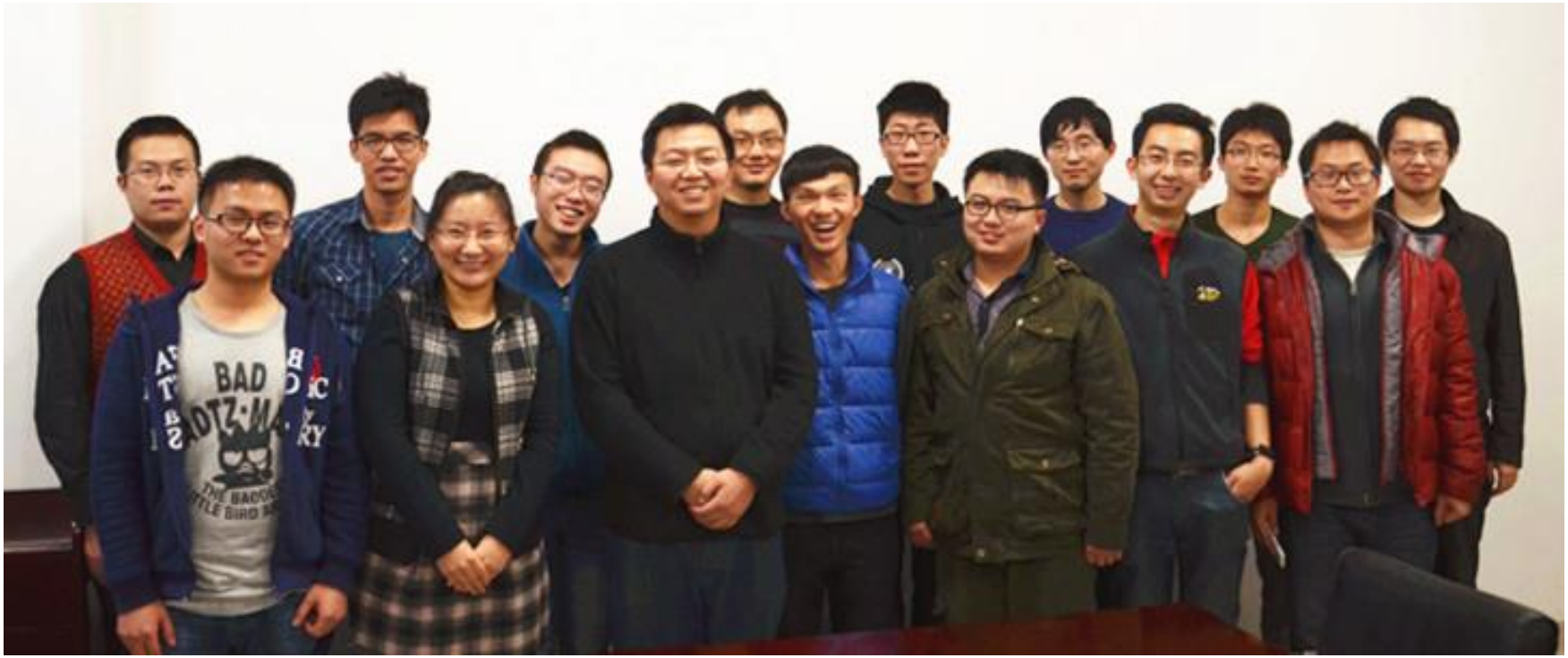
Unleashing the Performance Potential of GPUs for Atmospheric Dynamic Solvers

Haohuan Fu
haohuan@tsinghua.edu.cn
High Performance Geo-Computing (HPGC) Group
Center for Earth System Science
Tsinghua University

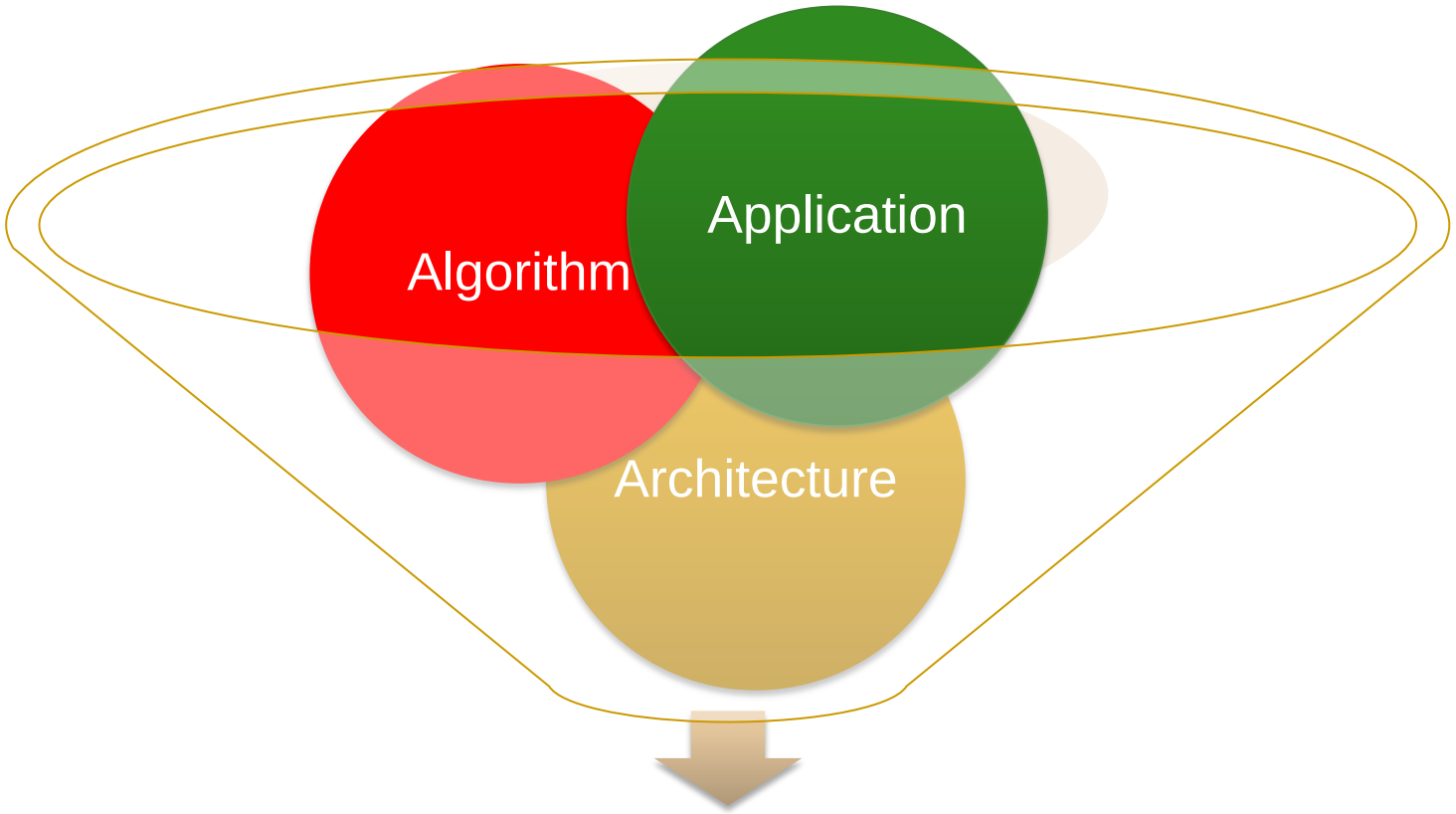
April/5th/2016

Tsinghua HPGC Group

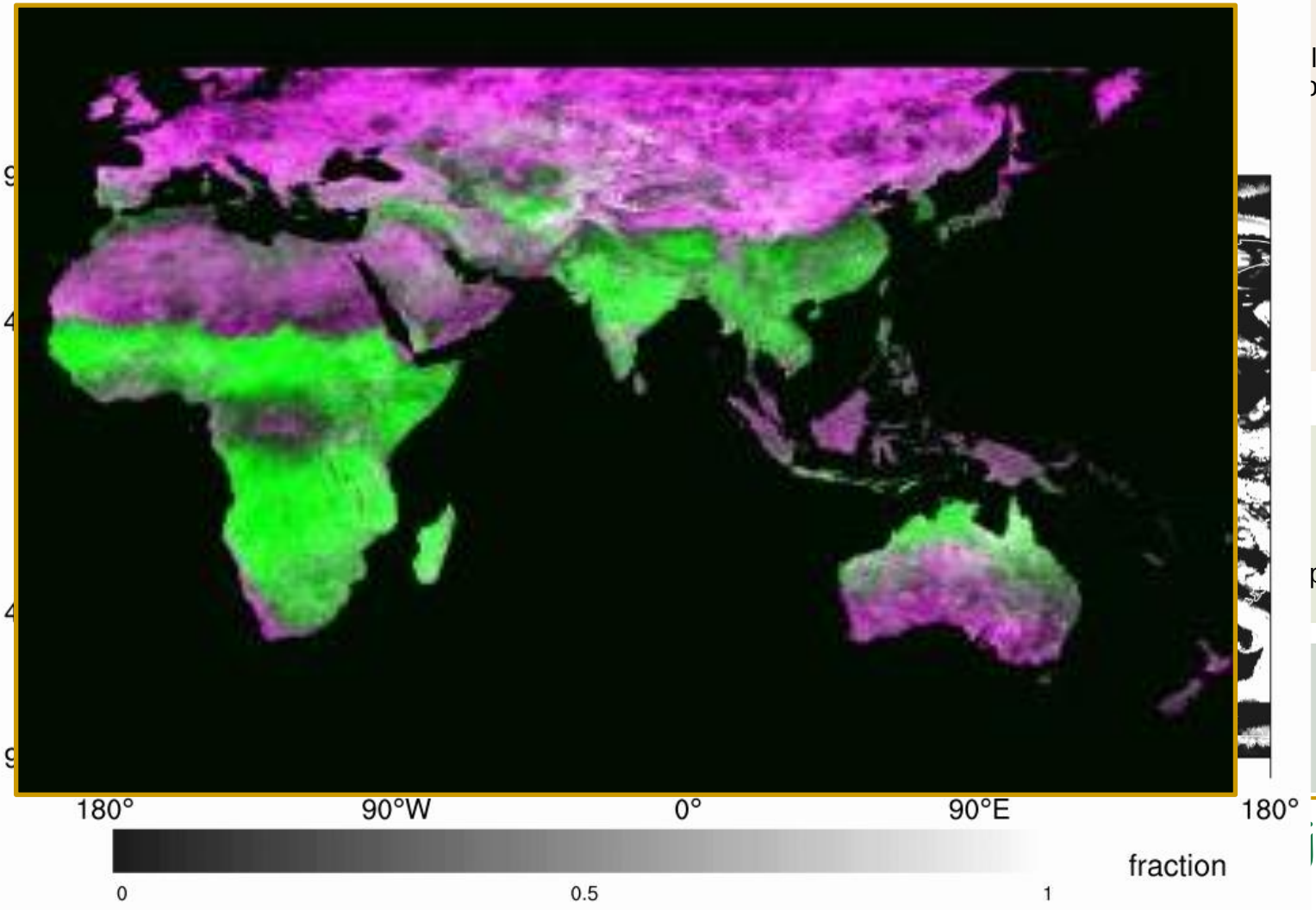
- HPGC: high performance geo-computing http://www.thuhpgc.org



A Design Process That Combines Optimizations from Different Layers

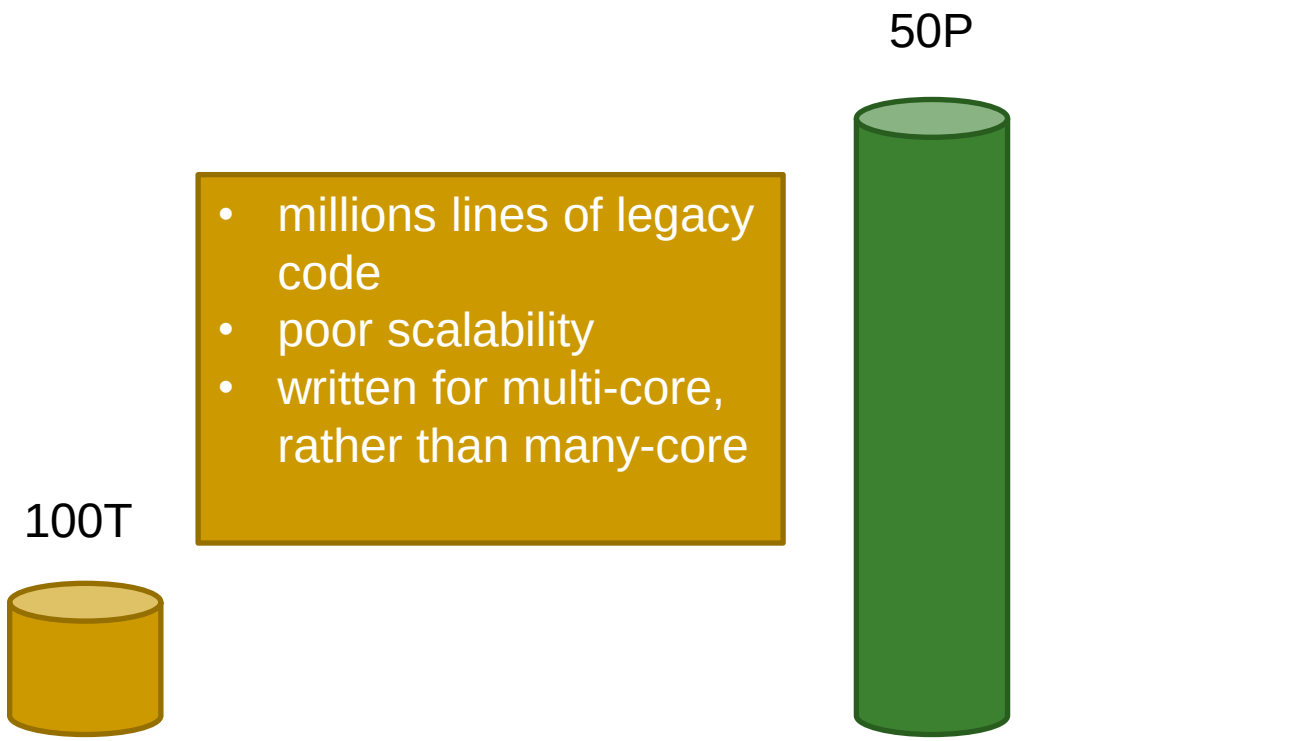


- Exploration Geophysics
  - GPU-based BEAM Migration (sponsored by Statoil)
  - GPU-based ETE Forward Modeling (sponsored by BGP)
  - Parallel Finite Element Electromagnetic Forward Modeling Method (sponsored



A Highly Scalable Framework for Atmospheric Modeling on Heterogeneous Supercomputers

The Gap between Software and Hardware



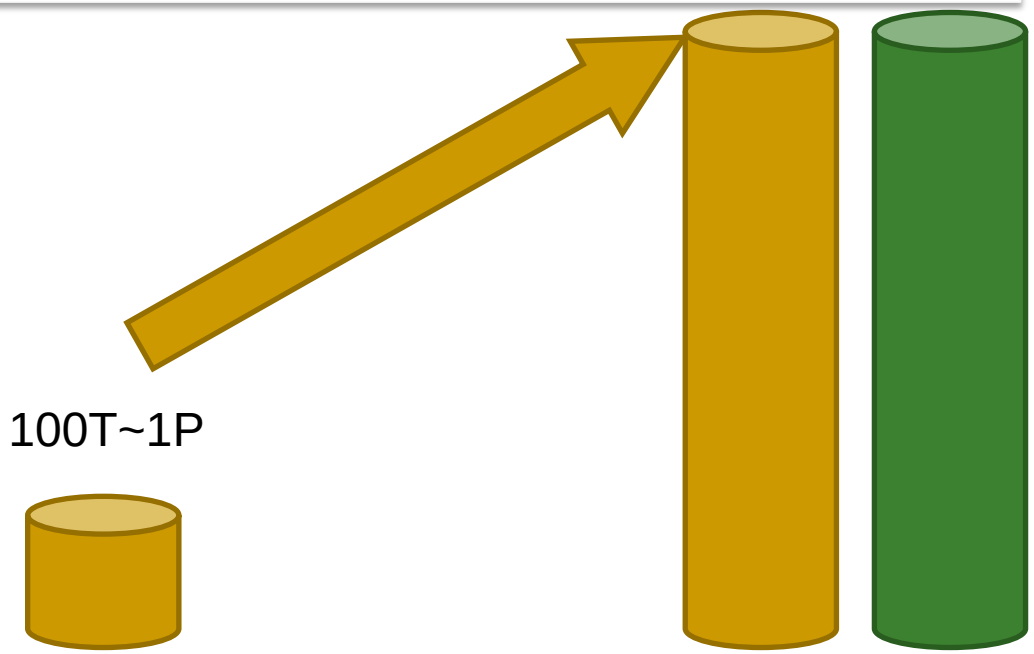
- millions lines of legacy code
- poor scalability
- written for multi-core, rather than many-core

China's models
- pure CPU code
- scaling to hundreds or thousands of cores

China's supercomputers
- heterogeneous systems with GPUs or MICs
- millions of cores

Our Research Goals

- highly scalable framework that can efficiently utilize many-core accelerators
- automated tools to with the legacy code

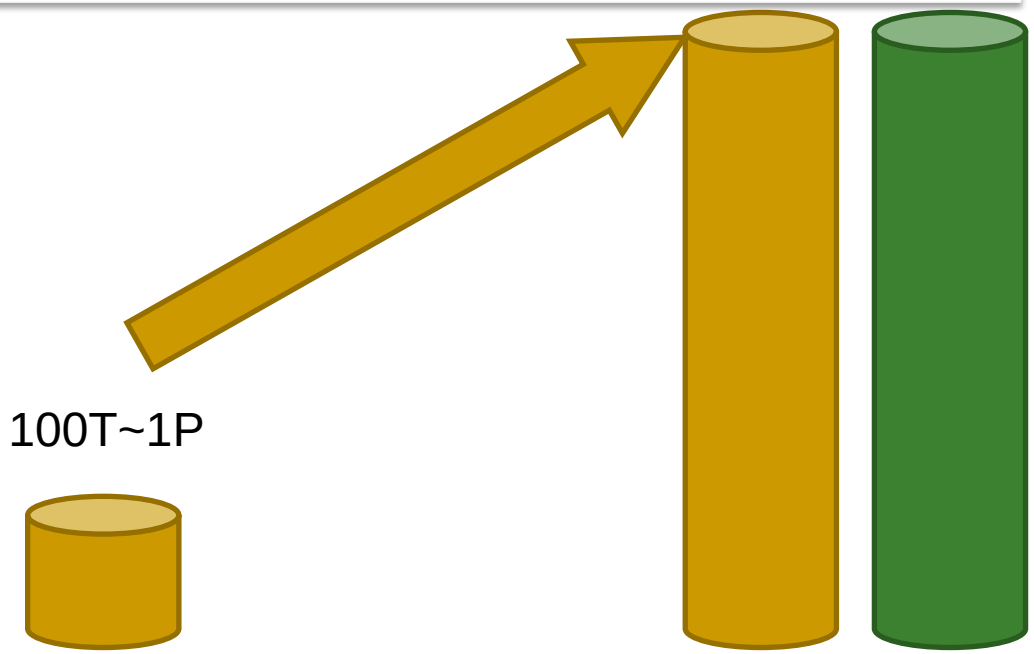


China's models
- pure CPU code
- scaling to hundreds or thousands of cores

China's supercomputers
- heterogeneous systems with GPUs or MICs
- millions of cores

Our Research Goals

- highly scalable framework that can efficiently utilize many-core accelerators
- automated tools to with the legacy code



China's models
- pure CPU code
- scaling to hundreds or thousands of cores

China's supercomputers
- heterogeneous systems with GPUs or MICs
- millions of cores

Example: Highly-Scalable Atmospheric Simulation Framework

Yang, Chao
Institute of Software, CAS
computational mathematics

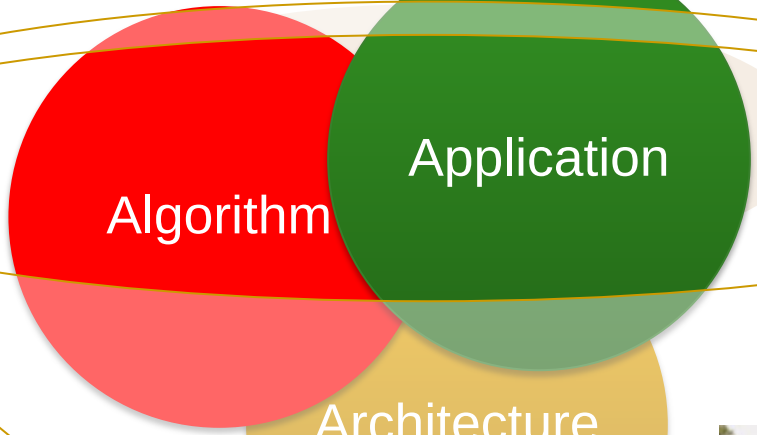
cube-sphere grid or other grid

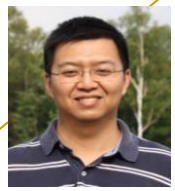
explicit, implicit, or semi-implicit method

cloud resolving

Wang, Lanning
Beijing Normal University
climate modeling

Xue, Wei
Tsinghua University
computer science





Fu, Haohuan
Tsinghua University
geo-computing

CPU, GPU, MIC, FPGA

C/C++, Fortran, MPI, CUDA, Java, …

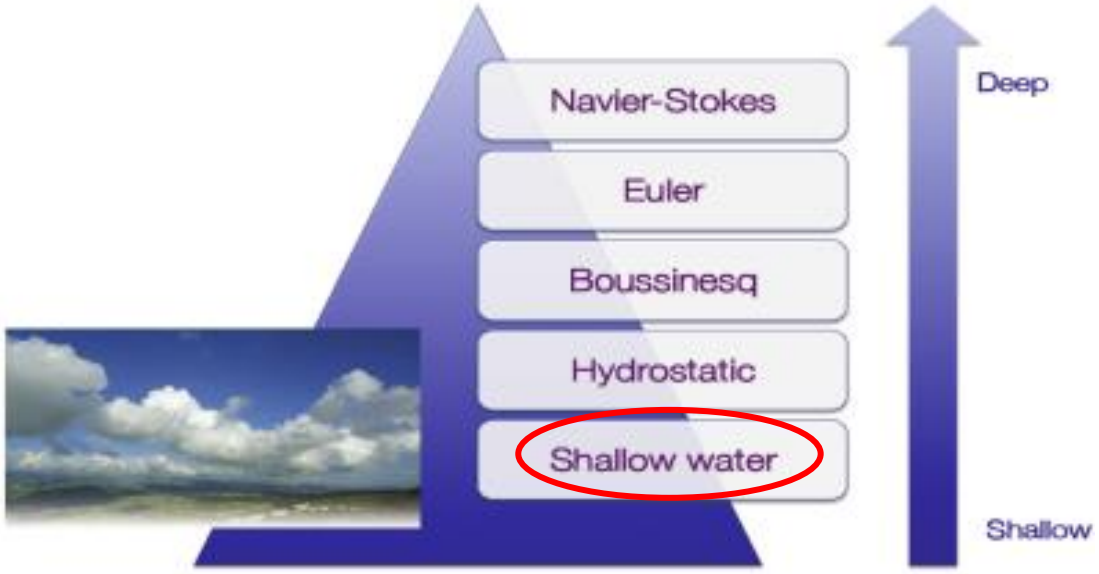
The “Best” Computational Solution

A Highly Scalable Framework for Atmospheric Modeling on Heterogeneous Supercomputers:

Previous Efforts

Highly-Scalable Framework for Atmospheric Modeling

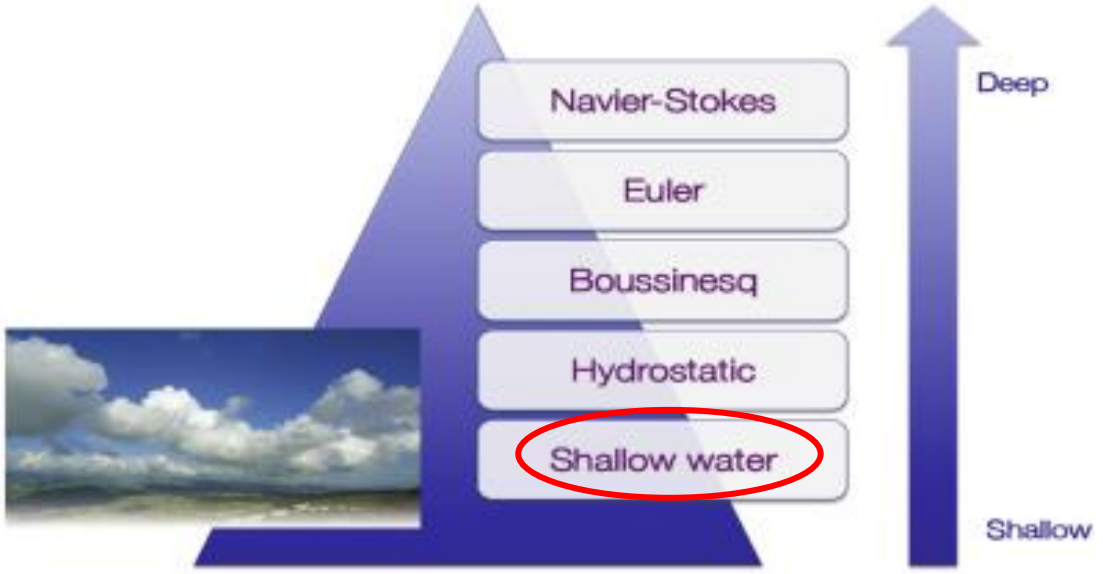
- 2012: solving 2D SWE using CPU + GPU
  - 800 Tflops on 40,000 CPU cores, and 3750 GPUs



For more details, please refer to our PPoPP 2013 paper: “A Peta-Scalable CPU-GPU Algorithm for Global Atmospheric Simulations”, in *Proceedings of the 18th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, pp. 1-12, Shenzhen, 2013. .

Highly-Scalable Framework for Atmospheric Modeling

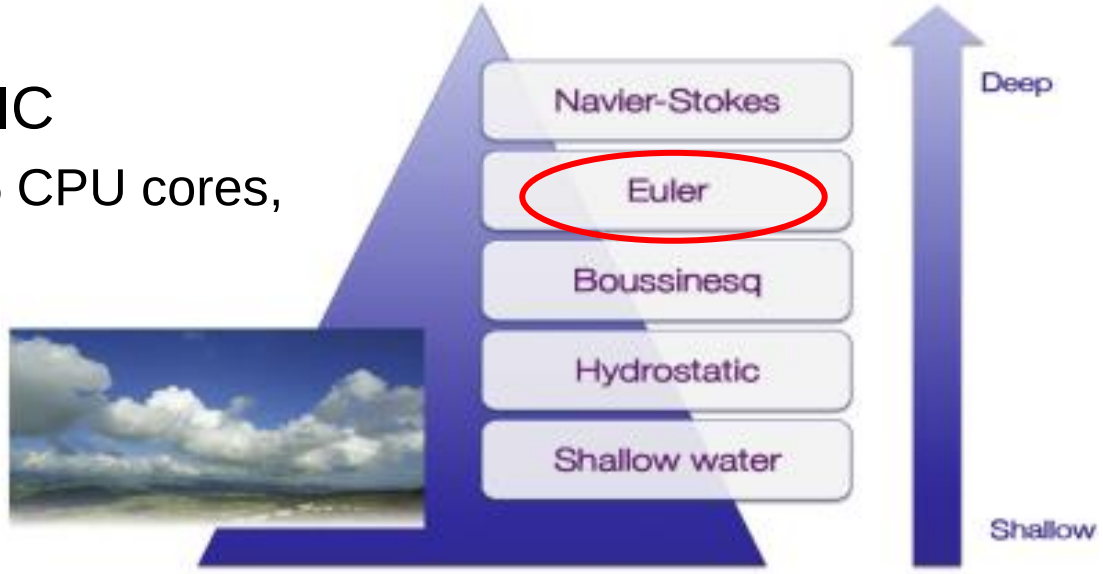
- 2012: solving 2D SWE using CPU + GPU
  - 800 Tflops on 40,000 CPU cores, and 3750 GPUs
- 2013: 2D SWE on MIC and FPGA
  - 1.26 Pflops on 207,456 CPU cores, and 25,932 MICs
  - another 10x on FPGA



For more details, please refer to our IPDPS 2014 paper: "Enabling and Scaling a Global Shallow-Water Atmospheric Model on Tianhe-2"; and our FPL 2013 paper: "Accelerating Solvers for Global Atmospheric Equations Through Mixed-Precision Data Flow Engine".

Highly-Scalable Framework for Atmospheric Modeling

- 2012: solving 2D SWE using CPU + GPU
  - 800 Tflops on 40,000 CPU cores, and 3750 GPUs
- 2013: 2D SWE on CPU+MIC and CPU+FPGA
  - 1.26 Pflops on 207,456 CPU cores, and 25,932 MICs
  - another 10x on FPGA
- 2014: 3D Euler on MIC
  - 1.7 Pflops on 147,456 CPU cores, and 18,432 MICs



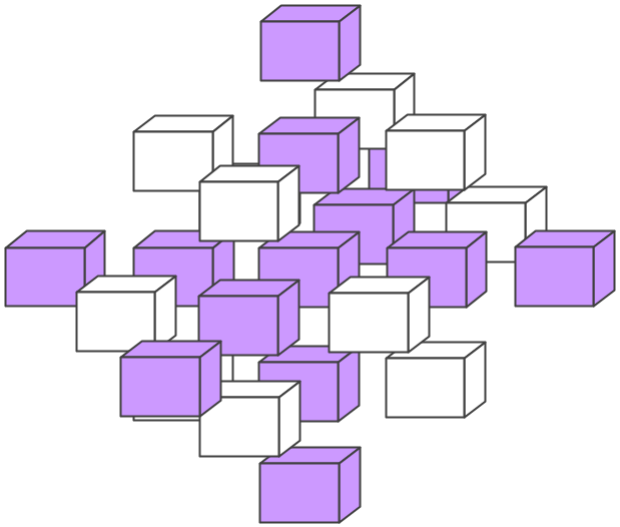
For more details, please refer to our paper: “Ultra-scalable CPU-MIC Acceleration of Mesoscale Atmospheric Modeling on Tianhe-2”, IEEE Transaction on Computers.

A Highly Scalable Framework for Atmospheric Modeling on Heterogeneous Supercomputers:

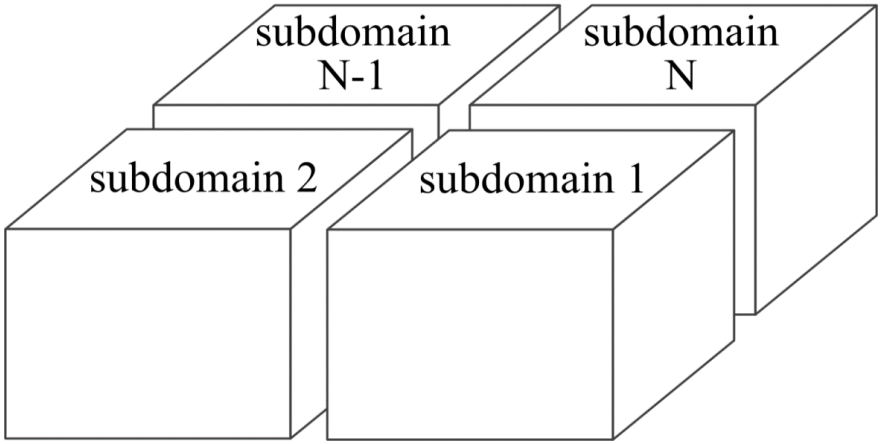
3D Euler on CPU+GPU

CPU-only Algorithm

- Parallel Version
  - Multi-node & Multi-core
  - MPI Parallelism



25 points stencil



3D channel

CPU-only Algorithm

- Parallel Version
  - *Multi-node & Multi-core*
  - *MPI Parallelism*
- CPU Algorithm
  - *Workflow*

CPU Algorithm per Stencil sweep

For each subdomain

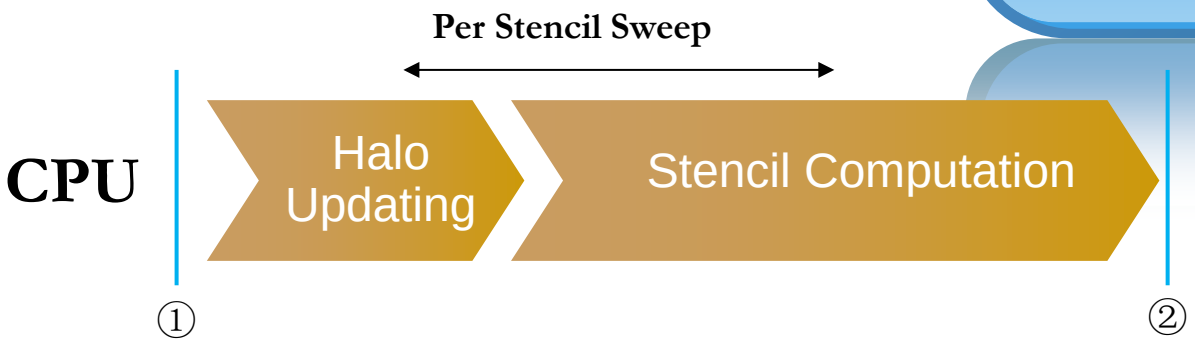
① Update Halo

② Calculate Euler stencil

a. Compute Local Coordinate

b. Compute Fluxes

c. Compute Source Terms



CPU Workflow

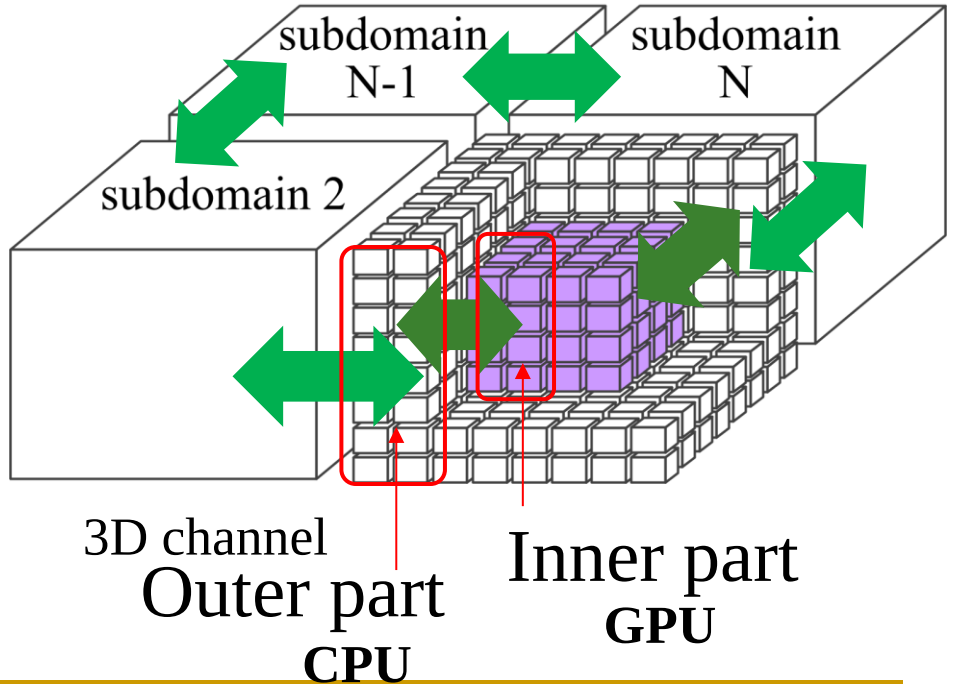
Hybrid (CPU+GPU) Algorithm

- Hybrid Partition
  - GPU → Inner Stencil Computation
  - CPU → Halo Updating & Outer Stencil Computation

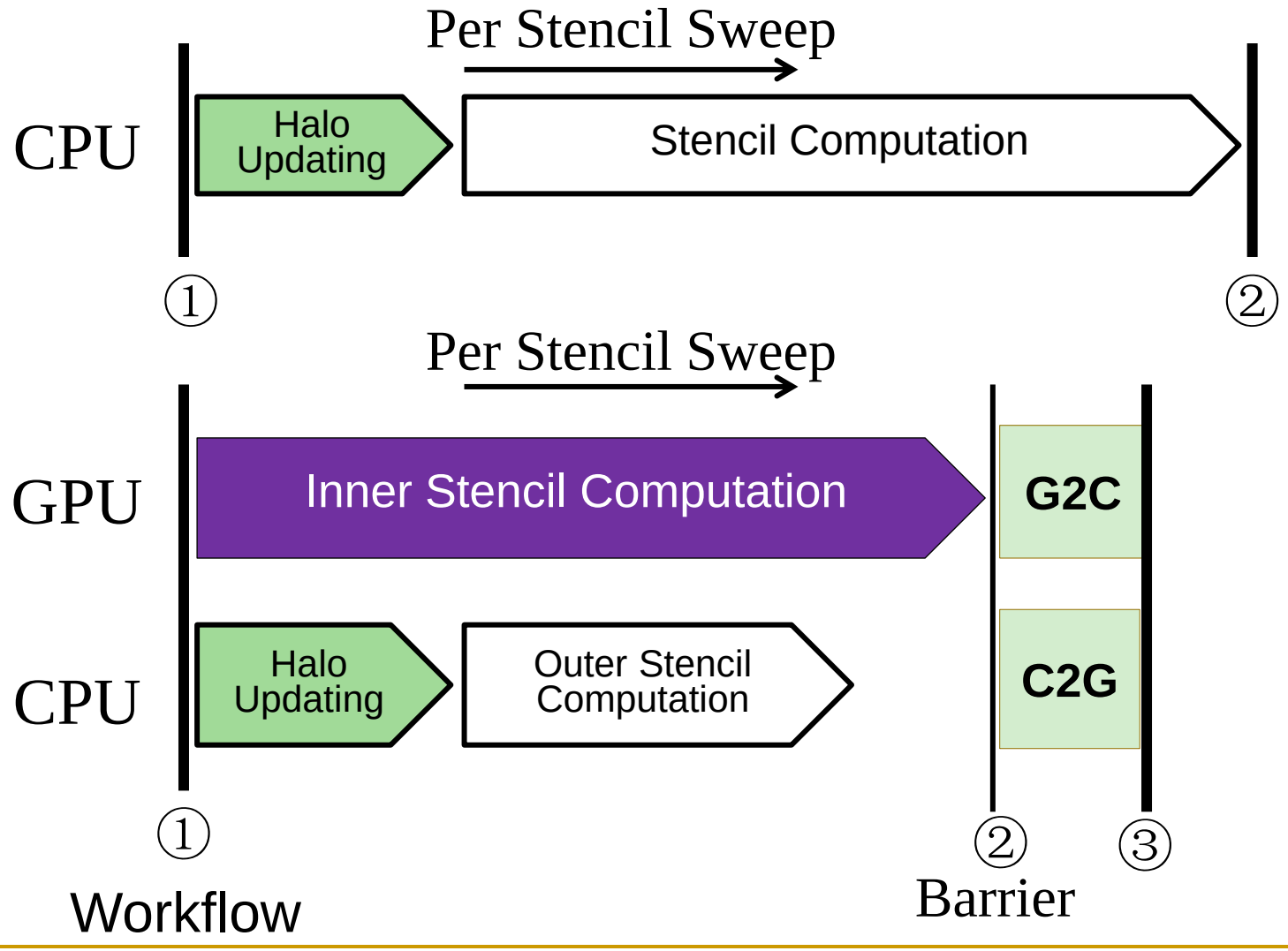
- CPU-GPU Hybrid Algorithm

❖*CPU-GPU Hybrid Algorithm Per Stencil Sweep*

For each subdomain
GPU side: *PETSc*
Inner-part Euler Stencil
CPU side:
① Update Halo
② Outer-part Euler stencil
BARRIER *4 layers*
CPU-GPU Exchange



Hybrid Algorithm Design



A Highly Scalable Framework for Atmospheric Modeling on Heterogeneous Supercomputers:

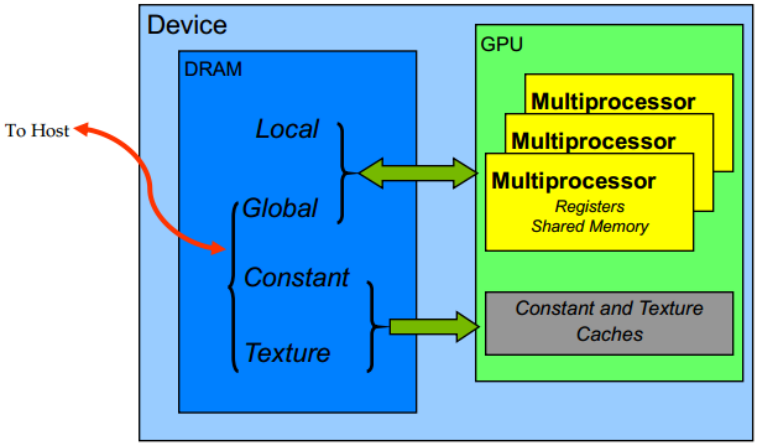
GPU-related Optimizations

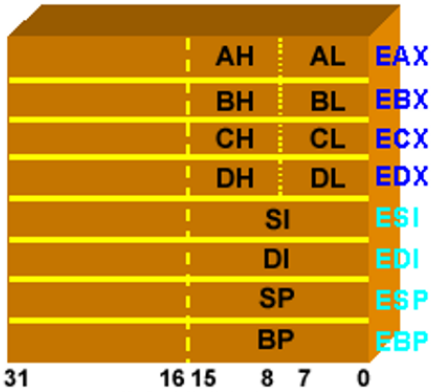
Optimizations

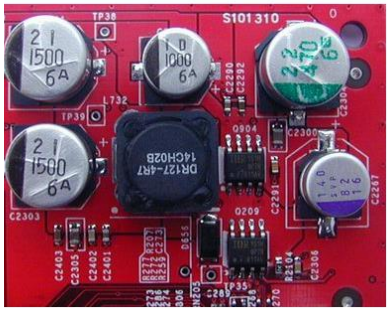
GPU Opt	Pinned Memory
	SMEM/L1
	AoS -> SoA
	Register Adjustment
	Kernel Splitting
	Other Methods
	Customizable Data Cache
	Inner-thread Rescheduling
CPU Opt	OpenMP
	SIMD Vectorization
	Cache blocking

Optimizations

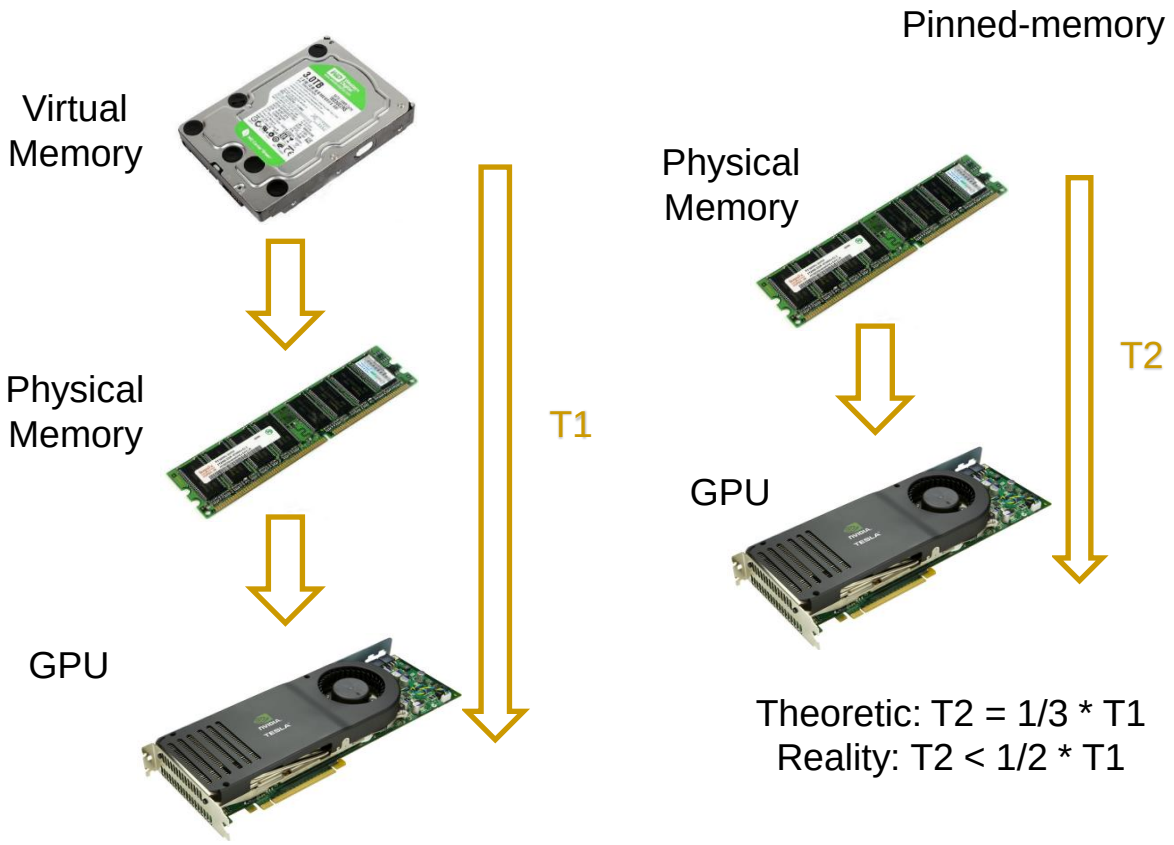
GPU Opt	**Pinned Memory**
	SMEM/L1
	AoS -> SoA
	Register Adjustment
	Kernel Splitting
	Other Methods
	Customizable Data Cache
	Inner-thread Rescheduling
CPU Opt	**OpenMP**
	SIMD Vectorization
	Cache blocking





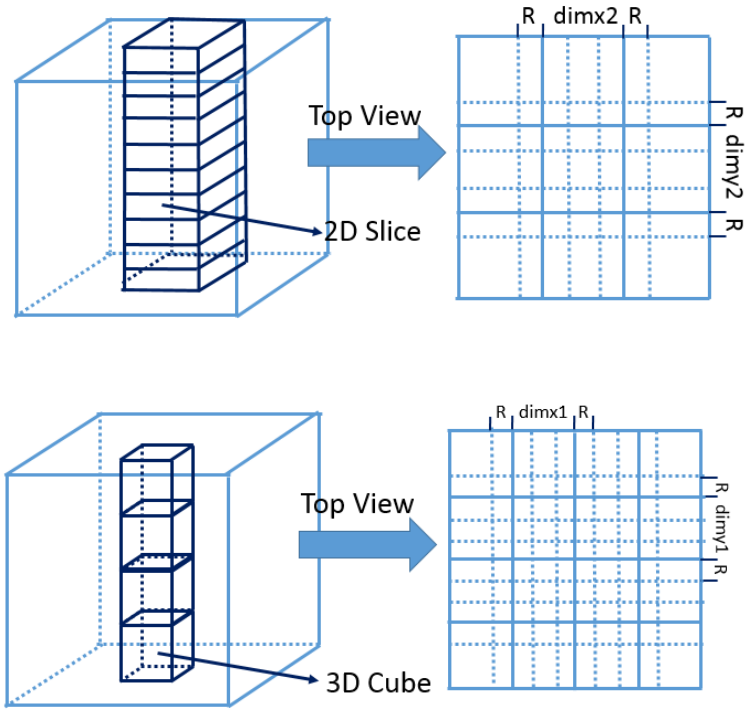


Optimizations



Optimizations

GPU Opt	Pinned Memory
	SMEM/L1
	AoS -> SoA
	Register Adjustment
	Kernel Splitting
	Other Methods
	Customizable Data Cache
	Inner-thread Rescheduling
CPU Opt	OpenMP
	SIMD Vectorization
	Cache blocking

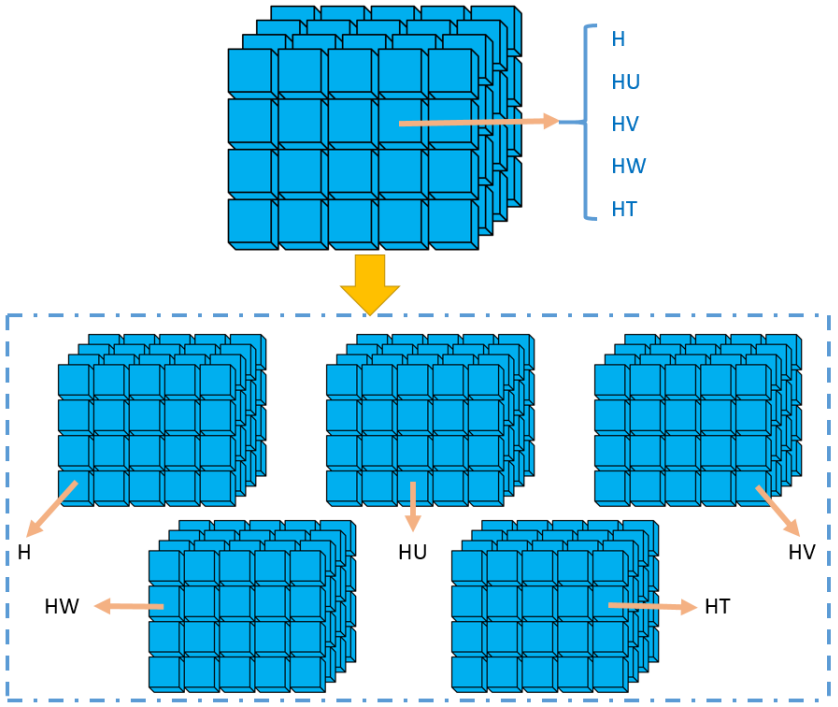


Compiler option

-Xptxas dlcm= ca

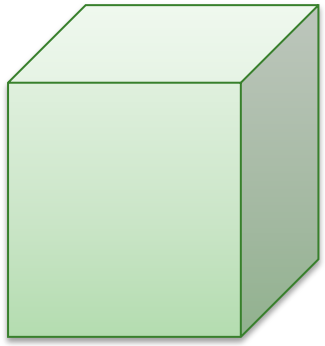
Optimizations

GPU Opt	Pinned Memory
	SMEM/L1
	AoS -> SoA
	Register Adjustment
	Kernel Splitting
	Other Methods
	Customizable Data Cache
	Inner-thread Rescheduling
CPU Opt	OpenMP
	SIMD Vectorization
	Cache blocking



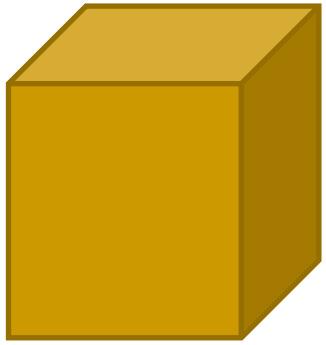
Optimizations

GPU Opt	Pinned Memory
	SMEM/L1
	AoS -> SoA
	Register Adjustment
	Kernel Splitting
	Other Methods
	Customizable Data Cache
	Inner-thread Rescheduling
CPU Opt	OpenMP
	SIMD Vectorization
	Cache blocking



Streaming Multi-Processor
64K Register
2048 threads

Rt: Register per thread
Occupancy = (64*1024) / (2048*Rt)

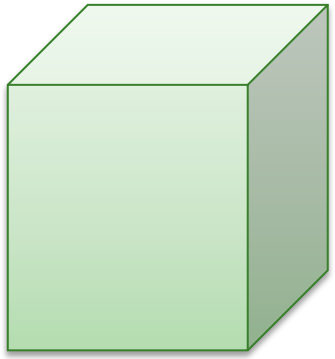


256 registers per threads
Rt = 256
1 Block per SM

Occupancy = (64*1024) / (2048*Rt) = 12.5%

Optimizations

GPU Opt	Pinned Memory
	SMEM/L1
	AoS -> SoA
	Register Adjustment
	Kernel Splitting
	Other Methods
	Customizable Data Cache
	Inner-thread Rescheduling
CPU Opt	OpenMP
	SIMD Vectorization
	Cache blocking



Streaming Multi-Processor
64K Register
2048 threads

Rt: Register per thread
Occupancy = (64*1024) / (2048*Rt)

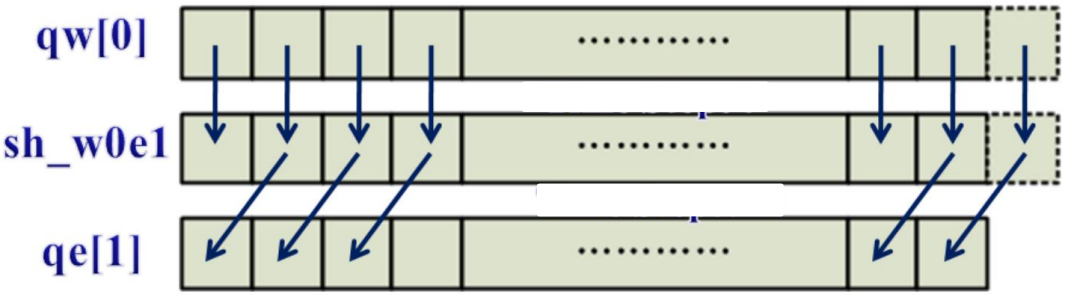
64 registers per threads
Rt = 64
4 Block per SM

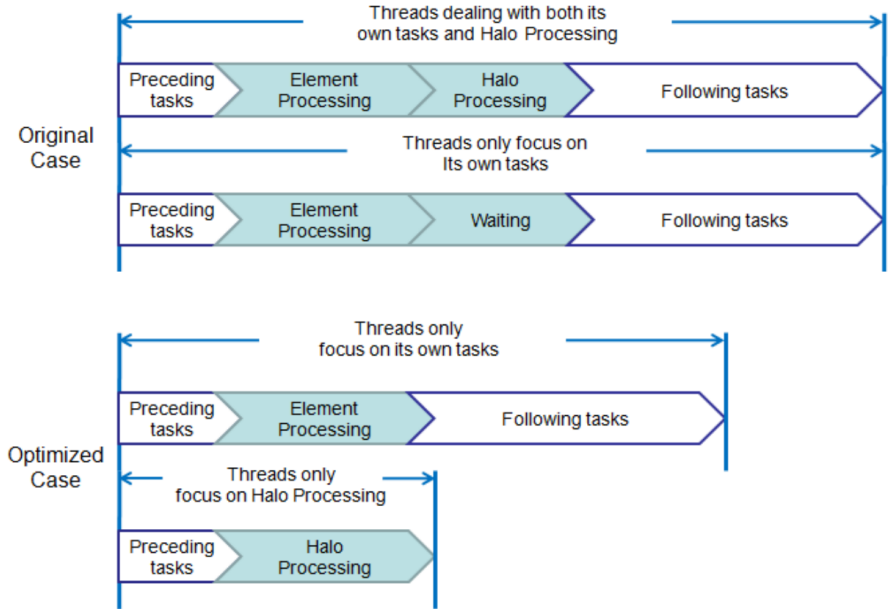
Occupancy = (64*1024) / (2048*Rt) = 50%

Compiler option
-maxrregcount = 64

Optimizations

GPU Opt	Pinned Memory
	SMEM/L1
	AoS -> SoA
	Register Adjustment
	Kernel Splitting
	Other Methods
	Customizable Data Cache
	Inner-thread Rescheduling
CPU Opt	OpenMP
	SIMD Vectorization
	Cache blocking





Optimizations

Algorithm 8 Demonstration of Euler State Reconstruction Step

X-axis original code: ($qe[0], qe[1], qw[0]$ and $qw[1]$ are intermediate variables in the step of State Reconstruction)

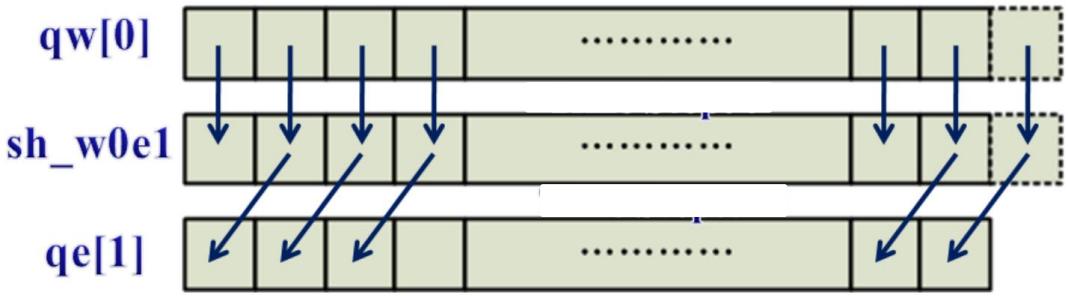
1: $qw[0]$ = (24*x[k,j,i] + 8*x[k,j,i-1] - 4*x[k,j,i+1] - (x[k,j+1,i] + x[k,j-1,i] + x[k+1,j,i] + x[k-1,j,i])) / 24;
2: $qe[0]$ = **(24*x[k,j,i] + 8*x[k,j,i+1] - 4*x[k,j,i-1] - (x[k,j+1,i] + x[k,j-1,i] + x[k+1,j,i] + x[k-1,j,i])) / 24**;
3: $qe[1]$ = (24*x[k,j,i+1] + 8*x[k,j,i] - 4*x[k,j,i+2] - (x[k,j+1,i+1] + x[k,j-1,i+1] + x[k+1,j,i+1] + x[k-1,j,i+1])) / 24;
4: $qw[1]$ = **(24*x[k,j,i-1] + 8*x[k,j,i] - 4*x[k,j,i-2] - (x[k,j+1,i-1] + x[k,j-1,i-1] + x[k+1,j,i-1] + x[k-1,j,i-1])) / 24**;

Optimizations

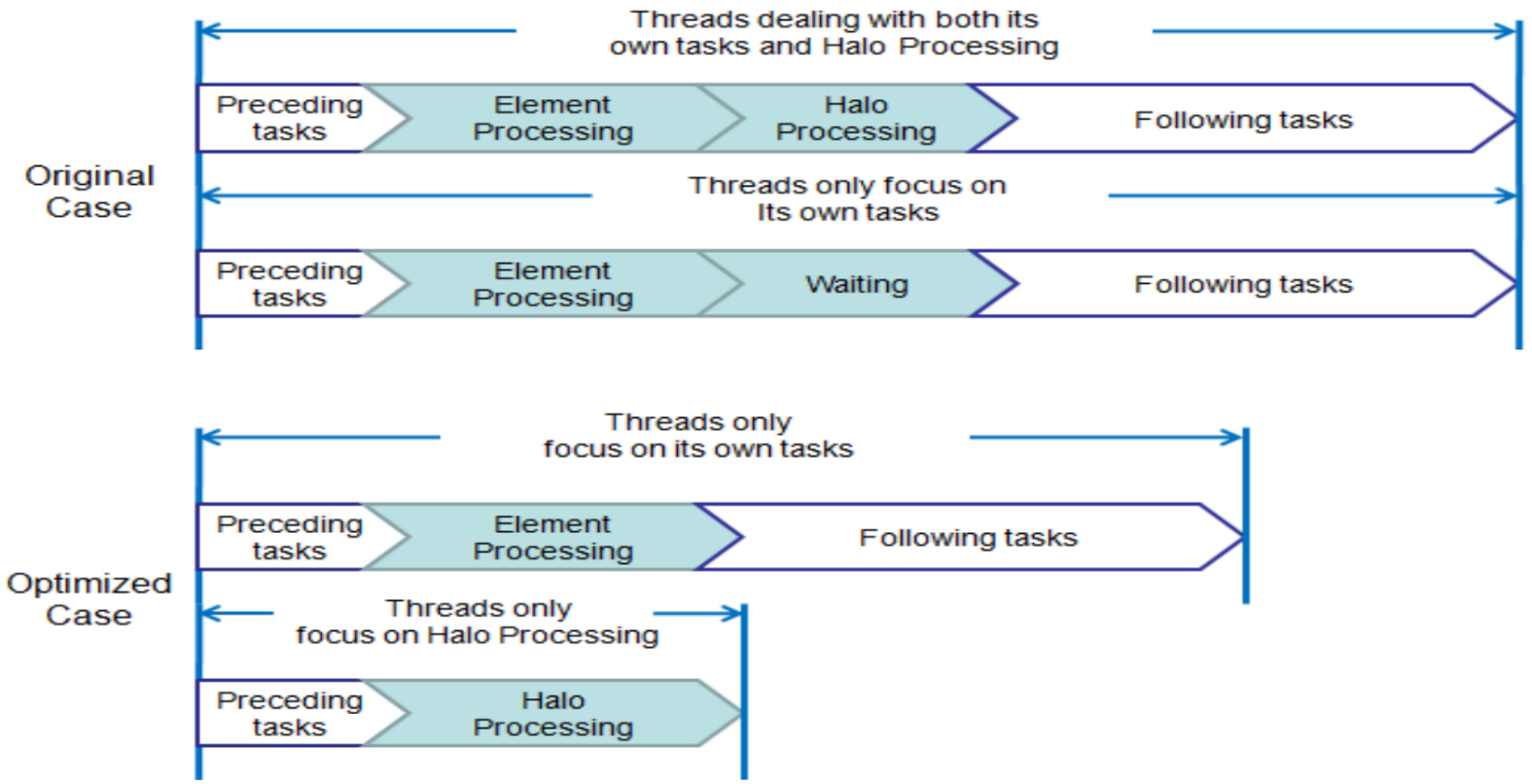
Algorithm 8 Demonstration of Euler State Reconstruction Step

X-axis original code: ($qe[0], qe[1], qw[0]$ and $qw[1]$ are intermediate variables in the step of State Reconstruction)

1: $qw[0]$ = (24*x[k,j,i] + 8*x[k,j,i-1] - 4*x[k,j,i+1] - (x[k,j+1,i] + x[k,j-1,i] + x[k+1,j,i] + x[k-1,j,i])) / 24;
2: $qe[0]$ = **(24*x[k,j,i] + 8*x[k,j,i+1] - 4*x[k,j,i-1] - (x[k,j+1,i] + x[k,j-1,i] + x[k+1,j,i] + x[k-1,j,i])) / 24**;
3: $qe[1]$ = (24*x[k,j,i+1] + 8*x[k,j,i] - 4*x[k,j,i+2] - (x[k,j+1,i+1] + x[k,j-1,i+1] + x[k+1,j,i+1] + x[k-1,j,i+1])) / 24;
4: $qw[1]$ = **(24*x[k,j,i-1] + 8*x[k,j,i] - 4*x[k,j,i-2] - (x[k,j+1,i-1] + x[k,j-1,i-1] + x[k+1,j,i-1] + x[k-1,j,i-1])) / 24**;



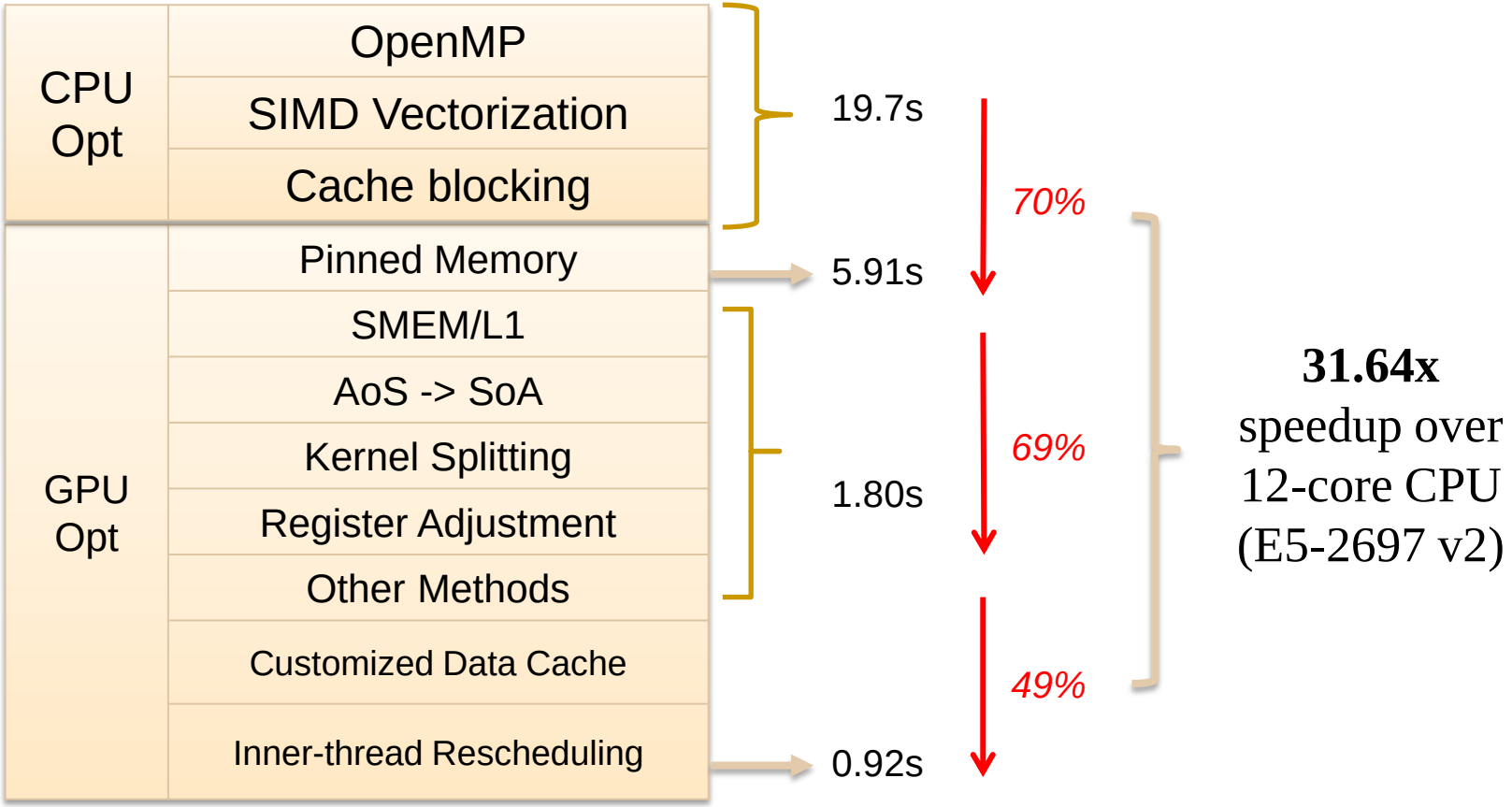
Optimizations



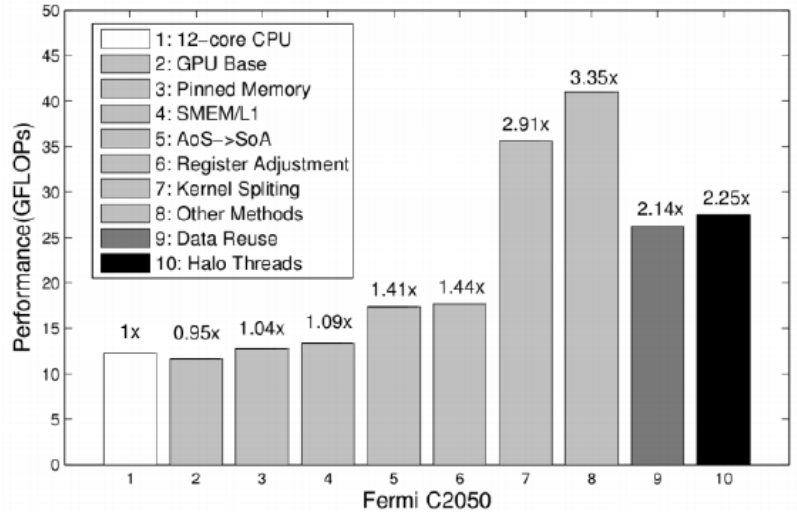
A Highly Scalable Framework for Atmospheric Modeling on Heterogeneous Supercomputers:

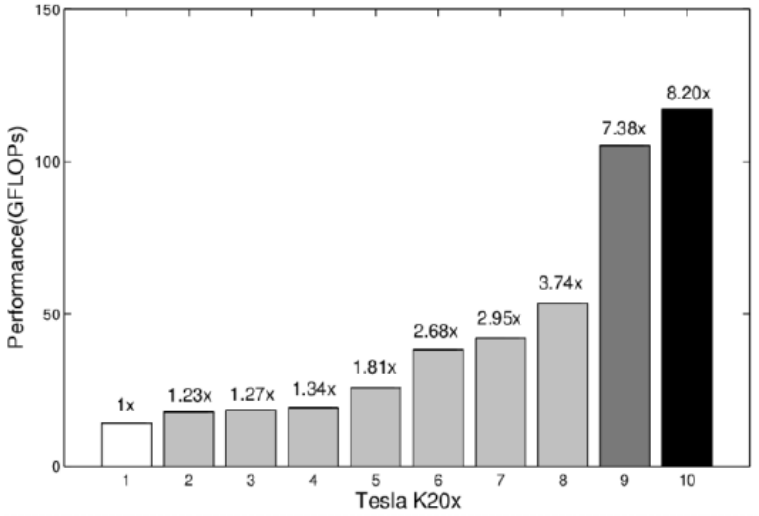
Results

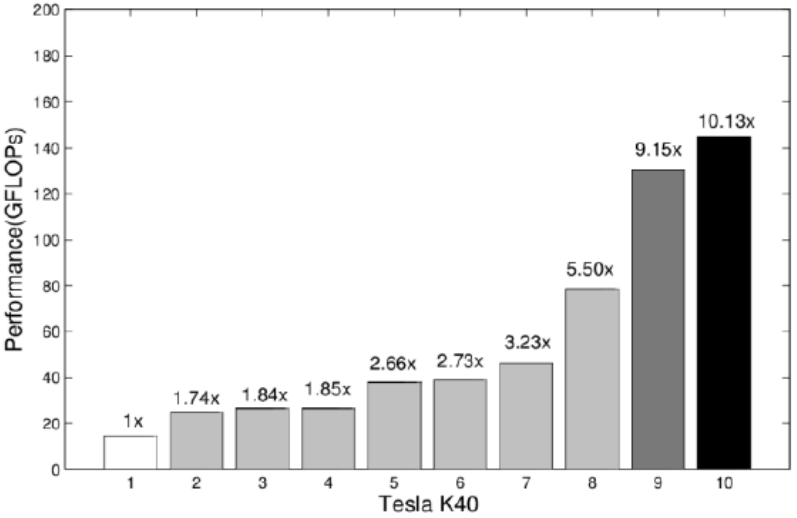
Experimental Result

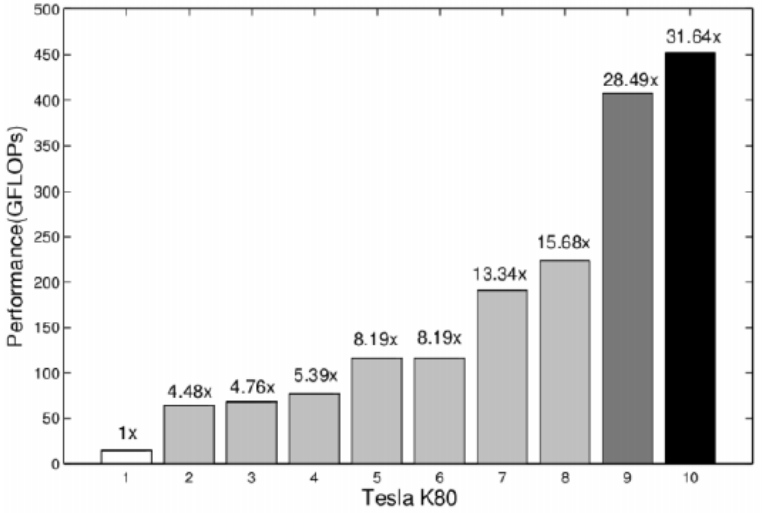


Experimental Result



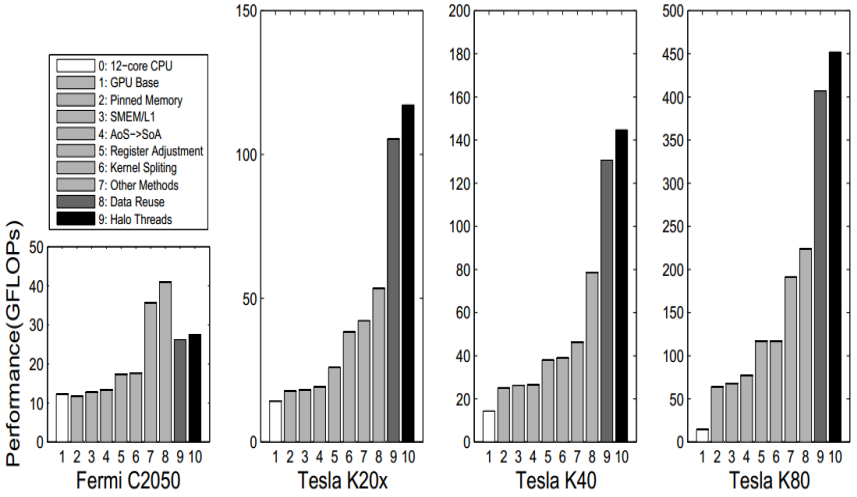


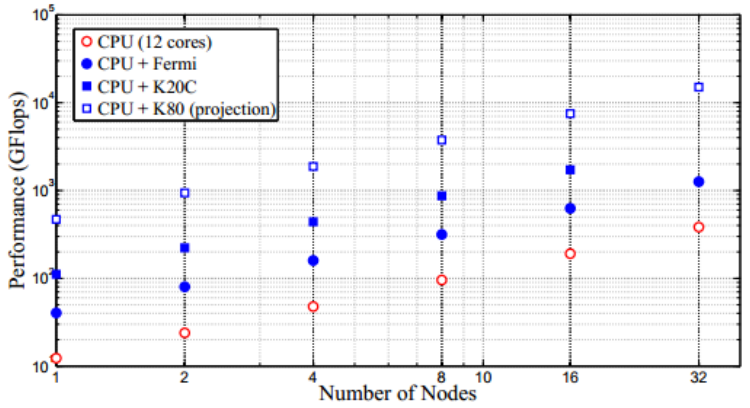




Experimental Result

- First-Round Optimizations
  - Five general-used optimizations
  - A 80GFlops performance result is achieved on a single Tesla K40
- Customized Optimizations
  - A customized cache mechanism & Inter-thread Rescheduling
  - A 146GFlops performance result is achieved on a single Tesla K40
- Experimental Results
  - 451GFLOPs on a single Tesla K80, which is 31.64x speedup over a 12-core CPU (E5-2697 v2)
  - 16.87% of peak based on Tesla K80
- Weak Scaling Result
  - 98.7% among 32 Node





Acknowledgement

国家自然科学
基金委员会
National Natural Science
Foundation of China

中国石油东方地球物理公司
BGP INC.,CHINA NATIONAL PETROLEUM CORPORATION

Thank You!

haohuan@tsinghua.edu.cn