



HPC Middleware (HPC-MW)

Infrastructure for Scientific Applications on HPC Environments

Overview and Recent Progress

Kengo Nakajima
RIST, FSIS/University of Tokyo and CREST/JST

4th ACTS Workshop, August 5-8, 2003.
Lawrence Berkeley National Laboratory
Berkeley, CA, USA.

Frontier Simulation Software for Industrial Science (FSIS)

<http://www.fsis.iis.u-tokyo.ac.jp>

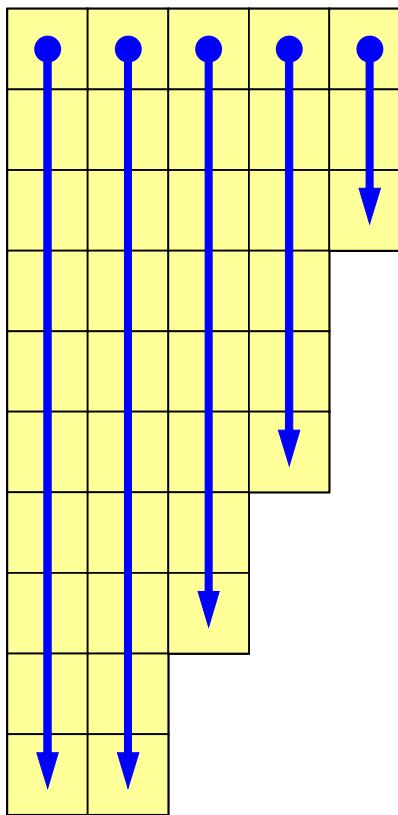
- Part of "IT Project" in MEXT (Ministry of Education, Culture, Sports, Science & Technology)
- HQ at Institute of Industrial Science, University of Tokyo
 - 5 years
 - 12M USD/yr.
 - 7 internal projects
 - > 100 people involved.
- **Focused on Industry Use, Public Use.**

Background

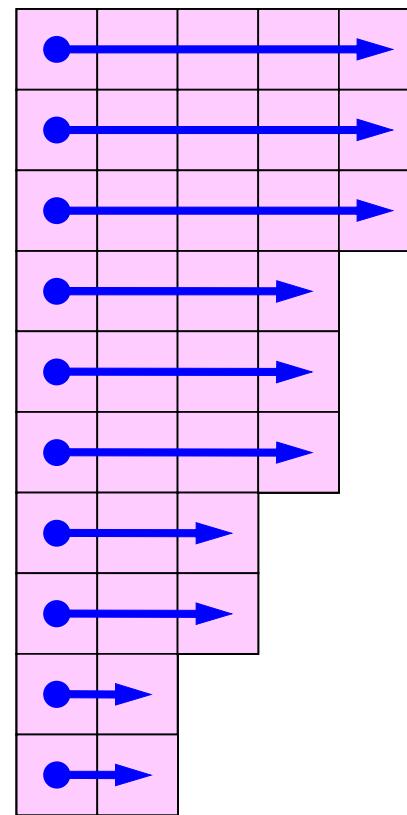
- Various Types of HPC Platforms
 - Parallel Computers
 - PC Clusters
 - MPP with Distributed Memory
 - SMP Cluster (8-way, 16-way, 256-way)
 - Power, HP-RISC, Alpha/Itanium, Pentium, Vector PE
 - **GRID Environment -> Various Resources**
- Parallel/Single PE Optimization is important !!
 - Portability under GRID environment
 - Machine-dependent optimization/tuning.
 - Everyone knows that ... but it's a big task especially for application experts, scientists.

Reordering for SMP Cluster with Vector PEs: ILU Factorization

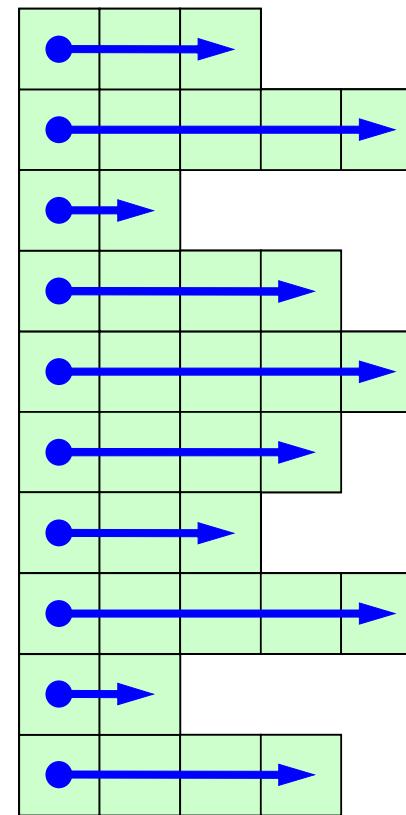
PDJDS/CM-RCM



**PDCRS/CM-RCM
short innermost loop**



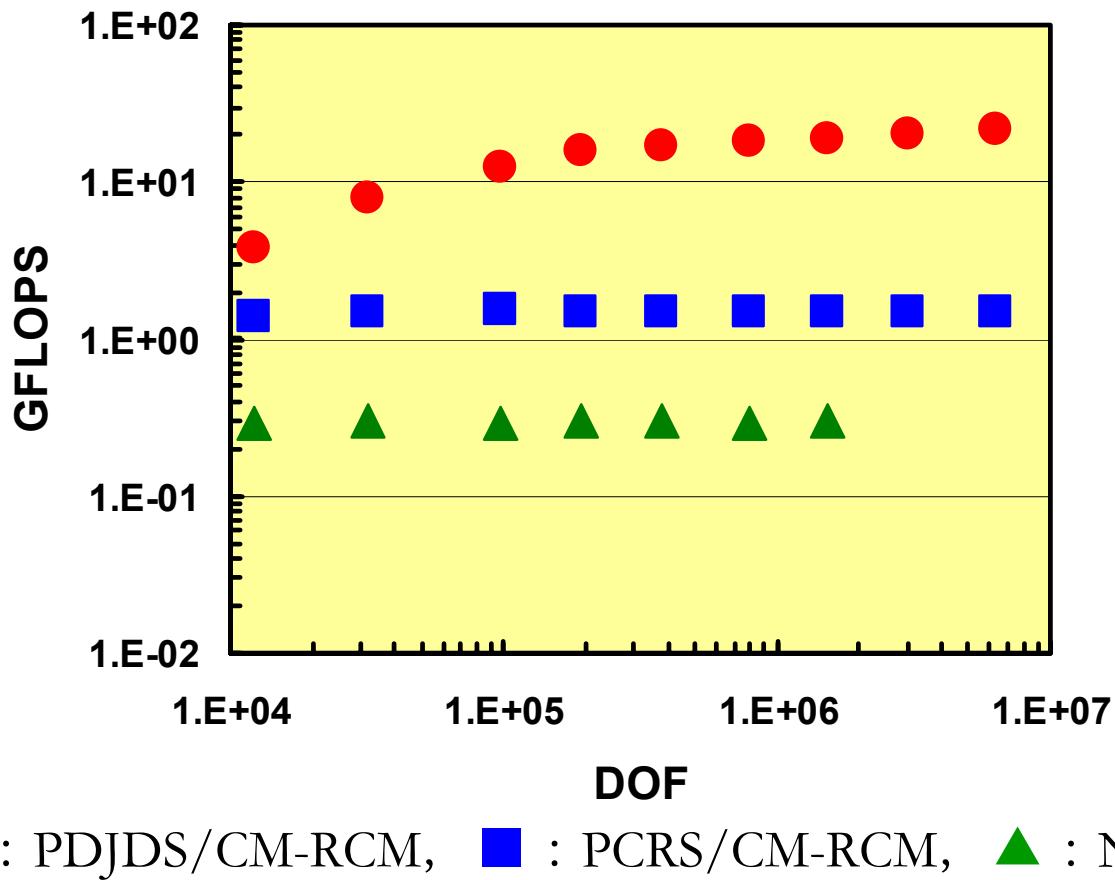
CRS no re-ordering



3D Elastic Simulation

Problem Size~GFLOPS

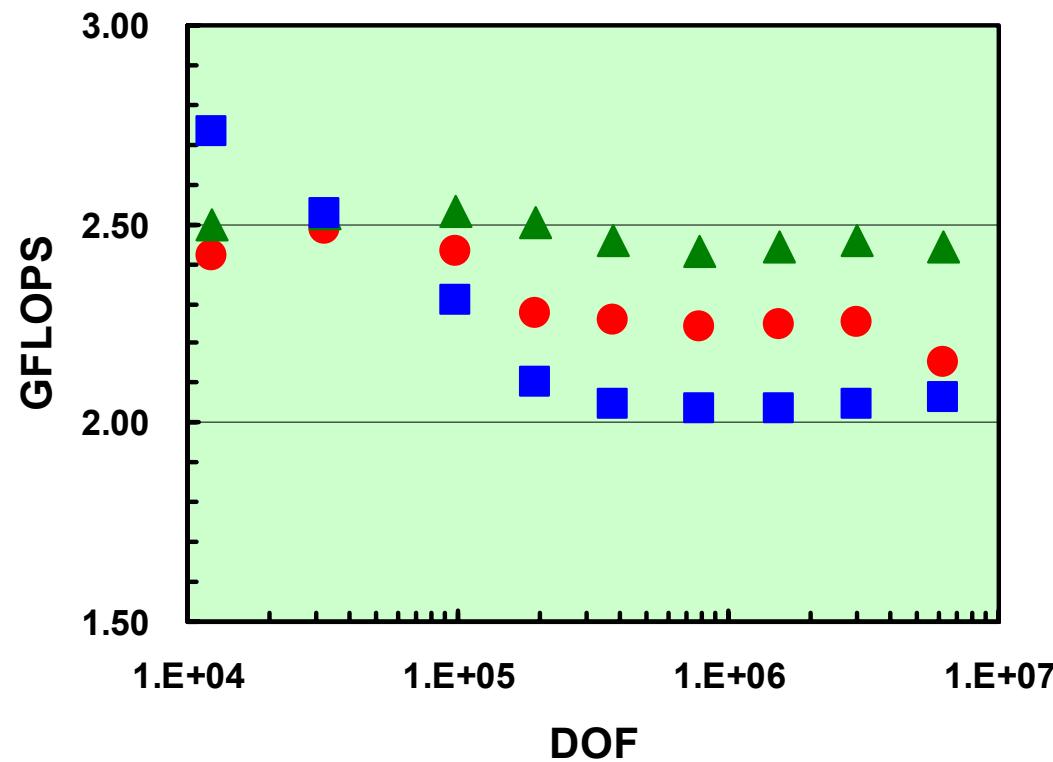
Earth Simulator/SMP node (8 PEs)



3D Elastic Simulation

Problem Size~GFLOPS

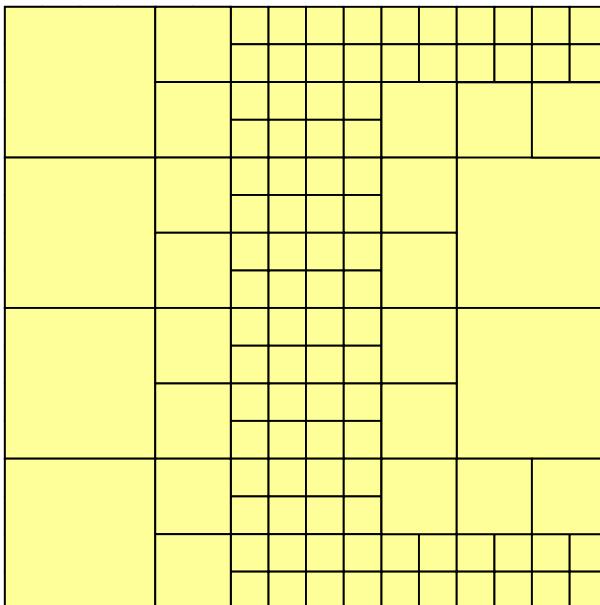
Intel Xeon 2.8 GHz, 8 PEs



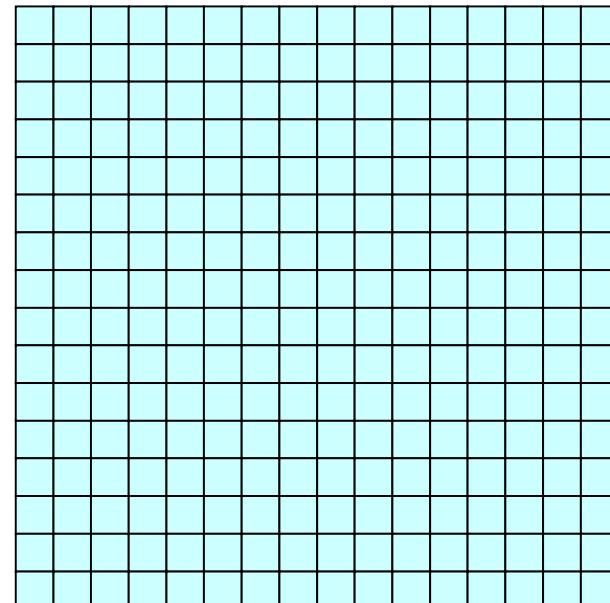
● : PDJDS/CM-RCM, ■ : PCRS/CM-RCM, ▲ : Natural Ordering

Volume Rendering Module using Voxels

On PC Cluster



On Earth Simulator

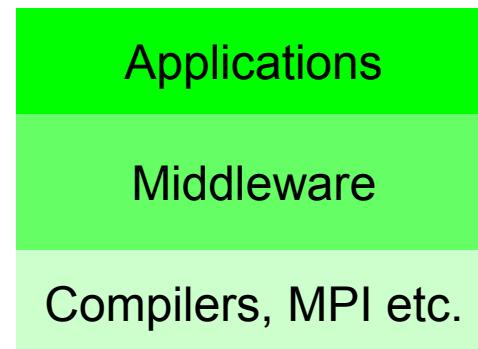


- Hierarchical Background Voxels
- Linked-List

- Globally Fine Voxels
- Static-Array

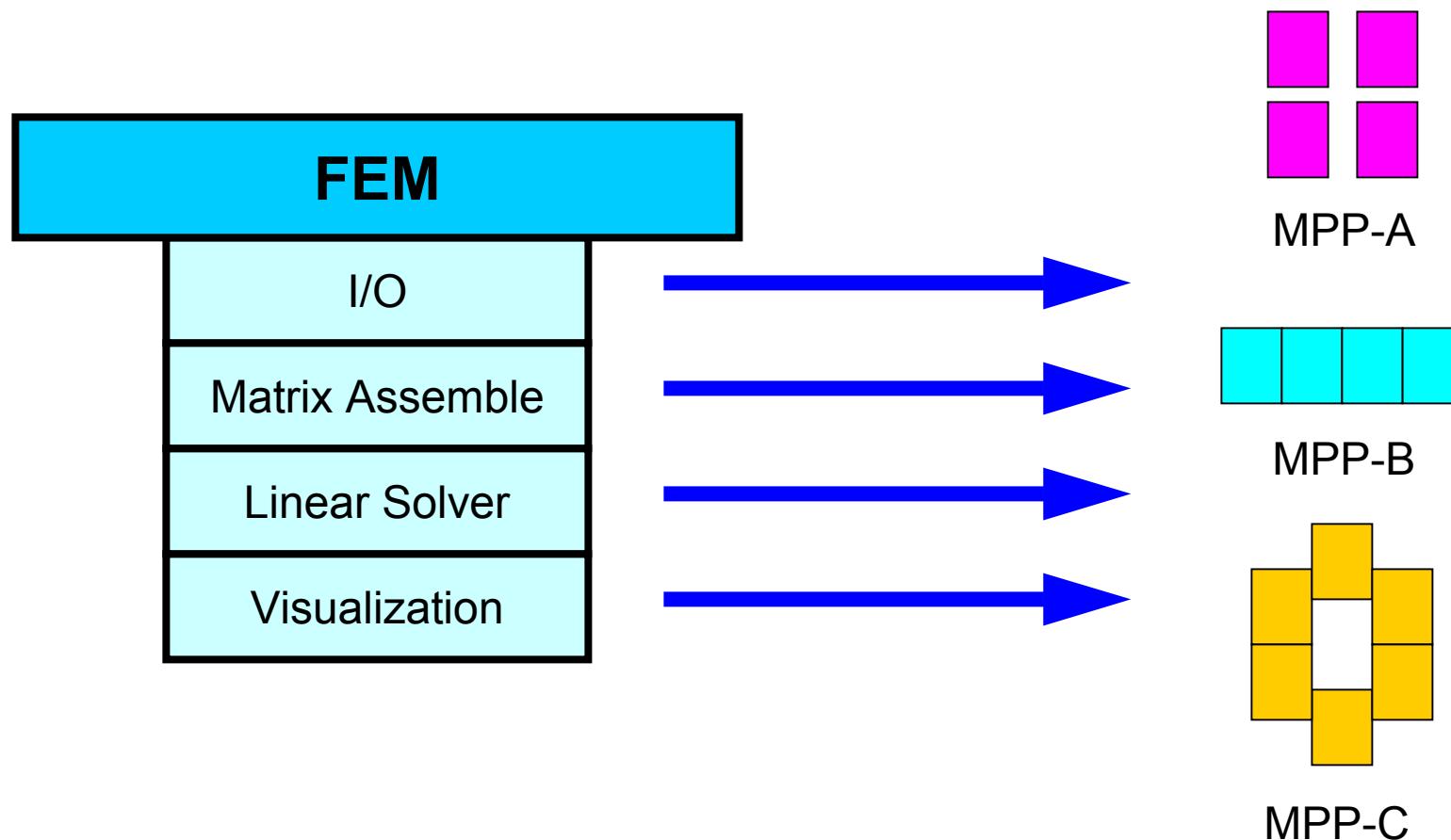
Background (cont.)

- Simulation methods such as FEM, FDM etc. have several typical processes for computation.
- How about "*hiding*" these process from users by **Middleware** between applications and compilers ?
 - Development: efficient, reliable, portable, easy-to-maintain
 - accelerates advancement of the applications (= physics)
 - **HPC-MW = Middleware close to "Application Layer"**



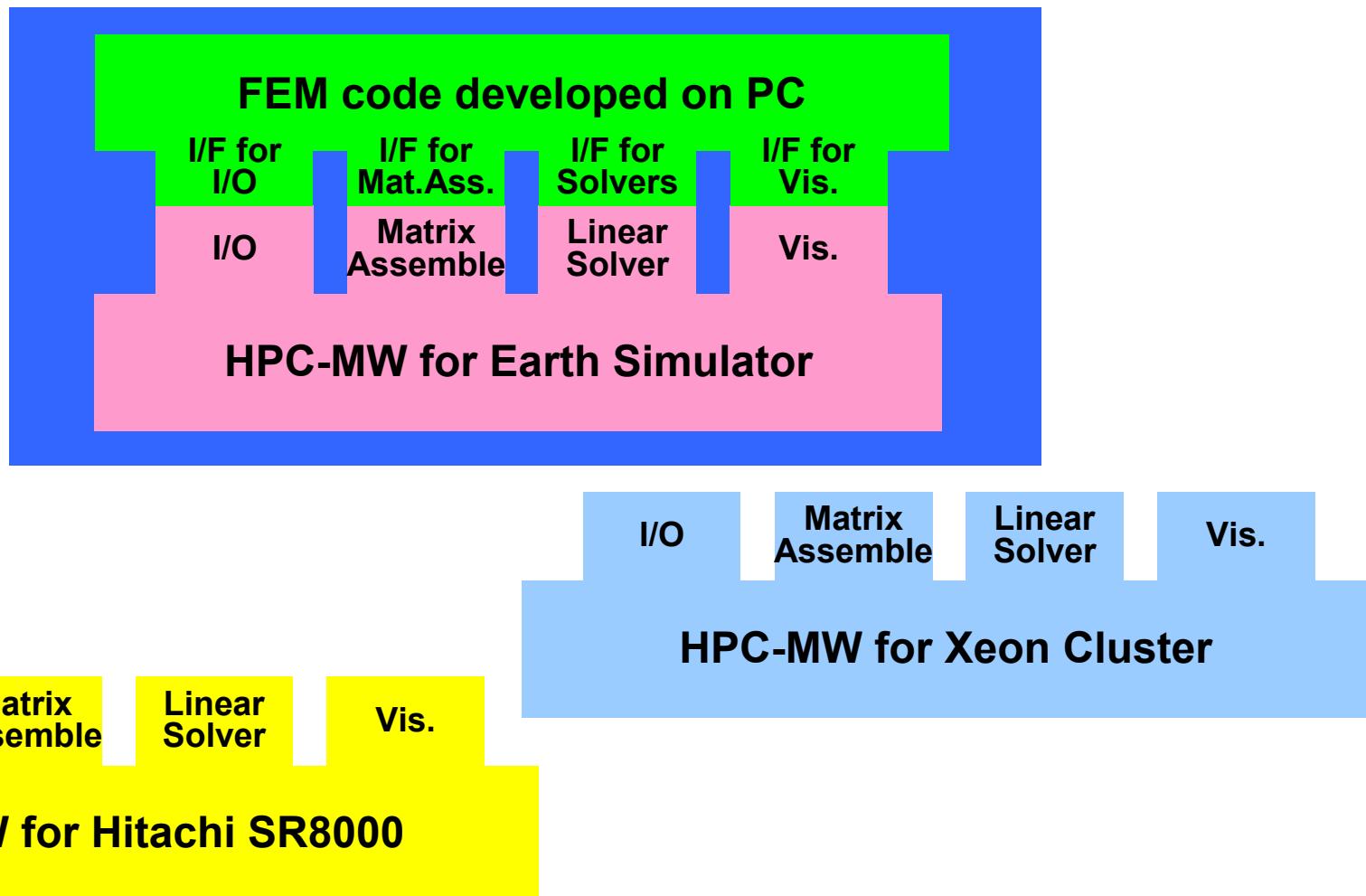
Example of HPC Middleware

Individual Process could be optimized for
Various Types of MPP Architectures



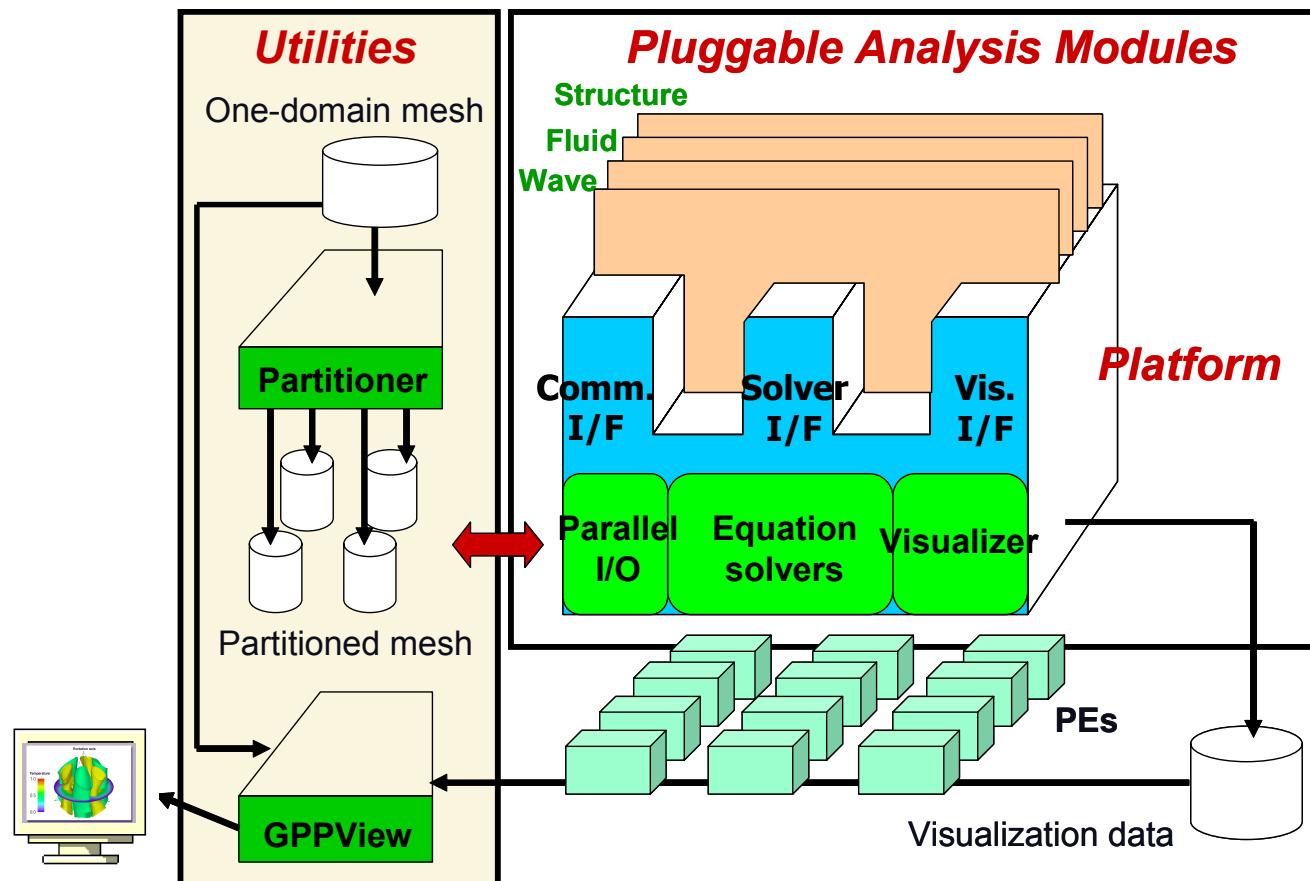
Example of HPC Middleware

Parallel FEM Code Optimized for ES



HPC Middleware (HPC-MW) ?

- Based on idea of "Plug-in" in GeoFEM.

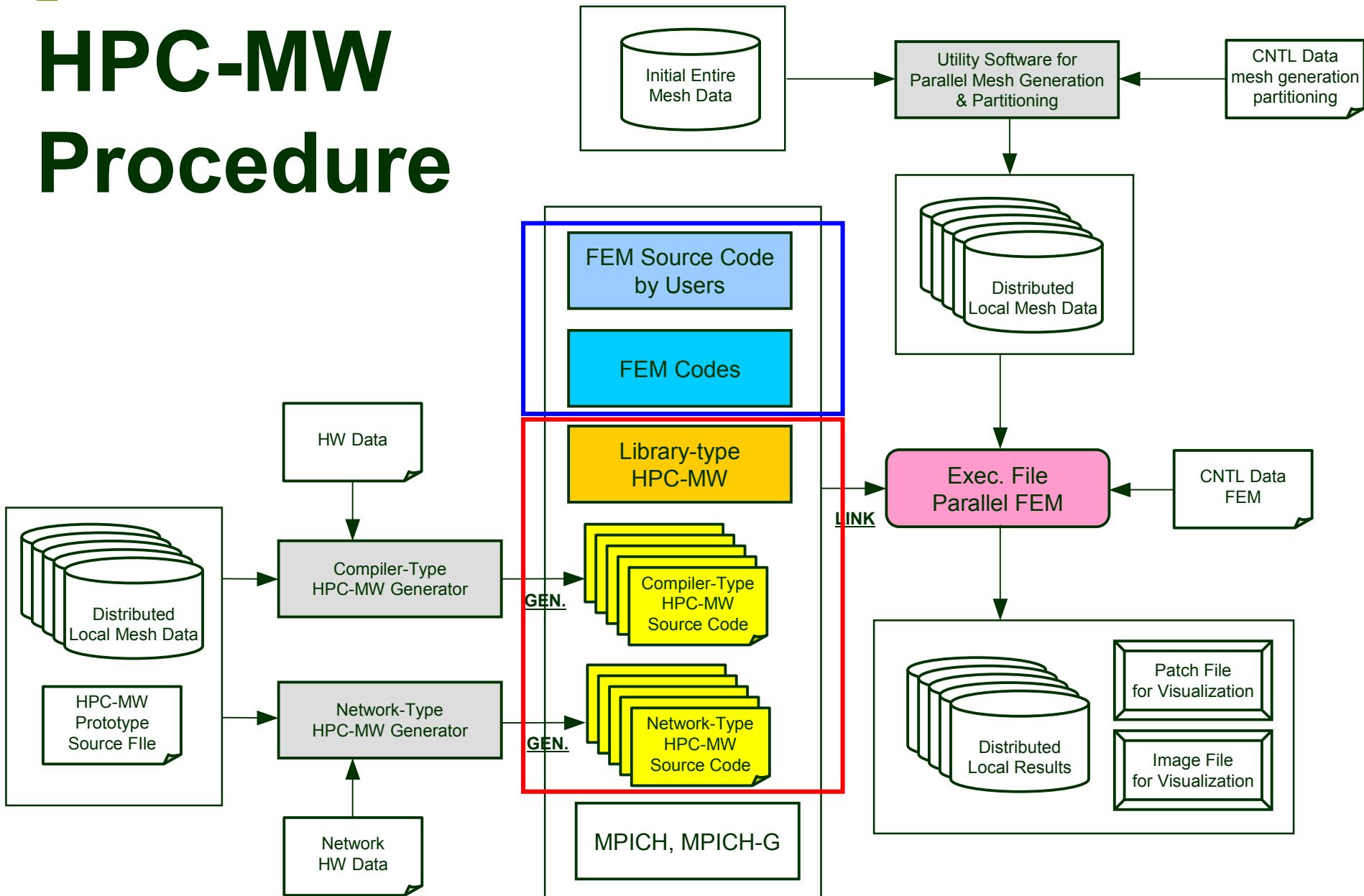


<http://geofem.tokyo.rist.or.jp/>

What can we do by HPC-MW ?

- We can develop optimized/parallel code easily on HPC-MW from user's code developed on PC.
- Library-Type
 - most fundamental approach
 - optimized library for individual architecture
- Compiler-Type
 - Next Generation Architecture
 - Irregular Data
- Network-Type
 - GRID Environment,
 - Large-Scale Computing (Virtual Petaflops), Coupling.

HPC-MW Procedure



What is new, What is nice ?

- Application Oriented (limited to FEM at this stage)
 - Various types of capabilities for parallel FEM are supported.
 - **NOT** just a library
- Optimized for Individual Hardware
 - Single Performance
 - Parallel Performance

Schedule

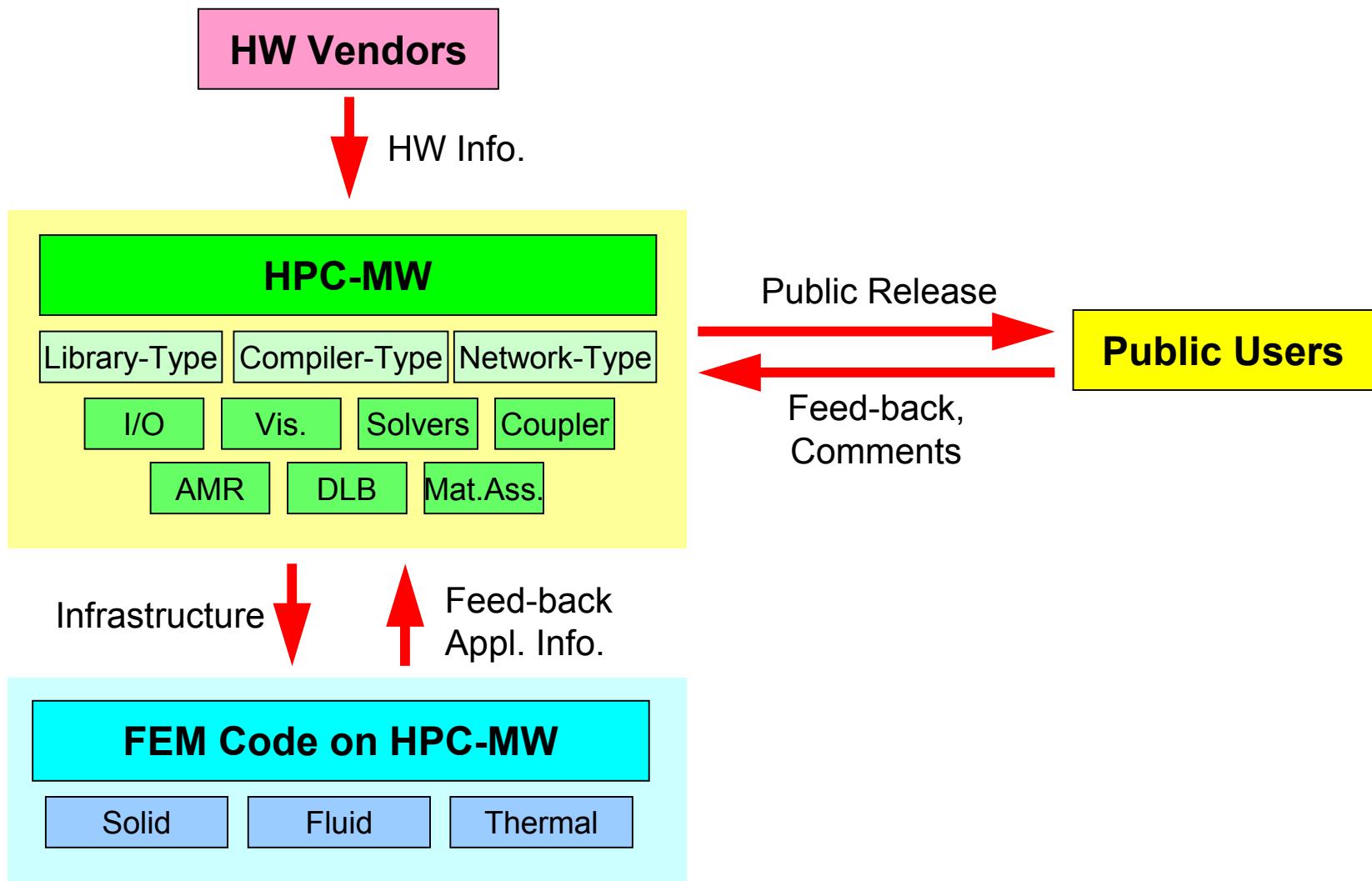
	FY.2002	FY.2003	FY.2004	FY.2005	FY.2006
Basic Design					
Prototype					
Library-type HPC-MW					
Compiler-type Network-type HPC-MW					
FEM Codes on HPC-MW					

Legend: ▲ Public Release

The chart illustrates the timeline for five key projects over a five-year period. The 'Basic Design' project begins in FY.2002. The 'Prototype' project starts in FY.2003. The 'Library-type HPC-MW' project begins in FY.2004. The 'Compiler-type Network-type HPC-MW' project begins in FY.2005. The 'FEM Codes on HPC-MW' project begins in FY.2006. Red arrows point to specific milestones: 'Scalar' in FY.2004, 'Vector' in FY.2005, 'Compiler' in FY.2006, and 'Network' in FY.2006. The 'Compiler-type Network-type HPC-MW' project is shown as a continuous blue bar spanning all five years.

▲ Public Release

System for Development





FY. 2003

- Library-Type HPC-MW
 - FORTRAN90, C, PC-Cluster Version
 - Public Release
 - Sept. 2003: Prototype Released
 - March 2004: Full version for PC Cluster
- Demonstration on Network-Type HPC-MW in SC2003, Phoenix, AZ, Nov. 2003.
- Evaluation by FEM Code

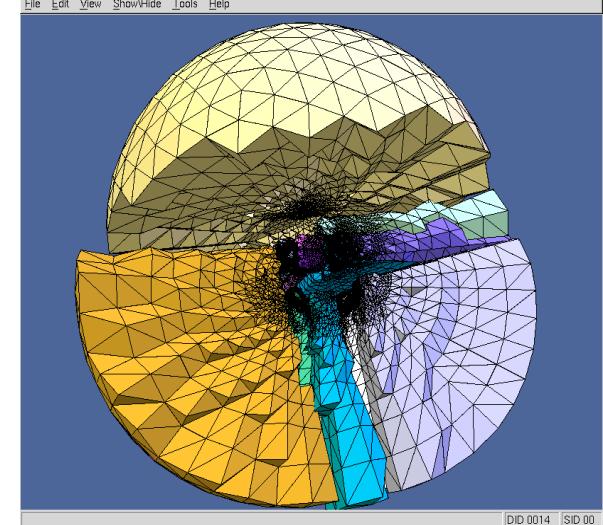
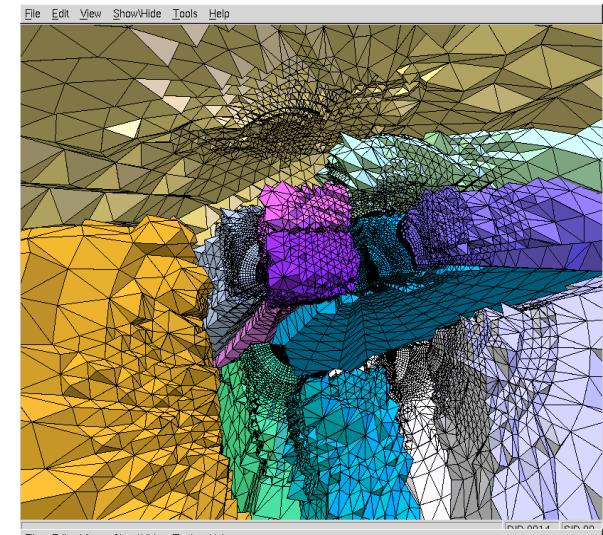
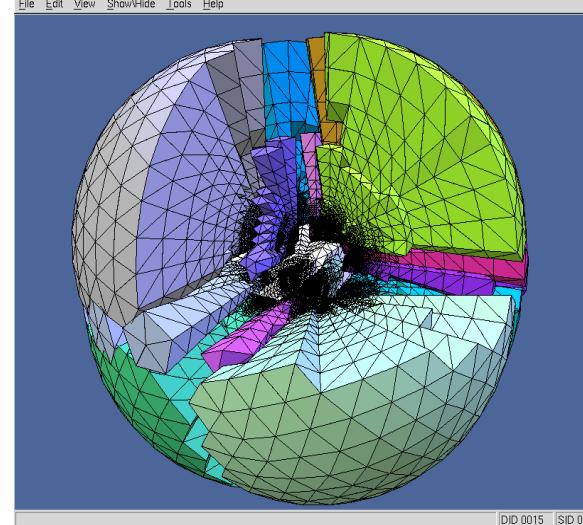
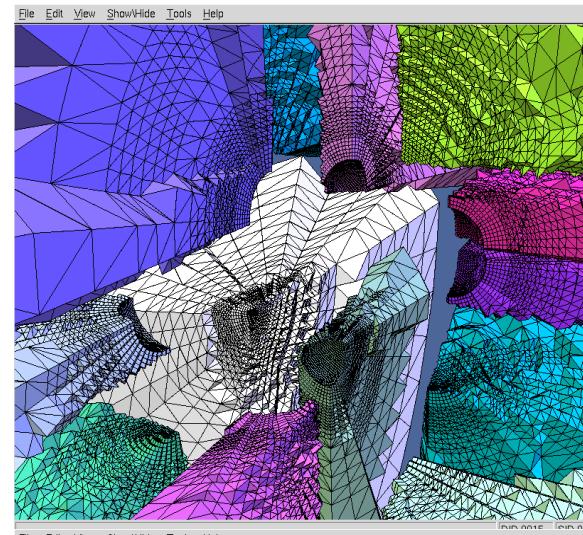
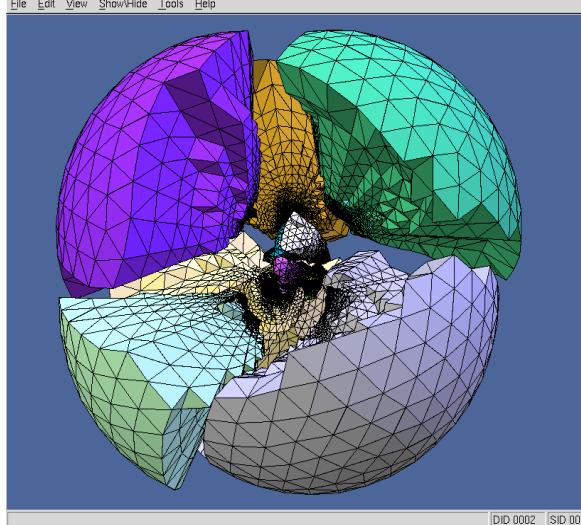
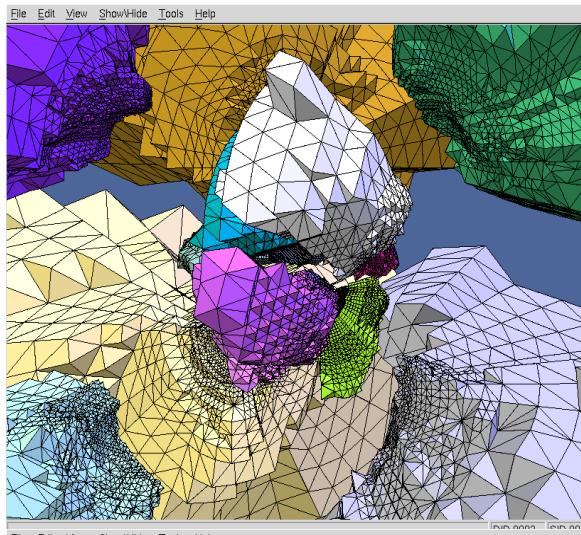
Library-Type HPC-MW

Mesh Generation is not considered

- Parallel I/O : I/F for commercial code (NASTRAN etc.)
- Adaptive Mesh Refinement (AMR)
- Dynamic Load-Balancing using **pMETIS** (DLB)
- Parallel Visualization
- Linear Solvers (GeoFEM + AMG, SAI)
- FEM Operations (Connectivity, Matrix Assembling)
- Coupling I/F
- Utility for Mesh Partitioning
- On-line Tutorial



AMR+DLB



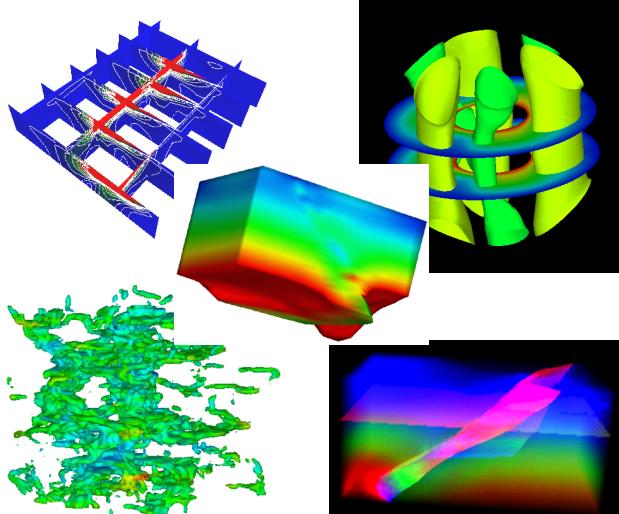
Parallel Visualization

Scalar Field

Surface rendering

Interval volume-fitting

Volume rendering



Vector Field

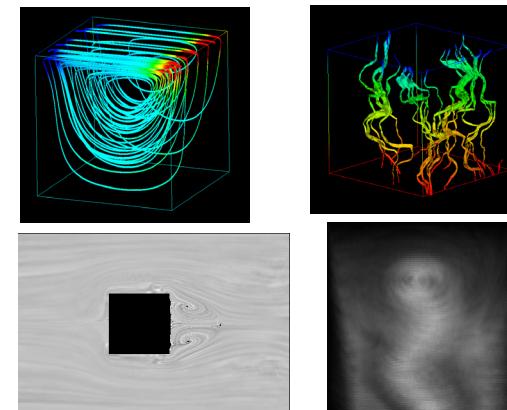
Streamlines

Particle tracking

Topological map

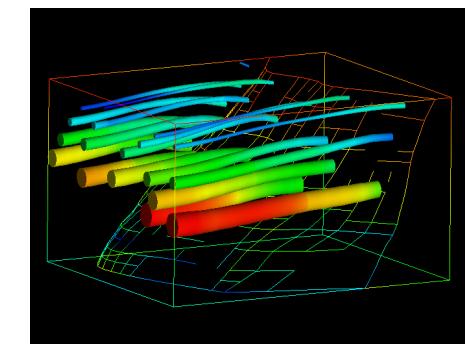
LIC

Volume rendering



Tensor Field

Hyperstreamlines



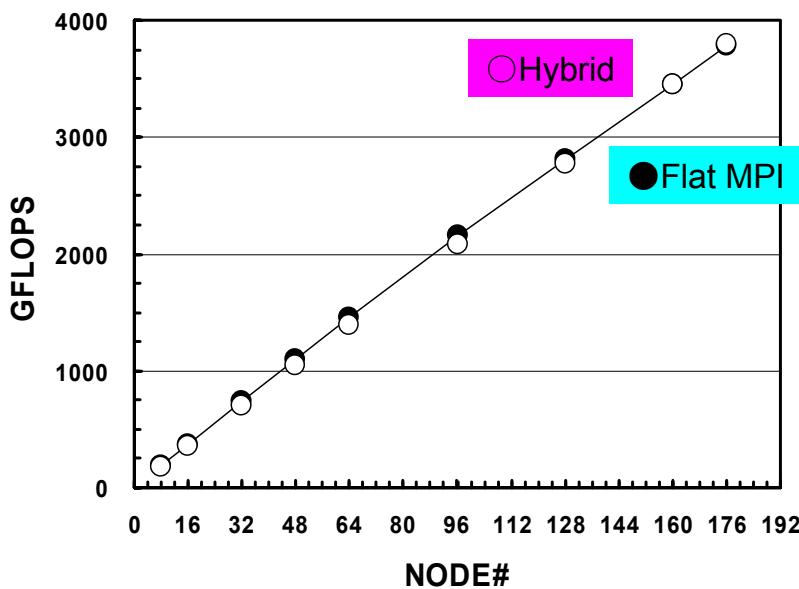
Extension of functions

Extension of dimensions

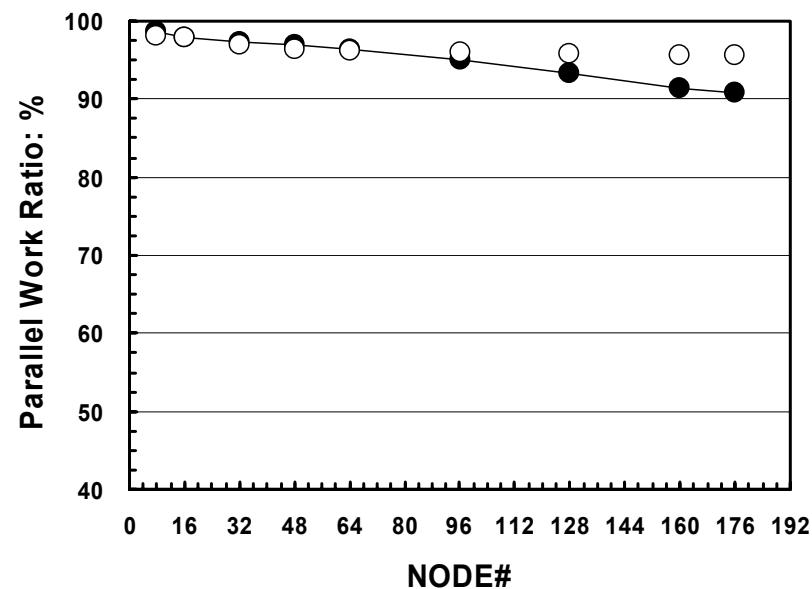
Extension of Data Types

Parallel Linear Solvers

GFLOPS rate

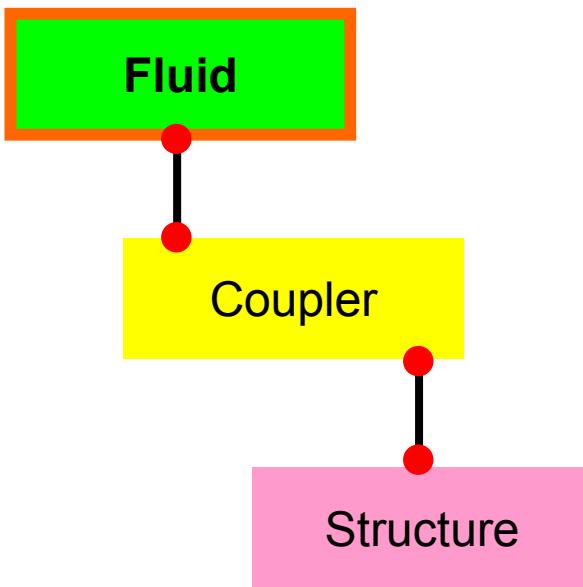


Parallel Work Ratio



On the Earth Simulator : 176 node, 3.8 TFLOPS

Coupler



```

module hpcmw_mesh
  type hpcmw_local_mesh
  ...
end type hpcmw_local_mesh
end module hpcmw_mesh
  
```

FLUID

```

program fluid

use hpcmw_mesh
type (hpcmw_local_mesh) :: local_mesh_b

...
call hpcmw_couple_PtoS_put
call structure_main
call hpcmw_couple_StoP_get
...
end program fluid
  
```

STRUCTURE

```

subroutine structure_main

use hpcmw_mesh
type (hpcmw_local_mesh) :: local_mesh_b

call hpcmw_couple_PtoS_get
...
call hpcmw_couple_StoP_put

end subroutine structure_main
  
```

Fluid= MAIN
Structure is called from **Fluid** as a
subroutine through **Couper**.

FEM Codes on HPC-MW

- Primary Target: Evaluation of HPC-MW itself !
- Solid Mechanics
 - Elastic, Inelastic
 - Static, Dynamic
 - Various types of elements, boundary conditions.
- Eigenvalue Analysis
- Compressible/Incompressible CFD
- Heat Transfer with Radiation & Phase Change

Release in Late September 2003

- Library-Type HPC-MW
 - Parallel I/O
 - Original Data Structure, GeoFEM, ABAQUS
 - Parallel Visualization
 - PVR, PSR
 - Parallel Linear Solvers
 - Preconditioned Iterative Solvers (ILU, SAI)
 - Utility for Mesh Partitioning
 - Serial Partitioner, Viewer
 - On-line Tutorial
- FEM Code for Linear-Elastic Simulation (prototype)

Technical Issues

- Common Data Structure
 - Flexibility vs. Efficiency
 - Our data structure is efficient ...
 - How to keep user's own data structure

- Interface to Other Toolkits
 - PETSc (ANL), Aztec/Trillinos (Sandia)
 - ACTS Toolkit (LBNL/DOE)
 - DRAMA (NEC Europe), Zoltan (Sandia)

Strategy for Public Use

- General Purpose Parallel FEM Code
- Environment for Development (1): for Legacy Code
 - "Parallelization"
 - F77->F90, COMMON -> Module
 - Parallel Data Structure, Linear Solvers, Visualization
- Environment for Development (2): from Scratch
 - Education
- Various type of collaboration

On-going Collaboration

- Parallelization of Legacy Codes
 - CFD Grp. in FSIS project
 - Mitsubishi Material: Groundwater Flow
 - JNC (Japan Nuclear Cycle Development Inst.): HLW
 - others: research, educations.

- Part of HPC-MW
 - Coupling Interface for Pump Simulation
 - Parallel Visualization: Takashi Furumura (ERI/U.Tokyo)

On-going Collaboration

- Environment for Development
 - ACESS (Australian Computational Earth Systems Simulator) Group
- Research Collaboration
 - JAERI ITBL
 - DOE ACTS Toolkit (Lawrence Berkeley National Laboratory)
 - NEC Europe (Dynamic Load Balancing)
 - ACES GRID
 - APEC Cooperation for Earthquake Simulation

Further Study/Works

- Develop HPC-MW
 - Collaboration
 - Simplified I/F for Non-Experts
 - Interaction is important !!: Procedure for Collaboration

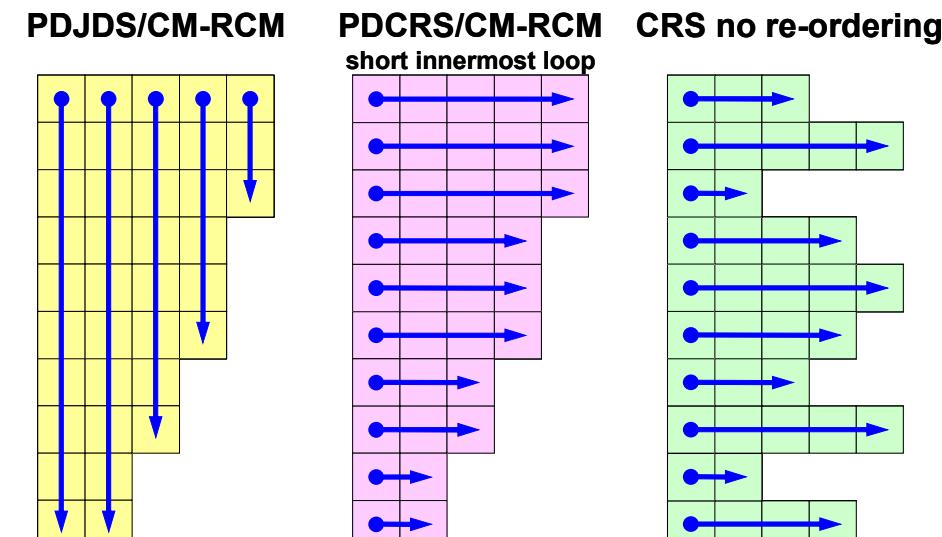
- Promotion for Public Use
 - Parallel FEM Applications
 - Parallelization of Legacy Codes
 - Environment for Development

Preliminary Study in FY.2002

- FEM Procedure for 3D Elastic Problem
 - Parallel I/O
 - Iterative Linear Solvers (ICCG, ILU-BiCGSTAB)
 - FEM Procedures (Matrix Connectivity/Assembling)
- Linear Solvers
 - SMP Cluster/Distributed Memory
 - Vector/Scalar Processors
 - CM-RCM/MC Reordering

Method of Matrix Storage

- Scalar/Distributed Memory
 - CRS with Natural Ordering
- Scalar/SMP Cluster
 - PDCRS/CM-RCM
 - PDCRS/MC
- Vector/Distributed & SMP Cluster
 - PDJDS/CM-RCM
 - PDJDS/MC



Main Program

```
program SOLVER33_TEST
use solver33
use hpcmw_all

implicit REAL*8 (A-H,O-Z)

call HPCMW_INIT
call INPUT_CNTL
call INPUT_GRID

call MAT_CON0
call MAT_CON1

call MAT_ASS_MAIN (valA,valB,valX)
call MAT_ASS_BC

call SOLVE33 (hpcmwIarray, hpcmwRarray)
call HPCMW_FINALIZE

end program SOLVER33_TEST
```

FEM program developed by users

- call same subroutines
- interfaces are same
- NO MPI !!

Procedure of each subroutine is different in the individual library

Mat.Ass. for SMP Cluster/Scalar

```

do ie= 1, 8
  ip = nodLOCAL(ie)
do je= 1, 8
  jp = nodLOCAL(je)

kk= 0
if (jp.gt.ip) then
  iiS= indexU(ip-1) + 1
  iiE= indexU(ip  )

  do k= iiS, iiE
    if ( itemU(k).eq.jp ) then
      kk= k
      exit
    endif
  enddo
endif

if (jp.lt.ip) then
  iiS= indexL(ip-1) + 1
  iiE= indexL(ip  )

  do k= iiS, iiE
    if ( itemL(k).eq.jp) then
      kk= k
      exit
    endif
  enddo
endif

PNXi= 0.d0
PNYi= 0.d0
PNZi= 0.d0
PNXj= 0.d0
PNYj= 0.d0
PNZj= 0.d0

VOL= 0.d0

```

```

do kpn= 1, 2
do jpn= 1, 2
do ipn= 1, 2
  coef= dabs(DETJ(ipn,jpn,kpn))*WEI(ipn)*WEI(jpn)*WEI(kpn)

  VOL= VOL + coef
  PNXi= PNX(ipn,jpn,kpn,ie)
  PNYi= PNY(ipn,jpn,kpn,ie)
  PNZi= PNZ(ipn,jpn,kpn,ie)

  PNXj= PNX(ipn,jpn,kpn,je)
  PNYj= PNY(ipn,jpn,kpn,je)
  PNZj= PNZ(ipn,jpn,kpn,je)

  a11= valX*(PNXi*PNXj+valB*(PNYi*PNYj+PNZi*PNZj))*coef
  a22= valX*(PNYi*PNYj+valB*(PNZi*PNZj+PNXi*PNXj))*coef
  a33= valX*(PNZi*PNZj+valB*(PNXi*PNXj+PNYi*PNYj))*coef

  a12= (valA*PNXi*PNYj + valB*PNXj*PNYi)*coef
  a13= (valA*PNXi*PNZj + valB*PNXj*PNZi)*coef
  ...

  if (jp.gt.ip) then
    PAU(9*kk-8)= PAU(9*kk-8) + a11
    ...
    PAU(9*kk  )= PAU(9*kk  ) + a33
  endif

  if (jp.lt.ip) then
    PAL(9*kk-8)= PAL(9*kk-8) + a11
    ...
    PAL(9*kk  )= PAL(9*kk  ) + a33
  endif

  if (jp.eq.ip) then
    D(9*ip-8)= D(9*ip-8) + a11
    ...
    D(9*ip  )= D(9*ip  ) + a33
  endif
enddo
enddo
enddo

enddo
endif
enddo
enddo
enddo

```

Mat.Ass. for SMP Cluster/Vector

```

do ie= 1, 8
  ip = nodLOCAL(ie)
if (ip.le.N) then
do je= 1, 8
  jp = nodLOCAL(je)
  kk= 0
  if (jp.gt.ip) then
    ipU= OLDTtoNEW_U(ip)
    jpU= OLDTtoNEW_U(jp)
    kp= PEon(ipU)
    iv= COLORon(ipU)
    nn= ipU - STACKmc((iv-1)*PEsmpTOT+kp-1)

  do k= 1, NUmaxHYP(iv)
    iS= indexU(npUX1*(iv-1)+PEsmpTOT*(k-1)+kp-1) + nn
    if ( itemU(iS).eq.jpU) then
      kk= iS
      exit
    endif
  enddo
endif

  if (jp.lt.ip) then
    ipL= OLDTtoNEW_L(ip)
    jpL= OLDTtoNEW_L(jp)
    kp= PEon(ipL)
    iv= COLORon(ipL)
    nn= ipL - STACKmc((iv-1)*PEsmpTOT+kp-1)

  do k= 1, NLmaxHYP(iv)
    iS= indexL(npLX1*(iv-1)+PEsmpTOT*(k-1)+kp-1) + nn
    if ( itemL(iS).eq.jpL) then
      kk= iS
      exit
    endif
  enddo
endif

  PNXi= 0.d0
  PNYi= 0.d0
  PNZi= 0.d0
  PNXj= 0.d0
  PNYj= 0.d0
  PNZj= 0.d0

  VOL= 0.d0

```

```

  do kpn= 1, 2
  do jpn= 1, 2
  do ipn= 1, 2
    coef= dabs(DETJ(ipn,jpn,kpn))*WEI(ipn)*WEI(jpn)*WEI(kpn)

    VOL= VOL + coef
    PNXi= PNX(ipn,jpn,kpn,ie)
    PNYi= PNY(ipn,jpn,kpn,ie)
    PNZi= PNZ(ipn,jpn,kpn,ie)

    PNXj= PNX(ipn,jpn,kpn,je)
    PNYj= PNY(ipn,jpn,kpn,je)
    PNZj= PNZ(ipn,jpn,kpn,je)

    a11= valX*(PNXi*PNXj+valB*(PNYi*PNYj+PNZi*PNZj))*coef
    a22= valX*(PNYi*PNYj+valB*(PNZi*PNZj+PNXi*PNXj))*coef
    a33= valX*(PNZi*PNZj+valB*(PNXi*PNXj+PNYi*PNYj))*coef

    a12= (valA*PNXi*PNYj + valB*PNXj*PNYi)*coef
    a13= (valA*PNXi*PNZj + valB*PNXj*PNZi)*coef
    ...

    if (jp.gt.ip) then
      PAU(9*kk-8)= PAU(9*kk-8) + a11
      ...
      PAU(9*kk )= PAU(9*kk ) + a33
    endif

    if (jp.lt.ip) then
      PAL(9*kk-8)= PAL(9*kk-8) + a11
      ...
      PAL(9*kk )= PAL(9*kk ) + a33
    endif

    if (jp.eq.ip) then
      D(9*ip-8)= D(9*ip-8) + a11
      ...
      D(9*ip )= D(9*ip ) + a33
    endif
  enddo
  enddo
  enddo

  enddo
  endif
  enddo
enddo

```

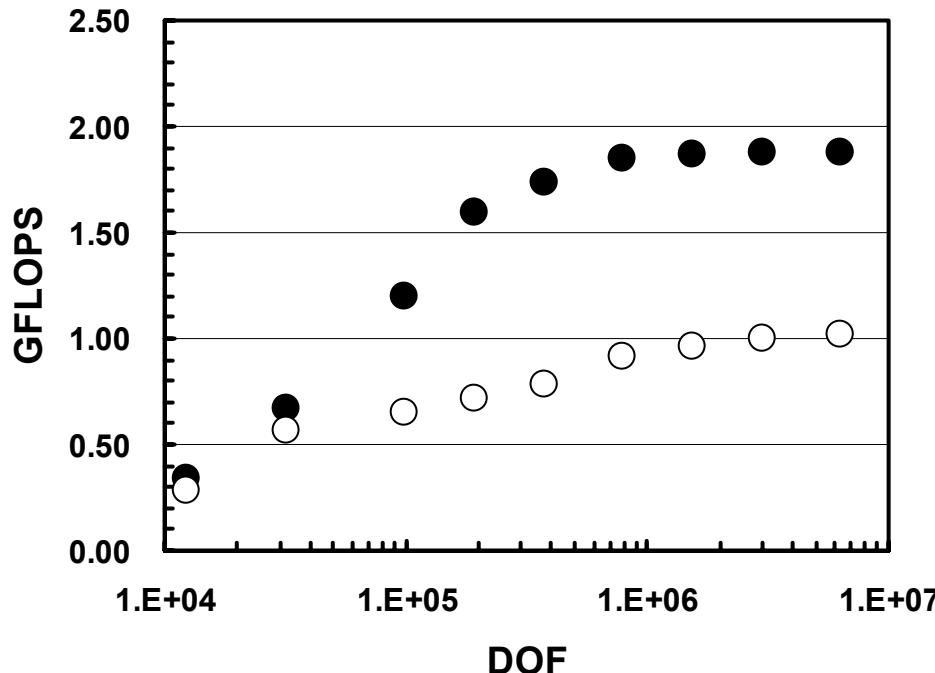
Hardware

- Hitachi SR8000/128
 - SMP Cluster
 - 8 PE/node
 - Pseudo-Vector
- Xeon 2.8 GHz Cluster
 - 2 PE/node, Myrinet
 - Flat MPI only
- Hitachi SR2201
 - Pseudo-Vector
 - Flat MPI

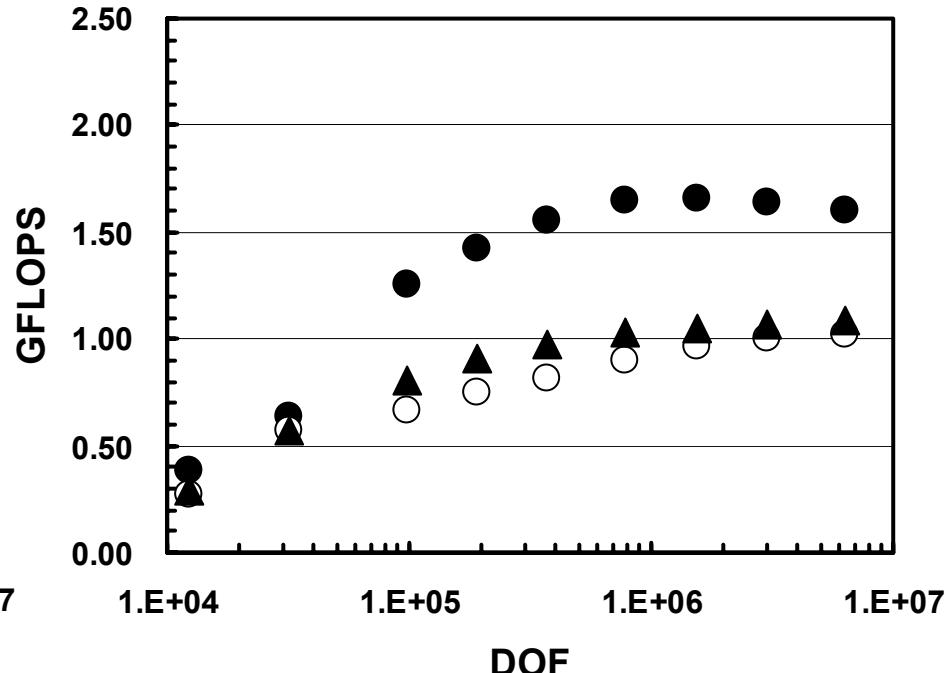
Hitachi SR8000/128

8 PEs/1-SMP node

SMP



Flat-MPI

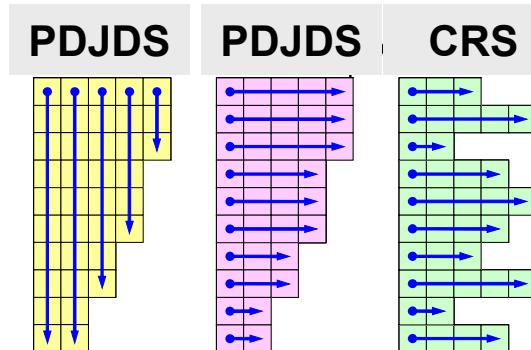


● : PDJDS, ○ : PDCRS, ▲ : CRS-Natural

Xeon & SR2201

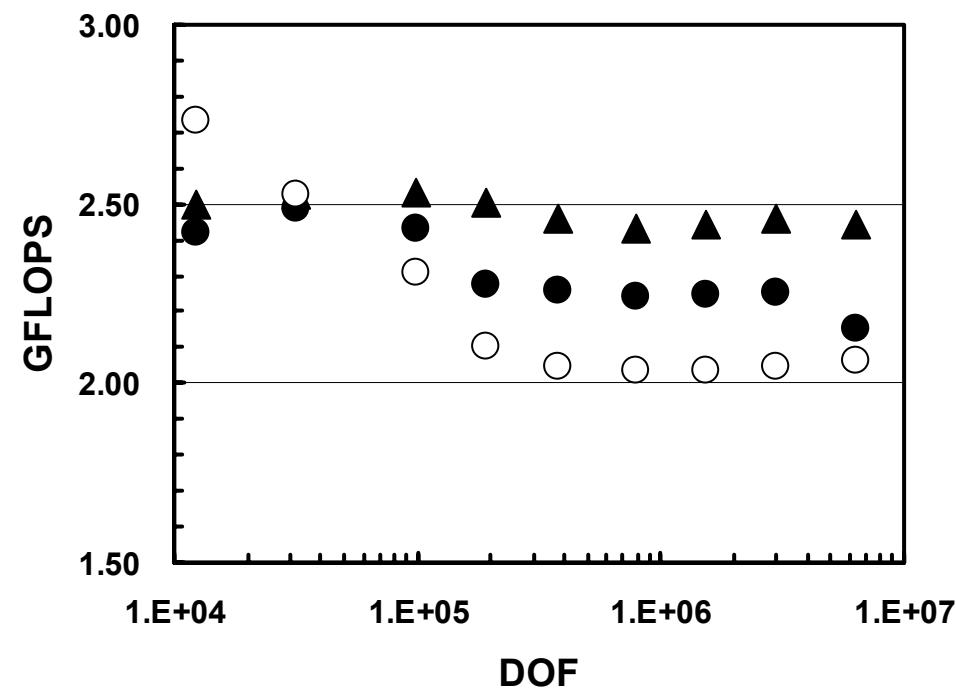
8 PEs

● : PDJDS, ○ : PDCRS, ▲ : CRS-Natural



Not so significant
reduce due to
pseudo-vector.

Xeon



SR2201

