h-Refined Schwarz-Krylov Solution for Hydrodynamics and Mass Transport in a PC Cluster

Ricardo Vargas Dorneles[1,2], Rogério Luis Rizzi[1,3], Tiaraju A. Diverio[1], Philippe O. A. Navaux[1].

[1] PPGC, Instituto de Informática, UFRGS
CP 15064, 91501-970, Porto Alegre, RS, Brasil.
[2] Departamento de Informática, Centro de Ciências Exatas e Tecnologia, UCS
Rua Francisco Getúlio Vargas, 1130, 95001-970, Caxias do Sul, RS, Brasil
[3] Centro de Ciências Exatas e Tecnológicas, UNIOESTE, Campus de Cascavel
Rua Universitária, 2069, 85801-110, Cascavel, PR, Brasil.
{cadinho, rizzi, diverio, navaux}@inf.ufrgs.br

Abstract—

This work presents a parallel solution, implemented in a PC cluster, using MPI library, to the simulation of hydrodynamics and mass transport in the Rio Guaiba. The governing equations of hydrodynamics and scalar transport of substances are defined in space-staggered grids, where the finer grid is nested in the coarser grid and built through interpolation. The PDEs are discretized using finite differences using upwind and centered difference techniques to generate a semi-implicit numerical scheme. To build the local subproblems a version of bissection algorithm was used to partitionate the numerical grid, and Schwartz additive method was used to build the overlapping Local solutions are obtained using Krylov subspace iterative methods.

Keywords—**Semi-implicit scheme, local refinement, grid partitioning, Schwarz-Krylov method, PC Cluster.**

I. Introduction

The motivation for this work comes from the fact that computational modelling of lakes, rivers, estuaries and seas has several practical applications such as the evaluation of the conditions for bath, navigation and water supplying.

The development of a high-resolution computer model allows the detailed simulation of both the hydrodynamics and the mass transport. The data obtained with this simulation is essential in the definition of an environmental policy. It can be used, for instance, to evaluate the impact in the environment of sewerage emission at certain points.

The contributions of this work are two. The first one is the construction, using h-method (interpolation), of a local refinement for the transport model, which is then solved, in a grid nested in the hydrodynamic grid using the Schwarz-Krylov method. The second one is the implementation of algorithms to partition the domain in rectangular subdomains.

II. Governing Equations and Boundary Conditions

The governing primitive variable equations describing constant density and free surface flows in rivers, estuarine embayments and coastal seas can be derived from the Navier-Stokes equations after turbulent averaging and under the simplifying assumption that the pressure is hydrostatic.

Furthermore, if the water body is well mixed, and the movement occurs mainly in the horizontal, and also the vertical scale is much smaller than the horizontal one, the governing equations can be the 2D integrated in vertical. The equations obtained with these approximations are the shallow water equations (SWE), which form a system of non-linear hyperbolic PDEs (partial differential equations) for an uncompressible fluid with a free surface. They aggregate the momentum equations (ME) and the continuity equation (CE) [WEI92], [RIZ00]. The governing equation for the advection-diffusion of constituent concentration, in the case of well-mixed water bodies, is mass transport equation (MTE), which can be described as in [LEE71].

For the sake of simplicity, we considered that in t=0, the velocity flow is zero in the entire domain and the initial distribution of concentration is zero everywhere $C(x,y,0)=C_0(x,y)=0$.

The lateral boundary conditions (BC) are specified as open or closed ones. The superior and inferior BCs are already aggregated to the governing PDEs. In the closed BCs the velocities (and water levels) in each face of the cells can be calculated considering that in solid boundaries the tangential velocities have the same velocity of the contour (no slip condition), and there is no penetration (no flux condition). In the hydrodynamics, in the open lateral BCs inflow and outflow, water levels or flow were defined. For transport, a concentration was specified in inflow BC and a null gradient in outflow BC.

III. Discretization and Numerical Scheme

The discretization of the PDEs must consider the domains of the spatial and temporal definitions. The spatial

discretization consists of building a grid throughout which the continuous space is approximated by a finite number of points, where the numerical values of the variables are determined. The temporal discretization of the systems of equations defines three families of methods: the explicit ones, the implicit ones and the semi-implicit ones, which combine the characteristics of the two methods.

In the construction of semi-implicit schemes, we look for a balance in the restraints of stability with the obtention of large time steps. This semi-implicit approach is largely used [LEE89], [CAS94], [GRO98], and such schemes appear when some terms that constitute the governing PDEs are approximated in an explicit form, and others in a implicit form. With this approach, we can construct systems of equations, which are linear, stable and computationally efficient. The degree of explicitness and implicitness of schemes can be controlled using the trapezoidal method [HIR92].

Furthermore, the critical terms can be conveniently approached by schemes like upwind, semi-Lagrangian (Eulerian-Lagrangian) [CHE84], TVD (total variation diminishing) [HAR83], MPDATA (multidimensional positive-definite advective transport algorithm) [SMO84], MARF (monotone area preserving flux-form advection algorithm) [BOT92], among others.

In this work, the semi-implicit numerical schemes, defined over a type C space-staggered Arakawa grid [MES98], is built such that the gradient of surface elevation in the ME and the velocity in the CE will be discretized implicitly. The Coriolis and horizontal viscosity terms in the ME will be discretized explicitly using central difference. The advective term is discretized by the upwind technique to positive and negative velocities. For accuracy and to avoid the numerical instability that appears in very shallow waters the coefficient of friction in the momentum equations will be discretized implicitly. The CE will be considered in its original conservative form, where velocities will be discretized implicitly, while the total water depth will be taken explicitly.

The resulting system is a linear system of equations symmetric and positive definite (SPD), which guarantees the existence and uniqueness of the solution [CAS90]. The unknowns of the systems of equations are the water levels and they are written as:

$$E\eta^{n+1}_{(i-1,j)} + C\eta^{n+1}_{(i,j)} + D\eta^{n+1}_{(i+1,j)} + F\eta^{n+1}_{(i,j-1)} + S\eta^{n+1}_{(i,j+1)} = f^{n}_{(i,j)} \quad (1)$$

where the coefficients in (1) can be seen in [RIZ01].

For MTE the advection is approximated by upwind scheme and the horizontal diffusion is approached by central difference, where concentration terms are discretized implicitly. The system of equations resulting from this discretization process is linear and has non-symmetrical 5-diagonal matrices. The unknowns of the systems are the concentrations and they are written as:

$$EC^{n+1}_{(i-1,j)} + CC^{n+1}_{(i,j)} + DC^{n+1}_{(i+1,j)} + FC^{n+1}_{(i,j-1)} + SC^{n+1}_{(i,j+1)} = G^{n}_{(i,j)} \quad (2)$$

where the coefficients in (2) can be seen in [RIZ01].

In this work the solution for semi-implicit numerical schemes is constructed over a type C space-staggered Arakawa grid. The subdomains generated by the partitioning algorithms are rectangular, but as in the simulation only the internal cells are considered, the subdomains can be considered irregular. Furthermore, when the boundaries are irregular, as it happens in the external boundaries of the domain, or isthmuses, islands or forks exist, the generated matrices are sparse and non-band type, what should be considered in operational models used as research tools and for simulation of realistic events. The models resulting from these choices are called, in this work, HIDRA-2D.

IV. Local Refinament: h – Method

In the solution of MTE, at least three issues must be considered. The first one is related to the fact that the contaminated subregion is, generally, much smaller than the definition domain, not being necessary a grid refined globally. The dimensions of this sub-region depend on factors such as meteorological conditions, climate, period of day, type of water (fresh or salted). The second issue is the importance of capturing details and indicating which subregions are most contaminated and suffer restrictions of water supplying. The third factor is numerical: in refined meshes, the Peclet number is such that the numerical oscillations of the solution are less important. For these reasons, the mesh of the MTE has dimensions $\Delta x=\Delta y$ varying from 50m to 25m

One alternative to compatibilize the construction of computationally efficient models with the necessity of high local resolution is the use of h-methods, defining locally refined meshes as in [ZEE93]. Local refinement is a strategy that allows the mesh to be concentrated in subregions where more activity occurs. In these sub-regions the mesh can be adjusted and refined to obtain accurate solutions, without the computational cost of a globally refined mesh. Furthermore, as the plume of contaminants can assume several configurations and directions along the time, the refined mesh must be built so that the grid accompanies the event in the time. A strategy consists of:

- defining a global IBVP (Initial and Boundary Values Problem) where the SWEs are solved providing U, V and η at each point of this mesh;

- building an interpolation scheme to U, V and η in the refined mesh from the values obtained from the global mesh;
- solving the MTE locally for these values.

The refined mesh is built through bilinear Lagrange interpolation because the solution for SWEs provides the values of U, V and η, at each point of the grid at each level of time.

These data are the input data for MTE. Thus, for regular grids, the local refinement is nested in the coarser grid and built through interpolation. It can also be observed, in figures 1 and 2, that inside the refined mesh, the cells with higher concentration of contaminants, composing the shape of the plume of contaminants. This shape is formed, basically, by the direction of the flow of water.

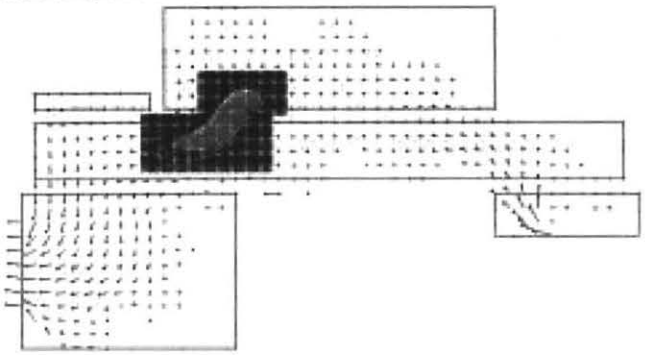

Fig. 1 Mass transport in a refined grid considering 3 subdomains after 541 cycles, corresponding to 1460 minutes of simulation

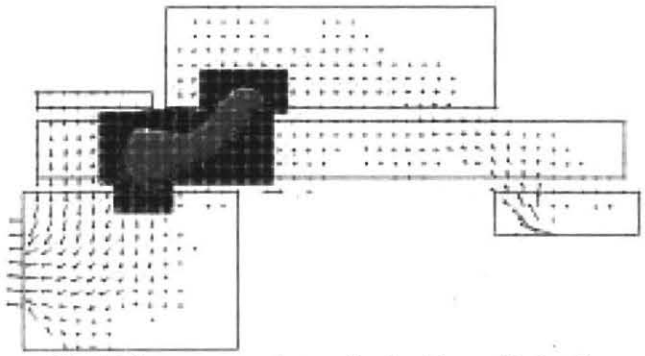

Fig. 2 Mass transport in a refined grid considering 3 subdomains after 734 cycles, corresponding to 1958 minutes of simulation.

It must be noted that the contaminant plume can vary during the simulation generating some load unbalancing. This problem is being treated through the development and implementation of dynamic load balancing algorithms.

Furthermore, the local solutions also need interpolation in time and the discrete SWE and MTE have distinct time steps, which depend on the stability conditions of the numerical schemes built.

V. SCHWARZ-KRYLOV SOLUTION METHODS

The matrices of the systems of equations, generated by the approaches to the governing equations, are sparse and large. Because of this nature, the systems are solved by Krylov subspace iterative methods. To solve the linear system $Ax = b$, the algorithms of this class seek an approximate solution x_m from an affine subspace m-dimensional $x_0 \perp K^m$, by imposing the Petrov-Galerkin $b - Ax_m \perp L^m$ condition, where L^m is another subspace of dimension m, and where $K^m(A, r_0) = span\{r_0, Ar_0, \ldots, A^{m-1}r_0\}$ is the Krylov subspace, with $r_0 = b - Ax_0$.

For the solution of systems of equations SPD or non-symmetrical, two important algorithms of this class are, respectively, the conjugate gradient (CG) and the generalized minimal residual (GMRES). The technical literature, and our own numerical experiments, have shown that these methods are efficient, robust and particularly attractive when combined with the domain decomposition methods, which are used to build the local subsolutions.

Hence, it is possible to solve the subproblems in parallel and independently using message passing libraries to explore the internodes parallelism. But in order to explore the potentialities offered by the multiprocessed nodes, i.e., the intranode parallelism, it is necessary the use of threads. We are currently implementing versions of CG and GMRES algorithms using threads, however, results obtained with the use of threads are not presented in this work.

A. The solution for the equations systems

In the solution of the SWEs the systems of equations (1) must be solved at each time step. As the matrix of coefficients is SPD, we can use the CG algorithm, an algorithm especially efficient to obtain the solution of SPD systems. The strategy of the CG algorithm is to advance, at each iteration, in the direction opposite to the gradient, so that the direction already explored is not repeated anymore. This process continues until the strict global minimum is found. The guarantee that the minimum is found is the solution of $\eta = A^{-1}b$ it elapses that A is SPD in relation to the inner product. The Preconditioned CG algorithm can be seen in [DOR00].

The solution of non-symmetrical equations systems like (2) is obtained using the Generalized Minimal Residual Method (GMRES) algorithm. This algorithm constructs the solution generating a sequence (base of the subspace) of orthogonal vectors using the Arnoldi orthogonalization

process. The vectors of this base are stored in the Hessenberg matrix, which is solved by a backward procedure after the elimination of the main diagonal through methods like Givens rotation, and a new residual is calculated with the new approximate solution found. If the convergence is not reached in a cycle, the most recent solution is used to start the next iteration.

The GMRES is a robust algorithm and obtains the approximated solution with a minimal residual norm. Its disadvantage is that the number of matrix-vector products grows linearly with the iterations and all the vectors in the Krylov subspace base must be stored, which is a problem when the dimension m of the subspace increases. One solution is to reinitialize the algorithm keeping the same dimension m of the subspace.

This strategy generates the GMRES (m) algorithm, which is not as robust as GMRES, since the convergence is not guaranteed anymore. However, if the matrix is real and positive, as it is the case in this work, the GMRES (m) converges [SAA96]. As the algorithm can converge faster or slower to a given value of m, the problem is to know the appropriate value. Silva [SIL97] suggests $5 \leq m \leq 30$ and, in our experiments, the value that better conciliates the needs of memory and convergence speed was $m = 20$.

B. Algebraic preconditioner

To speedup the convergence of Krylov subspace methods, it is necessary to preconditionate the matrices of systems equations. Thus in this work the approximations to the local inverses of the local subsystems $A_k \eta_k = f_k$, generated by the mesh partitioning algorithms, can be used to speed the solutions building the preconditioner $M_k^{-1} \approx A_k^{-1}$, where A_k^{-1} is an approximation for the inverse of A_k.

In the systems generated by equation (1), as the matrices are SPD and diagonally dominant, the polynomial approximation, described in [DOR00] with k=1, which maintains the sparseness of the original matrix, was used. With this preconditioning the number of iterations per cycle was reduced from 11 to 6, but this gain in the number of iterations was not enough to reduce the total execution time as the additional operations of the preconditioning have approximately the same cost.

Hence, it is necessary to improve the preconditioning. However, more effective preconditioners as the block-diagonal usually result in matrices which are not sparse. For this reason, up to now we have not used other preconditioner than the described above.

In the case of the systems generated by equation (2), the matrices are 5-diagonal, non-symmetrical, but well conditioned enough so the solution is obtained with in less than 3 iterations, dispensing the use of a preconditioner.

VI. Mesh Partitioning and Domain Decomposition

Parallel programming can be explicit or implicit. In the implicit one, the compiler generates code according to the degree of parallelism determined by the programmer considering the characteristics of the compiler. The explicit one requires the complete control over the strategies of implementation and over the implementation. This is the approach used here, in the solution of the PDEs, which must consider the locality of data, the load balancing, the construction of local solutions and the efficiency of the methods of solution.

The first issue is treated under the scope of mesh partitioning methods and load balancing algorithms, where it is considered the mesh partitioning such as to maximize computation and minimize communication. The second question is approached by domain decomposition methods (DDM) which couple the solutions of the subproblems generating a global solution. The third question is solved using the iterative methods of Krylov subspace, which are accelerated using a preconditioner.

A. Mesh partitioning: a graph problem

Parallelism is obtained by partitioning the problem among the many processors, but to ensure a good load balancing, it is necessary that the workload of each processor, proportional to the part of the domain processed by it, be proportional to its processing power. The efficiency of the parallel algorithm depends directly on how the numerical mesh is partitioned and on how the loads are mapped to the processors. In this work, the focus is only in the partitioning.

It must be noted that the partitioning of the numerical mesh can be seen as a problem of graph partitioning, when we consider the graph $G \equiv (V, E, w_e, w_v)$ composed by the set $V = \{0, \ldots, n-1\}$ with n nodes; the set $A \subseteq V \times V$ of edges; the edges weights w_e; and the vertices weights w_v. Hence, processes or data can be associated, to each node; communication or data dependencies to the edges; the computational load to the w_v weights; and the communication load to the w_e weights.

The problem of graph partitioning is to divide the graph in subgraphs so to minimize the number of edges between them. This problem is known as k-way partitioning, and the basic idea, in the simple case of the numerical mesh be generated by finite differences, is to pass by every effective cell of the domain (see figure 3), identifying them and associating to each one its computational load (vertices) and its data dependencies (edges). After that, a list containing the quantity of vertices and edges can be generated, and an enumeration of the relation between each pair of vertices. The problem is, then, obtaining a k-way partitioning for this

list, but this question is a NP-hard [GAR79] one and heuristic approaches are the only viable ones.

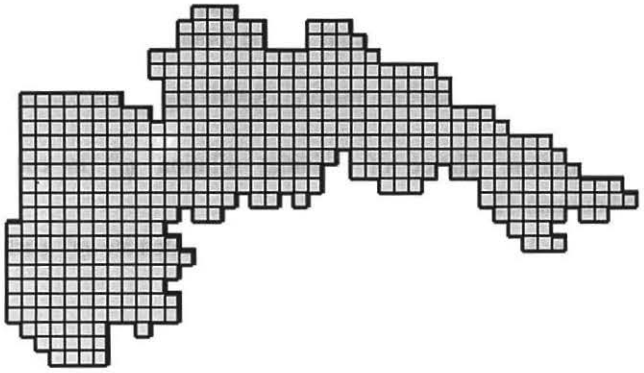

Fig. 3 Approximate configuration of Lago Guaiba, considering cells with width=height=1000m

The Lago Guaíba bathes the metropolitan region of Porto Alegre City, at the southern of Brazil. With 470 km^2 of surface, it receives the outflow of Jacuí delta, which is formed by the confluence of Jacuí, Caí, Sinos and Gravataí rivers, and flows into Lagoa dos Patos. It is about 50 km long and 15 km wide in some sections, and is situated between the 50° and 55° West parallels and 28° and 35° South latitudes [CAS 85]. The Lago Guaíba is quite important for water supplying, fluvial transport and soil irrigation for the cities on its region. However, it receives a lot of industrial and domestic contaminants.

It must be noted that the approximation (coarse) shown in figures 3 and 4 is used only to make easier the visualization. In our numerical experiments we used $\Delta x=\Delta y$ =200m to SWE and $\Delta x=\Delta y$ =25m to the MTE. With these dimensions the number of internal cells can vary from 150.000 up to 600.000, justifying the use of parallelism.

Some heuristics for the partitioning of Cartesian meshes, as the one used in this work, partitionate the mesh in any coordinate direction. Some examples are STRIP (stripwise partitioning), RCB (recursive coordinate bisection), ORB (orthogonal recursive bisection), SORB (straight orthogonal recursive bisection) [VOL97], [ROE97].

If the graph is a mesh, i.e., if it is inserted in an Euclidean space and geometrical coordinates are associated to its nodes, this extra information can be used to generate slightly better partitions. Some bissection methods by coordinates classify the nodes according to their coordinates x, y (2D problems) and then, divide the graph in these directions.

This is the approach used by the RCB method, which uses information about the coordinates of the mesh to do the bissection of the graph. The bissection algorithm is applied recursively to cut the graph in more than two parts keeping straight boundaries. Two versions of this algorithm were implemented and applied to HIDRA-2D to partition the domain graph, in a non-balanced way, in 2^k parts, with $k \in \mathbb{N}$. Figures 4 show the division of the domain for 16 processes using one version of the RCB generating partitions with straight boundaries (SRCB).

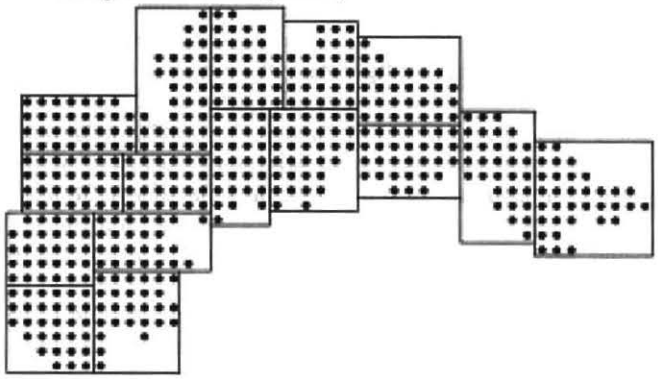

Fig. 4 Domain partitioning of the HIDRA-2D to 16 processes using the SRCB algorithm

Most methods of domain partitioning result in non-rectangular subdomains, ensuring a better balancing and less communication. However subdomains with irregular boundaries increase the complexity of managing the data exchange in the boundaries. For this reason rectangular subdomains were used in this work, generated by the SRCB algorithm.

The domain partitioning is done using a recursive coordinate bisection algorithm with straight boundaries (SRCB). In this algorithm the domain is recursively divided in two subdomains until the desired number of subdomains is reached. In each bisection, the algorithm looks for the longest axis of the domain and the cut is done orthogonal to this axis. The decision about the point where the cut will be done is based on the counting of internal cells in each subdomain generated

It must be noted that in the heuristics based in the bissection orthogonal to the longer axis, the load is considered well distributed, and the cut orthogonal to the longer axis tries to minimize the communication between neighbour subdomains. Thus, when using rectangular subdomains, where a perfect load balancing can not be assured, we considered the load distribution a more important criterion than the minimization of the communication.

When the SRCB algorithm is used, during the simulation, due to the data dependency between the subdomains, each process must use data from the cells of the boundaries of the neighbour subdomains. This data is necessary to generate the matrices corresponding to its subdomain. To keep these details updated, at each time step, each process must effectuate a data exchange with the processes keeping the neighbor subdomains. To effectuate this exchange, a structure describing which subdomains are neighbors of each processor is generated and kept. This structure is produced during the partitioning and consists of a

list of processes that maintain neighbour subdomains and the boundary cells of each one.

In our implementation, the domain partitioning is centralized in process 0, which reads from a file the description of the domain and creates a structure in memory, which describes its configuration of rows and columns, as well as the boundary conditions. The partitioning is executed using this structure and generates, for each subdomain, structures that describe it, as well as the structure of the neighbour subdomains. It must be noted that this partitioning is done using the structures that describe the domain, not being necessary the allocation of the cells of the entire domain.

After the partitioning is complete, these structures are sent to the neighbour processes. With these structures describing each subdomain, each process, including process 0, allocates memory for its entire subdomain and initializes the variables of each cell in the domain.

In relation to mass transport, in the beginning of the simulation, process 0 sends to each process the information about the contaminant emitters present in each subdomain, and the processes which contain an emitter allocate a refined mesh in the region next to the emitter. As the contaminant plume keeps growing, the refined mesh keeps also growing. Reaching the subdomain boundary, the process responsible by the neighbour subdomain receives this information and allocates a refined mesh so the contaminants transport can occur also in the neighbour subdomain and so on.

B. Domain decomposition: Schwarz additive method

The Domain Decomposition Methods (DDM) are attractive from the point of view of parallel computing, because they divide the computation in parts such that, generally, the communications are restricted to the boundaries. Some of the major motivations for the use of DDM are:

- the major use of local data. Global communication is then generally restricted to the synchronization of the solution;
- present flexibility to work with PDEs defined in regions with a complex geometry and/or which show different behaviors in different parts of the subdomain;
- allow the construction of preconditioners to speed up Krylov subspace solution methods.

The DDM used here is the Schwarz method with overlapping, which is characterized by the decomposition of the bounded open domain $\Omega \subset \mathbb{R}^{n\leq 3}$, $\overline{\Omega} \subseteq \bigcup_{k=1}^{N} \overline{\Omega}_k$ in N subdomains Ω_k overlapped, where the internal boundary conditions are denoted by $\Gamma_{ij} = \partial\Omega_i \cap \Omega_j$ (figure 5), such that the equations system $A\varphi = f$ finds its global solution φ through the partial solutions φ_k of the k subproblems $A_k\varphi_k = f_k$.

Thus, in the additive version, implemented here, all the subdomains use the solution of the last iteration as BC so that each subdomain can be solved independently, keeping the communication restricted to the boundaries [SMI96], [CHA94].

The Schwarz additive method (SAM) algorithm consists in:

Initialization: Choice of a initial solution φ_i^0 in each subdomain Ω_i;

Iteration: For $n \geq 1$, with φ_i^{n-1} known, calculate φ_i^n such that:

1. $A\varphi_i^n = f$ in Ω_i;
2. $\varphi_i^n = g$ in $\partial\Omega_i \cap \partial\Omega$;
3. $\varphi_i^n = \varphi_j^{n-1}$ in Γ_{ij} to $1 \leq j \leq N$ such that $\Omega_i \cap \Omega_j \neq \emptyset$;

until $\sup\left(\left\|\varphi_j^n - \varphi_j^{n-1}\right\|_2 / \left\|\varphi_j^{n-1}\right\|_2\right) \leq \varepsilon$, where ε is the desired precision.

Supposing that $\Omega_i \cap \Omega_j \cap \Omega_k \neq \emptyset$, $\forall i \neq j \neq k$, it can be shown that the algorithm converges. Moreover, the presence of overlapping regions ensures the continuity of the solution and its derivatives between the different subdomains [DEB98].

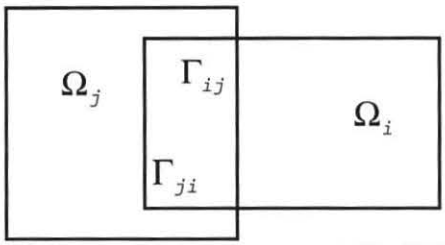

Fig. 5 Subdomains Ω_i and Ω_j generated by SAM, where Γ_{ij} and Γ_{ji} are the internal boundaries of the i- and j-adjacent subdomains

As the data dependency is restricted to the boundaries of the adjacent subdomains, the need for data exchange between the processors in each time step is restricted to these lines. Hence, due to the computational stencil used to build the semi-implicit numerical scheme, which is a 5-points one with natural ordering generating matrices, the minimal overlapping is of two rows and/or two columns and, therefore, the SAM applied to the computational mesh considered these overlapping in the subdomains obtained in the SRCB partitioning.

It should be noted that, due to the approximation considered here, where the subproblems are built considering that the meshes are regular and the subproblems are rectangular domains, for each choice of $k \in \mathbb{N}$ done in the SRCB algorithm, the set of subdomains generated provides, when executed in parallel, solutions slightly

different of the solution obtained when executed in a single processor. The difference depends only on the desired accuracy, which can be controlled varying ε.

VII. Platform and Results Obtained

The platform used in this work is the cluster available in the Instituto de Informática in UFRGS, which is multiprocessed. It is constituted by nodes interconnected by a Fast Ethernet communication network.

The model here described was developed over the Linux operating system, using C programming language and, for interprocessor communication, the MPICH 1.2.1, an implementation of MPI standard for message passing. In this library, at the beginning of the execution, it is defined the number of processes that will be created, as well as the node where each process will run. In this work we have defined the number of processes equal to the number of processors, running each process in one processor, and being assigned to each processor one subdomain.

A. Input Data

In the solution of the SWE the original computational domain was approximated by cells with the dimensions $\Delta x=\Delta y=1000$m and this grid was refined with different refinements R.

The hydrodynamics grid is refined in some parts, generating a finer grid for the MTE. The time step Δt of both numerical schemes is automatically calculated based on Courant number.

The BC used for the SWE are velocities U=(u,v) where $|U|=0.3$ m/s was used in outflow frontiers and $|U|=0.9$ m/s was used in inflow frontiers, generating the velocity field shown in figure 1. Other physical parameters as the initial level, the Coriolis force, the stress coefficients in the bottom of the water body, the wind velocity and the turbulent viscosity coefficient, among others, can be seen in [RIZ00]. The initial data and the physical parameters to the MTE are described in [DOR00].

B. Results for the partitioning of the hydrodynamics mesh

Table 1 shows some results obtained with the partitioning using the SRCB algorithm. The execution times in seconds TT (processing time + communication time + idle time) and communication times CT were measured for a number of processors #P from 1 to 8, with refinements R from 1 to 16. The original domain, without refinement, is composed by approximately 1500 cells. The most refined grid used (refinement of 16) results in a domain of about 400.000 cells. The cluster used in the test is composed of 4 nodes dual Pentium Pro 200 with 128 Mb of memory in each one. In the tests up to 4 processors, one processor per node was used. In the test with 8 processors, two processes per node were used, and the O.S. schedules one process to each processor of the node. The tests were made running 50 time steps for the hydrodynamics.

As can be seen in table 1, the processing time, for the same number of processors, is quadratically proportional to the refinement and the communication time is nearly linear with the refinement. An exceptional behavior occurred when running with a refinement of R=8, when the communication time grew more than expected.

Table 1 – Refinement × number of processors

	#P=1	#P=2		#P=4		#P=8	
	TT	TT	CT	TT	CT	TT	CT
R=1	1.5	1.1	0.4	0.8	0.4	0.9	0.7
R=2	6.2	3.3	0.6	2.1	1.3	1.7	0.9
R=4	28.7	12.6	0.9	7.1	5.4	4.9	2.0
R=8	159.1	77.3	13.7	48.3	24.8	29.5	17.7
R=16	582.2	289.2	3.3	147.5	8.7	81.1	25.4

It must be noted that, when more than one processor is used in the same node, the processors share resources as memory and cache. Although the communication between processes in the same node is expected to be faster, the contention in the resources can reduce the obtained speedup. We have made some tests to evaluate the contention. For instance, the product of two matrices 700 × 700, when executed in a single processor uses 98,4% of CPU, 10,6% of memory and results in an execution time of 32 s. When we run the same code in two processors, 98,8% of each CPU, 10,5% of memory and the execution time grows to 41,45 s.

The speedups obtained for the refinement of 16 were nearly linear with the number of processors, showing a good efficiency and scalability. Figure 6 shows the speedups obtained for a refinement of 8 and 16. The speedup obtained for 2 processors was 2.01. This superlinear speedup occurs because, when the rectangular domain is divided, each of the resulting subdomains is smaller than half the original domain and so is the processing time. The gain obtained in the processing time is greater than the communication overhead resulting in this superlinear speedup.

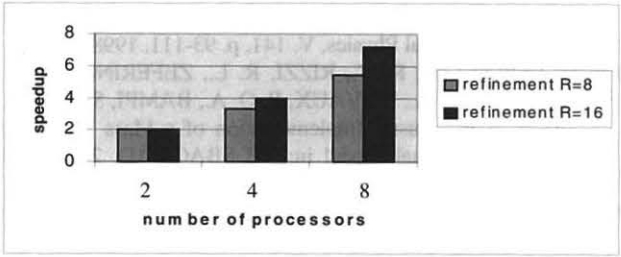

Fig. 6 Speedup x number of processors for refinements 8 and 16 using Fast Ethernet network

VIII. Conclusion and Future Works

In this work we presented a solution for the hydrodynamics and mass transport in 2D water bodies, where the mass transport mesh is built nested in the hydrodynamics mesh by an interpolation procedure. The local subproblems are built by mesh partitioning using bissection algorithms such to obtain problems with straight frontiers.

The numerical results show that the local solutions are efficiently obtained using Krylov subspace methods. However, since the hydrodynamics and mass transport simulations occur alternately, and the mass transport occurs only over a part of the domain, that is frequently changed during the simulation, a strong load unbalance occurs. Hence, the next step is to develop and implement strategies and algorithms for dynamic load balancing like diffusion or scracht-remap algorithms [DOR01].

Acknowledgments

This research was partially supported by CNPq, CAPES and FAPERGS.

References

[BOT92] BOTT, A. Monotone Flux Limitation in the Area-preserving Flux-Form Advection Algorithm. Monthly Weather Review, vol. 120, November 1992, pp. 2592-2602.

[CAS90] CASULLI, Vincenzo. Semi-Implicit Finite Difference Methods for the Two-Dimensional Shallow Water Equations. Journal of Computational Physics, v.86. p.56-74, 1990.

[CAS94] CASULLI, V., CATTANI, E., Stability, Accuracy and Efficiency of a Semi-Implicit Method for Three-Dimensional Shallow Water Flow. Computers Math. Applic., vol. 27, nr. 4, pp. 99-112, 1994.

[CHA94] CHAN, T. .F.; MATHEW, T. P. Domain Decomposition Algorithms. Acta Numerica, p.61-143. 1994.

[CHE84] CHENG, R. Eulerian-Lagrangian Solution of the Convection-Dispersion Equation in Natural Coordinates. Water Resources Research, v.20. p.944-952. July 1984.

[DEB98] DEBREU, L., BLAYO, E. On the Schwarz Alternating Method for Oceanic Models on Parallel computers. Journal of computacional Physics, V. 141, p. 93-111. 1998.

[DOR00] DORNELES, R. V., RIZZI, R. L., ZEFERINO, C. A., DIVERIO, T. A., NAVAUX, P. O. A., BAMPI, S, SUZIN, A. A. A PC cluster Implementation of a Mass Transport Two Dimensional Model in: XII SBAC-PAD. 2000, São Pedro.

[DOR01] DORNELES, R. V. Particionamento de Domínio e Balanceamento Dinâmico de Carga em Arquiteturas Heterogêneas: Aplicação a Modelos Hidrodinâmicos e de Transporte de Massa 2-D e 3-D. Proposta de Tese. PPGC-UFRGS. 2001.

[GRO98] GROSS, E. S., CASULLI, V., BONAVENTURA, L., KOSEFF, JEFFREY. A Semi-Implicit Method for Vertical transport in Multidimensional Models. Int. Journal for Numerical Methods in Fluids. Vol. 28. pp. 157-186. 1998.

[GAR79] GAREY, M. R., JOHNSON, D. S. Computer and Intractability: a guide to the theory of NP-completeness. Freeman, San Francisco, 1979.

[HAR83] HARTEN, A. High resolution schemes for hyperbolic conservation laws. Journal of Computational Physics, vol. 49. pp. 357-393. 1983.

[HIR92] HIRSCH, C., Numerical Computation of Internal and External Flows. Vol. 1: Fundamentals of Numerical discretization. John Wiley & Sons. Chichester. p. 514. 1992.

[LEE71] LEENDERTSE, J. J; GRITTON, E. C. A Water-Quality Simulation Model for Well-Mixed Estuaries and Coastal Seas: vol. II, Computation Procedures. Technical Report R-708-NYC, Santa Monica : The Rand Corp, 1971.

[LEE89] LEENDERTSE, J. J. A new approach to three-dimensional free-surface flow modeling. Technical Report R-3712-NETH/RC, Santa Monica, The Rand Corp., 1989.

[MES98] MESSINGER, F. Numerical Methods: The Arakawa Approach, Horizontal Grid. Global and Limited-Area Modeling. Camp Springs: Academic Press. 1998.

[RIZ00] RIZZI, R. L., ZEFERINO, C. A., DORNELES, R. V., NAVAUX, P. O. A., BAMPI, S., SUZIN, A. A., DIVERIO, T. A. Fluvial Flowing of Guaiba River Estuary: A Parallel Solution for the Shallow Water Equations Model in: Proceedings of the Fourth Vecpar. 2000. Part III, June 23, p. 895-896, Portugal. 2000.

[RIZ01] RIZZI, R.L. Modelo Computacional Paralelo para a Hidrodinâmica e para o Transporte de Massa 2-D e 3-D. Proposta de Tese. PPGC-UFRGS. 2001.

[ROE97] ROEST, M.R.T. Partitioning for Parallel Finite Difference Computations in Coastal Water Simulation. Delft: Technische Universiteit Delf, 1997 (Ph.D. Thesis).

[SAA96] SAAD, Y. Iterative Methods for Sparse Linear Systems. Boston, PWS Publishing Company, 447 p. 1996.

[SIL97] SILVA, R. S. et al. Iterative Local Solvers for Distributed Krylov-Schwarz Method Applied to Convection-Diffusion Problems. Computer Methods for Applied Mechanics and Engineering, Vol. 149, 353-362, 1997.

[SMI96] SMITH, Barry. et al. Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations. Cambridge University. 1996.

[SMO84] SMOLARKIEWINCZ, P. K. A Fully Multidimensional Positive Definite Advection Transport Algorithm with Small Implicit Diffusion. Journal of Computational Physics, vol. 54, pp. 325-362, 1984.

[VOL97] VOOLEBREGT, Edwin A. H. Parallel Software Development Techiques for Shallow Water Models. Delft: Technische Universiteit Delf, 1997 (Ph.D. Thesis).

[WEI92] WEIYAN, T. Shallow Water Hydrodynamics: Mathematical Theory and Numerical Solution for a Two-dimensional System of Shallow Water Equations. Water & Power Press, Beijing e Elsevier, Amsterdam, 434 p. 1992.

[ZEE93] ZEEUW, D., POWELL, K. G. An Adaptively Refined Cartesian Mesh Solver for the Euler Equations. Journal of Computational Physics, vol. 104. pp. 56-68, 1993.