GPU Acceleration of Algebraic Multigrid for Low-Frequency Finite Element Methods

Elia A. Attardo*, and Andrea Borsic*
*Thayer School Of Engineering
Dartmouth College, 8000 Cummings Hall, Hanover, NH 03755, USA
Email: {elia.a.attardo, andrea.borsic}@dartmouth.edu

***Abstract*—This paper introduces a GPU acceleration of a Wavelet-based Algebraic Multigrid used as preconditioner for solving the Laplace's equation discretized by Finite Element Method. We conduct some tests using a CPU-based direct solver, a CPU-based Preconditined Conjugate Gradient (PCG), and a GPU-based PCG. Finally, we report the solution time and the speed-up achieved in solving the discretized problem.**

I. Introduction

Multigrid Method (MG) is a well-known numerical technique for solving large sparse linear systems of equations [1]. Generally, the Laplace's equation is expressed as a Partial Differential Equation (PDE), and its discretization leads to a sparse linear system of equations. Then, a hierarchy of meshes is formed and used to solve the original problem. In the Algebraic Multigrid (AMG), instead of forming a hierarchy of meshes, the discretized (sparse) matrix is used to build a hierarchy of matrices. In this, mesh (geometry) information is not required, as the technique is entirely algebraic. AMG methods have been proved to be efficient in different applications, especially in solving problems that arise from unstructred meshes. AMG methods have been used as an efficient preconditioner for iterative solvers such as Conjugate Gradient (CG) or Generalized Minimal Residual (GMRES). Therefore, even though these methods are usually recognized as a *black box solver*, some attention must be devoted in choosing appropriately the operators which will create the matrices hierarchy (*restrictor*, and *prolongator* operators). In [2], the use of a Discrete Wavelet Transform (DWT) was proposed to interpolate between different levels of the hierarchy matrices. This scheme is generally known as Wavelet-based AMG (WAMG), and its use allows us to efficiently parallelize the solution of the discretized PDE.

An other disadvantage of traditional AMG methods is that they require the inspection of the whole system matrix to determine certain weight factors [1]. If the system matrix is stored across different computing nodes, this requires a certain amount of inter communication. WAMG methods do not require this inspection, and the resulting communication between different parallel execution units. In other words, WAMG methods are ideal candidates for the implementation on Graphic Processing Units (GPUs), thanks to their good parallelization properties.

In this work we consider the solution of the Laplace's equation solved with the Finite Element Method (FEM) using piece-wise constant basis functions. We study the acceleration of a GPU-based WAMG using a biomedical test case, where electric fields applied to the body through electrodes need to be computed in the volume of the body. Here we extend the work proposed in [3] providing a GPU acceleration for AMG methods.

Firstly, we set the notation (Sec. II) and we introduce the WAMG for solving the Laplace'equation. Finally, we show some results in terms of speeding up the solution of the system when GPUs are used (Sec. III-A).

II. Formulation

Consider the situation in which the electric field is assumed conservative and the conduction currents dominant with respect to the displacement currents. This condition leads to the following PDE:

$$\nabla \cdot \sigma \nabla u = 0 \qquad \text{on } \Omega \tag{1}$$

where σ is the conductivity or admittivity of the body to be imaged, u is the electric potential, and Ω the body to be imaged. Electrodes are commonly modeled with some boundary conditions [4], resulting in the following boundary condition for each portion of the boundary $\partial\Omega_\ell$ underneath electrode ℓ:

$$\int_{\partial\Omega_\ell} \sigma \frac{\partial u}{\partial \mathbf{n}} = I_\ell \qquad \ell = 1 \dots \mathrm{L} \tag{2}$$

where I_ℓ is the current injected at electrode ℓ and L is the number of electrodes.

Equations (1) to (2) together with a set of boundary conditions allow to compute the electrode voltages V_ℓ for any given conductivity distribution σ, and usually are solved with the Finite Element Method. The resulting system of linear equations is represented in a matrix fashion as follow:

$$\underline{\underline{A}}\,\underline{x} = \underline{b} \tag{3}$$

where the matrix $\underline{\underline{A}}$ is *weakly diagonal dominant* (*M-matrix*) [1], and $\underline{b}$ is the right-hand side.

III. Wavelet-based AMG on GPU

In the framework of AMG methods, the solution of the system matrix in eq. 3 is obtained iteratively. Indeed, only one direct solution is performed at the last level (coarsest

Fig. 1. CG with WAMG (dark), and with diagonal preconditioner (red). The desired tolerance of 10^{-9} is not reached within 300 iterations when the WAMG is not employed.

mesh) of the formed hierarchy, and then the errors, between different levels are updated. To do that, two operators are needed. The first is the so-called *restrictor*, which is used to build a matrix on a coarser level starting from a fine one; and the second is the *prolongator*, whose operates conversely. For WAMG these operators are derived from the DWT without need of inspecting the system matrix. We note that the previous scheme is known as *V-cycle* algorithm [1], and it is used to approximate the solution of the system in (3). In this paper, the WAMG is employed as a preconditioner for the CG iterative solver, preconditioned-CG (PCG). The restrictor and prolongator operators are, at each step of the V-cycle, computed on-the-fly; we use the second order Daubechies wavelets [5].

To show the speed-ups achievable by means of the GPUs, we run some tests using a direct solver. In [6] it is shown that PARDISO [7] can be significantly faster than the iterative methods proposed in literature. Therefore, direct solvers have the inconvenience that they require more memory than iterative solvers, as they need to store the factored matrix, and they lend themselves less well to parallelization. In the following, the direct solver (PARDISO) will run on CPU, while the WAMG on GPU.

A. Results

In this section we discuss numerical results relative to the acceleration of the solution of the system in eq. 3. We present the solution time while a CPU-based multiple core server, and a GPU computing server are used. The CPU-based system is a Dell Power Edge 1955 Blade Server. The server is based on two quad-core Xeon 5355 "Clovertown" CPUs, with an internal clock frequency of 2.66GHz and front-side bus speed of 1.33GHz, 8 cores at 64-bit. The GPU-based system is a NVIDIA Tesla S1070 server, which consists of four Tesla GPUs. Each GPU has 240 cores (960 in total) with an internal clock frequency of 1.5GHz. Windows 7 64-bit Enterprise edition is installed on the host CPU. For the time being we use only one of the four GPUs available.

At this stage we consider only real-valued problems. Development of complex-valued solver is under way. Also, the discretization procedure of the physical problem is conducted on different mesh density resorting in $\underline{\underline{A}}$ matrix with different size. The dimensions considered are 59,000; 146,000; 300,000; and 500,000. We solve these four problems by using the WAMG-PCG on GPU and PARDISO on CPU. Figure III-A shows the number of iterations versus the relative residual of the PCG when the WAMG is used (dark curve) and when only a diagonal preconditioner is employed (red curve) for $\underline{\underline{A}}$ with size equal to 146,000. It is possible to observe that the number of iteration required to get a desired tolerance ($\epsilon = 10^{-9}$) is reached within 30 iterations by using WAMG.

Finally, in Table I we report the timing in solving the systems, with different size, by using PARDISO and WAMG-PCG. For both methods the tolerance is equal to $\epsilon = 10^{-9}$.

TABLE I
SOLUTION TIME FOR REAL-VALUED PROBLEMS: PARDISO (CPU, MULTI-THREADS) VS WAMG (GPU).

Size matrix (K)	PARDISO (sec)	WAMG (sec)	Speed-up
59	3.5	1.7	2.05
146	10.83	3.73	3
300	76.86	21	3.5
500	787	96	8.2

IV. CONCLUSIONS

In this work we present a sparse linear solver based on WAMG, and we discuss its acceleration on GPUs. In particular, we report a significant speed-up in solving the system matrix on GPU with respect to a direct solver on CPU. For a test mesh with 500,000 unknowns a relative speed-up of almost 8 for real-valued problems has been obtained using only one GPU. We leave for the forthcoming papers the acceleration of system matrix by using all computational capabilities of the NVIDIA S1070 (eg. all four GPUs).

REFERENCES

[1] W. Briggs, V. Henson, and S. McCormick, *A Multigrid Tutorial*. SIAM, 2000.

[2] F. Pereira, S. Verardi, and S. Nabeta, "A wavelet-based algebraic multigrid preconditioner for sparse linear systems," *Applied Mathematics and Computation*, vol. 182, pp. 1098–1107, 2006.

[3] A. Borsic and R. Bayford, "Forward solving in electrical impedance tomography with wavelet based preconditioners and implementation results on multi-core and gpu platforms," No. 224 in Journal of Physics, 2010.

[4] E. Somersalo, M. Cheney, and D. Isaacson, "Existence and uniqueness for electrode models for electric current computed tomography," *SIAM J Appl Math*, vol. 52, pp. 1023–1040, 1992.

[5] I. Daubechies, *Ten lectures on wavelets*. Regional Conference Series in Applied Mathematics, 1992.

[6] A. Borsic, A. Hartov, and K. D. Paulsen, "Optimization of 3d impedance tomography reconstruction on mulit-core computing platforms," *9th Int. Conf. on Biom. App. of EIT, Hanover NH, USA*, 2008.

[7] "http://www.pardiso-project.org/."