Fast Algorithms for Many-Particle Simulations on Peta-Scale Parallel Architectures

Godehard Sutmann
Institute for Advanced Simulation (IAS) - Jülich Supercomputing Centre (JSC)
Research Centre Jülich

g.sutmann@fz-juelich.de
Outline

• Many-Body simulations
• Methods for long range interactions in many-body systems
• Multigrid Method
• Barnes-Hut Tree Method
• Fast Multipole Method
• Coupling Methods to Codes
Outline

• Many-Body simulations
• Methods for long range interactions in many-body systems
• Multigrid Method
• Barnes-Hut Tree Method
• Fast Multipole Method
• Coupling Methods to Codes
“Atomistic” Models of Matter

Li-F in water

Galaxy
Molecular dynamics

Particle systems are described by a Hamiltonian

\[ H_0 = \frac{1}{2} p^T M^{-1} p + \frac{1}{2} \pi^T \theta^{-1} \pi + \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} u(r_{ij}) + \sum_{i=1}^{N-2} \sum_{j=i+1}^{N-1} \sum_{k=j+1}^{N} u^{(3)}(r_{ij}, r_{ik}, r_{jk}) + \ldots \]

- **Kinetic Energy**
- **Potential Energy**

Particle trajectories are generated by solving classical equations of motion

\[ \dot{q} = \frac{d}{dp} H(q, p) \quad \quad \dot{p} = -\frac{d}{dq} H(q, p) \]
Long range interactions

Typical form for a force field (AMBER / Sander)

\[ U = \sum_{\text{bonds}} K_r (r - r_{eq})^2 + \sum_{\text{angles}} K_\theta (\theta - \theta_{eq})^2 + \sum_{\text{dihedrals}} \frac{V_n}{2} [1 + \cos(n\phi - \gamma)] \]

\[ + \sum_{i<j} \left[ \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^6} \right] + \sum_{\text{H-bonds}} \left[ \frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^6} \right] + \sum_{i<j} \frac{q_i q_j}{r_{ij}} \]

Classification of potentials

\[ I = \int_{\mathbb{R}^D} d^D r \frac{A}{r^n} = \begin{cases} \infty : n \leq D & \text{long range} \\ \text{finite} : n > D & \text{short range} \end{cases} \]

- Examples: Coulomb interactions (charge-charge, dipole-dipole)
- Gravitational potential
Long Range Interactions

Potential energy calculations

\[ U = \frac{1}{2} \sum_{i=1}^{N} q_i \phi_i = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{q_i q_j}{\| r_i - r_j \|_2} = \frac{1}{2} q^T A q \]

\[ A_{ij} = \frac{1}{\| r_i - r_j \|_2} \]

\[ q^T = \{ q_1, \ldots, q_N \} \]

\[ O(N^2) \]
Long Range Interactions

Hierarchical potential energy calculations

\[ U = \sum_{i=1}^{N} q_i \phi_i = \sum_{i=1}^{N} \sum_{j=1}^{\log(N)} q_i A_{ij} \]

\[ \mathcal{O}(N \log(N)) \]

\[ U = \sum_{i=1}^{N} q_i \phi_i = \sum_{i=1}^{N} q_i B[i] \]

\[ \mathcal{O}(N) \]
Outline

- Many-Body simulations
- Methods for long range interactions in many-body systems
- Multigrid Method
- Barnes-Hut Tree Method
- Fast Multipole Method
- Coupling Methods to Codes
Methods for long range interactions

Periodic systems

- Grid free:
  - Plain Ewald summation → \( O(N^{3/2}) \)
  - Fast multipole method (FMM) → \( O(N) \)
  - Tree-algorithms (Barnes-Hut) → \( O(N \log(N)) \)

- Grid based:
  - Particle Mesh Ewald (PME) → \( O(N \log(N)) \)
  - Particle-Particle Particle-Mesh Method (P³M) → \( O(N \log(N)) \)

Open systems

- Grid free:
  - Tree-algorithms (Barnes-Hut) → \( O(N \log(N)) \)
  - Fast multipole method (FMM) → \( O(N) \)

- Grid based:
  - Multigrid based method (P³MG) → \( O(N) \)

List not complete…
Classification of Algorithms

- Fully Hierarchical
  - complete subdivision of space
  - continuous description from sources to far-field
  - hierarchical ordering of field contributions
  - mesh-less description
  - sources: charges, multipoles
  - res.: physical potentials, fields
  - expl: Fast Multipole Method, Barnes-Hut Tree Method

- Hierarchical Splitting Methods
  - complete subdivision of space
  - splitting between short- and long-range contributions
  - fast methods to solve long range contributions
  - (often) grid-based description
  - sources: pseudo-charges
  - res.: lr: pseudo-potentials / -fields
  - sr: potentials + corrections
  - expl: P3M, SPME, multigrid
Why different methods?

• **Multigrid**
  ▪ Elliptic PDE solver (Poisson, Helmholtz) & non-linear (Poisson-Boltzmann)
  ▪ allows for dielectric shading
  ▪ prescription of desired boundary potentials (capacitors)

• **Barnes-Hut Tree Method**
  ▪ strongly inhomogeneous particle distributions
  ▪ very complex system geometries

• **Fast Multipole Method**
  ▪ high accuracy calculations
  ▪ full error control

*All methods are scalable on parallel architectures*
Schematic differences of hierarchical methods

- Multigrid

- Barnes-Hut Tree method

- Fast multipole method
Acknowledgements

- FMM
  - Holger Dachsel, Ivo Kabadshow

- Barnes-Hut
  - Paul Gibbon, Lukas Arnold, Robert Speck, Mathias Winkel

- Multigrid
  - Matthias Bolten, Bernhard Steffen

- Parallel Sorting
  - Michael Hofmann (TU Chemnitz)

- Interface
  - Lidia Westphal, Rene Halver

Financial support from BMBF
Outline

• Many-Body simulations
• Methods for long range interactions in many-body systems
• Multigrid Method
• Barnes-Hut Tree Method
• Fast Multipole Method
• Coupling Methods to Codes
Multigrid for Poisson Equation

- Solve PDE on the finest grid with Dirichlet boundary conditions
  \[ \Delta u(r) = -f(r) \quad , \quad r \in \Omega \quad \quad u(r) = g \quad , \quad r \in \Gamma \]
- Use finite differences to approximate PDE
  \[ \Delta u(r) = -f(r) \quad \Rightarrow \quad L_h u_h = -f_h \]
- Get an approximation for the potential from where an error \( \epsilon \) and residuum \( r \) is obtained
  \[ r_h = L_h \tilde{u}_h + f_h \quad \quad \epsilon_h = u_h - \tilde{u}_h \]
- These obey the same differential equation
  \[ L_h \epsilon_h = -r_h \quad \quad \epsilon_h(r) = 0 \quad , \quad r \in \Gamma_h \]
- Get correction to approximation: \( \tilde{u}_h = \tilde{u}_h + \epsilon_h \)
  (repeat until convergence)
Multigrid for Poisson Equation

- Multigrid: solve for the error onto a hierarchy of coarse grids $2^l h$

  Low frequency components on the fine grid get high frequency components on the coarse grid (fast error reduction)

- Forward MG: Restrict residuum

- Backward MG: Prolongate residuum

  \[ L_H \epsilon_H = I_H^H r_h \]
  \[ L_H = I_H^H L_h I_H^h \]
  \[ L_h \epsilon_h = I_H^h r_H \]

  e.g. V-cycle
Complexity

Natural complexity of long-range pair-wise interactions is $O(N^2)$

Multigrid solver reduces complexity of field calculations to $O(N)$

\[
N_g = \alpha N
\]

\[
\#(ops) \propto n_{iter} N_g \left( 1 + \frac{1}{8} + \frac{1}{64} + \frac{1}{512} + \ldots \right)
\]

\[
\approx 1.1428 \, n_{iter} N_g
\]

$\rightarrow O(N_g)$

$\rightarrow O(N)$
Convergence of Multigrid

Test system of 4096 particles with \((2^5+1)^3\) grid points

Multigrid calculation
Convergence of Multigrid

Test system of 4096 particles with \((2^5)^3\) grid points

Single grid calculation

![Graph showing convergence of Multigrid](image)
Electrostatics: finite difference solution for point particles

- In molecular simulations, source terms are point charges, which create a potential, diverging at the origin
- Sampling problem on a discrete grid
- Resolution problem depending on grid spacing
- Large error close to the source term
- Also far field poisoned by discretization errors
Electrostatic Regularization

- Splitting of the potential into near- and far-field (analogon to Ewald summation)

\[ \Delta \Phi_s(r) = -4\pi \sum_{i=1}^{N} q_i \rho_s(\|r - r_i\|_2) \quad \int_{\Omega} d^3r \, \rho_s(r) = 1 \]

\[ \phi(r_i) = \Phi_s(r_i) - \phi_s(0) + q_i \sum_{j \in \text{supp}} \left\{ \frac{q_j}{\|r_i - r_j\|_2} - \phi_s(\|r_i - r_j\|_2) \right\} \]
### Scaling

<table>
<thead>
<tr>
<th># particles</th>
<th>grid size</th>
<th>( \frac{E_{\text{pot}} - E_{\text{pot}}^<em>}{E_{\text{pot}}^</em>} )</th>
<th>time/s</th>
<th>sampling</th>
<th>solution of PDE</th>
<th>back interp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>33(^3)</td>
<td>1.579 \times 10^{-2}</td>
<td>0.25</td>
<td>0.14</td>
<td>0.16</td>
<td></td>
</tr>
<tr>
<td>8000</td>
<td>65(^3)</td>
<td>1.989 \times 10^{-3}</td>
<td>2.01</td>
<td>3.46</td>
<td>1.41</td>
<td></td>
</tr>
<tr>
<td>64000</td>
<td>129(^3)</td>
<td>1.033 \times 10^{-2}</td>
<td>16.34</td>
<td>35.18</td>
<td>12.29</td>
<td></td>
</tr>
<tr>
<td>512000</td>
<td>257(^3)</td>
<td>2.481 \times 10^{-3}</td>
<td>132.30</td>
<td>340.05</td>
<td>108.95</td>
<td></td>
</tr>
</tbody>
</table>

![Graph showing scaling results](image-url)
Multigrid Parallelization

- Exchange of neighbor grid points on each hierarchy level
- Data non-locality, examples

- \( N_g = 128^3 \), i.e. \((2^7)^3\), i.e. 7 MG-levels
- \( N_p = 512 \), i.e. \(8^3\), i.e. 5 local levels \((128, 64, 32, 16, 8)\)
  \( r \sim 2.5 \times 10^4 \)
- \( N_g = 2048^3 \), i.e. \((2^{11})^3\), i.e. 11 MG-levels
- \( N_p = 65536 \), i.e. \(32^3\), i.e. 7 local levels \((2048, 1024, 512, 256, 128, 64, 32)\)
  \( r \sim 2 \times 10^6 \)
Parallel Fast Multipole Method

- Parallelization:
  - Domain decomposition
  - MPI (cartesian communicators)
  - Asynchronous communications
  - only nearest neighbor communication

Strong scaling: \( N=256^3 \)
7-point stencil

8\(^3\) unknowns/core
Outline

• Many-Body simulations
• Methods for long range interactions in many-body systems
• Multigrid Method
• Barnes-Hut Tree Method
• Fast Multipole Method
• Coupling Methods to Codes
Tree Algorithms

- Organize physical space in a hierarchical way (relative to each particle)
- Interactions are weighted according to their importance
- Trees may be organized in different ways, e.g. 2\text{d} trees: binary-, quad-, oct-trees in 1,2,3 dimensions
- Principle is to sort particles into cells, where on the lowest level only one particle is inside a cell
- Original work by Barnes & Hut
Tree Algorithms: grouping criterion

• Interaction between individual particles and distant pseudo-particles (multipoles)

• Which criterion is adopted for grouping?

If $s/d < \theta$: internal structure of the pseudoparticle is ignored

If $s/d > \theta$: the node is resolved and the criterion is checked for every child

At latest this procedure stops, when a leaf node is reached
Sorted list of keys maps 3D particle coordinates onto 1D space-filling curve

Sort keys according to their size in ASCII representation. This sequence will be a space filling curve (Morton, Z, Lebesque - curve)
PEPC

- Barnes-Hut algorithm implemented into **PEPC** (Pretty Efficient Parallel Coulomb-Solver)
  - developed by Paul Gibbon (JSC) for plasma-laser interaction
  - reduction of the n-body $O(N^2)$ complexity to $O(N \log N)$
- OpenSource, freely available
- F90 and MPI
- part of ScaFaCoS library
- part of DEISA and PRACE benchmark suites
Scientific Application

- PEPC is highly flexible and comes in different flavours
  - PEPC-E
    Coulomb interaction
  - PEPC-B
    Magnetic fields
    laser-plasma interaction
  - PEPC-V
    vortex fluid methods
  - PEPC-G
    Newtonian gravitation
    extends to smooth particle hydrodynamics
Parallel Performance (pure MPI version)

PEPC on IBM BG/P
Hybrid PEPC

- MPI + POSIX threads
- reduce metadata overhead
- communication thread + multiple worker threads
- overlap of communication and computation
- first successful runs on 300k cores of jugene
- system size $N > 2 \times 10^9$
Outline

- Many-Body simulations
- Methods for long range interactions in many-body systems
- Multigrid Method
- Barnes-Hut Tree Method
- Fast Multipole Method
- Coupling Methods to Codes
Long range contributions

For general orientations of coordinate vectors the expansion is

\[
\frac{1}{|r - a|} = \sum_{l=0}^{\infty} \sum_{m=-l}^{m=l} \frac{(l - m)!}{(l + m)!} \frac{a^l}{r^{l+1}} P_{lm}(\cos \alpha) P_{lm}(\cos \theta) e^{-im(\beta - \varphi)}
\]

Define

\[
\frac{1}{|r - a|} = \sum_{l=0}^{\infty} \sum_{m=-l}^{m=-l} O_{lm}(a) M_{lm}(r)
\]
Extensions: rotation-based FMM

- Conventional FMM has complexity $O(L^4)$ with respect to the multipole length

\[ O_{lm}(a, \alpha, \beta) = \frac{1}{(l+m)!} a^l P_{lm}(\cos \alpha) e^{-im\beta} \]

\[ M_{lm}(r, \theta, \varphi) = (l-m)! \frac{1}{r^{l+1}} P_{lm}(\cos \theta) e^{im\varphi} \]

- Costs can be reduced to $O(L^3)$ if Legendre polynomials are used

\[ O_{lm}(a, 0, 0) = \frac{1}{(l+m)!} a^l \delta_{m0} \]

\[ M_{lm}(r, 0, 0) = (l-m)! \frac{1}{r^{l+1}} \delta_{m0} \]
Fast electrostatics – Fast Multipole Method

• FMM is subdivided into 5+1 passes
  ▪ Pass 0: determine optimal FMM parameter set
  ▪ Pass 1: form and shift multipole expansions
  ▪ Pass 2: transform distant multipole expansions
  ▪ Pass 3: shift Taylor like expansions
  ▪ Pass 4: calculate far field energy
  ▪ Pass 5: calculate near field energy

Enhanced FMM Workflow
Fast Multipole Method

- **Pass 0**
  - Determine FMM parameters
    - *Seperation criteria ws*
    - *Depth of the FMM tree*
    - *Length of the multipole expansion L*
  - Optimisation of run time behavior
    \[
    \frac{\partial t}{\partial w_s} = 0 \quad \frac{\partial t}{\partial D} = 0 \quad \frac{\partial t}{\partial L} = 0
    \]
    \[
    \Delta E(D, L, w_s) \leq \epsilon
    \]

- **Pass 4**
  - Sum up far field energies
    \[
    E_{FF} = \sum_{i_{box}} \sum_{l=0}^{L} \sum_{m=-l}^{l} \omega_{lm} \mu_{lm}
    \]

- **Pass 5**
  - Sum up near field energies
    \[
    E_{NF} = \frac{1}{2} \sum_{i_{box}} \sum_{i=1}^{N_{i_{box}}} \sum_{j_{box}=1}^{N_{j_{box}}} \sum_{j=1}^{N_{i_{box}} (ws+1)^3 - 1} \frac{q_i q_j}{\|r_i - r_j\|_2}
    \]
Fast electrostatics – Fast Multipole Method

- Computational complexity: \[ 1 + \frac{1}{8} + \frac{1}{64} + \ldots \approx 1.1 \rightarrow O(N) \]

Every Pass consists of operations of \( O(N) \),

- High Precision Cross-Over
  - \( \Delta E = 10^{-12} \)
  - 4000 particles

- Low Precision Cross-Over
  - \( \Delta E = 10^{-3} \)
  - 500 particles

-> Fast Multipole Method has optimal scaling with number of particles!
Periodic Boundary Conditions

- Implementation straightforward (although involved) for 1d, 2d, 3d-periodicity

**Scaling of O(N) vs. Full error control**

- Graph showing CPU time vs. number of particles for different periodic conditions.
- Graph showing FMM relative error vs. requested relative error for different Madelung models.

**Legend:**
- Red: Open system
- Green: 1d - periodic
- Blue: 2d - periodic
- Pink: 3d - periodic
- Cyan: Ideal
FMM Parallelization

- Utilize data locality
- Use fastest one-sided memory access (put)
- Employ blocking and non-blocking data transfer
- Use MPI for collectives only
- Avoid accumulate and non-contiguous send operations
- Use minimal communication functionality (put/reduce/gather)
- Exploit Fortran-C interoperability
- Use thin-wrapper to map C library functions to Fortran
- Avoid direct access to communication library functions
Parallel Fast Multipole Method

• Parallelization:
  - Remote memory access (RMA)
  - Point-to-point:
    One sided communication
      - *Put, Fence, Notify, Notifywait*
  - Contiguous data transfer
    - ARMCI / A1
  - Collective operations
    - MPI
      - *Allgather, Allreduce, Barrier*
Parallel Fast Multipole Method

Weak scaling on Jugene and Jupopa
(Load: $8^6=262144$ particles/core)

![Graph showing weak scaling on Jugene and Jupopa](image)
Parallel Fast Multipole Method

Strong scaling on Jugene  (Load: $N = 1024^3$)
Parallel Fast Multipole Method

Strong scaling on Juropa  (Load: N = 1024^3)
FMM World record (Dec. 2010)

- After memory optimization: \( \approx 50 \) Bytes / particle
- \( 512 \) Mbytes / core \( \times \) \( 294912 \) cores / \( 50 \) Bytes / particle
  \[ \approx 3.0 \times 10^{12} \] particles
- Largest simulation with
  \[ N = 14441^3 \] particles
  \[ = 3011561968121 \]
  \[ = 3.0 \) Trillion \]
Outline

• Many-Body simulations
• Methods for long range interactions in many-body systems
• Multigrid Method
• Barnes-Hut Tree Method
• Fast Multipole Method
• Coupling Methods to Codes
BMBF Project ScaFaCoS
Scalable Library for Fast Coulomb Solver

Coordinator:
• Forschungszentrum Jülich (IAS/JSC)

University Partners:
• Bonn
• Chemnitz
• Stuttgart
• Wuppertal

Research Centres:
• MPI Mainz (Polymer)

Industry:
• BASF Ludwigshafen
• Cognis Düsseldorf (BASF)
• IBM Dresden
Scalable Fast Coulomb Solvers

- libLR
  - Communication
  - Ewald
  - P3M
  - FMM
  - BHT
  - P3MG
  - Fast summation

- libSR
  - Communication
  - Neighbor search
  - Potential fcts

- Particle2Grid
- Multipole
- Raumf. Kurven
- ...
Insert essential solver-specific data into a new FCSHandle-object

Call fcs_init to set method and MPI communicator

Calls:
- fcs_FMM_init
- fcs_P3M_init
- fcs_PP3MG_init
- fcs_NFFT_init
- fcs_PEPC_init
- fcs_direct_init

Insert optional solver-specific data into FCSHandle-object

Call fcs_tune to calculate solver-specific parameters based on positions / charges

Calls:
- fcs_FMM_tune
- fcs_P3M_tune
- fcs_PP3MG_tune
- fcs_NFFT_tune
- fcs_PEPC_tune
- fcs_direct_tune

Call fcs_run to calculate a time step with the chosen solver

Calls:
- fcs_FMM_run
- fcs_P3M_run
- fcs_PP3MG_run
- fcs_NFFT_run
- fcs_PEPC_run
- fcs_direct_run

Next time step with same configuration?

Yes

Change solver?

No

Calculate another time step at all?

Yes

Call fcsHandle_destroy to deallocate FCSHandle-object

Simulation ends
Thank you for your Attention