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

BOSTON UNIVERSITY

Lorena A Barba[1], Rio Yokota[2]

[1] Boston University, [2] KAUST

## ***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.

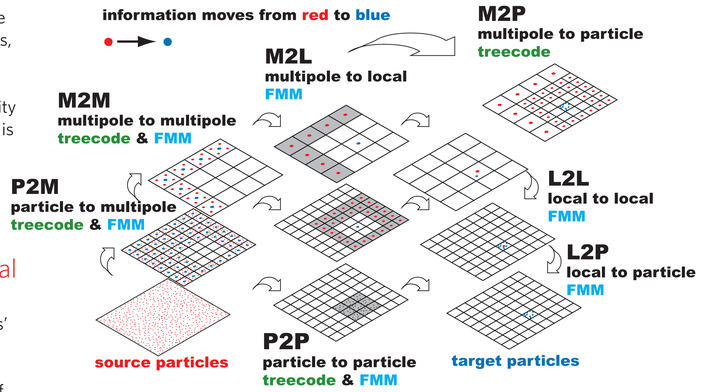
Despite its importance, there is a lack of open community code, which arguably has affected its wider adoption. It is also a difficult algorithm to understand and to program, making availability of open-source implementations even more desirable.

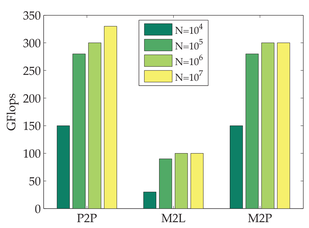
## ***Method:*** Multipole-based, hierarchical

There are two common variants: treecodes and FMMs. Both use tree data structures to cluster 'source particles' into a hierarchy of cells. *See Figure to the right.*

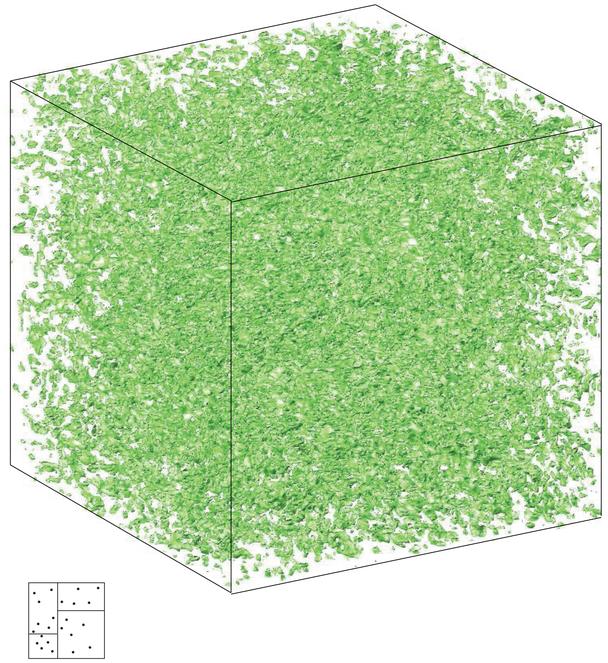
Multipole expansions are used to represent the effect of a cluster of particles, and are built for parent cells by an easy transformation (M2M). Local expansions are used to evaluate the effect of multipole expansions on many target points locally. The multipole-to-local (M2L) transformation takes most of the runtime.

Treecodes do not use local expansions, and compute the effect of multipole expansions directly on target points. They thus scale as O(N log N), for N particles. The FMM has the ideal O(N) scaling.



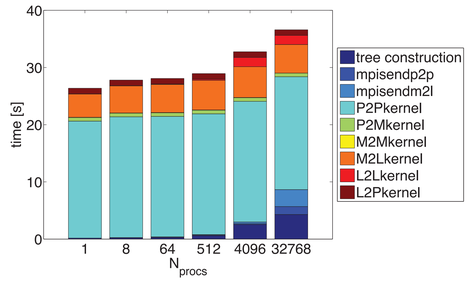




*Left* — Actual performance on the GPU of three core kernels, for four different values of N. The M2P kernel of the treecode is able to deliver more flop/s on GPU than the M2L kernel of the FMM.

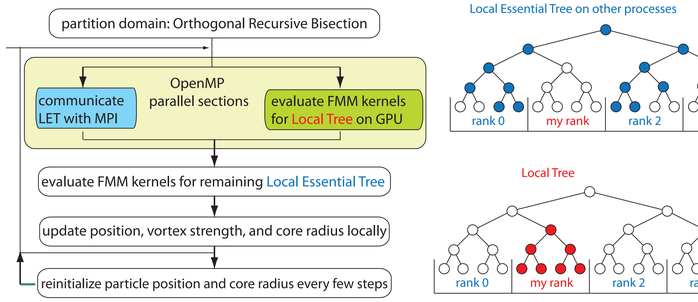


*Left* — The simulation of homogeneous isotropic turbulence is one of the most challenging benchmarks for computational fluid dynamics. Here, we show results of a $2048^3$ system, computed with a fast multipole vortex method on 2048 GPUs. This simulation ***achieved 0.5 petaflop/s*** on TSUBAME 2.0.

*Below* — Scaling on a massively parallel CPU systems is also excellent. The plot shows MPI weak scaling (with SIMD within each core) from 1 to ***32,768 processes***, and timing breakdown of the different kernels, tree construction and communications. The test uses 10 miilion points per process, and results in 72% parallel efficiency on 32 thousand processes (Kraken system).






*Above* — The flow of the vortex method caluclation using parallel FMM, where the domain is partitioned by an orthogonal recursive bisection and communication of the local essential tree (LET) is overlapped with the GPU calculation of the local tree. The remaning parts of the LET are calculated and updates are performed locally.

## ***Results:*** Parallel, multi-GPU performance

Besides the results shown above, strong scaling runs on CPUs for 10 million bodies per core have shown 93% parallel efficiency at 2048 cores without SIMD, and 54% parallel efficiency at 2048 cores with SIMD kernels.

Strong scaling runs on GPUs for 100 million bodies show 65% parallel efficiency on 512 GPUs, while weak scaling runs on GPUs with 4 million bodies per GPU show 72% parallel efficiency on 2048 GPUs.

The calculation of isotropic turbulence on a $2048^3$ grid (as shown in the figure above) uses 4 million bodies per GPU on 2048 GPUs.

The same calculation using FFTW on 2048 CPU cores of the same machine obtains 27% parallel efficinecy on 2048 cfores. The good scalability of FMMs becomes an advantage on massively parallel systems.

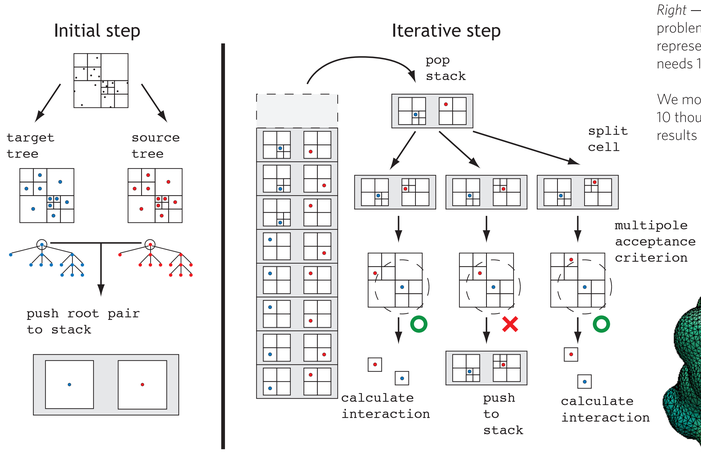
## ***Features:*** Parallel, multi-GPU, with auto-tuning (did we say open source?)

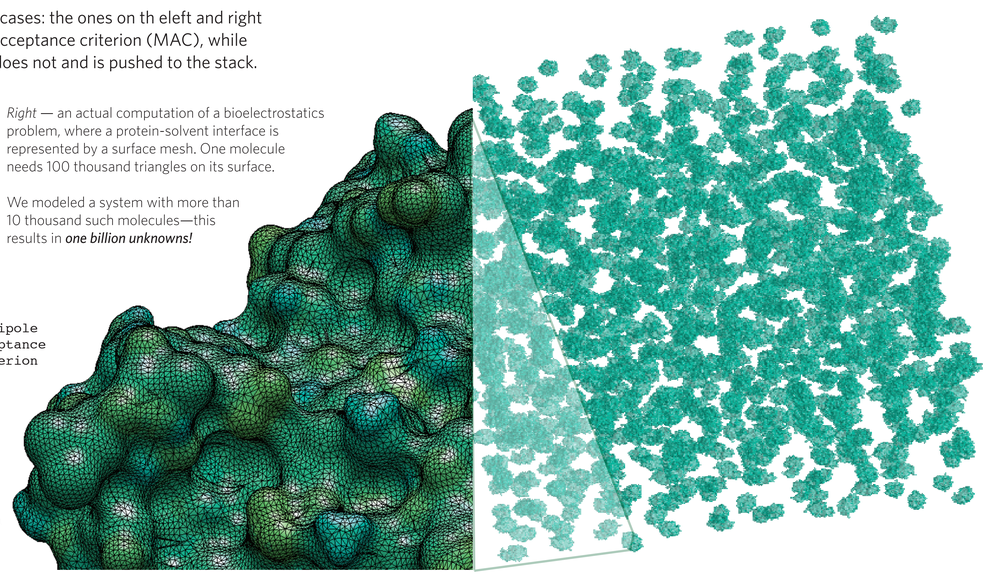
We developed a novel treecode-FMM hybrid algorithm with auto-tuning capabilities, that is O(N) and chooses the most efficient type of interaction. It is highly parallel and GPU-capable.

The algorithm uses a dual tree traversal for the target and source trees. *See Figure below*. In the initial step, both trees are formed and their root cells are paired and pushed to an empty stack.

Further iterations consist of popping a pair of cells from the stack, splitting the larger one, and applying an acceptance criterion to determine if multipole interaction is valid or not. If not, a new pair is pushed to the stack.

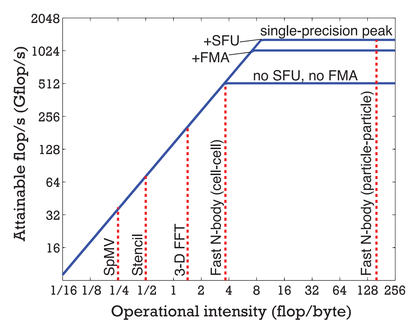
The figure shows tree cases: the ones on th eleft and right satisfy the multipole acceptance criterion (MAC), while the one in the center does not and is pushed to the stack.



*Right* — an actual computation of a bioelectrostatics problem, where a protein-solvent interface is represented by a surface mesh. One molecule needs 100 thousand triangles on its surface.

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




*Right* — This 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.

## ***Want more?*** Papers and software are online

All the codes developed in our group are free (like free beer) and open source. To download them, follow the links from our group website:

***http://barbagroup.bu.edu***

Also on the website are up-to-date bibliographic references, and papers for download. ***Please visit!***

**GTC Express @ SC'11,** November 2011