Introduction
Overview of the Class

Kengo Nakajima
Information Technology Center

Programming for Parallel Computing (616-2057)
Seminar on Advanced Computing (616-4009)

- Target: Parallel FEM
- Supercomputers and Computational Science
- Overview of the Class
- Future Issues

Programming for Parallel Computing
Seminar on Advanced Computing
並列計算プログラミング・先端計算機演習

- Instructor
  - Kengo Nakajima
  - Professor, Information Technology Center, The University of Tokyo
- Topics
  - Finite-Element Method (FEM)
  - Parallel FEM using MPI and OpenMP

This 9-day intensive class provides introduction to large-scale scientific computing using the most advanced massively parallel supercomputers. Topics cover:

- Finite-Element Method (FEM)
- Message Passing Interface (MPI)
- Parallel FEM using MPI and OpenMP
- Parallel Numerical Algorithms for Iterative Linear Solvers

Several sample programs will be provided and participants can review the contents of lectures through hands-on-exercise/practices using Fujitsu PRIMEHPC FX10 at the University of Tokyo (Oakleaf-FX).

Finite-Element Method is widely-used for solving various types of real-world scientific and engineering problems, such as structural analysis, fluid dynamics, electromagnetics, and etc. This lecture course provides brief introduction to procedures of FEM for 1D/3D steady-state heat

conduction problems with iterative linear solvers and to parallel FEM. **Lectures for parallel FEM will be focused on design of data structure for distributed local mesh files, which is the key issue for efficient parallel FEM.** Introduction to MPI (Message Passing Interface), which is widely used method as "de facto standard" of parallel programming, is also provided.

Solving large-scale linear equations with sparse coefficient matrices is the most expensive and important part of FEM and other methods for scientific computing, such as Finite-Difference Method (FDM) and Finite-Volume Method (FVM). Recently, families of Krylov iterative solvers are widely used for this process. In this class, details of implementations of parallel Krylov iterative methods are provided along with parallel FEM.

Moreover, lectures on programming for multicore architectures will be also given along with brief introduction to OpenMP and OpenMP/MPI Hybrid Parallel Programming Model.

Motivation for Parallel Computing (and this class)

- Large-scale parallel computer enables fast computing in large-scale scientific simulations with detailed models. Computational science develops new frontiers of science and engineering.
- Why parallel computing ?
  - faster & larger
  - “larger” is more important from the view point of “new frontiers of science & engineering”, but “faster” is also important.
  - + more complicated
  - Ideal: Scalable
    - Weak Scaling, Strong Scaling

Scalable, Scaling, Scalability

- Solving N^x scale problem using N^x computational resources during same computation time
  - for large-scale problems: **Weak Scaling, Weak Scalability**
  - e.g. CG solver: more iterations needed for larger problems
- Solving a problem using N^x computational resources during 1/N computation time
  - for faster computation: **Strong Scaling, Strong Scalability**

Kengo Nakajima (1/2)

- Current Position
  - Professor, Supercomputing Research Division, Information Technology Center　情報基盤センター
  - Professor, Department of Mathematical Informatics, Graduate School of Information Science & Engineering　数理情報学専攻
  - Professor, Department of Electrical Engineering & Information Systems, Graduate School of Engineering 電気系工学専攻
  - Visiting Senior Researcher, Advanced Institute for Computational Science (AICS), RIKEN
- Research Interest
  - High-Performance Computing
  - Parallel Numerical Linear Algebra (Preconditioning)
  - Parallel Programming Model
  - Computational Mechanics, Computational Fluid Dynamics
  - Adaptive Mesh Refinement, Parallel Visualization

Kengo Nakajima (2/2)

- Education
  - B.Eng (Aeronautics, The University of Tokyo, 1985)
  - M.S. (Aerospace Engineering, University of Texas, 1993)
  - Ph.D. (Quantum Engineering & System Sciences, The University of Tokyo, 2003)
- Professional Background
  - Mitsubishi Research Institute, Inc. (1985-1999)
  - Research Organization for Information Science & Technology (1999-2004)
  - The University of Tokyo
    - Department Earth & Planetary Science (2004-2008)
    - Information Technology Center (2008-)
  - JAMSTEC (2008-2011), part-time
  - RIKEN (2009-), part-time

Scientific Computing = SMASH

Science

Modeling

Algorithm

Software

Hardware

- You have to learn many things.
- Collaboration (or Co-Design) will be important for future career of each of you, as a scientist and/or an engineer.
  - You have to communicate with people with different backgrounds.
  - It is more difficult than communicating with foreign scientists from same area.
- (Q): Your Department ?

This Class ...

Science

Modeling

Algorithm

Software

Hardware

- Parallel FEM using MPI and OpenMP
- Science: Heat Conduction
- Modeling: FEM
- Algorithm: Iterative Solvers etc.
- You have to know many components to learn FEM, although you have already learned each of these in undergraduate and high-school classes.

Road to Programming for "Parallel" Scientific Computing

Programming for Parallel Scientific Computing (e.g. Parallel FEM/FDM)

Programming for Real World Scientific Computing (e.g. FEM, FDM)

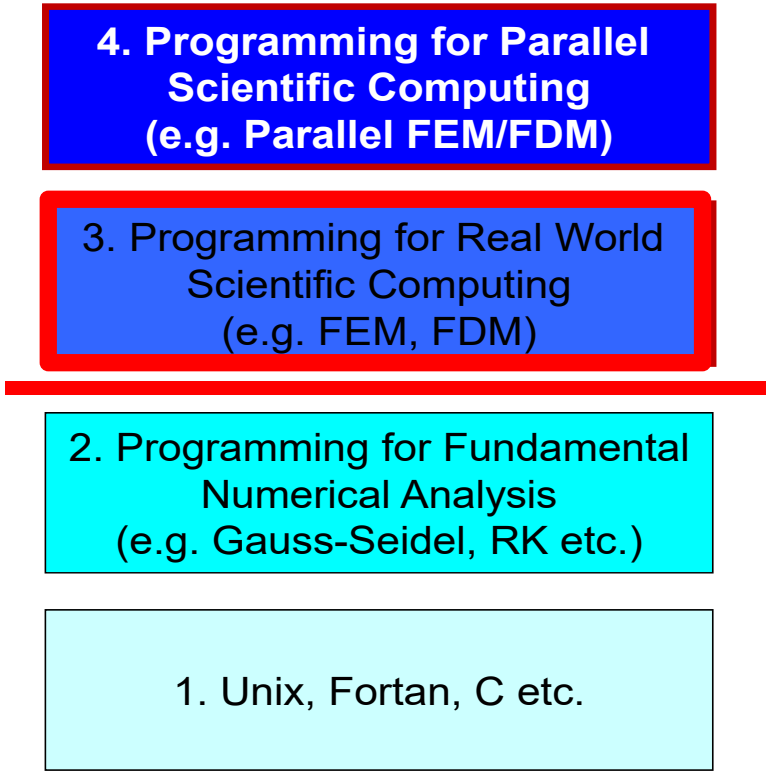
Big gap here !!

Programming for Fundamental Numerical Analysis (e.g. Gauss-Seidel, RK etc.)

Unix, Fortan, C etc.

The third step is important !

- How to parallelize applications ?
  - How to extract parallelism ?
  - If you understand methods, algorithms, and implementations of the original code, it's easy.
  - "Data-structure" is important



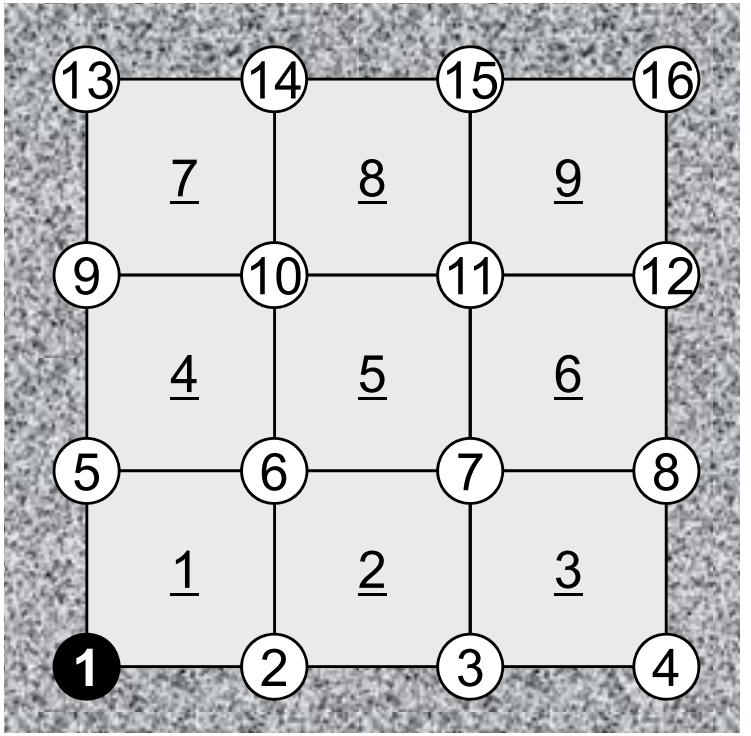
- How to understand the code ?
  - Reading the application code !!
  - It seems primitive, but very effective.
  - In this class, "reading the source code" is encouraged.

Finite-Element Method (FEM)

- One of the most popular numerical methods for solving PDE's.
  - elements (meshes) & nodes (vertices)
- Consider the following 2D heat transfer problem:

$$\lambda\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}\right)+Q=0$$

  - 16 nodes, 9 bi-linear elements
  - uniform thermal conductivity (λ=1)
  - uniform volume heat flux (Q=1)
  - T=0 at node 1
  - Insulated boundaries



Galerkin FEM procedures

- Apply Galerkin procedures to each element:

$$\int_V [N]^T \left\{ \lambda \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) + Q \right\} dV = 0$$

where $T = [N]\{\phi\}$ in each elem.

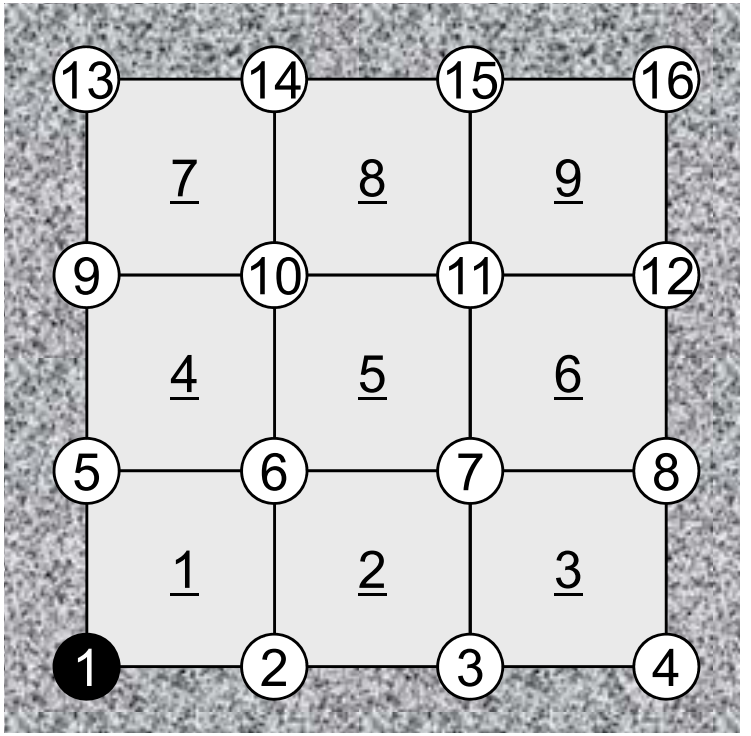
$\{\phi\}$: T at each vertex

$[N]$: Shape function (Interpolation function)

- Introduce the following "weak form" of original PDE using Green's theorem:

$$-\int_V \lambda \left(\frac{\partial [N]^T}{\partial x} \frac{\partial [N]}{\partial x} + \frac{\partial [N]^T}{\partial y} \frac{\partial [N]}{\partial y} \right) dV \cdot \{\phi\}$$

$$+\int_V Q[N]^T dV = 0$$

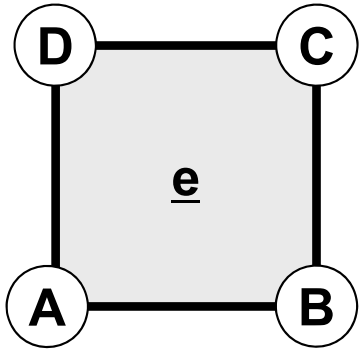


Element Matrix

- Apply the integration to each element and form "element" matrix.

$$-\int_V \lambda\left(\frac{\partial[N]^T}{\partial x}\frac{\partial[N]}{\partial x}+\frac{\partial[N]^T}{\partial y}\frac{\partial[N]}{\partial y}\right)dV\cdot\{\phi\}$$

$$+\int_V Q[N]^T dV = 0$$

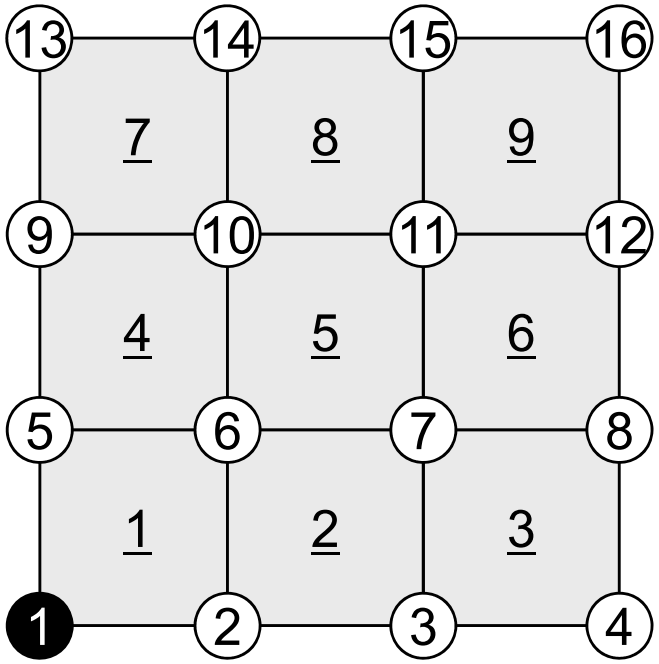


$$[k^{(e)}]\{\phi^{(e)}\} = \{f^{(e)}\}$$

$$\begin{bmatrix} k_{AA}^{(e)} & k_{AB}^{(e)} & k_{AC}^{(e)} & k_{AD}^{(e)} \\ k_{BA}^{(e)} & k_{BB}^{(e)} & k_{BC}^{(e)} & k_{BD}^{(e)} \\ k_{CA}^{(e)} & k_{CB}^{(e)} & k_{CC}^{(e)} & k_{CD}^{(e)} \\ k_{DA}^{(e)} & k_{DB}^{(e)} & k_{DC}^{(e)} & k_{DD}^{(e)} \end{bmatrix} \begin{Bmatrix} \phi_A^{(e)} \\ \phi_B^{(e)} \\ \phi_C^{(e)} \\ \phi_D^{(e)} \end{Bmatrix} = \begin{Bmatrix} f_A^{(e)} \\ f_B^{(e)} \\ f_C^{(e)} \\ f_D^{(e)} \end{Bmatrix}$$

Global (Overall) Matrix

Accumulate each element matrix to “global” matrix.

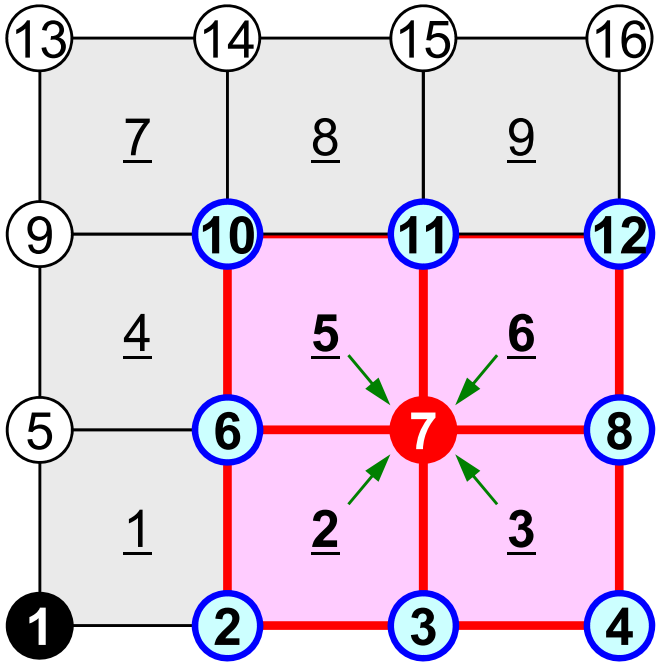


$$[K]\{\Phi\} = \{F\}$$

$$\begin{bmatrix}
D & X & & & X & X & & & & & & & & & & \\
X & D & X & & X & X & X & & & & & & & & & \\
 & X & D & X & & X & X & X & & & & & & & & \\
 & & X & D & & & X & X & & & & & & & & \\
X & X & & & D & X & & & X & X & & & & & & \\
X & X & X & & X & D & X & & X & X & X & & & & & \\
 & X & X & X & & X & D & X & & X & X & X & & & & \\
 & & X & X & & & X & D & & & X & X & & & & \\
 & & & & X & X & & & D & X & & & X & X & & \\
 & & & & X & X & X & & X & D & X & & X & X & X & \\
 & & & & & X & X & X & & X & D & X & & X & X & X \\
 & & & & & & X & X & & & X & D & & & X & X \\
 & & & & & & & & X & X & & & D & X & & \\
 & & & & & & & & X & X & X & & X & D & X & \\
 & & & & & & & & & X & X & X & & X & D & X \\
 & & & & & & & & & & X & X & & & X & D
\end{bmatrix}
\begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \\ \Phi_6 \\ \Phi_7 \\ \Phi_8 \\ \Phi_9 \\ \Phi_{10} \\ \Phi_{11} \\ \Phi_{12} \\ \Phi_{13} \\ \Phi_{14} \\ \Phi_{15} \\ \Phi_{16} \end{Bmatrix}
=
\begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \\ F_7 \\ F_8 \\ F_9 \\ F_{10} \\ F_{11} \\ F_{12} \\ F_{13} \\ F_{14} \\ F_{15} \\ F_{16} \end{Bmatrix}$$

To each node …

Effect of surrounding elem's/nodes are accumulated.

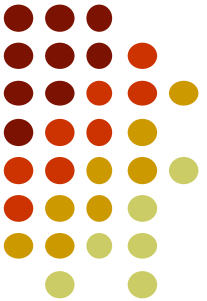


$$[K]\{\Phi\} = \{F\}$$

$$\begin{bmatrix}
D & X & & & X & X & & & & & & & & & & \\
X & D & X & & X & X & X & & & & & & & & & \\
 & X & D & X & & X & X & X & & & & & & & & \\
 & & X & D & & & X & X & & & & & & & & \\
X & X & & & D & X & & & X & X & & & & & & \\
X & X & X & & X & D & X & & X & X & X & & & & & \\
 & X & X & X & & X & D & X & & X & X & X & & & & \\
 & & X & X & & & X & D & & & X & X & & & & \\
 & & & & X & X & & & D & X & & & X & X & & \\
 & & & & X & X & X & & X & D & X & & X & X & X & \\
 & & & & & X & X & X & & X & D & X & & X & X & X \\
 & & & & & & X & X & & & X & D & & & X & X \\
 & & & & & & & & X & X & & & D & X & & \\
 & & & & & & & & X & X & X & & X & D & X & \\
 & & & & & & & & & X & X & X & & X & D & X \\
 & & & & & & & & & & X & X & & & X & D
\end{bmatrix}
\begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \\ \Phi_6 \\ \Phi_7 \\ \Phi_8 \\ \Phi_9 \\ \Phi_{10} \\ \Phi_{11} \\ \Phi_{12} \\ \Phi_{13} \\ \Phi_{14} \\ \Phi_{15} \\ \Phi_{16} \end{Bmatrix}
=
\begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \\ F_7 \\ F_8 \\ F_9 \\ F_{10} \\ F_{11} \\ F_{12} \\ F_{13} \\ F_{14} \\ F_{15} \\ F_{16} \end{Bmatrix}$$

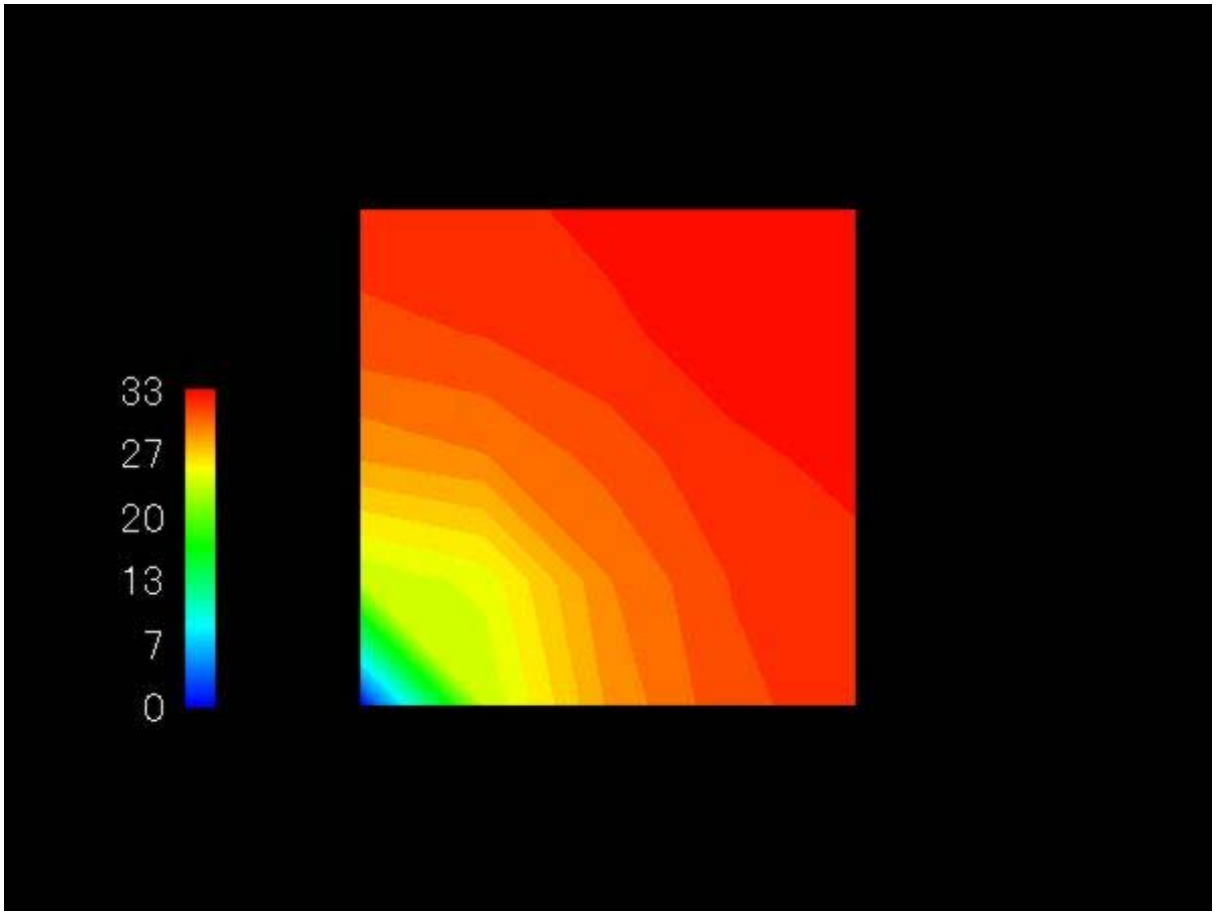
Solve the obtained global eqn's

under certain boundary conditions
(Φ_1=0 in this case)

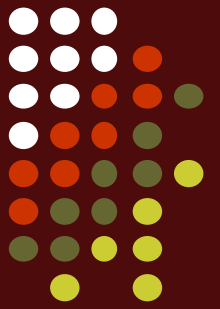


$$
\begin{bmatrix}
D & X & & & X & X & & & & & & & & & & \\
X & D & X & & X & X & X & & & & & & & & & \\
 & X & D & X & & X & X & X & & & & & & & & \\
 & & X & D & & & X & X & & & & & & & & \\
X & X & & & D & X & & & X & X & & & & & & \\
X & X & X & & X & D & X & & X & X & X & & & & & \\
 & X & X & X & & X & D & X & & X & X & X & & & & \\
 & & X & X & & & X & D & & & X & X & & & & \\
 & & & & X & X & & & D & X & & & X & X & & \\
 & & & & X & X & X & & X & D & X & & X & X & X & \\
 & & & & & X & X & X & & X & D & X & & X & X & X \\
 & & & & & & X & X & & & X & D & & & X & X \\
 & & & & & & & & X & X & & & D & X & & \\
 & & & & & & & & X & X & X & & X & D & X & \\
 & & & & & & & & & X & X & X & & X & D & X \\
 & & & & & & & & & & X & X & & & X & D
\end{bmatrix}
\begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \\ \Phi_6 \\ \Phi_7 \\ \Phi_8 \\ \Phi_9 \\ \Phi_{10} \\ \Phi_{11} \\ \Phi_{12} \\ \Phi_{13} \\ \Phi_{14} \\ \Phi_{15} \\ \Phi_{16} \end{Bmatrix}
=
\begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \\ F_7 \\ F_8 \\ F_9 \\ F_{10} \\ F_{11} \\ F_{12} \\ F_{13} \\ F_{14} \\ F_{15} \\ F_{16} \end{Bmatrix}
$$

Result …



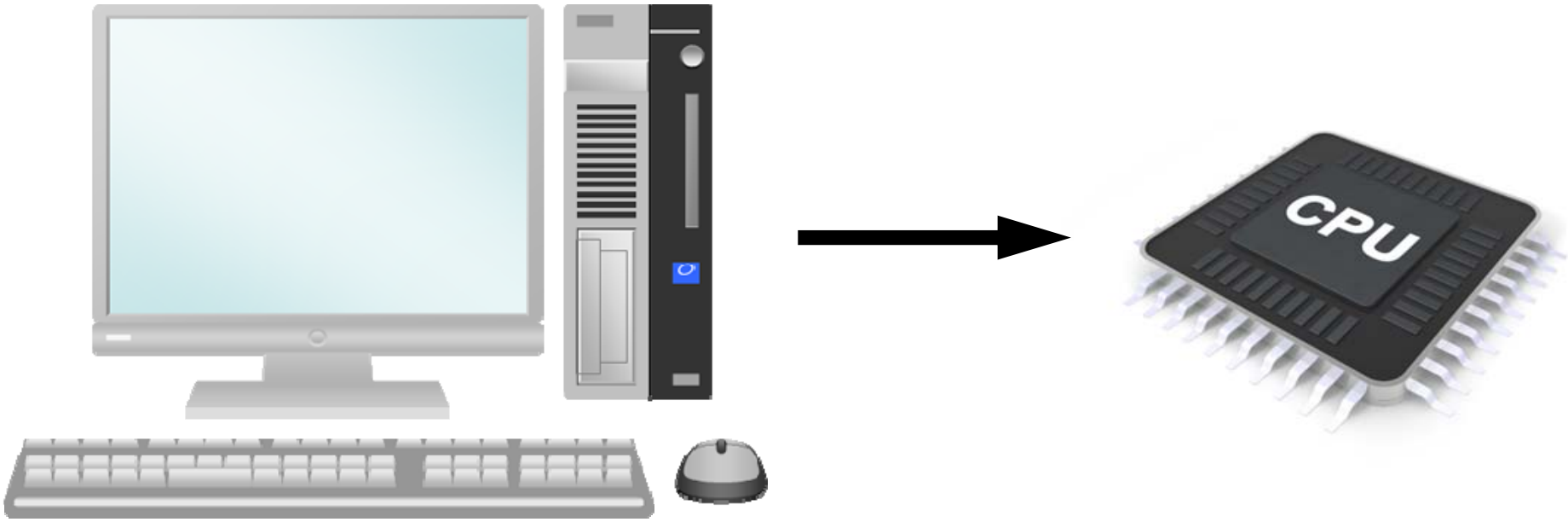
Features of FEM applications



- Typical Procedures for FEM Computations
  - Input/Output
  - Matrix Assembling
  - Linear Solvers for Large-scale Sparse Matrices
  - Most of the computation time is spent for matrix assembling/formation and solving linear equations.
- **HUGE** “indirect” accesses
  - memory intensive
- Local “element-by-element” operations
  - sparse coefficient matrices
  - suitable for parallel computing
- Excellent modularity of each procedure

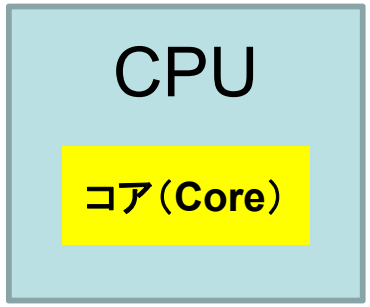
- Target: Parallel FEM
- **Supercomputers and Computational Science**
- Overview of the Class
- Future Issues

Computer & CPU

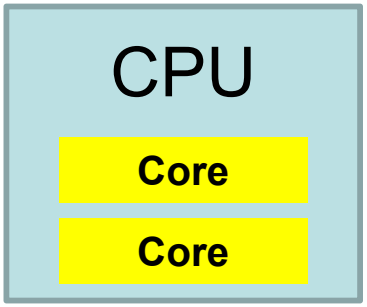


- Central Processing Unit（中央処理装置）:CPU
- CPU’s used in PC and Supercomputers are based on same architecture
- GHz: Clock Rate
  - Frequency: Number of operations by CPU per second
    - GHz -> 10^9 operations/sec
  - Simultaneous 4-8 instructions per clock

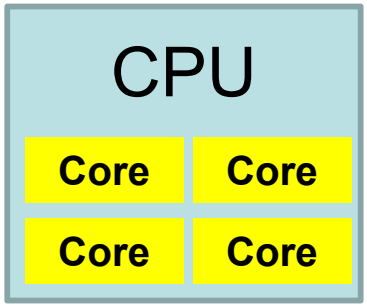
Multicore CPU



Single Core
1 cores/CPU

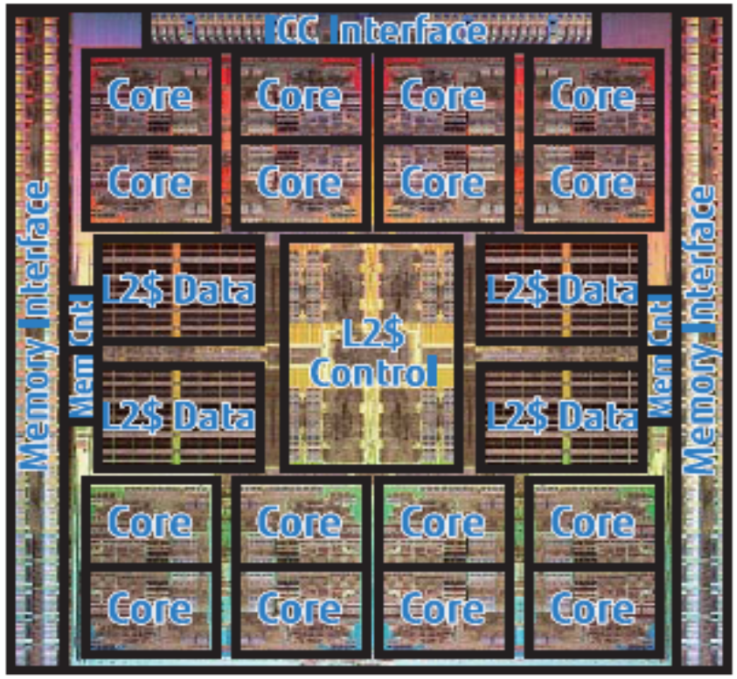


Dual Core
2 cores/CPU



Quad Core
4 cores/CPU

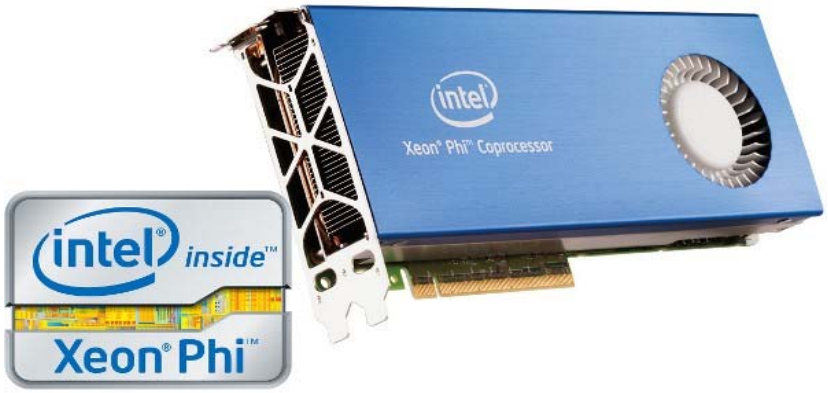
- Core= Central part of CPU
- Multicore CPU's with 4-8 cores are popular



- GPU: Manycore
  - $O(10^1)$-$O(10^2)$ cores
- More and more cores
  - Parallel computing
- Oakleaf-FX at University of Tokyo: 16 cores
  - SPARC64™ IXfx

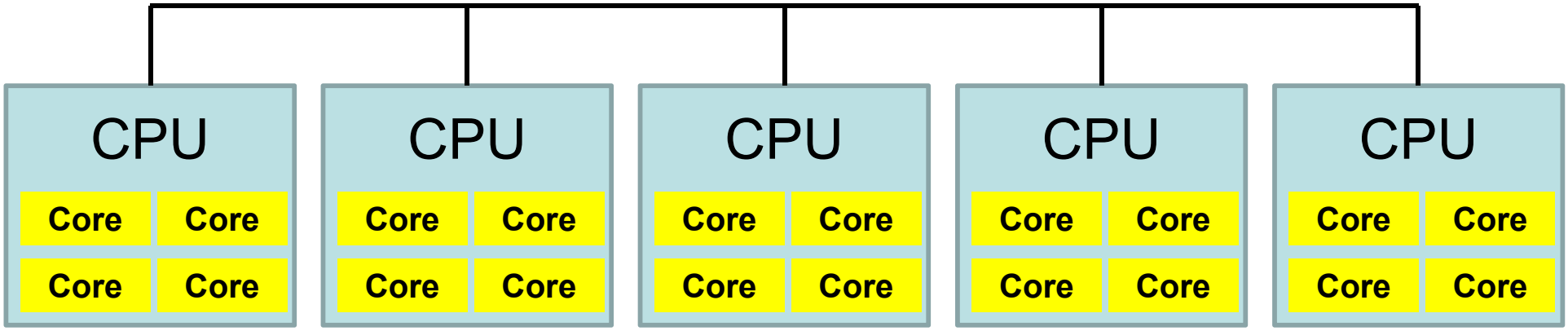
GPU/Manycores

- GPU : Graphic Processing Unit
  - GPGPU: General Purpose GPU
  - $O(10^2)$ cores
  - High Memory Bandwidth
  - Cheap
  - NO stand-alone operations
    - Host CPU needed
  - Programming: CUDA, OpenACC
- Intel Xeon/Phi: Manycore CPU
  - 60 cores
  - High Memory Bandwidth
  - Unix, Fortran, C compiler
  - Currently, host CPU needed
    - Stand-alone will be possible soon

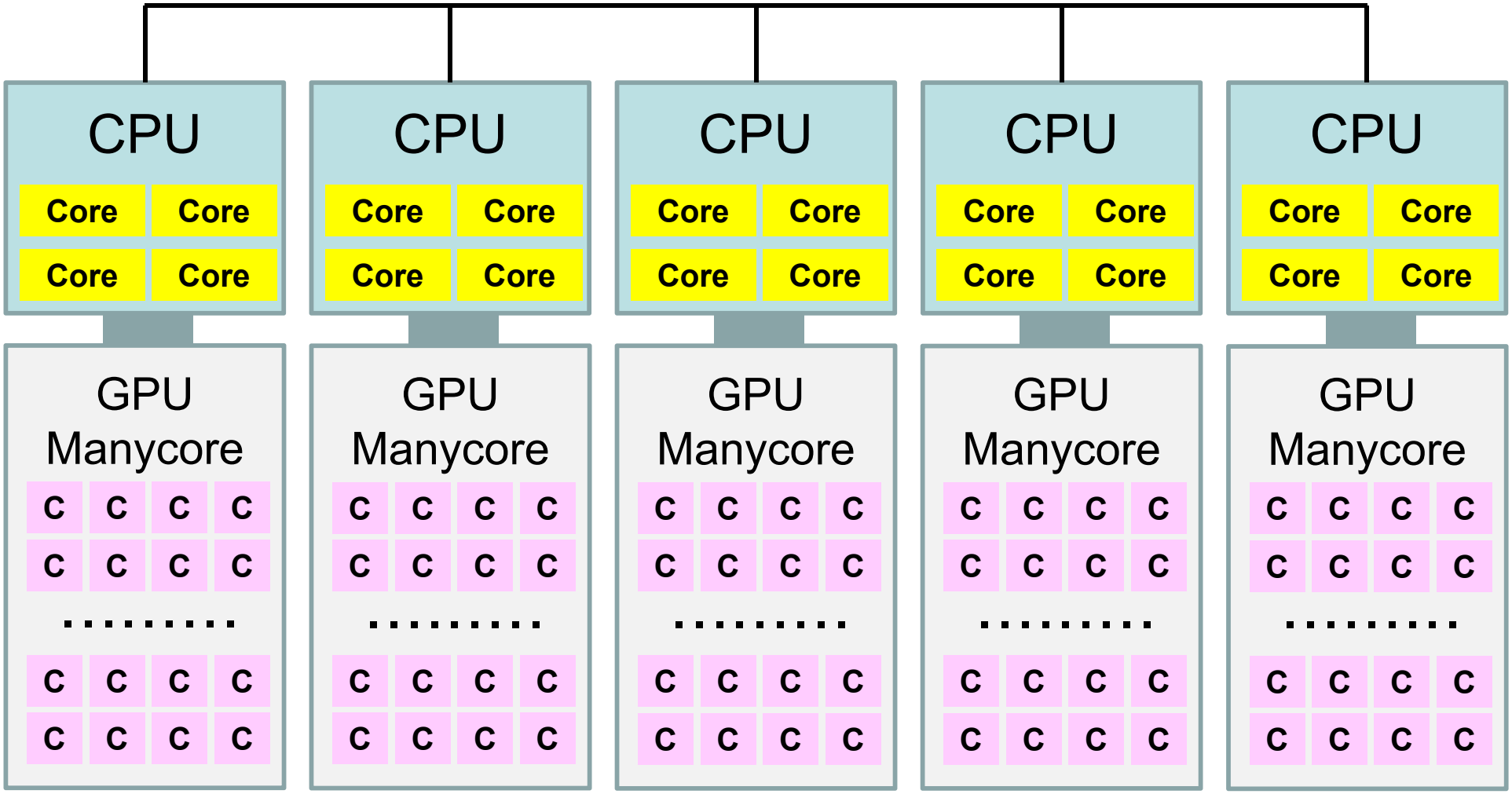


Parallel Supercomputers

Multicore CPU's are connected through network



Supercomputers with Heterogeneous/Hybrid Nodes



Performance of Supercomputers

- Performance of CPU: Clock Rate
- FLOPS (Floating Point Operations per Second)
  - Real Number
- Recent Multicore CPU
  - 4-8 FLOPS per Clock
  - (e.g.) Peak performance of a core with 3GHz
    - $3 \times 10^9 \times 4(\text{or } 8) = 12(\text{or } 24) \times 10^9$ FLOPS=12(or 24)GFLOPS

    - 10^6 FLOPS= 1 Mega FLOPS = 1 MFLOPS
    - 10^9 FLOPS= 1 Giga FLOPS = 1 GFLOPS
    - 10^{12} FLOPS= 1 Tera FLOPS = 1 TFLOPS
    - 10^{15} FLOPS= 1 Peta FLOPS = 1 PFLOPS
    - 10^{18} FLOPS= 1 Exa FLOPS = 1 EFLOPS

Peak Performance of Oakleaf-FX

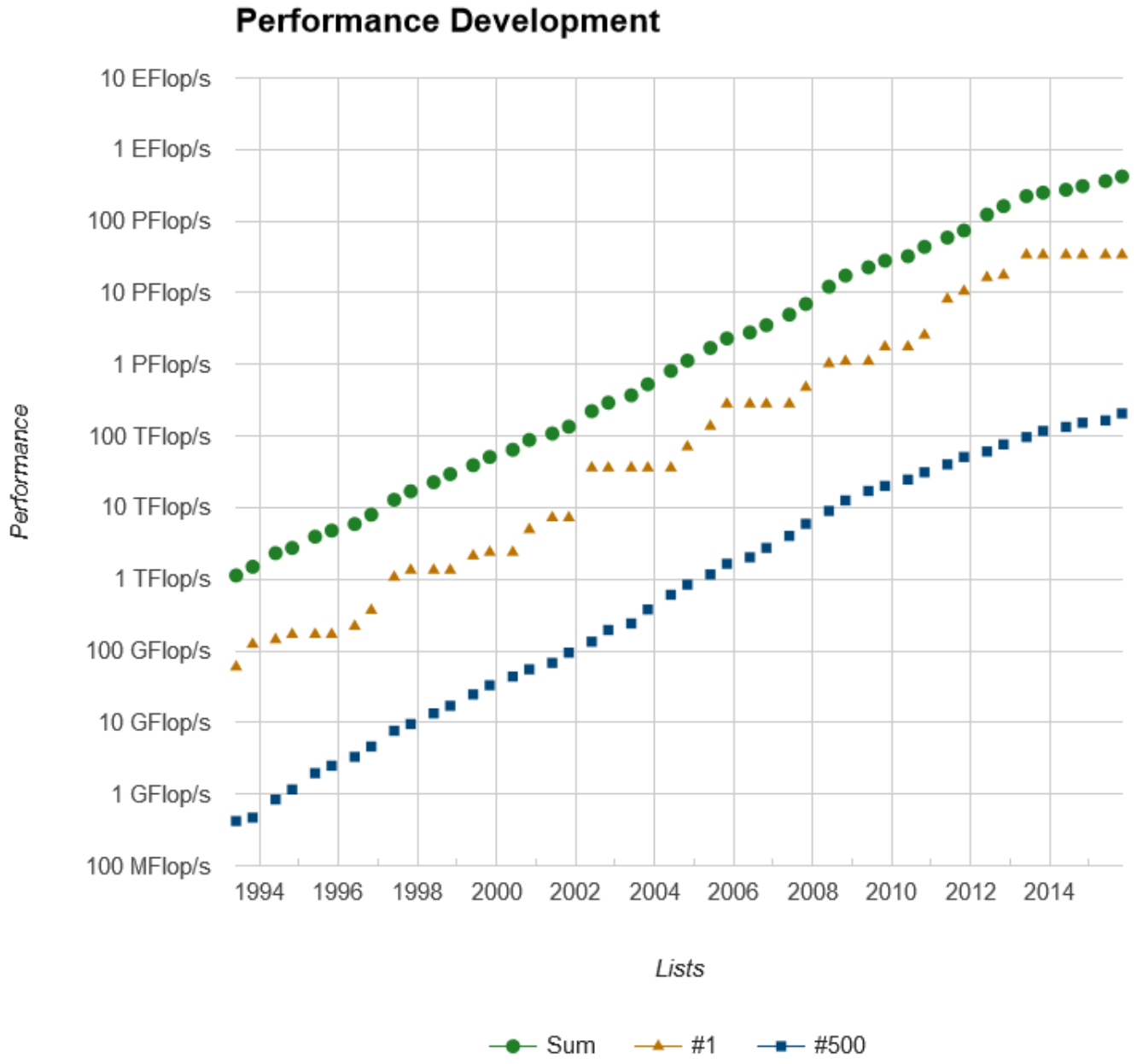
Fujitsu PRIMEHPC FX10 at U.Tokyo

- 1.848 GHz
- 8 FLOP operations per Clock
- Peak Performance (1 core)
  - 1.848 × 8= 14.78 GFLOPS
- Peak Performance (1 node/16 cores)
  - 236.5 GFLOPS
- Peak Performance of Entire Performance
  - 4,800 nodes, 76,800 cores
  - 1.13 PFLOPS

TOP 500 List

http://www.top500.org/

- Ranking list of supercomputers in the world
- Performance (FLOPS rate) is measured by "Linpack" which solves large-scale linear equations.
  - Since 1993
  - Updated twice a year (International Conferences in June and November)
- Linpack
  - iPhone version is available



- PFLOPS: Peta (=10^{15}) Floating OPerations per Sec.
- Exa-FLOPS (=10^{18}) will be attained after 2023 …

http://www.top500.org/

47th TOP500 List (June, 2016)

	Site	Computer/Year Vendor	Cores	R_{max}	R_{peak}	Power
1	National Supercomputing Center in Wuxi, China	Sunway TaihuLight - Sunway MPP, Sunway SW26010 260C 1.45GHz, 2016 NRCPC	10649600	93015 (= 93.0 PF)	125436	15371
2	National Supercomputing Center in Tianjin, China	**Tianhe-2** Intel Xeon E5-2692, TH Express-2, Xeon Phi, 2013 NUDT	3120000	33863 (= 33.9 PF)	54902	17808
3	Oak Ridge National Laboratory, USA	**Titan** Cray XK7/NVIDIA K20x, 2012 Cray	560640	17590	27113	8209
4	Lawrence Livermore National Laboratory, USA	**Sequoia** BlueGene/Q, 2011 IBM	1572864	17173	20133	7890
5	RIKEN AICS, Japan	K computer, SPARC64 VIIIfx , 2011 Fujitsu	705024	10510	11280	12660
6	Argonne National Laboratory, USA	**Mira** BlueGene/Q, 2012 IBM	786432	8587	10066	3945
7	DOE/NNSA/LANL/SNL, USA	Trinity - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, 2016 Cray	301056	8101	11079	3945
8	Swiss Natl. Supercomputer Center, Switzerland	**Piz Daint** Cray XC30/NVIDIA K20x, 2013 Cray	115984	6271	7789	2325
9	HLRS-Stuttgart, Germany	Hazel Hen - Cray XC40, Xeon E5-2680v3 12C 2.5GHz, 2016 Cray	185088	5640	7404	-
10	KAUST, Saudi Arabia	**Shaheen II** Cray XC40, Xeon E5-2698, 2015 Cray	196608	5537	7235	2834

R_{max}: Performance of Linpack (TFLOPS)
R_{peak}: Peak Performance (TFLOPS), Power: kW

http://www.top500.org/

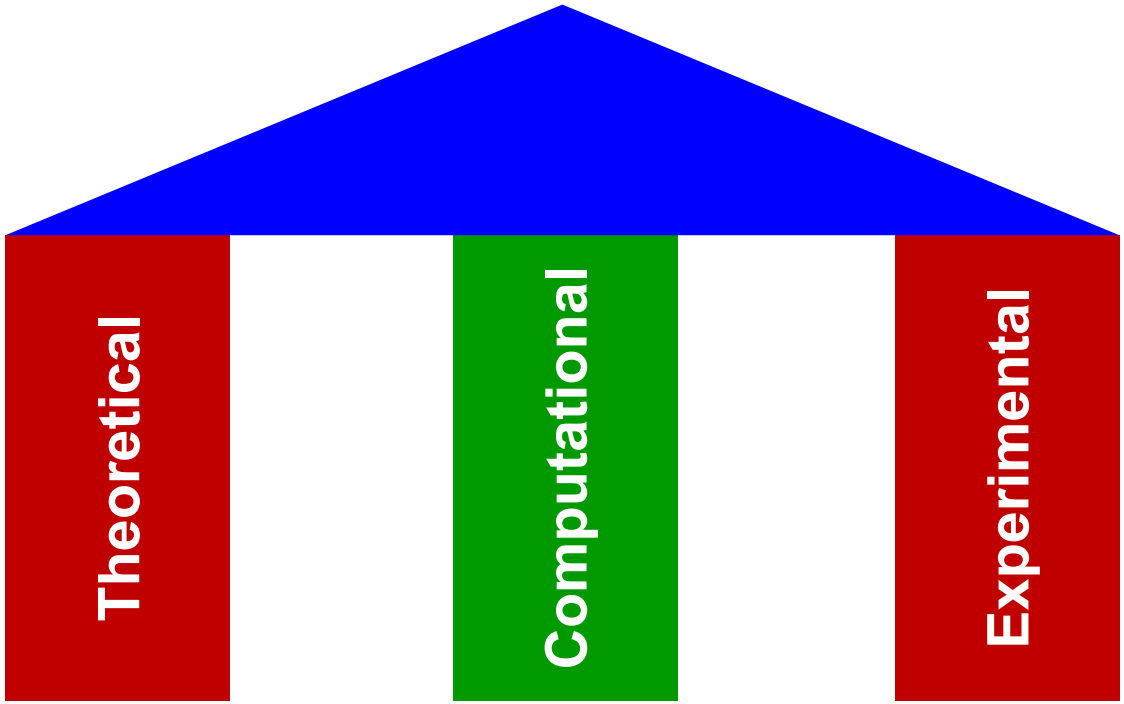
47th TOP500 List (June, 2016)

	Site	Computer/Year Vendor	Cores	R_{max}	R_{peak}	Power
1	National Supercomputing Center in Wuxi, China	Sunway TaihuLight - Sunway MPP, Sunway SW26010 260C 1.45GHz, 2016 NRCPC	10649600	93015 (= 93.0 PF)	125436	15371
2	National Supercomputing Center in Tianjin, China	**Tianhe-2** Intel Xeon E5-2692, TH Express-2, Xeon Phi, 2013 NUDT	3120000	33863 (= 33.9 PF)	54902	17808
3	Oak Ridge National Laboratory, USA	**Titan** Cray XK7/NVIDIA K20x, 2012 Cray	560640	17590	27113	8209
4	Lawrence Livermore National Laboratory, USA	**Sequoia** BlueGene/Q, 2011 IBM	1572864	17173	20133	7890
5	RIKEN AICS, Japan	K computer, SPARC64 VIIIfx , 2011 Fujitsu	705024	10510	11280	12660
6	Argonne National Laboratory, USA	**Mira** BlueGene/Q, 2012 IBM	786432	8587	10066	3945
7	DOE/NNSA/LANL/SNL, USA	Trinity - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, 2016 Cray	301056	8101	11079	3945
8	Swiss Natl. Supercomputer Center, Switzerland	**Piz Daint** Cray XC30/NVIDIA K20x, 2013 Cray	115984	6271	7789	2325
9	HLRS-Stuttgart, Germany	Hazel Hen - Cray XC40, Xeon E5-2680v3 12C 2.5GHz, 2016 Cray	185088	5640	7404	-
10	KAUST, Saudi Arabia	**Shaheen II** Cray XC40, Xeon E5-2698, 2015 Cray	196608	5537	7235	2834
85	ITC/U. Tokyo Japan	Oakleaf-FX SPARC64 IXfx, 2012 Fujitsu	76800	1043	1135	1177

http://www.top500.org/

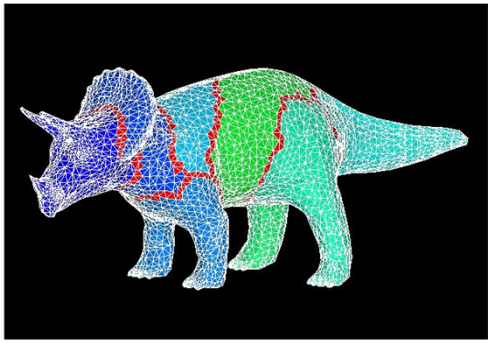
Computational Science
The 3rd Pillar of Science

- Theoretical & Experimental Science
- Computational Science
  - The 3rd Pillar of Science
  - Simulations using Supercomputers

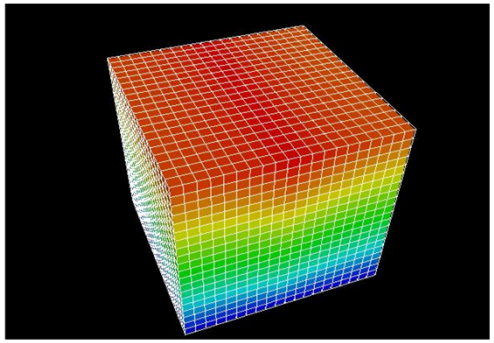


Methods for Scientific Computing

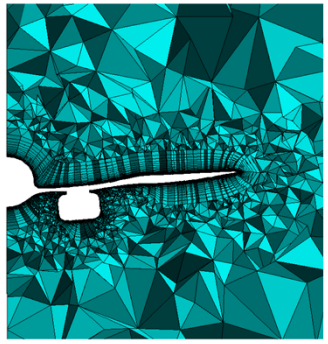
- Numerical solutions of PDE (Partial Diff. Equations)
- Grids, Meshes, Particles
  - Large-Scale Linear Equations
  - Finer meshes provide more accurate solutions



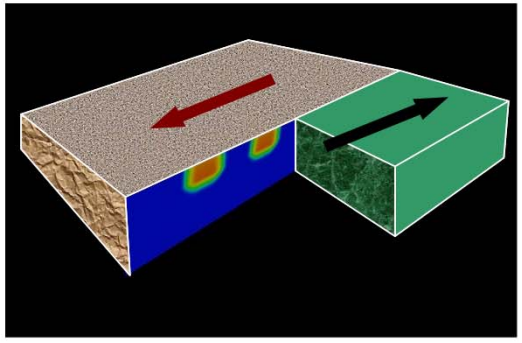
有限要素法
Finite Element Method
FEM



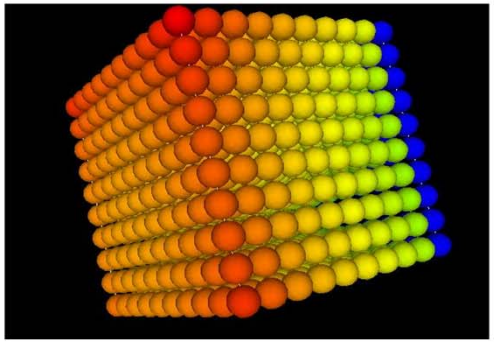
差分法
Finite Difference Method
FDM



有限体積法
Finite Volume Method
FVM



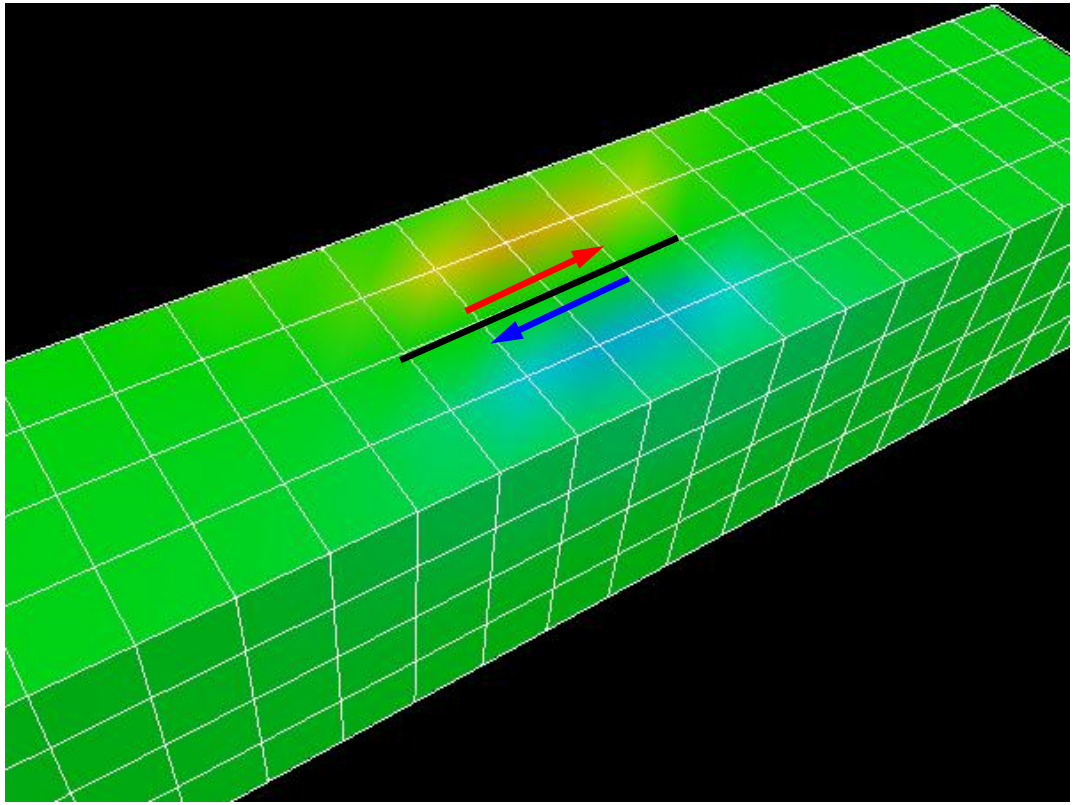
境界要素法
Boundary Element Method
BEM

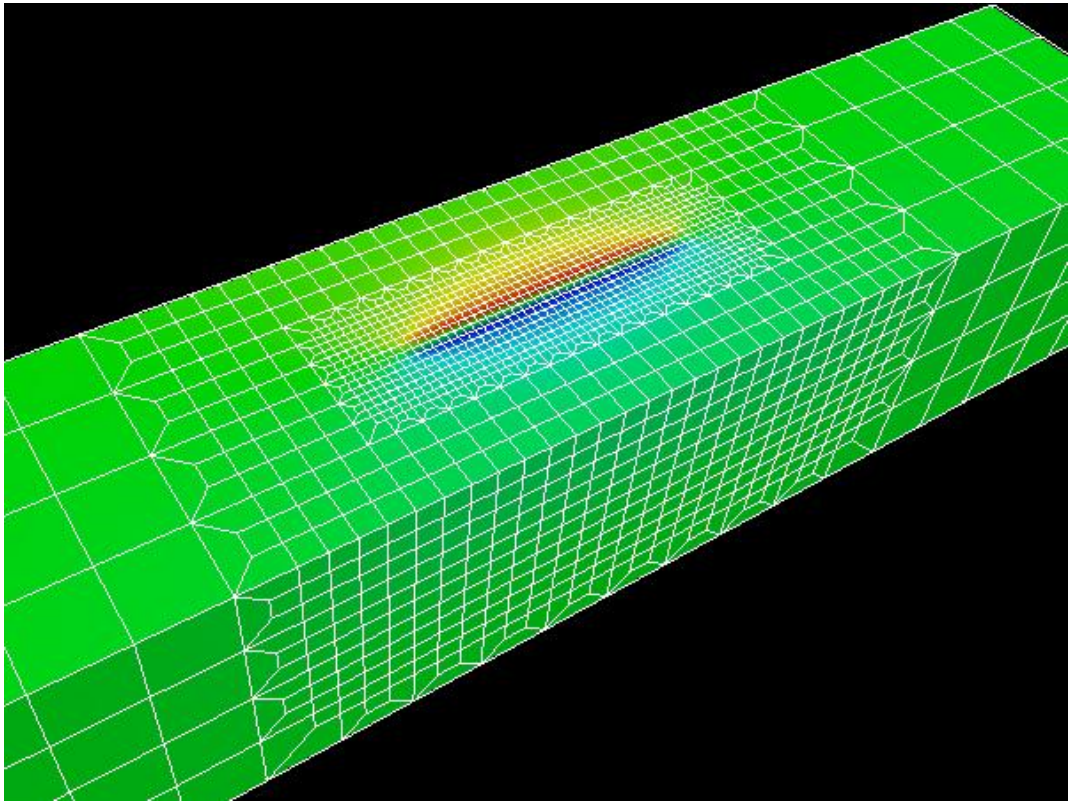


個別要素法
Discrete Element Method
DEM

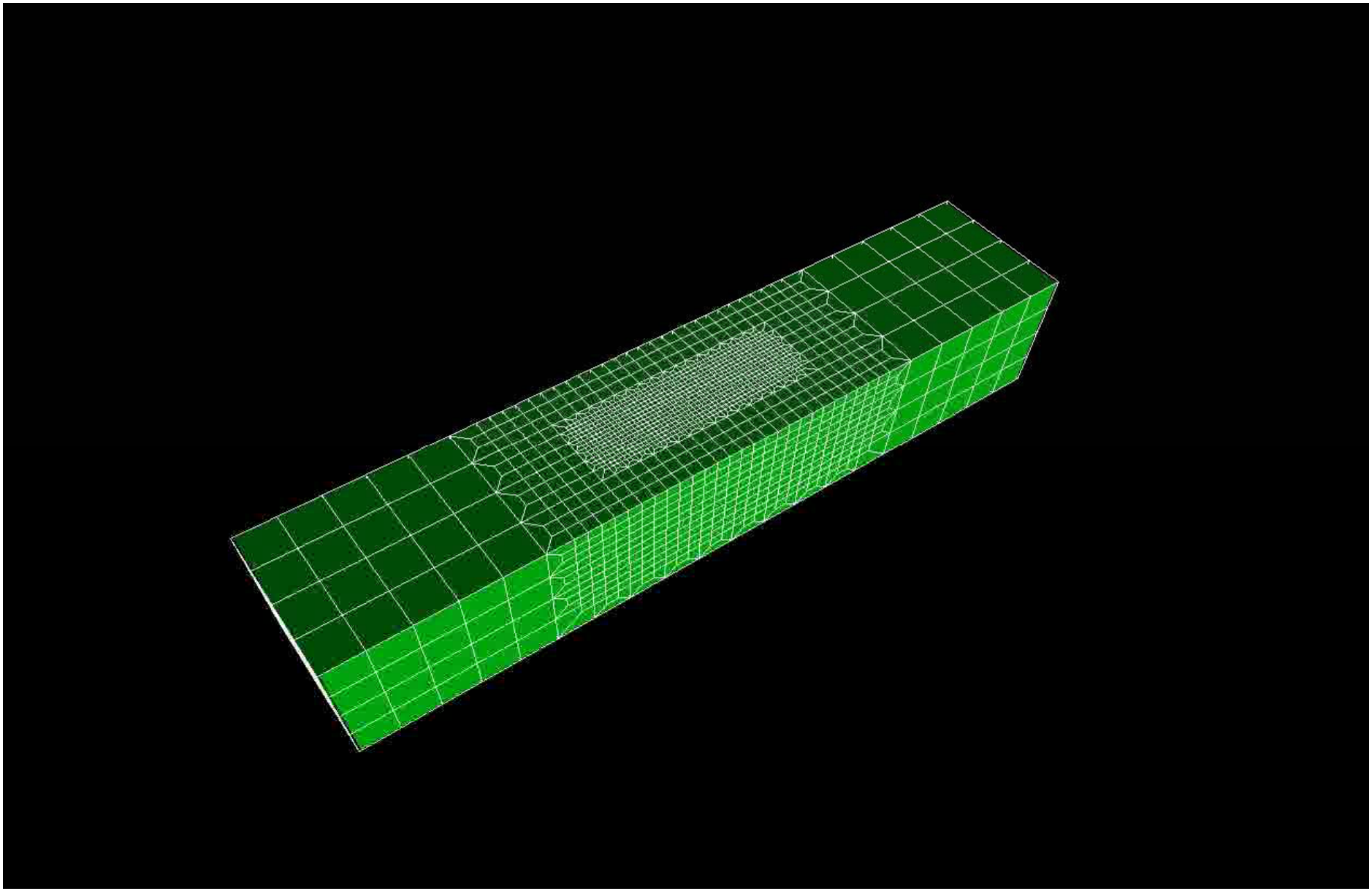
3D Simulations for Earthquake Generation Cycle San Andreas Faults, CA, USA

Stress Accumulation at Transcurrent Plate Boundaries

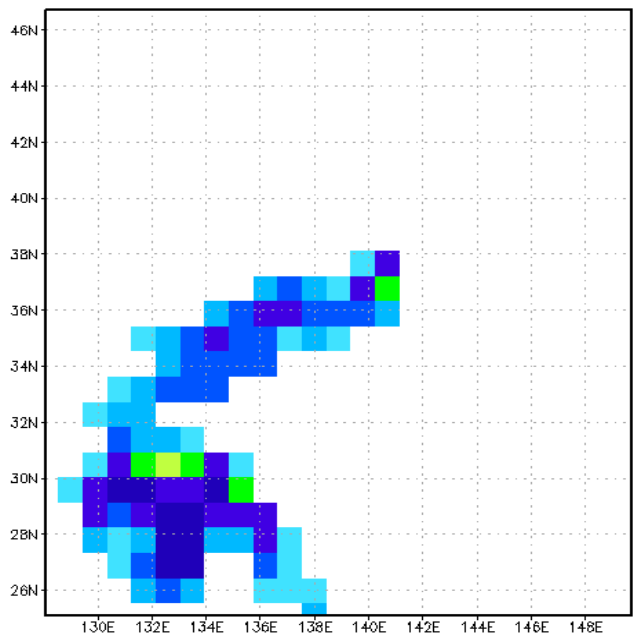




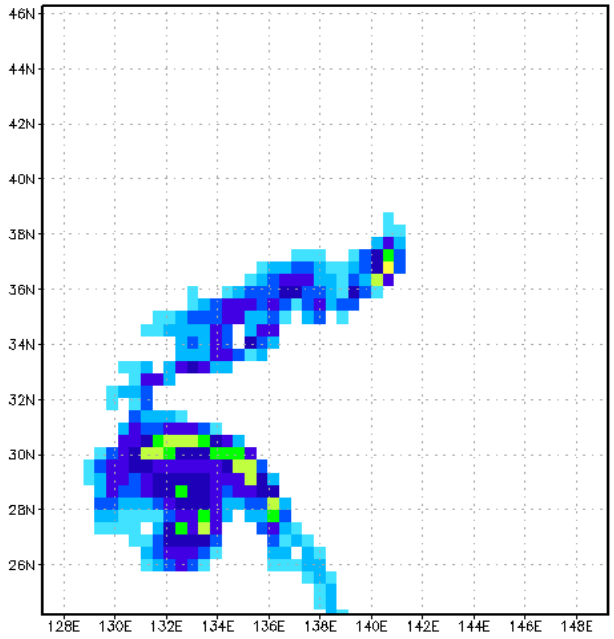
Adaptive FEM: High-resolution needed at meshes with large deformation (large accumulation)



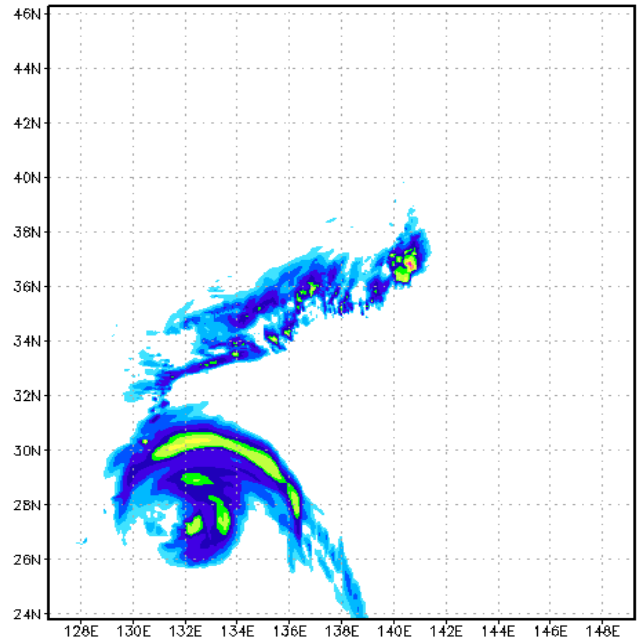
Typhoon Simulations by FDM
Effect of Resolution



Δh=100km



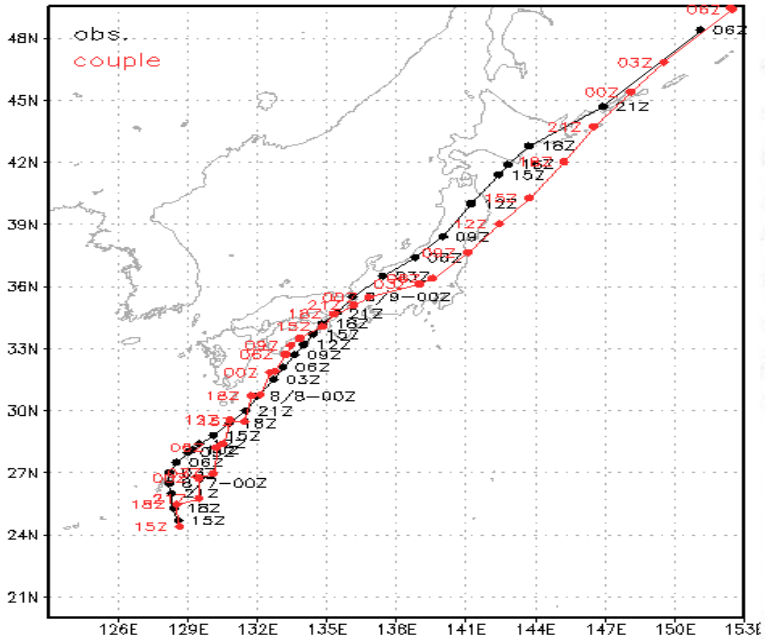
Δh=50km

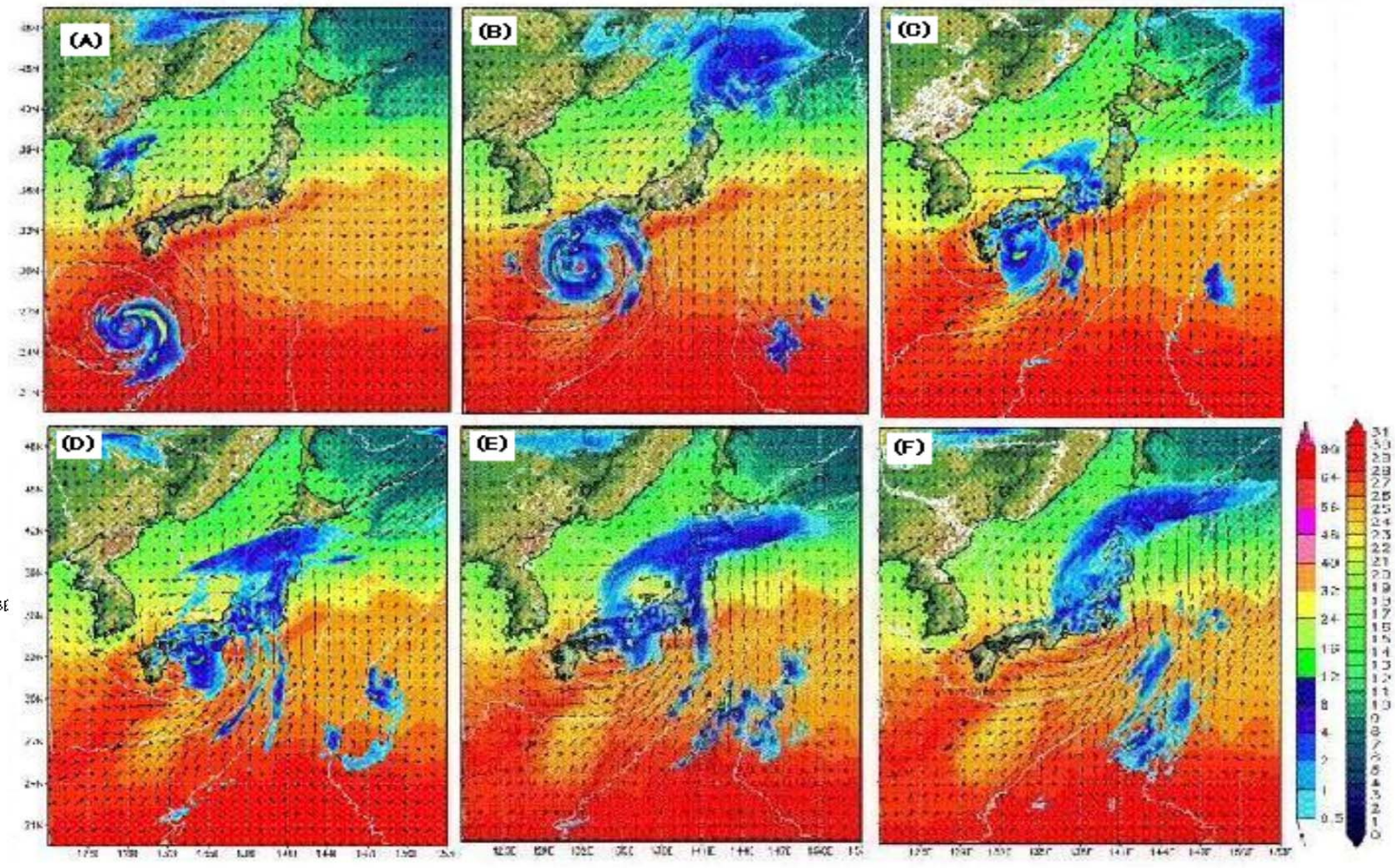


Δh=5km

[JAMSTEC]

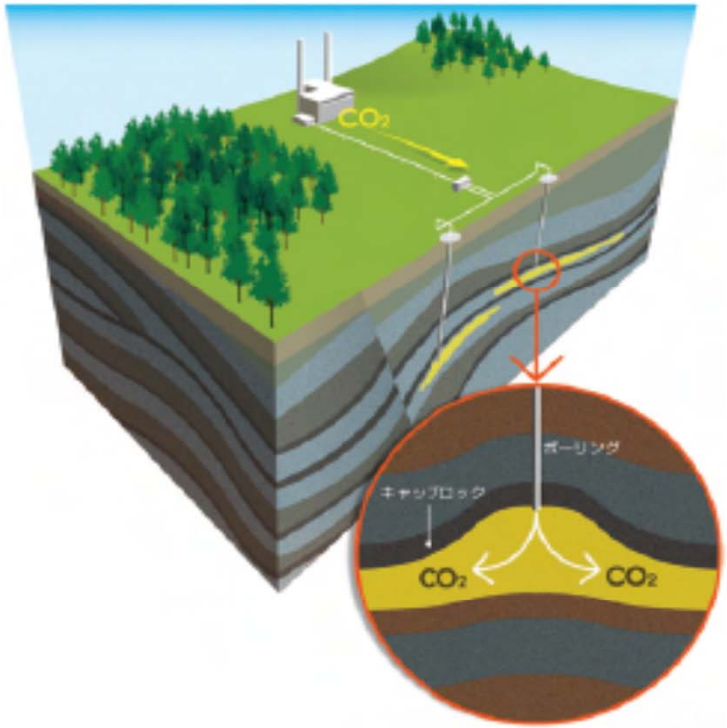
Simulation of Typhoon MANGKHUT in 2003 using the Earth Simulator





[JAMSTEC]

Simulation of Geologic CO_2 Storage



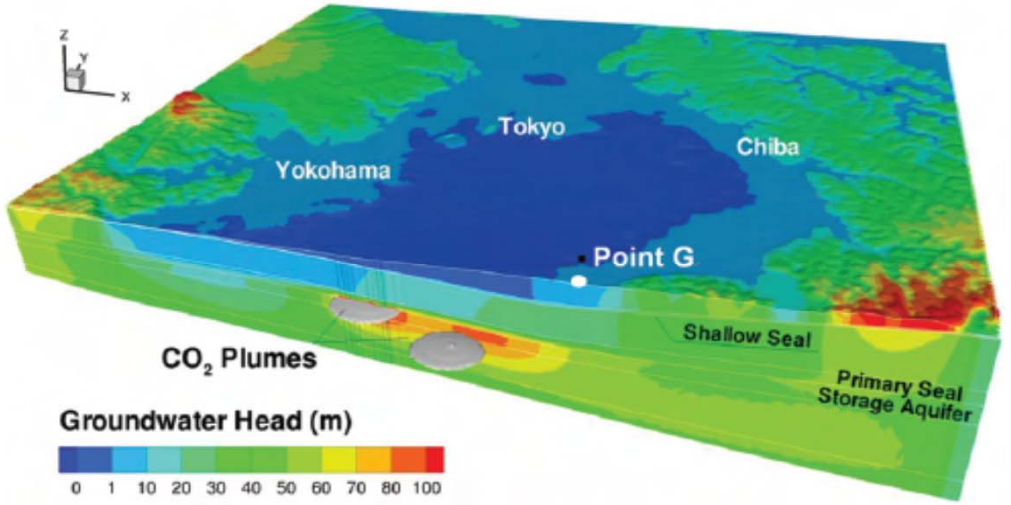
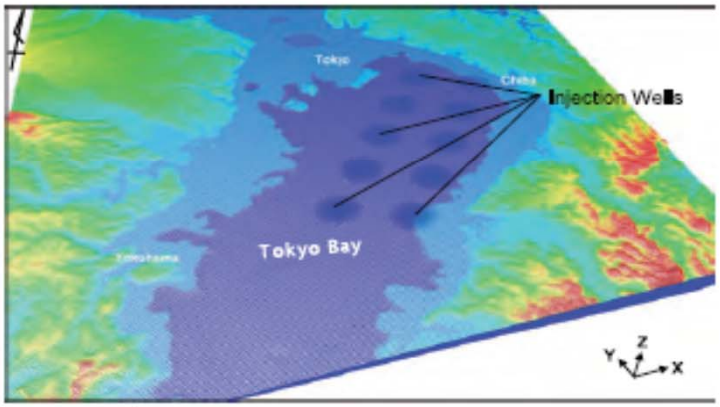
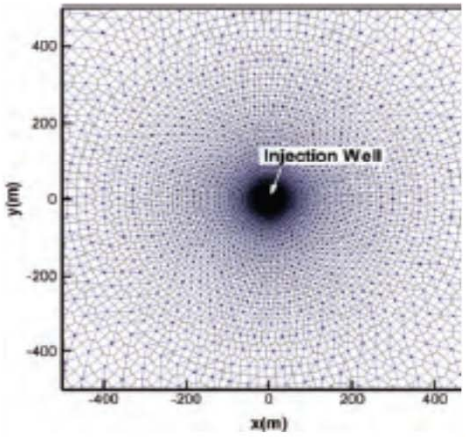
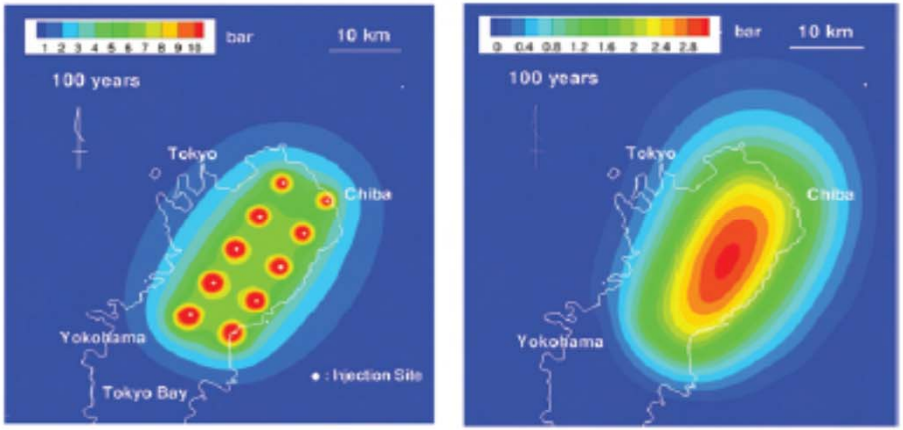


図 -4　CO_2 圧入後の地下水圧（全水頭換算）の分布（100 年後）







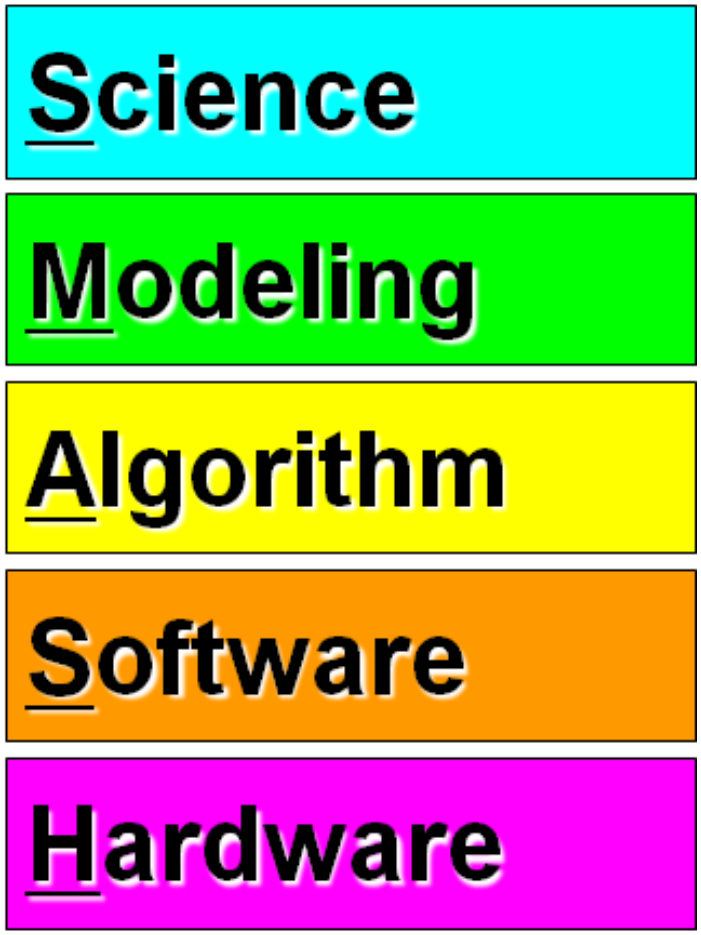
(a) 深部遮蔽層下面　　(b) 浅部遮蔽層下面

図 -5　圧力上昇量の平面分布（初期状態からの増分、圧入開始から 100 年後）

[Dr. Hajime Yamamoto, Taisei]

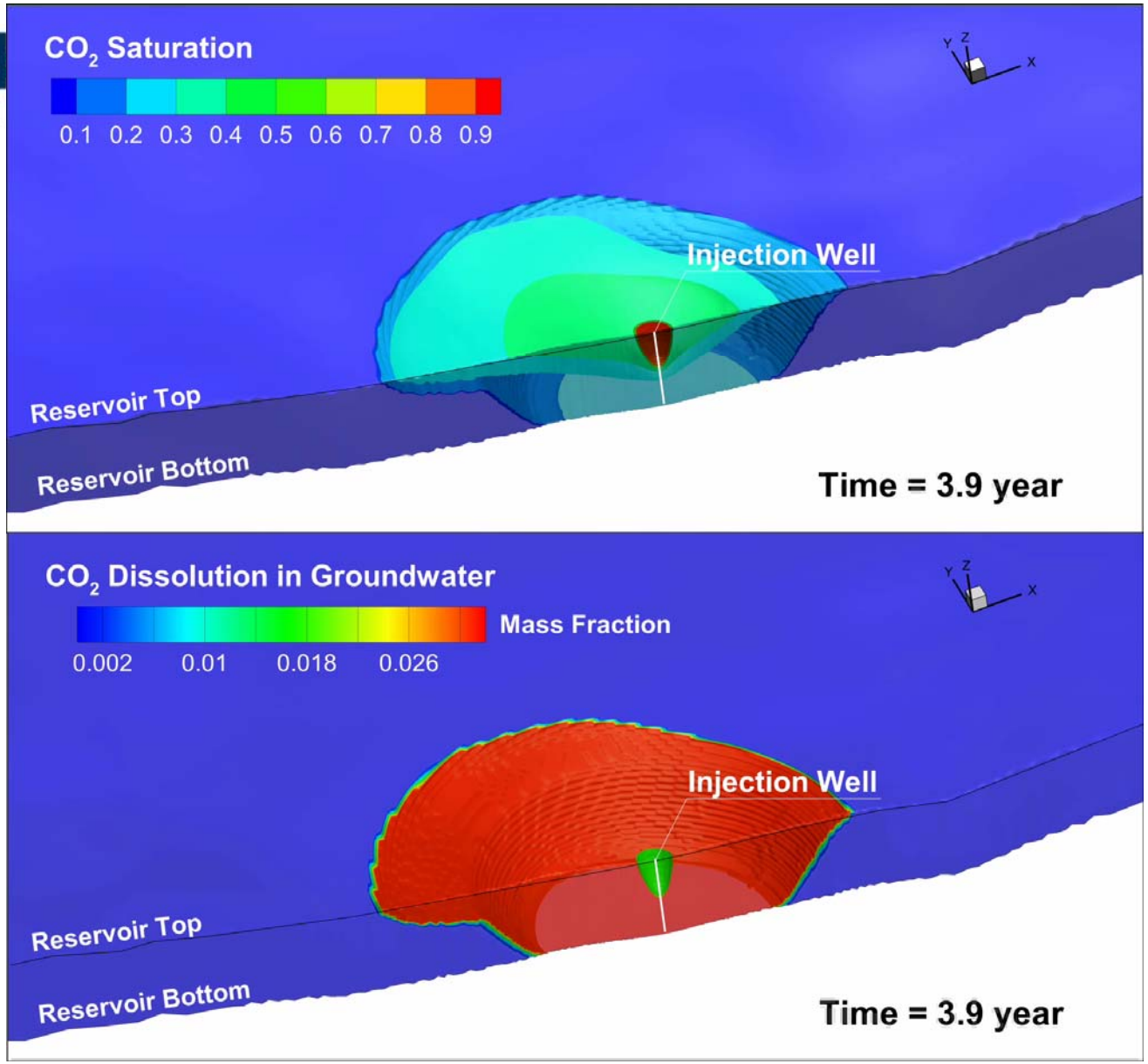
Simulation of Geologic CO_2 Storage

- International/Interdisciplinary Collaborations
  - Taisei (Science, Modeling)
  - Lawrence Berkeley National Laboratory, USA (Modeling)
  - Information Technology Center, the University of Tokyo (Algorithm, Software)
  - JAMSTC (Earth Simulator Center) (Software, Hardware)
  - NEC (Software, Hardware)
- 2010 Japan Geotechnical Society (JGS) Award



Simulation of Geologic CO_2 Storage

- Science
  - Behavior of CO_2 in supercritical state at deep reservoir
- PDE's
  - 3D Multiphase Flow (Liquid/Gas) + 3D Mass Transfer
- Method for Computation
  - TOUGH2 code based on FVM, and developed by Lawrence Berkeley National Laboratory, USA
    - More than 90% of computation time is spent for solving large-scale linear equations with more than 10^7 unknowns
- Numerical Algorithm
  - Fast algorithm for large-scale linear equations developed by Information Technology Center, the University of Tokyo
- Supercomputer
  - Earth Simulator (Peak Performance: 130 TFLOPS)
    - NEC, JAMSEC



Density convections for 1,000 years:

Flow Model

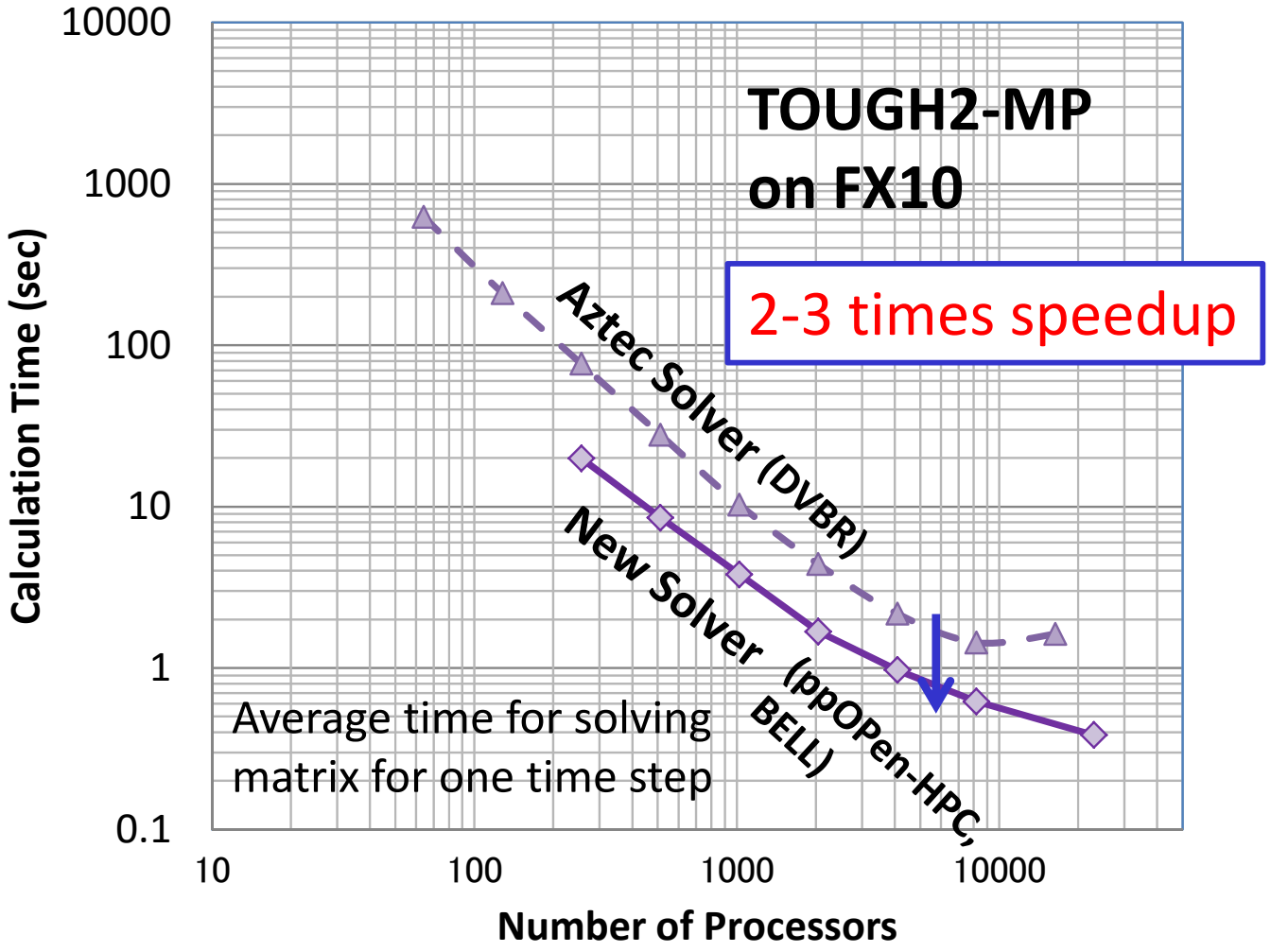
Only the far side of the vertical cross section passing through the injection well is depicted.

[Dr. Hajime Yamamoto, Taisei]

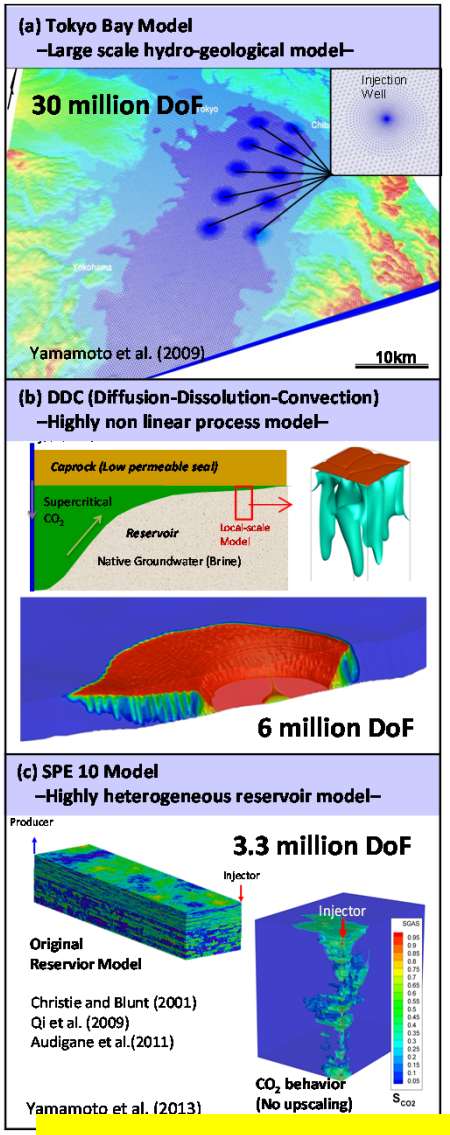
- The meter-scale fingers gradually developed to larger ones in the field-scale model
- Huge number of time steps (> 10^5) were required to complete the 1,000-yrs simulation
- Onset time (10-20 yrs) is comparable to theoretical (linear stability analysis, 15.5yrs)

Simulation of Geologic CO_2 Storage

30 million DoF (10 million grids × 3 DoF/grid node)



[Dr. Hajime Yamamoto, Taisei]



※

Fujitsu FX10（Oakleaf-FX），30M DOF: 2x-3x improvement

3D Multiphase Flow (Liquid/Gas) + 3D Mass Transfer

Motivation for Parallel Computing, again

- Large-scale parallel computer enables fast computing in large-scale scientific simulations with detailed models. Computational science develops new frontiers of science and engineering.
- Why parallel computing ?
  - faster
  - larger
  - “larger” is more important from the view point of “new frontiers of science & engineering”, but “faster” is also important.
  - + more complicated
  - Ideal: Scalable
    - Weak Scaling, Strong Scaling

- Target: Parallel FEM
- Supercomputers and Computational Science
- **Overview of the Class**
- Future Issues

Information of this Class

- Instructor
  - Kengo Nakajima (Information Technology Center)
    - Information Technology Center (Asano) Annex 3F #36 ex: 22719
    - e-mail: nakajima(at)cc.u-tokyo.ac.jp
- Schedule
  - September 5,6,8,9, 12-15
  - 09:00-10:30, 10:45-12:15, 13:30-15:00, 15:15-16:45
  - **http://nkl.cc.u-tokyo.ac.jp/16e/**
- Practice
  - Time for exercise
- Lecture Room
  - Information Technology Center (Asano) Seminar Room #2 (1F)
    - No Foods, No Drinks

Prerequisites

- Knowledge and experiences in fundamental methods for numerical analysis (e.g. Gaussian elimination, SOR)
- Knowledge and experiences in UNIX
- Experiences in programming using FORTRAN or C
- “Seminar on Advanced Computing (35616-4009)” should be also registered
- Account for Educational Campuswide Computing System (ECC System) should be obtained in advance:
  - http://www.ecc.u-tokyo.ac.jp/ENGLISH/index-e.html

Grading by Reports ONLY

- MPI (Collective Communication) (S1)
- MPI (1D Parallel FEM) (S2)
  - If you complete (S1-S2), you get credits of “Programming for Parallel Computing (616-2057)” .
- Parallel FEM (P1)
  - If you complete (P1), you get credits of “Seminar on Advanced Computing (616-4009)” are graded.

- Sample solutions will be available
- Deadline: October 22nd (Sat) 17:00
  - By E-mail: nakajima(at)cc.u-tokyo.ac.jp
  - You can bring hard-copy’s to my office ...

Homepage

- **http://nkl.cc.u-tokyo.ac.jp/16e/**
  - General information is available
  - No hardcopy of course materials are provided (Please print them by yourself)

参考文献(1/2)

- 菊地「有限要素法概説(新訂版)」, サイエンス社, 1999.
- 竹内, 樫山, 寺田(日本計算工学会編)「計算力学:有限要素法の基礎」, 森北出版, 2003.
- 登坂, 大西「偏微分方程式の数値シミュレーション 第2版」, 東大出版会, 2003.
  - 差分法, 境界要素法との比較
- 福森「よくわかる有限要素法」, オーム社, 2005.
  - ヘルムホルツ方程式
- 矢川, 宮崎「有限要素法による熱応力・クリープ・熱伝導解析」, サイエンス社, 1985.(品切)
- Segerlind, L.(川井監訳)「応用有限要素解析 第2版」, 丸善, 1992.(品切)

参考文献（より進んだ読者向け）

- 菊池，岡部「有限要素システム入門」，日科技連，1986.
- 山田「高性能有限要素法」，丸善，2007.
- 奥田，中島「並列有限要素法」，培風館，2004.

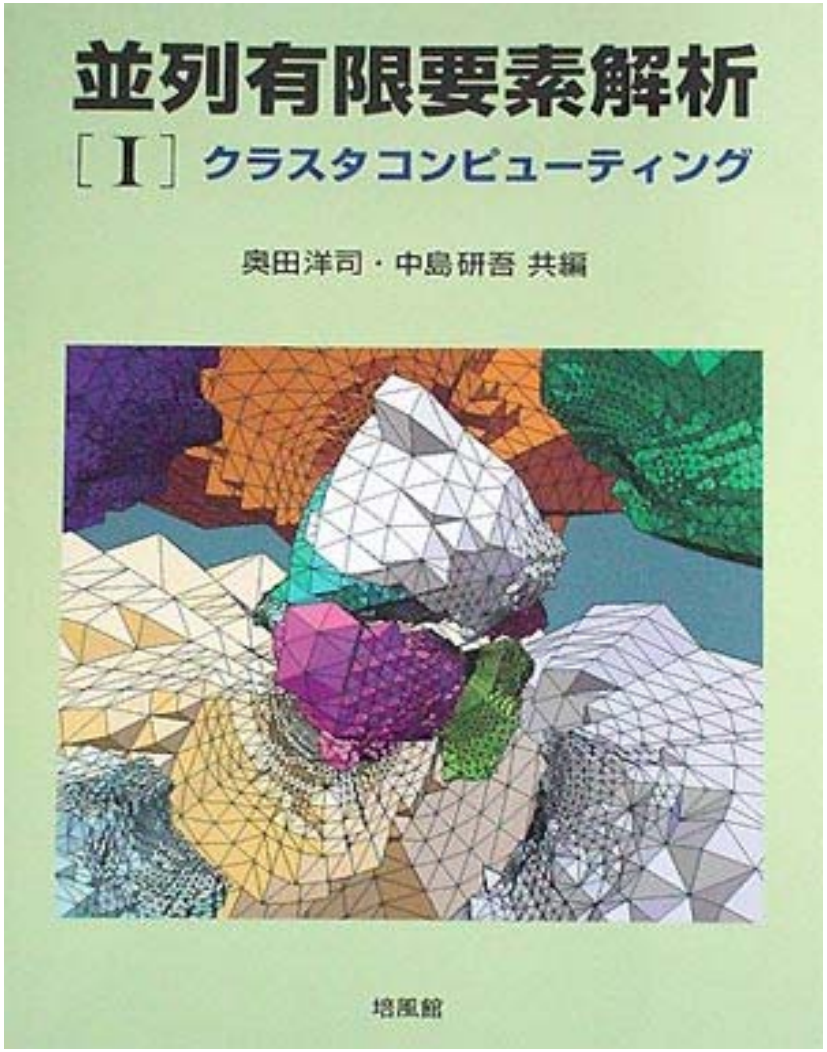
- Smith, I. 他「Programming the Finite Element Method (4th edition)」，Wiley.

奥田，中島編「並列有限要素解析〔I〕クラスタコンピューティング」

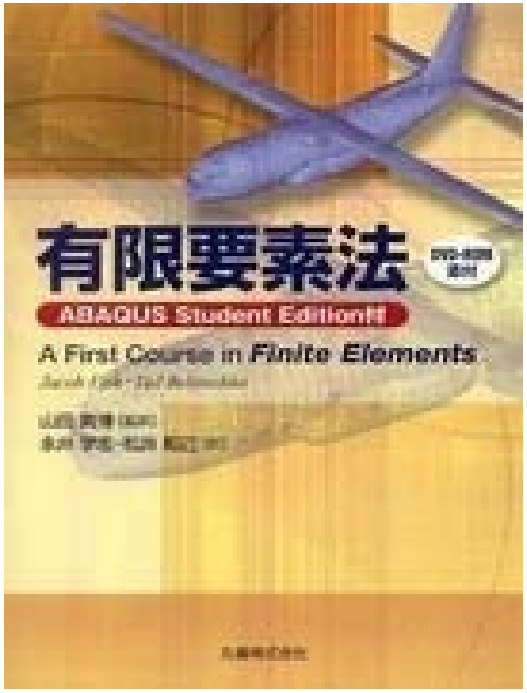
培風館，2004.

- 「GeoFEM」の成果のまとめ
  - **http://geofem.tokyo.rist.or.jp**
- 「地球シミュレータ」上での最適化，シミュレーション結果を紹介

- 初心者向けでは無い
- 高い・・・
  - 若干残部があるので希望者には貸し出します。



References



- Fish, Belytschko, A First Course in Finite Elements, Wiley, 2007
  - Japanese version is also available
  - “ABAQUS Student Edition” included
- Smith et al., Programming the Finite Element Method (4th edition), Wiley, 2004
  - Parallel FEM
- Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Dover, 2000

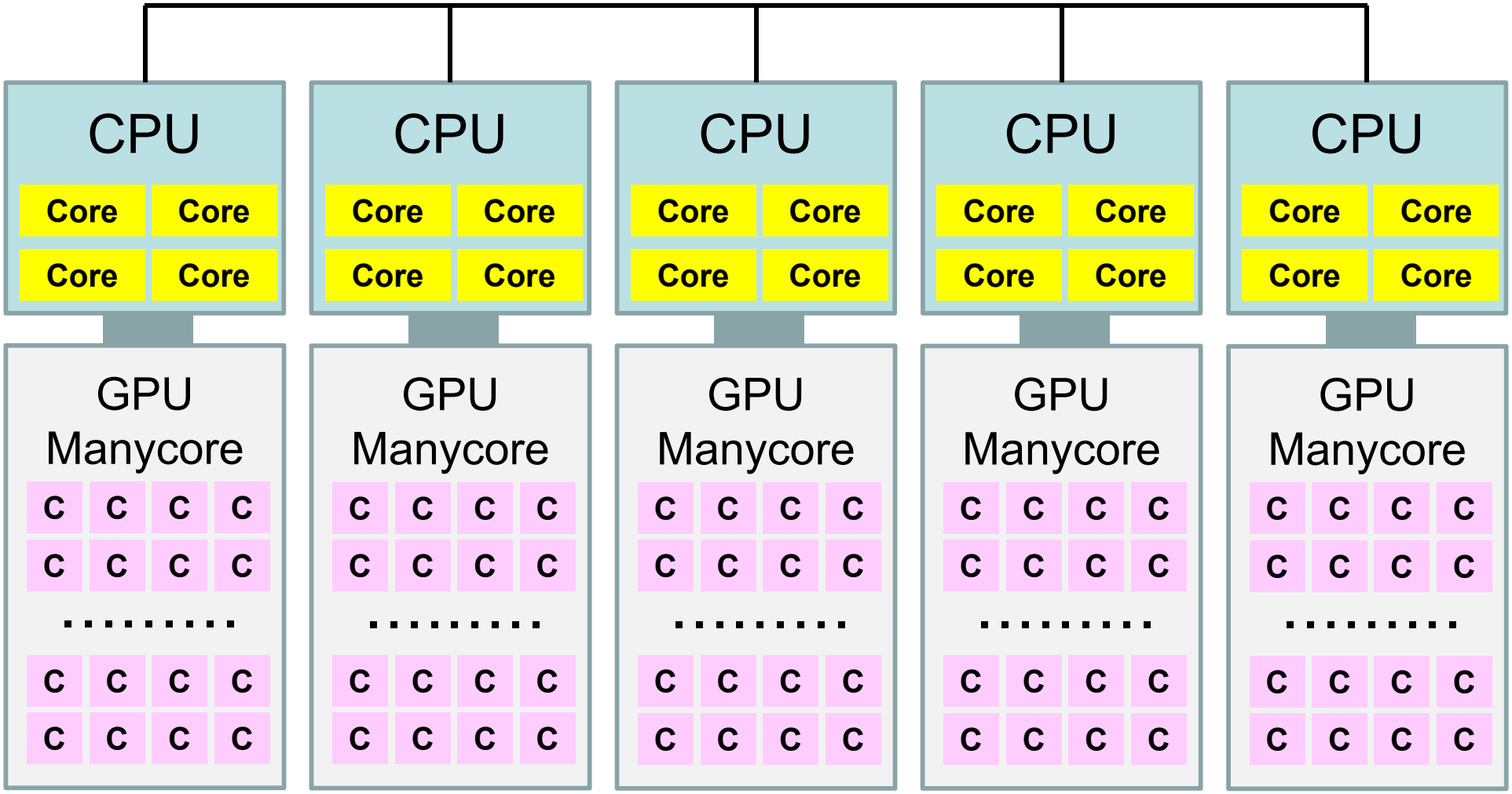
- Target: Parallel FEM
- Supercomputers and Computational Science
- Overview of the Class
- **Future Issues**

Key-Issues towards Appl./Algorithms on Exa-Scale Systems

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

- Hybrid/Heterogeneous Architecture
  - Multicore + GPU/Manycores (Intel MIC/Xeon Phi)
    - Data Movement, Hierarchy of Memory
- Communication/Synchronization Reducing Algorithms
- Mixed Precision Computation
- Auto-Tuning/Self-Adapting
- Fault Resilient Algorithms
- Reproducibility of Results

Supercomputers with Heterogeneous/Hybrid Nodes

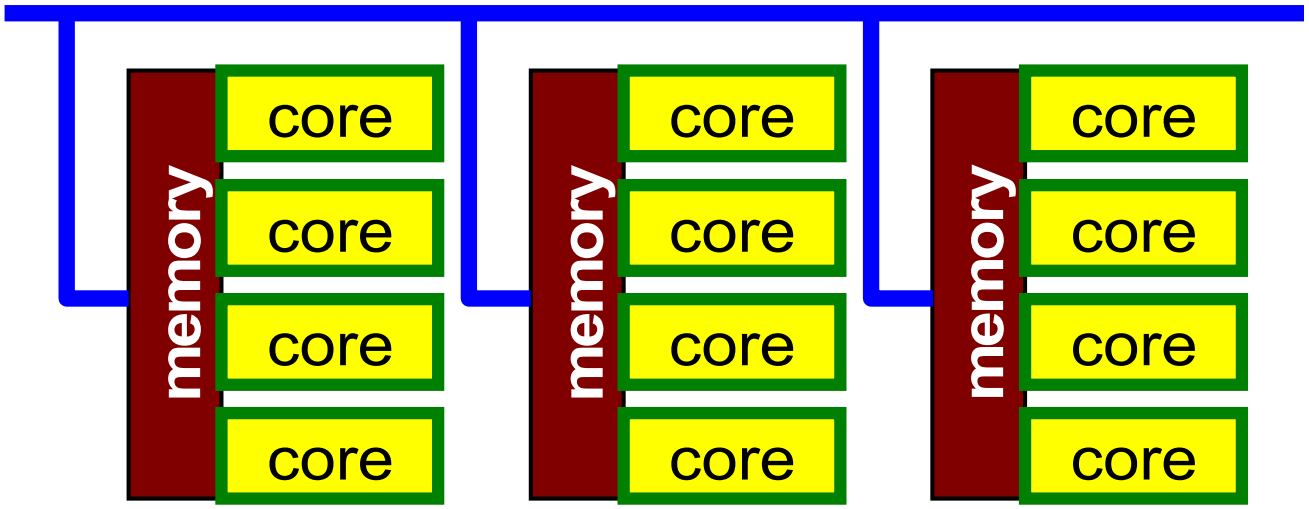


Hybrid Parallel Programming Model is essential for Post-Peta/Exascale Computing

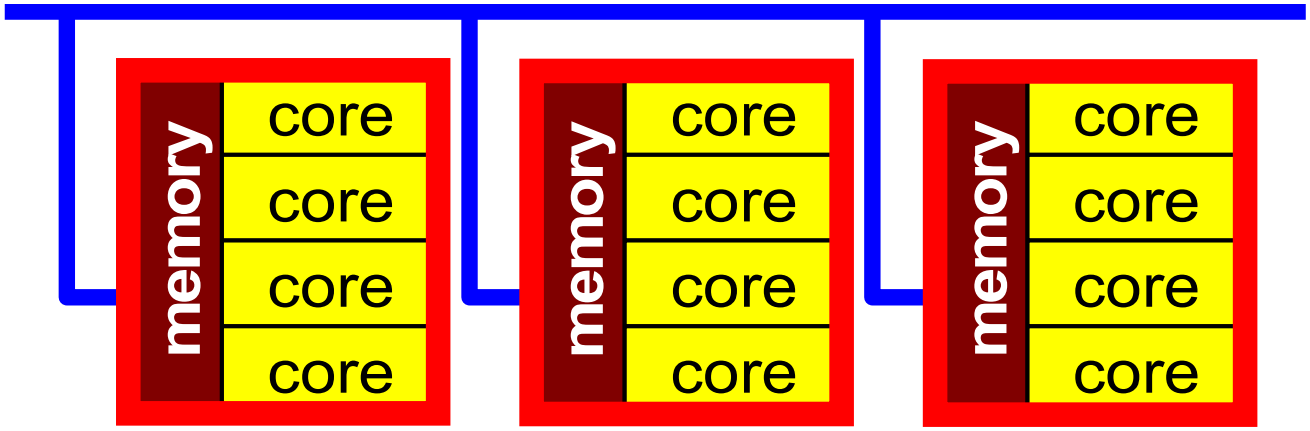
- Message Passing (e.g. MPI) + Multi Threading (e.g. OpenMP, CUDA, OpenCL, OpenACC etc.)
- In K computer and FX10, hybrid parallel programming is recommended
  - MPI + Automatic Parallelization by Fujitsu's Compiler
- Expectations for Hybrid
  - Number of MPI processes (and sub-domains) to be reduced
  - O(10^8-10^9)-way MPI might not scale in Exascale Systems
  - Easily extended to Heterogeneous Architectures
    - CPU+GPU, CPU+Manycores (e.g. Intel MIC/Xeon Phi)
    - MPI+X: OpenMP, OpenACC, CUDA, OpenCL

Flat MPI vs. Hybrid

Flat-MPI：Each PE -> Independent



Hybrid：Hierarchal Structure



In this class...

- Very brief introduction of OpenMP and OpenMP/MPI Hybrid Parallel Programming Model will be provided.

- MPI is essential for large-scale scientific computing. If you want to something new using supercomputers, you must learn MPI, then OpenMP.
  - You don’t have to be attracted by PGAS (e.g. HPF), automatic parallelization（自動並列化）, etc.

Example of OpnMP/MPI Hybrid

Sending Messages to Neighboring Processes

MPI: Message Passing, OpenMP: Threading with Directives

```
!C
!C- SEND

      do neib= 1, NEIBPETOT
        II= (LEVEL-1)*NEIBPETOT
        istart= STACK_EXPORT(II+neib-1)
        inum  = STACK_EXPORT(II+neib  ) - istart
!$omp parallel do
        do k= istart+1, istart+inum
          WS(k-NE0)= X(NOD_EXPORT(k))
        enddo

        call MPI_Isend (WS(istart+1-NE0), inum, MPI_DOUBLE_PRECISION,   &
     &                  NEIBPE(neib), 0, MPI_COMM_WORLD,                &
     &                  req1(neib), ierr)
      enddo
```

Parallel Programming Models

- Multicore Clusters (e.g. K, FX10)
  - MPI + OpenMP and (Fortan/C/C++)
- Multicore + GPU (e.g. Tsubame)
  - GPU needs host CPU
  - MPI and [(Fortan/C/C++) + CUDA, OpenCL]
    - complicated,
  - MPI and [(Fortran/C/C++) with OpenACC]
    - close to MPI + OpenMP and (Fortran/C/C++)
- Multicore + Intel MIC/Xeon-Phi (e.g. Stampede)
  - Xeon-Phi needs host CPU (currently)
  - MPI + OpenMP and (Fortan/C/C++) is possible
    - + Vectorization

Future of Supercomputers (1/2)

- Technical Issues
  - Power Consumption
  - Reliability, Fault Tolerance, Fault Resilience
  - Scalability (Parallel Performancce)
- Petascale System
  - 2MW including A/C, 2M$/year, O($10^5$～$10^6$) cores
- Exascale System (10^3x Petascale)
  - After 2020, 2023 ?
    - 2GW (2 B$/year !), O($10^8$～$10^9$) cores
  - Various types of innovations are on-going
    - to keep power consumption at 20MW (100x efficiency)
    - CPU, Memory, Network ...
  - Reliability

Future of Supercomputers (2/2)

- Not only hardware, but also numerical models and algorithms must be improved:
  - 省電力アルゴリズム（Power-Aware/Reducing）
  - 耐故障アルゴリズム（Fault Resilient）
  - 通信削減アルゴリズム（Communication Avoiding/Reducing）

- Co-Design by experts from various area (SMASH) is important
  - Exascale system will be a special-purpose system, not a general-purpose one.