A Distributed Parallel Approach using the Conjugate Gradient Method and the N-Scheme on FEM cases

J. Eyng[1], J.P.A. Bastos[1], M.A.R. Dantas[2], M. Júnior[3], M. Fischborn[4]
[1]GRUCAD/EEL/CTC, Universidade Federal de Santa Catarina, CP. 476, Florianópolis, SC
[2]LaPeSD/INE/CTC, Universidade Federal de Santa Catarina, CP. 476, Florianópolis, SC
[3]CFM, Universidade Federal de Santa Catarina, CP. 476, Florianópolis, SC
[4]UFTPR, Universidade Tecnológica Federal do Paraná, CP. 271, Florianópolis, SC
julianaeyng@gmail.com, jpab@grucad.ufsc.br, mario.dantas@ufsc.br, maginot.junior@gmail.com, fisch@utfpr.edu.br

***Abstract*— A technique for solving static problems with the finite element method (FEM) was developed and is called N-Scheme. The method solves Finite Element (FE) problems without assembling the matrix system *Ax=b*. It calculates the node potential unknowns in a much simpler way than traditional techniques. The assembling and solution of the matrix system are considered in a single procedure and operations are similar to the Gauss-Seidel method with over-relaxation (or Successive Over Relaxation – SOR), which provides good convergence. However, the computational time of N-Scheme method is larger when compared with the classical FE implementation using the ICCG (Incomplete Choleski Conjugate Gradient) method. One possible way to improve the method computational time is to apply parallel programming techniques. Studies have shown that the Conjugate Gradient method applied in conjunction with the N-Scheme reduces the computational time significantly. This work aims to show that the new N-Scheme method associated with parallel programming techniques offers a still better computational timing to solve large problems involving large finite element 3D meshes. Some examples show that the proposed technique with parallel programming allows fast calculation time.**

***Index Terms*— Finite Elements, Parallel Programming, Numerical Solver.**

I. Introduction

The technique called N-Scheme method was considered in [1]. It calculates the unknown potential nodes of the mesh in a much simpler way than the traditional techniques used for FEM cases. The FEM applied to static cases requires the calculation of the rigidity matrix and source vectors. The evaluation of such matrices is normally performed element by element and assembled in the global matrix system. Boundary conditions are inserted and the system is solved by a direct method (as Gauss Elimination) [2] [3] or by an iterative method as ICCG (Incomplete Choleski Conjugate Gradient) [1] [3].

The N-Scheme method is presented as new alternative for solving linear systems generated by FEM. It proposes a different approach to solve the system $Ax = b$. The method, associated with parallel programming tools for fast computation, will be compared to classical sequential solving. In order to depict good performances with parallel techniques, when compared to sequential calculations, the FE meshes must be large, operating with millions of elements. The parallel development is processed using MPI (Message Passing Interface) library [4] [5]. The environment used for testing is a cluster, available in *Sharcnet* (Shared Hierarchical Academic Research Computing Network) [6], with a high-speed local area networking system. Timing is measured in seconds and results are available in [7] [8]. For didactical goals, we replicate here some parts of [8], where the N-Scheme with parallel programming was introduced. We will call this technique N-Scheme/SOR.

After some investigation, we noticed that application of CG method associated to the Jacobi pre-conditioner is possible since it does not affect the principles of the N-Scheme method [9]. As result, the processing time obtained for sequential calculations is significantly better than timing obtained with the N-Scheme/SOR method. Therefore, applying parallel programming techniques in the modified method is the main contribution of this work. It provides accurate results and the processing time for large 3D problems is significantly reduced. For better understanding, we recall some concepts presented in [10] for a regular mesh as well as some parallel programming principles applied on multicore clusters.

This article is organized as follows: the characteristics of the new method N-Scheme of solving the matrix system $Ax = b$ are discussed in Section 2. Section 3 shows the N-Scheme in connection with the Conjugate Gradient method and its features. 3. Section 4 considers the experimental environment and the results obtained by the N-Scheme/CG techniques. Finally, a conclusion is presented.

II. The N-Scheme/SOR and N-Scheme/CG Method

The N-Scheme/SOR works by nodes (instead the classical assembling by elements) using cells of elements around nodes. The method is solved as follows:

- For each *n* unknown node:
  - Calculate the elemental matrices of all the elements of the cell around *n*, making the appropriate sums and operating as shown in [1];
  - Thus, a first approximation of the potential for this unknown is obtained;
- Repeat the process until the convergence is obtained.

For the implementation, three loops are necessary:

- the external loop for the iterations;
- the second loop for the unknown nodes;
- the internal loop is related to cell elements of the unknown node.

The above presented method has three loops. When adapting the method for applying the CG method (N-Scheme/CG), only the two following loops are necessary [10]:

- the external loop for the iterations
- the internal loop for the elements

The N-Scheme is applied on the two more time demanding operations of the algorithm. The first one is the computation of the residue vector. A second one requiring attention can also be computed similarly without assembling *A* [6]. For the preconditioner, we utilize the Jacobi preconditioner, which is a vector containing the diagonal elements of the matrix *A*.

III. Results

In this section we present the results obtained with the method N-Scheme/GC using multicore parallel programming concepts. In these executions, we evaluate the performance achieved by its implementation on sequential and parallel.

The method can not be fully parallelized, but the calculation of the element nodes of the 3D mesh occur in parallel. The results are shown in graphs (comparative runtime curves and number of iterations).

The program is developed using C Language and the hardware architecture used has x86 compatible with the following specs: Intel Core i7 3770k processor with 8GB of memory and the operating system we used the GNU/Linux 3.2.0 32bit.

The parallel code and also the sequential calculations are performed in two ways: without the flag optimization in the parallel code compilation Fig. 1 and with the optimization flag Fig. 2 [11]. The optimization flag allows the compiler to perform various in a fast way, correcting parts of the code in order to make it more efficient while being compiled. The Fig. 3 shows the speedup behavior for the performed calculations.

IV. Conclusions

The N-Scheme has a processing time larger when compared with a classical FEM implementation using, for instance, an ICCG solver. In order to enhance it, we are investigating other possible solver techniques while keeping the N-Scheme main principles. For the graphs presented, the mesh will be referenced by the variable *Nx* (equal to *Ny* and *Nz*): Nº of elements = *Nx.Ny.Nz* and Nº of nodes = (*Nx+1*).(*Ny+1*).(*Nz+1*).

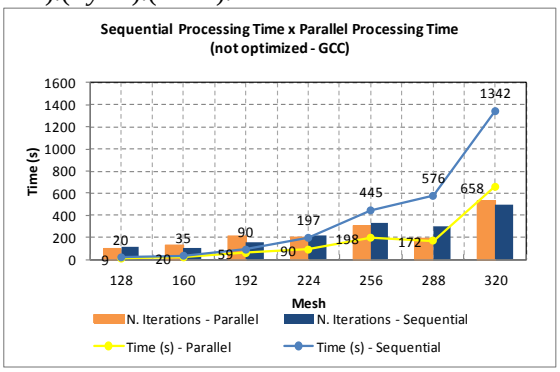

Fig. 1 – Sequential x Parallel processing time and iterations, without the flag optimization.

Finally, we point out that the main goal of this work is the application of parallelization techniques on the N-Scheme/CG and the obtained results are encouraging.

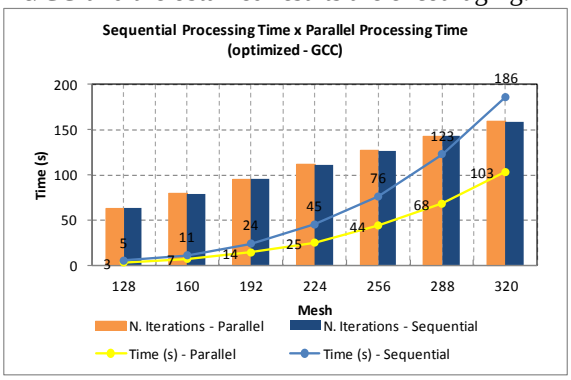

Fig. 2 – Sequential x Parallel processing time and iterations, with the flag optimization.

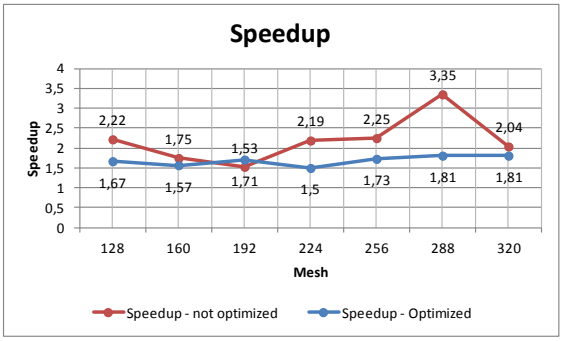

Fig. 3 – Values of the speedup.

References

[1] J.P.A. Bastos, "Is it possible to solve a FEM static case without assembling, storing and solving an Ax = b matrix system?" *ICS Newsletter*, vol 16, number 1, March 2009, UK.

[2] Ida, N., Bastos, J.P.A, *Electromagnetic and Calculation of Fields*, edited by Springer-Verlag, 1997.

[3] Stoer, J., Burlisch, R., *Indrotuction to numerical analysis*, Springer-Verlag, 1980.

[4] J. Dongarra et al, "*MPI: the complete reference*". Massachusetts Institute of Tecnology, (http://www.netlib.org/utk/papers/mpi-book/mpi-book.html)

[5] H. Jordan, G. Alaghband, *Fundamentals of Parallel Processing*, Prentice-Hall (Upper Saddle River, N.J), 2003.

[6] SHARCNET homepage, Shared Hierarchical Academic Research Computing Network. Available in: < http:/www.SHARCNet.ca>.

[7] Eyng, J.; Bastos, J.P.A.; Sadowski, N.; Fischborn, M.; Dantas, M.A.R.; Ferreira, D.J., "Parallel programming apllied to the N-Scheme for solving FE cases without assembling an Ax = b system." *The 14th Biennial IEEE Conference on Electromagnetic Field Computation – CEFC 2010*, vol. 1, p. 1, May de 2010, Chicago, Illinois, USA.

[8] Eyng, J., Bastos, J.P.A., Sadowski, N., Fischborn, M., Dantas, M.A.R., Applying Parallel Programming to the N- Scheme for solving FEM cases without assembling an Ax=b system, 18th International Conference on the Computation of Electromagnetic Fields – COMPUMAG 2011, vol. 1, p. 1 a 2, 12 a 15 de Julho de 2011, Sydney/Austrália.

[9] R. Barret, M. Berry, T.F. Chan, J. Demmel, J.M. Donatto, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. Van der Vorts, "*Templates for the solution of linear systems: building blocks for iterative procedures*", Society for Industrial and Applied Mathematics, (http://www.siam.org/books).

[10] J. P. A. Bastos and N. Sadowski, "A new method to solve 3-D magnetodynamic problems without assembling an *Ax=b* system", IEEE Trans. on Magn., vol. 46, no. 8, pp. 3365-3368, August 2010.

[11] Brian J. Gough, foreword by Richard M. Stallman. *An Introduction to GCC - for the GNU compilers gcc and g++*. Revised August 2005.