Introduction
Overview of the Class

Kengo Nakajima
Information Technology Center
The University of Tokyo

- 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 is last course given my me**

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 Reedbush-U system at the University of Tokyo.

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 中島研吾　(1/2)

- Current Position
  - Professor, Supercomputing Research Division, Information Technology Center, The University of Tokyo（情報基盤センター）
    - Department of Mathematical Informatics, Graduate School of Information Science & Engineering, The University of Tokyo（情報理工・数理情報学）
    - Department of Electrical Engineering and Information Systems, Graduate School of Engineering, The University of Tokyo（工・電気系工学）
  - Deputy Director, RIKEN CCS (Center for Computational Science) (Kobe) (20%) (2018.Apr.-)
- 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
  - 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 (2009-2017), 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

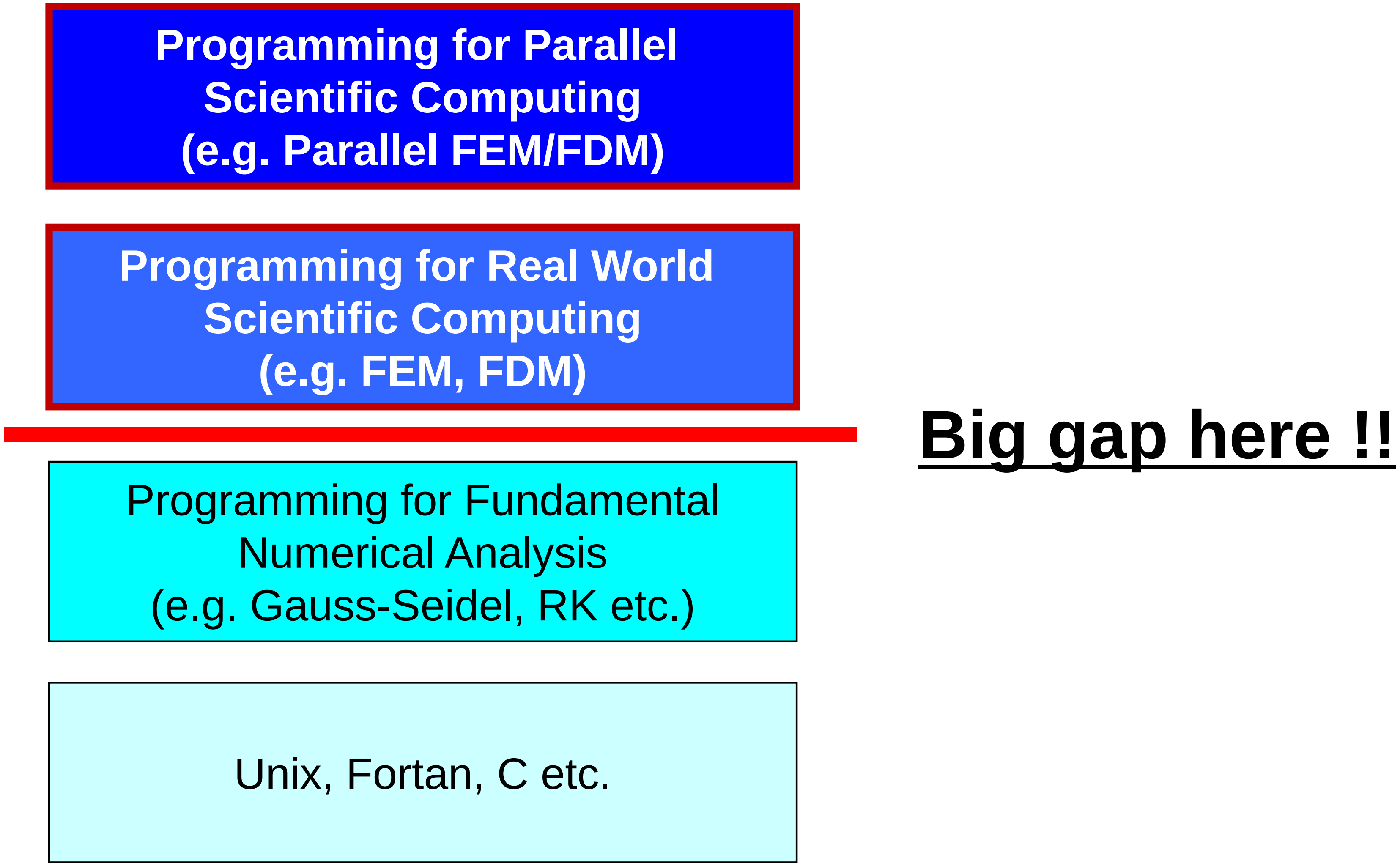
Hardware

- 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)

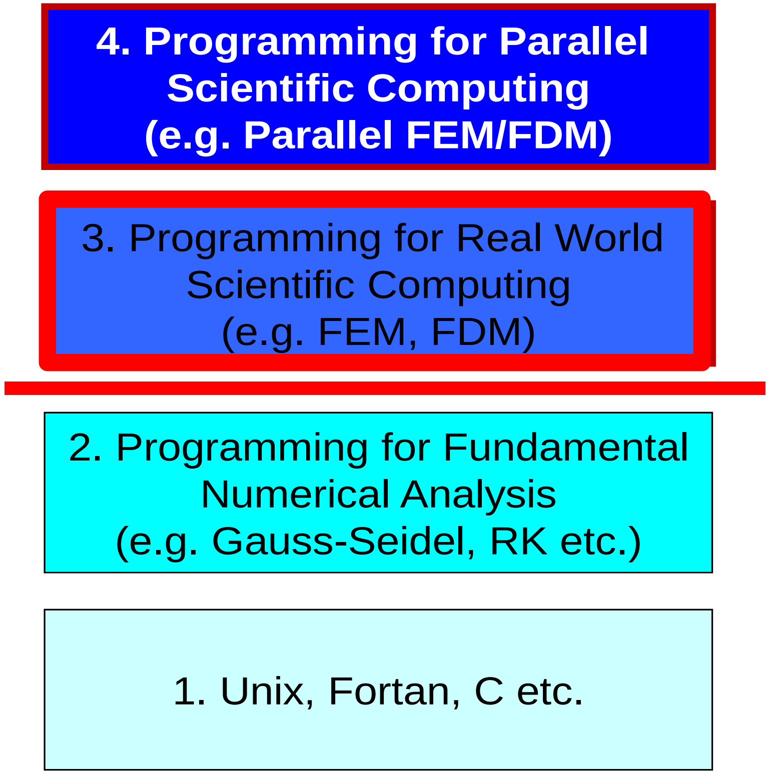
Programming for Real World Scientific Computing (e.g. FEM, FDM)

Big gap here !!

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

Unix, Fortan, C etc.

The third step is important !



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

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

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

1. Unix, Fortan, C etc.

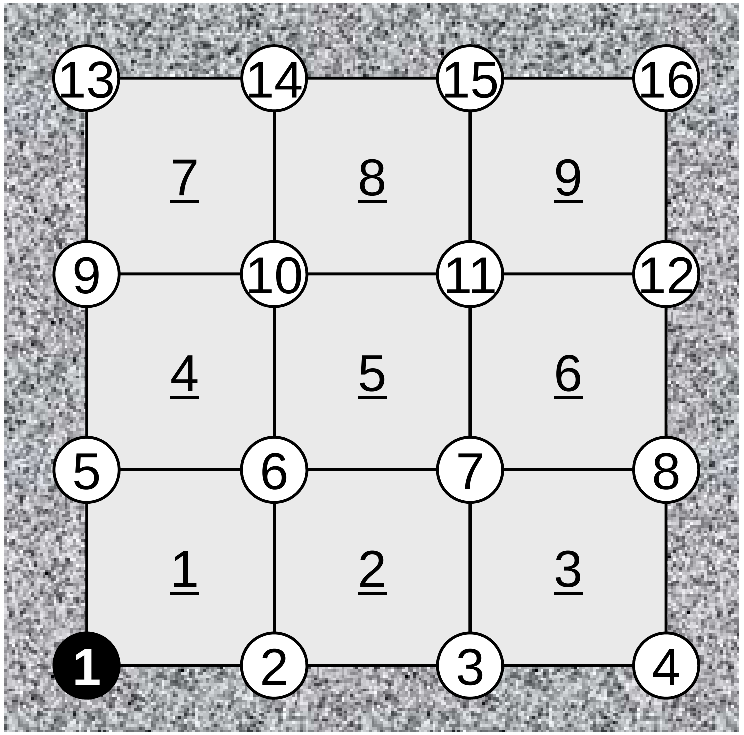
- How to parallelize applications ?
  - How to extract parallelism ?
  - If you understand methods, algorithms, and implementations of the original code, it's easy.
  - "Data-structure" is important
- How to understand the code ?
  - Reading the application code !!
  - It seems primitive, but very effective.
  - 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.

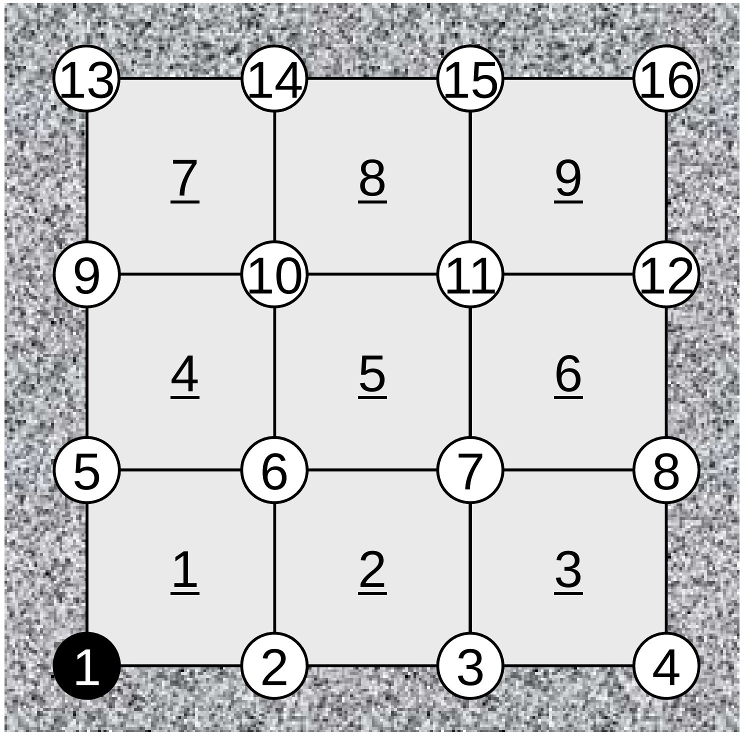
$\{\phi\}$: T at each vertex

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

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

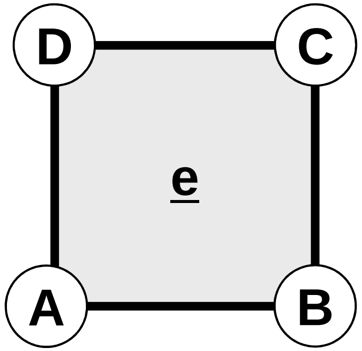
$$-\int_V \lambda \left(\frac{\partial [N]^T}{\partial x} \frac{\partial [N]}{\partial x} + \frac{\partial [N]^T}{\partial y} \frac{\partial [N]}{\partial y} \right) dV \cdot \{\phi\}$$

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



Element Matrix

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



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

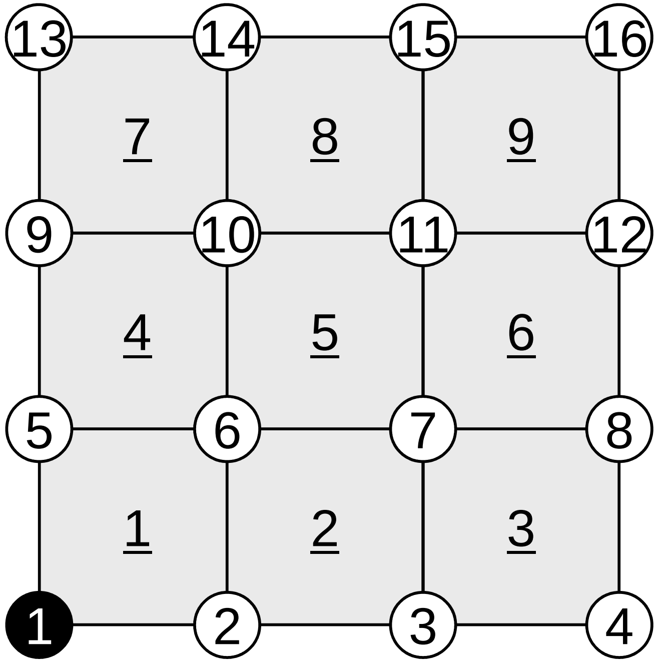
$$
+\int_V Q[N]^T dV = 0
$$

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

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

Global (Overall) Matrix

Accumulate each element matrix to “global” matrix.

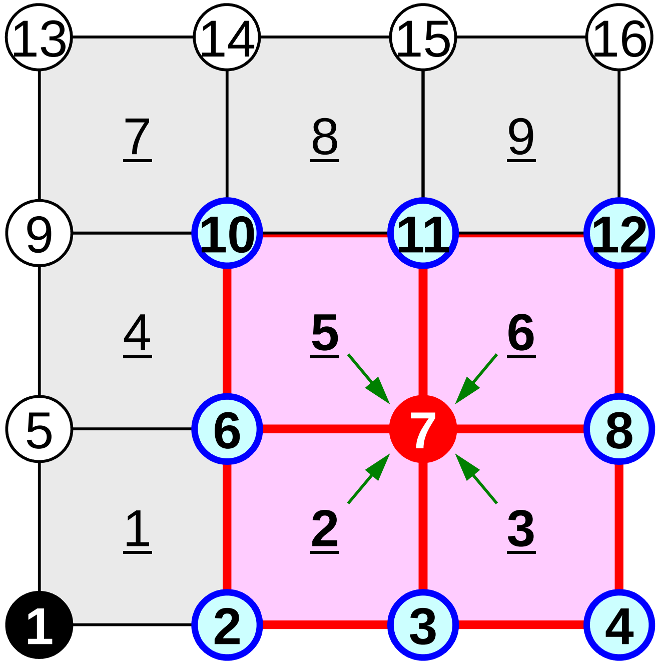


$$[K]\{\Phi\}=\{F\}$$

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

To each node …

Effect of surrounding elem’s/nodes are accumulated.



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

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

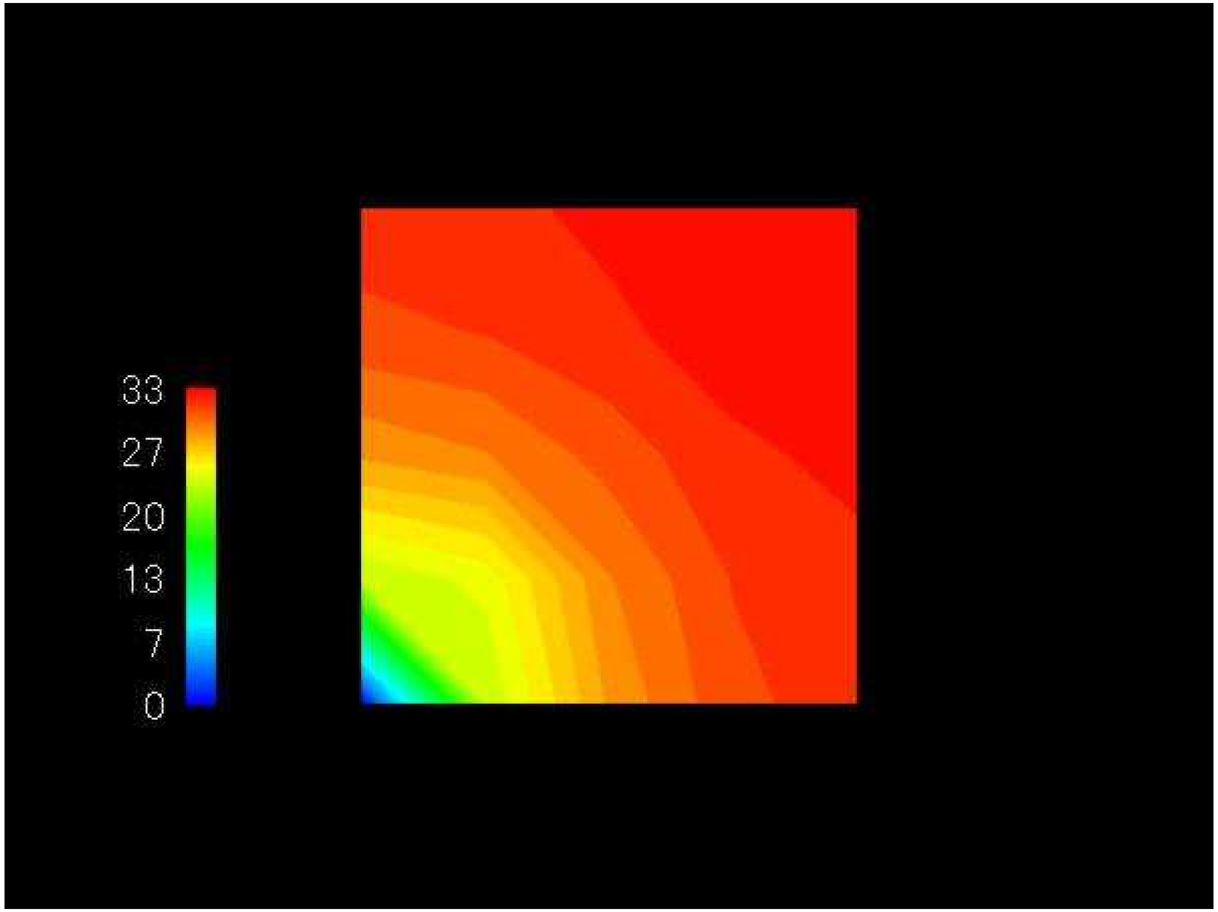
Solve the obtained global eqn’s

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 …

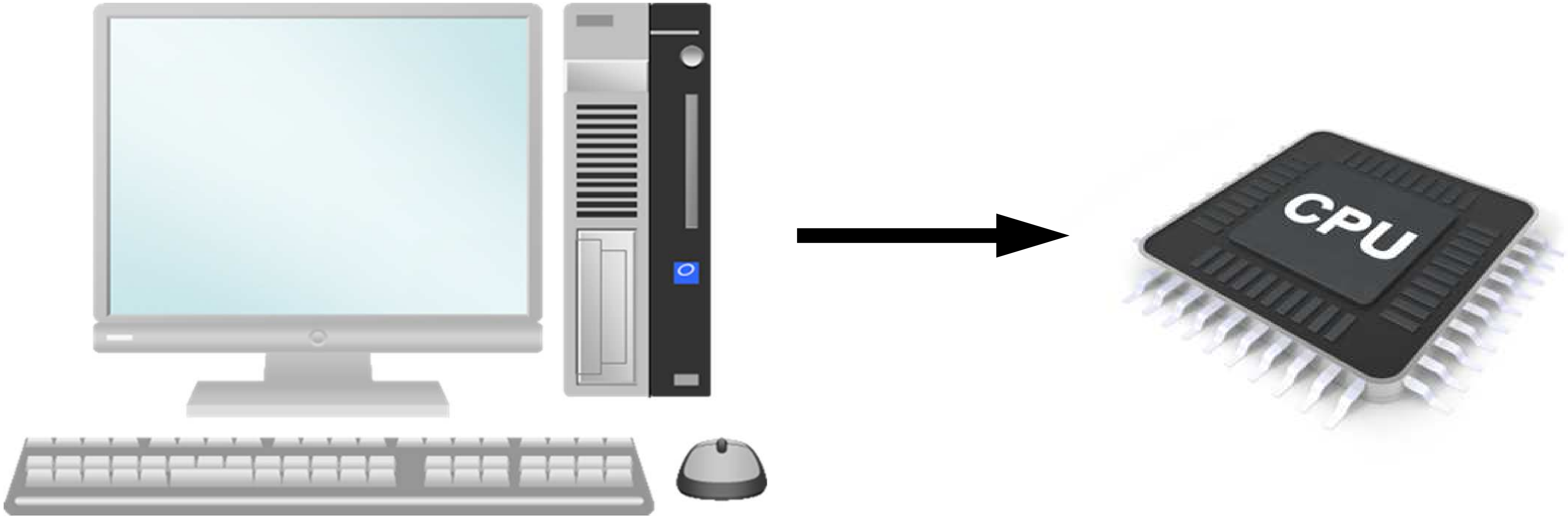


Features of FEM applications

- 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

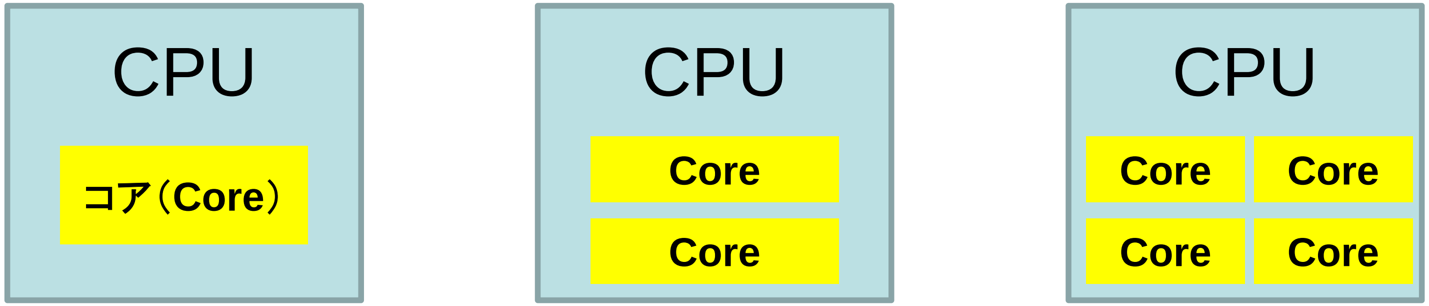
- Target: Parallel FEM
- **Supercomputers and Computational Science**
- Overview of the Class
- Future Issues

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

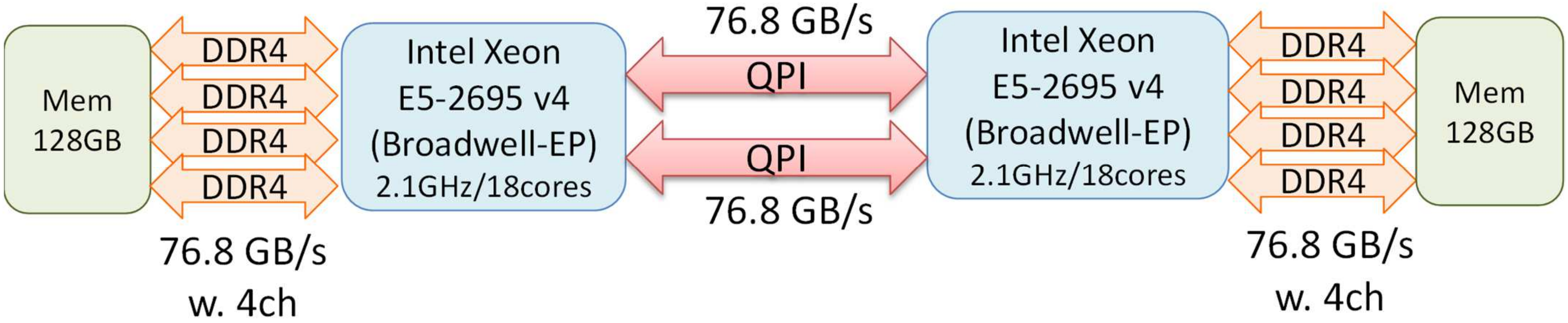


Single Core
1 cores/CPU

Dual Core
2 cores/CPU

Quad Core
4 cores/CPU

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

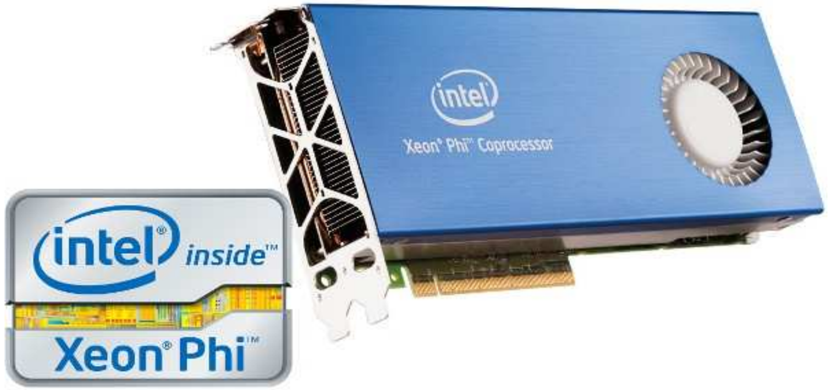


- GPU: Manycore
  - $O(10^1)$-$O(10^2)$ cores
- More and more cores
  - Parallel computing

- Reedbush-U: 18 cores x 2
  - Intel Xeon Broadwell-EP

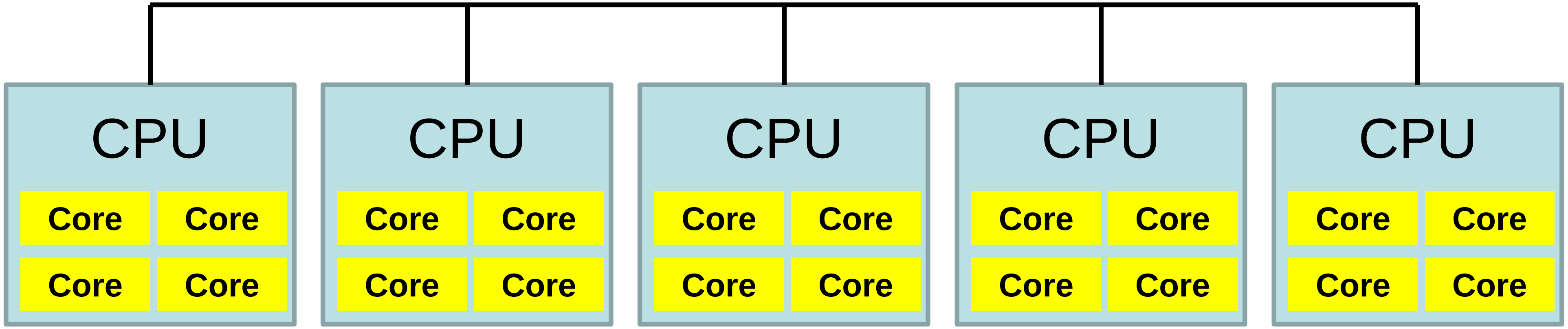
GPU/Manycores

- 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
  - Host CPU needed in the 1st generation
    - Stand-alone is possible now (Knights Landing, KNL)

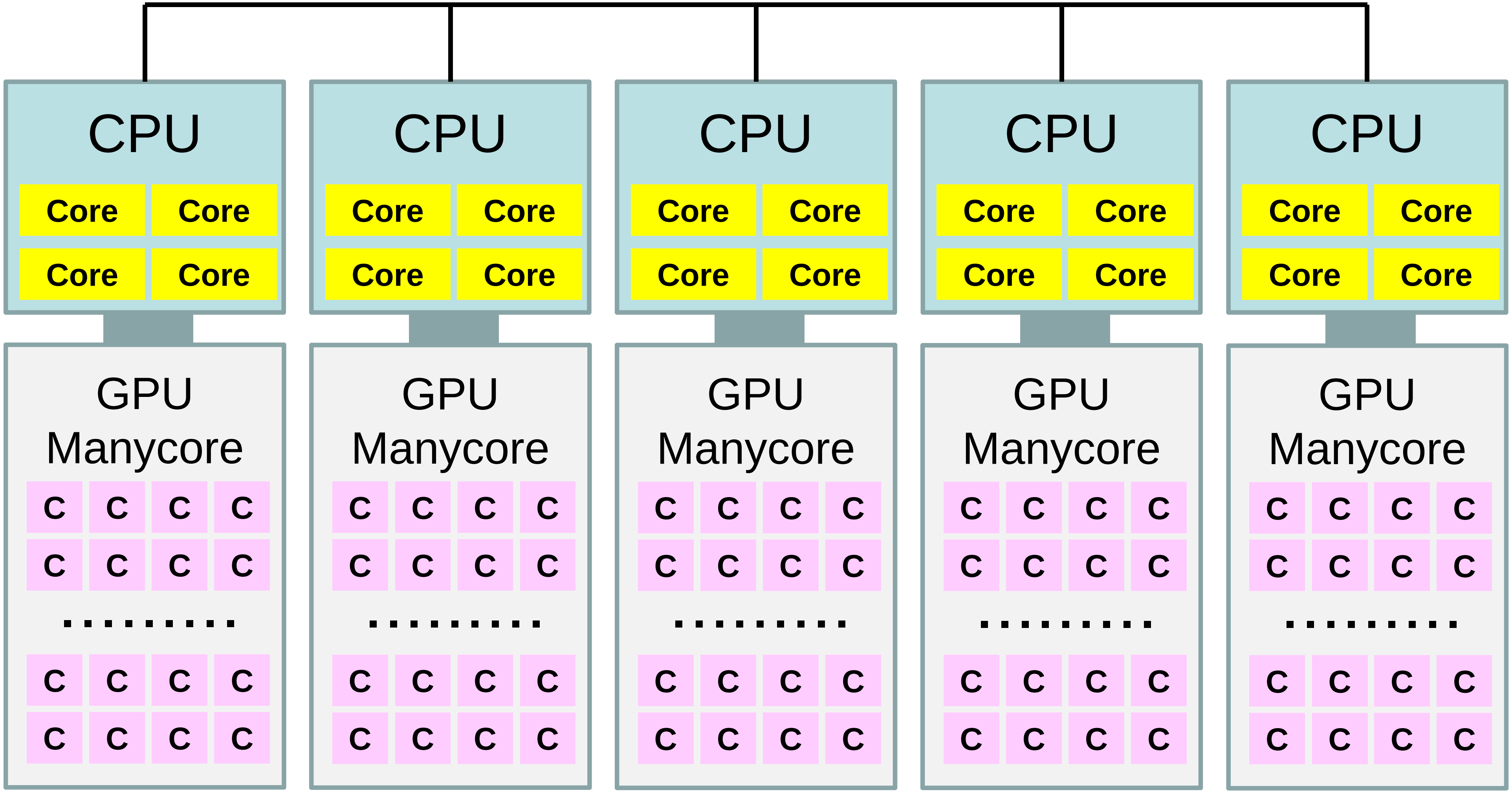


Parallel Supercomputers

Multicore CPU’s are connected through network



Supercomputers with Heterogeneous/Hybrid Nodes



CPU
Core Core
Core Core

GPU
Manycore
c c c c
c c c c
……….
c c c c
c c c c

CPU
Core Core
Core Core

GPU
Manycore
c c c c
c c c c
……….
c c c c
c c c c

CPU
Core Core
Core Core

GPU
Manycore
c c c c
c c c c
……….
c c c c
c c c c

CPU
Core Core
Core Core

GPU
Manycore
c c c c
c c c c
……….
c c c c
c c c c

CPU
Core Core
Core Core

GPU
Manycore
c c c c
c c c c
……….
c c c c
c c c c

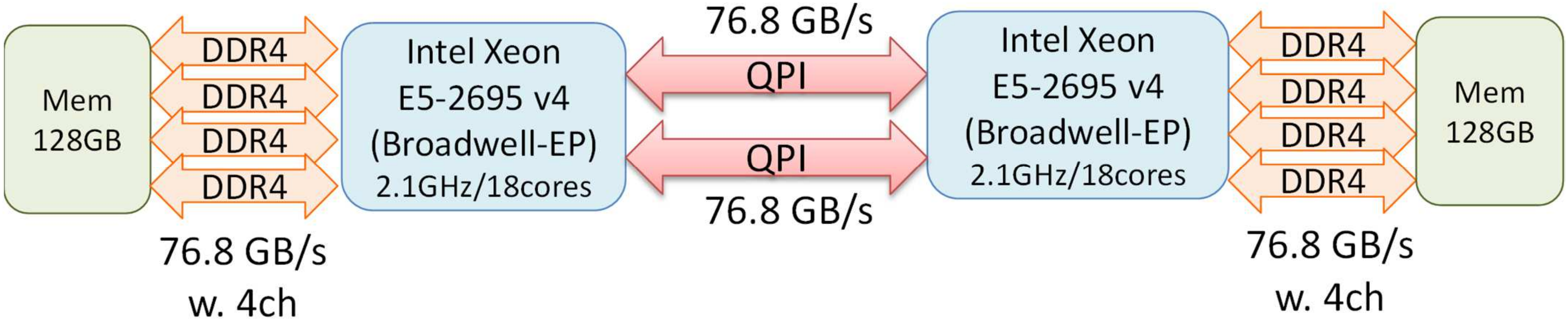
Performance of Supercomputers

- 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

Peak Performance of Reedbush-U

Intel Xeon E5-2695 v4 (Broadwell-EP)

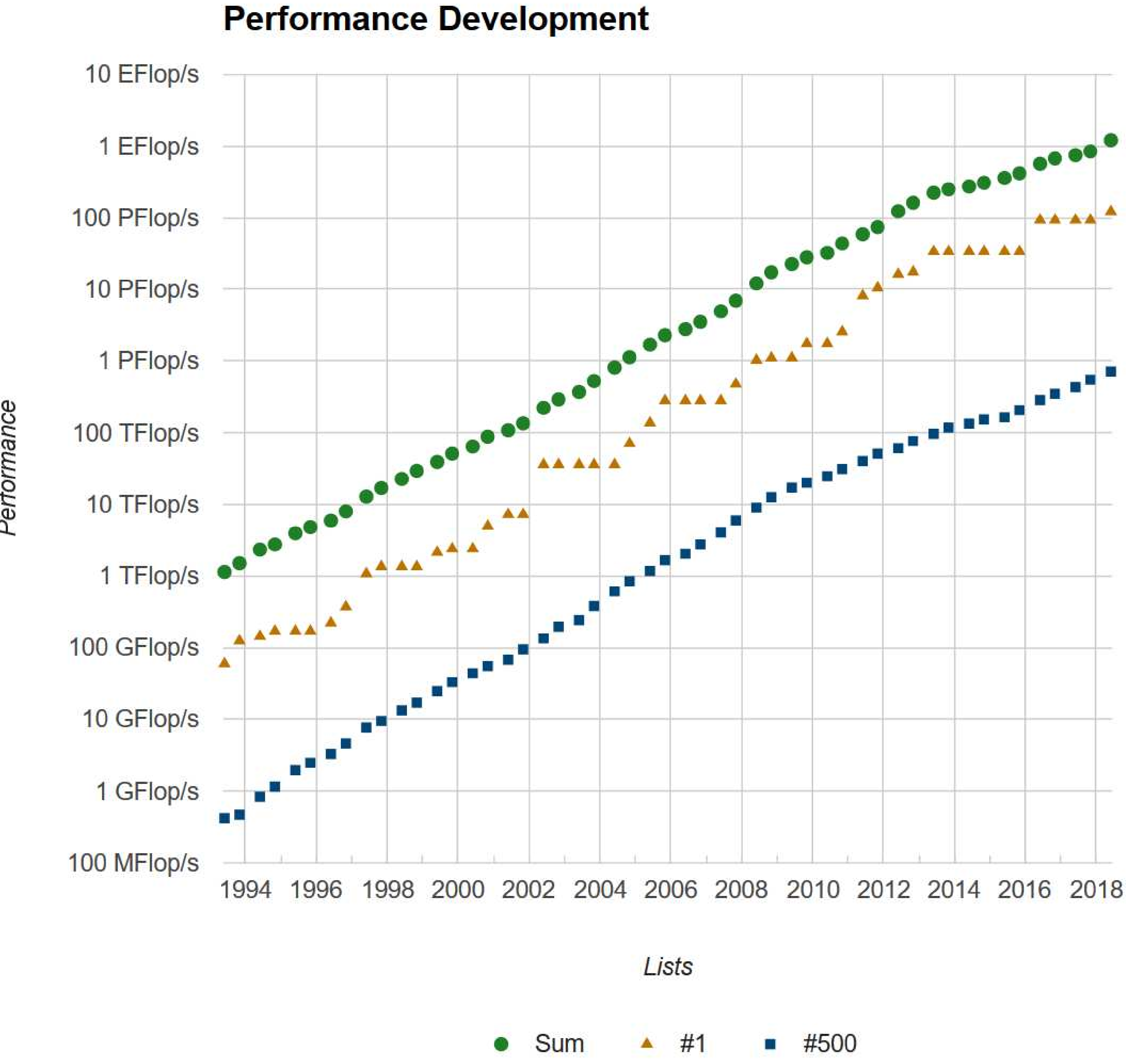


- 2.1 GHz
  - 16 DP (Double Precision) FLOP operations per Clock
- Peak Performance (1 core)
  - 2.1 × 16= 33.6 GFLOPS
- Peak Performance
  - 1-Socket, 18 cores: 604.8 GFLOPS
  - 2-Sockets, 36 cores: 1,209.6 GFLOPS 1-Node

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 in 2021 …

http://www.top500.org/

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)

51th TOP500 List (June, 2018)

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

http://www.top500.org/

	Site	Computer/Year Vendor	Cores	R_{max} (TFLOPS)	R_{peak} (TFLOPS)	Power (kW)
1	**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	2,282,544	122,300 (= 122.3 PF)	187,659	8,806
2	**Sunway TaihuLight, 2016, China** National Supercomputing Center in Wuxi	Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway	10,649,600	93,015	125,436	15,371
3	**Sieera, 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	71,610	119,194	
4	**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	61,445	100,679	18,482
5	**ABCI (AI Bridging Cloud Infrastructure), 2018, Japan** National Institute of Advanced Industrial Science and Technology (AIST)	PRIMERGY CX2550 M4, Xeon Gold 6148 20C 2.4GHz, NVIDIA Tesla V100 SXM2, Infiniband EDR	391,680	19,880	32,577	1,649
6	**Piz Daint, 2017, Switzerland** Swiss National Supercomputing Centre (CSCS)	Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100	361,760	19,590	25,326	2,272
7	**Titan, 2012, USA** DOE/SC/Oak Ridge National Laboratory	Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x	560,640	17,590	27,113	8,209
8	**Sequoia, 2011, USA** DOE/NNSA/LLNL	BlueGene/Q, Power BQC 16C 1.60 GHz, Custom	1,572,864	17,173	20,133	7,890
9	**Trinity, 2017, USA** DOE/NNSA/LANL/SNL	Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect	979,968	14,137	43,903	3,844
10	**Cori, 2016, Japan** DOE/SC/LBNL/NERSC	Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect	622,336	14,016	27,881	3,939
12	**Oakforest-PACS, 2016, Japan** Joint Center for Advanced High Performance Computing	PRIMERGY CX1640 M1, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path	556,104	13,556	24,913	2,719

HPCG Ranking (June, 2018)

	Computer	Cores	HPL Rmax (Pflop/s)	TOP500 Rank	HPCG (Pflop/s)	Peak
1	**Summit**	2,392,000	122.300	1	2.926	1.5%
2	**Sierra**	835,584	71.610	3	1.796	1.5%
3	**K computer**	705,024	10.510	16	0.603	5.3%
4	**Trinity**	979,072	14.137	9	0.546	1.8%
5	**Piz Daint**	361,760	19.590	6	0.486	1.9%
6	**Sunway TaihuLight**	10,649,600	93.015	2	0.481	0.4%
7	**Oakforest-PACS**	557,056	13.555	12	0.385	1.5%
8	**Cori**	632,400	13.832	10	0.355	1.3%
9	**Tera-1000-2**	522,240	11.965	14	0.334	1.4%
10	**Sequoia**	1,572,864	17.173	8	0.330	1.6%

http://www.hpcg-benchmark.org/

http://www.top500.org/

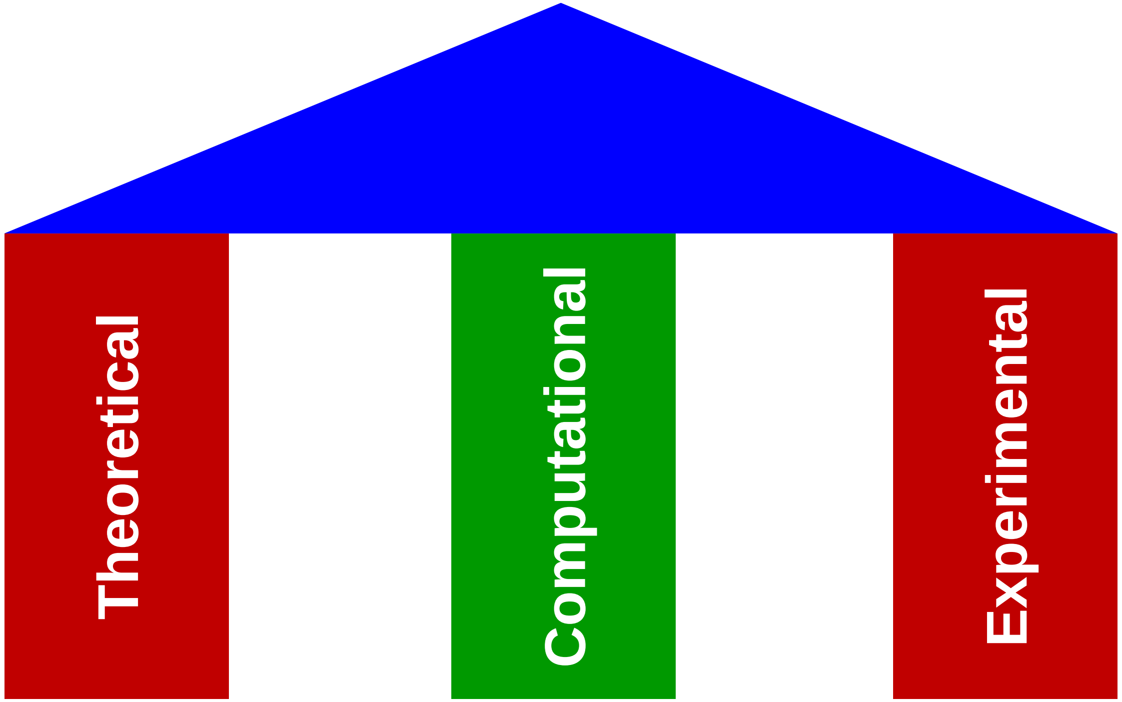
Green 500 Ranking (June, 2018)

	TOP 500 Rank	System	Cores	HPL Rmax (Pflop/s)	Power (MW)	GFLOPS/W
1	359	**Shoubu system B, Japan**	794,400	858.	47	18.404
2	419	**Suiren2, Japan**	762,624	798.	47	16.835
3	385	**Sakura, Japan**	794,400	825.	50	16.657
4	227	**DGX SaturnV Volta, USA**	22,440	1,070.	97	15.113
5	1	**Summit, USA**	2,282,544	122,300.	8,806	13.889
6	19	**TSUBAME3.0, Japan**	135,828	8,125.	792	13.704
7	287	**AIST AI Cloud, Japan**	23,400	961.	76	12.681
8	5	**ABCI, Japan**	391,680	19,880.	1,649	12.054
9	255	**MareNostrum P9 CTE, Spain**	19,440	1,018.	86	11.865
10	171	**RAIDEN GPU, Japan**	35,360	1,213.	107	11.363
13	411	**Reedbush-L, U.Tokyo, Japan**	16,640	806.	79	10.167
19	414	**Reedbush-H, U.Tokyo, Japan**	17,760	802.	94	8.575

Computational Science

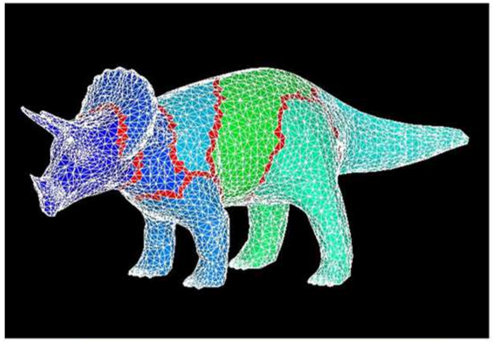
The 3rd Pillar of Science

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

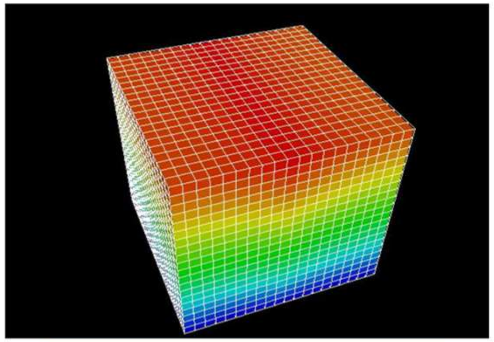


Methods for Scientific Computing

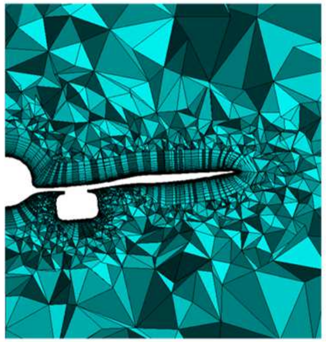
- Numerical solutions of PDE (Partial Diff. Equations)
- Grids, Meshes, Particles
  - Large-Scale Linear Equations
  - Finer meshes provide more accurate solutions



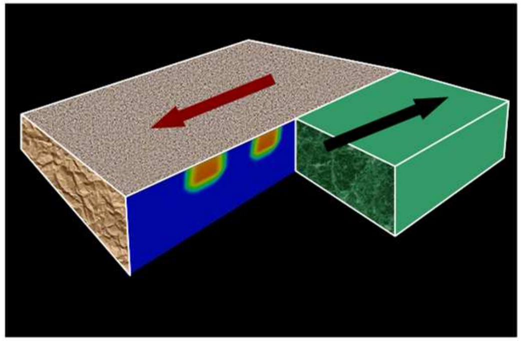
有限要素法
Finite Element Method
FEM



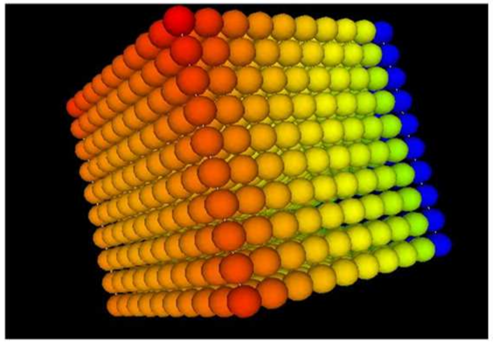
差分法
Finite Difference Method
FDM



有限体積法
Finite Volume Method
FVM



境界要素法
Boundary Element Method
BEM

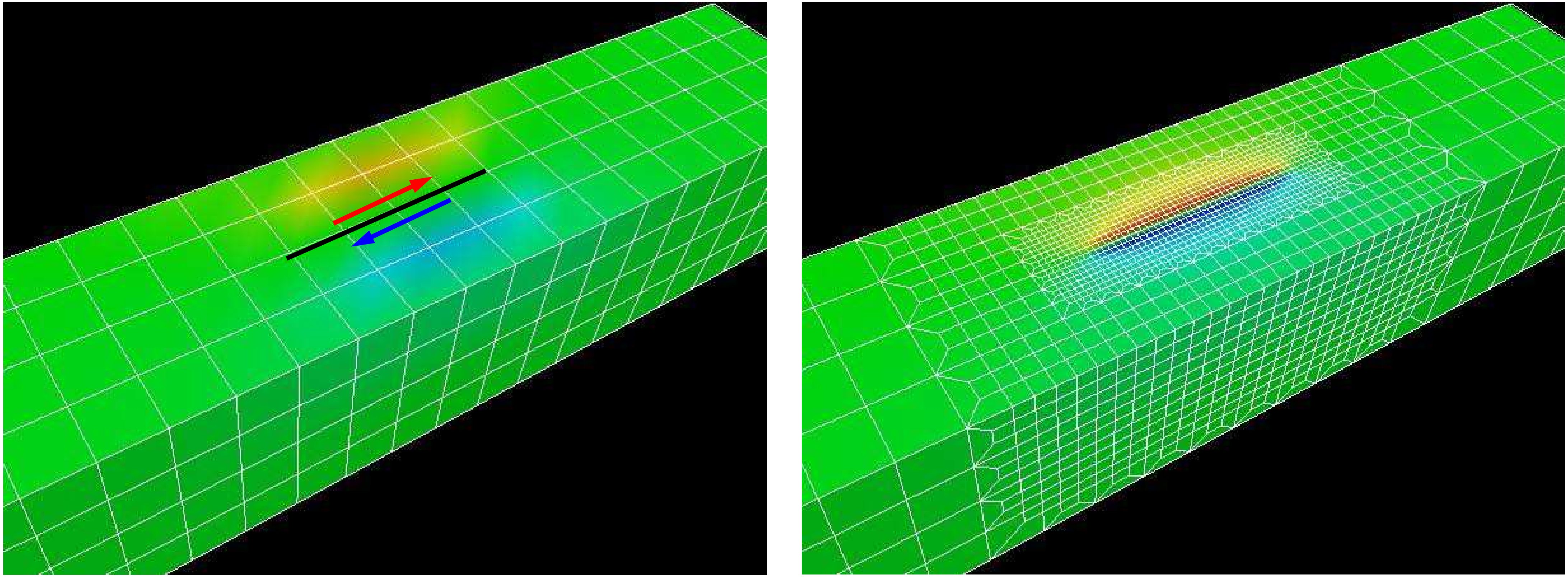


個別要素法
Discrete Element Method
DEM

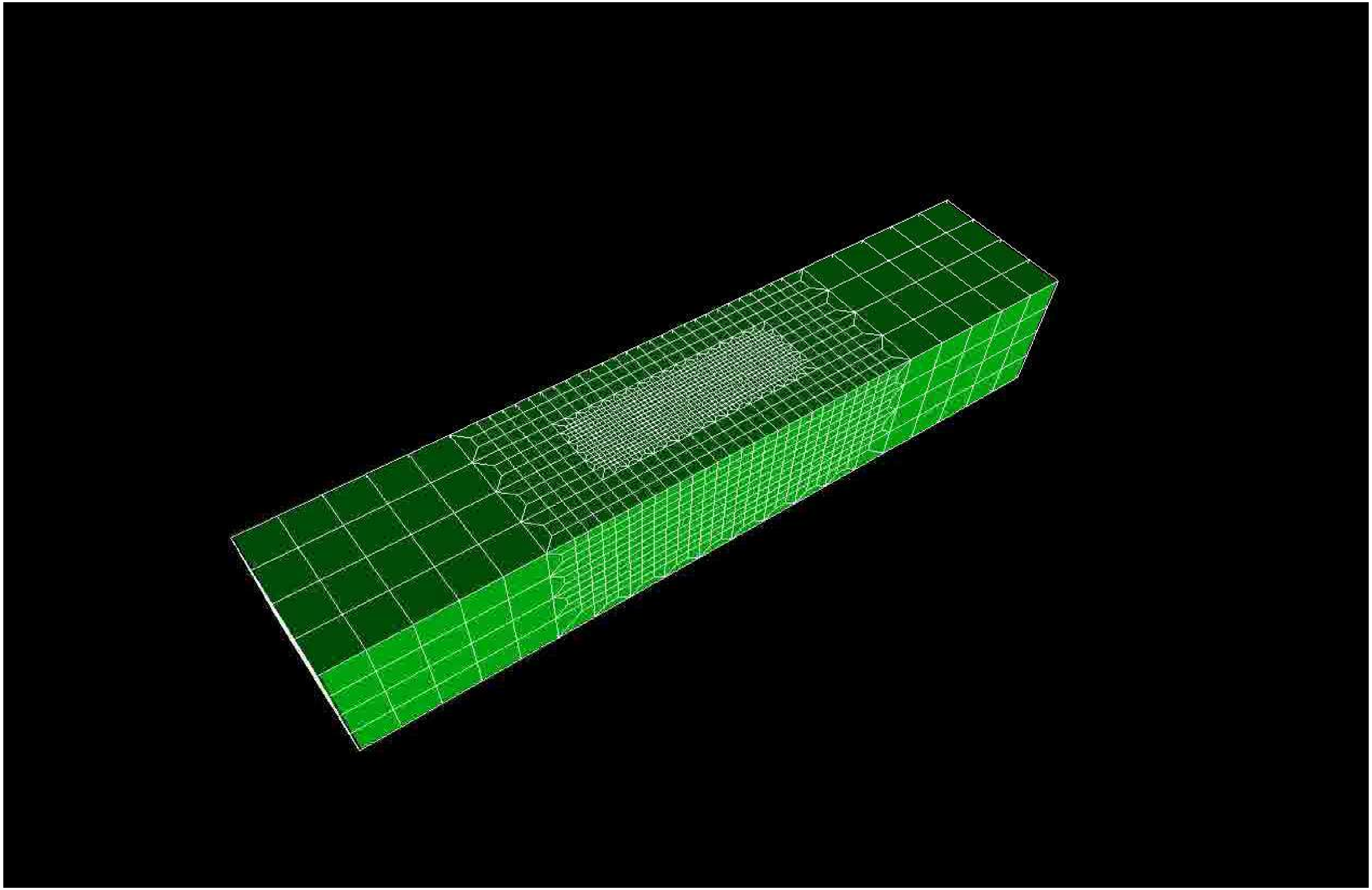
3D Simulations for Earthquake Generation Cycle

San Andreas Faults, CA, USA

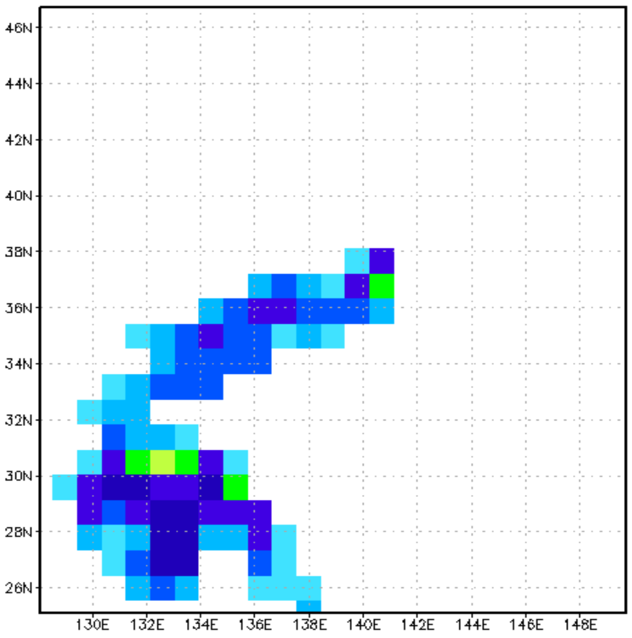
Stress Accumulation at Transcurrent Plate Boundaries



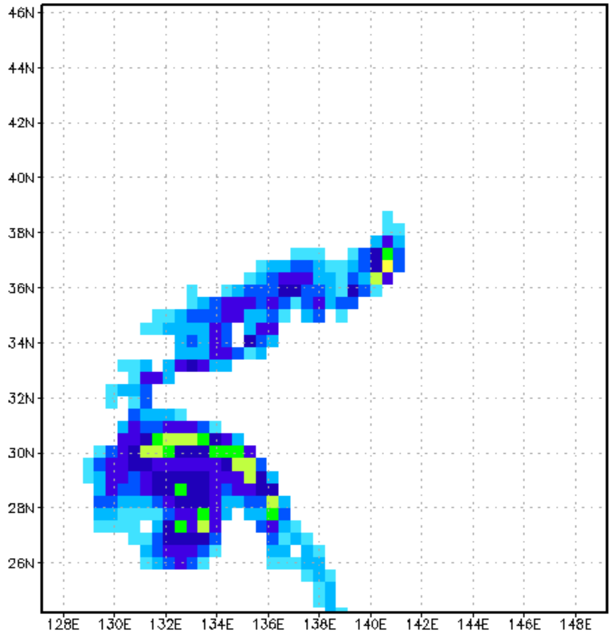
Adaptive FEM: High-resolution needed at meshes with large deformation (large accumulation)



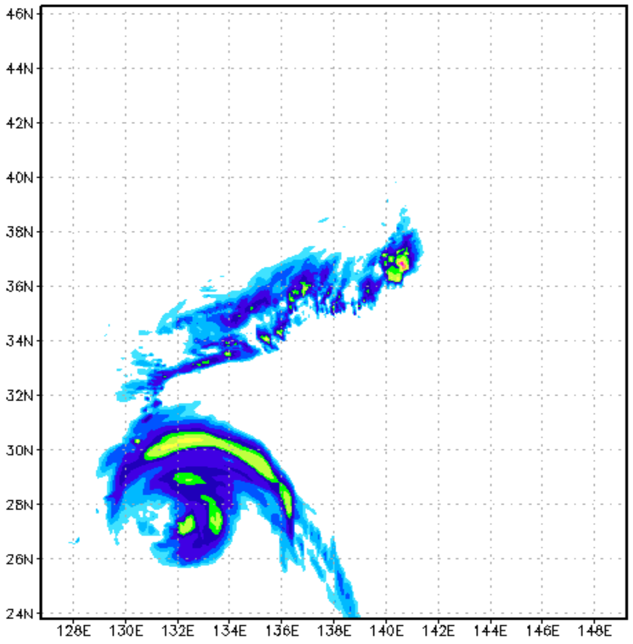
Typhoon Simulations by FDM
Effect of Resolution



Δh=100km



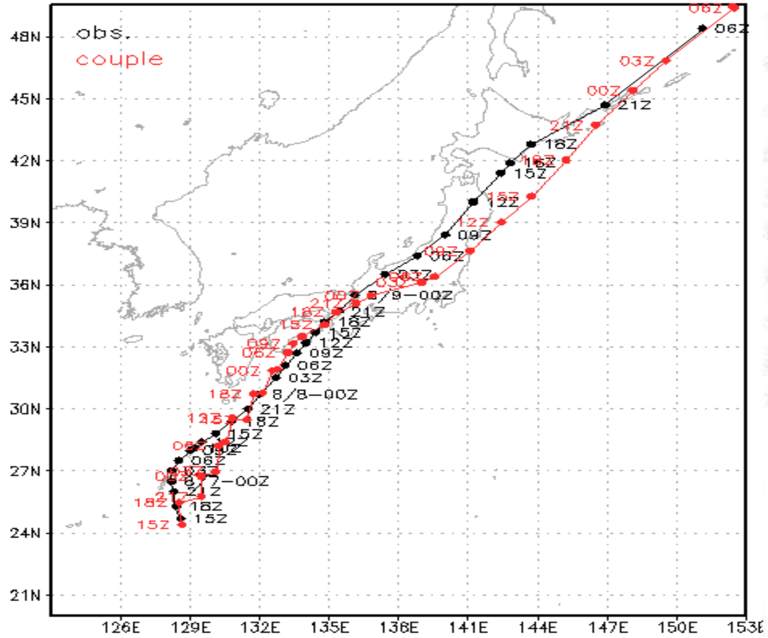
Δh=50km

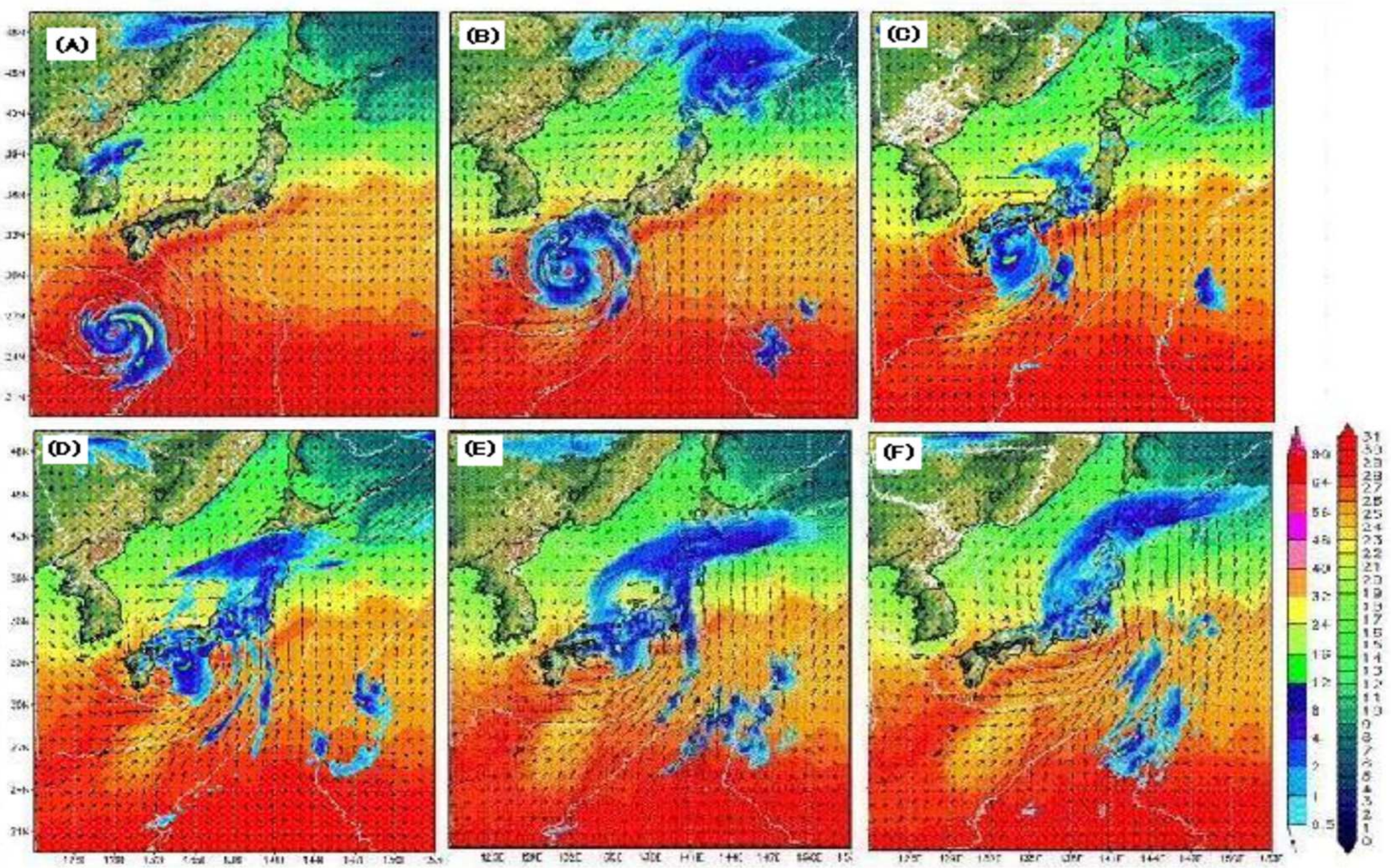


Δh=5km

[JAMSTEC]

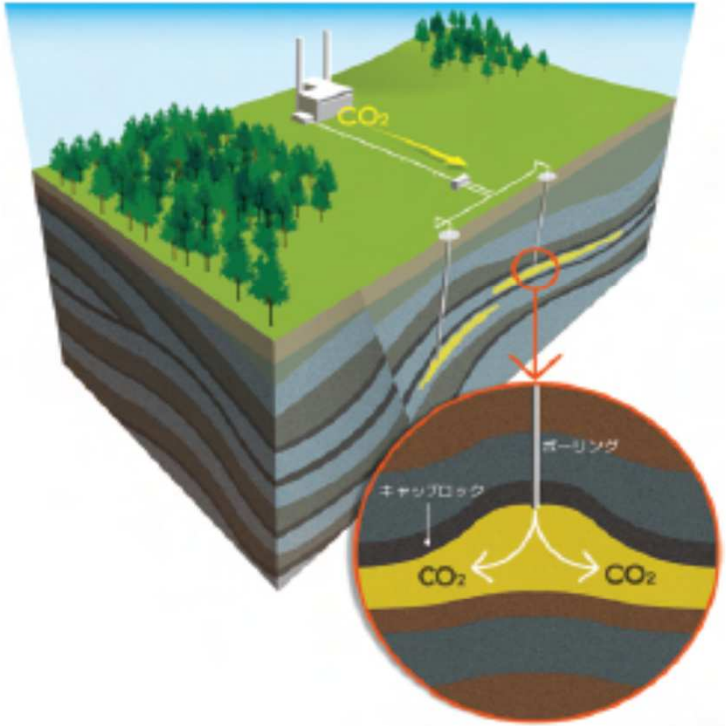
Simulation of Typhoon MANGKHUT in 2003 using the Earth Simulator





[JAMSTEC]

Simulation of Geologic CO_2 Storage



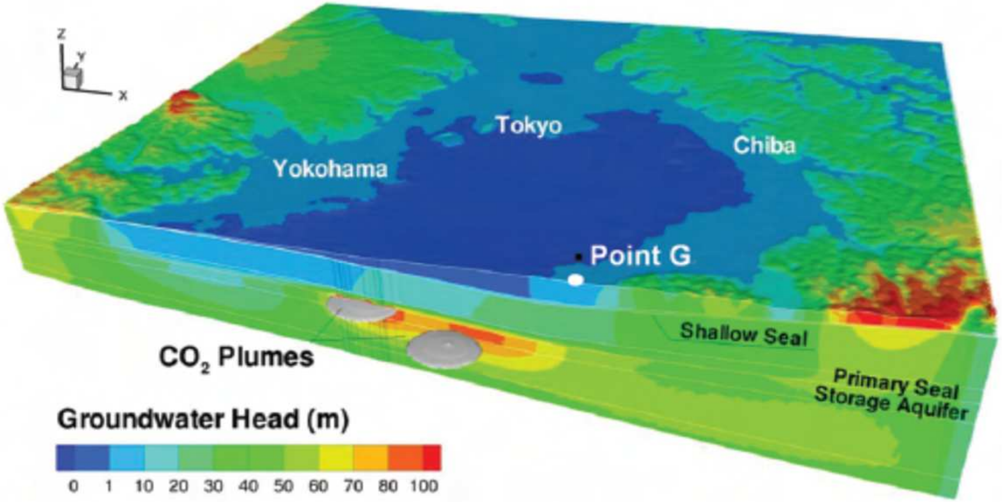
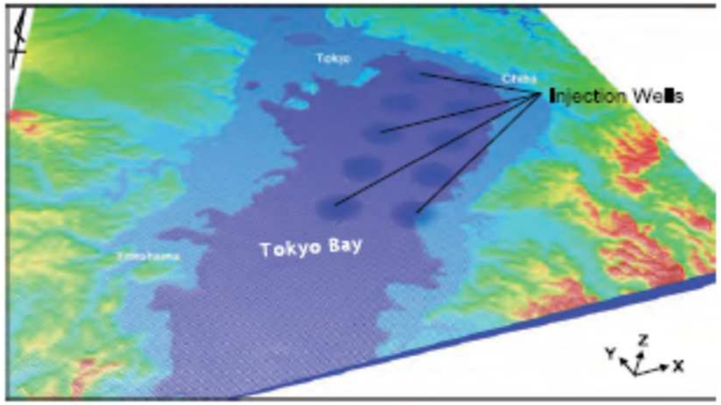
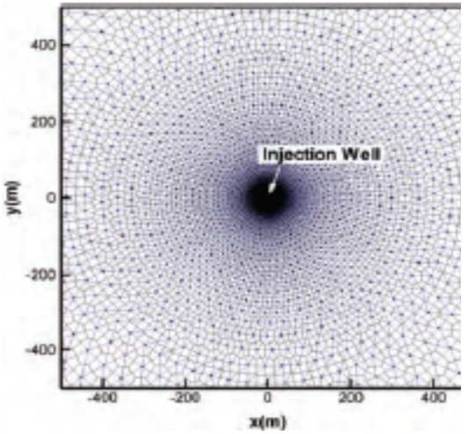
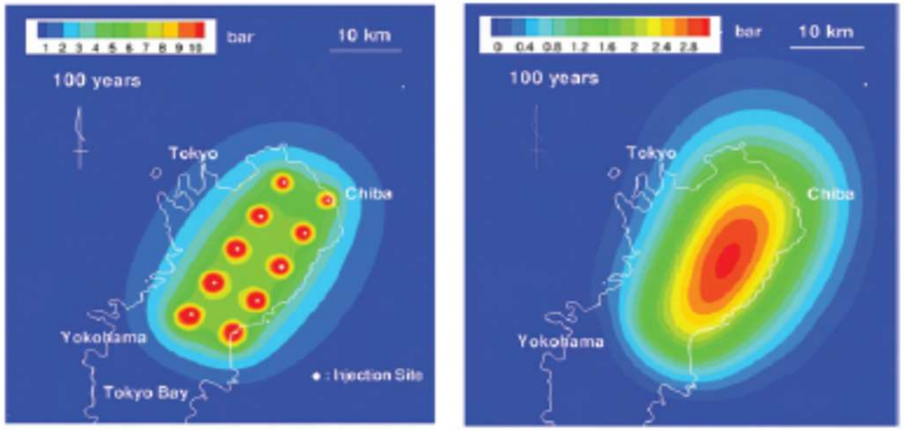


図-4 CO_2 圧入後の地下水圧（全水頭換算）の分布（100 年後）







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

Science

Modeling

Algorithm

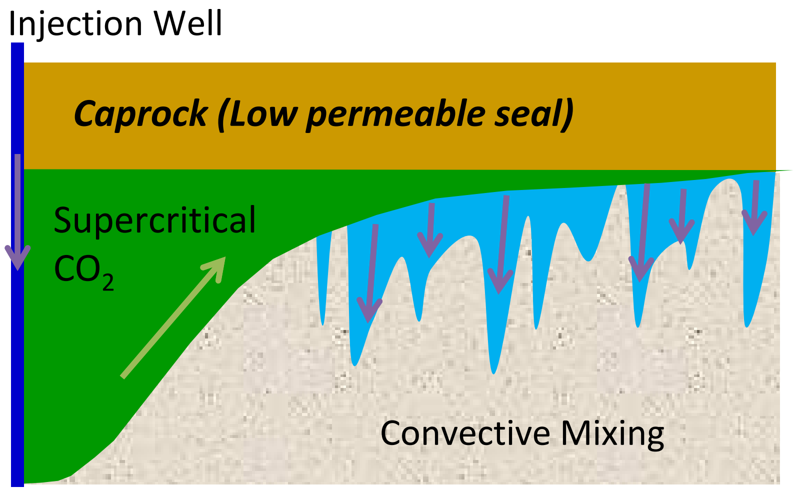
Software

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

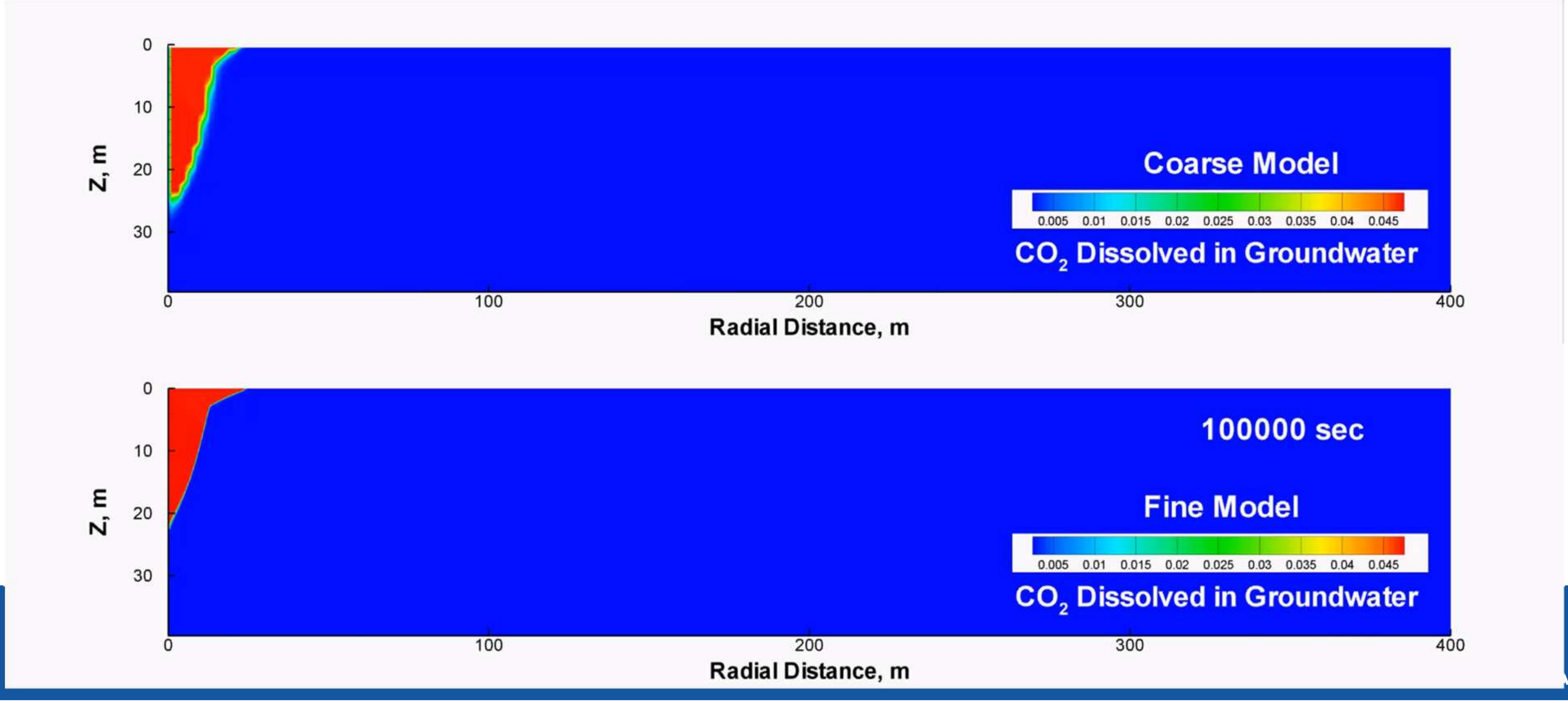
Diffusion-Dissolution-Convection Process



- **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**

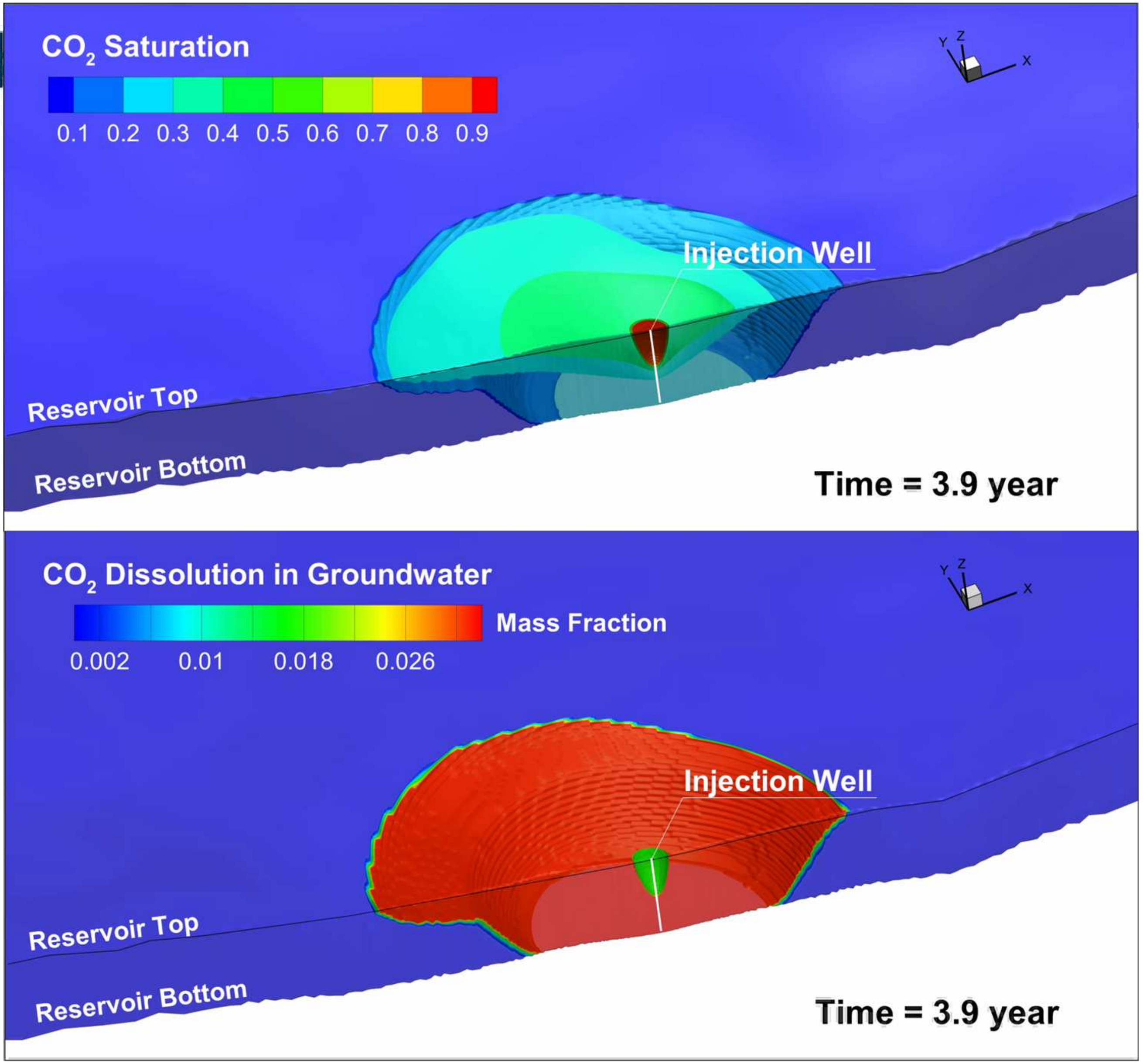
The mixing significantly enhances the CO_2 dissolution into groundwater, resulting in more stable storage

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

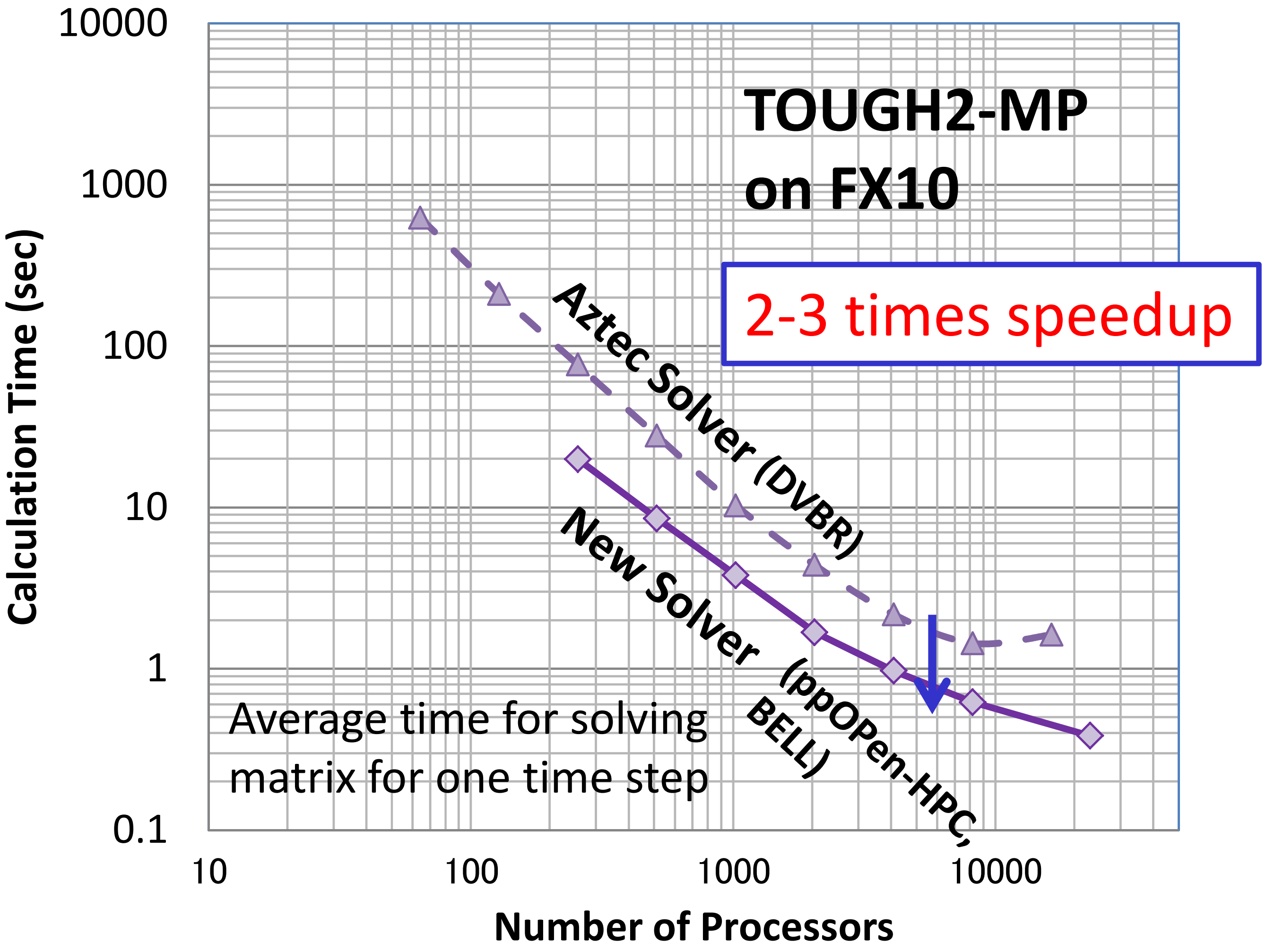
- Permeability: 100 md
- Porosity: 20%
- Pressure: 3MPa
- Temperature: 100°C
- Salinity: 15wt%

[Dr. Hajime Yamamoto, Taisei]

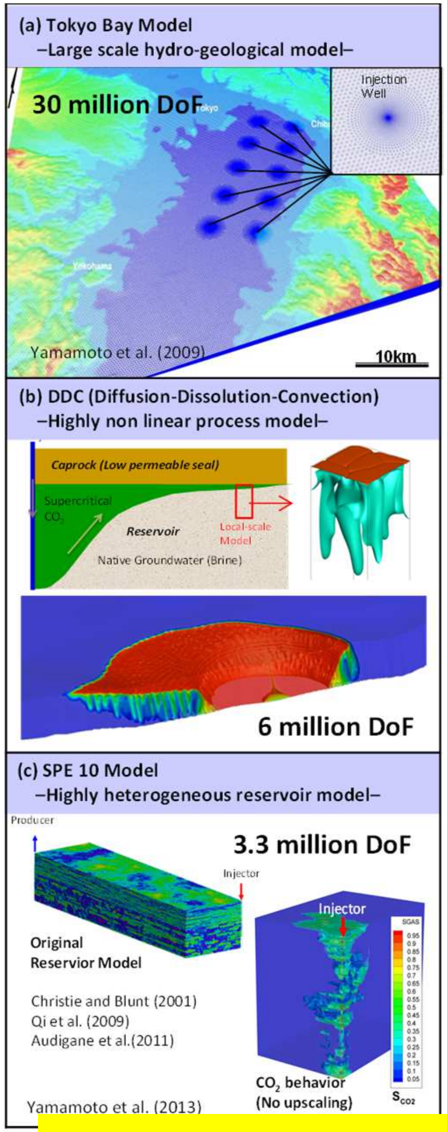
- The meter-scale fingers gradually developed to larger ones in the field-scale model
- Huge number of time steps (> 10^5) were required to complete the 1,000-yrs simulation
- Onset time (10-20 yrs) is comparable to theoretical (linear stability analysis, 15.5yrs)

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

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 10-14, 18-21
  - 09:00-10:30, 10:45-12:15, 13:30-15:00, 15:15-16:45
  - **http://nkl.cc.u-tokyo.ac.jp/18e/**
- 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 28th (Sun) 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/18e/**
  - General information is available
  - No hardcopy of course materials are provided (Please print them by yourself)

Date	Hour	ID	Content
Sept.10 (M)	0900-1030	CE01	Introduction
	1045-1215	CE02	Introdution to FEM
	1330-1500	CE03	1D-FEM (1/2)
	1515-1645	Practice	Practice
Sept. 11 (T)	0900-1030	CE04	1D-FEM (2/2)
	1045-1215	CE05	3D-FEM (1/2)
	1330-1500	CE06	3D-FEM (2/2)
	1515-1645	Practice	Practice (Instructor is not available)
Sept. 12 (W)	0900-1030	CE07	Introduction to Parallel FEM
	1045-1215	CE08	Login to Reedbush-U
	1330-1500	CE09	Parallel Programming by MPI (1) (1/2)
	1515-1645	Practice	Practice (Instructor is not available)
Sept. 13 (Th)	0900-1030	Practice	Practice (Instructor is not available)
	1045-1215	Practice	Practice (Instructor is not available)
	1330-1500	CE10	Parallel Programming by MPI (1) (2/2)
	1515-1645	Practice	Practice
Sept. 14 (F)	0900-1030	CE11	Report S1
	1045-1215	Practice	Practice

Date	Hour	ID	Content
Sept. 14 (F)	1330-1500	CE12	Parallel Programming by MPI (2) (1/2)
	1515-1645	CE13	Parallel Programming by MPI (2) (2/2)
Sept.18 (T)	0900-1030	CE14	Introduction to Tuning
	1045-1215	CE15	Report S2
	1330-1500	Practice	Practice (Instructor is not available)
	1515-1645	Practice	Practice (Instructor is not available)
Sept. 19 (W)	0900-1030	CE16	Parallel FEM (1/3)
	1045-1215	CE17	Parallel FEM (2/3)
	1330-1500	CE18	Parallel FEM (3/3)
	1515-1645	Practice	Practice
Sept. 20 (Th)	0900-1030	Practice	Practice (Instructor is not available)
	1045-1215	Practice	Practice (Instructor is not available)
	1330-1500	CE19	Hybrid Parallel FEM (1/2)
	1515-1645	Practice	Practice (Instructor is not available)
Sept. 21 (F)	0900-1030	Practice	Practice (Instructor is not available)
	1045-1215	Practice	Practice (Instructor is not available)
	1330-1500	CE20	Hybrid Parallel FEM (2/2)
	1515-1645	Practice	Practice (Instructor is not available)

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.

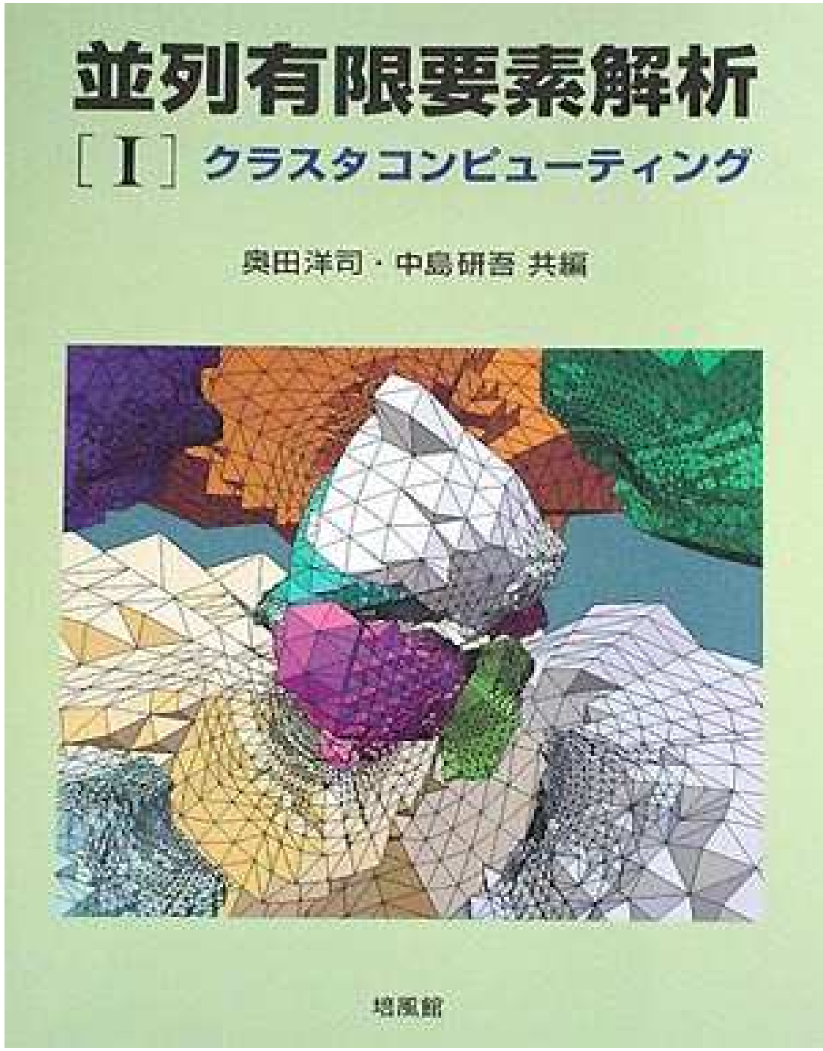
- Smith, I. 他「Programming the Finite Element Method (4th edition)」，Wiley.

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」

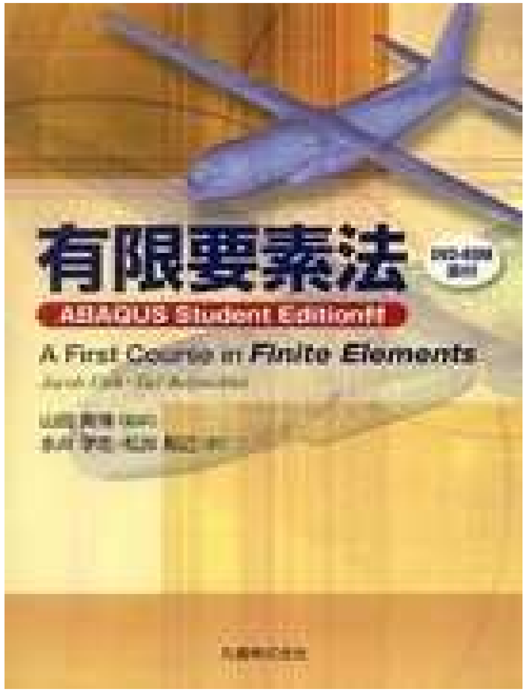
培風館，2004.

- 「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

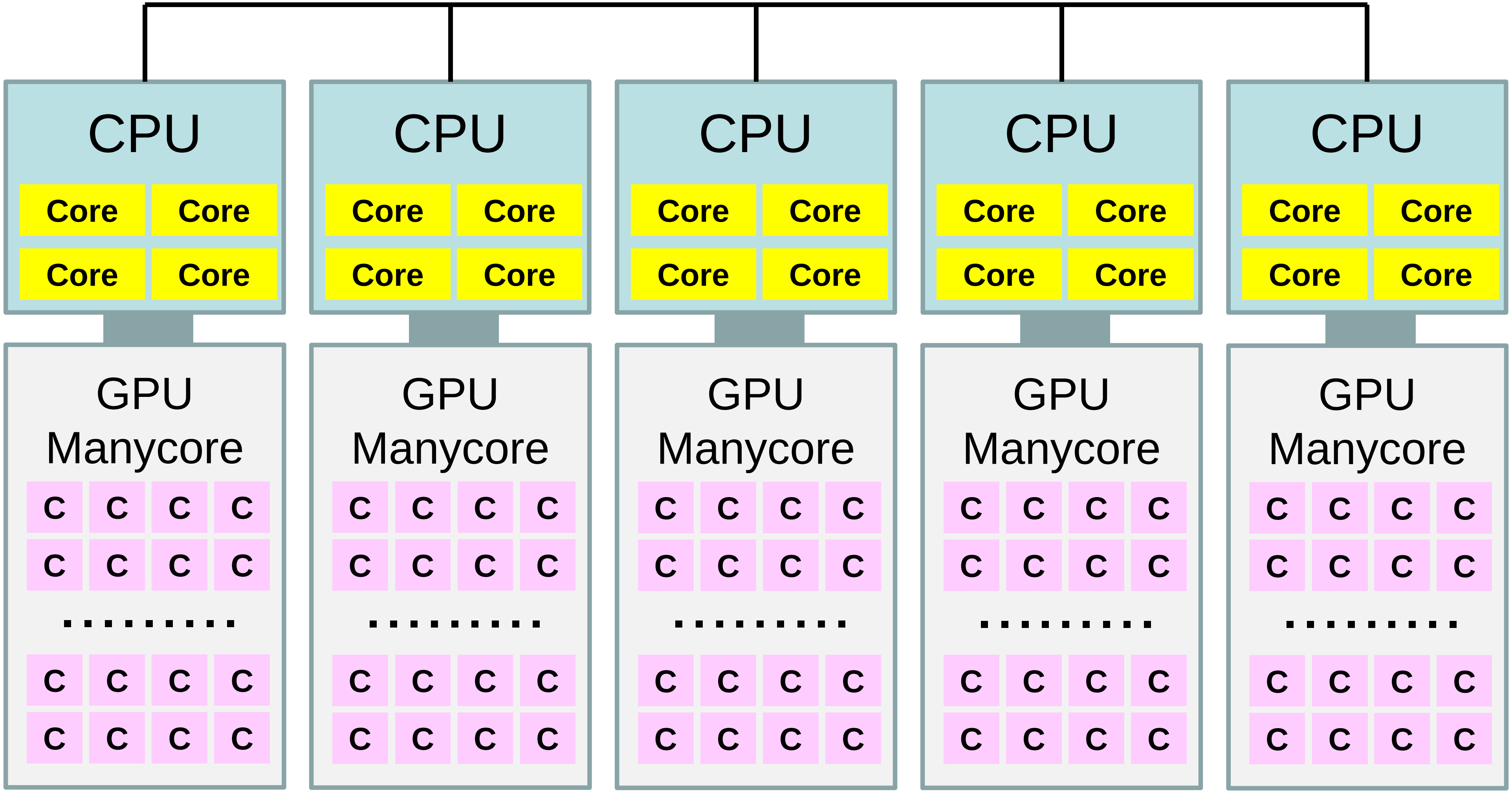
- Target: Parallel FEM
- Supercomputers and Computational Science
- Overview of the Class
- **Future Issues**

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

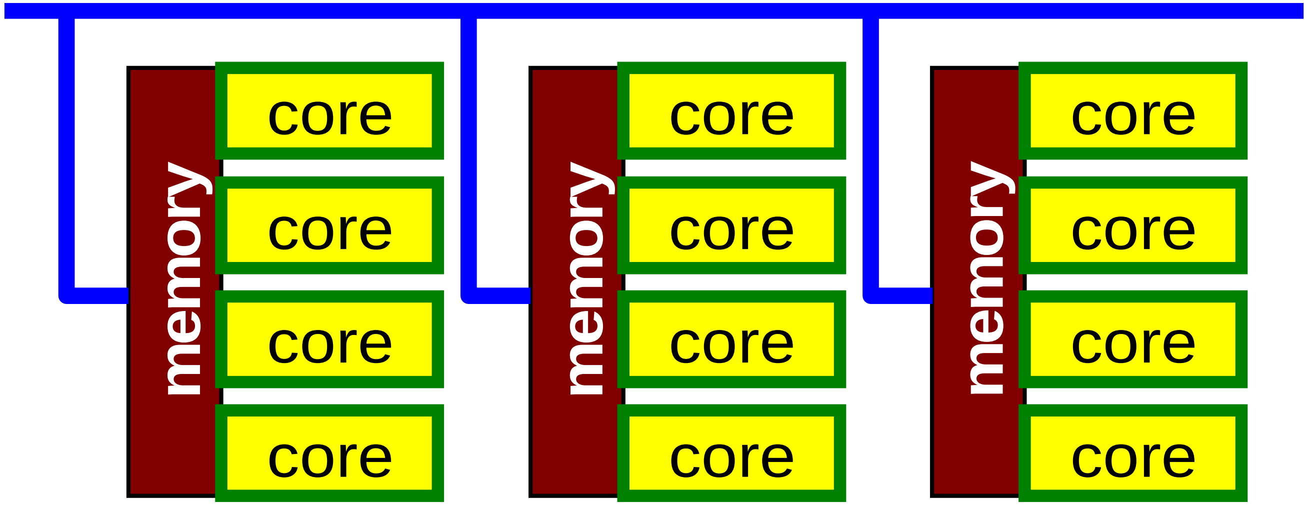


Hybrid Parallel Programming Model is essential for Post-Peta/Exascale Computing

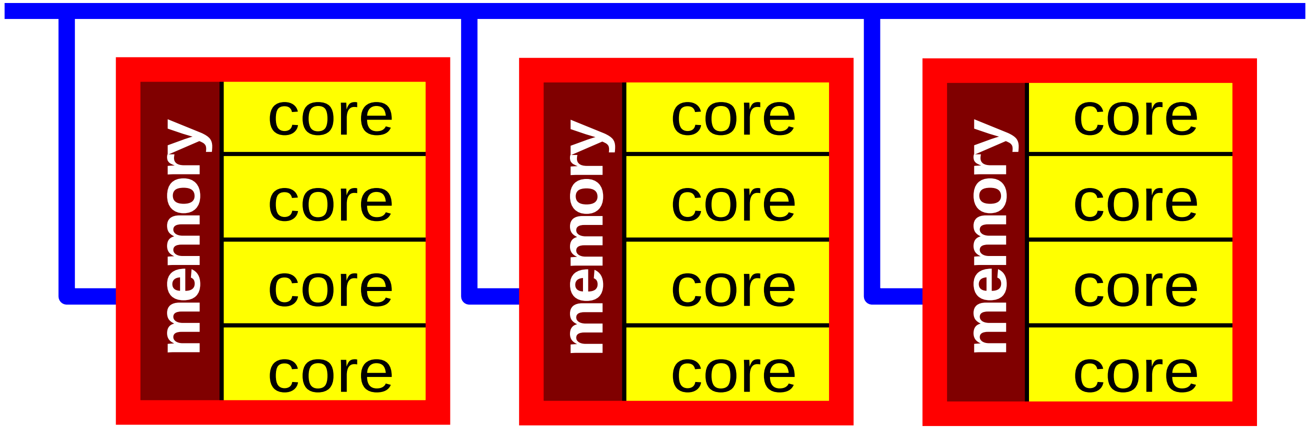
- Message Passing (e.g. MPI) + Multi Threading (e.g. OpenMP, CUDA, OpenCL, OpenACC etc.)
- In K computer and FX10, hybrid parallel programming is recommended
  - MPI + Automatic Parallelization by Fujitsu’s Compiler
- Expectations for Hybrid
  - Number of MPI processes (and sub-domains) to be reduced
  - 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-NEO)= X(NOD_EXPORT(k))
        enddo

        call MPI_Isend (WS(istart+1-NEO), 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)
  - 2021 (A21 by US-DOE, Department of Energy)
    - 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

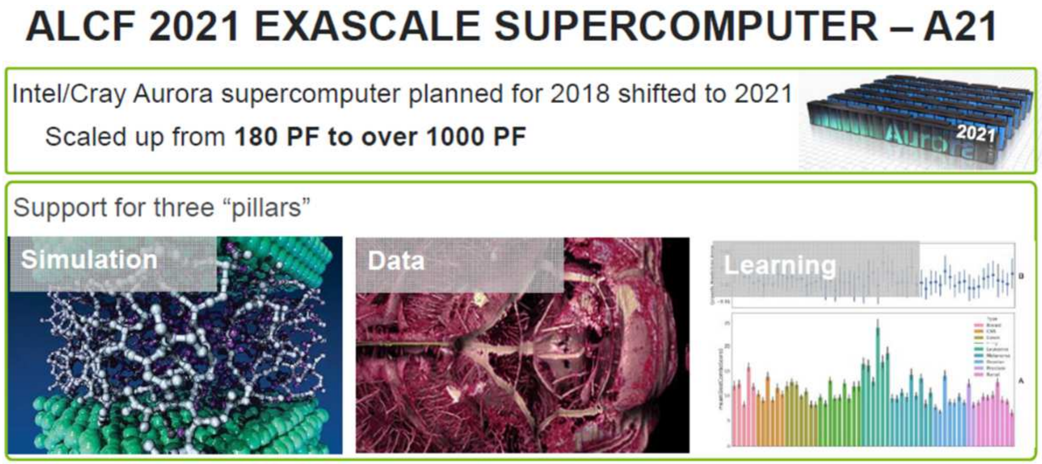
Future of Supercomputers (2/2)

- 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.

Towards Exascale/Post Moore Era

- Moore's law is the observation that the number of transistors in a dense integrated circuit doubles about every two years (18-24 months).
- First Exascale System (A21, US-DOE) in 2021 ?
- Supercomputing is changing
  - More “Intelligent” Supercomputing by integration of (Simulation + Data + Learning)
- Power Consumption
  - Various types of Workload, Optimum HW
    - CPU, GPU, FPGA, Quantum/Neuromorphic, Custom Chips

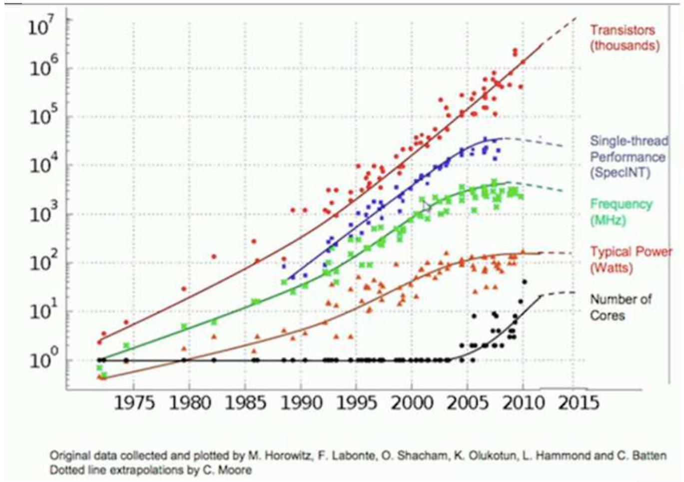


ALCF 2021 EXASCALE SUPERCOMPUTER – A21

Intel/Cray Aurora supercomputer planned for 2018 shifted to 2021

Scaled up from **180 PF to over 1000 PF**

Support for three “pillars”



Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore