Solving the Euler Equations for Unstructured Grids on Graphical Processor Units

Mario J. Martin[1], Esther Andres[1], Carlos Carreras[2] and Francisco Palacios[3]

[1] INTA – Fluid Dynamics, Crta. Ajalvir Km 4, 28850, Madrid, Spain.
[2] ETSIT, Ciudad Universitaria s/n, 28040, Madrid, Spain.
[3] IMDEA Mathematics, Facultad de Ciencias UAM, Ciudad Universitaria de Cantoblanco, 28049, Madrid, Spain.

Abstract. The long computation times required for fluid dynamics simulations (CFD) has lead the industry to look for some alternatives to boost high performance computing (HPC). This paper is focused on the acceleration of fluid dynamics simulations for industrial complex configurations using modern graphics cards (GPUs) that exhibits a substantial parallel architecture. First, some general facts about general purpose programming with GPUs (GPGPU) are presented. Then, we explain the implementation of a CFD solver for the resolution of 2D Euler equations on unstructured grids and some problems related to the parallelization of this algorithm. Finally, two approaches based on hybrid CPU-GPU, and full GPU implementations are compared.

Keywords: Graphics Processor Unit, Hardware Acceleration, Euler Equations, Unstructured Grid, Computational Fluid Dynamics, CUDA.

1 Introduction

Nowadays, modern graphics hardware outperforms the traditional desktop CPU in terms of computational processing power by several magnitudes with a very attractive cost/performance ratio [1]. Commodity graphics processing units (GPUs) have evolved into high performance parallel architectures, driven by multiple cores and very high memory bandwidths. GPGPU stands for General-Purpose computation on GPUs. With the increasing programmability and flexibility of graphics processing units, this hardware is capable of performing not only computations for image rendering applications, but also computations in a wide variety of fields: from sparse matrix multiplications techniques [2] to multigrid, conjugate-gradient solvers for systems of partial differential equations [3], N-body simulations [4][5] and physical simulations such as fluid mechanics solvers [6][7][8].

In Computer Fluid Dynamics (CFD) applications, the increasing demands for accuracy and simulation capabilities produce an exponential growth of the required computational resources. This situation calls for new simulation platforms based on heterogeneous architectures in which conventional processors and specific hardware modules work together. Some alternatives for High Performance Computing (HPC)

for scientific applications are the acceleration using GPUs, FPGAs, ASICs [9], and the Cell processor [10].

This paper describes how commodity graphics cards can be used to perform fluid dynamics simulations for industrial complex configurations. This work is focused on the acceleration of CFD solvers for compressible Euler equations on unstructured grids, which is one of the most demanding applications used in aeronautical simulation. There are previous similar works in this area [11] based on regular grids where each pixel represents a cell of the grid, but solving the Euler equations on unstructured grids has remained a challenge.

1.1 Euler Equations

The equations to be solved are the 2D Euler equations for compressible inviscid flow in integral conservative formulation over a general region Ω with boundary $\partial\Omega$.

$$\frac{d}{dt}\iint_{\Omega} w\,dx\,dy + \oint_{\partial\Omega} (f\,dy - g\,dx) = 0 \qquad (1)$$

where w is the vector of conserved flow variables, and f, g are the Euler flux vectors for compressible inviscid flow defined as

$$w = \begin{Bmatrix} \rho u \\ \rho u + p \\ \rho v \\ \rho E \end{Bmatrix}, f = \begin{Bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u H \end{Bmatrix}, g = \begin{Bmatrix} \rho u \\ \rho u v \\ \rho v^2 + p \\ \rho v H \end{Bmatrix} \qquad (2)$$

where ρ, u, v, H, E, p are the density, Cartesian velocity components, total energy, total enthalpy, and pressure, respectively.

1.2 Implementation

A finite volume method on unstructured meshes [12] is used for the numerical discretization of the fluid dynamics equations. The Euler equations are numerically solved by local time-stepping using an upwind scheme with Roe's approximate Riemann solver on a triangular mesh. A Runge Kutta formulation is used for the time integration.

An in-house code for the resolution of Euler equations has been developed using GPUs as the core computational resource for the CFD solver and two strategies has been tested: The first one is a hybrid CPU-GPU computation where the costly portions of the solver are implemented in the GPU, and some considerations should be taken about the transfer time through the PCI express. The second one is a full

GPU implementation of the main loop of the solver (see Fig. 1) where a strategy based on vertex partitioning is proposed in order to ensure the non-interference of concurrent threads and fully exploits the intrinsic parallelism on modern GPUs.

An equivalent solver is made in C++ to be run in a CPU for benchmarking purposes. Because the GPU only supports single point-precision, both CPU and GPU implementations work with single point-precision in order to be as similar as possible.

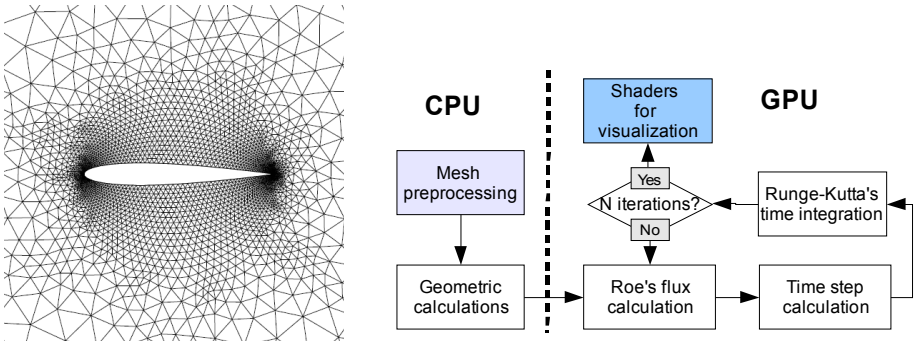

Fig. 1. Unstructured grid for the NACA0012 airfoil (*left*) and process flowchart (*right*) used in this example to solver the Euler equations using a first order upwind scheme.

1.3 Numerical Results

We have selected a two dimensional NACA0012 airfoil mesh, with 7.883 nodes, 23.221 edges and 15.338 triangles. The execution times for one iteration loop of the different solver implementations are given in Table 1, as well as the performance obtained from the same solver running on the CPU. The implementation of the solver in the graphics card has been developed using *Compute Unified Device Architecture* (CUDA) and tested on one Nvidia 8800GTX graphic card. The CPU implementation is coded in C++ and executed on a CPU Intel Core 2 Duo E6600 (but using only one core).

Table 1. Execution times of the main loop of the CFD solver implemented.

Implementation	Times in ms	Speedup
Full one core CPU	13.906	---
Hybrid CPU-GPU	2.854	x5
Full GPU	1.029	x13.5

1.4 Conclusions

A speedup of almost 14 has been achieved using a full implementation of the main loop of the algorithm in the GPU while the hybrid implementation has achieved a

speedup of 5 times. The most efficient way is a full implementation, because avoids data transfers through the PCI express bus. On the other hand, multi-thread programming model of GPUs is quite different from the sequential instruction model of most CPUs architectures. Traditional algorithms can be implemented in parallel architectures, like modern GPUs, but some algorithms must substantially be modified in order to achieve full performances in parallel architectures.

With emerging environments for programming applications on GPUs for general purposes such as Standfor's Brook, AMD's C*lose to The Metal* (CTM), and NVIDIA's *Compute Unified Device Architecture* (CUDA), the real challenge is to develop efficient algorithms for streaming computing, where thousands of threads are executed simultaneously

References

1. NVIDIA Corporation, "Compute Unified Device Architecture Programming Guide", http://www.nvidia.com, (2007).
2. Krüger J., Westermann R., "Linear algebra operatoes for GPU implementation of numerical algorithms", *ACM Transactions on Graphics* 22-3, (2003).
3. Bolz J., Farmer I., Grinspun E., Schröder P., "Sparce matrix solvers on the GPU: Conjugate gradients and multigrid", *ACM Transactions on Graphics* 22, 3, 917-924, (2003).
4. Belleman, R.G. Bédorf, J., and Zwart, S.F.P., "High performance direct gravitacional N-body simulations on graphics processing units II: An implementation in CUDA", *New Astronomy*, Vol 13, Nº 2, 103-112, (2008).
5. Mark J. Stock, and Adrin Gharakhanii, "Toward efficient GPU-accelerated N-body simulations", *46th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA 2008-608, (2008).
6. Harris M., "Fast fluid dynamics simulation on the GPU", GPU Gems, Addison Wesley, 637-665, (2004).
7. Trond Runar Hagen, Knut-Andreas Lie, and Jostein r. Natvig, "Solving the Euler equations on graphics processing units", *Lecture Notes in Computer Science*, Vol 3994, 220-227, (2006).
8. Tobias Brandvik and Grahan Pullan, "Acceleration of a 3D Euler solver using commodity graphics hardware", *46th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA 2008-607, (2008).
9. E.Andres, C.Carreras, G.Caffarena, O.Nieto-Taladriz, F.Palacios, "CFD acceleration through Reconfigurable hardware", *46th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA-2008-481, (2008).
10. Williams, S., Shalf, J., Oliker, L., Kamil, S., Husbands, P., and Yelick, K., "The potential of the Cell processor for scientific computing. In CF 06: Proceedings of the ACM International Conference on Computing Frontiers, p.p. 9-20, (2006)
11. D. Owens, D. Luebke, N. Govindaraju, M. Harris, "A survey of general-purpose computation on graphics hardware," in Eurographics 2005, State of the Art Reports, pp. 21–51, (2005)
12. Timothy J. Barth "Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes equations". AGARD Report 787, 6.1-6.61, (1992).