Efficient GPU parallelization of the Fast Multipole Method with periodic boundary conditions
Fast Multipole Method (FMM)

Calculation of long-ranged forces in the n-body problem (Greengard and Rokhlin, 1987)

- Tree – based approach
- $O(n)$ complexity
  \[ n := \text{number of particles} \]
- Multipole expansion
  of long-range interactions
Fast Multipole Method (FMM)

1/r Long-Range interactions

Molecular Dynamics

Plasma Physics

Astrophysics
Molecular Dynamics Simulation

Simulations on the atomistic level
Molecular Dynamics Simulation

Describing the energy of the system

Bonded interactions

\[ E = \sum_{\alpha} E_{\alpha} + \sum_{ij} (E_{ij}^{\text{Coul.}} + E_{ij}^{\text{vdW}}) \]
Molecular Dynamics Simulation

Describing the energy of the system

Non-bonded interactions

\[ E = \sum_{\alpha} E_\alpha + \sum_{ij} \left( E_{ij}^{\text{Coul.}} + E_{ij}^{\text{vdW.}} \right) \]
Molecular Dynamics Simulation

Describing the energy of the system

Non-bonded interactions

\[ E = \sum_{\alpha} E_\alpha + \sum_{ij} (E_{ij}^{\text{Coul.}} + E_{ij}^{\text{vdW}}) \]

N-body problem \( \mathcal{O}(n^2) \) + periodic boundaries

08.05.2017

B. Kohnke
Particle Mesh Ewald (PME)

**Basic idea of PME**

- Splitting the computation of electrostatic potential in two parts

- Direct
  - Compute the particle-particle interactions directly within a cutoff

- Reciprocal (FFT based)
  - Extrapolate charges on the grid
  - FFT of the charge grid
    - Computation $O(n \log n)$
    - Communication $O(\text{nodes}^2)$

\[ E = E_{direct} + E_{reciprocal} \]
PME vs FMM

Massive parallelization, 150000 Particles

Parallel efficiency

PME

FMM (Near Field)

Efficiency

replication factor

08.05.2017

B.Kohnke
Basic Idea

Classical $O(n^2)$ approach
Basic Idea

Classical $O(n^2)$ approach

Tree-code $O(n \log n)$
Basic Idea

Classical $O(n^2)$ approach

Tree-code $O(n \log n)$

FMM $O(n)$
FMM – Parameters

Controlling the accuracy of the approximation and performance

$d$ – depth of the FMM tree
FMM – Parameters

Controlling the accuracy of the approximation and performance

ws – separation criterion
FMM – Parameters

Controlling the accuracy of the approximation and performance

p – multipole order

\[
\frac{1}{d} = \sum_{l=0}^{p} \sum_{m=-l}^{l} \left[ \frac{a^l}{(l + m)!} P_{lm}(\cos \alpha) \right] \left[ \frac{(l - m)!}{r^{l+1}} P_{lm}(\cos \theta) \right] e^{-im(\beta - \phi)}
\]
FMM – Workflow

Preprocessing

Particle input – positions and charges (x, y, z, q)

Set algorithm parameters

- \( ws \) – separation criterion
- \( p \) – multipole order
- \( d \) – tree depth
FMM – Workflow

Preprocessing

Clustering particles

Build a tree according to chosen parameter $d$

- $ws$ – separation criterion
- $p$ – multipole order
- $d$ – tree depth
Preprocessing

Particle to multipole (P2M)

Expand particle positions and charges to multipole moments $\omega$

- $O(np^2)$ – operation
- Fill boxes on the lowest level
FMM – Workflow

Pass 1

Multipole to multipole (M2M)

Translate the multipole moments $\omega$ up the FMM-tree

- $O(p^4)$ – operation
- One operation per box
- Vertical operator

$$\omega (a + b) = \sum_{l=0}^{p} \sum_{m=-l}^{l} \sum_{j=0}^{l} \sum_{k=-j}^{j} \omega_{jk} (a) O_{l-j, m-k} (b)$$
FMM – Workflow

Pass 2

Multipole to local (M2L)

Transform multipole moments $\omega$ into Taylor moments $\mu$

- $\mathcal{O}(p^4)$ – operation
- 189 operations per box ($ws = 1$)
- Horizontal operator

$$
\mu(b - a) = \sum_{l=0}^{p} \sum_{m=-l}^{l} \sum_{j=0}^{l} \sum_{k=-j}^{j} \omega_{jk}(a) M_{l+j,m+k}(b)
$$
FMM – Workflow

Pass 3

Local to local (L2L)

Translate local moments $\mu$ down the tree

- $O(p^4)$ – operation
- One operation per box
- Vertical operation

\[
\mu(r - b) = \sum_{l=0}^{p} \sum_{m=-l}^{l} \sum_{j=l}^{p} \sum_{k=-j}^{j} \mu_{jk}(r) M_{j-l,k-m}(b)
\]
FMM – Workflow

Pass 4 and 5

Forces computing

- Compute $\Phi_{FF}, F_{FF}, E_{FF}$
- $O(np^2)$ – operation

Far field contribution from $\mu$ and $\omega$

Near field contribution (particle to particle)

- Compute $\Phi_{NF}, F_{NF}, E_{NF}$
- $O(n_{cutoff}^2)$ – operation
Far field

Coefficient Matrix, Generalized Storage Type

- Matrix size $\mathcal{O}(p^2)$
- Triangular shape
- One complex value per element

Used as storage for

- Multipole moments $\omega$
- Taylor moments $\mu$
- Operators $M$

Physical memory alignment
M2L Operation – Tree structure

Tree loop and Box – Neighbor Structure, \( ws=1 \)

**M2L Operation**

- For all boxes in the tree
  - Determine the interaction set
    - Children of direct neighbors of the own parent
  - Determine M2L operator
  - Compute one M2L operation
    - For each valid \( \omega, \mu \) pair
M2L Operation – Tree structure

Tree loop and Box – Neighbor Structure, ws=1

M2L Operation

- For all boxes in the tree
- Determine the interaction set
  - Children of direct neighbors of the own parent
- Determine M2L operator
- Compute one M2L operation
  - For each valid $\omega, \mu$ pair
M2L Operation – Tree structure

Tree loop and Box – Neighbor Structure, ws=1

M2L Operation

- For all boxes in the tree
- Determine the interaction set
  - Children of direct neighbors of the own parent
- Determine valid M2L operator
- Compute one M2L operation
  - For each valid \( \omega, \mu \) pair
M2L Operation – Tree structure

Tree loop and Box – Neighbor Structure, ws=1

M2L Operation

- For all boxes in the tree
- Determine the interaction set
  - Children of direct neighbors of the own parent
- Determine valid M2L operator
- Compute one M2L operation
  - For each valid \( \omega, \mu \) pair

\[
\mu(b - a) = \sum_{l=0}^{p} \sum_{m=-l}^{l} \sum_{j=0}^{l} \sum_{k=-j}^{l} \omega_{jk}(a)M_{l+j,m+k}(b)
\]
Translating multipole expansion to local expansion, $p^4$ loop structure

**One M2L operation**

\[
\mu_{lm}(b - a) = \sum_{j=0}^{l} \sum_{k=-j}^{j} \omega_{jk}(a) M_{l+j, m+k}(b)
\]

```
for (int l = 0; l <= p; ++l)
    for (int m = 0; m <= l; ++m)
    {
        omega_l_m=0;
        for (int j = 0; j <= p; ++j)
        {        
            for (int k = -j; k <= j; ++k)
            {    
                omega_l_m += M[m_idx](l+j, m+k) * omega[o_idx](j,k);
            }
        }
        mu[mu_idx](l, m) += omega_l_m
    }
```
M2L – GPU dynamic parallelism

Tree loop and Box – Neighbor Structure, ws=1

M2L Operation

- Start one parent kernel for each \( \omega \)
- Parent kernel
  - Computes the interaction set
  - Spawns the child kernels
- Child kernel
  - Compute one \( p^4 \) operation (\( p^2 \) threads)
M2L – GPU dynamic parallelism

Tree loop and Box – Neighbor Structure, ws=1

M2L Operation

- Start one parent kernel for each $\omega$

- Parent kernel
  - Computes the interaction set
  - Spawns the child kernels

- Child kernel
  - Compute one $p^4$ operation ($p^2$ threads)

```
parent_kernel<<<(1, 1, 1)(3, 3, 3)>>>
```
M2L – GPU dynamic parallelism

Tree loop and Box – Neighbor Structure, ws=1

M2L Operation

- Start one parent kernel for each $\omega$

- Parent kernel
  - Computes interaction set
  - Spawns the child kernels

- Child kernel
  - Compute one $p^4$ operation ($p^2$ threads)

$parent\_kernel\lll(1,1,1)(3,3,3)\rrr$
$child\_kernel\lll(2,2,2)(p,p,1)\rrr$

- Blocksize too small for small $p$ values
- Small grids (streams help to utilize the GPU)
M2L Operation – Internal Structure

Translating multipole expansion to local expansion, $p^4$ loop structure

One M2L operation

$$\mu_{lm}(\mathbf{b} - \mathbf{a}) = \sum_{j=0}^{l} \sum_{k=-j}^{j} \omega_{jk}(\mathbf{a})M_{l+j,m+k}(\mathbf{b})$$

for (int $l = 0$; $l <= p$; ++$l$)
for (int $m = 0$; $m <= l$; ++$m$)
{
    omega_l_m = 0;
    for (int $j = 0$; $j <= p$; ++$j$)
    {
        for (int $k = -j$; $k <= j$; ++$k$)
        {
            omega_l_m += M[m_idx](l+j, m+k) * omega[o_idx](j,k);
        }
    }
    mu[mu_idx](l, m) += omega_l_m
}
M2L Dynamic Scheme + Shared Memory

Depth 4, 4096 Boxes, periodic boundaries, 4096*189 = 774144 p^4 operations

![Diagram](https://via.placeholder.com/150)

- **Computational effort (estimate)**
- **Dynamic**

<table>
<thead>
<tr>
<th>multipole order</th>
<th>computational effort (estimate)</th>
<th>time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.2E-04</td>
<td>1.2E-04</td>
</tr>
<tr>
<td>2</td>
<td>2.0E-03</td>
<td>2.0E-03</td>
</tr>
<tr>
<td>3</td>
<td>3.1E-02</td>
<td>3.1E-02</td>
</tr>
<tr>
<td>4</td>
<td>4.9E-04</td>
<td>4.9E-04</td>
</tr>
<tr>
<td>5</td>
<td>7.8E-03</td>
<td>7.8E-03</td>
</tr>
<tr>
<td>6</td>
<td>1.3E-01</td>
<td>1.3E-01</td>
</tr>
<tr>
<td>7</td>
<td>5.0E-01</td>
<td>5.0E-01</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

08.05.2017
M2L – enhanced parallelization
Better tree abstraction

Idea – precompute interaction sets

- For each box in the tree store:
  - $\omega$
  - List of pointers to the targets $\mu_i$
  - List of pointers to the operators $M_i$

- Overhead only in initialization step
- Increased memory requirement
  - Redundant pointers storage
M2L – enhanced parallelization

Better tree abstraction

Idea – precompute interaction sets

- For each box in the tree store:
  - \( \omega \)
  - List of pointers to the targets \( \mu_i \)
  - List of pointers to the operators \( M_i \)

- Periodic boundary conditions
  - All interaction lists have the same size
M2L – precomputed interaction sets

Enhancing memory access patterns and index computation

Parallelization

- Precompute interactions lists
- Reorganize memory SoA → AoS
- Threading at tree level (for d >=3)
- One kernel launch per Symmetry_Type

```
Kernel<Symmetry_Type>//d=3,p=10
<<<(189,100,1)(64,1,1)>>>
```

# of operators p^2  # of boxes and p^2 (sequential)
M2L – precomputed interaction sets

Enhancing memory access patterns and index computation

Parallelization

Kernel<
Symmetry_Type>
// d=3, p=10
<<<(189, 100, 1)
(64, 1, 1)>>>.

- Threads within the same block share the Operator (shared memory)
- Nearly no index computation for $\omega$
  - $\omega_i = \text{ThreadId} + \text{offset}($Symmetry_Type$)$
- Targets $\mu_i$ accessed through list
  - Presorted to achieve coalescing
M2L – precomputed interaction sets

Depth 2, 64 Boxes, periodic boundaries, $64 \times 189 = 12096$ $p^4$ operations

![Graph showing computational effort vs. multipole order]
M2L – precomputed interaction sets

Depth 3, 512 Boxes, periodic boundaries, $512 \times 189 = 96768$ p$^4$ operations

- computational effort (estimate)
- dynamic
- presorted SoA

Time in seconds vs. multipole order

08.05.2017
B. Kohnke
M2L – precomputed interaction sets

Depth 4, 4096 Boxes, periodic boundaries, 4096*189 = 774144 \( p^4 \) operations

![Graph showing computational effort and time in seconds for different multipole orders. The graph includes lines for computational effort (estimate), dynamic, and presorted SoA.]
M2L – reduced operator set
Exploring operator symmetry

Symmetry of the operators

- The whole operator set contains 316 operators (3D)
- There are 56 unique operators (3D)
- Operators of the same length are rotational symmetric
- Only the signs of the symmetric counterparts elements differ
Exploring operator symmetry

Symmetry of the operators

- The whole operator set contains 316 operators (3D)
- There are 56 unique operators (3D)
- Operators of the same length are rotational symmetric
- Only the signs of the symmetric counterparts elements differ
M2L – reduced operator set

Exploring operator symmetry

Symmetry of the operators

- The whole operator set contains 316 operators (3D)
- There are 56 unique operators (3D)
- Operators of the same length are rotational symmetric
- Only the signs of the symmetric counterparts elements differ
M2L – reduced operator set

Exploring operator symmetry

Different symmetry groups

- The interaction sets are asymmetric
  - 8 distinct types according to the position within the parent box

- There are 4 symmetry groups (3D)
  - 7 operators are unique
  - 21 operators are 2-way symmetric
  - 21 operators are 4-way symmetric
  - 7 operators are 8-way symmetric
M2L – reduced operator set

Exploring operator symmetry

Different symmetry groups

- The interaction sets are asymmetric
  - 8 distinct types according to the position within the parent box
- There are 4 symmetry groups (3D)
  - 7 operators are unique
  - 21 operators are 2-way symmetric
  - 21 operators are 4-way symmetric
  - 7 operators are 8-way symmetric
M2L – reduced operator set

Exploring operator symmetry

Different symmetry groups

- The interaction sets are asymmetric
  - 8 distinct types according to the position within the parent box
- There are 4 symmetry groups (3D)
  - 7 operators are unique
  - 21 operators are 2-way symmetric
  - 21 operators are 4-way symmetric
  - 7 operators are 8-way symmetric
M2L – reduced operator set

Exploring operator symmetry

Different symmetry groups

- The interaction sets are asymmetric
  - 8 distinct types according to the position within the parent box
- There are 4 symmetry groups (3D)
  - 7 operators are unique
  - 21 operators are 2-way symmetric
  - 21 operators are 4-way symmetric
  - 7 operators are 8-way symmetric
M2L – precomputed interaction sets

Enhancing memory access patterns and index computation

Parallelization of reduced operator

- Preload the operator data into shared memory
  - Only once for the whole symmetry group
- Preload the bitsets of signs for the symmetrical counterparts
- One full dot product to get (1,2,4,8) results depending on symmetry group
M2L – precomputed interaction sets

Enhancing memory access patterns and index computation

Parallelization of reduced operator

- Preload the operator data into shared memory
  - Only once for the whole symmetry group
- Preload the bitsets of signs for the symmetrical counterparts
- One full dot product to get (1,2,4,8) results depending on symmetry group

\[ \chi_{0} \chi_{1} \chi_{2} \chi_{3} \Omega_{\omega} \mu \mu_{M} x, y x, y x, y \]

08.05.2017

B.Kohnke 49
M2L – precomputed interaction sets

Enhancing memory access patterns and index computation

Parallelization of reduced operator

- Preload the operator data into shared memory
  - Only once for the whole symmetry group
- Preload the bitsets of signs for the symmetrical counterparts
- One full dot product to get (1,2,4,8) results depending on symmetry group

1110110101
0110110101

1010011101
110110101

08.05.2017
B.Kohnke
M2L – precomputed interaction sets

Enhancing memory access patterns and index computation

Parallelization of reduced operator

- **Gridsize** reduced by a factor of ~ 4
- Logic added to get and change signs of the result

```
Kernel<Symmetry_Type, Operator> <<<(7,100,1)(64,1,1)>>>(...)
Kernel<Symmetry_Type, Operator> <<<(21,100,1)(64,1,1)>>>(...)
Kernel<Symmetry_Type, Operator> <<<(21,100,1)(64,1,1)>>>(...)
Kernel<Symmetry_Type, Operator> <<<(7,100,1)(64,1,1)>>>(...)
```
M2L – reduced operator

Depth 4, 4096 Boxes, periodic boundaries, 4096*189 = 774144 p⁴ operations
M2L – precomputed interaction sets

Enhancing memory access patterns and index computation

Parallelization of reduced operator

- Uncoalesced memory access for the symmetrical counterparts targets
- Solution
  - Store coalesced pointers for all (l,m) – targets

Kernel<Symmetry_Type, Operator><<<<(7,100,1)(64,1,1)>>>(…)
Kernel<Symmetry_Type, Operator><<<<(21,100,1)(64,1,1)>>>(…)
Kernel<Symmetry_Type, Operator><<<<(21,100,1)(64,1,1)>>>(…)
Kernel<Symmetry_Type, Operator><<<<(7,100,1)(64,1,1)>>>(…)

08.05.2017

B.Kohnke
M2L – reduced operator

Depth 4, 4096 Boxes, periodic boundaries, 4096*189 = 774144 p^4 operations

![Graph showing computational effort vs. multipole order](image)
FMM vs GROMACS - 285119 particles – channel protein TehA

FMM – DEPTH 3, 512 BOXES
Intel i7-6800 CPU, 4 Cores (8 Hyperthreads), GeForce GTX 1060
GROMACS – fourierspacing 0.12 nm, cutoff 1.0 nm, interpolation order 4

08.05.2017
B.Kohnke
FMM vs GROMACS - 285119 particles – channel protein TehA

FMM – DEPTH 4, 4096 BOXES
Intel i7-6800 CPU, 4 Cores (8 Hyperthreads), GeForce GTX 1060
GROMACS – fourierspacing 0.12 nm, cutoff 1.0 nm, interpolation order 4

08.05.2017
B.Kohnke

0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
1
2
3
4
5
6
7
8
9
10
11
time in seconds
FORCES
L2L
LATTICE
M2L
M2M
P2M
COPY AND FLUSH
DISTRIBUTE PARTICLES
P2P
GROMACS
P2P
FMM vs GROMACS - 285119 particles – channel protein TehA

**FMM – DEPTH 3.32, 1000 BOXES**
Intel i7-6800 CPU, 4 Cores (8 Hyperthreads), GeForce GTX 1060
GROMACS – fourierspacing 0.12 nm, cutoff 1.0 nm, interpolation order 4

08.05.2017
B.Kohnke
Summary

All stages of the FMM run on a GPU
- Distributing particles
- Building multipoles
- M2M (*compute bound*)
- M2L (*compute bound*)
- L2L (*compute bound*)
- Lattice (dealing with periodicity)
- Computing far-field forces
- P2P (*compute bound*)

Enhancements are still possible
- M2L (exploring more symmetry)
- Redistributing the work between the CPU and GPU (CPU is nearly idle)
Thank You

Questions?