Shared memory parallelism and low-rank approximation techniques applied to direct solvers in FEM simulation

P. Amestoy[1], A. Buttari[2], G. Joslin[3], J.-Y. L'Excellent[4,5], M. Sid-Lakhdar[5], C. Weisbecker[1], M. Forzan[6], C. Pozza[6], R. Perrin[7], and V. Pellissier[7]

[1]INPT-IRIT, Toulouse, France, [2]CNRS-IRIT, Toulouse, France, [3]CERFACS, Toulouse, France
[4]INRIA, Montbonnot, France, [5]ENS Lyon, France
[6]Dep.t of Industrial Engineering, University of Padova, 35131 Italy
[7]CEDRAT, Meylan, France

Abstract— **In this paper, the performance of a parallel sparse direct solver on a shared memory multicore system is presented. Large size test matrices arising from finite element simulation of induction heating industrial applications are used in order to evaluate the performance improvements due to low-rank representations and multicore parallelization.**

Index Terms— **Eddy currents, finite element methods, sparse matrices, parallel algorithms, approximation algorithms.**

I. Introduction

In 3D finite element simulation of induction heating processes, the solution time is a limiting factor in the design and optimization of new devices. Time-harmonic electromagnetic problems coupled with thermal problems are solved in sequence, and the linear system solution in the electromagnetic problem is often the bottleneck. Direct solvers are preferred to iterative ones when convergence and stability issues. Therefore, an efficient direct method to solve large sparse complex matrices should exploit parallelization and reduce memory consumption. In this paper, a reduction of memory requirements through low-rank techniques and factorization times through shared memory parallelism in MUMPS (MUltifrontal Massively Parallel sparse direct Solver) will be discussed [2]. Matrices arise from the modelization of induction heating industrial devices. Heating of a susceptor by pancake coils and gear induction hardening are taken as test benchmarks. Geometric model design, meshing and matrix building are performed by a commercial software [3]. Starting from the same geometry (Pancake or Gear), meshes are gradually refined in order to solve problems of different sizes leading to matrices ranging from 320k to 1.5M degrees of freedom.

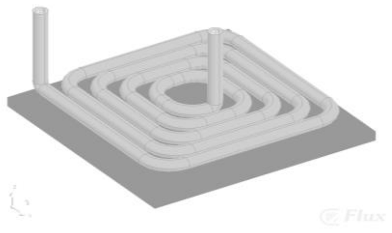

Fig. 1. Geometry of the pancake coil

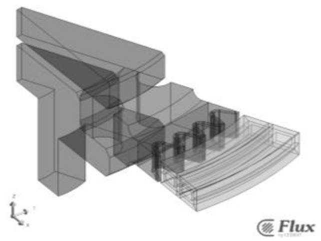

Fig. 2. The figure represents only a slice of the whole inductor and workpiece geometry because of symmetries

TABLE I
TESTED MATRICES

Test matrix	Pancake 1	Pancake 2	Pancake 3	Pancake 4	Gear 1
Degrees of freedom	320k	630k	1M	1.5M	370k
Number of entries in LU factors	990M	2.1G	5.2G	8.7G	700M

II. Improvements by Low-Rank Approximation Techniques

A low-rank matrix can be represented in a form which decreases its memory requirements and the complexity of basic linear algebra operations it is involved in, such as matrix-matrix products. This is formalized by Definition 3.1 in [6].

Let A be a matrix of size $m \times n$. Let k_ε be the approximated numerical rank of A at accuracy ε. A is said to be a low-rank matrix if three matrices W of size $m \times k_\varepsilon$, Z of size $n \times k_\varepsilon$ and E of size $m \times n$ exist such that :

$$A = W \cdot Z^T + E . \tag{1}$$

where $\|E\|^2 \le \varepsilon$ and $k_\varepsilon (m + n) < mn$.
k_ε is commonly referred to as the numerical rank at precision ε and can be computed, together with W and Z with a Rank-Revealing QR (RRQR) factorization. Low-rank approximation techniques are based upon the idea to ignore E and simply represent A as the product of W and Z^T, at accuracy ε.

In practice, matrices coming from applicative problems are not low-rank, which means that they cannot be directly approximated. However, low-rank approximations can be performed on sub-blocks defined by an appropriately chosen partitioning of matrix indices [5]. Theoretical studies based on mathematical properties of the underlying operators have shown that variable sets that are far away in the domain tend to exhibit weak interactions, and the corresponding matrix block has a low rank. To benefit from this property, a new low-rank format called Block Low-Rank (BLR) has been developed and exploited within internal data structures of the multifrontal method used in the MUMPS software, in order to decrease the memory consumption and the operation count of the solver.

Preliminary results show that the described method is more efficient on large problems, which is a good property in the context of large scale computing. As reported in Table 2, in two test cases the memory footprint is reduced by a factor almost of three and the complexity is almost halved. A few steps of iterative refinement are performed to recover full precision from the original approximated precision.

TABLE II
LOW-RANK IMPROVEMENTS

Test matrix	Pancake 2	Pancake 4	
LR threshold ε	10^{-8}	10^{-10}	10^{-14}
Memory saved for factors storage	35%	34%	14%
Operations saved for factorization	60%	60%	29%
Scaled residual	2.3×10^{-16}	1×10^{-12}	3.5×10^{-16}

III. IMPROVEMENTS OF SHARED MEMORY PARALLELISM IN MUMPS

In order to solve a linear system, the multifrontal method transforms the initial sparse matrix into an elimination tree of much smaller dense matrices. This tree is traversed in a topological order and each node is computed following a partial LU decomposition. Then, a two-pass solve operation is applied on each node of the tree in order to find the solution of the original system.

The structure of the tree offers an inner parallelism, called tree parallelism, in which different processes work on data subsets, on different nodes of a network. This kind of parallelism is already exploited by MUMPS in distributed memory environments [5]. However, in shared memory environments, only node parallelism is applied: many processes collaborate on the decomposition, working on the same node. In this approach, threaded BLAS libraries are preferred in order to parallelize dense matrix operations. In order to go further, the fork-join model of parallelism has been implemented in code based on OpenMP directives. Our first goal was then to exploit tree parallelism in shared memory environments, by adopting algorithms commonly used in distributed memory environments and by rearranging them to multithreading. Therefore, the so called AlgL0 algorithm consists in finding a separating layer in the tree, called L0, such that node parallelism will still be applied above it, but tree parallelism will be applied under it through OpenMP. The use of adequate memory mapping policies allows to improve the performance of MUMPS on SMP (Symmetric Multi Processor) and NUMA (Non-Uniform Memory Access) architectures. The localalloc policy, which consists in mapping the memory pages on the local memory of the processor that first touches them, is applied on data structures used under L0, in order to achieve a better data locality and cache exploitation. The interleave policy, which consists in allocating the memory pages on all memory banks in a round-robin fashion such that the allocated memory is spread over all the physical memory, has been used on data structures above L0 in order to improve the bandwidth.

Both MUMPS 4.10.0 and MUMPS 4.10.0 with new AlgL0 algorithm have been tested on the set of test matrices, with the stated memory allocation techniques. As reported in Table 3, this approach brings a remarkable reduction of computational time on all matrices even if it tends to decrease on large matrices, because the portion of work in the top of the tree (above L0) increases in comparison to the workload in the bottom of the tree. Furthermore, this gain is expected to rise through the use of the interleave policy.

TABLE III
TIME SAVE BY USING MUMPS 4.10.0 WITH ALGL0 ALGORITHM

Test matrix	Pancake 1	Pancake 2	Pancake 3	Gear 1
Time saving	13%	11%	8%	21%

IV. CONCLUSIONS

Significant improvements of parallel sparse direct solver MUMPS have been successfully tested in 3D numerical simulation of industrial applications of induction heating. Preliminary results show a remarkable reduction of both memory usage and number of operations to be performed. Furthermore, both tree and node parallelism are exploited in order to reduce the solution time.

REFERENCES

[1] F. Dughiero, M. Forzan, C. Pozza and E. Sieni, "A translational coupled electromagnetic and thermal innovative model for induction welding of tubes", IEEE Trans. On Magn., vol. 48, no. 2, 2012.
[2] MUMPS, http://graal.ens-lyon.fr/MUMPS/, MUMPS users guide.
[3] CEDRAT, http://www.cedrat.com, Flux users guide.
[4] S. Börm, "Efficient numerical methods for non-local operators", European Mathematical Society, 2010.
[5] P. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L'Excellent and C. Weisbecker, "Improving multifrontal methods by means of Low-Rank Approximations techniques", SIAM 2012 Conference on Applied Linear Algebra, LA12, Valencia, Spain, 2012.
[6] M. Bebendorf, Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems (Lecture Notes in Computational Science and Engineering), Springer, 1 ed., 2008.