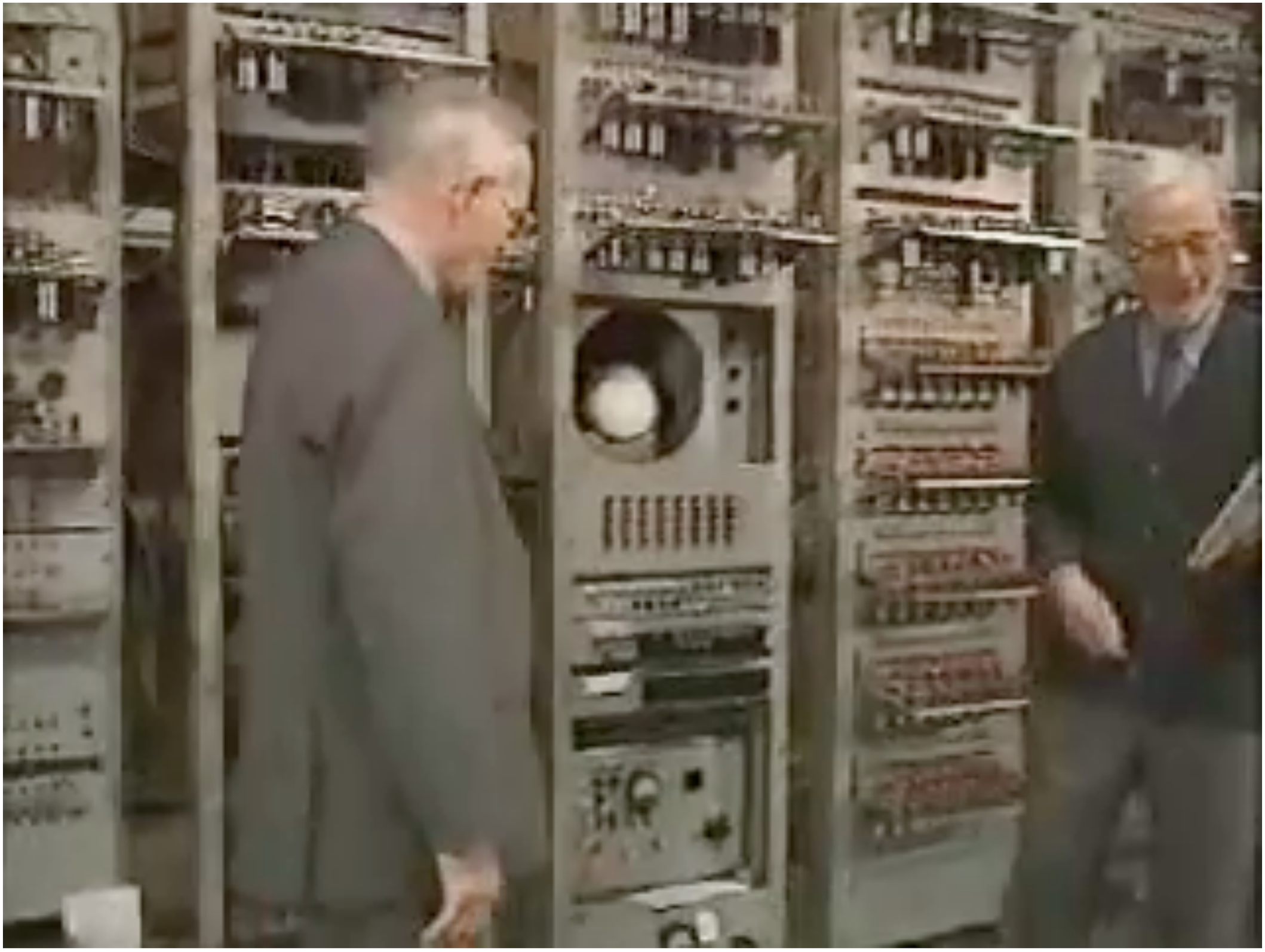
New England Numerical Analysis Day
University of Massachusetts, Amherst
Amherst, Apr. 21, 2012

The success of multipole methods: Are we there yet?

Lorena A Barba, Boston University

1948

The world's first computation by a stored-program computer: The Manchester Baby

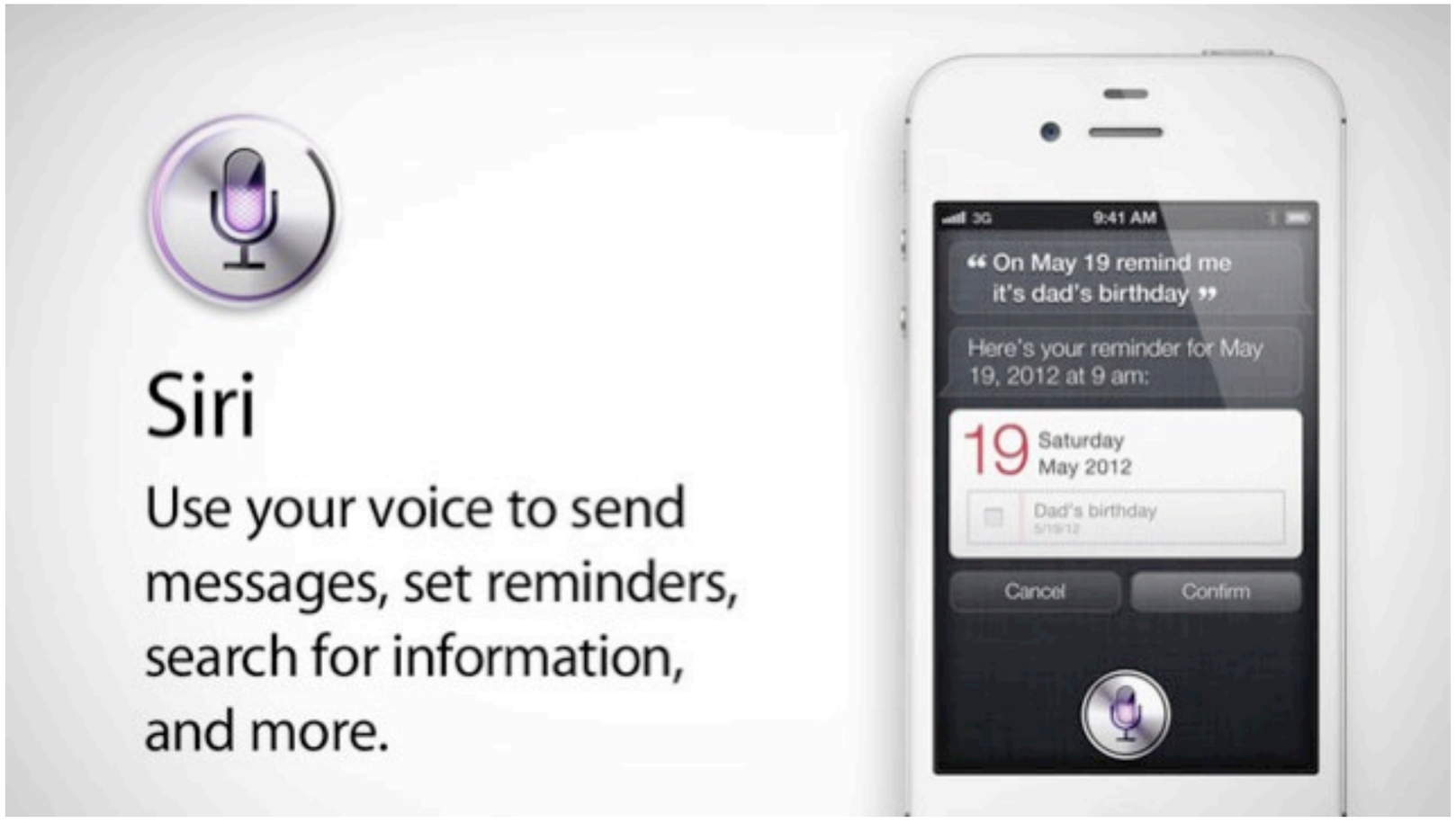


1998

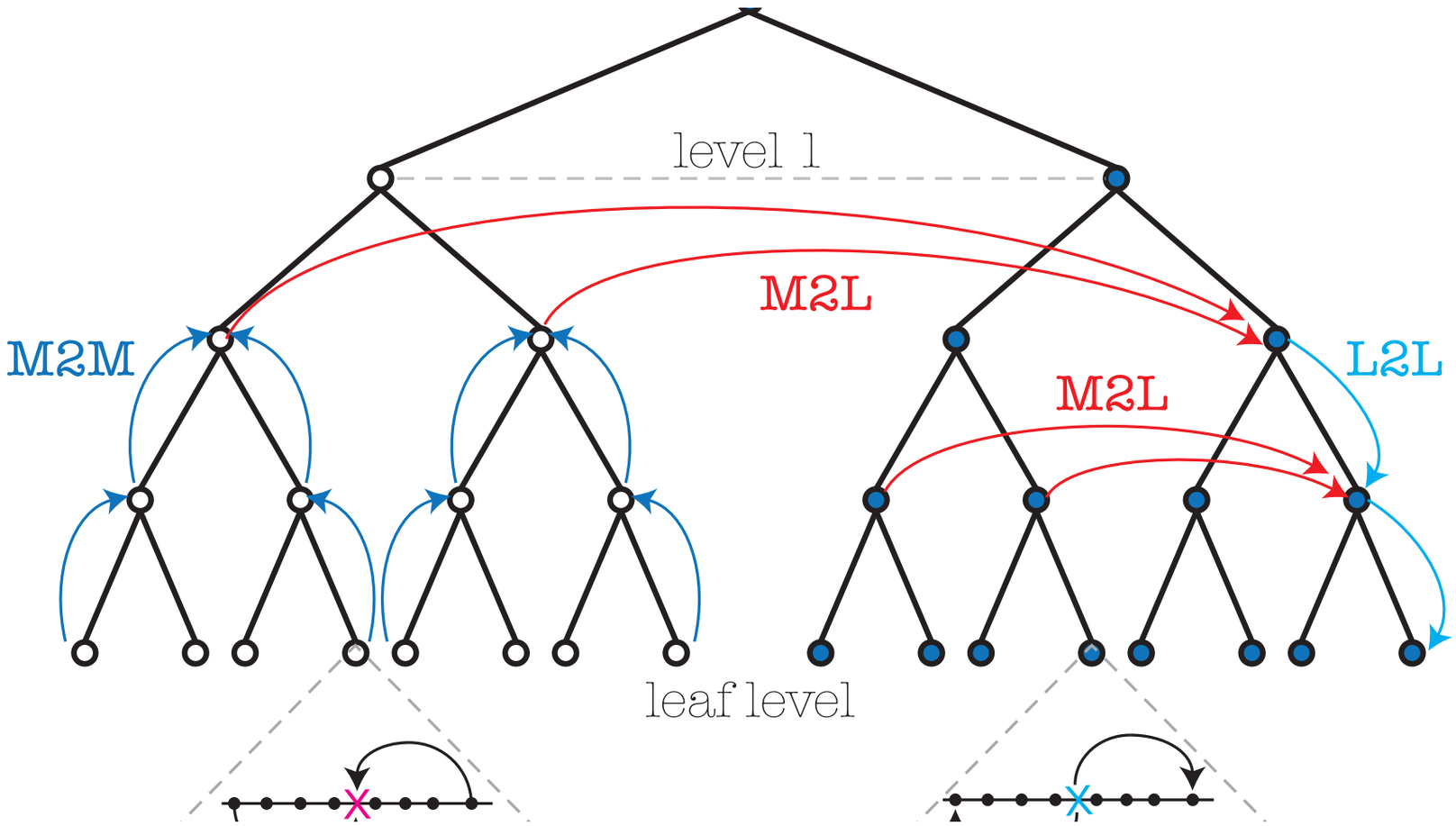
Lloyd Trefethen: Predictions for Scientific Computing 50 Years From Now

Professor Lloyd N Trefethen FRS
Head of Oxford's Numerical Analysis Group
President of SIAM (2011-2012)

#1. We may not be here



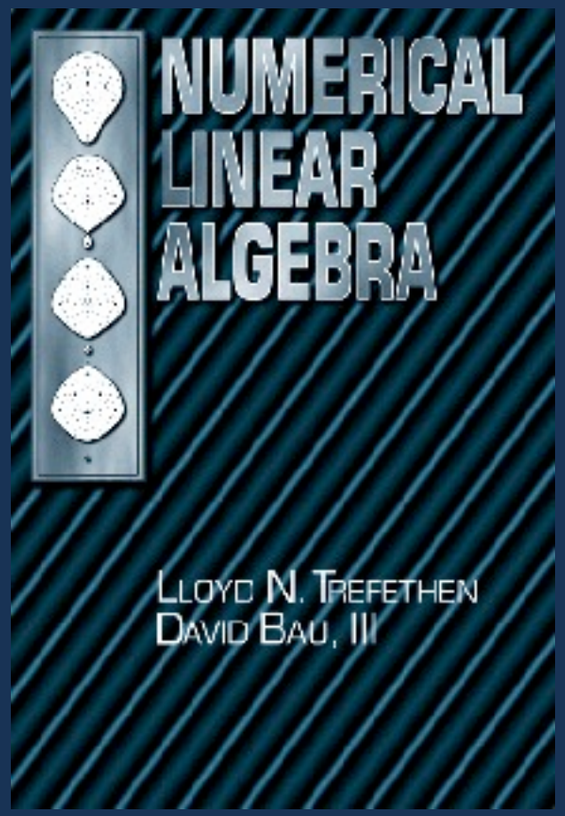
#2. We'll talk to computers more often than type to them, and they'll respond with pictures more often than numbers.



#7. Multipole methods and their descendants will be ubiquitous

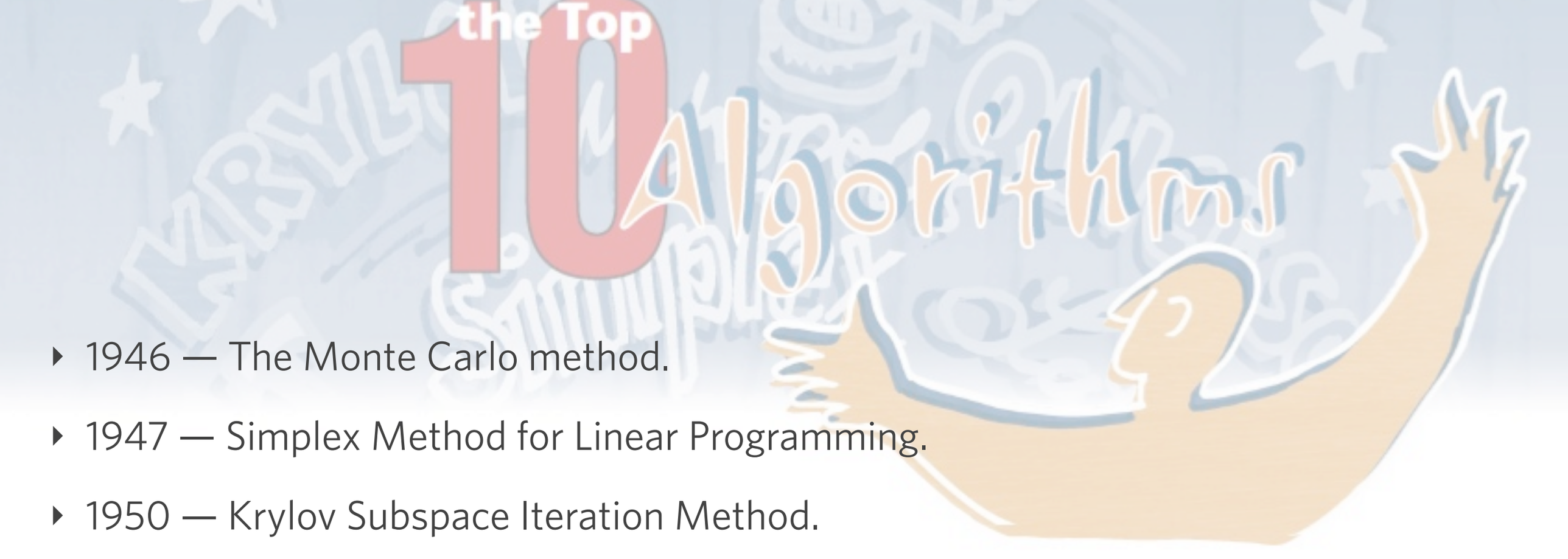
"... the fundamental law of computer science [is]: the faster the computer, the greater the importance of speed of algorithms"

Trefethen & Bau "Numerical Linear Algebra" SIAM



Fast multipole method

- *O(N) solution of N-body problem*
- *Top 10 Algorithm of the 20th century*

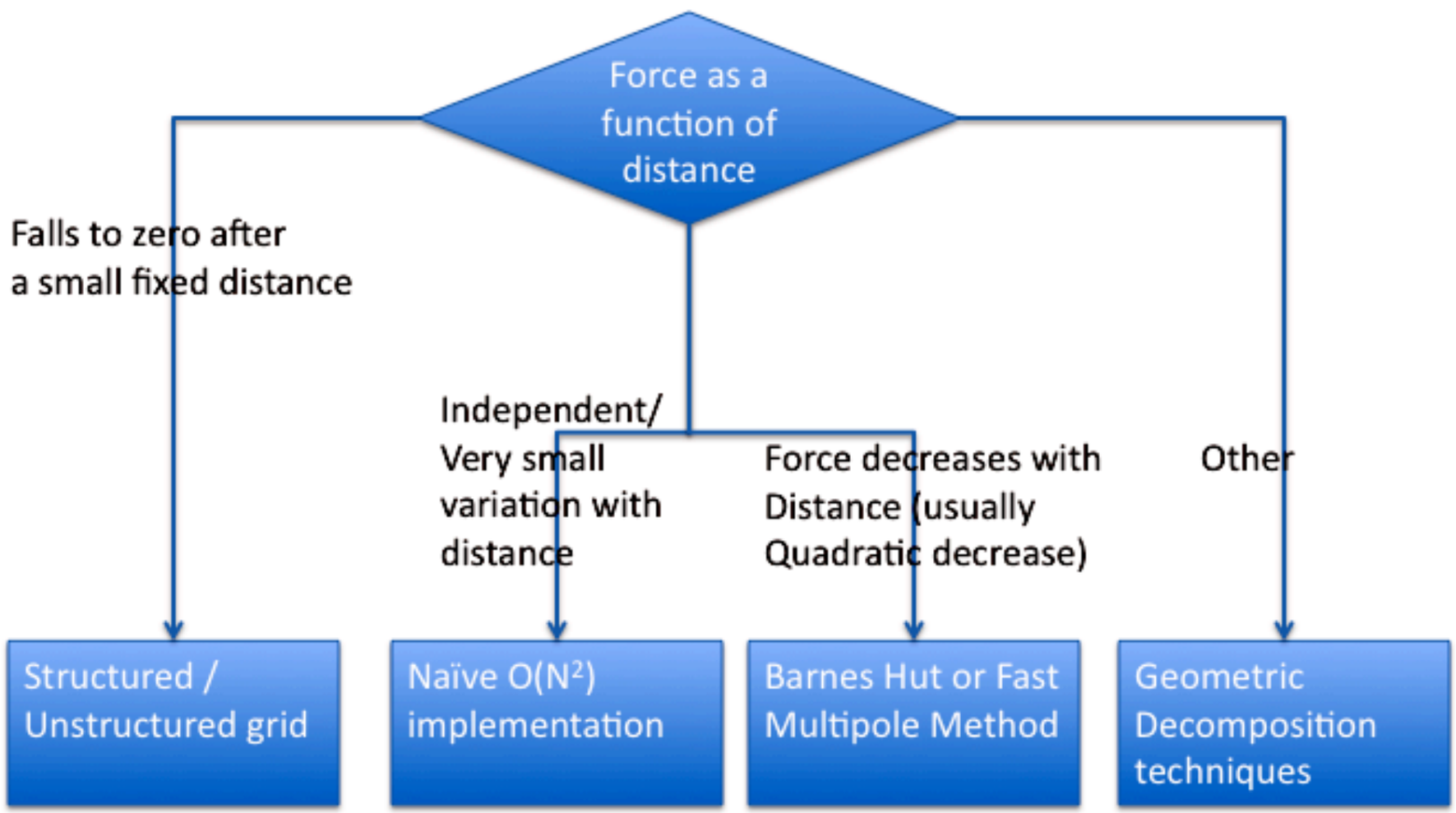


- 1946 — The Monte Carlo method.
- 1947 — Simplex Method for Linear Programming.
- 1950 — Krylov Subspace Iteration Method.
- 1951 — The Decompositional Approach to Matrix Computations.
- 1957 — The Fortran Compiler.
- 1959 — QR Algorithm for Computing Eigenvalues.
- 1962 — Quicksort Algorithms for Sorting.
- 1965 — Fast Fourier Transform.
- 1977 — Integer Relation Detection.
- 1987 — **Fast Multipole Method**

Dongarra& Sullivan, IEEE Comput. Sci. Eng., Vol. 2(1):22–23 (2000)

N-body

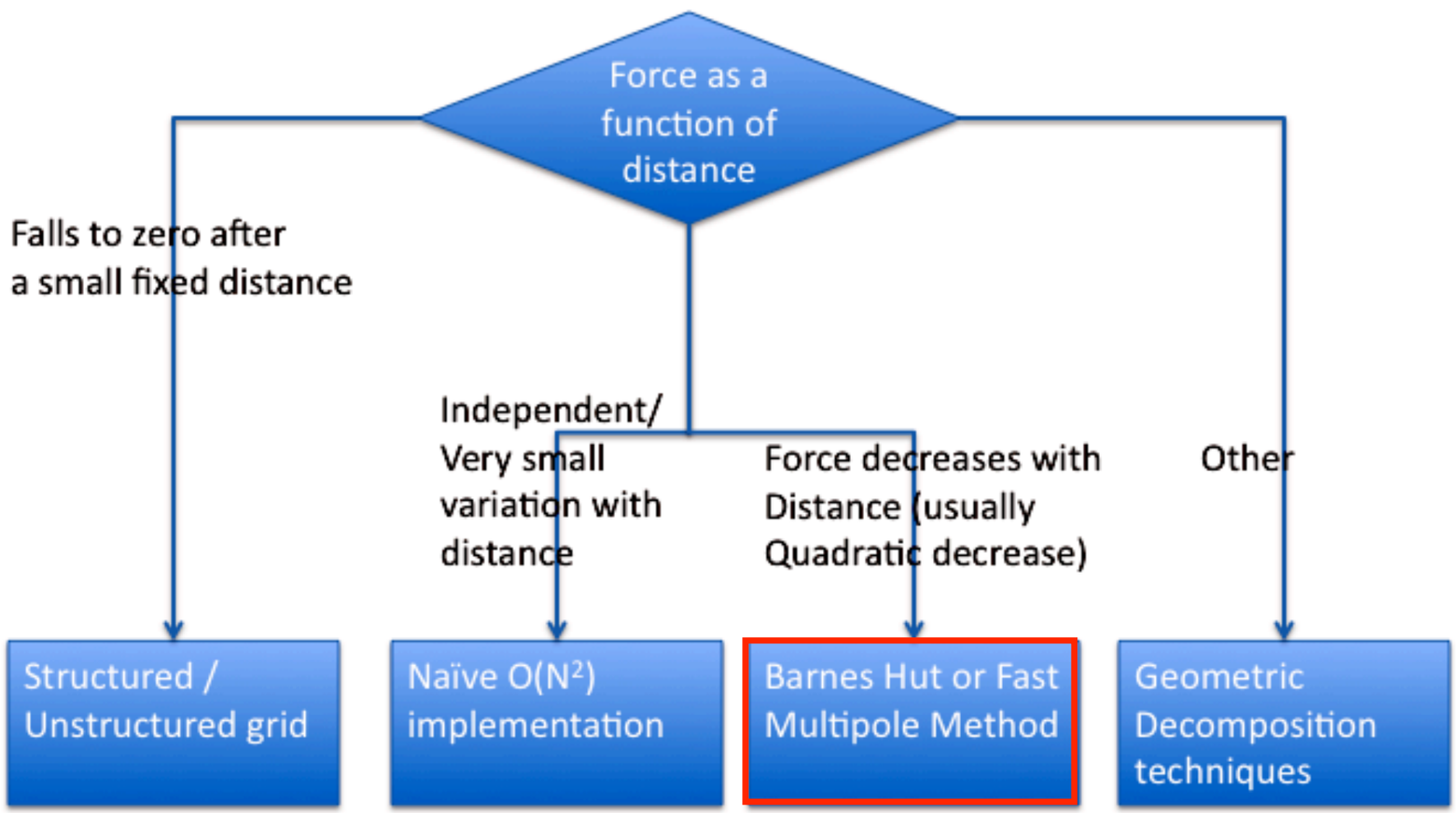
- **Problem:**
  "updates to a system where each element of the system rigorously depends on the state of every other element of the system."



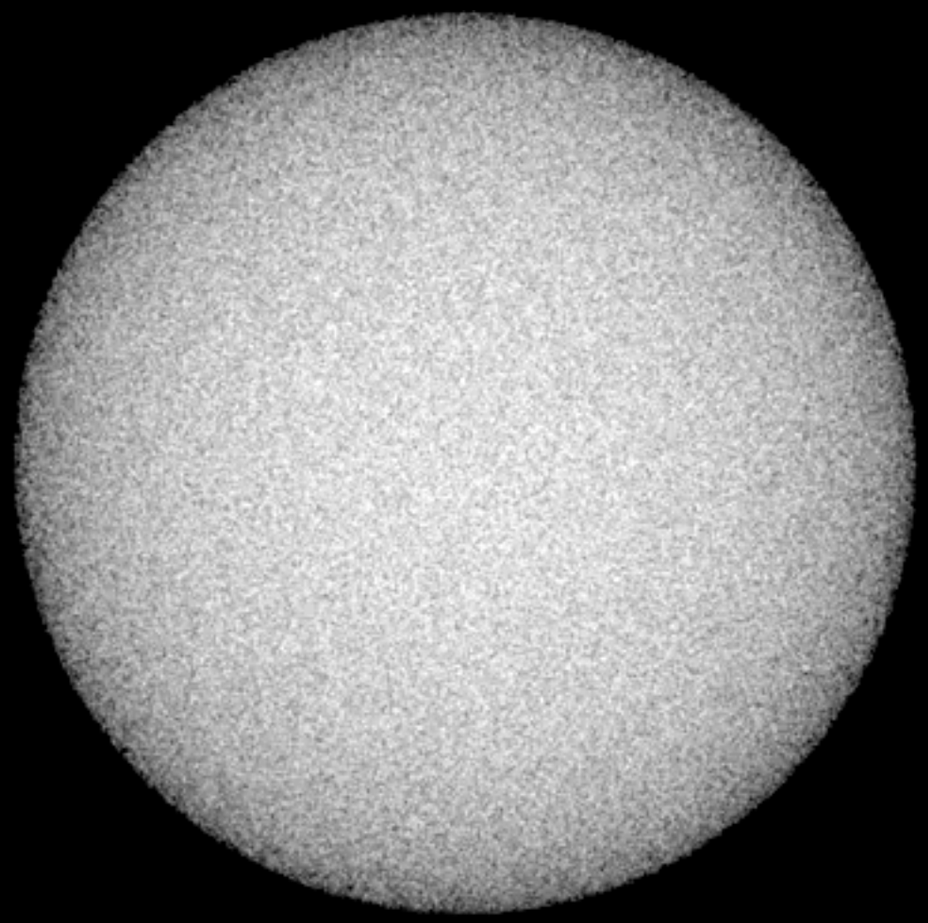
http://parlab.eecs.berkeley.edu/wiki/patterns/n-body_methods

N-body

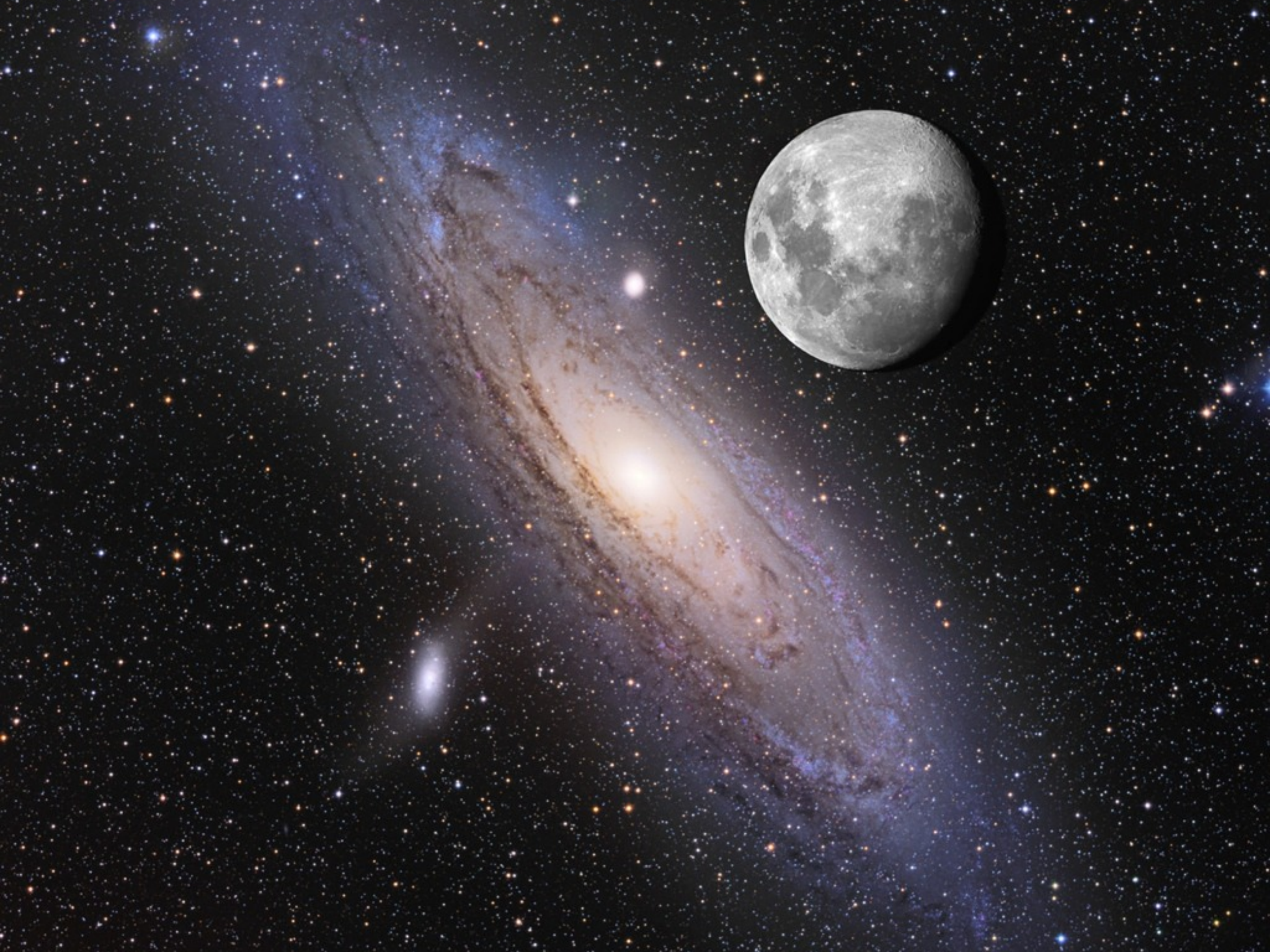
- **Problem:**
  "updates to a system where each element of the system rigorously depends on the state of every other element of the system."



http://parlab.eecs.berkeley.edu/wiki/patterns/n-body_methods



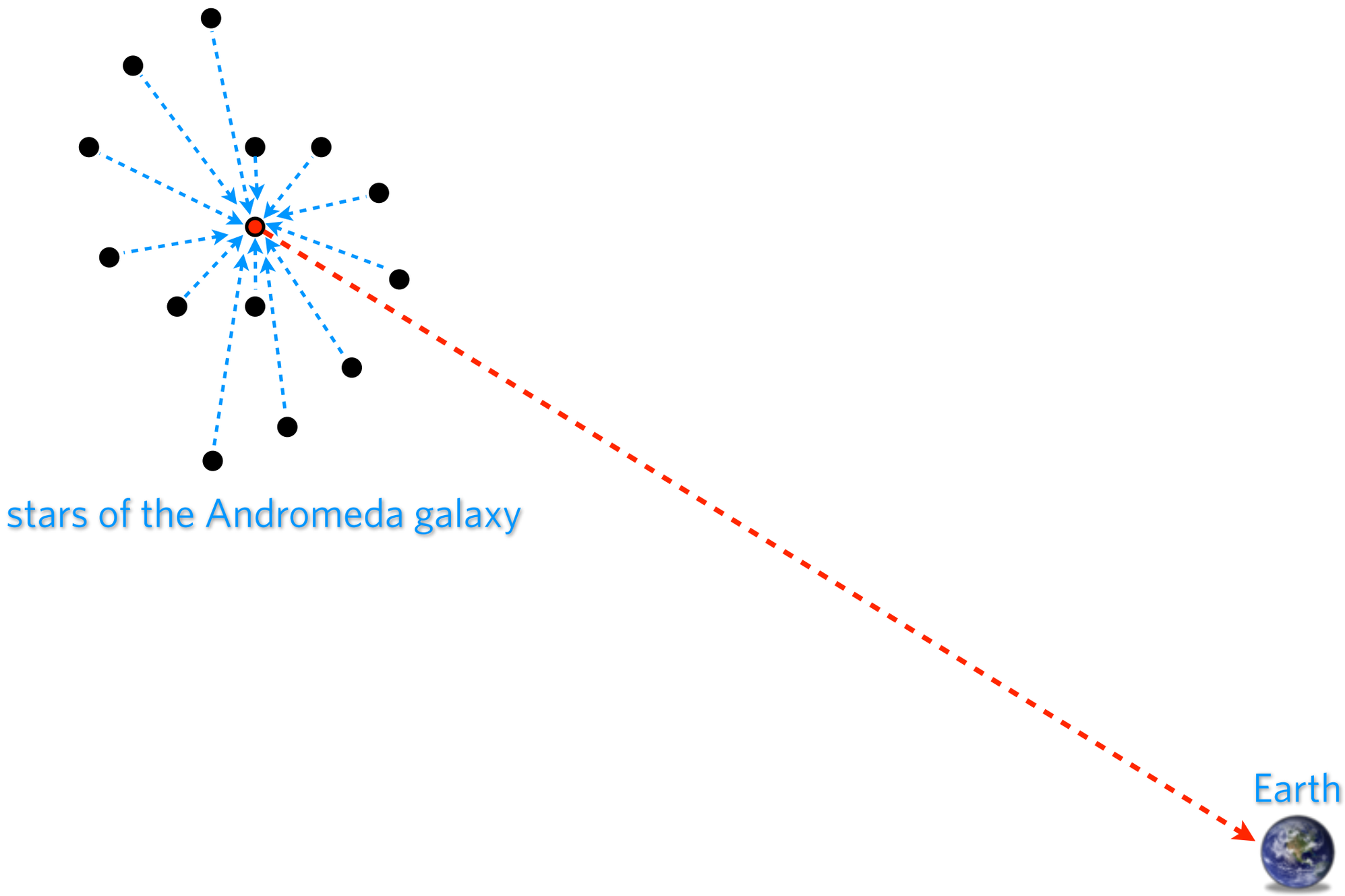
Credit: **Mark Stock**



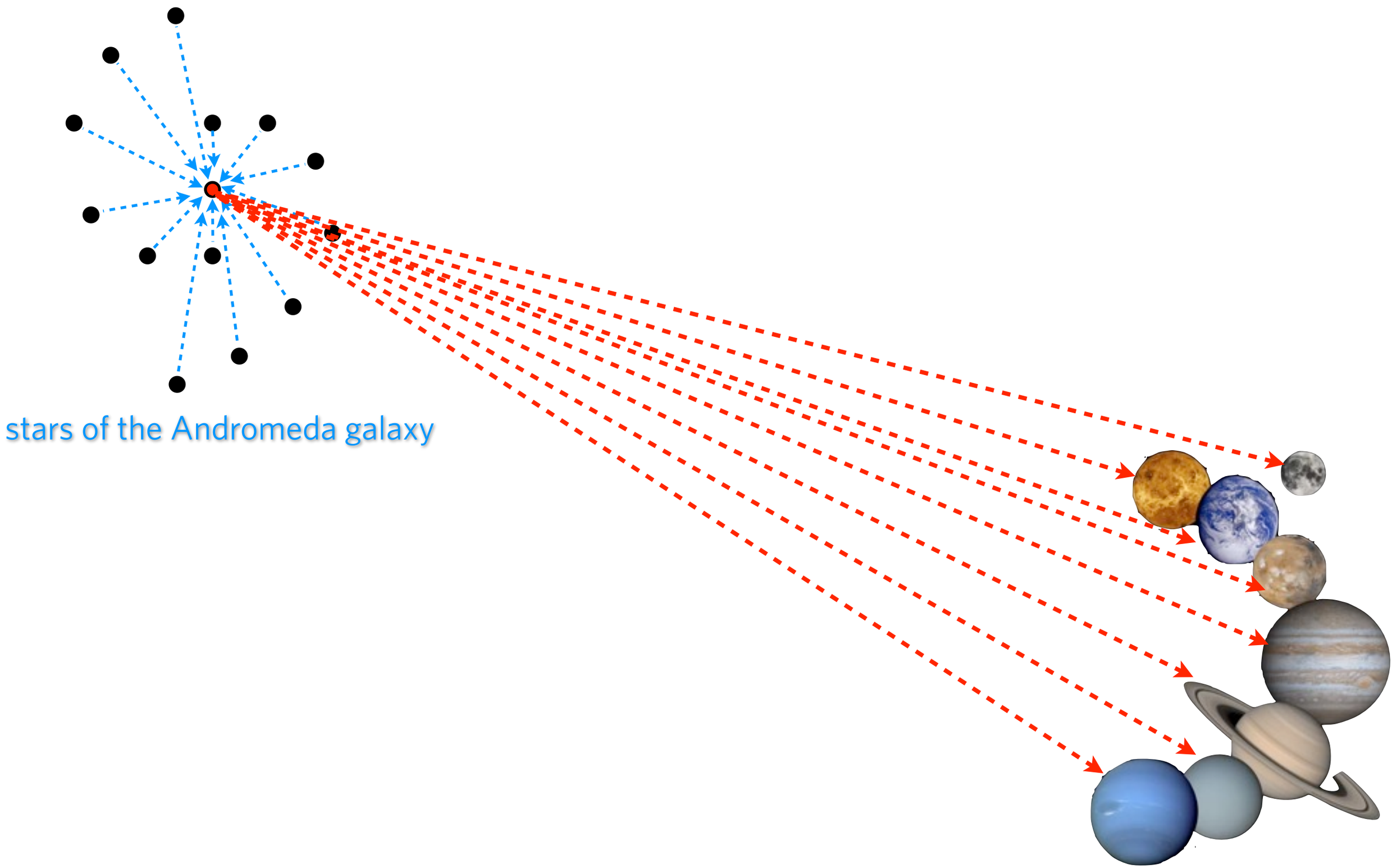

M31 Andromeda galaxy
stars: 10^{12}


CASSIOPEIA
PEGASUS
CAMELOPARDALIS
Andromeda Galaxy
ANDROMEDA
ALGOL
PERSEUS
TRIANGULUM
PISCES
Capella
ARIES
East

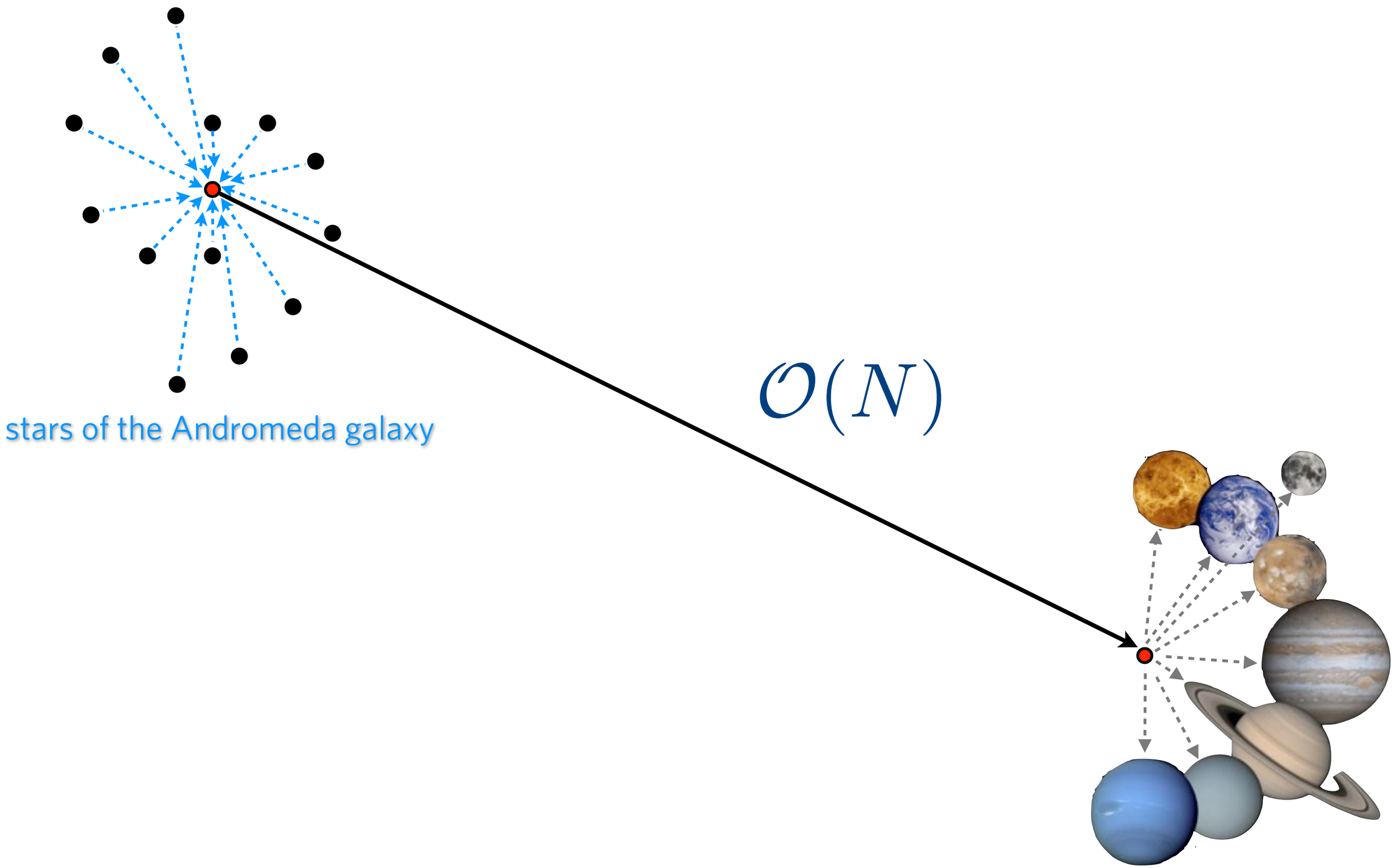
Fast N-body method



Fast N-body method



Fast N-body method



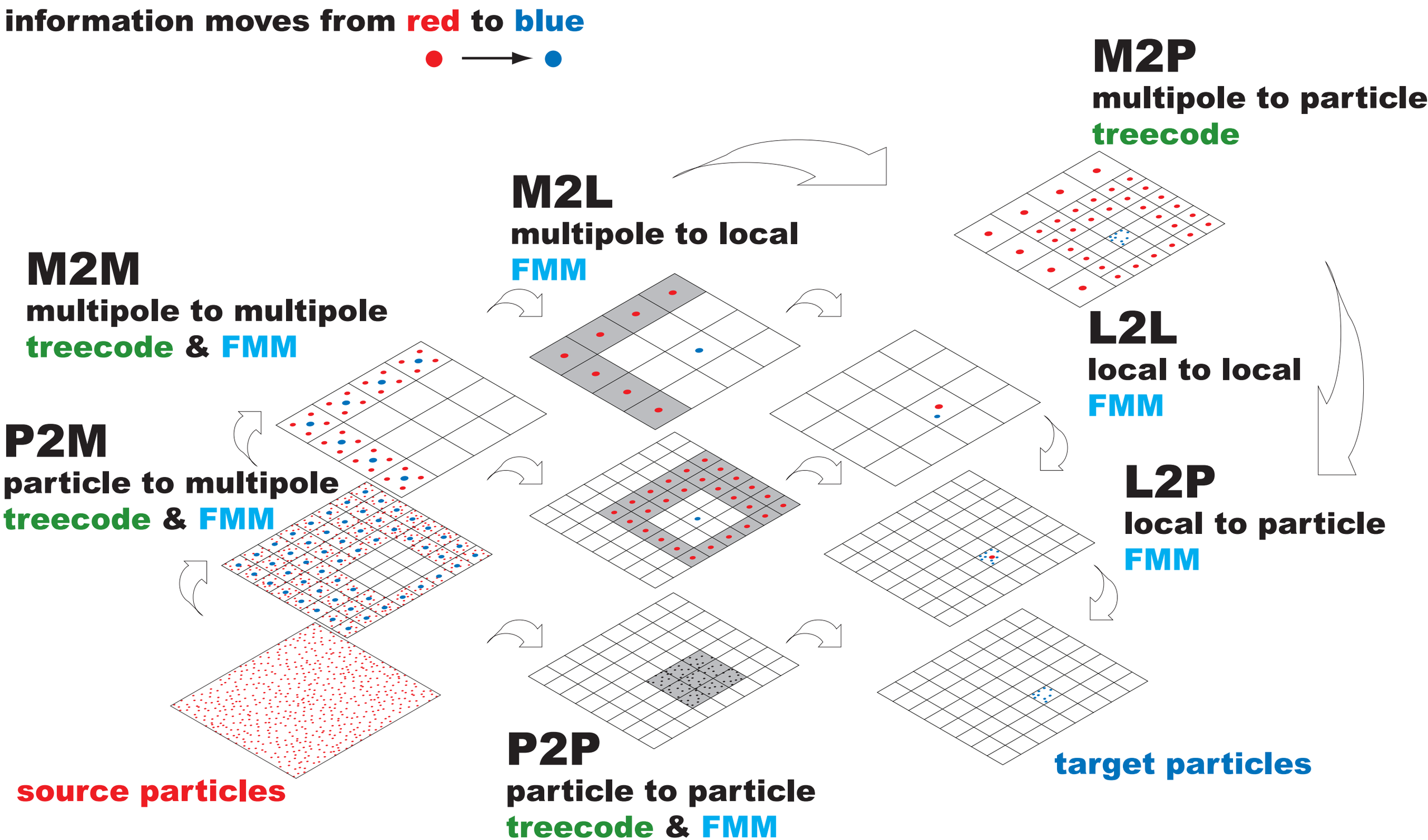
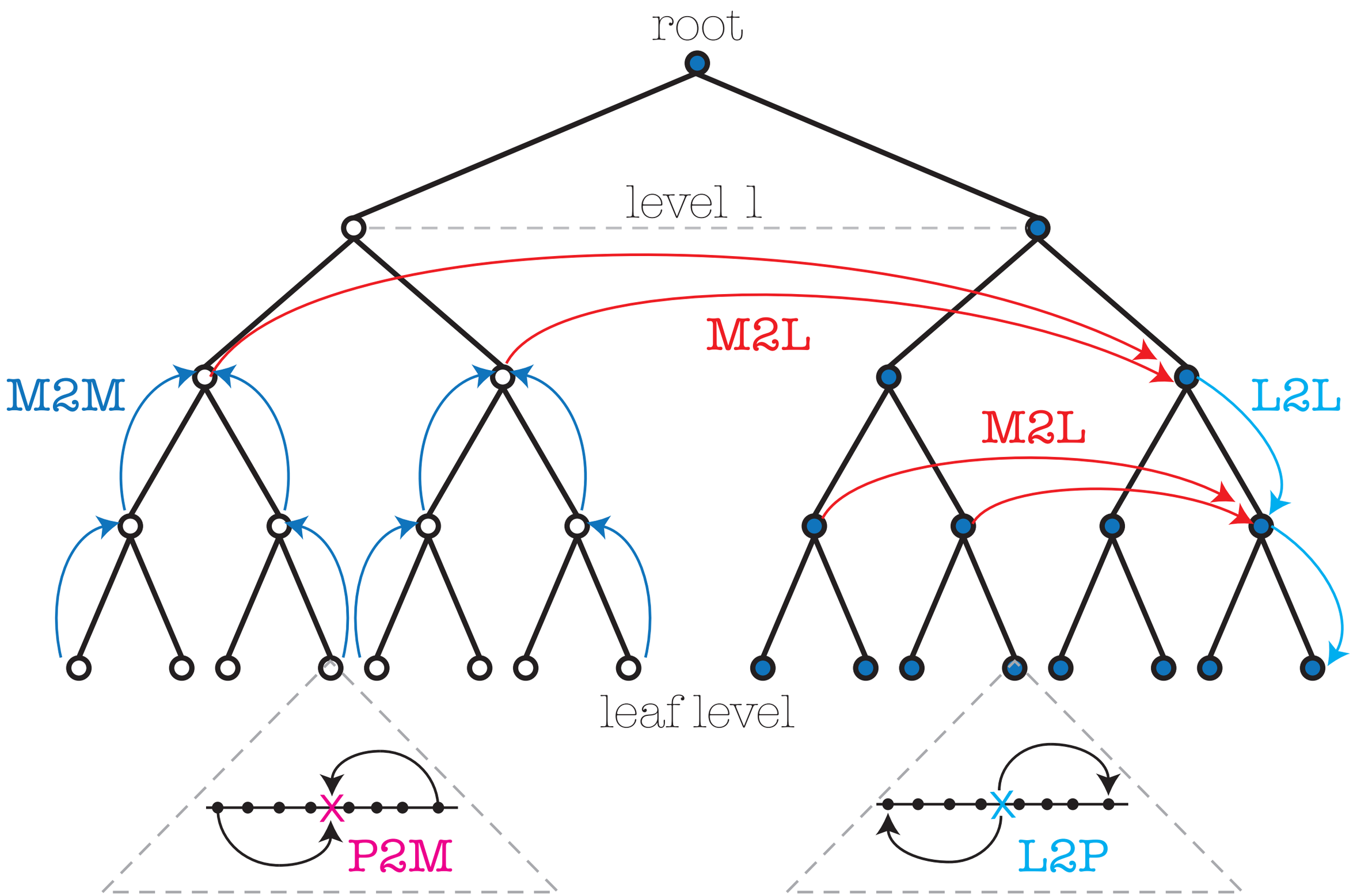


Image: "Treecode and fast multipole method for N-body simulation with CUDA", Rio Yokota, Lorena A Barba, Ch. 9 in *GPU Computing Gems Emerald Edition*, Wen-mei Hwu, ed.; Morgan Kaufmann/Elsevier (2011) pp. 113–132.

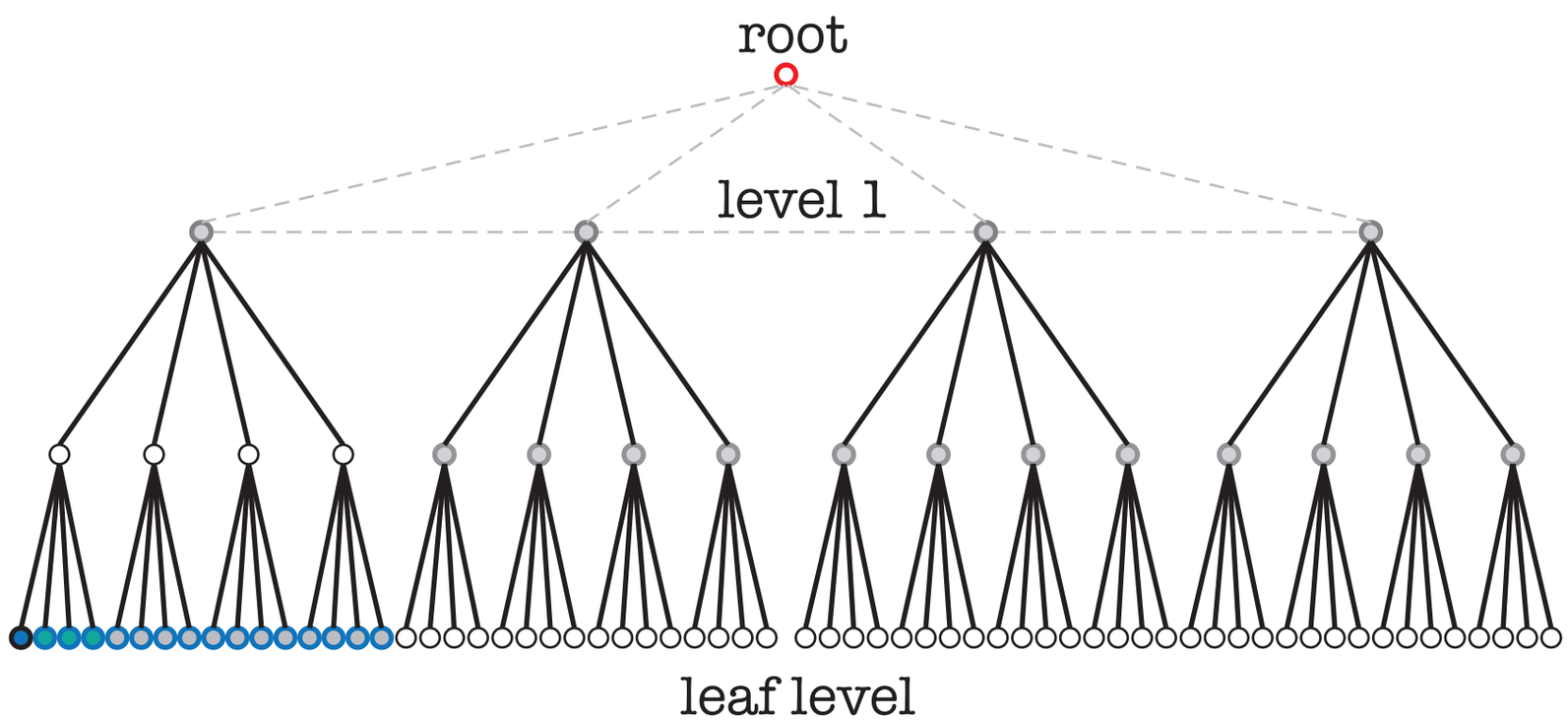


"A tuned and scalable fast multipole method as a preeminent algorithm for exascale systems", Rio Yokota, L A Barba. *Int. J. High-perf. Comput.*, Jan. 2012 — This figure shared under CC-BY on figshare.com

Treecode & Fast multipole method

- reduces operation count from **O(N²) to O(N log N) or O(N)**

$$f(y) = \sum_{i=1}^{N} c_i \mathbf{K}(y - x_i) \qquad y \in [1 \ldots N]$$



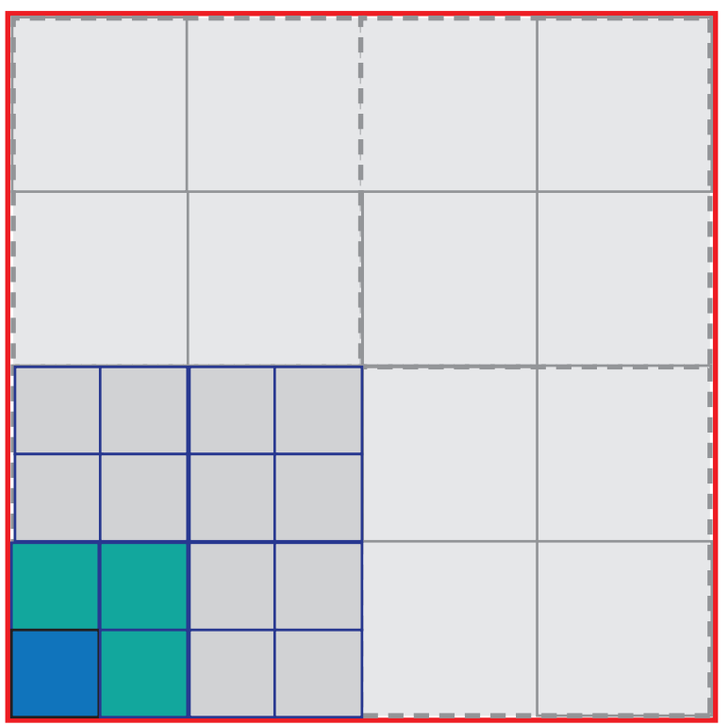
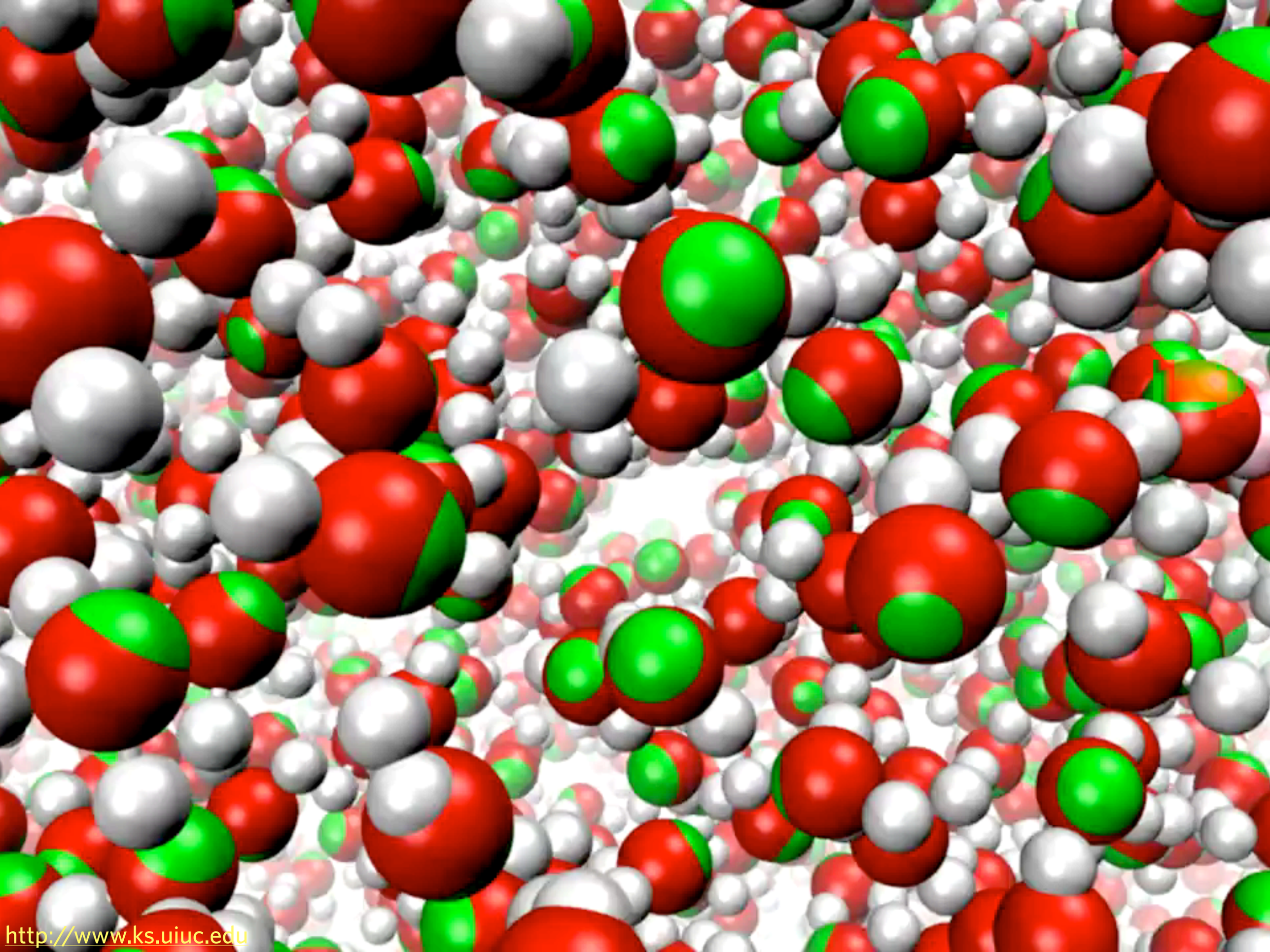


Image: "A tuned and scalable fast multipole method as a preeminent algorithm for exascale systems", Rio Yokota, L A Barba. *Int. J. High-perf. Comput*. Accepted (2011) — To appear; preprint arXiv:1106.2176


http://www.ks.uiuc.edu

Diversity of N-body problems

atoms/ions in electrostatic or van der Waals forces

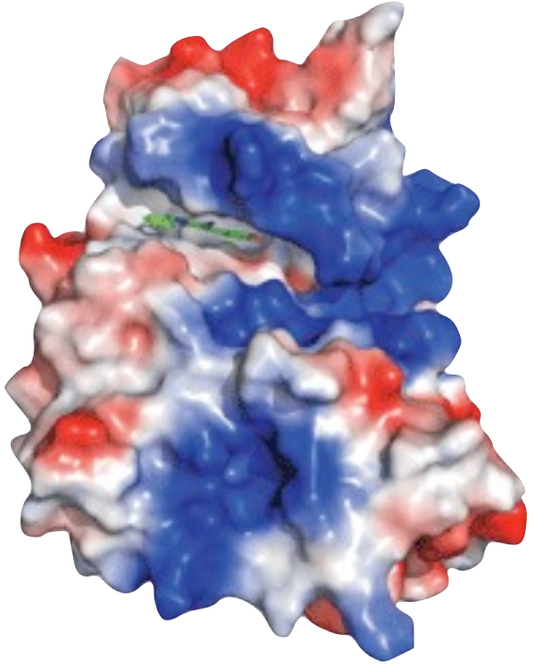
- integral formulation of elliptic PDE

$$\nabla^2 u = f \quad \rightarrow \quad u = \int_\Omega G f \, d\Omega$$

Numerical integration

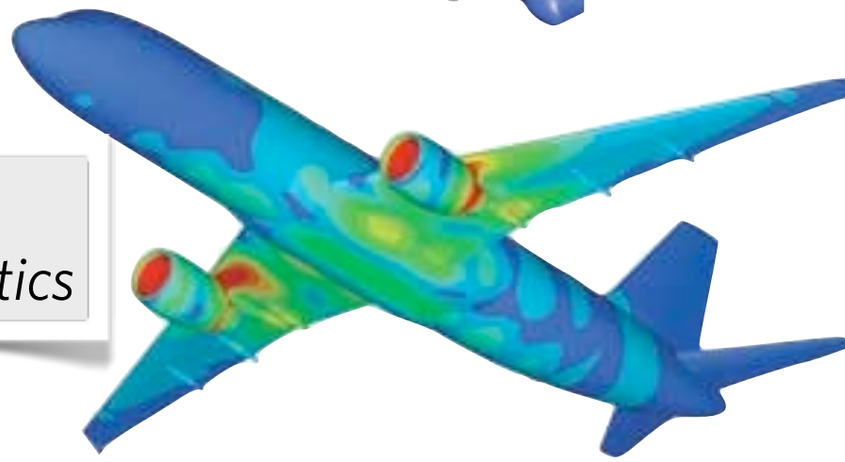
Applications of the FMM

$$\nabla^2 u = f \quad \rightarrow \quad u = \int_\Omega G f \, d\Omega$$



- Poisson $\nabla^2 u = -f$

Astrophysics
Electrostatics
Fluid mechanics



- Helmholtz $\nabla^2 u + k^2 u = -f$

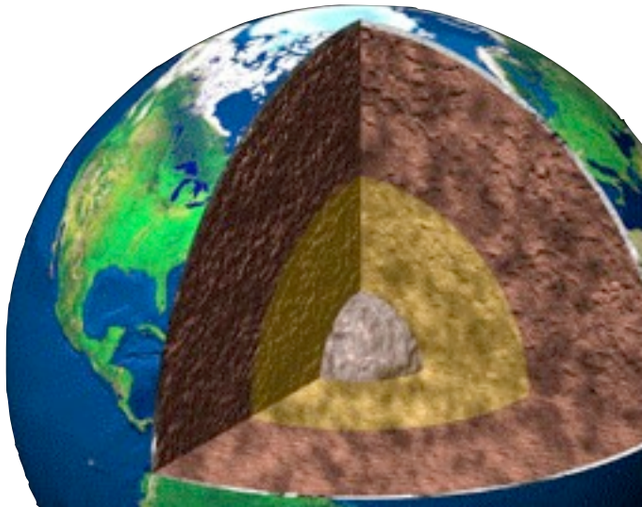
Acoustics
Electromagnetics

- Poisson-Boltzmann

$$\nabla \cdot (\epsilon \nabla u) + k^2 u = -f$$

Geophysics
Biophysics

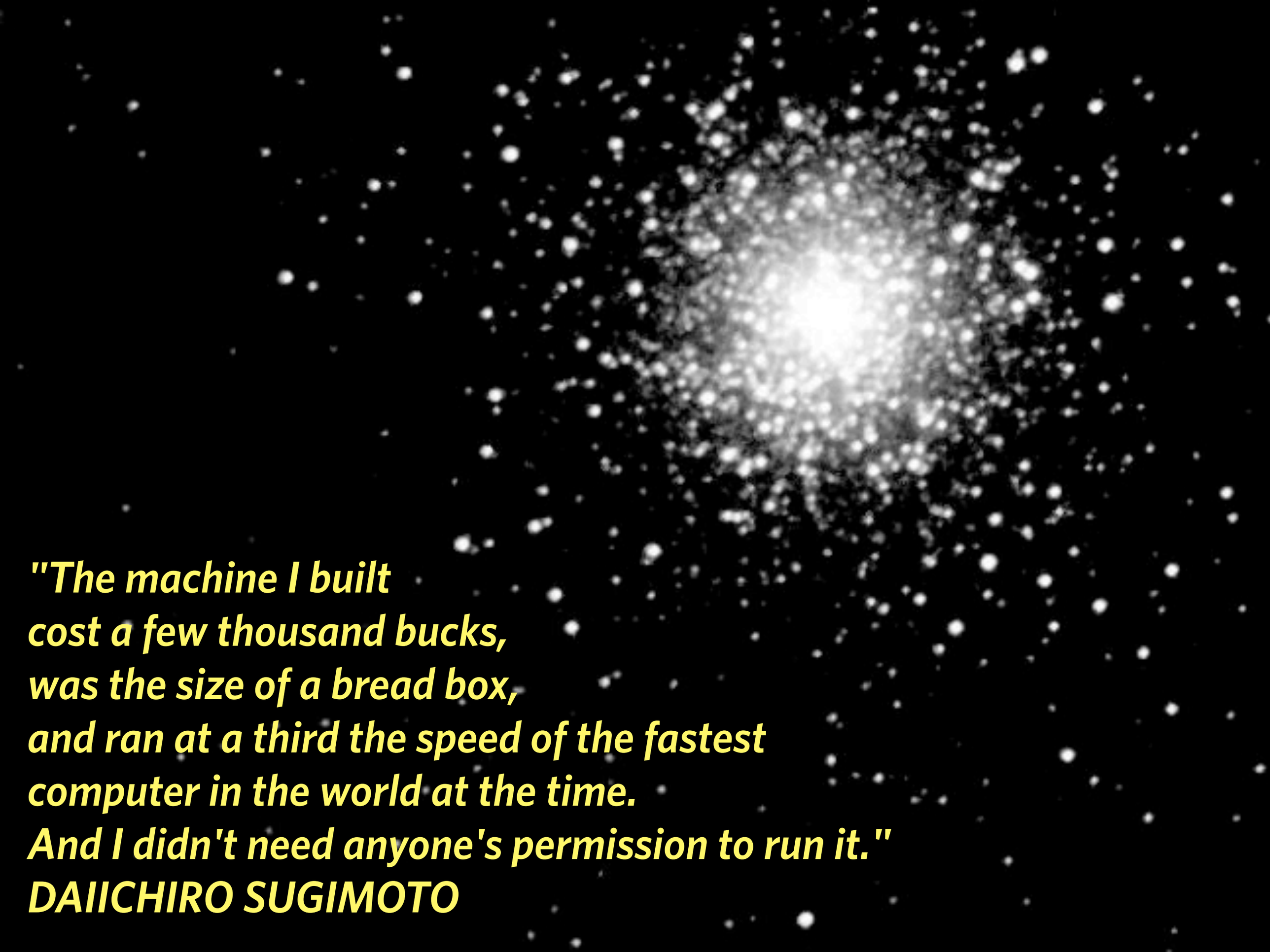
- *fast mat-vec:*
  - accelerate iterations of Krylov solvers
  - speeds-up Boundary Element Method (BEM) solvers



Background:

a bit of history and current affairs

N-body prompted a series of special-purpose machines (GRAPE) & has resulted in fourteen Gordon Bell awards overall



"The machine I built
cost a few thousand bucks,
was the size of a bread box,
and ran at a third the speed of the fastest
computer in the world at the time.
And I didn't need anyone's permission to run it."
DAIICHIRO SUGIMOTO

GRAPE (GRAvity PipE)

1st gen — 1989, 240 Mflop/s ...

4th gen — 1995, broke 1Tflop/s ... first Gordon Bell prize

seven GRAPE systems have received GB prizes

"Not only was GRAPE-4 the first teraflop supercomputer ever built, but it confirmed Sugimoto's theory that globular cluster cores oscillate like a beating heart."

The Star Machine, Gary Taubes, *Discover* 18, No. 6, 76-83 (June 1997)

GRAPE (GRAvity PipE)

1st gen — 1989, 240 Mflop/s ...

4th gen — 1995, broke 1Tflop/s ... first Gordon Bell prize

seven GRAPE systems have received GB prizes

14 Gordon Bell awards for N-body

- Performance 1992 — Warren & Salmon, 5 Gflop/s
  - Price/performance 1997 — Warren et al., 18 Gflop/s / $1 M
  - Price/performance 2009 — Hamada et al., 124 Mflop/s / $1
- Performance 2010 — Rahimian et al., 0.7 Pflop/s on Jaguar

Petascale direct numerical simulation of blood flow on 200K cores and heterogeneous architectures

Abtin Rahimian*, Ilya Lashuk*, Shravan K. Veerapaneni†, Aparna Chandramowlishwaran*
Dhairya Malhotra*, Logan Moon*, Rahul Sampath‡, Aashay Shringarpure*,
Jeffrey Vetter‡, Richard Vuduc*, Denis Zorin†, and George Biros*

14 Gordon Bell awards for N-body

- Performance 1992 — Warren & Salmon, 5 Gflop/s
  - Price/performance 1997 — Warren et al., 18 Gflop/s / $1 M
  - Price/performance 2009 — Hamada et al., 124 Mflop/s / $1
- Performance 2010 — Rahimian et al., 0.7 Pflop/s on Jaguar

6200x cheaper

Petascale direct numerical simulation of blood flow on 200K cores and heterogeneous architectures

Abtin Rahimian*, Ilya Lashuk*, Shravan K. Veerapaneni†, Aparna Chandramowlishwaran*
Dhairya Malhotra*, Logan Moon*, Rahul Sampath‡, Aashay Shringarpure*,
Jeffrey Vetter‡, Richard Vuduc*, Denis Zorin†, and George Biros*

14 Gordon Bell awards for N-body

- Performance 1992 — Warren & Salmon, 5 Gflop/s
  - Price/performance 1997 — Warren et al., 18 Gflop/s / $1 M
  - Price/performance 2009 — Hamada et al., 124 Mflop/s / $1
- Performance 2010 — Rahimian et al., 0.7 Pflop/s on Jaguar

34x more than Moore's law

6200x cheaper

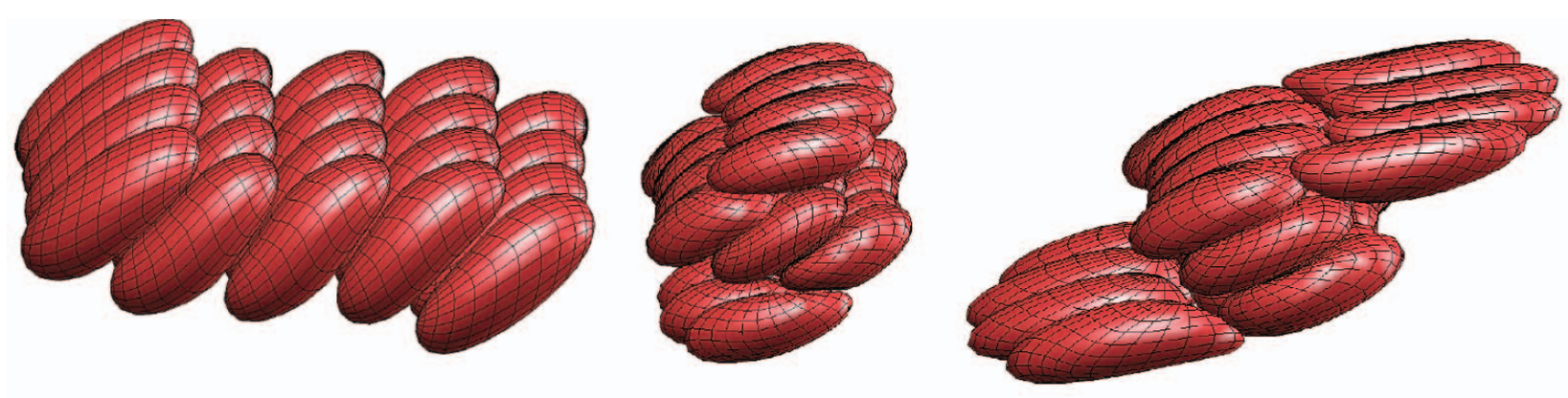
Petascale direct numerical simulation of blood flow on 200K cores and heterogeneous architectures

Abtin Rahimian*, Ilya Lashuk*, Shravan K. Veerapaneni†, Aparna Chandramowlishwaran*
Dhairya Malhotra*, Logan Moon*, Rahul Sampath‡, Aashay Shringarpure*,
Jeffrey Vetter‡, Richard Vuduc*, Denis Zorin†, and George Biros*

Petascale direct numerical simulation of blood flow on 200K cores and heterogeneous architectures

Abtin Rahimian*, Ilya Lashuk*, Shravan K. Veerapaneni†, Aparna Chandramowlishwaran*
Dhairya Malhotra*, Logan Moon*, Rahul Sampath‡, Aashay Shringarpure*,
Jeffrey Vetter‡, Richard Vuduc*, Denis Zorin†, and George Biros*

- largest simulation — ***90 billion unknowns***
- scale — 256 GPUs of Lincoln cluster / 196,608 cores of Jaguar
- numerical engine: FMM (kernel-independent version, 'kifmm')



N-body simulation on GPU hardware

The algorithmic and hardware speed-ups multiply

Early application of GPUs

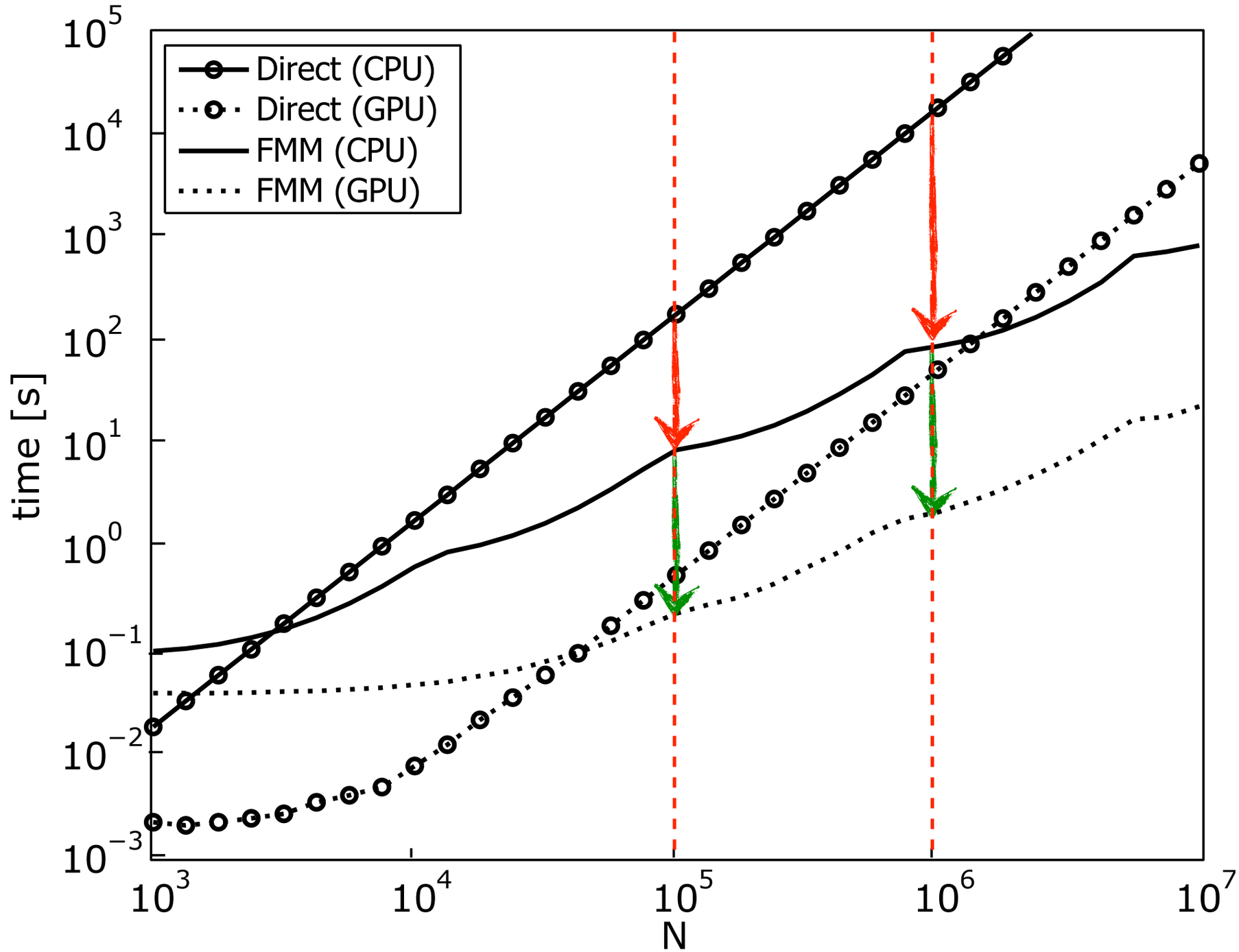
- 2007, Hamada & Iitaka — ‘CUNbody’
  - distributed source particles among thread blocks, requiring reduction
- 2007, Nyland et al. — GPU Gems 3
  - target particles were distributed, no reduction necessary
- 2008, Belleman et al. — ‘Kirin’ code
- 2009, Gaburov et al. — ‘Sapporo’ code

FMM on GPU — multiplying speed-ups

Note:

p=10

L^2-norm error (normalized): 10^{-4}



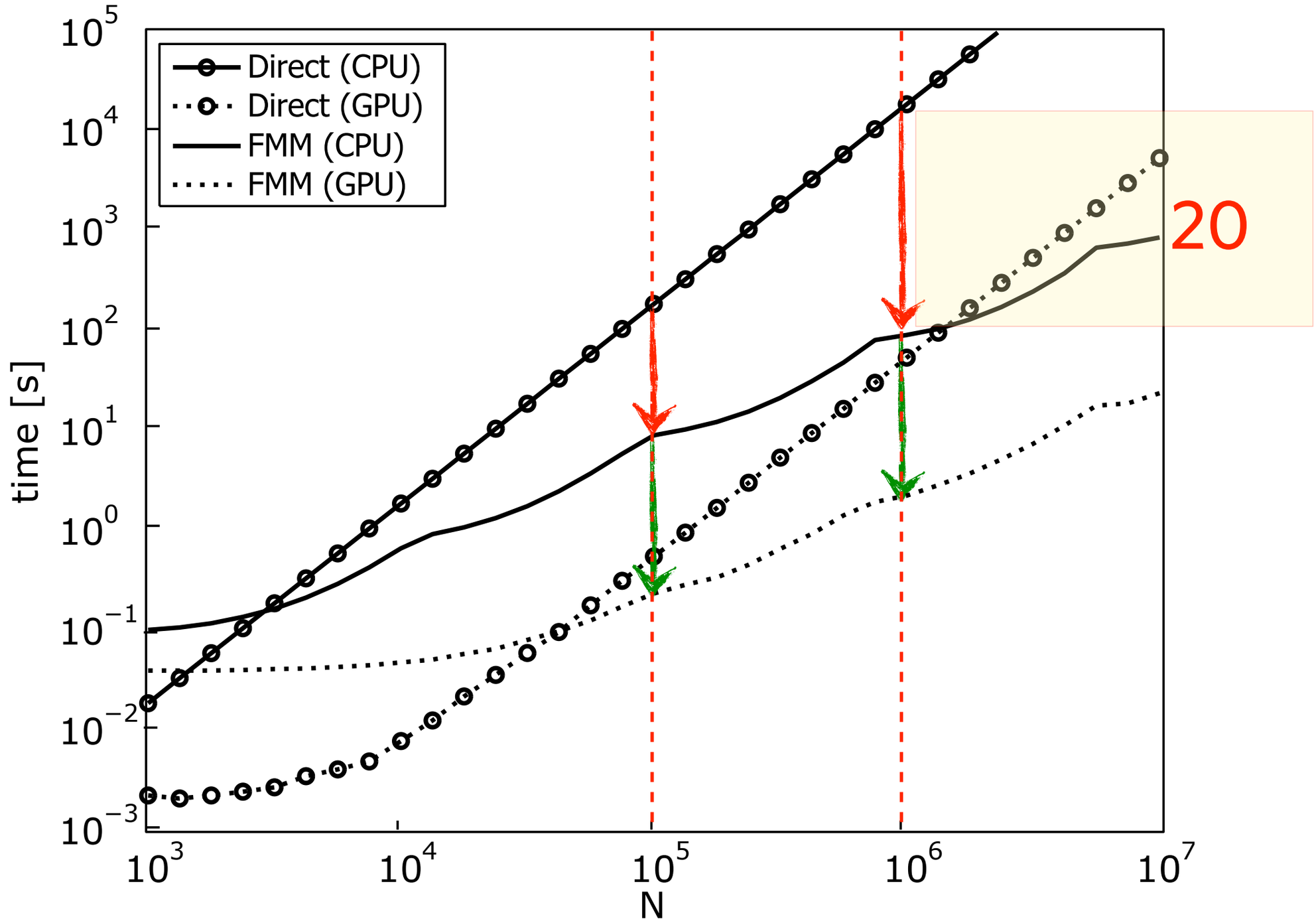
"Treecode and fast multipole method for N-body simulation with CUDA", R Yokota & L A Barba, Ch. 9 in ***GPU Computing Gems Emerald Edition,*** Elsevier/Morgan Kaufman (2011)

FMM on GPU — multiplying speed-ups

Note:

p=10

L^2-norm error (normalized): 10^{-4}



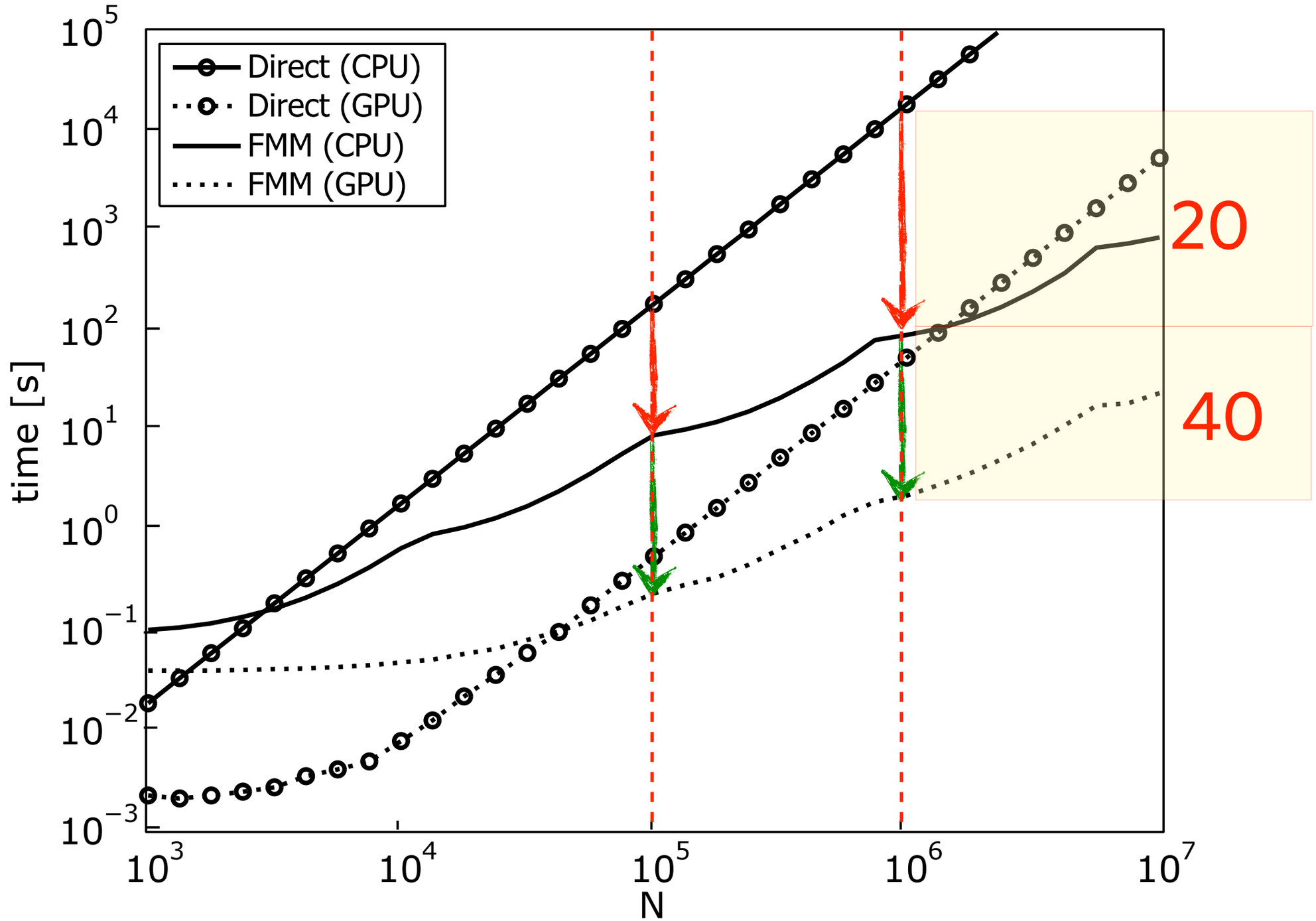
"Treecode and fast multipole method for N-body simulation with CUDA", R Yokota & L A Barba, Ch. 9 in ***GPU Computing Gems Emerald Edition,*** Elsevier/Morgan Kaufman (2011)

FMM on GPU — multiplying speed-ups

Note:

p=10

L^2-norm error (normalized): 10^{-4}



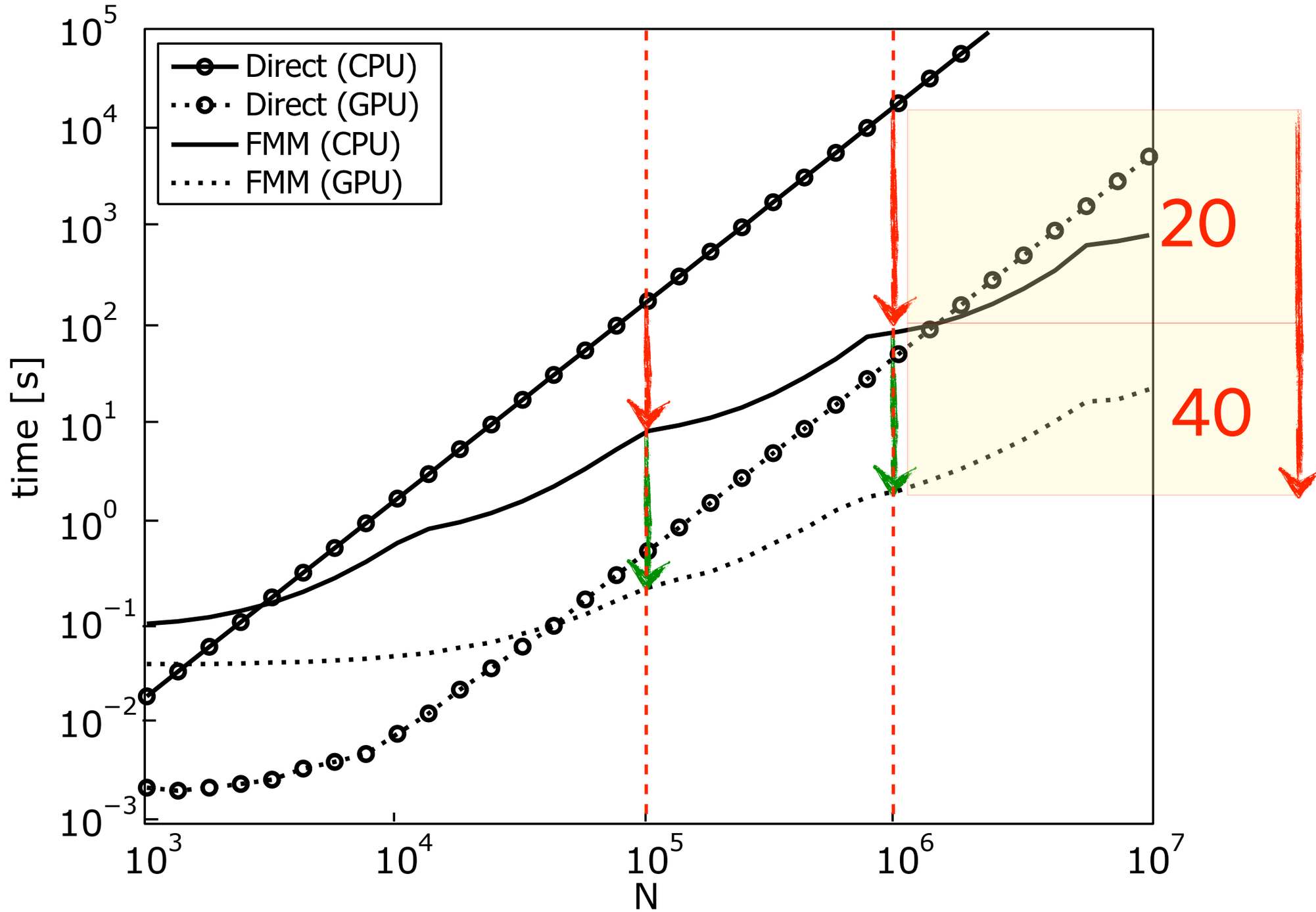
"Treecode and fast multipole method for N-body simulation with CUDA", R Yokota & L A Barba, Ch. 9 in ***GPU Computing Gems Emerald Edition,*** Elsevier/Morgan Kaufman (2011)

FMM on GPU — multiplying speed-ups

Note:

p=10

L^2-norm error (normalized): 10^{-4}

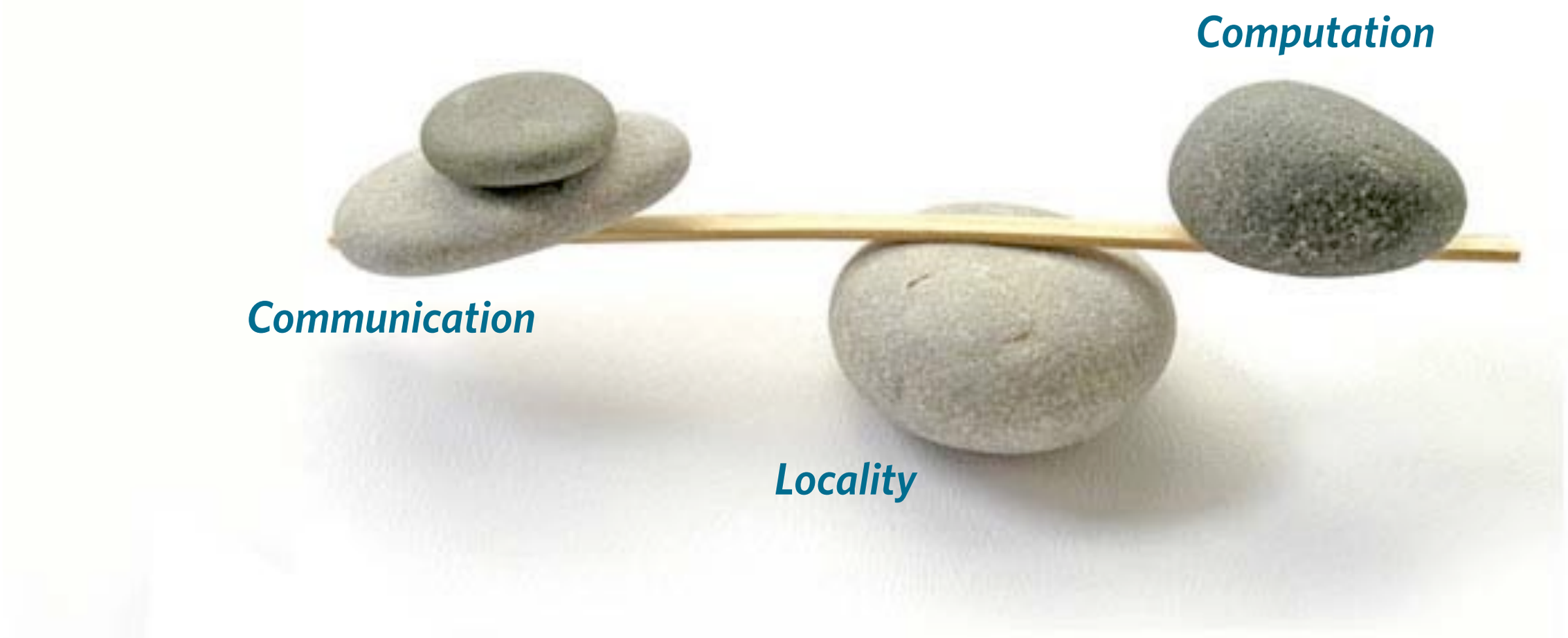


"Treecode and fast multipole method for N-body simulation with CUDA", R Yokota & L A Barba, Ch. 9 in ***GPU Computing Gems Emerald Edition,*** Elsevier/Morgan Kaufman (2011)

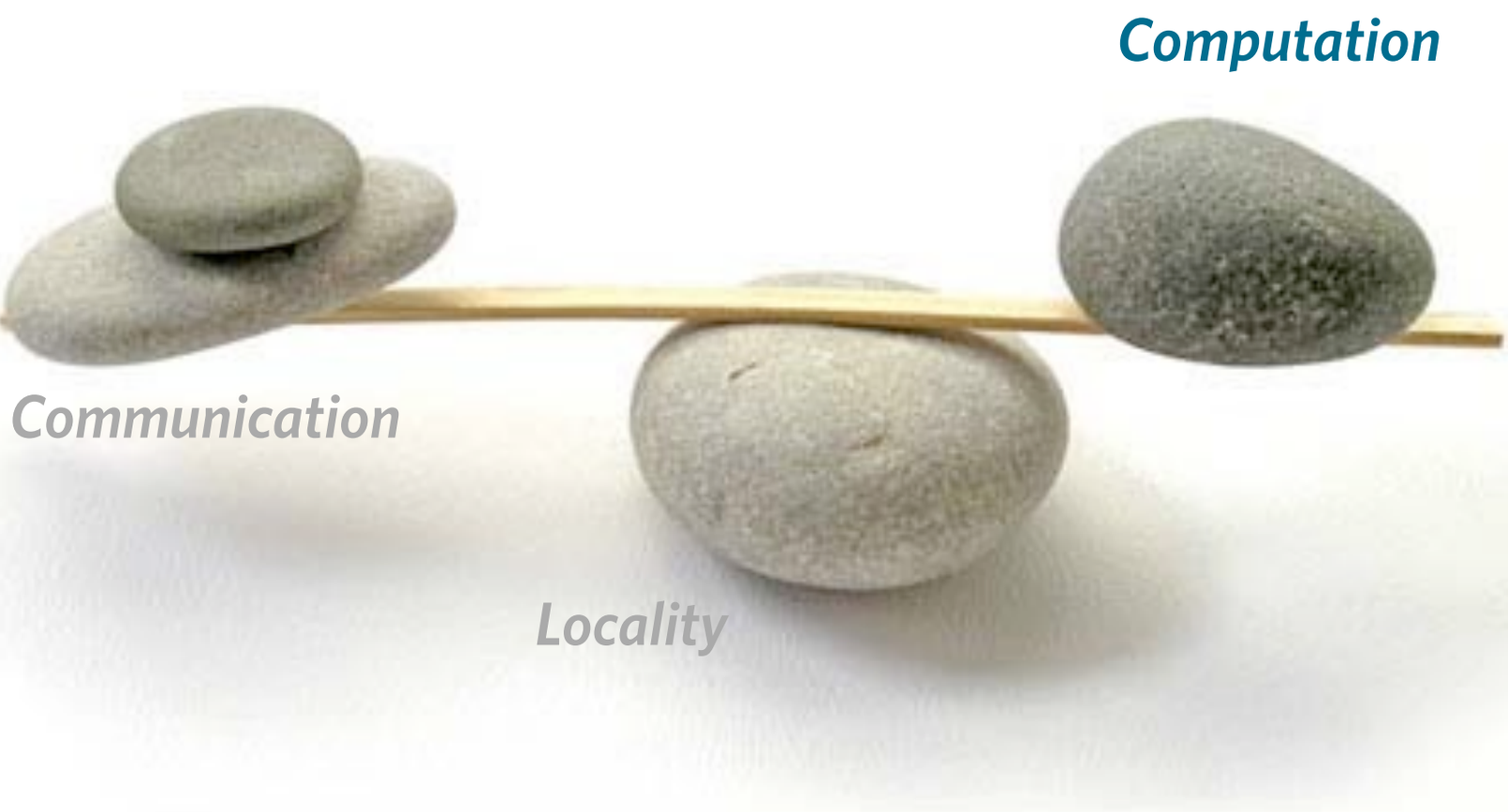
Advantage of N-body algorithms on GPUs

- quantify using the ***Roofline Model***
  - shows hardware barriers ('ceiling') on a computational kernel

- *Components of performance:*



Performance: Computation



Metric:

- Gflop/s
- dp / sp

Peak achivable if:

- exploit FMA, etc.
- non-divergence (GPU)

- Intra-node parallelism:
  - explicit in algorithm
  - explicit in code

Source: ParLab, UC Berkeley

Performance: Communication

Metric:

- GB/s

Peak achivable if optimizations are explicit

- prefetching
- allocation/usage
- stride streams
- coalescing on GPU

Source: ParLab, UC Berkeley

Performance: Locality

"Computation is free"

- Maximize locality > minimize communication
- Comm lower bound
  - minimize capacity misses
  - minimize conflict misses

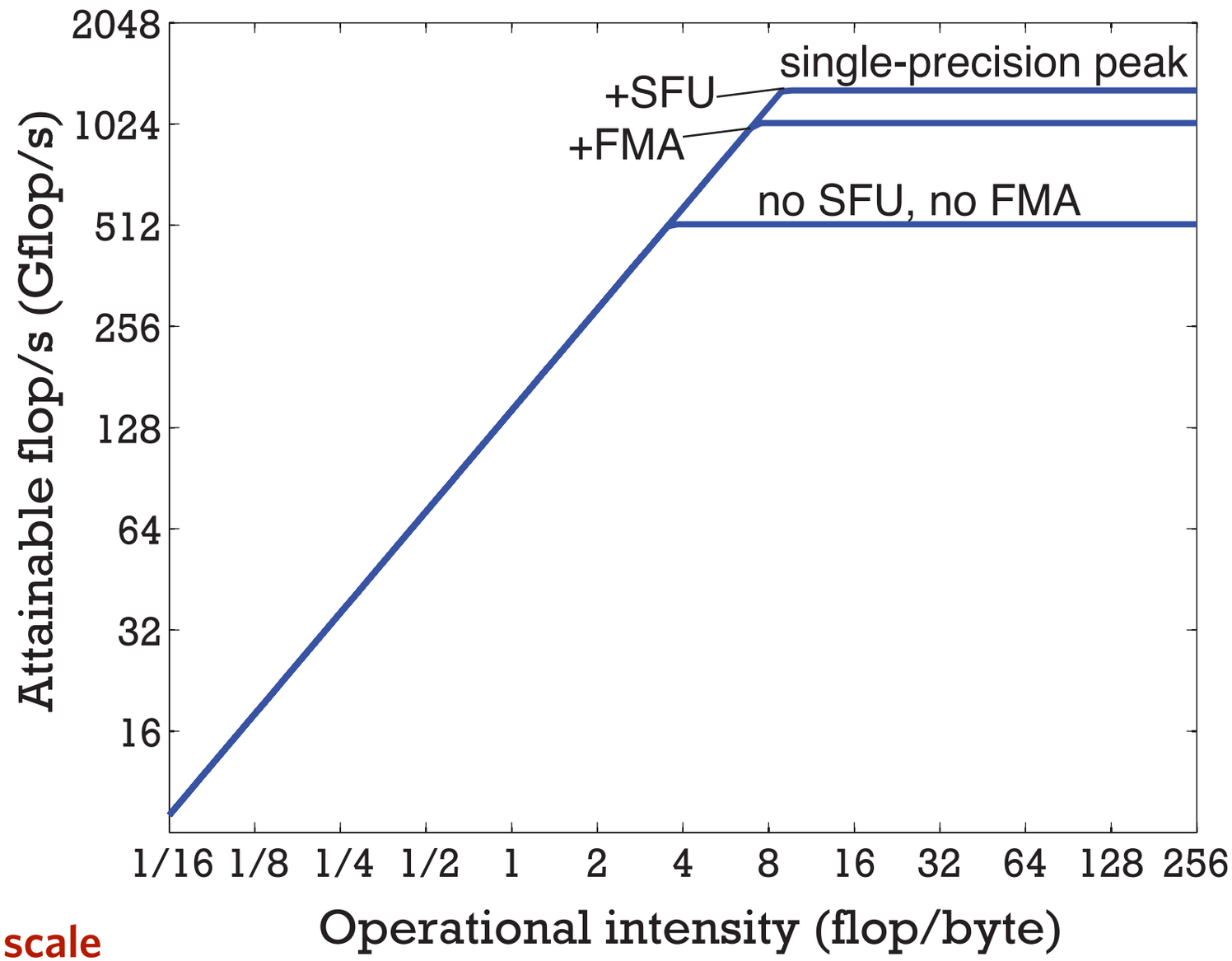
Hardware aids	*Optimizations via software*
cache size	blocking
associativities	padding

Source: ParLab, UC Berkeley

"Roofline: An Insightful Visual Performance Model for Multicore Architectures", S. Williams, A. Waterman, D. Patterson. *Communictions of the ACM*, April 2009.

Roofline model

- Operational intensity = total flop / total byte = Gflop/s / GB/s



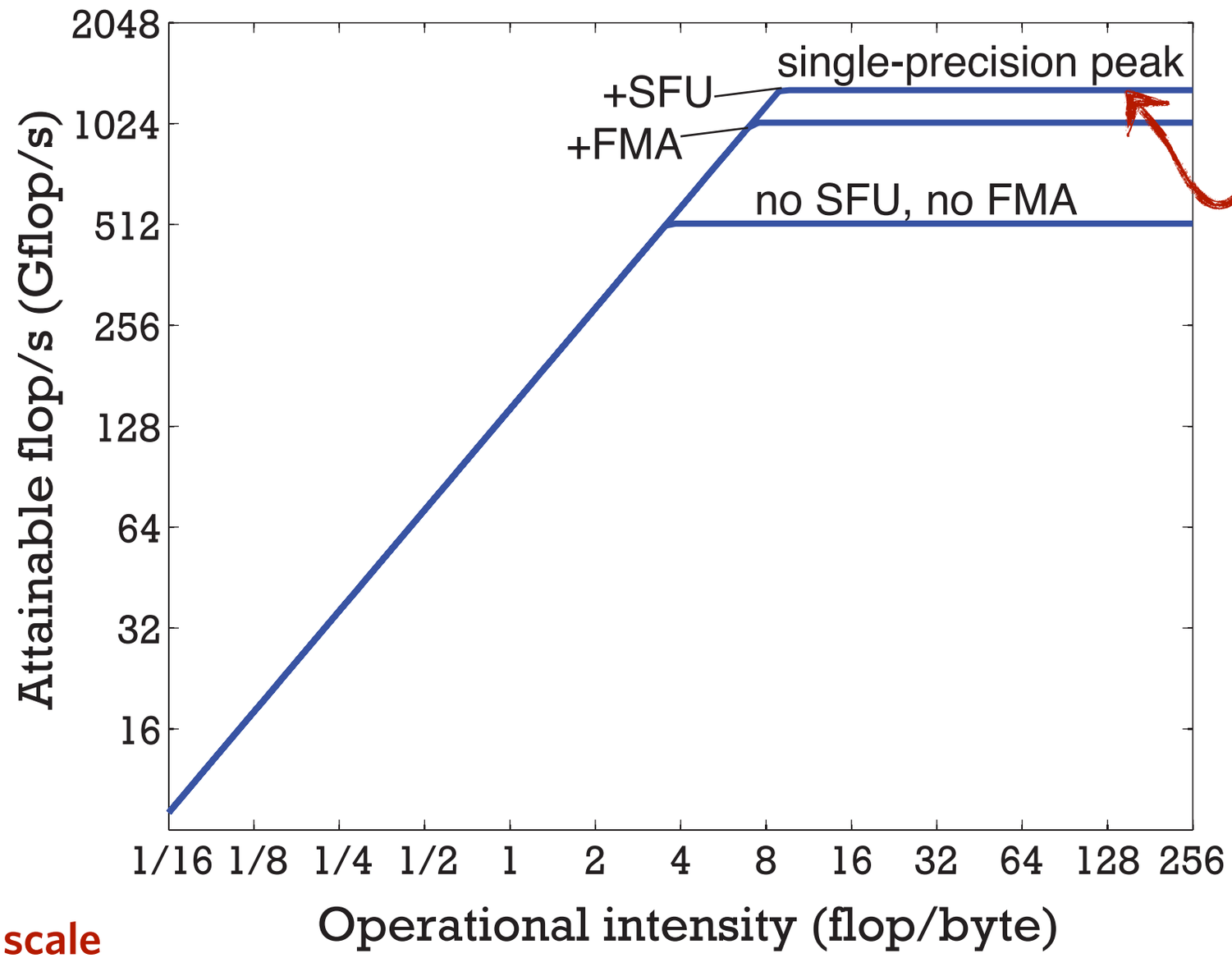
NVIDIA C2050

log/log scale

"Roofline: An Insightful Visual Performance Model for Multicore Architectures", S. Williams, A. Waterman, D. Patterson. *Communictions of the ACM*, April 2009.

Roofline model

- Operational intensity = total flop / total byte = Gflop/s / GB/s



peak floating-point performance

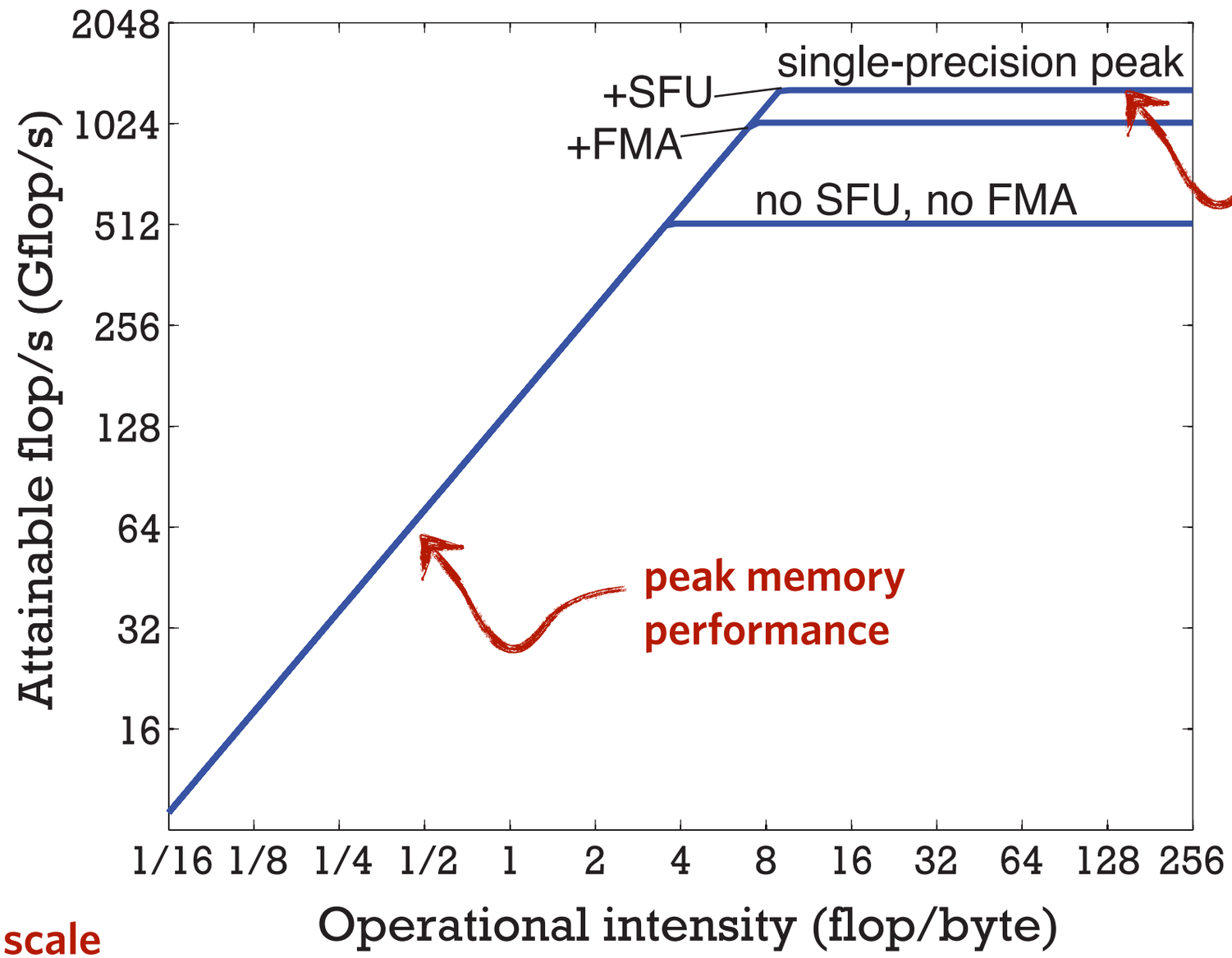
NVIDIA C2050

log/log scale

"Roofline: An Insightful Visual Performance Model for Multicore Architectures", S. Williams, A. Waterman, D. Patterson. *Communictions of the ACM*, April 2009.

Roofline model

- Operational intensity = total flop / total byte = Gflop/s / GB/s

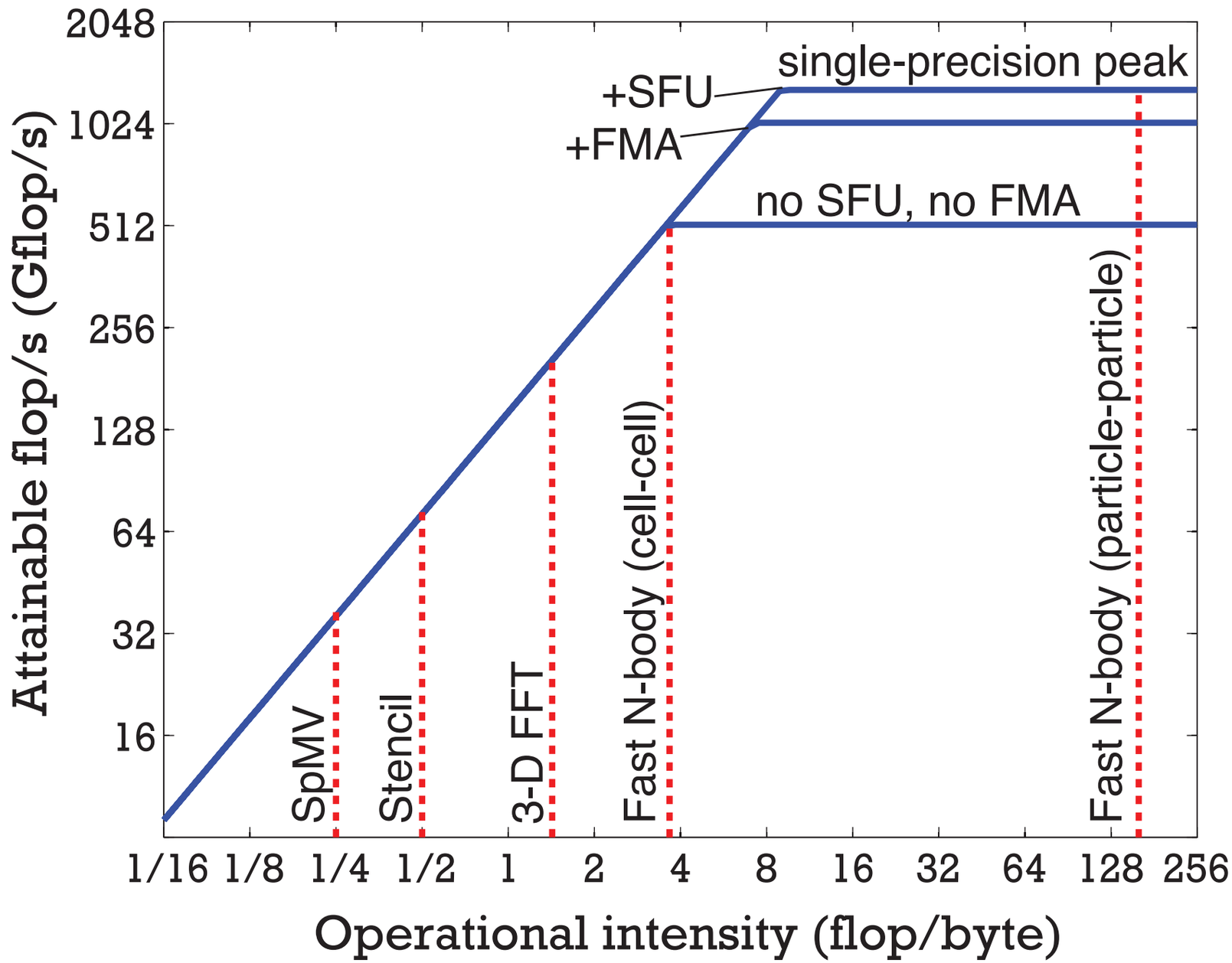


peak floating-point performance

NVIDIA C2050

log/log scale

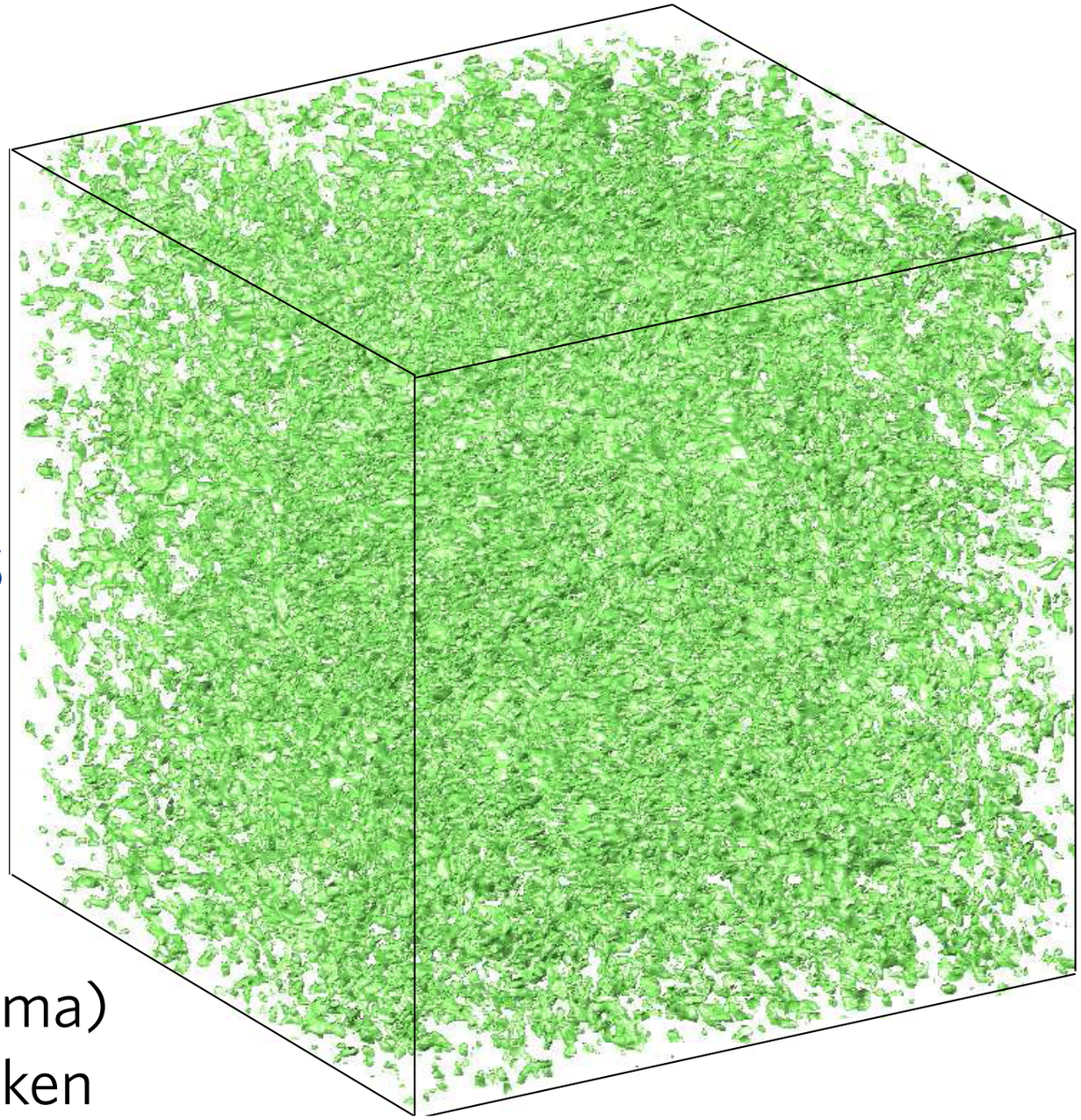
Advantage of N-body algorithms on GPUs



NVIDIA C2050

Image: "Hierarchical N-body simulations with auto-tuning for heterogeneous systems", Rio Yokota, L A Barba. *Computing in Science and Engineering (CiSE)*, 3 January 2012, IEEE Computer Society, doi:10.1109/MCSE.2012.1.

Scalability in many-GPUs & many-CPU systems



Our own progress so far:

1) 1 billion unknowns on 512 GPUs (Degima)
2) 32 billion on 32,768 processors of Kraken
3) 69 billion on 4096 GPUs of Tsubame 2.0
   achieved **1 petaflop/s on turbulence simulation**

http://www.bu.edu/exafmm/

Computer Physics Communications 182 (2011) 1272–1283

Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

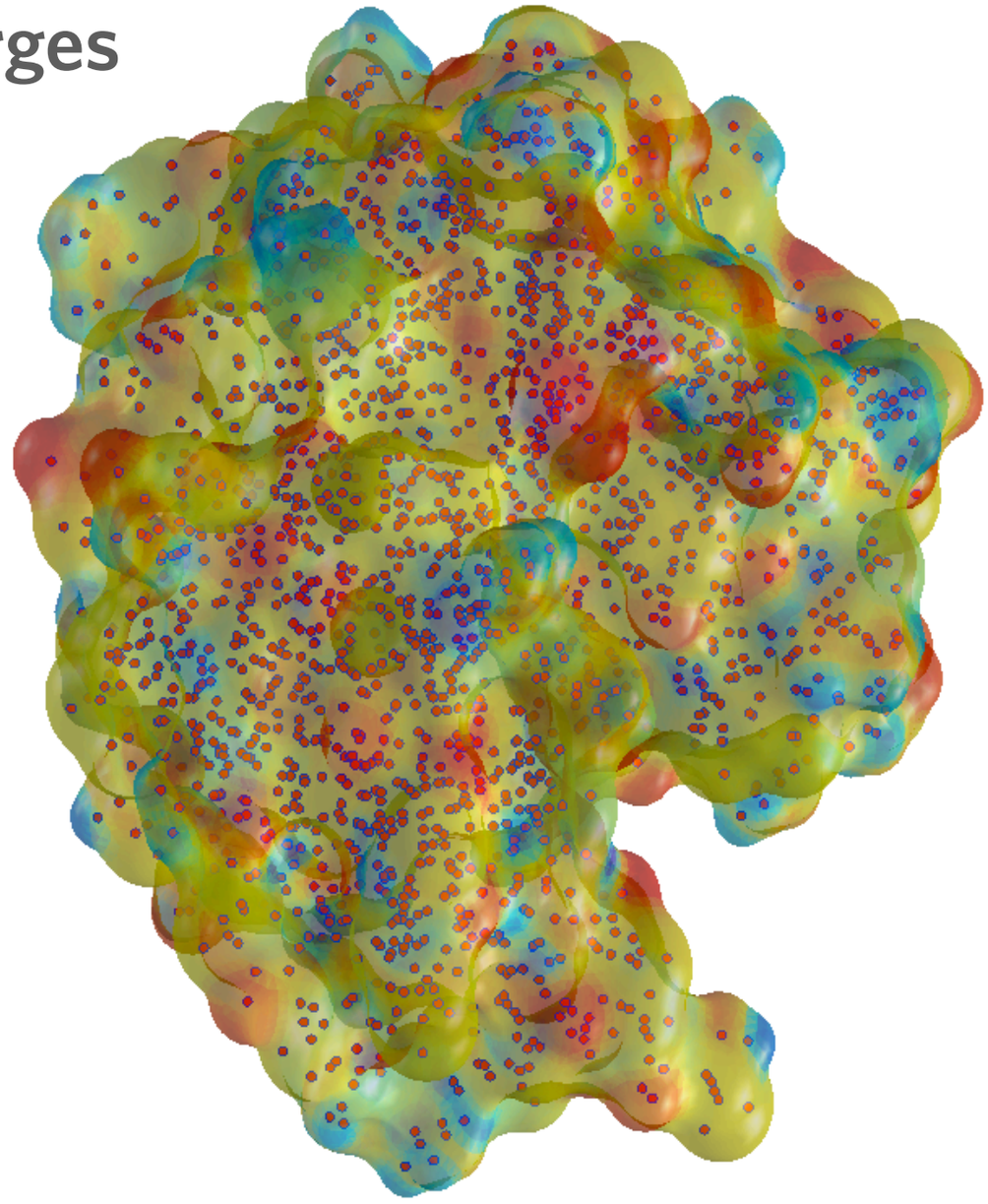
Biomolecular electrostatics using a fast multipole BEM on up to 512 GPUs and a billion unknowns

Rio Yokota [a], Jaydeep P. Bardhan [b], Matthew G. Knepley [c], L.A. Barba [a,*], Tsuyoshi Hamada [d]

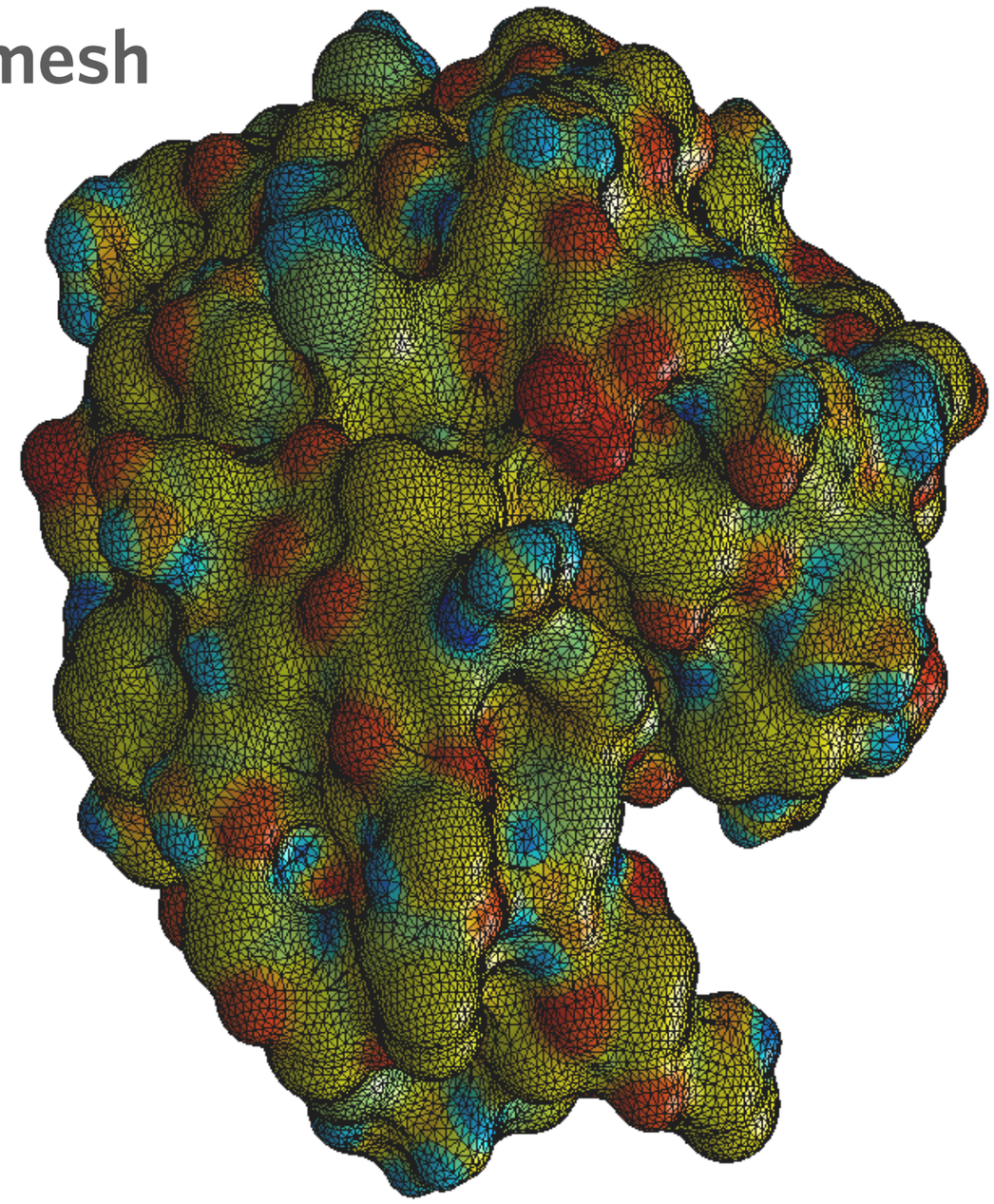
[a] Department of Mechanical Engineering, Boston University, Boston, MA 02215, United States
[b] Dept. of Molecular Biophysics and Physiology, Rush University Medical Center, Chicago, IL 60612, United States
[c] Computation Institute, University of Chicago, Chicago, IL 60637, United States
[d] Nagasaki University, Advanced Computing Center (NACC), Nagasaki, Japan

Lysozyme molecule

charges

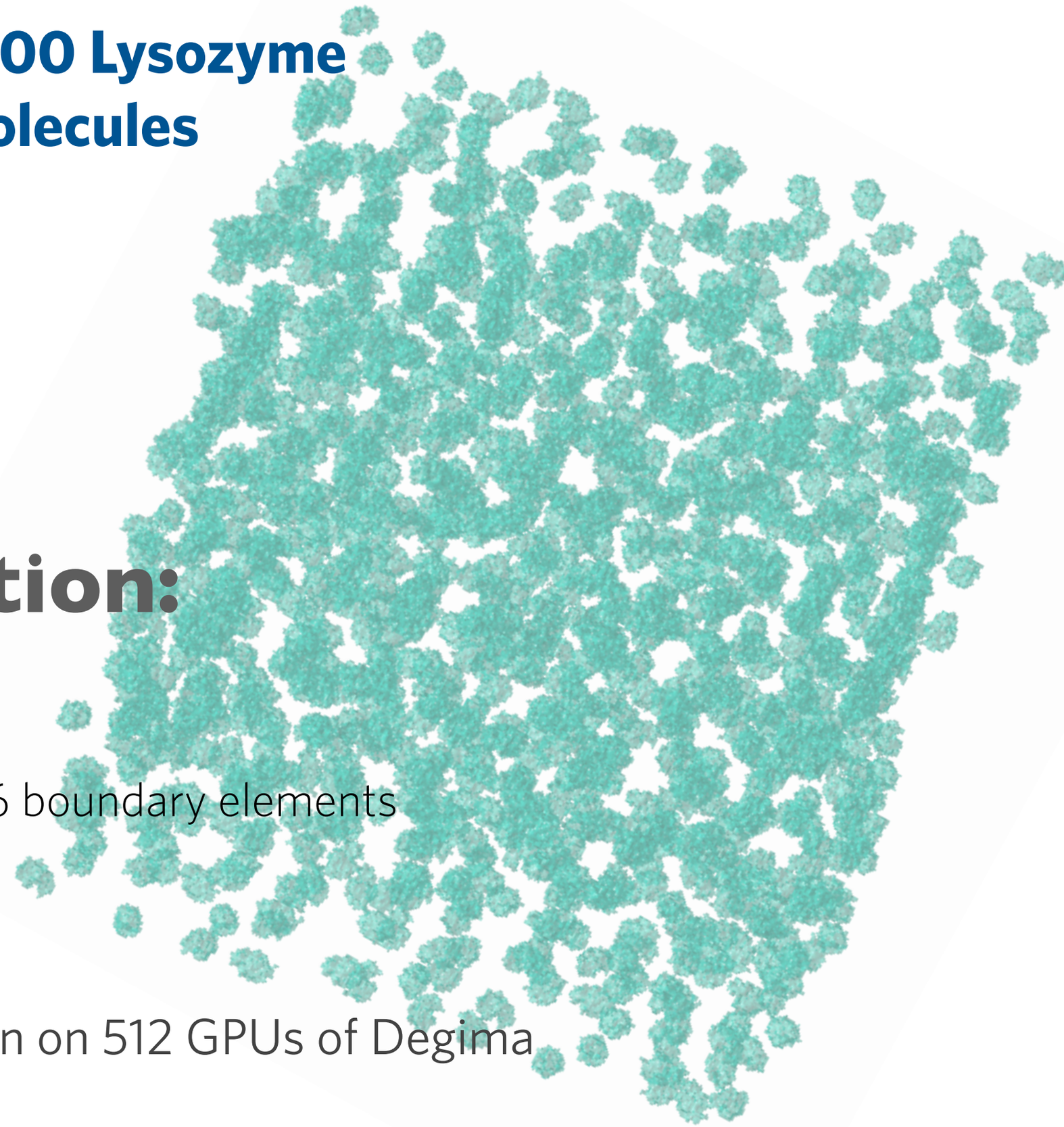


mesh



discretized with 102,486 boundary elements

1000 Lysozyme molecules



largest calculation:

- 10,648 molecules
- each discretized with 102,486 boundary elements
- more than 20 million atoms
- **1 billion unknowns**

one minute per iteration on 512 GPUs of Degima

Degima cluster

Nagasaki Advanced Computing Center

Kraken

Cray XT5 system at NICS, Tennessee:
9,408 nodes with 12 CPU cores each,
16 GB memory

peak performance is 1.17 Petaflop/s.
 # 11 in Top500 (Jun'11 & Nov'11)

TeraGrid

Weak scaling on Kraken

p=3
N=10⁶, per process

parallel efficiency
72% at 32,768
processors

largest run:

32 Billion points

time to solution:

<40 s

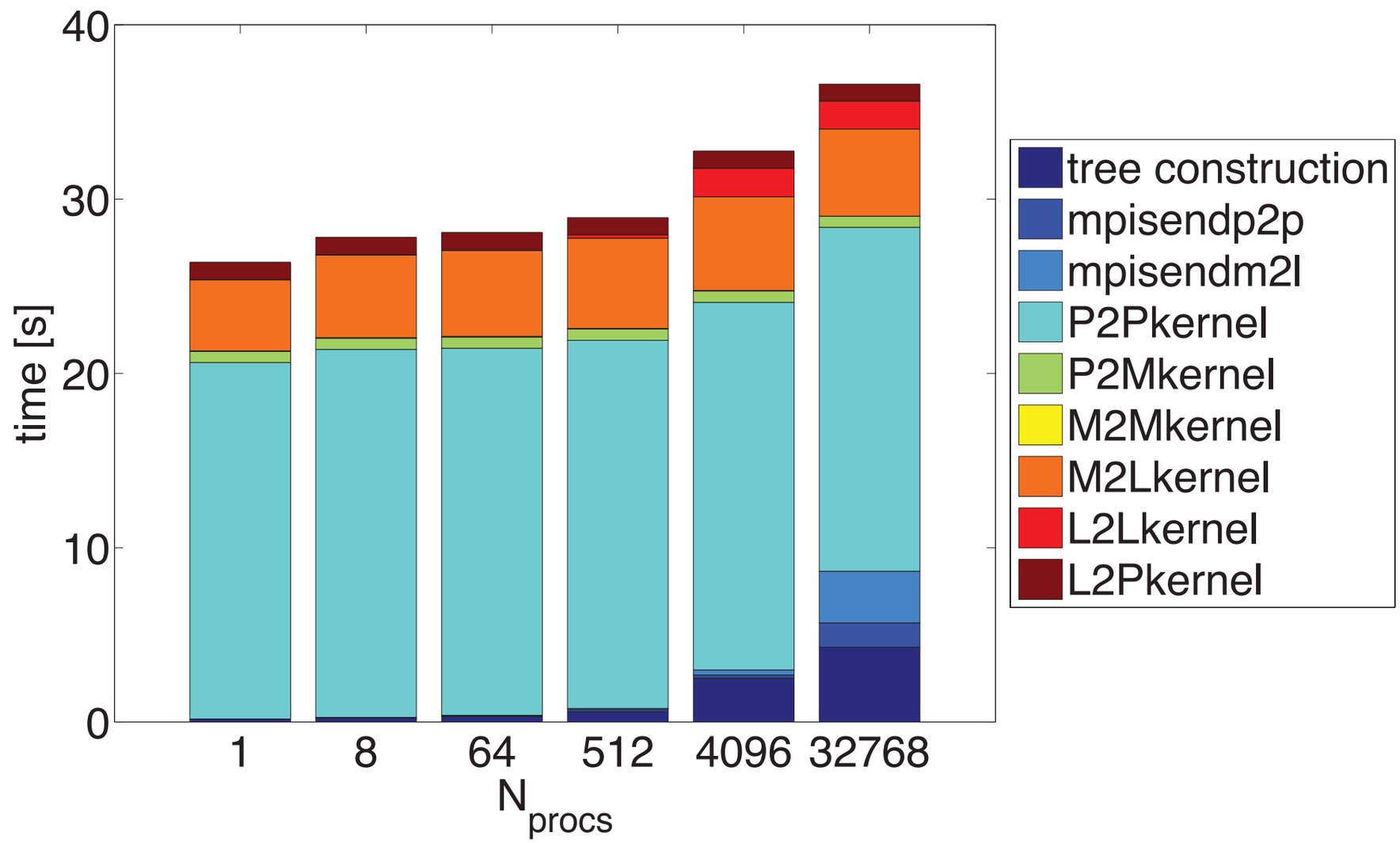


Image: "A tuned and scalable fast multipole method as a preeminent algorithm for exascale systems", Rio Yokota, L A Barba. *Int. J. High-perf. Comput*. Accepted (2011) — To appear.

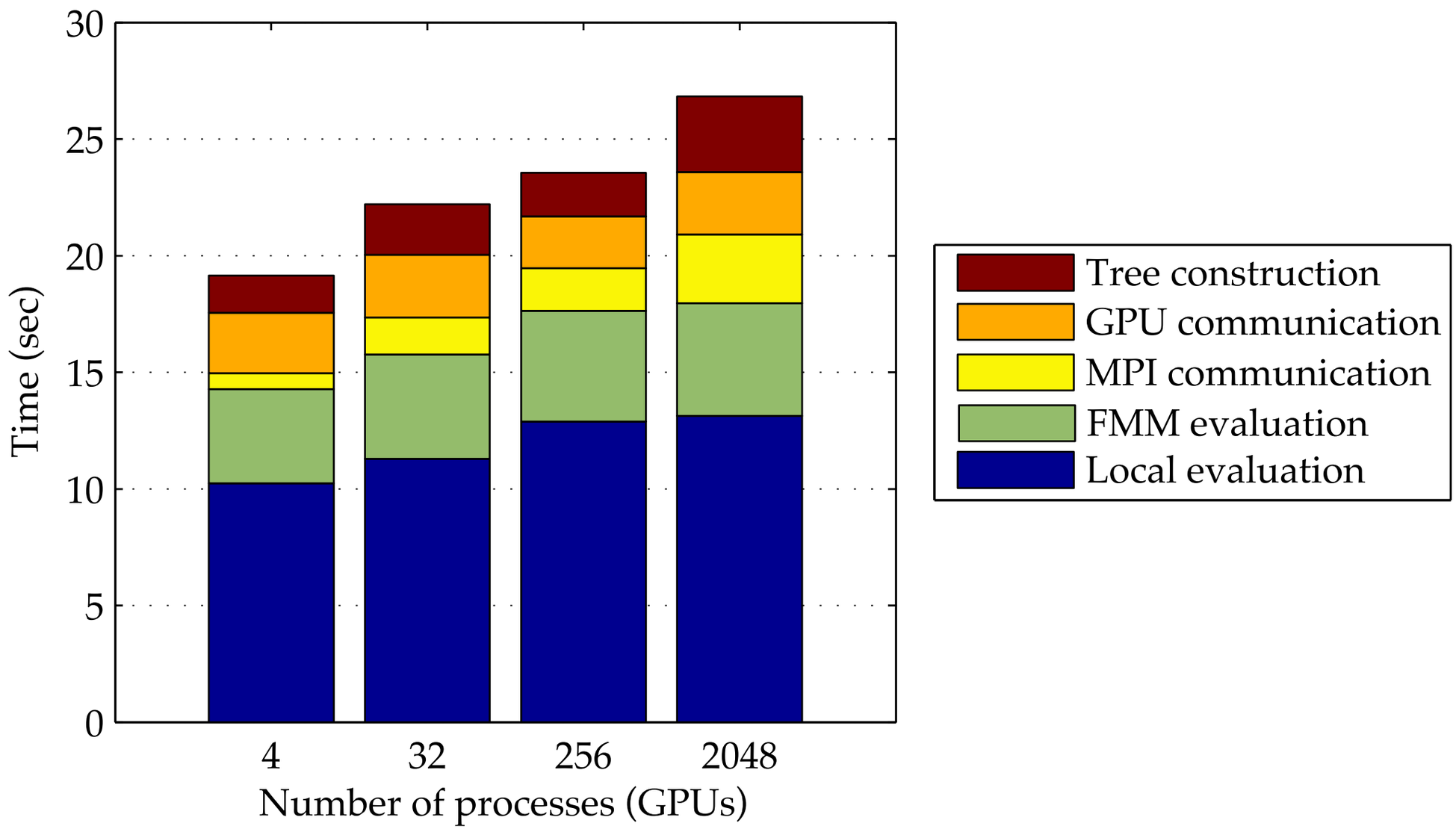
Tsubame 2.0

1408 nodes with 12 CPU cores each,
3 NVIDIA M2050 GPUs,
54 GB of RAM.

Total of 4224 GPUs
peak performance 2.4 Petaflop/s.
 # 5 in Top500 (Jun'11 & Nov'11)

Weak scaling on Tsubame

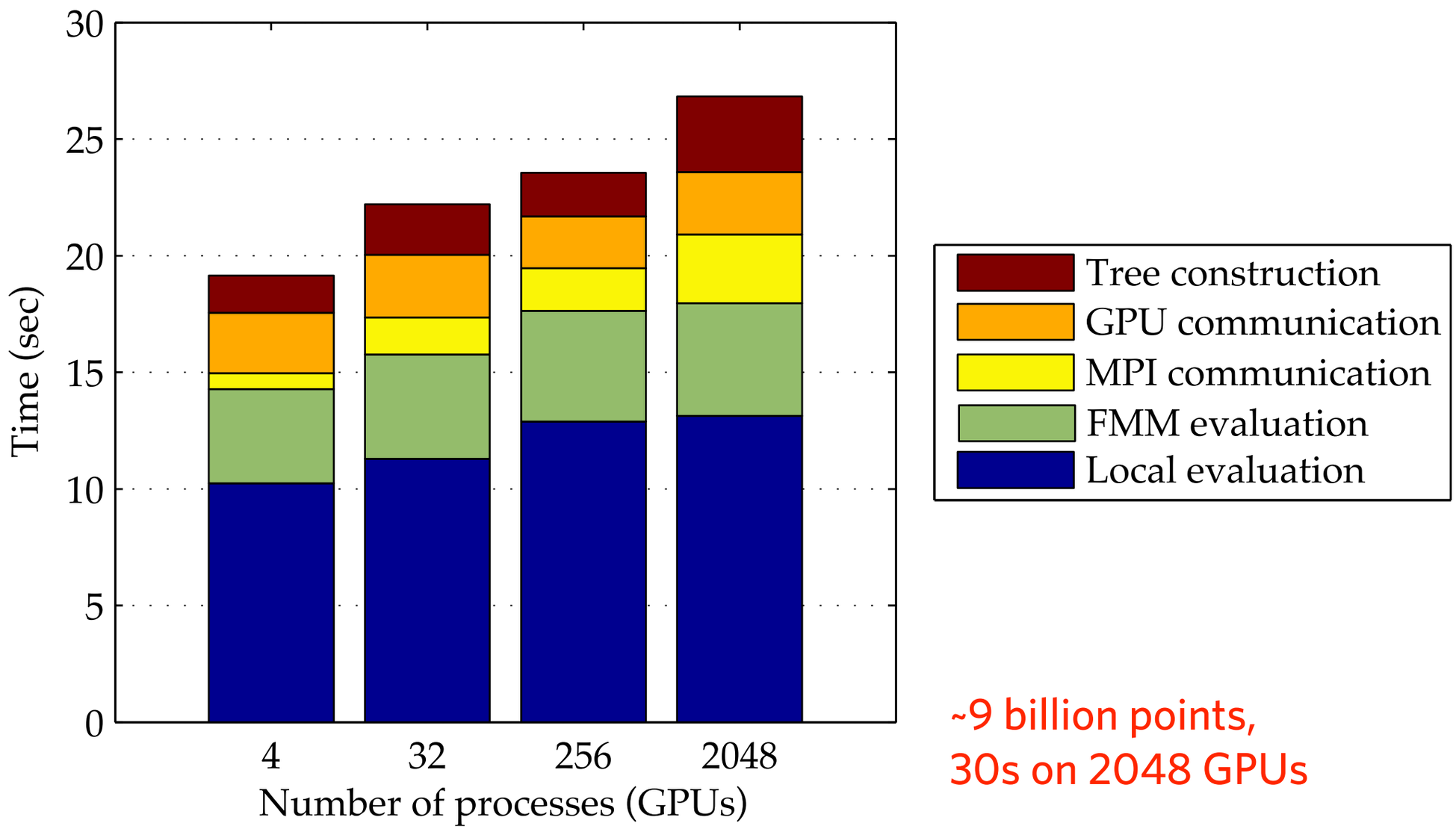
- 4 million points per process



"Petascale turbulence simulation using a highly parallel fast multipole method", Rio Yokota, L A Barba, Tetsu Narumi, Kenji Yasuoka. *Comput. Phys. Commun.*, under revision (minor)
Preprint arXiv:1106.5273

Weak scaling on Tsubame

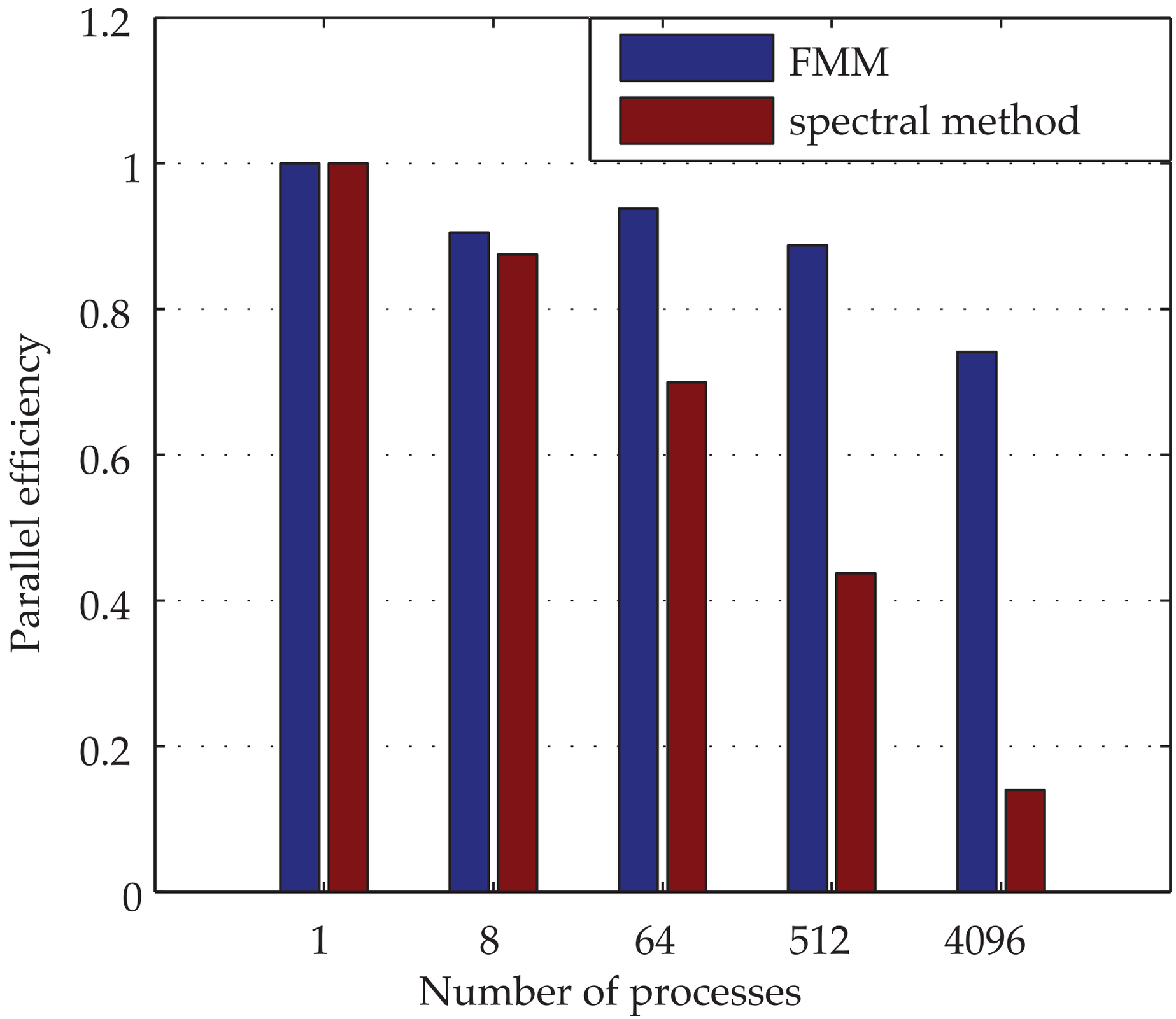
- 4 million points per process



~9 billion points,
30s on 2048 GPUs

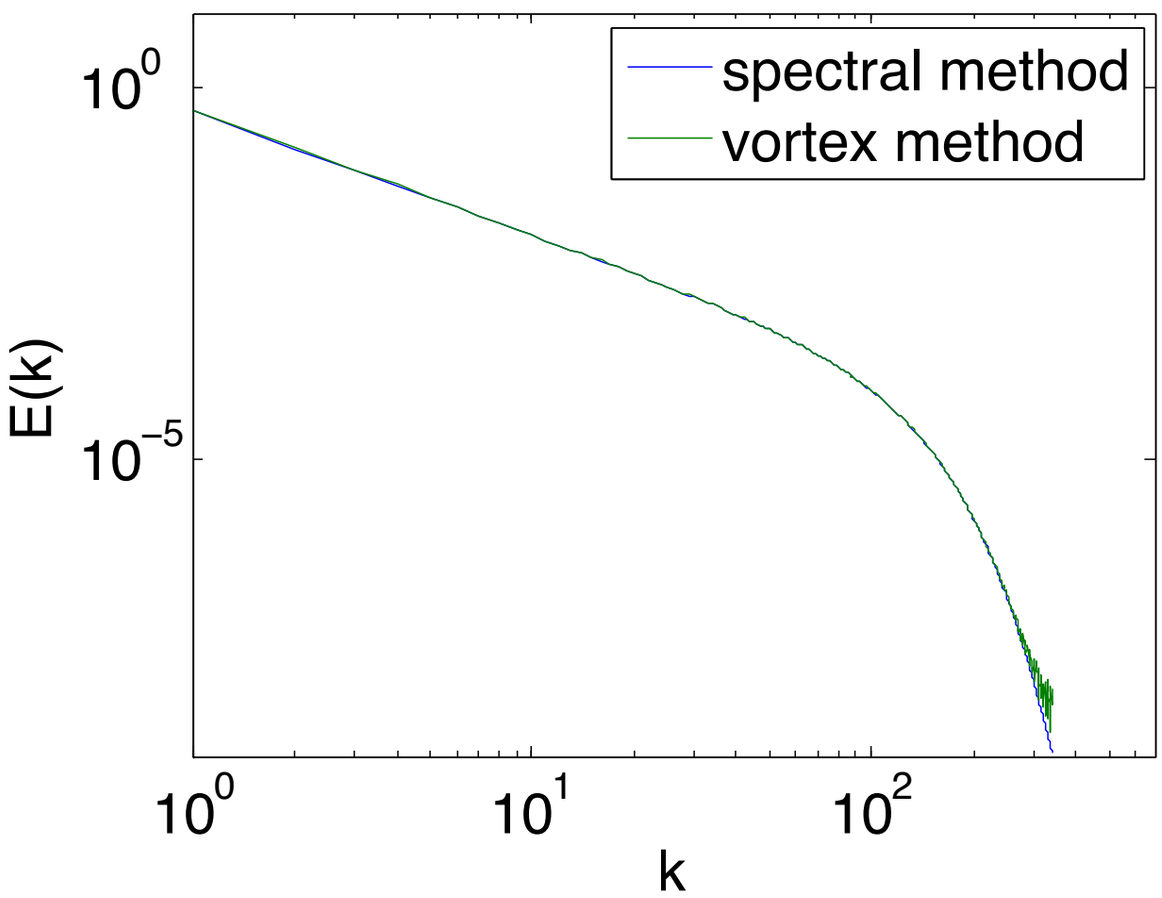
"Petascale turbulence simulation using a highly parallel fast multipole method", Rio Yokota, L A Barba, Tetsu Narumi, Kenji Yasuoka. *Comput. Phys. Commun.*, under revision (minor)
Preprint arXiv:1106.5273

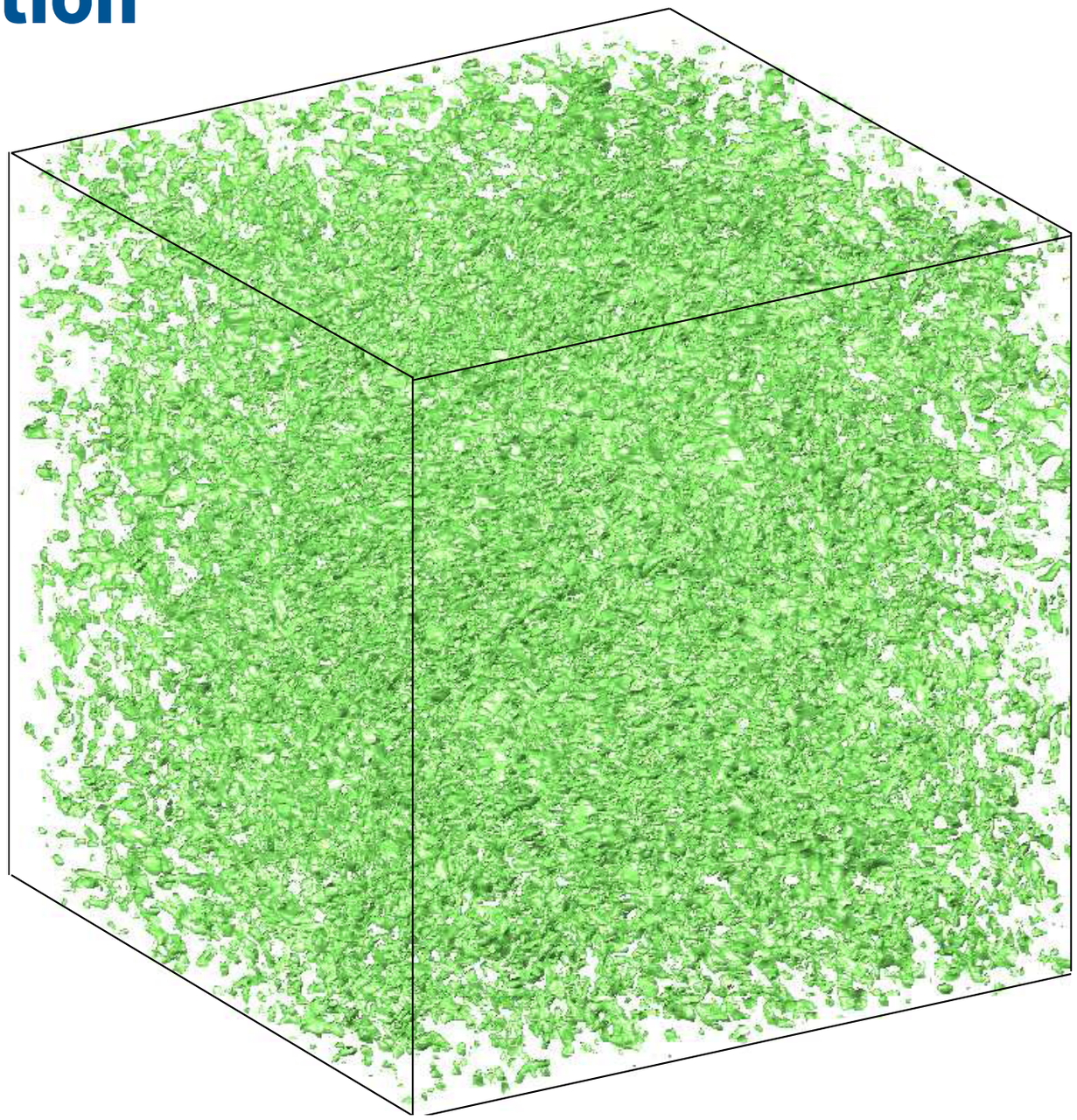
FMM vs. FFT, weak scaling



Petascale turbulence simulation

- using vortex method
- $4{,}096^3$ grid, **69 billion** points
- **1 Pflop/s**
- Energy spectrum well-matched





"Petascale turbulence simulation using a highly parallel fast multipole method", Rio Yokota, L A Barba, Tetsu Narumi, Kenji Yasuoka.
Comput. Phys. Commun., under revision (minor)
Preprint arXiv:1106.5273

New hybrid Treecode/FMM with auto-tuning

ExaFMM code base: www.bu.edu/exafmm

Computing in Science & Engineering

Hierarchical N-body simulations with auto-tuning for heterogeneous systems (PDF)

PrePrint
ISSN: 1521-9615

Rio Yokota, Boston University, Boston
Lorena Barba, Boston University, Boston

DOI Bookmark: http://doi.ieeecomputersociety.org/10.1109/MCSE.2012.1

ABSTRACT

Algorithms designed to efficiently solve this classical problem of physics fit very well on GPU hardware, and exhibit excellent scalability on many GPUs. Their computational intensity makes them a promising approach for many other applications amenable to an N-body formulation. Adding features such as auto-tuning makes multipole-type algorithms ideal for heterogeneous computing environments.

"Hierarchical N-body simulations with auto-tuning for heterogeneous systems",
Rio Yokota, L A Barba.
Computing in Science and Engineering (CiSE), 3 January 2012, IEEE Computer Society, doi:10.1109/MCSE.2012.1.
Preprint arXiv:1108.5815

Hybrid treecode/FMM — Purpose of auto-tuning

- *Choices:*
  - Cartesian vs. spherical expansions
  - rotation-based vs. plane wave-based translations
  - cell-cell vs. cell-particle interactions for far field
  - order of expansion (p) vs. MAC-based error control
- *Depend on:*
  - required accuracy
  - hardware
  - implementation

So What?

Hybrid Treecode/FMM liberates the user from
(i) deciding between treecode & FMM for their application
(ii) there is no need to tweak parameters, e.g., particles per cell

License

For maximum freedom of use, **ExaFMM is distributed under The MIT License (MIT).** Please note that you must give proper attribution in all derived works.

www.bu.edu/exafmm

to conclude

Hierarchical N-body algorithms are well-suited for achieving exascale

FMM is a particularly favorable algorithm for the emerging heterogeneous, many-core architectural landscape.

Spatial and temporal locality

In the sense of: Bergman et al. (2008) “Exascale Computing Study”, DARPA IPTO

Spatial and temporal locality

- Algorithm has intrinsic geometric locality

In the sense of: Bergman et al. (2008) “Exascale Computing Study”, DARPA IPTO

Spatial and temporal locality

- Algorithm has intrinsic geometric locality
- Acces patterns *could* be non-local

In the sense of: Bergman et al. (2008) “Exascale Computing Study”, DARPA IPTO

Spatial and temporal locality

- Algorithm has intrinsic geometric locality
- Acces patterns *could* be non-local
  - work with sorted particle indices, access via a start-offset combination

In the sense of: Bergman et al. (2008) “Exascale Computing Study”, DARPA IPTO

Spatial and temporal locality

- Algorithm has intrinsic geometric locality
- Acces patterns *could* be non-local
  - work with sorted particle indices, access via a start-offset combination
- Temporal locality:

In the sense of: Bergman et al. (2008) “Exascale Computing Study”, DARPA IPTO

Spatial and temporal locality

- Algorithm has intrinsic geometric locality
- Acces patterns *could* be non-local
  - work with sorted particle indices, access via a start-offset combination
- Temporal locality:
  - queue GPU tasks before execution, buffer the input and output of data making memory access contiguous

In the sense of: Bergman et al. (2008) “Exascale Computing Study”, DARPA IPTO

Spatial and temporal locality

- Algorithm has intrinsic geometric locality
- Acces patterns *could* be non-local
  - work with sorted particle indices, access via a start-offset combination
- Temporal locality:
  - queue GPU tasks before execution, buffer the input and output of data making memory access contiguous

➡ *The FMM is **not** a locality-sensitive application*

In the sense of: Bergman et al. (2008) "Exascale Computing Study", DARPA IPTO

Global data comunications and synchronization

- Two most time-consuming in the FMM:
  - P2P — purely local
  - M2L — "hierarchical synchronization"