Coastal Ocean Modeling of the U.S. West Coast with Multiblock Grid and Dual-Level Parallelism

Phu Luong
Engineer Research and Development Center
Major Shared Resource Center
Vicksburg, MS 39180

and

Clay P. Breshears
KAI Software
A Division of Intel Americas, Inc.

and

Le N. Ly
Department of Oceanography
Naval Postgraduate School
Monterey, CA 93943

Abstract

In coastal ocean modeling, a one-block rectangular grid for a large domain has large memory requirements and long processing times. With complicated coastlines, the number of grid points used in the calculation is often the same or smaller than the number of unused grid points. These problems have been a major concern for researchers in this field.

Multiblock grid generation and dual-level parallel techniques are solutions that can overcome these problems. The Multiblock Grid Princeton Ocean Model (MGPOM) uses Message Passing Interface (MPI) to parallelize computations by assigning each grid block to a unique processor. Since not all grid blocks are of the same size, the workload between MPI processes varies. Pthreads is used to improve load balance.

Performance results from the MGPOM model on a one-block grid and a 29-block grid simulation for the U.S. west coast demonstrate the efficacy of both the MPI-Only and MPI-Pthreads code versions.

Keywords: Multiblock Grid Princeton Ocean Model, U.S. west coast simulation, coastal ocean circulation model.

1 Introduction

Over the years, the traditional one-block rectangular grid has been used for ocean circulation modeling. This technology encounters difficulty on computational grids with high resolution owing to the large memory and computing requirements. For a large body of water, such as an ocean with complicated coastlines, the number of grid points used in the calculation (water points) is often the same or even smaller than the number of unused grid points (land points).

Domain decomposition can be used to partition the traditional one-block grid into subdomains that reduce the unused grid points, and MPI [1] can be used to parallelize this type of computation [2]. Domain decomposition often requires a preprocessing step to determine the most efficient work distribution for the subdomains in order to avoid severe load imbalances. Moreover, in this technique, each domain is allowed to communicate with only one adjacent neighbor subdomain at the interface between the two. Along complicated coastlines, subdomains may be composed mostly of land-grid points that can cause load balance inequities, without special treatment in the preprocessing step.

Another approach, known as multiblock grid generation, can be used to reduce the unused grid points as well as to improve the performance of the model. This methodology allows the elimination of blocks composed mainly of land-grid points and the choice of the grid with minimum land-grid points along the coastline. In addition, high horizontal grid resolution in an area of interest can be easily handled. MPI is used for parallelization of the ocean model by assigning each grid block to a unique processor. Workload (number of used grid points) for each block can be determined during the grid generation process to achieve a more even load balance. Another advantage of the multiblock grid over the domain decomposition is that each MPI process can communicate with more than one adjacent neighbor grid block at the interfaces.

In this study, the MPI parallel version of the Multiblock Grid Princeton Ocean Model (MGPOM) [3] is used for simulation of the U.S. west coast. Pthreads is also used as a second-level parallelism within one routine of the code for improving load imbalance produced by MPI processes. The U.S. west coast has been chosen as the prototypical problem to demonstrate the effectiveness of the multiblock grid technique utilizing the dual-level parallel execution model. A brief description of grids used, one-block and multiblock grids, is presented in Section 2. Models, as well as data, used for simulation in this study are presented in Section 3. The Pthreads implementation in the routine PROFQ of the MGPOM code is described in Section 4. Performance results of the parallel codes (MPI-Only and MPI-Pthreads) on the multiblock grid are discussed in Section 5. Conclusions from this study are presented in Section 6.

2 Grids Used

The physical geographic area for this study extends from 116 West to 135 West in longitude and from 30 North to 49 North in latitude (Figure 1). The 4-min resolution grid for this area yields a total number of grid points for a one-block rectangular grid (01BLK) of 81,796 (286×286). The number of used and unused

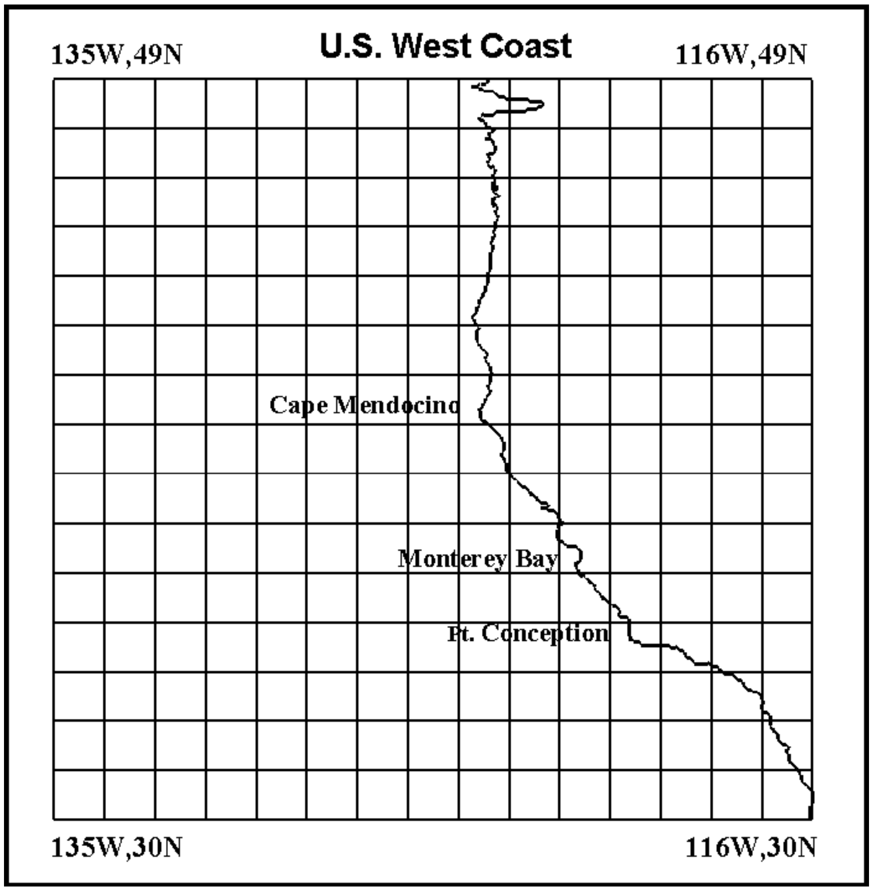

Figure 1: The U.S. west coast coastline.

grid points is 56,146 and 25,650, respectively. More than 31 percent of the total grid is unused (Figure 2).

The 29-block grid (29BLK) (Figure 3) was generated through the use of the EAGLEView interactive software package [4]. Details of this process can be found in [5]. Both the 01BLK and 29BLK grids are used for the simulation of the U.S. west coast in this study. It should be mentioned that the number of unused grid points for the 29BLK is only 4,007.

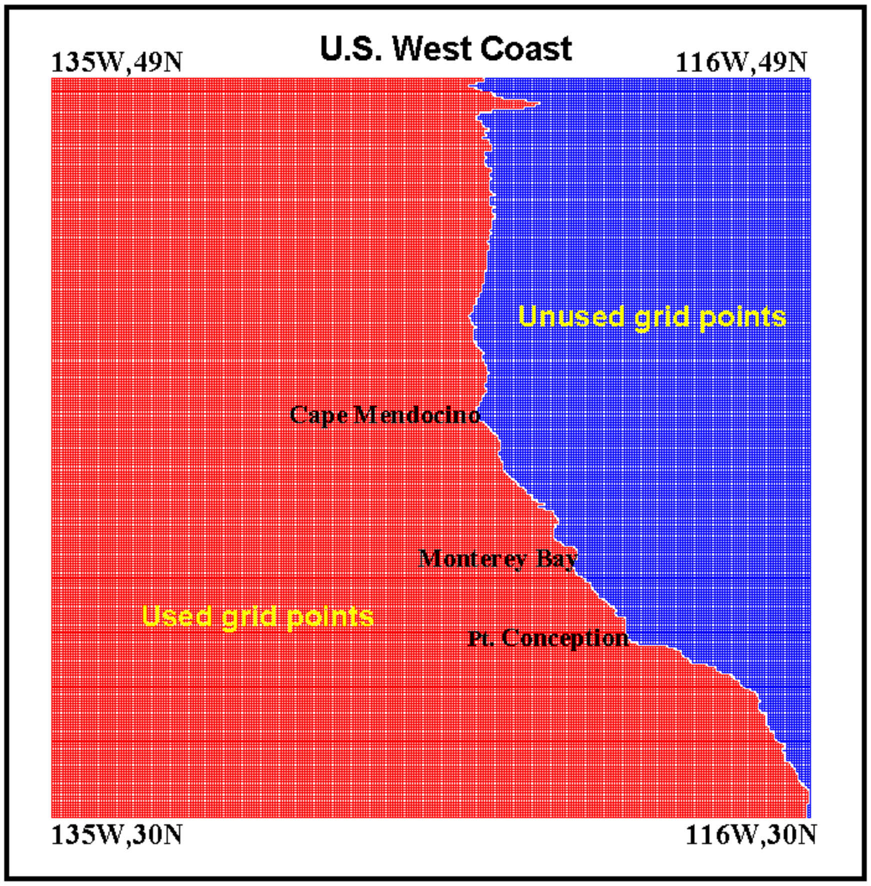

Figure 2: Used and unused grid points.

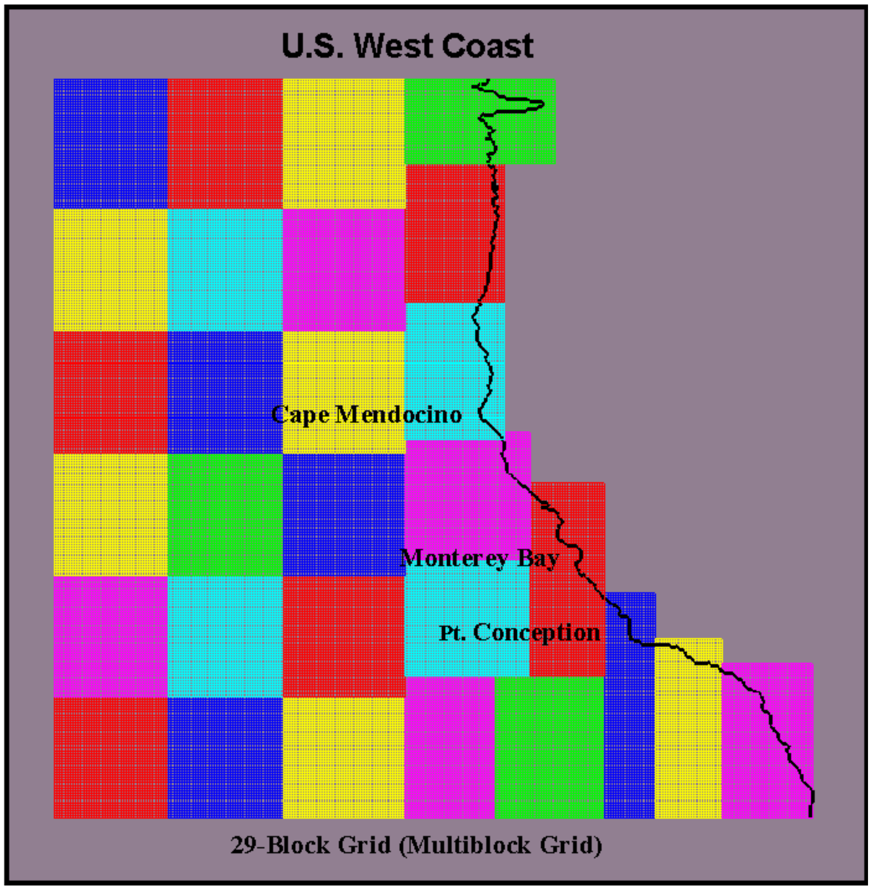

Figure 3: Coastline on the 29BLK grid.

3 The U.S. West Coast Simulation

MGPOM is a three-dimensional (3-D), primitive equations, time-dependent, σ coordinates, and free-surface coastal ocean circulation model. The primitive equations in this model describe the velocity, surface elevation, salinity, and temperature fields of the ocean. The ocean is assumed to be hydrostatic and incompressible. The model uses curvilinear orthogonal coordinates in the horizontal for the computational model grid. σ coordinates are used for the vertical direction from surface to the seafloor in which the coordinate is scaled on the water column depth. The model is embedded with a second moment turbulence closure submodel [6] to provide vertical mixing coefficients. The model has a free surface and a split time-step. The external (barotropic) mode portion of the model is two-dimensional (2-D) and uses the explicit numerical scheme for the time differencing. The internal (baroclinic) mode is 3-D and the vertical differencing is implicit. The latter eliminates time constraints for the vertical coordinate and permits the use of fine vertical grid resolution in the surface and the bottom boundary layers. More details of the model can also be found in [6].

Data for the U.S. west coast simulation were obtained from the Naval Oceanographic Office (NAVOCEANO) database. Bathymetry for the 01BLK and 29BLK grids were computed by interpolation from the 2-min resolution bathymetry database. Initial temperature and salinity for these grids were also obtained by interpolation from the 10-min resolution temperature and salinity Generalized Digital Environmental Model (GDEM) database.

The U.S. west coast computational domain has three open boundaries. At these boundaries, the internal normal velocities are governed by a Sommerfeld radiation condition. The open boundary condition for the surface elevation is zero gradient normal to the boundary. Temperature, salinity, and tangential velocities are upwinded at the open boundaries. The model is spun up for 30 days (diagnostic mode) in which the density distribution at all points on the computational grids are held fixed in time. The time-step scales used in this simulation are 40 seconds for the external (barotropic) mode and 240 seconds for the internal (baroclinic) mode. The time-step scale for the external mode is based on the CFL (Courant, Friedrichs, and Lewy) condition and the external wave speed, while the internal mode is based on the CFL condition and the internal wave speed.

After 30 days of diagnostic mode, the model is then run for 60 days. Numerical solutions for surface currents for the 01BLK grid (Figure 4) and 29BLK grid (Figure 5) yield identical results. Results of the computation for surface temperature on the 01BLK grid (Figure 6) and 29BLK grid (Figure 7) for the 90-day model output were also identical.

The serial version of MGPOM code for a 10-day simulation has an execution time of 67,000 seconds on a single IBM SP processor. A parallel version of MGPOM uses MPI asynchronous sends and receives to exchange data between adjacent blocks at the interfaces. OpenMP [7] has been used as a second level of parallelization within each MPI process in the simulation of the Arabian Gulf [8]. In this study, Pthreads is used as the second level of parallelism within each MPI process.

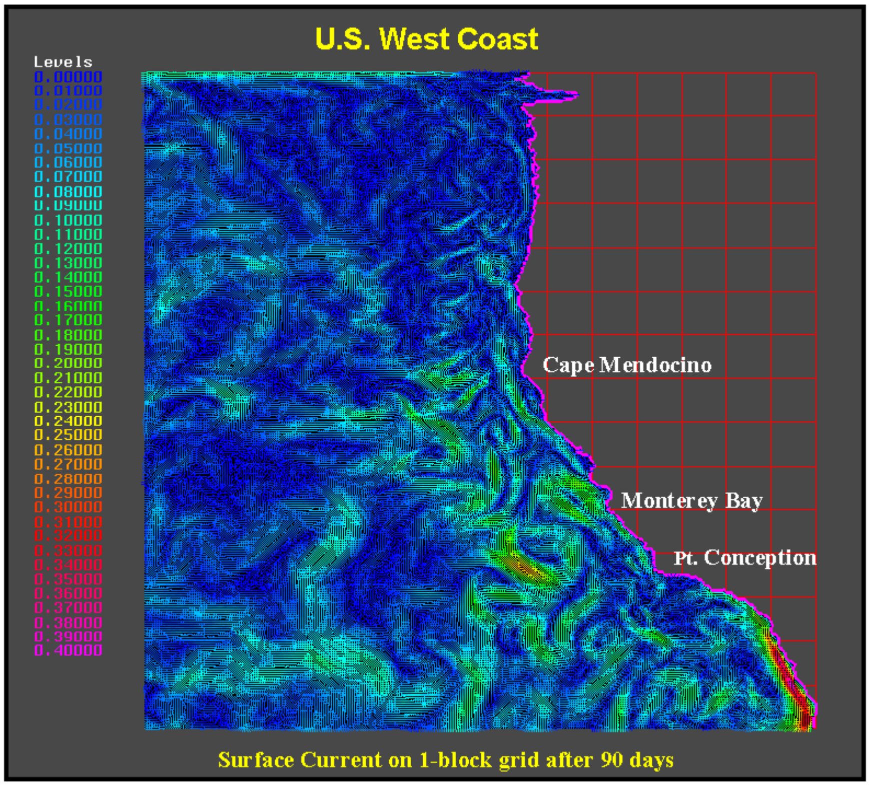

Figure 4: Surface current on the 01BLK grid.

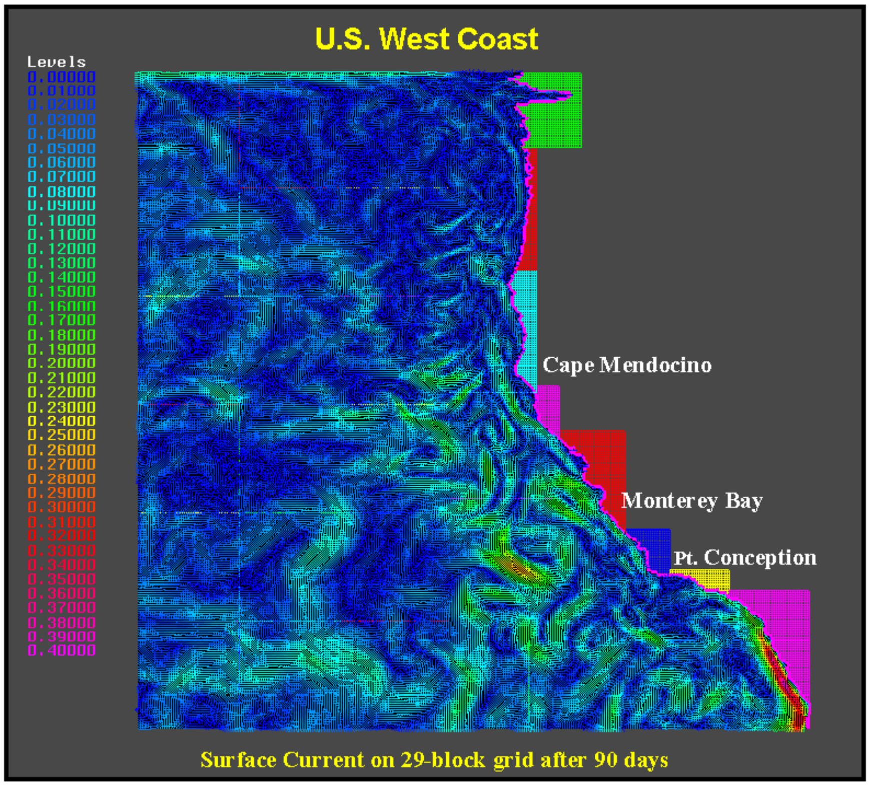

Figure 5: Surface current on the 29BLK grid.

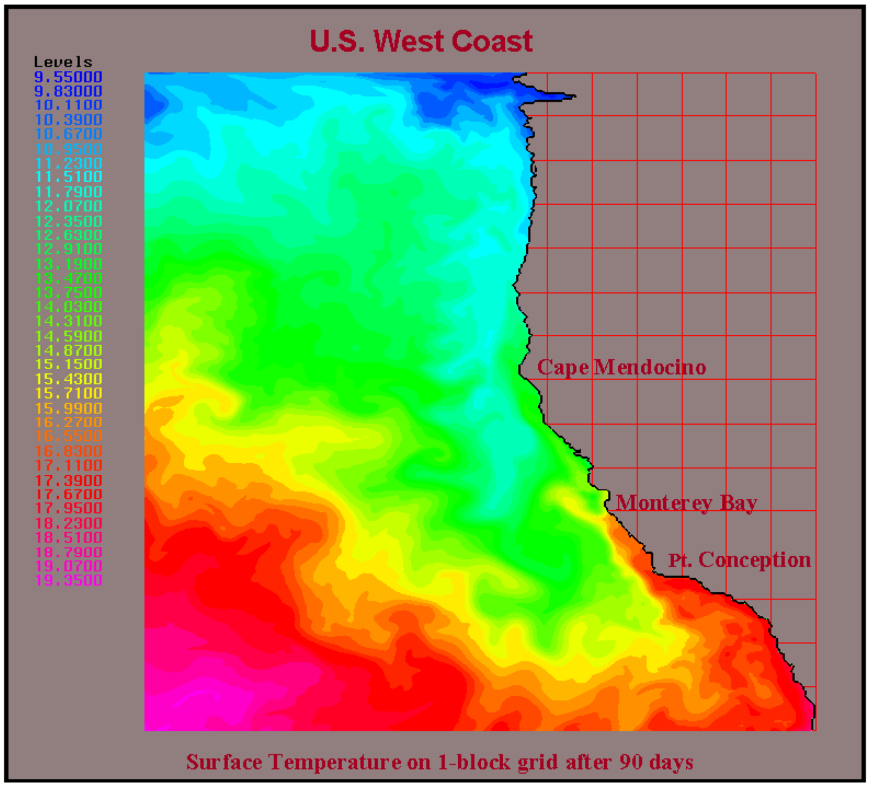

Figure 6: Surface temperature on the 01BLK grid.

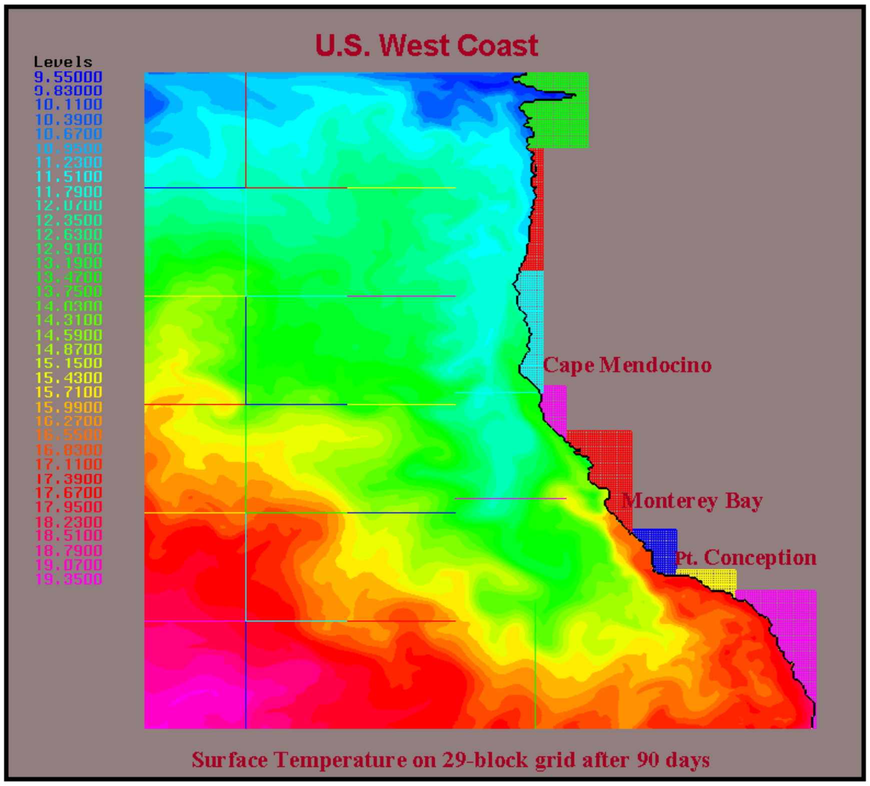

Figure 7: Surface temperature on the 29BLK grid.

4 Pthreads Implementation

Pthreads is the library of POSIX standard functions for concurrent, multithreaded programming. The POSIX standard [9] only defines an application interface (API) to the C programming language, not to Fortran. The FPTHRD package [9] consists of a Fortran module and file of C functions. The module defines Fortran-derived types, parameters, interfaces, and routines to provide Fortran programmers the capabilities of the Pthread API. The C functions provide the interface from Fortran subroutine calls and map parameters into corresponding POSIX routines and function arguments.

The structure of the MGPOM code contains one major loop within the main program. This loop iterates the time-steps of the execution simulation. Within this loop are several subroutine calls (four of these are for MPI communication, while the others perform computation). The computation subroutines are composed of multiple doubly and triply nested loops operating on 2-D and 3-D arrays. Profiling the parallel MGPOM code revealed several routines that accounted for more than half of the total execution time in each processor. The top one among those routines is PROFQ, which takes nearly 20 percent of the total execution time. This routine was the obvious first choice for threading.

Within each MPI process, based on the size of the block grid, a 2-D decomposition of the first two indices of the block into subblocks is then computed: one subblock is created per thread. The exact decomposition depends upon the number of threads assigned to the block and the relative lengths of the sides of the block. A decomposition that would yield the "squarest" subblocks is the ultimate goal. The index boundaries of the subblocks within the block grid are saved into a shared array and are used by each thread as loop iteration bounds within the PROFQ routine. After the block grid is decomposed into subblocks, the main routine is modified so that threads are created at run time according to the number of subblocks to be used within PROFQ.

5 Parallel Performance

While MGPOM processes use asynchronous communication, those processes must synchronize to some degree; e.g., processes with a small number of used grid points within assigned blocks are forced to wait on the actual receipt of data from processes with large blocks that have been performing more calculations prior to communication. The greater the difference in size between adjacent blocks, the larger the load imbalance of computation will be. In order to quantify the degree of load imbalance within a given code segment, *idle overhead* is defined as the ratio of total execution time to maximum possible execution time expressed as a percentage:

$$\%\text{idle} = 100\% \times \left(1. - \frac{\sum_{i=0}^{N-1} t_i}{N \times t_{max}}\right) \tag{1}$$

where N is the number of MPI processes, t_i is execution time of process i, and t_{max} is the largest time t_i.

The cumulative timing results of PROFQ (Figure 8) for the MPI-Only version for a run of 10 simulated days on the 29BLK shows an idle overhead of 8 percent. The total wall-clock execution time of this run was 2,018 seconds when run on 29 IBM SP processors, a 33-times speedup as compared with the 01BLK grid serial time. The cumulative timing results of PROFQ for the MPI-Pthreads version (Figure 9) yielded only 4 percent of idle overhead time. This run was performed on 29 IBM SP SMP nodes (each with four processors) using a maximum of two threads per process (58 processors) based on assigned workload. The total wall-clock execution time of this run was 1,625 seconds, a 41-times speedup over the 01BLK grid version. The performance results achieved were the result of threading a single MGPOM routine. Other candidate routines within the code are still available.

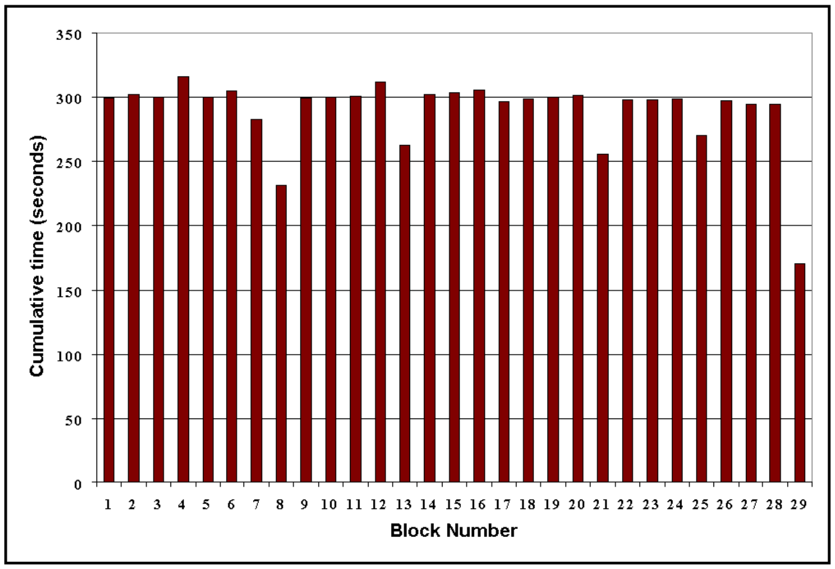

Figure 8: PROFQ (MPI-Only) cumulative time in seconds.

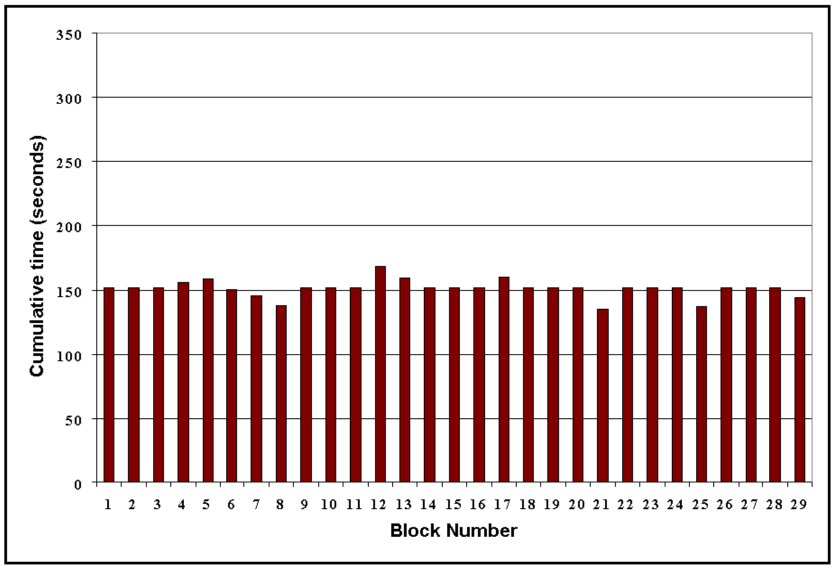

Figure 9: PROFQ (MPI-Pthreads) cumulative time in seconds.

6 Conclusions

The timing results show a significant improvement in the execution time over the MPI-Only execution. Through use of the hand-coded, dynamic threading using Pthreads, load balance between the MPI processes of the multiblock grid can be improved.

Acknowledgment

This work was funded by the DoD High Performance Computing Modernization Program (HPCMP) U.S. Army Engineer Research and Development Center (ERDC)

Major Shared Resource Center (MSRC) through Programming Environment and Training (PET).

References

[1] Snir, M., Otto, S., Huss-Lederman, S., Walker, D., and Dongarra, J., *MPI—The Complete Reference: Volume 1, the MPI Core*, MIT Press, Cambridge, 1998.

[2] Oberpriller, W.D., Sawdey, A.C., O'Keefe, M.T., and Gao, S., "Paralleling the Princeton Ocean Model using TOPAZ," `http://topaz.lcse.umn.edu`.

[3] Luong, P.V., Breshears, C.P., and Ly, L.N., "Application of Multiblock Grid and Dual-Level Parallelism in Coastal Ocean Circulation Modeling," *Journal of Applied Mathematical Modeling*, In Review.

[4] Stokes, M., Jiang, M., and Remotique, M., "EAGLEview Grid Geration Package," EAGLE-View Version 2.4 Manual. Missisippi State University/National Science Foundation Engineering Research Center for Computational Field Simulation, December 1992.

[5] Luong, P.V., Breshears, C.P., and Ly, L.N., "Comparison of Multiblock Grid and Domain Decomposition in Coastal Ocean Circulation Modeling," DoD Users Group Conference, 18-21 June, 2001, Biloxi, MS.

[6] Blumberg, A.F., and Mellor, G.L., "A Description of a Three-Dimensional Coastal Ocean Circulation Model." In *Three-Dimensional Coastal Models,* Coastal and Estuaries Sciences. Heaps, N.S., editor, AGU Geophysical Monograph Board, 1987, 1.

[7] OpenMP Architecture Review Board, "OpenMP Fortran Application Program Interface, Version 1.0," `http://www.openmp.org`, October 1997.

[8] Luong, P.V., Breshears, C.P., and Ly, L.N., "Dual-Level Parallelism and Multiblock Grids in Coastal Ocean Circulation Modeling," Technical Report ERDC MSRC/PET, TR/00-08, February 2000.

[9] Hanson, R.J., Breshears, C.P., and Gabb, H.A., "A Fortran Interface to POSIX Threads," Technical Report ERDC MSRC/PET, TR/00-18, February 2000.