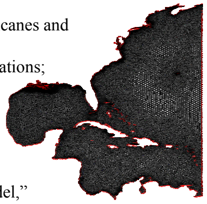
Benchmarking Studies of the Combined Predictor-Corrector, Parallel ADCIRC Code

K.M. Dresback, R.L. Kolar, J.C.Dietrich, E.P. Gilje, H. Neeman — University of Oklahoma
J.J. Westerink — University of Notre Dame
C.N. Dawson, M.F. Wheeler, V. Parr — University of Texas

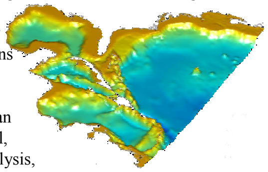
School of Civil Engineering and Environmental Science

1 History of ADCIRC

- ADCIRC (ADvanced CIRCulation) is a finite element hydrodynamic model that solves the generalized form of the wave continuity equation for elevation and the non-conservative momentum equation for the velocity field.
- The model is used for a number of applications, on a variety of basins, including:
  - Predicting surges due to hurricanes and other storms;
  - Coordinating naval fleet operations;
  - Producing tidal charts for coastal areas;
  - Studying general circulation patterns in near-shore and oceanic waters.



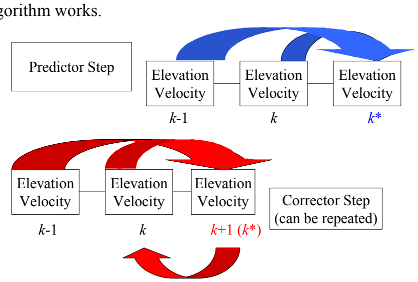
- For example the "East Coast Model," used for coastal tidal charts and storm surges, has a 30,000 node grid (right) and irregular bathymetry (below).
- The model was originally developed for the U.S. Army Corps of Engineers over 10 years ago, by R. Luettich and J.J. Westerink, and has been modified by many others since then.
- Numerical Algorithm:
  - Spatial discretization – Galerkin finite elements on linear triangles (2nd order accuracy)
  - Time discretization – sequential, finite difference
    - GWCE – . (linear free surface gradient term implicit, everything else explicit)
    - Momentum equations – u,v (U,V) (advection, diffusion explicit, matrix diagonal in horizontal, tri-diagonal in vertical)
    - 3D continuity – w
    - Transport equations – s,T,C



- Model features – modular code, spherical or Cartesian coordinates, tidal potential, "on the fly" harmonic analysis, cold and hot start, wetting and drying, "barrier" elements, grid generator/visualization integration, well documented online users manual.

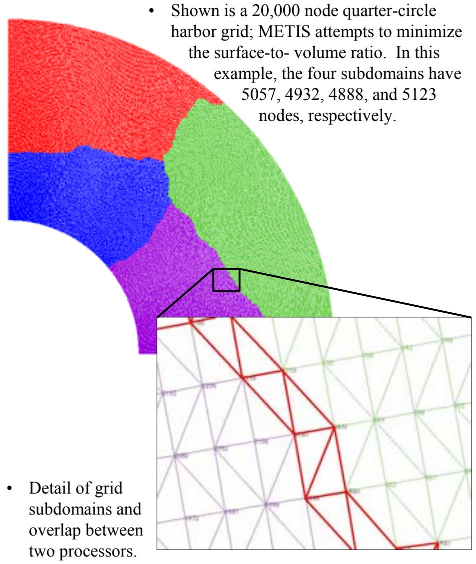
2 Predictor-Corrector Algorithm

- The original ADCIRC code was limited to a Courant number significantly less than one.
- This problem was most evident in domains with finely resolved coastlines that include inlets and barrier islands, where the nonlinear terms are most significant.
- The form of ADCIRC's time-stepping algorithm for the nonlinear terms was changed from explicit to implicit using an iterative predictor-corrector algorithm that fits within the framework of the original model.
- This new predictor-corrector algorithm was shown to improve both stability (up to a 8-fold increase) and temporal accuracy.
- Shown below is a schematic of how the predictor-corrector algorithm works.

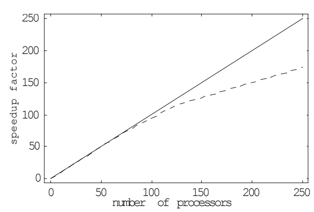


3 Parallelization Efforts

- ADCIRC uses a domain decomposition algorithm, METIS, developed at the University of Minnesota.
- METIS is based on multi-level graph partitioning, and has been proven to be more robust and produces better decompositions than alternative algorithms (e.g., space filling curves).
- METIS acts as a stand-alone library to perform grid decomposition *a priori*, then during runtime the subdomains exchange information via the MPI interface and an overlapping layer of elements (see insert below).
  - Sample of METIS grid decomposition on four processors.
  - Shown is a 20,000 node quarter-circle harbor grid; METIS attempts to minimize the surface-to- volume ratio. In this example, the four subdomains have 5057, 4932, 4888, and 5123 nodes, respectively.



- Detail of grid subdomains and overlap between two processors.
- The grid points on the interface between two subdomains (shown in red in the figure) are shared by both processors. These "ghost" nodes allow information to be shared between two processors so that a globally-continuous solution exists throughout the entire domain.
- Speed-up curves for the original ADCIRC code.



East Coast Grid
140,000 Nodes

Theoretical (solid)
Cray T3E (dot dash)
IBM SP (dash)

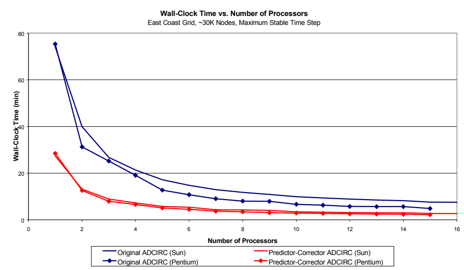
- Factors are shown for up to 32 processors on the IBM SP and 256 processors on the Cray T3E. Speedup factors are near optimal through 128 processors.

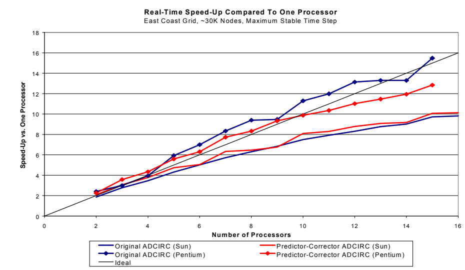
4 Benchmarking Platforms

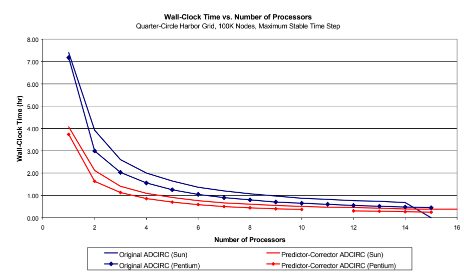
- Benchmarking was performed on two workstation clusters:

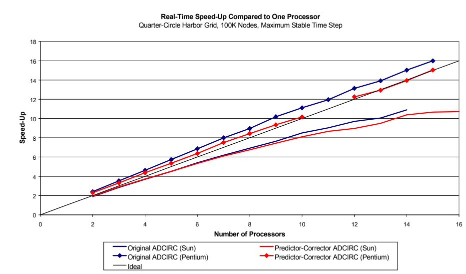
	Sun Ultra Sparc IIe	Intel Pentium III
Processors	16	16
Speed	500 MHz	1 GHz
Cache	256 KB	256 KB
Memory	128 MB	256 KB
Comm.	100 Mb/s	100 Mb/s
Compiler	Sun Forte 6.0	NAG
MPI	Sun ClusterTools	MPIch
Total Cost	$19,300	$27,500

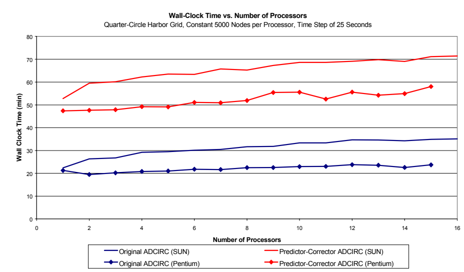
5 Benchmarking Results

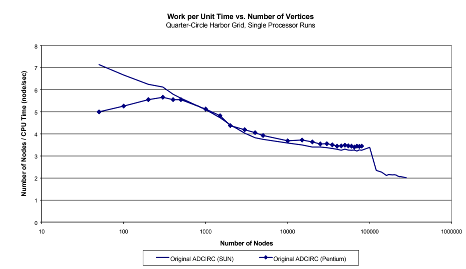












6 Closing Remarks

- The combined predictor-corrector/parallel algorithm significantly reduces simulation times.
- The predictor-corrector algorithm scales similarly to the original algorithm.
- Maximum efficiency is realized by keeping the problem size large enough to overcome communication costs but small enough to keep the problem in memory.

Acknowledgements:
- Funding from NSF ACI-9623592
- U.S. Department of Education GAANN Fellowship
- Funding from NSF EEC-9912319