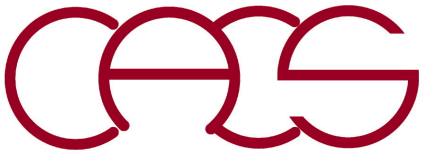
Parallel Quantum Molecular Dynamics

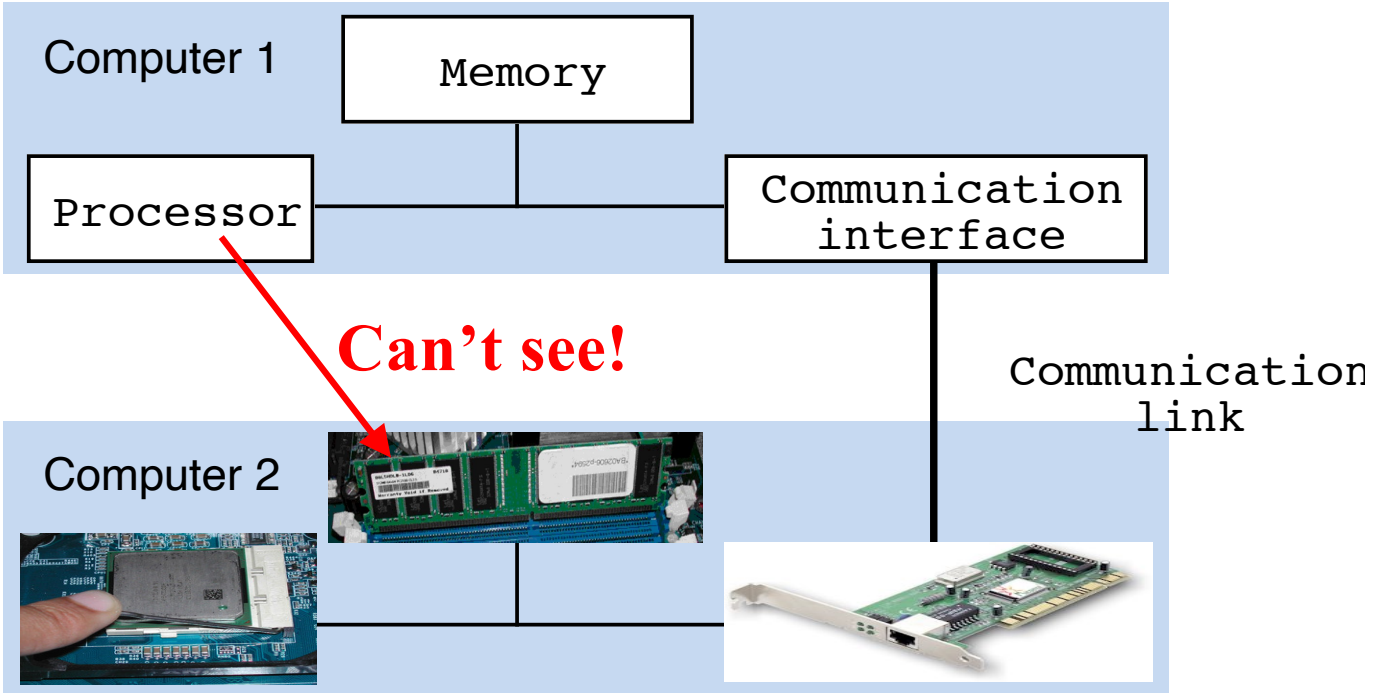
Aiichiro Nakano

Collaboratory for Advanced Computing & Simulations
Department of Computer Science
Department of Physics & Astronomy
Department of Quantitative & Computational Biology
University of Southern California

Email: anakano@usc.edu



Parallel Computing Hardware

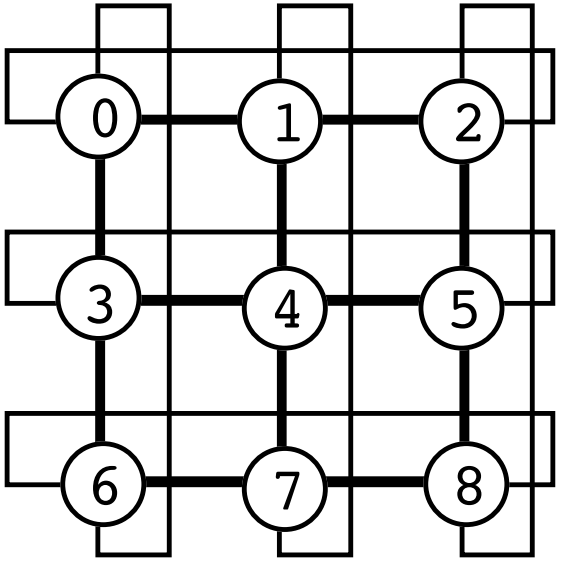


- **Processor: Executes arithmetic & logic operations**
- **Memory: Stores program & data (stored program computer)**
- **Communication interface: Performs signal conversion & synchronization between communication link & a computer**
- **Communication link: A wire capable of carrying a sequence of bits as electrical (or optical) signals**

See https://aiichironakano.github.io/cs596.html

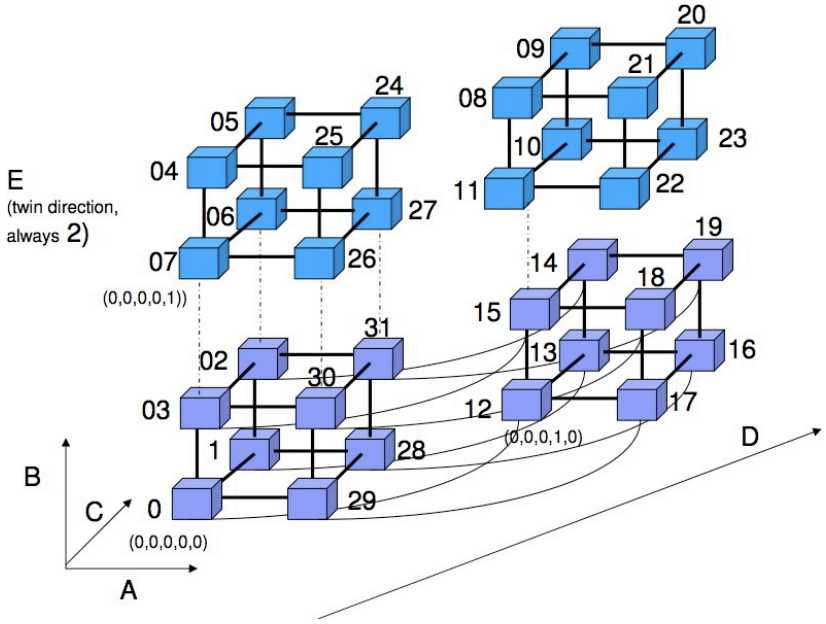
Communication Network

Mesh (torus)

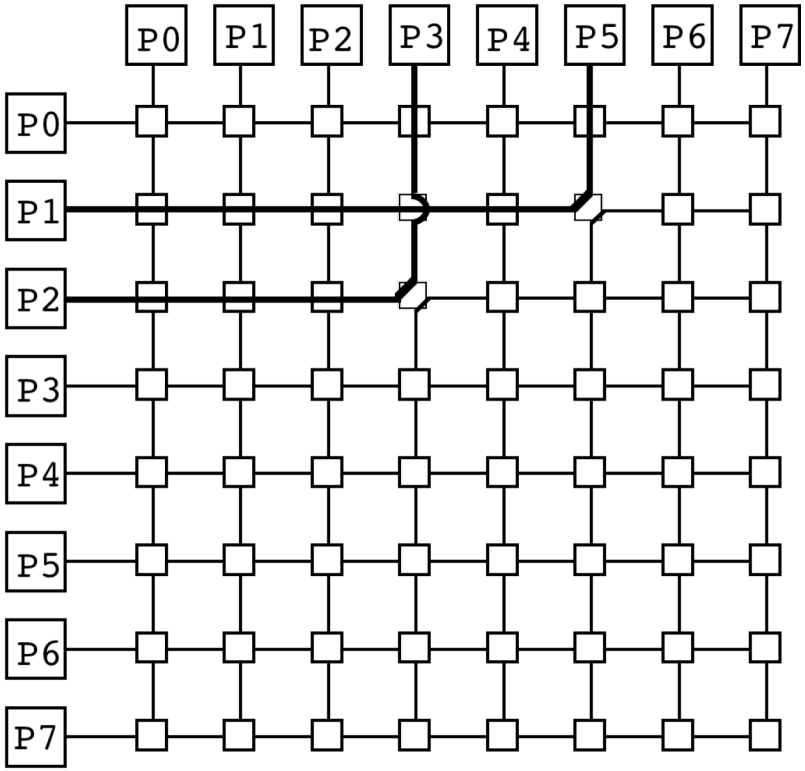


NEC Earth Simulator (640x640 crossbar)

IBM Blue Gene/Q (5D torus)



Crossbar switch



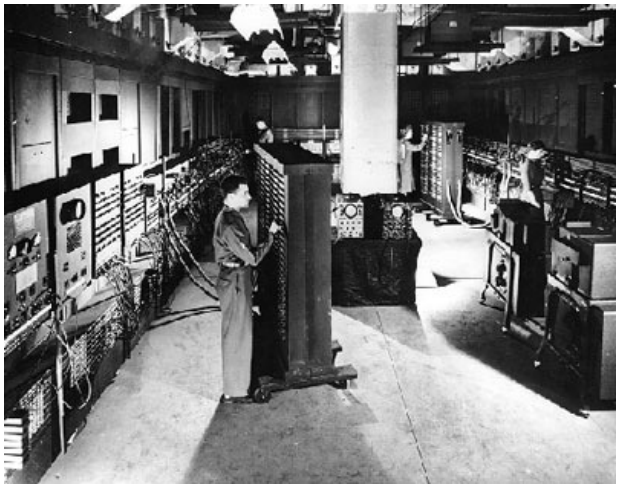
History of Supercomputers

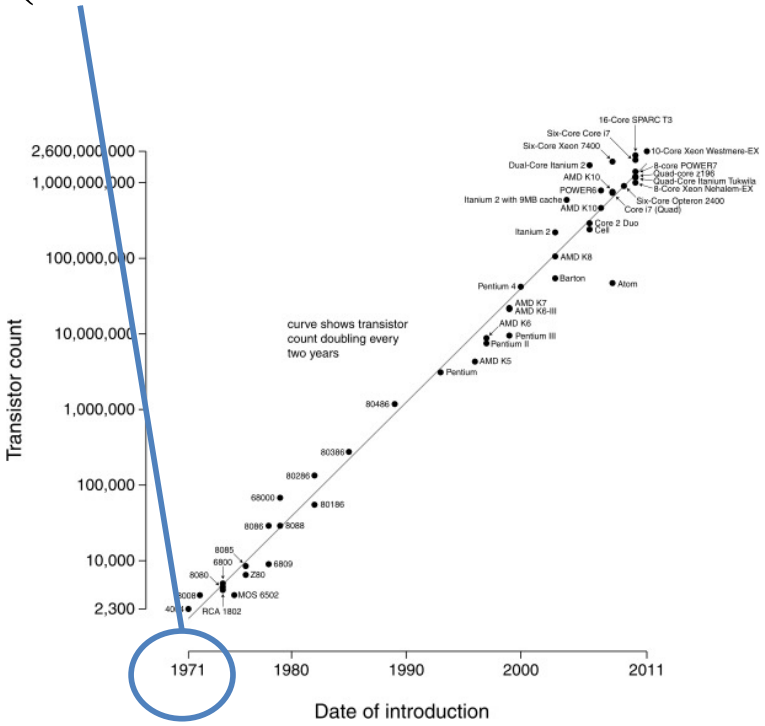
Early ’40s: ENIAC by Presper Eckert & John Mauchly at Univ. of Pennsylvania—First general-purpose electronic computer

’76: Cray 1 by Seymour Cray—beginning of vector supercomputer era

Late 80’s: massively parallel computers such as the Thinking Machines CM-2

(’71: Intel 4004—invention of microprocessor)





See lecture on MD machines

Merge of PC & Supercomputers

Rank	System	Cores	Rmax (PFlop/s)	Rpeak (PFlop/s)	Power (kW)
1	**Frontier** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,699,904	1,206.00	1,714.81	22,786
2	**Aurora** - HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max, Slingshot-11, Intel DOE/SC/Argonne National Laboratory United States	9,264,128	1,012.00	1,980.01	38,698
3	**Eagle** - Microsoft NDv5, Xeon Platinum 8480C 48C 2GHz, NVIDIA H100, NVIDIA Infiniband NDR, Microsoft Azure Microsoft Azure United States	2,073,600	561.20	846.84	
4	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	7,630,848	442.01	537.21	29,899
5	**LUMI** - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE EuroHPC/CSC Finland	2,752,704	379.70	531.51	7,107

Theoretical performance

Measured performance (in Pflop/s)

Flop/s = floating-point operations/second

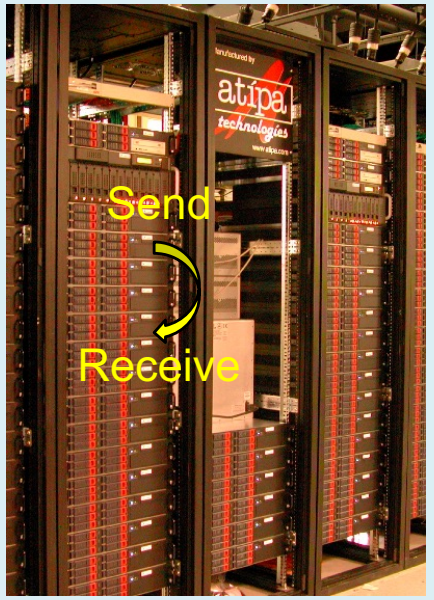
M (mega) = 10^6
G (giga) = 10^9
T (Tera) = 10^{12}
P (Peta) = 10^{15}
X (Exa) = 10^{18}

http://www.top500.org (June '24)

- **USC-CARC: 30,000 cores**
- **CACS: 4,096 cores**
- **CACS-INCITE: 4M node-hours/year on exaflop/s Aurora at Argonne Nat'l Lab**

MPI Programming

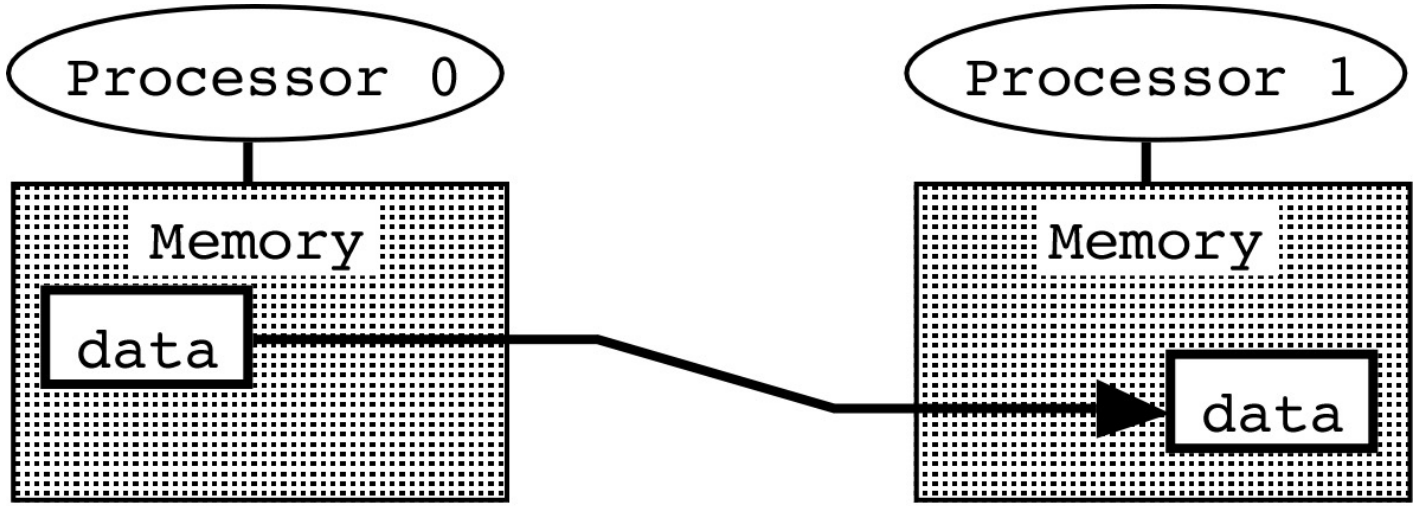
```
#include "mpi.h"
#include <stdio.h>
main(int argc, char *argv[]) {
  MPI_Status status;
  int myid;
  int n;
  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &myid);
  if (myid == 0) {
    n = 777;
    MPI_Send(&n, 1, MPI_INT, 1, 10, MPI_COMM_WORLD);
  }
  else {
    MPI_Recv(&n, 1, MPI_INT, 0, 10, MPI_COMM_WORLD, &status);
    printf("n = %d\n", n);
  }
  MPI_Finalize();
}
```



Only need two (send & receive) functions!

https://www.mcs.anl.gov/research/projects/mpi

Single Program Multiple Data (SPMD)



Process 0

```
if (myid == 0) {
  n = 777;
  MPI_Send(&n,...);
}
else {
  MPI_Recv(&n,...);
  printf(...);
}
```

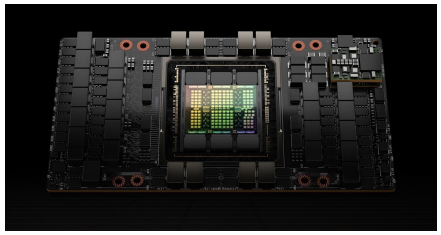
Process 1

```
if (myid == 0) {
  n = 777;
  MPI_Send(&n,...);
}
else {
  MPI_Recv(&n,...);
  printf(...);
}
```

Parallel computing: Specifies "Who does what"

Many-core CPU/GPU Computing

Intel 80-core chip
1.01 teraflop/s @62W



14,592 cores
67 teraflop/s

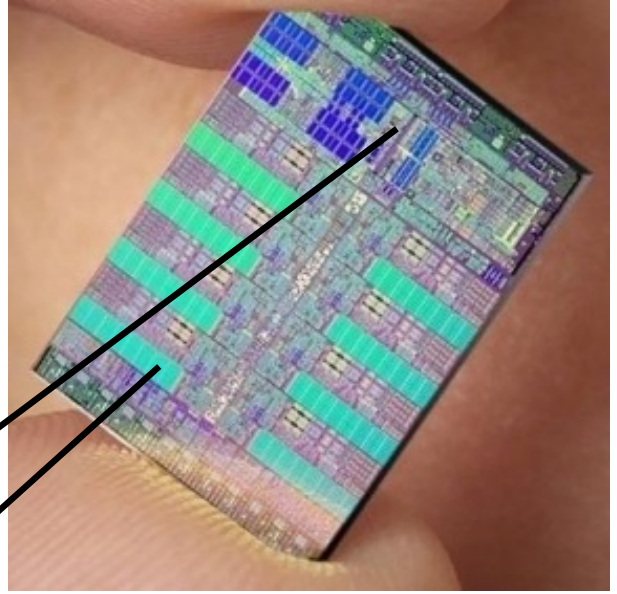
NVIDIA H100

10^2-cluster Grid

10^5-node cluster

10^3-core node

Cell Broadband Engine



64bit PowerPC

8 synergistic processing elements

Godson-T Many-core Architecture

J. Parallel Distrib. Comput. 73 (2013) 1469–1482

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

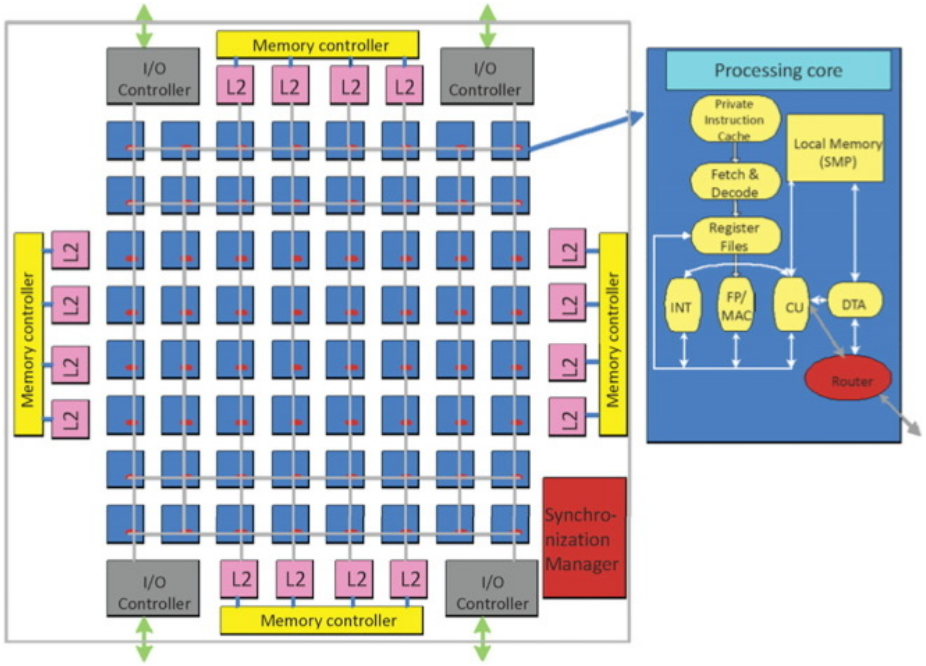
Scalability study of molecular dynamics simulation on *Godson-T* many-core architecture

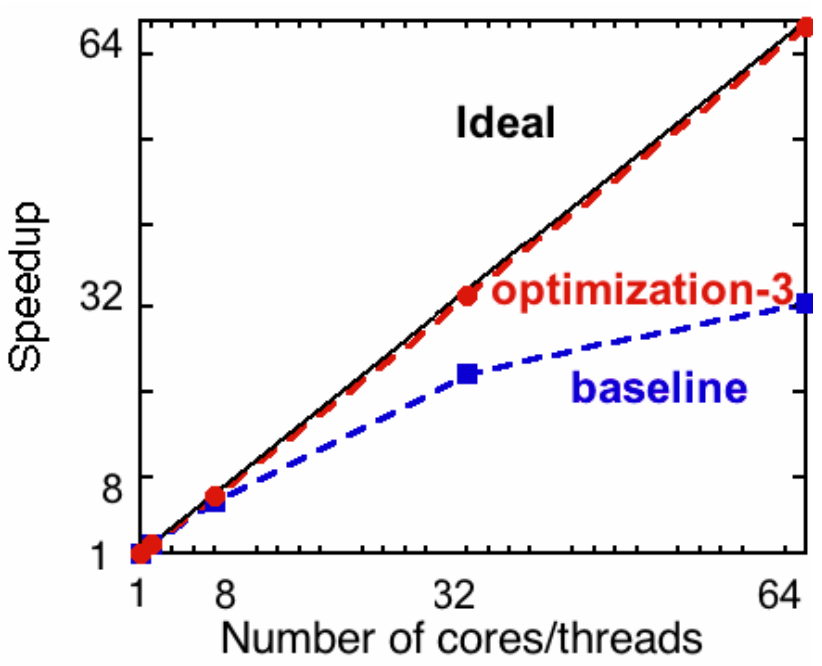
狗剩

Liu Peng [a,*], Guangming Tan [b,*], Rajiv K. Kalia [a], Aiichiro Nakano [a], Priya Vashishta [a], Dongrui Fan [b], Hao Zhang [b], Fenglong Song [b]

[a] *Collaboratory for Advanced Computing and Simulations, University of Southern California, Los Angeles, CA, 90089, USA*

[b] *Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China*

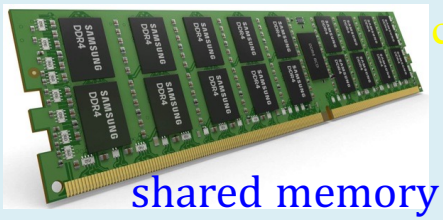


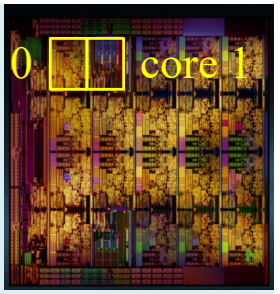


OpenMP Programming

```
#include <stdio.h>
#include <omp.h>
void main () {
  int nthreads,tid;
  nthreads = omp_get_num_threads();
  printf("Sequential section: # of threads = %d\n",nthreads);
  /* Fork multi-threads with own copies of variable */
  #pragma omp parallel private(tid)
  {
    /* Obtain & print thread id */
    tid = omp_get_thread_num();
    printf("Parallel section: Hello world from thread %d\n",tid);
    /* Only master thread does this */
    if (tid == 0) {
      nthreads = omp_get_num_threads();
      printf("Parallel section: # of threads = %d\n",nthreads);}
  } /* All created threads terminate */
}
```

parallel section



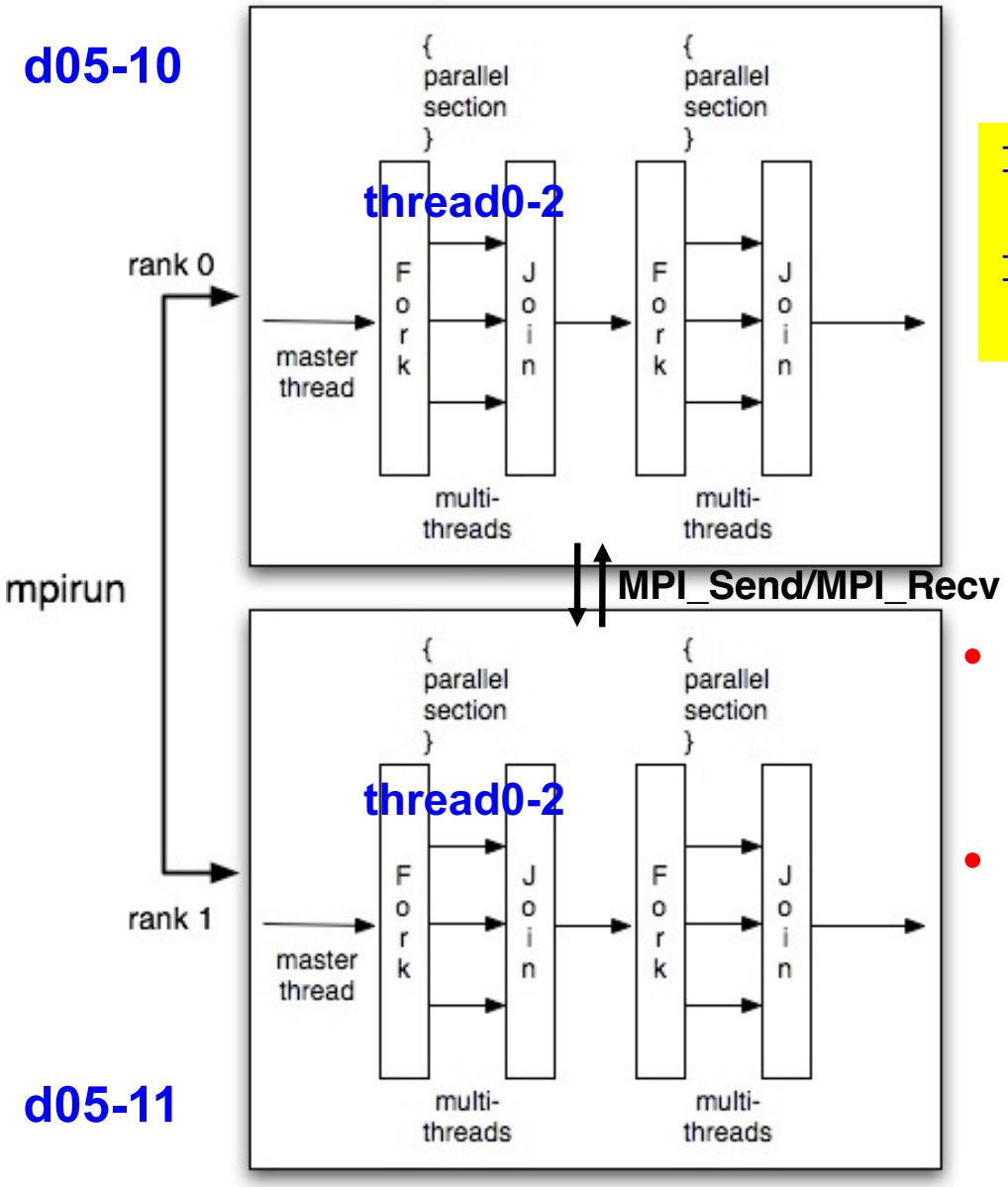


- **Obtain the number of threads & my thread ID**
- **By default, all variables are shared unless selectively changing storage attributes using private clauses**

http://www.openmp.org

Hybrid MPI+OpenMP Programming

Each MPI process spawns multiple OpenMP threads




```
In a PBS script:
  mpirun –n 2
In the code:
  omp_set_num_threads(3);
```

- MPI processes communicate by sending/receiving messages
- OpenMP threads communicate by writing to/reading from shared variables

SIMD Vectorization: MD

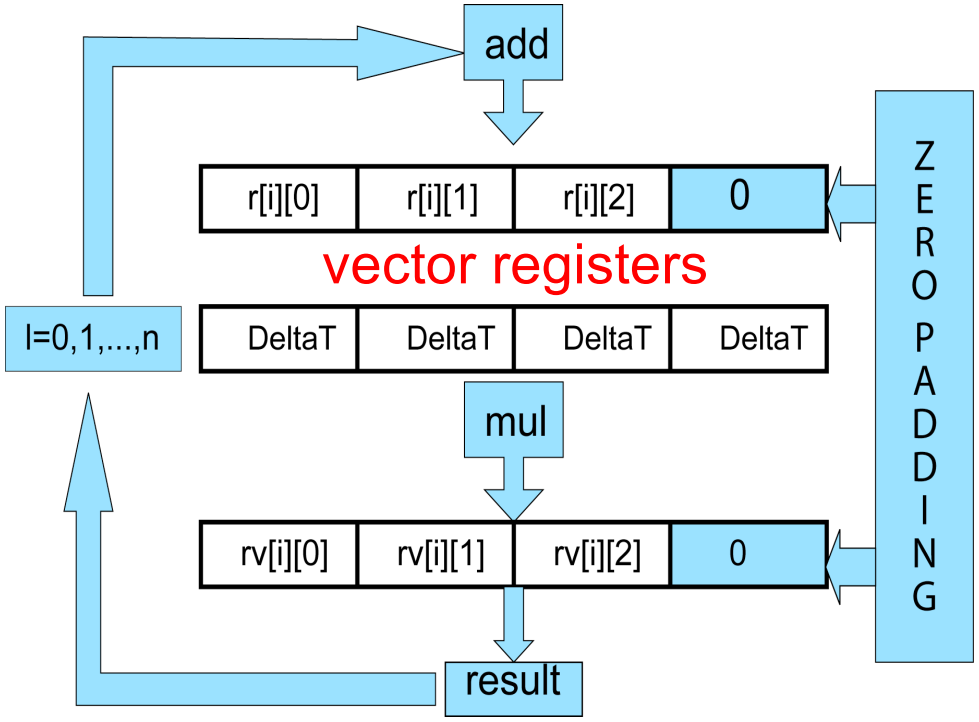
- **Single-instruction multiple-data (SIMD) parallelism using vector registers**

(Example) Zero padding to align complex data in molecular dynamics

Original solution

```
for (i=0; i<N; i++)
  for (a=0; a<3; a++)
    r[i][a] =
    r[i][a] +
    DeltaT*rv[i][a];
```

SIMD solution

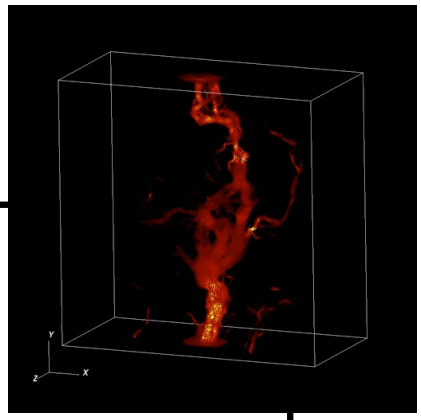


Peng *et al.*, *PDPTA 2009*; *UCHPC 2010; J. Supercomputing* **57**, 20 ('11)

SIMD Vectorization: LBM

- **Translocated statement fusion in lattice-Boltzmann flow simulation**

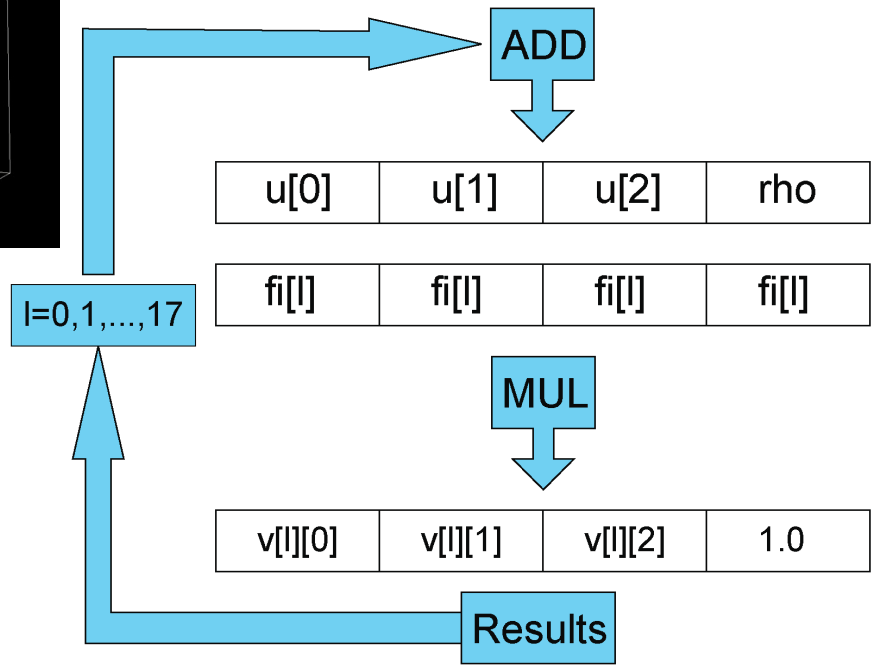
Original solution



```
for(i=0;i<3;i++){
  u[i]=0.0; rho=0.0;
  for(l=0;l<18;l++){
    fi[l] = f[18*cnz+1];
    u[i] += fi[l]*v[l][i];
    rho += fi[l];
  }
}
```

3×18×5 = 270 computation

SIMD solution



18×4 = 72 computation

Ideal Speedup 3.5

SIMDizable mathematical formulations:

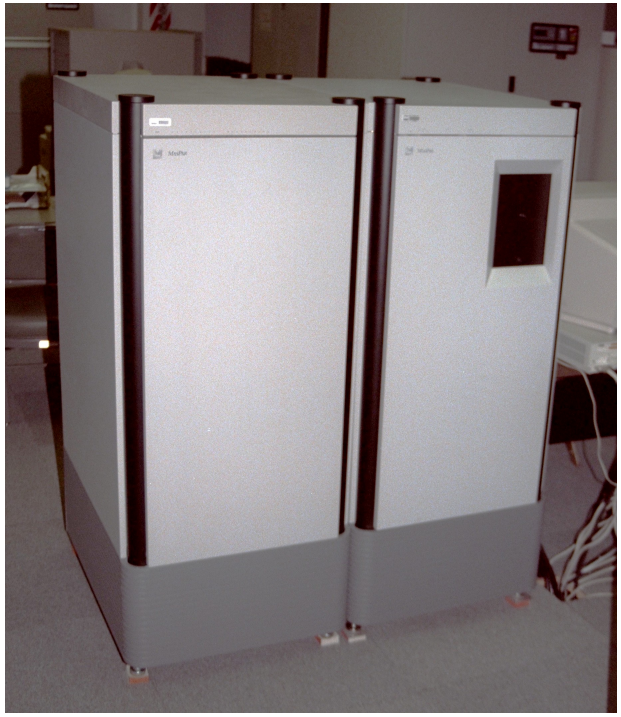
Special relativity, quaternion, *etc.*

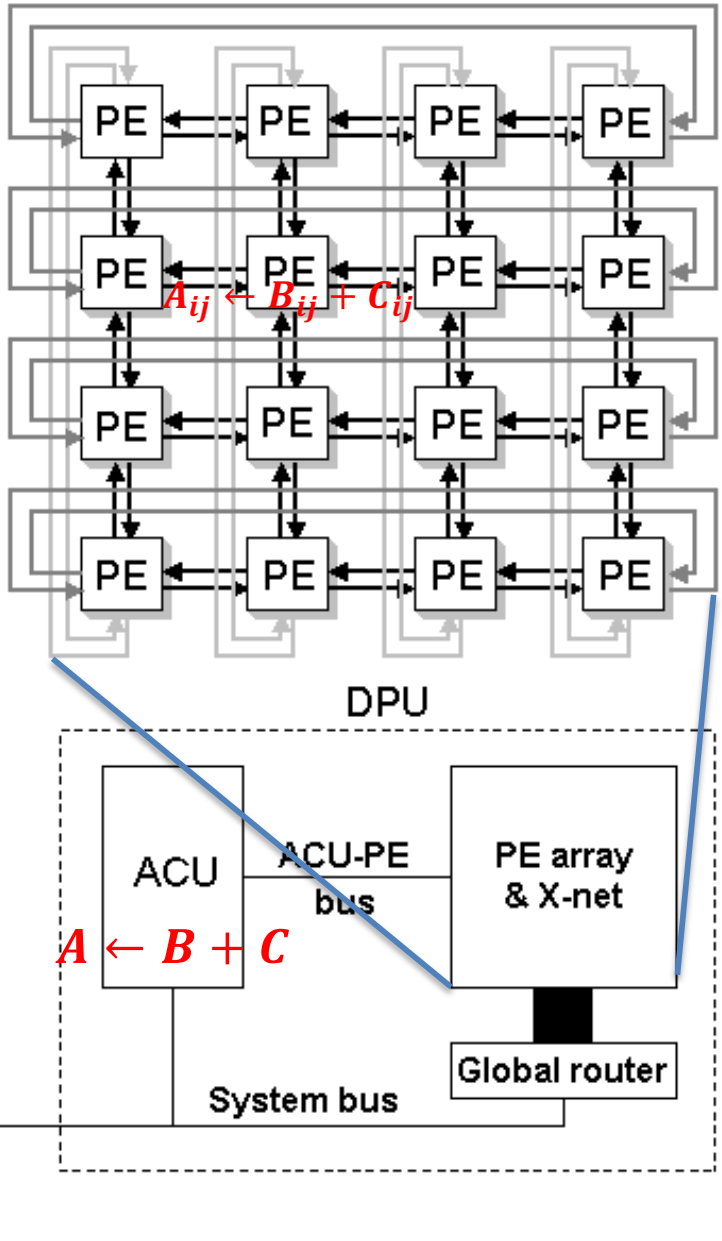
$$J^{\alpha} = (c\rho, j^1, j^2, j^3)$$

$$A^{\alpha} = (\phi/c, A^1, A^2, A^3)$$

$$\boxdot A^{\alpha} = \left(\frac{1}{c^2}\frac{\partial^2}{\partial t^2} - \nabla^2\right) A^{\alpha} = \frac{4\pi}{c} J^{\alpha}$$

Massive SIMD Data Parallelism





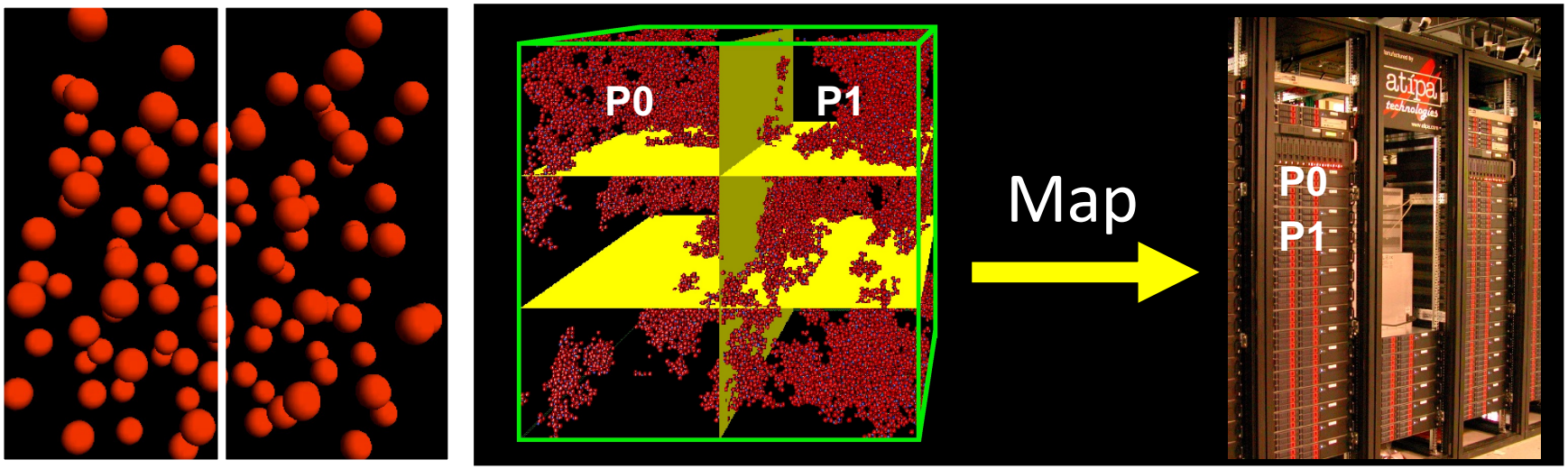
Quantum dynamics on 8,192-processor (128 × 64) MasPar 1208B

Nakano,
Comput. Phys. Commun.
83, 181 ('94)

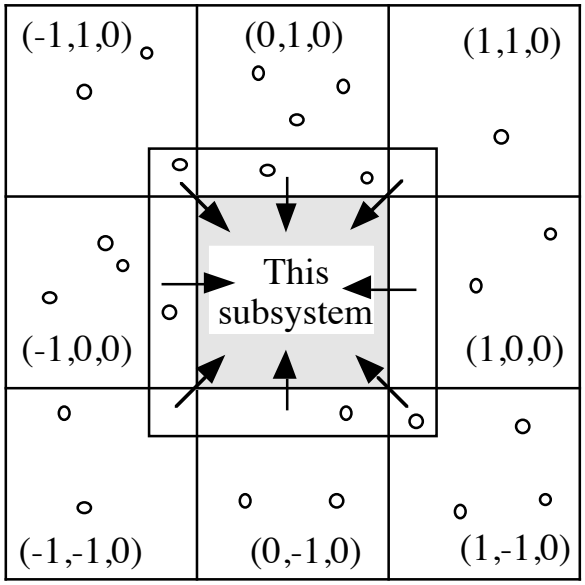
See lecture on pre-Beowulf parallel computing

Parallel Molecular Dynamics

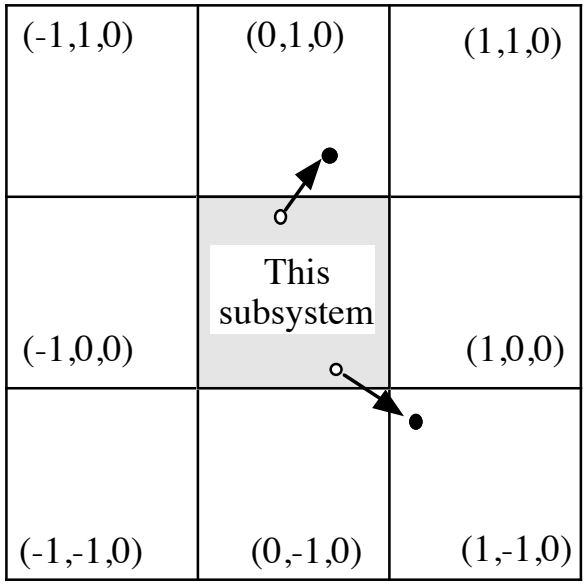
Spatial decomposition (short ranged): $O(N/P)$ computation



Atom caching: $O((N/P)^{2/3})$



Atom migration



See also parallel quantum dynamics lecture

History of Particle Simulations

'44 **John von Neumann memo on a stored-program computer: *"Our present analytical methods seem unsuitable for the solution of the important problems arising in connection with nonlinear partial differential equations. The really efficient high-speed computing devices may provide us with those heuristic hints which are needed in all parts of mathematics for genuine progress"***

'53 **First Monte Carlo simulation of liquid by Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller on MANIAC at Los Alamos Nat'l Lab**

'55 **Enrico Fermi, John Pasta, and Stanislaw Ulam studied the dynamics of an one-dimensional array of particles coupled by anharmonic springs on MANIAC**

'56 **Dynamics of hard spheres (billiards) studied by Alder and Wainwright at the Lawrence Livermore Nat'l Lab.**

'60 **Radiation damage in crystalline Cu studied with short-range repulsion and uniform attraction toward the center by George Vineyard's group at Brookhaven Nat'l Lab**

'64 **First MD simulation of liquid (864 argon atoms) using interatomic potentials by Aneesur Rahman at the Argonne Nat'l Lab on a CDC 3600**

Born-Oppenheimer Approximation

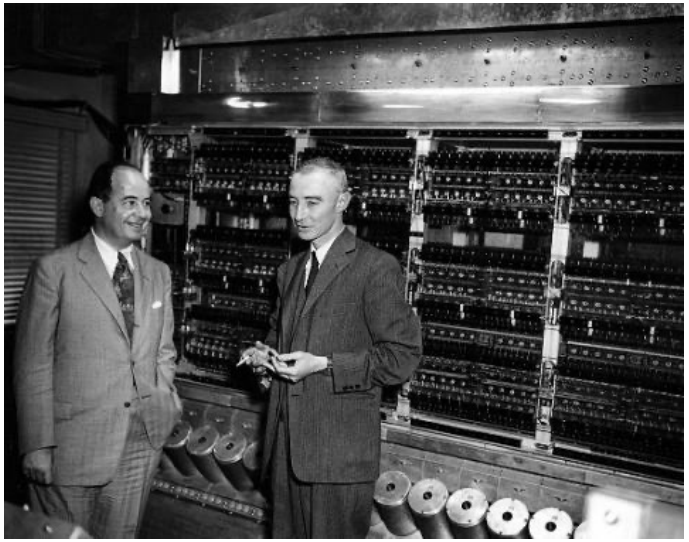
1927 № 20

ANNALEN DER PHYSIK

VIERTE FOLGE. BAND 84

1. *Zur Quantentheorie der Molekeln;*
von M. Born und R. Oppenheimer

US Supercomputers



MANIAC computer at Los Alamos

Aiichiro Nakano of the University of Southern California uses the Argonne Leadership Computing Facility to study how light can change the structures and properties of atomically thin materials to create better, cheaper semiconductors. (Image by Argonne National Laboratory.)

ASCR Discovery
ADVANCING SCIENCE THROUGH COMPUTING

Search ASCR Discovery

Subscribe

Archives

Categories

Most Recent Stories

APRIL 2024
Quanta in bulk

MARCH 2024
Aiming exascale at black holes

FEBRUARY 2024
Flying green

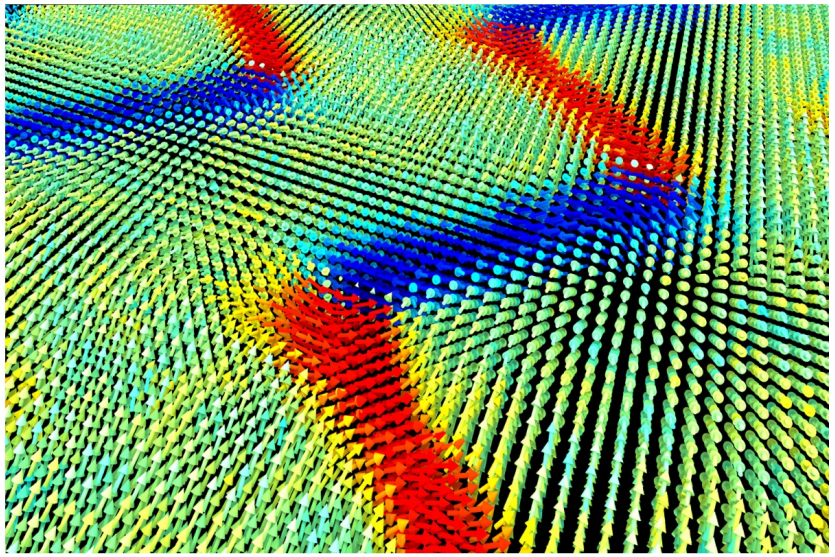
JANUARY 2024
Refining finite elements

Nanomaterials, Quantum

APRIL 2024

Quanta in bulk

A USC computer scientist wants to produce quantum materials at scale, with help from Argonne supercomputers.



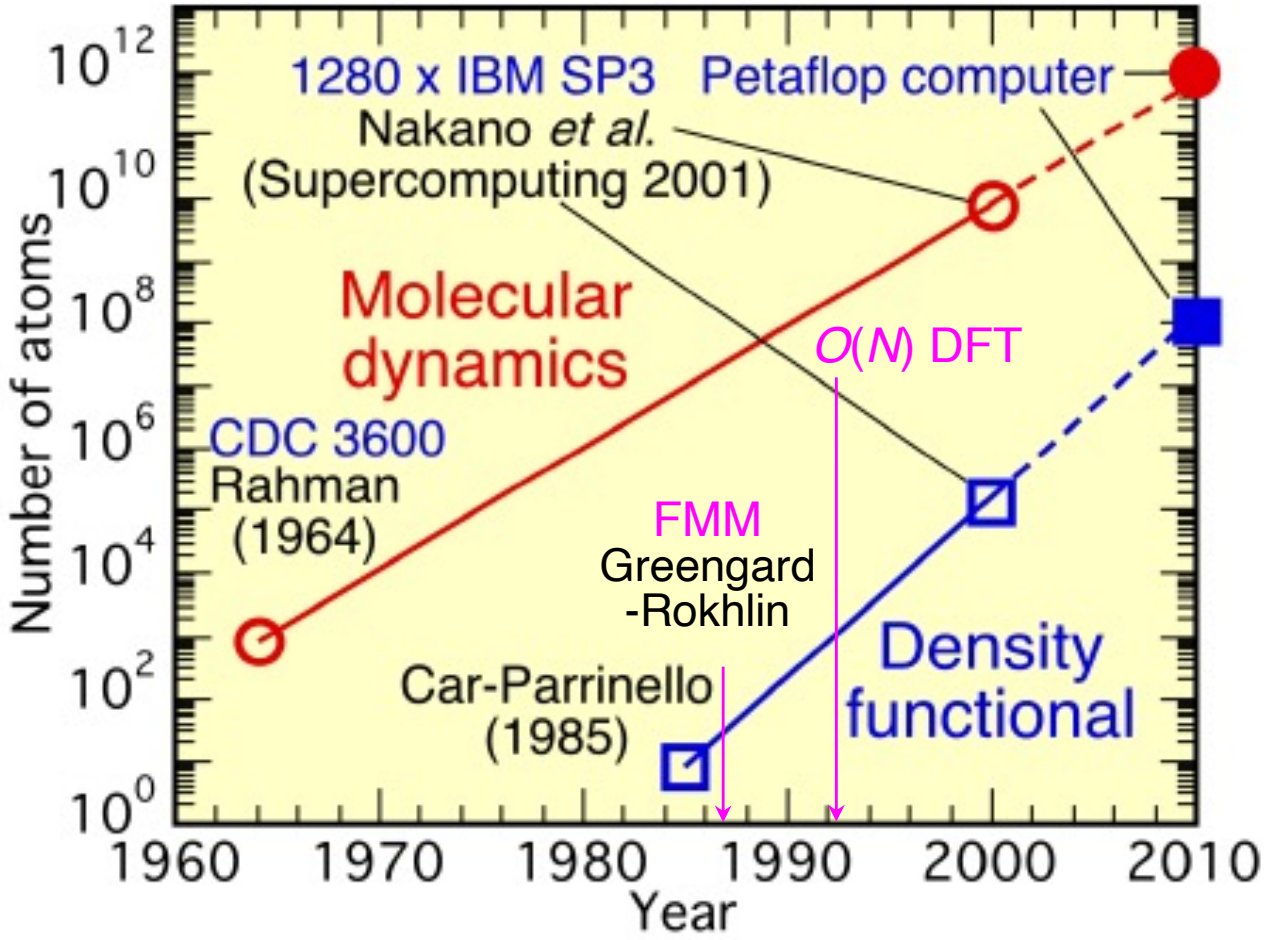
Topological structure during ferroelectric switching in PbTiO3, where arrows represent electrical polarization colored according to its magnitude. *Image courtesy of Thomas Linker and Ken-ichi Nomura/USC Viterbi School of Engineering.*

Supercomputing at Argonne National Laboratory now

Moore's Law in Scientific Computing

Number of particles in MD simulations has doubled:

- **Every 19 months in the past 50 years for classical MD**
- **Every 22 months in the past 30 years for DFT-MD**



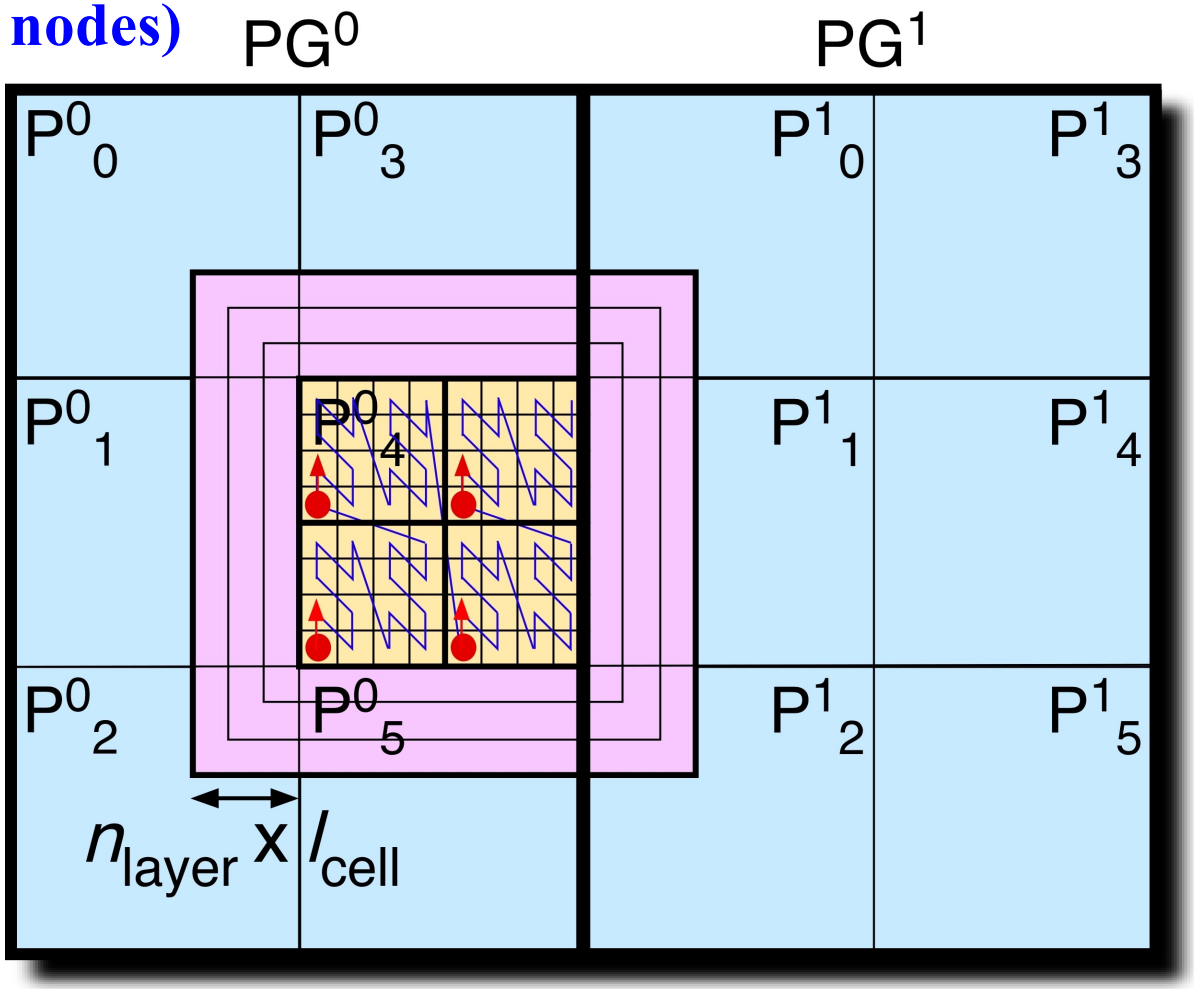
2014: 10^{12}-atom MD & 10^{8}-electron DFT on a 10 petaflop/s Blue Gene/Q with advances in algorithmic & parallel-computing techniques

Tunable Hierarchical Cellular Decomposition

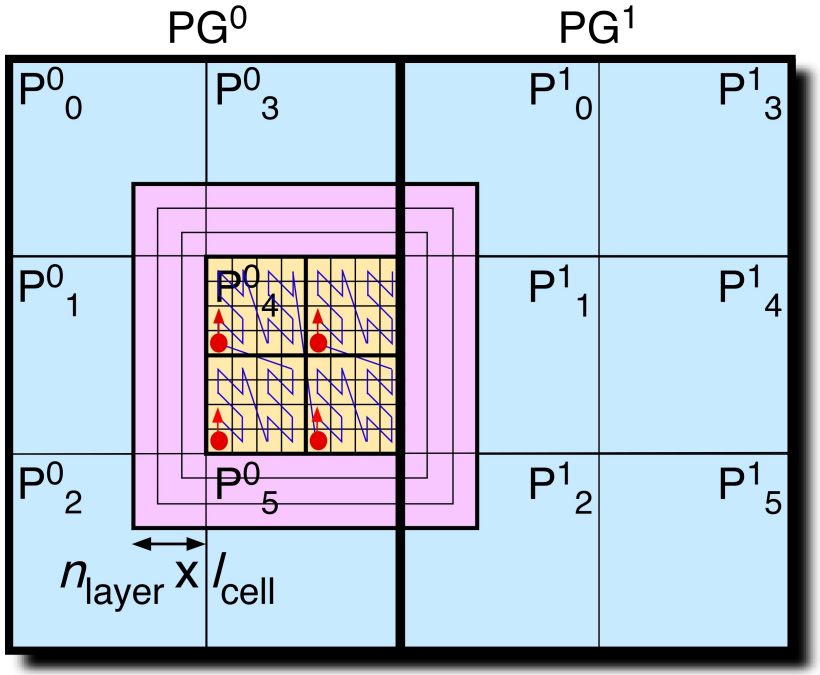
Mapping $O(N)$ divide-&-conquer algorithms onto memory hierarchies

- Spatial decomposition with data "caching" & "migration"
- Computational cells (*e.g.* linked-list cells in MD) < cell blocks (threads) < processes ($P^{\gamma}{}_{\pi}$, spatial decomposition subsystems) < process groups (P^{γ}, Grid nodes)
- Multilayer cellular decomposition (MCD) for n-tuples (n = 2–6)
- Tunable cell data & computation structures: Data/computation re-ordering & granularity parameterized at each decomposition level
- Tunable hybrid MPI + OpenMP + SIMD implementation

Nomura *et al.*, *IPDPS 2009*

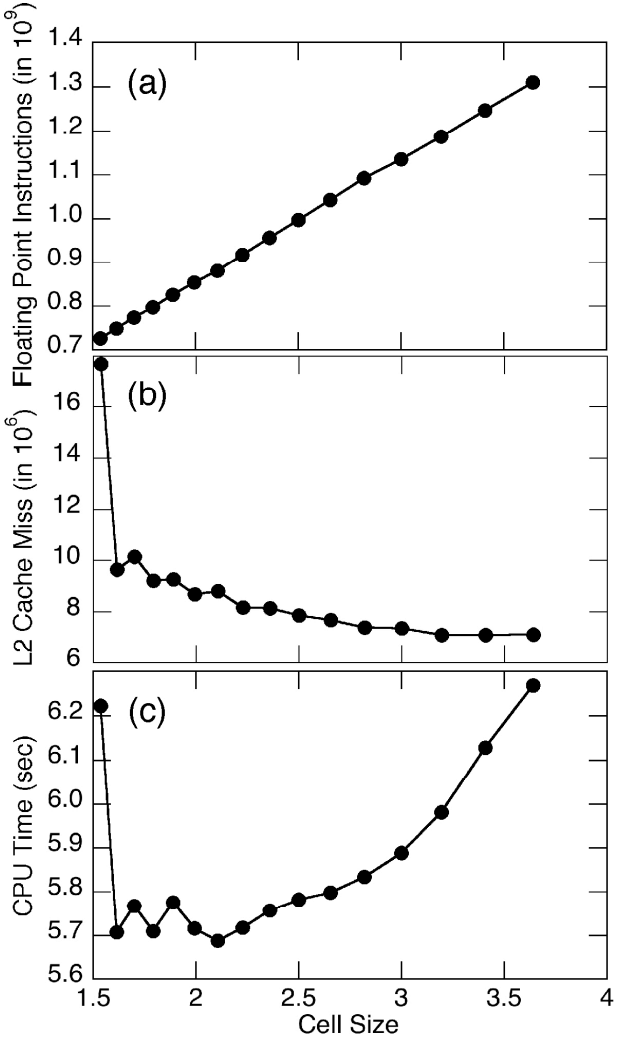


Performance Tunability



Floating-point operation/L2 cache miss trade-off:

331,776-atom silica MRMD on 1.4GHz Pentium III



MPI/OpenMP parallelism trade-off:

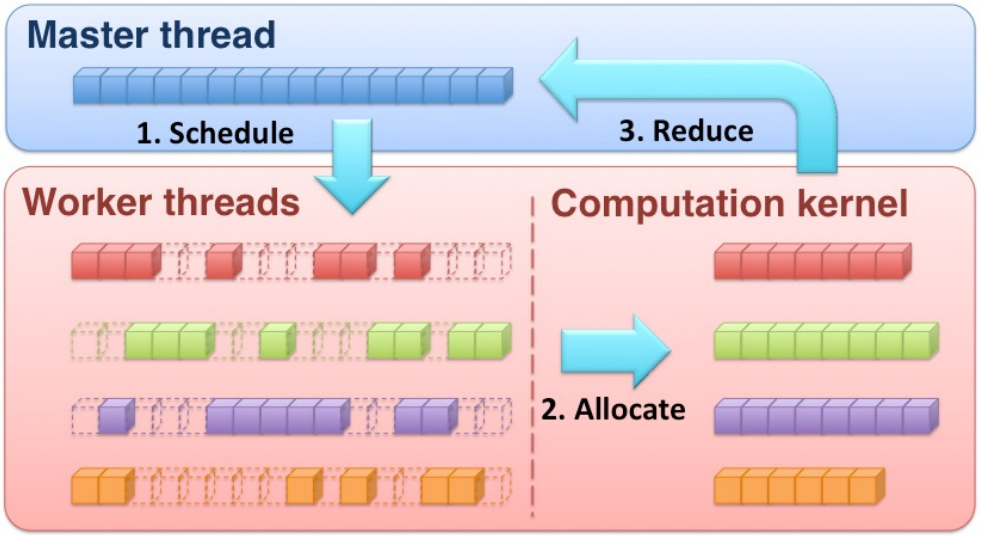
8,232,000-atom silica MRMD & 290,304-atom RDX F-ReaxFF on 8-way 1.5 GHz Power4

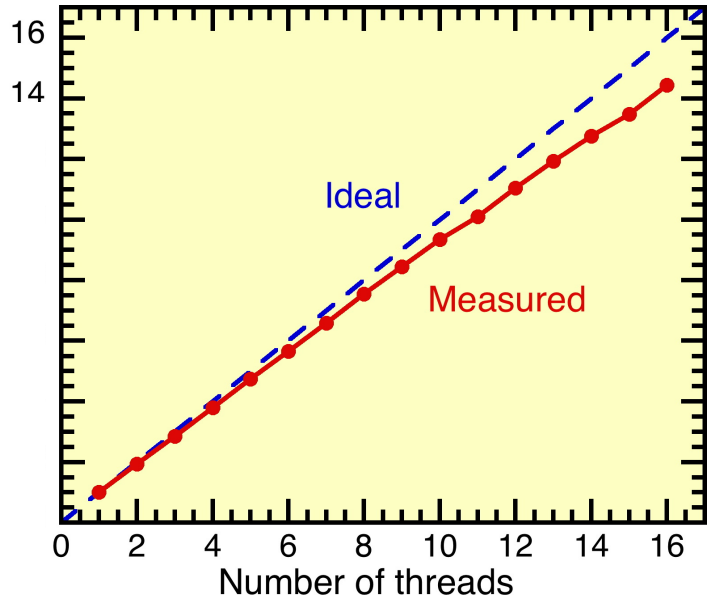
Number of OpenMP threads, n_{td}	Number of MPI processes, n_p	Execution time/MD time step (sec)	
		MRMD	P-ReaxFF
1	8	4.19	62.5
2	4	5.75	58.9
4	2	8.60	54.9
8	1	12.5	120

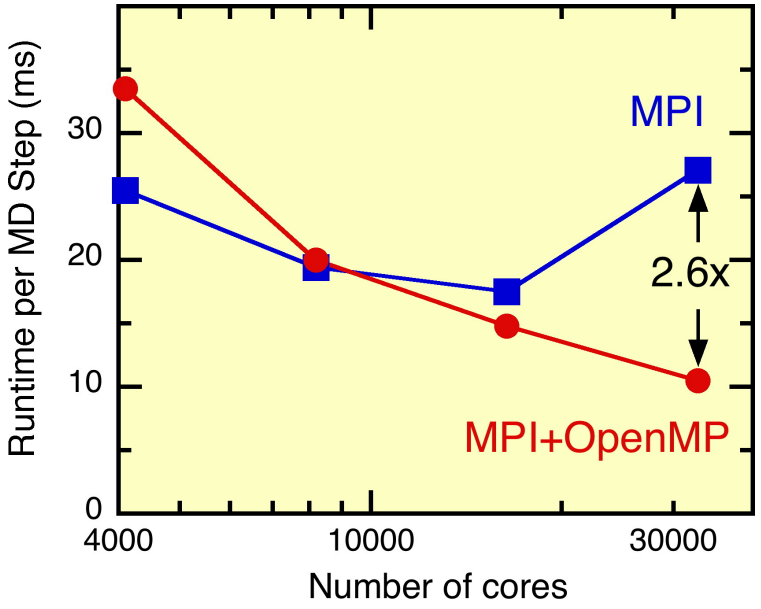
Spatially Compact Thread Scheduling

Concurrency-control mechanism: Data privatization

- **Reduced memory:** # of atoms / # of threads
  $\Theta(nq) \rightarrow \Theta(n+n^{2/3}q^{1/3})$
- **Strong scaling parallel efficiency 0.9 on quad quad-core AMD Opteron**
- **2.6× speedup over MPI by hybrid MPI+OpenMP on 32,768 IBM Blue Gene/P cores**







M. Kunaseth *et al.*, *PDPTA'11*; *J. Supercomput.* ('13)

Concurrency-Control Mechanisms

A number of concurrency-control mechanisms (CCMs) are provided by OpenMP to coordinate multiple threads:

- **Critical section: Serialization**
- **Atomic update: Expensive hardware instruction**
- **Data privatization: Requires large memory $\Theta(nq)$**
- **Hardware transactional memory: Rollbacks (on IBM Blue Gene/Q)**

# of atoms per node

# of threads

CCM performance varies:

- **Depending on computational characteristics of each program**
- **In many cases, CCM degrades performance significantly**

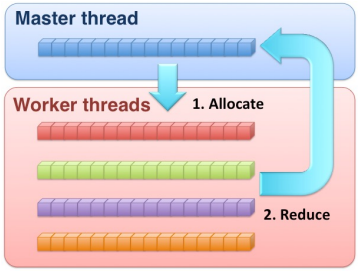
HTM/critical section

```
#pragma omp <critical|tm_atomic>
{
  ra[i][0] += fa*dr[0];
  ra[i][1] += fa*dr[1];
  ra[i][2] += fa*dr[2];
}
```

Atomic update

```
#pragma omp atomic
ra[i][0] += fa*dr[0];
#pragma omp atomic
ra[i][1] += fa*dr[1];
#pragma omp atomic
ra[i][2] += fa*dr[2];
```

Data privatization



Goal: Provide a guideline to choose the "right" CCM

Hardware Transactional Memory

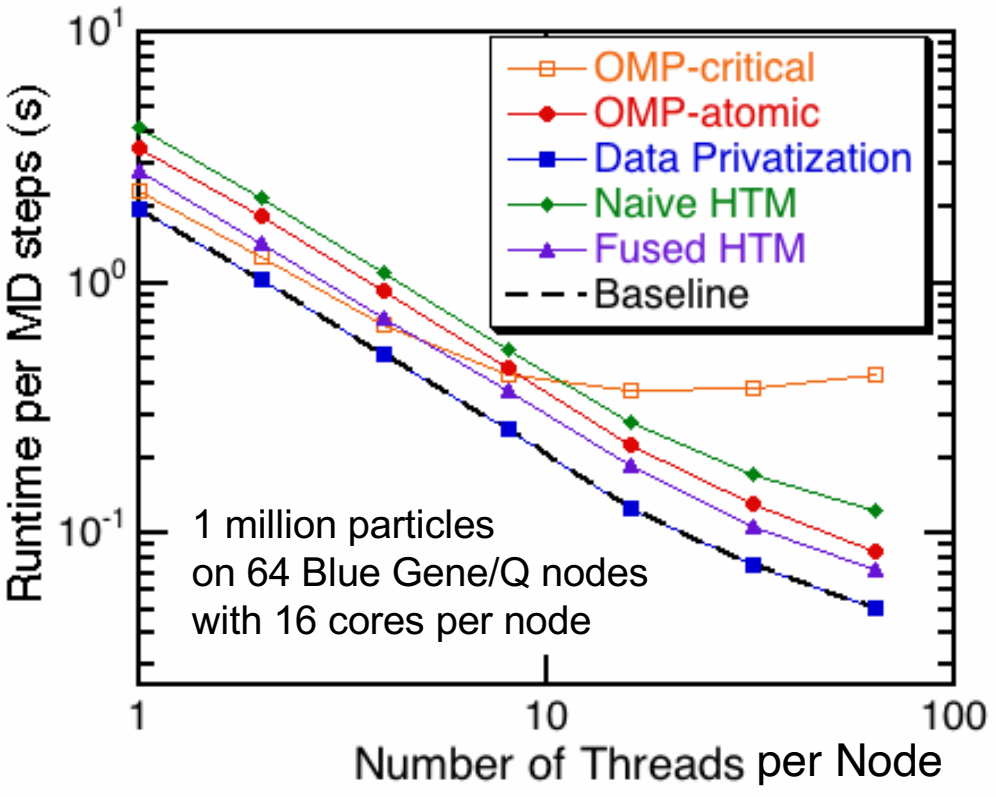
Transactional memory (TM): An opportunistic CCM

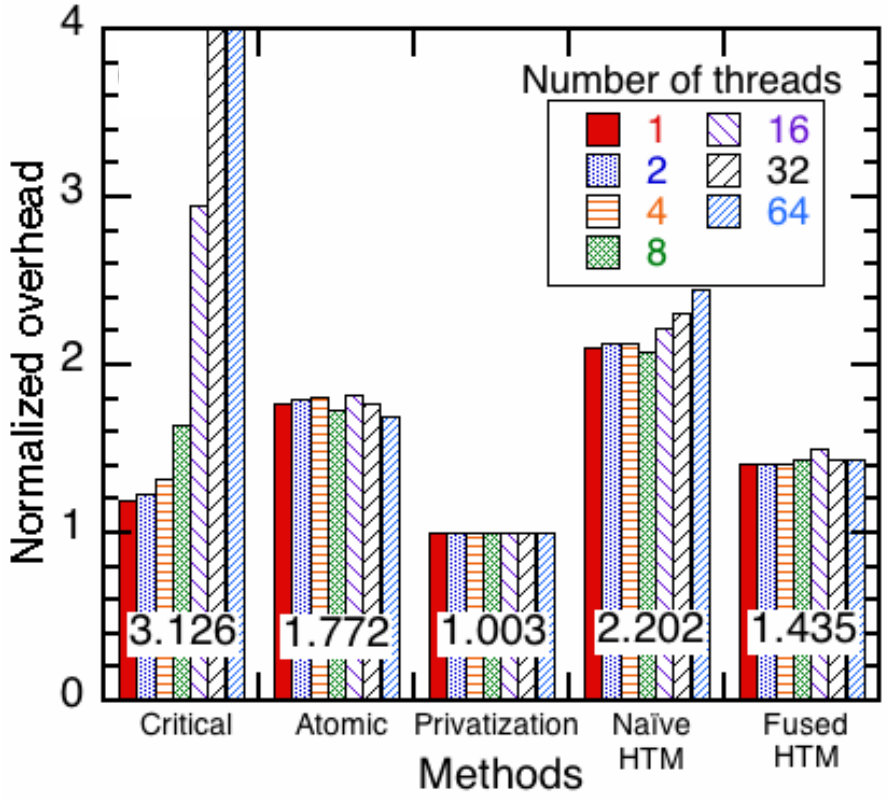
- **Avoids memory conflicts by monitoring a set of speculative operations (*i.e.* transaction)**
- **If two or more transactions write to the same memory address, transaction(s) will be restarted—a process called rollback**
- **If no conflict detected in the end of a transaction, operations within the transaction becomes permanent (*i.e.* committed)**
- **Software TM usually suffers from large overhead**

Hardware TM on IBM Blue Gene/Q:

- **The first commercial platform implementing TM support at hardware level *via* multiversioned L2-cache**
- **Hardware support is expected to reduce TM overhead**
- **Performance of HTM on molecular dynamics has not been quantified**

Strong-Scaling Benchmark for MD





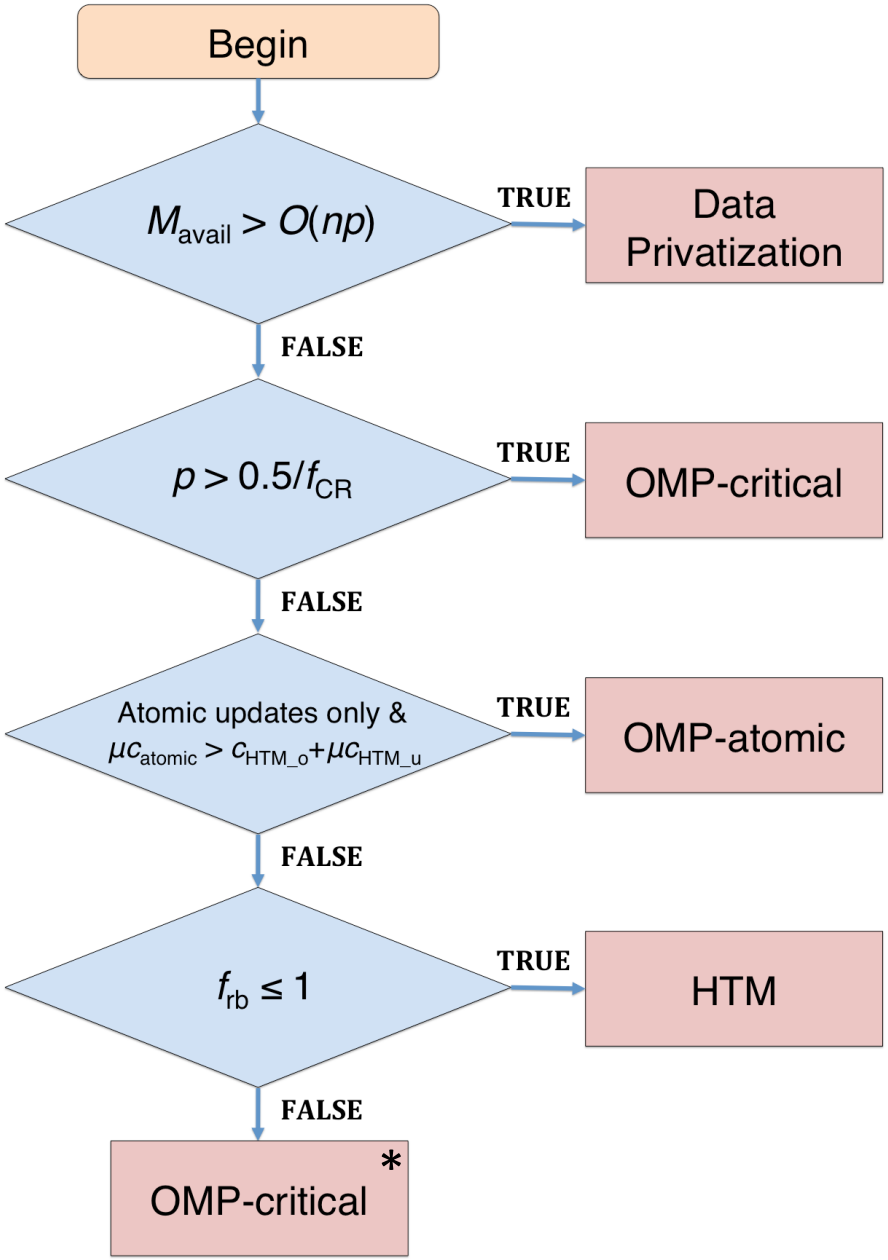
*Baseline: No CCM; the result is wrong

Developed a fundamental understanding of CCMs:

- **OMP-critical has limited scalability on larger number of threads ($q > 8$)**
- **Data privatization is the fastest, but it requires $\Theta(nq)$ memory**
- **Fused HTM performs the best among constant-memory CCMs**

M. Kunaseth *et al.*, *PDSEC'13*

Threading Guideline for Scientific Programs



M. Kunaseth *et al.*, *PDSEC'13*

Focus on minimizing runtime (best performance):

- **Have enough memory → data privatization**
- **Conflict region is small → OMP-critical**
- **Small amount of updates → OMP-atomic**
- **Conflict rate is low → HTM**
- **Other → OMP-critical* (poor performance)**

Concurrency control mechanism	Parallel efficiency
OMP-critical	$e = \min(\frac{1}{pf_{CR}}, 1)$
OMP-atomic	$e = \frac{t_{total}}{t_{total} + m\mu c_{atomic}}$
Data privatization	$e = \frac{t_{total}}{t_{total} + c_{reduction} n \log p}$
HTM	$e = \frac{t_{total}}{t_{total} + m(c_{HTM\_overhead} + \mu c_{HTM\_update})}$

IEEE PDSEC Best Paper & Beyond

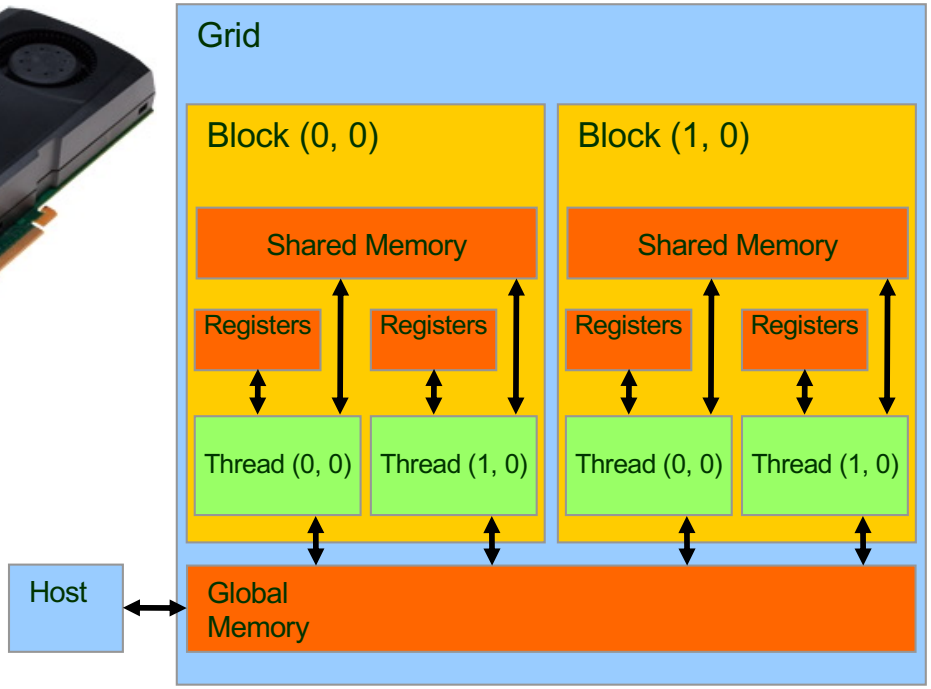
GPU Programming: CUDA

- **Compute Unified Device Architecture**
- **Integrated host (CPU) + device (GPU) application programming interface based on C language developed at NVIDIA**
- **CUDA homepage**
  `http://www.nvidia.com/object/cuda_home_new.html`
- **Compilation**

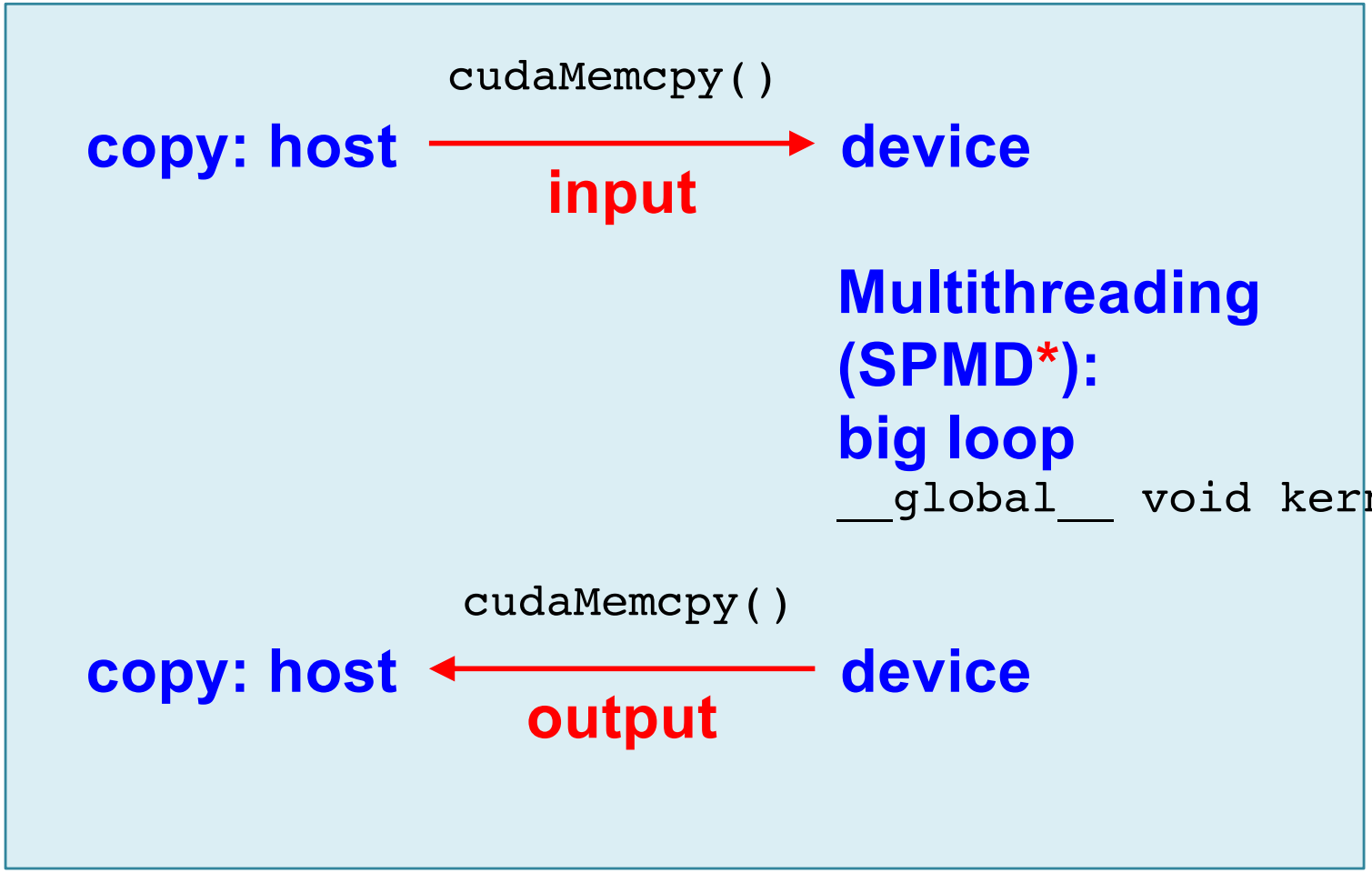
```
$ nvcc pi.cu
```

- **Execution**

```
$ a.out
PI = 3.141593
```



Summary: CUDA Computing

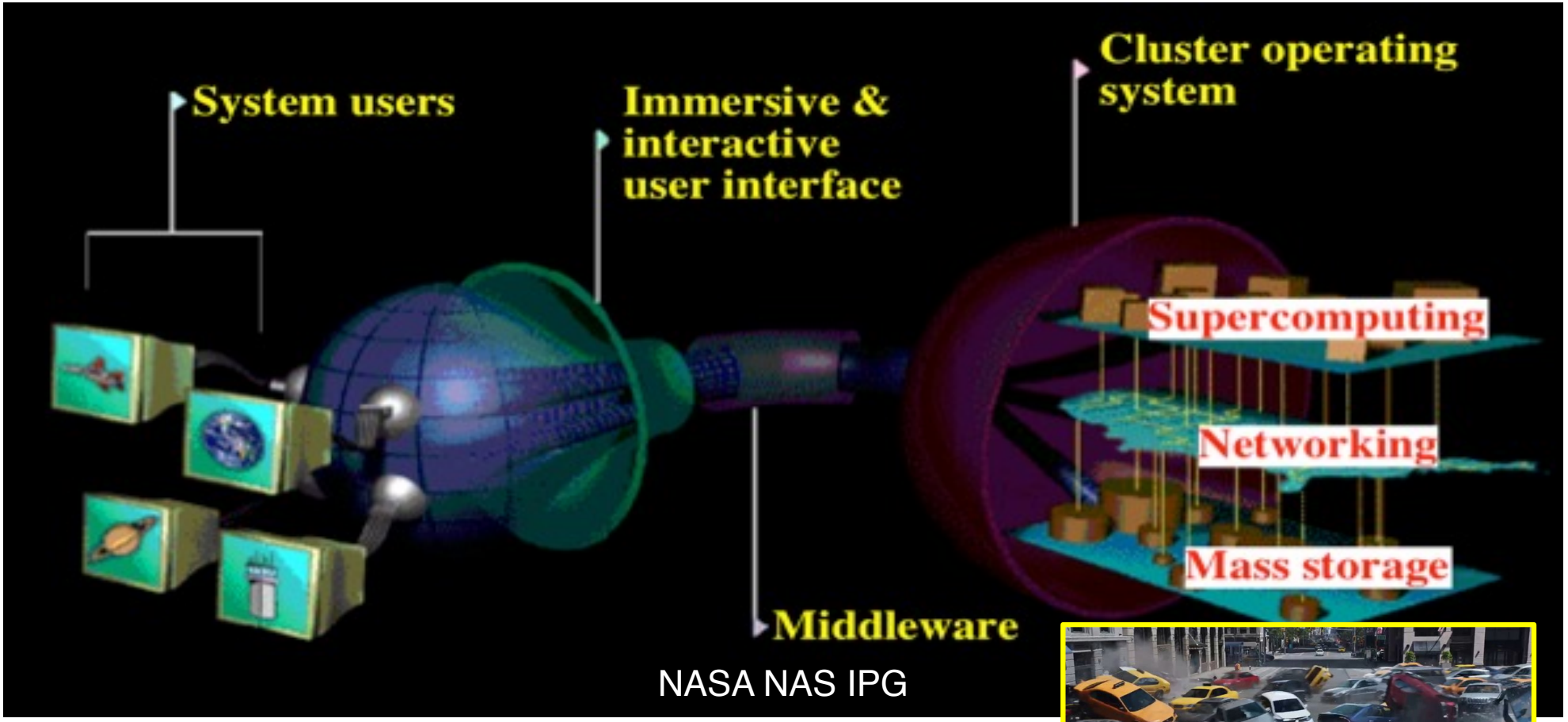


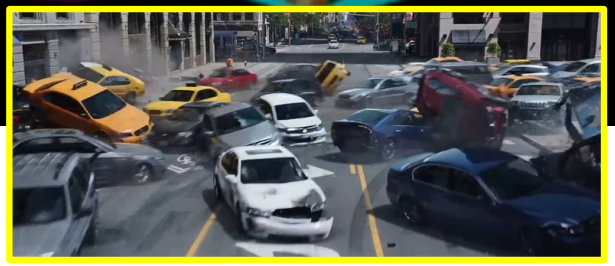
* Single program multiple data we have learned is called single instruction multiple threads (SIMT) in GPU terminology

See https://aiichironakano.github.io/cs596/CUDA.pdf

Grid Computing

- **World Wide Web:** Universal interface to digital library on the Internet
- **Information Grid:** Pervasive (from any place in the world at any time) access to everything (computing, mass storage, experimental equipments, distributed sensors, *etc*., on high-speed networks)

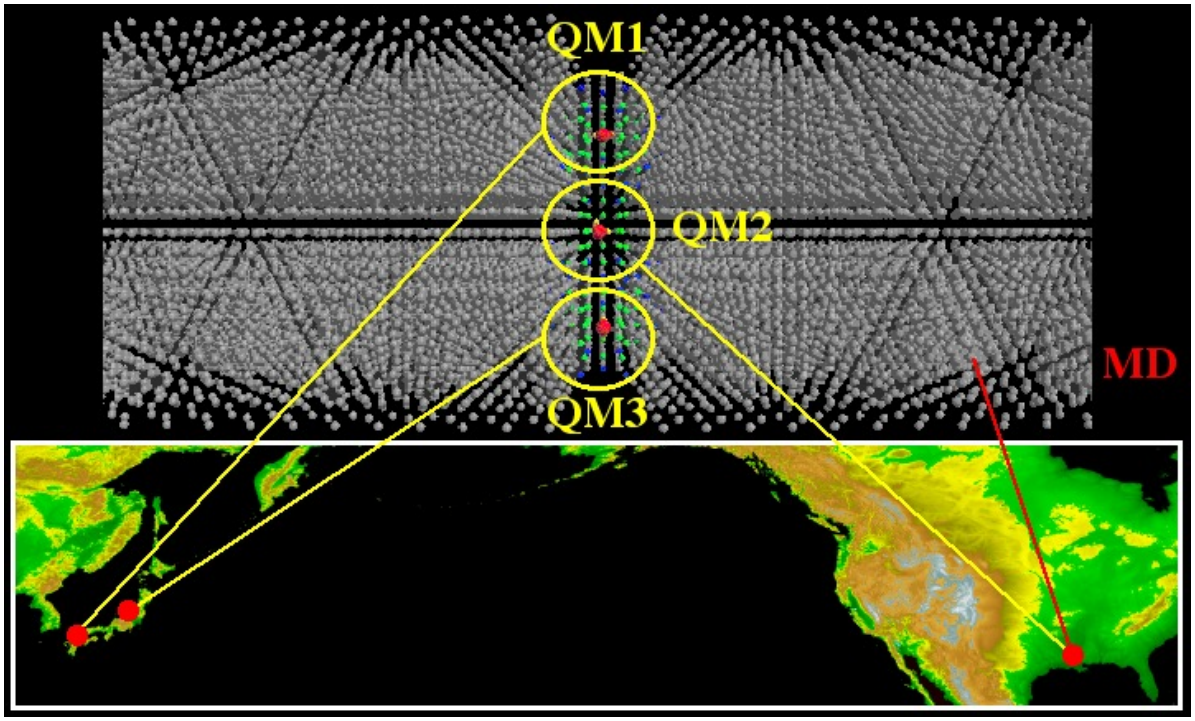


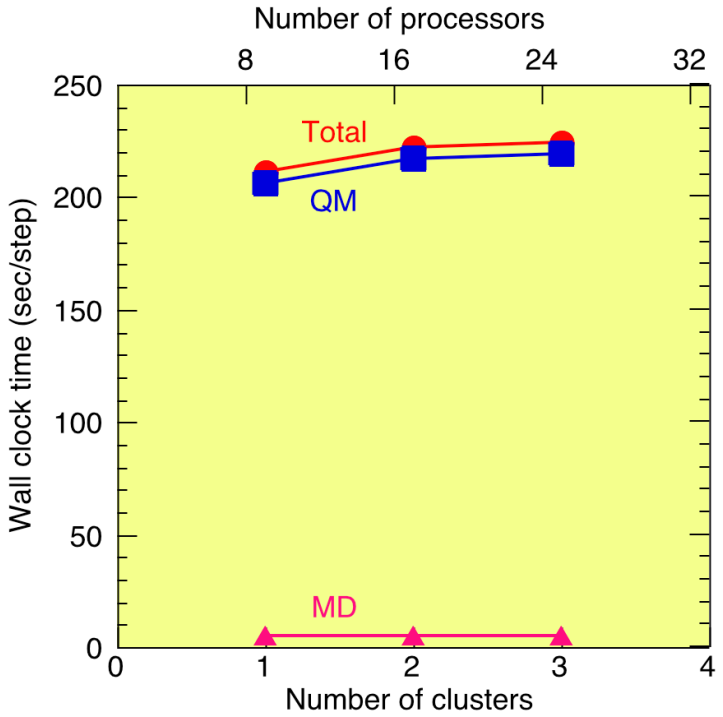


Global Collaborative Simulation

Multiscale MD/QM simulation on a Grid of distributed PC clusters in the US & Japan

- Task decomposition (MPI Communicator) + spatial decomposition
- MPICH-G2/Globus





Japan: Yamaguchi—65 P4 2.0GHz
Hiroshima, Okayama, Niigata—3×24 P4 1.8GHz
US: Louisiana—17 Athlon XP 1900+

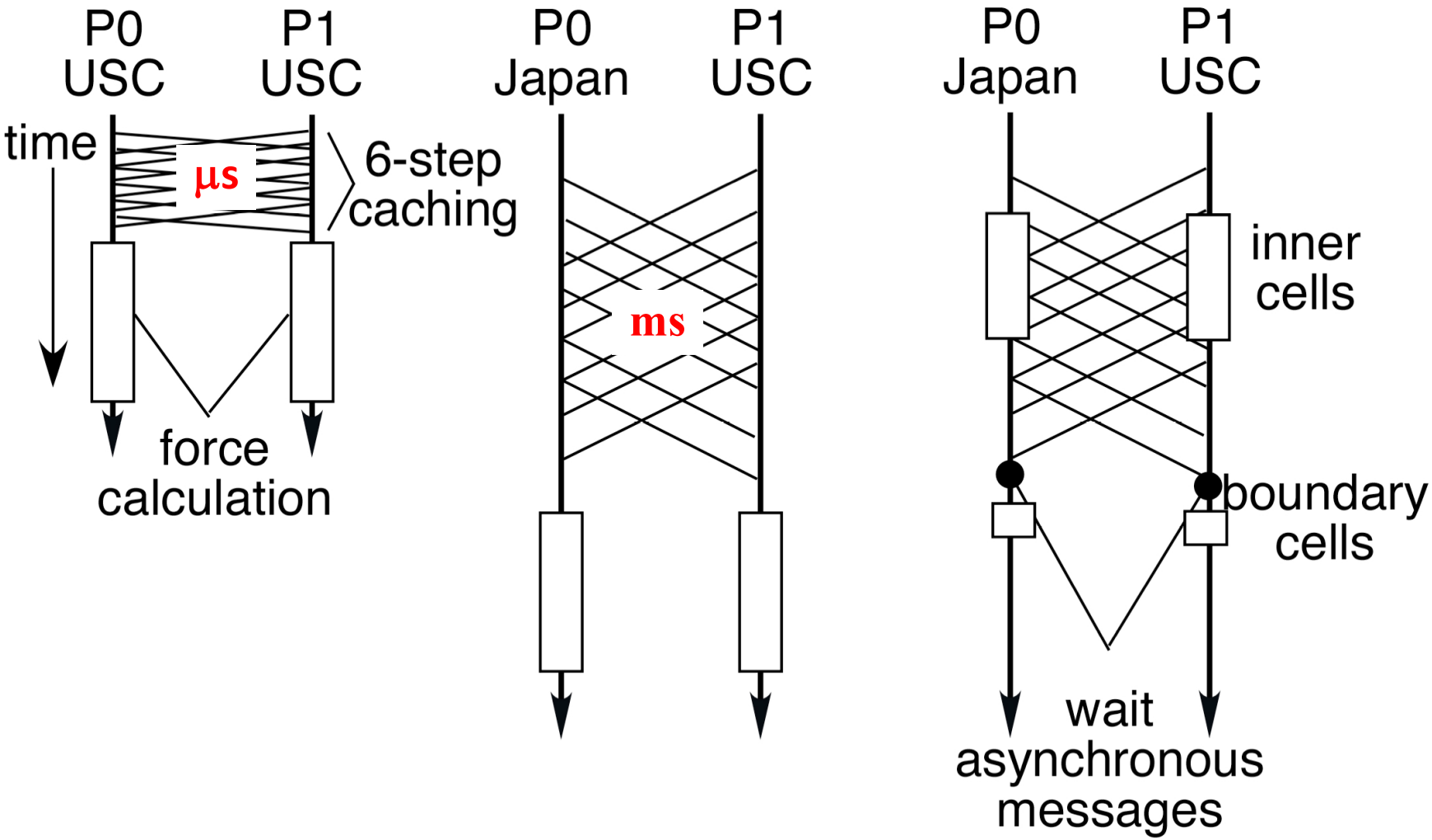
MD — 91,256 atoms
QM (DFT) — 76*n* atoms on *n* clusters

- Scaled speedup, $P = 1$ (for MD) + $8n$ (for QM)
- Efficiency = 94.0% on 25 processors over 3 PC clusters

Kikuchi *et al.*
IEEE/ACM SC02

Internode Optimization

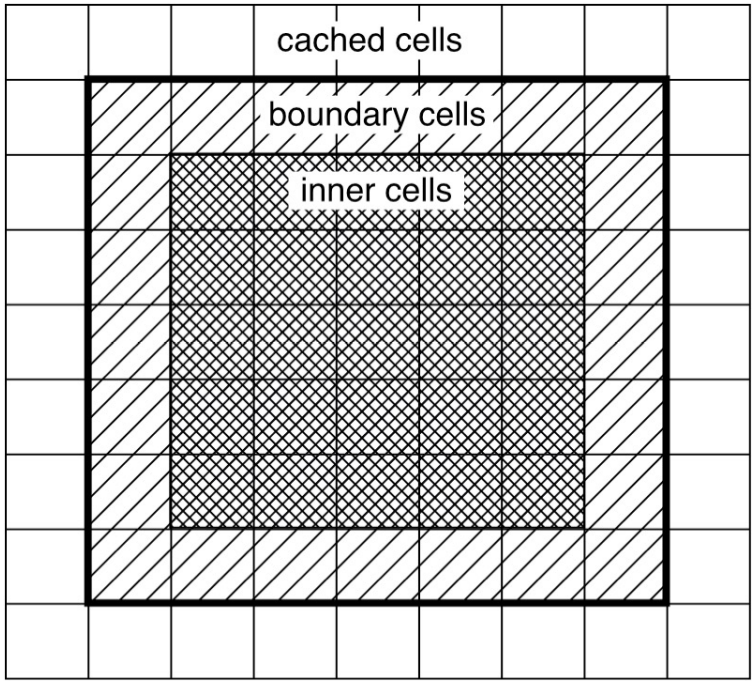
- **Communication bottleneck in metacomputing on a Grid**



Grid-Enabled MD Algorithm

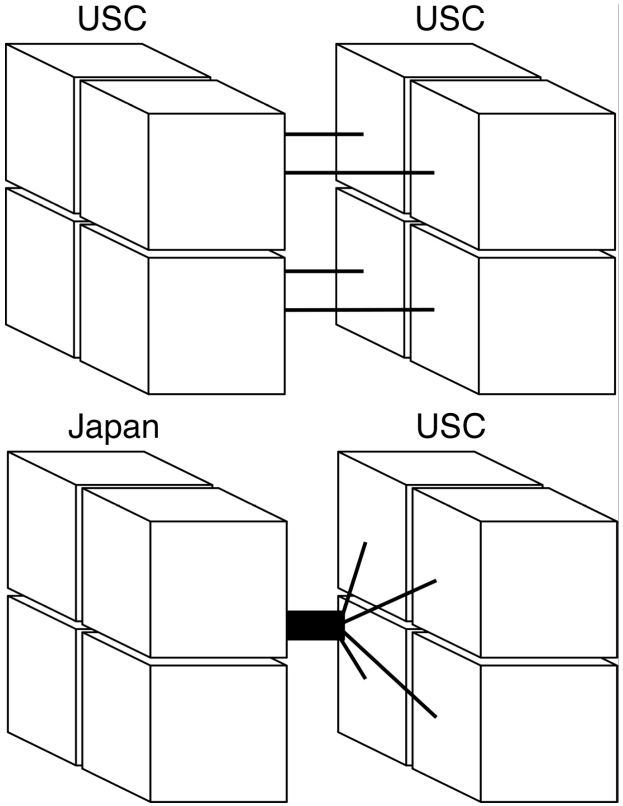
Grid MD algorithm:

1. asynchronous receive of cells to be cached **MPI_Irecv()**
2. send atomic coordinates in the boundary cells
3. compute forces for atoms in the inner cells
4. wait for the completion of the asynchronous receive **MPI_Wait()**
5. compute forces for atoms in the boundary cells



Renormalized Messages:

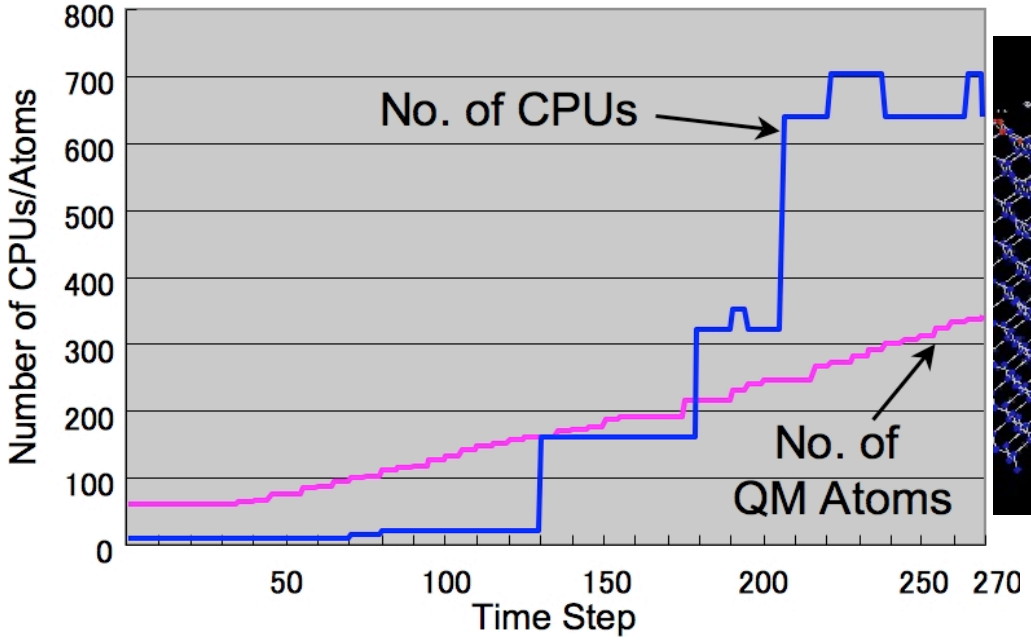
Latency can be reduced by composing a large cross-site message instead of sending all processor-to-processor messages

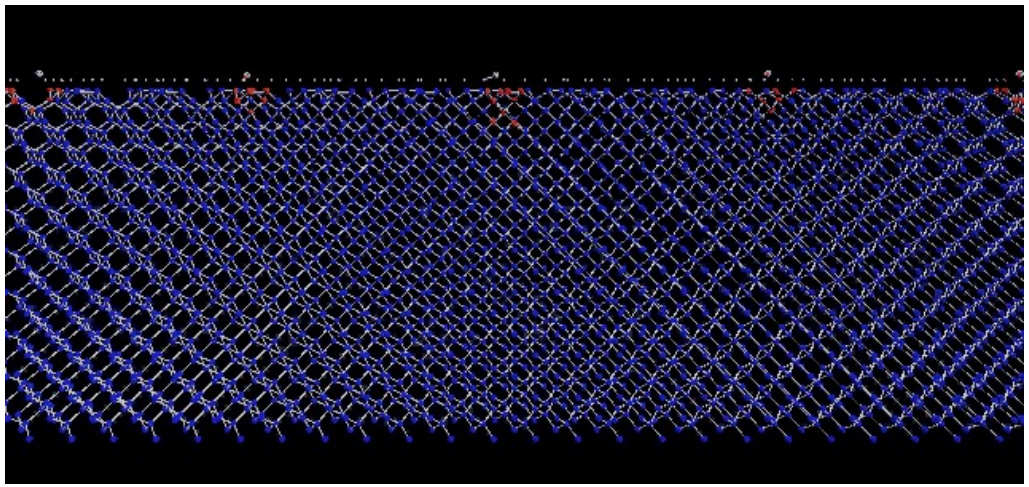


Sustainable Grid Supercomputing

- **Sustained (> months) supercomputing (> 10^3 CPUs) on a Grid of geographically distributed supercomputers**
- **Hybrid Grid remote procedure call (GridRPC) + message passing (MPI) programming**
- **Dynamic allocation of computing resources on demand & automated migration due to reservation schedule & faults**

Ninf-G GridRPC: ninf.apgrid.org; MPICH: www.mcs.anl.gov/mpi





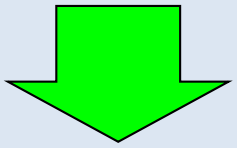
Takemiya *et al.*, *IEEE/ACM SC06*
Song *et al.*, *IJCS* ('09)

Multiscale QM/MD simulation of high-energy beam oxidation of Si

Grid Remote Procedure Call (RPC)

- **Simple RPC API (application program interface)**
- **Existing libraries & applications into Grid applications**
- **IDL (interface definition language) embodying call information, with minimal client-side management**

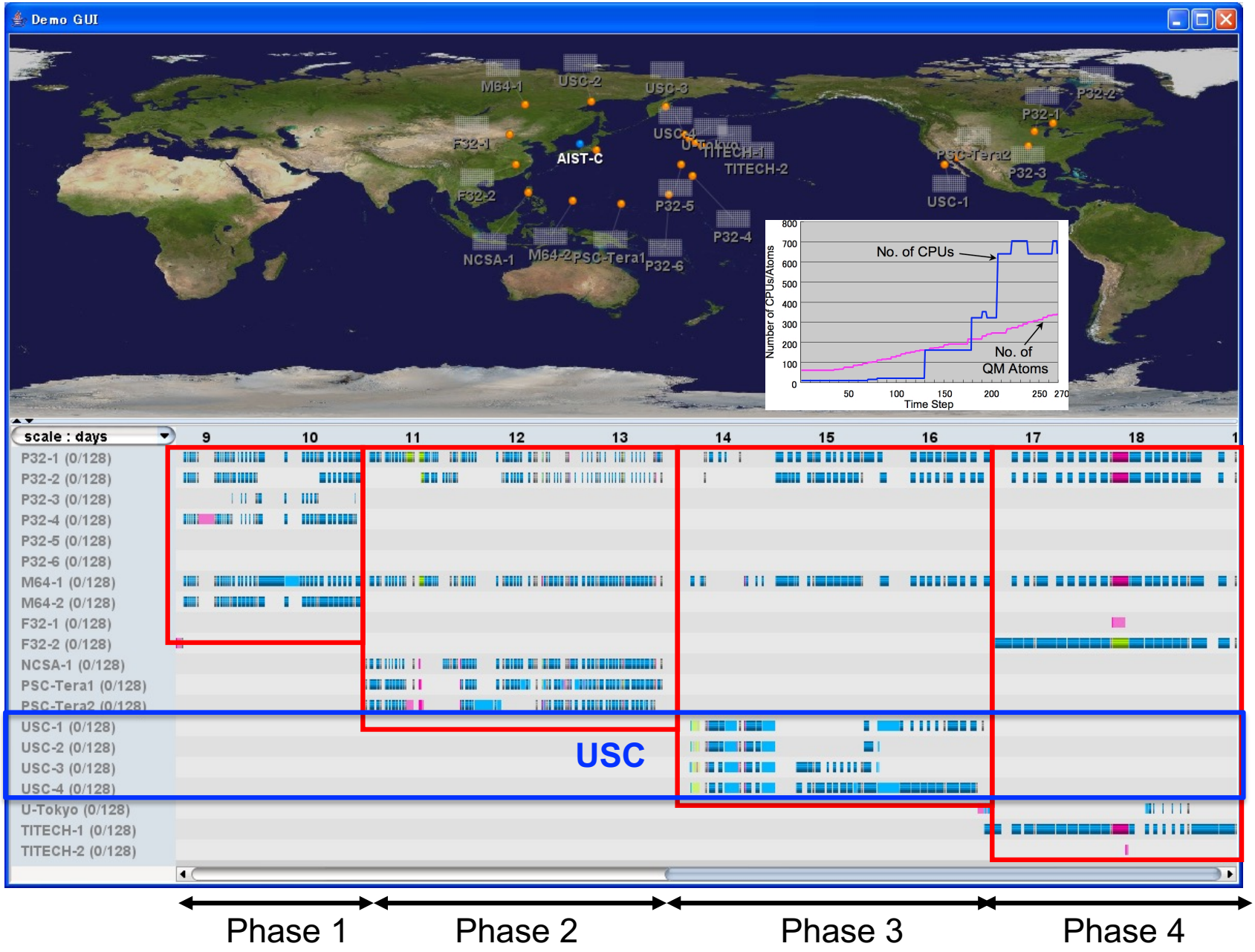
```
double A[n][n],B[n][n],C[n][n];      /* Data Declaration */
dmmul(n,A,B,C);                      /* Call local function */
```



```
grpc_function_handle_default(&hdl, "dmmul");
grpc_call(hdl,n,A,B,C);   /* Call server side routine */
```

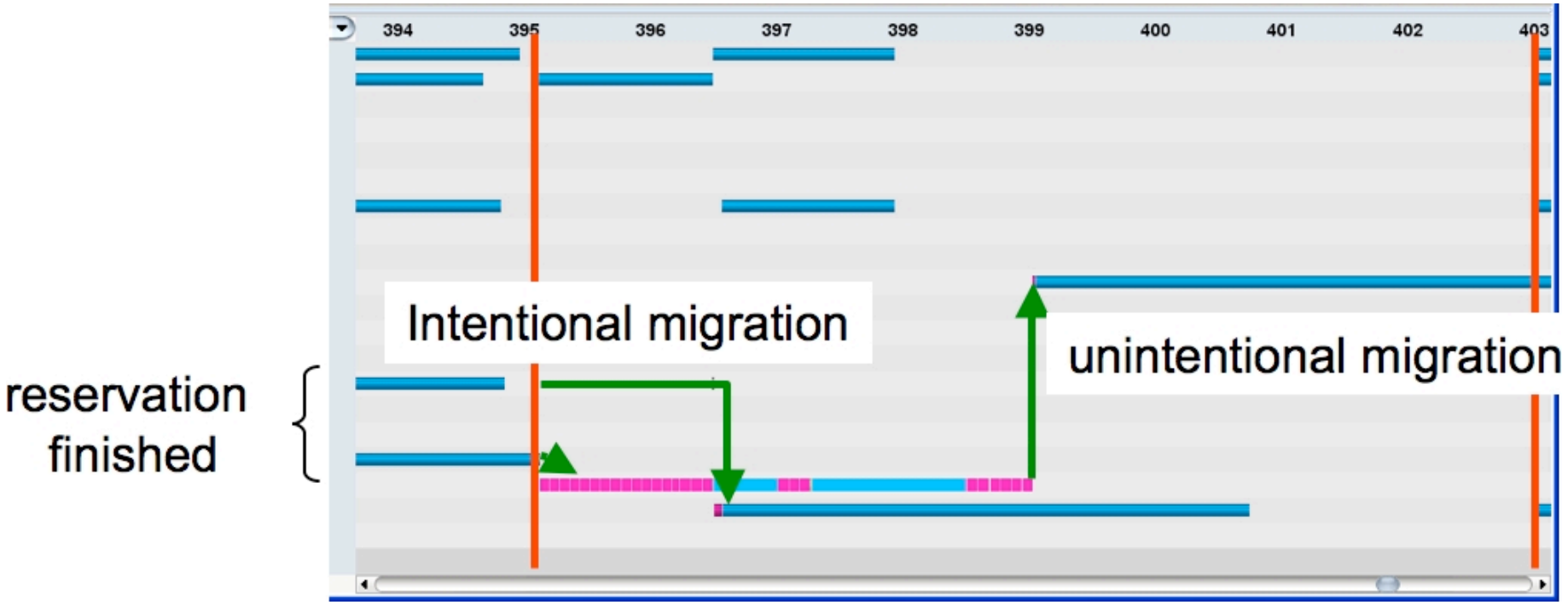
- **Ninf–G Grid RPC system**
  http://ninf.apgrid.org

US-Japan Grid Testbed



Fault Tolerance

- **Automated migration in response to unexpected faults**



Current & Future Supercomputing

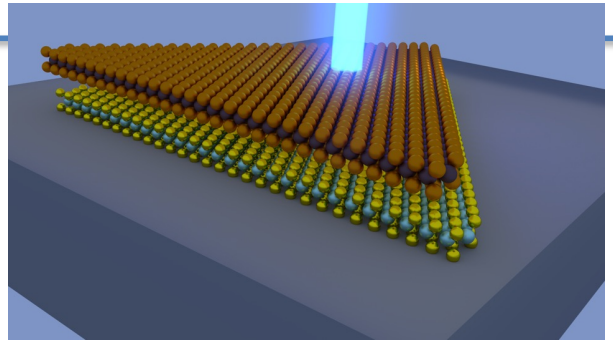
- **Won two DOE supercomputing awards to develop & deploy metascalable ("design once, scale on future platforms") simulation algorithms**

U.S. DEPARTMENT OF ENERGY
INCITE
LEADERSHIP COMPUTING
Innovative & Novel Computational Impact on Theory & Experiment

- **Atomistic simulations on million cores (pre-exascale)**

Title: AI-Guided Exascale Simulations of Quantum Materials Manufacturing and Control
PI and Co-PIs: Aiichiro Nakano–PI, Rajiv K. Kalia, Ken-ichi Nomura, Priya Vasishta

786,432-core IBM Blue Gene/Q
281,088-core Intel Xeon Phi
560-node (2,240-GPU) AMD/NVIDIA Polaris



AURORA | EARLY SCIENCE PROGRAM

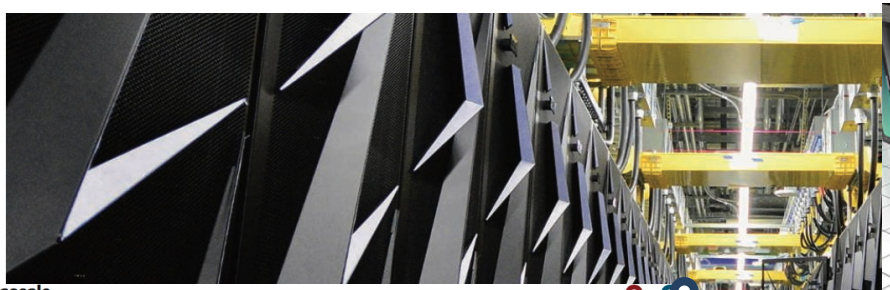
Early Science Projects for Aurora Supercomputer Announced
Metascalable layered materials genome
Investigator: Aiichiro Nakano, University of Southern California

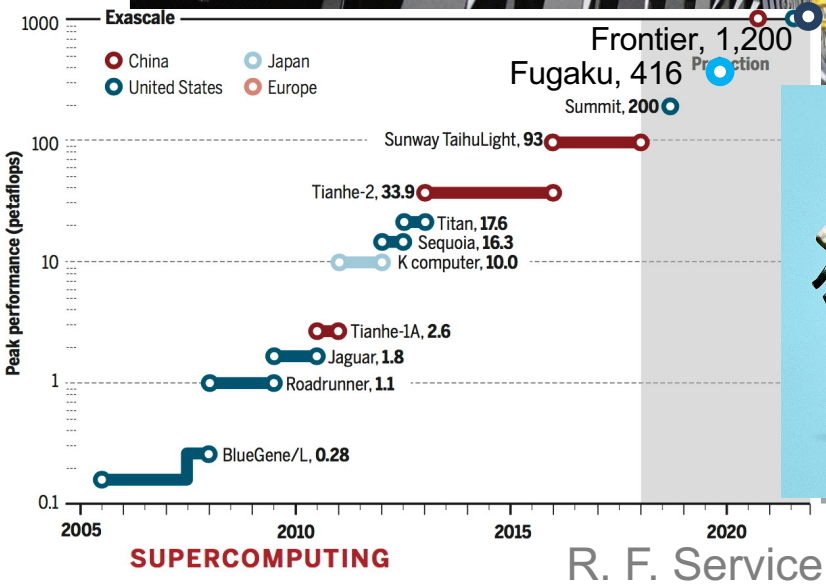
1.01 exaflop/s
Intel Aurora

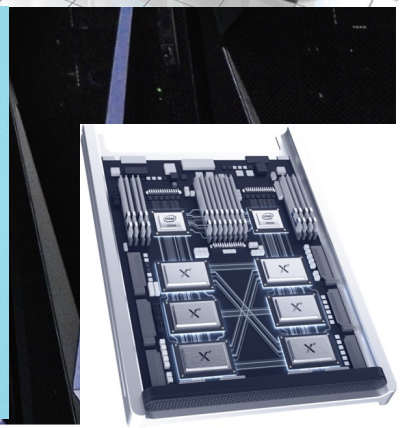
exaflop/s = 10^{18} mathematical operations per second

- **One of the initial simulation users of the next-generation DOE supercomputer**

CACS@Aurora in the Global Exascale Race







R. F. Service, *Science* 359, 617 ('18)

Design for U.S. exascale computer takes shape

Competition with China accelerates plans for next great leap in supercomputing power

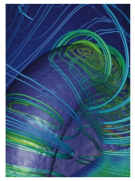
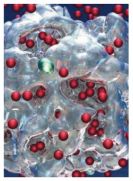
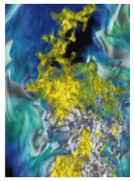
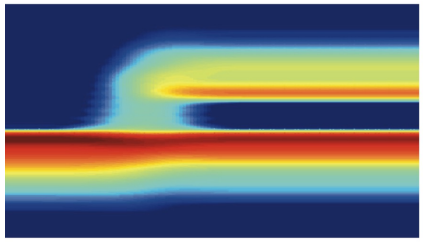
Exa(peta)flop/s = 10^{18} (10^{15}) floating-point operations per second

By **Robert F. Service**

In 1957, the launch of the Sputnik satellite vaulted the Soviet Union to the lead in the space race and galvanized the United States. U.S. supercomputer researchers are today facing their own Lemont, Illinois. That's 2 years earlier than planned. "It's a pretty exciting time," says Aiichiro Nakano, a physicist at the University of Southern California in Los Angeles who uses supercomputers to model materials made by layering stacks of atomic sheets like graphene. pace reflects a change of strategy by DOE officials last fall. Initially, the agency set up a "two lanes" approach to overcoming the challenges of an exascale machine, in particular a potentially ravenous appetite for electricity that could require the output of a small nuclear plant.

https://www.tomshardware.com/news/two-chinese-exascale-supercomputers

BES

Exa-leadership

BASIC ENERGY SCIENCES

EXASCALE REQUIREMENTS REVIEW

An Office of Science review sponsored jointly by
Advanced Scientific Computing Research and Basic Energy Sciences

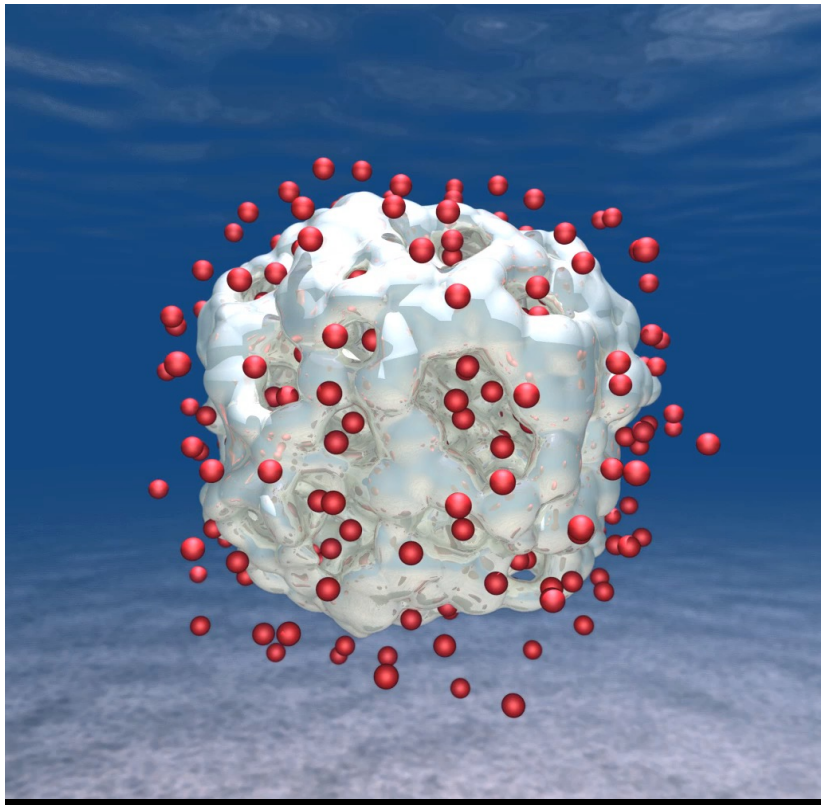
NOVEMBER 3–5, 2015

ROCKVILLE, MARYLAND

16,661-atom QMD

Shimamura *et al.*,
Nano Lett.
14, 4090 ('14)

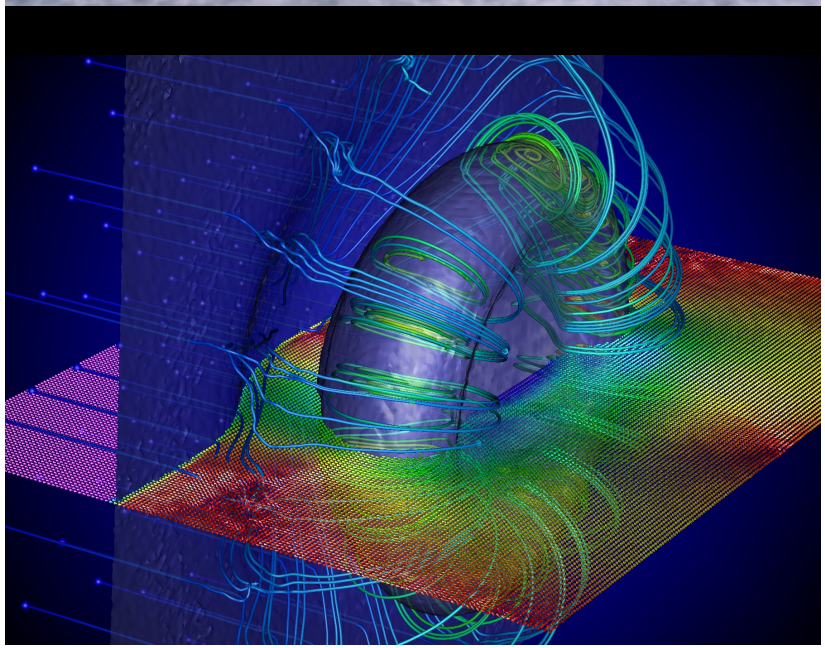
On-demand hydrogen production from water



10^9-atom RMD

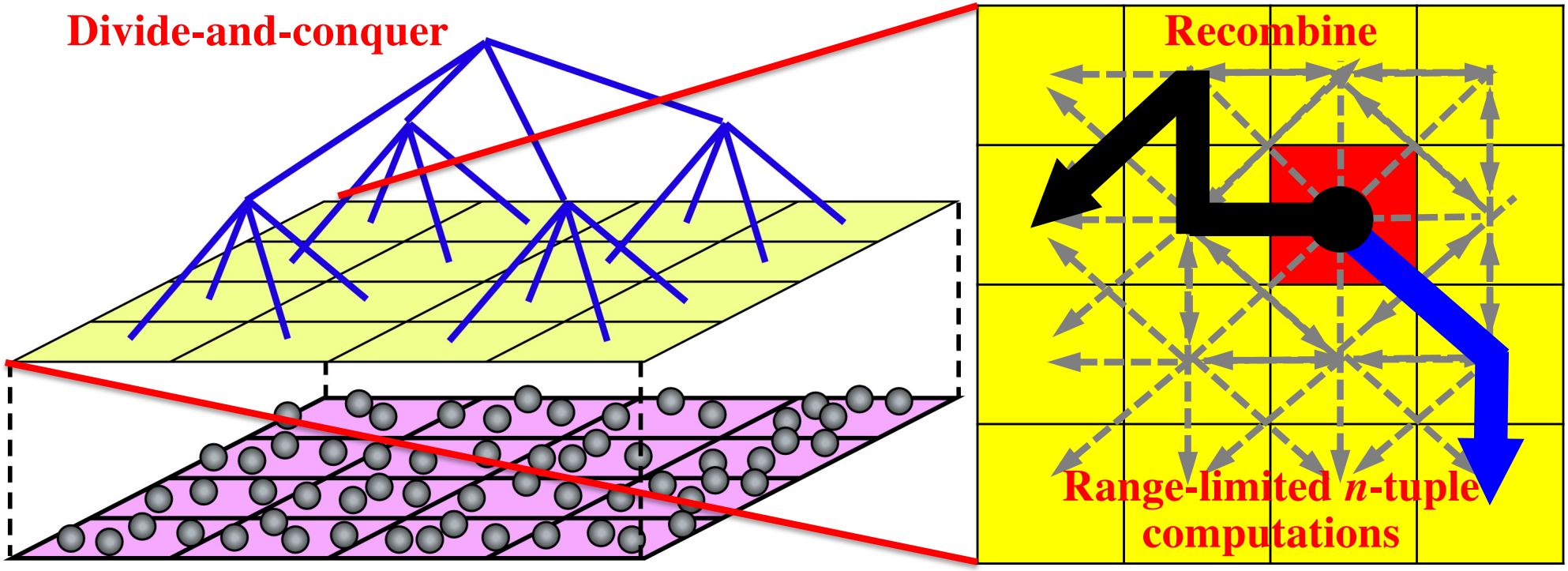
Shekhar *et al.*,
Phys. Rev. Lett.
111, 184503 ('13)

Fluid dynamics atom-by-atom



U.S. DEPARTMENT OF ENERGY

Divide-Conquer-Recombine (DCR) Engines



M. Kunaseth *et al.*, *ACM/IEEE SC13*

See lecture on "shift-collapse" algorithm

- **Lean divide-&-conquer density functional theory (LDC-DFT) algorithm minimizes the prefactor of $O(N)$ computational cost**

  F. Shimojo *et al.*, *J. Chem. Phys.* **140**, 18A529 ('14); K. Nomura *et al.*, *IEEE/ACM SC14*

- **Extended-Lagrangian reactive molecular dynamics (XRMD) algorithm eliminates the speed-limiting charge iteration**

  K. Nomura *et al.*, *Comput. Phys. Commun.* **192**, 91 ('15)

Divide-Conquer-(Re)combine

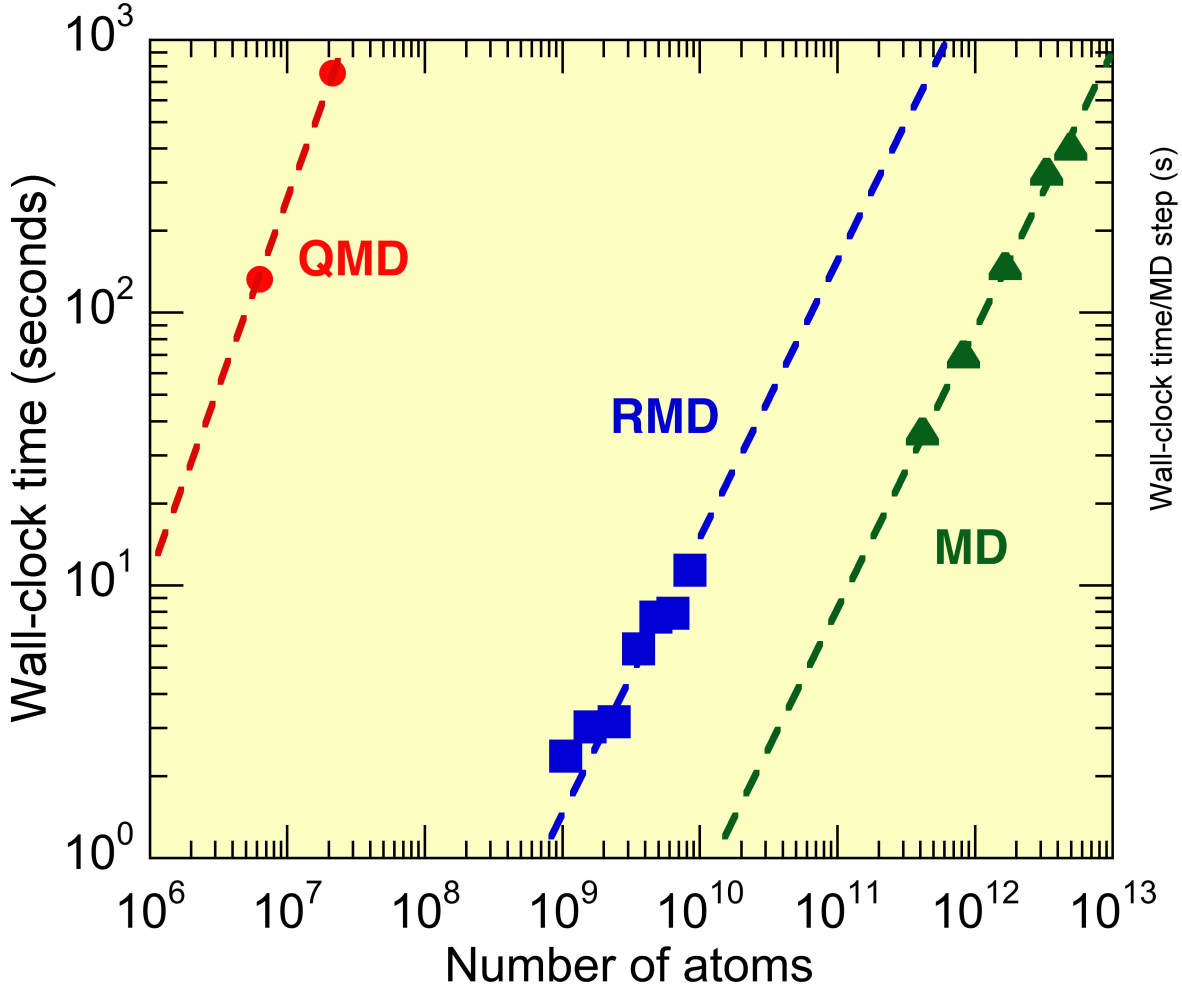
- **"The first was to never accept anything as true which I could not accept as obviously true. The second was to divide each of the problems in as many parts as I should to solve them. The third, beginning with the simplest and easiest to understand matters, little by little, to the most complex knowledge. And the last resolution was to make my enumerations so complete and my reviews so general that I could be assured that I had not omitted anything." (René Descartes, *Discourse on Method*, 1637)**

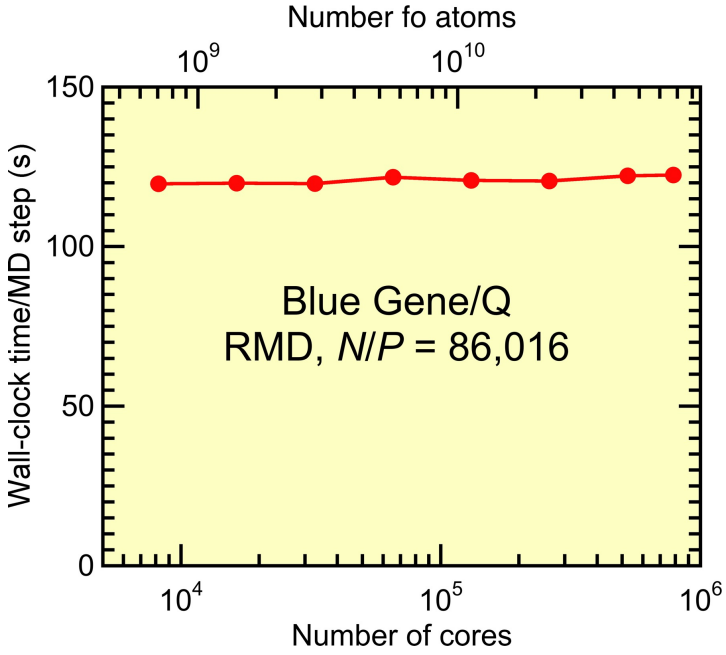
- **「モデルの分割ー再統合の方法の優れた点は、分割した要素的概念を、モデルの理解に役立つように再構成することができ、そこに創造の入り込む余地があるという点にある。」（福井謙一学問の創造、1987）**

room for creativity

Kenichi Fukui [*Nobel Chemistry Prize*, '81]

Scalable Simulation Algorithm Suite





QMD (quantum molecular dynamics): DC-DFT

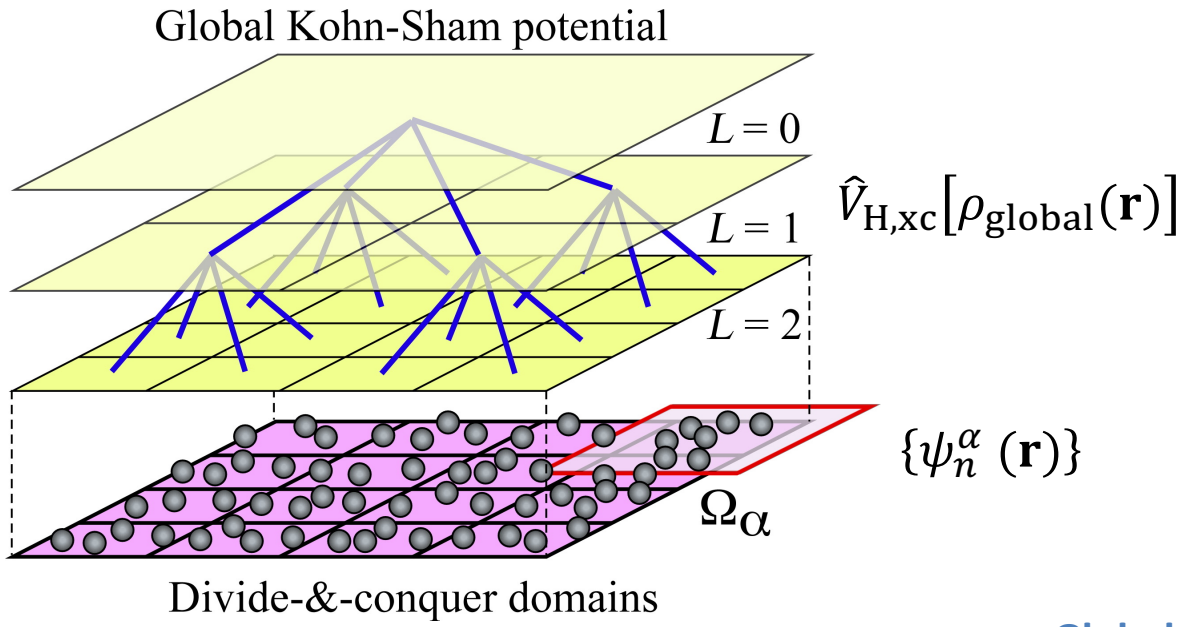
RMD (reactive molecular dynamics): F-ReaxFF

MD (molecular dynamics): MRMD

- **4.9 trillion-atom space-time multiresolution MD (MRMD) of SiO_2**
- **67.6 billion-atom fast reactive force-field (F-ReaxFF) RMD of RDX**
- **39.8 trillion grid points (50.3 million-atom) DC-DFT QMD of SiC**

parallel efficiency 0.984 on 786,432 Blue Gene/Q cores

Divide-&-Conquer Density Functional Theory



- **Overlapping spatial domains:** $\Omega = \cup_\alpha \Omega_\alpha$
- **Domain Kohn-Sham equations**

Global-local self-consistent field (SCF) iteration

$$\left(-\frac{1}{2}\nabla^2 + \hat{V}_{\mathrm{ion}} + \hat{V}_{\mathrm{H,xc}}[\rho_{\mathrm{global}}(\mathbf{r})]\right)\psi_n^\alpha(\mathbf{r}) = \epsilon_n^\alpha \psi_n^\alpha(\mathbf{r})$$

- **Global & domain electron densities**

$$\rho_{\mathrm{global}}(\mathbf{r}) = \Sigma_\alpha\, p_\alpha(\mathbf{r})\rho_\alpha(\mathbf{r}) \leftarrow \rho_\alpha(\mathbf{r}) = \Sigma_n |\psi_n^\alpha|^2\, \Theta(\mu - \epsilon_n^\alpha)$$

Domain support function

$$\Sigma_\alpha\, p_\alpha(\mathbf{r}) = 1$$

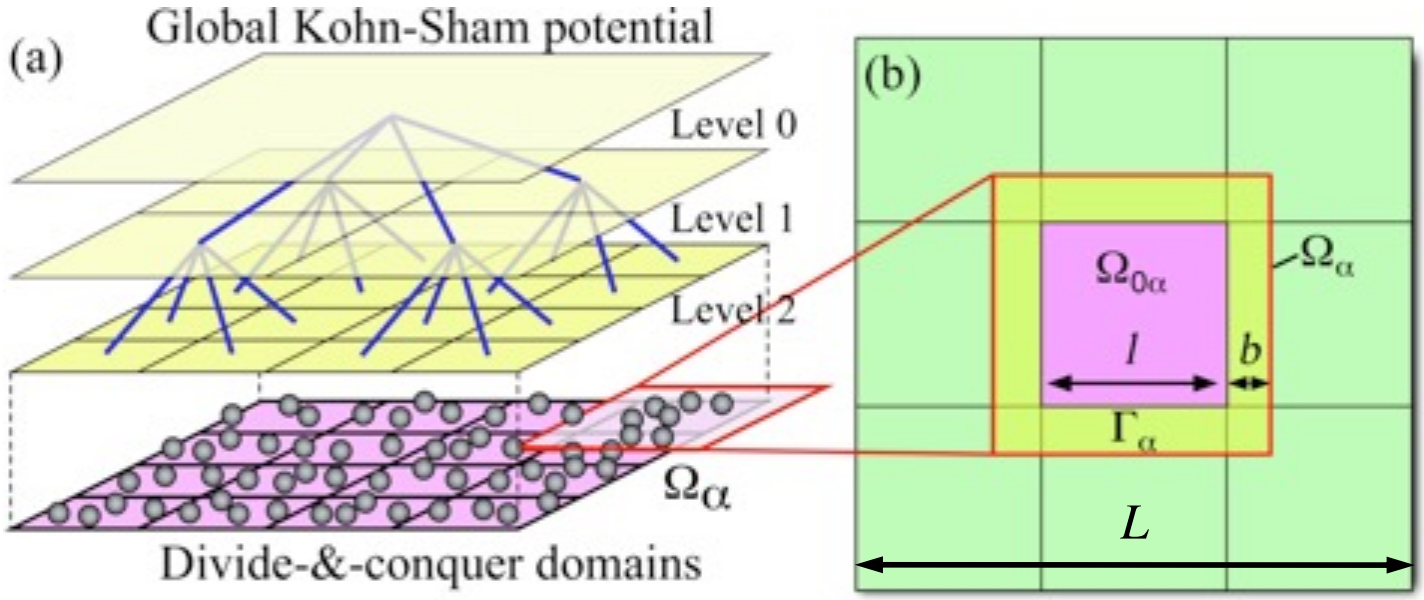
Global chemical potential

$$N = \int d\mathbf{r}\, \rho_{\mathrm{global}}(\mathbf{r})$$

cf. subsystem DFT [W. Mi *et al.*, *Comput. Phys. Commun.* **269**, 108122 ('21)]

Optimization of Divide-&-Conquer DFT

- **Computational parameters of DC-DFT = domain size (l) + buffer thickness (b)**



- **Complexity analysis to optimize the domain size l**

$$l_* = \mathrm{argmin}\left(T_{\mathrm{comp}}(l)\right) = \mathrm{argmin}\left(\left(\frac{L}{l}\right)^3 (l+2b)^{3\nu}\right) = \frac{2b}{\nu-1}$$

Per-domain computational complexity of DFT = $O(n^\nu)$: $\nu = 2$ or 3 ($n <$ or $> 10^3$)

- **Error analysis: Buffer thickness b is dictated by the accuracy requirement**

$$b = \lambda \ln\left(\max\left\{|\Delta\rho_\alpha(\mathbf{r})| \,|\, \mathbf{r} \in \partial\Omega_\alpha\right\}\right)/\varepsilon\langle\rho_\alpha(\mathbf{r})\rangle \qquad |\Delta\rho|e^{-b/\lambda} = \varepsilon\langle\rho\rangle$$

Decay length — $\rho_\alpha(\mathbf{r}) - \rho_{\mathrm{global}}(\mathbf{r})$ — Error tolerance

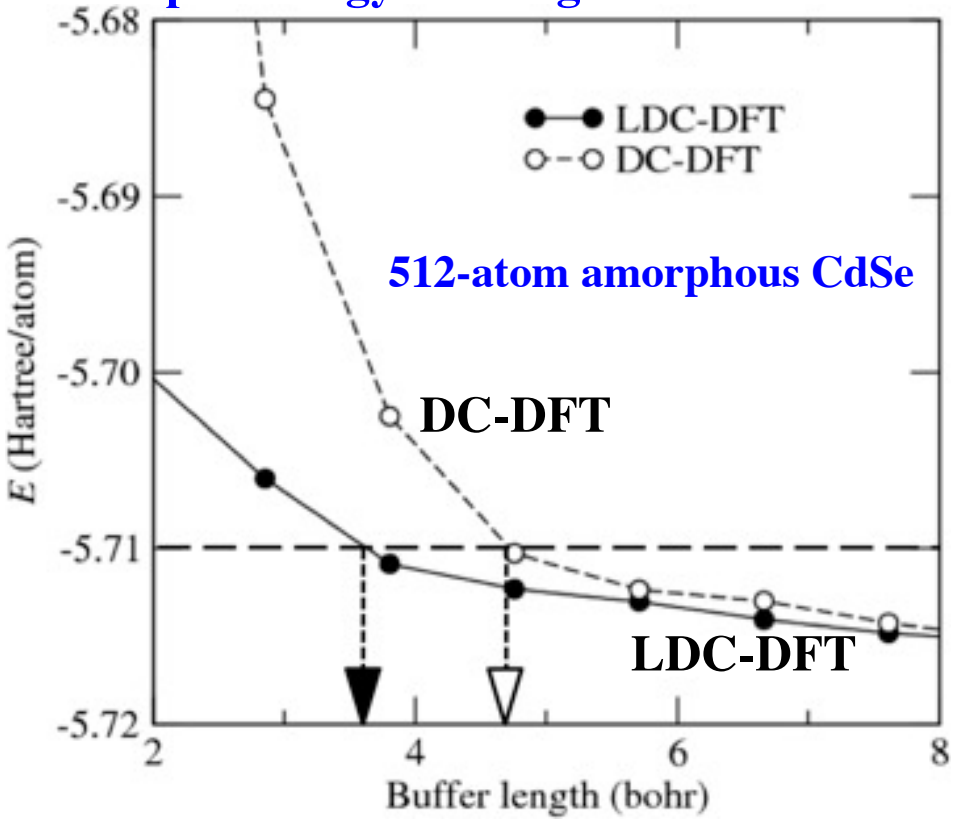
cf. quantum nearsightedness [Kohn, *Phys. Rev. Lett.* **76**, 3168 ('96); Prodan & Kohn, *P. Nat. Acad. Sci.* **102**, 11635 ('05)]

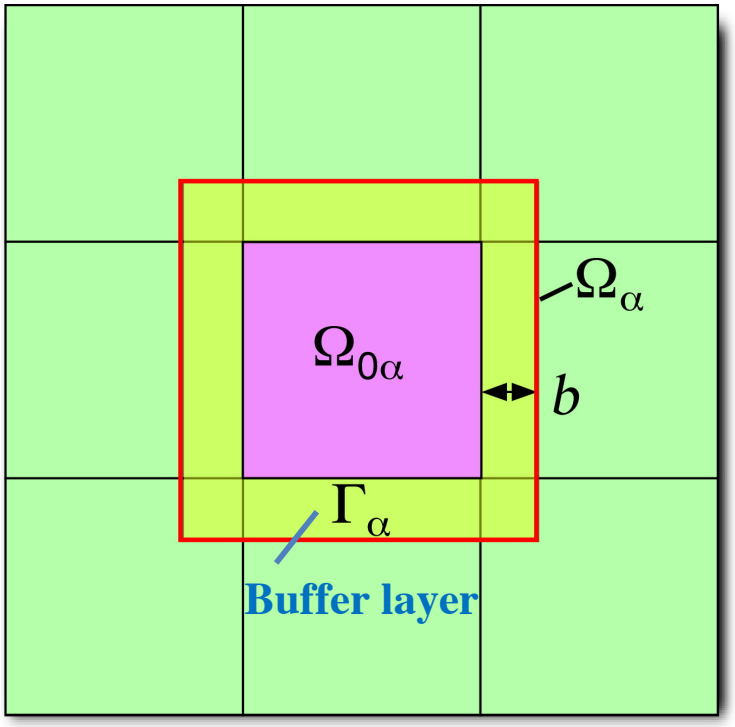
Lean Divide-&-Conquer (LDC) DFT

- **Density-adaptive boundary potential to reduce the *O*(*N*) prefactor** local approximation

$$v_\alpha^{\mathrm{bc}}(\mathbf{r}) = \int d\mathbf{r}' \frac{\partial v(\mathbf{r})}{\partial \rho(\mathbf{r}')} \left(\rho_\alpha(\mathbf{r}') - \rho_{\mathrm{global}}(\mathbf{r}') \right) \cong \frac{\rho_\alpha(\mathbf{r}) - \rho_{\mathrm{global}}(\mathbf{r})}{\xi}$$

- **More rapid energy convergence of LDC-DFT compared with nonadaptive DC-DFT**



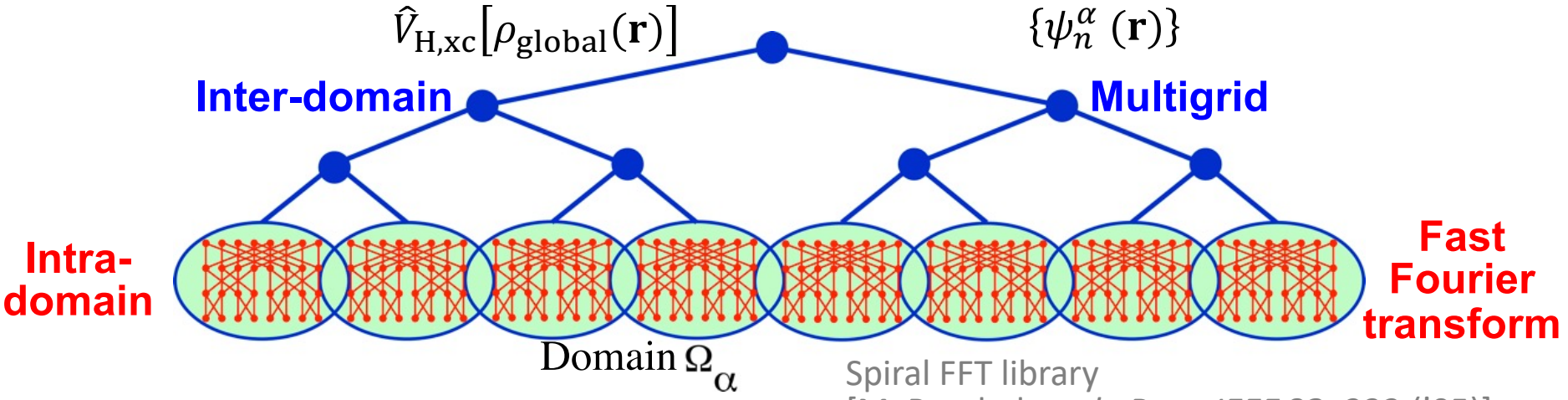


- **Factor 2.03 (for ν = 2) ~ 2.89 (for ν = 3) reduction of the computational cost with an error tolerance of 5×10^{-3} a.u. (per-domain complexity: n^ν)**

F. Shimojo *et al.*, *J. Chem. Phys.* **140**, 18A529 ('14);
Phys. Rev. B **77**, 085103 ('08); *Comput. Phys. Commun.* **167**, 151 ('05)

Hierarchical Computing

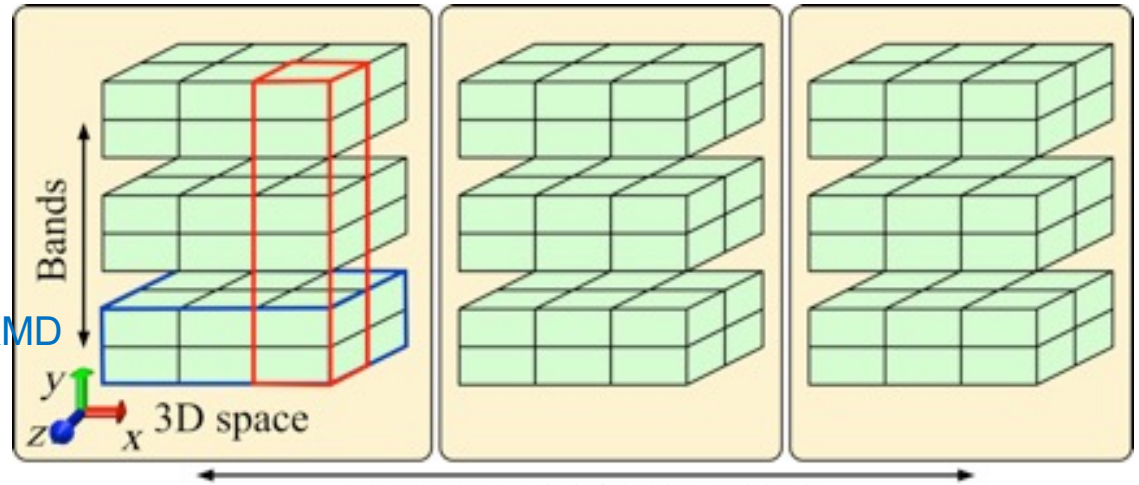
- **Globally scalable (real-space multigrid) + locally fast (plane wave) electronic solver**



Spiral FFT library
[M. Puschel *et al.*, *Proc. IEEE* **93**, 232 ('05)]

cf. globally- sparse-yet-locally-dense eigensolver [J. H. Lam *et al.*, *Nature Commun*. **15**, 3479 ('24)]

- **Hierarchical band (*i.e.*, Kohn-Sham orbital) + space + domain (BSD) decomposition**



QM-nodes in QXMD

MD-nodes in QXMD software

Parallel Efficiency

- **Execution time: $T(W,P)$**
  W: Workload
  P: Number of processors

- **Speed:** $$S(W,P) = \frac{W}{T(W,P)}$$

- **Speedup:** $$S_P = \frac{S(W_P,P)}{S(W_1,1)} = \frac{W_P T(W_1,1)}{W_1 T(W_P,P)}$$

- **Efficiency:** $$E_P = \frac{S_P}{P} = \frac{W_P T(W_1,1)}{P W_1 T(W_P,P)}$$

Ideal speedup

How to scale W_P with P?

See https://aiichironakano.github.io/cs596.html

Fixed Problem-Size Scaling

$W_P = W$—constant (strong scaling)

- **Speedup:** $S_P = \dfrac{T(W,1)}{T(W,P)}$

$$S_P = \frac{S(W_P,P)}{S(W_1,1)} = \frac{W_P T(W_1,1)}{W_1 T(W_P,P)}$$

- **Efficiency:** $E_P = \dfrac{T(W,1)}{PT(W,P)}$

$$E_P = \frac{S_P}{P} = \frac{W_P T(W_1,1)}{PW_1 T(W_P,P)}$$

Solving the same problem faster using more processors!

- **Amdahl's law: f (= sequential fraction of the workload) limits the asymptotic speedup**

$$S_P = \frac{T(W,1)}{T(W,P)} \leq P$$

$$T(W,P) = fT(W,1) + \frac{(1-f)T(W,1)}{P}$$

$$\therefore S_P = \frac{T(W,1)}{T(W,P)} = \frac{1}{f + (1-f)/P}$$

$$\therefore S_P \to \frac{1}{f} \quad (P \to \infty)$$

Isogranular Scaling

$W_P = Pw$ (weak scaling)
w = constant workload per processor (granularity)

- **Speedup:** $$S_P = \frac{S(P \bullet w, P)}{S(w,1)} = \frac{P \bullet w / T(P \bullet w, P)}{w / T(w,1)} = \frac{P \bullet T(w,1)}{T(P \bullet w, P)}$$

- **Efficiency:** $$E_P = \frac{S_P}{P} = \frac{T(w,1)}{T(P \bullet w, P)}$$

$$S_P = \frac{S(W_P, P)}{S(W_1, 1)} = \frac{W_P T(W_1, 1)}{W_1 T(W_P, P)}$$

$$E_P = \frac{S_P}{P} = \frac{W_P T(W_1, 1)}{P W_1 T(W_P, P)}$$

Solving larger problems within the same time using more processors!

$$E_P = \frac{T(w,1)}{T(Pw,P)} \leq 1$$

Analysis of Parallel MD

- **Parallel execution time:**
  Workload ∝ Number of atoms, *N* (linked-list cell algorithm)

MPI_Allreduce()

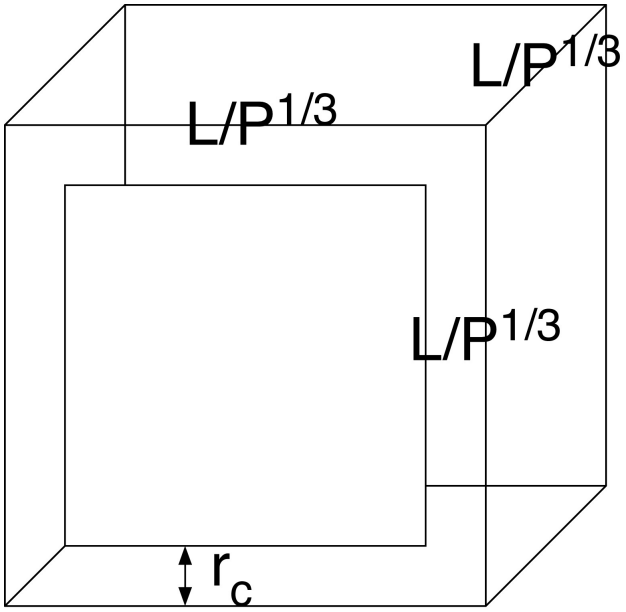
$$T(N,P) = T_{\text{comp}}(N,P) + T_{\text{comm}}(N,P) + T_{\text{global}}(P)$$
$$= a\frac{N}{P} + b\left(\frac{N}{P}\right)^{2/3} + c\log P$$

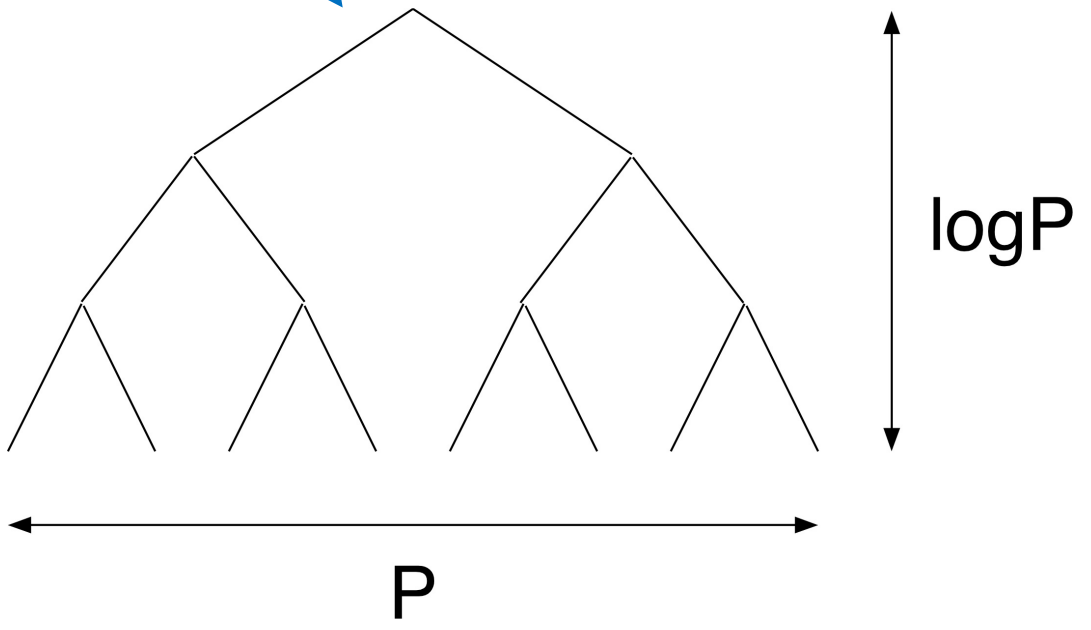
facets: $\overbrace{6}$

cached volume: $\overbrace{\frac{L^2}{P^{2/3}} r_c}$ × atom density: $\overbrace{\rho}$

$$= 6r_c \frac{N^{2/3}/\rho^{2/3}}{P^{2/3}} \rho$$
$$= 6r_c \rho^{1/3} \left(\frac{N}{P}\right)^{2/3}$$

$\left(\because \frac{N}{L^3} = \rho \Rightarrow L^2 = \frac{N^{2/3}}{\rho^{2/3}}\right)$





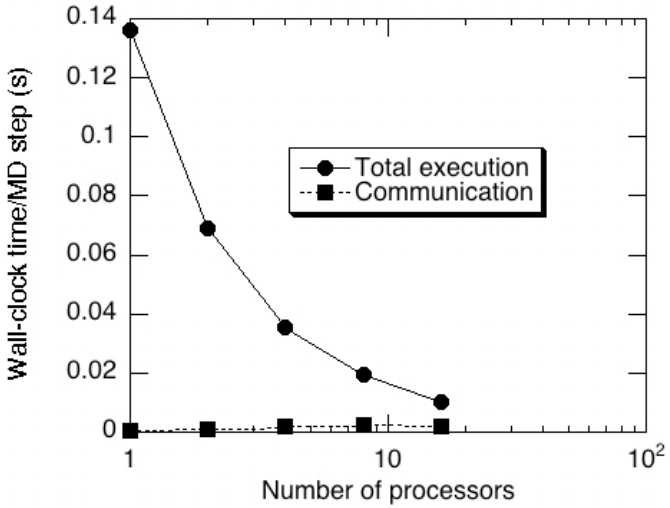
Fixed Problem-Size Scaling

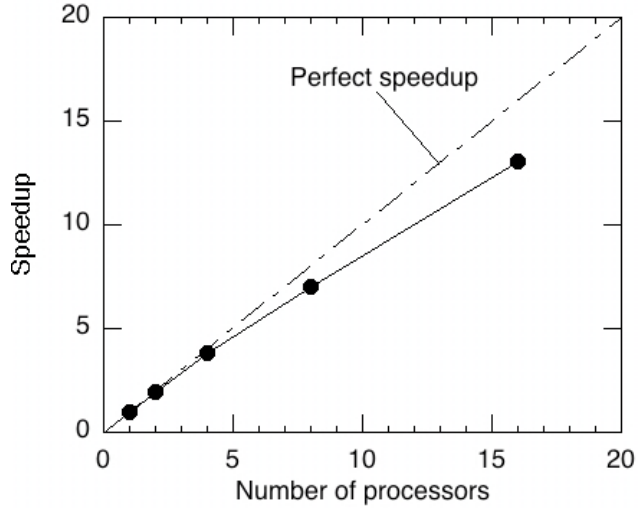
- Speedup:

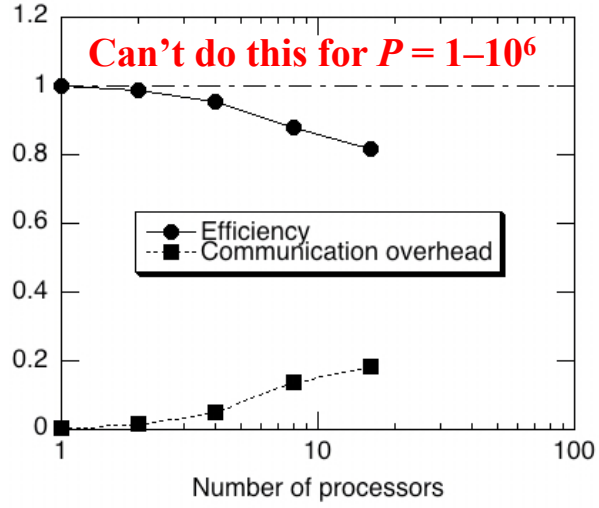
$$S_P = \frac{T(N,1)}{T(N,P)} = \frac{aN}{\frac{aN/P + b(N/P)^{2/3} + c\log P}{P}} = \frac{}{1 + \frac{b}{a}\left(\frac{P}{N}\right)^{1/3} + \frac{c}{a}\frac{P\log P}{N}}$$

- Efficiency:

$$E_P = \frac{S_P}{P} = \frac{1}{1 + \frac{b}{a}\left(\frac{P}{N}\right)^{1/3} + \frac{c}{a}\frac{P\log P}{N}}$$





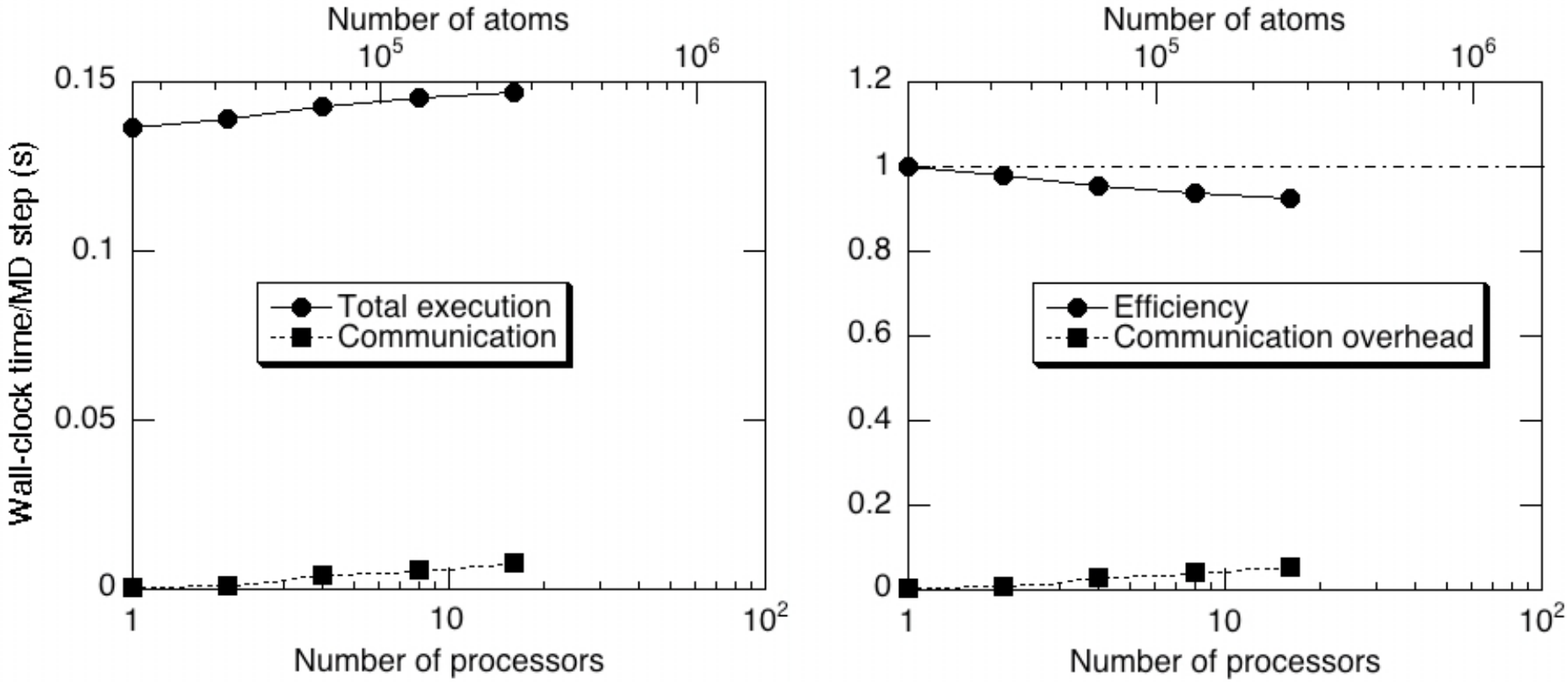


pmd.c: N = 16,384, on HPC (predecessor of CARC)

Isogranular Scaling of Parallel MD

- $n = N/P$ = constant: doable for arbitrarily large P
- Efficiency:

$$E_P = \frac{T(n,1)}{T(nP,P)} = \frac{an}{an + bn^{2/3} + c\log P} = \frac{1}{1 + \frac{b}{a}n^{-1/3} + \frac{c}{an}\log P}$$

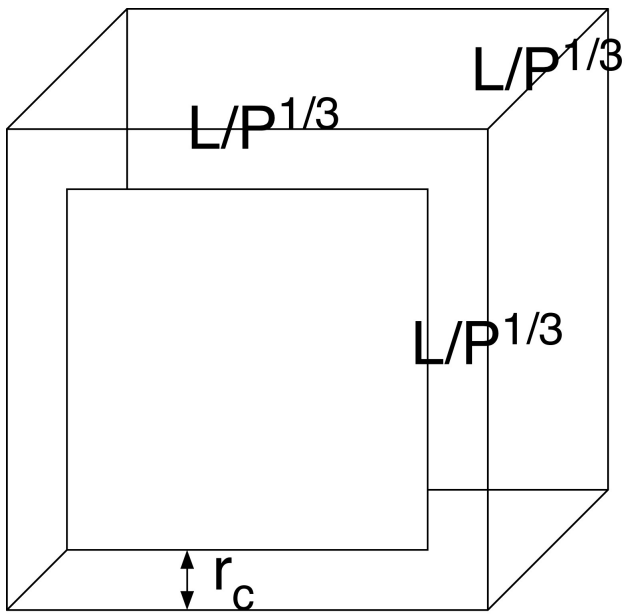


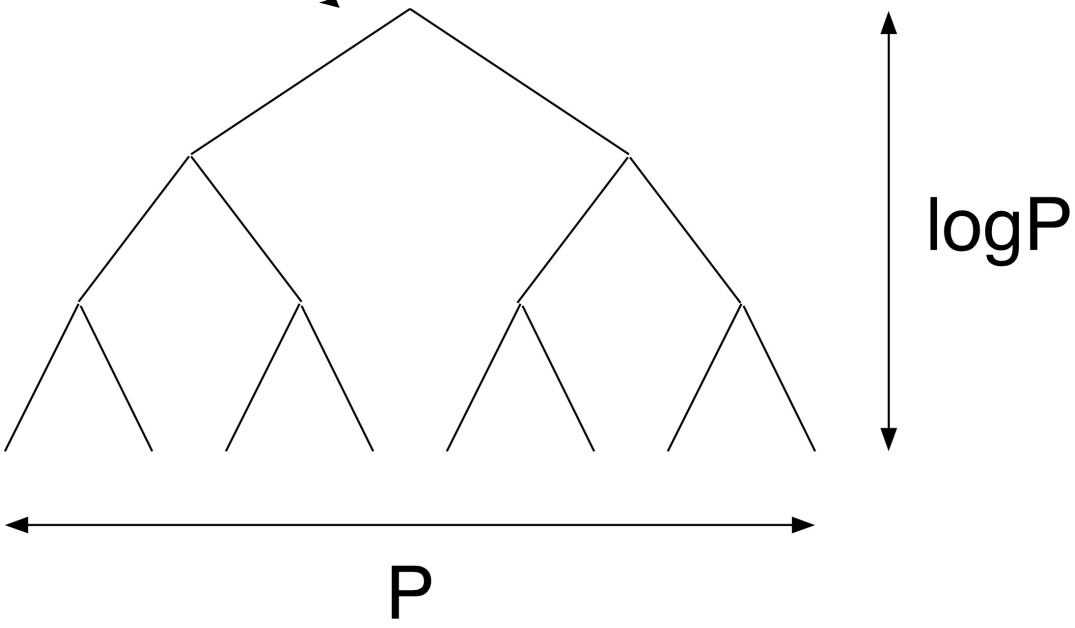
pmd.c: N/P = 16,384, on HPC (predecessor of CARC)

Analysis of Parallel MD

- **Parallel execution time:**
  Workload ∝ Number of atoms, *N* (linked-list cell algorithm)

$$T(N,P) = T_{\text{comp}}(N,P) + T_{\text{comm}}(N,P) + T_{\text{global}}(P)$$
$$= a\frac{N}{P} + b\left(\frac{N}{P}\right)^{2/3} + c\log P$$





Fixed Problem-Size Scaling

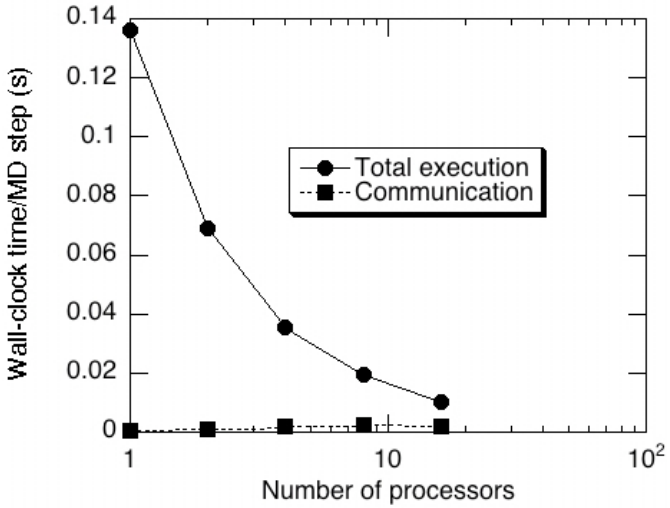
- **Speedup:**

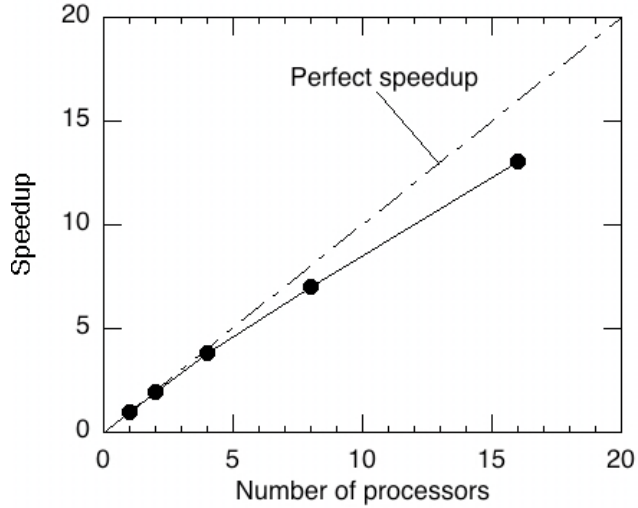
$$S_P = \frac{T(N,1)}{T(N,P)} = \frac{aN}{aN/P + b(N/P)^{2/3} + c\log P}$$

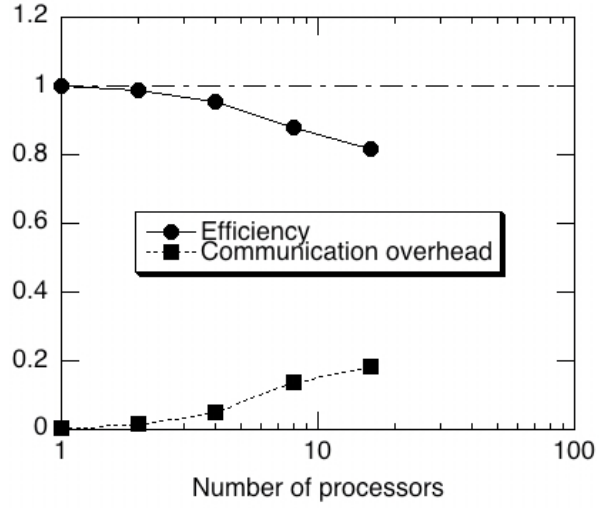
$$= \frac{P}{1 + \frac{b}{a}\left(\frac{P}{N}\right)^{1/3} + \frac{c}{a}\frac{P\log P}{N}}$$

- **Efficiency:**

$$E_P = \frac{S_P}{P} = \frac{1}{1 + \frac{b}{a}\left(\frac{P}{N}\right)^{1/3} + \frac{c}{a}\frac{P\log P}{N}}$$





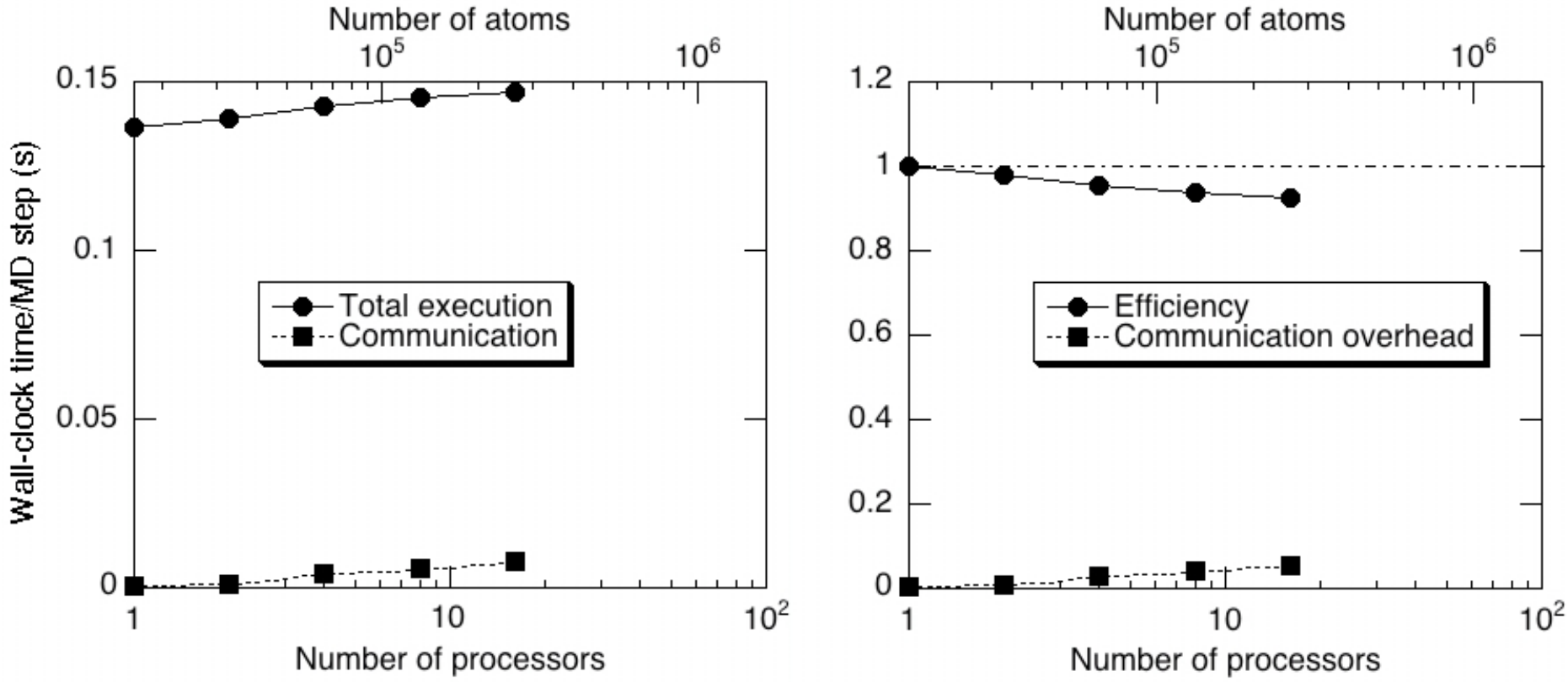


pmd.c: N = 16,384, on HPC

Isogranular Scaling of Parallel MD

- $n = N/P$ = constant
- **Efficiency:**

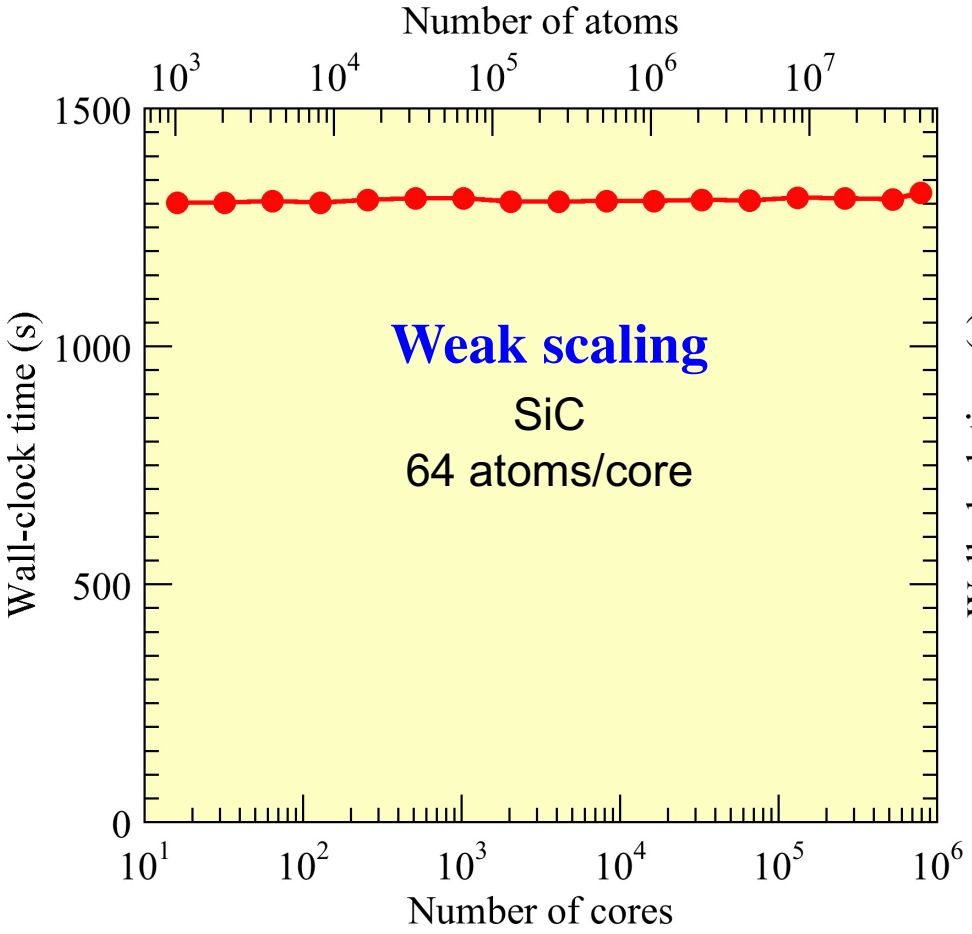
$$E_P = \frac{T(n,1)}{T(nP,P)} = \frac{an}{an + bn^{2/3} + c\log P} = \frac{1}{1 + \frac{b}{a}n^{-1/3} + \frac{c}{an}\log P}$$

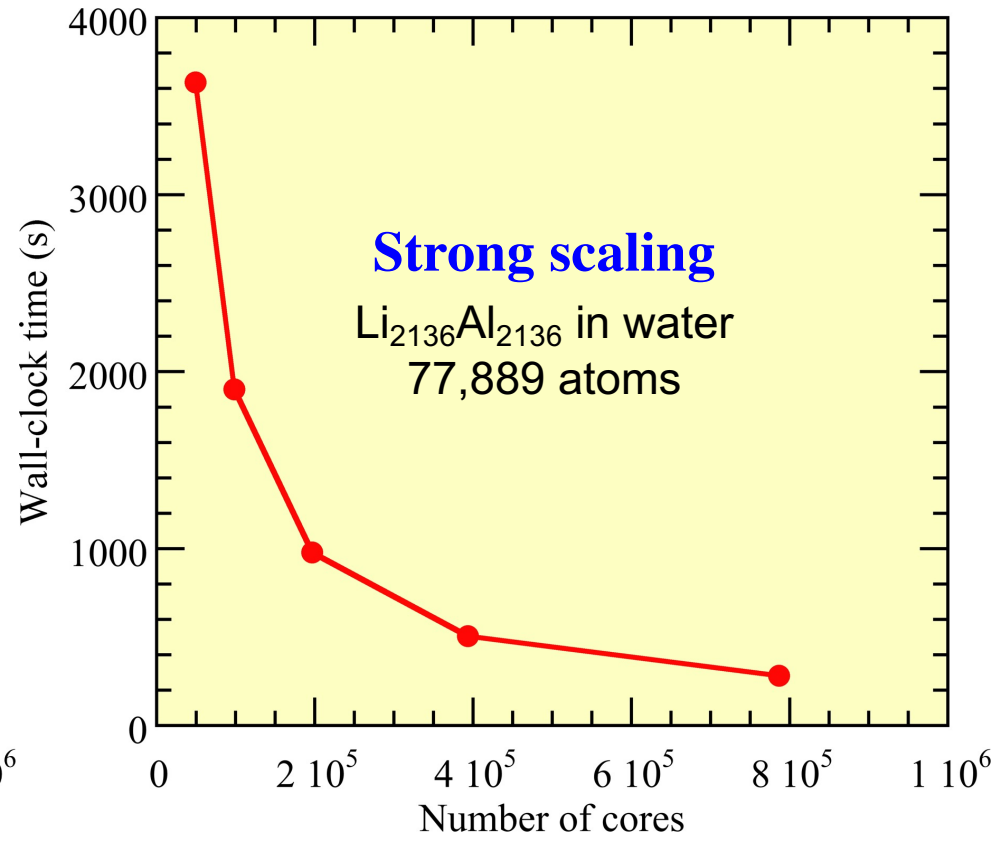


pmd.c: N/P = 16,384, on HPC

Parallel Performance of QXMD

- **Weak-scaling parallel efficiency is 0.984 on 786,432 Blue Gene/Q cores for a 50,331,648-atom SiC system**
- **Strong-scale parallel efficiency is 0.803 on 786,432 Blue Gene/Q cores**





- **62-fold reduction of time-to-solution** [441 s/SCF-step for 50.3M atoms] **from the previous state-of-the-art** [55 s/SCF-step for 102K atoms, Osei-Kuffuor *et al.*, *PRL* '14]

K. Nomura *et al.*, *IEEE/ACM Supercomputing, SC14* ('14)

BLASification

- **Transform from band-by-band to all-band computations to utilize a matrix-matrix subroutine (DGEMM) in the level 3 basic linear algebra subprograms (BLAS3) library**
- **Algebraic transformation of computations**

Example: Nonlocal pseudopotential operation

D. Vanderbilt, *Phys. Rev. B* **41**, 7892 ('90)

$$\hat{v}_{\mathrm{nl}}|\psi_n^\alpha\rangle = \sum_I^{N_{\mathrm{atom}}} \sum_{ij}^{L_{\max}} |\beta_{i,I}\rangle D_{ij,I} \langle\beta_{j,I}|\psi_n^\alpha\rangle \quad (n = 1, \ldots, N_{\mathrm{band}})$$

$$\boldsymbol{\Psi} = \left[|\psi_1^\alpha\rangle, \ldots, |\psi_{N_{\mathrm{band}}}^\alpha\rangle\right] \widetilde{\mathbf{B}}(i) = \left[|\beta_{i,1}\rangle, \ldots, |\beta_{i,N_{\mathrm{atom}}}\rangle\right] \left[\widetilde{\mathbf{D}}(i,j)\right]_{I,J} = D_{ij,I}\delta_{IJ}$$

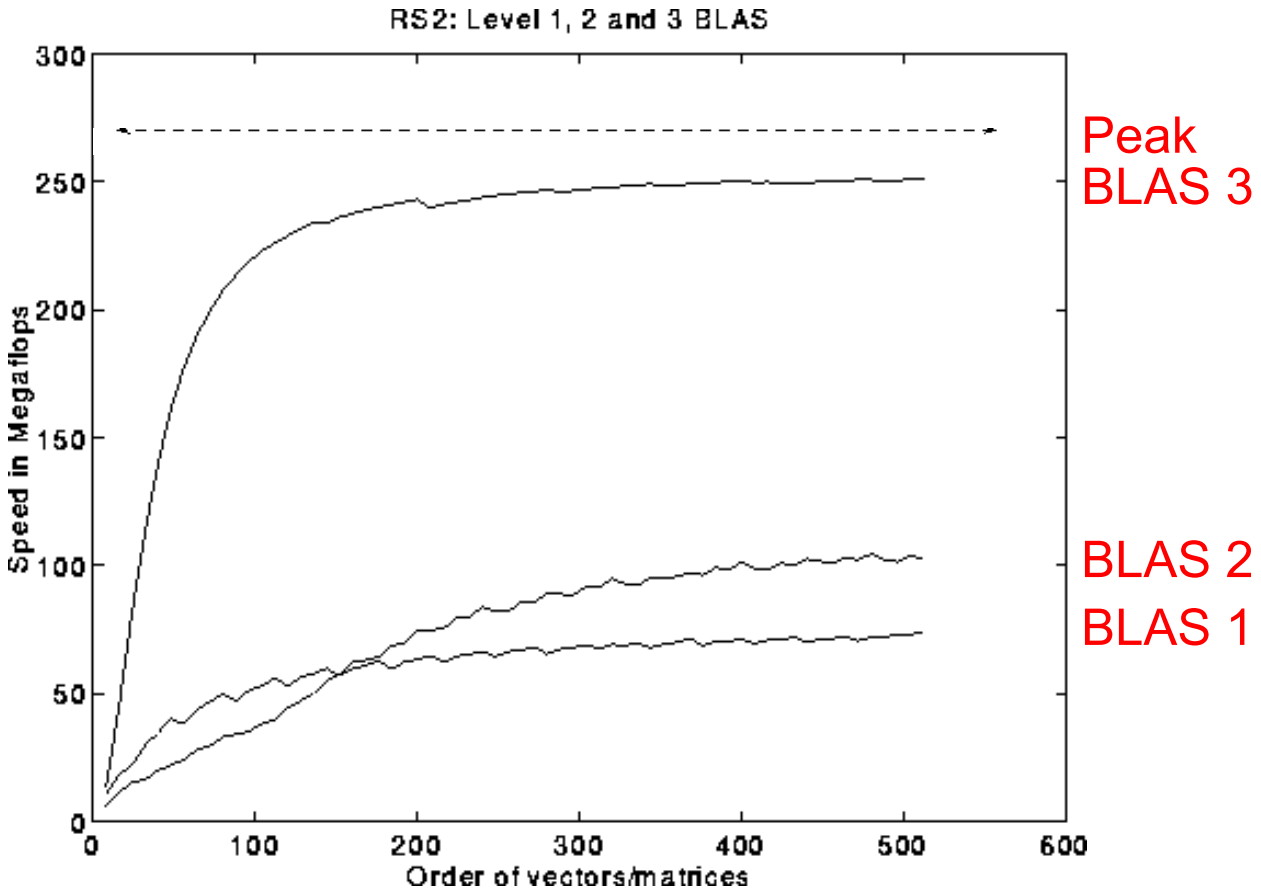
$$\hat{v}_{\mathrm{nl}}\boldsymbol{\Psi} = \sum_{i,j}^{L} \widetilde{\mathbf{B}}(i)\widetilde{\mathbf{D}}(i,j)\widetilde{\mathbf{B}}(j)^T$$

- **50.5% of the theoretical peak FLOP/s performance on 786,432 Blue Gene/Q cores (entire Mira at the Argonne Leadership Computing Facility)**
- **55% of the theoretical peak FLOP/s on Intel Xeon E5-2665**

K. Nomura *et al.*, *IEEE/ACM Supercomputing, SC14* ('14)

BLAS3-Performance Molecular Dynamics?

- **BLAS3: q = flop/memory access = (block size)$^{1/2}$**



- **Molecular dynamics: $q = O(n^2)/O(n) = O(n$: block size)**
  > Use of SIMD (single instruction multiple data) instructions on Cell, multicore (SSE)?

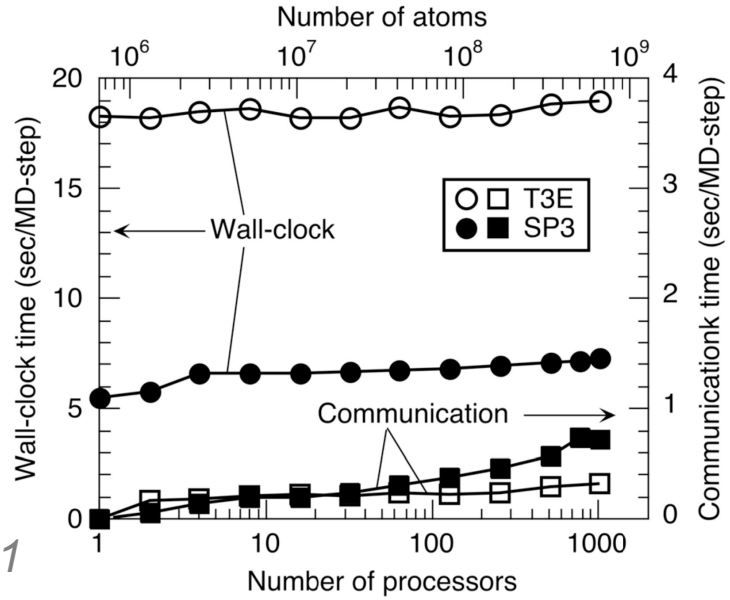
Quantum MD@Scale

Quantum dynamics at scale: ultrafast control of emergent functional materials

S. C. Tiwari, P. Sakdhnagool, R. K. Kalia, A. Krishnamoorthy, M. Kunaseth, A. Nakano, K. Nomura, P. Rajak, F. Shimojo, Y. Luo & P. Vashishta

Best Paper in *ACM HPC Asia 2020*

19 years since



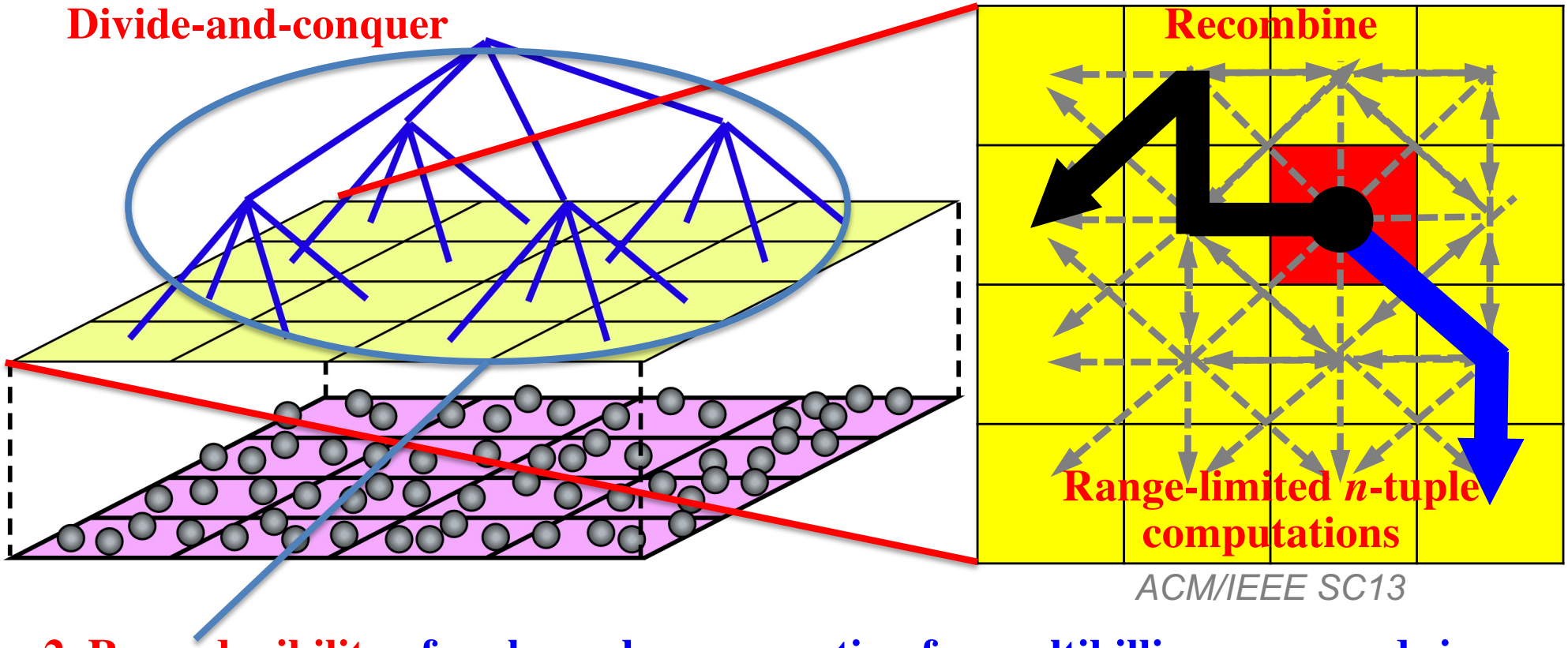
Scalable atomistic simulation algorithms for materials research, A. Nakano *et al.*, Best Paper, *IEEE/ACM Supercomputing 2001, SC01*

Exascale Computing Challenge

1. Scalability for billion-way parallelism

J. Chem. Phys. **140**, 18A529 ('14)
IEEE/ACM SC14
IEEE Computer **48(11)**, 33 ('15)

Divide-conquer-recombine (DCR) algorithmic framework
Metascalable ("design once, scale on future architectures")

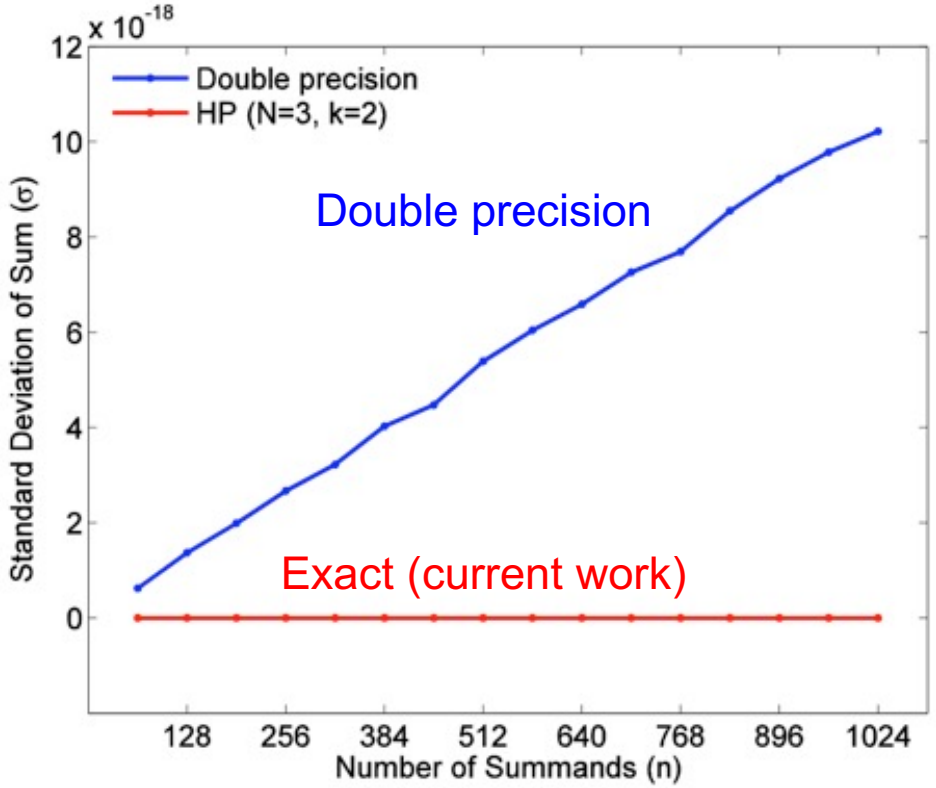


ACM/IEEE SC13

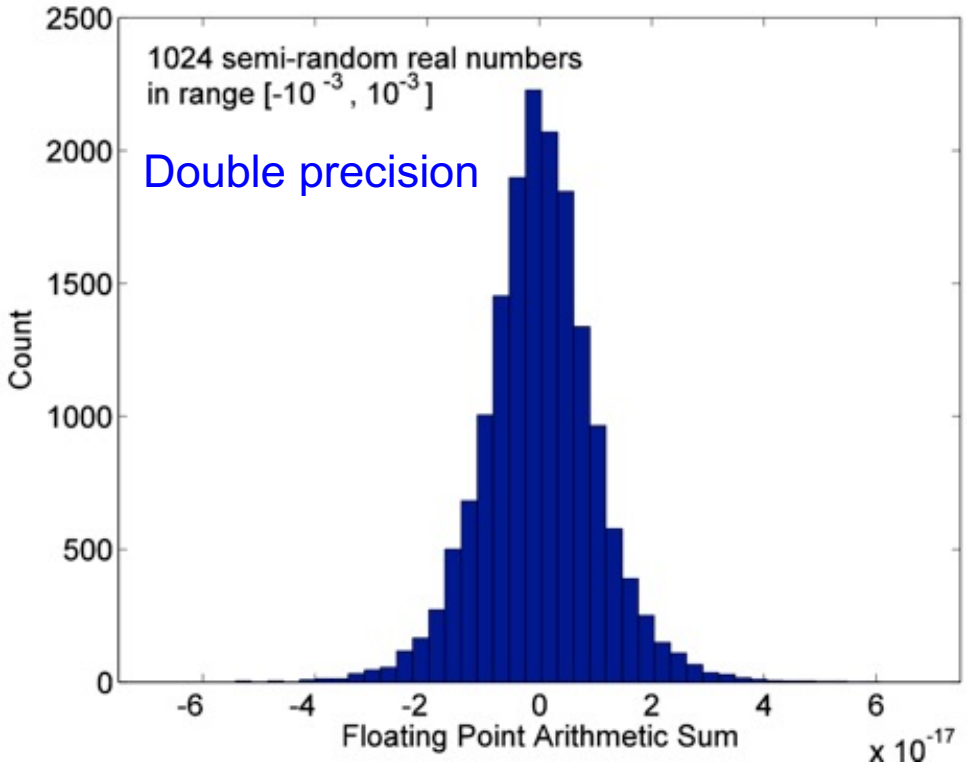
2. Reproducibility of real-number summation for multibillion summands in the global sum; double-precision arithmetic began to produce different results on different high-end architectures

Reproducibility Challenge

- **Rounding (truncation) error makes floating-point addition non-associative**



Standard deviation of sum with random summation orders



Distribution of sum with random summation orders

- **Sum becomes a random walk across the space of possible rounding error**

High-Precision (HP) Method

- **Propose an extension of the order-invariant, higher-precision intermediate-sum method by Hallberg & Adcroft** [*Par. Comput.* **40**, 140 ('14)]
- **The proposed variation represents a real number *r* using a set of *N* 64-bit unsigned integers, a_i ($i = 0, N\text{-}1$)**

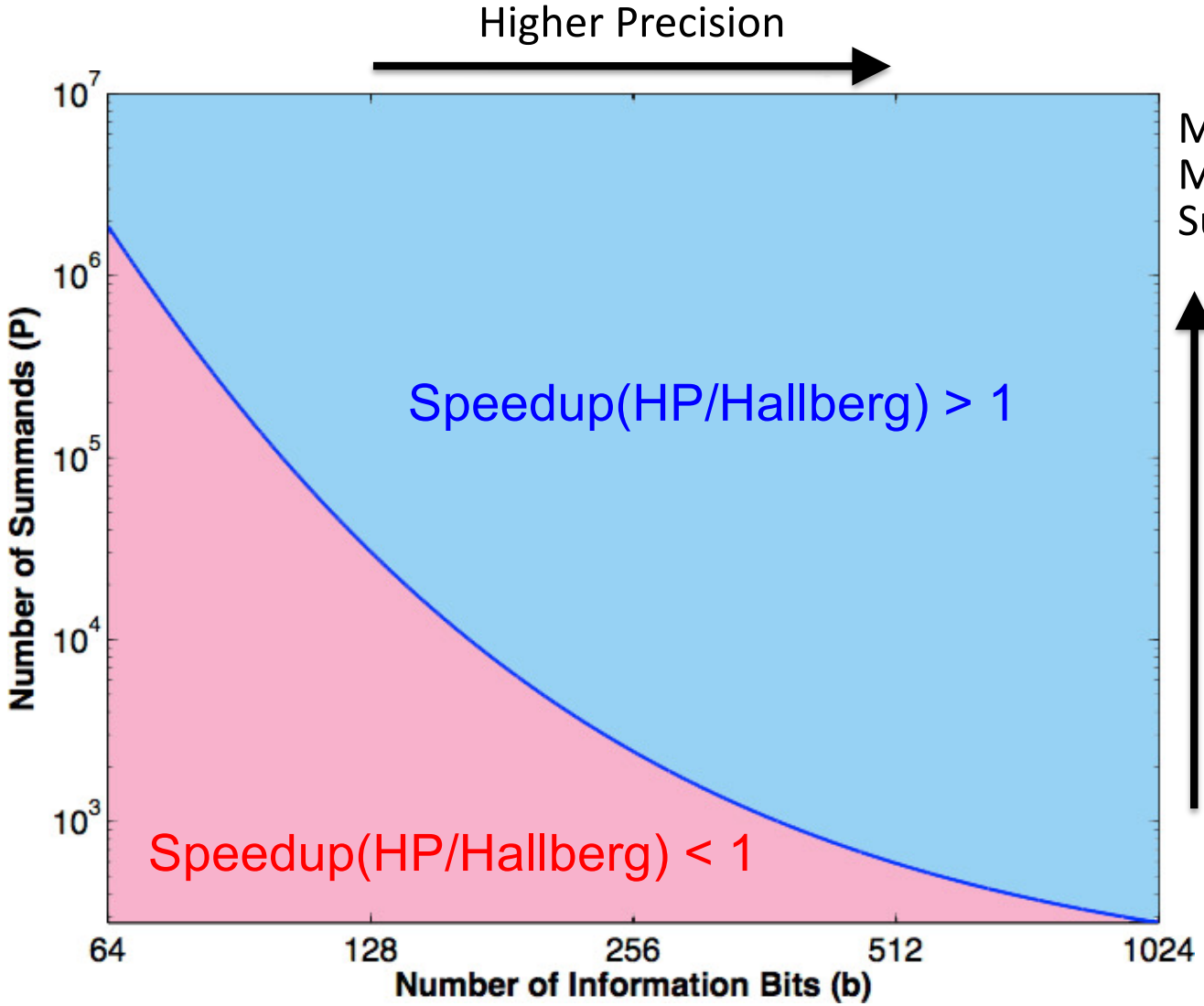
$$r = \sum_{i=0}^{N-1} a_i 2^{64(N-k-i-1)}$$
$$= \underbrace{a_0 2^{64(N-k-1)} + \cdots + a_{N-k-1}}_{N-k} + \cdots + \underbrace{a_{N-k} 2^{-64} + \cdots + a_{N-1} 2^{-64k}}_{k}$$

- ***k* is the number of 64-bit unsigned integers assigned to represent the fractional portion of *r* ($0 \le k \le N$), whereas *N*-*k* integers represent the whole-number component**
- **Negative number is represented by two's complement in integer representation, using only 1 bit**

P. E. Small *et al.*, *IEEE IPDPS 2016*

Performance Projection

- **HP sum is faster than Hallberg sum for higher precision & larger numbers of summands**



P. E. Small *et al.*, *IEEE IPDPS 2016*

Gordon Bell Prizes

aka Nobel prize of supercomputing

- **F. Gygi *et al.*, "Large-scale electronic structure calculations of high-Z metals on the BlueGene/L platform" ('06)**
- **M. Eisenbach *et al.*, "A scalable method for ab initio computation of free energies in nanoscale systems" ('09)**
- **Y. Hasegawa *et al.*, "First-principles calculations of electron states of a silicon nanowire with 100,000 atoms on the K computer" ('11)**
- **A. N. Ziogas *et al.*, "A data-centric approach to extreme-scale ab initio dissipative quantum transport simulations" ('19)**
- **S. Das *et al.*, "Large-scale materials modeling at quantum accuracy: *Ab initio* simulations of quasicrystals and interacting extended defects in metallic alloys" ('23)**

See the reading list