PHPC : Theorethical analysis - Shallow Water Wave Equation with a finite volume solver

Arnaud Pannatier
(Dated: May 12, 2018)

I. INTRODUCTION

This project propose a way to increase the perfomance of a sequential code using classical parallelism technics, in particular the . It is based on a Matlab code of [author of matlab code] that simulate the evolution of a tsunami. The purpose of this report is to describe the state of the current projet and to describe how the sequential code will parallelized. It also give the expected results for the speed up, as well as the strong and weak scaling of the problem.

II. DESCRIPTION OF THE CODE

The code is arranged in the following way. The algorithm will compute the evolution of the height of the wave $H(x, y)$, based on the knowledge of the topography and the evolution of the speed of the wave in the direction x and y. In order to start the computation, the algorithm first read the initial conditions for the differents values (H,HU,HV, Zdx, Zdy) in binary files. The duration of the evolution T_{end} is specified by the user. While this time is not reached, the algorithm will compute the next variable timestep and the evolution of the variable H,HU,HV. The update of the variables only needs informations of the direct neighbor of a point (x,y). When the simulation is over the height of the wave at the final timestep is stored in a file.

III. PARALLELIZATION

The sequential code will be parallelized using `MPI`. The reader, writer and timestep computation as well as the update of the variables H, HU, HV will be implemented.

The parallelization of the reader and writer will make use of the Parallel I/O of MPI. Computing the next timestep implies that the max over a grid should be found. This can be done in a parallel fashion using the following design : as `C++` is a [INSERT major row] language, the grid will be splitted between nodes in block of $\frac{N}{P}$ complete row. Each processor will then return the max over his subgrid and the final max will be computed at the end, this final computation will be done on a single node. An hybrid approach using OpenMP to parallelized inside a node will be tested. The parallelization of the update of H, HU, HV will be approach in a similar maner. The grid will be splitted as before. before and after each subgrid (except the first one and the last one) a row of ghost cells will be added, in order to allow the computation. Most of the code will therefore be parallelized.

IV. PROFILING

V. THEORETHICAL RESULTS