HETEROGENEOUS PARALLEL IMPLEMENTATION OF A SEMI-IMPLICIT 3D HYDRODYNAMIC MODEL

Mario Acosta, Universidad de Granada, 660012701 34, marioa@ugr.es

1. Mario C. Acosta, Arquitectura y Tecnología de Computadores, Instituto del Agua
2. Mancia Anguita, Arquitectura y Tecnología de Computadores
3. Francisco J. Rueda, Ingenieria Civil, Instituto del Agua
4. F. Javier Fernández-Baldomero, Arquitectura y Tecnología de Computadores

This work presents the first heterogeneous-platform parallel implementation of the SI3D hydrodynamic model [1] that makes use of both the multiple cores and the GPU of the nodes of a cluster. The implementation takes advantage of the cores by using either OpenMP or MPI and of the Nvidia GPU by using CUDA.

High performance computations are being increasingly demanded in water sciences to get detailed descriptions of the flow fields that develop in natural ecosystems within reasonable lengths of time by means of simulations conducted with three-dimensional (3D) numerical algorithms solving the governing equations of fluid motion. Many of the 3D hydrodynamic models currently used in lake research are based on the solution of a Navier-Stokes equations simplification, the shallow water equations (SWE), in which the vertical pressure gradients are assumed hydrostatic. The spatially-varying horizontal velocities (u, v) and the water surface elevation (η) are the main state variables in the SWE. Although SWE models have a moderate computational cost, they are still time and memory consuming when high density spatial grids are used or when they are used to simulate the long-term behavior of natural water systems. The hydrodynamic model improved in this work, SI3D [1], is a SWE model widely validated [2]. SI3D decomposes the domain in the horizontal using a structured grid of square water columns; the vertical is composed of a set of layer according to the depth.

The OpenMP and the MPI parallel SI3D implementation [3] uses domain decomposition to divide the workload among the core. These subdomains are not completely independent, so is necessary to add communication/synchronization or redundant computation at certain points. A computational cost analysis of SI3D showed that assembling and solving a large number of independent small tridiagonal systems (there is one for each column) at five points in the program consume 70% of the total execution time. In the heterogeneous implementation proposed, the GPU constructs and solves all the tridiagonal systems in parallel. While the GPU performs this work, the CPU is not idle, it is evaluating explicit terms in x-momentum and y-momentum equations. GPU receives the work and returns the results by asynchronous communication. Using a computer with one Intel Xeon multi-core (4 cores) and one GForce GTX 470 GPU, a speed-up, compare to the execution in a core, of 16 to 20 is obtained. This outperforms the speed-up of 3.6 achieved with the 4 cores.

Moreover, we have developed an API (Application Programming Interface) to access CUDA from FORTRAN due to the limitations of the already available CUDA-aware programming tools for FORTRAN programming, which avoided to reaching a good speed-up.