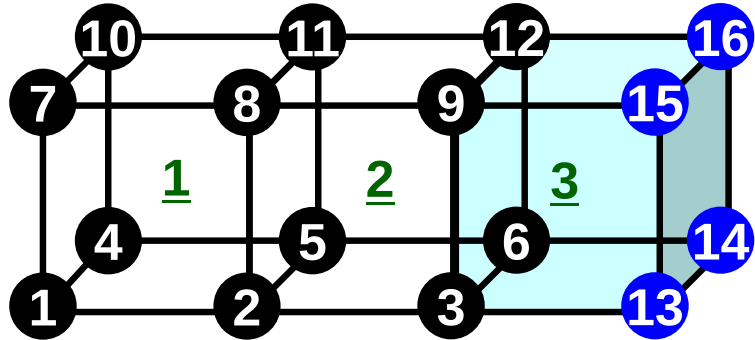


2-Lev. Adapted

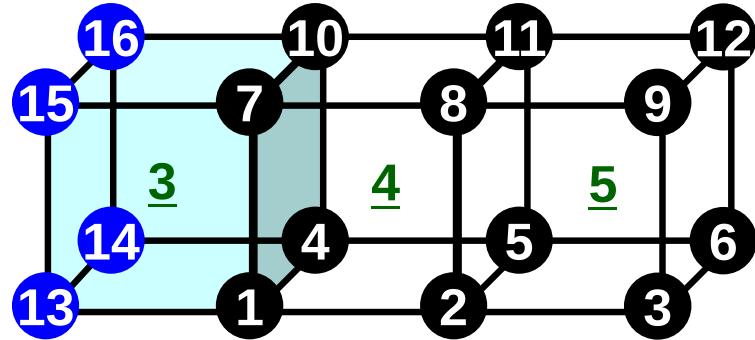
			<u>before</u>	<u>after</u>	<u>before</u>	<u>after</u>
PE0	137	-	793	652	3834	2527
PE1	137	-	696	650	2769	2526
PE2	136	-	668	652	2703	2522
PE3	136	-	448	651	1390	2524

Internal, External Nodes

pc.0



pc.1

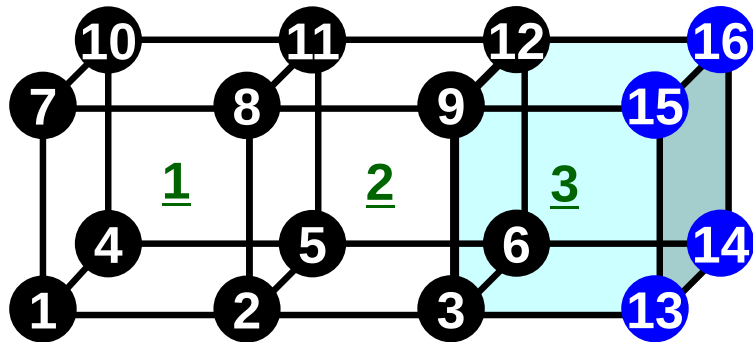


0				
1				
1				
16		12		
1	0	0.00	0.00	0.00
2	0	1.00	0.00	0.00
3	0	2.00	0.00	0.00
4	0	0.00	1.00	0.00
5	0	1.00	1.00	0.00
6	0	2.00	1.00	0.00
7	0	0.00	0.00	1.00
8	0	1.00	0.00	1.00
9	0	2.00	0.00	1.00
10	0	0.00	1.00	1.00
11	0	1.00	1.00	1.00
12	0	2.00	1.00	1.00
1	1	3.00	0.00	0.00
4	1	3.00	1.00	0.00
7	1	3.00	0.00	1.00
10	1	3.00	1.00	1.00

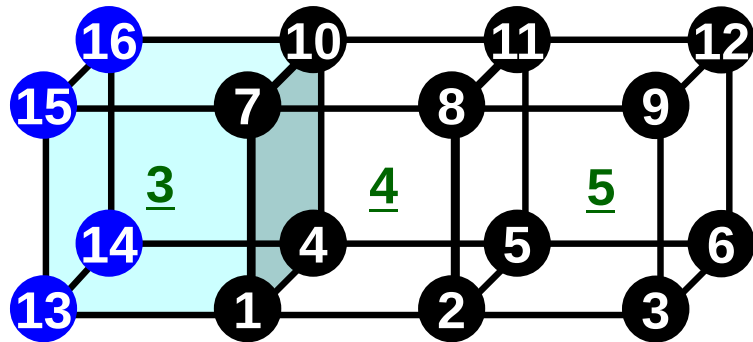
1				
1				
0				
16		12 (Node #: Total, Internal)		
1	1	3.00	0.00	0.00
2	1	4.00	0.00	0.00
3	1	5.00	0.00	0.00
4	1	3.00	1.00	0.00
5	1	4.00	1.00	0.00
6	1	5.00	1.00	0.00
7	1	3.00	0.00	1.00
8	1	4.00	0.00	1.00
9	1	5.00	0.00	1.00
10	1	3.00	1.00	1.00
11	1	4.00	1.00	1.00
12	1	5.00	1.00	1.00
3	0	2.00	0.00	0.00
6	0	2.00	1.00	0.00
9	0	2.00	0.00	1.00
12	0	2.00	1.00	1.00

Local Numbering: Nodes

pc.0



pc.1



0					
1					
1					
16		12			
1	0	0.00	0.00	0.00	①
2	0	1.00	0.00	0.00	②
3	0	2.00	0.00	0.00	③
4	0	0.00	1.00	0.00	④
5	0	1.00	1.00	0.00	⑤
6	0	2.00	1.00	0.00	⑥
7	0	0.00	0.00	1.00	⑦
8	0	1.00	0.00	1.00	⑧
9	0	2.00	0.00	1.00	⑨
10	0	0.00	1.00	1.00	⑩
11	0	1.00	1.00	1.00	⑪
12	0	2.00	1.00	1.00	⑫
1	1	3.00	0.00	0.00	⑬
4	1	3.00	1.00	0.00	⑭
7	1	3.00	0.00	1.00	⑮
10	1	3.00	1.00	1.00	⑯

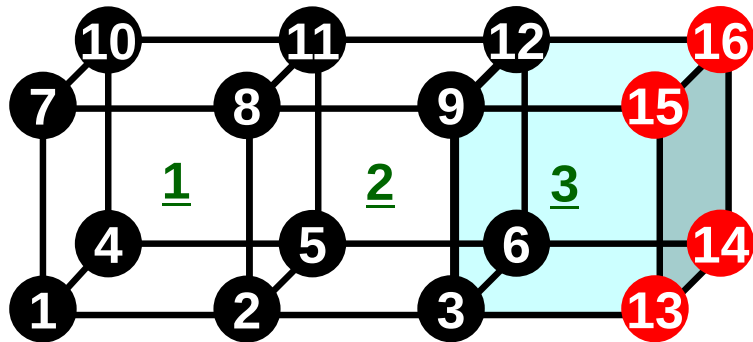
Local ID, "Home" PE, Coordinates

1					
1					
0					
16		12			
1	1	3.00	0.00	0.00	①
2	1	4.00	0.00	0.00	②
3	1	5.00	0.00	0.00	③
4	1	3.00	1.00	0.00	④
5	1	4.00	1.00	0.00	⑤
6	1	5.00	1.00	0.00	⑥
7	1	3.00	0.00	1.00	⑦
8	1	4.00	0.00	1.00	⑧
9	1	5.00	0.00	1.00	⑨
10	1	3.00	1.00	1.00	⑩
11	1	4.00	1.00	1.00	⑪
12	1	5.00	1.00	1.00	⑫
3	0	2.00	0.00	0.00	⑬
6	0	2.00	1.00	0.00	⑭
9	0	2.00	0.00	1.00	⑮
12	0	2.00	1.00	1.00	⑯

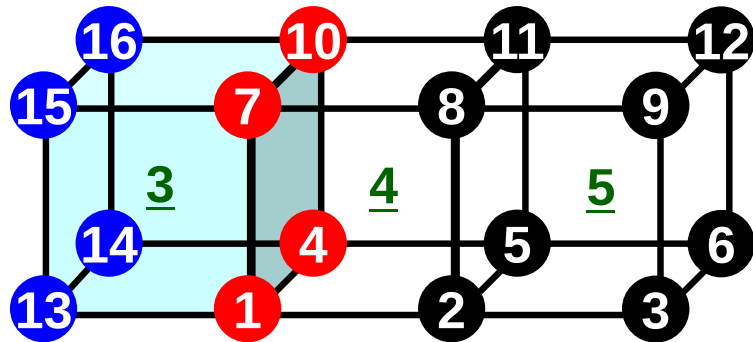
Local ID, "Home" PE, Coordinates

Local Numbering: Nodes

pc.0



pc.1



0					
1					
1					
16		12			
1	0	0.00	0.00	0.00	①
2	0	1.00	0.00	0.00	②
3	0	2.00	0.00	0.00	③
4	0	0.00	1.00	0.00	④
5	0	1.00	1.00	0.00	⑤
6	0	2.00	1.00	0.00	⑥
7	0	0.00	0.00	1.00	⑦
8	0	1.00	0.00	1.00	⑧
9	0	2.00	0.00	1.00	⑨
10	0	0.00	1.00	1.00	⑩
11	0	1.00	1.00	1.00	⑪
12	0	2.00	1.00	1.00	⑫
1	1	3.00	0.00	0.00	⑬
4	1	3.00	1.00	0.00	⑭
7	1	3.00	0.00	1.00	⑮
10	1	3.00	1.00	1.00	⑯

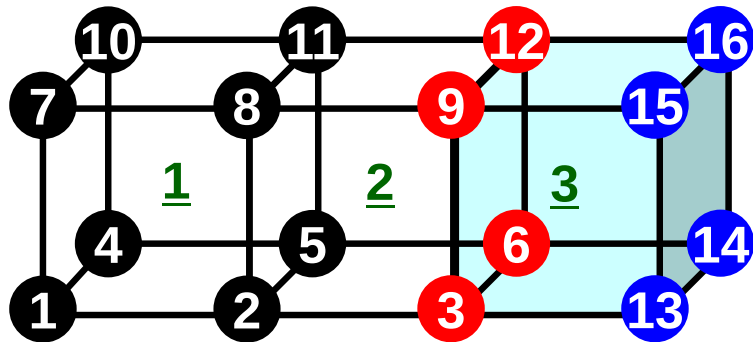
Local ID, "Home" PE, Coordinates

1					
1					
0					
16		12			
1	1	3.00	0.00	0.00	①
2	1	4.00	0.00	0.00	②
3	1	5.00	0.00	0.00	③
4	1	3.00	1.00	0.00	④
5	1	4.00	1.00	0.00	⑤
6	1	5.00	1.00	0.00	⑥
7	1	3.00	0.00	1.00	⑦
8	1	4.00	0.00	1.00	⑧
9	1	5.00	0.00	1.00	⑨
10	1	3.00	1.00	1.00	⑩
11	1	4.00	1.00	1.00	⑪
12	1	5.00	1.00	1.00	⑫
3	0	2.00	0.00	0.00	⑬
6	0	2.00	1.00	0.00	⑭
9	0	2.00	0.00	1.00	⑮
12	0	2.00	1.00	1.00	⑯

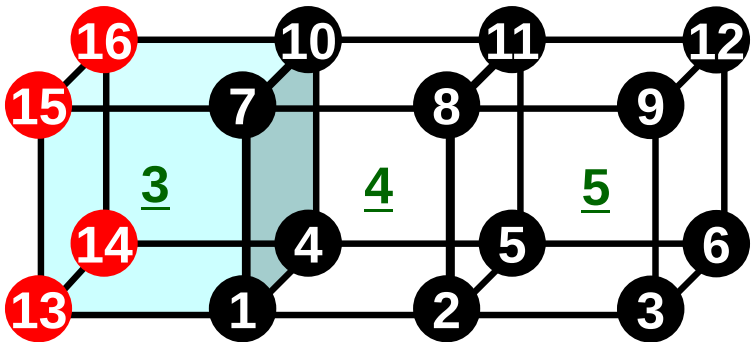
Local ID, "Home" PE, Coordinates

Local Numbering: Nodes

pc.0



pc.1



0					
1					
1					
16		12			
1	0	0.00	0.00	0.00	①
2	0	1.00	0.00	0.00	②
3	0	2.00	0.00	0.00	③
4	0	0.00	1.00	0.00	④
5	0	1.00	1.00	0.00	⑤
6	0	2.00	1.00	0.00	⑥
7	0	0.00	0.00	1.00	⑦
8	0	1.00	0.00	1.00	⑧
9	0	2.00	0.00	1.00	⑨
10	0	0.00	1.00	1.00	⑩
11	0	1.00	1.00	1.00	⑪
12	0	2.00	1.00	1.00	⑫
1	1	3.00	0.00	0.00	⑬
4	1	3.00	1.00	0.00	⑭
7	1	3.00	0.00	1.00	⑮
10	1	3.00	1.00	1.00	⑯

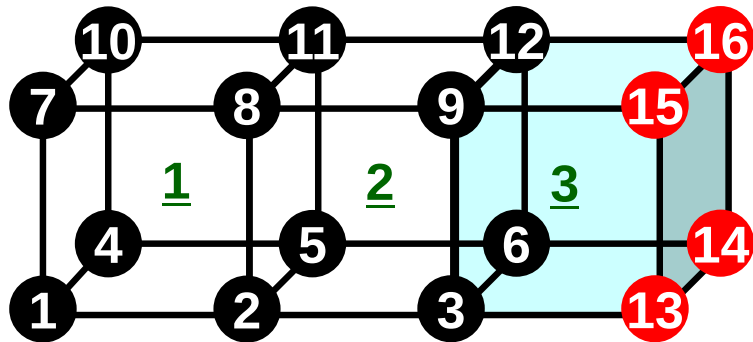
Local ID, "Home" PE, Coordinates

1					
1					
0					
16		12			
1	1	3.00	0.00	0.00	①
2	1	4.00	0.00	0.00	②
3	1	5.00	0.00	0.00	③
4	1	3.00	1.00	0.00	④
5	1	4.00	1.00	0.00	⑤
6	1	5.00	1.00	0.00	⑥
7	1	3.00	0.00	1.00	⑦
8	1	4.00	0.00	1.00	⑧
9	1	5.00	0.00	1.00	⑨
10	1	3.00	1.00	1.00	⑩
11	1	4.00	1.00	1.00	⑪
12	1	5.00	1.00	1.00	⑫
3	0	2.00	0.00	0.00	⑬
6	0	2.00	1.00	0.00	⑭
9	0	2.00	0.00	1.00	⑮
12	0	2.00	1.00	1.00	⑯

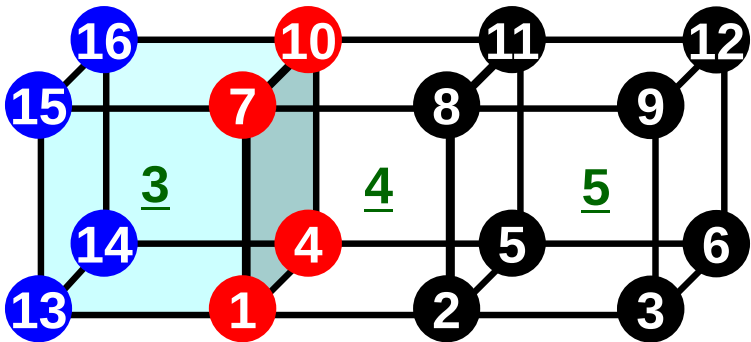
Local ID, "Home" PE, Coordinates

Local Numbering: Nodes

pc.0



pc.1



0					
1					
1					
16		12			
1	0	0.00	0.00	0.00	①
2	0	1.00	0.00	0.00	②
3	0	2.00	0.00	0.00	③
4	0	0.00	1.00	0.00	④
5	0	1.00	1.00	0.00	⑤
6	0	2.00	1.00	0.00	⑥
7	0	0.00	0.00	1.00	⑦
8	0	1.00	0.00	1.00	⑧
9	0	2.00	0.00	1.00	⑨
10	0	0.00	1.00	1.00	⑩
11	0	1.00	1.00	1.00	⑪
12	0	2.00	1.00	1.00	⑫
1	1	3.00	0.00	0.00	⑬
4	1	3.00	1.00	0.00	⑭
7	1	3.00	0.00	1.00	⑮
10	1	3.00	1.00	1.00	⑯

Local ID, "Home" PE, Coordinates

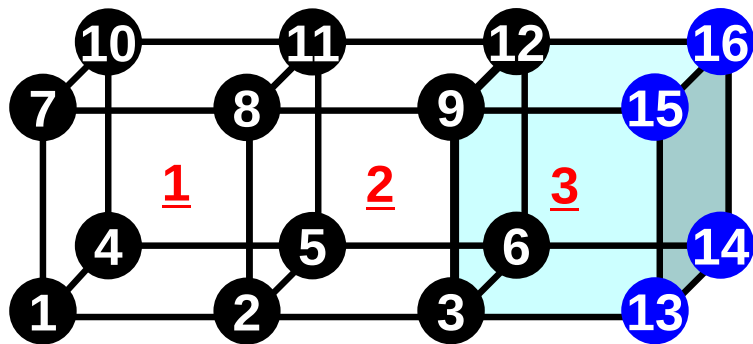
1					
1					
0					
16		12			
1	1	3.00	0.00	0.00	①
2	1	4.00	0.00	0.00	②
3	1	5.00	0.00	0.00	③
4	1	3.00	1.00	0.00	④
5	1	4.00	1.00	0.00	⑤
6	1	5.00	1.00	0.00	⑥
7	1	3.00	0.00	1.00	⑦
8	1	4.00	0.00	1.00	⑧
9	1	5.00	0.00	1.00	⑨
10	1	3.00	1.00	1.00	⑩
11	1	4.00	1.00	1.00	⑪
12	1	5.00	1.00	1.00	⑫
3	0	2.00	0.00	0.00	⑬
6	0	2.00	1.00	0.00	⑭
9	0	2.00	0.00	1.00	⑮
12	0	2.00	1.00	1.00	⑯

Local ID, "Home" PE, Coordinates

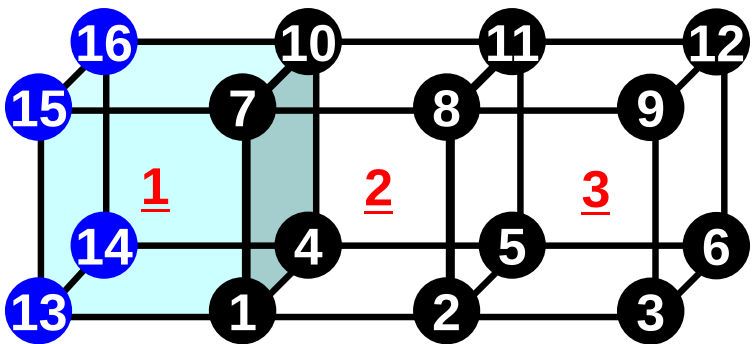
Only "local" ID's (numbers enclosed in circles) are used in the program

Local Numbering: Elements

pc.0

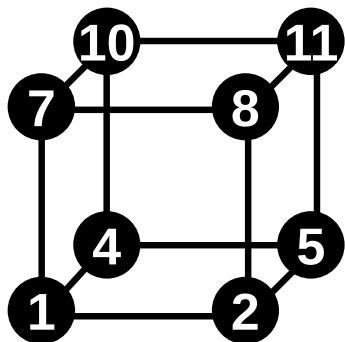


pc.1



3	3										
361	361	361									
1	0		1	1	2	5	4	7	8	11	10
2	0		1	2	3	6	5	8	9	12	11
3	0		1	3	13	14	6	9	15	16	12
1	2	3									

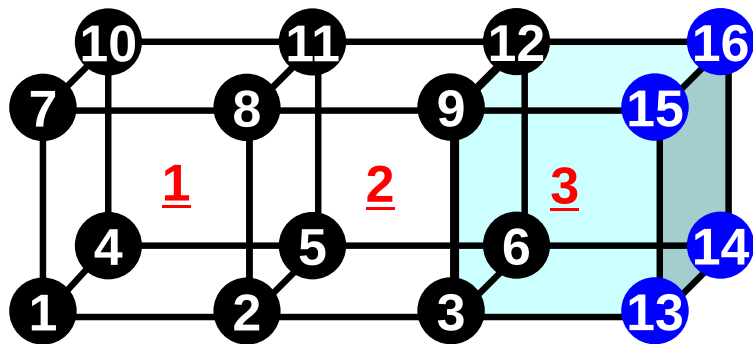
3	2	(Element #: All, Local)									
361	361	361									
3	0		1	13	1	4	14	15	7	10	16
1	1		1	1	2	5	4	7	8	11	10
2	1		1	2	3	6	5	8	9	12	11
2	3										



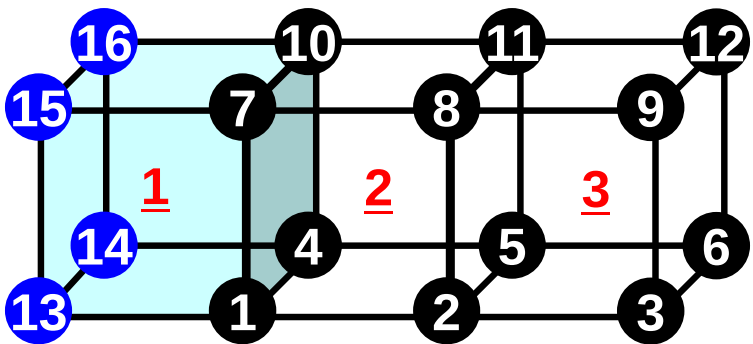
- “Home” PE of Element
 - Defined by “home” of 8 nodes
 - If all of 8 nodes are internal pts., “home” of the element is that of 8 nodes.
 - If external nodes are included, the smallest number of ID of “home” of the nodes is selected.
 - In this case, “home” PE’s of elements in overlapped region are all “0”.

Local Numbering: Elements

pc.0



pc.1



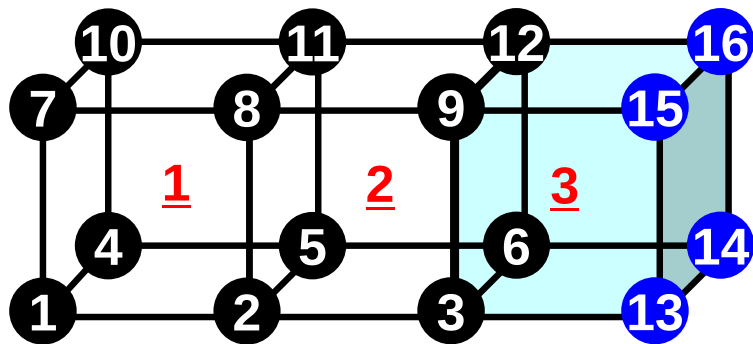
<u>3</u>	3											
361	361	361										
1	0	1	1	2	5	4	7	8	11	10	<u>1</u>	
2	0	1	2	3	6	5	8	9	12	11	<u>2</u>	
3	0	1	3	13	14	6	9	15	16	12	<u>3</u>	
1	2	3										

<u>3</u>	2											
361	361	361										
3	0	1	13	1	4	14	15	7	10	16	<u>1</u>	
1	1	1	1	2	5	4	7	8	11	10	<u>2</u>	
2	1	1	2	3	6	5	8	9	12	11	<u>3</u>	
2	3											

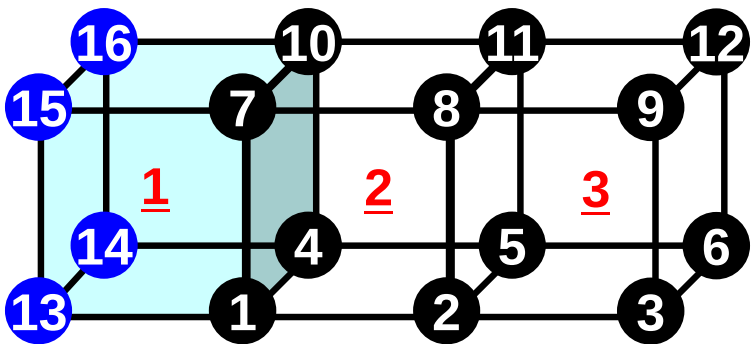
- Double Numbering for Element
 - Local ID at “home” PE: **ELEM_ID(i, 1)**
 - ID of “home” PE: **ELEM_ID(i, 2)**
- Material ID
- 8 Nodes
- Underlined local ID is used in the program

Local Numbering: Elements

pc.0



pc.1



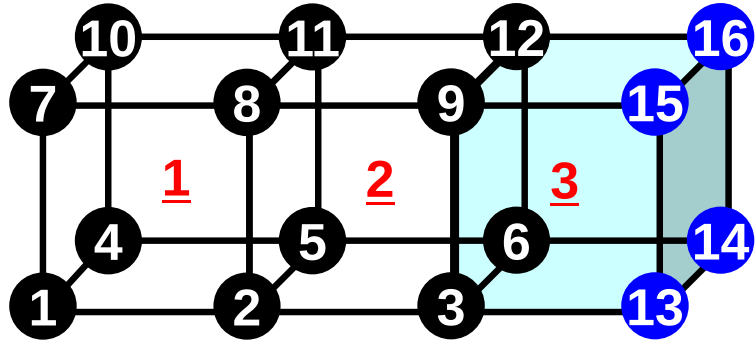
3	<u>3</u>														
361	361	361													
1	0	1	1	2	5	4	7	8	11	10	1				
2	0	1	2	3	6	5	8	9	12	11	2				
3	0	1	3	13	14	6	9	15	16	12	3				
1	2	3													

3	<u>2</u>														
361	361	361													
3	0	1	13	1	4	14	15	7	10	16	1				
1	1	1	1	2	5	4	7	8	11	10	2				
2	1	1	2	3	6	5	8	9	12	11	3				
2	3														

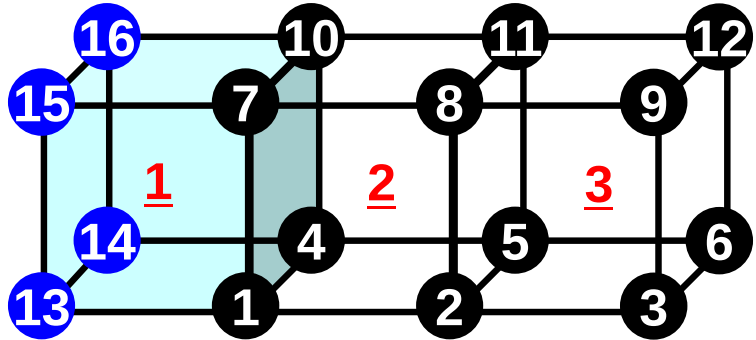
- pc.0
 - 1, 2, 3 are “Local Elements” (“Home Elements”)
- pc.1
 - 2, 3 are “Local Elements” (“Home Elements”)

Communication Tables

pc.0



pc.1



4
13
14
15
16
4
3
6
9
12

4
13
14
15
16
4
1
4
7
10

PE-to-PE Communication

Generalized Communication Tables

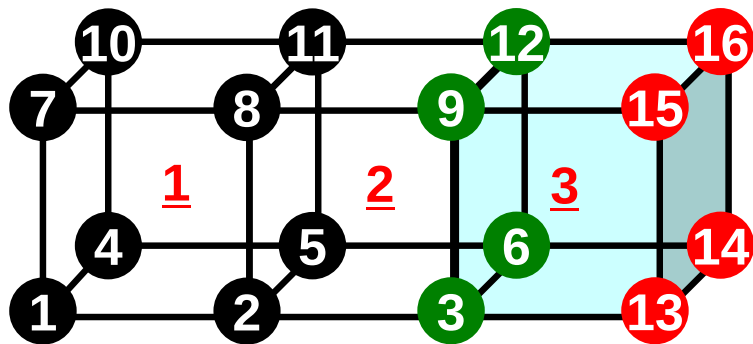
- “Communication” in parallel FEM means obtaining information of “external points” from their “home” PE’s
- “Communication Tables” describe relationship of “external points” among PE’s
 - Send/Export, Recv/Import
- Sending information of “boundary points”
- Receiving information of “external points”

Generalized Comm. Table: Send

- Neighbors
 - NeibPETot, NeibPE[NeibPETot]
- Message size for each neighbor
 - export_index[NeibPETot+1]
- ID of **boundary** points
 - export_item[export_index[NeibPETot+1]]
- Messages to each neighbor
 - SendBuf[export_index[NeibPETot+1]]

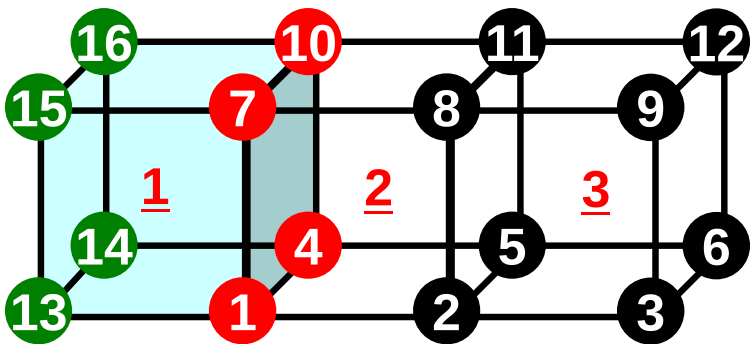
Communication Table (Send/Export)

pc.0



4
13
14
15
16
4
3
6
9
12

pc.1

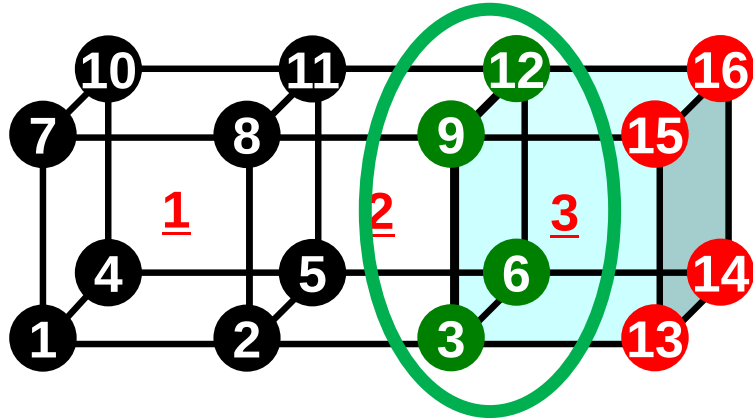


4
13
14
15
16
4
1
4
7
10
export_index(neib)
export_item

- export_index Size of Messages sent to Each Neighbor
 - # Neighbors= 1 in this case
- export_item Local ID of boundary points

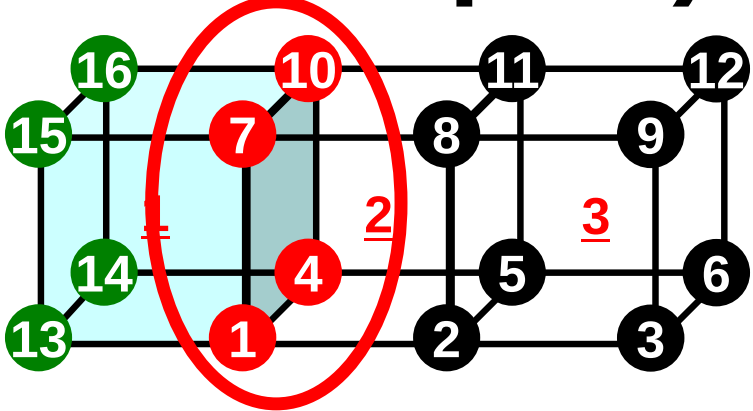
Communication Table (Send/Export)

pc.0



4
13
14
15
16
4
3
6
9
12

pc.1



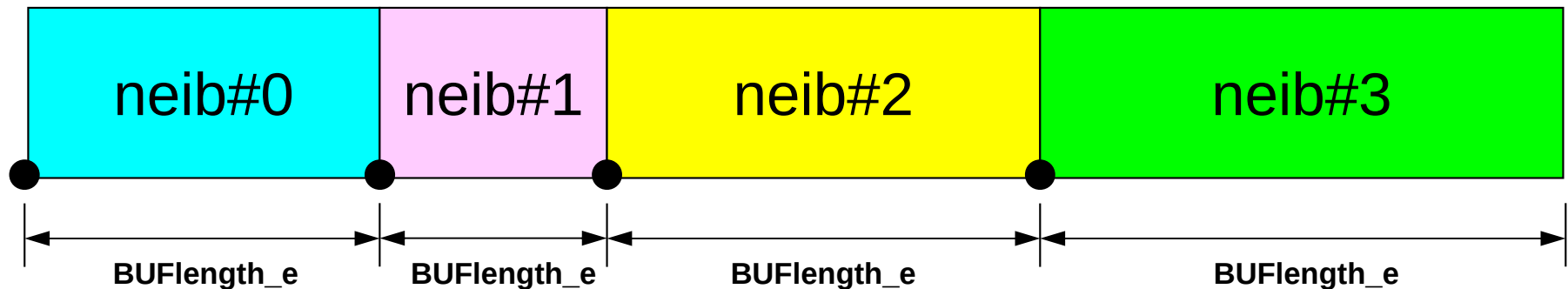
4
13
14
15
16
4
1
4
7
10
export_index(neib)
export_item

- export_index Size of Messages sent to Each Neighbor
 - # Neighbors= 1 in this case
- export_item Local ID of boundary points

SEND: MPI_Isend/Irecv/Waitall

C

SendBuf



export_index[0] export_index[1] export_index[2] export_index[3] export_index[4]

export_item (export_index[neib]:export_index[neib+1]-1) are sent to neib-th neighbor

```
for (neib=0; neib<NeibPETot;neib++){
    for (k=export_index[neib];k<export_index[neib+1];k++){
        kk= export_item[k];
        SendBuf[k]= VAL[kk];
    }
}
```

Copied to sending buffers

```
for (neib=0; neib<NeibPETot; neib++){
    tag= 0;
    iS_e= export_index[neib];
    iE_e= export_index[neib+1];
    BUFlength_e= iE_e - iS_e

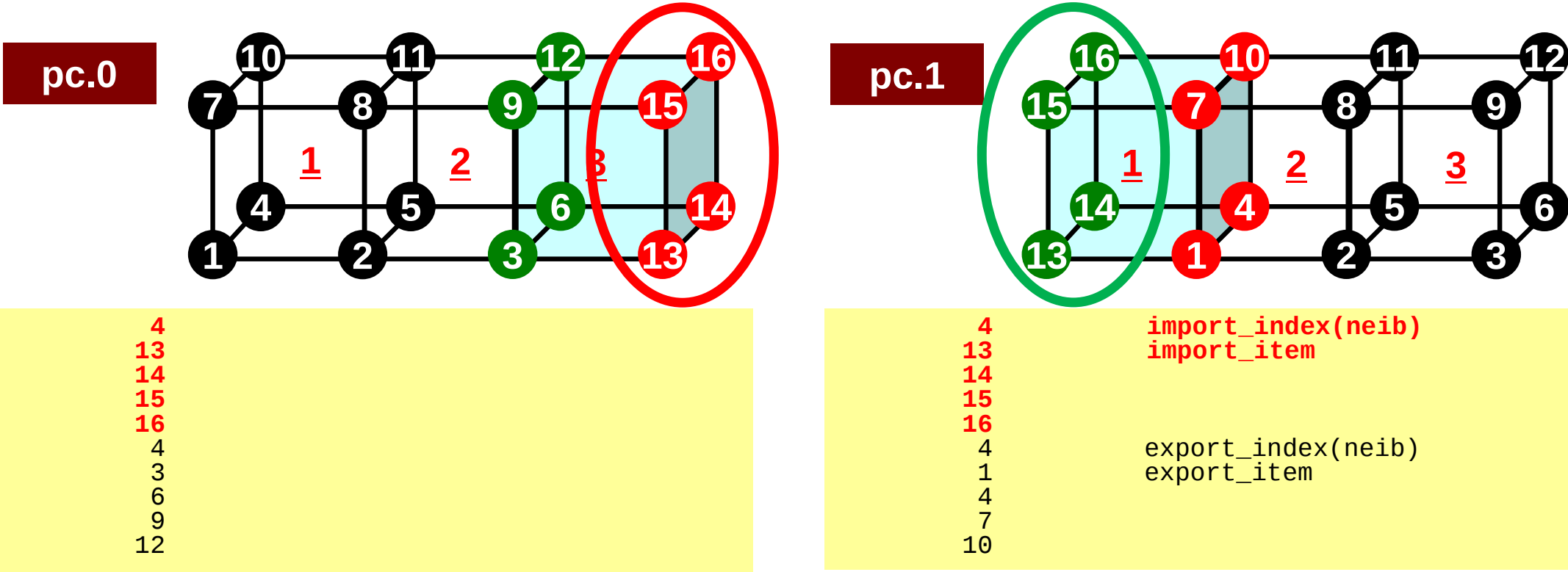
    ierr= MPI_Isend
        (&SendBuf[iS_e], BUFlength_e, MPI_DOUBLE, NeibPE[neib], 0,
         MPI_COMM_WORLD, &ReqSend[neib])
}
```

```
MPI_Waitall(NeibPETot, ReqSend, StatSend);
```

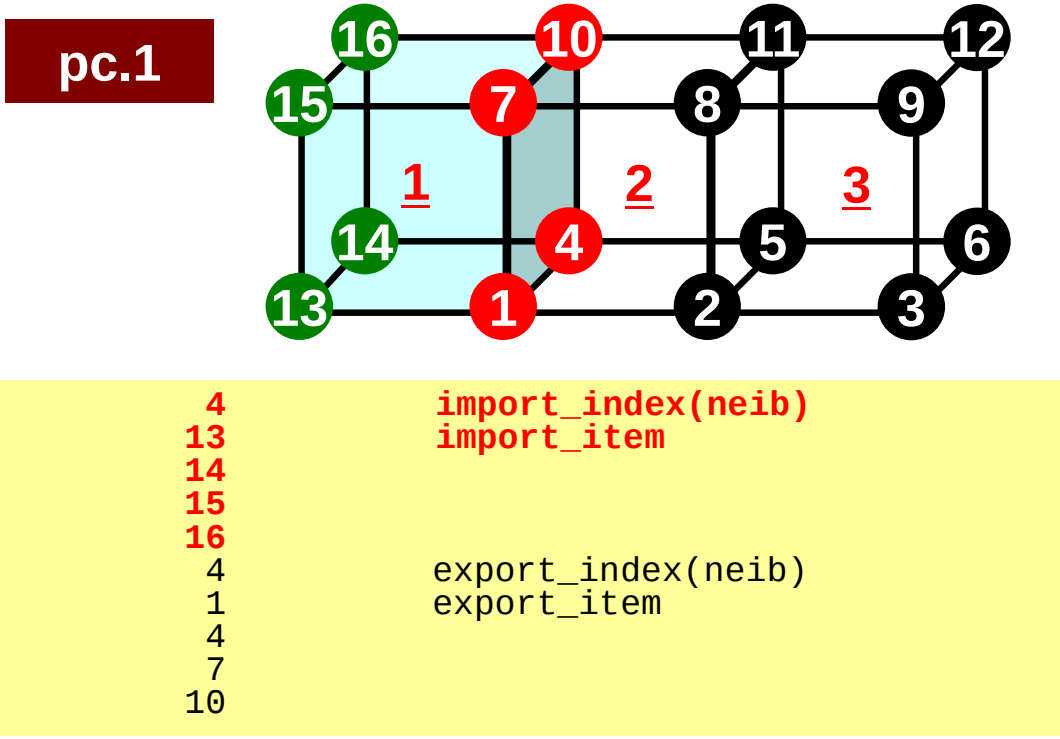
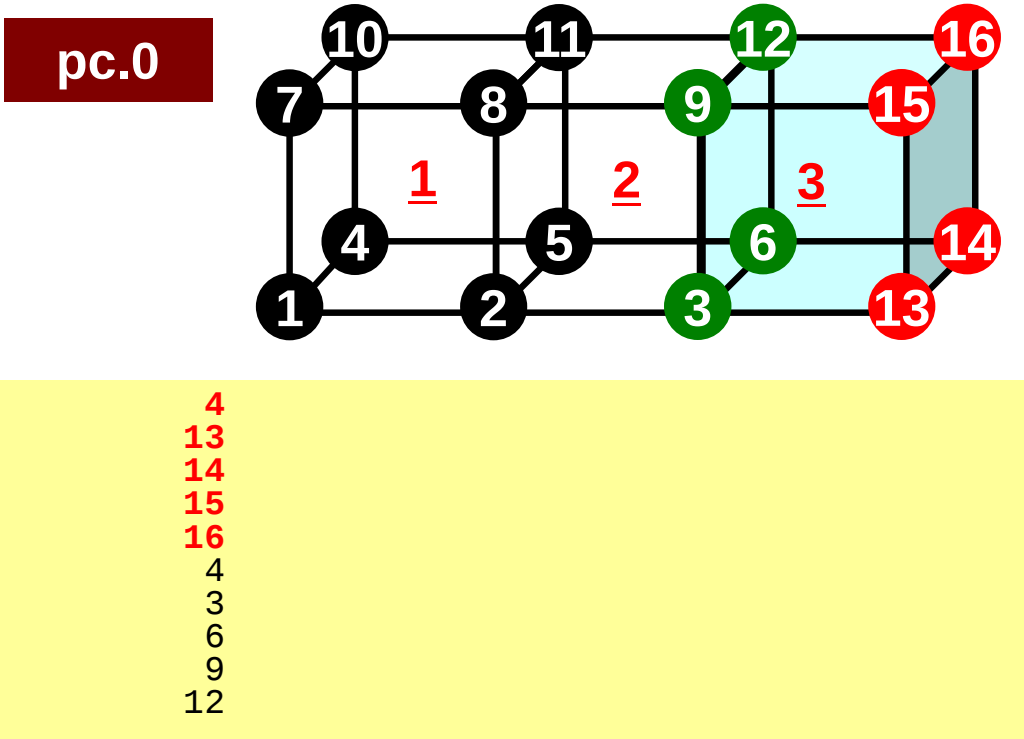
Generalized Comm. Table: Receive

- Neighbors
 - NeibPETot, NeibPE[NeibPETot]
- Message size for each neighbor
 - import_index [NeibPETot+1]
- ID of **external** points
 - import_item [import_index[NeibPETot+1]]
- Messages from each neighbor
 - RecvBuf [import_index[NeibPETot+1]]

Communication Table (Recv/Import)

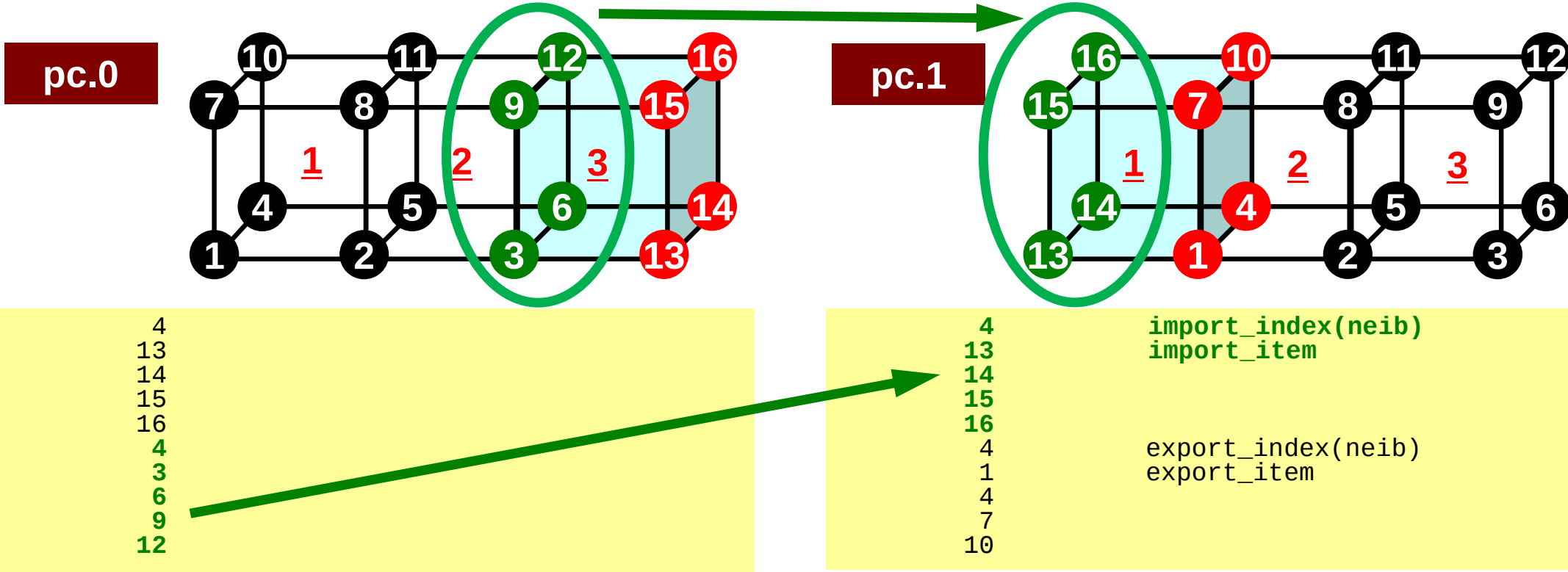


Communication Table (Recv/Import)



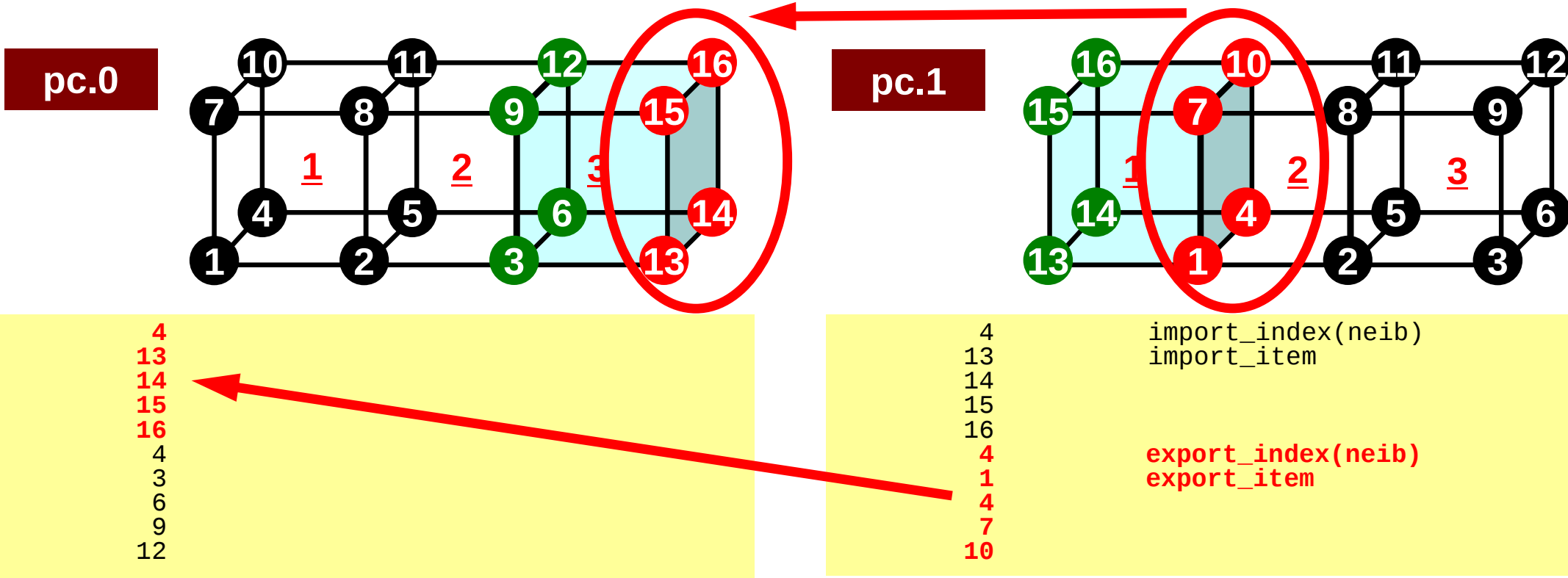
- `import_index` Size of Messages recv. from Each Neighbor
 - # Neighbors= 1 in this case
- `import_item` Local ID of external points

Communication Table (Recv/Import)



- import_index Size of Messages recv. from Each Neighbor
 - # Neighbors= 1 in this case
- import_item Local ID of external points

Communication Table (Recv/Import)



- **import_index** Size of Messages recv. from Each Neighbor
 - # Neighbors= 1 in this case
- **import_item** Local ID of external points



RECV: MPI_Isend/Irecv/Waitall

```
for (neib=0; neib<NeibPETot; neib++){
    tag= 0;
    iS_i= import_index[neib];
    iE_i= import_index[neib+1];
    BUFlength_i= iE_i - iS_i

    ierr= MPI_Irecv
        (&RecvBuf[iS_i], BUFlength_i, MPI_DOUBLE, NeibPE[neib], 0,
         MPI_COMM_WORLD, &ReqRecv[neib])
}

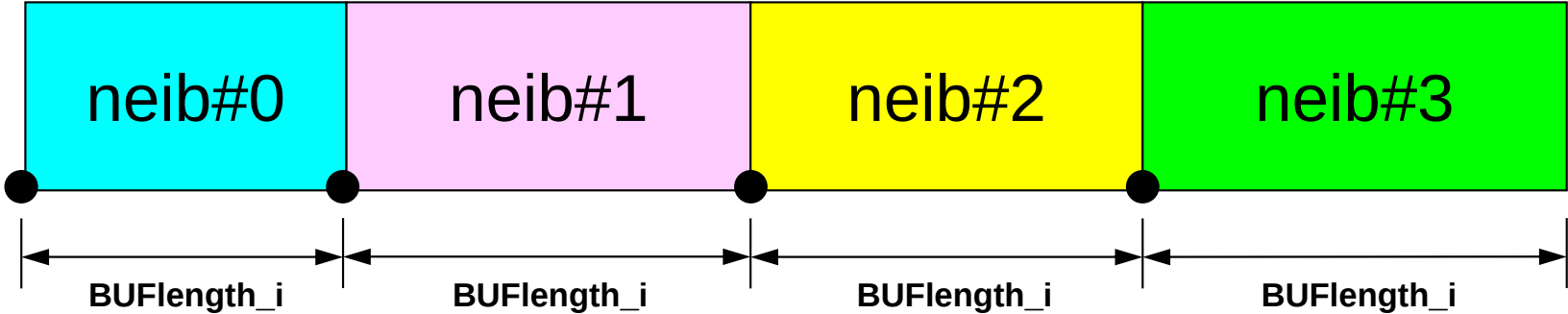
MPI_Waitall(NeibPETot, ReqRecv, StatRecv);

for (neib=0; neib<NeibPETot;neib++){
    for (k=import_index[neib];k<import_index[neib+1];k++){
        kk= import_item[k];
        VAL[kk]= RecvBuf[k];
    }
}
```

Copied from receiving buffer

import_item (import_index[neib]:import_index[neib+1]-1) are received from neib-th neighbor

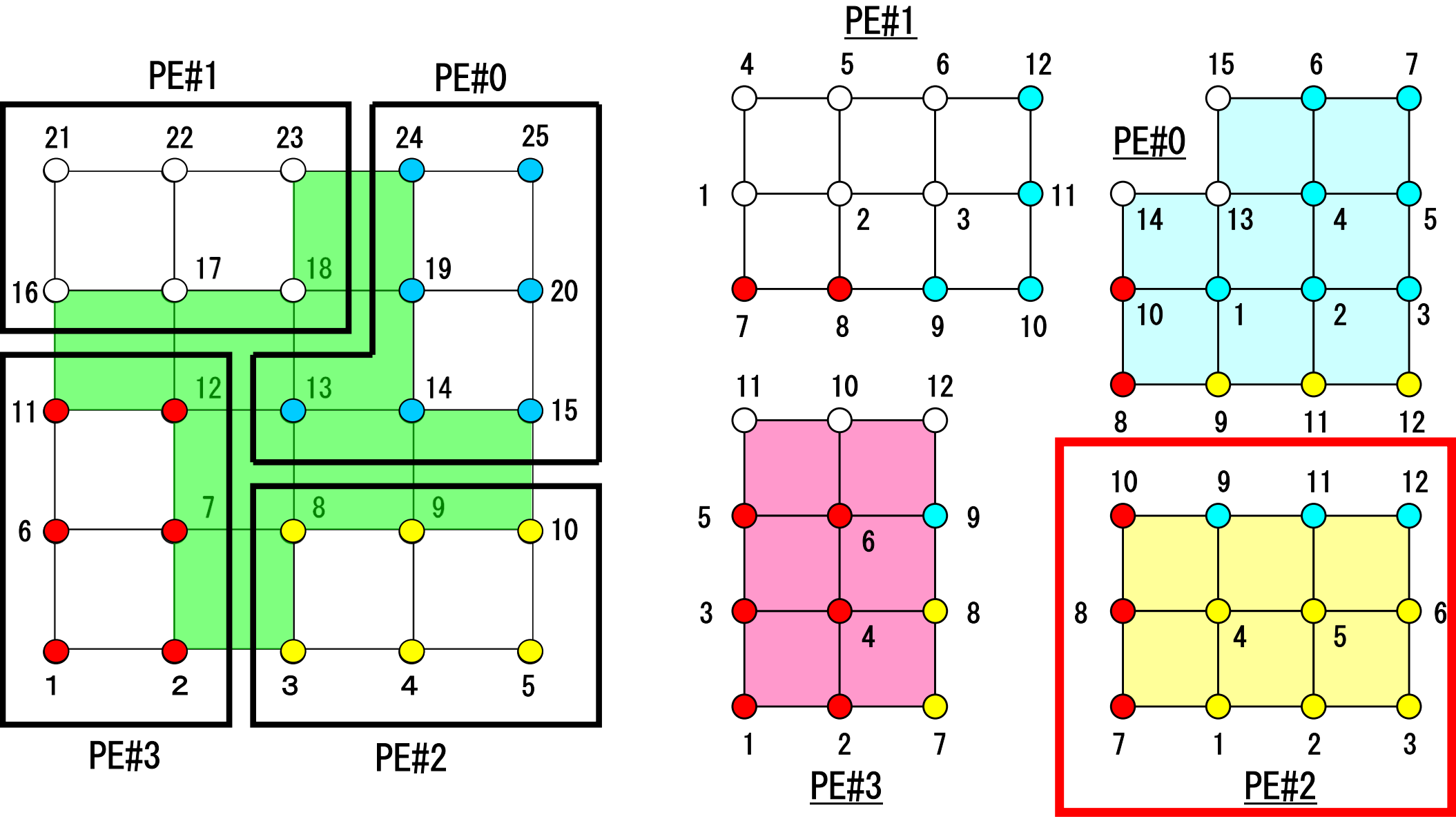
RecvBuf



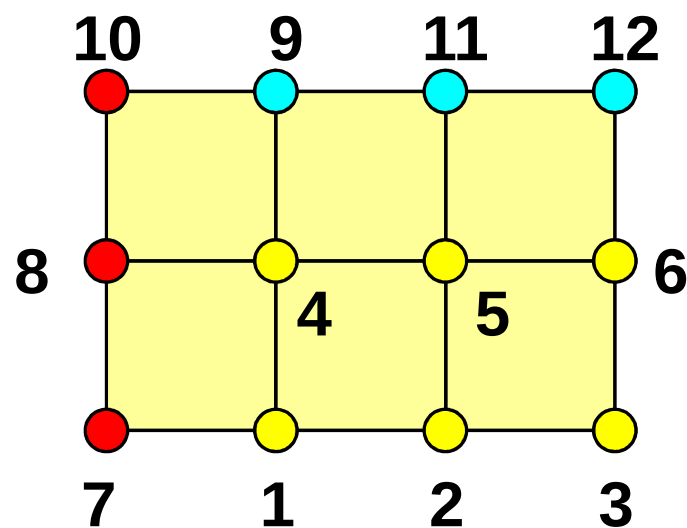
import_index[0] import_index[1] import_index[2] import_index[3] import_index[4]

Node-based Partitioning

internal nodes - elements - external nodes



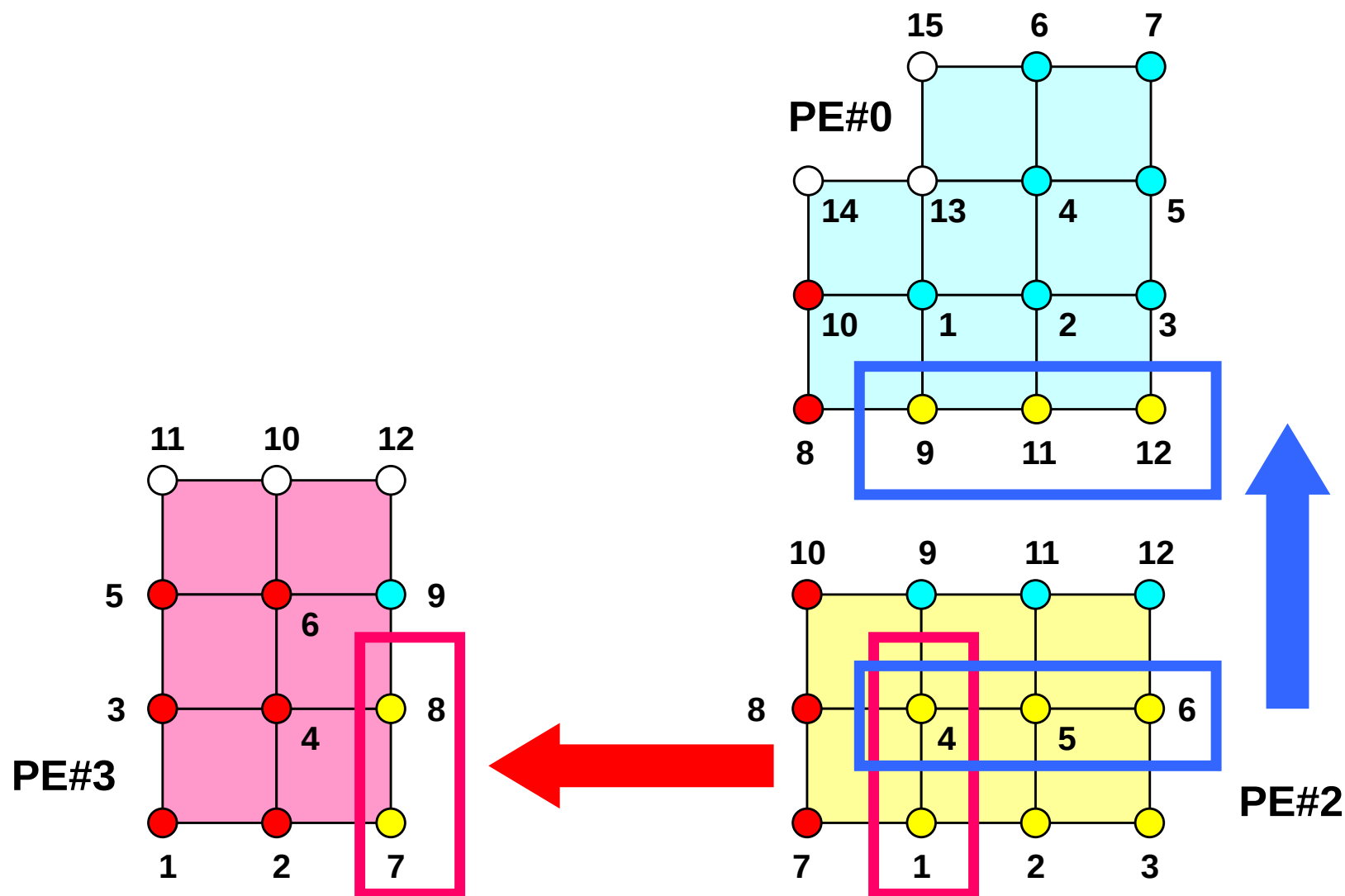
Description of Distributed Local Data



- Internal/External Points
 - Numbering: Starting from internal pts, then external pts after that
- Neighbors
 - Shares overlapped meshes
 - Number and ID of neighbors
- External Points
 - From where, how many, and which external points are received/imported ?
- Boundary Points
 - To where, how many and which boundary points are sent/exported ?

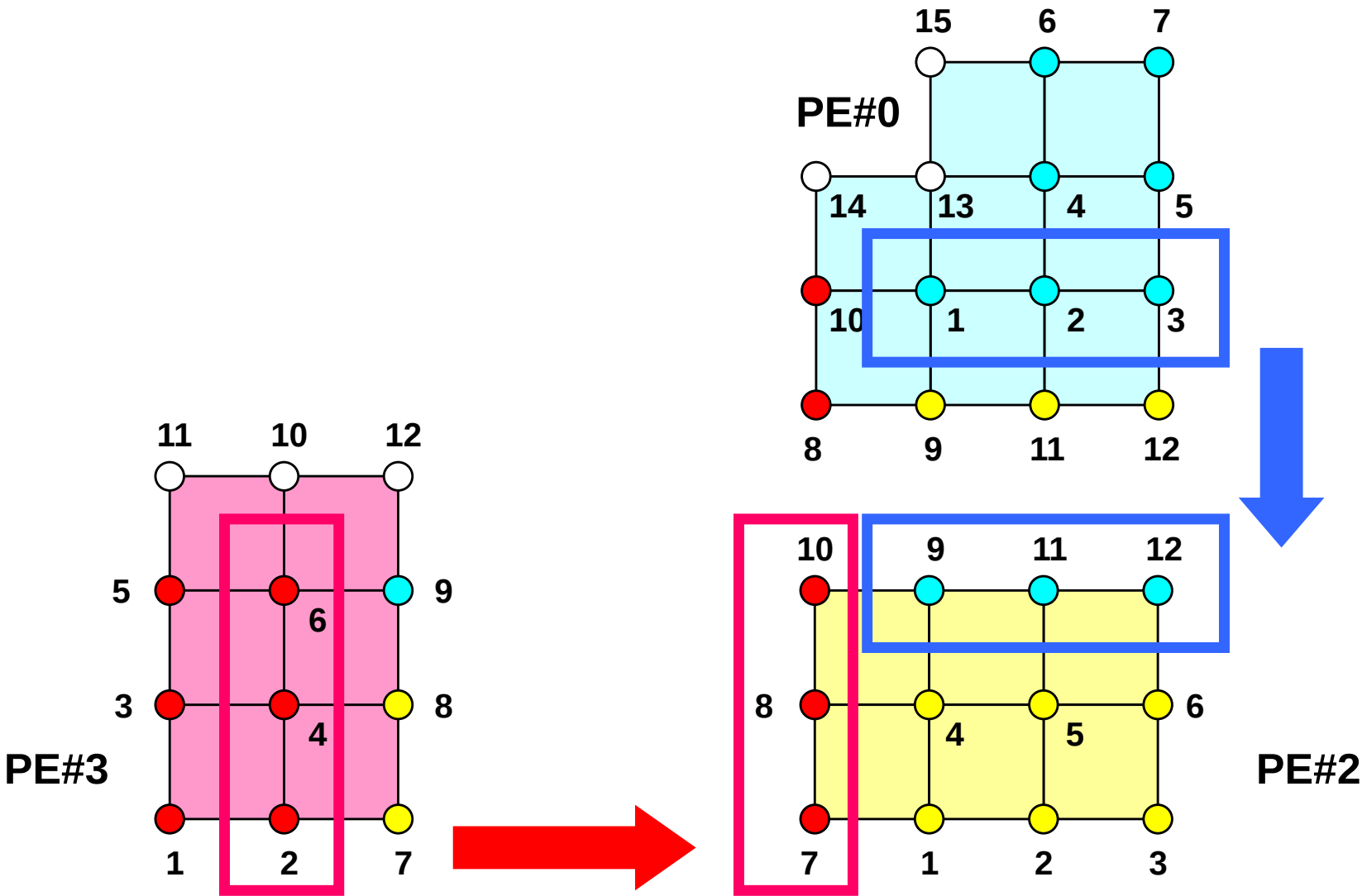
Boundary Nodes (境界点) : SEND

PE#2 : send information on “boundary nodes”

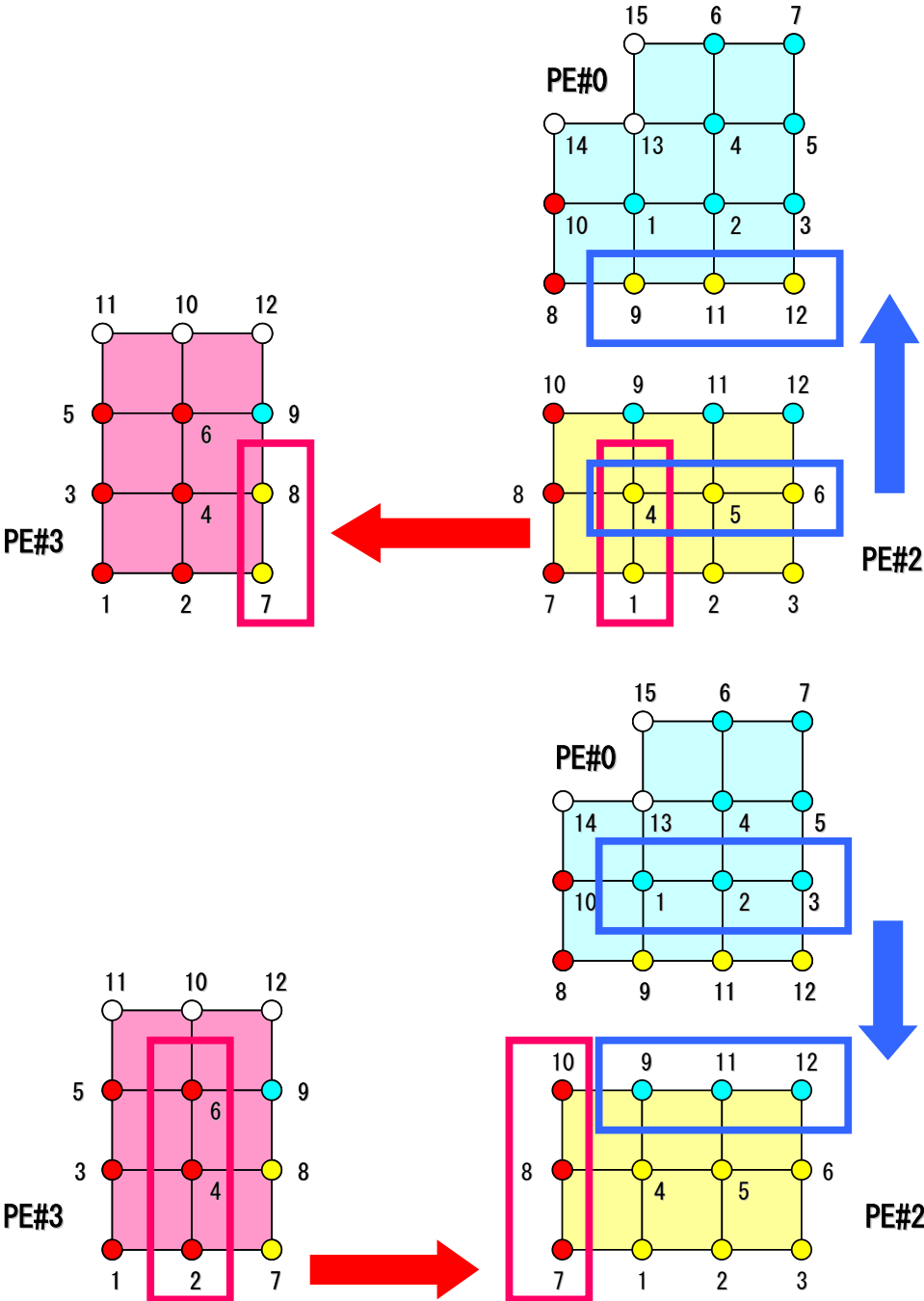


External Nodes (外点) : RECEIVE

PE#2 : receive information for “external nodes”



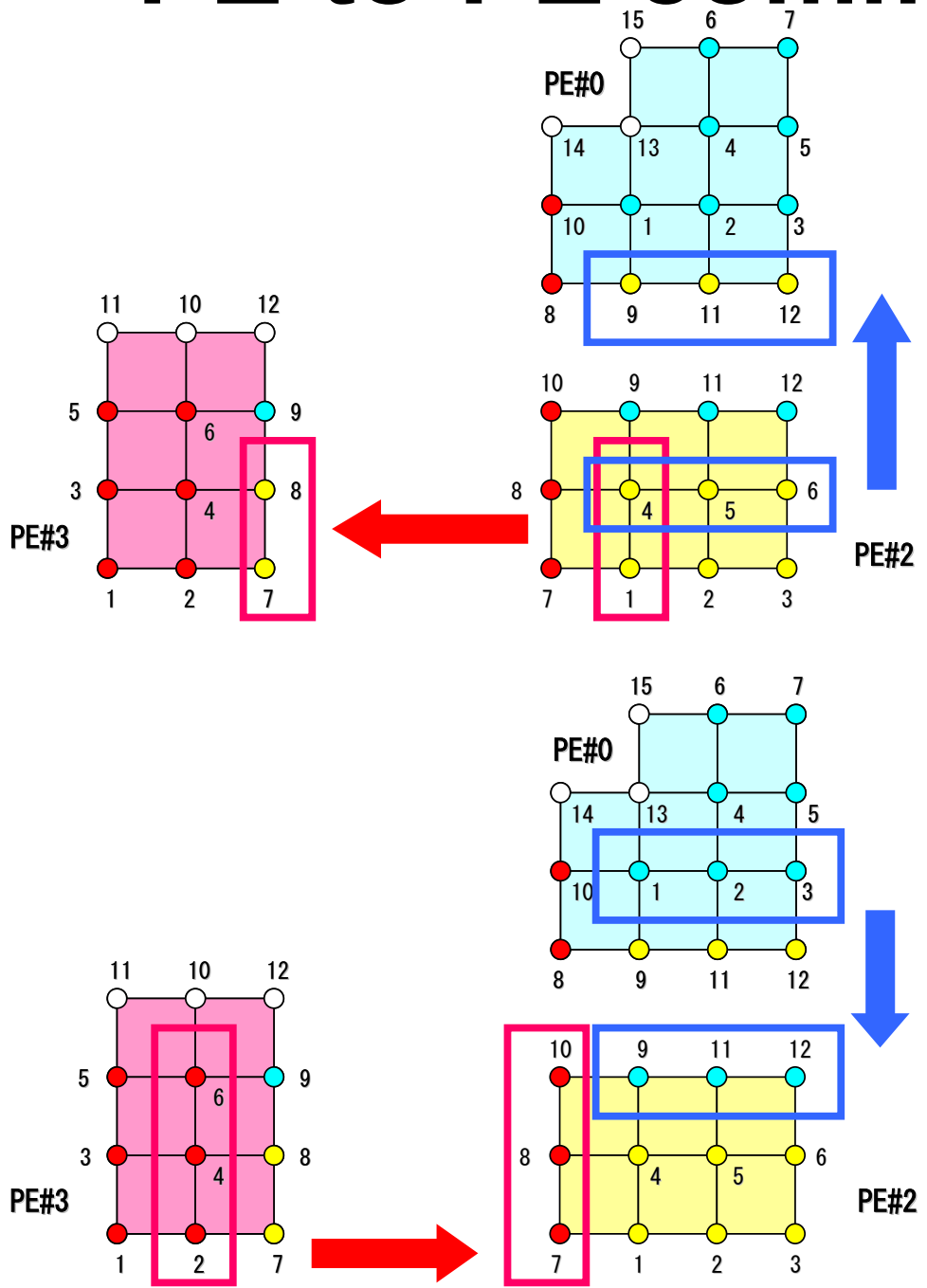
PE-to-PE comm. : Local Data



(中略)

2	
2	
3	0
3	6
7	
8	
10	
9	
11	
12	
2	5
1	
4	
4	
5	
6	

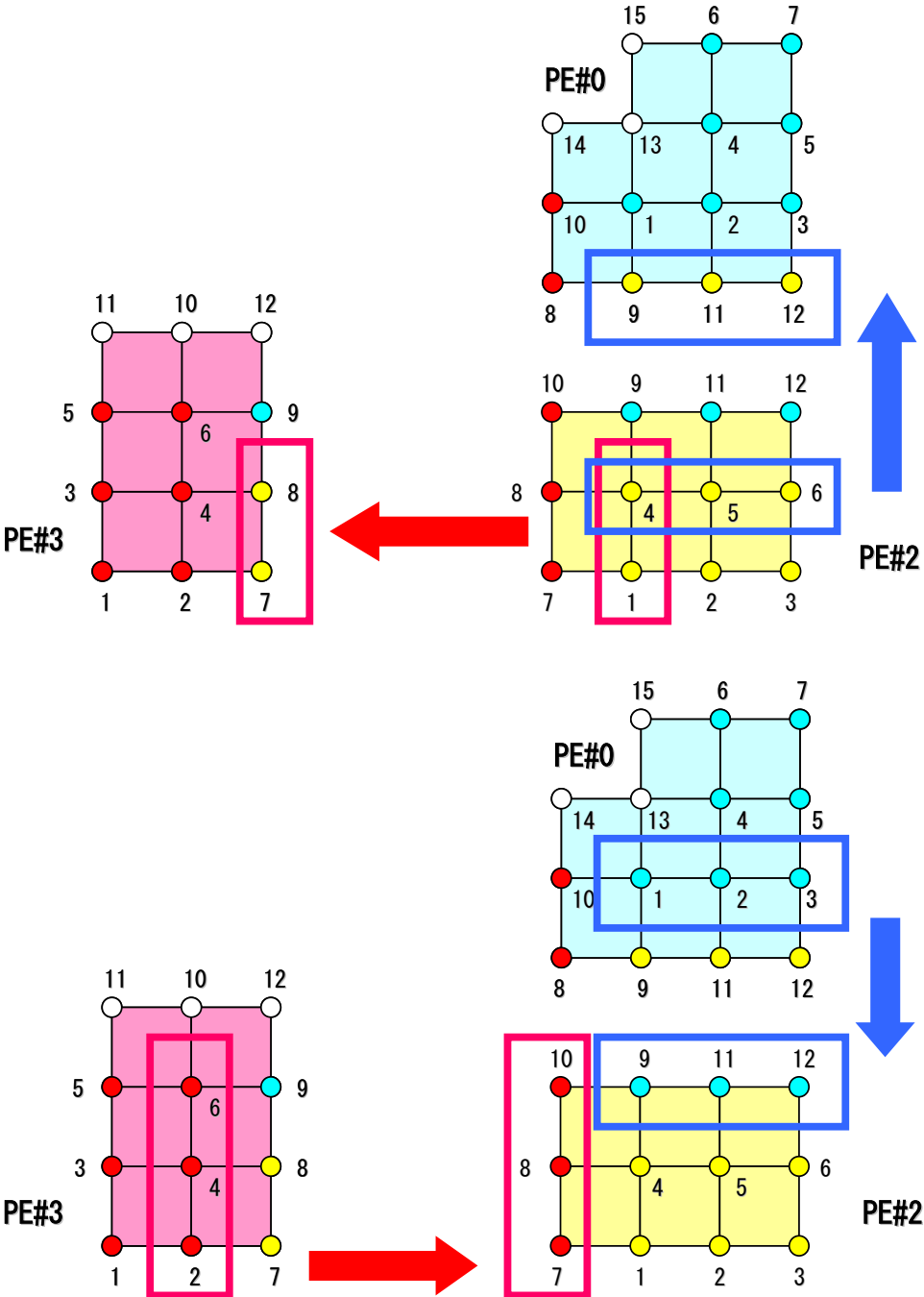
PE-to-PE comm. : Local Data (C)



	2	ID of process
	2	Num. of Neighbors
	3	ID of Neighbors
(中略)	0	
	3	6
	7	
	8	
	10	
	9	
	11	
	12	
	2	5
	1	
	4	
	4	
	5	
	6	

NEIBPETOT= 2
NEIBPE[0]=3, NEIBPE[1]= 0

PE-to-PE comm. : SEND (C)



(中略)

2	
2	
3	0
3	6
7	
8	
10	
9	
11	
12	
2	
1	
4	
4	
5	
6	

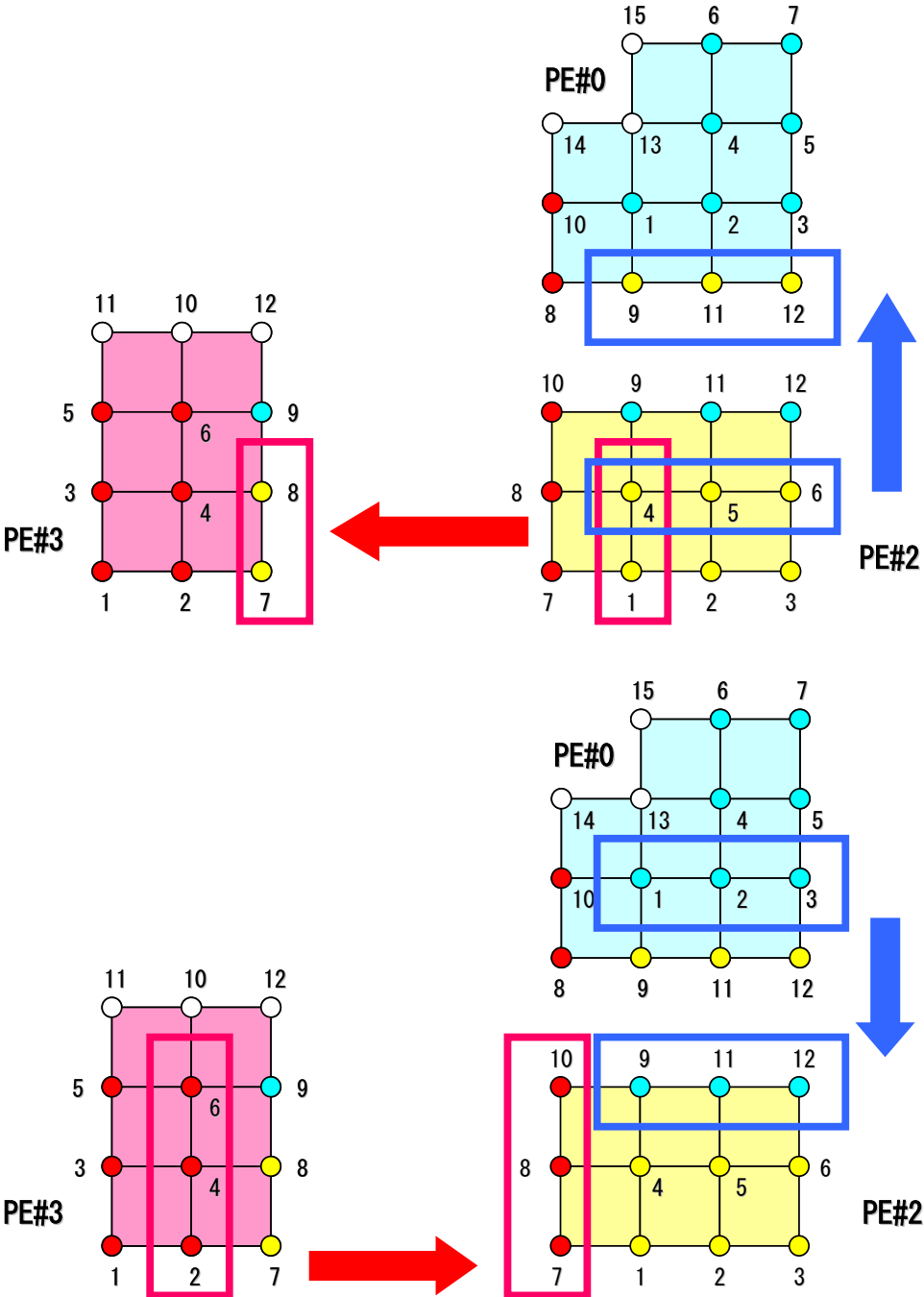
5 export_index

export_index[0]= 0
export_index[1]= 2
export_index[2]= 2+3 = 5

export_item[0-4]=1,4,4,5,6

Node "4" is sent to two processes (PE)

PE-to-PE comm. : RECV (C)



(中略)

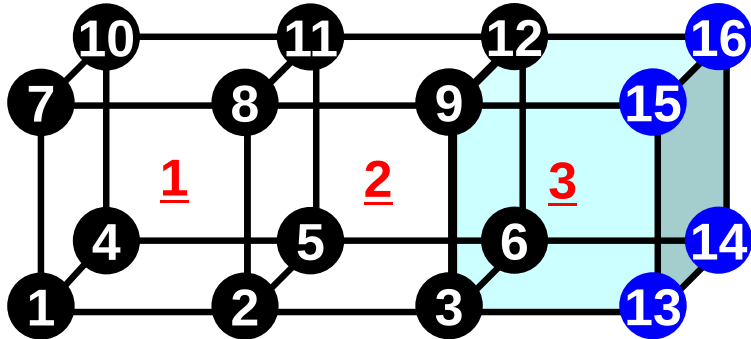
2	
2	
3	0
3	6
7	import_index
8	
10	
9	
11	
12	
2	5
1	
4	
4	
5	
6	

```
import_index[0]= 0
import_index[1]= 3
import_index[2]= 3+3 = 6

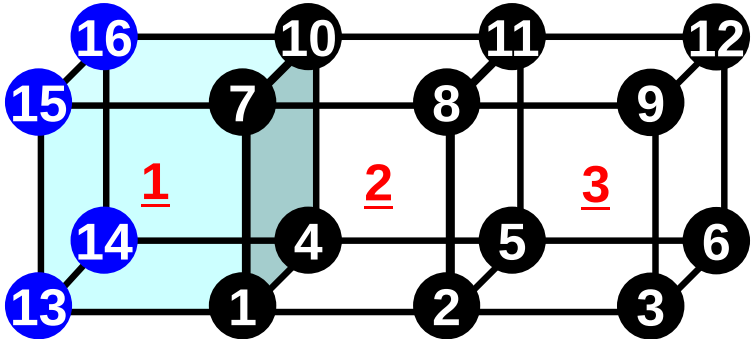
import_item[0-5]=7,8,10,9,11,12
```

Node Group

pc.0



pc.1



4							
4	12	20	28				
Xmin							
1	4	7	10				
Ymin							
1	2	3	13	7	8	9	15
Zmin							
1	2	3	13	4	5	6	14
Zmax							
7	8	9	15	10	11	12	16

4							
0	8	16	24				
Xmin							
Ymin							
13	1	2	3	15	7	8	9
Zmin							
13	1	2	3	14	4	5	6
Zmax							
15	7	8	9	16	10	11	12

- pc.1
 - Because there are node nodes which belong to “Xmin”, number of node is “0”.