

# Toward Parallel Environmental Transport Modeling on Computing Grids

Dali Wang<sup>1</sup>, J.R. Manson<sup>2</sup> and S. G. Wallis<sup>3</sup>

<sup>1</sup> The Institute for Environmental Modeling, University of Tennessee, Knoxville, 37996, USA (wang@tiem.utk.edu)

<sup>2</sup> Macaulay Land Use Research Institute, Craigiebuckler, Aberdeen AB15 8QH, UK (r.manson@macaulay.ac.uk)

<sup>3</sup> School of the Built Environment, Heriot-Watt University, Riccarton, Edinburgh EH14 4AS, UK (steve@civ.hw.ac.uk)

**Abstract.** In this paper the authors present a methodology to pursue parallel environmental transport computations on computing grids. A Natural Coordinate System (NCS) is introduced to transform the two-dimensional environmental transport problem into a series of one-dimensional advection systems coupled with transverse and longitudinal diffusion. Then appropriate numerical methods are deployed to solve the transport problem at larger timesteps. Two kinds of computing grids are used: one is a Sun 220R Enterprise Cluster and the other is a non-dedicated network of PCs. The experiments demonstrate the high scalability of the model performance. The authors argue that the combination of the usage of the NCS and suitable numerical schemes is the reason for the inherent scalability of this model.

## 1 Introduction

High performance computation are required in many environmental simulation. Traditional parallel programming requires a suitable computing platform and algorithm to achieve the speedup goal. The expense of buying and operating a dedicated parallel computer or the difficulty of access to existing parallel computers, often deter practicing engineers from this route. Grid computing, which usually at low cost, is becoming one of alternative of parallel computing to environmental engineers in design offices. Virtually all networked workstations/PCs can be utilized as low cost computing grids. However, these networks are usually heterogeneous, so that computational models and algorithms must be appropriately designed to achieve scalability on those computing environment. The main goal of this research was to develop a methodology for parallel environmental modeling on computing grids.

## 2 Model Development

Environmental transport refers to the process wherein mass (usually the mass of some pollutant of interest) is transported by a moving fluid (usually water

or air). In two-dimensions mass transport is usually presented in a Cartesian Coordinate System (CCS) as shown below,

$$\frac{\partial c}{\partial t} + \frac{\partial uc}{\partial x} + \frac{\partial vc}{\partial y} = \frac{\partial}{\partial x} \left[ E_x \frac{\partial c}{\partial x} \right] + \frac{\partial}{\partial y} \left[ E_y \frac{\partial c}{\partial y} \right], \quad (1)$$

in which  $c(x, y, t)$  is concentration of contaminants;  $u = (u(x, y), v(x, y))$  is the vector of fluid velocity in the Cartesian coordinates  $x$  and  $y$ ;  $E_x$  and  $E_y$  are the diffusion coefficients in each coordinate direction; and  $t$  is the time. Although equation (1) is typically presented in the CCS it can be transformed into a natural coordinate system (NCS) in which the principal coordinate aligns with the flow-field streamlines and the secondary coordinate is mutually orthogonal [1]. The resulting equation is:

$$\frac{\partial c}{\partial t} + \frac{\partial qc}{\partial \xi} = \frac{\partial}{\partial \xi} \left[ D_\xi \frac{\partial c}{\partial \xi} \right] + \frac{\partial}{\partial \eta} \left[ D_\eta \frac{\partial c}{\partial \eta} \right], \quad (2)$$

where  $q$  is the resultant velocity,  $q = \sqrt{u^2 + v^2}$ ,  $u(x, y)$  and  $v(x, y)$  are the velocities in the  $x$  and  $y$  directions.  $D_\xi$  is the streamwise diffusion coefficient,  $D_\eta$  is the transverse diffusion coefficient,  $\xi$  is the streamwise coordinate direction,  $\eta$  is the transverse coordinate direction and  $t$  is time.

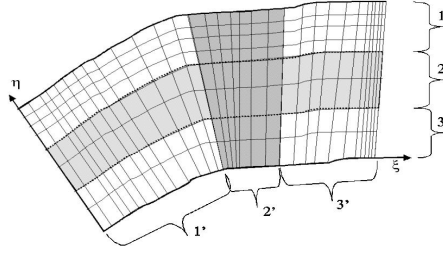
In essence, the entire computational methodology can be summarized as solving equation (2) numerically on a computational mesh based on a natural coordinate system, referred to as an NCS mesh [1]. Considering that the advection process and diffusion process are caused by different mechanism, equation (2) can be transformed into following coupled equations:

$$\frac{\partial c}{\partial t} + \frac{\partial qc}{\partial \xi} = 0, \quad (3)$$

$$\frac{\partial c}{\partial t} = \frac{\partial}{\partial \xi} \left[ D_\xi \frac{\partial c}{\partial \xi} \right], \quad (4)$$

$$\frac{\partial c}{\partial t} = \frac{\partial}{\partial \eta} \left[ D_\eta \frac{\partial c}{\partial \eta} \right]. \quad (5)$$

Equation (3) represents the pure advection process, equation (4) describes the longitudinal diffusion processes and equation (5) describes the transverse diffusion processes. The implications of this formulation for distributed computing are significant. Assume that an NCS mesh consists of  $J = 8$  streamtubes, and each streamtube consists of  $I = 30$  cells (see Fig. 1). For pure advection problems, only equation (3) need be solved. The collection of streamtubes in regions 1, 2 and 3, shown in Fig. 1, can be mapped to three different computational nodes (e.g., node 1, 2 and 3), where equation (3) is solved for each of the streamtubes independently. For advection-diffusion problems, equations (3) and (4) can be solved as before, i.e., the collection of streamtubes in regions 1, 2 and 3 on three independent computation nodes (node 1, 2 and 3). Followed that, the transverse diffusion is then solved. The collection of "diffusion-tubes" within region 1' can



**Fig. 1.** Transport computation in an NCS mesh

be mapped to node 1 for computation with regions 2' and 3' mapped to nodes 2 and 3 respectively. Therefore, equation (5) can be solved simultaneously. In this case, communications between computational nodes is inevitable. One efficient way to reduce the number of synchronizations and data exchange is to adapt large timesteps. In the model presented here, the advection term is solved by a conservative flux-based semi-Lagrangian method [2–4], and the diffusion terms are solved using either fully implicit method or the numerical method of lines.

### 3 Mesh Generation

Mesh generation is an essential and non-trivial part of numerical computation in an NCS framework. A general approach to generate an NCS mesh based on an known velocity field consists of the following steps: (1) Calculate streamlines starting at inflow boundary sections using velocity vectors. These streamlines form the longitudinal axes of an NCS; (2) Calculate cross-streamlines starting at boundary sections using the normal vectors of velocity. These cross-streamlines form the transverse axes of the NCS; (3) Find the straddle points of those streamlines and cross-streamlines, which are vertices of the mesh for the NCS; (4) Four adjacent vertices construct a cell, based on that streamtubes and cross-streamtubes can be constructed.

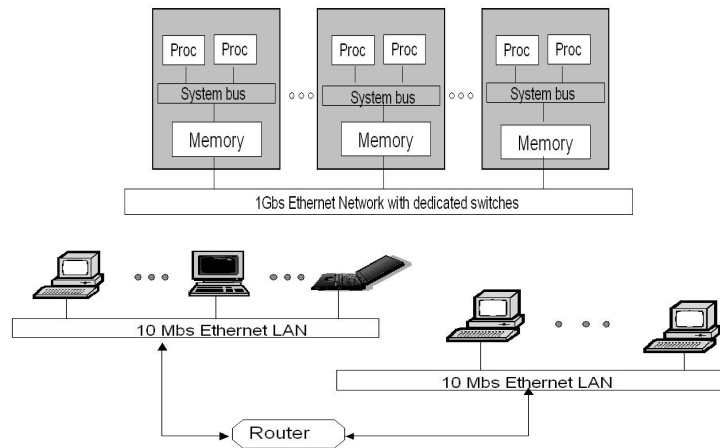
### 4 Job Scheduling

Considering the heterogeneity and reliability of nodes in a computing grid, a two-layer (server/client) communication model was adapted in this research. Since the computation is based on an NCS mesh, a dynamic job assignment algorithm was implemented to achieve better model performance. In this algorithm, the server first splits the computation into many tasks to be performed, assigns one job to each client (node), and waits for completion. When a client completes its task, the server will assign another task for it. This assignment continues until all tasks are assigned out. Although this dynamic assignment requires more communication and synchronization than static assignment, it is more flexible and suitable for heterogeneous or non-dedicated computing grids.

## 5 Computing Grid

### 5.1 Dedicated Computing Grid

A new computing infrastructure project called the Scalable Intracampus Research Grid (or SInRG) is currently being deployed at the University of Tennessee, Knoxville. This project will mirror the underlying technologies and the interdisciplinary research collaborations that are characteristic of the emerging national technology grid. SInRG's primary purpose and a more detailed description of its technology are discussed in [12]. The main building blocks of the SInRG architecture are called Grid Service Clusters (GSCs). The GSC used in this research is a Sun 220R Enterprise cluster of 16 nodes. Each node has 512 MB of memory with 4 MB L2 cache and dual Sun Ultra Sparc-II 64 bits RISC processors of 450MHz running under the Solaris 7.0 operating system. Each node in the cluster was connected via a 1 Gbits Ethernet connection. The general architecture of this GSC is shown in Fig. 2(top). In this paper, an implementation of MPI standard library, MPICH, was selected to support the message-passing communication in parallel computations.



**Fig. 2.** General Architecture of a dedicated computing grid(top) and a non-dedicated computing grid(bottom)

### 5.2 Non-Dedicated Computing Grid

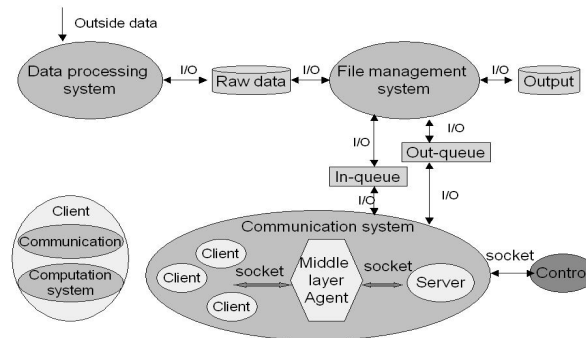
Rather than run the model over an actual non-dedicated computing grid (for which we would have had little control), the model was implemented on a simulated non-dedicated computing grid, i.e. a dedicated networked PCs was setup

to replicate a non-dedicated system by altering environmental conditions. Each PC (node) in this system had dual operating systems (Windows and Linux) and was connected to the network via a 10Mbit/s Ethernet. Those PCs belonged to two different subnets, and each had its own terminal. Fig. 2(bottom) shows the general architecture of the networked PCs.

In order to simulate the computational environment of a non-dedicated grid, we introduced two parameters to define the reliability of environmental conditions on the dedicated networked PCs. The first parameter is a network reliability coefficient (NRC), which is designed to represent the possibility of a node (client) failure within the computation system. The NRC can take any value between 0 and 1, which represents the percentage of valid connections that a server received. The second parameter is a computing efficiency coefficient (CEC), which represents the relative computational ability of each node (client). The CEC is a random number generated by a participating node (client), ranging from 0.4 to 2.5. The smaller the value, the faster is the computation performance. By introducing these parameters we could essentially configure the network to replicate a real heterogeneous, non-dedicated computing grid.

### 5.3 Distributed Communication System for Non-Dedicated Computing Grid

In this research, a distributed communication system was developed in Java to support the parallel computations on the non-dedicated grid, see Fig. 3. Structurally, the distributed communication system consists of four major sub-systems: a data processing system, a file-management system, a communication system, and a computation system. The main function of the data processing system is to collect necessary data and store them into an appropriate format according to pre-defined protocols. The file management system is a multi-thread system to create and maintain two data queues, namely, *in\_queue* and *out\_queue*, shown in Figure 3. The communication system basically consists of two parts:



**Fig. 3.** Distributed communication system for non-dedicated computing grid

a server and a client, among which data are exchanged via sockets. The computation system only exists within clients. The client only parses an incoming message, executes the task according to the underlying protocols, and sends the result back to the server. See [1] for more details.

## 6 Numerical Experiments

### 6.1 Advection-Diffusion in a Rotating Flow Field

A Gaussian distribution profile transported in a rotating flow field is a common test case in environmental transport modeling [6]. In this case, pollutant mass initially distributed in a Gaussian pattern is transported by a circular flow field. The initial concentration is a Gaussian distribution, which is described by the equation

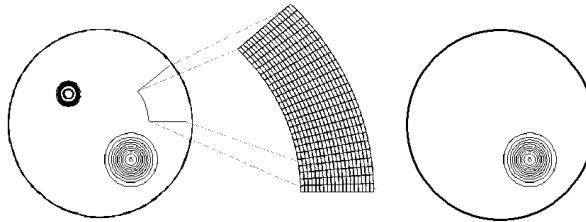
$$c_o(x, y) = \exp\left(-\frac{(x - x_c)^2 + (y - y_c)^2}{2\sigma^2}\right) \quad (6)$$

where  $(x_c, y_c)$  is the center of the Gaussian profile,  $\sigma$  is the standard deviations ( $= \sqrt{0.5}$ ). The diffusion coefficient  $D$  is given as  $10^{-2}$ . The corresponding analytical solution of this case with constant diffusion coefficient  $D$  is given by:

$$c(x, y, t) = \frac{2\sigma^2}{2\sigma^2 + 4Dt} \exp\left(-\frac{(x - x_s)^2 + (y - y_s)^2}{2\sigma^2 + 4Dt}\right) \quad (7)$$

where  $(x_s, y_s)$  is the center of Gaussian profile at Lagrangian coordinates, which changes with time. After half revolution,  $(x_s, y_s)$  is at the symmetric location of  $(x_c, y_c)$ .

The computation domain consists of 80 streamtubes. The width of each streamtube is 0.2 and the number of boxes in each streamtube is 200. Therefore there were 16000 computational grid points. The two-dimensional plots (plan view) of the initial condition, the contours of exact solution after half revolution, as well as the computation domain, are shown in Fig. 4. The simulation time of this test is fixed as 200 time units for half revolution. The timestep is set as 100 time units, and so the simulation takes two steps and the Courant



**Fig. 4.** Initial condition and exact solution (left), portion of computational domain (middle) and numerical result after two steps (right)

number reaches 50. The lateral diffusion number at the center of Gaussian pulse is about 16.0. The transverse diffusion number at the center of Gaussian pulse is about 25. In this experiment, contour plots with an interval of 0.01 are used to show model results. The numerical result (right graph of Figure 4) is very similar to the exact solution showing that the model accuracy is very high. A complete accuracy picture is presented elsewhere [1]. Since the simulation only take two steps, it is highly suitable for grid computing.

## 7 Parallel Performance

### 7.1 Model Performance on a Dedicated Computing Grid

In this series of experiments, the number of participating processors varied from 1 to 10. Speedup factors (defined as the ratio of the runtime of a serial solution to a problem to the parallel runtime) are calculated and shown in Fig. 5. The scalability is satisfied. The speedup factor using 10 processors is larger than 7. It is worth mentioning here, that by adopting an NCS mesh, the computation domain is automatically, optimally partitioned, and the computations required are linearly related to the number of steps. In these experiments, the computational time required to complete the simulation is significantly reduced by adopting large timesteps. The Courant number in the experiment reaches 50. If we used a traditional Eulerian method, such as first order upwind differencing, the computational time would be at least 50 times larger than what we achieved in this experiments.

### 7.2 Model Performance on a non-Dedicated Computing Grid

In this series of simulations, the number of participated nodes (clients) varied from 1 to 10. NRC was set as 0.8 and CEC was fixed as 1. The results of the speedup factor are shown in Fig. 5. The model scalability is high. The speedup factor in the condition of 10 processors is nearly 9. But the most impressive aspect of this computation is its super reliability: (1) the computation



Fig. 5. Model performance on computing grids

can be resumed at any point, even after all clients have been terminated; (2) the participating number of clients are only limited by the capacity of the server, theoretically any networked idle computers can join this distributed system. The total execution time on a non-dedicated grid is much larger than that on a dedicated grid, however, these disadvantages do not deter us from seeking high-performance computations on a non-dedicated computing grid because the basic idea behind this kind of approach is to reuse all idle and available CPU cycles from contributing nodes. This basic approach lies somewhere in the area between grid computing and P2P (peer to peer) computing.

## 8 Conclusions and Future Challenges

In this paper the authors describe a methodology to efficiently predict the fate of episodically released pollutants in two-dimensions. The method employs a computational mesh that aligns with the natural coordinate system of the flow field resulting in a simpler advection-diffusion system that is well suited for grid computing. The method's usefulness was demonstrated with reference to two transport problems and in both cases the present method was shown to provide excellent results. Finally the method was shown to be well-suited for implementation on both dedicated and non-dedicated computing grids.

## 9 Acknowledgement

This research used the resources of the Scalable Intracampus Research Grid (SInRG) Projects at the University of Tennessee, supported by the National Science Foundation CISE Research Infrastructure Award EIA-9972889. The networked PCs were part of the campus network at Rensselaer Polytechnic Institute.

## References

1. Wang, D.: Ph.D. Thesis, Environmental Transport Modeling in a Distributed Computing Environment. (2002) Rensselaer Polytechnic Institute
2. Manson, J.R. and Wallis, S.G.: An Accurate Numerical Algorithm for Advective Transport. *Communications in Numerical Methods in Engineering* **11** (1995), 1039-1045
3. Manson, J. R., and Wallis, S.G.: A Conservative, semi-Lagrangian Fate and Transport Model for Fluvial Systems-I. Theoretical Development. *Water Research* **34(15)** (2000) 3769-3777
4. Manson, J. R., Wallis, S.G. and Wang, D.: A Conservative, semi-Lagrangian Fate and Transport Model for Fluvial Systems-II. Numerical Testing and Practical Applications. *Water Research* **34(15)** (2000) 3778-3785
5. Innovative Computing Laboratory at the University of Tennessee, SInRG: Scalable Intracampus Research Grid, Knoxville, TN, 2002, <http://icl.cs.utk.edu/sinrg>
6. Wang, H., Dahle, H.K., Ewing, R. E., Espedal, M. S., Sharpley, R. C., Man, S.: An ELLAM scheme for Advection-Diffusion Equations in Two Dimensions. *SIAM Journal of Scientific Computation* **20(6)** (1999) 2160-2194