FMM based solution of electrostatic and magnetostatic field problems on a PC-cluster

André Buchau
Wolfgang Hafla
Friedemann Groh
Wolfgang M. Rucker
Outline

- Introduction
- Vectorization
- Multithreading
- Multiprocessing
- Numerical examples
- Conclusions
Introduction

Numerical formulation

- Electrostatic field problems
- Boundary element method
- Indirect formulation based on charges
- Galerkin method
- Second order boundary elements
- Iterative solver GMRES with Jacobi preconditioner
- Fast multipole method
Introduction

Direct BEM formulation

- Electrostatics
- Steady current flow fields
- Green’s theorem

\[ c(r)u(r) = \oint \frac{\partial u(r')}{\partial n'} \frac{1}{|r - r'|} dA' - \oint u(r') \frac{\partial}{\partial n'} \frac{1}{|r - r'|} dA' \]

- Dirichlet boundary conditions
- Neumann boundary conditions
Introduction

Indirect BEM formulation

- Electrostatics
- Magnetostatics
- Charge densities

\[ u(r) = \frac{1}{4\pi \varepsilon_0} \int_A \frac{\sigma(r')}{|r-r'|} \, dA' \]

- Dirichlet boundary conditions
- Neumann boundary conditions
Introduction

Initial situation
- BEM with compressed matrix
- Very high compression rates (90 % to 99 %)
- Typical problem size: 10000 to 100000 unknowns
- Typical memory requirements: 100 MByte to 1 GByte
- Computing time for linear problems: up to a few hours
- Computing time for nonlinear problems: up to a few days

Aim of parallelization
- Reduction of computing time
Vectorization

Properties

- Parallel execution of multiple instructions on a single CPU
- Hardware and compiler dependent
- Recommended for dense data structures

\[ a \times b_1 \quad a \times b_2 \quad a \times b_3 \]

\[ a \times b_1 \quad a \times b_2 \quad a \times b_3 \]
Vectorization

Multipole transformations

- Classical multipole-to-local transformation
- Dense transformation matrix
- Processor optimized numerical libraries
- \( O(L^4) \)

\[
L^m_n = \sum_{k=0}^{L} \sum_{l=-k}^{k} M^l_k \frac{j^{m-l-|m|-|l|}}{A^l_k A^m_n Y^{l-m}_{k+n}(\mu, \nu)} \frac{(-1)^k}{\rho^{k+n+1}} A^{l-m}_{n+k}
\]

\[
\{L\} = \left[T_{M2L}\right]\{M\}
\]
Vectorization

Multipole transformations

- Modified multipole-to-local transformation
- Sparse transformation matrices
- $O(L^3)$

$$[T] = [z_{\text{inv}}][y_{\text{inv}}][M 2L_z][y][z]$$

$$M'^m_n = M^n_m e^{im\beta}$$

$$M'^m'_n = \sum_{m=-n}^{-1} R(n, m, m', \alpha)(-1)^m (M^n_m)^* + \sum_{m=0}^{n} R(n, m, m', \alpha) M^n_m$$

$$L^m_n = \sum_{k=m}^{L} M^m_k \frac{Y^0_{k+n}(0, 0)(-1)^{k+m}(n+k)!}{\rho^{k+n+1} \sqrt{(k-m)!(k+m)!(n-m)!(n+m)!}}$$
Vectorization

Summary

- Fast for dense operations
- Sparse operators are faster
- Use all special properties of the operators
- Reduce number of sub-operations
Multithreading

Properties

- Easy to implement
- Only time consuming parts are parallelized
- Dynamic load distribution during runtime
- Shared memory access
Multipole transformations

- Operations of the octree cubes can be computed independent of each other
Properties

- Whole program runs in parallel
- Deterministic algorithm for load distribution
- Synchronization between processes
Multithreading

Multipole transformations

- Data transfer between processes is necessary
Numerical examples

Hardware

2 Intel Pentium III

2 Intel Pentium III

Ethernet
Numerical examples

Coated sphere

- 9892 second order quadrilateral elements
- 29680 unknowns
- 66 linear iteration steps
- 280 MByte
- Homogeneous mesh
Numerical examples

Coated sphere

- Potential at a radial line
Numerical examples

Coated sphere

- Computing times

<table>
<thead>
<tr>
<th></th>
<th>Serial</th>
<th>OpenMP</th>
<th>OpenMP + MPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processes</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Threads</td>
<td>1</td>
<td>2</td>
<td>2*2</td>
</tr>
<tr>
<td>Matrix assembly</td>
<td>2931 s</td>
<td>1466 s</td>
<td></td>
</tr>
<tr>
<td>Reduction</td>
<td></td>
<td>50 %</td>
<td></td>
</tr>
<tr>
<td>Solution linear</td>
<td>3851 s</td>
<td>2387 s</td>
<td>1268 s</td>
</tr>
<tr>
<td>equation system</td>
<td></td>
<td>38 %</td>
<td>67 % (38 % OpenMP, 47 % MPI)</td>
</tr>
<tr>
<td>Reduction</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Numerical examples

Gas insulated high voltage system
Numerical examples

Gas insulated high voltage system

- 9529 second order quadrilateral elements
- 28855 unknowns
- 178 linear iteration steps
- 305 MByte
- Problem oriented mesh
Numerical examples

Gas insulated high voltage system

![Graph showing electric field strength](image-url)
Numerical examples

Gas insulated high voltage system

- Computing times

<table>
<thead>
<tr>
<th></th>
<th>Serial</th>
<th>OpenMP</th>
<th>OpenMP + MPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processes</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Threads</td>
<td>1</td>
<td>2</td>
<td>2*2</td>
</tr>
<tr>
<td>Solution linear</td>
<td>6730 s</td>
<td>4194 s</td>
<td>3360 s</td>
</tr>
<tr>
<td>equation system</td>
<td></td>
<td></td>
<td>2548 s</td>
</tr>
<tr>
<td>Reduction</td>
<td>38 %</td>
<td>50 %</td>
<td>62 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(19 % MPI)</td>
<td>(39 % MPI)</td>
</tr>
<tr>
<td>Octree level</td>
<td></td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>
Conclusions

- Compressed BEM matrices (fast multipole method)
- Vectorization, multithreading, multiprocessing
- Reduction of computing time
- Easy-to-implement approach
- Limits of numerical algorithms in combination with parallelization methods