

Parallel Computing for the Simulation of 3D Free Surface Flows in Environmental Applications

Paola Causin¹ and Edie Miglio¹

MOX - Modeling and Scientific Computing, Dipartimento di Matematica "F.Brioschi"
Politecnico di Milano, via Bonardi 9, 20133 Milano - Italy
Paola.Causin@mate.polimi.it, Edie.Miglio@mate.polimi.it,
WWW home page: <http://www.mox.polimi.it>

Abstract. The numerical simulation of 3D free surface flows in environmental fluid-dynamics requires a huge computational effort. In this work we address the numerical aspects and the computer implementation of the parallel finite element code *Stratos* for the solution of medium and large scale hydrodynamics problems. The code adopts the MPI protocol to manage inter-processor communication. Particular attention is paid to present some original details of the software implementation as well as to show the numerical results obtained on a Cray T3E machine.

1 Introduction and motivation

Free surface flows are encountered in various natural phenomena, such as tidal flows, large water basins and river courses. The huge computational effort required by a full 3D numerical simulation has led in the past to developing simplified 1D and 2D models. These latter models are nowadays well established, both in terms of a sound mathematical formulation and of a robust numerical implementation []. Nonetheless, many situations of relevant interest occur that require a more accurate physical description of the problem in order to model all the significant phenomena.

The so-called *Quasi-3D Shallow Water Equation* model (Q3D-SWE) considered in the present work enriches the 2D models by a proper selection of 3D features (*e.g.*, a velocity field that depends on all the three spatial variables), with a computational cost not severely exceeding the cost of pure 2D models.

However, it is a fact that even adopting the Q3D-SWE model, hydrodynamic problems in environmental applications inherently require a very large number of unknowns. This fact, considering furthermore the fairly long characteristic times involved in these problems (often several days), lead to exceedingly expensive computations. Consider as an example the simulation of the tides in the Venice Lagoon whose numerical results will be presented in Sect.4. Such simulation, spanning the tidal cycle of two days, has been carried out using 20 layers in the vertical direction and about 31500 triangles in the horizontal plane covering an area of 1800 km², for a total of approximately of 2 millions degrees of freedom with a time step of 15 minutes. These latter figures clearly demonstrate that a parallel approach is in order. The results presented in this work have been obtained running the code *Stratos* on a T3E Cray machine using up to

8 processors. The code adopts the MPI Message Passing Protocol in order to guarantee the maximum portability; it is indeed planned to run the simulations in the future on a cluster of PCs.

The paper is organized as follows: in Sect.2 we present the Q3D-SWE mathematical model; in Sect.3 we address the finite element discretization of the Q3D-SWE model and the parallel computer implementation; in Sect.4 we present the numerical results, while in Sect.5 we draw the conclusions and we present the future directions of the work.

2 The Q3D-SWE mathematical model

The Q3D-SWE model is derived from the three dimensional incompressible Navier-Stokes equations by integrating the continuity equation along the vertical direction,

Before presenting the model, we need to set up some basic notations. Indicating by $\widehat{\Omega}$ a parallelepipedon which includes the water body for the whole time interval of interest and denoting by Ω the base of this parallelepipedon, the bottom surface and the free surface of the domain are described by the bottom bathymetry $z = -h(x, y)$ and by the elevation of the free surface $z = \eta(x, y, t)$, respectively. The total depth of the fluid at a generic point (x, y) is denoted by $H(x, y, t) = h(x, y) + \eta(x, y, t)$. Under the assumption of *long wave phenomena*, vertical accelerations can be neglected, leading to the following *hydrostatic approximation* for the pressure

$$p = p_0 + \rho g(\eta - z), \quad (1)$$

where p_0 is the given atmospheric pressure and g is the gravity acceleration. The Q3D-SWE model reads [1999]:

$\forall t \in (0, T]$, find (\mathbf{u}, w, η) such that

$$\begin{cases} \frac{D\mathbf{u}}{Dt} = -g\nabla_{xy}\eta + \nu \frac{\partial^2 \mathbf{u}}{\partial z^2} + \mathbf{f}, \\ \frac{\partial \eta}{\partial t} + \nabla_{xy} \cdot \left(\int_{-h}^{\eta} \mathbf{u} dz \right) = 0, \\ \frac{\partial w}{\partial z} + \nabla_{xy} \cdot \mathbf{u} = 0, \end{cases} \quad (2)$$

where ∇_{xy} denotes the spatial gradient in the horizontal plane $D/Dt = \partial/\partial t(\cdot) + \mathbf{u} \cdot \nabla(\cdot)$ is the Lagrangian time derivative, ν is the vertical eddy viscosity coefficient, $\mathbf{u} = (u, v)^T$ is the horizontal velocity, w is the vertical velocity, \mathbf{f} is the external force vector (typically the Coriolis force). Notice that relation (1) has been employed to express the gradient of the pressure in the momentum equation as a function of the sole elevation gradient. System (2) is completed by a proper set of initial and boundary conditions that specify the elevation or the velocity of the fluid on the boundary of the domain (water-water boundary or water-solid wall boundary, respectively). Wind effects as well friction effects can be also accounted for by forcing to a given value the tangential component of the normal stress on the free surface or on the bottom, respectively.

3 Parallel implementation of the Q3D-SWE model

In this section we address the issue of the discretization of model (2) and we focus on the aspects concerning its computer implementation in a parallel architecture.

3.1 Discretization and partitioning of the domain

The discretization method adopted in this work decouples the vertical and the horizontal directions. This procedure is a natural consequence both of the mathematical form of the Q3D-SWE equations and of the geometrical characteristics of the computational domain. Namely, the parallelepipedon $\hat{\Omega}$ is sliced in the vertical direction into several horizontal parallel layers, with possible different heights. An unstructured triangulation is used in the horizontal plane xy , in order to provide an accurate description of the irregular planar geometries that frequently occur in these problems. The replication of the same triangular grid in the middle point of each layer creates a mesh of right-angle prisms. At each time level, a prism may lay entirely or partially under the free surface level or may be dry. Tracking the free surface position is thus reconducted to managing efficiently the emptying and filling of the prisms. Very irregular bathymetries of the bottom, as obtained from experimental measurements, can be handled as well. In Fig. 1 we show the very irregular bathymetry of the bottom (left) and the finite element triangulation in the xy plane (right) for the Venice Lagoon. Notice how the maximum density of the triangulation is reached in correspondance of the lagunar isles and of the channels that exchange the fresh water from the Adriatic Sea to the Lagoon, where more resolution is needed.

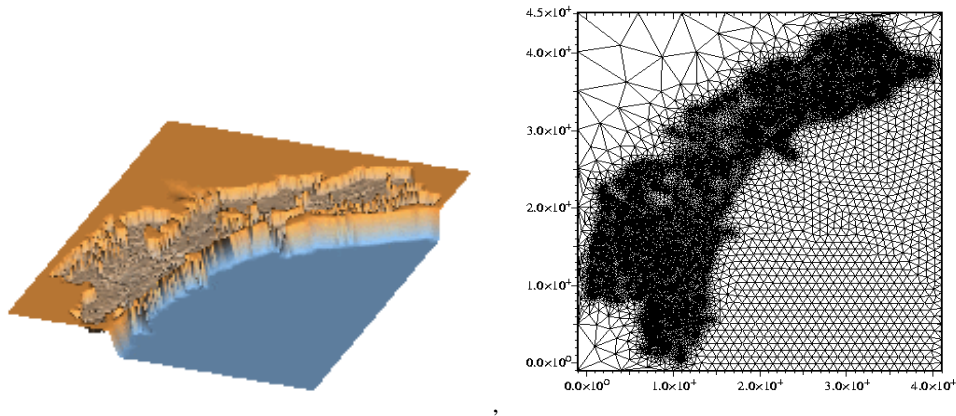


Fig. 1. The complex bathymetry of the Venice Lagoon (left) and the mesh in the horizontal plane (right)

The basic prismatic cell one layer deep in the z direction may be regarded as the most *fine grained* decomposition level of the problem. The granularity of the partition is increased by agglomerating all the layers of an element into a column. Each column

of cells is then mapped to a processor. This choice significantly reduces communication costs by increasing locality. The package Metis [2001] has been used to partition the mesh in the horizontal plane, since each column is unambiguously identified through a tag that is the number of its base triangle in the xy plane. Fig. 2 shows an exploded 3D view of the subdomains assigned to each of the 8 processors used in the simulation.

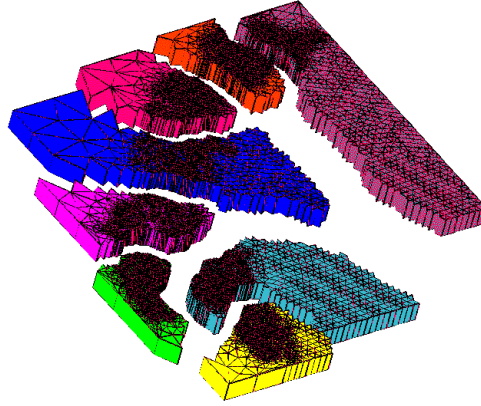


Fig. 2. An exploded view of the domain partitioning for the simulation of the Venice lagoon

3.2 Finite element approximation of the 3D-SWE model

We refer to [1999] and [2002] for a detailed discussion of the finite element spaces adopted for the approximation of the Q3D-SWE model. Here we limit ourselves to present the aspects that are relevant to the parallel computer implementation. Among the main issues to deal with, we mention:

1. the management of unstructured grids in a distributed architecture that demands for additional connectivity structures;
2. the existence of several procedures that require a search in all the elements of the domain (as happens in the computation of the starting point of the characteristic line, that will be discussed in the following) must be suitably specialized to the multiprocessor environment;
3. the objective of obtaining the maximum efficiency of the code must be pursued avoiding undeterministic behaviors that may often occur during the interprocessor message passing phase in a noncorrectly planned algorithm.

On this basis, it has appeared convenient to adopt an object-oriented programming concept, so to exploit the intrinsic links of data and underlying structures in association with the tasks (methods) they have to carry out in the parallel environment. The introduction of abstract data types can be applied to describe geometric objects such as elements and edges as well as to collect a set of tasks such as creating the element

matrix or updating the variables. Let us consider two important classes of mesh objects: the *element class* and the *edge class*. We may think that the first class is associated to the finite element discretization of the elevation, that is piecewise constant. Each element object contains the required connectivity data (its name, the names of its edges and vertices and the names of its adjacent elements), the *ids* of the processors that own the element itself and the adjacent elements, the geometrical data (area, coordinates of the circumcenters and of the vertices) and the numerical values of the quantities associated with the element (elevations and vertical velocity). Let us consider as an example Fig.3.2, where the area of a larger domain is depicted and let us focus the attention on the elements labeled by ⑤ and ⑭. The abstract element class and the corresponding

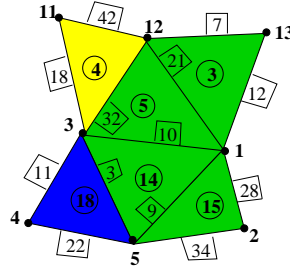


Fig. 3. Each color in the figure identifies a different processor. Circle: element numbers; square: edge numbers; solid points: vertex numbers.

instances for elements ⑤ and ⑭ are represented in Fig.5 (left). The second class of objects, edge objects, retain the information regarding the number and the thickness of the active layers that at each temporal step describe the actual shape of the fluid and the bathymetry. Moreover, they represent the fundamental geometrical structure for the horizontal velocity in the xy plane. Indeed, Raviart-Thomas finite elements of lowest degree are employed, yielding a finite volume-like approach that uses as degrees of freedom the fluxes of the unknown quantity across the edges of the triangulation. Edge objects are assigned the fundamental role to manage inter-processor communication. Indeed, the element class does not perform message passing but it delegates communication to the edge class. The edges collect and broadcast data to be sent, distribute received data and retain all the information required for a proper organization of the message passing itself. To improve efficiency, while the interface edges perform *non-blocking* message passing actions, internal edges overlap computation on processor-owned data. The abstract edge class and the corresponding instances for edges [32] and [3] are represented in Fig.5 (right).

3.3 Computation of the Lagrangian time derivative

In order to ensure numerical stability when dealing with the convective terms in (2), a Lagrangian approach is considered. This turns out to add a number of complications in

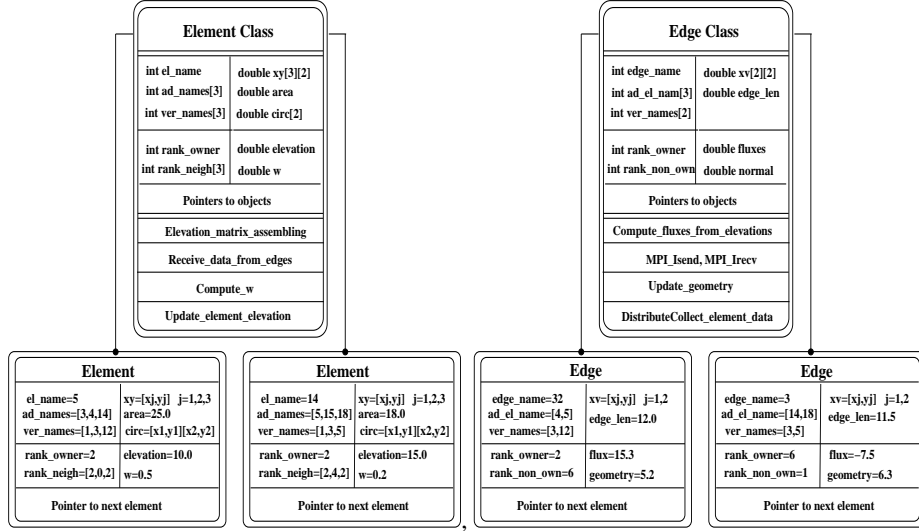


Fig. 4. The element class with its attributes and methods and two instances (left) and the element class with its attributes and methods and two instances (right).

the parallel implementation. Denoting by \mathbf{x}_{mi} the midpoint of a vertical face of a prism, the Lagrangian discretization of the convective term reads:

$$\frac{D\mathbf{u}}{Dt}(t_{n+1}, \mathbf{x}_{mi}) \simeq \frac{\mathbf{u}(t_{n+1}, \mathbf{x}_{mi}) - \mathbf{u}(t_n, \mathbf{X}(t_n; t_{n+1}, \mathbf{x}_{mi}))}{\Delta t},$$

where $\mathbf{X}(s; t, \mathbf{x})$ is the solution of the following problem:

$$\begin{cases} \frac{d\mathbf{X}(s; t, \mathbf{x})}{ds} = \mathbf{V}(s, \mathbf{X}(s; t, \mathbf{x})) \text{ for } s \in (0, t), \\ \mathbf{X}(t; t, \mathbf{x}) = \mathbf{x}. \end{cases} \quad (4)$$

From a geometric point of view $\mathbf{X}(\cdot) = \mathbf{X}(\cdot; t, \mathbf{x})$ is the parametric representation of the streamlines: *i.e.* $\mathbf{X}(s; t, \mathbf{x})$ is the position at time s of a particle which has been driven by the field $\mathbf{V} = (\mathbf{u}, w)$ and that occupied the position \mathbf{x} at time t .

Equation (4) can be solved using an explicit Euler method (see Fig. 5) which gives the following approximation of $\mathbf{X}(t_n; t_{n+1}, \mathbf{x})$:

$$\mathbf{X}(s; t, \mathbf{x}) = \mathbf{x}_{mi} - \mathbf{V}\Delta t \quad (5)$$

where the velocity \mathbf{V} is the velocity evaluated at \mathbf{x}_{mi} at time t_n . From the left part of Fig.5 it is evident that the Euler method may produce very bad results in presence of strongly curved streamlines unless a very small time step is used. To overcome this problem we solve problem (4) using a *composite* Euler scheme: the time step Δt is splitted into N_δ substeps (generally $N_\delta \simeq 4$ and in each substep a simple Euler scheme

is used (see right part of figure 5). In this way the streamlines are better approximated and if the number of substeps is properly chosen for the problem at hand the computed streamlines do not cross solid boundaries.

It is clear that the characteristic method relies strongly upon an efficient searching algorithm, since we have to find the element in which the foot of the characteristic line (denoted by $\tilde{\mathbf{X}}$) starting from the generic point \mathbf{x}_{mi} is located. Let us briefly describe the searching algorithm. The searching algorithm can be conveniently decoupled into a search along the vertical direction, which is discretized using a structured grid and does not present difficulties and a search in the xy plane that requires more care. In particular, we proceed as follows: let $\tilde{\mathbf{x}}_{mi}$ and $\tilde{\mathbf{X}}$ be the projections of the starting point, \mathbf{x}_{mi} , and of the foot of the characteristic, \mathbf{X} , on the xy plane. The point $\tilde{\mathbf{x}}_{mi}$ is the middle point of the edge ℓ_i of the 2D mesh. First of all we check if $\tilde{\mathbf{X}}$ belongs to one of the two triangles sharing the edge ℓ_i , in which case the procedure ends. Otherwise we take one of these two triangles as the starting element (we will denote it with T) and we go through the following steps

1. build the segment s joining $\tilde{\mathbf{x}}_{mi}$ and $\tilde{\mathbf{X}}$;
2. determine the edge ℓ_t of T crossed by s and find the adjoining element (T') of T sharing with T the edge ℓ_t .
3. if the element T' is owned by a different processor, set $\tilde{\mathbf{X}} = \tilde{\mathbf{x}}_{mi}$. Otherwise check if $\tilde{\mathbf{X}}$ belongs to T' , if not, iterate the procedure.

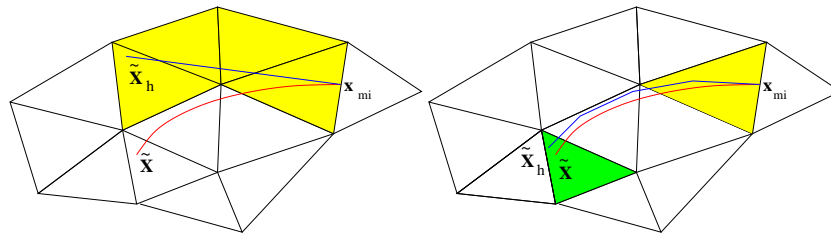


Fig. 5. Representation of the discretization of the streamlines using the Euler method: simple Euler method (left), composite Euler method (right).

3.4 The algorithm implemented in the code *Stratos*

In Fig.6 it is represented a block diagram that illustrates the phases of the computation, showing when interprocessor communications is needed.

Notice how a very limited number of interprocessor communications (red boxes) are required. Among these, we cite the solution of the linear algebraic system on the elevations (see [1999] for details) that is performed using the Aztec parallel library [1999]. The other sources of coupling are due to the computations of the variables in the sole xy plane. Let us consider for example the horizontal velocity field \mathbf{u} . Along the vertical direction piecewise continuous polynomials are adopted. This implies a coupling between the points of the same column, but since the entire column is owned

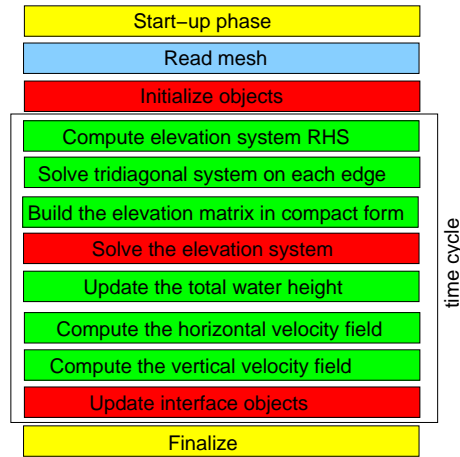


Fig. 6. Block diagram. Colors indicate the following: yellow: initialization phase; red: interprocessor communication; green: parallel work; cyan: I/O phase.

by the same processor, this process is completely *local*. The computation in the xy plane requires interprocessor communication between the edges located along interprocessor boundaries. These edges are duplicated, the flux is computed twice by two different processors and then averaged (this is due to a possible different computation of the Lagrangian derivative, see below).

4 Numerical results

To assess the properties of scalability of the code, we have performed the 3D Gaussian hill test case using up to 8 processors (the maximum number we were allowed) on a Cray T3E machine. The results indicate a speed-up factor of the code that is almost linear. The following figure show instead the computation on the Venice Lagoon. In Fig.8 (right) it is shown the water elevation. while in Fig.8 (left) it is shown the the velocity in the Venice Lagoon.

5 Conclusions

In this work we have addressed the numerical aspects and the MPI computer implementation of the Q3D-SWE model for the simulation of free surface flows in environmental problems. The presence of the free surface requires to deal with a number of issues that are not commonly present in fixed domain fluid-dynamics problems. In this sense, some original details of the implementation have been discussed. Future research will be devoted to develop and implement models able to couple free surface flows with permeable soils through which the surface water can filter. This aspect is of relevant importance, especially in presence of pollutants.

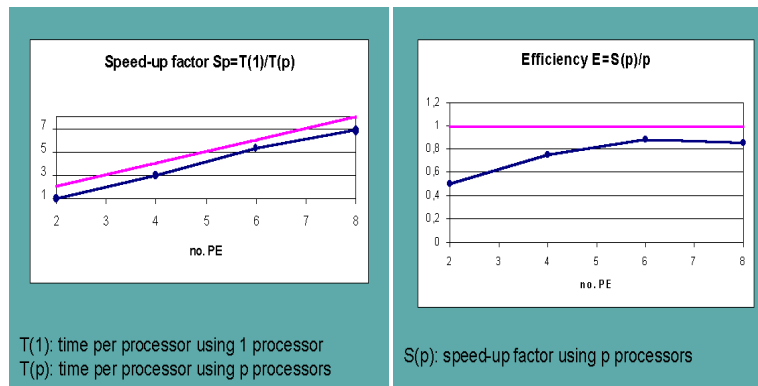


Fig. 7. Speed-up factor and efficiency for the 3D Gaussian hill test case on Cray T3E.

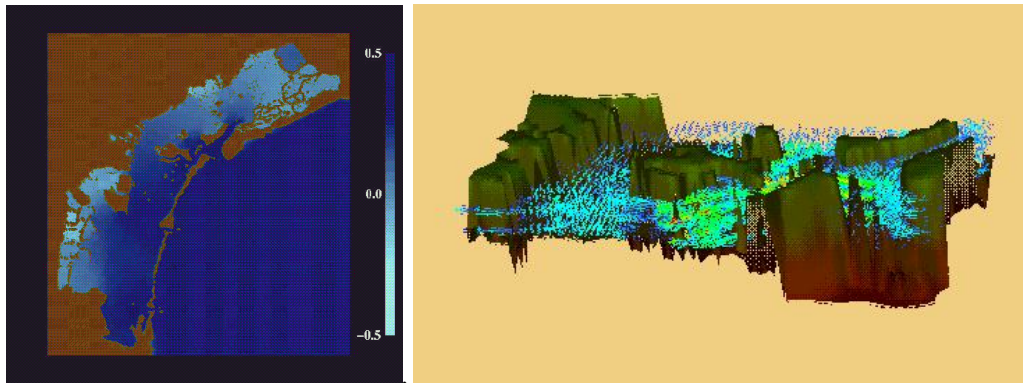


Fig. 8. Three dimensional velocity field in a region of the Venice Lagoon (right).

References

- [2002] Causin, P., Miglio, E., Saleri, F.: Algebraic factorizations for non-hydrostatic 3D free surface flows, to appear on *CVS* **5** (2), pre-print available at the web-site: <http://mox.polimi.it>.
- [1989] Chaudry, M.H.: Implicit methods for two dimensional unsteady free surface flows, **27** 321–332.
- [1999] Fontana, L., Miglio, E., Quarteroni, A., Saleri, F.: A Finite Element Method for 3D Hydrostatic Water Flows, *CVS*, Vol.2/2-3, 85–93.
- [2001] Karypis, G., Kumar, V., 'Metis, A Software Package for Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices', University of Minnesota.
- [1999] Tuminaro, R.S., Heroux, M., Hutchinson, S.A., Shadid, J.N.: Official Aztec User's Guide: Version 2.1.