# Implementation of a Message Passing Interface into a Cloud-Resolving Model for Massively Parallel Computing

Hann-Ming Henry Juang[1], Wei-Kuo Tao[2], Xiping Zeng[2,3], Chung-Lin Shie[2,3], Joanne Simpson[2] and Steve Lang[4],

[1]*Environmental Modeling Center*
*NCEP, NOAA*
*Washington, DC*

[2]*Laboratory for Atmospheres*
*NASA/Goddard Space Flight Center*
*Greenbelt, MD 20771, USA*

[3]*Goddard Earth Sciences and Technology Center*
*University of Maryland, Baltimore County*
*Baltimore, MD*

[4]*Science Systems and Applications Inc.*
*Lanham, MD 20706*

June 2, 2004

*Mon. Wea. Rev.*

[1] Corresponding author address: Dr. Hann-Ming Henry Juang, Environmental Modeling Center, NCEP, NOAA, W/NP5 Room 204, 5200 Auth Road, Camp Springs, MD 20746

## Abstract

The capability for massively parallel programming (MPP) using a message passing interface (MPI) has been implemented into a three-dimensional version of the Goddard Cumulus Ensemble (GCE) model. The design for the MPP with MPI uses the concept of maintaining similar code structure between the whole domain as well as the portions after decomposition. Hence the model follows the same integration for single and multiple tasks (CPUs). Also, it provides for minimal changes to the original code, so it is easily modified and/or managed by the model developers and users who have little knowledge of MPP.

The entire model domain could be sliced into one- or two-dimensional decomposition with a halo regime, which is overlaid on partial domains. The halo regime requires that no data be fetched across tasks during the computational stage, but it must be updated before the next computational stage through data exchange via MPI. For reproducible purposes, transposing data among tasks is required for spectral transform (Fast Fourier Transform, FFT), which is used in the anelastic version of the model for solving the pressure equation.

The performance of the MPI-implemented codes (i.e., the compressible and anelastic versions) was tested on three different computing platforms. The major results are: 1) both versions have speedups of about 99% up to 256 tasks but not for 512 tasks; 2) the anelastic version has better speedup and efficiency because it requires more computations than that of the compressible version; 3) equal or approximately-equal numbers of slices between the x- and y-

directions provide the fastest integration due to fewer data exchanges; and 4) one-dimensional slices in the x-direction result in the slowest integration due to the need for more memory relocation for computation.

## 1. Introduction

Cloud-resolving models (CRMs), which are based the non-hydrostatic equations of motion, have been extensively applied to cloud-scale and mesoscale processes during the past four decades (see a brief review by Tao 2003). Table 1 lists the major foci and some (not all) of the key contributors to CRM development over the past four decades. Because cloud-scale dynamics are treated explicitly, uncertainties stemming from convection that have to be parameterized in large-scale (hydrostatic) models are obviated, or at least mitigated, in CRMs. Also, CRMs solve the equations of motion with much higher spatial and temporal resolution and use more sophisticated and physically realistic parameterizations of cloud microphysical processes (although by no means perfect yet). CRMs also allow explicit interactions between clouds, radiation and surface processes. For this reason, the Global Energy and Water Cycle Experiment (GEWEX) formed the GEWEX Cloud System Study (GCSS), which chose CRMs as the primary approach to improve the representation of moist processes in large-scale models (GCSS Science Plan 1993; Randall *et al.* 2003). Global models will use a non-hydrostatic framework with horizontal resolutions of 5-10 km one to two decades from now.

In recent years, exponentially increasing computer power has extended CRM integrations from hours to months (i.e., Wu *et al.* 1998) and the number of computational grid points from less than a thousand to close to ten million (Grabowski and Moncrieff 2001).

Three-dimensional CRMs are now more prevalent. Much attention is being devoted to precipitating cloud systems where the crucial 1 km scales are resolved in horizontal domains as large as 10,000 km in two-dimensions and 1,000 x 1,000 km$^2$ in three-dimensions. However, many CRMs need to be re-programmed (re-coded) in order to fully utilize the fast advancement of computing technology (i.e., massive parallel processors)[1].

In this paper, the design for massively parallel programming (MPP) with a message passing interface (MPI) that is implemented into a three-dimensional (3D) version of a CRM, the Goddard Cumulus Ensemble (GCE) model will be presented. The concept of MPI implementation, along with the method of domain decomposition and data-communication to avoid aforementioned risks, will be presented. In section 2, a brief description of the GCE model will be given. The MPI implementation will be described in section 3. The performance of the model with MPI, implementation regarding the model dynamics (anelastic and compressible), stability, speedup, efficiency, reproducibility and wall-clock comparisons among different decompositions, tasks and dimensions using three different computing platforms will be given in section 4. The summary and conclusion are given in section 5 with future model developments.

## 2. Goddard Cumulus Ensemble (GCE) model

### *2,1 GCE model description and applications*

1 IBM Blue Gene Lite supercomputer (estimated to be 5-10 times more powerful than the Japanese Earth Simulator by IBM representatives) will be delivered in 12 months with 65,000 CPU nodes and as many as

The Goddard Cumulus Ensemble (GCE) model has been developed and improved at NASA/Goddard Space Flight Center over the past two decades. The development and main features of the GCE model have been extensively published by Tao and Simpson (1993) and Tao *et al.* (2003a). Recent improvements and testing were presented in Ferrier (1994), Tao *et al.* (1996), Wang *et al.* (1996), Lynn *et al.* (1998), Baker *et al.* (2001) and Tao *et al.* (2003b). A Kessler-type two-category liquid water (cloud water and rain) microphysical formulation is mainly used with a choice of two three-class ice formulations (3ICE), namely that by Lin *et al.* (1983) and the Lin scheme modified to adopt slower graupel fall speeds as reported by Rutledge and Hobbs (1984). An improved four-class, multiple-moment ice scheme (4ICE) has also been developed (Ferrier 1994) and tested for several convective systems in different geographic locations (Ferrier *et al.* 1995). The 4ICE scheme only requires minimal tuning compared to the 3ICE schemes. Recently, two detailed spectral-bin microphysical schemes (Khain *et al.* 2000; Chen and Lamb 1999) were also implemented into the GCE model. The formulation for the explicit spectral-bin microphysical processes is based on solving stochastic kinetic equations for the size distribution functions of water droplets and several types of ice particles. Each type is described by a special size distribution function containing many categories (i.e., 33 bins). Atmospheric aerosols are also described using number density size-distribution functions. Significant computation is required in applying this explicit spectral-bin microphysics to study cloud-aerosol interactions and nucleation scavenging of aerosols, as well as the impact of different concentrations and size distributions

655,361 CPU nodes with optimal parallel efficiency leaving a lot of room for future supercomputer of aerosol particles upon cloud formation. These new microphysics, however, require the use of a multi-dimensional Positive Definite Advection Transport Algorithm (MPDATA, Smolarkiewicz and Grabowski 1990) to avoid "decoupling" between mass and number concentration[2]. The positive definite advection scheme also produces more light precipitation, which is in better agreement with observations (Johnson *et al.* 2002; Lang *et al.* 2003). Solar and infrared radiative transfer processes (Chou and Suarez 1999; Chou *et al.* 1999) have been included, and their impact on cloud development as well as several hypotheses associated with cloud-radiation interaction have been assessed (Tao *et al.* 1996; Sui *et al.* 1998). A sophisticated seven-layer soil/vegetation land process model has also been implemented into the GCE model (Lynn *et al.* 1998). Subgrid-scale (turbulent) processes in the GCE model are parameterized using a scheme based on Klemp and Wilhelmson (1978) and Soong and Ogura (1980), and the effects of both dry and moist processes on the generation of subgrid-scale kinetic energy have been incorporated. Table 2 shows the major characteristics of the GCE model.

The application of the GCE model to the study of precipitation processes can be generalized into fourteen categories (see Table 2 in Tao 2003). They are: 1) the mechanisms associated with cloud-cloud interactions and mergers (Tao and Simpson 1984, 1989a), 2) $Q_1$ and $Q_2$ Budgets and their individual components in different geographic locations (Soong and

development with microchips.

2 Decoupling means that a grid point has mass without number concentration or has number concentration without mass. The decoupling is caused by large phase errors associated with the spatially centered (second or fourth order) advection scheme.

Tao 1980; Tao and Soong 1986), 3) statistical characteristics of clouds, convective updrafts and downdrafts (Tao *et al.* 1987), 4) role of the horizontal pressure gradient force on momentum transport and budget (Soong and Tao 1984; Tao *et al.* 1995), 5) ice processes and their role in stratiform rain formation and the associated mass, $Q_1$ and $Q_2$ budgets (Tao and Simpson 1989b, Tao *et al.* 1989), 6) the redistribution of trace gases by convection and enhancement of $O_3$ production in the tropics (Scala *et al.* 1990; Pickering *et al.* 1992a, b; and a review by Thompson *et al.* 1997), 7) precipitation efficiency (Ferrier *et al.* 1996; Tao *et al.* 2004), 8) cloud radiation interaction and their impact on diurnal variation of precipitation (Tao *et al.* 1996), 9) the horizontal transport of hydrometeors and water vapor from convective towers into the stratiform region (Tao *et al.* 1993a, Tao 1995), 10) the effects of surface fluxes on precipitation processes, CAPE and boundary layer structure (Wang *et al.* 1996; 2003), 11) mesoscale circulations induced by soil gradients and their effects on precipitation and initialization of convection (Lynn *et al.* 1998), 12) physical processes associated with idealized climate variations and convective-radiative quasi-equilibrium in the Tropics (Sui *et al.* 1994; Tao *et al.* 1999, 2001a), 13) enhancing the performance of TRMM rainfall retrieval algorithms by providing realistic cloud profiles (Tao *et al.* 1990; 1993b and a review by Simpson *et al.* 1996), and 14) developing algorithms for retrieving the four-dimensional vertical structure of latent heating over the global tropics (Tao *et al.* 2000), A review on the application of the GCE model to the understanding of precipitation processes can be found in Simpson and Tao (1993) and Tao (2003). Figure 1 shows recent 3D GCE simulations capturing detailed convective systems typical of the SCSMEX and KWAJEX regions[3].

Several national and international universities and research institutions (i.e., University of Maryland, University of Virginia, Columbia University, University of New York at Albany, Texas Austin College, Florida State University, University of Washington, Monash University, Australia, Hebrew University of Jerusalem, Israel; Seoul National University, Korea, National Central University, Taiwan) are using the GCE model and its results in their research. These professors and researchers are important partners because they can inform us about the model performance. In addition, the GCE microphysical processes, land processes and cloud-radiation interactive processes were implemented into two community mesoscale models (Penn State U/NCAR MM5 and Advanced Regional Prediction System, ARPS).

## *2.2 Anelastic and compressible versions*

The GCE model flow can be either anelastic (Ogura and Phillips 1962), filtering out sound waves, or compressible (Klemp and Wilhelmson 1978), which allows the presence of sound waves. The sound waves are not important in thermal convections, but their processes can place severe restrictions on the time step in numerical integrations because of their high propagation speed. For this reason, most cloud modelers use an anelastic system of equations in which sound waves have been removed by eliminating certain terms in the

3 SCSMEX stands for South China Sea Monsoon Experiment, KWAJEX for Kwajalein Experiment.

compressible system (*e.g.*, neglecting the local variation of air density with time in the mass continuity equation). A 3D diagnostic (elliptic) pressure equation can be solved using direct (*e.g.*, Fast Fourier Transform, FFT) or iterative methods. However, when terrain is added, the direct methods become more complex and sometimes time consuming.

In the compressible system, the pressure-equation is derived by taking the derivative of the thermodynamic equation and using the compressible continuity equation (Klemp and Wilhelmson 1978). Due to the presence of sound waves, a very small time step (2 s for a 1000 m spatial resolution) is needed for time integration of the entire model equation set. However, Klemp and Wilhelmson (1978) developed a semi-implicit time-splitting scheme, in which the equations are split into sound-wave and gravity-wave components, to achieve computational efficiency. The small time step integration is semi-implicit in the vertical using a 2 s time step. The remaining time integration can use a 10 s time-step. One advantage of the compressible system is its computational simplicity and flexibility. The numerical code then remains a set of explicit prognostic equations and alterations such that stretched or nested grids, surface terrain and boundary conditions (*e.g.*, radiative upper boundary) can be incorporated into the numerical model without complicating the solution procedure.

Anderson *et al.* (1985) have tested an anelastic system using a 4 s time step against the results from a fully compressible system without using the time-splitting technique (which needs a 0.3 s time step). They found that the anelastic system produced essentially identical results to those of the compressible system for 2-D cool pool experiments lasting

500 s where the convection was initiated by a cool pool. Ikawa (1988) also compared the anelastic and the compressible systems for a 2-D case involving orography. His results indicated that both simulated systems are similar if sound waves are damped enough in the compressible system. A pair of sensitivity tests, anelastic versus compressible, were also performed with an initial condition associated with a mid-latitude squall line using the GCE model (Tao and Simpson 1993). All model physics were activated in these two runs for a 12-h time simulation in order to maximize the possibility of differences. It showed that the GCE model does not produce identical results with the two different systems. The differences between the anelastic and compressible systems are much smaller, however, than those obtained by changing microphysical processes and advection schemes. Several observed features, such as the propagation speed of the squall system, the weakly evolving multi-cellular structure, the meso-low aloft, the squall pressure high, and the rear inflow from the middle troposphere, were well-simulated in both systems. No significant differences were found in the simulated cloud updraft/downdraft structure or cloud top heights between the two systems.

## 3. Massively parallel implementation

There is no need to emphasize the point that the improvement of numerical model development is strongly based on the advancement of computer resources. It is known that numerical models have to be modified to fully take advantage of the advanced computer architectures as compilers cannot keep up with the new architectures. When machines change from scalar to vector, independent computations in the innermost loop had to be provided by the models. When multi-processors were added into single-box machines, directives before the loop or outer-most loop for multi-tasking in a shared-memory multi-processor machine had to be inserted in the models, and this is known as a single instruction multiple data (SIMD) computation. When multi-processor one-box machines with shared-memory architecture evolved into distributed-memory machines with multi-processors in several boxes for multi-process parallel (MPP) computing, numerical models were recoded with domain decomposition moving the computation from SIMD to MIMD (multiple instruction multiple data computation). Furthermore, when hybrid machines with some processors in shared memory arrive and some processors in distributed memory, then multi-thread multi-tasking and domain decomposition multi-parallel-tasking will have to be implemented into numerical models.

The programming requirement in numerical modeling for MPP architecture comprises domain decomposition and data communication. The techniques in domain decomposition and data communication may depend on particular numerical methods used in the models. Certain methods may do better with certain decompositions. Thus, several different domain decompositions need to be performed. Since atmospheric models are mainly three-dimensional, domain decomposition can be up to three dimensions. However, due to domain dependent computations, such as column physics, which require all model grids in the vertical, decomposition is used only up to two dimensions. Therefore, we can have either one-dimensional (1-D) or two-dimensional (2-D) decompositions. Any 1-D decomposition is usually simple and easy to implement, but it limits the maximum number of tasks to be the number of the specific direction. Two-dimensional decomposition can have a larger number of tasks limited by the product of the two given dimensions and is known to have less total data to transfer in terms of exchanging halo data (Johnson *et al.* 1994). Two-dimensional decomposition was concluded to be more scalable than 1-D decomposition in Skalin (1997a, b). Nonetheless, it also depends on the model and the platforms used. In the case of transposition between different decompositions, 1-D was found to have the same performance as 2-D in an IBM-SP and was better than 2-D in the VPP5000. This conclusion can be realized because 1-D has longer lengths for vector computation in the VPP5000 (Juang and Kanamitsu 2001).

There are several packages for MPP computing. The Message Passing Interface (MPI) package is the most popular MPP package in the meteorological community. Nevertheless there are several ways to implement MPI into a numerical model. Some introduce rules that add routines, and some even require re-coding the entire model to fully satisfy the MPI implementation in order to obtain peak performance. Severely modifying the model, while the model is modified severely to obtain peak performance, introduces two risks: 1) the code may be hard to read for further scientific and/or numerical improvements, and 2) it may not be reversible to a single code and difficult or tedious to implement future packages for advanced architectures. One of the solutions is to design the MPI implementation with an option to be able to run with a single-processor either for a new model being developed (Juang *et al.* 2003) or in an existing model (Purnell and Revell 1995). Thus, the model code without MPI can be obtained whenever it is needed providing readability and easily future improvements.

In this section, it will be illustrated how MPI can be implemented into a model (GCE model) without lots of code rewriting. The concept of adding MPI or grid-decomposition without disturbing the existing model structure and computation will be introduced, and the decomposition and addition of the halo regimes, if necessary, will be described. The way data is exchanged among all the halo regimes and transposed for FFT will also be shown.

## *3.1 Concept of MPI implementation*

As mentioned, the main concept of this MPI implementation is to preserve readability for further scientific development and be reversible to be able to recover the original code for future computer architect changes. From this concept comes some bottom-line solutions for the MPI implementation. First, all the model structures, array indices and computations will be kept original so the code is more readable for further scientific development. Secondly, MPI will be an add-on package containing all of the necessary MPI capabilities in one call at one place and not added throughout the code. A preprocessor is used to provide options as well as the ability to recover the original single-processor code. The C preprocessor is the most common used in UNIX platforms and FORTRAN programming; thus it is adopted for GCE model. It is required that the GCE model scientists become familiar with it.

## 3.2 *Methods of decomposition*

Following the main concept in the previous sub-section, the decomposition in grid-point space is illustrated in Fig. 2 with an example of 12 tasks in terms of three columns and four rows. The upper cube shown by the solid line represents the entire domain of the GCE model; the bottom cube also shown by the solid line represents a single partial model domain for any task. The dashed lines form the inner portion of the entire model and partial model domains for the upper and bottom cubes. The remaining outer portions (between the solid lines and dashed lines) are called lateral boundaries for entire model domain in the upper cube and halo regions for the partial domain in the bottom cube. The decomposition is conducted with as equal a number of grid points as possible for computational balance in the inner domain of the model. After each task obtains its portion of the model domain shown by the dotted lines from the upper cube, it also gets its halo grids to form the bottom cube. The shape between the entire domain and the partial domain may be different, but both have an inner portion and an outer portion, again, which is a lateral boundary for the entire domain and either a halo or lateral boundary for the partial domain. As mentioned, the array indexes are not changed throughout the code to represent the partial domains, but three indexes are introduced to represent the entire domain in three directions.

In order to illustrate decomposition in quantity, the following formula is introduced:

$$N_x = INT\left[\frac{F_x - 2L_x - 1}{M_{col}}\right] + 1 + 2L_x$$
$$N_y = INT\left[\frac{F_y - 2L_y - 1}{M_{row}}\right] + 1 + 2L_y \qquad (1)$$

where N, F, M and L are all integers, and INT followed by brackets indicates the result in the brackets is round to be an integer as in FORTRAN, F is the number of full grid points, L the number of lateral boundary and/or halo grid points, N the number partial grid points, and M the number of decomposition slices. The subscripts x, y, col and row are x-direction, y-direction, column and row. In the case of a single cpu, $M_{col} = M_{row} = 1$, then N = F. For 1-D decomposition, then either $M_{col}$ or $M_{row}$ equals one. For 2-D decomposition, both $M_{col}$ and $M_{row}$ are larger than one. F-2L may not be divisible by M without a remainder. The remainder R (always less than M,) is distributed to tasks by adding one to each task for R tasks. Thus N is the maximum dimension for any partial domain. Any given task may have a grid number of N or N-1. For the anelastic version, slices in the vertical are required for FFT, so the subscript z can be used to replace either x or y in Eq. 1.

## 3.3 *Data exchange for halo and lateral boundaries*

Since the GCE model is a finite difference model, any given point requires neighboring points in order to compute its derivative. After one complete time step, the lateral boundary (halo portion) of the partial domain has to be updated. Each partial domain is assigned initially knowing its lateral boundary and/or halo conditions. If it is a lateral boundary condition, it will apply either an open boundary condition using its own data or a cyclic boundary condition using another related partial domain. If it is a halo boundary condition, the halo portion has to be updated by the neighboring partial domain. For example, task 5 in Fig. 2 has lateral boundary conditions on the east side but halo boundary conditions on the west, north and south sides. It needs to exchange data with tasks 8, 4 and 2 (side neighbors) and tasks 1 and 7 (corner neighbors) to update its halo. It also needs to exchange data with side neighbor task 3 and corner neighbor tasks 0 and 6 for cyclic boundary conditions.

Except for open lateral boundary conditions, all lateral and halo portions have to be updated by exchanging data among neighboring grids or grids in related with related partial domain. Figure 3 illustrates the data exchange in two steps with side neighbors. It is a completed data exchange without involving corner neighbors. Shaded areas indicate updated data. The base of the arrow indicates where data is ready to be sent out, and the point of the arrow indicates where data are to be received and update. First the data is exchanged in the x-direction as shown in Fig. 3a. After updating the in x-direction, the data is exchanged in the y-direction as shown in Fig. 3b. After Fig. 3b, all halo-portions should be updated. The lateral boundary has two arrows: the short one represents an open boundary condition that is updated by internal data and the long one for cyclic boundary conditions, which is updated by the task at the furthest end of the model domain.

Since all tasks work together, assuming the initial time for MPI communication is negligible, the wall clock time can be related to the amount of data exchanged in one complete data exchange (Fig. 3) by the task, which spends the longest running time. The amount of data exchanged can be measured by using Eq. 1 and Fig. 3. From Fig. 3(a), the data exchanged along x are

$$2L_x\left[\frac{F_y-2L_y-1}{M_{row}}+1\right] \tag{2}$$

where $L_x$ is the data depth and the quantity in the brackets is the data length. Then from Fig. 3(b), the data exchanged along y are

$$2L_y\left[\frac{F_x-2L_x-1}{M_{col}}+1+2L_x\right] \tag{3}$$

where, again, $L_y$ is data depth, and the quantity inside the brackets is the data length, which includes lateral boundary grids. For simplicity, assume $L_x = L_y$ so that combining Eqs. 2 and 3 together gives

$$2L\left[\frac{F_y-2L-1}{M_{row}}+\frac{F_x-2L-1}{M_{col}}\right]+4L^2+4L \tag{4}$$

where the values inside the bracket determine the variation of data exchange. The two terms in the brackets indicate the number of inner grid points in the y and x directions of the partial domain, respectively. This shows that minimizing the value in the brackets requires minimizing the values of both terms, and it indicates that the larger the M, the smaller the value of each term will be. Thus, 2-D decomposition results in less data exchange than 1-D decomposition. Some evidence will be shown in section 4.

## *3.4 Transposing data for FFT*

In an anelastic system, FFT is used to solve for the diagnostic pressure equation (Ogura and Phipps 1962). For FFT in horizontal, x and y direction., grid point values for the entire domain are needed. In this case, the model domain should not be sliced in the x and y directions, only the z-direction can be sliced. The consequence of this is that only 1-D decomposition in the z-direction is possible for FFT. The number of tasks is thus limited to the dimension of the vertical direction. Fortunately, the FFT in the original design of the GCE model is a one-dimensional FFT. In other words, it transforms in the x-direction first, then transforms in the y-direction to obtain grid-point values from spectral coefficients by spectral transform. Thus, the MPI transpose can be used between each transformation to provide entire domain grid-point values or spectral coefficients in one direction then in the other direction. And 2-D decomposition can be used because only one direction cannot be sliced due to transformation; another direction and the vertical direction can be sliced. In this case, a large number of tasks can be used but limited by the vertical grid point number multiplied by the smaller of the two horizontal grid numbers in the x or y directions.

Figure 4 shows the entire process for FFT with an example of 12 tasks in 2-D decomposition with 3 columns and 4 rows. In the case of 1-D decomposition, the entire process is redone without the three transposes marked by the hollow arrows and no slicing for the 3 columns, only for the 4 rows. The four thin arrows indicate the transformation between grid-point values and spectral coefficients, and the three thick arrows indicate transposes for either 1-D or 2-D decomposition. Starting from the bottom left cube in grid-point space, the process follows the arrows to the spectral space in the second row from the left of the bottom cube. Then, after the spectral computations, it follows the arrows back to the top cube on the left in the grid-point space.

Figure 5 shows a schematic diagram illustrating the MPI transpose in 2-D decomposition. The 1-D MPI transpose is illustrated in Fig. 2 of section 3 of Juang *et al.* (2003). The 2-D transpose can be described here in the same fashion. The solid lines indicate the existing decomposition for the upper cube and after transpose in the bottom cube. The dashed lines indicate the slicing in each task (large-font numbers) before its sliced data is sent out to the tasks (small-font numbers) in the upper cube. The dashed lines in the bottom cube for each task (large numbers) indicate the portion of sliced data received from other tasks (small numbers). This example shows the transpose in the x-y direction between the entire grid for the x and y directions for FFT either in the x- or y-directions.

## *3.5 Summary of routine additions and code changes*

A user package of callable routines linking between the MPI binding and the GCE model is created to handle all MPI calls for the model. Each routine has a grid arrangement that calls MPI then rearranges back to the model grid for computation. These routines include: 1) gathering (scattering) data for output (input), 2) gathering all data for summation, and 3) exchanging data among subgroups of tasks or neighboring tasks.

Since all the necessary actions for adding MPI care contained in a call statement, the original code is not contaminated by too many new statements. There are only two model routines that call the transpose routine, three model routines that call the data exchange, and eight routines that call reproducible summation or non-reproducible quick summation routines in the user package. There are many model routines with the option to use a single task to print out a message. Model array indices are not changed because the entire domain and partial decomposed domains use the same index names for their own domain definition. The partial domain index and whole domain index are defined and used in the user package. The variable names for the whole domain dimensions are introduced. It is up to the user to provide the whole domain dimensions. The model dimensions for the partial domain dimensions are then computed by the user-provided decomposition. In order to easily manage the code for single and multi-task experiments, a C preprocessor, as mentioned, is used on the model code controlled by a user definition file included in each routine. Thus, any option can be changed in the user definition file. Furthermore, the options, as mentioned, provide the ability to run single-cpu or any numbers of multi-tasks with any decomposition.

## 4. Performance

This section contains several sub-sections. First, the stability of repeat runs and one long continuous run and the performance of different decompositions have to be examined to establish a benchmark for evaluating the performance of single and short-period runs. Next, the performance of different tasks in terms of wall-clock time, speedup and efficiency, for versions with and without Fourier transforms, different resolutions, and different platforms are shown.

Performance in terms of wall-clock-time in this section is based on model integrations of 4 hours with time steps of 12 seconds in all cases for two different versions of the model, except for the long-period (3 day) stability runs. All results hereafter are run with the in-line statistics option for a complete performance test. The statistics option is very costly. For example with 4 hr integrating, using statistics took about 300 seconds and without statistics took only 200 seconds for a case using 256x256x34 grid points (260x260 if lateral boundary grids are included) for compressible model with 512 tasks on HALEM. So, the statistics took about the one-third of total wall-clock time in this case. Three platforms, HALEM (a Dec alpha cluster), the IBM-SP (a Power4 cluster), and CHAPMAN (a SGI Origin 2000 cluster), are used to test the performance of the model parallelization.

## *4.1 Performance Stability*

Before showing results for single runs under all conditions, the stability of the performance has to be examined. Table 3 gives the wall-clock times for five arbitrary runs on HALEM and the IBM-SP using the same model configuration and decomposition of 8x8 (64 tasks). The same model configuration on CHAPMAN but for 1-hour integration is also listed with four members. The maximal time differences among different runs vary by 25 seconds on HALEM, and about 2.9 % of the mean of 866 seconds. For the IBM-SP, the maximal time difference among different runs is 74 seconds, about 6.6% of the mean of 1121 seconds. For CHAPMAN, there is an 8 second possible difference, which is about 2.3% of the mean of 339 seconds. These percentages are needed to measure the possible variation of all the results shown hereafter since there is only one run for each condition. In other words, results shown hereafter can be variated around these percentages.

Table 4 lists the wall-clock times for 3-day and 4-hour integrations using the same configuration (256x256x34 grid points) and 128 tasks on HALEM. The amount of wall-clock time with respect to the model integration time is about the same between the short-period 4-hour run and the long-period 3-day run. This indicates there is no need to conduct long integrations for performance evaluations.

## *4.2 Different decompositions*

As mentioned, the design of the decomposition that was implemented can be integrated with any number of tasks, whether the number is a prime number or not. If it is a prime number, the model will run 1-D decomposition by itself. If the number of tasks is not prime, it can be run as either a 1-D or 2-D decomposition. Since any given non-prime number can have several decompositions, it would be good to know the performance of all the decompositions so that an optimal decomposition can be selected.

Figure 6 gives examples of the performance for several different possible decompositions on the HALEM, the IBM-SP, and the CHAPMAN. There are 7 different decompositions for 64 tasks: 1x64, 2x32, 4x16, 8x8, 16x4, 32x2 and 64x1 given as the number of slices in the x-direction times the number of slices in the y-direction. In the figure, the seven groups of bars represent the seven decompositions. Each group of bar plot has three parts, the left part of each bar is for HALEM, the central part for the IBM-SP, and the right part for CHAPMAN. From these platforms, there are two major conclusions: 1) the closer the numbers of the two slices are, the better the performance and 2) a decomposition using a smaller number of slices in x than in y is better than the reverse. Hence, 8x8 is the best then 4x16 (16x4) followed by 2x32 (32x2) and finally 1x64 (64x1). And 4x16 is better than 16x4; 2x32 is better than 32x2; and 1x64 is better than 64x1.

The difference between the best decomposition (8x8) and the worst decomposition (64x1) is significant. Hence, having the best decomposition for any given number of tasks should be the default configuration, though the model is designed to be flexible. The reason for the significant differences comes from the combination of array indices and loops for computations, the amount of data communication (see Eq. 4), and the architecture of the scalar machine. Nevertheless, this conclusion may not be valid for a vector cluster such as the VPP5000 vector machine as shown in Fig. 3 in Juang and Kanamitsu (2001). In a vector machine, the longer the array length in the x-direction, the shorter the wall-clock time. So it can be expected that 1x64 will achieve the best performance among vector machines. Thus, the decomposition design is platform dependent.

## 4.3 *Wall-clock times for different tasks, platforms and versions*

Following the previous two sub-sections, a short integration period using the best decomposition can be used to evaluate the performance of a number of different tasks, platforms and versions of the model. Figures 7, 8, and 9 show the wall-clock time with respect to number of tasks for model dimensions of 256x256x34 using (a) compressible and (b) anelastic versions of the GCE model. Due to the geometric increments in the number of tasks, a logarithmic scale is applied to both axes. The numbers of tasks along with their decomposition used in these figures are 1, 4 (2x2), 16 (4x4), 32 (4x8), 64 (8x8), 128 (8x16), 256 (16x16), and 512 (16x32). The symbol "X" in the figure indicates the location of the wall-clock time for each number of tasks, and the number beneath the symbol "X" is the value of the wall-clock time. In order to show the relative performance, one solid line, one dotted curve, and one dashed curve are draw in each plot along with the individual runs. They represent the theoretical wall-clock time for numbers of tasks between 1 and 512. The solid line represents perfect performance, which is a line based on wall-clock time for a single task divided by the given number of tasks between 1 and 512. The dotted and dashed curves indicate 99% and 95% parallelization, respectively. They can be obtained from

$$T(n) = T(1)(1-p) + \frac{T(1)p}{n} + c(n) \qquad (5)$$

where c(n) is the communication cost (which is neglected to simplify discussion), T(n) the wall-clock time for n number of tasks, T(1) the wall-clock time for a single task (which is the largest value shown on the y axis for each plot), and p the percentage of parallelization in. When p = 100%, T(n) is the solid line; when p = 99%, T(n) is the dotted curve; and when p = 95% , T(n) is the dashed curve. The perfect line and curves here are drawn for the case of c(n) = 0.

Figure 7 shows the wall-clock time results for HALEM for (a) the compressible version and (b) the anelastic version of the GCE model. The results show a similar trend. The performance is close to 95% for 4 tasks, 99% for 16 tasks, almost perfect parallelization for 128 tasks, then returns to 99%. Figure 8 is the same as Fig. 7 except that it is for the IBM-SP. Wall-clock times are 99% parallelization. Strict speaking, two versions performed very similar; close to 95% for 4 tasks, 99% for 64 tasks, and less than 99% for up to 512 tasks. Figure 9 is the same as Fig. 7 except for CHAPMAN. The results show both versions of the model perform excellently in parallelization between 16 and 256 tasks.

Comparing different versions of the model on the different platforms, the HALEM, the IBM-SP, and CHAPMAN, reveals: 1) the compressible version is faster for a single cpu or small number of tasks, 2) the anelastic version becomes faster than or equal to the compressible version for large numbers of tasks, 3) the IBM-SP is faster than HALEM or CHAPMAN for a small number of tasks but not for a larger number of tasks. Results 1) and 2) are due to additional computations required in the anelastic version, and for 3) indicates that the IBM is a faster computational machine but with slower communications or slower IO. The reason CHAPMAN achieved almost excellent parallelization may be related to its cache advantage for the proper length of array. Furthermore, the results indicates that it is no need to run these dimensions with more than 256 tasks.

## 4.4 *Speedup and efficiency*

In addition to wall-clock time, parallel jobs can also be checked in terms of speedup and efficiency. Wall-clock time can show the related speedup for each configuration for a different number of tasks, but different versions running on different machines cannot be compared on one plot. With speedup and efficiency plots, different versions run on different platforms can be put in one plot, such as in Figs. 10 and 11. The formula for speedup is given as follows:

$$S_p(n) = \frac{T(1)}{T(n)} \tag{6}$$

and substituting Eq. 5 into Eq. 6 yields the theoretical speedup as

$$S_p(n) = \frac{T(1)}{T(1)(1-p) + \frac{T(1)p}{n} + c(n)} \tag{7}$$

where the communication portion c(n) is neglected (as in most published literature) in drawing the theoretical perfect, 99% and 95% speedup in Fig. 8. Following Figs. 6 and 7, the solid line is perfect speedup, the dotted curve 99%, and the dashed curve 95%.

In Fig. 10, compressible and anelastic versions run on the IBM-SP, HALEM, and CHAPMAN are plotted together. General speaking, all of the results are along or better than 99% parallelization. It shows clearly that CHAPMAN provides the best speedup when the number of tasks is larger than 64. The results are less distinctive for smaller numbers of tasks. Also, the anelastic version has a better speedup than the compressible on all platforms. Of course, these results are consistent with the results shown in the previous sub-section (Figs. 7, 8, and 9). Though Fig. 10 shows the relative speedup all together, and it is recognized that the anelastic version has a better speedup than compressible. If the meteorological performance is nearly equal between them, it is faster to use the anelastic version on the IBM-SP, the compressible version on HALEM, and any version on CHAPMAN if 256 tasks are available.

The efficiency of all versions and platforms can be checked in one plot. The formula for efficiency is

$$E(n) = \frac{S_p(n)}{n} 100\% \qquad (8)$$

where E(n) is the efficiency with n tasks. Figure 11 show the efficiency of all versions running on all platforms. The results from the IBM-SP show less distinction between the two versions, and efficiency dropping from around 85% with 4 tasks to 10% with 512 tasks. On the contrary, the results from HALEM show more of a difference between the two versions and no consistent drop in efficiency as with the IBM-SP. Though the compressible version on HALEM drops from 80% efficieny with 32 tasks to 24% with 512 tasks, the anelastic version on HALEM has efficiencies of 91% and 97% for 32 and 128 tasks. The special case of the anelastic version on HALEM having close to perfect speedup was shown in Figs. 9(b) and 10; however, this appears much more clearly in the efficiency plot. Thus it is the most efficient configuration and should be used. Except for 512 tasks, the two versions perform efficiently (above 90%) on CHAPMAN and achieve 120% efficiency with 32 and 64 tasks. Furthermore, in an operational environment, this configuration (128 task anelastic version) should be the choice for routine integrations.

## 4.5 *Different resolutions*

So far, only dimensions of 256x256x34 have been used to test performance. These may not be the optimal dimensions for the GCE model. It can be expected that larger and larger dimensions for higher and higher resolution will be needed. For case of measurement, wall-clock time will be examined for increasing dimension not increasing resolution.

Figure 12 shows wall-clock time for (a) HALEM, (b) the IBM-SP, and (c) CHAPMAN for three different dimensions. They are 256x256x34, 512x512x34 and 1024x1024x34 and used the compressible version with 128 tasks. The number of grid points in 1024x1024x34 is four times larger than that of 512x512x34, which is four times larger than that of 256x256x34. The wall-clock times are marked with the symbol "X" with the associated times written beneath. The solid curve, again, is the idealized performance produced by multiplying the increase in dimensions by the wall-clock time for 256x256x34. The dotted curve is 80% of the idealized performance, and dashed curve is 60%.

In Fig. 11(a), the wall-clock times from the different dimensions correspond to the idealized performance (solid curve) indicating that the size of the dimensions can be increased up to 16 times without losing efficiency on HALEM. In Fig. 11(b), for the IBM-SP, the results fall below the idealized curve. The result for a 4 times increase in the size of the dimensions is located on the 80% curve, and the result for a 16 times increase is located between the dotted and dashed curves at about 70%. For CHAPMAN, Fig. 12(a) shows slightly less wall-clock time for 512x512 but much more than the idealized wall-clock time for 1024x1024. This indicates that the IBM-SP has the best performance with large dimensions followed by HALEM then CHAPMAN.

## *4.6 Reproducibility*

If the computational sequence is the same between different decompositions or no decomposition, the results are the same, we call the decomposition is reproducible. There are some computations, such as obtaining the mean value from the entire domain, that could use a reduced collective MPI call to save wall clock time; however, reduced collective calls may not have the same computational sequence among different decompositions. Thus, a flag was introduced in the code as an option for reproducibility. When the reproducible flag is on, the reduced collective MPI call is not used but the gather MPI routine, which is used to gather data into one task to do the global sum. In this case, the model results for different numbers of tasks are identical, down to binary data comparison, due to the same computational sequences being implemented. When the reproducible flag is off in order to save time, the reduced collective MPI call is used, thus binary results are different for different runs as well as different numbers of tasks. But how different? It has to be checked.

Figure 13 shows total domain rainfall in mm/h (a) between 0 and 24 h and (b) between 72 and 76 h of integration and 256x256x34 grid points. The solid thin curve is from single-cpu integration, and the thick dashed curve is from the non-reproducible option with 64 cpus. Though the binary output between these two experiments is different as in the binary rainfall output, the figure here shows the difference is un-distinguishable, not only initial 24 h of integration but also after 72 h of integration. Figure 14 shows the accumulated rainfall over the domain for a) 1 cpu and b) 64 cpus. The patterns and values are very similar. Thus, the model can produce approximately the same results using different numbers of tasks.

## 5. Conclusions

The concept of MPI implementation with little modification to the original model code as well as having flexibility and reproducibility running any number of tasks, including single-cpu, has been adopted in the GCE model. The GCE model has two different dynamic options: compressible and anelastic[4]. Both versions are decomposed in grid point space in the same fashion as required to have the necessary data exchange by MPI to update the halo and lateral boundaries. In order to have the entire grid points in any given direction available for FFT computation in the anelastic version, a MPI transpose method similar to the data exchange is implemented into the anelastic version for solving the pressure derivative in the horizontal. The preprocessor used in C language programming is adopted into the original code to manage the model options, so that the model can be run in many different combinations of versions, decompositions and numbers of tasks.

Since there are many options available, several sensitivity tests were conducted in order to obtain the optimal values for better performance. Three platforms, HALEM (Dec alpha), the IBM SP (power4) and CHAPMAN (SGI Origin 2000) were selected to test the performance of the MPI implementation. Repeated runs using the same configuration on different platforms showed about 5% difference in wall-clock time. Long- and short-period runs have about the same ratio of wall-clock time to forecast time indicating the performance of the implementation on these platforms is quite consistent.

The performance of the different decompositions supports the theoretical concept that the best performance is with 2-D decomposition using equal numbers of columns and rows. The worst performance is with 1-D decomposition with one row sliced into columns. These results are mainly due to the smaller amount of data exchange in 2-D decomposition and the x-direction grid length in the inner loop. Both versions of the model have wall-clock times and speedups that are along 99% of the theoretical curves up to 256 tasks but not for 512 tasks. It implies that saturated speedup for dimensions of 256x256x34 is about 256

4 Note that many large-eddy simulation (LES, Siebesma *et al.* 2003) models also use the anelastic system.

tasks or more precisely about 128 tasks. The anelastic version has better speedup and efficiency compared to the compressible version due to the greater number of computations. There is highly efficient configuration for the anelastic system with 128 tasks on HALEM that produces wall-clock times about the same as the compressible version with the same number of tasks. CHAPMAN is the slowest machine but has the best speedup and efficiency for dimensions less than 256x256x34. The IBM-SP is the slowest for large numbers of tasks; however, its performance using large dimensions with 128 tasks is the best among these three machines.

Even though the performance of the current MPI implementation is reasonably good, there are still some portions of the code that can be improved using MPI, for example input and output (IO). MPI-IO was not included in the first version of MPI (Gropp *et al.* 1999a). Most IO either uses direct access, an IO server with one IO task, or writes out each portion of a file that is then collected by other program. MPI-2 (Gropp *et al.* 1999b) has several advanced MPI routines including MPI-IO. However, it has not been fully implemented into the current platforms (either partially implemented or not available). Hence, it was not implemented in the current GCE model. The GCE model will need further MPI implementation after MPI-2 is available and stable on most of the platforms in the future. Nevertheless, the concept of the current MPI implementation with a C preprocessor provides a viable mechanism for model development giving the GCE model the capability to plug/un-plug MPI and/or any future advanced package to easily keep up with the rapidly changing computer architectures.

**Acknowledgment**

This work is supported both by the Atmospheric Dynamics and Thermodynamics Program under NASA headquarters and by the NASA Tropical Rainfall Measuring Mission (TRMM). The authors are grateful to Dr. R. Kakar at NASA headquarters for his support of Goddard Cumulus Ensemble (GCE) model development.

Acknowledgment is also made to Dr. T. Lee at NASA HQ and Dr. T. Clune at NASA/NCCS. The NASA/Goddard Space Flight Center, NCEP, and AMES are acknowledged for computer time used in this research.

## References

Anderson, J. R., K. K. Droegemeier and R. B. Wilhelmson, 1985: Simulation of the thunderstorm subcloud environment. **Preprint**, *14th Conf. on Severe Local Storms*. 147-150.

Baker, R. D., B. H. Lynn, A. Boone, W.-K. Tao and J. Simpson, 2001: The influence of soil moisture, coastline curvature, and the land-breeze circulation on sea-breeze initiated precipitation. *J. of Hydrometeorology*, **2**, 193-211.

Chen, J-.P., and D. Lamb, 1999: Simulation of cloud microphysical and chemical processes using a multi-component framework. Part II: Microphysical evolution of a wintertime orographic cloud. *J. Atmos. Sci.*, 56, 2293-2312.

Chou, M.-D., and M. J. Suarez, 1999: A shortwave radiation Parameterization for atmospheric studies. 15, NASA/TM-104606. pp40.

Chou, M.-D., K.-T. Lee, S.-C. Tsay, and Q. Fu, 1999: Parameterization for cloud longwave scattering for use in atmospheric models. *J. Climate*, **12**, 159-169.

Ferrier, B. S., 1994: A double-moment multiple-phase four-class bulk ice scheme. Part I: Description. *J. Atmos. Sci.*, **51**, 249-280.

Ferrier, B. S., W.-K. Tao, and J. Simpson, 1995: A double-moment multiple-phase four-class bulk ice scheme. Part II: Simulations of convective storms in different large-scale environments and comparisons with other bulk parameterizations. *J. Atmos. Sci.*, **52**, 1001-1033.

Ferrier, B. S., J. Simpson and W.-K. Tao, 1996: Factors responsible for different precipitation efficiencies between midlatitude and tropical squall simulations. *Mon. Wea. Rev..* **124**, 2100-2125.

GEWEX Cloud System Study (GCSS), 1993: *Bull. Amer. Meteor. Soc.,* **74**, 387-400.

Grabowski, W.W. and Moncrieff M. W., 2001: Large-scale organization of tropical convection in two-dimensional explicit numerical simulations. *Quart. J. Roy. Meteor. Soc.*, **127**, 445-468.

Gropp, W., E. Lusk, and A. Skjellum, 1999a: Using MPI: portable parallel programming with the message-passing interface, 2nd ed. The MIT Press, Cambridge, Massachusetts, 371 pp.

Gropp, W., E. Lusk, and R. Thakur, 1999b: Using MPI-2: advanced features of the message-passing interface. The MIT Press, Cambridge, Massachusetts, 382 pp.

Ikawa, M., 1988: Comparison of some schemes for non hydrostatic models with orography. *J. Meteor. Soc. Japan*, **66**, 753-776.

Johnson, D., W.-K. Tao, J. Simpson, and C.-H. Sui, 2002: A study of the response of deep tropical clouds to large-scale processes, Part I: Model set-up strategy and comparison with observation, *J. Atmos. Sci.*, **59**, 3492-3518.

Johnson, K. W., J. Bauer, G. A. Riccardi, K. K. Droegemeier, and M. Xue, 1994: Distributed processing of a regional prediction model. Mon. Wea. Rev., 122, 2558-2572.

Juang, H.-M. H., and M. Kanamitsu, 2001: The computational performance of the NCEP seasonal forecast model on FUJITSU VPP5000 at ECMWF. *Developments in Tera*

*Computing, proceedings of the ninth ECMWF workshop on the use of high performance computing in Meteorology, Reading, UK, 13-17 November, 2000,* 338 – 347.

Juang, H.-M. H., C.-H, Shiao, and M.-D. Cheng, 2003: The Taiwan Central Weather Bureau Regional Spectral Model for seasonal prediction: multi-parallel implementation and preliminary results. *Mon. Wea. Rev.*, **131**, 1832-1847.

Khain, A. P., M. Ovtchinnikov, M. Pinsky, A. Pokrovsky, and H. Krugliak, 2000: Notes on the state-of-the-art numerical modeling of cloud microphysics. *Atmosph. Res.*, **55**, 159-224.

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

Lang, S., W.-K. Tao, J. Simpson and B. Ferrier, 2003: Modeling of convective-stratiform precipitation processes: Sensitivity to partitioning methods, *J. Appl. Meteor.* **42**, 505-527.

Lau, K. M., C. H. Sui, and W.-K. Tao, 1993: A preliminary study of the tropical water cycle and its sensitivity to surface warming. *Bull. Amer. Meteor. Soc.*, **74**, 1313-1321.

Lau, K. M., C. H. Sui, M.-D. Chou, and W.-K. Tao, 1994: An enquiry into the cirrus-cloud thermostat effect for tropical sea surface temperature. *Geophys. Res. Lett.*, **21**, 1157-1160.

Lin, Y.-L., R. D. Farley and H. D. Orville, 1983: Bulk parameterization of the snow field in a cloud model. *J. Clim. Appl. Meteor.*, **22**, 1065-1092.

Lynn, B.H., W.-K. Tao, P.J. Wetzel, 1998: A study of landscape generated deep moist convection. *Mon. Wea. Rev.*, **126**, 928-942.

Lynn, B. H., and W.-K. Tao, 2001: A parameterization for the triggering of landscape generated moist convection, Part II: Zero order and first order closure, *J. Atmos. Sci.,* **58**, 593-607.

Lynn, B. H., and W.-K. Tao, and F. Abramopoulos, 2001: A parameterization for the triggering of landscape generated moist convection, Part I: Analyses of high resolution model results, *J. Atmos. Sci.,* **58,** 575-592

Ogura, Y., and N. A. Phillips, 1962: Scale analysis of deep and shallow convection in the atmosphere. *J. Atmos. Sci.*, **19**, 173-179.

Purnell, D. K., and M. J. Revell, 1995: Field-object design of a numerical weather prediction model for uni- and multiprocessors. Mon. Wea. Rev., 123, 401-429.

Randall, D.A., J. Curry, P. Duynkerke, S. K. Krueger, M. W. Moncrieff, B. Ryan, D. O. Starr, M. Miller, W. Rossow, G. Tseliudis, B. A. Wielikci, 2003: The GEWEX cloud system study: A view from 2001. *Bull. Amer. Meteor. Soc.*, **84**, 455-469.

Rutledge, S. A., and P. V. Hobbs, 1984: The mesoscale and microscale structure and organization of clouds and precipitation in mid-latitude clouds. Part XII: A diagnostic modeling study of precipitation development in narrow cold frontal rainbands. *J. Atmos. Sci.*, **41**, 2949-2972.

Scala, J. R., M. Garstang, W.-K. Tao, K. E. Pickering, A. M. Thompson, J. Simpson, V. W. J. H. Kirchoff, E. V. Browell, G. W. Saschse, A. L. Torres, G. L. Gregory, R. A. Rasmussen, and M. A. K. Khalil, 1990: Cloud draft structure and trace gas transport. *J Geophys. Res.*, **95**, 17015-17030.

Siebesma, A. P., C. S. Bretherton, A. Brown, A. Chilond, J. Cuxart, P. G. Duynkerke, H. Jiang, M. Khairoutdinov, D. Lewellen, C.-H. Moeng, E. Sanchez, B. Stevens and D. E. Stevens, 2003: A large eddy simulation intercomparison study of shallow cumulus convection. *J. Atmos. Sci.*, **60**, 1201-1219.

Simpson, J., and W.-K. Tao, 1993: The Goddard Cumulus Ensemble Model. Part II: Applications for studying cloud precipitating processes and for NASA TRMM. *Terrestrial, Atmospheric and Oceanic Sciences*, **4**, 73-116.

Simpson, J., C. Kummerow, W.-K. Tao, and R. Adler, 1996: On the Tropical Rainfall Measuring Mission (TRMM), *Meteor. and Atmos. Phys.* , **60**, 19-36, 1996.

Skalin, R., 1997a: Scalability of parallel gridpoint limited-area atmospheric models. Part I: Explicit time-integration schemes. *J. Atmos. Oceanic Technol.*, **14**, 427-441.

Skalin, R., 1997b: Scalability of parallel gridpoint limited-area atmospheric models. Part II: Semi-implicit time-integration scheme. *J. Atmos. Oceanic Technol.*, **14**, 442-455.

Smolarkiewicz, P. K., and W. W. Grabowski, 1990: The multidimensional positive advection transport algorithm: Nonoscillatory option. *J. Comput. Phys.*, 86, 355-375.

Soong, S.-T., and Y. Ogura, 1980: Response of tradewind cumuli to large-scale processes. *J. Atmos. Sci.*, 37, 2035-2050.

Soong, S.-T., and W.-K. Tao, 1980: Response of deep tropical clouds to mesoscale processes. *J. Atmos. Sci.*, 37, 2016-2036.

Sui, C.-H., and K.-M. Lau, W.-K. Tao, and J. Simpson, 1994: The tropical water and energy cycles in a cumulus ensemble model. Part I: Equilibrium climate. *J. Atmos. Sci.*, **51**, 711-728.

Sui, C.-H., and K.-M. Lau, and X. Li, 1998: Convective-radiative interaction in simulated diurnal variations of tropical cumulus ensemble. *J. Atmos. Sci.*, **55**, 2345-2357.

Tao, W.-K., and J. Simpson, 1984: Cloud interactions and merging: Numerical simulations. *J. Atmos. Sci.,* **41**, 2901-2917.

Tao, W.-K., and S.-T. Soong, 1986: A study of the response of deep tropical clouds to mesoscale processes: Three-dimensional numerical experiments. *J. Atmos. Sci.*, **43**, 2653-2676.

Tao, W.-K., J. Simpson, and S.-T. Soong, 1987: Statistical properties of a cloud ensemble: A numerical study. *J. Atmos. Sci.*, **44**, 3175-3187.

Tao, W.-K., and J. Simpson, 1989a: A further study of cumulus interaction and mergers: Three-dimensional simulations with trajectory analyses. *J. Atmos. Sci.*, **46**, 2974-3004.

Tao, W.-K., and J. Simpson, 1989b: Modeling study of a tropical squall-type convective line. *J. Atmos. Sci.*, **46**, 177-202.

Tao, W.-K., and J. Simpson, and M. McCumber, 1989: An ice-water saturation adjustment. *Mon. Wea. Rev.*, **117**, 231-235.

Tao, W.-K., and J. Simpson, S. Lang, M. McCumber, R. Adler and R. Penc, 1990: An algorithm to estimate the heating budget from vertical hydrometeor profiles. *J. Appl. Meteor.,* **29**, 1232-1244.

Tao, W.-K., and J. Simpson, and S.-T. Soong, 1991: Numerical simulation of a sub-tropical squall line over Taiwan Strait. *Mon. Wea. Rev.*, **119**, 2699-2723.

Tao, W.-K., and J. Simpson, 1993: The Goddard Cumulus Ensemble Model. Part I: Model description. *Terrestrial, Atmospheric and Oceanic Sciences,* **4**, 19-54.

Tao, W.-K., and J. Simpson, C.-H. Sui, B. Ferrier, S. Lang, J. Scala, M.-D. Chou, and K. Pickering, 1993a: Heating, moisture and water budgets of tropical and mid-latitude squall lines: Comparisons and sensitivity to longwave radiation. *J. Atmos. Sci.*, **50**, 673-690.

Tao, W.-K., and J. Simpson, S. Lang, J. Simpson and R. Adler, 1993b: Retrieval Algorithms for estimating the vertical profiles of latent heat release: Their applications for TRMM. *J. Meteor. Soc. Japan* , **71**, 685-700.

Tao, W.-K., J. Scala, and J. Simpson, 1995: The effects of melting processes on the development of a tropical and a mid-latitudes squall line. *J. Atmos. Sci.*, **52**, 1934-1948.

Tao, W.-K., 1995: Interaction of parameterized convection and explicit stratiform cloud microphysics. WMO/WCRP-90, Cloud Microphysics Parameterizations in Global Atmospheric General Circulation Models, Ed. D. Randall, 199-210.

Tao, W.-K., S. Lang, J. Simpson, C.-H. Sui and B. Ferrier and M.-D. Chou, 1996: Mechanisms of Cloud-radiation interaction in the tropics and midlatitudes. *J. Atmos. Sci.* **53**, 2624-2651.

Tao, W.-K., J. Simpson, C.-H. Sui, C.-L. Shie, B. Zhou, K. M. Lau, and, M. Moncrieff, 1999: On equilibrium states simulated by Cloud-Resolving Models, *J. Atmos. Sci.,* **56,** 3128-3139.

Tao, W.-K., S. Lang, J. Simpson, W. S. Olson, D. Johnson, B. Ferrier, C. Kummerow and R. Adler, 2000: Vertical profiles of latent heat release and their retrieval in TOGA COARE convective systems using a cloud resolving model, SSM/I and radar data, *J. Meteor. Soc. Japan,* **78**, 333-355.

Tao, W.-K., C.-L. Shie and J. Simpson, 2001a: Comments on The sensitivity study of radiative-convective equilibrium in the Tropics with a convective resolving model, *J. Atmos. Sci.,* **58**, 1328-1333.

Tao, W.-K., S. Lang, W. S. Olson, S. Yang, R. Meneghini, J. Simpson, E. Smith, C. Kummerow, E. Smith and J. Halverson, 2001b: Retrieved Vertical Profiles of Latent Heat Release Using TRMM Products for February 1998, *J. Appl. Meteor.,* **4,** 957-982.

Tao, W.-K., 2003: Goddard Cumulus Ensemble (GCE) model: Application for understanding precipitation processes, *AMS Meteorological Monographs - Cloud Systems, Hurricanes and TRMM.* 103-138.

Tao, W.-K., J. Simpson, D. Baker, S. Braun, M.-D. Chou, D. Johnson, B. Ferrier, A. Khain, S. Lang, B. Lynn, C.-L. Shie, D. Starr, C.-H. Sui, Y. Wang and P. Wetzel, 2003a: Microphysics, Radiation and Surface Processes in a Non-hydrostatic Model. *Meteorology and Atmospheric Physics*, **82**, 97-137.

Tao, W.-K., C.-L. Shie, R. Johnson, S. Braun, J. Simpson, and P. E. Ciesielski, 2003b: Convective Systems over South China Sea: Cloud-Resolving Model Simulations. *J. Atmos. Sci.*, **60**, 2929-2956.

Tao, W.-K., D. Johnson, C.-L. Shie, and J. Simpson. 2004: Atmospheric energy budget and large-scale precipitation efficiency of convective systems during TOGA COARE, GATE, SCSMEX and ARM: Cloud-resolving model simulations, *J. Atmos. Sci.*, (in press).

Thompson, A. M., W.-K. Tao, K. E. Pickering, J. Scala, and J. Simpson, 1997: Tropical deep convection and ozone formation, *Bull. Amer. Meteor. Soc.*, **78**, 1043-1054.

Wang, Y., W.-K. Tao, and J. Simpson, 1996: The impact of ocean surface fluxes on a TOGA COARE convective system. *Mon. Wea. Rev.*, **124**, 2753-2763.

Wang, Y., W.-K. Tao, J. Simpson, and S. Lang, 2003: The sensitivity of tropical squall lines (GATE and TOGA COARE) to surface fluxes: 3-D Cloud resolving model simulations, *Q. J. R. Met. Soc.*, **129**, 987-1007.

Wu, X., W. W. Grabowski and M. W. Moncrieff, 1998: Long-term behavior of cloud systems in TOGA COARE and their interactions with radiative and surface processes. Part I: Two-dimensional modeling study. *J. Atmos. Sci.*, **55**, 2693-2714.

## Figures Captions

Fig. 1 Radar observations (dBZ) from SCSMEX (upper left panels), and KWAJEX (lower left panels). Linear cloud systems typically propagated from west to east in SCSMEX. Less organized and short-lived clouds/cloud systems dominated in KWAJEX.

Fig. 2 Schematic diagram showing decomposition in grid-point space for an example using 12 tasks over the entire model domain and a partial domain with a halo-region (regime) for task 4. It shows a similar structure between the entire and partial domains if the dashed lines are eliminated from the entire domain.

Fig. 3 Schematic diagram showing data exchange in **a)** the east-west direction and **b)** the north-south direction for grid-point space. The dashed lines indicate the inner-border of the halo regimes, and the arrows indicate the direction of data movement. The thin solid lines show the area of data with other lines indicating where it is to be moved. The lateral boundary conditions determine which data to move and where using two arrows at the lateral boundaries.

Fig. 4 Transpose of data between different decompositions for different grid-point and spectral spaces for Fourier transform in the anelastic version of the model. Thin arrows indicate Fast Fourier Transforms between grid and spectral spaces, thick

arrows MPI transposes needed in any decomposition, and hollow arrows MPI transposes when 2-D decomposition is used.

Fig. 5 Schematic diagram showing an MPI transpose using 2-D decomposition.

Fig. 6 Bar plot for different decompositions using the compressible version of the model. The left bar is for results on HALEM, the central bar on the IBM-SP, and the right bar on CHAPMAN. The numbers on the x-axis indicate the decomposition in terms of the number of columns times the number of rows.

Fig. 7 Wall-clock time in seconds with respect to number of tasks for **(a)** the compressible and **(b)** the anelastic versions with 256 grid points in the x- and y-directions and 34 in the z direction on HALEM. The solid line indicates perfect parallelization based on wall-clock time for a single task. Wall-clock time is shown on the y-axis. The dotted curve indicates 99% parallelization, and the dashed curve indicates 95%. The numbers next to the symbol 'X' are the actual wall-clock times.

Fig. 8 Same as Fig. 7 except for the IBM-SP environment.

Fig. 9 Same as Fig. 7 except for the CHAPMAN environment.

Fig. 10 Speed-up for the compressible and anelastic versions of the model with 256 grid points in the x- and y-directions and 34 in the z-direction on HALEM, the IBM-SP and CHAPMAN. The solid line indicates perfect speedup, the dotted curve 99%, and the dashed curve 95%.

Fig. 11 Efficiency for the compressible and anelastic versions of the model with 256 grid points in the x- and y-directions and 34 in the z-direction on HALEM, the IBM-SP and CHAPMAN.

Fig. 12 Wall-clock time in seconds for different dimensions of the model horizontal domain for **(a)** the IBM-SP and **(b)** HALEM for the compressible version of the model. The solid curve indicates wall-clock times linearly extrapolated from the 256x256 model domain, the dotted curve indicates 80% of the performance of solid curve and the dashed curve 60% in each environment.

Fig. 13 Rainfall amount over a given domain in mm/hr with respect to time evolution from 72 to 76 hr of integration. Solid curve indicates results from a single task and dashed curve from arbitrary multiple tasks.

Fig. 14 Accumulated rainfall amount in mm/hr after 76 hr of integration for results from **(a)** 1 cpu and **(b)** 64 tasks on HALEM.

## Table Captions

Table1 Major highlights of cloud modeling development over the past four decades.

Table 2 The main characteristics of the Goddard Cumulus Ensemble (GCE) model.

Table 3 Wall-clock times from repeated runs.

Table 4 Wall-clock times for 3-day and 4-hour runs.

**Table 1**

| | **Highlights** |
|---|---|
| **1960's** | Loading, Buoyancy and Entrainment |
| **1970's** | Slab- vs axis-symmetric models<br><br>Cloud Seeding<br><br>Super Cell Dynamics<br><br>Cloud Dynamics & Warm Rain<br><br>Wind Shear Effect on Cloud Organization |
| **1980's** | Ensembles of Clouds - Cumulus Parameterization<br><br>Cloud Interactions and Mergers<br><br>Ice Processes<br><br>Squall Lines<br><br>Convective and Stratiform<br><br>Wind Shear and Cool Pools<br><br>Gravity Waves and Density Currents<br><br>Large-Scale and Cloud-Scale Interactions<br><br>Cloud Radiation Interaction |
| **1990's** | 2D vs 3D<br><br>Land and Ocean Processes<br><br>Multi-scale Interactions<br><br>Cloud Chemistry<br><br>Process modeling - Climate Variation Implications<br><br>GEWEX Cloud System Study (GCSS)<br><br>Coupling with Microwave Radiative Transfer Models for TRMM |

## Table 2

| Parameters/Processes | GCE Model |
|---|---|
| Dynamics | Anelastic or compressible<br>2-D (slab- and axis-symmetric) and 3-D |
| Vertical Coordinate | Z (p) |
| Explicit Convective Processes | 2-class water & 2-moment<br>4-class ice, 2- or 3-class ice<br>Spectral-Bin Microphysics * |
| Implicit Convective Processes | Betts & Miller or Kain & Fritsch |
| Numerical Methods | Positive Definite Advection for Scalar Variables;<br>4-th Order for Dynamic Variables |
| Initialization | Initial Conditions with Forcing<br>from Observations/Large-Scale Models |
| FDDA | Nudging |
| Radiation | k-distribution and four-stream discrete-ordinate scattering<br>(8 bands)<br>Explicit Cloud-radiation Interaction |
| Sub-Grid Diffusion | TKE (1.5 order) |
| Topography | Sigma-z(p)** |
| Two-Way Interactive Nesting | Radiative-Type* |
| Surface Energy Budget | 7-Layer Soil Model (PLACE)<br>CLM - LIS<br>TOGA COARE Flux Module |
| Parallelization | OPEN-MP and MPI |

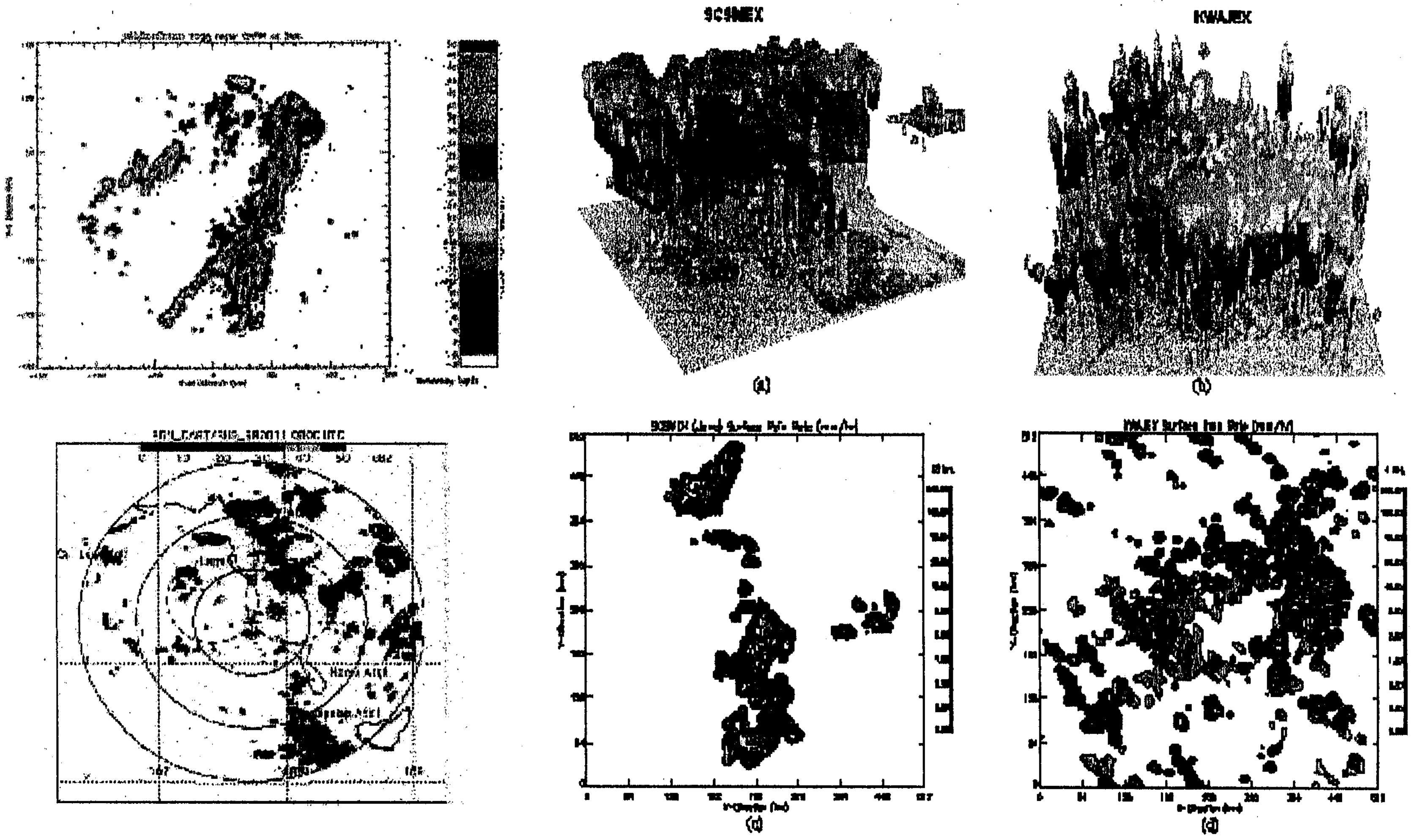
**Table 3**

| Platform | Dimension | Task | Integration | Wall-clock time |
|---|---|---|---|---|
| *HALEM* | 256x256x34 | 64(8x8) | 4 hours | 854 sec<br>858 sec<br>865 sec<br>875 sec<br>879 sec |
| *IBM-SP* | 256x256x34 | 64(8x8) | 4 hours | 1085 sec<br>1107 sec<br>1114 sec<br>1139 sec<br>1159 sec |
| *CHAPMAN* | 256x256x34 | 64(8x8) | 1 hour | 334 sec<br>339 sec<br>340 sec<br>342 sec |

**Table 4**

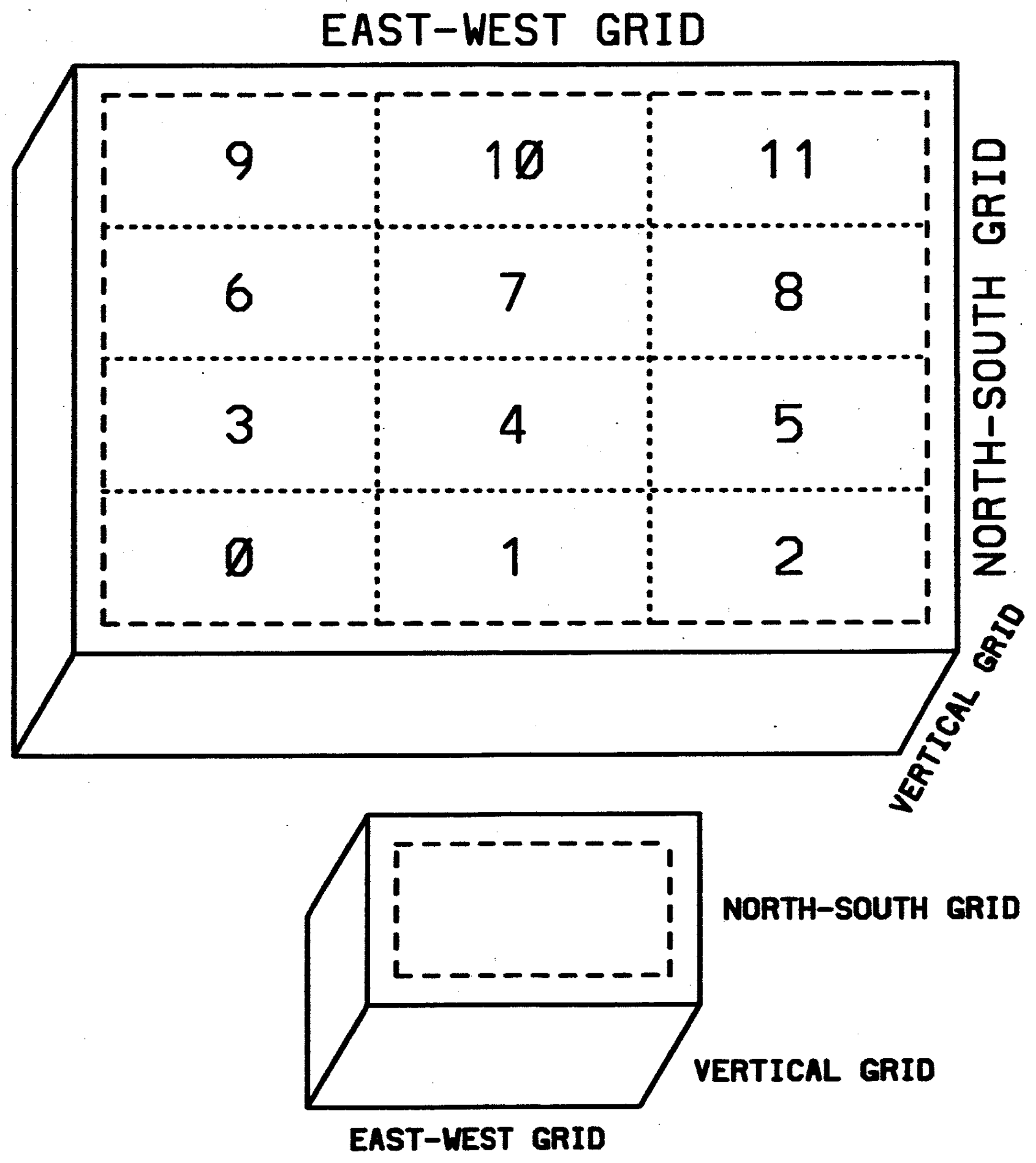
| Platform | Dimension | Task | Integration | Wall-clock time |
|---|---|---|---|---|
| *HALEM* | 256x256x34 | 128(8x16) | 3 days<br>4 hours | 7803 sec<br>460 sec |
| *IBM-SP* | 256x256x34 | 128(8x16) | 3 days<br>4 hours | 13680 sec<br>754 sec |

## SCSMEX (S. China Sea) and KWAJEX (W. Pacific)



**Radar Observations (dBZ) from SCSMEX (upper left panel)and KWAJEX (lower left panels). Linear cloud systems typically propagated from west to east in SCSMEX. Less organized and short-lived clouds/cloud systems dominated in KWAJEX**

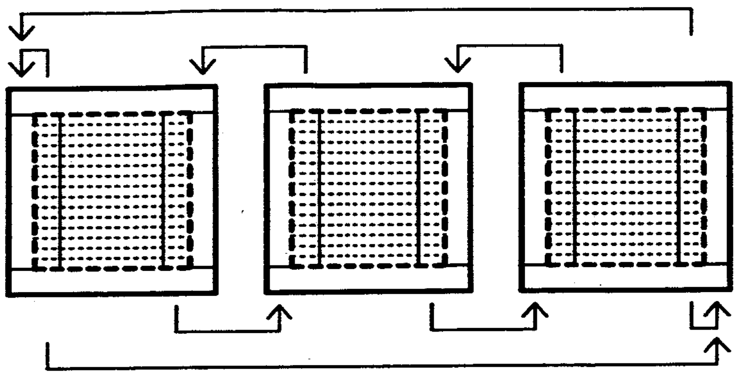
Fig. 1



EAST-WEST GRID
9
10
11
6
7
8
3
4
5
0
1
2
NORTH-SOUTH GRID
VERTICAL GRID
NORTH-SOUTH GRID
VERTICAL GRID
EAST-WEST GRID

Fig. 2

(a)



(b)

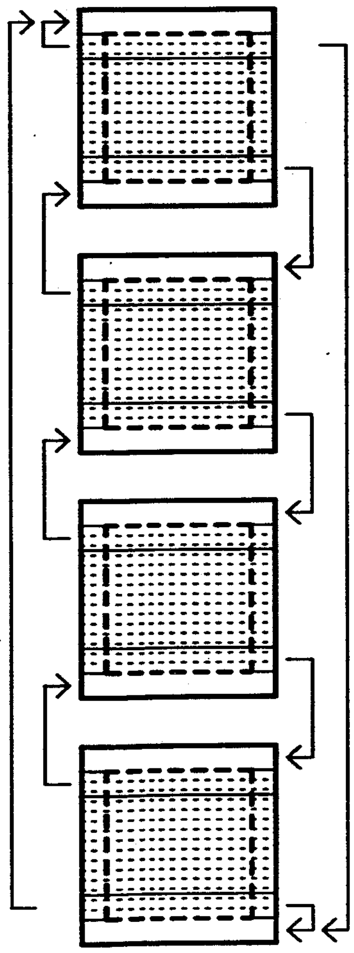
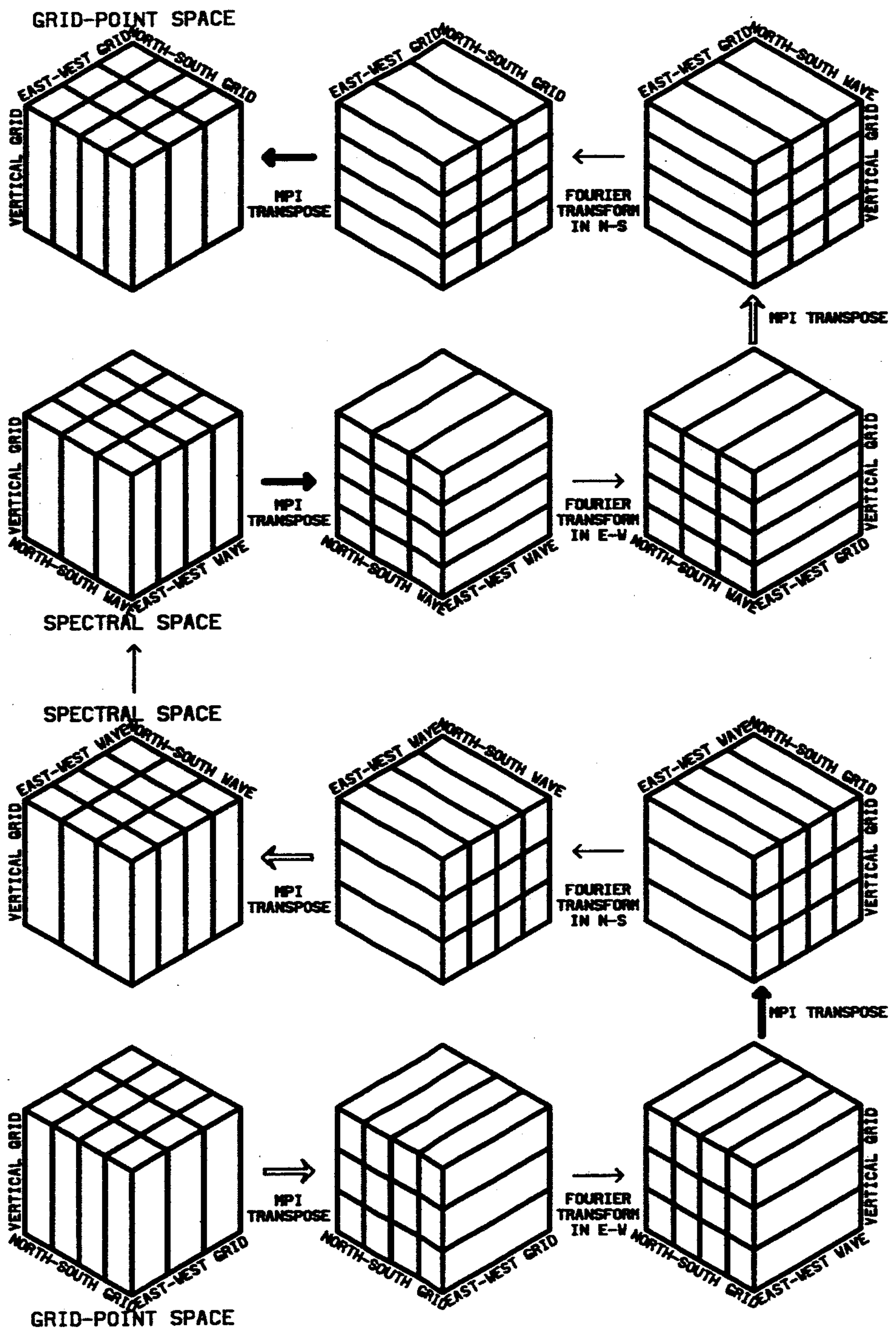


Fig. 3



GRID-POINT SPACE
EAST-WEST GRID
NORTH-SOUTH GRID
VERTICAL GRID
MPI TRANSPOSE
EAST-WEST GRID
NORTH-SOUTH GRID
FOURIER TRANSFORM IN N-S
EAST-WEST GRID
NORTH-SOUTH WAVE
VERTICAL GRID
MPI TRANSPOSE
VERTICAL GRID
NORTH-SOUTH WAVE
EAST-WEST WAVE
MPI TRANSPOSE
NORTH-SOUTH WAVE
EAST-WEST WAVE
FOURIER TRANSFORM IN E-W
NORTH-SOUTH WAVE
EAST-WEST GRID
VERTICAL GRID
SPECTRAL SPACE
SPECTRAL SPACE
EAST-WEST WAVE
NORTH-SOUTH WAVE
VERTICAL GRID
MPI TRANSPOSE
EAST-WEST WAVE
NORTH-SOUTH WAVE
FOURIER TRANSFORM IN N-S
EAST-WEST WAVE
NORTH-SOUTH GRID
VERTICAL GRID
MPI TRANSPOSE
VERTICAL GRID
NORTH-SOUTH GRID
EAST-WEST GRID
MPI TRANSPOSE
NORTH-SOUTH GRID
EAST-WEST GRID
FOURIER TRANSFORM IN E-W
NORTH-SOUTH GRID
EAST-WEST WAVE
VERTICAL GRID
GRID-POINT SPACE

Fig. 4

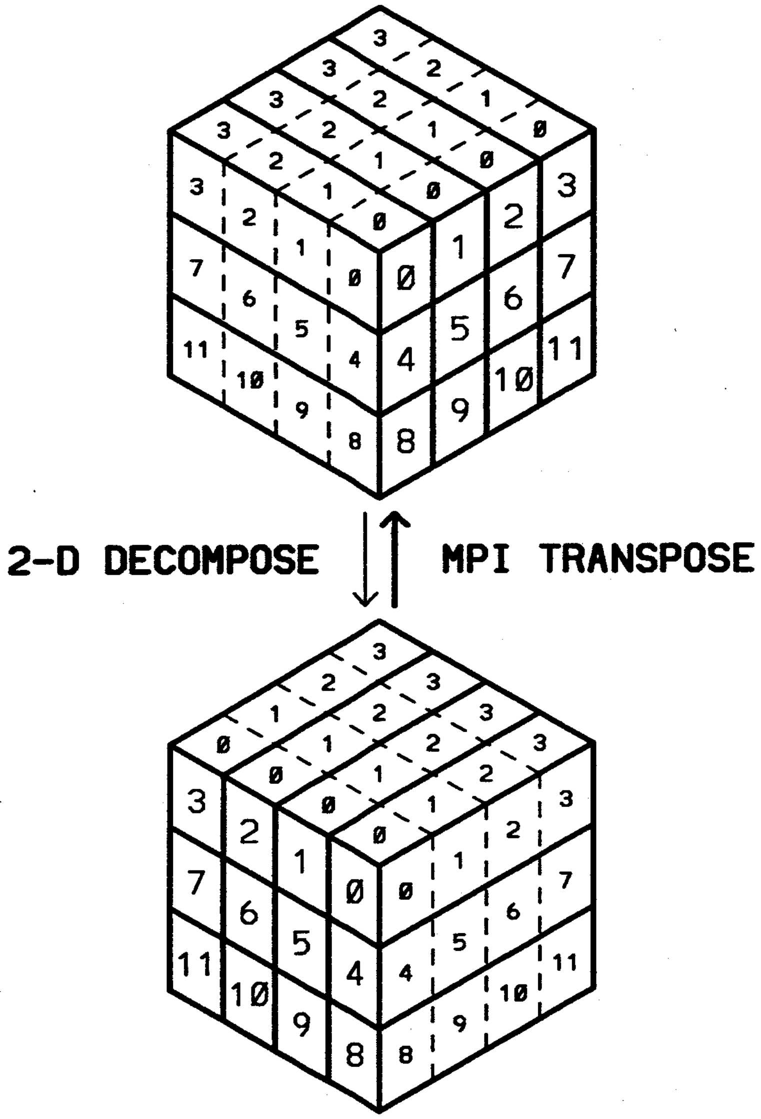


Fig. 5

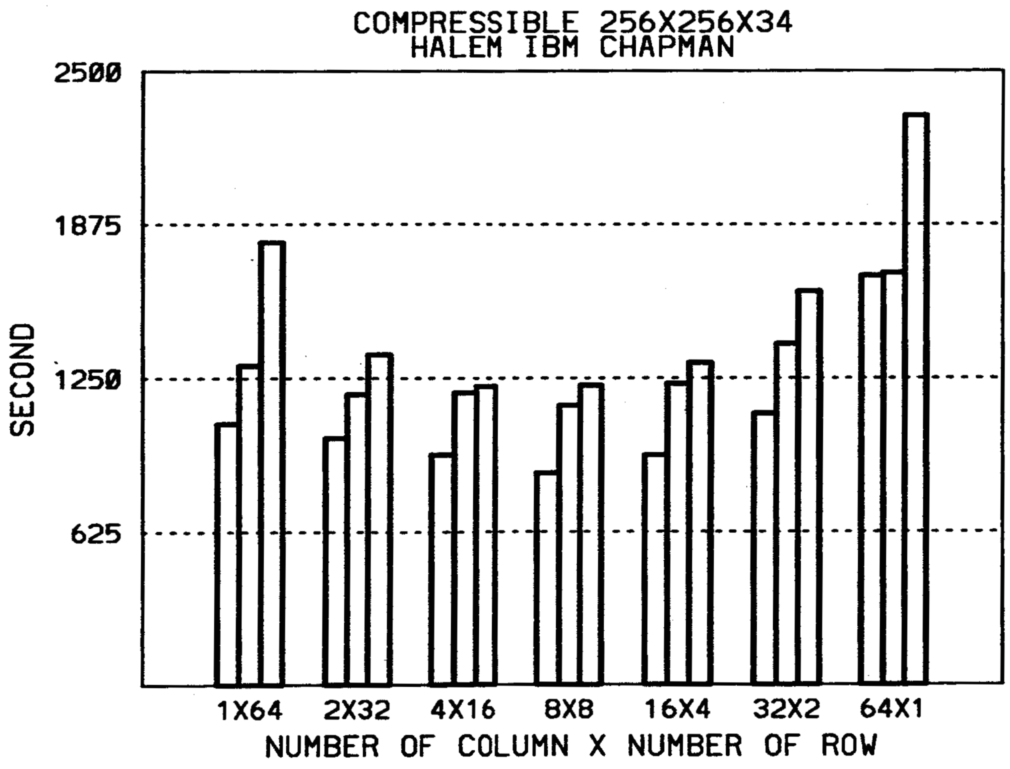
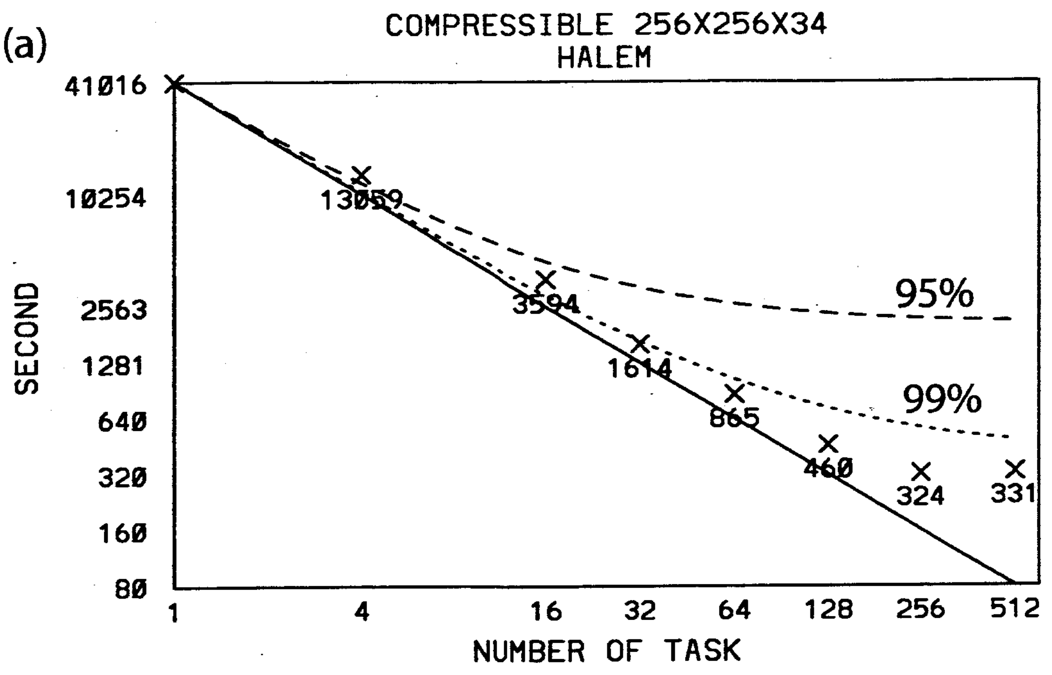

COMPRESSIBLE 256X256X34
HALEM IBM CHAPMAN
2500
1875
1250
625
SECOND
1X64
2X32
4X16
8X8
16X4
32X2
64X1
NUMBER OF COLUMN X NUMBER OF ROW

Fig. 6

(a)


COMPRESSIBLE 256X256X34
HALEM
41016
10254
2563
1281
640
320
160
80
SECOND
13059
3594
1614
865
460
324
331
95%
99%
1
4
16
32
64
128
256
512
NUMBER OF TASK

(b)

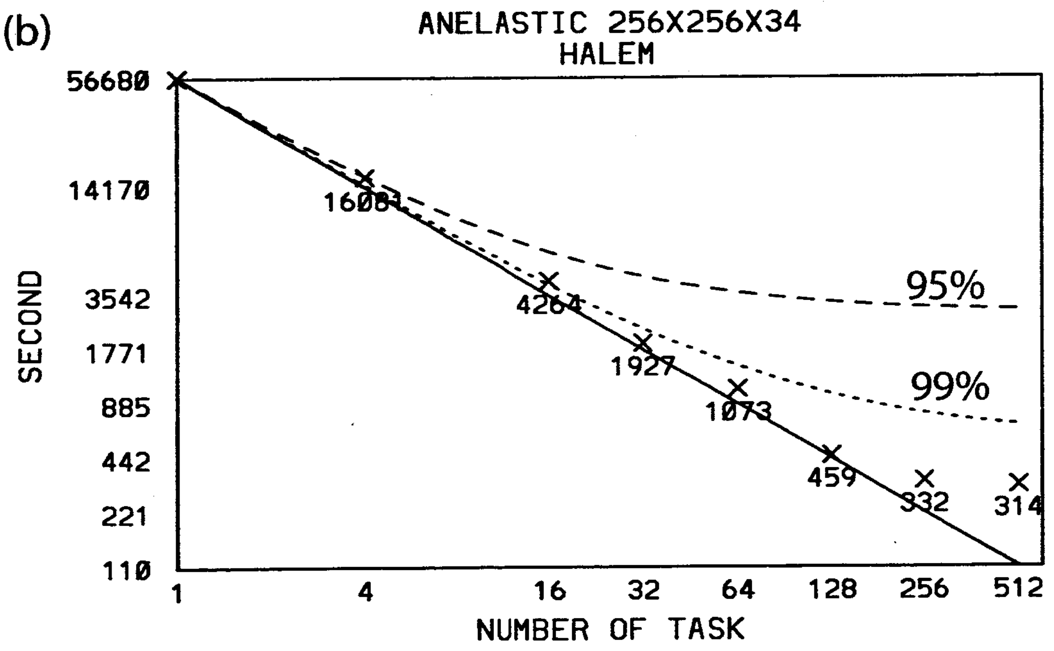
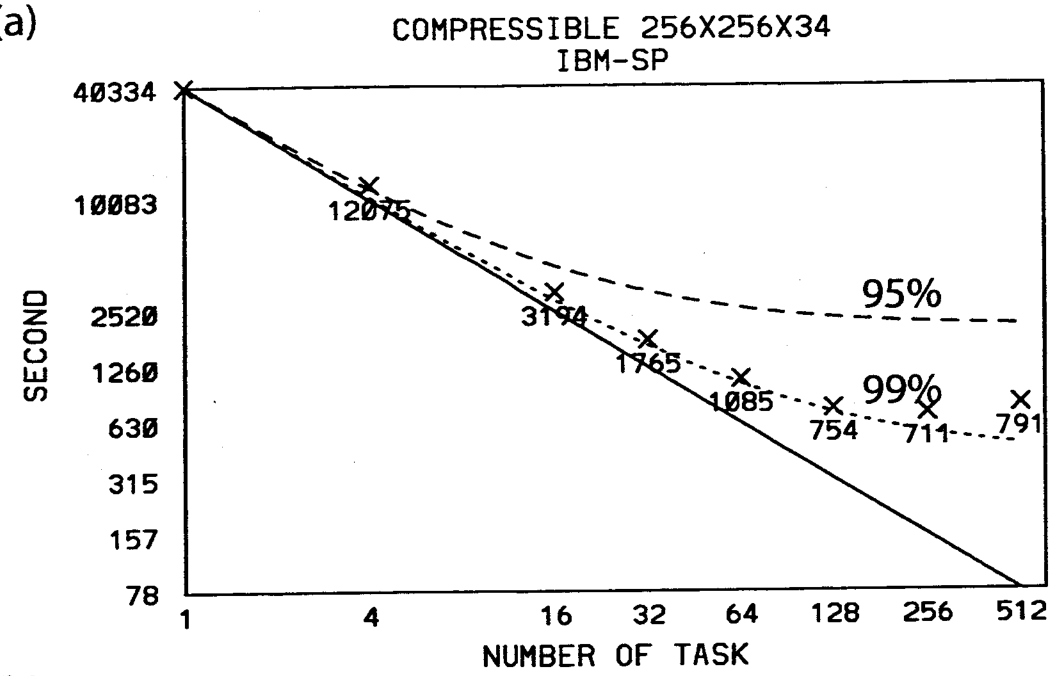

ANELASTIC 256X256X34
HALEM
56680
14170
3542
1771
885
442
221
110
SECOND
16081
4264
1927
1073
459
332
314
95%
99%
1
4
16
32
64
128
256
512
NUMBER OF TASK

Fig. 7

(a)


COMPRESSIBLE 256X256X34
IBM-SP
40334
10083
2520
1260
630
315
157
78
SECOND
12075
3194
1765
1085
754
711
791
95%
99%
1
4
16
32
64
128
256
512
NUMBER OF TASK

(b)

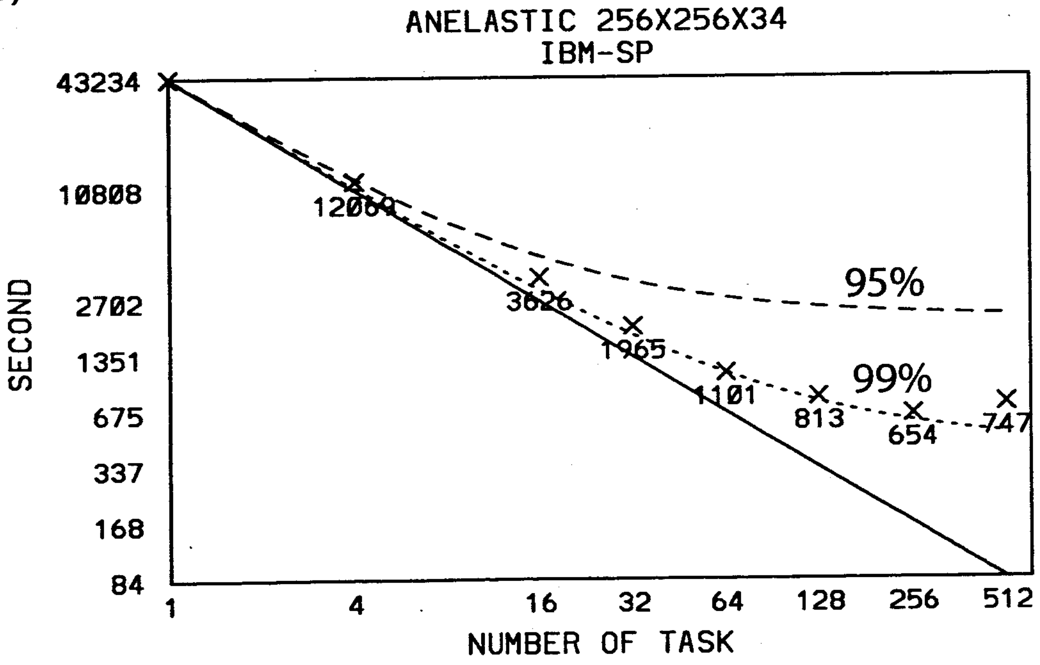
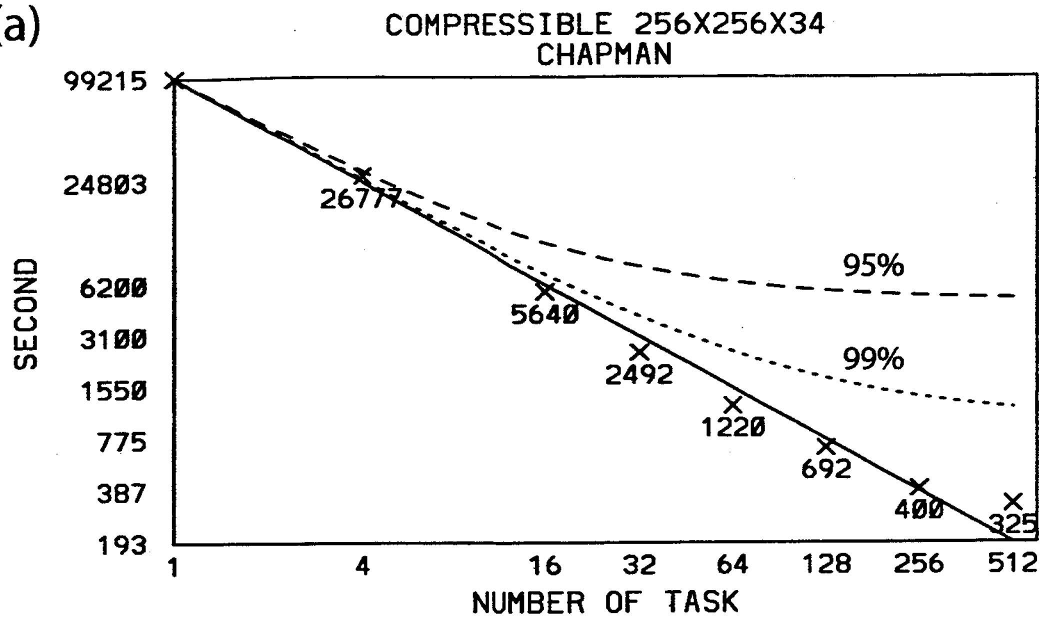

ANELASTIC 256X256X34
IBM-SP
43234
10808
2702
1351
675
337
168
84
SECOND
12069
3626
1965
1101
813
654
747
95%
99%
1
4
16
32
64
128
256
512
NUMBER OF TASK

Fig. 8

(a)


COMPRESSIBLE 256X256X34
CHAPMAN
99215
24803
6200
3100
1550
775
387
193
SECOND
1
4
16
32
64
128
256
512
NUMBER OF TASK
26777
5640
2492
1220
692
400
325
95%
99%

(b)

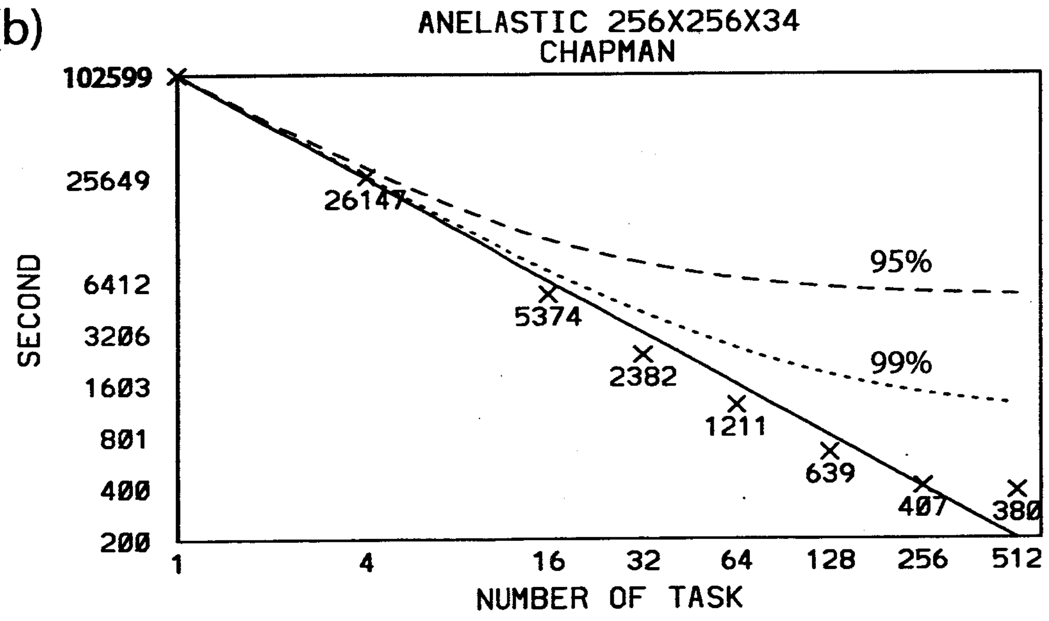
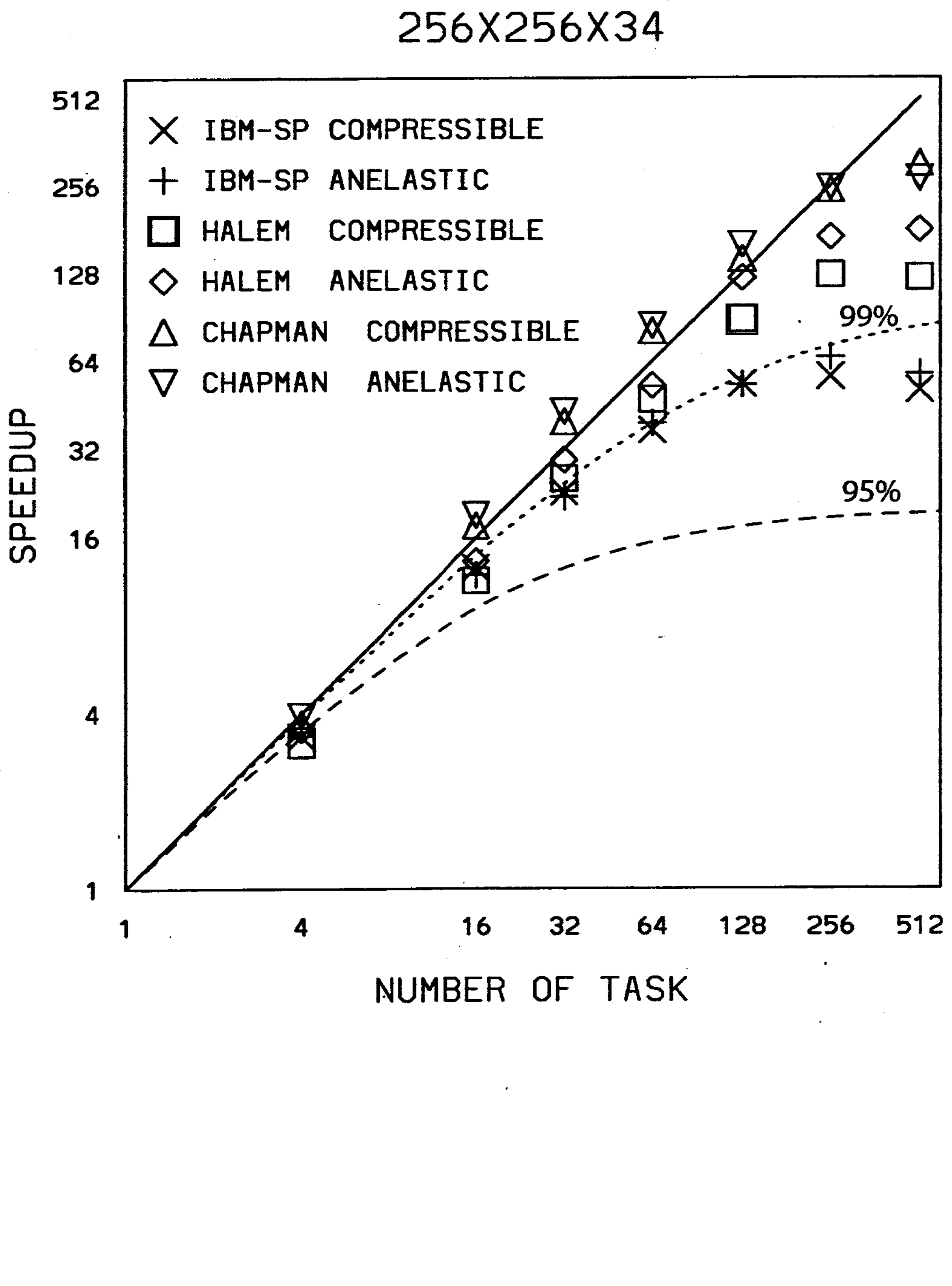

ANELASTIC 256X256X34
CHAPMAN
102599
25649
6412
3206
1603
801
400
200
SECOND
1
4
16
32
64
128
256
512
NUMBER OF TASK
26147
5374
2382
1211
639
407
380
95%
99%

Fig. 9



256X256X34
512
256
128
64
32
16
4
1
SPEEDUP
1
4
16
32
64
128
256
512
NUMBER OF TASK
IBM-SP COMPRESSIBLE
IBM-SP ANELASTIC
HALEM COMPRESSIBLE
HALEM ANELASTIC
CHAPMAN COMPRESSIBLE
CHAPMAN ANELASTIC
99%
95%

Fig. 10

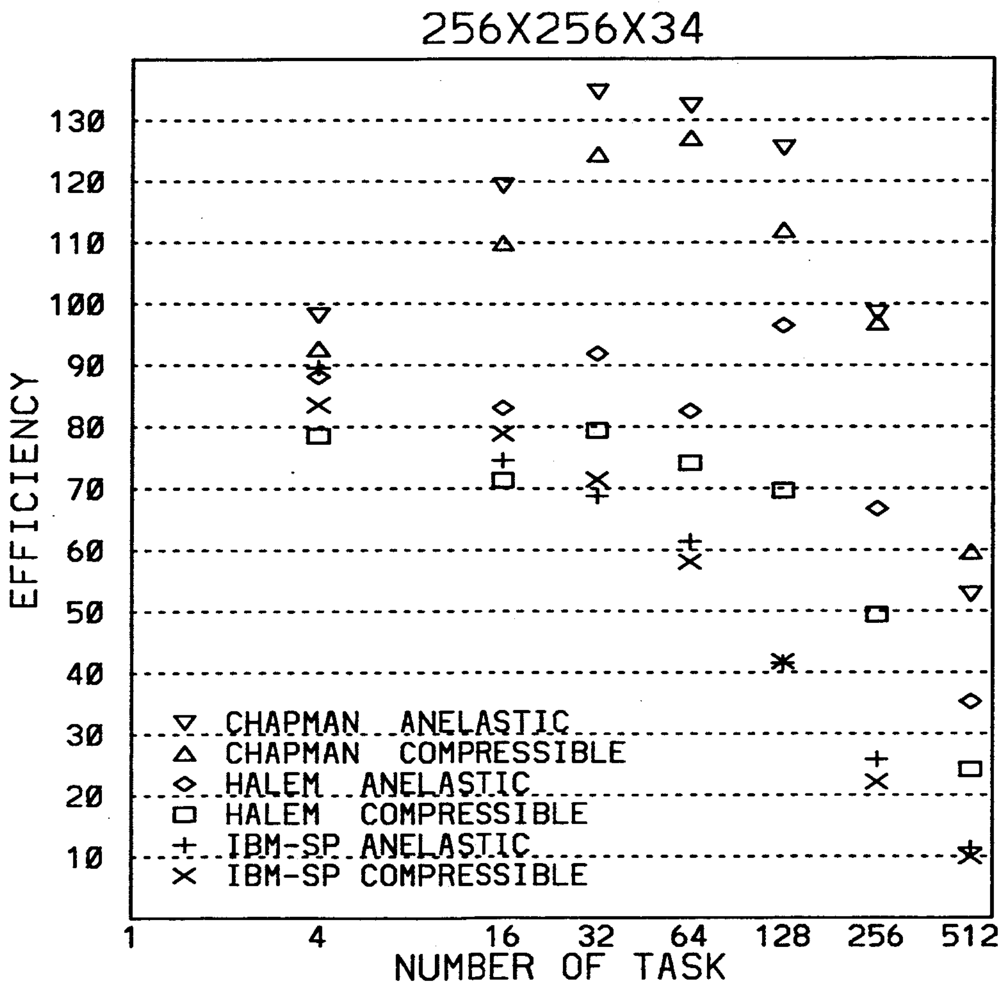

256X256X34
130
120
110
100
90
80
70
60
50
40
30
20
10
EFFICIENCY
1
4
16
32
64
128
256
512
NUMBER OF TASK
▽ CHAPMAN ANELASTIC
△ CHAPMAN COMPRESSIBLE
◇ HALEM ANELASTIC
□ HALEM COMPRESSIBLE
+ IBM-SP ANELASTIC
× IBM-SP COMPRESSIBLE

Fig. 11

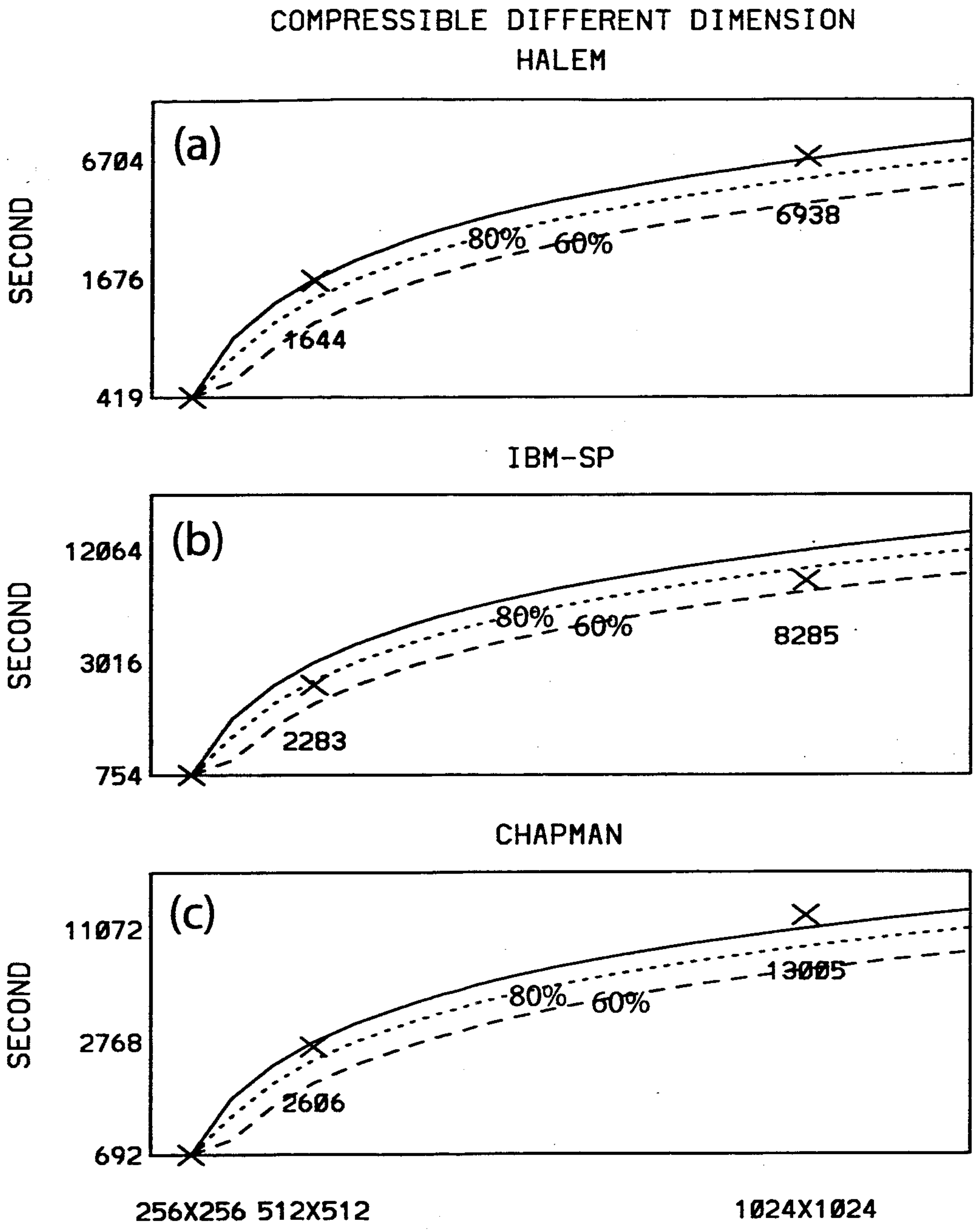
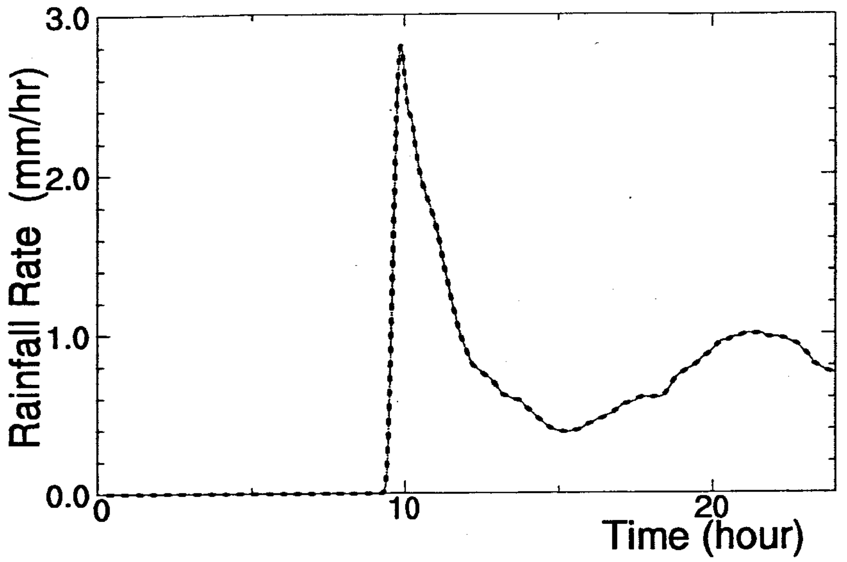

COMPRESSIBLE DIFFERENT DIMENSION
HALEM
(a)
6704
1676
419
SECOND
80%
60%
6938
1644
IBM-SP
(b)
12064
3016
754
SECOND
80%
60%
8285
2283
CHAPMAN
(c)
11072
2768
692
SECOND
80%
60%
13005
2606
256X256
512X512
1024X1024

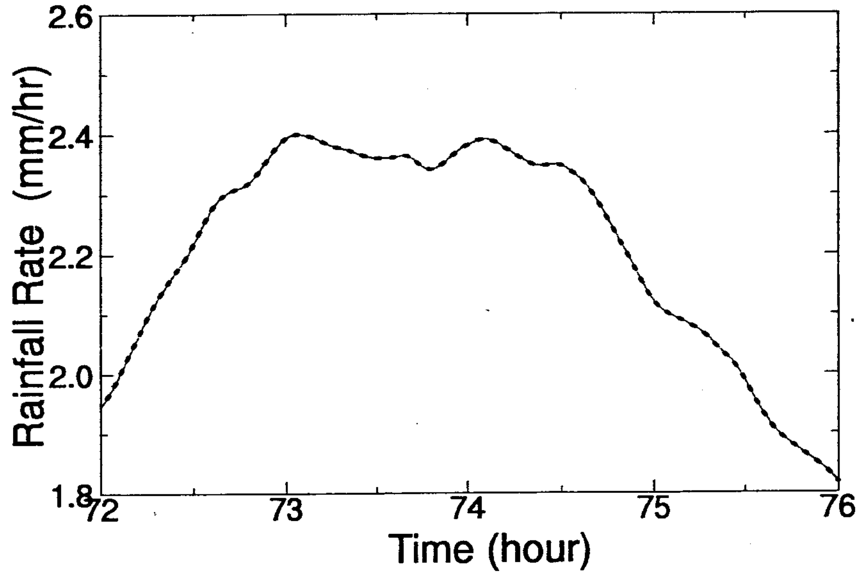
Fig. 12

(a)



Rainfall Rate (mm/hr)
3.0
2.0
1.0
0.0
0
10
20
Time (hour)

(b)



Rainfall Rate (mm/hr)
2.6
2.4
2.2
2.0
1.8
72
73
74
75
76
Time (hour)

Fig. 13

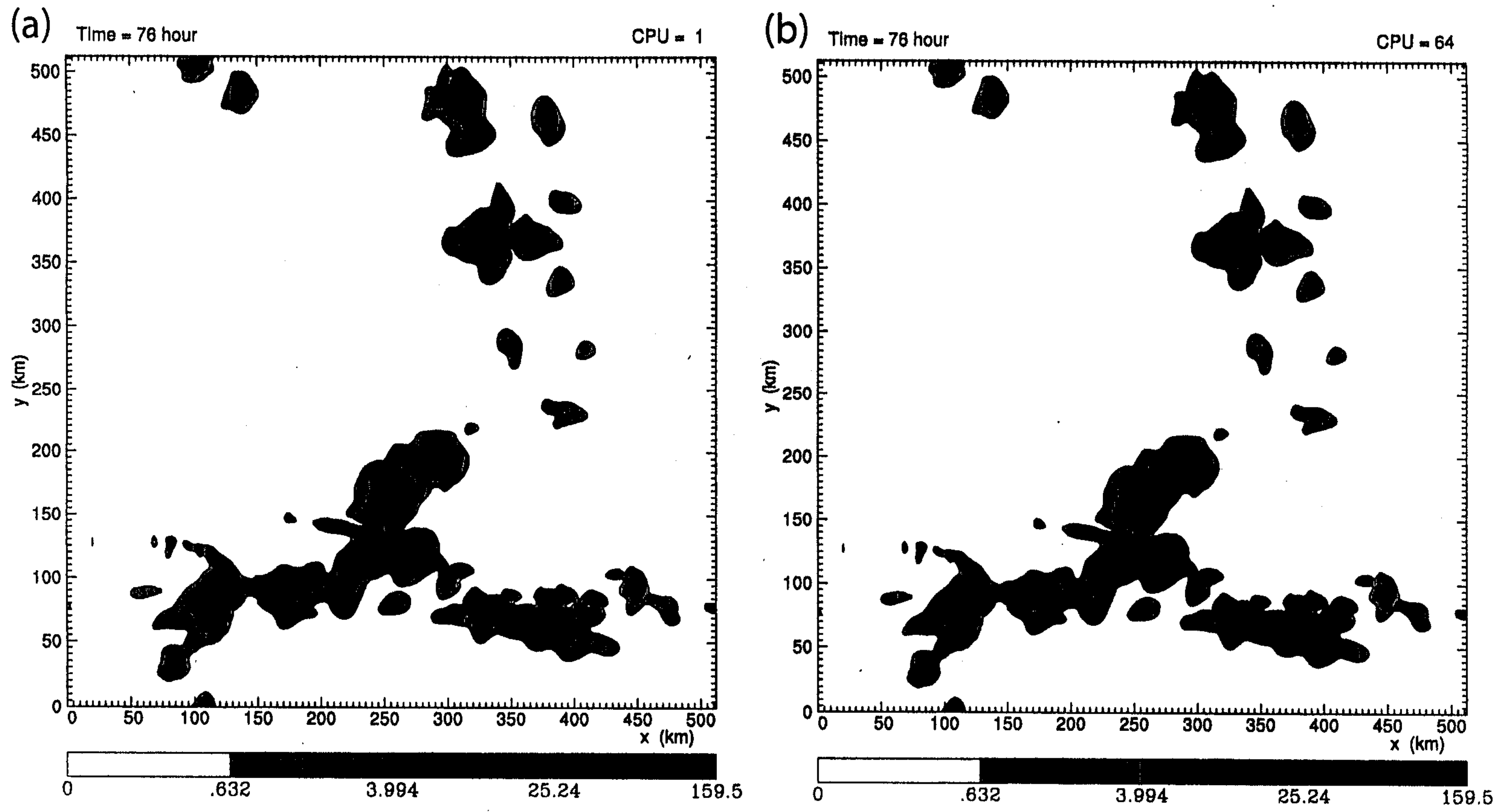

(a)
Time = 76 hour
CPU = 1
y (km)
x (km)
0
50
100
150
200
250
300
350
400
450
500
0
.632
3.994
25.24
159.5
(b)
Time = 76 hour
CPU = 64

Fig. 14

# Implementation of a Message Passing Interface into a Cloud-Resolving Model for Massively Parallel Computing

Hann-Ming Henry Juang, Wei-Kuo Tao, Xiping Zeng, Chung-Lin Shie, Joanne Simpson and Steve Lang

*Mon. Wea. Rev.*

## Popular Summary

The capability for massively parallel programming (MPP) using a message passing interface (MPI) has been implemented into a three-dimensional version Cumulus Ensemble (GCE) model. The design for the MPP with MPI uses the concept of maintaining similar code structure between the whole domain as well as the portions after decomposition. Hence the model follows the same integration for single and multiple tasks (CPUs). Also, it provides for minimal changes to the original code, so it is easily modified and/or managed by the Goddard model developers and users who have little knowledge of MPP.

The entire model domain could be sliced into one- or two-dimensional decomposition with a halo regime, which is overlaid on partial domains. The halo regime requires that no data be fetched across tasks during the computational stage, but it must be updated before the next computational stage through data exchange via MPI. For reproducible purposes, transposing data among tasks is required for spectral transform (Fast Fourier Transform, FFT), which is used in the anelastic version of the model for solving the pressure equation.

The performance of the MPI-implemented codes (i.e., the compressible and anelastic versions) was tested on three different computing platforms. The major results are: 1) both versions have speedups of about 99% up to 256 tasks but not for 512 tasks; 2) the anelastic version has better speedup and efficiency because it requires more computations than that of the compressible version; 3) equal or approximately-equal numbers of slices between the x- and y-directions provide the fastest integration due to fewer data exchanges; and 4) one-dimensional slices in the x-direction result in the slowest integration due to the need for more memory relocation for computation.

The MPI implementation can broaden the applications of the GCE model to the multi-scale (from micro, cloud, cloud system, local and regional) precipitation processes, hydrologic cycles and regional climates.