



Lawrence Livermore National Laboratory

Case Study: Beyond Homogeneous Decomposition with Qbox

Scaling Long-Range Forces on Massively Parallel Systems

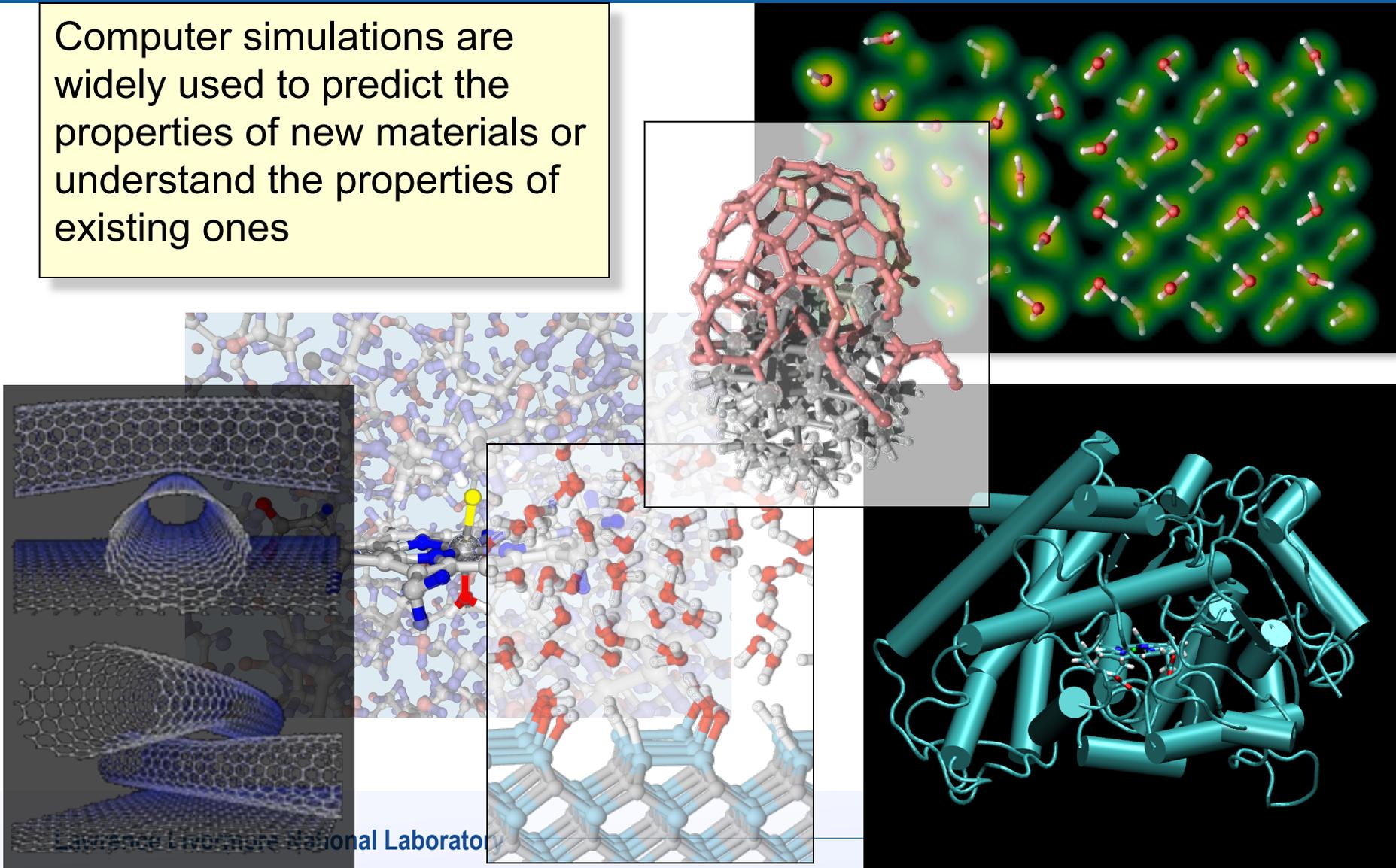
Case Study: Outline

- Problem Description
- Computational Approach
- Changes for Scaling



Computer simulations of materials

Computer simulations are widely used to predict the properties of new materials or understand the properties of existing ones



Simulation of Materials from First-Principles

First-principles methods:

Calculate properties of a given material directly from fundamental physics equations.

- **No empirical parameters**

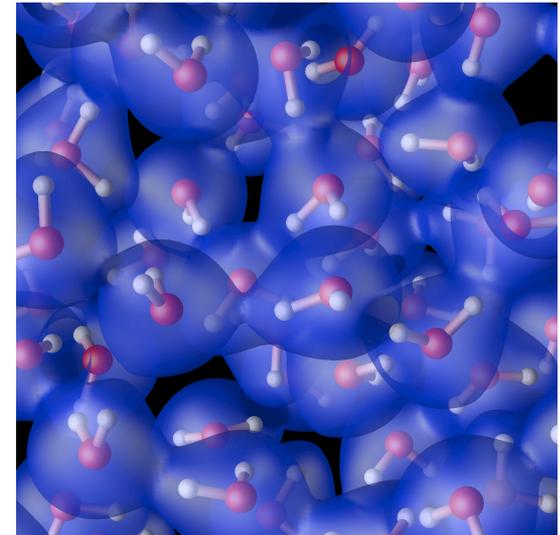
Can make predictions about complex or novel materials, under conditions where experimental data is unavailable or inconclusive.

- **Chemically dynamic**

As atoms move, chemical bonds can be formed or broken.

- **Computationally expensive**

Solving quantum equations is time-consuming, limiting systems sizes (e.g. hundreds of atoms) and simulation timescales (e.g. picoseconds)



Electron density surrounding water molecules, calculated from first-principles

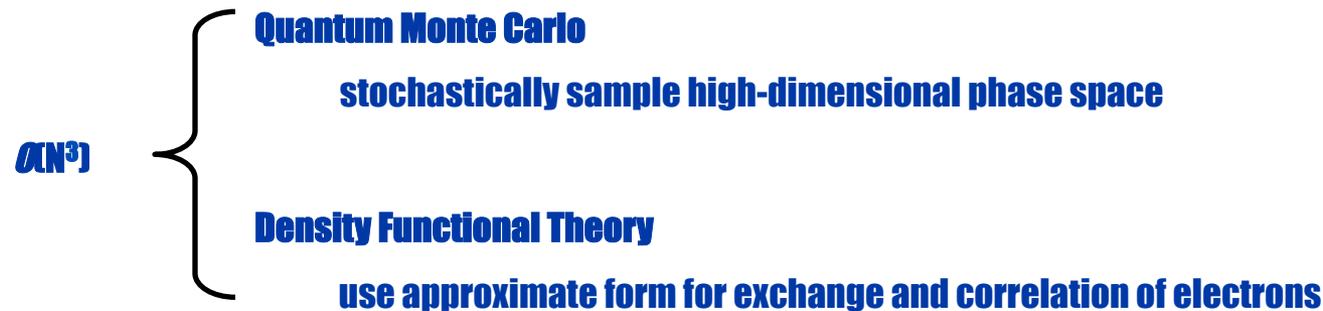
Quantum Mechanics is Hard

Properties of a many-body quantum system are given by Schrödinger's Equation:

$$\left[\frac{-\hbar^2}{2m} \nabla^2 + V_{\text{ion}}(\mathbf{r}) + V_{\text{H}}(\mathbf{r}) + V_{\text{xc}}(\mathbf{r}) \right] \psi_i(\mathbf{r}) = \varepsilon_i \psi_i(\mathbf{r})$$

Exact numerical solution has exponential complexity in the number of quantum degrees of freedom, making a brute force approach impractical (e.g. a system with 10,000 electrons would take 2.69×10^{43} times longer to solve than one with only 100 electrons!)

Approximate solutions are needed:



Both methods are $O(N^3)$, which is expensive, but tractable



Density Functional Theory

Density functional theory: total energy of system can be written as a functional of the electron density

$$E[\{\psi_i\}, \{R_I\}] = 2 \sum_i^{occ} \int d\mathbf{r} \psi_i^*(\mathbf{r}) \left(-\frac{1}{2} \nabla^2 \right) \psi_i(\mathbf{r}) + \int d\mathbf{r} \rho(\mathbf{r}) V^{ion}(\mathbf{r}) + \frac{1}{2} \sum_{J \neq I} \frac{Z_I Z_J}{|R_I - R_J|} \\ + E^{xc}[\rho] + \frac{1}{2} \int d\mathbf{r} \rho(\mathbf{r}) \int d\mathbf{r}' \frac{\rho(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|}$$

many-body effects are approximated by a density functional

exchange-correlation

Solve Kohn-Sham equations self-consistently to find the **ground state** electron density defined by the minimum of this energy functional (for a given set of ion positions)

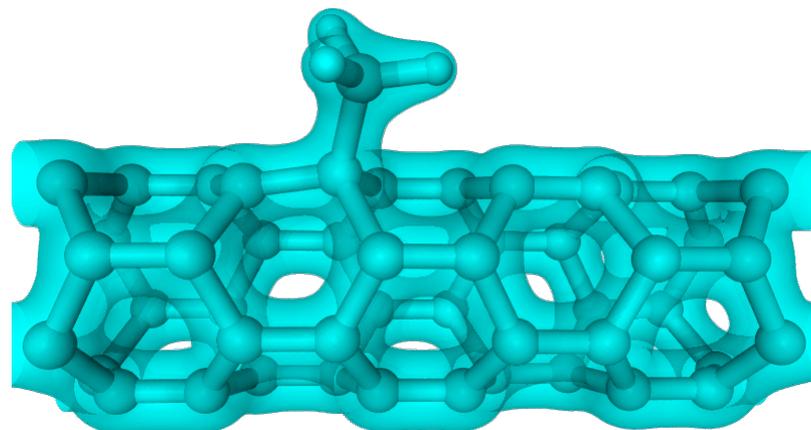
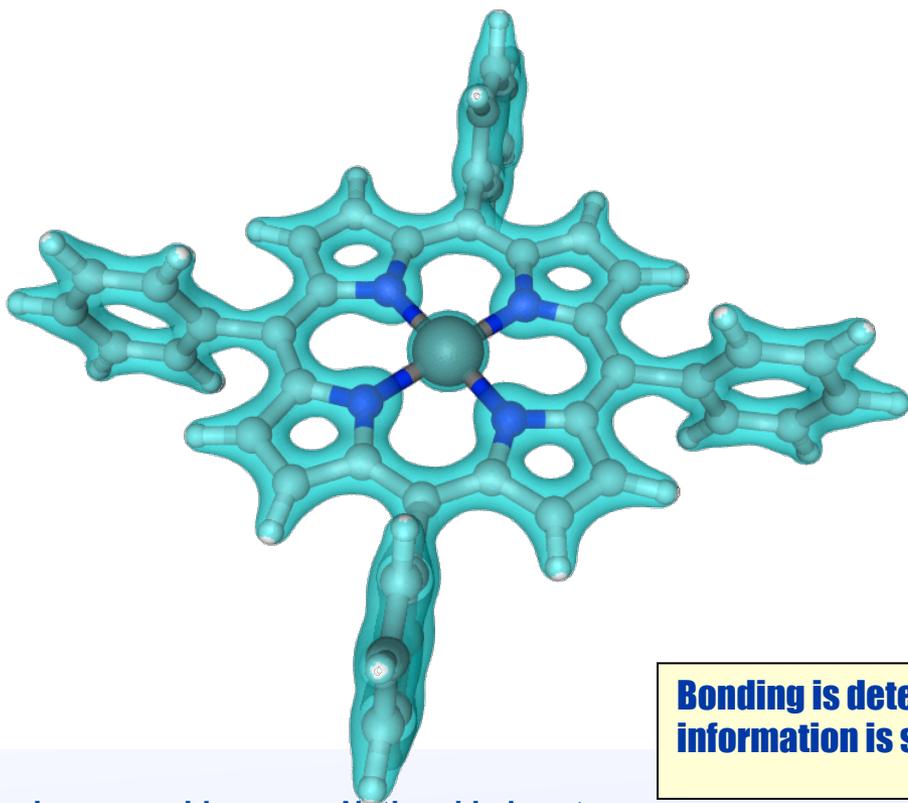
The total energy allows one to directly compare different structures and configurations



Static Calculations: Total Energy

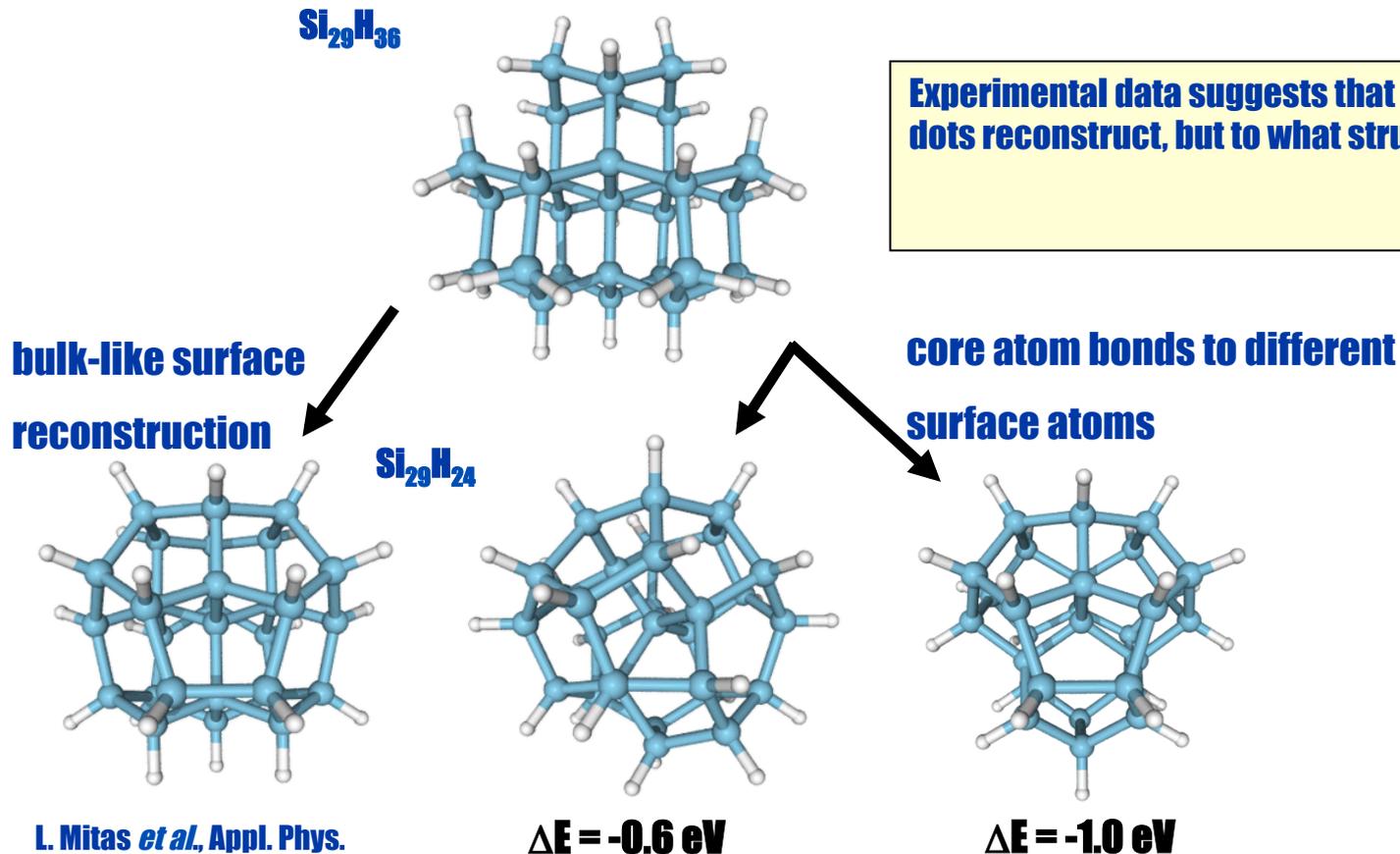
Basic Idea:

For a given configuration of atoms, we find the electronic density which gives the lowest total energy



Bonding is determined by electronic structure, no bonding information is supplied *a priori*

Energy Differences Can Predict Structure



Experimental data suggests that silicon quantum dots reconstruct, but to what structure?

L. Mitas *et al.*, *Appl. Phys. Lett.* **78**, 1918 (2001)

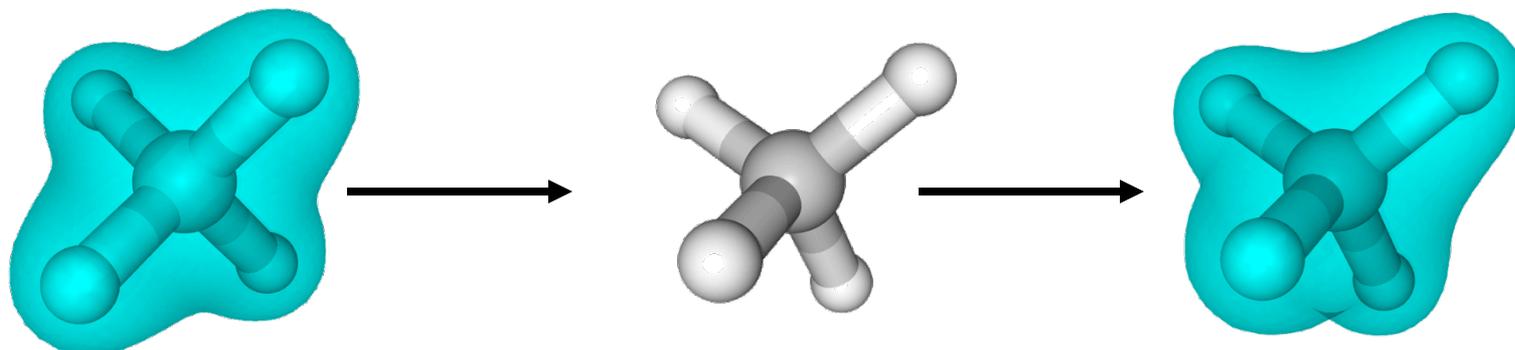
Using quantum simulation tools, we can determine the most energetically-favorable structure for a given stoichiometry.

First-Principles Molecular Dynamics (FPMD)

For dynamical properties, both ions and electrons have to move simultaneously and consistently:

- 1. Solve Schrodinger's Equation to find ground state electron density.**
- 2. Compute inter-atomic forces.**
- 3. Move atoms incrementally forward in time.**
- 4. Repeat.**

Because step 1 represents the vast majority (>99%) of the computational effort, we want to choose time steps that are as large as possible, but small enough so that previous solution is close to current one

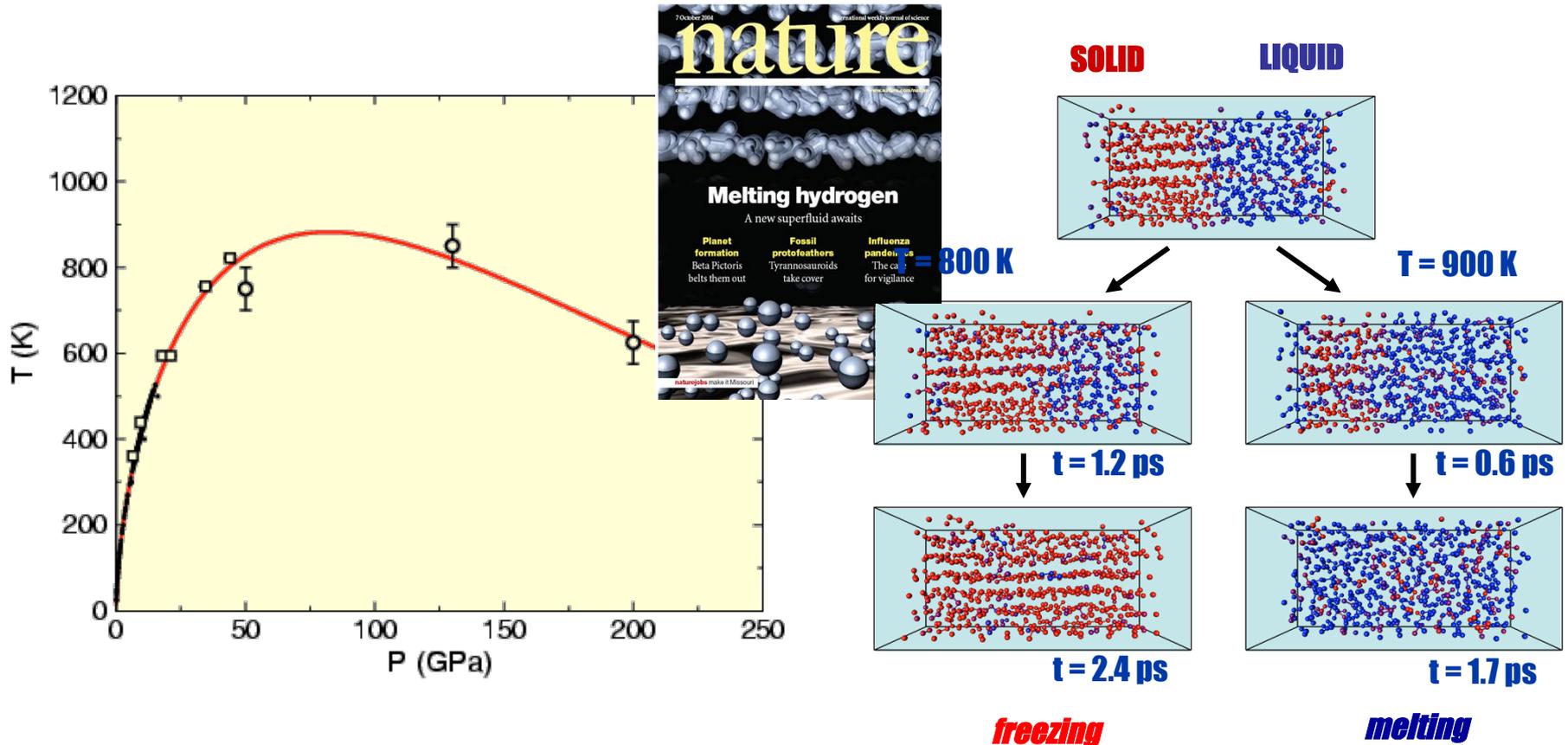


time t

move atoms

time t + dt

Predictive simulations of melting

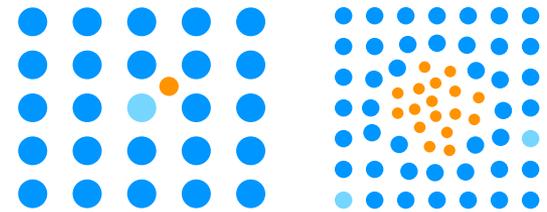


A two-phase simulation approach combined with local order analysis allows one to calculate phase behavior such as melting lines with unprecedented accuracy.

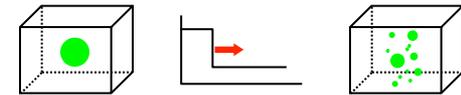
High-Z Metals from First-Principles

- A detailed theoretical understanding of high-Z metals is important for stockpile stewardship.
- Simulations may require many atoms to capture anisotropy. Dynamical properties such as melting require enough atoms to minimize finite-size effects.
- Computational cost depends on number of electrons, not atoms. A melting simulation of a high-Z metal with 10-20 valence electrons/atom will be **1000-8000 times as expensive** as hydrogen!
- The complex electronic structure of high-Z metals requires large basis sets and multiple k-points.

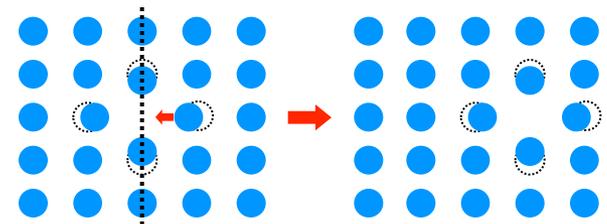
Effect of radioactive decay



Response to shocks

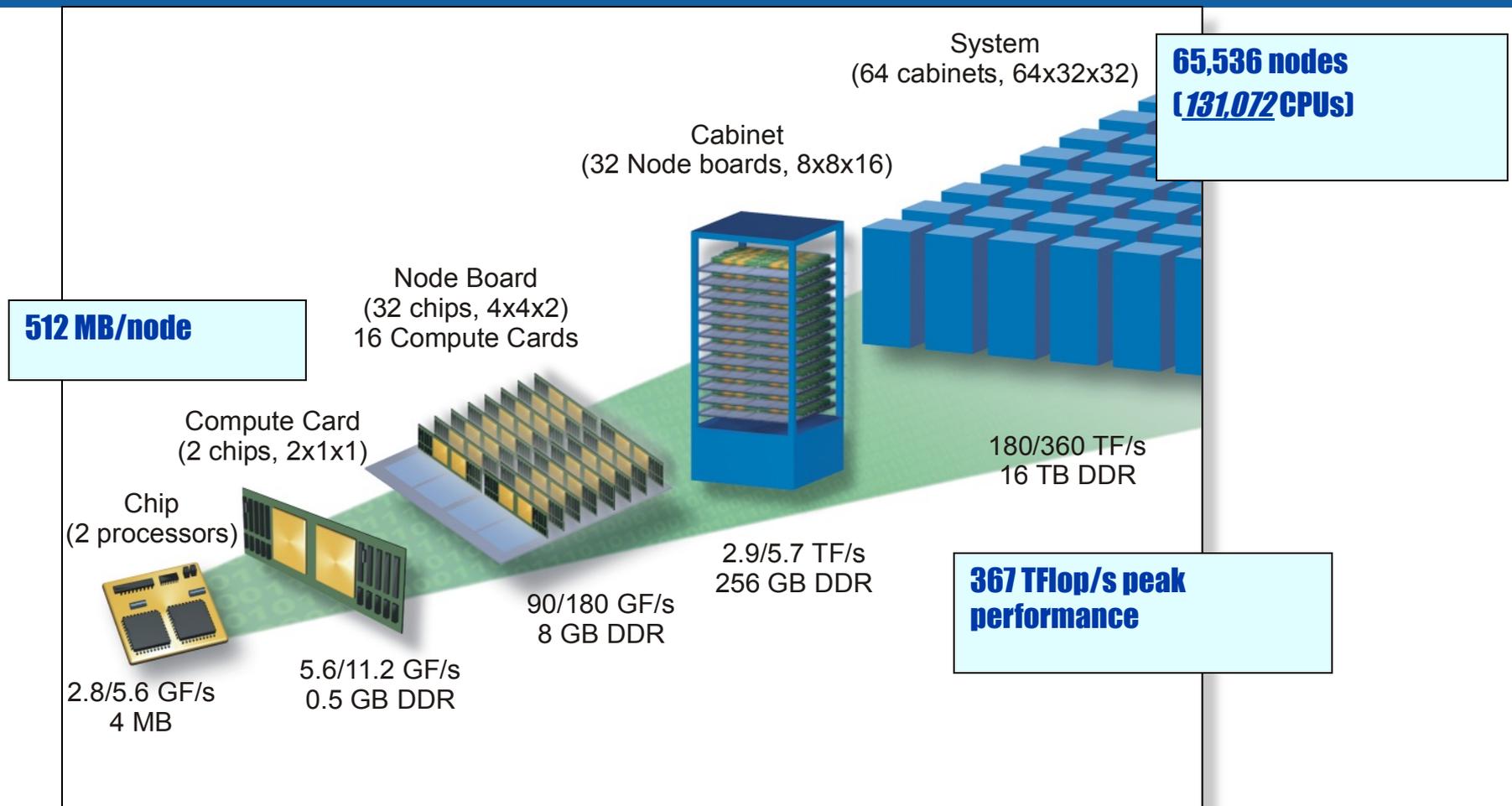


Evolution of aging damage



We need a really big computer!

The Platform: BlueGene/L



Can an FPMD code use 131,072 CPUs efficiently?

The Code: Qbox

Qbox, written by François Gygi, was designed from the ground up to be massively parallel

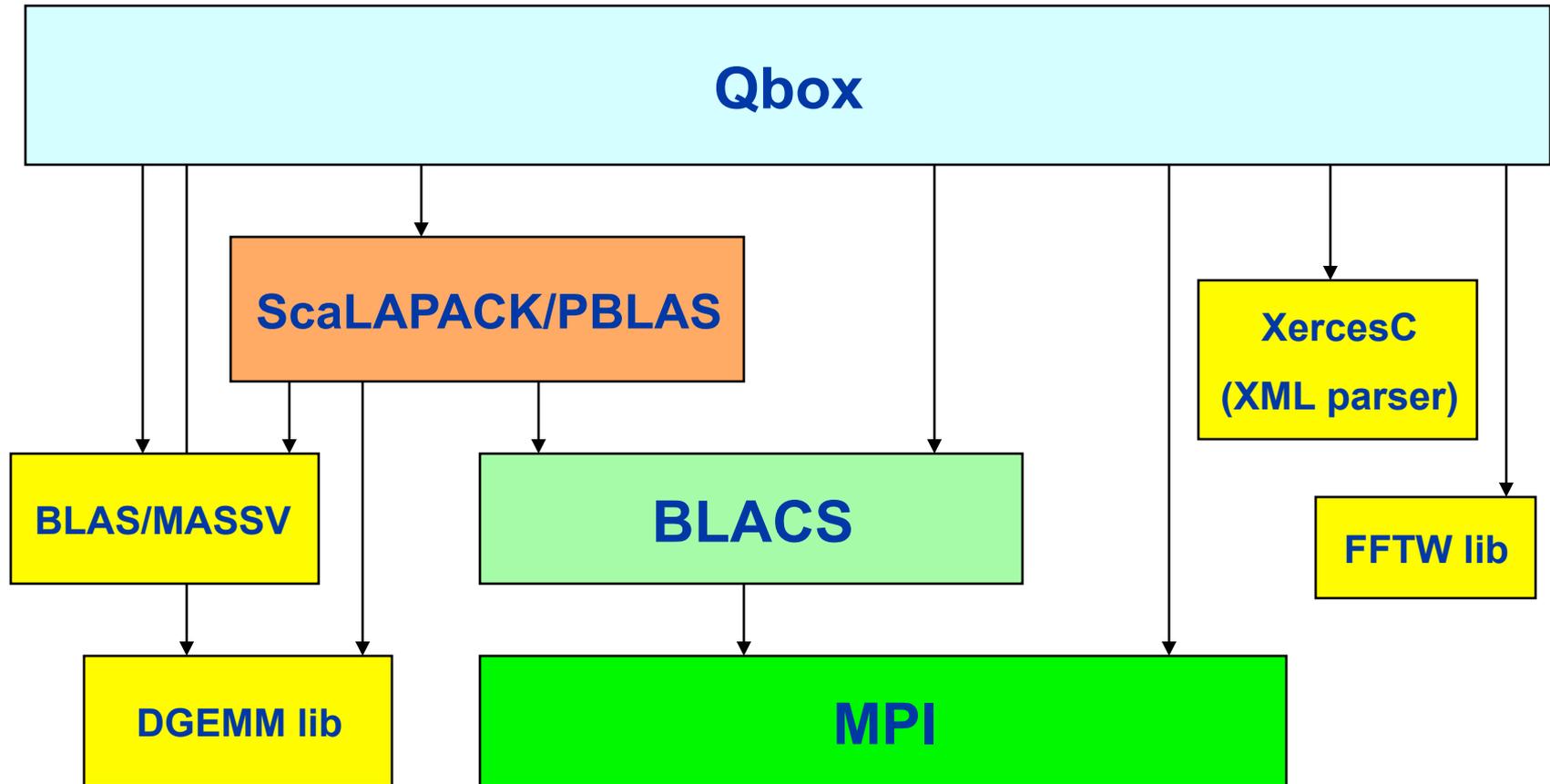
- **C++/MPI**
- **Parallelized over plane waves and electronic states**
- **Parallel linear algebra handled with ScaLAPACK/PBLAS libraries.**
- **Communication handled by BLACS library and direct MPI calls.**
- **Single-node dual-core dgemm, zgemm optimized by John Gunnels, IBM**
- **Uses custom distributed 3D complex Fast Fourier Transforms**

Implementation details:

- **Wave function described with a plane wave basis**
- **Periodic boundary conditions**
- **Pseudopotentials are used to represent electrons in atomic core, and thereby reduce the total number of electrons in the simulation.**



Qbox Code Structure



Parallel Solution of Kohn-Sham Equations

How to efficiently distribute problem over tens of thousands of processors?

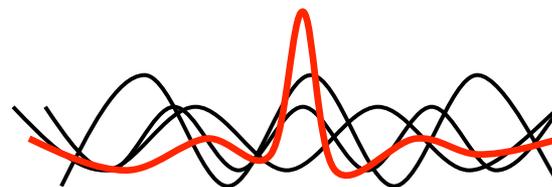
Kohn-Sham equations

- solutions represent molecular orbitals (one per electron)
- molecular orbitals are complex scalar functions in \mathbb{R}^3
- coupled, non-linear PDEs
- periodic boundary conditions

$$\left\{ \begin{array}{l} -\Delta\varphi_i + V(\rho, \mathbf{r})\varphi_i = \varepsilon_i\varphi_i \quad i = 1 \dots N_{\text{el}} \\ V(\rho, \mathbf{r}) = V_{\text{ion}}(\mathbf{r}) + \int \frac{\rho(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} d\mathbf{r}' + V_{\text{XC}}(\rho(\mathbf{r}), \nabla\rho(\mathbf{r})) \\ \rho(\mathbf{r}) = \sum_{i=1}^{N_{\text{el}}} |\varphi_i(\mathbf{r})|^2 \\ \int \varphi_i^*(\mathbf{r}) \varphi_j(\mathbf{r}) d\mathbf{r} = \delta_{ij} \end{array} \right.$$

We use a plane-wave basis for orbitals:

$$\varphi_{\mathbf{k},n}(\mathbf{r}) = \sum_{|\mathbf{k}+\mathbf{q}|^2 < E_{\text{cut}}} c_{\mathbf{k}+\mathbf{q},n} e^{i\mathbf{q}\cdot\mathbf{r}}$$



Represent function as a Fourier series

Careful distribution of wave function is key to achieving good parallel efficiency

Algorithms used in FPMD

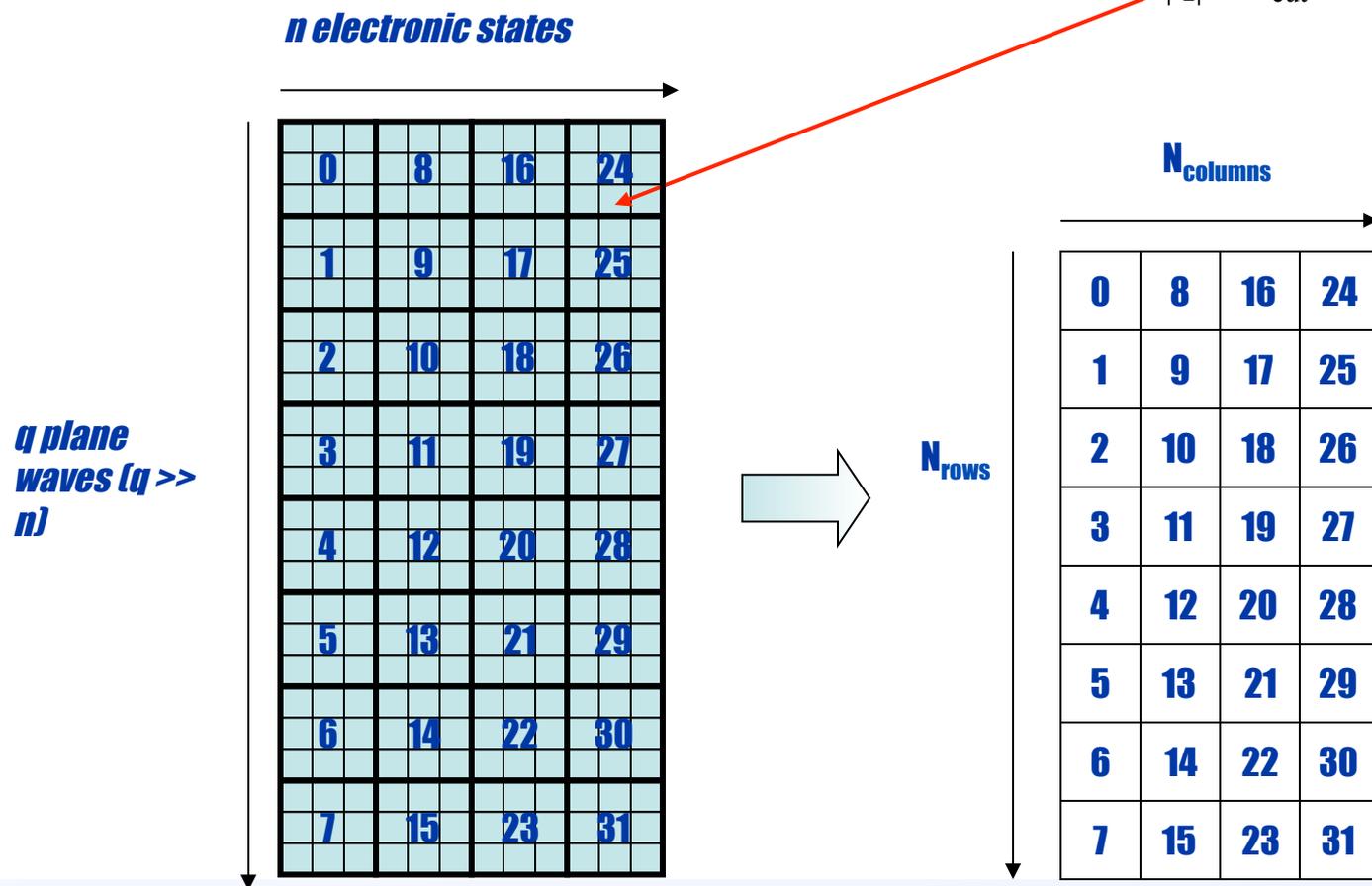
$$\left\{ \begin{array}{l} -\Delta\varphi_i + V(\rho, \mathbf{r})\varphi_i = \varepsilon_i\varphi_i \quad i = 1 \dots N_{\text{el}} \\ V(\rho, \mathbf{r}) = V_{\text{ion}}(\mathbf{r}) + \int \frac{\rho(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} d\mathbf{r}' + V_{\text{XC}}(\rho(\mathbf{r}), \nabla\rho(\mathbf{r})) \\ \rho(\mathbf{r}) = \sum_{i=1}^{N_{\text{el}}} |\varphi_i(\mathbf{r})|^2 \\ \int \varphi_i^*(\mathbf{r}) \varphi_j(\mathbf{r}) d\mathbf{r} = \delta_{ij} \end{array} \right.$$

- Solving the KS equations: a constrained optimization problem in the space of coefficients c_{qn}
- Poisson equation: 3-D FFTs
- Computation of the electronic charge: 3-D FFTs
- Orthogonality constraints require dense, complex linear algebra (e.g. $A = C^H C$)

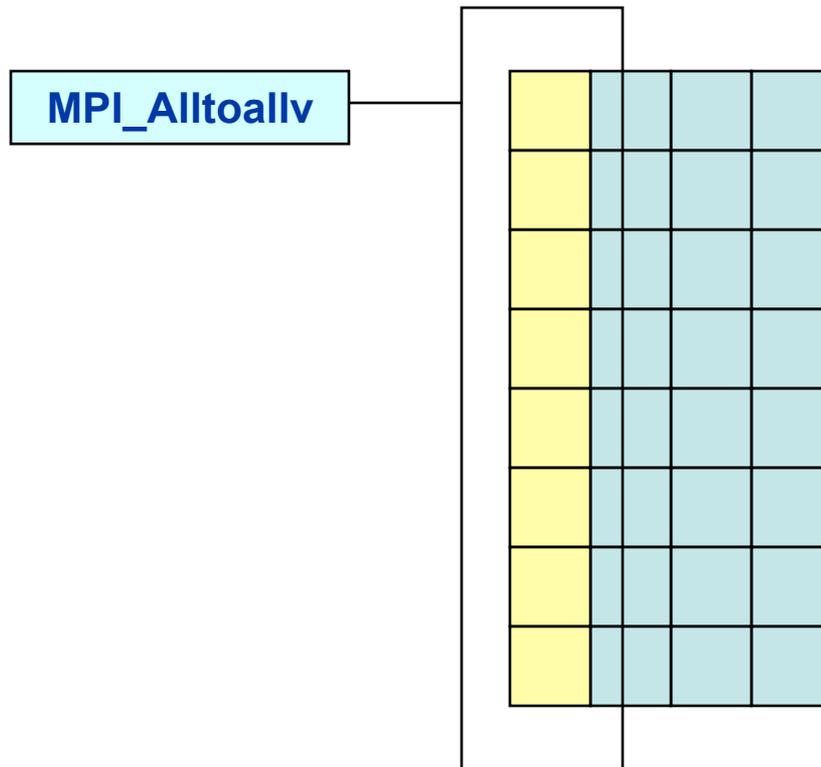
Overall cost is $O(N^3)$ for N electrons

The matrix of coefficients $c_{q,n}$ is block distributed (ScaLAPACK data layout)

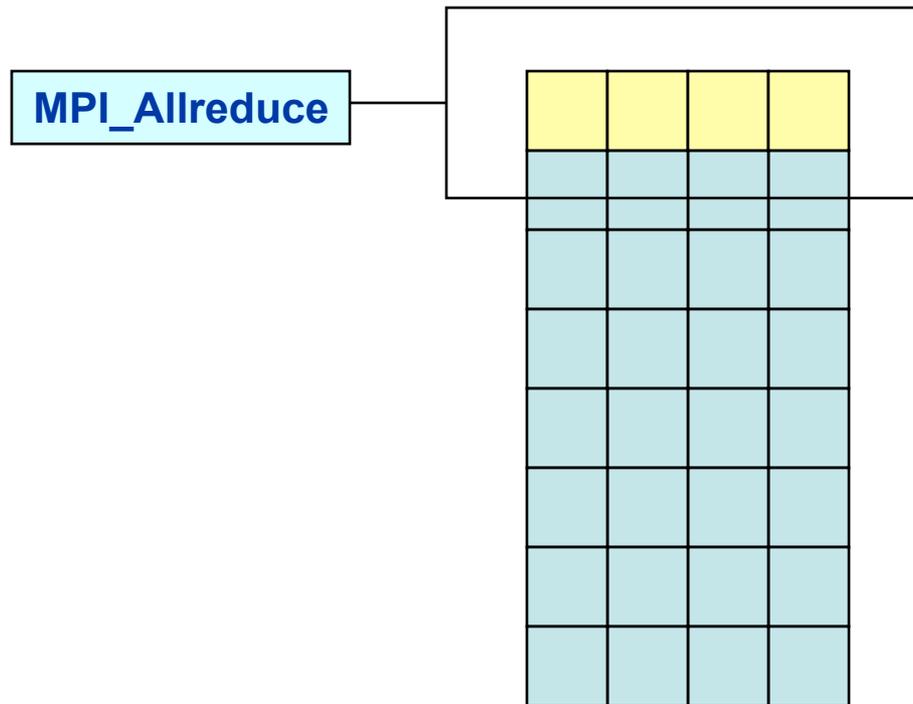
$$\varphi_n(\mathbf{r}) = \sum_{|\mathbf{q}|^2 < E_{\text{cut}}} c_{q,n} e^{i\mathbf{q}\cdot\mathbf{r}}$$



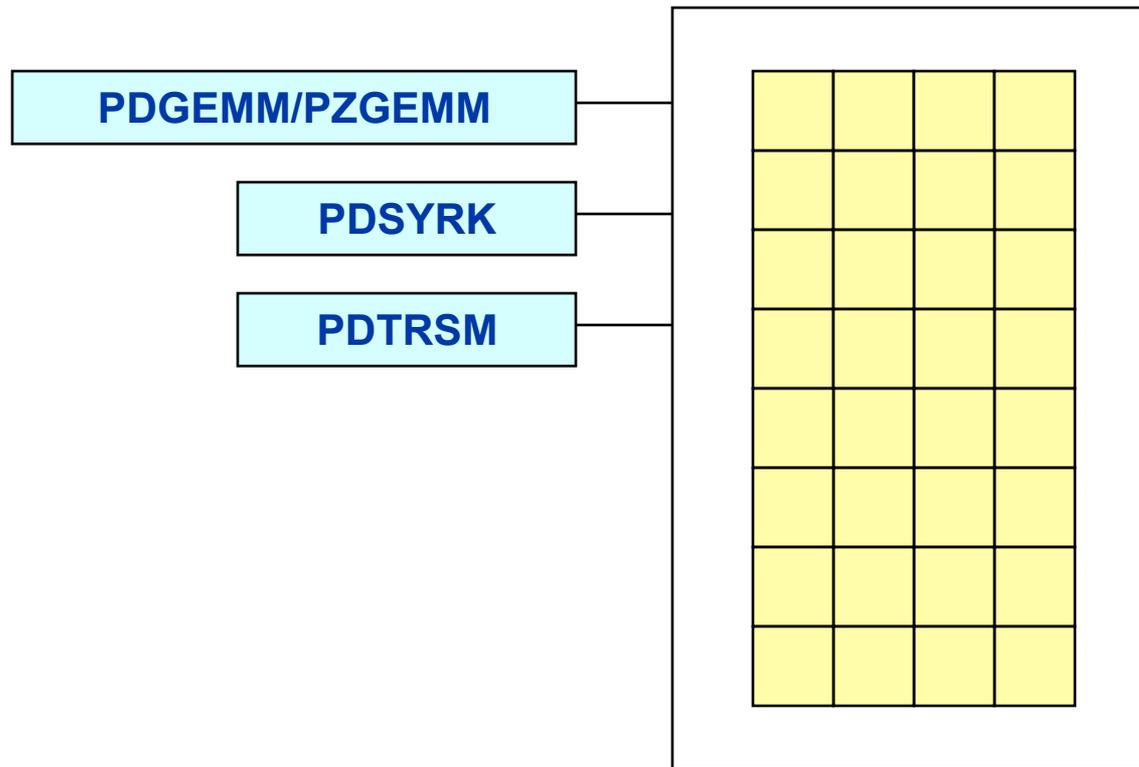
- Computation of 3-D Fourier transforms



- Accumulation of electronic charge density



- ScaLAPACK dense linear algebra operations



Controlling Numerical Errors

- For systems with complex electronic structure like high-Z metals, the desired high accuracy is achievable only through careful control of all numerical errors.

- Numerical errors which must be controlled:

- Convergence of Fourier series

$$\varphi_n(\mathbf{r}) = \sum_{|\mathbf{q}|^2 < E_{\text{cut}}} c_{\mathbf{q},n} e^{i\mathbf{q}\cdot\mathbf{r}}$$

- Convergence of system size (number of atoms)

- Convergence of k-space integration

$$\rho(\mathbf{r}) = \int_{BZ} |\psi_{\mathbf{k},n}(\mathbf{r})|^2 d^3\mathbf{k}$$

- We need to systematically increase

- Plane-wave energy cutoff

- Number of atoms

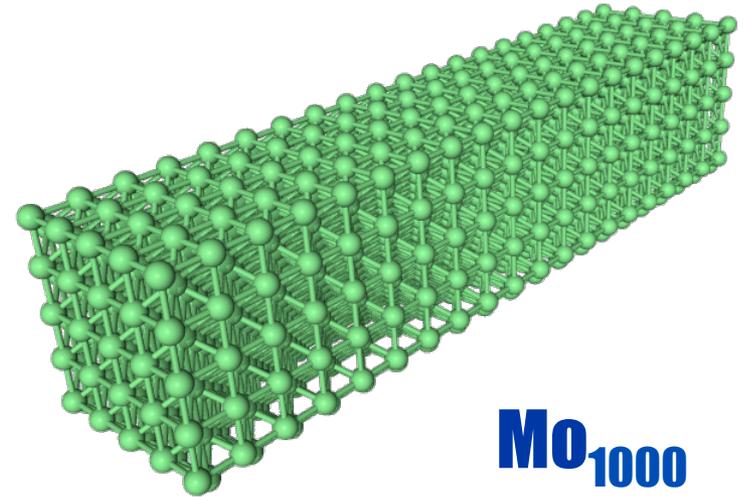
- Number of k-points in the Brillouin zone integration

Let's start with these

BlueGene/L allows us to ensure convergence of all three approximations

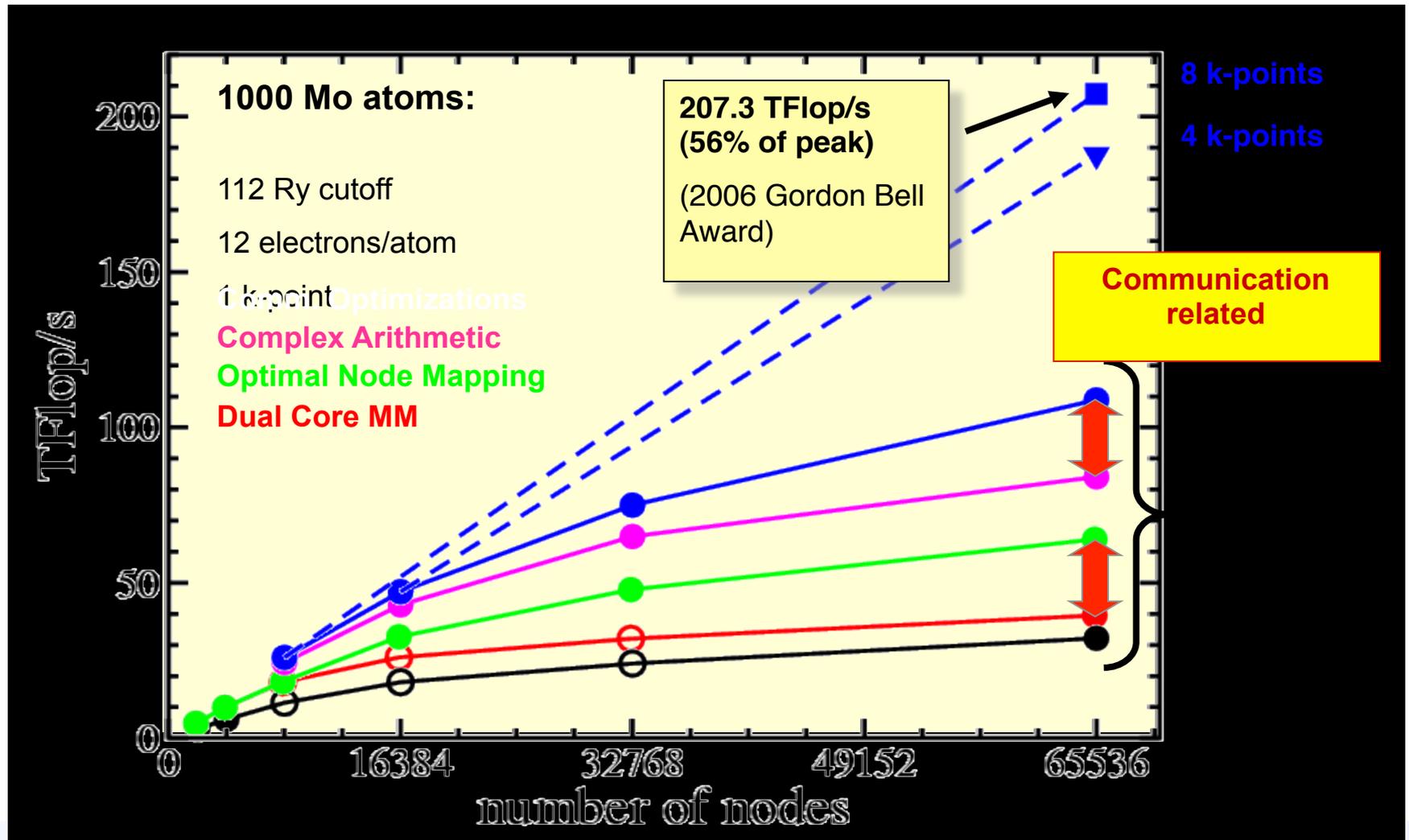
High-Z test problem: Mo_{1000}

- **Electronic structure of a 1000-atom Molybdenum sample**
- **12,000 electrons**
- **32 non-local projectors for pseudopotentials**
- **112 Ry plane-wave energy cutoff**
- **High-accuracy parameters**



We wanted a test problem which was representative of the next generation of high-Z materials simulations while at the same time straightforward for others to replicate and compare performance

QBox Performance



Performance measurements

- We use the PPC440 HW performance counters
- Access the HW counters using the APC library (J. Sexton, IBM)
 - Provides a summary file for each task
- Not all double FPU operations can be counted
 - DFPU fused multiply-add: ok
 - DFPU add/sub: not counted
- FP operations on the second core are not counted

Performance measurements

- Using a single-core DGEMM/ZGEMM library
 - Measurements are done in three steps:
 - 1) count FP ops without using the DFPU
 - 2) measure timings using the DFPU
 - 3) compute flop rate using 1) and 2)
- Problem: some libraries use the DFPU and are not under our control
- DGEMM/ZGEMM uses mostly fused multiply-add ops
- In practice, we use 2). Some DFPU add/sub are not counted

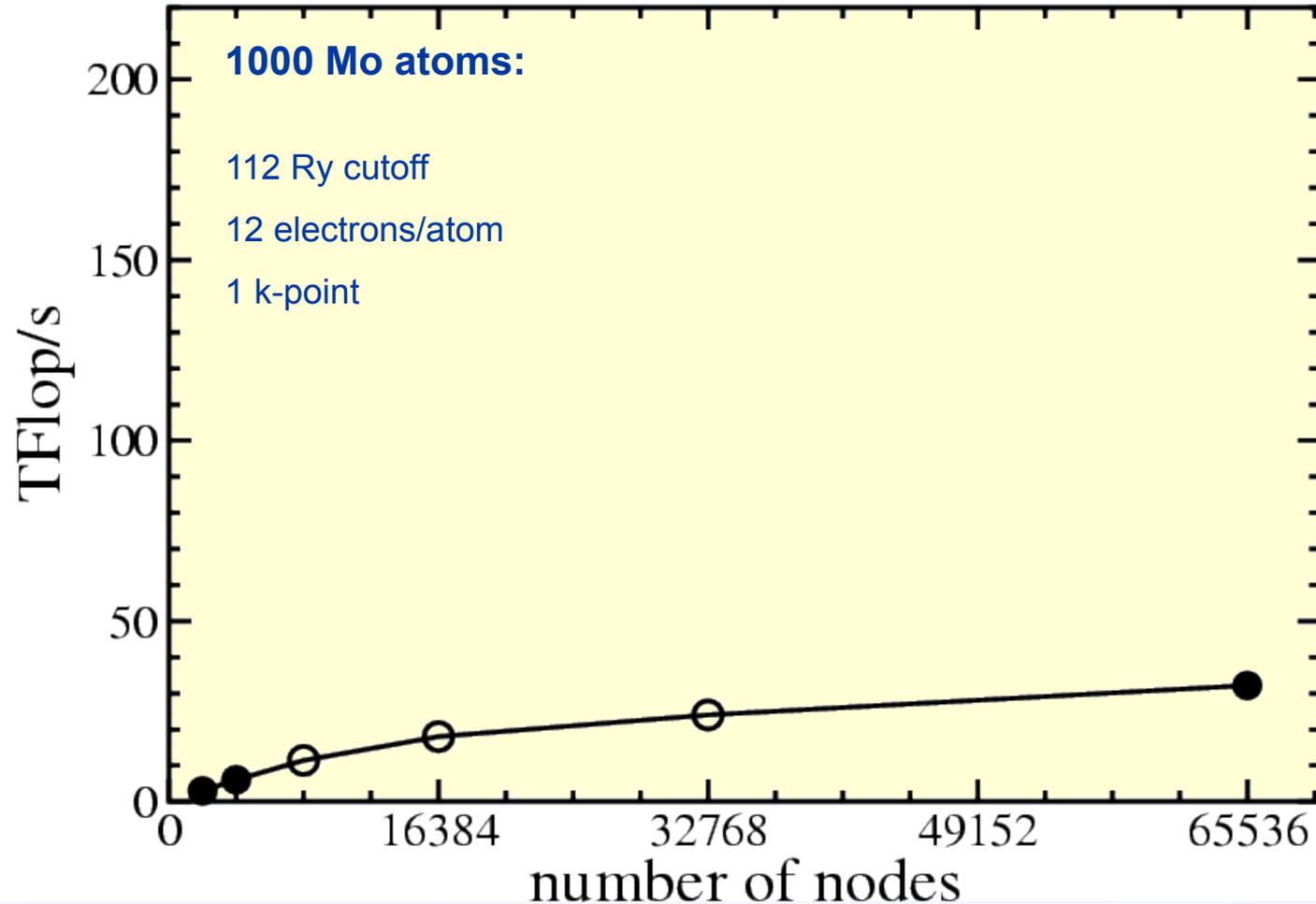
Performance measurements

- Using a dual-core DGEMM/ZGEMM library:
 - FP operations on the second core are not counted
 - In this case, we *must* use a 3 step approach:
 - 1) count FP ops using the single-core library
 - 2) measure timing using the dual-core library
 - 3) compute the flop rate using 1) and 2)

Our performance numbers represent a strict lower bound of the actual performance

Mo₁₀₀₀ scaling results

sustained performance



Single-node kernels

Exploiting the BG/L hardware

- Use double FPU instructions (“double hummer”)
- Use both CPUs on the node
 - use virtual node mode, *or*
 - program for two cores (not L1 coherent)
- We use BG/L in co-processor mode
 - 1 MPI task per node
 - Use second core using dual-core kernels

DGEMM/ZGEMM kernel (John Gunnels, IBM)

- Hand optimized, uses double FPU very efficiently
- Algorithm tailored to make best use of L1/L2/L3
- Dual-core version available: uses all 4 FPUs on the node

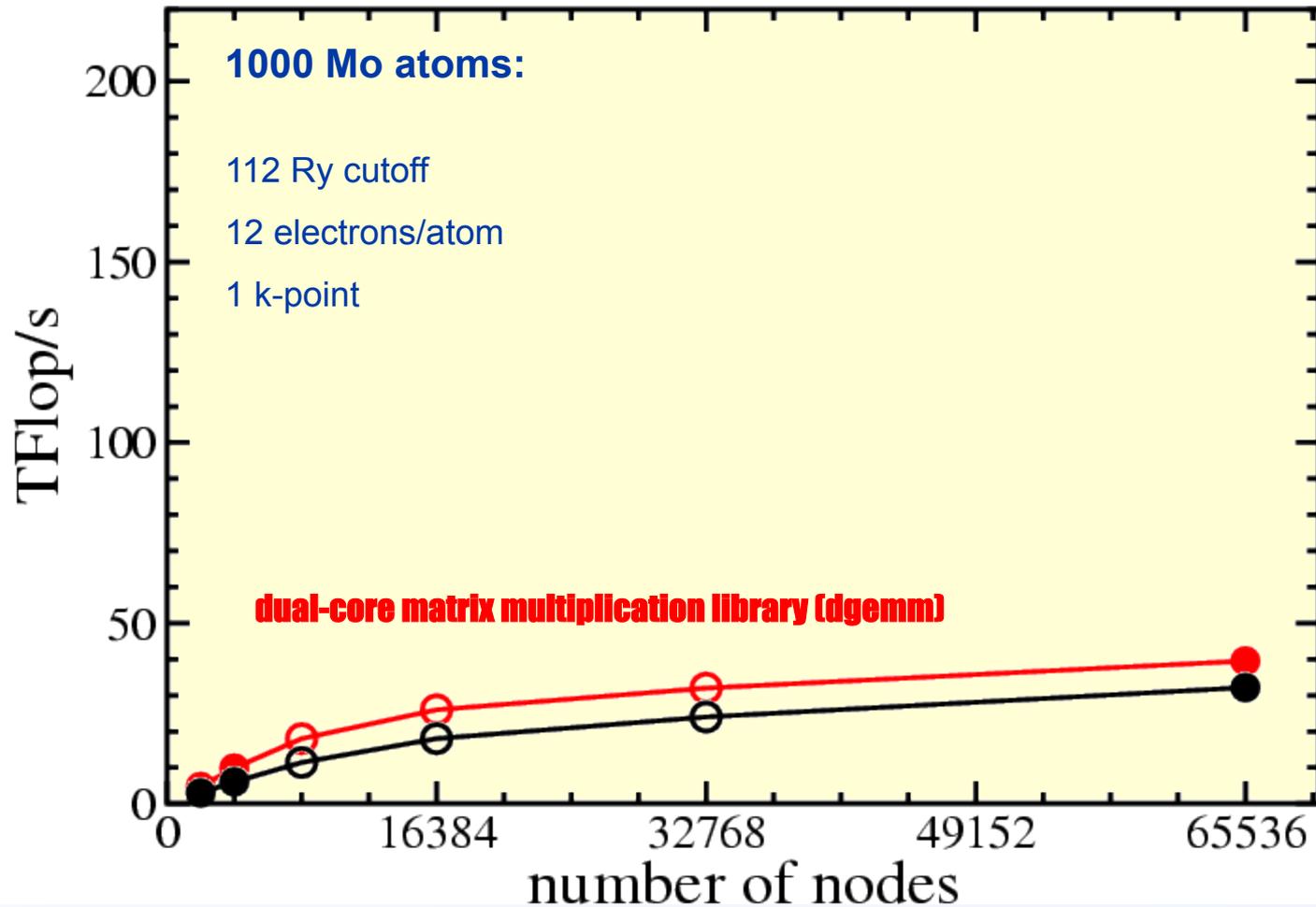
FFTW kernel (Technical University of Vienna)

- Uses hand-coded intrinsics for DFPU instructions



Mo₁₀₀₀ scaling results

sustained performance



Need for Scalable Tools

- Support complete development cycle
 - Debugging
 - Performance analysis
 - Optimization/transformation
- New challenges with scalability
 - Large volumes of data to store and analysis
 - Central processing/control infeasible
 - Light-weight kernel
- New tool strategies
 - Scalable infrastructures
 - Application specific tool support
 - Flexible and interoperable toolboxes



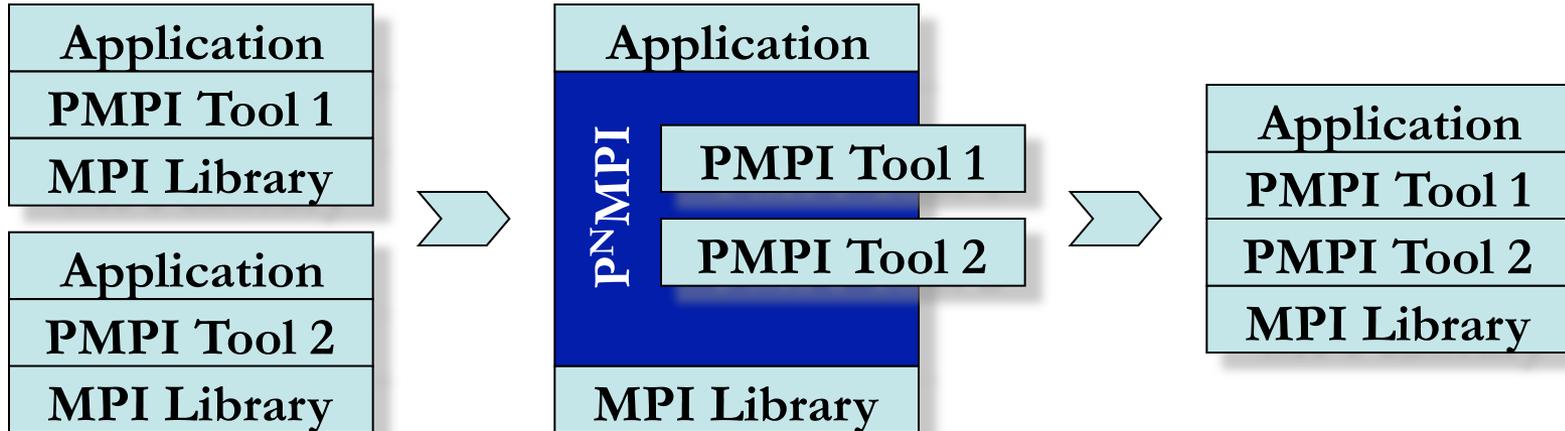


Assembling Application Specific Tool Prototypes

- Tools need to react to specific application requirements
 - Limited use for monolithic tools with their own stacks
 - New and unprecedented scenarios
 - Limit data collection and specialize data analysis
- Example: Dynamic tool assembly with PⁿMPI

Virtualizing MPI Tools

- MPI tools based on the PMPI interface
 - Intercept arbitrary MPI calls using tool wrappers
 - Transparently track modify, or replace MPI calls

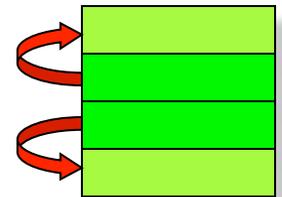
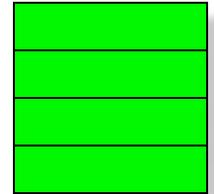


- P^NMPI: Dynamically stack multiple (binary PMPI) tools
 - Provide generic tool wrappers
 - Transparently modify MPI calls

```
module mpitrace
module mpiprofile
```

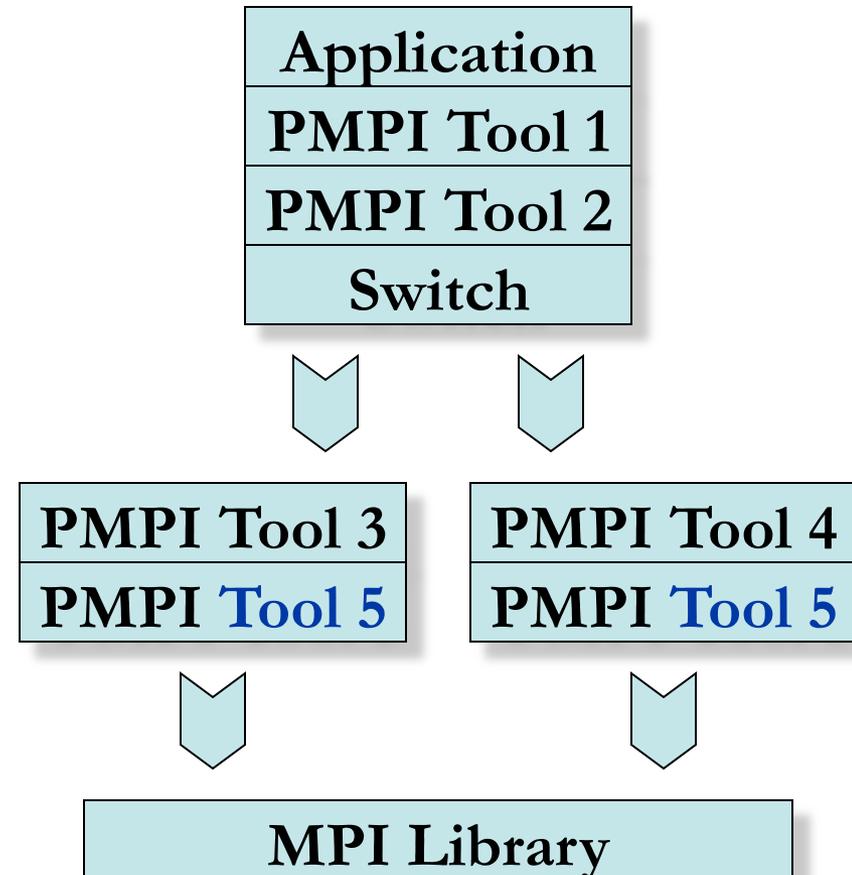
Usage Scenarios

- Concurrent execution of transparent tools
 - Tracing and profiling
 - Message perturbation and MPI Checker
- Tool cooperation
 - Encapsulate common tool operations
 - Publish/Subscribe interface for services
 - E.g.: datatype walking, request tracking
- Tool multiplexing
 - Apply tools to subsets of applications
 - Run concurrent copies of the same tool



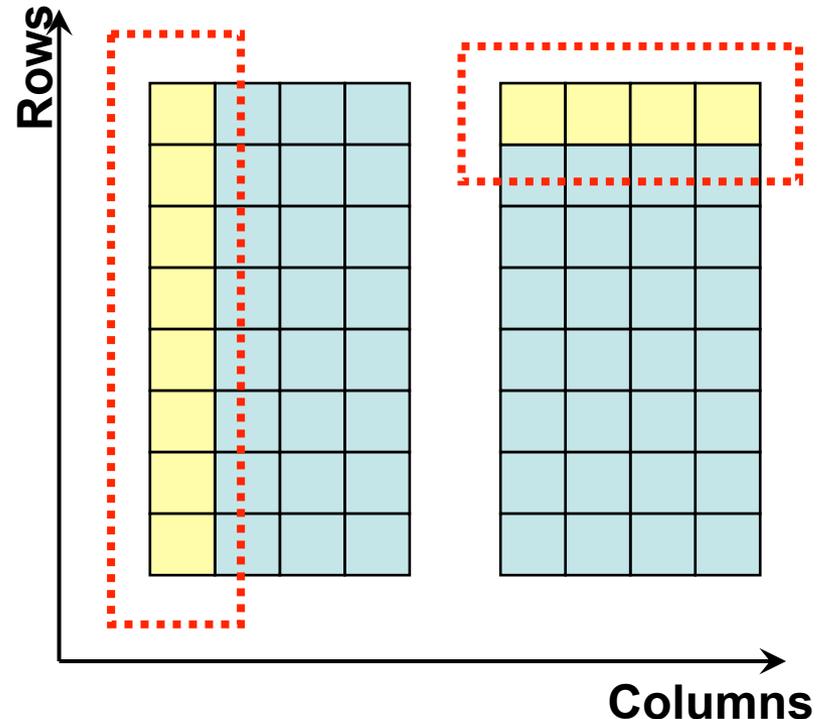
P^NMPI Switch Modules

- Multiple tool stacks
 - Defined independently
 - Initial tool stack called by application
- Switch modules
 - Dynamic stack choice
 - Based on arguments or dynamic steering
- Duplication of tools
 - Multiple contexts
 - Separate global state



Distinguishing Communicators

- Example: dense matrix
 - Row and column communicators
 - Global operations
- Need to profile separately
 - Different operations
 - Separate optimization
- Switch module to split communication (111 lines of code)
 - Create three independent tool stacks
 - Apply unmodified profiler (mpiP) in each stack



Profiling Setup

- Configuration file:

```
module commsize-switch  
argument sizes 8 4  
argument stacks column row  
module mpiP
```

Default
Stack

Target
Stack 1

Target
Stack 2

```
stack row  
module mpiP1
```

```
stack column  
module mpiP2
```

Switch Module

Arguments
controlling
switch module

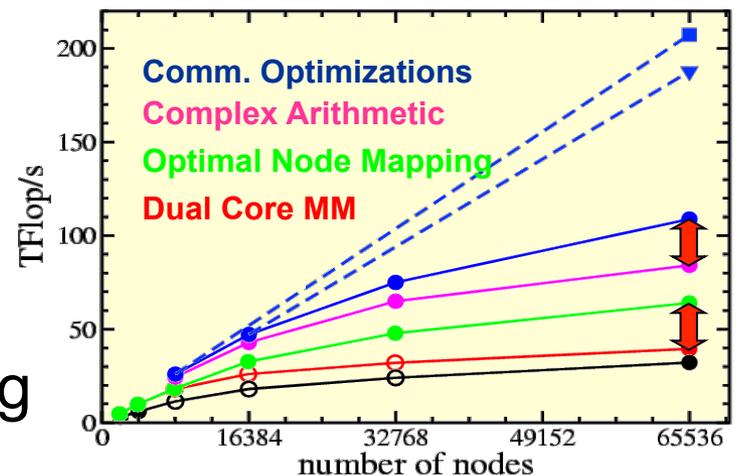
Multiple profiling
instances

Communicator Profiling Results for FPMD Code

Operation	Sum
Send	317245
Allreduce	319028
Alltoallv	471488
Recv	379265
Bcast	401042

- Information helpful for ...
 - Understanding behavior
 - Locating optimization targets
 - Optimizing of collectives
 - Identifying better node mapping

AMD Opteron/Infiniband Cluster



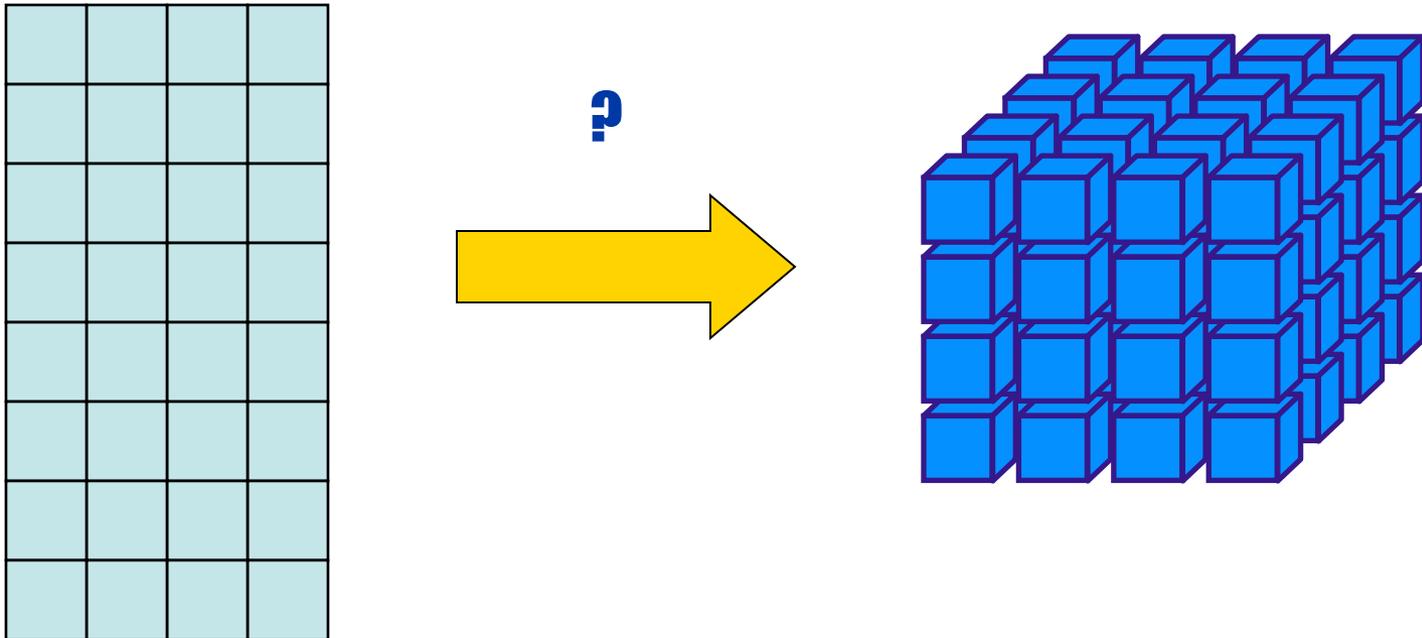
Conclusions

- System size growing towards Petascale
 - Tools must scale with systems and codes
 - New concepts and infrastructures necessary
- Scalable debugging with STAT
 - Lightweight tool to narrow search space
 - Tree-based stack trace aggregation
- Dynamic MPI tool creation with P^NMPI
 - Ability to quickly create application specific tools
 - Transparently reuse and extend existing tools
- Tool interoperability increasingly important



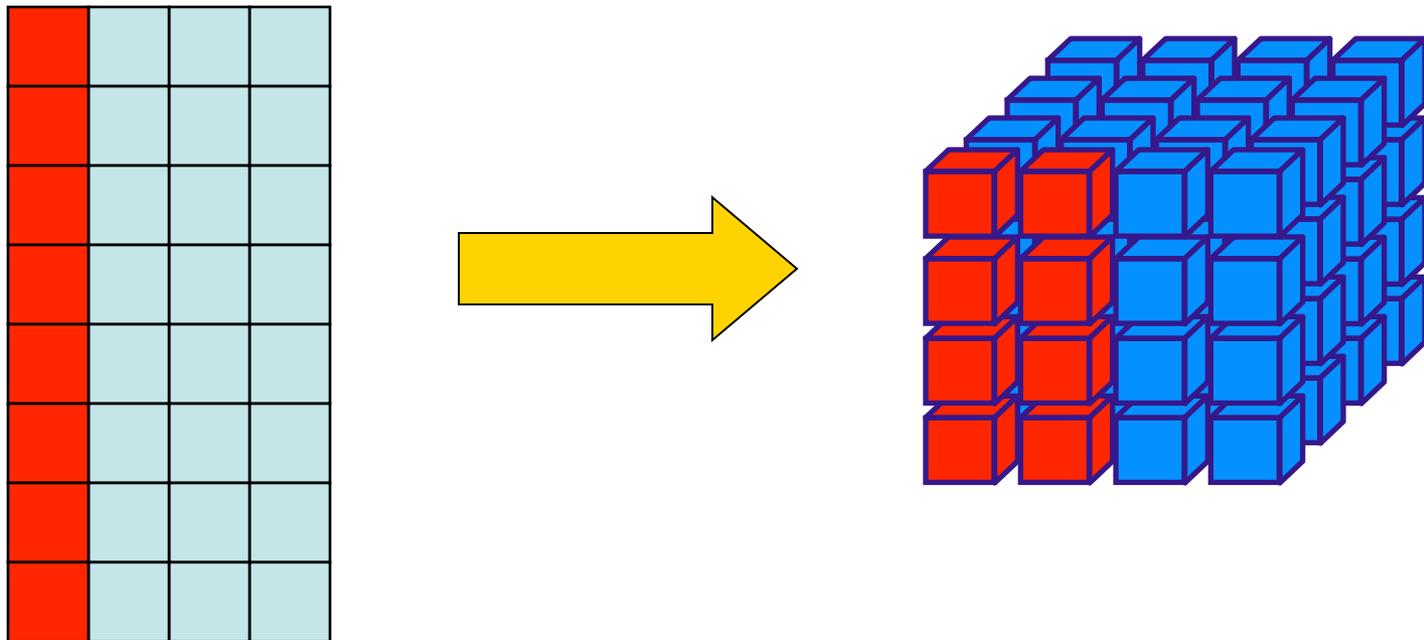
Mapping tasks to physical nodes

BG/L is a 3-dimensional torus of processors. The physical placement of MPI tasks can thus be expected to have a significant effect on communication times



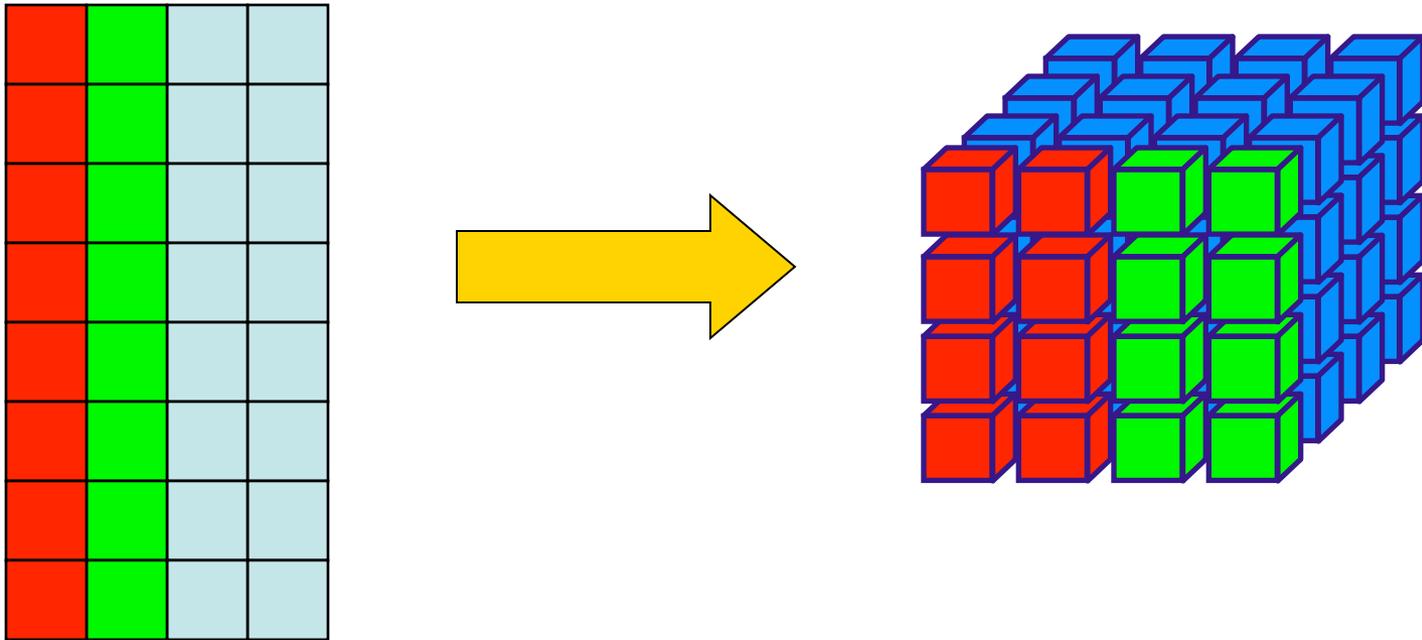
Mapping tasks to physical nodes

BG/L is a 3-dimensional torus of processors. The physical placement of MPI tasks can thus be expected to have a significant effect on communication times

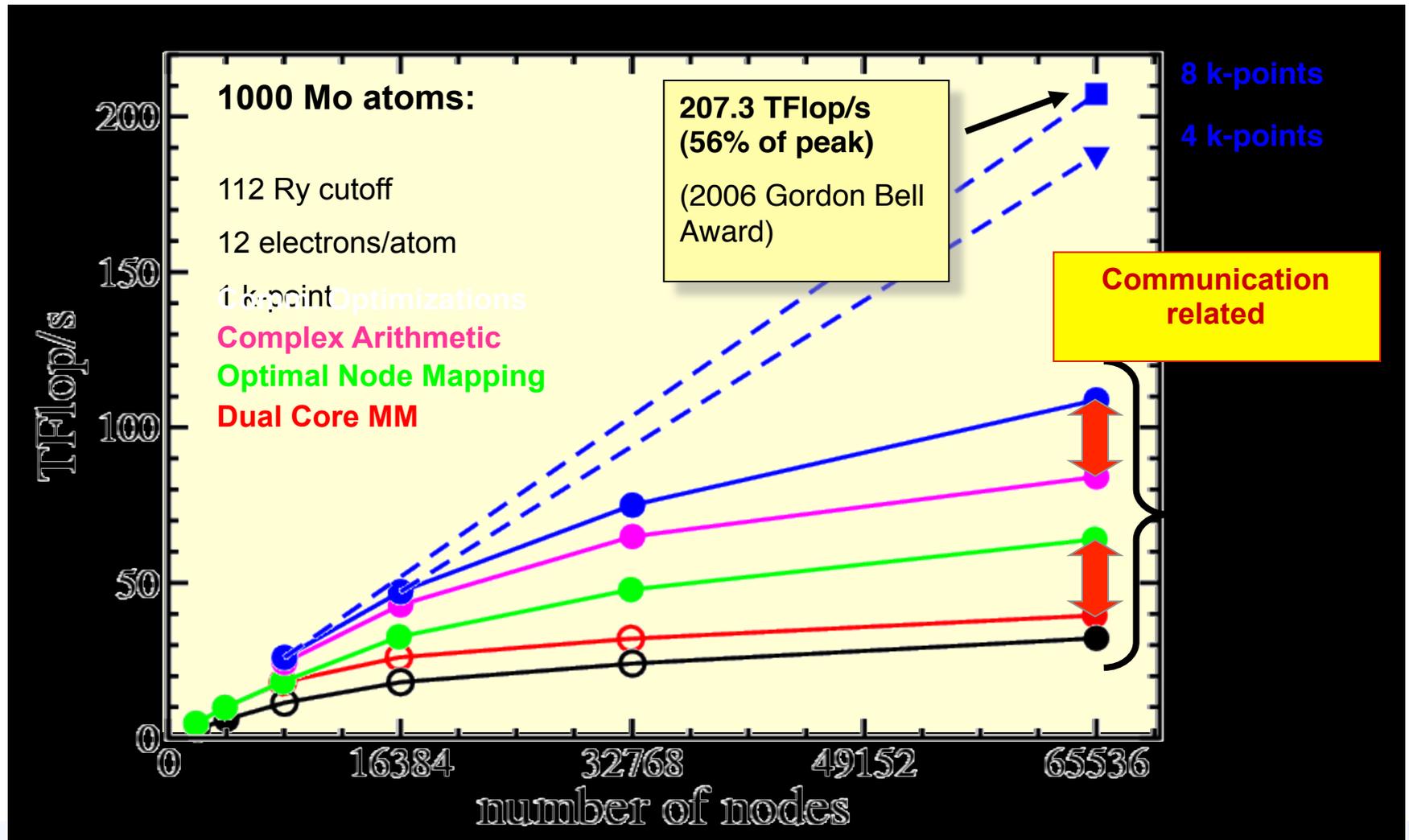


Mapping tasks to physical nodes

BG/L is a 3-dimensional torus of processors. The physical placement of MPI tasks can thus be expected to have a significant effect on communication times

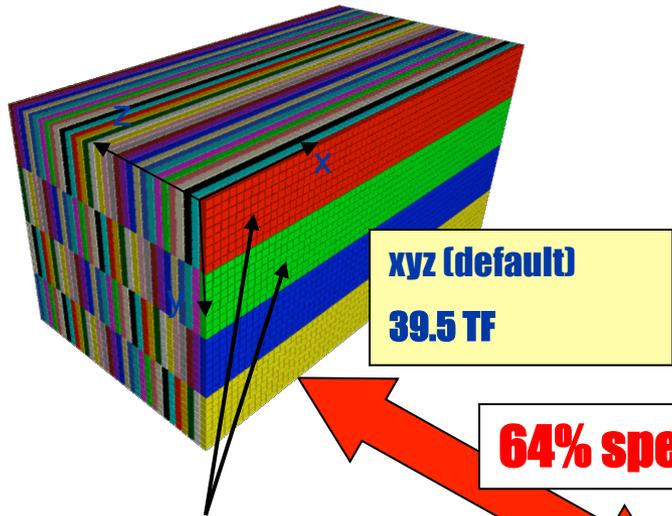


QBox Performance



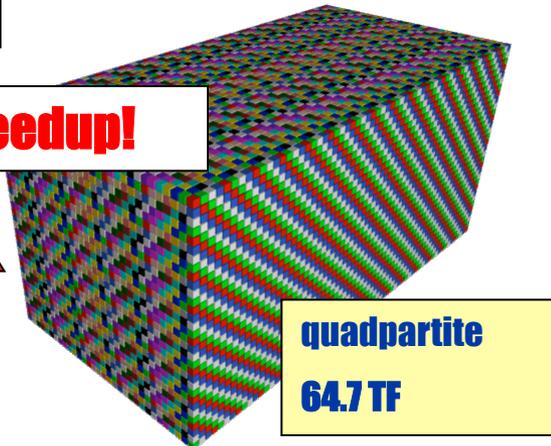
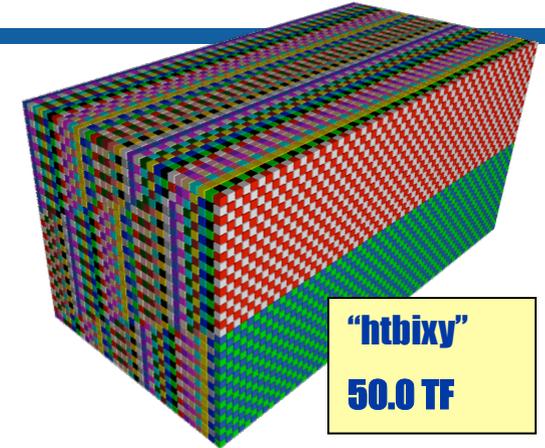
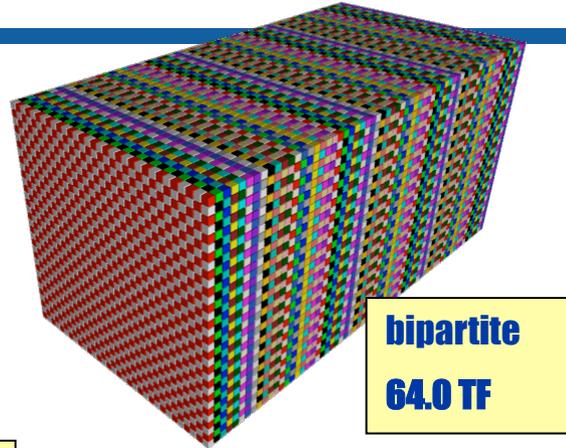
BG/L node mapping

65536 nodes, in a
64x32x32 torus



512 tasks per MPI
subcommunicator

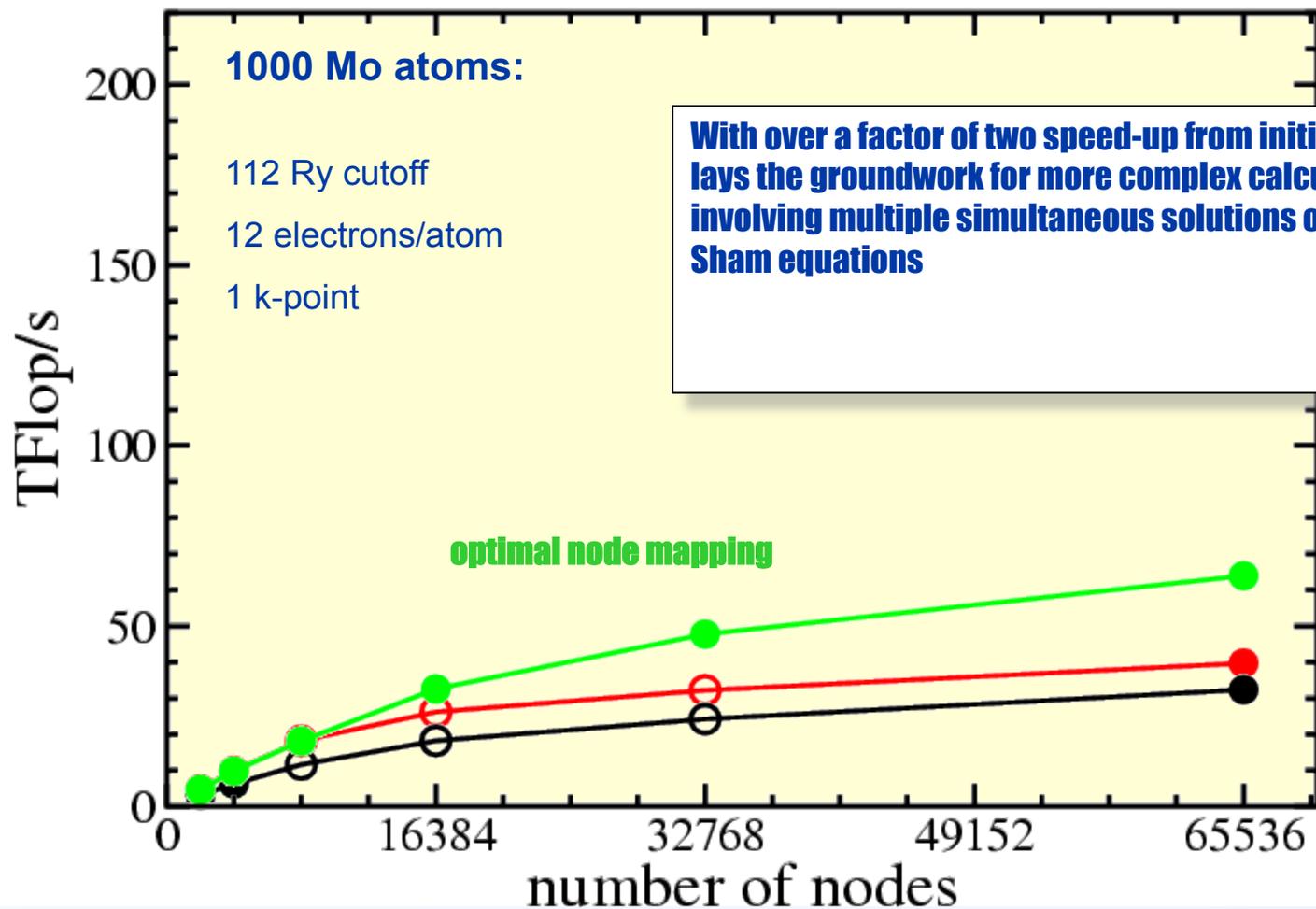
64% speedup!



Physical task distribution can significantly affect performance

Mo₁₀₀₀ scaling results

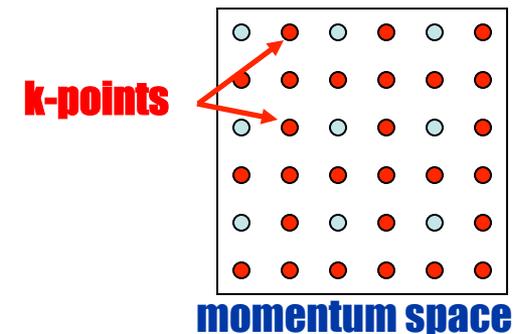
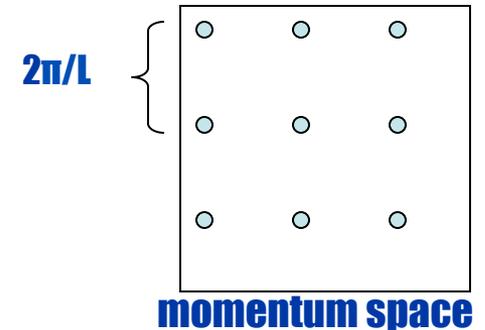
sustained performance



High symmetry requires k-point sampling

Crystalline systems require explicit sampling of the Brillouin zone to accurately represent electrons:

- Typically, a Monkhorst-Pack grid is used, with symmetry-equivalent points combined.
- each wave vector (“k-point”) requires a separate solution of Schrödinger’s Equation.

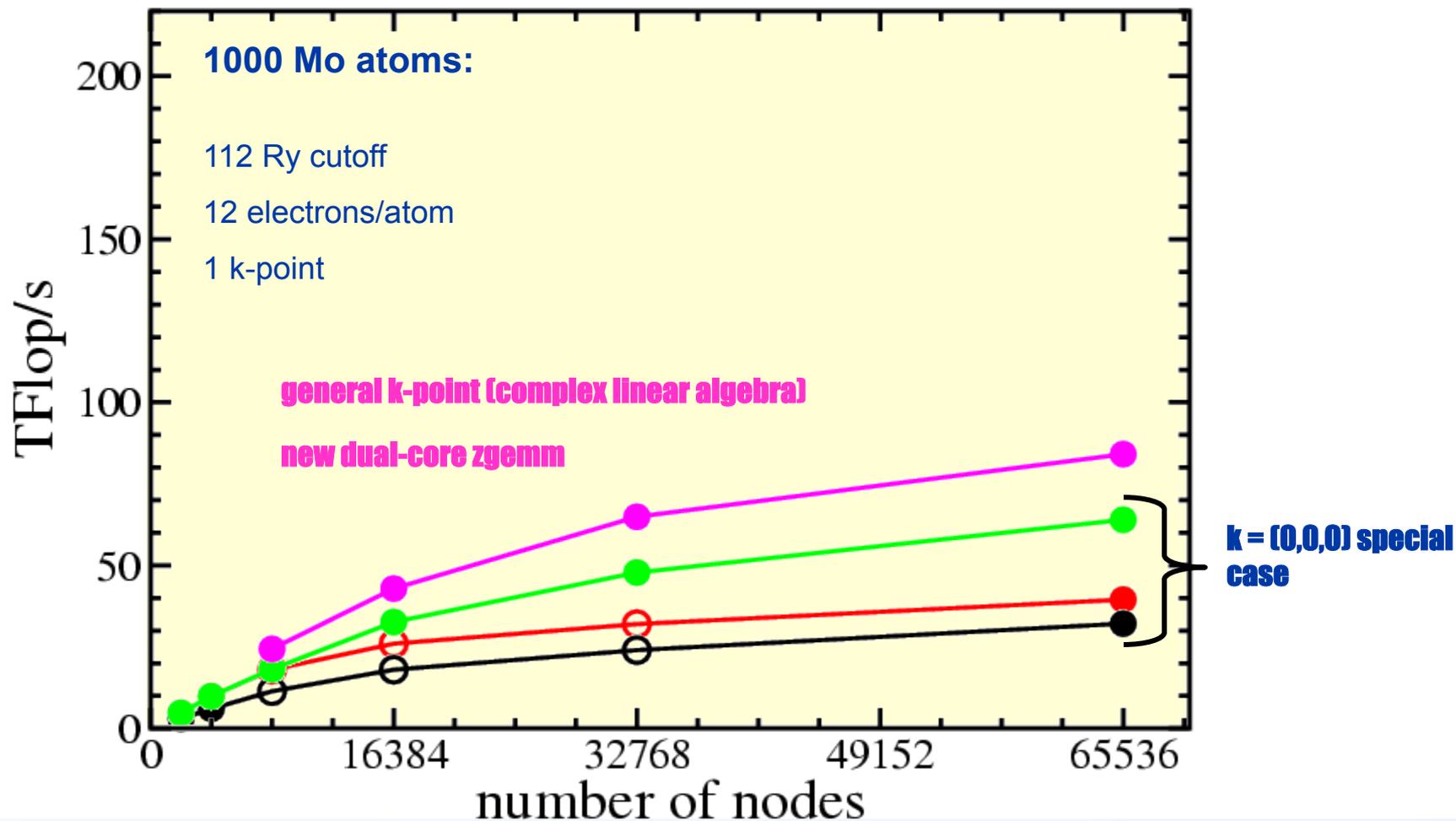


Qbox was originally written for non-crystalline calculations and assumed $k=0$. Rewriting the code for k-points had several challenges:

- **more complicated parallel distribution**
- **complex linear algebra**

Mo₁₀₀₀ scaling results

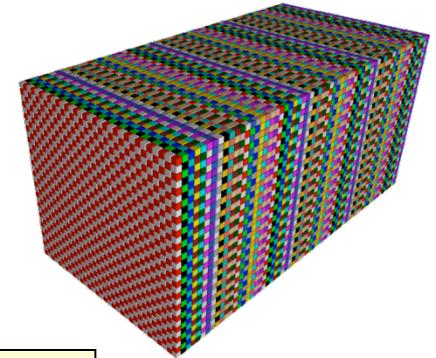
sustained performance



Node Mapping Optimization

Working with Bronis de Supinski and Martin Schulz, we conducted a detailed analysis of communication patterns within the bipartite node mapping scheme to further improve performance.

- Analysis of the MPI tree broadcast algorithm in sub-communicators uncovered communication bottlenecks.



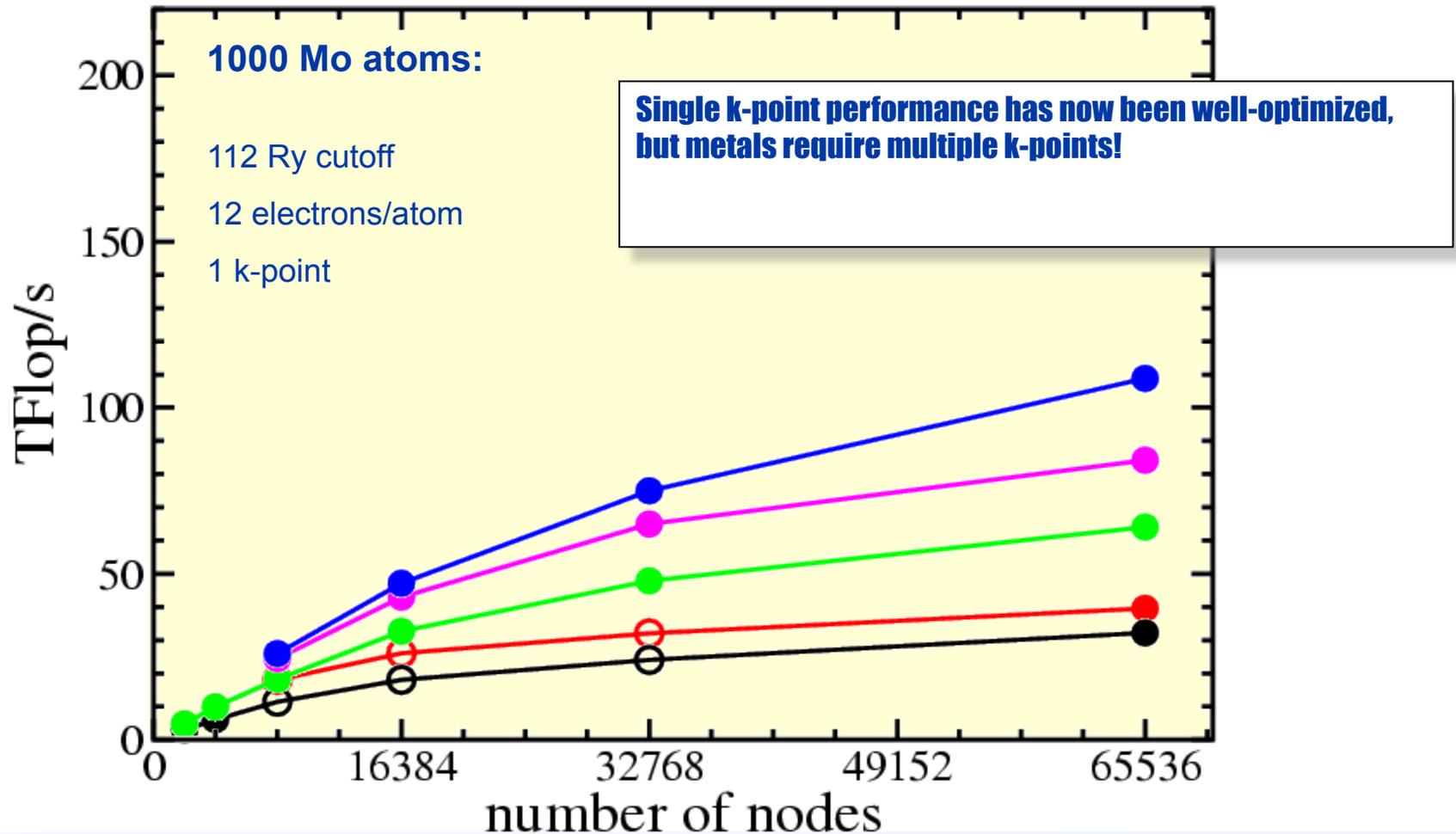
Reordering tasks within each “checkerboard” using a modified space-filling curve improves performance

- **The BLACS library was using an overly general communication scheme for our application, resulting in many unnecessary `Type_commit` operations.**

BLACS was rewritten to include a check for cases where these operations are unnecessary

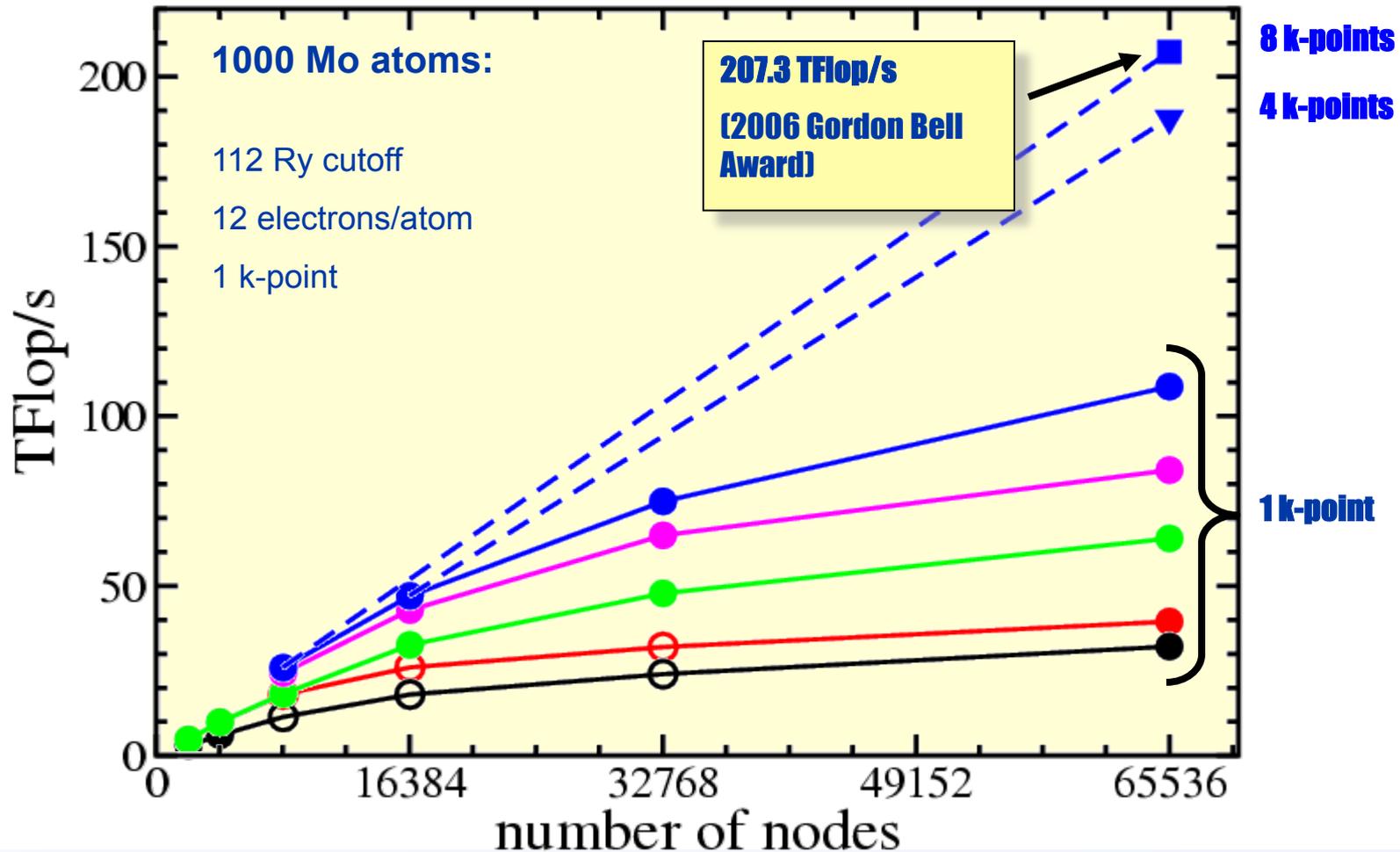
Mo₁₀₀₀ scaling results

sustained performance



Mo₁₀₀₀ scaling results

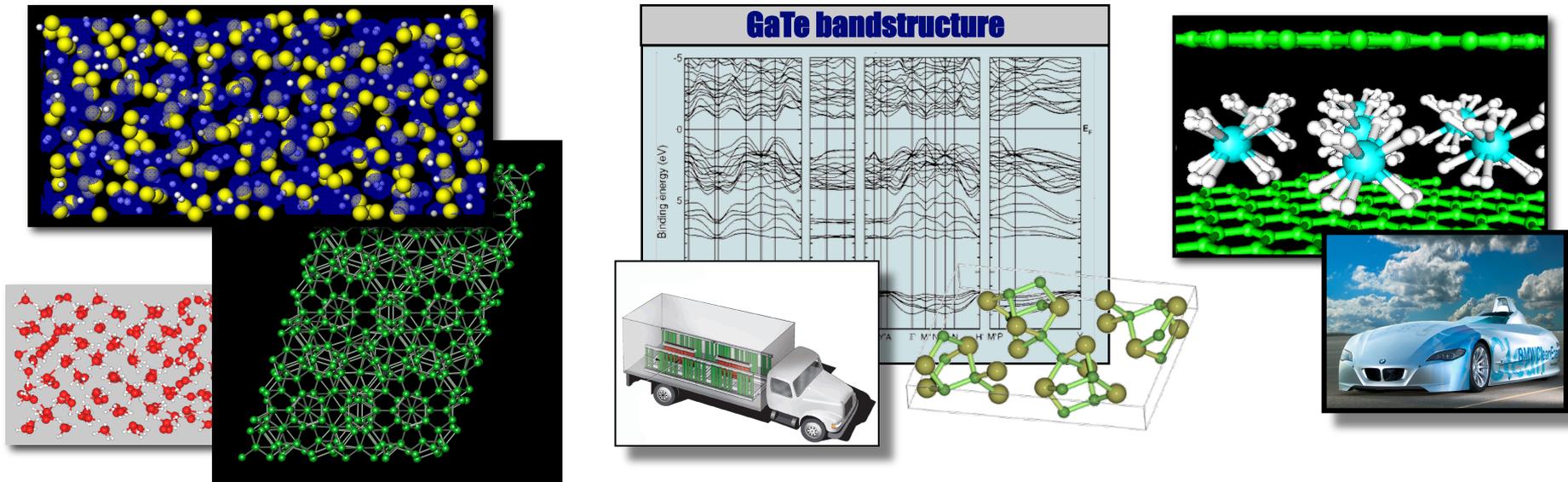
sustained performance



Now that it's fast, what do we do with it?

We now have a first-principles code capable of doing high accuracy calculations of high-Z metals on 131,072 processors. This will allow us to make accurate predictions of the properties of important high-Z systems that previously weren't possible, **a key programmatic goal**.

In addition, Qbox can enable other important research:



Deeper understanding of existing materials

Designing new materials for radiation detection and hydrogen storage

Conclusions

- Use existing numerical libraries (where possible)
 - ScaLAPACK/BLAS
- System-specific practices
 - Use optimized routines where necessary
 - single-node dual-core dgemm, zgemm
 - custom 3D complex FFT routines
 - mapping of tasks to physical nodes
- Important to have suitable tool set
 - May be necessary to modify or develop new tools to assist in scaling



Lessons for Petascale Tools

Tools are essential in Petascale efforts

- Need to debug at large scale
- Performance optimization to exploit machines
- Need performance measurement tools for large scale use
- Centralized infrastructures will not work
 - Tree-based aggregation schemes
 - Distributed storage and analysis
 - Node count independent infrastructures
- Need for flexible and scalable toolboxes
 - Integration and interoperability
 - Comprehensive infrastructures
 - Community effort necessary



Acknowledgments

Qbox Team:

Eric Draeger

Center for Applied Scientific Computing, Lawrence Livermore National Laboratory

François Gygi

University of California, Davis

Martin Schulz, Bronis R. de Supinski,

Center for Applied Scientific Computing, Lawrence Livermore National Laboratory

Franz Franchetti

Department of Electrical and Computer Engineering, Carnegie Mellon University

Stefan Kral, Juergen Lorenz, Christoph W. Ueberhuber

Institute of Analysis and Scientific Computing, Vienna University of Technology

John A. Gunnels, Vernon Austel, James C. Sexton

IBM Thomas J. Watson Research Center, Yorktown Heights, NY

