Speed-up of Magnetic-Electric Matrices Assembly Computation by Means of a Multi GPUs Environment

A. Chiariello[1], S. Mastrostefano[2], M. Nicolazzo[3], G. Rubinacci[3], A. Tamburrino[2], S. Ventre[2], F. Villone[2]

[1]Consorzio CREATE, DIII, Seconda Università di Napoli, Via Roma 29, Aversa, Italy
[2]Consorzio CREATE, DIEI, Università di Cassino e del Lazio Meridionale, Cassino (FR), Italy
[3]Consorzio CREATE, DIETI, Università Federico II di Napoli, Via Claudio 21, 80125, Napoli,Italy

This paper deals with an accelerated implementation of the assembly of the matrices accounting for the magnetic-electric interactions arising from an integral formulation of the magneto-quasi-static problem. The use of integral formulations leads to fully populated matrices whose computational effort is O(N^2), N being the number of degrees of freedoms (DoFs) related to the finite element mesh. Although the inversion procedure is O(N^3) for a direct solver, for "medium size" problems the assembly can be very time consuming, especially when a great accuracy is required and/or a great deal of geometrical details should be investigated. In this work we will prove that a significant speed-up can be achieved by means of an "ad-hoc" use of GPUs, although its implementation is really challenging w.r.t. traditional parallel computational systems. Two kinds of applications are shown: one in the framework of Non-Destructive Testing (NDT), the other in the field of plasma fusion devices modelling.

***Index Terms*— GPUs, HPC, integral formulation, NDT, fusion plasma devices**

I. Introduction

MAGNETIC materials are widely used in many kinds of applications for NDT (Non-Destructive Testing) and Plasma Fusion Devices.

One leading application in NDT concerns the electromagnetic (nondestructive) characterization of ferromagnetic materials. As a matter of fact, the material microstructure affects both the mechanical and the magnetic properties of ferromagnetic materials. Measurement of the magnetic properties (for instance, measurements of the incremental permeability along a hysteresis loop, multi-frequency eddy currents induced in a hysteretic medium, Barkhausen noise and harmonic analysis) can be used to infer mechanical properties in a nondestructive manner. The numerical modelling (see [1] and references therein) of these very complex measurement systems represents an essential tool for understanding the signature of the measurements, for probe optimization etc.

In fusion devices ferromagnetic materials are necessary for two main purposes. The first one is related to the shielding of the neutrons produced by fusion reactions: indeed in next-generation devices (ITER, DEMO), very high neutron fluence is expected. The second main field of application of ferromagnetic materials in fusion devices, which will be addressed in the present paper, is in the development of so-called ferromagnetic inserts to improve the toroidal field (TF) ripple.

The TF ripple, which can be deleterious for the overall performances of the device, originate from the spurious poloidal components produced by the ports need to guarantee accessibility of the interior of the machine that break the continuity of the toroidal field coils. In order to improve the situation, suitable pieces of ferromagnetic materials (the so-called ferromagnetic inserts) are placed along the torus.

The setup of quantitative methods to treat complex geometries and constitutive relationships (like ferromagnetic materials) could result in discretized models whose requirements (memory and computational time) could be beyond the actual limits of a standard computational unit. For this reason, the use of High Performance Computing (HPC) environment, resorting to parallel architectures, is mandatory. In modern workstations, tens of CPUs and one (or more) GPUs (Graphical Processing Unit) are usually available. Moreover it is not unusual to have a cluster of such workstations communicating by using a fast network interface. The architecture of the GPUs, where hundreds of cores are present, perfectly matches the parallel nature of graphics rendering: each pixel of the picture is an entry of a large matrices and all the graphic operations are implemented as the same mathematical operations on matrices, this type of computation is called Single Instruction Multiple Data (SIMD). On the contrary, a traditional CPU have a large part of the chip area which is dedicated to a sophisticated control logic and to a large cache memory in order to easily fit the requirement of codes for completely different applications. A scientific computational application is very different from standard graphical applications. Hence, an "ad-hoc" reformulation of the numerical methods and algorithms is mandatory to take full benefit from the large number of cores available in the device [2]. However, their use in FE codes for the computation of magnetic field is limited to few examples, since the iterative inversion such as the ICCG includes sequential computations[3]. Also, integral formulations can be easily parallelized, as already shown for a cluster of CPUs [4-5]. In this paper, the use of multi-GPUs for the computation of the matrices related to the solution of the magneto-quasi-static problem in presence of ferromagnetic materials is presented, with particular attention to fusion and NDT applications.

II. Numerical Model

We use an source integral formulation in which unknowns are the induced current density **J** and the magnetization **M**. The resulting coupled equations, for time-harmonic operations and linear materials, are [6]:

$$\left(\underline{\underline{R}} + j\omega\underline{\underline{L}}\right)\underline{I} + j\omega\underline{\underline{F}}\underline{M} = -j\omega\underline{v}_S \qquad (1)$$

$$j\omega\underline{\underline{F}}^T\underline{I} + j\omega\underline{\underline{E}}\underline{M} = -j\omega\underline{u}_S \qquad (2)$$

where the discrete unknowns are $\underline{I}$ and $\underline{M}$ (representing **J** and **M**, respectively), $\underline{\underline{R}}$ and $\underline{\underline{L}}$ are related to the conducting regions, $\underline{\underline{E}}$ is related to magnetic regions and $\underline{\underline{F}}$ is related to interaction between the electric and magnetic regions. The quantities $\mathbf{v}_s$ ad $\mathbf{u}_s$ are prescribed and related to the exciting coils. Matrix $\underline{\underline{R}}$ is sparse, whereas the others are fully populated matrices. The assembly effort is O(N^2), N being the number of DoFs. Since the system (1)-(2) is dense, a direct solver requires a computational cost increasing as O(N^3). However, in "medium size" problems, the assembly phase can be very time consuming, especially when a great accuracy is required, a great deal of geometrical details should be investigated, or it is necessary to evaluate several geometrical configurations (for instance a coil scan in NDT application). Hence, it is mandatory to parallelize also the assembly of the aforementioned matrices.

III. GPU-based Assembly

In the following, we give a brief description of the GPU-based $\underline{\underline{E}}$ matrix assembly; in the full paper, we will extend the method to other matrices in (1)-(2) ($\underline{\underline{F}}$ and $\underline{\underline{L}}$). Specifically, the $\underline{\underline{E}}$ matrix is built by computing the interactions between elements. These interactions are equally distributed between the available GPUs (domain decomposition). Each GPU works on a portion of the $\underline{\underline{E}}$ matrix, each thread compute an interaction elements to elements, all these contributes are finally scattered on the complete $\underline{\underline{E}}$ matrix on the CPU side. In this way, the total load is partitioned into single kernel calls and the possible race condition due to the summation of the contribute in the total matrix is avoided. An accurate tuning of the matrices access pattern has been done in order to minimize the uncoalesced access to the global GPU memory, this is obtained optimizing the overall device memory access, in the way that consecutive access to the threads is granted. The number of the threads and the grid size in a single "kernel shot" are optimized taking into account the resources to be used (global GPU and register memory).

IV. Test Cases and Numerical Results

Two examples are given to validate the performances: one in the framework of quantitative methods for NDT [1], and the other one in the field of plasma fusion device. Concerning the NDT case, we refer to the modelling of the so-called incremental permeability in material characterization of ferromagnetic materials. The system used to test the method has been described in [1]. In this system a magnetic field is applied to the sample under investigation via an ferrite U-shaped inductor, driven by two coils fed by high amplitude voltage at low frequency. A small coil (pick-up coil) is placed onto the specimen at center of the two legs of the yoke and driven by a low intensity current at high frequency. The measured quantity is the impedance of this pick-up coil as a function of the magnetic field measured via a Hall-sensor. This quantity is somehow related to the incremental permeability along the hysteresis loop (see [1]).

The second example is about a fusion application. The effect of ferromagnetic inserts on the TF ripple of DEMO [7], a device currently under design, is investigated.

The mesh (see Fig. 1) used for NDT (resp. fusion) test case has the number of elements equal to 6912 (resp. 5148); the corresponding number of elements of the matrix **E** is equal to 20736 (resp. 15444). Using one CPU the time required to assembly the matrix **E** is equal to 6251 s for NDT test case and 140075 s for fusion example. It is worth noting that using a standard parallelization, ideally the computational time scales linearly with the number of CPUs used in computation.

The computational cluster used for the evaluation of the numerical performances of the method is a 4 Nodes HP ProLiant ML370 G6 Base: 8 cores Intel Xeon E5345, 2.33GHZ , 72 GB RAM, 8MB L2. On each node is present a GPU NVIDIA Tesla C2050, 3 GB. Let the speedup parameter be defined as the ratio between the time needed for matrix assembly on one CPU, divided by assembly time using the GPUs. In Fig. 2 we report the speedup parameter versus the number of the GPUs used in the computation. The gain is extremely significant and scales almost linearly with the number of GPUs.

This work was supported in part by Italian MIUR under PRIN grant 2010SPS9B3.

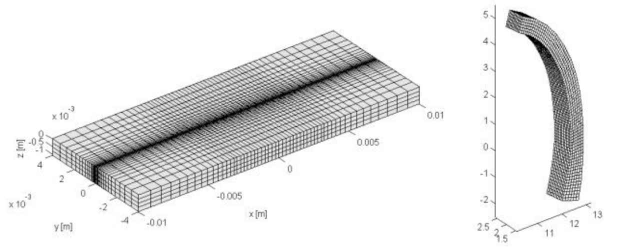

Fig. 1. The meshes used for NDT (left) and for fusion (right) test cases

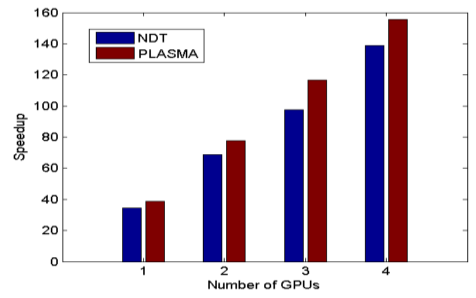

Fig. 2. The speedup versus the number of GPUs.

References

[1] M. D'Aquino et al., *IEEE Trans. Mag.* **50** (2014) p. 7001104
[2] M. Bernaschi et al., *Comp. Phys. Comm.* **184** (2013) p. 329-341
[3] T. Okimura et al., *IEEE Trans. Mag.* **49** (2013) p. 1557-1560
[4] G. Rubinacci et al., *J. Comp. Phys.* **228** (2009) p. 1562–1572
[5] F. Villone et al., *Plasma Phys. Control. Fusion* **54** (2012) p. 085003
[6] G. Rubinacci et al., *Nondestr. Testing Eval.* **24** (2009) p.165-194
[7] G. Federici et al., *Fus. Eng. Des.* **89** (2014) p.882-889