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

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

Direct BEM formulation

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

$$c(\boldsymbol{r})u(\boldsymbol{r}) = \oint \frac{\partial u(\boldsymbol{r}')}{\partial n'} \frac{1}{|\boldsymbol{r}-\boldsymbol{r}'|} \mathrm{d}A' - \oint u(\boldsymbol{r}') \frac{\partial}{\partial n'} \frac{1}{|\boldsymbol{r}-\boldsymbol{r}'|} \mathrm{d}A'$$

- Dirichlet boundary conditions
- Neumann boundary conditions

Indirect BEM formulation

- Electrostatics
- Magnetostatics
- Charge densities

$$u(\boldsymbol{r}) = \frac{1}{4\pi\varepsilon_0} \int_A \frac{\sigma(\boldsymbol{r}')}{|\boldsymbol{r}-\boldsymbol{r}'|} \mathrm{d}A'$$

- Dirichlet boundary conditions
- Neumann boundary conditions

Introduction

Initial situation

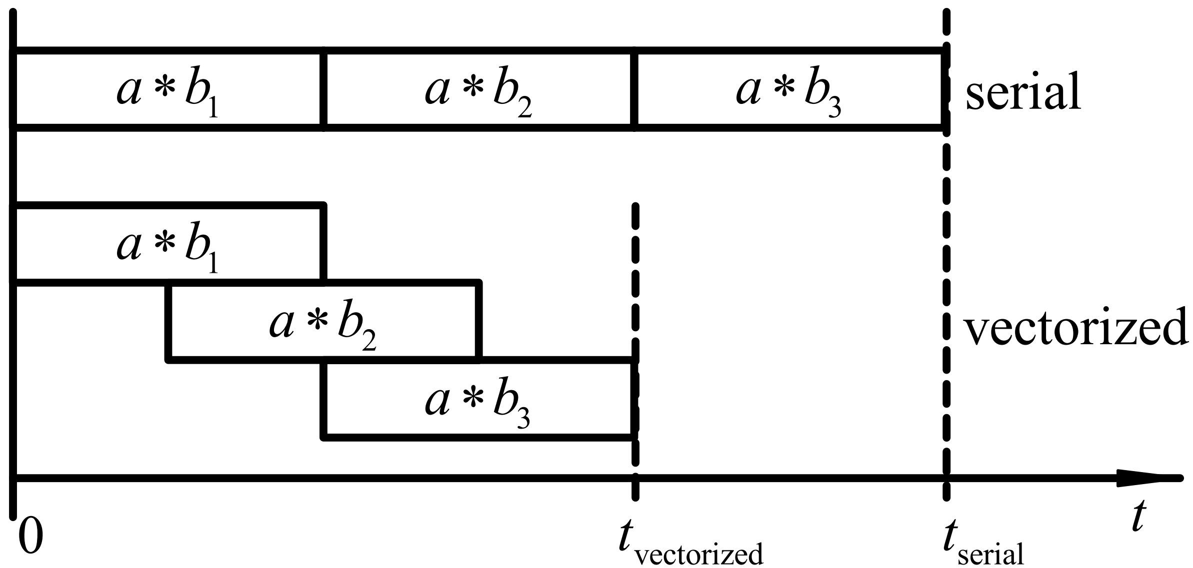
- BEM with compressed matrix
- Very high compression rates (90 % to 99 %)
- Typical problem size: 10000 to 100000 unknowns
- Typical memory requirements: 100 MByte to 1 GByte
- Computing time for linear problems: up to a few hours
- Computing time for nonlinear problems: up to a few days

Aim of parallelization

- Reduction of computing time

Properties

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



Multipole transformations

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

$$L_n^m = \sum_{k=0}^{L} \sum_{l=-k}^{k} \frac{M_k^l \, \mathrm{j}^{|m-l|-|m|-|l|} \, A_k^l A_n^m Y_{k+n}^{l-m}(\mu,\nu)}{(-1)^k \, \rho^{k+n+1} A_{n+k}^{l-m}}$$

$$\{L\} = [T_{M2L}]\{M\}$$

Multipole transformations

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

$$[T]=[z_{\text{inv}}][y_{\text{inv}}][M2L_z][y][z]$$

$$M'^{m}_{n}=M^{m}_{n}\,\mathrm{e}^{\mathrm{j}m\beta}$$

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

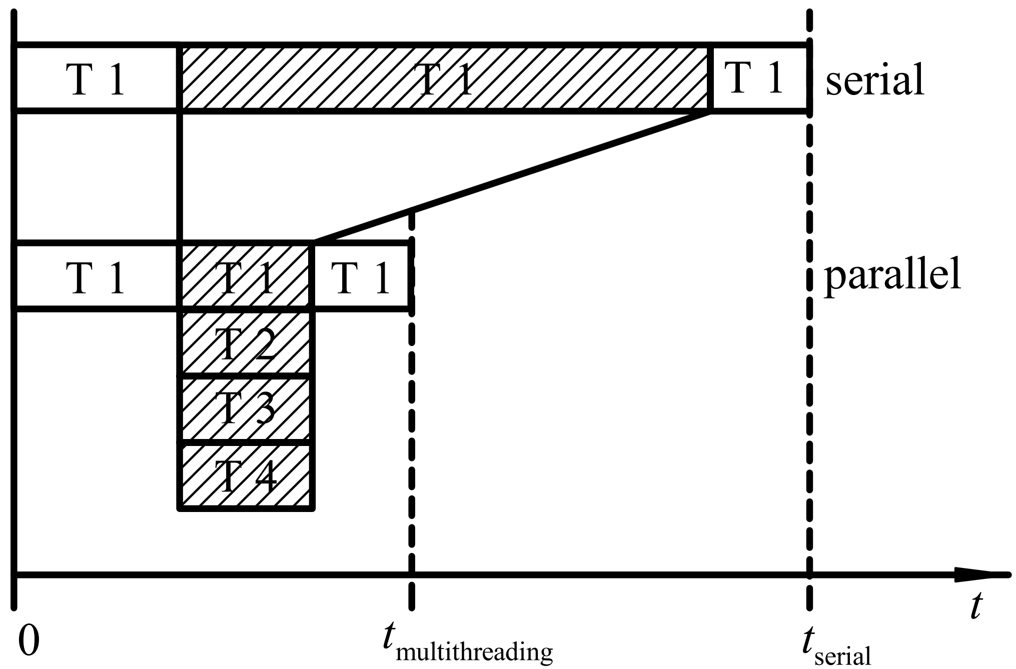
$$L^{m}_{n}=\sum_{k=m}^{L}M^{m}_{k}\frac{Y^{0}_{k+n}(0,0)(-1)^{k+m}(n+k)!}{\rho^{k+n+1}\sqrt{(k-m)!(k+m)!(n-m)!(n+m)!}}$$

Summary

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

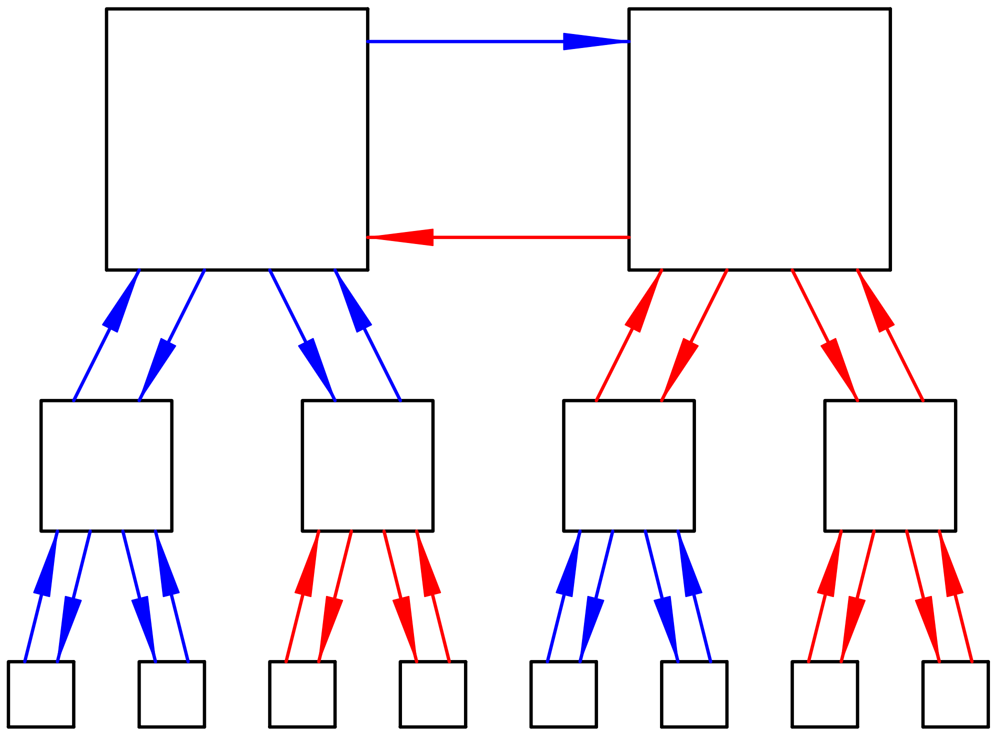
Properties

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



Multipole transformations

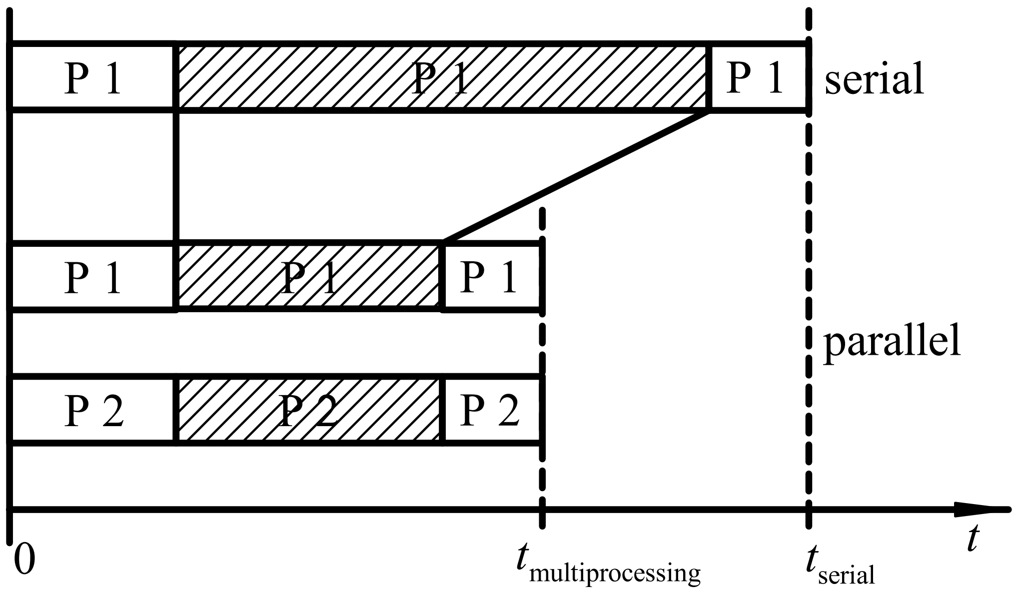
- Operations of the octree cubes can be computed independent of each other



Multiprocessing

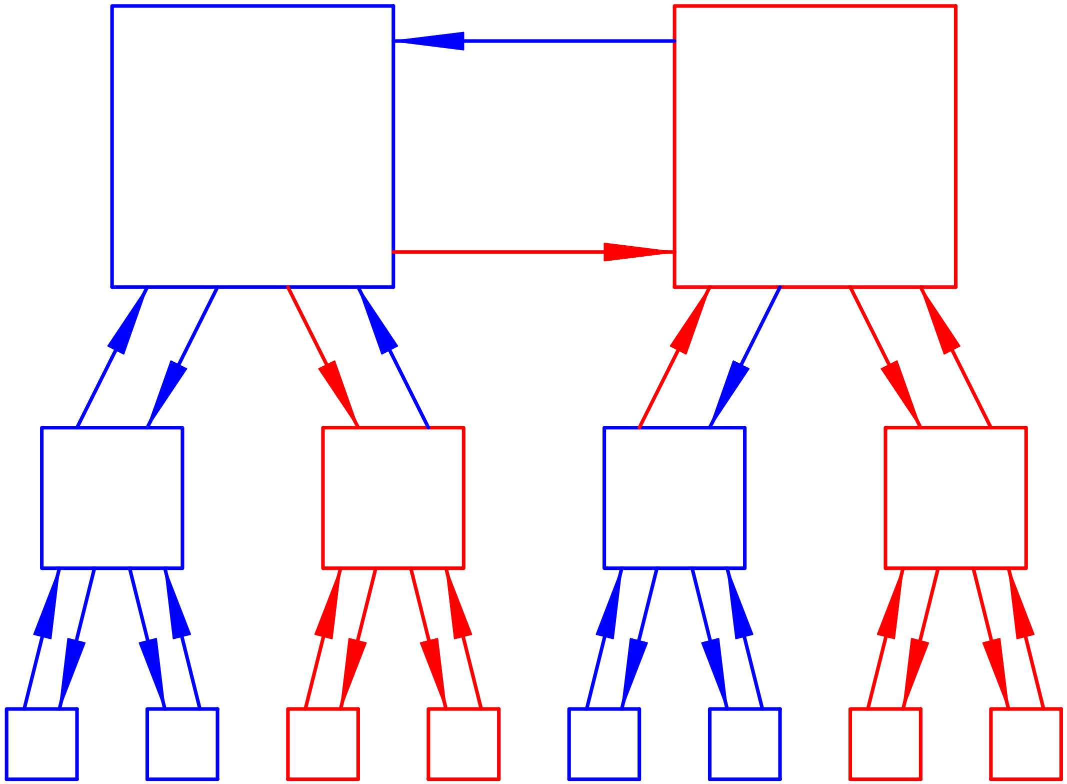
Properties

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

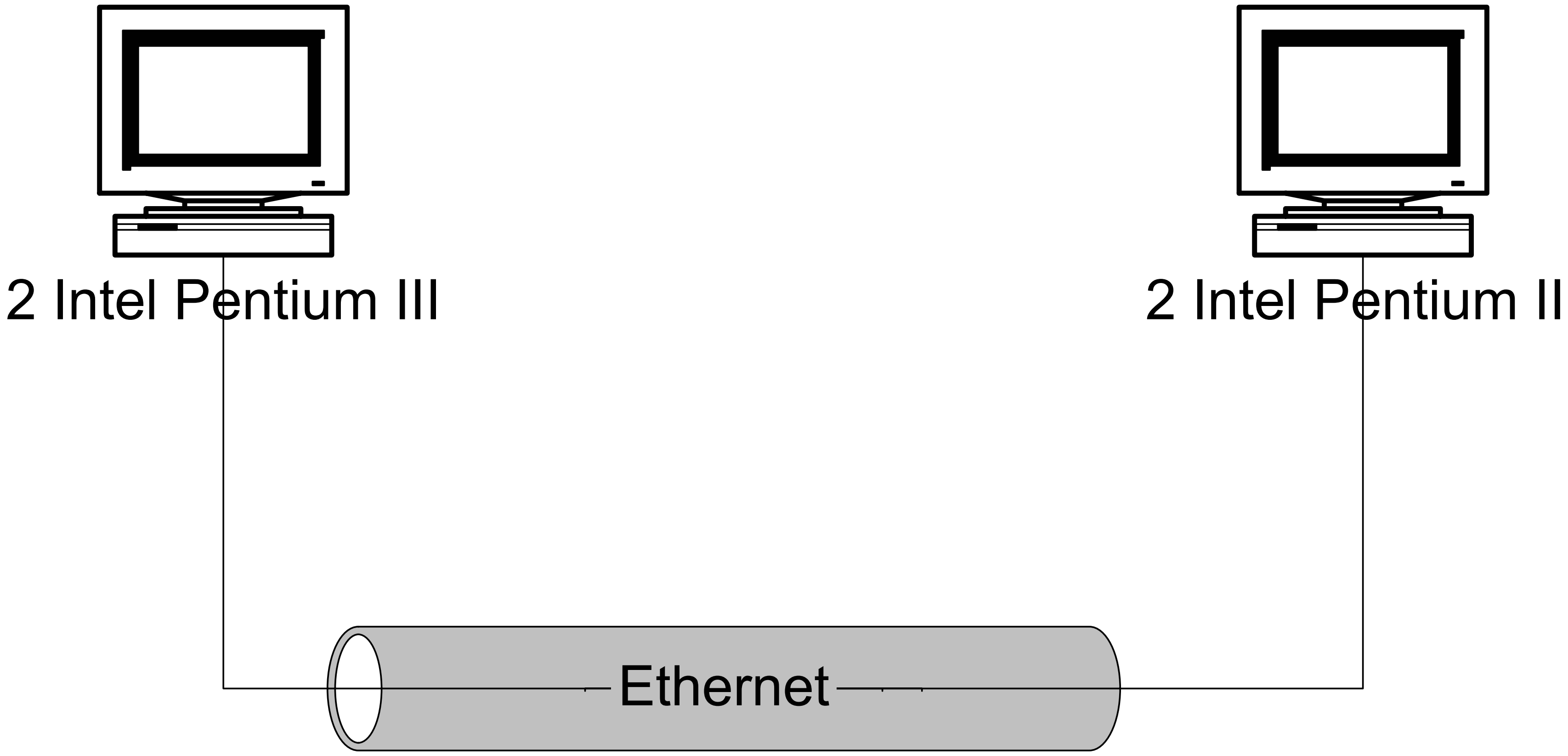


Multipole transformations

- Data transfer between processes is necessary

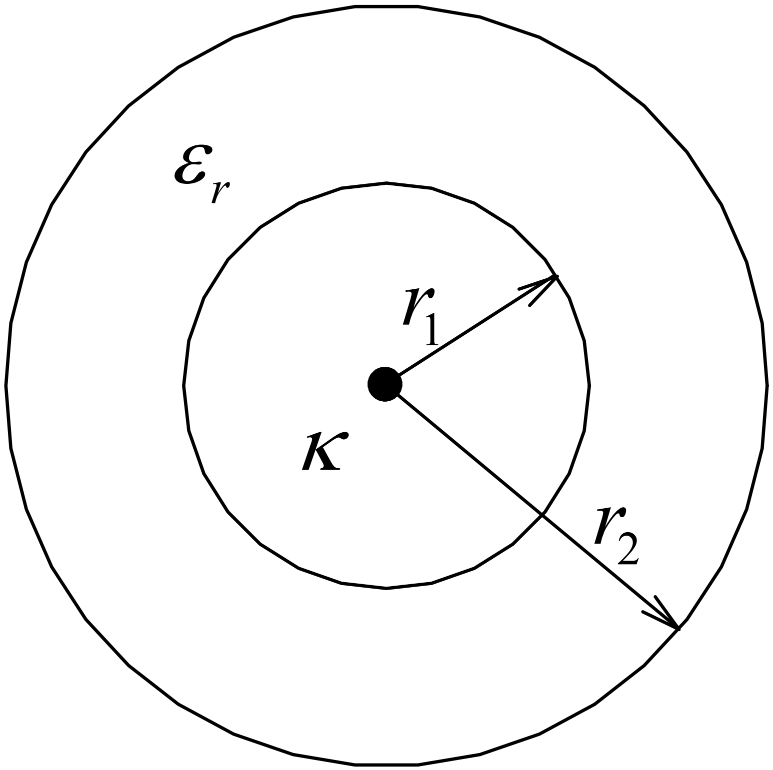


Hardware



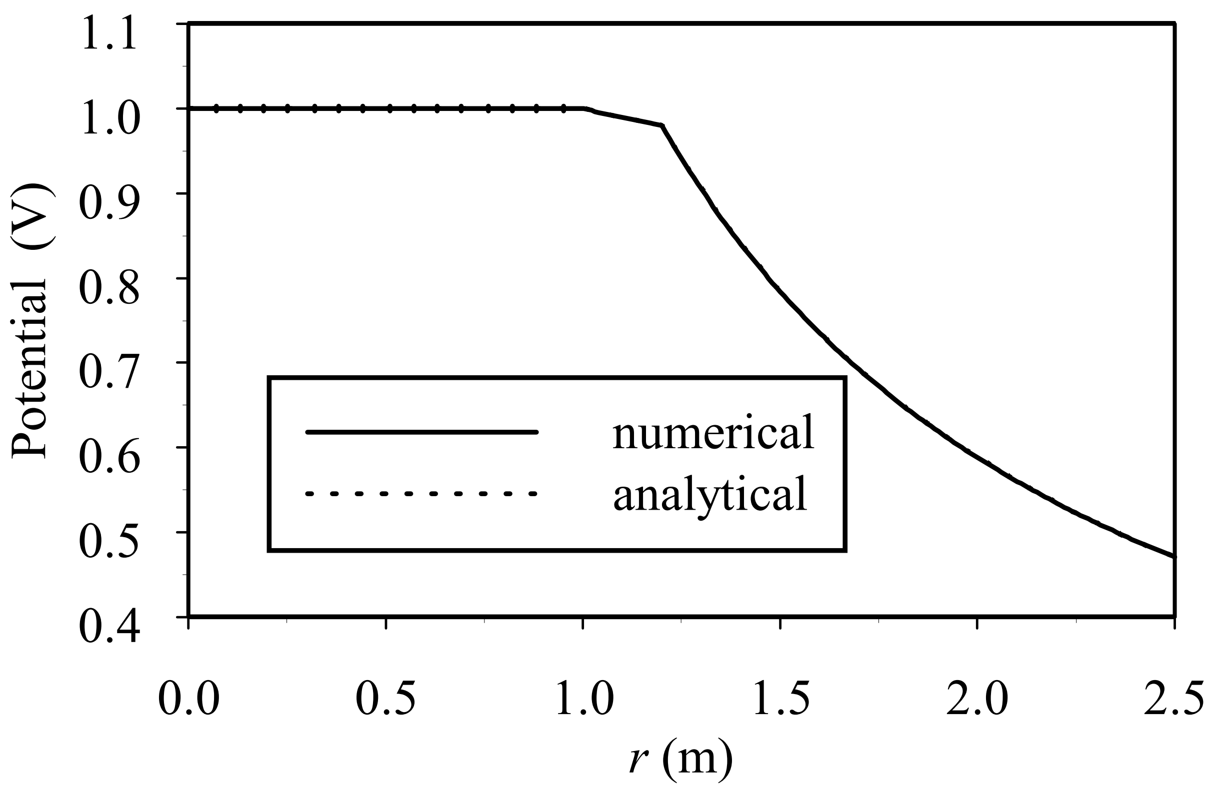
Coated sphere

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



Coated sphere

- Potential at a radial line

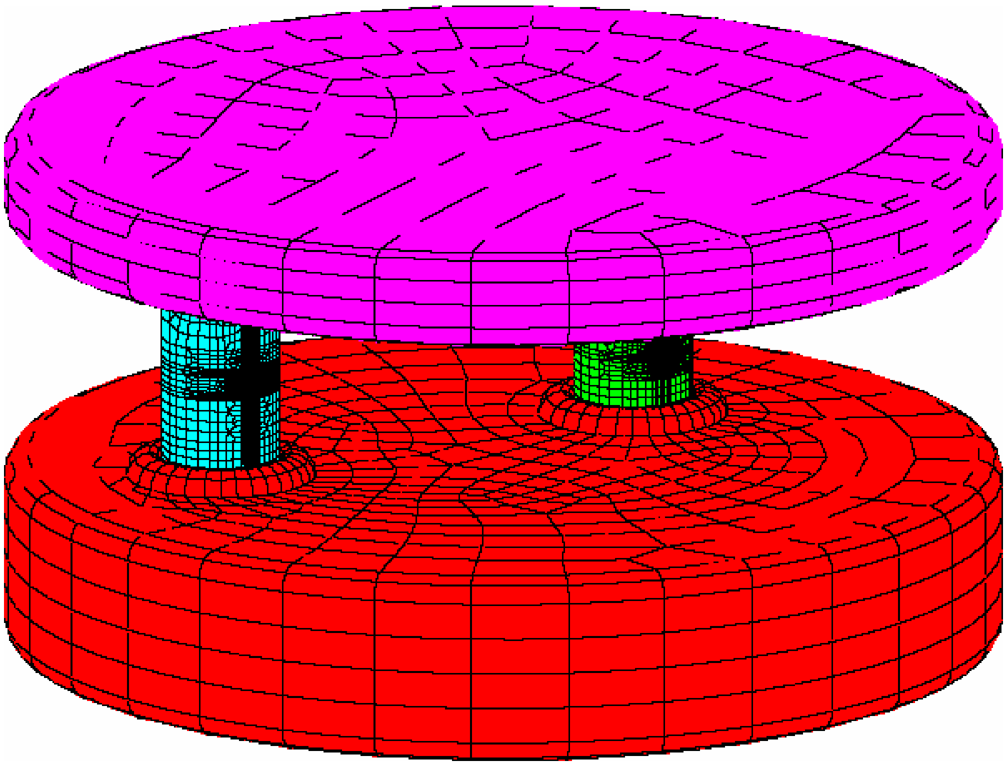


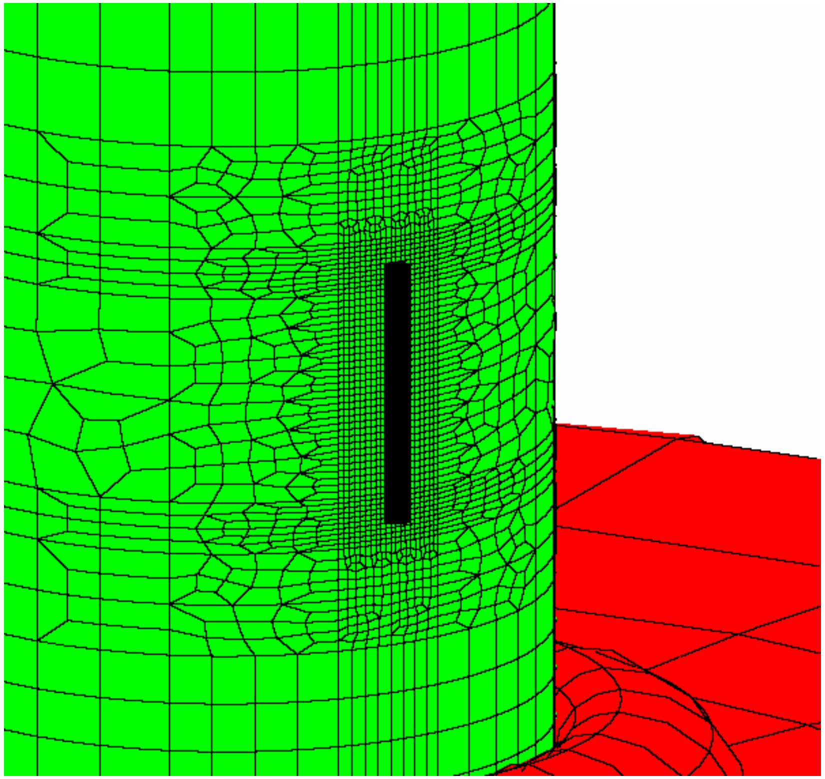
Coated sphere

- Computing times

	Serial	OpenMP	OpenMP + MPI
Processes	1	1	2
Threads	1	2	2*2
Matrix assembly	2931 s	1466	
Reduction		50 %	
Solution linear equation system	3851 s	2387 s	1268 s
Reduction		38 %	67 % (38 % OpenMP, 47 % MPI)

Gas insulated high voltage system

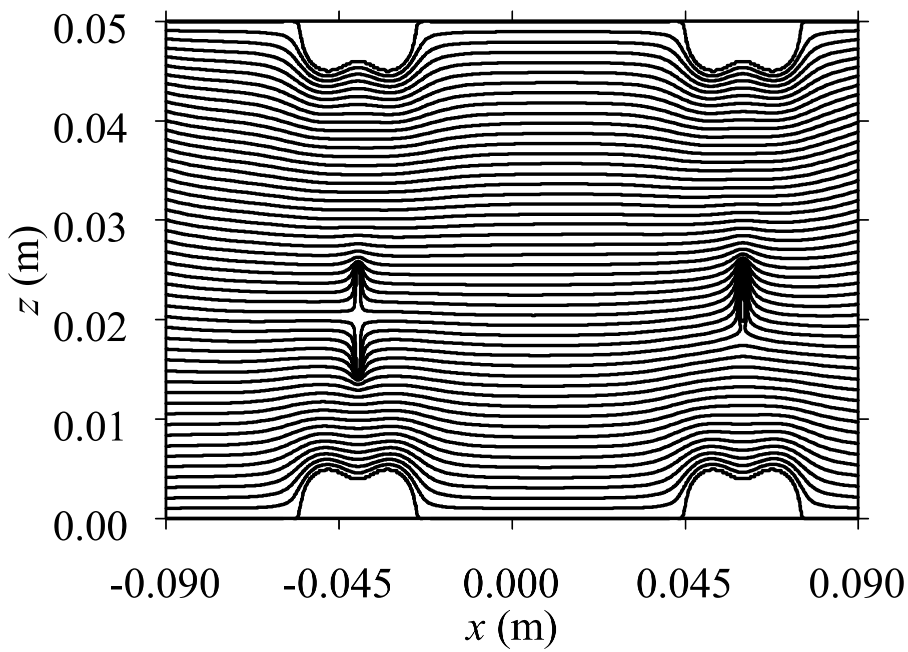


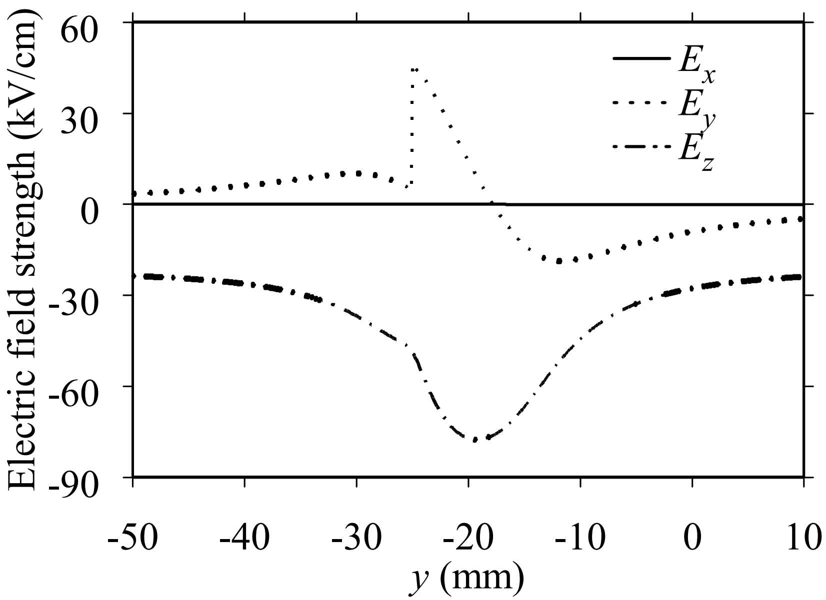


Gas insulated high voltage system

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

Gas insulated high voltage system





Gas insulated high voltage system

- Computing times

	Serial	OpenMP	OpenMP + MPI	
Processes	1	1	2	
Threads	1	2	2*2	
Solution linear equation system	6730 s	4194 s	3360 s	2548 s
Reduction		38 %	50 % (19 % MPI)	62 % (39 % MPI)
Octree level			4	7

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