PARALLEL COMPUTATIONS OF 3-D ELECTRO-MAGNETIC FIELDS ON TRANSPUTERS

Ch. Hafner, St.Kiener
Swiss Federal Institute of Technology
CH-8092 Zurich, Switzerland

Abstract

The MMP (Multiple MultiPoles) programs are based on the Generalized Multipole Technique (GMT) which has been shown to be very efficient especially for computations of electrodynamic fields [1]. Since this method is relatively simple there was no problem to implement it on PC's. The power of PC's may be increased very much by INMOS T800 transputers which allow parallel processing. In this paper the parallel versions of the 3-D MMP programs are presented together with a comparison of the speed of different configurations.

Basics of the MMP programs

The MMP programs are based on the Generalized Multipole Technique (GMT) which 1.) expands the fields themselves to avoid Coulomb and similar integrals and 2.) applies a generalized point-matching technique (PMT) to avoid those integrals which occur in the scalar products of the projection technique (PT) used by the method of moments (MoM). Instead of this, the generalized PMT simply uses overdetermined systems of equations which are solved in the least- squares sense.

The expansion functions used in the GMT are exact solutions of the field equations in a domain with linear, homogeneous and isotropic materials. As a consequence only the boundaries of the domains have to be discredited even in the case of lossy materials. This simplifies the input very much, which is important especially for PC's. The most important expansion functions of the GMT are called multipoles. They show a 'local behavior'. This easily produces well conditioned matrices which may be solved by a simple updating routine using Given's plane rotations.

Although multipole functions are more complicated than the ones preferred by other methods (linear functions, spline functions, harmonic functions etc.) the computation time of the functions is much smaller than the computation time for the solution of the system of equations and may be neglected therefore (except for small problems).

Since the GMT avoids time consuming numerical integrations almost the whole computation time is used for the solution of the system of equations. One might believe that this leads to inferior results. In fact, if the *usual* PMT is applied to the MoM this may be true. But it has been shown [1] that the *extended* PMT with an appropriate weighting of the equations is equivalent to a PT with the optimal choice (Galerkin) of testing functions (and a numerical computation of the scalar products). The same results may be derived by an error method with a convenient definition of the error. In fact these equivalences have just been used to solve the problem of the correct weighting.

Parallel Computations on PC's

The rapid growth of modern computers – especially in the PC market – makes it very important that numerical codes can be easily adapted to new machines. Since the MMP code is quite small and the method quite simple, the programs could be adapted to the new T800 transputers (which allow parallel computations on PC's) within a few weeks.

For our applications a T800 transputer has about the speed of a 80386 based 20MHz PC with 80387 coprocessor and Weitek mW1167 accelerator or 12 times the speed of an AT03 with 80287 coprocessor or 1/3 of the speed of a Cyber 855 mainframe.

It has been mentioned that almost the whole computation time is needed for the solution of the overdetermined system of equations by an updating routine. As a consequence only the updating procedure needs to be analyzed and paralleling. Only an upper triangular matrix U needs to stored in a vector A. U is updated starting with the first column, from top to bottom, as indicated in figure 1. Finally U is solved by back substitution.

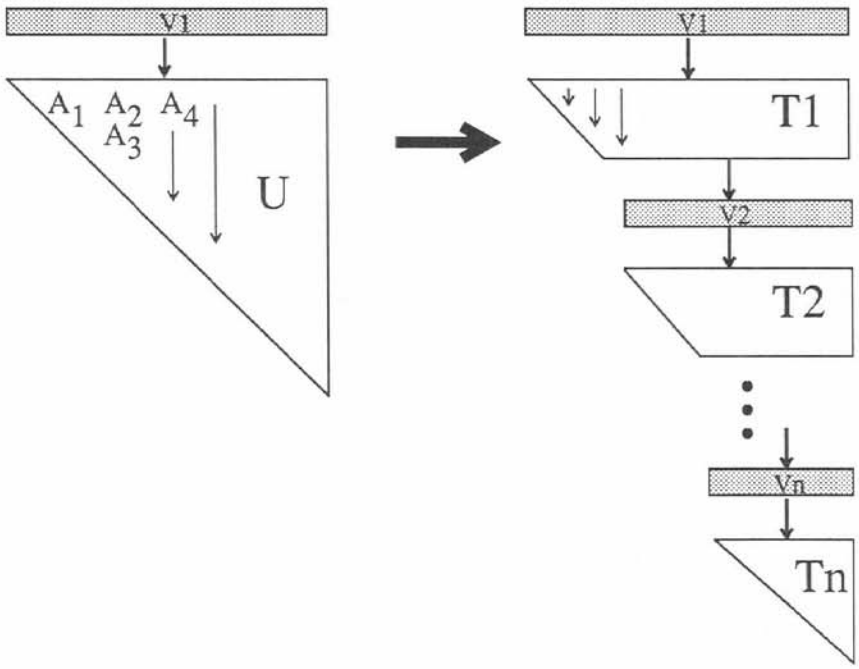

Figure 1 Splitting of the triangular matrix U

To parallelize this procedure for N transputers, U is simply splitter in N trapezoidal parts T1,T2,...TN (see figure 1) with a similar number of elements stored in the vectors A1,A2,...AN. When a new row (a new equation) stored in a vector V1 has to be updated, the first transputer (which stores T1) takes it, updates T1 and gives a (shorter) vector V2 to the second transputer and so on. Since the computation time is approximately proportional to the number of elements, the computation time for each of the transputers is almost the same.

Of course it is impossible with N transputers to get N times the speed of a single one. On the first (root) transputer not only the matrix T1 is computed but also the whole

input/output, the calculation of the elements of the vectors V1 and so on has to be done. For this reason less memory for the storage of T1 is available than for T2, T3,...TN. At the beginning only the first transputer is working. Later on the other transputers start working (pipeline architecture). Even if all transputers are busy, one updating step will never need exactly the same time for every transputer. For these reasons the computation time depends not only on the number of transputers but also on the problem to be solved.

The splitting of the matrix U has another important advantage: It allows to solve problems with an almost unlimited number of unknowns simply by adding enough transputers (Cards with up to 20 T800 transputers with 1MBytes memory each are available for AT compatible PC's.). For the present paper a board with four T800 transputers with 4MByte memory each has been used in a portable Toshiba T3200 PC. Our tests showed that for big problems such a portable machine needs less execution time and - since it is a single user machine - much less turn-around time than a Cyber 855.

Examples

To compare the speed some simple examples with different sizes have been selected. The examples have been solved with 1, 2, and 4 transputers. To get an impression of the efficiency of the parallelization one should use somehow similar problems. We have chosen the scattering of a plane wave on 1 or several metallic cylinders of finite length (half wavelength) and thickness (radius equal to 1/7 of the length). The aim of these computations is the comparison of the speed of different machines and configurations rather than to show the speed and practical value of the programs. Figure 2 shows the magnetic field for the case of five cylinders.

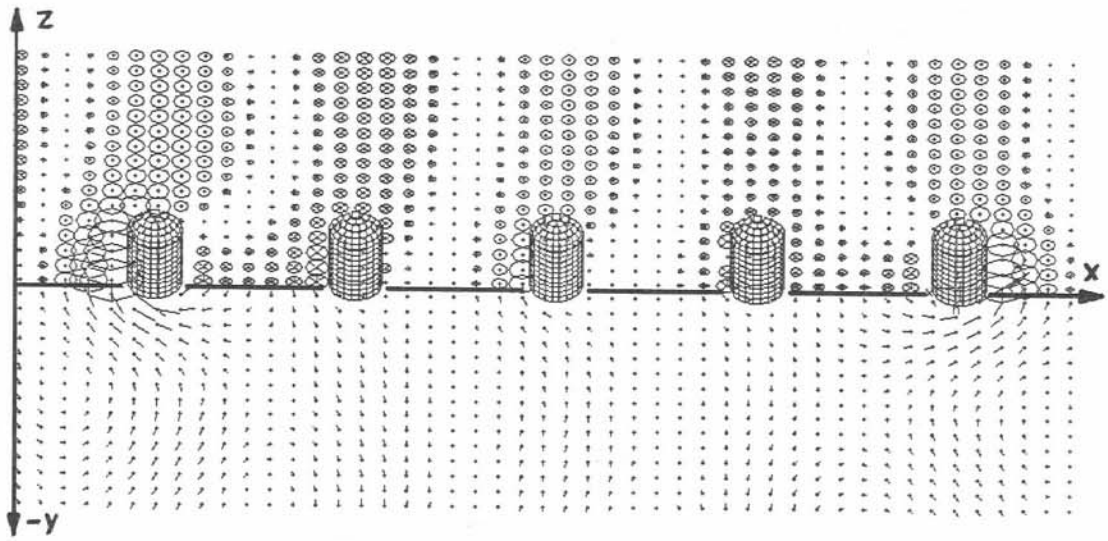

Figure 2 Magnetic field in the xz- and xy-plane for a plane wave incident from the right hand side on five thick cylinders.

In order to compare the computation time of different problems, the size S of a problem is defined as the number of equations M multiplied by the square of the number of unknowns

N (divided by 10.0E6 to get reasonable numbers). It is expected that the computation time of large problems is approximately proportional to S. For this reason the computation time relative to S is shown in figure 3. Two points should be mentioned: 1.) The relative computation times (computation time divided by the size S of the problem) should decrease for increasing size of the problem. This is true for any number of transputers. But on workstations this is the case only for problems small enough to fit in the central memory. 2.) The turn- around time for large problems on workstations (and mainframes) may be much higher than the computation time even if only one single job is running. These effects are due to the paging.

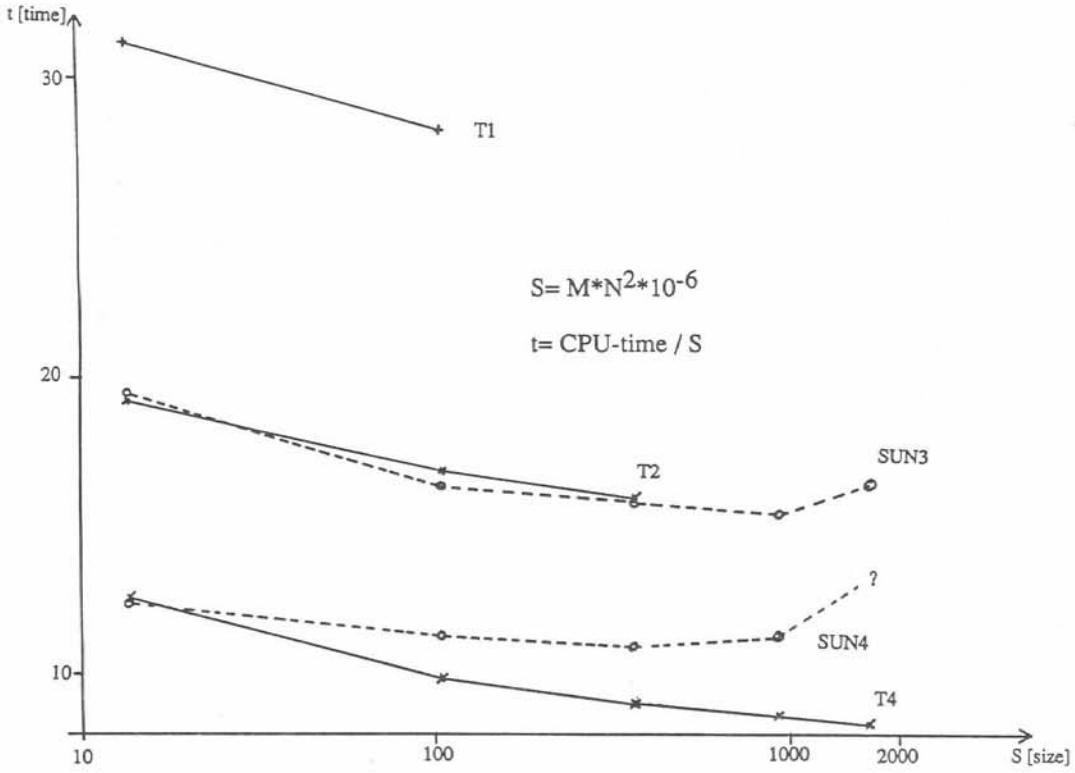

Figure 3 Relative computation times for examples of different size on 1, 2, 4 transputers and SUN 3/260, SUN 4/110 workstations with floating point accelerators.

Reference

[1] Ch.Hafner: *Numerische Berechnung elektromagnetischer Felder*, Springer, Berlin, 1987.