A scalable version of the Navy Operational Global Atmospheric Prediction System spectral forecast model

Thomas E. Rosmond
Naval Research Laboratory, Monterey, California, 7 Grace Hopper Ave, Stop 2, Monterey, CA 93943-5502, USA
Tel.: +1 408 656 4736; Fax: +1 408 656 4769;
E-mail: rosmond@nrlmry.navy.mil

The Navy Operational Global Atmospheric Prediction System (NOGAPS) includes a state-of-the-art spectral forecast model similar to models run at several major operational numerical weather prediction (NWP) centers around the world. The model, developed by the Naval Research Laboratory (NRL) in Monterey, California, has run operational at the Fleet Numerical Meteorological and Oceanographic Center (FNMOC) since 1982, and most recently is being run on a Cray C90 in a multi-tasked configuration. Typically the multi-tasked code runs on 10 to 15 processors with overall parallel efficiency of about 90%. Operational resolution is T159L30, but other operational and research applications run at significantly lower resolutions.

A scalable NOGAPS forecast model has been developed by NRL in anticipation of a FNMOC C90 replacement in about 2001, as well as for current NOGAPS research requirements to run on DOD High-Performance Computing (HPC) scalable systems. The model is designed to run with message passing (MPI). Model design criteria include bit reproducibility for different processor numbers and reasonably efficient performance on fully shared memory, distributed memory, and distributed shared memory systems for a wide range of model resolutions.

Results for a wide range of processor numbers, model resolutions, and different vendor architectures are presented. Single node performance has been disappointing on RISC based systems, at least compared to vector processor performance. This is a common complaint, and will require careful re-examination of traditional numerical weather prediction (NWP) model software design and data organization to fully exploit future scalable architectures.

1. Introduction

The Navy Operational Global Atmospheric Prediction System (NOGAPS) is the heart of the Fleet Numerical Meteorological and Oceanographic Center (FNMOC) operational NWP support to all branches of the Department of Defense. The Naval Research Laboratory (NRL) is responsible for NOGAPS design and computer implementation. NOGAPS has been operational at FNMOC since 1982 and has been through several computer system upgrades and design changes during that time. The spectral forecast model component of NOGAPS [1] is similar in formulation to global models run at other major operational NWP centers around the world. Operationally it runs multi-tasked on a Cray C90 using 10–15 processors with a sustained performance of 400 Mflops/processor. The operational resolution is currently T159L30. The 159 value is the number of Fourier coefficients carried in east-west direction on each latitude ring. The 'T' indicates a triangular truncation of the spherical harmonic spectra of the model's dependent variables.

In addition to the operational application NOGAPS is run by NRL scientists at a variety of lower resolutions for coupled atmosphere/ocean modeling research, data assimilation studies, long-term integrations, and singular vector/adjoint model research.

Price/performance considerations are driving many supercomputer applications away from expensive vector architectures and toward scalable architectures built around commodity-based components. Numerical weather prediction models such as the NOGAPS spectral forecast model is an example of such an application. FNMOC is currently planning a switch to a scalable architecture for their primary computational resource over the next 2–3 years, and NOGAPS is the most prominent application to be ported to the new system. In anticipation of a new operational platform for NOGAPS, a distributed memory NOGAPS based on mes-

Scientific Programming 8 (2000) 31–38
ISSN 1058-9244 / $8.00

sage passing (MPI) has been developed and is being tested and optimized. In part 2 of this report the design criteria and priorities of the new code are discussed. Part 3 is a brief overview of the physical processes simulated by the NOGAPS forecast model. Part 4 describes the design of the computationally intensive spherical harmonic transforms. Part 5 discusses some overall model performance issues and load balance problems. Part 6 presents some conclusions, lessons learned, and future plans. Acknowledgments are in part 7.

2. Design objectives

Because of uncertainty in the commercial marketplace for the new architectures, portability among candidate systems is a high priority in the new code. Single node performance is also being emphasized because diabatic processes dominate the computational cost of NOGAPS, and these are embarrassingly parallel. An important consideration of the new code is to retain as much of the current C90 vector code as possible, for two reasons: (a) we do not want to recode the 30–40 thousand lines of code that make up the model's diabatic processes, and (b) multiple processor shared memory "nodes" are likely to be part of many next-generation systems, and existing parallel vector codes should port gracefully to them. Specifically, this means we preserve the "long-vector" legacy of the past as much as possible, although the code has run time granularity factors that allow control of actual on-processor array sizes and loop lengths. We believe the potential performance penalties this strategy will impose on cache-based processors will be minor.

The first application of the scalable NOGAPS has been as a benchmark code for a FNMOC procurement. Therefore portability across a wide spectrum of potential platforms is essential. Message passing (MPI) is the obvious choice to maximize this portability. The proposed OpenMP standard for on-node shared memory architectures is being anticipated, but not yet implemented. The ultimate goal is to have a single code which can run as a pure MPI application on a single processor/node MPP platform, a hybrid MPI/OpenMP application on a distributed shared memory system, or as a purely shared memory application similar to the current C90 parallel/vector code. The main motivation for this is configuration management; we cannot maintain separate NOGAPS codes for three different architectures. Some overhead in computational cost and memory is inevitable with such a generalized code, but we are prepared to accept this.

3. NOGAPS description

Only a brief description of the physical parameterizations and mathematical details of the NOGAPS spectral model is presented here. Readers interested in more details should see [1]. A global spectral NWP model such as NOGAPS is composed of two major computational areas: dynamics and diabatic processes. The dynamics are the equations of motion for fluid motion on the surface of the Earth. The equations are simplified in ways that eliminate meteorologically irrelevant sound waves and certain kinds of gravity waves. In NOGAPS these equations are formulated in terms of spherical harmonic transforms in the horizontal and finite differences in the vertical. The transforms are two dimensional, requiring fast Fourier transforms (FFT's) in the east-west direction, and Legendre transforms [2] in the north-south direction. During a model integration the transforms allow conversion of the model's dependent variables (e.g., wind, temperature, moisture), between spectral "space" and grid point "space". All non-linear computations are done in the grid point space, while linear calculations and the time stepping of the model variables is done in the spectral space. Every model time step requires repeated cycling of the model's variables back and forth between the two spaces. The transforms are global, so every grid point on the Earth's surface is influenced by every other point on the surface. This has serious implications for efficient implementation of the transforms on MPP architectures because of the amount of global communication required.

The diabatic processes are all the physical processes we normally associate with the real atmosphere and weather. There are parameterization schemes for cumulus precipitation, large scale precipitation, planetary boundary layer, gravity wave drag, solar radiation, and long wave radiation. The computational cost of these processes dominates NOGAPS runtime, typically being about 65% of the total. Since the calculations are done in grid point space with only vertical dependences, they are embarrassingly parallel, requiring no communication between horizontal grid points. However, there is potential for large load imbalances unless care is taken in distributing the points in the horizontal domain among the processors. An obvious source of such imbalance is the solar radiation, since the sun is shining on only half the Earth's surface at any moment, and that subset of points is always changing. An even more serious imbalance is due to the cumulus convection, which is concentrated in the tropics, and also tends to occur during daylight hours in response to

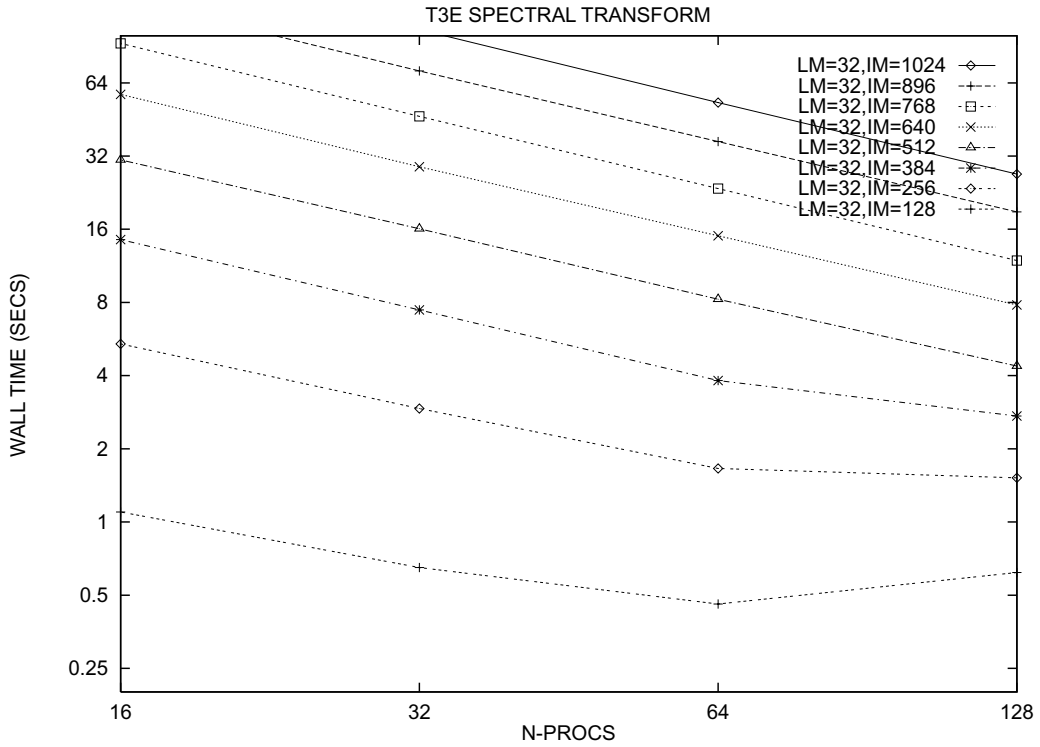

Fig. 1. Spherical harmonic transforms on the T3E, showing scaling performance for a variety of spectral resolutions and processor numbers. In the upper right 'LM' is the number of layers, 'IM' is the number of grid points around a latitude circle

solar heating. In part 5 the strategy used in NOGAPS to minimize the effects of load imbalance in the solar radiation and cumulus convection is described.

4. Spherical harmonic transforms

Other authors [3,4] have described parallel versions of the spherical harmonic transform. A common approach is the transpose method, where all communication is confined to matrix transposes that organize data so that all computation can be "on-processor", ensuring bit reproducibility of results for varying numbers of processors, a vital property for model validation and debugging. The NOGAPS transforms[1] are similar in design to those of other groups, i.e, using the transpose method, but we have coded them with several different MPI communication modes to allow performance comparisons on a variety of platforms. Specifically we have compared explicit send/recv matrix transposes using combinations of blocking, unblocking, synchronous, and non- synchronous MPI, as well as the MPI_ALLTOALLV collective function.

During the transformation of model variables back and forth between grid point and spectral space, it is important to maintain mathematical consistency. Of particular importance is alias-free representation of the quadratic non-linear terms in the dynamical equations. This is ensured if the number of grid points around a latitude circle is IM $\approx$ 3(J+1), where J is the spectral resolution. Therefore for the T159 model IM $=$ 480. Additional computational considerations dictate that the number of latitude rings be IM/2, i.e., 240 for the T159 NOGAPS. In the subsequent discussion and figures the results will frequently be presented in terms of the horizontal dimension IM and the vertical dimension LM.

An important feature of the NOGAPS transform algorithm is that the the data is organized as three-dimensional arrays. The transforms are two-dimensional in the horizontal, so the vertical dimension is passive. Transform cost is proportional to $(\mathrm{IM})^2 \ln(\mathrm{IM})$ for the FFT's and $(\mathrm{IM})^3$ for the Legendre transforms , but only linearly with LM. The presence of this third dimension is critical for computational efficiency, however,

[1] Source code available on request from *rosmond@nrlmry.navy.mil.*

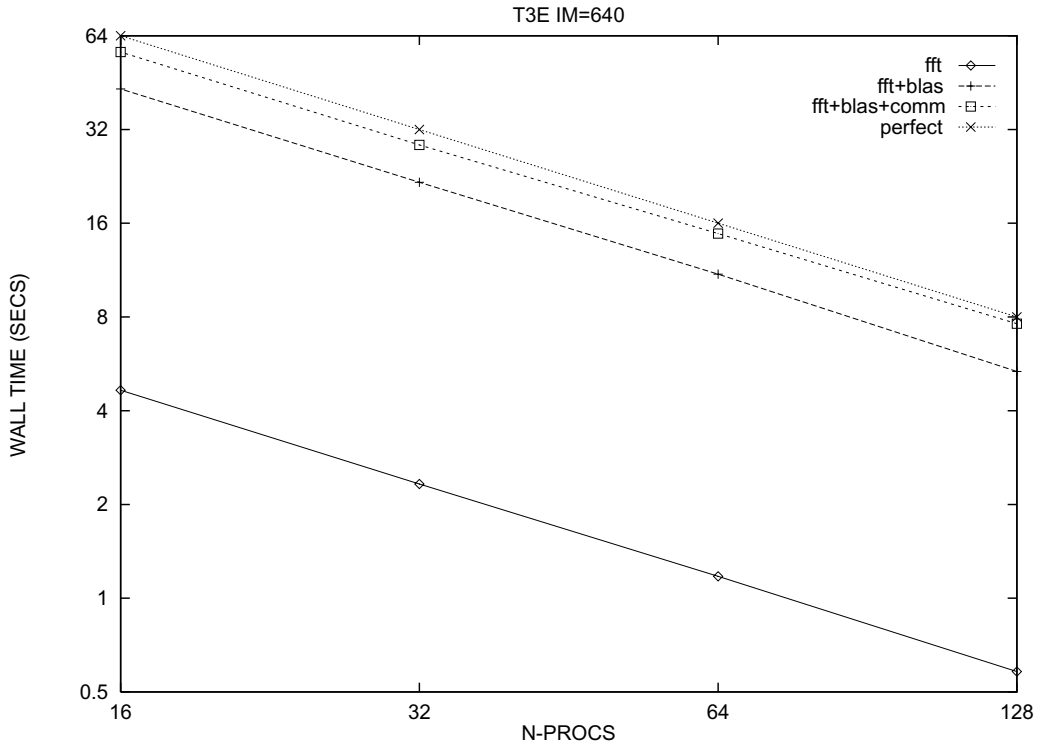

Fig. 2. FFT, Legendre transform (BLAS SGEMM), and communication times for T213L32. IM = 640 is the transform grid resolution matching the T213 spectral resolution. The "perfect" line shows hypothetically perfect scaling.

because it allows the use of BLAS 3 matrix-matrix multiply software for the Legendre transforms. Figure 1 shows some performance results for the NOGAPS transforms for a range of spectral resolutions and processor numbers on the Cray T3E.[2] Figure 2 shows a breakdown for one spectral resolution (T213, IM = 640), showing how the FFT, Legendre transform (BLAS matrix multiply), and communication scale for the same range of processor numbers. A perfect scaling reference line, showing a factor of two reduction in running time for each doubling of processor numbers, is also shown. Note that with this definition, only the slope of the perfect line is relevant. For this resolution scaling performance is excellent. Only the line including communication costs shows significant departure from parallelism with the perfect scaling line.

5. Overall model performance

Figure 3 shows, for 15, 30, 60, 120, and 240 processors, the total T3E runtime of a T159L32 NOGAPS 48hr forecast (the 'tottime' line), and a breakdown of the forecast into several components. 'Diabat' is the time taken for all the diabatic physical processes of the model, e.g., precipitation and radiation. 'Hist_writ' is the I/O time taken to output the raw model histories for this forecast period. 'P2sig' is the pre_processing time taken to interpolate gridded constant pressure surface initial fields to the model's vertical and horizontal sigma coordinate grids. This time is not included in 'tottime'. 'MPI_trans' is the time spent actually doing MPI communication in the spherical harmonic transforms. 'Dry_dyn' is the time taken to integrate the equations of motion. This includes the computational cost of FFT's, Legendre transforms (BLAS matrix multiplies), and non-linear advection terms. 'Dagnos' is the time spent computing "on-the-fly" informative diagnostics that allow quick inspection of the model's meteorological performance from printed output. 'Perfect' is a reference line, as described previously, which shows hypothetical performance for perfect scaling over this range of processor numbers.

Clearly there is a wide range of scaling performance among the various components of the NOGAPS model. 'Dry_dyn' scales the best, with essentially per-

[2] 450MHz EV5 alpha processors, UNICOSmk_2.0, CF90.3.1.0.0.

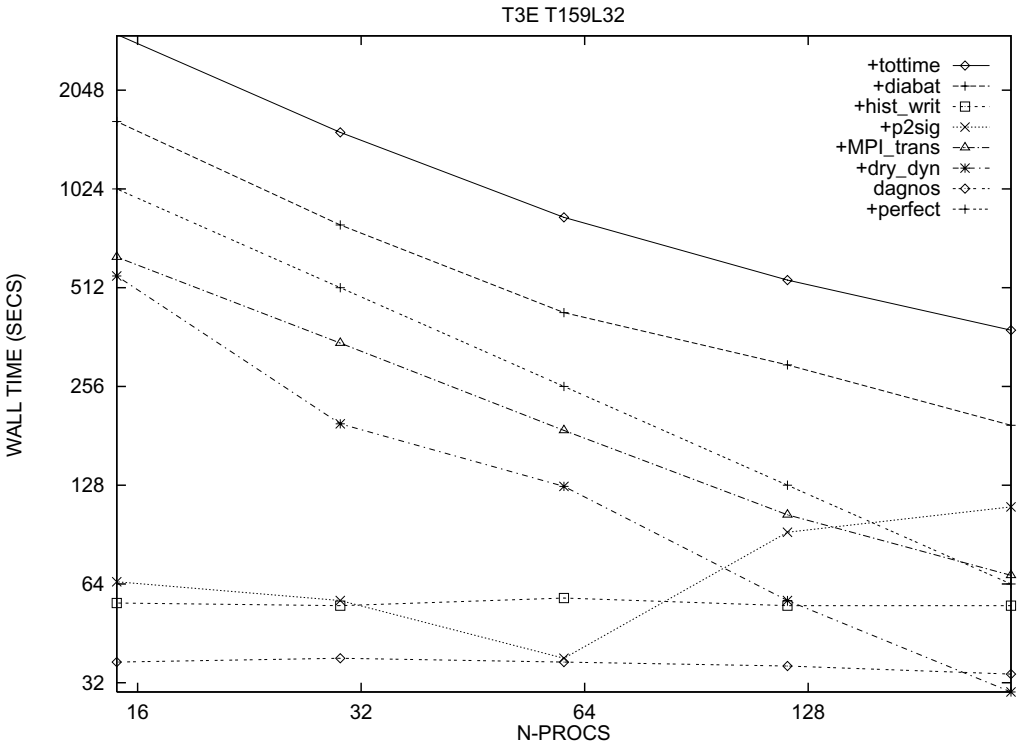

Fig. 3. T3E scaling performance for various components of T159L32 NOGAPS

fect speedup over the full range of processor numbers. The somewhat erratic behavior of 'Dry_dyn' is related to cache reuse efficiency in the BLAS matrix multiplies, which varies with on-processor granularity. 'Diabat' shows good scaling up to 60 processors, but degrades beyond. This is almost entirely due to a load imbalance in the cumulus convection, and is discussed below. The MPI communication scales reasonably well, and can probably be improved by better algorithms for the matrix transposes.

The three remaining components, 'hist_writ', 'dagnos', and 'p2sig', are conspicuous for the complete lack of scalability, with either essentially constant runtime for all processor numbers or, for 'p2sig', a disastrous increase in runtime above 60 processors. The behavior of the first two is not surprising. Because the MPI-2 I/O standard was not implemented at the time this work was done, all the I/O in the NOGAPS model is done by MPI collective gathering of data onto a single processor for file output, or single processor file reads and MPI collective scattering of data for input. This almost guarantees serial performance. Likewise 'dagnos' involves MPI collective gathering of global diagnostic data onto a single processor, where FORTRAN printed output is performed. However, neither of these two components is a cause for great concern. An efficient implementation of the MPI-2 I/O standard should eventually solve the I/O bottleneck, and the diagnostic output is a convenient luxury that can be turned off for operational NOGAPS runs.

The performance of 'p2sig', however, is more troubling. Every run of NOGAPS, regardless of forecast length, must start with a pre-processing step on a standard set of initial meteorological fields. For historical reasons, the vertical and horizontal interpolations of the pre-processing are done with hand coded cubic spline routines that are highly vectorized and multi-tasked for parallel vector architectures, e.g. Cray C90, and rely heavily on a global shared memory. The MPI implementation of these codes is very communication intensive, and most of the communication is in the form of MPI collective gathers and scatters. Certainly alternative interpolation algorithms, which are much more local in nature than cubic splines, are possible, but they are significantly less accurate and therefore not very attractive. A more likely solution will be to relegate the 'p2sig' step to a nearby shared memory system such as an individual multi-processor node of a large multi-node system.

Figure 4 shows the distribution of load of the cumulus convection for (a) 60 and (b) 120 processors. The

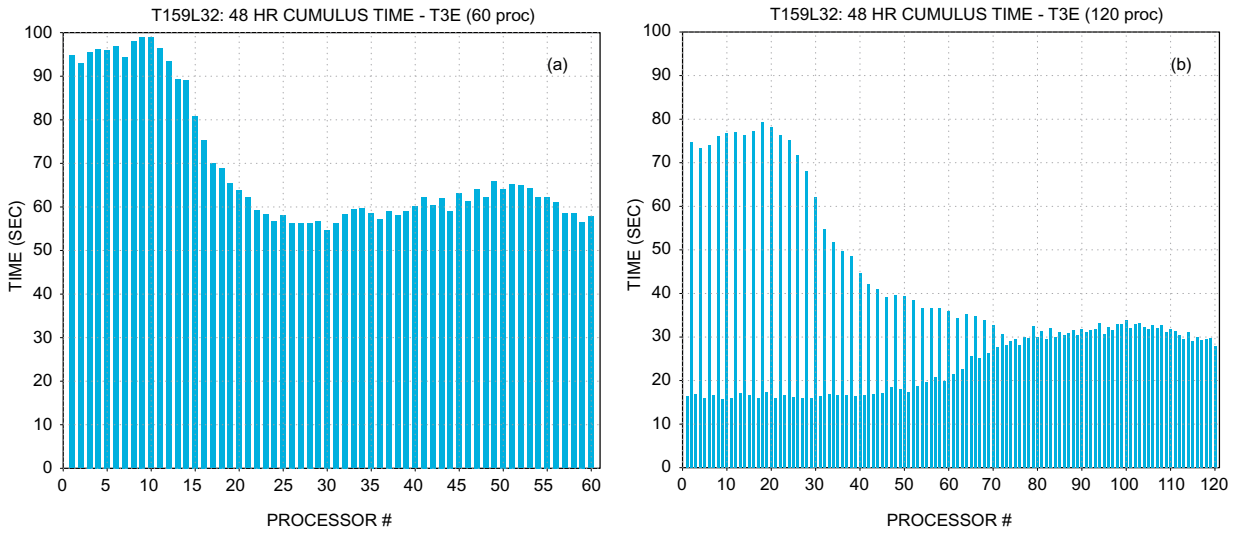

Fig. 4. Cumulus convection load balance: (a) 60 proc, (b) 120 proc. In (a) each column of the graph is the wall time used by one of the 60 processors, where each processor contains 4 latitude rings of data. The longer columns near the left side of (a) are the result of combining two mirror image latitudes rings near the equator with two mirror image latitudes near the poles. The shorter columns near the right side contain two northern hemisphere mid-latitude rings with the mirror image set from the southern hemisphere. In (b), with 120 latitudes, only two latitude rings fit on a processor. Near the left side of (b) each long column represents two tropical latitudes and each short column two polar latitudes. Only in the mid-latitudes is there reasonable load balance.

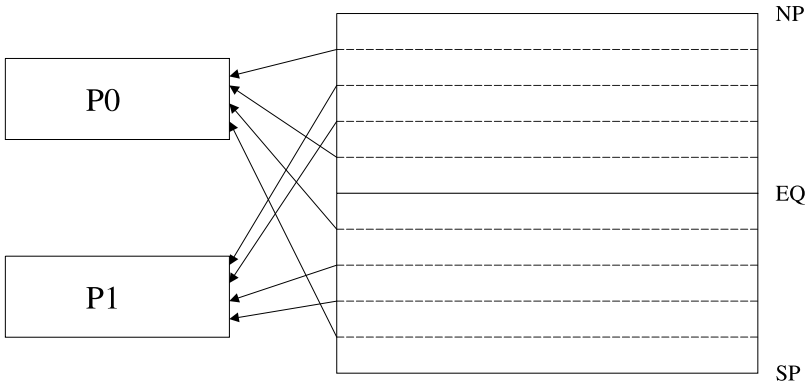

Fig. 5. MPI interleaving of NOGAPS latitudes to satisfy symmetry requirements of Legendre transforms and cumulus load balancing. NP = north pole, SP = south pole, EQ = equator. P0 would correspond to a column near the left side of Fig 4(a), P1 to a column near the right side.

transform grid for the T159 NOGAPS has 240 Gaussian latitudes, each with 480 longitude grid points around the Earth. Efficient implementation of the Legendre transforms requires that each processor always carry sets of mirror image latitudes about the equator, i.e., latitude X is paired with latitude $-X$, etc. Tropical latitudes dominate the cumulus convection workload, so the best load balance strategy is to combine a pair of tropical latitudes with a high latitude pair where the convection load is quite small. A schematic example of the strategy for an 8 latitude ring case is shown in Fig. 5. Notice that for successful application of this strategy the number of latitudes must be at least 4 times the number of processors. This criteria is satisfied for the 60 processor case. For the 120 processor case, however, only the Legendre transform criteria can be satisfied and serious load imbalances are inevitable. Notice that the maximum wall time for the 60 processor case is just under 100 secs, while for the 120 processor case it is about 80 secs, only a 20% improvement. The solution is to partition the transform grid across processors in the longitude direction as well, so a minimum of 4 partial latitudes are always on-processor for cumulus convection load balance. This requires additional communication to restore full length latitudes rings for

Table 1
T21L18 NOGAPS single node performance (24 hour fcst, time in secs)

	ALPHA 300 SMP	ALPHA 450 T3E	O2000	Cray C90
Total time	106.4	192.8	115.6	32.0
Transforms	9.7	25.6	12.0	3.6
Diabatic	91.4	162.0	100.1	26.2
Cumulus	9.4	12.4	13.2	3.9
Longwave rad	27.2	56.4	32.7	6.7
Solar rad	23.4	30.0	22.5	5.4

on-processor FFT's in the spectral transforms, and has not yet been implemented in the NOGAPS code.

Table 1 shows single node performance for a quite small (T21) NOGAPS, chosen to fit on the relatively small T3E on-node memory (128MB). The model is essentially the current operational code, highly vectorized for the C90, although at this resolution the average vector lengths are quite short. In the table the total time is followed by the spherical harmonic transform time and diabatic processes time. Cumulus convection and radiation dominate the cost of the diabatic processes, as the last three lines of the table show. Otherwise, perhaps the most conspicuous result is the relatively poor performance of the T3E relative to the DEC ALPHA 8400 SMP and O2000. In spite of a significantly faster processor speed, the smaller cache of the T3E ALPHA processor[3] gives substantially poorer performance than for the DEC 8400[4] and the SGI Origin 2000.[5]

6. Summary

We have begun the process of converting a large operational NWP model code, optimized for a parallel vector architecture, to a yet to be determined scalable architecture. The code is as general as possible to ensure reasonably graceful porting to a variety of candidate architectures and programming models. Some inefficiencies are inevitable with this philosophy, but if we understand the reasons for these problems, we believe future refinements of the model will eliminate them. An important point to be made is that no effort has yet been made to redesign the model for more optimum performance on a distributed memory, cache-based microprocessor system. Specifically, the model carries its time level histories in spectral space, rather than Gaussian grid point space. This conserves memory, but generally requires more transform operations per time step. On shared memory vector platforms such as the C90 this is an attractive design strategy, since transforms are relatively cheap, but on distributed memory architectures there is a considerable penalty in both computational and communication cost with this approach. The model also preserves a rich complement of in-line global diagnostics which are critical for monitoring meteorological performance but would be an expensive luxury in a scalable production code.

One of the greatest challenges of moving to these scalable architectures is accepting the fact that these systems are not all-purpose, and many of the "whistles and bells" that our models now contain will have to be removed, or at least made optional. This has potentially important impacts on the user-friendliness of many models which are often run by relatively naive users. NOGAPS is such a model, and we cannot ignore the implications on software design and configuration management.

[3] 450 MHz EV5, 96KB L2 cache
[4] 300 MHz EV5, 8MB L2 cache
[5] 195MHz R10000, 4MB L2 cache

Acknowledgments

This work was sponsored by the Office of Naval Research under program element 0602435N, project number 035-71, and the Department of Defense Common HPC Software Support Initiative under program element 0603704N. Most computations were performed at the Naval Oceanographic Office HPC Major Shared Resource Center, Stennis Space Center, MS.

References

[1] Hogan, T.F. and T.E. Rosmond, The description of the Navy Global Operational Prediction System's spectral forecast model, *Mon. Wea. Rev.* **119** (1991), 1786–1815.

[2] Abramowitz M. and I. Stegun, *Handbook of Mathematical Functions,* Dover Publications, Inc., New York., 1970, pp. 1046.

[3] Foster, I. and P. Worley, *Parallelizing the spectral transform method: A comparison of alternative parallel algorithms,* Proceedings of the 1994 Scalable High Performance Computing Conference, Knoxville, TN, May 23–25, IEEE, Los Alamitos, Ca., 1993, pp. 726–733.

[4] Barros, S.R.M., D. Dent, L. Isaksen and G. Robinson, The IFS Model – Overview and parallel strategies, in: *Coming of Age: Proceedings of the Sixth ECMWF Workshop on the Use of Parallel Processors in Meteorology,* Hoffman, G.-R. and N. Kreitz, eds., World Scientific, Singapore, 1995, pp. 303–318.