The Regional Atmospheric Modeling System (RAMS): Development for Parallel Processing Computer Architectures

Craig J. Tremback
*Mission Research Corporation / *ASTER Division*
Fort Collins, Colorado

Robert L. Walko
Department of Atmospheric Science, Colorado State University
and
*Mission Research Corporation / *ASTER Division*
Fort Collins, Colorado

1. Introduction

This paper describes the development of the parallel processing version of the Regional Atmospheric Modeling System (RAMS), developed at Colorado State University and MRC/*ASTER. RAMS is a multipurpose, numerical prediction model designed to simulate atmospheric circulations spanning in scale from the hemisphere down to large eddy simulations (LES) of the planetary boundary layer. Its most frequent applications are to simulate atmospheric phenomena on the mesoscale (horizontal scales from 2 km to 2000 km) for purposes ranging from operational weather forecasting to air quality regulatory applications to support of basic research. The predecessor codes of RAMS were developed initially to perform research into the areas of modeling physiographically-driven weather systems and simulating convective clouds, mesoscale convective systems, cirrus clouds, and precipitating weather systems in general. Since the early versions, the use of RAMS has increased greatly with more than 100 RAMS installations in more than 30 different countries.

2. RAMS Development History

The RAMS concept was born in the early 1980's at Colorado State University. There existed at the Department of Atmospheric Science three models that had a great deal of overlap, the CSU cloud/mesoscale mode (Tripoli and Cotton, 1982), a hydrostatic version of the cloud model (Tremback, 1990), and the sea breeze model described by Mahrer and Pielke (1977). The sea breeze model and the cloud/mesoscale model had development histories dating back to the early 1970s. It was decided to merge these three models into a unified system, and the cloud model and the hydrostatic version were merged in 1983, forming the first version of RAMS.

The original RAMS and its predecessors had been executed exclusively on the NCAR CRAY-1 machine. Because of the size of the central memory on that machine (1 Mw), there were various deign constructs that were necessary (such as disk I/O) that were required to run what we would consider today to be small runs. As the 1980's progressed and computers were built with significantly more memory, some of the design constructs became obsolete and stood in the way of efficient programming. Therefore, in 1986, a re-write of the entire RAMS code was started to remove the obsolete features and also to include a few of the parameterizations from the sea breeze model. The first version of the "new" RAMS was released in 1988 as version 0a. The first widely distributed version was 2c which was released in 1991.

The development of a parallel version began in 1991 at CSU. MPI did not exist at the time, so PVM was used for the message passing package. The development continued slowly (about ¼ full-time equivalent). A reasonably complete version was finished in 1994, with a substantial number of modifications and improvements carried out in 1995 and 1996, including support for MPI. A prototype operational version of the parallel RAMS was installed at Kennedy Space Center in late 1995.

3. Summary of RAMS Options

RAMS contains a number of options which makes it amenable for use in a wide range of applications. It is designed so that the code contains a variety of structures and features ranging from hydrostatic to non-hydrostatic codes, resolution ranging from less than a meter to the order of a hundred kilometers, domains from a few kilometers to an entire hemisphere, and a suite of physical options. This allows for an easy selection of the appropriate options for a different spatial scale or different locations. RAMS is well suited for parallelization since it does not use physical/numerical routines that are global. Pressure, for example, is solved locally either using the hydrostatic approximation or non-hydrostatically using a time-split compressible approximation. Advection is calculated using local finite difference operators rather than using non-local spectral methods. The advantage of the non-global character of finite-difference schemes can be used to great advantage for parallelization with good computational accuracy.

3.1 RAMS Physical and Numerical Options

Any implementation of RAMS involves selecting from among a wide variety of options and features in order to optimize the model for the purpose at hand. RAMS is equipped with a multiple grid nesting scheme which allows the model equations to be solved simultaneously on any number of interacting computational meshes of differing spatial resolution. The highest resolution meshes are used to model the details of smaller-scale atmospheric systems, such as flow over complex terrain and surface-induced thermal circulations. The coarser meshes are used to model the environment of these smaller systems, thus providing boundary conditions to the fine mesh region. In addition, the coarser meshes are used to simulate larger scale atmospheric systems which interact with the mesoscale systems resolved on the finer grids.

Following is a list of the RAMS options available.

Table 1. Range of options for configuring RAMS

BASIC EQUATIONS	• Hydrostatic incompressible or compressible • Non-hydrostatic time-split compressible
DIMENSIONALITY	• 2-dimensional • 3-dimensional
VERTICAL COORDINATE	• Standard Cartesian coordinate • Terrain-following height coordinate
HORIZONTAL COORDINATE	• Standard Cartesian coordinate • Rotated polar-stereographic transformation
GRID STRUCTURE	• Arakawa-C grid stagger

	• Unlimited number of nested grids • User-specified space and time nesting ratios • Ability to add and subtract nests • Moveable nests
TIME DIFFERENCING	• Leapfrog • Forward • Hybrid combination
TURBULENCE CLOSURE	• Deformation/stability-based (Smagorinsky-Lilly type) • Turbulent kinetic energy (Mellor-Yamada type) • Turbulent kinetic energy (Deardorff type)
CONDENSATION	• Grid points fully saturated or unsaturated • No condensation
CLOUD MICROPHYSICS	• Warm rain processes • Five ice condensate species
RADIATION	• Chen and Cotton (1988) long/shortwave model • Mahrer and Pielke (1977) long/shortwave model
LOWER BOUNDARY	• Surface layer similarity theory (Louis, 1979) • Tremback/Kessler (1985) soil temperature and moisture model • Vegetation temperature and moisture model
UPPER BOUNDARY CONDITION	• Rigid lid • Raleigh friction layer • Prognostic surface pressure • Gravity wave radiation condition
LATERAL BOUNDARY CONDITION	• Klemp and Wilhelmson (1978 ab) radiative condition • Orlanski (1976) radiative condition • Klemp and Lilly (1978) radiative condition • Davies (1976) nudging condition • Cyclic conditions
INITIALIZATION	• Horizontally homogeneous • Above plus variations to force cloud initiation • Analysis of grid point data/obs on isentropic surfaces • Grid point data interpolated to model grid

Various aspects of the RAMS structure and options will now be described in greater detail.

<u>GRID STRUCTURE</u>: The Arakawa C grid is employed (Messinger and Arakawa, 1976). All thermodynamic and moisture variables are defined in the center of a grid volume with the velocity components staggered 1/2 of a grid space in their normal direction. This stagger has several advantages including isotropy of the velocity component locations relative to the thermodynamic variables which is very important for mass and flux conservation.

HORIZONTAL COORDINATE SYSTEM: The horizontal grid transformation is a rotated polar stereographic projection. The pole of the map projection is rotated to a user-specified location, typically the center of the coarsest model nest. This reduces the amount of distortion between model grid spacings and the corresponding distance on the earth. Also, for smaller domain sizes, the horizontal structure is very close to Cartesian or UTM coordinates.

VERTICAL COORDINATE SYSTEM: The vertical coordinate transformation is the terrain-following sigma-z coordinate representation as described by Gal-Chen and Somerville (1975) and Clark (1977). The sigma-z coordinate has the advantage over a sigma-p coordinate in that the model levels are at the same absolute height above the ground at all times during a simulation.

NESTING BOUNDARY CONDITIONS: RAMS is equipped with a multiple grid nesting scheme which allows the model equations to be solved simultaneously on the interacting computational meshes of differing spatial resolution. The two-way interaction between the nested grids is performed following the scheme by Clark and Farley (1984) and Clark and Hall (1990). The time dependent model solution is first updated on the coarsest grid. A tri-quadratic spatial interpolation is then performed to obtain values which are assigned to the spatial boundaries of a finer grid nested within a coarser grid. The model fields on the finer grid are then updated using the coarser grid interpolated values as the spatial boundary conditions. Once the finer grid is at the same time level as the coarser grid, local spatial averages of the fine grid fields are obtained and used to overlay the coarse grid fields. Care has been taken in the implementation of the scheme to make sure that the interpolation and averaging cycle is reversible and that mass and momentum are conserved across the grid interfaces.

BASIC MODEL PHYSICS AND NUMERICS: The equation set most used by RAMS is the quasi-Boussinesq non-hydrostatic equations described by Tripoli and Cotton (1982). There are prognostic equations for all state variables including u, v, w, potential temperature, mixing ratio and Exner function (scaled pressure).

FINITE DIFFERENCE FORMULATION: The model is basically second-order in space and time. The advection terms are formulated in flux-conservative mode so that mass, momentum and energy are conserved. The model also has a "time-split" scheme to handle sound wave terms. This scheme, along with slowing the actual propagation speed of sound, computes those terms responsible for the sound waves on a smaller timestep than longer time scale terms such as advection (Tripoli and Cotton, 1982).

SOIL MODEL: For the lower boundary condition of the atmospheric model, the surface layer and soil parameterizations described in detail by Tremback and Kessler (1985) are used. This scheme is a modification of the scheme described by McCumber and Pielke (1981) in which the numerous iterative processes have been removed. This involves formulating prognostic equations for the soil surface temperature and water content by assuming a finite depth soil/atmosphere interface layer. The surface layer fluxes of heat, momentum, and water vapor into the air will be computed with the scheme of Louis (1979). This scheme approximates the profile functions of Businger et al. (1971) with analytic expressions. The Louis scheme also exhibits more realistic behavior than the Businger scheme in the free convective limit and in very stable conditions.

LAND USE: Variable land use characteristics are employed using high resolution USGS digital data bases. The input distinguishes between water and land surfaces, and the latter further subdivided into approximately eighteen classes. Each model grid cell is assigned its predominant land use type (or percent land, percent water where appropriate) by analyzing the mode of the distribution of types within each cell. Each grid cell is then characterized by a roughness length and albedo derived from a look-up table.

VEGETATIVE COVERAGE: The current vegetation parameterization is derived from the work of Avissar (1988). For each grid column, a dominant vegetation class is defined. The types used in the

BATS model (Dickinson, 1986) are included. The interactions between the vegetation, soil and atmosphere are then accounted for.

<u>MICROPHYSICS AND CONVECTIVE SCHEME</u>: On coarser grids, RAMS uses a convective parameterization scheme based upon Kuo (1974). On all grids, the explicit microphysics parameterization may be activated. The development of a completely new bulk microphysical code for RAMS has been completed. The new scheme is a generalization of the older microphysical code in that it can treat each water category (cloud water, rain, pristine ice, snow, aggregates, graupel and hail) as a generalized gamma distribution. It includes hail as a new category and allows graupel and hail hydrometers to contain some liquid water. New schemes have been implemented for homogeneous and heterogeneous nucleation of pristine ice, and for the conversion of ice between the large and small pristine categories resulting from vapor deposition or sublimation. A very efficient solver for the stochastic collection equation has been implemented based on new analytic solutions to the collection integral. A new sedimentation routine allows differential fall speeds based on the gamma size distribution. The code has been designed with computational efficiency as a major goal, and it runs considerably faster than the old RAMS microphysics module.

<u>RADIATION</u>: RAMS currently contains two sets of radiative schemes, the schemes described by Mahrer and Pielke (1977) (MP) and the schemes described by Chen and Cotton (1983) (CC), which were based on the schemes of Webster and Stephens (1977). The MP schemes possess the advantage that they are very efficient to execute. However, they do not include the effects of clouds. The CC schemes include the effects of clouds but are more costly.

3.2 Model initialization

For weather forecasting simulations, RAMS needs data analyses for initial conditions and large scale lateral boundary tendencies. Various observational datasets are combined and processed with a mesoscale isentropic data analysis package (Tremback, 1990) which has been termed RAMS/ISAN (ISentropic ANalysis package). Isentropic coordinates have many advantages over other coordinate systems when applied to data analysis. Since the synoptic scale flow is, to a first approximation, adiabatic, an objective analysis performed on an isentropic surface will better approximate the interstation variability of the atmospheric fields. Also, isentropes tend to be ``packed” in frontal areas, thus providing enhanced resolution along discontinuities. In addition, because isentropes are sloped in the vicinity of fronts, short wavelength features are transformed into longer wavelengths that can be more accurately analyzed objectively with much smoothing than with other coordinate systems. However, there are some disadvantages to isentropic coordinates, namely that the vertical resolution decreases as the atmospheric stability decreases (e.g., in the planetary boundary layer) and that the isentropes frequently intersect the ground. Therefore, one of the main features of ISAN has been the inclusion of a “hybrid” vertical coordinate, a mixture of isentropic and the terrain-following, σ coordinates which help to alleviate the problems that occur with isentropic coordinates near the ground.

ISAN can perform a separate data analysis to any or all of the nested grids specified for the RAMS simulation. This allows the user to create a higher resolution analysis over, for instance, an intensive field site. If data resolution does not warrant a separate analysis, a nested grid may be interpolated from the next coarser grid.

The first step in the analysis procedure is to access the available gridded forecast datasets. These data are accessed over the area of interest and interpolated onto the RAMS polar-stereographic grids, creating a polar-stereographic/pressure coordinate dataset. Then, the data are interpolated vertically to both the isentropic vertical coordinate and the terrain-following, σ coordinate. Once the large scale data has been processed, any available (routine, special, or bogus) rawinsondes observations may be accessed. All significant and mandatory level wind, temperature, and moisture data can be used from the rawinsonde reports. The horizontal wind components, pressure, and relative humidity are interpolated vertically to the same isentropic levels or the σ levels as was the large scale data.

An separate objective analysis is then performed on the isentropic and σ datasets. The Barnes (1973) objective analysis scheme is applied to the wind, pressure, and relative humidity on the isentropes and the wind, temperature, and relative humidity on the σ levels. User-specified parameters control the smoothing characteristics of the objective analysis and the relative importance between the rawinsondes and the large scale data. The atmospheric variables at the earth's surface are analyzed in a similar manner to the upper air variables. Wind components, potential temperature, and relative humidity are objectively analyzed using the Barnes scheme.

At this point, there are three objectively-analyzed, gridded datasets on the polar-stereographic RAMS grids, the isentropic and σ upper air datasets and the surface dataset. These data are blended in the following manner. First, a layer is chosen, usually starting at a level just above the boundary layer, extending for 1-3 km. From the surface to the bottom of this layer, the analysis will be defined completely from the σ data. From the top of this layer to the top of the model domain, the analysis will be defined completely from the isentropic data. In the layer, there is a simple weighted average of the isentropic and σ data where the weighting function is a linear function of height. The surface analysis is then blended into the upper level analysis in the lower levels.

4. Parallel Design Considerations

There are many issues to consider when taking a large serial code and modifying it for multi-processor execution. This section will describe some of these considerations.

In approaching the modification of a large code such as RAMS for parallel computation, we initially identified several design requirements and considerations to take into account. The following were our initial goals; we will mention if we met these goals for each point.

- We concentrated on the "large" three-dimensional computational configurations of RAMS, not the smaller two-dimensional runs that can be requested. In general, the larger the three-dimensional domain (measured in numbers of grid points), the more efficient the parallelization should become. This goal has been met; only three-dimensional runs are allowed to be run in parallel.

- No compromises should be made in the physics or numerics. It would be possible to reduce numerical accuracies in some schemes such as advection to reduce the amount of communication among the processors. This goal has been met, although the sixth-order advective scheme has not been implemented yet in the newest parallel version.

- We targeted the distributed memory MIMD architecture. With SIMD computers or shared-memory systems, a micro-tasking structure in which the parallelization is done on the individual loop level could be explored, which would require a distinctly different code structure than a MIMD machine. A distributed-memory architecture usually requires a much coarser-grained parallelism, since communication between processors needs to be kept to a minimum. This goal has been met. It has turned out that the MIMD parallel structure has been extremely efficient on shared memory platforms tested (see Section 7).

- Scalability of the parallel algorithms to massively-parallel platforms was considered. Although our initial goals were to target a workstation cluster (< 8 nodes), we have ported RAMS to massively-parallel platforms (>64 nodes). This goal has been partially met; we are continuing to work on the scalability.

- For ease of software maintenance, a single code version should be developed for both uniprocessor and parallel platforms. This goal has been met.

- There should not be any performance degradation for the code on uniprocessor platforms. This goal has been met and we continue to work on overall efficiency improvements such as better cache usage.

Therefore, with these requirements and considerations, the code modification has progressed. The following sections will summarize the features of the parallel RAMS and the basic techniques used in accomplishing the goals.

5. RAMS Parallel Version Differences

The parallel version of RAMS was designed to be very compatible with the latest single-processor version. The differences that do exist between the latest serial RAMS version (3b) and the parallel version are because of unimplemented features in the parallel version or at the code level.

- **Unimplemented features:** There are a number of RAMS features that have not been used very extensively in recent years which were not chosen for implementation in the parallel version. These include:
    - hydrostatic equation set options
    - forward and leapfrog time differencing (only the “hybrid” scheme was implemented)
    - “old” microphysics scheme

- **Code changes**: At the code level, there are a large number of changes mostly due to the modifications for the parallel execution. The major changes fall into four categories:

    - **allowing the code to execute on a *sub-domain* -** The serial code was generally of a structure which executed a process or set of instructions over the full set of grid points for any grid. This had to be modified to allow it to execute these instructions over a subset of these points. Along with the considerations for parallel efficiency (such as communication/computation overlap), care needed to be taken to make sure that operations were scheduled correctly so that the boundary regions of the sub-domains had access to the correct information that is passed from adjoining nodes.
    - **message passing -** Code was developed to handle the passing of data (messages) between the sub-domains. Since, at different places in the execution, different amounts of data need to be passed, different sets of code were implemented to handle these tasks. Code to handle the bookkeeping chores also was developed. This includes the tasks of domain decomposition and dynamic load balancing.
    - **additional C language code**: Generally, the message passing libraries have been developed and written in the C language. Therefore, it is advantageous at times to interface to these libraries with C routines. A few additional RAMS code files have been rewritten in C, most notably the main program. This also allows for the capability to include standard portable command line arguments when specifying the executable name. Also, several of the FORTRAN modules are now passed through the C preprocessor to allow for conditional compilation.
    - **code improvements -** Several improvements to the code were made as development of version 4a progressed, either from efficiency considerations or for ease of use. Examples of these changes include a completely rewritten advection scheme (which combines the leapfrog and forward advective schemes in the same set of code) and a new method of setting

up memory and output variable specifications. Because of this latter change, a new input data file is now required to set up the model variable structure.

6. Parallel components and structure

6.1 Basic Code Structure

The original version of the parallel RAMS, which began its development at CSU in 1991, used the Parallel Virtual Machine (PVM) software, developed at Oak Ridge National Laboratory, for communication among the processors of the parallel computer platform (Bequelin et al., 1991). However, since then, a new de facto standard called MPI (Message Passing Interface) has been developed by a consortium of industry, government, and university computer scientists. RAMS has been modified to use MPI in addition to retaining the capability to execute under PVM. RAMS is structured in a standard master-node configuration, where the master process is a main controlling process handling model initialization and output while the nodes are the main workers, performing virtually all of the floating point computations needed for the model simulations.

The basic structure of RAMS in this master-node configuration is the standard method of *domain decomposition* where each processor is given a portion of the modeling domain on which to perform the computations. Each node is given a rectangular set of grid points and a surrounding boundary region, which will be referred to as the *subdomain.* Again, all computation is done on the node processes. During each model timestep, the nodes must exchange information at the subdomain boundaries.

6.2 Domain decomposition

Domain decomposition is the process of spatially subdividing any computational grid domain in the model into two or more subdomains, each one to be handed to a separate processor for carrying out the necessary computations for time integration. There are many possible ways of decomposing a 3-D grid domain, but the following considerations point strongly to one best method.

- Some computational algorithms in the model solve tri-diagonal linear systems in the vertical, requiring simultaneous knowledge of all grid cell values in the vertical column. Thus, it is to great advantage to keep an entire column on the same processor and to decompose only in one or both horizontal directions.
- Although 1-D horizontal decomposition, i.e., grouping not only an entire column but also all columns in a given constant-x or constant-y slab into the same subdomain, is conceptually and algorithmically simpler than 2-D decomposition, it increases both the total memory requirement and, more importantly, the amount of information that must be communicated between processors. Thus, we adopted the more general approach of decomposing in both horizontal directions.
- We desire the flexibility to utilize any number of processors that may be available including prime numbers.
- Parallel efficiency dictates that all processors should perform their computational tasks in nearly the same amount of time so that no processor needs to waste time waiting for the results from another. This requirement, coupled with the facts that (1) processors may run at different speeds (e.g., in a parallel cluster of dissimilar workstations) and (2) certain regions of a model grid require more computational steps than others (e.g., where clouds occur), dictates that model grid subdomains must be allowed to have unequal sizes in order to properly balance the computational load on each processor.

The domain decomposition algorithm in RAMS satisfies all the above requirements. The algorithm requires as input the number of grid points (vertical columns) in each horizontal direction, the number of processors to be used, the relative speed of each processor, and a set of “column work factors” or relative computational workload (CPU time) of each vertical column. Both the relative processor speeds and column work factors may be evaluated from the performance over the preceding timesteps or, at the beginning of a model run, estimated.

A diagram of grid subdomains resembles bricks laid in tiers, with straight, unbroken lines separating each tier. The lines separating the bricks in each tier generally terminate at the top and bottom of the tier but in some cases coincide with lines in adjacent tiers. The number of tiers is an integer close to the square root of the number of processors, as is the number of bricks in each tier, but adjustments are made as necessary to accommodate the exact number of processors available. Different tiers in general have different thicknesses, and bricks in a tier in general have different lengths. These dimensions, i.e., the horizontal dimensions of each subdomain, are collectively determined from the relative processor speeds and the column work factors in order to achieve precise load balancing.

An entire model grid consists of interior vertical grid columns where field variables are prognosed plus a single row of grid columns in which diagnostic lateral boundary conditions are applied forming a lateral perimeter. In the domain decomposition, each interior prognostic column is placed in only one subdomain and prognosed only in that subdomain which completely avoids repetitive prognostic computations between different processors. A single row of grid columns (the “halo” or overlap region) is added to form a perimeter around each subdomain prognostic region which duplicates a modest number of columns comprising the entire grid. These columns are used to store field values communicated from adjacent subdomain interiors where they are prognosed.

In applications where nested grids are employed in RAMS, domain decomposition is completely independent for each grid. Thus, decomposition is based only on the grid size and work factors plus the number and speed of the processors and does not depend on where the grid is placed within its parent grid, where finer grids are placed within it, or how the finer and parent grids are decomposed.

6.3 Types of communication

There are seven distinct types of message-passing events in the RAMS structure. These are:

- Initialization: During initialization, the master process will compute the grid decomposition and send all necessary information and data to the compute nodes. The full subdomain of information is sent for almost all variables. This will also occur during the dynamic load balancing.

- Long timestep overlap region: At the beginning of a timestep, the nodes will exchange the overlap regions of the prognostic variables. We have reduced the overlap region to one row.

- Long timestep overlap region (turbulence): In order to reduce the overlap region to one row, it was necessary to add an extra communication step where the turbulence exchange coefficients in the overlap region are exchanged between nodes. This technique has added an additional 5-10% parallel efficiency, depending on the platform, to the RAMS runs.

- Small timestep overlap region: During the computation of the small acoustic timestep, the compute nodes will exchange the overlap regions of the u and v velocity components and the pressure. This is only needed for one overlap row.

- Nested grid boundaries: Nodes containing a coarse grid will send the necessary data to a fine grid node for interpolation of the nested grid boundaries.

- Nested grid feedback: The compute nodes will average their fine grid information to the coarse grid structure and transfer that data to the appropriate coarse grid nodes.

- File output: When it is time for file output, the compute nodes will send the full subdomain of the prognostic variables to the master process.

All node to node communication types package variables into a single message before sending to the receiving nodes.

6.4 Concurrent computation/communication

It is desirable that communication between each processor of the parallel system be kept to a minimum if the overhead involved in the communication between processors in a system is significant compared to the computations performed. One way of minimizing the communication cost is to "schedule" communications so that they occur concurrently with computation. The modular structure of the RAMS code is very conducive to this type of technique. The timestep computational structure has been rearranged so that many of the model computational schemes that do not require information from the subdomain boundary regions (those schemes that operate either at a single grid point or in a vertical column) are executed first during a timestep. In addition, some of the model routines compute the interior portion of subdomains first. While these are occurring, the node processes exchange the subdomain boundary information. When the node is finished with the first set of computations, the subdomain boundary information will have been received so that the node can continue with the remainder of the computations. For smaller number of codes, the concurrent scheduling of communication and computation can even allow for reasonable efficiency in the use of a cost-effective, standard Ethernet for communication in a clustered workstation system (over a small number of nodes), rather than the more expensive optical fiber components.

6.5 Bookkeeping arrays

Once the domain decomposition is performed, all required communications between subdomain processors are determined. This involves tabulating the exact set of grid point locations for each variable to be sent and received between all relevant pairs of subdomains. In order to minimize communication, only required variables from required locations are sent and received. In some cases, only a single grid column needs to be communicated between processors while in others a rectangular block of columns must be sent. The starting and ending grid coordinates in both horizontal directions for each block are stored in a 5-D integer array. The arguments of this array are (1) the source or sending subdomain or processor, (2) the destination or receiving subdomain or processor, (3) the model grid number (since decomposition is different in general for each grid), (4) the designation of one of the four block coordinates, and (5) the type of communication.

6.6 File output

The file output of RAMS is accomplished through the master process. When the time comes to output the files, the compute processes send all necessary information back to the master, which can then write the files to local disk. Other techniques have been used where each node will output its portion of the data, then a separate process will recombine the individual parts. The concept of the master I/O process has several advantages including:

- The transfer of data is done over the usually higher speed message passing hardware. Network File System software is not used.
- No local disk space is needed (for MIMD platforms).
- After the nodes have transferred their data, they can continue with their processing without waiting for the master process to finish writing the files.
- The output files are immediately available from the master node for other activities.

6.7 Nested grid considerations

As mentioned, RAMS contains a two-way interactive grid nesting scheme where, during a timestep, the coarser grid gives boundary information to a fine grid and a fine grid averages its domain and overlays the appropriate area of the coarse grid. Nested grids can be either telescoping or simultaneously nested within a coarser grid. There is no limit imposed in RAMS as to the number of nests or the spatial nesting ratio between nests. This situation does pose a challenge for the development of an efficient parallel version of RAMS which uses domain decomposition, dynamic load balancing, and strives to minimize communication among the nodes.

The nesting scheme has been modified to increase its scalability. The new scheme decomposes each of the nested grids independently. Then the appropriate communications are done for the two-way nesting algorithms. For the feedback process, the fine grid data is averaged on the fine grid data is averaged on the fine grid nodes, then transferred to the coarse nodes. For the nested boundary interpolations, the coarse grid nodes send the data to the fine grid nodes where the interpolation is done. Note that in each of these schemes, only coarse grid data is transferred, thus reducing the amount of communication as much as possible.

6.8 Dynamic load balancing

With the techniques of domain decomposition, inefficiencies will arise when one processor takes much longer with its subdomain than the other processors. This can happen frequently in RAMS because of spatial differences in model physics and hence the complexity of the physical parameterizations the model has to compute. If this happens, the remainder of the processors will sit idle. It would be possible to make all computations at all grid points regardless of the necessity. However, we have implemented a technique for *dynamic load balancing,* an objective way to allow for the adjustment of the computational load among the processors as a model simulation progresses.

The dynamic balancing in RAMS is accomplished in the following manner. At the end of each timestep, the compute nodes will send the CPU time and the wall clock time from its subdomains back to the master process. The master process then can determine if imbalances are occurring. When the compute nodes have sent all the subdomain fields back to the master process, which occurs when it is time to write an output file to disk, the master process is able to basically repeat the initialization procedure. This consists of computing the work factor for each grid column, decomposing the model grids according to the work factors, and sending out new subdomain information and data to the compute processes.

7. Parallel efficiency results

Numerous tests have been run with the parallelized version RAMS on several parallel platforms. As of this writing, we have ported the code to workstation clusters, the IBM SP2, and the HP/Convex Exemplar. Other groups have ported the parallel RAMS to the CRAY J90, T3D, and T3E and the SGI Power Challenge. Obviously, the parallel performance is dependent on the hardware characteristics including the machine architecture, speed of CPU, and type and speed of the CPU interconnect. But parallel efficiency is also dependent on the configuration of the model simulation, with such things as numbers of grid points, number of grids, and complexity of physics needing to be considered.

For the ports we have made, we will state generally some of the parallel efficiencies we have attained.

- **Workstation clusters**: We have run on several clusters (2-8 nodes) of IBM RS/6000 workstations using standard Ethernet as the networking hardware. Efficiencies attained have ranged from 60% to 85% depending on model configuration and number of nodes.

- **IBM SP2**: On the SP2 using the high-speed switch, efficiencies have ranged from about 90% on larger simulations using 8 nodes to about 68% using 64 nodes.

- **HP/Convex Exemplar**: The Exemplar (or SPP 1600) is an 8 processor, shared memory machine using a crossbar switch. For large multi-grid runs, we routinely achieve 90-95% efficiency on the 8 processors. For moderate size multi-grid runs (number of horizontal grid points: 50 by 50), we are able to get better than 100% efficiency because of better cache utilization.

- **SGI Origin**: We have access to a Silicon Graphics Origin 200, a 4 processor, shared memory machine. For medium-size, multi-grid runs, we have achieved 90-95% efficiency.

8. Future developments

RAMS continues to be developed and capabilities expanded. Following are some of the issues we will be considering in the near future as related to the computational performance.

- ***Increased serial code efficiency:*** We continue to look for ways to increase the single processor performance of the code. These include better algorithmic ways to program various numerical techniques and to better utilize the CPU cache.

- ***Investigate additional concurrent communication/computation possibilities:*** Additional possibilities exist for overlapping the computations and communications such as computing more terms on the sub-domain interior first.

- ***More complicated “work” factor:*** The computation of the work factor, which is used in the domain decomposition, is not a straightforward technique, as it relies on the presence of a wide range of different variables and the knowledge of how these variables affect the computational cost. We are continuing to experiment with better algorithms to define this quantity.

- ***Continue to explore shared-memory architecture considerations:*** Computer manufacturers are beginning to introduce MIMD-type platforms where each node is a symmetric multi-processor. We will be looking at this architecture to determine the most efficient ways to utilize these platforms. Also, several shared memory platforms with a moderate number of processors (8-32 CPUs) have come on to the market recently. Although our initial impression is favorable as to the efficiency of a MIMD code structure on these platforms, we will continue to investigate the code performance

9. References

Avissar, R. and R.A. Pielke, 1989: A Parameterization of Heterogeneous Land Surfaces for Atmospheric Numerical Models and Its Impact on Regional Meteorology. *Mon. Wea. Rev*., **117**, 2113-2136.

Bames, S.L. , 1973: Mesoscale Objective Map Analysis Using Weighted Time Series Observations. NSSL Tech. Memo, NSSL-62, 60 pp.

Bequelin, A., J. Dongarra, G.A. Geist, R. Manchek and V. Sunderam, 1991: A User's Guide to PVM - Parallel Virtual Machine, Technical Report ORNL/TM-11826, Knoxville, TN.

Businger, J.A., J.C. Wyngaard, Y. Izumi, and E.F. Bradley, 1971: Flux-profile relationship in the atmosphere surface layer. *J. Atmos. Sci*., **28**, 181-189.

Chen, S. and W.R. Cotton, 1988: The Sensitivity of a Simulated Extratropical Mesoscale Convective System to Long Wave Radiation and Ice-Phase Microphysics. *J. Atmos. Sci*., **45**, 3897-3910

Clark, T.L., and R.D. Farley, 1984: Severe downslope windstorm calculations in two and three spatial dimensions using anelastic interactive grid nesting: A possible mechanism for gustiness. *J. Atmos. Sci*., **41**, 329-350.

Clark, T.L., and W.D. Hall, 1991: Multi-domain simulations of the time dependent Navier-Stokes equations: Benchmark error analysis of some nesting procedures. *J. Comput. Phys*., **92**, 456-481.

Gal-Chen, T., and R.C.J. Somerville, 1975: On the use of a coordinate transformation for the solution of the Navier-Stokes equations. *J. Comput. Phys*., **17**, 209-228.

Klemp, J.B. and D.K. Lilly, 1978: Numerical simulation of hydrostatic mountain waves. *J. Atmos. Sci*., **35**, 78-107.

Klemp, J.B. and R.B. Wilhelmson, 1978a: The simulation of three-dimensional convective storm dynamics. *J. Atmos. Sci*., **35**, 1070-1096.

Kuo, H.L., 1974: Further studies of the parameterization of the influence of cumulus convection on large-scale flow. *J. Atmos. Sci*., **31**, 1232, 1240.

Louis, J.F., 1979: A parametric model of vertical eddy fluxes in the atmosphere. *Boundary-Layer Meteorol*., **17**, 187-202.

Mahrer, Y. and R.A. Pielke, 1977: A Numerical Study of the Air Flow over Irregular Terrain. *Contrib. Atmos. Phys*., **50**, 98-113.

Orlanski, I., 1976: A simple boundary condition for unbounded hyperbolic flows. *J. Comput. Phys*., **21**, 251-269.

Mesinger, F. and A. Arakawa, 1976: Numerical methods used in atmospheric models. GARP Publication Series, No. 14, WMO/ICSU Joint Organizing Committee, 64 pp.

Tremback, C.J. and R. Kessler, 1985: A surface temperature and moisture parameterization for use in mesoscale numerical models. Preprints, 7th Conference on Numerical Weather Prediction, 17-20 June 1985, Montreal, Canada, AMS.

Tremback, C.J., 1990: Numerical Simulation of a Mesoscale Convective Complex: Model Development and Numerical Results. Ph.D. dissertation, Colorado State University.

Tripoli, G.J. and W.R. Cotton, 1982: The Colorado State University Three-Dimensional Cloud/Mesoscale Model. Part I: General Theoretical Framework and Sensitivity Experiments. *J. de Rech. Atmos*., **16**, 185-195.