Introduction
Overview of the Class

http://nkl.cc.u-tokyo.ac.jp/22w/

Kengo Nakajima
Information Technology Center
The University of Tokyo

Descriptions of Class

- Technical & Scientific Computing II (4820-1028)
  - 科学技術計算Ⅱ
  - Department of Mathematical Informatics
- Seminar on Computational Science Alliance II (4810-1205)
  - 計算科学アライアンス科学特別講義Ⅱ
  - Department of Computer Science
- Hybrid Distributed Parallel Computing (3747-111)
  - ハイブリッド分散並列コンピューティング
  - Department of Electrical Engineering & Information Systems

- 2009-2014
  - Introduction to FEM Programming
    - FEM: Finite-Element Method：有限要素法
  - Summer (I) : FEM Programming for Solid Mechanics
  - Winter (II): Parallel FEM using MPI
    - The 1st part (summer) is essential for the 2nd part (winter)
- Problems
  - Many new (international) students in Winter, who did not take the 1st part in Summer
  - **They are generally more diligent than Japanese students**
- 2015
  - Summer (I) : Multicore programming using OpenMP
  - Winter (II): FEM + Parallel FEM using MPI for Heat Conduction
  - Part I & II are mostly independent
- 2016: Reedbush-U
- **2017: English**
- 2019: OBCX, 2020: Online, **2022: Odyssey** …

- Instructor
  - Kengo Nakajima
  - Professor, Information Technology Center, The University of Tokyo
- Topics
  - Finite-Element Method (FEM)
    - Heat Conduction: easier than solid mechanics
  - Parallel FEM using MPI and OpenMP

Kengo Nakajima (1/2) 中島研吾

- 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
  - Deputy Director, RIKEN Center for Computational Science (R-CCS) (2018 April -)
- 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

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

This 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 **Wisteria/BDEC-01 (Odyssey) with A64FX Processors**.

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.

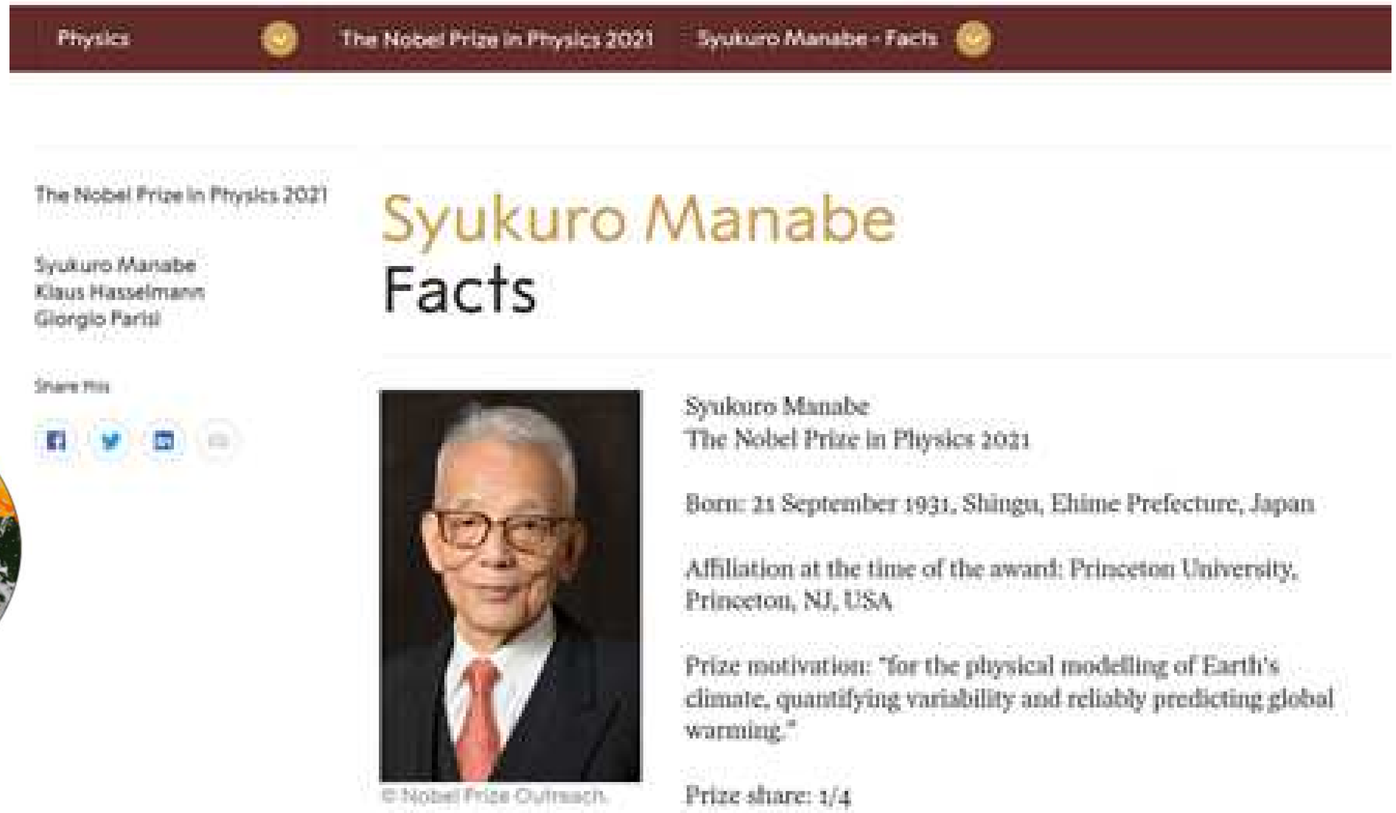
Lectures for parallel FEM will be focused on design of data structure for distributed local mesh files, which is the key issue for efficient parallel FEM. Introduction to MPI (Message Passing Interface), which is widely used method as "de facto standard" of parallel programming, is also provided.

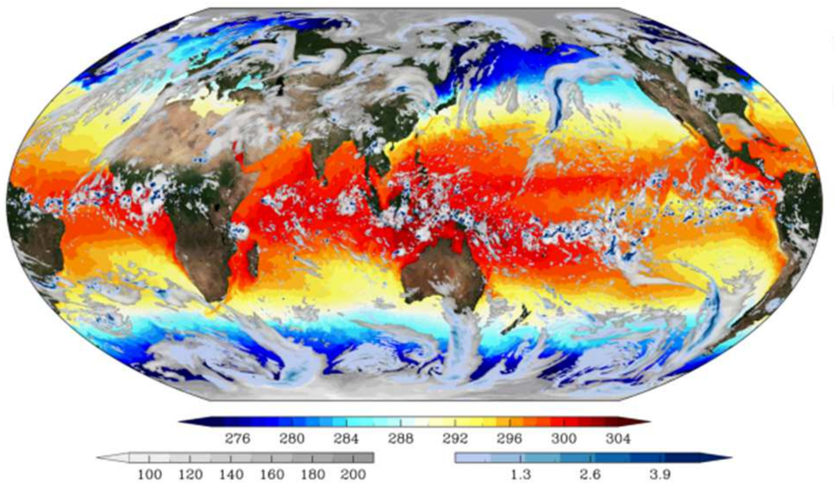
Solving large-scale linear equations with sparse coefficient matrices is the most expensive and important part of FEM and other methods for scientific computing, such as Finite-Difference Method (FDM) and Finite-Volume Method (FVM). Recently, families of Krylov iterative solvers are widely used for this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





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**

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 ?
  - Science (Physics, Chemistry, Bio etc.)
  - Engineering
  - Math/Applied Math
  - Computer Science

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)

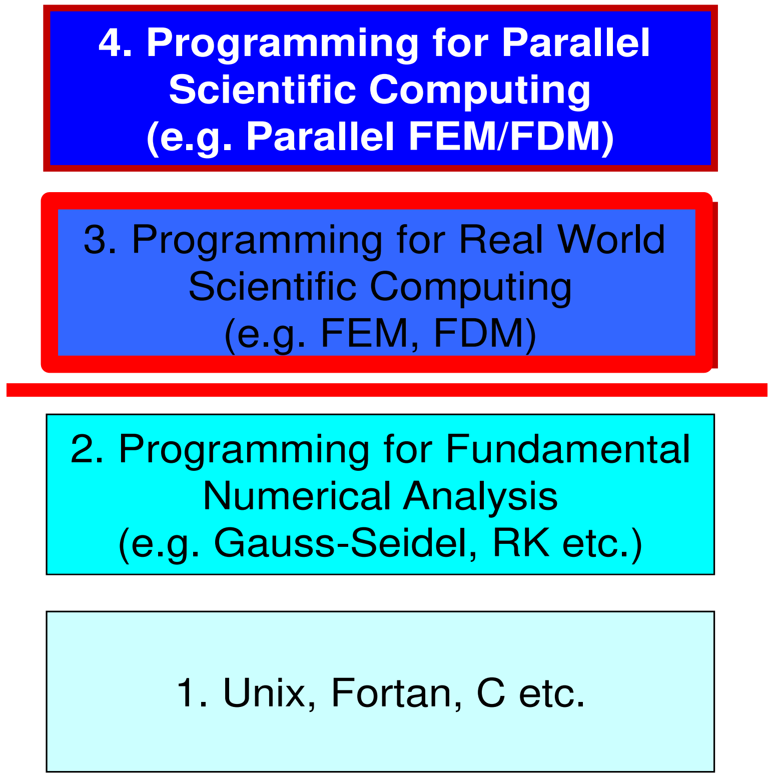
Big gap here !!

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



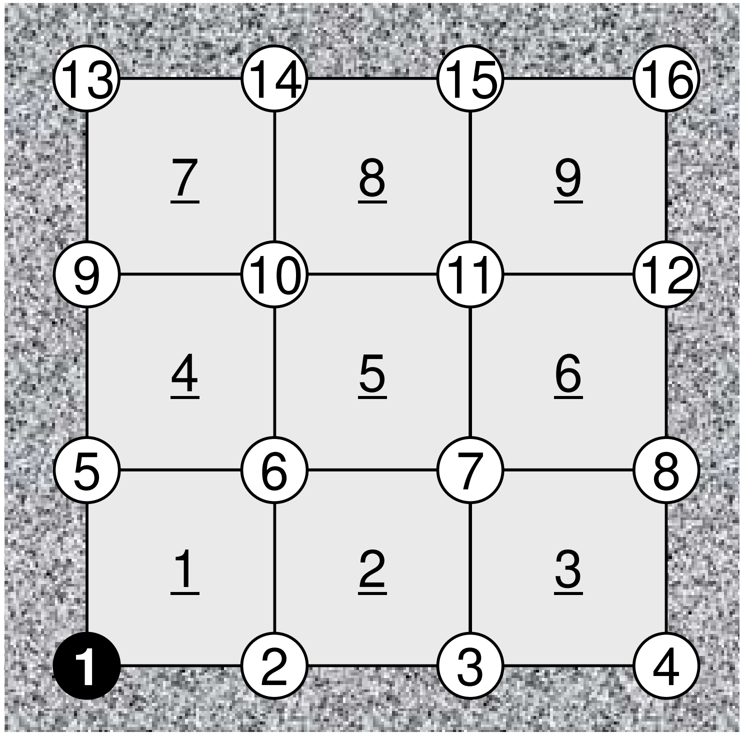
- How to understand the code ?
  - Reading the application code !!
  - It seems primitive, but very effective.
  - In this class, "reading the source code" is encouraged.

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.

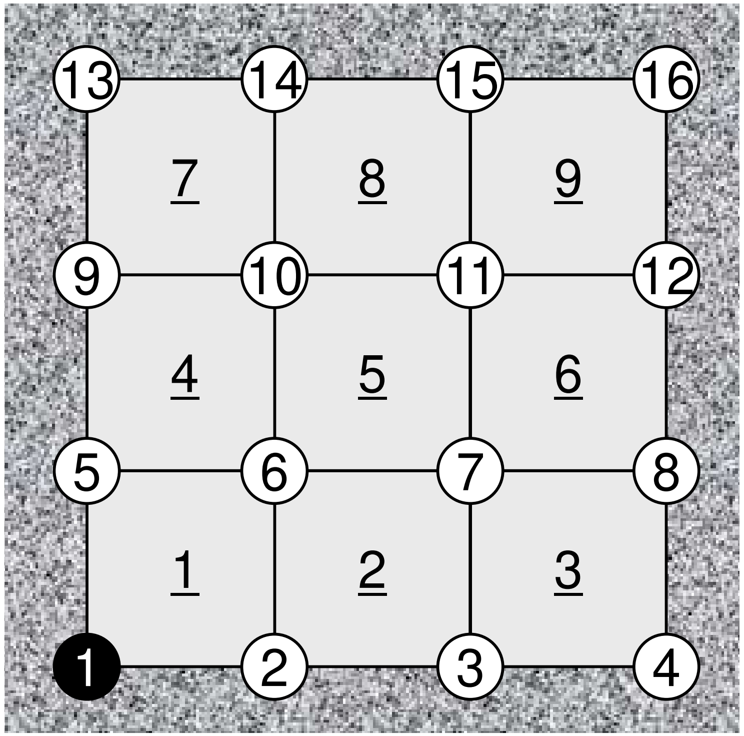
$\{\phi\}$: T at each vertex

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

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

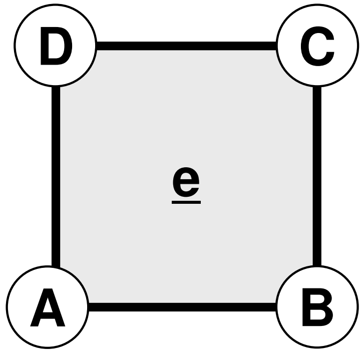
$$-\int_V \lambda \left(\frac{\partial [N]^T}{\partial x} \frac{\partial [N]}{\partial x} + \frac{\partial [N]^T}{\partial y} \frac{\partial [N]}{\partial y} \right) dV \cdot \{\phi\}$$

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



Element Matrix

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



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

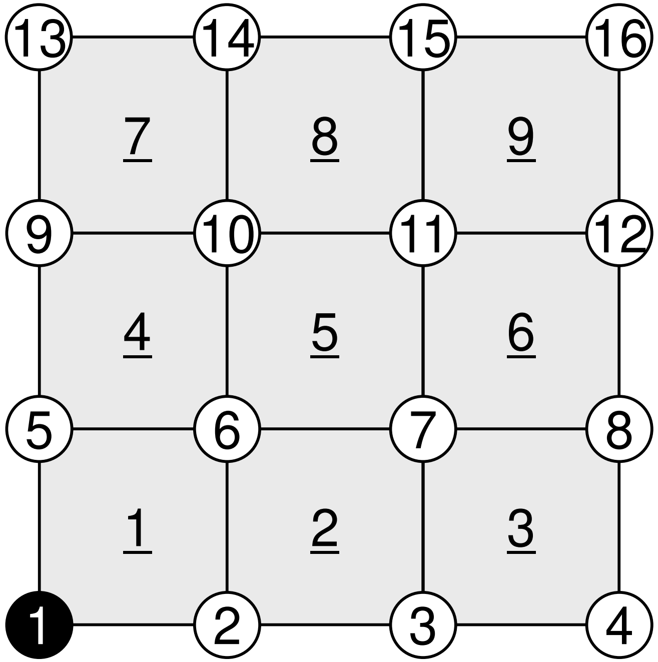
$$+\int_V Q[N]^T dV = 0$$

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

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

Global (Overall) Matrix

Accumulate each element matrix to “global” matrix.

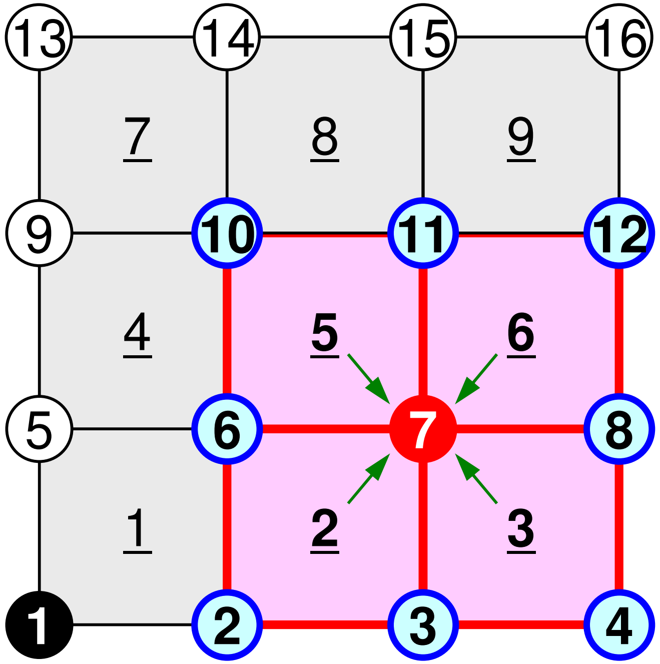


$$[K]\{\Phi\}=\{F\}$$

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

To each node …

Effect of surrounding elem's/nodes are accumulated.



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

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

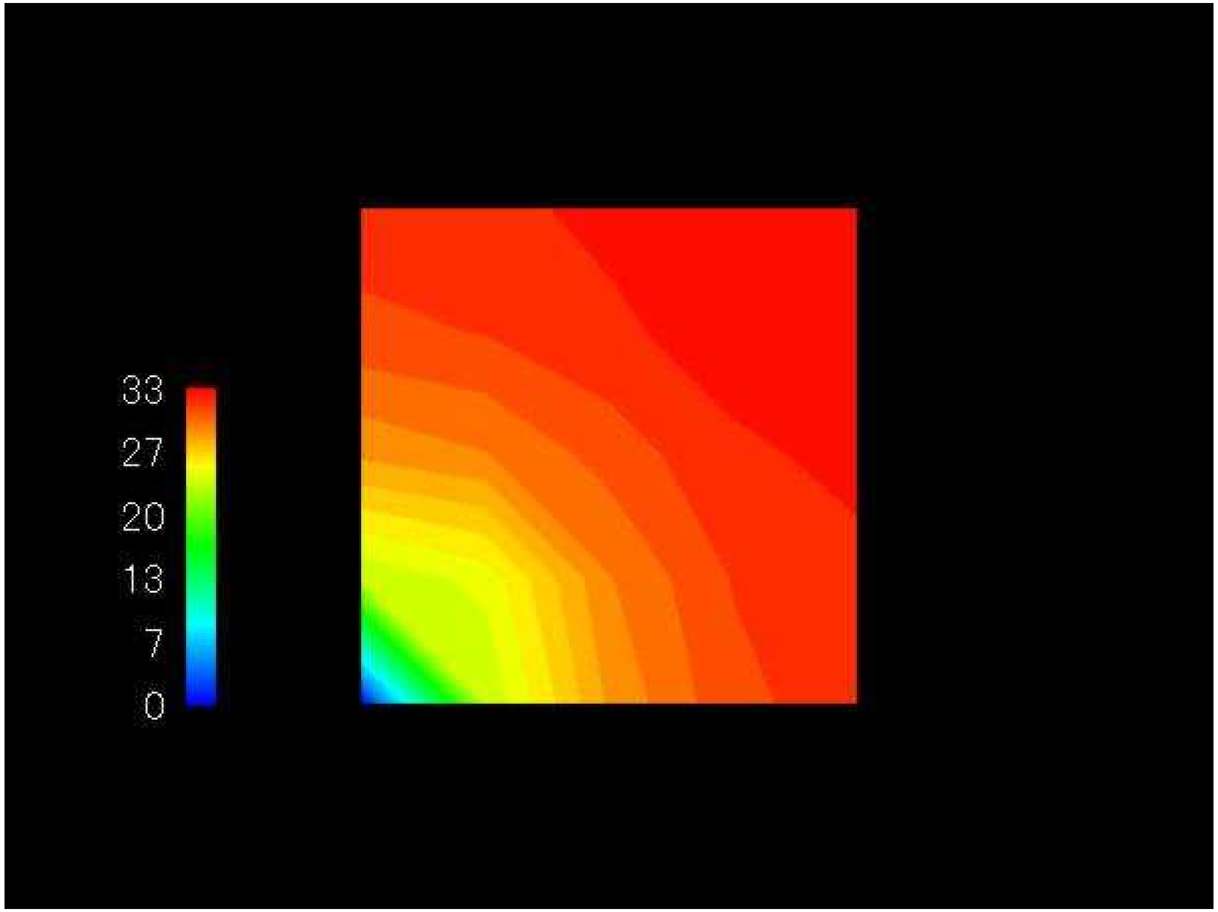
Coefficient matrix is very "sparse", many "0's"
Contributions from only neighbors

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 …

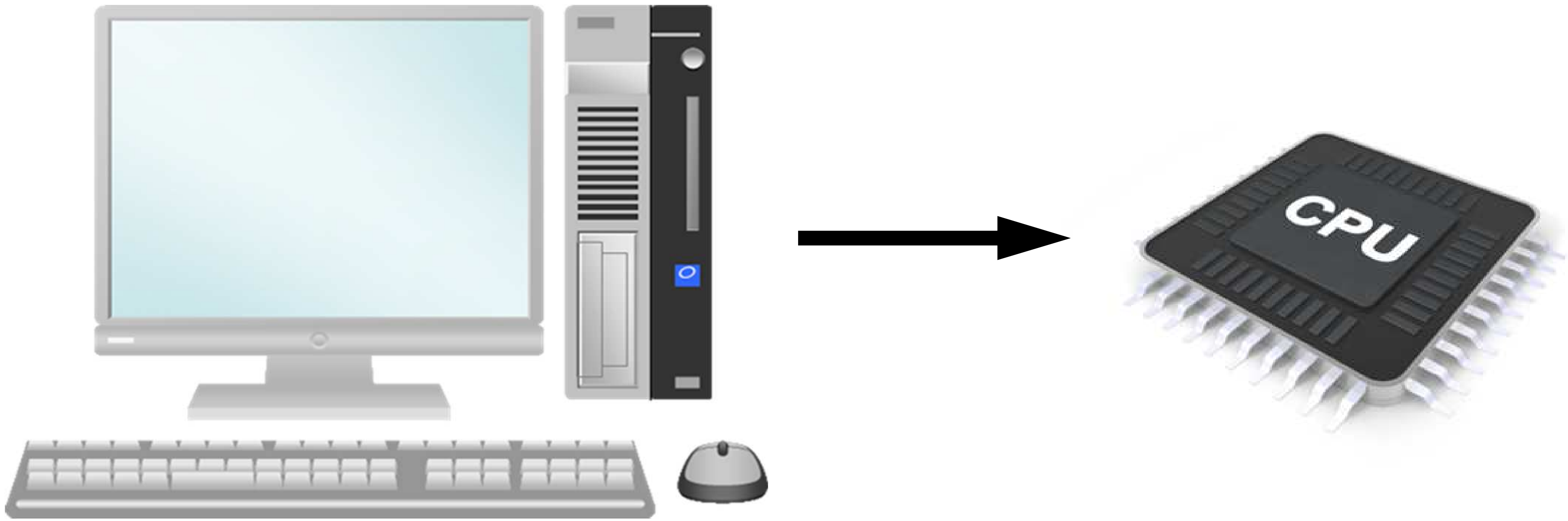


Features of FEM applications

- 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

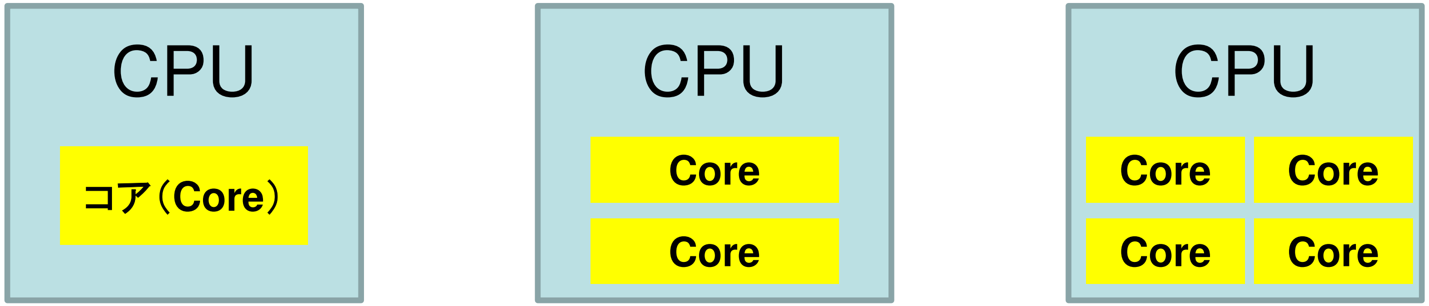
- Target
  - Parallel FEM
- **Supercomputers and Computational Science**
- Overview of the Class
- Future Issues

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 (or more) instructions per clock

Multicore CPU



Single Core
1 cores/CPU

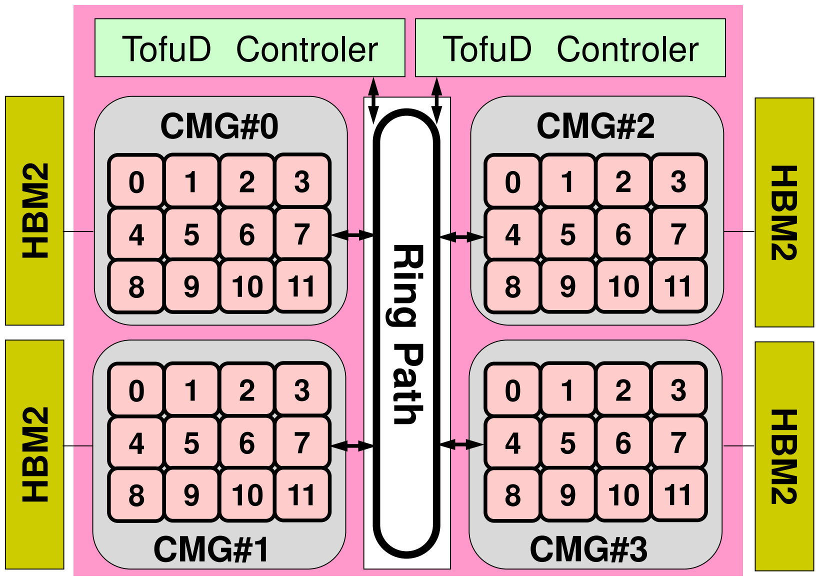
Dual Core
2 cores/CPU

Quad Core
4 cores/CPU

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

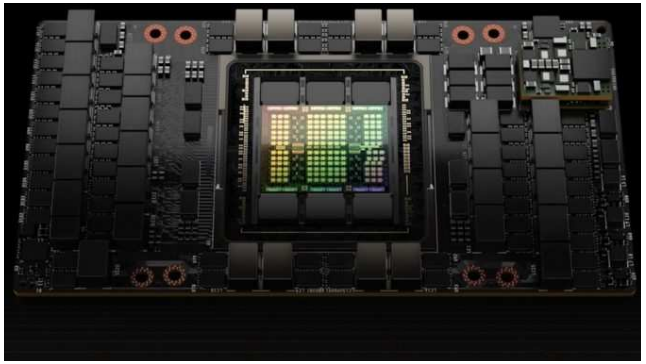
- GPU: Manycore
  - O(10^1)-O(10^2) cores
- More and more cores
  - Parallel computing

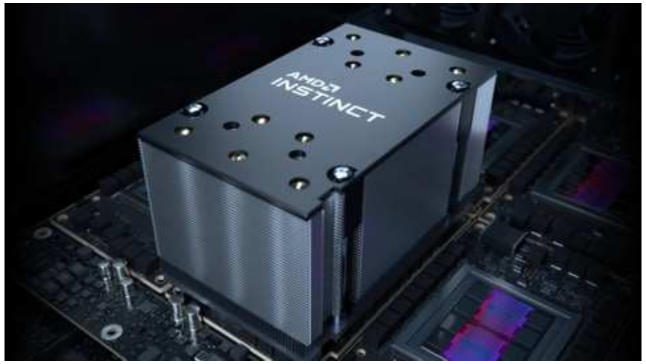
- **Odyssey: 48-cores/node**
  - **Fujitsu/ARM A64FX**

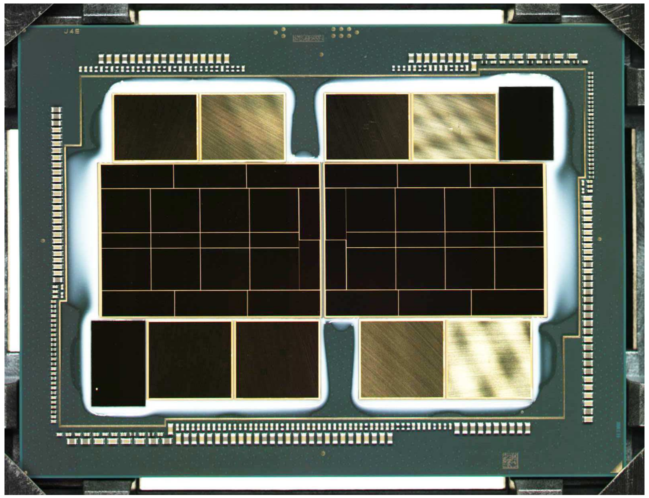


GPU/Accelerators

- GPU：Graphic Processing Unit
  - GPGPU: General Purpose GPU
  - O(10^2) cores
  - High Memory Bandwidth
  - (was) cheap
  - NO stand-alone operations
    - Host CPU needed
  - Programming: CUDA, OpenACC etc.
- NVIDIA, AMD, Intel …

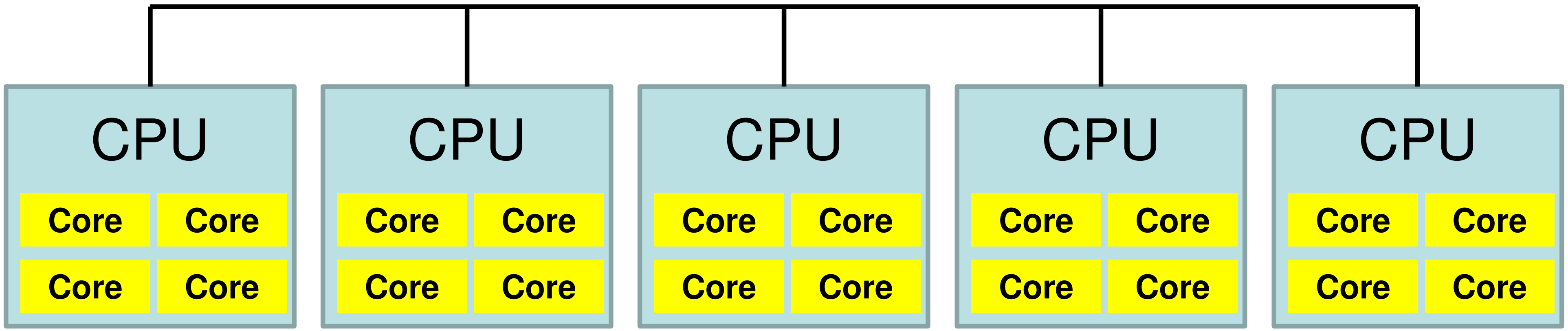




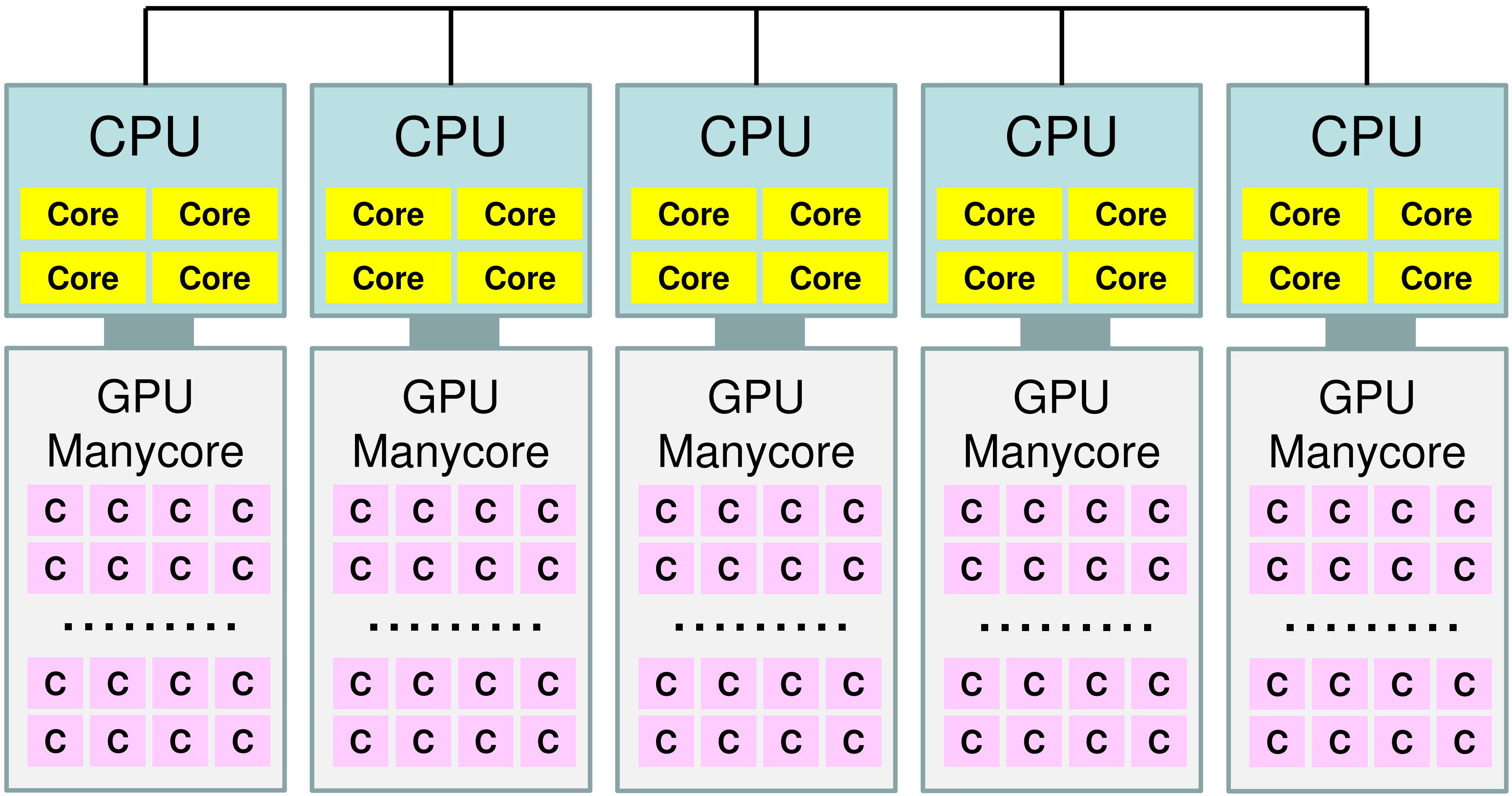


Parallel Supercomputers

Multicore CPU’s are connected through network



Supercomputers with Heterogeneous/Hybrid Nodes



Performance of Supercomputers

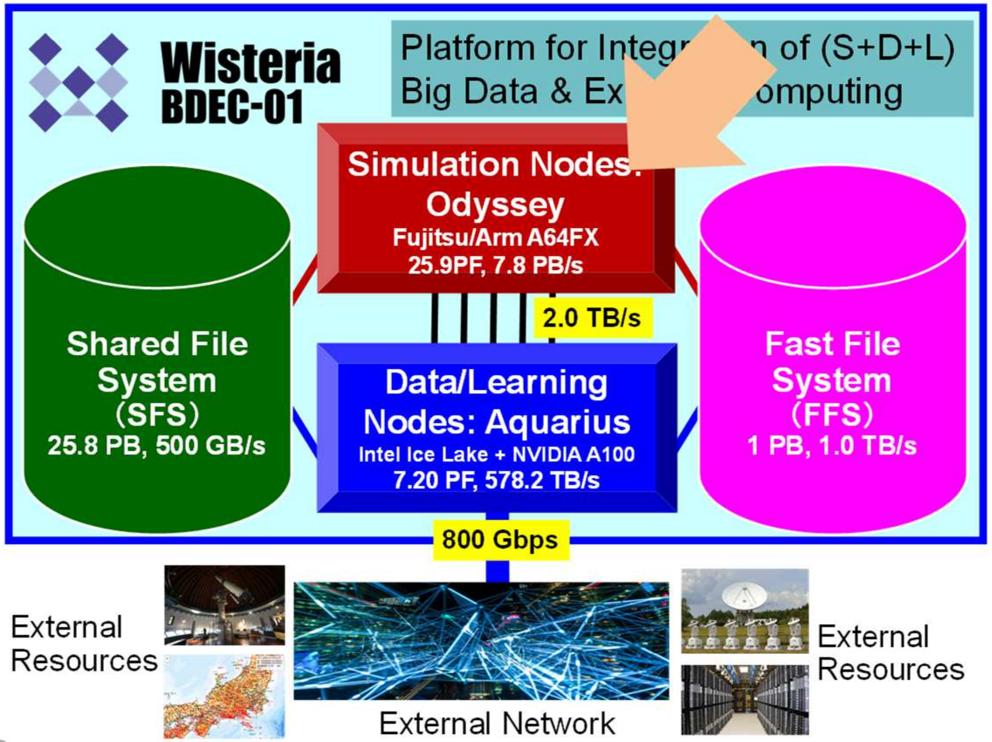
- Performance of CPU: Clock Rate
- FLOPS (Floating Point Operations per Second)
  – Real Number
- Recent Multicore CPU
  – 4-8 FLOPS per Clock
  – (e.g.) Peak performance of a core with 3GHz
    - $3 \times 10^9 \times 4$(or 8)=12(or 24)$\times 10^9$ FLOPS=12(or 24)GFLOPS

    - 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

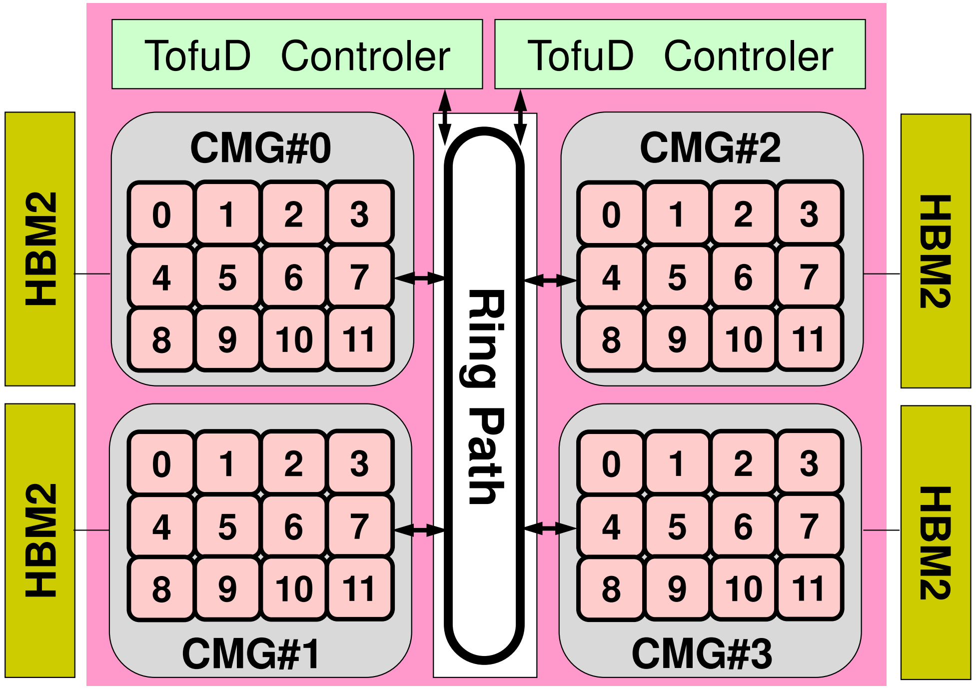
Wisteria/BDEC-01 (Odyssey)

The 1st BDEC System (Big Data & Extreme Computing)
Platform for Integration of (S+D+L)

- Operation starts on May 14, 2021
- 33.1 PF, 8.38 PB/sec by **Fujitsu**
  - ~4.5 MVA with Cooling, ~360m²
- **2 Types of Node Groups**
  - Hierarchical, Hybrid, Heterogeneous (h3)
  - **Simulation Nodes: Odyssey**
    - **Fujitsu PRIMEHPC FX1000 (A64FX), 25.9 PF**
      - **7,680 nodes (368,640 cores), Tofu-D**
      - **General Purpose CPU + HBM**
      - **Commercial Version of "Fugaku"**
  - Data/Learning Nodes: Aquarius
    - Data Analytics & AI/Machine Learning
    - Intel Xeon Ice Lake + NVIDIA A100, 7.2PF
      - 45 nodes (90x Ice Lake, 360x A100), IB-HDR
    - Some of the DL nodes are connected to external resources directly
- File Systems: SFS (Shared/Large) + FFS (Fast/Small)



A64FX Processor on Odyssey



Name	A64FX
Processor # (Core #)	1 (48+ 2or4 Assistant Cores)
Frequency	2.2 GHz
Peak Performance	3.3792 TFLOPS
Memory Size	32 GiB
Memory Bandwidth	1,024 GB/s
L1 Cache	64 KiB/core (Inst/Data)
L2 Cache	8 MiB/CMG

- 4 CMG's (Core Memory Group), 12 cores/CMG
  - 48 Cores/Node (Processor)
  - 2.2GHz × 32DP × 48= 3379.2 GFLOPS= 3.3792 TFLOPS
- NUMA Architecture (Non-Uniform Memory Access)
  - Each core of a CMG can access to the memory on other CMG's
  - Utilization of the local memory is more efficient

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)
- **“Fugaku” has been #1 from June 2020 to Nov.2021 (4 times)**
- **“Frotier” is the 1st Exaflop System in June 2022**
- Linpack
  - iPhone version is available

10^{19}= 10 ExaFlops

10^{18}= 1 ExaFlops

10^{17}=100 PetaFlops

10^{16}= 10 PetaFlops

10^{15}= 1 PetaFlops

10^{14}=100 TeraFlops

10^{13}= 10 TeraFlops

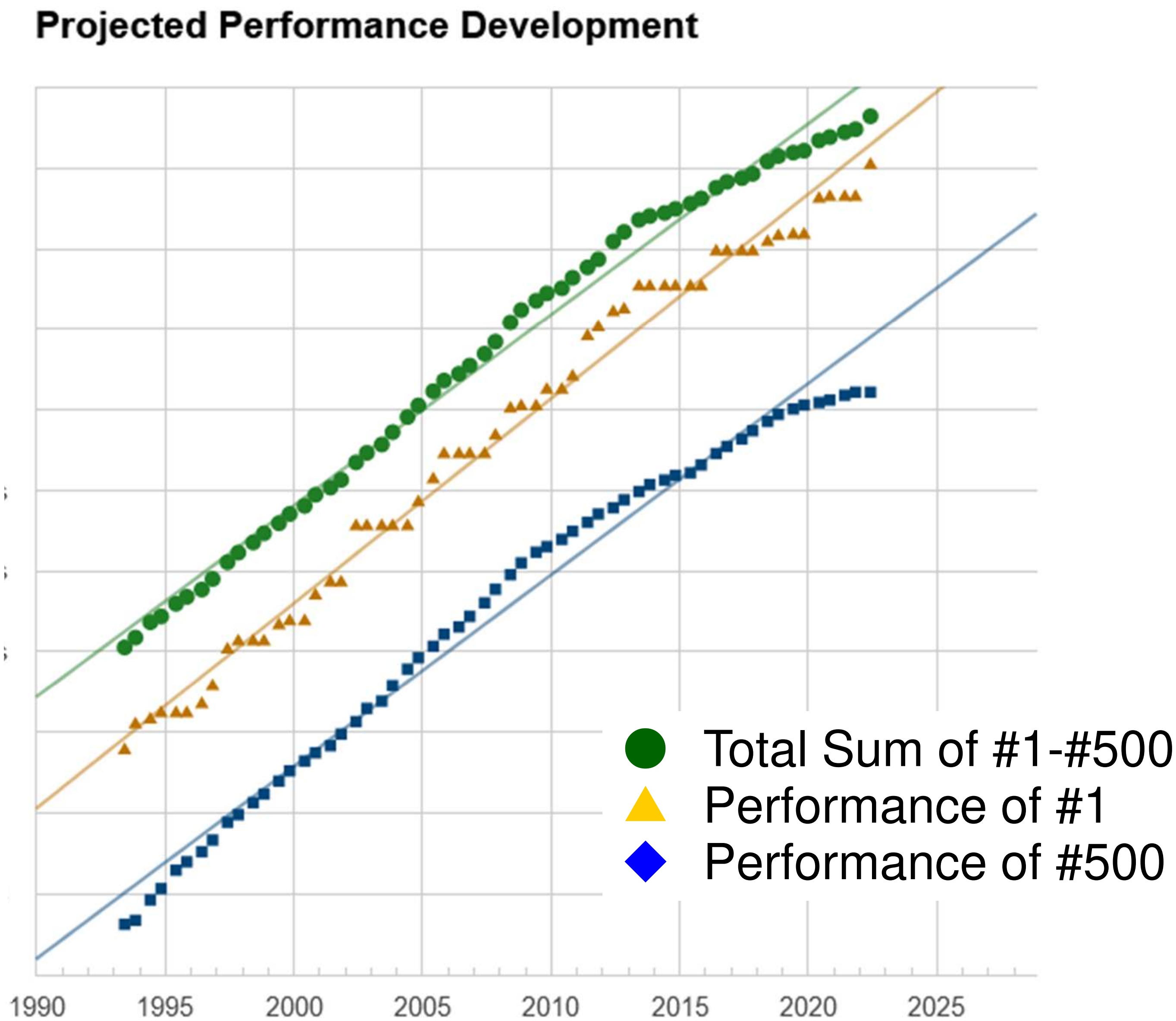
10^{12}= 1 TeraFlops

10^{11}=100 GigaFlops

10^{10}= 10 GigaFlops

10^{9}= 1 GigaFlops

10^{8}=100 MegaFlops



http://www.top500.org/

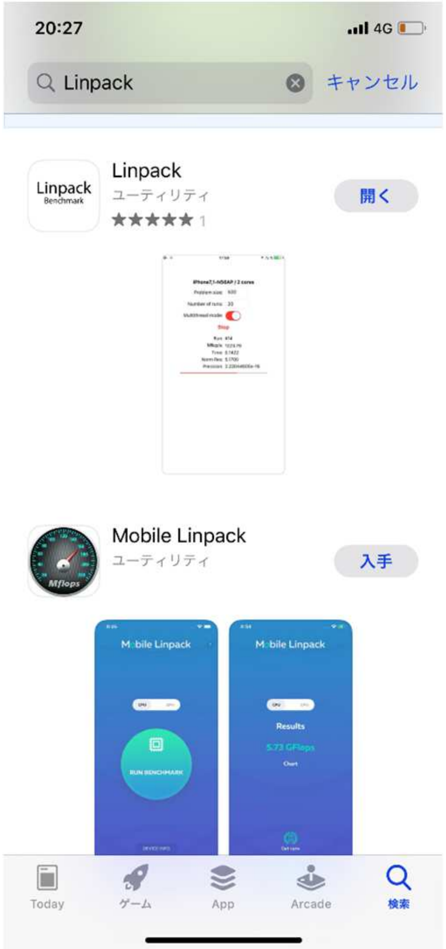
59th TOP500 List (June, 2022)

http://www.top500.org/

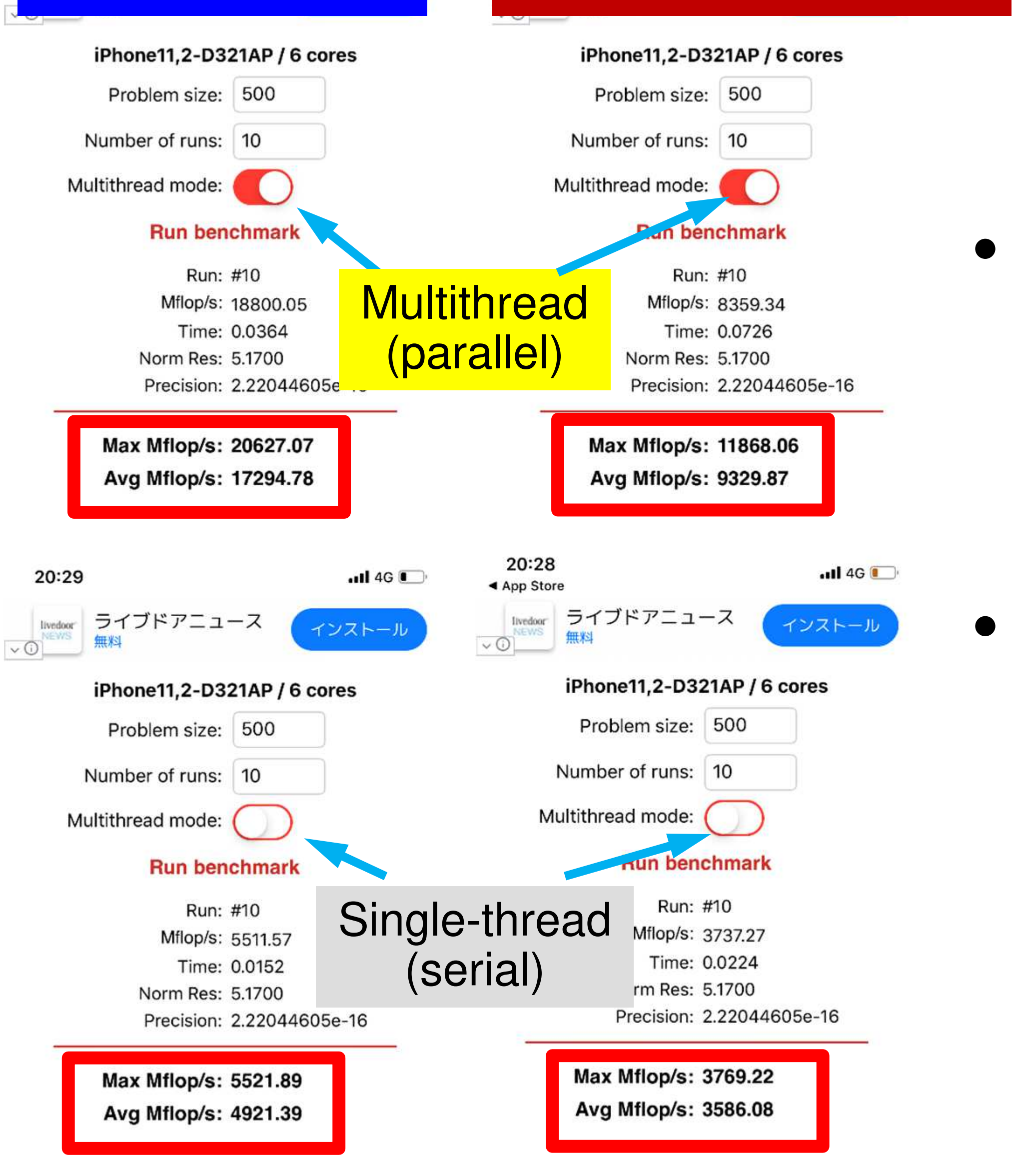
R_{max}: Performance of Linpack (TFLOPS)
R_{peak}: Peak Performance (TFLOPS),
Power: kW

	Site	Computer/Year Vendor	Cores	R_{max} (PFLOPS)	R_{peak} (PFLOPS)	Power (kW)
1	**Frontier, 2022, USA** DOE/SC/Oak Ridge National Laboratory	HPE Cray EX235a, AMD Optimized 3rd Gen. EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11	**8,730,112**	**1,102.00 (=1.102 EF)**	**1,685.65**	**21,100**
2	**Fugaku, 2020, Japan** R-CCS, RIKEN	Fujitsu PRIMEHPC FX1000, Fujitsu A64FX 48C 2.2GHz, Tofu-D	7,630,848	442,010 (= 442.0 PF)	537,212.0	**29,899**
3	**LUMI, 2022, Finland** EuroHPC/CSC	HPE Cray EX235a, AMD Optimized 3rd Gen. EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11	1,110,144	151.90	214.35	**2,942**
4	**Summit, 2018, USA** DOE/SC/Oak Ridge National Laboratory	IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR InfiniBand	2,414,592	148.60	200.79	**10,096**
5	**Sierra, 2018, USA** DOE/NNSA/LLNL	IBM Power System S922LC, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR InfiniBand	1,572,480	94.64	125.71	**7,438**
6	**Sunway TaihuLight, 2016, China** National Supercomputing Center in Wuxi	Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway	10,649,600	93.01	125.44	**15,371**
7	**Perlmutter, 2021, USA** DOE/NERSC/LBNL	HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10	761,856	70.87	93.75	**2,528**
8	**Selene, 2020, USA** NVIDIA	NVIDIA DGX A100 SuperPOD, AMD EPYC 7742 64C 2.25GHz, NVIDIA GA100, Mellanox Infiniband HDR	555,520	63.46	79.22	**2,646**
9	**Tianhe-2A, 2018, China** National Super Computer Center in Guangzhou	TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHz, TH Express-2, Matrix-2000	4,981,760	61.44	100.68	**18,482**
10	**Adastra, 2022, France** GENCI-CINES	HPE Cray EX235a, AMD Optimized 3rd Gen. EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11	319,072	46.10	61.61	**921**

Linpack on My iPhone XS

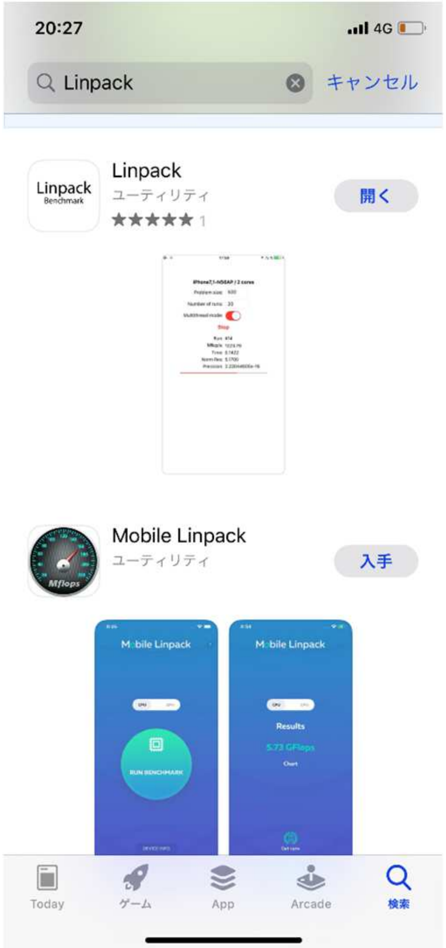


Normal Mode | **Low-Power Mode**

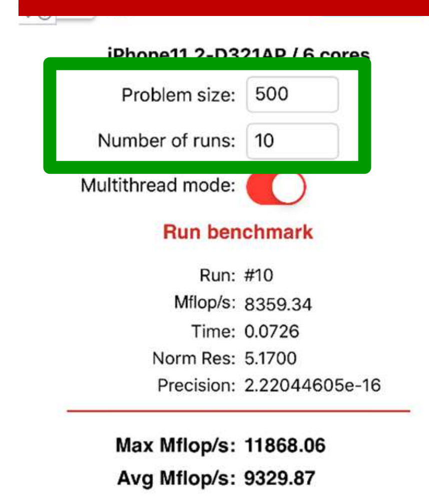


- Performance of my iPhone XS is about 20,000 Mflops
  - Fugaku: 3.83 Tflops
- Cray-1S
  - Supercomputer of my company in 1985 with 80 Mflops
  - I do not know the price, but we had to pay 10 USD for 1 sec. computing !

Linpack on My iPhone XS



Normal Mode | **Low-Power Mode**



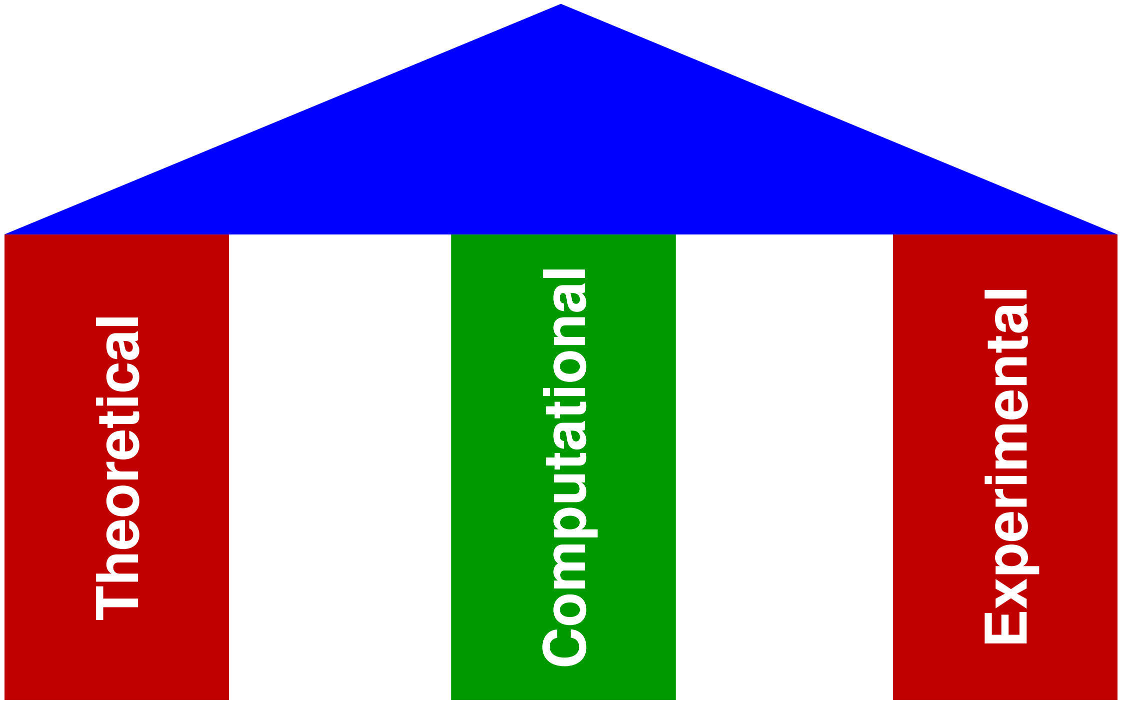
- You can change Problem size, and # of runs.
  - "Size=500" means linear equations Ax=b with 500 unknowns are solved
- Actually, problem size affects performance of computing so much !!

Benchmarks

- TOP 500 (Linpack, HPL(High Performance Linpack))
  - Direct Linear Solvers, FLOPS rate
  - Regular Dense Matrices, Continuous Memory Access
  - Computing Performance
- HPCG
  - Preconditioned Iterative Solvers, FLOPS rate
  - Irregular Sparse Matrices derived from FEM Applications with Many “0” Components
    - Irregular/Random Memory Access,
    - Closer to “Real” Applications than HPL
  - Performance of Memory, Communications
- Green 500
  - FLOPS/W rate for HPL (TOP500)

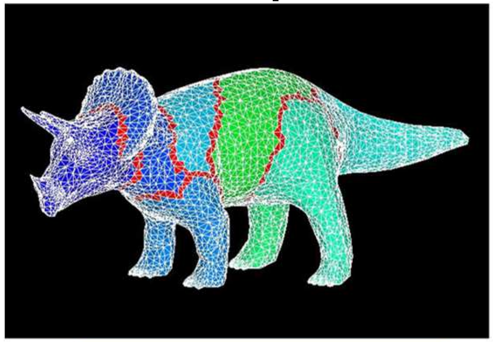
Computational Science
The 3rd Pillar of Science

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

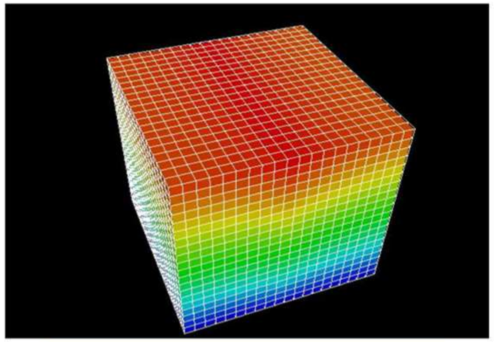


Methods for Scientific Computing

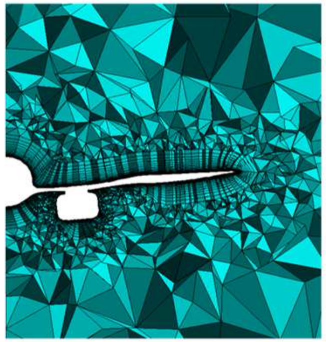
- Numerical solutions of PDE (Partial Diff. Equations)
- Grids, Meshes, Particles
  - Large-Scale Linear Equations
  - Finer meshes provide more accurate solutions



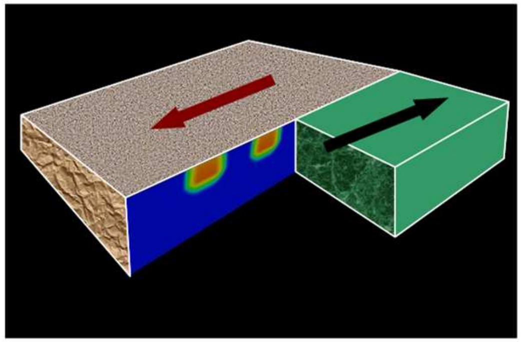
有限要素法
Finite Element Method
FEM



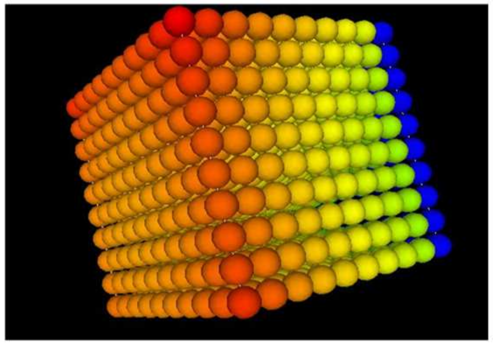
差分法
Finite Difference Method
FDM



有限体積法
Finite Volume Method
FVM



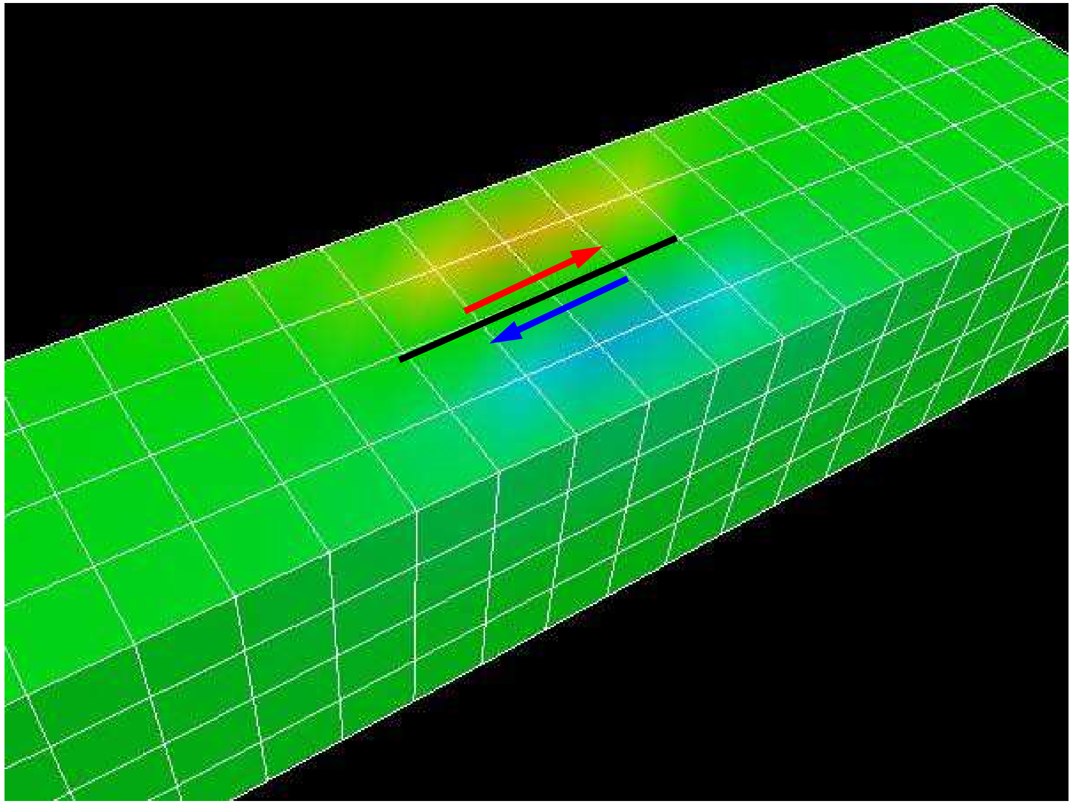
境界要素法
Boundary Element Method
BEM

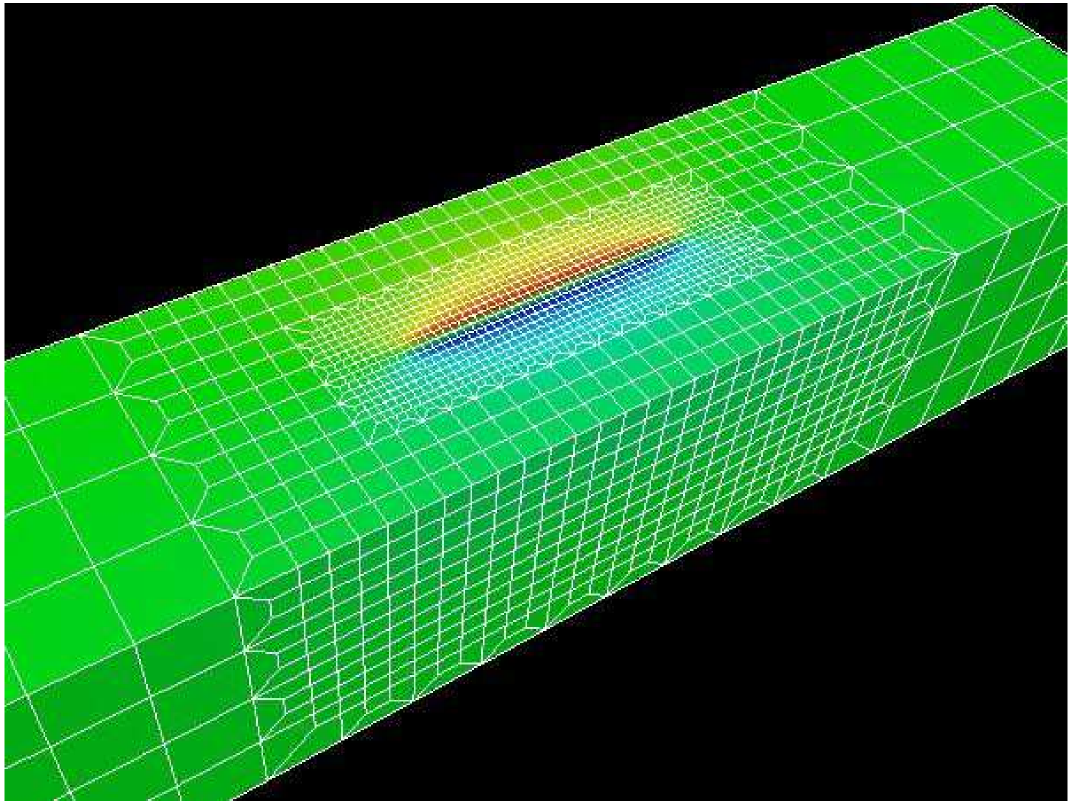


個別要素法
Discrete Element Method
DEM

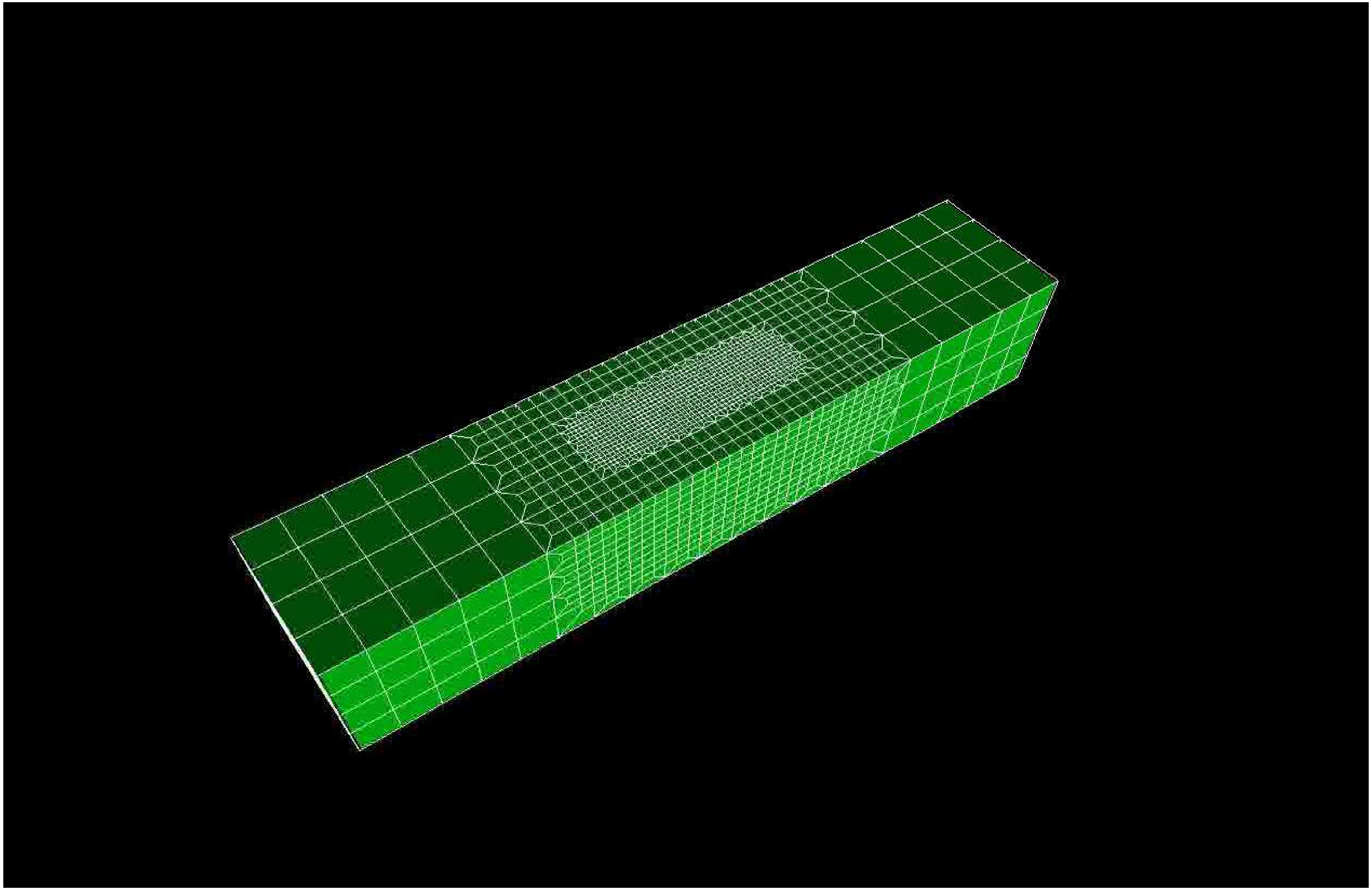
3D Simulations for Earthquake Generation Cycle San Andreas Faults, CA, USA

Stress Accumulation at Transcurrent Plate Boundaries

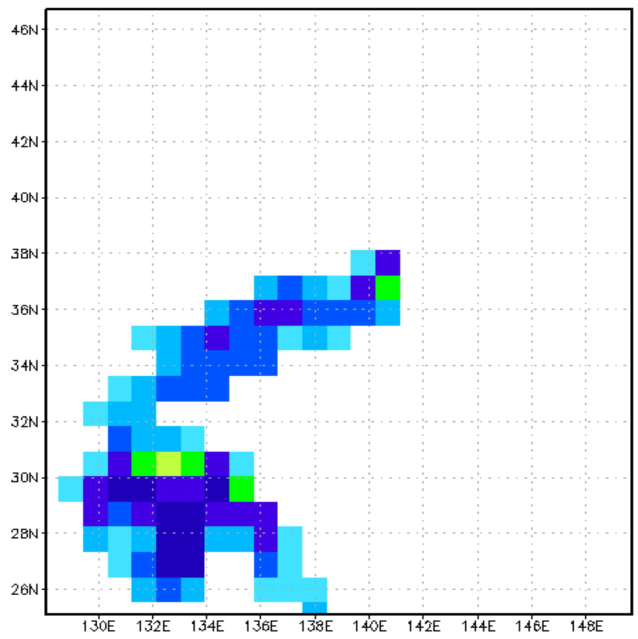




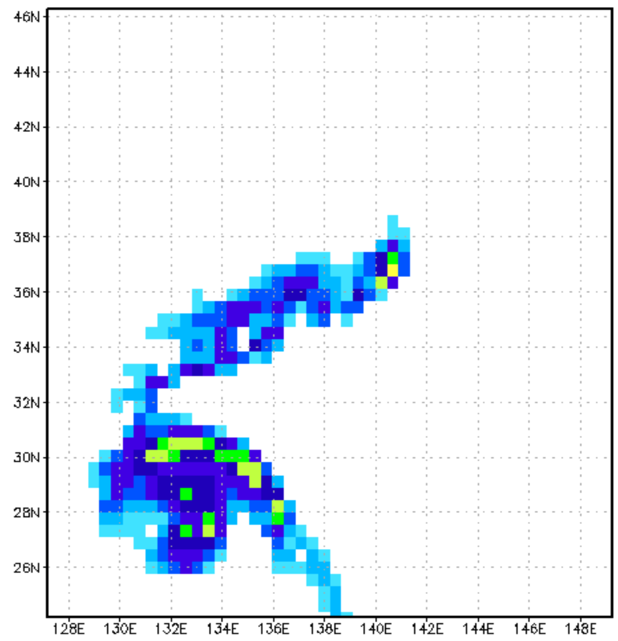
Adaptive FEM: High-resolution needed at meshes with large deformation (large accumulation)



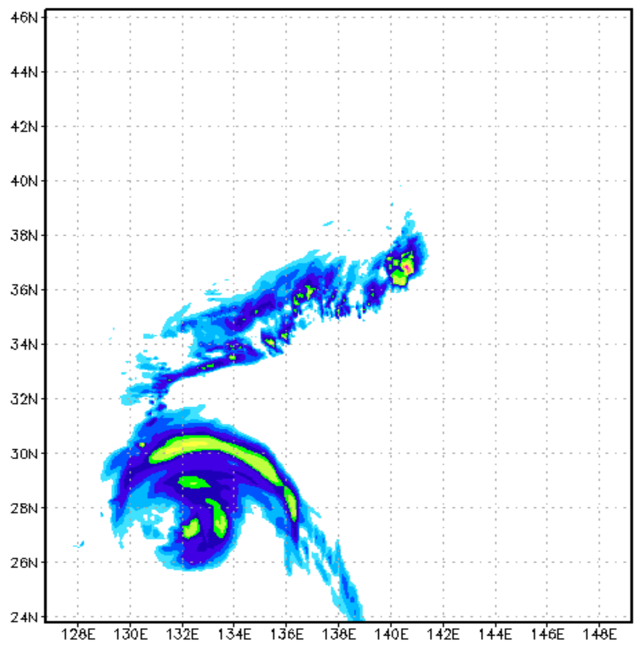
Typhoon Simulations by FDM
Effect of Resolution



Δh=100km



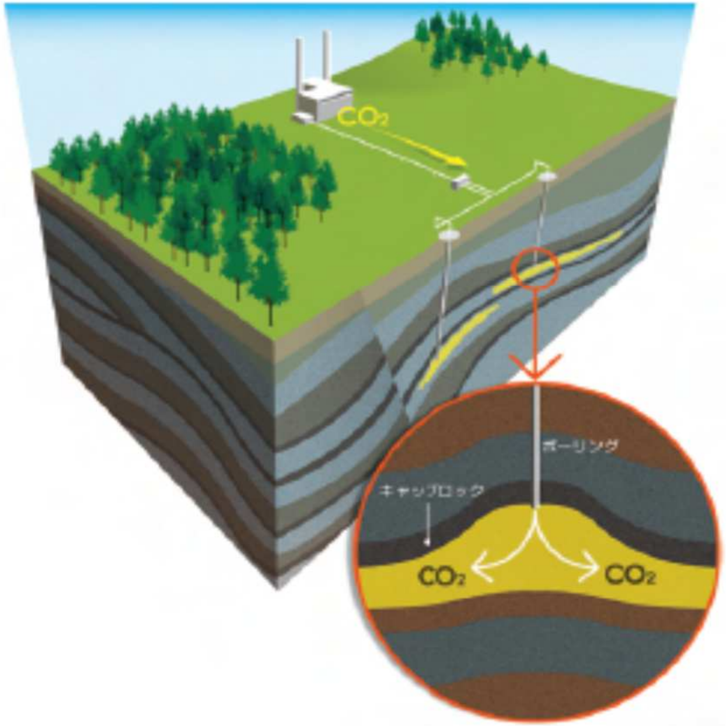
Δh=50km



Δh=5km

[JAMSTEC]

Simulation of Geologic CO_2 Storage



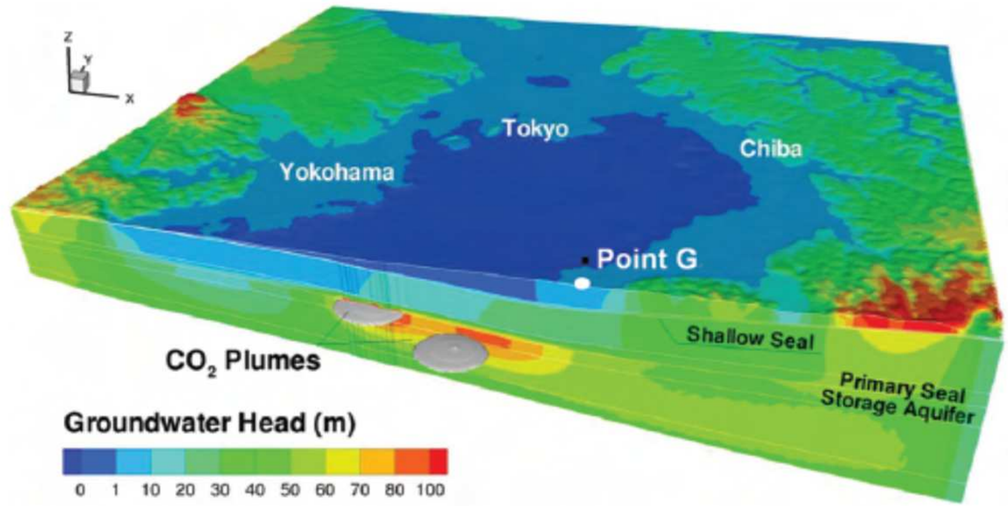
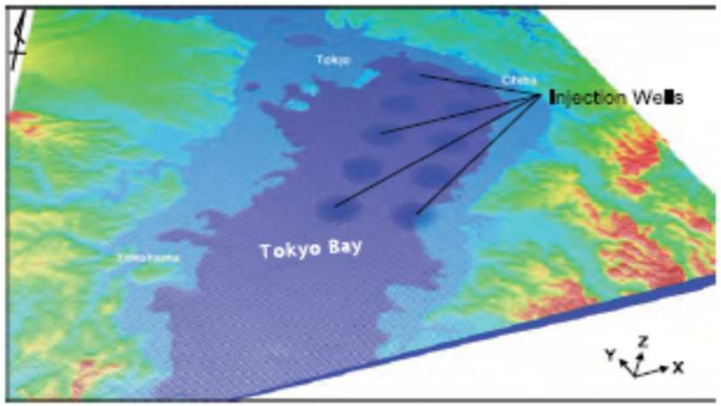
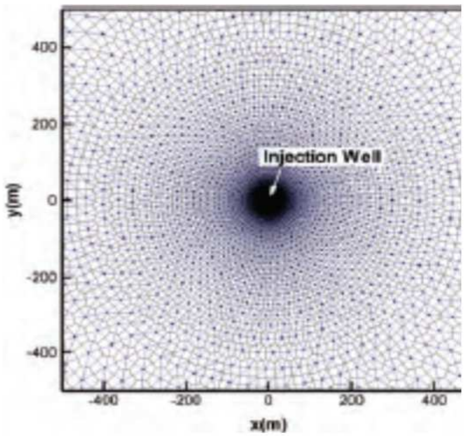
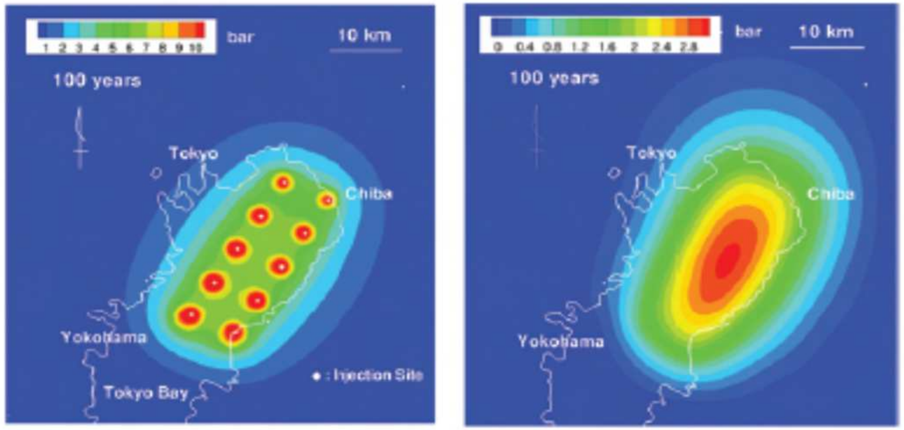


図 -4　CO_2 圧入後の地下水圧（全水頭換算）の分布（100 年後）







(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)
  - JAMSTEC (Earth Simulator Center) (Software, Hardware)
  - NEC (Software, Hardware)
- 2010 Japan Geotechnical Society (JGS) Award

Science

Modeling

Algorithm

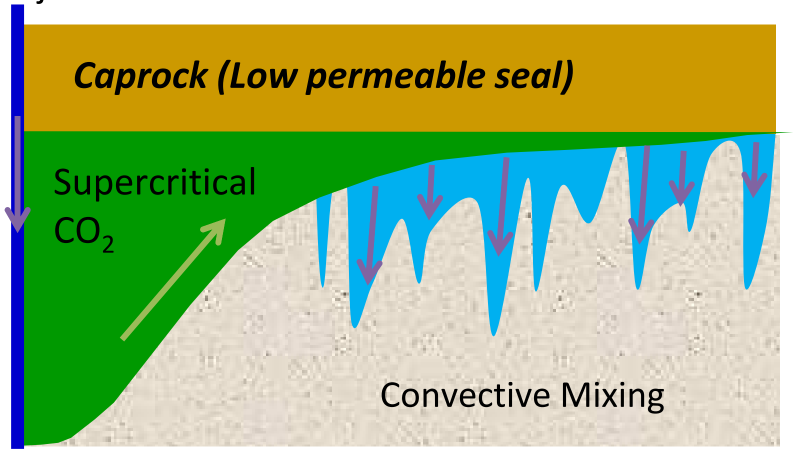
Software

Hardware

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 II (NEX SX9, JAMSTEC, 130 TFLOPS)
  - Oakleaf-FX (Fujitsu PRIMEHP FX10, U.Tokyo, 1.13 PFLOPS

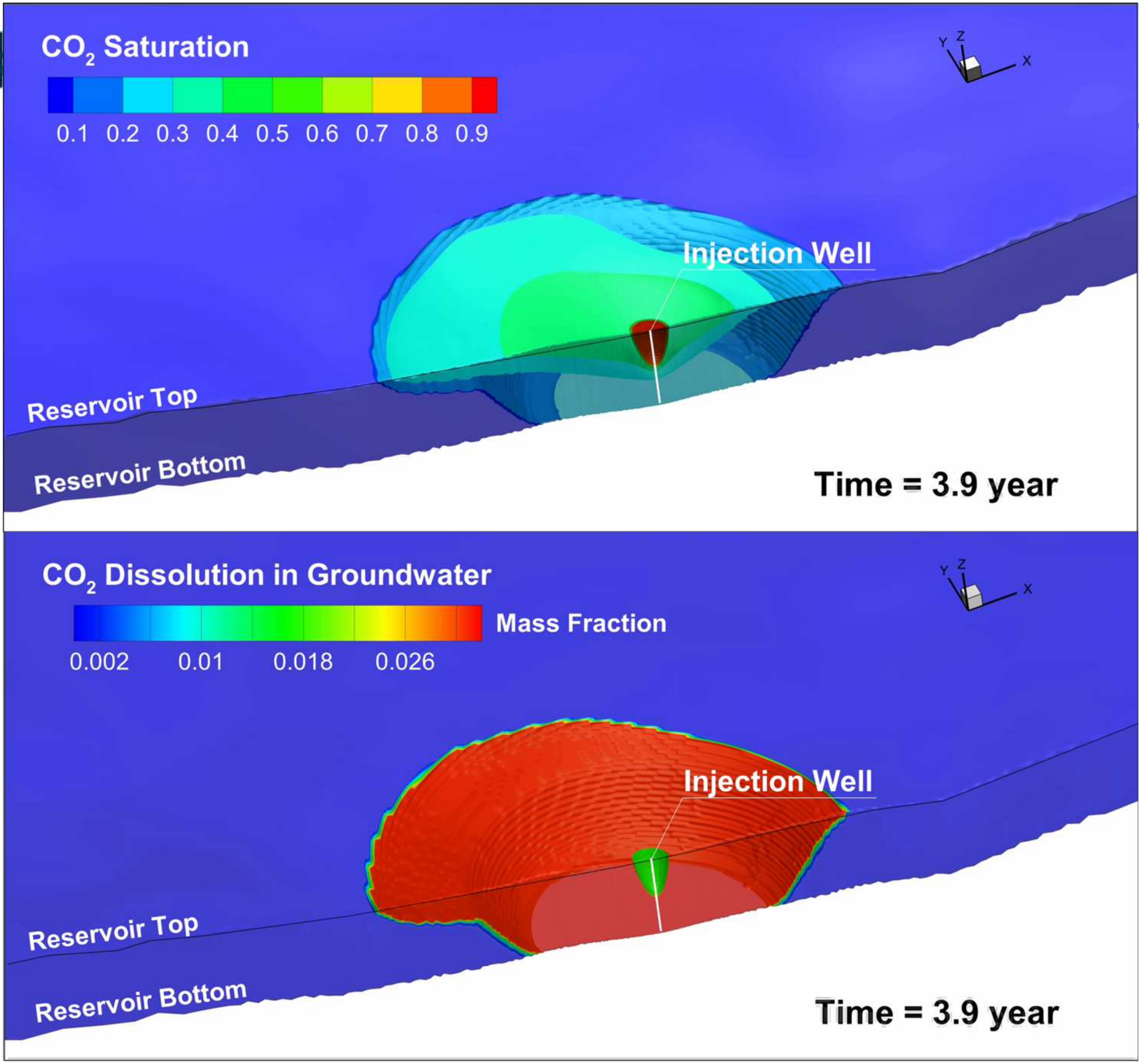
Diffusion-Dissolution-Convection Process



- **Buoyant $scCO_2$ overrides onto groundwater**
- **Dissolution of CO_2 increases water density**
- **Denser fluid laid on lighter fluid**
- **Rayleigh-Taylor instability invokes convective mixing of groundwater**

The mixing significantly enhances the CO_2 dissolution into groundwater, resulting in more stable storage

Preliminary 2D simulation (Yamamoto et al., GHGT11) [Dr. Hajime Yamamoto, Taisei]



Density convections for 1,000 years:

Flow Model

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

Reservoir Condition

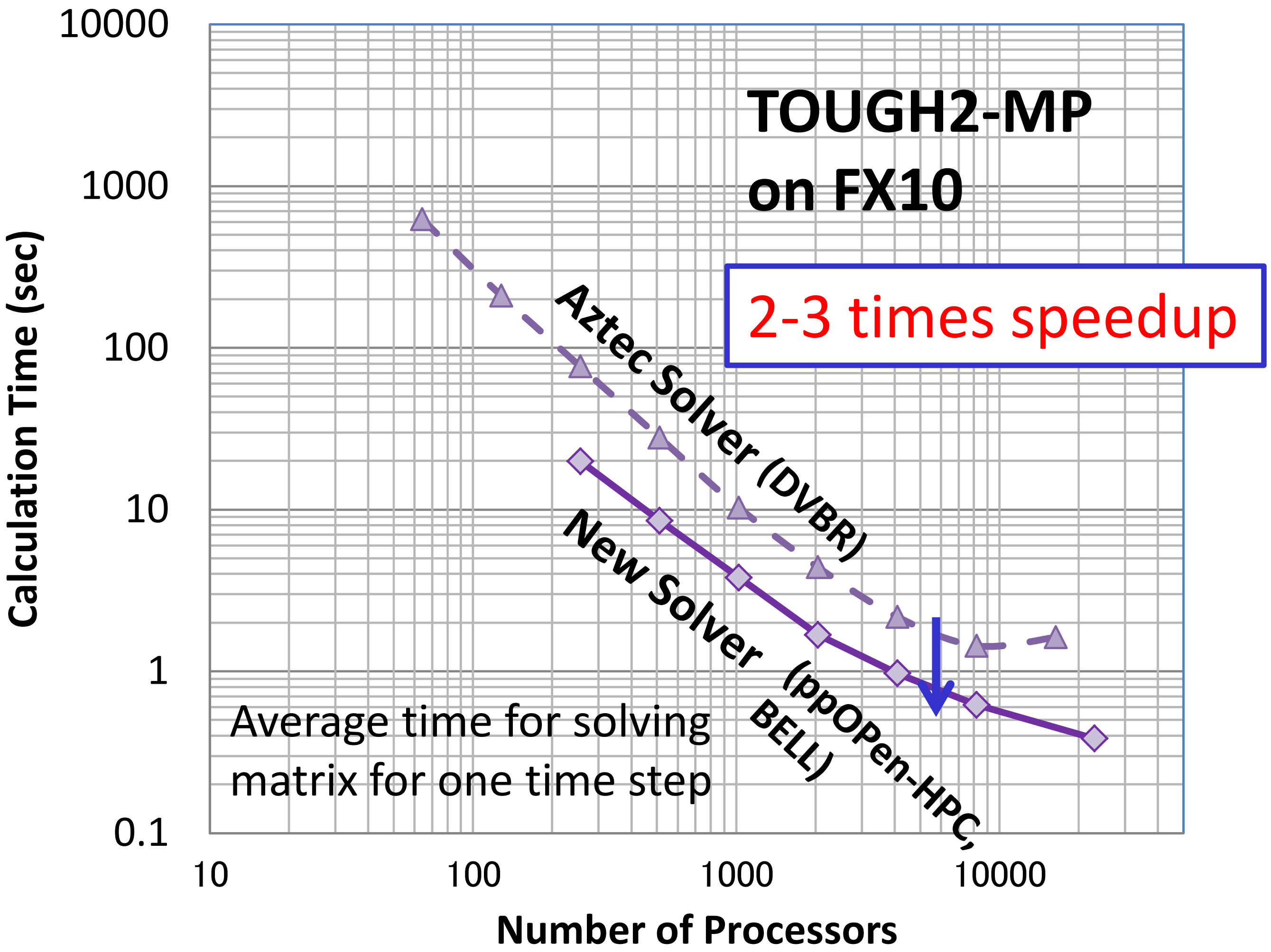
- Permeability: 100 md
- Porosity: 20%
- Pressure: 3MPa
- Temperature: 100°C
- Salinity: 15wt%

[Dr. Hajime Yamamoto, Taisei]

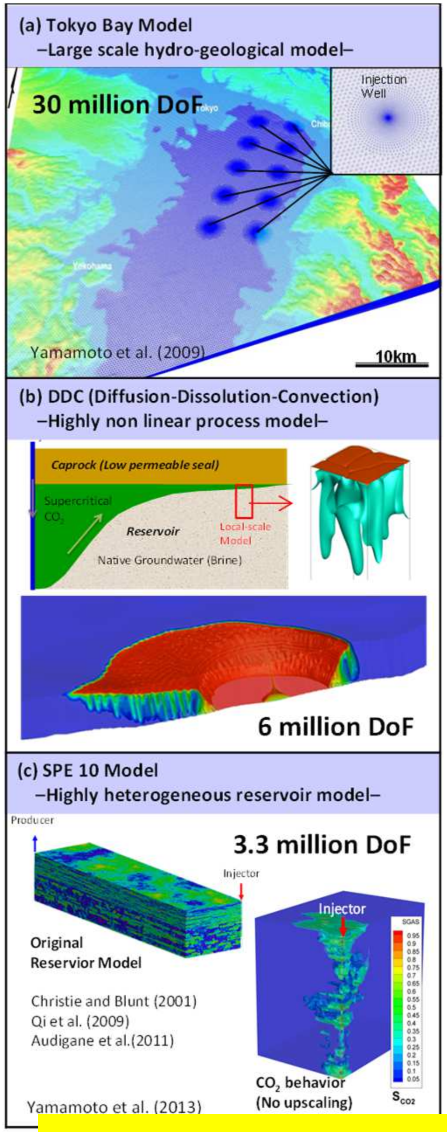
- The meter-scale fingers gradually developed to larger ones in the field-scale model
- Huge number of time steps (> 10^5) were required to complete the 1,000-yrs simulation
- Onset time (10-20 yrs) is comparable to theoretical (linear stability analysis, 15.5yrs)

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

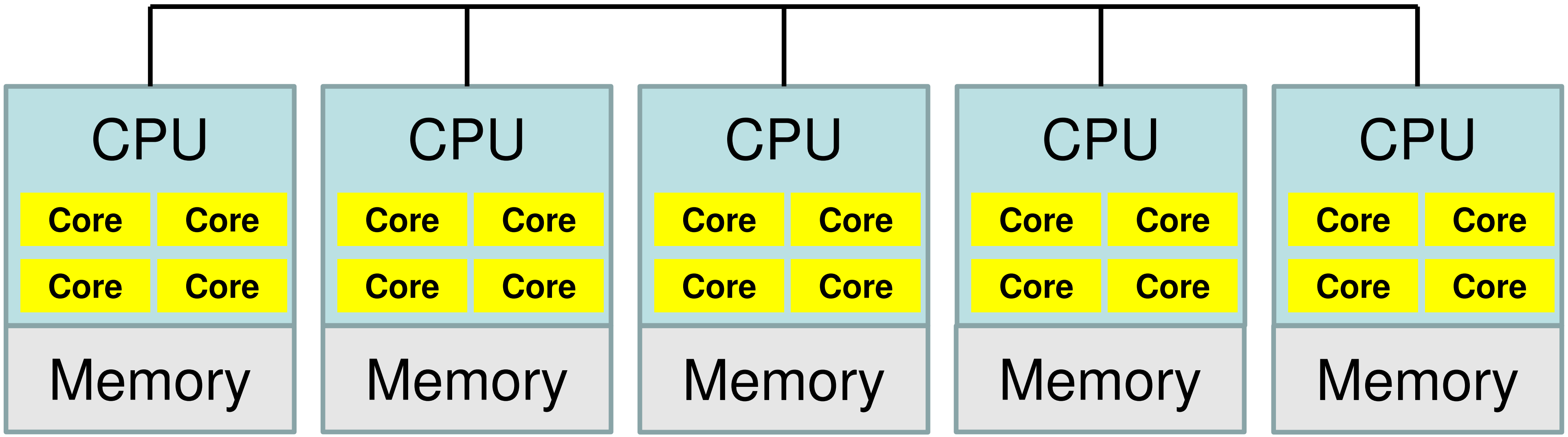
Motivation for Parallel Computing, again

- 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

Our Current Target: Multicore Cluster

Multicore CPU's are connected through network

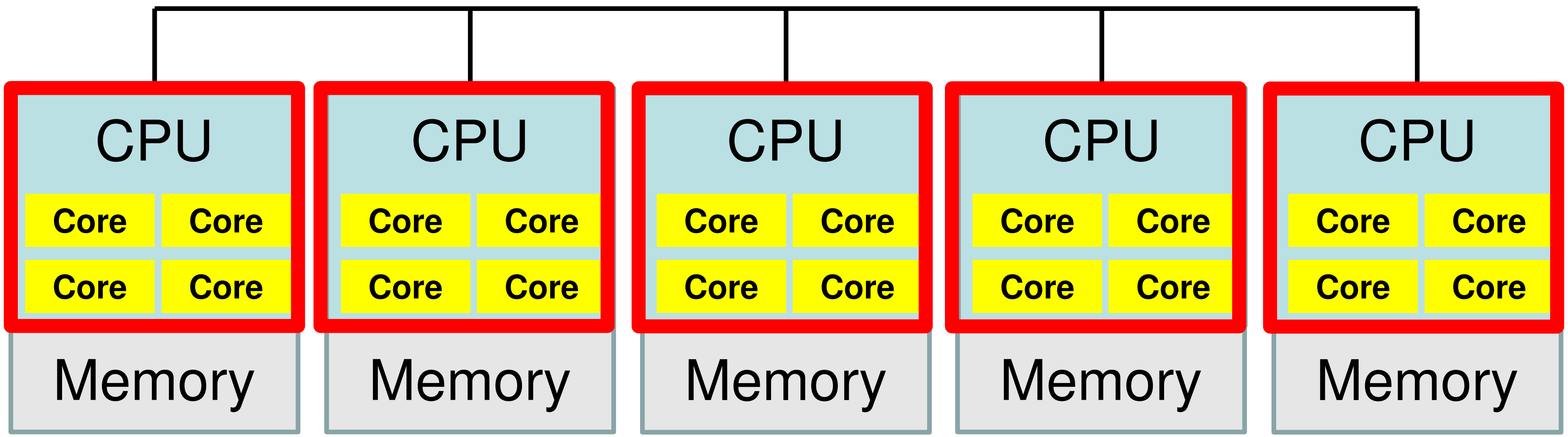


- OpenMP
  - ✓ Multithreading
  - ✓ Intra Node (Intra CPU)
  - ✓ Shared Memory

- MPI
  - ✓ Message Passing
  - ✓ Inter Node (Inter CPU)
  - ✓ Distributed Memory

Our Current Target: Multicore Cluster

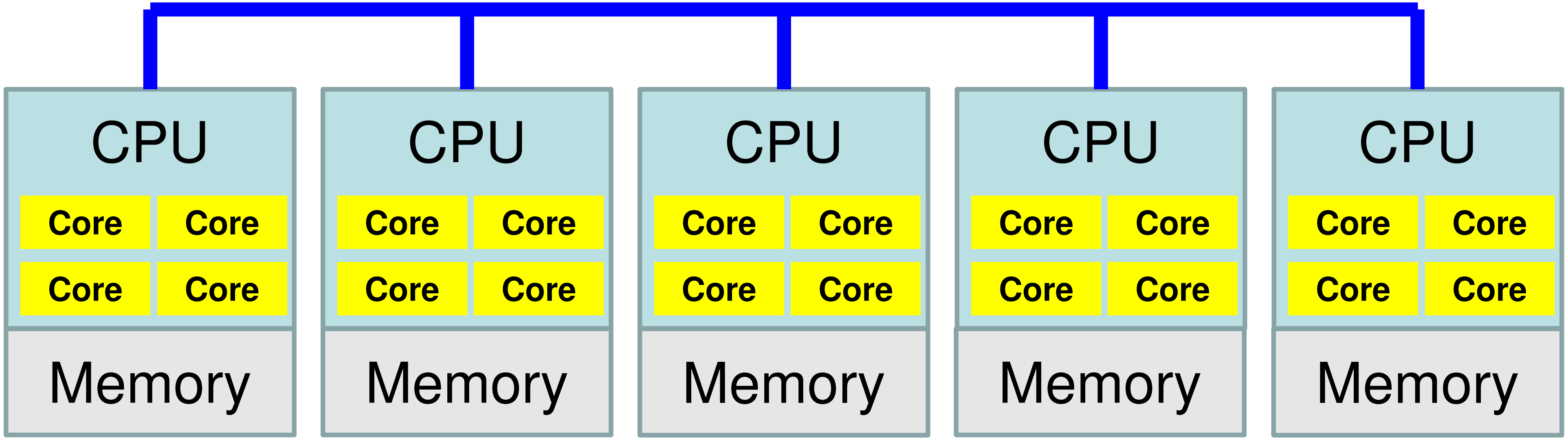
Multicore CPU's are connected through network



- OpenMP
  - ✓ Multithreading
  - ✓ Intra Node (Intra CPU)
  - ✓ Shared Memory
- MPI
  - ✓ Message Passing
  - ✓ Inter Node (Inter CPU)
  - ✓ Distributed Memory

Our Current Target: Multicore Cluster

Multicore CPU’s are connected through network

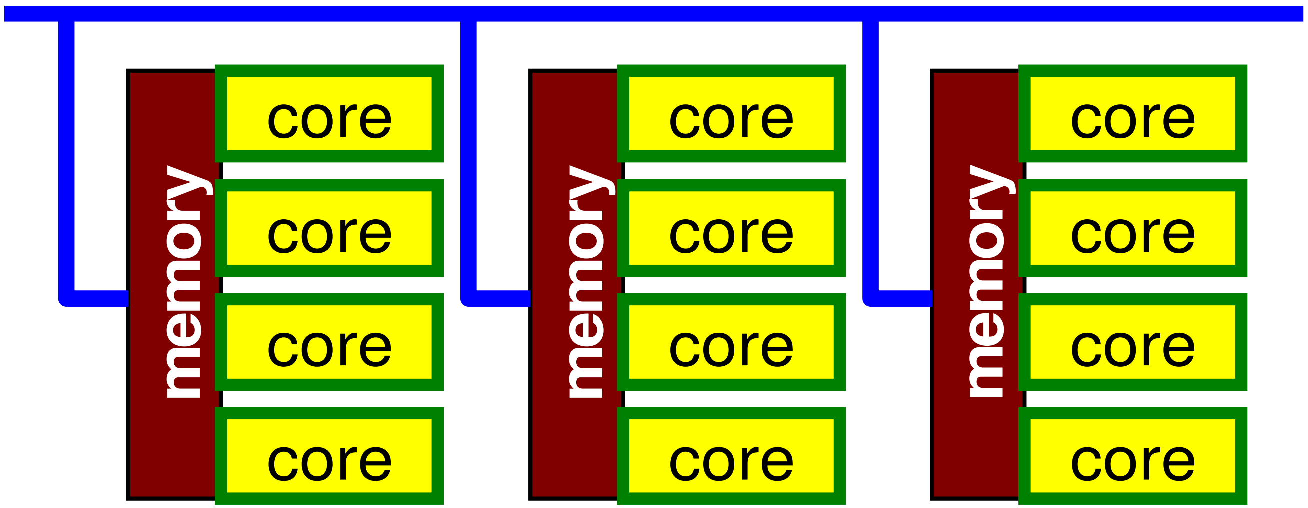


- OpenMP
  - ✓ Multithreading
  - ✓ Intra Node (Intra CPU)
  - ✓ Shared Memory
- MPI (after October)
  - ✓ Message Passing
  - ✓ Inter Node (Inter CPU)
  - ✓ Distributed Memory

Flat MPI vs. Hybrid

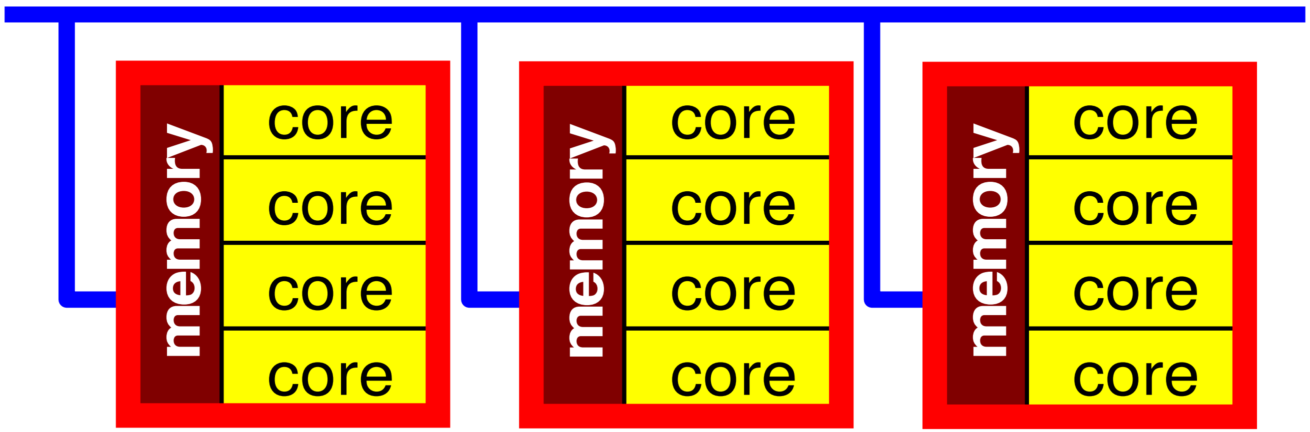
Flat-MPI : Each Core -> Independent

- MPI only
- Intra/Inter Node



Hybrid : Hierarchal Structure

- OpenMP
- MPI



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
```

Information of this Class

- Instructor
  - Kengo Nakajima (Information Technology Center)
    - Kashiwa II Campus
    - e-mail: nakajima(at)cc.u-tokyo.ac.jp
    - Zoom meeting is possible up on request
- Schedule
  - Every Wednesday
  - 08:30-10:15
  - **http://nkl.cc.u-tokyo.ac.jp/22w/**
- Practice
  - Time for exercise
- Lecture Room
  - Online
- Slack channel will be prepared

	Date	Time	Title
1	Oct.05(W)	0830-1015	Introduction, Introduction to FEM (1/2)
2	Oct.12(W)	0830-1015	Introduction to FEM (2/2), 1D/3D FEM (1/4)
3	Oct.19(W)	0830-1015	1D/3D FEM (2/4)
4	Oct.26(W)	0830-1015	1D/3D FEM (3/4)
5	Nov.02 (W)	0900-1015	1D/3D FEM (4/4)
6	Nov.09 (W)	0830-1015	Introduction to Parallel FEM, Login to Odyssey, MPI (1/5)
7	**Nov.16 (W)**	**0830-1015**	**MPI (2/5) (Video Recorded)**
	Nov.23 (W)		**National Holiday (No Class)**
8	Nov.30 (W)	0830-1015	MPI (3/5)
9	Dec.07 (W)	0830-1015	Report S1, MPI (4/5)
10	Dec.14 (W)	0830-1015	MPI (5/5)
11	Dec.21 (W)	0830-1015	Report S2, Parallel FEM (1/4)
12	Jan.04 (W)	0830-1015	Parallel FEM (2/4)
13	Jan.11 (W)	0830-1015	Parallel FEM (3/4)
14	Jan.18 (M)	0830-1015	Parallel FEM (4/4), Hybrid OpenMP/MPI (1/2)
15	Jan.25 (W)	0830-1015	Hybrid OpenMP/MPI (2/2)
16	Feb.01 (W)	0830-1015	Q/A

Prerequisites

- Knowledge and experiences in fundamental methods for numerical analysis (e.g. Gaussian elimination, SOR)
- Knowledge and experiences in Unix/Linux
  - Essential for using Supercomputers
  - Google “Introduction to Unix”, “Introduction to Linux”
  - **`cd, ls, cp, mv, rm, scp`**
  - Editor: **`vi, emacs`** etc.
- Experiences in programming using FORTRAN or C
- Account for Educational Campuswide Computing System (ECC System) should be obtained in advance:
  - http://www.ecc.u-tokyo.ac.jp/ENGLISH/index-e.html
  - http://www.ecc.u-tokyo.ac.jp/user.html

Grading by Reports ONLY

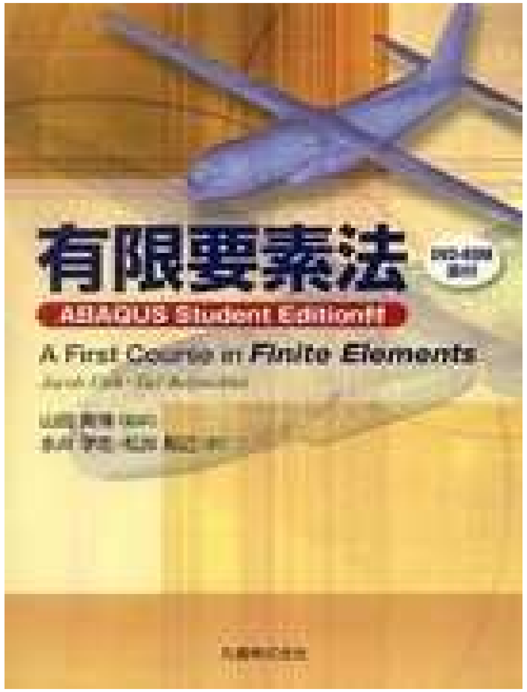
- MPI (Collective Communication) (S1)
- MPI (1D Parallel FEM) (S2)

- Sample solutions will be available
- Deadline: January 25th (Wed), 2023, 17:00
  - ITC-LMS

Homepage

- http://nkl.cc.u-tokyo.ac.jp/22w/
  - General information is available
  - No hardcopy of course materials are provided (Please print them by yourself)

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 included
- Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Dover, 2000

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.

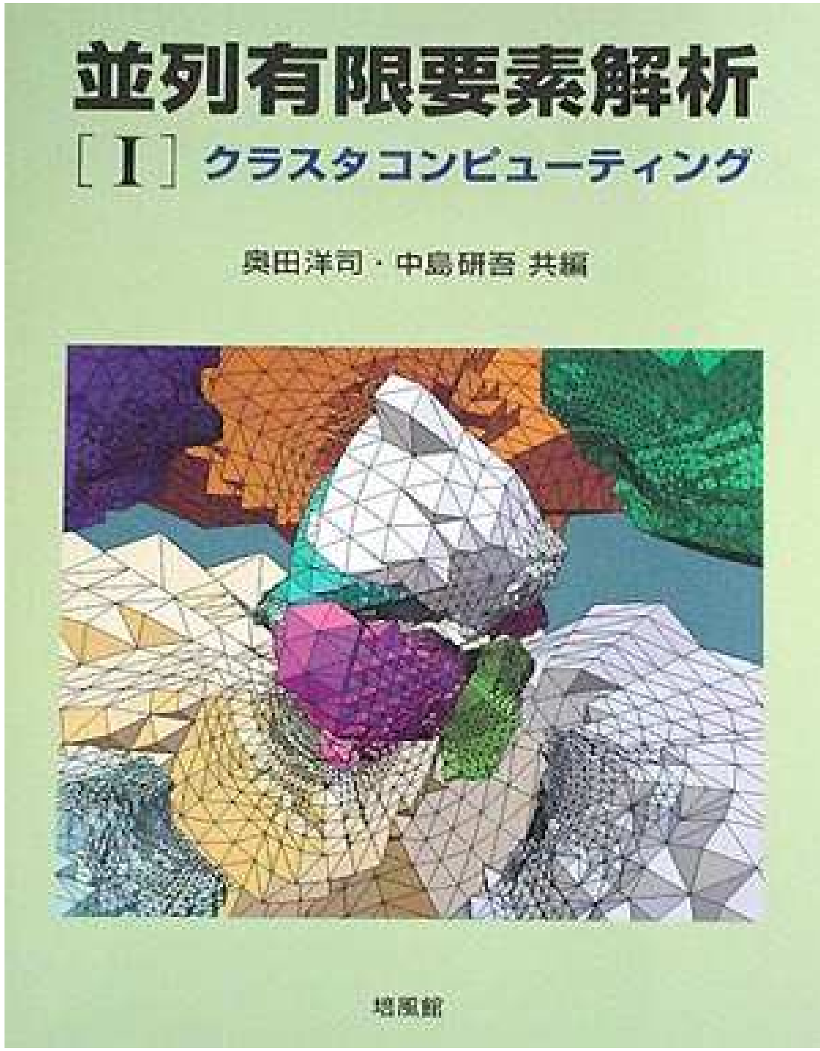
- Smith, I. 他「Programming the Finite Element Method (4th edition)」，Wiley.

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。



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

Technical Issues: Future of Supercomputers

- Power Consumption
- Reliability, Fault Tolerance, Fault Resilience
- Scalability (Parallel Performance)

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