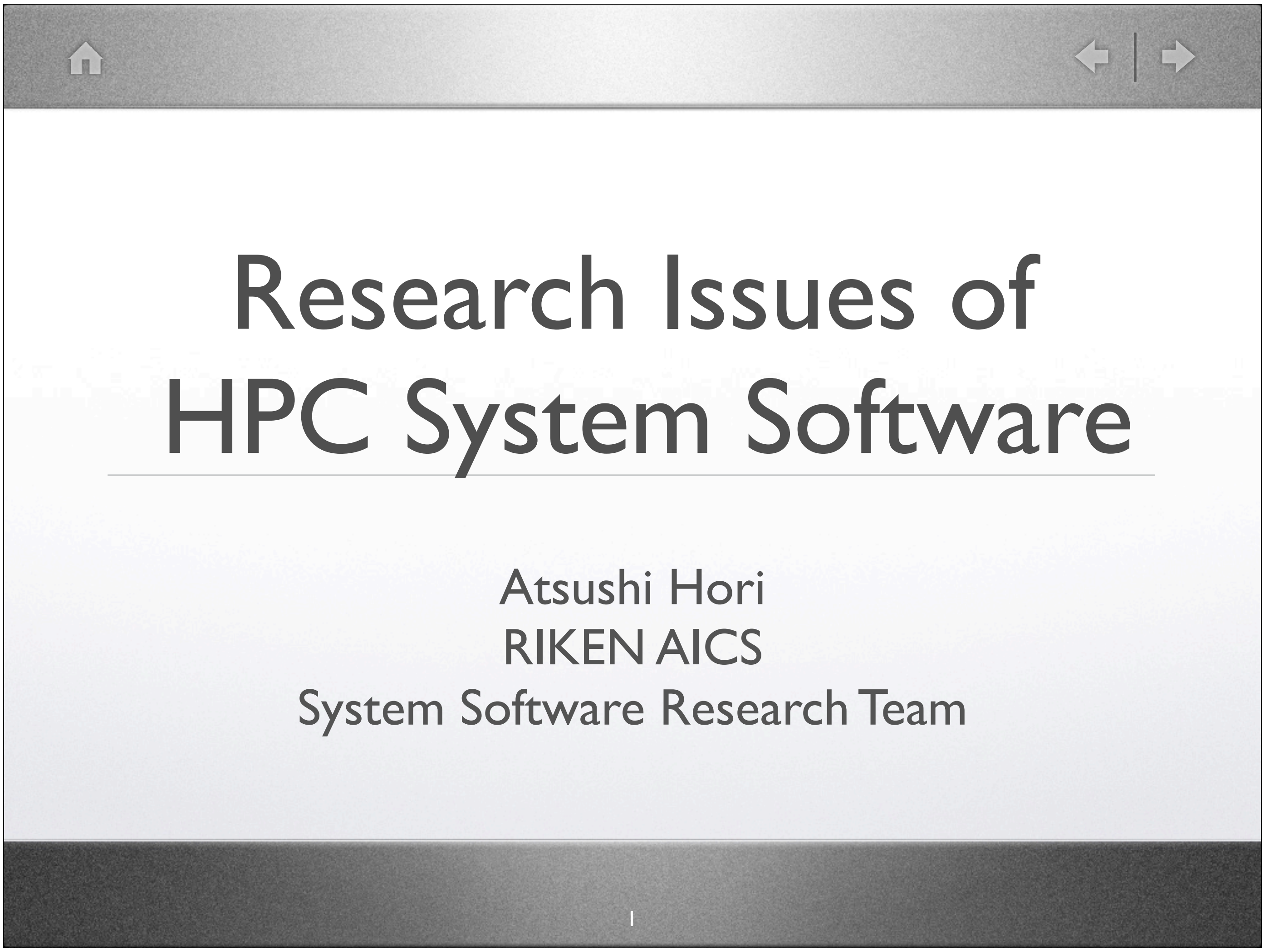
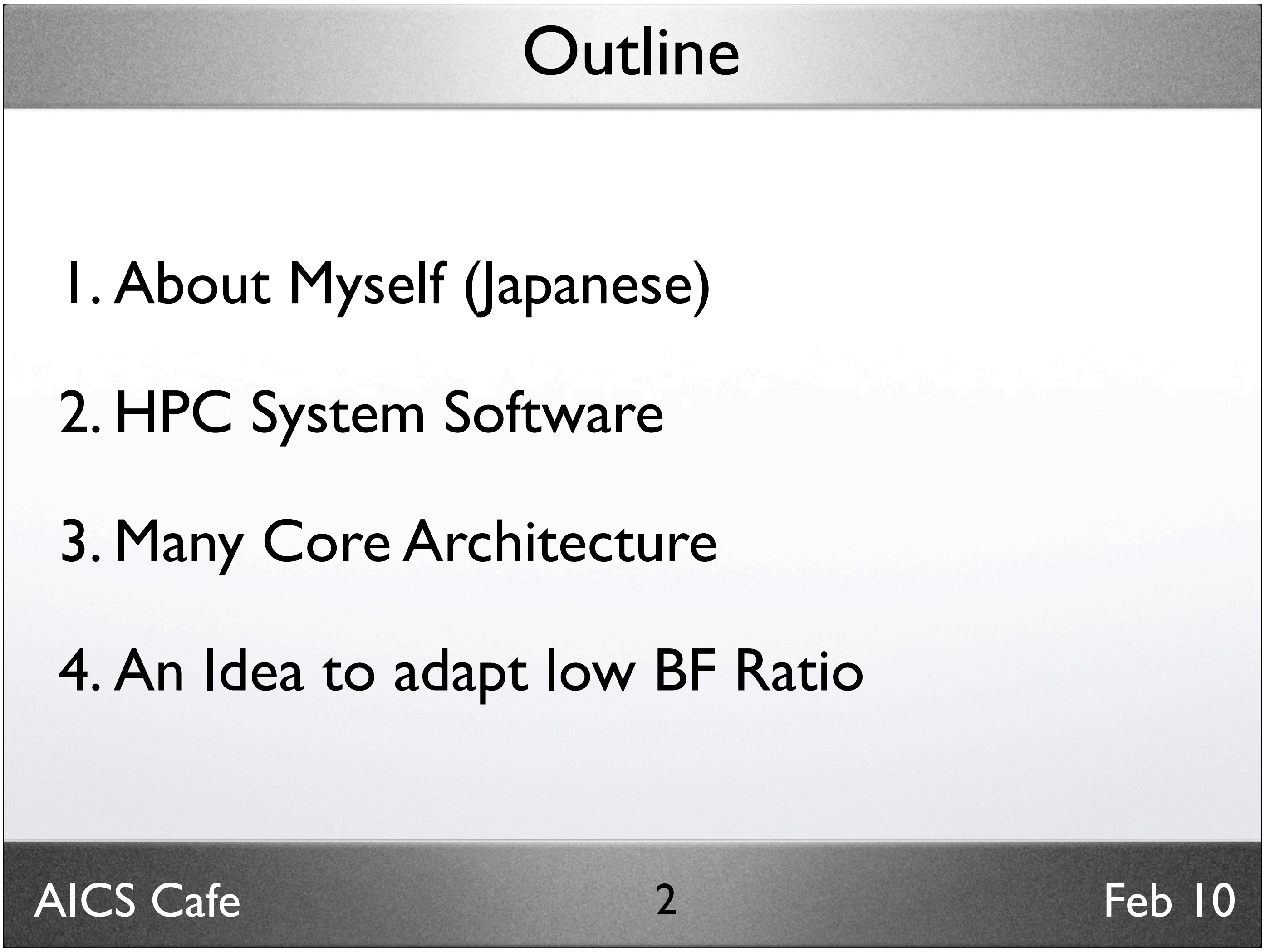


Research Issues of HPC System Software

Atsushi Hori
RIKEN AICS
System Software Research Team



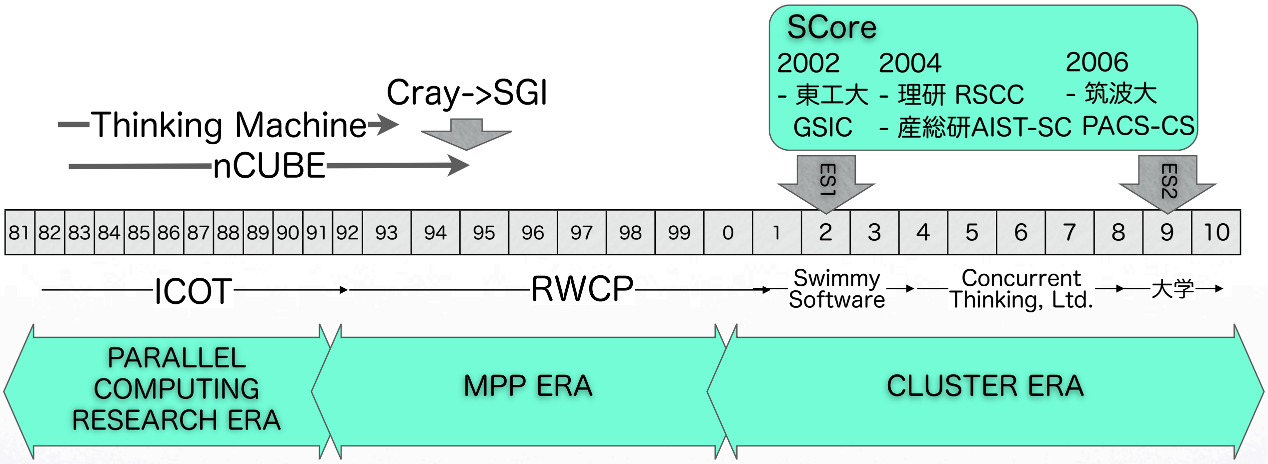
Outline

1. About Myself (Japanese)
2. HPC System Software
3. Many Core Architecture
4. An Idea to adapt low BF Ratio

Outline

1. About Myself (Japanese)

HPC and Myself



SCore
2002 - 東工大 GSIC
2004 - 理研 RSCC - 産総研AIST-SC
2006 - 筑波大 PACS-CS

ES1

ES2

Cray->SGI

Thinking Machine

nCUBE

81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 0 1 2 3 4 5 6 7 8 9 10

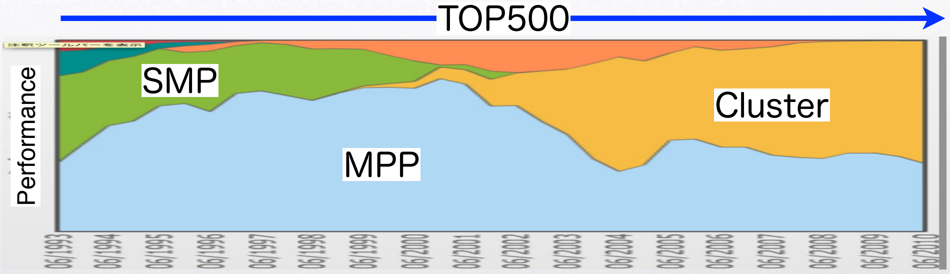
ICOT → RWCP → Swimmy Software → Concurrent Thinking, Ltd. → 大学

PARALLEL COMPUTING RESEARCH ERA

MPP ERA

CLUSTER ERA

TOP500



My First Super Computer

Ph.D Thesis

- Title: An Efficient Implementation of Time Sharing Scheduling for Distributed Memory Computers
- SCore *[es-core]* Cluster System Software
  - The second (and last ?) practical time sharing scheduling system for clusters
- Almost nobody wants to use this (T_T)
  - Weak Scaling => Exec. time does’t get shorter
  - Batch scheduling is enough

Outline

2. HPC System Software

Why are supercomputers so hard to use

- ★ Evolution is specialization
  - ★ No high performance, general purpose machines ever
  - ★ High performance machines are hard to use
- ★ Machines are tools
  - ★ Machines amplify human abilities
  - ★ Practices are needed to utilize them

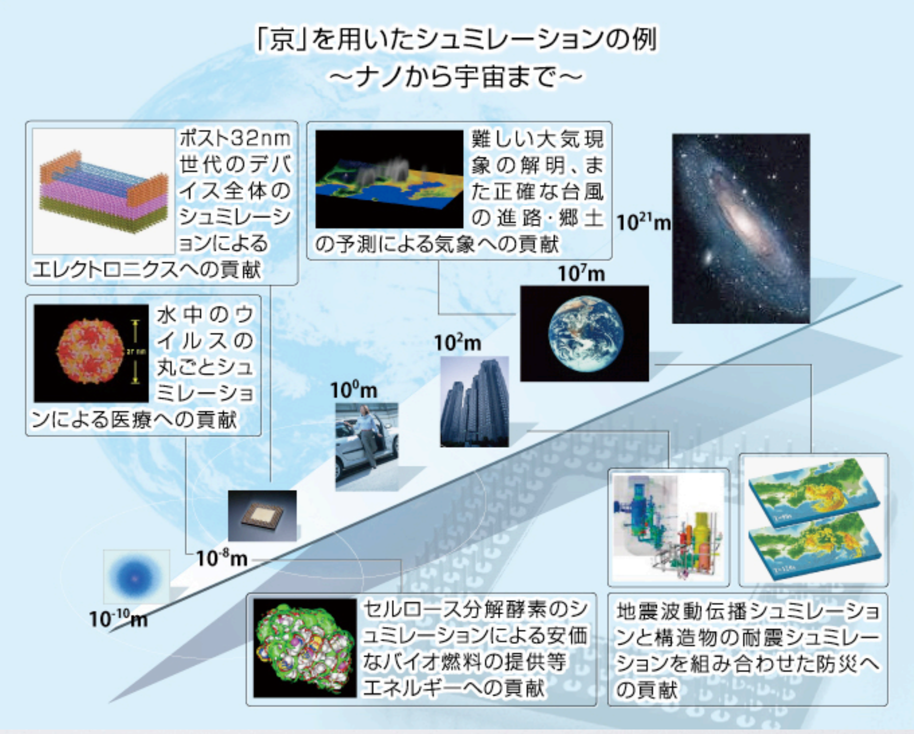
https://waterplanet.homeunix.net/~ohno/car-history/toyota_museeing.html

Where is System Software ?

- ★ Compuetr = Hardware + Software

©RIKEN

- ★ No pictures on System Software !!



「京」を用いたシュミレーションの例
～ナノから宇宙まで～

ポスト32nm世代のデバイス全体のシュミレーションによるエレクトロニクスへの貢献

難しい大気現象の解明、また正確な台風の進路・郷土の予測による気象への貢献

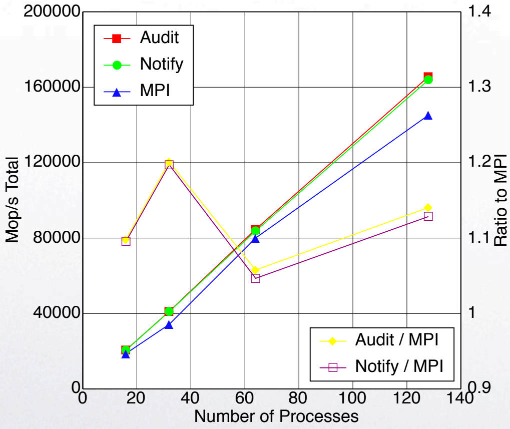
水中のウイルスの丸ごとシュミレーションによる医療への貢献

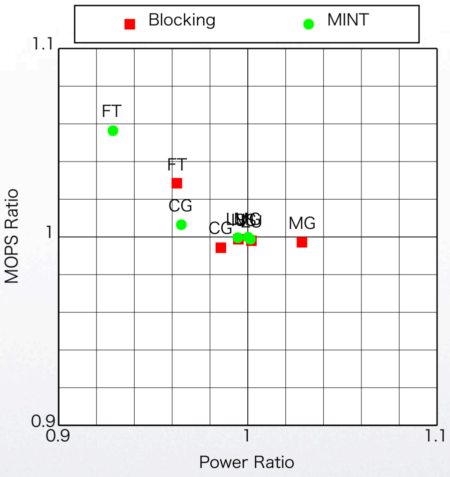
セルロース分解酵素のシュミレーションによる安価なバイオ燃料の提供等エネルギーへの貢献

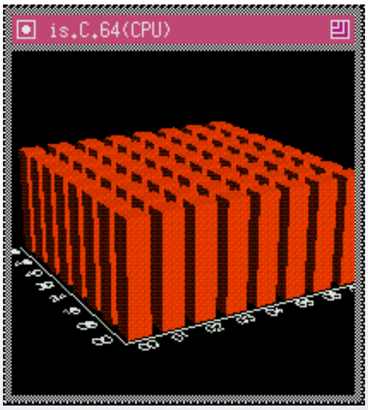
地震波動伝播シュミレーションと構造物の耐震シュミレーションを組み合わせた防災への貢献

Research on System Software

- ★ In many cases, small improvements are discussed.
- ★ No Scientific DISCOVERY !!
- ★ Not Continuous







System Software

- My definition of “System Software”
  - System software is a set of programs and libraries to make computers easy to use
    - Easy to program from the programmers view
    - Easy to run from the users view
    - Easy to operate from the operators view
    - ~~Easy to buy~~

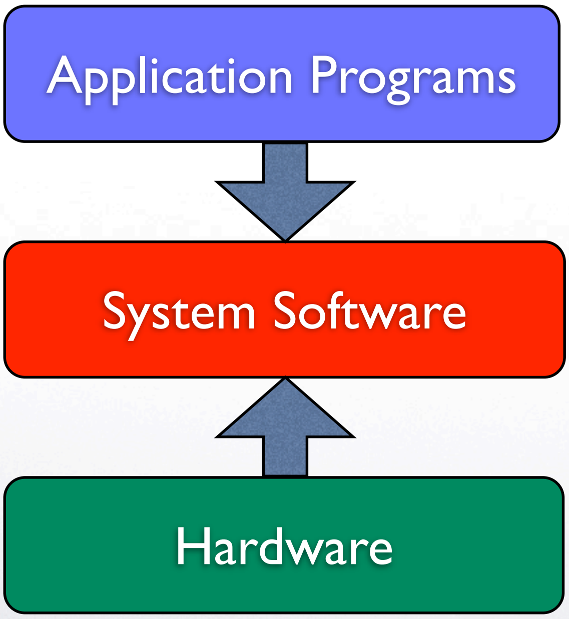
HPC System Software

- HPC
  - Speed, Speed, ..., and Speed followed by Speed
- HPC System Software
  - Control H/W and give users an abstracted H/W to make user programs run faster
  - Components
    - Operating System
    - Library
    - Tools
    - Language and Compiler
    - Mathematical Library, ...

HPC System Software

- ★ My definition of “HPC System Software”
  - ★ System software is a set of programs and libraries to make computers run faster
  - ★ even if it is harder to program, use and operate.

Position of System Software



Application Programs

System Software

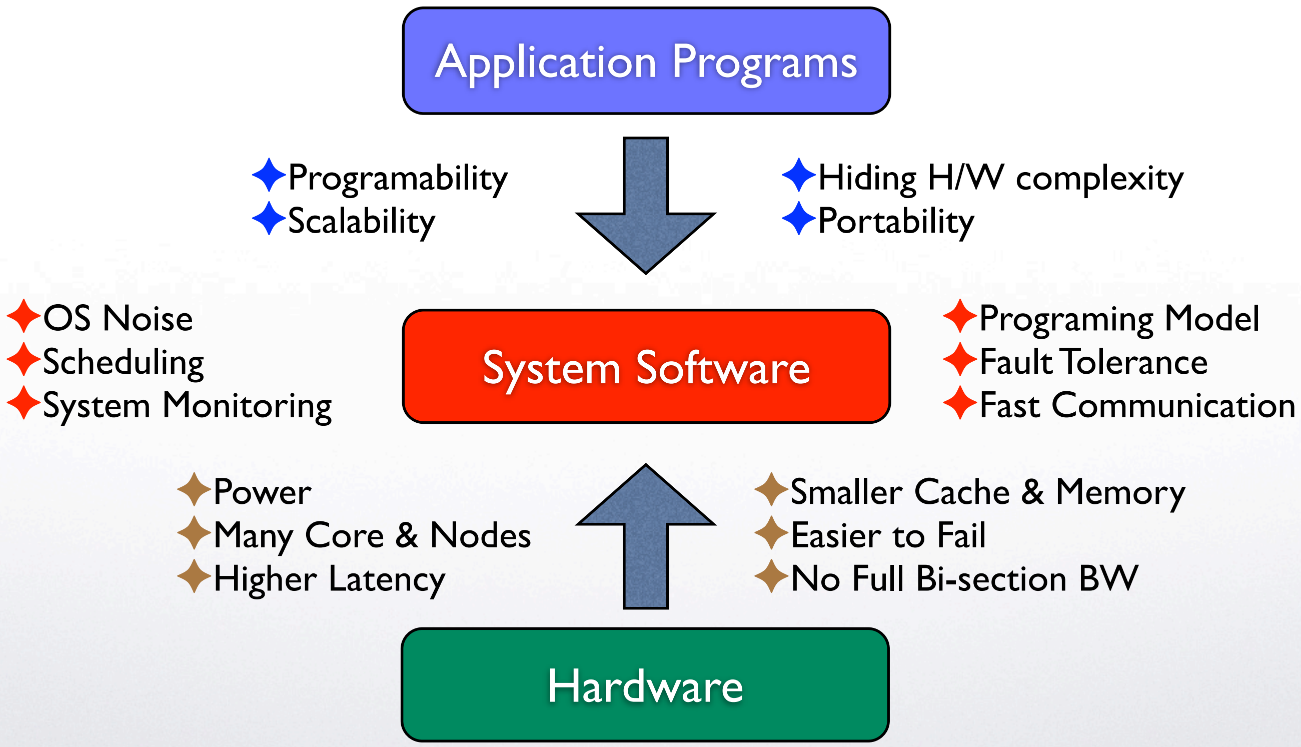
Hardware

Conservative

Hard to change

Radical

HPC System Software Issues



Application Programs

- Programability
- Scalability

- Hiding H/W complexity
- Portability

- OS Noise
- Scheduling
- System Monitoring

System Software

- Programing Model
- Fault Tolerance
- Fast Communication

- Power
- Many Core & Nodes
- Higher Latency

- Smaller Cache & Memory
- Easier to Fail
- No Full Bi-section BW

Hardware

Outline

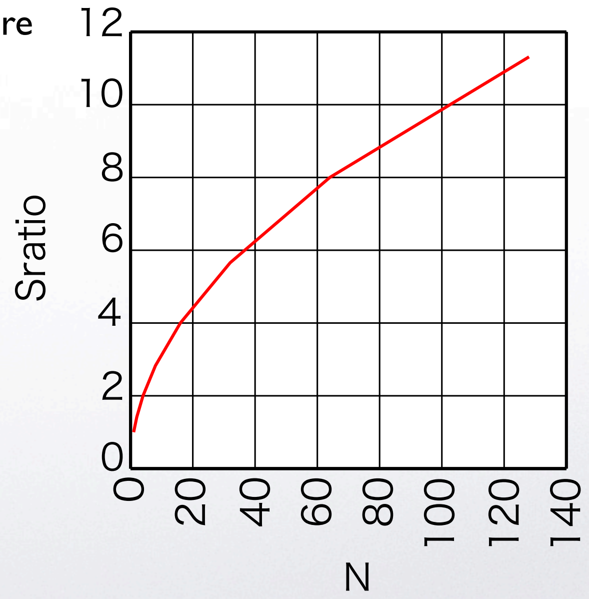
3. Many Core Architecture

Towards & Beyond Exa-Flops

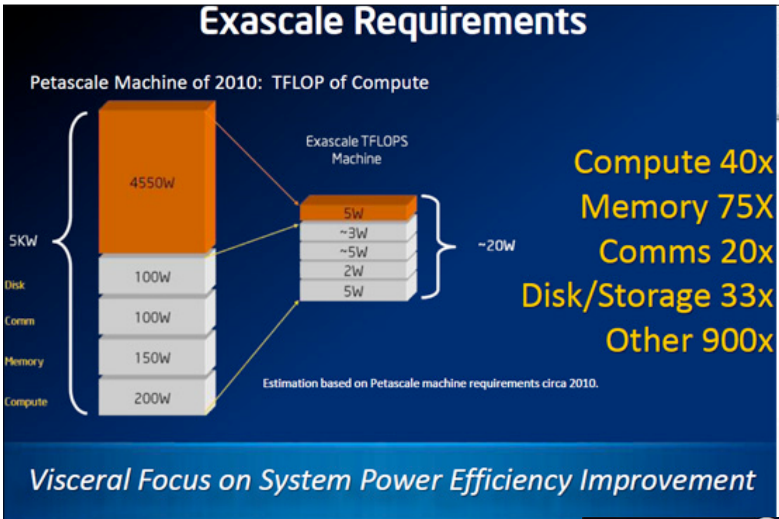
★ Number of Racks		1,000	(“K” : 864)
★ Number of Chips in a Rack		1,000	(“K” : 382)
★ Number of Cores in a Chip		100	(FX-10 : 16)
★ Total Number of Chips		10^6	
★ Chip Performance		10 TFLOPS	
★ Fujitsu FX10	236.5 GFLOPS		
★ Intel KnightsCorner	1 TFLOPS		
★ Total Peak Performance		10 EFLOPS	
★ Memory per Chip		1 TB	
★ Total Memory		1 EB (2^{60})	
★ Power Cap		20 MW (10 Watt/Chip)	
★ Fujitsu FX10	110 Watt		

Why Many Core Architecture ?

- ★ Rule of Thumb
  - ★ A: Chip (Die) Size, N : # Cores, S: Speed
  - ★ $N = A / A_{core}$, $S = N * S_{core}$
  - ★ $S_{core} = \alpha \ \mathrm{sqrt}(A_{core})$
  - ★ $S = \alpha \ N \ \mathrm{sqrt}(A_{core})$
  - ★ $Sr(N) = S_N / S_1 = N \ \mathrm{sqrt}(1 / N)$
- ★ Intel Knights series



Intel MICs



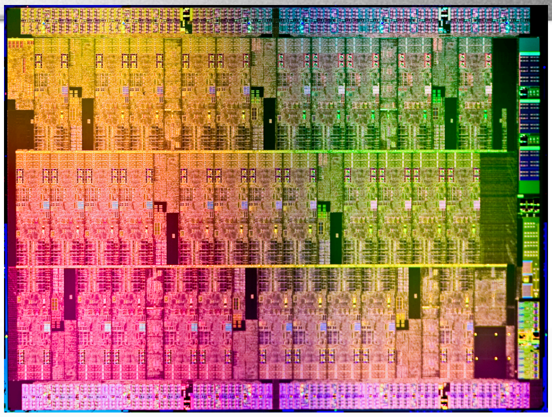
Exascale Requirements

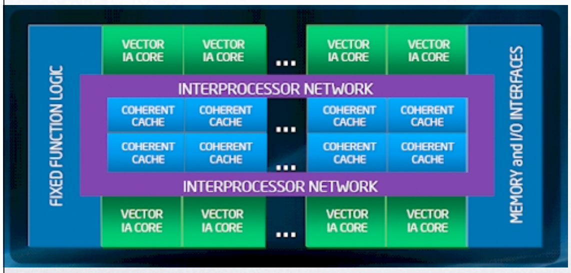
Petascale Machine of 2010: TFLOP of Compute

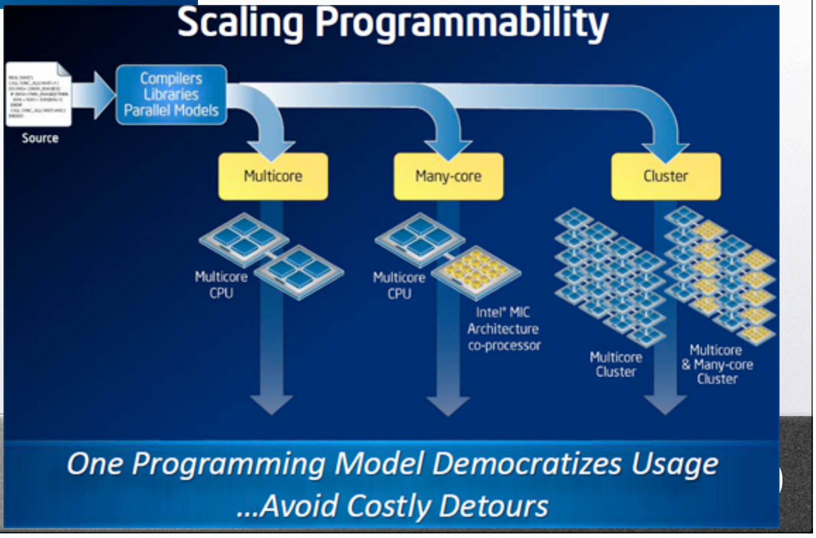
Compute 40x
Memory 75X
Comms 20x
Disk/Storage 33x
Other 900x

Estimation based on Petascale machine requirements circa 2010.

Visceral Focus on System Power Efficiency Improvement







Scaling Programmability

One Programming Model Democratizes Usage
...Avoid Costly Detours

MIC vs. GPGPU

- ★ MIC
  - ★ Richer ISA
  - ★ OS can run
    - ★ Bootable (theoretically)
    - ★ Dense Installation
  - ★ Higher Portability
  - ★ Intel KnightsCorner
    - ★ 1 TFLOPS (R*8)

- ★ GPGPU
  - ★ SIMD ISA
  - ★ OS cannot run
    - ★ Unable to boot itself Need of Control CPU
    - ★ Coarser Installation
  - ★ Lower Portability
  - ★ nVidia Tesla C2075
    - ★ 0.5 TFLOPS (R*8)

Outline

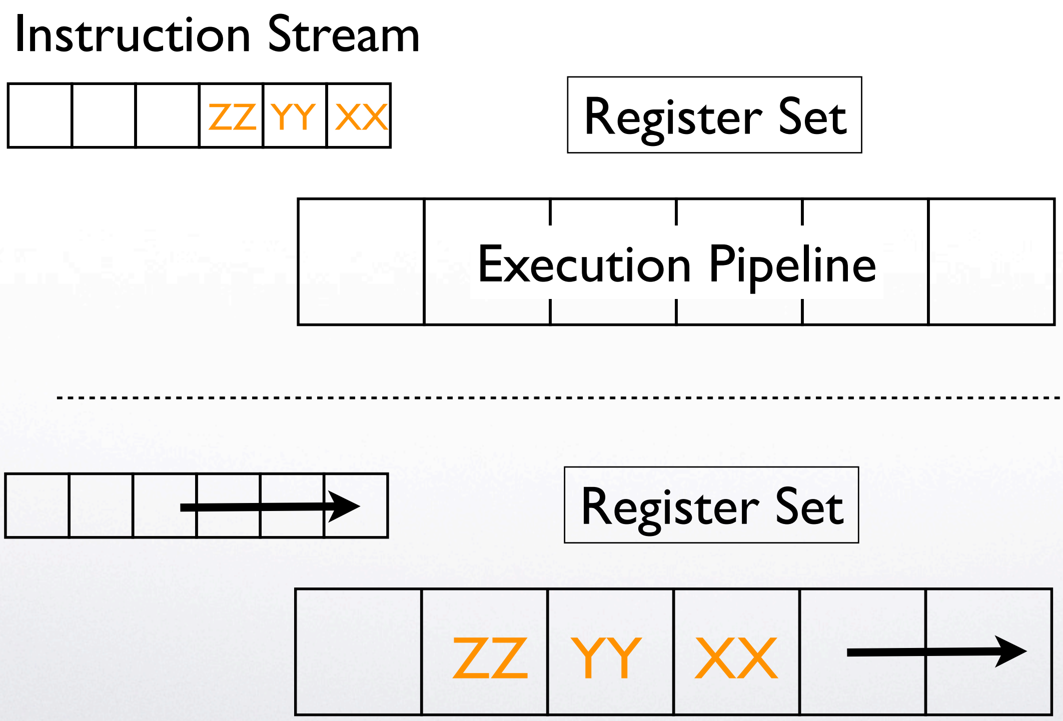
4. An Idea to adapt low BF Ratio

Just an Idea ...

- Scale (Hetale) Co-Design
- Climate Code
  - Stencil computation requires high BF value.
  - ClimateCode = DynamicsCode + PhysicsComp
    - DynamicsCode Memory Bound
    - PhysicsCode Compute Bound

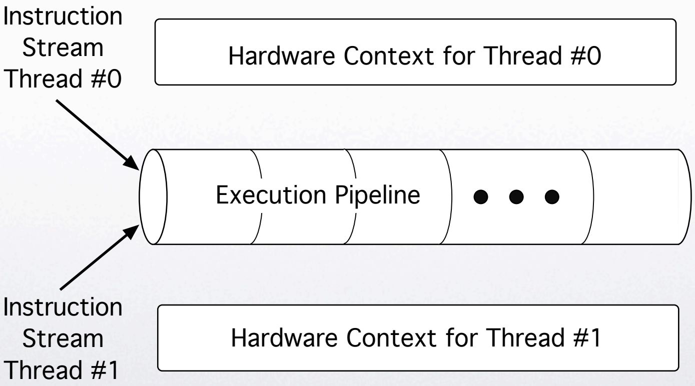
💡 What if DynamicsCode and PhysicsCode are blended ?

Execution Pipeline

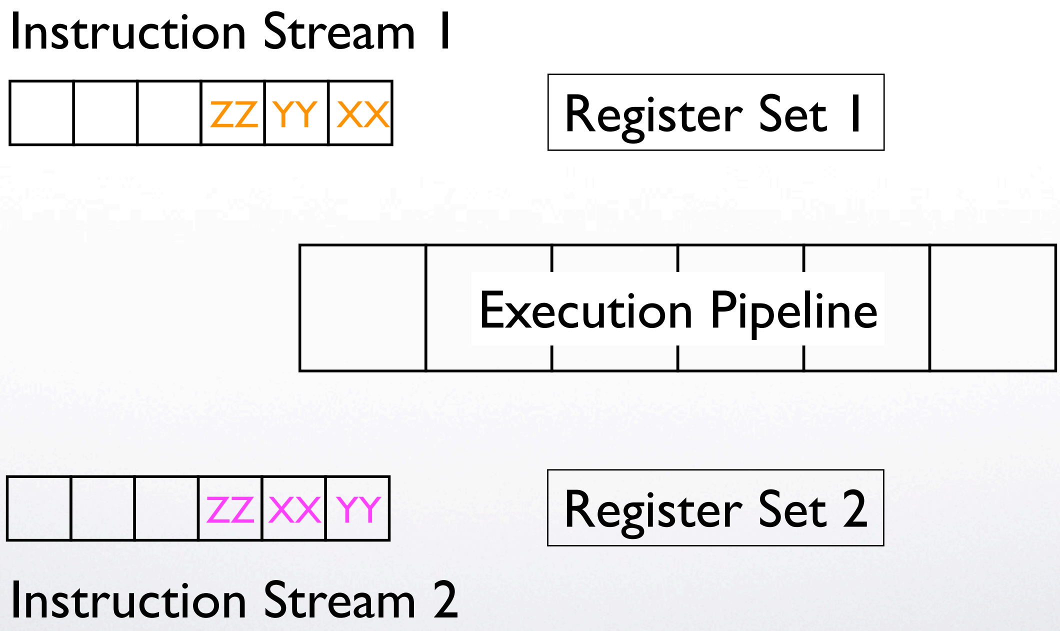


Simultaneous Multithreading

- ★ Hyper Threading (Intel)
  - ★ "3 % H/W investment improves 10 % performance"
- ★ So far, HPC system software almost ignore this.
  - ★ Cache size is the same while working set can get larger.


Instruction Stream Thread #0
Hardware Context for Thread #0
Execution Pipeline
• • •
Instruction Stream Thread #1
Hardware Context for Thread #1

SMT (1)



Instruction Stream 1

ZZ YY XX

Register Set 1

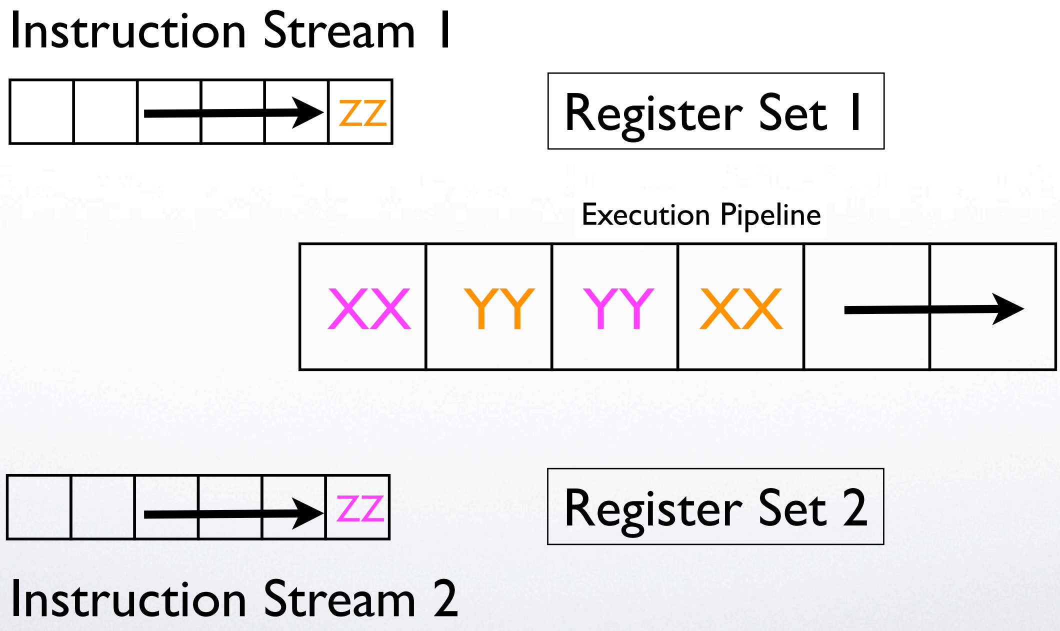
Execution Pipeline

ZZ XX YY

Register Set 2

Instruction Stream 2

SMT (2)



Instruction Stream 1

ZZ

Register Set 1

Execution Pipeline

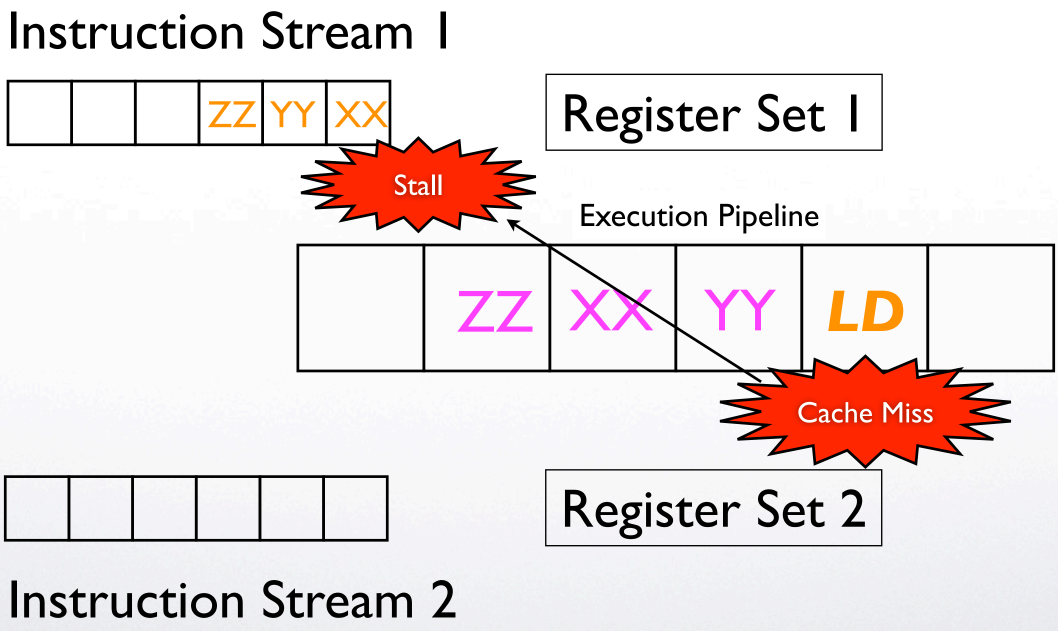
XX YY YY XX

ZZ

Register Set 2

Instruction Stream 2

SMT (2)



Instruction Stream 1

ZZ YY XX

Register Set 1

Stall

Execution Pipeline

ZZ XX YY LD

Cache Miss

Register Set 2

Instruction Stream 2

Load Blending

- ★ Utilizing SMT to increase performance
  - ★ Pipeline stall bubbles can be filled with the other SMT.
  - ★ If MB thread and CB thread are mixed
    - ★ then efficiency might be improved
- ★ Shadow Thread
  - ★ A thread library to utilize SMT

Load Blending

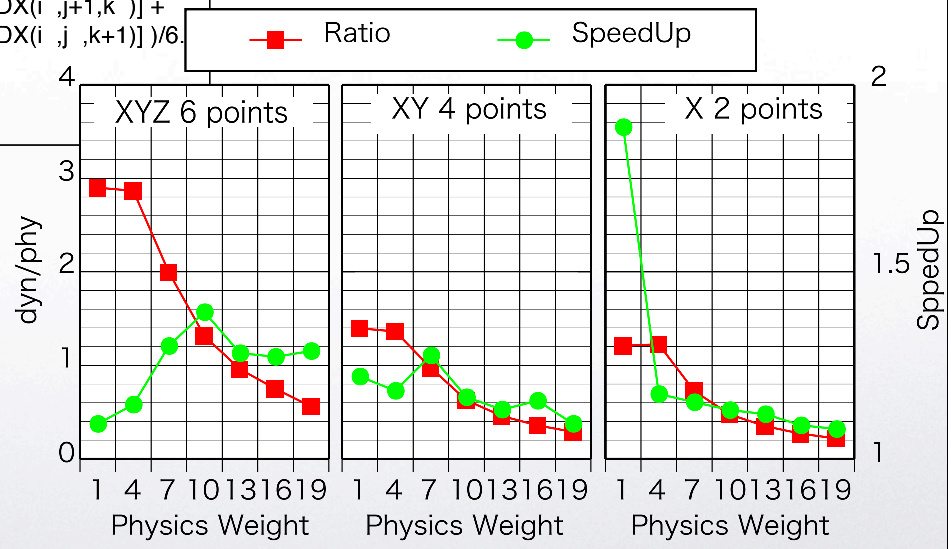
Dynamics code (XYZ)

```
for( k=1; k<SZ-1; k++ ) {
  for( j=1; j<SZ-1; j++ ) {
    for( i=1; i<SZ-1; i++ ) {
      array[IDX(i,j,k)] =
        ( array[IDX(i-1,j  ,k  )] + array[IDX(i+1,j  ,k  )] +
          array[IDX(i  ,j-1,k  )] + array[IDX(i  ,j+1,k  )] +
          array[IDX(i  ,j  ,k-1)] + array[IDX(i  ,j  ,k+1)] )/6.
    }
  }
}
```

Physics code

```
for( k=0; k<SZ; k++ ) {
  for( j=0; j<SZ; j++ ) {
    for( i=0; i<SZ; i++ ) {
      x = array[IDX(i,j,k)];
      for( l=0; l<m; l++ )  {
        x = ( x + 1.2 ) * 0.9;
      }
      array[IDX(i,j,k)] = x;
    }
  }
}
```

Xeon X5560, 2.8 GHz
2 Way SMT



Welcome to the CS world !

- ★ What does “CS” stand for ?
  - ★ Computer Science
  - ★ Complex (and/or Chaos) System
- ★ Indeed, current HPC systems are TOO COMPLEX
  - ★ No simple assumption hits the point !
  - ★ DATA can tell the truth
    - ★ The phenomenon hard to explain is the point !
  - ★ Trial-and-error can lead us to the point
    - ★ Never give up !