Parallel Climate Data Assimilation PSAS Package Achieves 18 GFLOPS on 512-node Intel Paragon

Hong Q. Ding, Clara Chan, Donald B. Gennery, and Robert D. Ferraro
Jet Propulsion Laboratory, California Institute of Technology

1.0 Overview

An important aspect in short-term numerical weather prediction and long-term cl mate modeling is incorporating the observational data into the simulation systems. Since observational data come with various uncertainties and errors, the data are incorporated in a statistical sense, i.e., they are filtered through a Kalman filter. Because the observational points are irregularly distributed on the surface of the earth (both latitude, longitude and elevation), and they change from time to time, the filtered observations must to be interpolated to regular grids on which model systems are based.

The Physical-space Statistical Analysis System (PSAS) developed at the Data Assimilation Office (DAO) at Goddard Space Flight Center[1] is an advanced system which provides a general framework to perform the above data assimilation tasks. This software system is designated to replace the existing operational system at the DAO by 1998. Currently, a brute-force application of PSAS for a complete analysis requires about 4-7hrs on Cray C90. This falls far short of the DAO requirement of 120 re-analyses per day in real time,

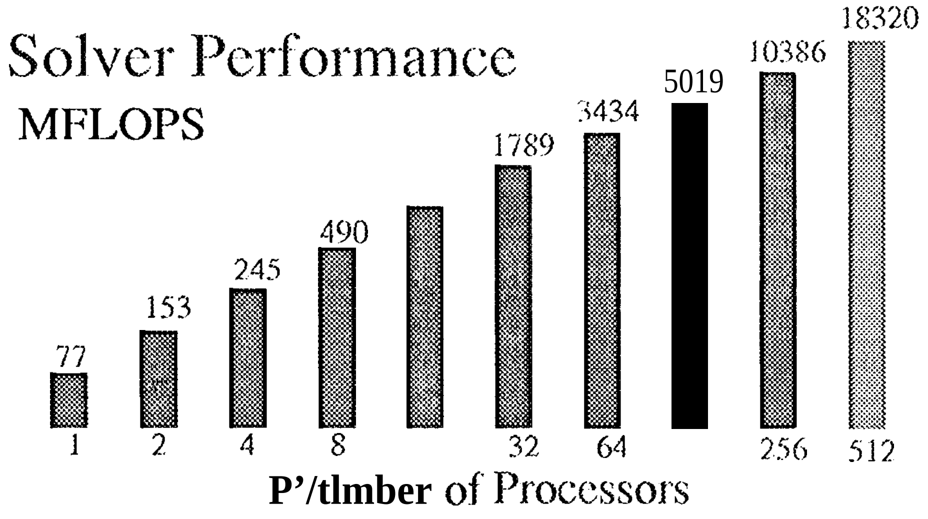

Fig. 1 Performance of the equation solver.

Recently, we have implemented several major parts of the PSAS package on the Intel Paragon. We designed and incorporated several new algorithms which are efficient and scale well to very large numbers (thousand) of processors. As a result, the parallel PSAS package solves (without interpolation, see Section 3) an 80,000 observation problem in just 1.47 minutes (including 12% of the time spent on reading data from disk) on the 512-node Paragon, in contrast, the same problem takes 152 min CPU time on the Cray C90 on the same tasks. This represents a 100-fold reduction in solution time, Thus the parallel version of PSAS now meets and substantially exceeds the time requirements and will enhance the DAO operations significantly.

The time-critical part of the package is the sparse linear equation solver, which achieves a sustained speed of 18.3 GFLOPS on a 512-node Paragon for an 85000 observation problem (see Figure 1). This represents 36% of the theoretical total peak speed (512* 100MFLOPS/node) of the 512-node Paragon. The solver achieves 77% peak speed on one node, 54% on 64 nodes, etc. These numbers indicate the high efficiency the package has achieved. The solver spends 27.5 % of the time on communications for this problem on 512-node. Most of the com - munication time is spent on sending or receiving (average) 536 messages per processor with (average) 166 floating point numbers. These percentage numbers on peak total speed and on communication indicate the highly scalable nature of the underlying algorithms.

2.0 Solving the Correlation Equation

The critical part of the current PSAS is the solution of a large linear system of equations, the correlation equation, Mx=b , with 105 unknowns. The matrix M contains the complex physical correlations calculated using a large number of existing routines, and the uncertainties of observational data, which are preprocessed in a quality control process and are stored together with other data in input data files.

The challenges of the problem lie in the size of the matrix involved. The symmetric correlation matrix M has a size of 105X105 with 26$% nonzero, due to the cutoff approximation of the correlations at 6000km on the surface. To store the entire matrix would require 10 GB (in single precision, or 20 GB in double precision) computer memory, exceeding the capacities of any existing sequential computer. This difficulty is resolved in the Cray C90 codes by recomputing the matrix on the fly, at considerable expense of CPU time.

The memory-bound problem fits well to distributed-memory parallel architecture, which cou Id have much large total residence memory. The "sparse" correlation matrix, however, does not fit to conventional sparse matrix techniques (for matrices of nonzeros typically far lower than 26%, maybe around 2% or less). This difficulty is resolved by imposing a structure to the sparse matrix: observations are divided into regions with equal numbers using a concurrent partitioned we have previousl y developed [2], and the correlation cutoff is enforced at region level (see Figure 2). This approximate on introduces a sm al l error (our rigorous com - parison indicates a 1 -2% rms error in solution), but increase the calculation speed dramatically. The imposed structure is a block structure with 74% of matrix blocks are identically zero (see Figure 3). Now the matrix-vector multiplication is carried out at 13 LAS 2 level speed, instead of the scalar speed due to the indirect indexing in conventional sparse matrix techniques. As the result, our solver runs at 77 MFLOPS on 1-node Paragon, in contrast to 3-10MFLOPS of typical scalar speed.

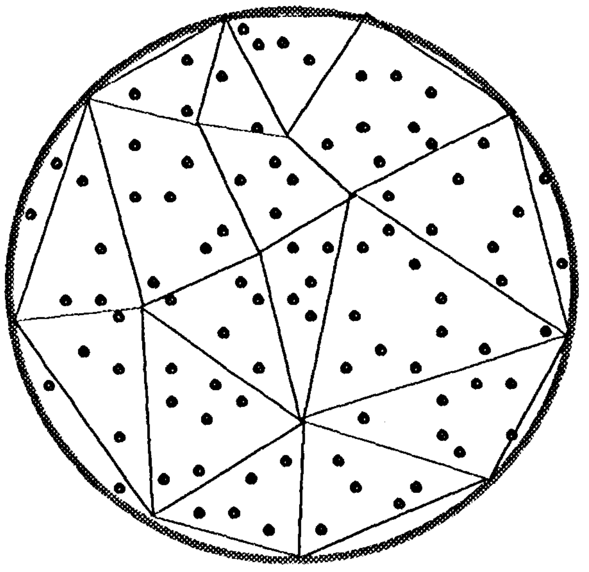

Fig.2 Regions of observations.

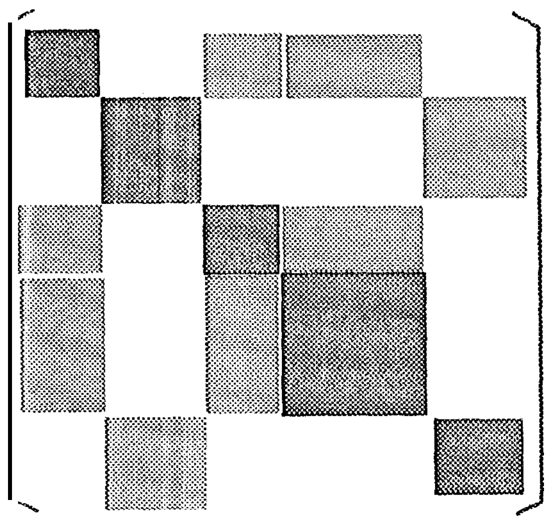

Fig.3 Block structure of the correlation mat rix. Only nonzero blocks are shown.

A large number of nonzero matrix blocks of this irregular problem must be distributed among the processors in a load-balanced way (e.g., there arc 34907 matrix blocks in the 512-node case). This is an optimization (linear programming) problem on 34907 variables with various constraints. We designed a fast iterative algorithm which finds a near-optimal distribution in just a few seconds.

Once the correlation matrix blocks are generated according to the distribution, the correlation equation is solved by a conjugate-gradient iterative solver, of which the key part is the global matrix-vector multiplication[3]. Given the imposed matrix block structure, the multiplication proceeds similar to the parallel block approach for dense matrix-vector multiplication. However, there are two important differences. First, we only store the upper-right matrix blocks due to symmetry; this allows each non-diagonal matrix block to be used twice in each matrix-vector multiplication, and therefore increases computation/com munication ratio. Second, the communication here is irregular due to the absence of 74% of matrix blocks (which would exist in dense matrix case); and storing only upper-right half matrix adds more irregularity to the communication pattern. When everything is properly implemented, this new algorithm for the "not-so-sparse" sparse matrix-vector multiplication is highly efficient and scales well to large number of processors, as indicated by the performance numbers shown above.

The partitioned, the distributor and the solver parts have been integrated and fully debugged with a system of monitoring and debugging. We have. also rigorously and systematically verified the accuracy of the solver on smaller problems where the sequential results can be obtained readily .

3.0 Interpolation

The solution of the correlation equation needs to be interpolated back to a regular 2°x2.50 grids on which the simulation models are based. The present sequential PSAS only interpolates the solution to the 500 mb level, which takes about 5% of Cray CPU time. However, the new DAO requirement of interpolating to all 14 levels represents a very significant computational task. This repetitive interpolation differs substantially from the solution of the correlation equation. Parallel implementation is in progress. We emphasize that all timing measurements presented in this paper includes every steps up to (but not including) the interpolation tasks.

Acknowledgment This work is supported by the NASA HPCC ESS project. We thank Peter Lyster, Arlindo Da Silva, Jing Guo at DAO for making the Cray C90 codes available to us and helping us in understanding the problem. Access to the 512-node Paragon at Caltech is provided by the NASA HPCC Program.

References

[1] A. Da Silva, J. Pfaendtner, J. Guo, M. Sienkiewicz, and S.E. Cohn, Assessing the Effects of Data Selection with DAO's Physical-space Statistical Analysis System, Proceedings of International Symposium on Assimilation of Observations, Tokyo, Japan, March, 1995.

[2] H.Ding and R.Ferraro, "Slices: A Scalable Concurrent Partitioner for Unstructured Finite Element Meshes" , Proceedings of SIAM 7th Conference for Parallel Processing, p.492, 1995.

[3] H.Ding and R.Ferraro, "A General Purpose Parallel Sparse Matrix Solvers Package", Proceedings of 9th International Parallel Processing Symposium, p.70, April 1995.