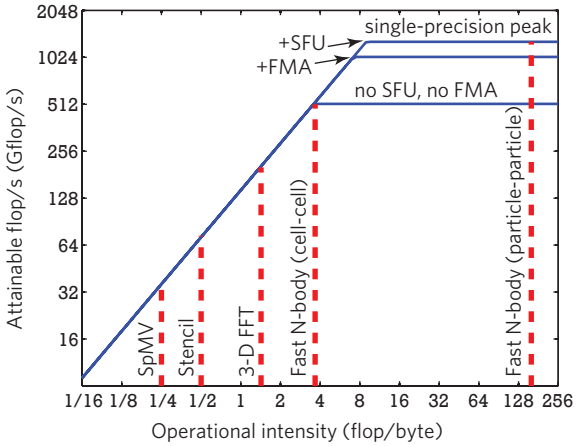
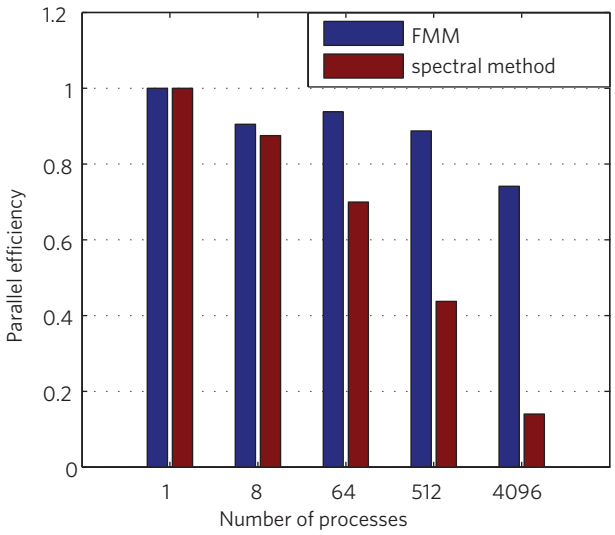
ExaFMM: An open source library for Fast Multipole Methods aimed towards Exascale systems

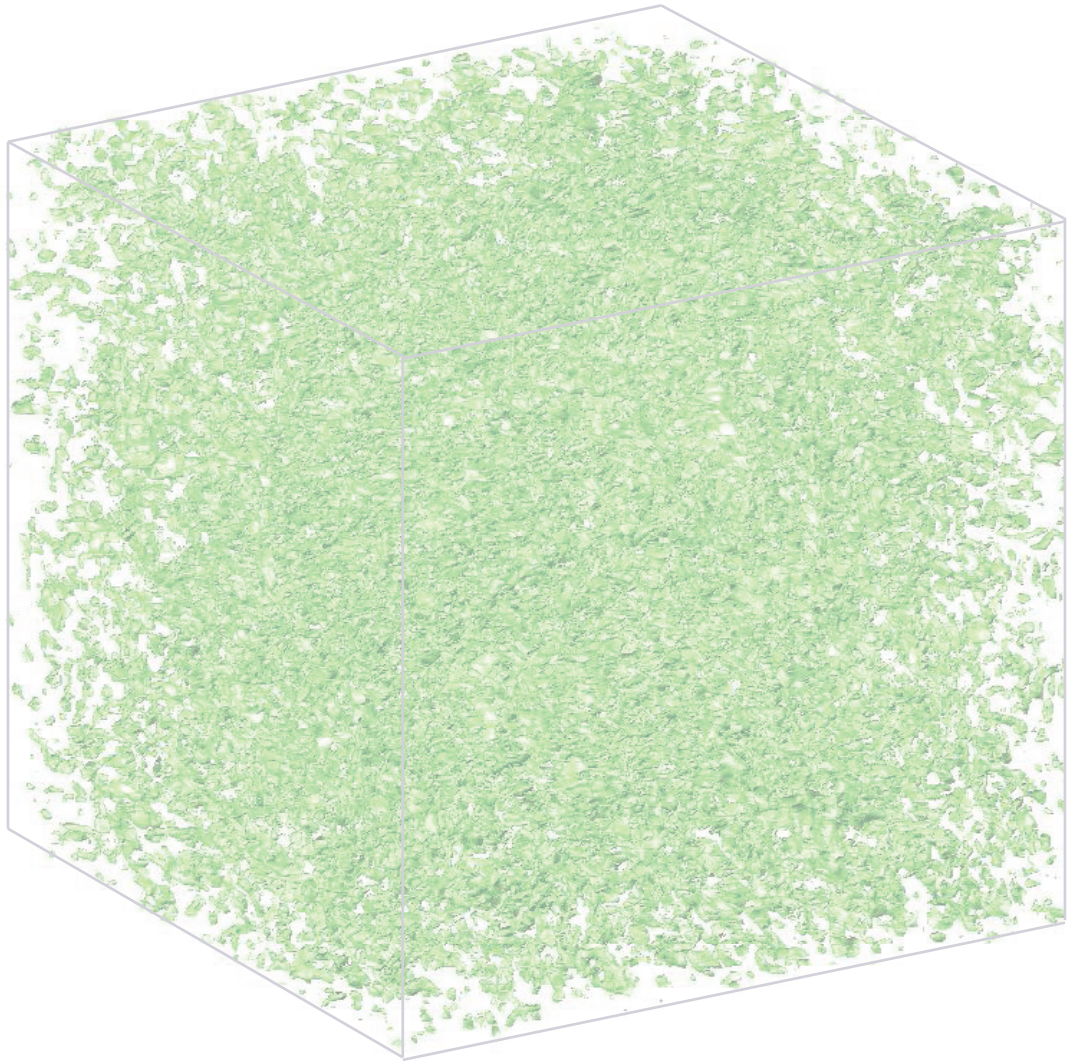
GPU relevance: Compute-bound

The roofline model shows the high operational intesity of FMM kernels compared to SpMV, Stencil, and 3-D FFT kernels. The model is for an NVIDIA Tesla C2050 GPU, where the single-precision peak can only be achieved when special function units (SFU) and fused multiply add (FMA) operations are fully utilized.



Parallel efficiency of the FMM on a weak scaling test achieved more than 70% on 4096 processes (with GPUs), while a similar test of a parallel FFT (without GPUs) shows a dramatic degradation of efficiency at this number of processes.





Application: Turbulence simulations

The simulation of homogeneous isotropic turbulence is one of the most challenging benchmarks for computational fluid dynamics. We have matched the current world record in turbulence simulation, computing a 4096^3 system on 4096 GPUs with our fast multipole vortex method. This simulation ***achieved 1 petaflop/s*** on TSUBAME 2.0.

Although the FMM has been taken to petascale before, the present work represents the first time that this is done on GPU architecture. Also, this is the largest direct numerical simulation with vortex method to date, with almost ***70 billion particles*** used in the cubic volume.

BOSTON UNIVERSITY

Details & download:
www.bu.edu/exafmm

More from this group:
www.barbagroup.bu.edu

See our poster at the NVIDIA booth.

ExaFMM: An open source library for Fast Multipole Methods aimed towards Exascale systems

What's new? An open source FMM

The fast multipole method (FMM) is a numerical engine used in many applications, from acoustics, electrostatics, fluid simulations, wave scattering, and more.

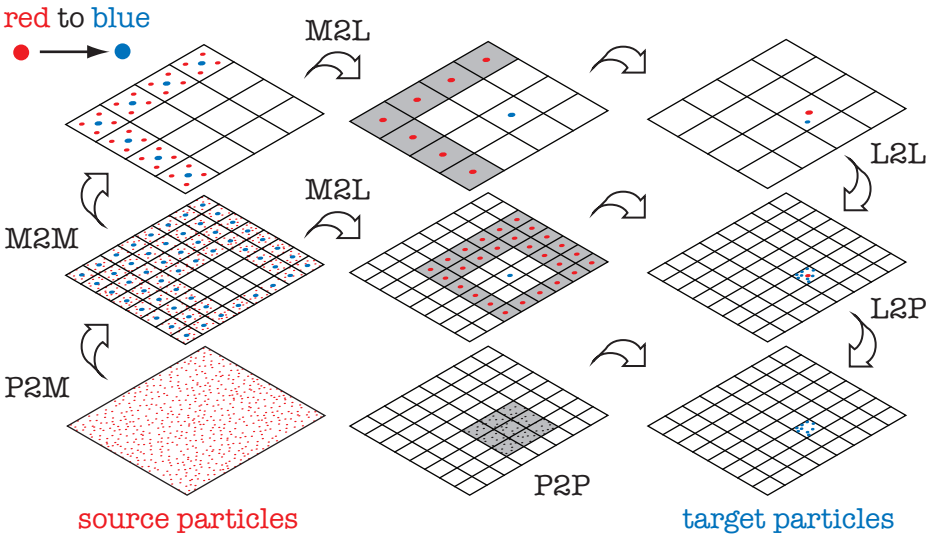
Despite its importance, there is a lack of open community code, which arguably has affected its wider adoption. It is also a difficult algorithm to understand and to program, making availability of open-source implementations even more desirable.

Method: Multipole-based, hierarchical

Two common variants: treecodes and FMMs, both use tree data structures to cluster 'source particles' into a hierarchy of cells. *See Figure below.*

Multipole expansions represent the effect of a cluster of particles, and are built for parent cells by an easy transformation (M2M). Local expansions are used to evaluate the effect of multipole expansions on many 'target points' locally. The multipole-to-local (M2L) transformation is used only on th FMM, while treecodes compute the effect of multipole expansions directly on target points, with O(N log N) scaling for N particles. The FMM scales as O(N).

We developed a novel treecode-FMM hybrid algorithm with auto-tuning capabilities, that is O(N) and chooses the most efficient type of interaction.



Features: (did we say open source?)

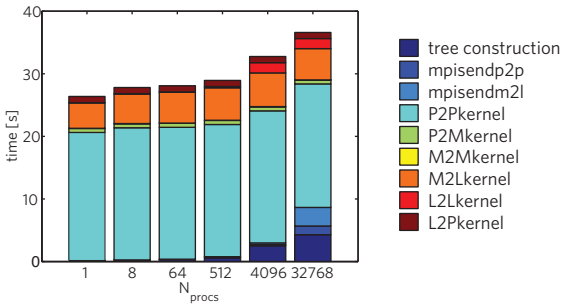
- Inter-node parallelism with MPI
- Intra-node multithreading with OpenMP
- GPU-enabled using CUDA
- Auto-tuning
- Released under the MIT license

Performance: Strong and weak scaling

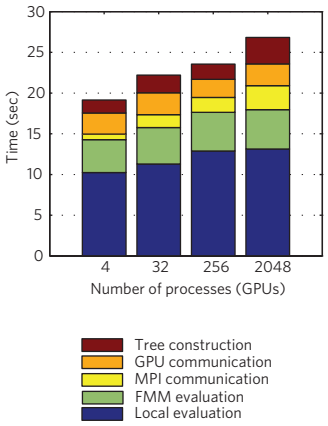
On multi-core systems (Kraken supercomputer), strong scaling with 10^8 particles on 2048 processes achieved:

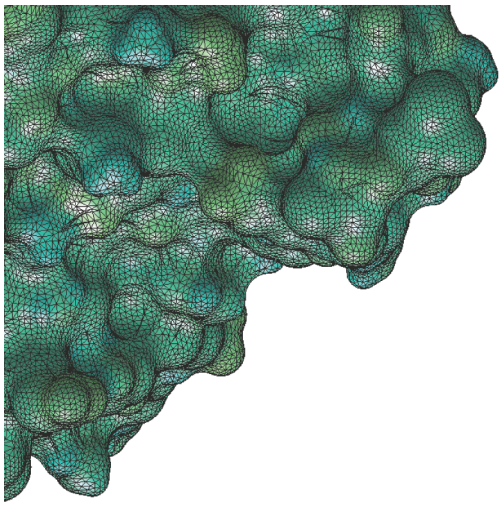
- 93% parallel efficiency for the non-SIMD code, and
- 54% for the SIMD-optimized version (2x faster).

Weak scaling with 10^6 particles per node achieved 72% efficiency on 32,768 processes. *See Figure below.*



On a GPU system (Tsubame 2.0 supercomputer), weak scaling with $4x10^6$ particles per process on 2048 GPUS achieved 72% efficiency. *See Figure below.*





Application: Electrostatics

In a computation of a biomolecular electrostatics problem with the implicit-solvent model, a protein-solvent interface is represented by a surface mesh. A lysozyme molecule needs 100k triangles on its surface.

We modeled a system with more than 10 thousand such molecules—this results in ***one billion unknowns!***

BOSTON UNIVERSITY

Details & download:
www.bu.edu/exafmm

More from this group:
www.barbagroup.bu.edu

See our poster at the NVIDIA booth.