A Kernel-independent Adaptive Fast Multipole Method

Lexing Ying

Caltech

Joint work with George Biros and Denis Zorin

Problem Statement

Given

- G an elliptic PDE kernel, e.g. $G(x, x') = \frac{1}{|x - x'|}$
- $\{\mathbf{x}_i\}$ points in $\mathbf{R}^d$ $(d = 2, 3)$
- $\{\phi_i\}$ charges

Evaluate potential

$$u_i = \sum_{j=1}^{N} G(\mathbf{x}_i, \mathbf{x}_j)\phi_j, \quad i = 1, \ldots, N$$

- Direct evaluation: $\mathcal{O}(N^2)$
- Barnes-Hut algorithm: $\mathcal{O}(N \log N)$
- Fast Multipole Method (FMM): $\mathcal{O}(N)$
  - Greengard and Rokhlin (JCP, 1987)

Contribution

New kernel-independent FMM

- Use only kernel evaluations
- Extend FMM to general kernels
- Efficiency and accurate

MPI based parallel implementation

- Scale up to 3000 processors
- 1.6 Tflops peak / 1.13Tflops sustained performance on 2.1 billion unknowns

3D high-order BIE solver

- Laplace and Stokes equation
- Smooth boundary

Applications

Electrostatics

Gravitation

Fluid dynamics

Molecular dynamics

…

Outline

Classical FMM

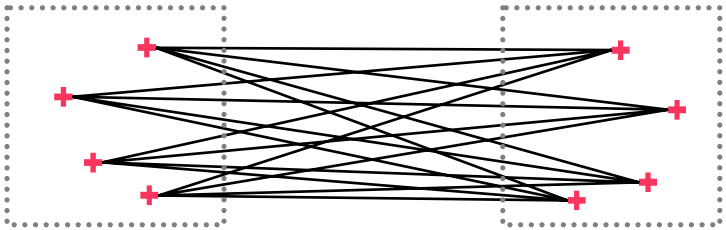
New algorithm

Parallel algorithm

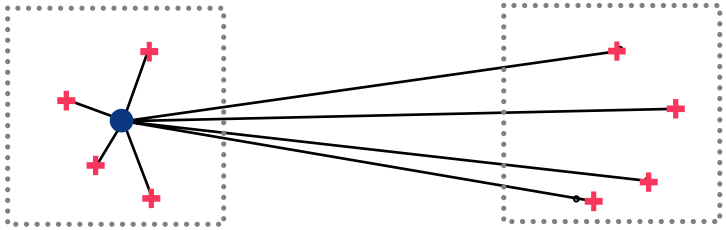
Boundary integral equation solver

Idea of Classical FMM

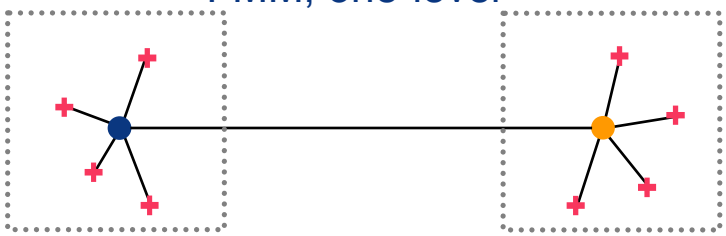
Direct evaluation



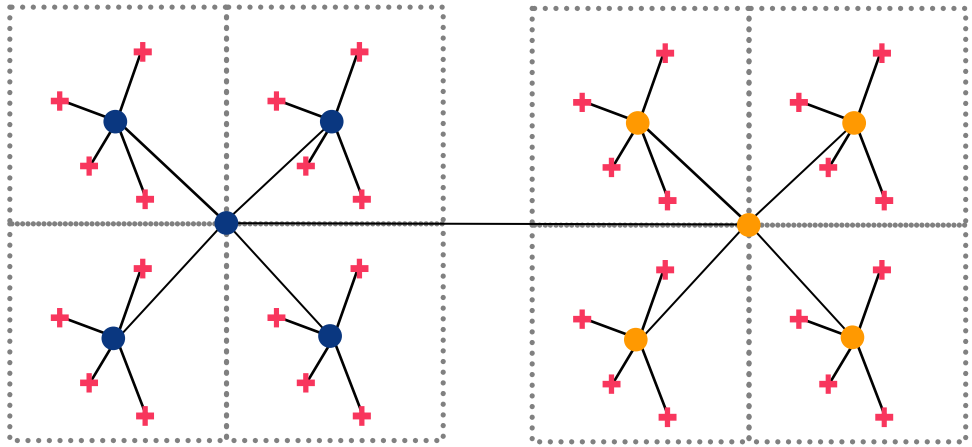
Barnes-Hut



FMM, one-level



FMM, 2-level



Well separated

2 efficient representation

3 efficient translation

Domain Partitioning

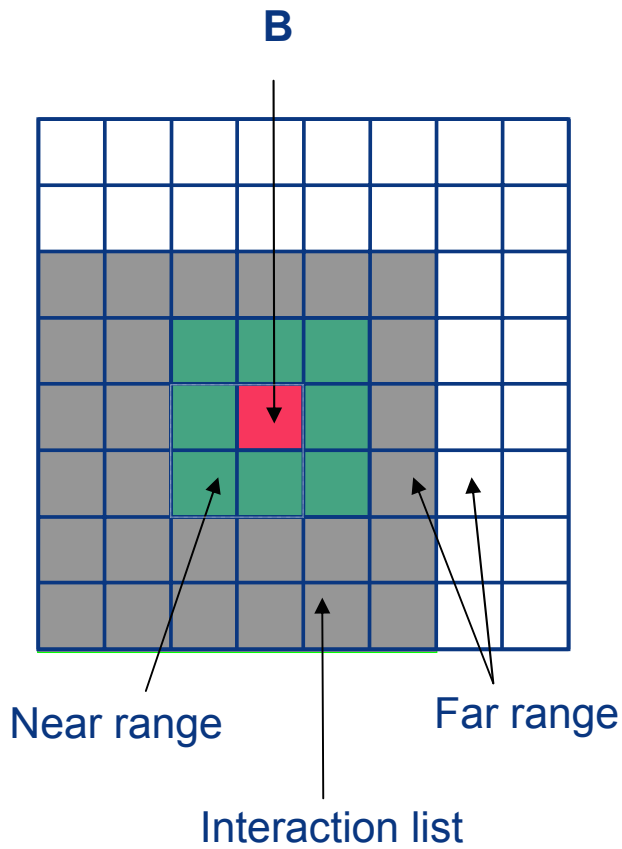
Quadtree / Octree

- Leaf box contains at most *s* charges

For each box **B**, define

- Near range
- Far range
- Interaction list

B and its far range are well separated



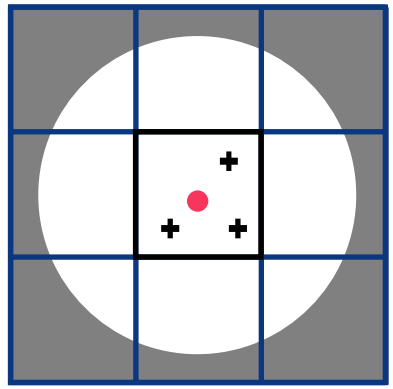
Efficient Representations

Multipole expansion (ME)

$\{\phi_j\}$ at $\{z_j\} : |z_j - z_C| < r$, for $|z - z_C| > R$

$$u(z) \approx a_0 \log(z - z_C) + \sum_{k=1}^{p} \frac{a_k}{(z - z_C)^k}$$

$$a_0 = \sum_{j=1}^{m} \phi_j \qquad a_k = \sum_{j=1}^{m} \frac{-\phi_i (z_i - z_C)^k}{k}.$$

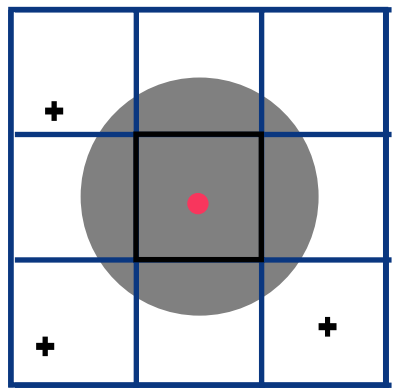


Local expansion (LE)

$\{\phi_j\}$ at $\{z_j\} : |z_j - z_C| > R$, for $|z - z_C| < r$

$$u(z) \approx \sum_{k=0}^{p} c_k (z - z_C)^k$$

$$c_0 = \sum_{j=1}^{m} \phi_j \log(z_C - z_j) \qquad c_l = \sum_{j=1}^{m} \frac{-\phi_j}{l \cdot (z_j - z_C)^l}.$$

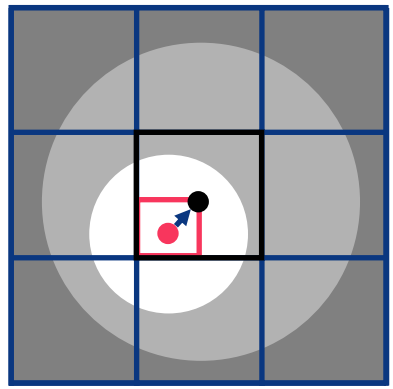


m is a constant → Both expansions are efficient

Efficient Translations

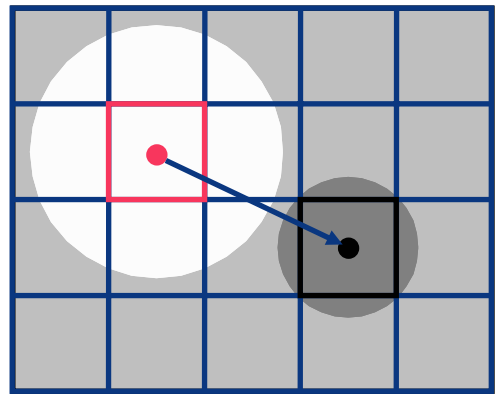
M2M: multipole to multipole

- From child to parent box



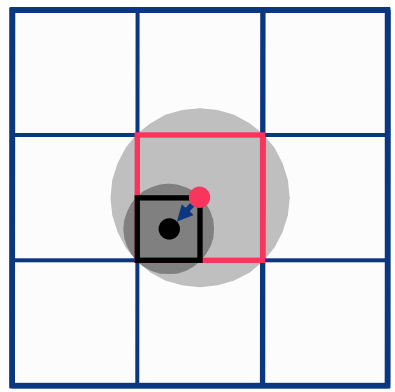
M2L: multipole to local

- From B's interaction list to B



L2L: local to local

- From parent to child box



All translations are efficient

Algorithm

1. Compute multipole expansion (ME) for each box
   - If leaf box, compute from exact sources in that box
   - If non-leaf box, compute from its children's ME using M2M
2. Compute local expansion (LE) for each box
   - Use M2L to translate the ME of the boxes in the interaction list and add to the LE of the current box
   - Use L2L to translate the LE of the parent box and add to the LE of the current box

Invariance: LE represents the potential from far field accurately

3. For each box, compute potential at each of its point by combining
   - Contribution from LE of this box
   - Potential from points in near range of this box

Kernel Dependency

Expansions and translations depend on underlying PDE

3D implementation only available for Laplace and Helmholtz

10 years to get efficient implementation for Laplace kernel

FMM can be extremely difficult for kernels look like this:

$$\mathbf{D}(\mathbf{x},\mathbf{y}) = A\left((\mathbf{r}\cdot\mathbf{n})\mathbf{I} + \mathbf{n}\otimes\mathbf{r}\right) + B(\mathbf{r}\otimes\mathbf{n}) + C(\mathbf{r}\cdot\mathbf{n})(\mathbf{r}\otimes\mathbf{r})$$

$$A = -\frac{f_{rrr}}{r} - (d-3)\frac{f_{rr}}{r^2} + (d-3)\frac{f_r}{r^3} \qquad B = -p + 2\frac{f_{rr}}{r^2} - 2\frac{f_r}{r^3} \qquad C = 2\frac{f_{rrr}}{r^3} - 6\frac{f_{rr}}{r^4} + 6\frac{f_r}{r^5}$$

$$f = \begin{cases} \frac{1}{2\pi\lambda^2}(\ln(\frac{1}{r}) - k_0(\lambda r)) & (2D) \\ \frac{1}{4\pi\lambda^2}(\frac{1}{r} - \frac{1}{r}e^{-\lambda r}) & (3D) \end{cases} \qquad p = \begin{cases} \frac{1}{2\pi}\frac{1}{r^2} & (2D) \\ \frac{1}{4\pi}\frac{1}{r^4} & (3D) \end{cases}$$

Outline

Classical FMM

New algorithm

Parallel algorithm

Boundary integral equation solver

Previous Work

FMM without multipole

- Anderson ’92
- Not fully kernel independent

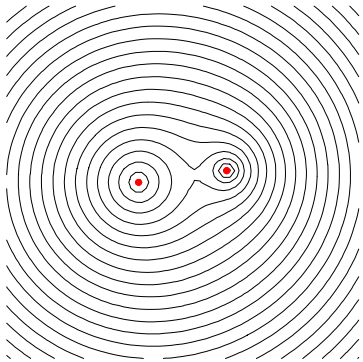
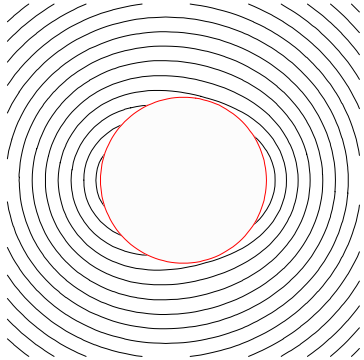
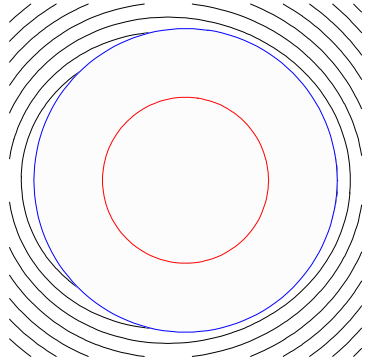
FFT-based methods

- Phillips, White ’97, pre-corrected FFT
- Bruno, Kunyansky ’01, improved version

Wavelet-based methods

- BCR ’91 and Alpert et al. ’93

Idea of New Algorithm

An electrostatic example

- Represent sources with equivalent densities on the circle
- Match check potential on the boundary of the far field

Extend this idea to FMM

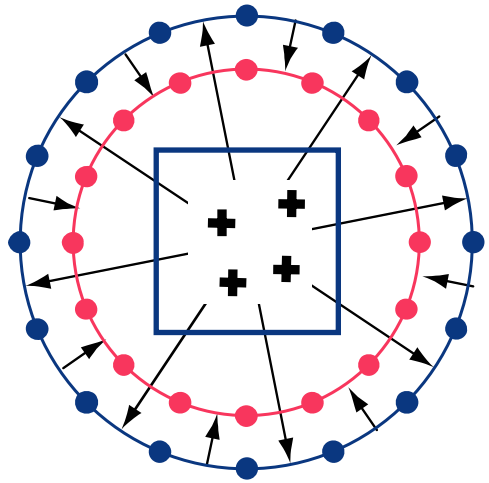
- Multipole expansion—upwards equivalent density
- Local expansion—downs equivalent density
- M2M, M2L and L2L translation—linear solution for the matching process

Efficient Representations

Upwards equivalent density

$$\int_{\mathbf{y}^{B,u}} G(\mathbf{x},\mathbf{y})\boxed{\phi^{B,u}}\mathrm{d}\mathbf{y} = \sum_{i\in I_s^B} G(\mathbf{x},\mathbf{y}_i)\phi_i = u^{B,u}$$

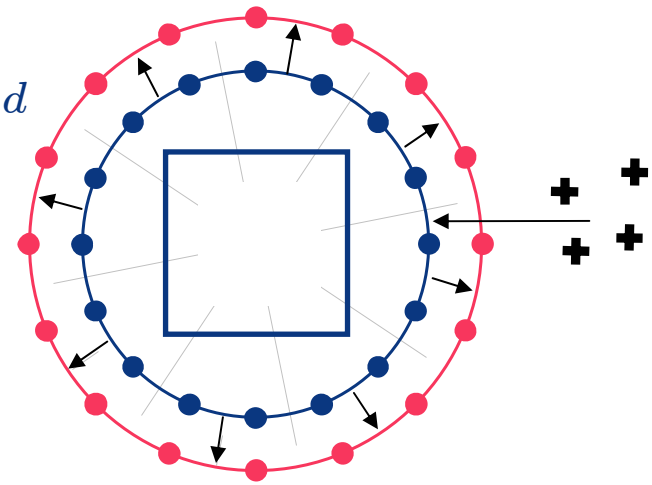
for any $\mathbf{x} \in \mathbf{x}^{B,u}$.



Downwards equivalent density

$$\int_{\mathbf{y}^{B,d}} G(\mathbf{x},\mathbf{y})\boxed{\phi^{B,d}}\mathrm{d}\mathbf{y} = \sum_{i\in I_s^{\mathcal{F}^B}} G(\mathbf{x},\mathbf{y}_i)\phi_i = u^{B,d}$$

for any $\mathbf{x} \in \mathbf{x}^{B,d}$.



Efficient Translations

M2M

$$\int_{\mathbf{y}^{B,u}} G(\mathbf{x},\mathbf{y})\phi^{B,u}(\mathbf{y})\,\mathrm{d}\mathbf{y} = \int_{\mathbf{y}^{A,u}} G(\mathbf{x},\mathbf{y})\phi^{A,u}(\mathbf{y})\,\mathrm{d}\mathbf{y}$$

- From child to parent

for all $\mathbf{x} \in \mathbf{x}^{B,u}$

M2L

$$\int_{\mathbf{y}^{B,d}} G(\mathbf{x},\mathbf{y})\phi^{B,d}(\mathbf{y})\,\mathrm{d}\mathbf{y} = \int_{\mathbf{y}^{A,u}} G(\mathbf{x},\mathbf{y})\phi^{A,u}(\mathbf{y})\,\mathrm{d}\mathbf{y}$$

- From interaction list to box

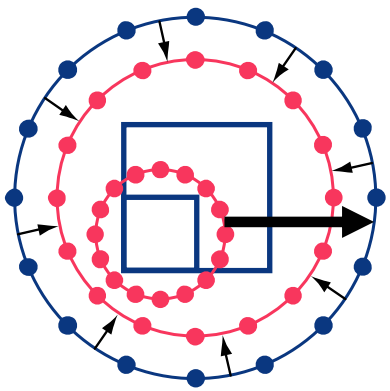
for all $\mathbf{x} \in \mathbf{x}^{B,d}$

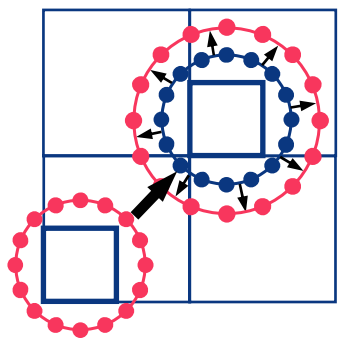
L2L

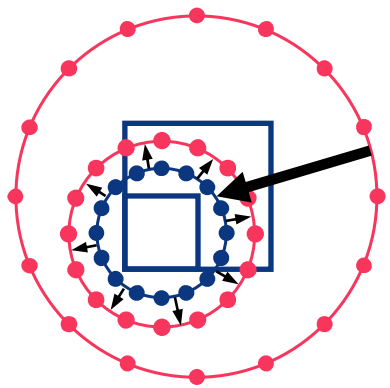
$$\int_{\mathbf{y}^{B,d}} G(\mathbf{x},\mathbf{y})\phi^{B,d}(\mathbf{y})\,\mathrm{d}\mathbf{y} = \int_{\mathbf{y}^{A,d}} G(\mathbf{x},\mathbf{y})\phi^{A,d}(\mathbf{y})\,\mathrm{d}\mathbf{y}$$

- From parent to child

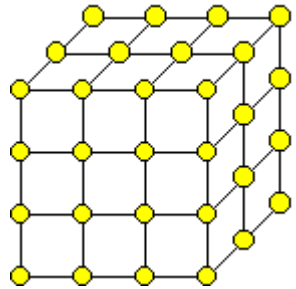
for all $\mathbf{x} \in \mathbf{x}^{B,d}$







Discretization



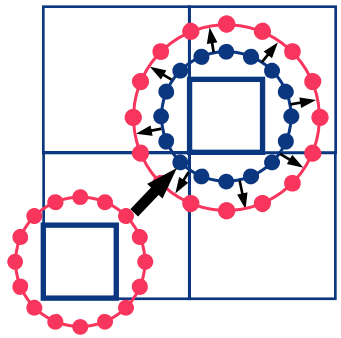
3D case

- Equivalent and check surfaces are cubes
- Boundary points of Cartesian grid as discretization points

Discretized translations (M2M, M2L and L2L) are small matrices

- Independent of the position of the box involved
- Pre-compute and store them

Acceleration of M2L

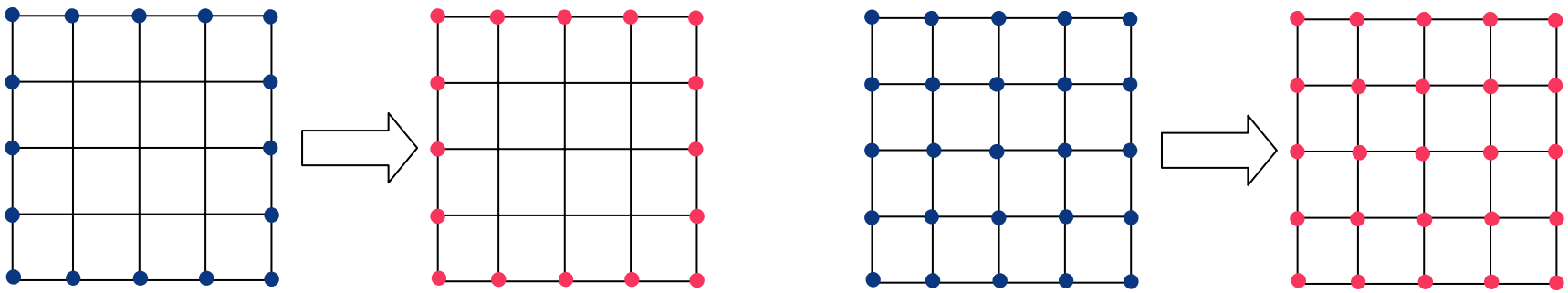


2D

- Store the SVD of the M2L matrix

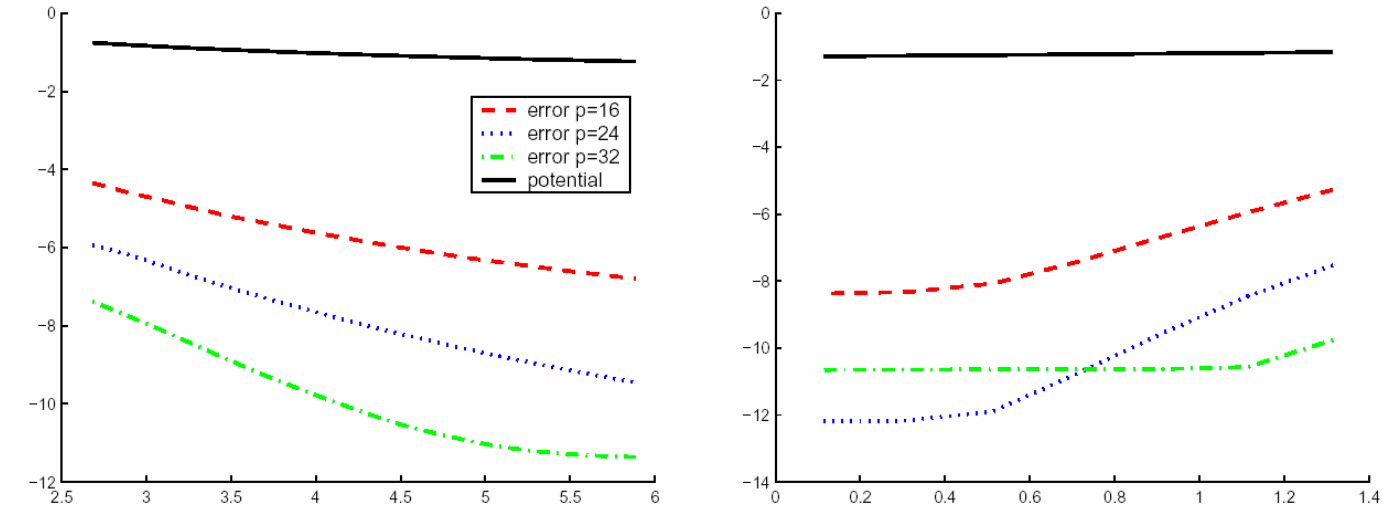
3D

- Pad interior Cartesian nodes with zeros
- Kernel translation invariant
- M2L translation = small 3D discrete convolution
- Use 3D FFT to compute it



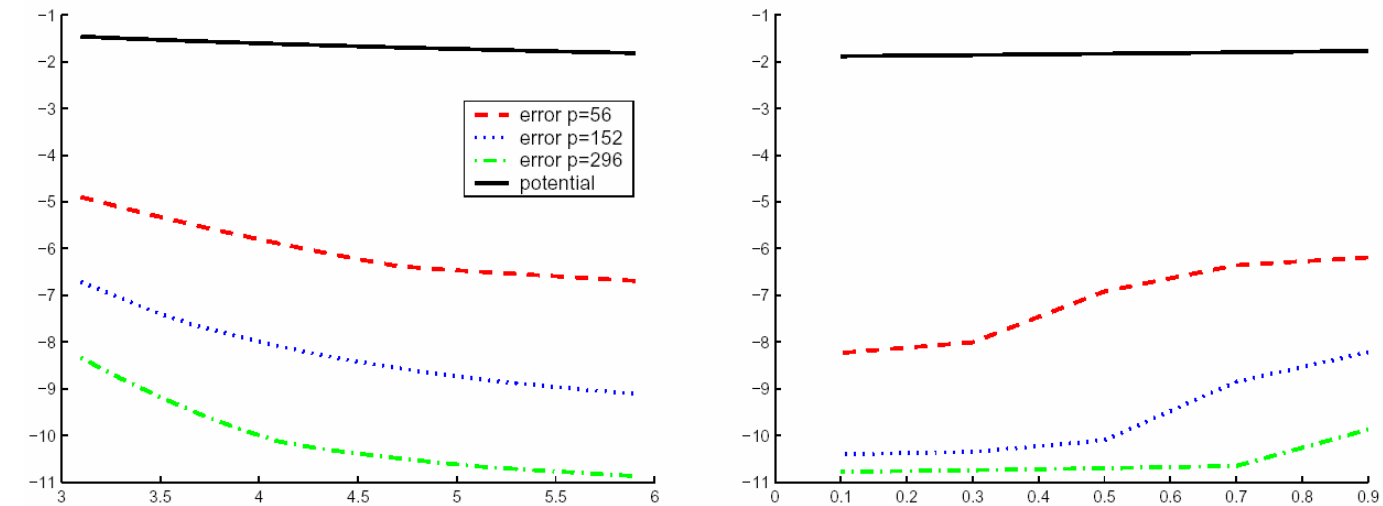
Eq. Density Approximation

2D



Single-layer Navier.

3D

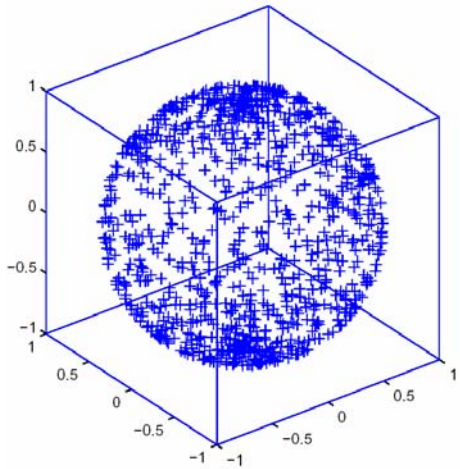


Single-layer Laplacian.

3D Laplace Kernel

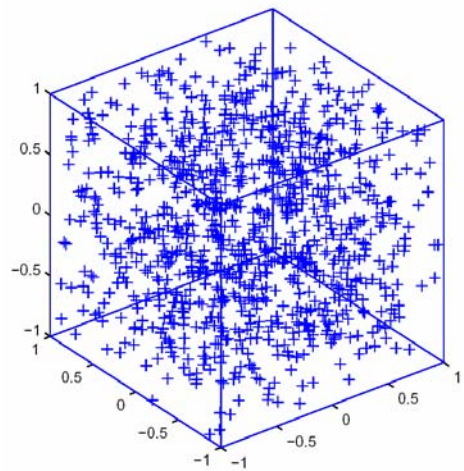
N	R	M	p	s	Storage (Mb)	T_{fmm} (sec)	T_{dir} (sec)	Error
24576	6	1377	56	60	1.72e+00	5.72e+00	9.74e+01	2.12e-05
98304	7	5049	56	60	1.72e+00	2.38e+01	1.56e+03	3.21e-05
393216	8	19065	56	60	1.72e+00	9.51e+01	2.49e+04	6.08e-05
1572864	9	76185	56	60	1.72e+00	3.82e+02	3.99e+05	6.03e-05
24576	5	585	152	150	5.90e+00	1.16e+01	9.74e+01	3.34e-07
98304	6	2289	152	150	5.90e+00	4.76e+01	1.56e+03	5.86e-08
393216	7	11193	152	150	5.90e+00	2.18e+02	2.49e+04	2.45e-07
1572864	9	44145	152	150	5.90e+00	8.35e+02	3.99e+05	3.08e-07
24576	4	273	296	250	1.47e+01	1.81e+01	9.74e+01	1.59e-09
98304	6	1449	296	250	1.47e+01	8.15e+01	1.56e+03	1.40e-09
393216	7	5073	296	250	1.47e+01	3.41e+02	2.49e+04	1.10e-09
1572864	8	19161	296	250	1.47e+01	1.38e+03	3.99e+05	2.81e-09

The particles are distributed on the surface of a sphere.



N	R	M	p	s	Storage (Mb)	T_{fmm} (sec)	T_{dir} (sec)	Error
24576	4	585	56	60	1.72e+00	6.40e+00	9.74e+01	6.64e-06
98304	5	3657	56	60	1.72e+00	3.11e+01	1.56e+03	1.27e-05
393216	7	28233	56	60	1.72e+00	1.30e+02	2.49e+04	5.00e-05
1572864	8	88137	56	60	1.72e+00	4.08e+02	3.99e+05	5.84e-05
24576	4	585	152	150	5.90e+00	1.60e+01	9.74e+01	1.54e-08
98304	5	3657	152	150	5.90e+00	9.28e+01	1.56e+03	4.70e-08
393216	6	14409	152	150	5.90e+00	3.18e+02	2.49e+04	1.10e-07
1572864	7	37449	152	150	5.90e+00	8.47e+02	3.99e+05	2.13e-07
24576	4	585	296	250	1.47e+01	3.65e+01	9.74e+01	5.25e-10
98304	4	585	296	250	1.47e+01	1.11e+02	1.56e+03	4.57e-10
393216	5	3657	296	250	1.47e+01	4.31e+02	2.49e+04	6.85e-10
1572864	6	17481	296	250	1.47e+01	1.46e+03	3.99e+05	1.46e-09

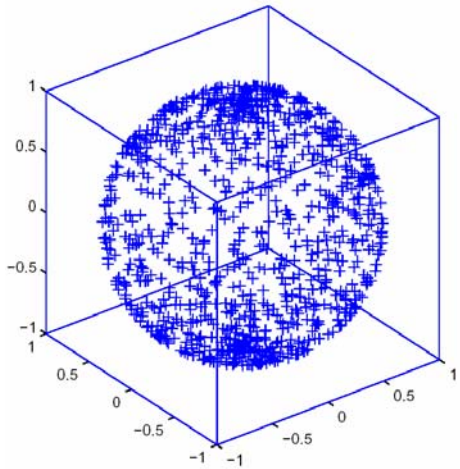
The particles are uniformly distributed inside a cube.



3D Navier Kernel

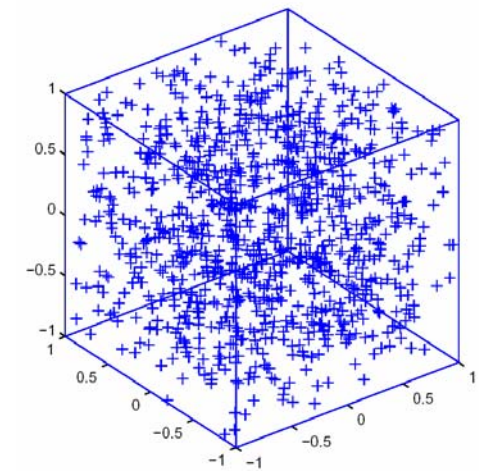
N	R	M	p	s	Storage (Mb)	T_{fmm} (sec)	T_{dir} (sec)	Error
6144	5	441	56	60	1.55e+01	1.29e+01	5.91e+01	8.54e-05
24576	6	1377	56	60	1.55e+01	4.93e+01	9.46e+02	6.71e-05
98304	7	5049	56	60	1.55e+01	1.98e+02	1.51e+04	6.32e-05
6144	4	225	152	150	5.50e+01	3.29e+01	5.91e+01	1.07e-06
24576	5	585	152	150	5.50e+01	1.10e+02	9.46e+02	1.66e-06
98304	6	2289	152	150	5.50e+01	4.59e+02	1.51e+04	1.02e-06
6144	3	57	296	250	1.08e+02	3.28e+01	5.91e+01	7.30e-09
24576	4	273	296	250	1.36e+02	1.82e+02	9.46e+02	8.51e-09
98304	6	1449	296	250	1.36e+02	8.51e+02	1.51e+04	8.73e-09

The particles are distributed on the surface of a sphere.



N	R	M	p	s	Storage (Mb)	T_{fmm} (sec)	T_{dir} (sec)	Error
6144	4	585	56	60	1.55e+01	3.41e+01	5.91e+01	3.70e-05
24576	4	585	56	60	1.55e+01	6.65e+01	9.46e+02	4.82e-05
98304	5	3657	56	60	1.55e+01	3.13e+02	1.51e+04	6.68e-05
6144	3	73	152	150	4.94e+01	2.19e+01	5.91e+01	1.81e-07
24576	4	585	152	150	5.50e+01	1.62e+02	9.46e+02	3.50e-07
98304	5	3657	152	150	5.50e+01	9.48e+02	1.51e+04	4.86e-07
6144	3	73	296	250	1.18e+02	3.78e+01	5.91e+01	2.56e-09
24576	4	585	296	250	1.36e+02	4.22e+02	9.46e+02	3.58e-09
98304	4	585	296	250	1.36e+02	1.00e+03	1.51e+04	4.39e-09

The particles are uniformly distributed in a cube.



Outline

Classical FMM

New algorithm

Parallel algorithm

Boundary integral equation solver

Motivation

Real applications solve huge N-body problem

- E.g. astrophysics, molecular dynamics
- New algorithm can still be slow

Parallel linear solvers are available

- E.g. PetSc
- FMM (often serves as matvec) better be parallel as well

FMM algorithm can be nicely parallelized

- Most operations are local
- E.g. parent to child, box to its interaction list …

Two Problems

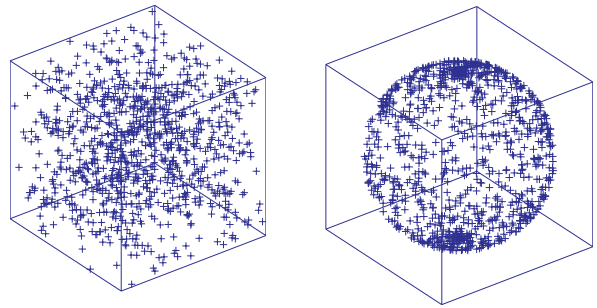
Octree can be huge

- Needs to be generated and stored in a distributed way
- Each processor keeps its Local Essential Tree (LET)
- Most work is done on LET

Synchronization due to data dependence

- Parent node's ME depends on child node's ME, but two nodes can reside in different processors
- Use linearity to remove data dependence
  - All translations are linear operator

Results



Tests setup

- Laplace, Stokes, Navier and modified versions
- Uniform and non-uniform distributions in box $[-1, 1]^3$

Software

- C++, uses PETSC, FFTW

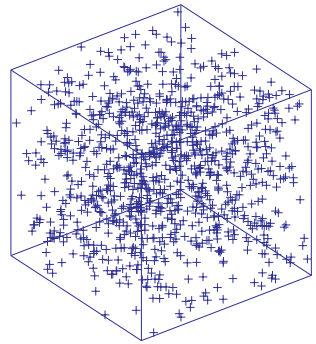
Hardware

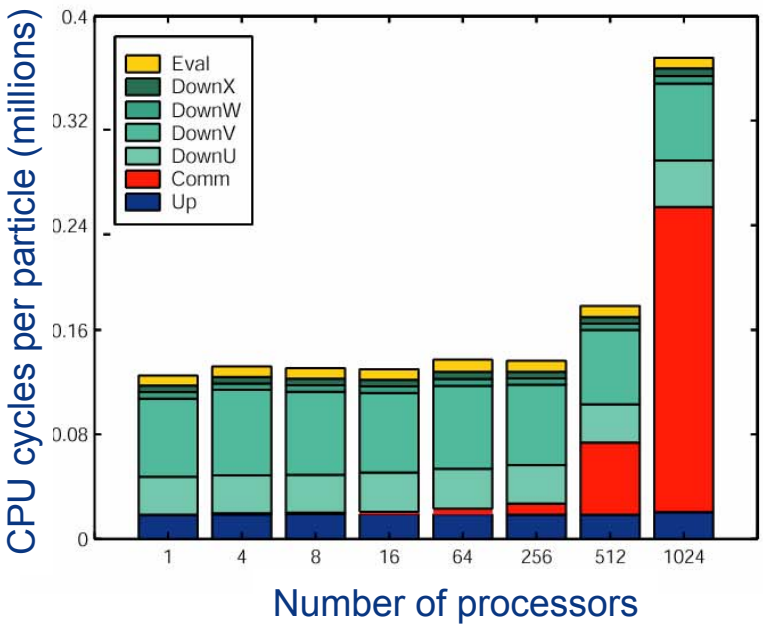
- TCS1 HP Alpha cluster at Pittsburgh supercomputing center
  - 750 compute nodes
  - 4 1GHz processors and 4GB memory per node
  - 187GB/s quadrics interconnection network

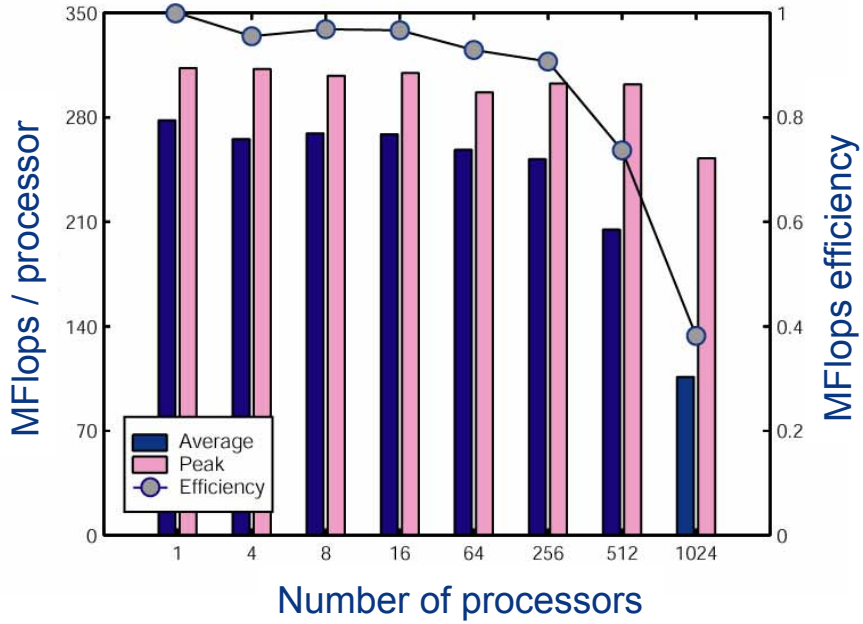
Fixed Size Scalability

Laplace equation,
uniform distribution

3.2 million points in total



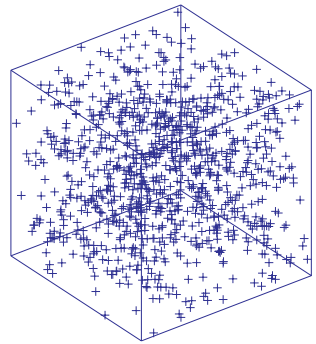


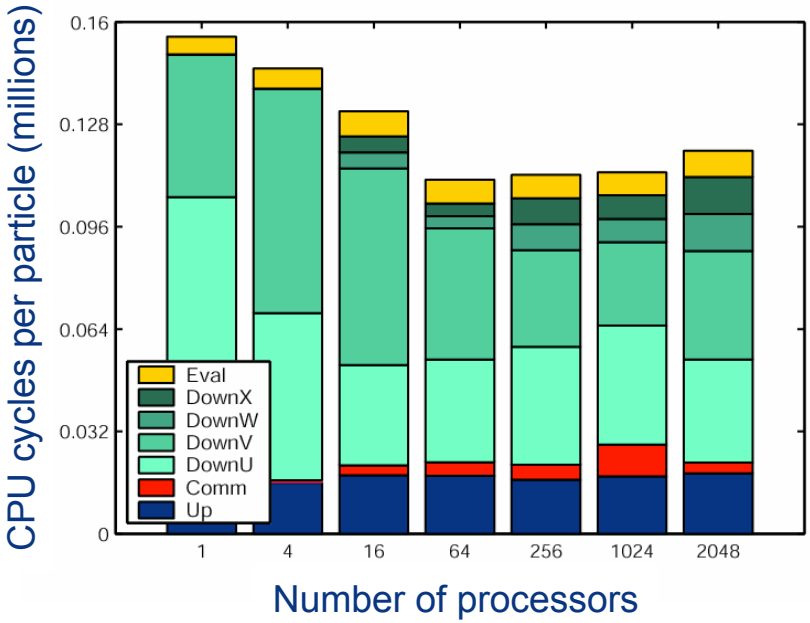


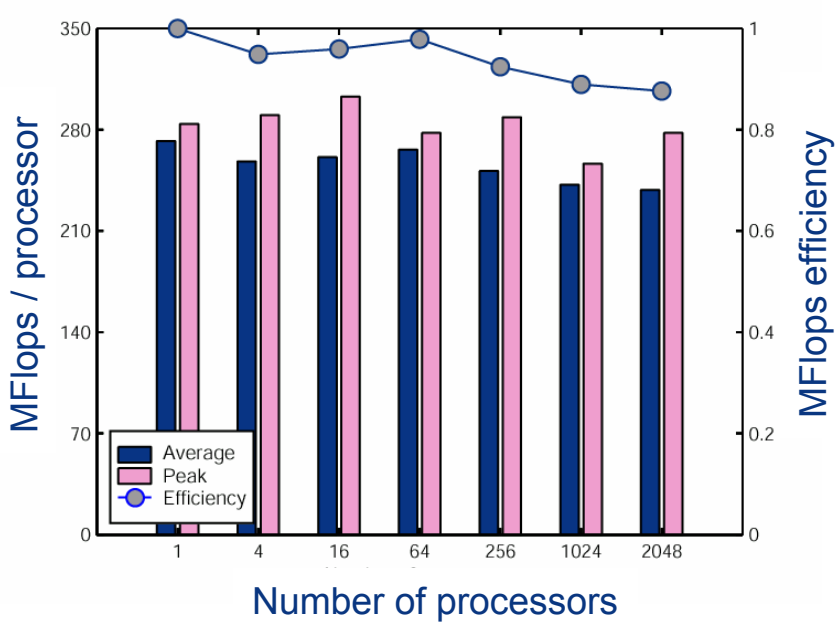
Isogranular Scalability

Laplace equation,
uniform distribution

0.2 million points per processor







3000 Processor Run

Stokes equation, uniform distribution

	Interaction computation						
	Time (sec)					GFlops/s	
unknowns	*Total*	*Ratio*	*Comm*	*Up*	*Down*	*Avg*	*Peak*
0.300 B	7.63	1.5	1.03	2.43	5.69	696.8	802.2
0.690 B	21.59	2.2	3.23	4.13	15.29	789.3	972.1
2.070 B	65.97	1.8	3.06	9.87	62.10	1134	1587

Summary

- Fixed size tests, 80% efficiency up to 256 processors
- Isogranular tests, good efficiency up to 3000 processors
- 1.6 Tflops peak / 1.13 Tflops sustained for 2.1 billion unknowns
- Tree construction bottleneck for 1024+ processors
- Some load imbalance in non-uniform distribution

Outline

Classical FMM

New algorithm

Parallel algorithm

Boundary integral equation solver

3D BIE Solver

Properties

- Uses second-kind integral formulation
- Works for Laplace, Stokes equations
- General smooth domain boundary
- Based on Nyström discretization
- High-order accurate
- Efficient

Components

High-order accuracy → high-order boundary representation

- Manifold-based boundary representation method (ACM Transaction of Graphics 23(3), ’04)

Smooth (dense) part of the integral operator

- Kernel independent FMM algorithm

Singular (diagonal) part of the integral operator

- Use local partition of unity to isolate singularity
- Local high-order integration in polar coordinates
- Similar to (Bruno and Kunyansky ’01)

Summary

New kernel-independent FMM

- Use only kernel evaluations
- Extend FMM to general kernels
- Efficiency and accurate

MPI based parallel implementation

- Scale up to 3000 processors
- 1.6 Tflops peak / 1.13Tflops sustained performance on 2.1 billion unknowns

3D high-order BIE solver

- Laplace and Stokes equation
- Smooth boundary

Thank you