PROCEEDINGS OF SCIENCE

Seasonal forecast modeling application on the GARUDA Grid infrastructure

Ramesh Naidu Laveti[1], **S. Janakiraman, Mohit Ved and B. B. Prahlada Rao**
Centre for Development of Advanced Computing
CDAC Knowledge Park, Byappanahalli, Bangalore, Karnataka, India
E-mail: {rameshl, jraman, mohitv, prahladab}@cdac.in

Seasonal Forecast Model (SFM) is an atmosphere general circulation model used for predicting the Indian summer monsoon rainfall in advance of a season. Ensemble forecasting methodology is used to minimize the effect of uncertainties in the initial conditions. The inherent parallelism available with the ensemble forecast methodology makes it a suitable application that can effectively utilize the power of Grid computing paradigm.

SFM is implemented on the GARUDA Infrastructure, a national Grid computing initiative of India. Initially, the SFM is run at T62 resolution (Equivalent to 200 km x 200 km physical grid resolution). GridWay meta-scheduler is used to schedule and monitor the jobs with Portable Batch System (PBS) as the local resource manager. GARUDA Visualization Gateway (GVG) tool is used to gather and visualize the outputs from different sites. This prototype run is executed on compute clusters at five different geographical locations. Due to the heterogeneous nature of the GARUDA Infrastructure, variations in performance were noticed. The results of the prototype runs were also used by the GARUDA operational community to fine tune the configurations of compute clusters at various sites of GARUDA.

High resolution SFM at T320 resolution (Equivalent to 37 km x 37 km physical grid resolution) is also implemented to understand the scalability of the application on the Grid. Further work is underway to increase the ensemble size. Large ensemble size requires the use of considerable amount of computing power and storage which typically cannot be found at one or two locations. In this work, we describe our experience in conducting the ensemble runs of SFM on GARUDA Grid. We attempt to provide a perspective on the desirable features of a Grid middleware for easier uptake to Grid computing by the climate modeling community.

The International Symposium on Grids and Clouds (ISGC) 2012
Academia Sinica, Taipei, Taiwan
February 26 – March 2, 2012

[1] Speaker

1. Introduction

Weather forecast is an estimation of future state of the atmosphere by first estimating the current state of the atmosphere and then calculating how this state will evolve in time using a weather prediction model. The initial state of the atmosphere can be estimated only with certain accuracy. As the atmosphere is a chaotic system, very small error in the initial state or initial conditions can lead to large errors in the forecast. Ensemble forecasting methodology is widely used to minimize the effect of uncertainties in the initial conditions [1] [4].

In ensemble forecasting, number of simulations using slightly different but similar initial conditions will be made. It requires considerable amount of computing power and storage which typically cannot be found in one location, thus there is a need for infrastructure like Grid. The ensemble forecasting problem can be seen as a set of independent tasks, where a single application is run many times with different parameters and/or input files [2]. The inherent parallelism available with the ensemble forecasting methodology makes it a suitable application that can effectively utilize the power of Grid computing paradigm. In the last decade, the weather and climate modeling community has started migrating toward Grid technologies to fulfill their increasing CPU power and storage requirements to do large ensemble experiments.

Seasonal forecast model (SFM) is used to conduct the ensemble experiments, which is an atmospheric general circulation model developed by Experimental Climate Prediction Center (ECPC), USA. SFM is an efficient, stable, state-of-the-art atmospheric model designed for seasonal prediction and climate research. SFM is an open resource available to the research and academic communities under research license [6]. The present work describes the experience of conducting ensemble forecast experiments with SFM on the GARUDA Grid Infrastructure. GARUDA is an acronym for Global Access to Resources Using Distributed Architecture. It is India's National Grid Computing initiative bringing together academic, scientific and research communities for developing their data and compute intensive applications. GARUDA Grid is an aggregation of resources comprising of compute clusters, mass storage and scientific instruments distributed across the country [5].

The details of SFM are briefly discussed in Section 2. Section 3 describes the components of GARUDA Grid and its schematic structure as viewed by the user. Section 4 presents the implementation details of SFM on GARUDA, prototype experiments and ensemble simulations with high resolution model configuration. Section 5 presents the benefits and challenges of using Grid computing for climate modeling community. Some key requirements that need to be addressed by the Grid middleware for easier uptake to Grid computing by the climate modeling community are also presented in Section 5. Finally, Section 6 presents concluding remarks and future work direction.

2. Seasonal Forecast Model

Seasonal forecast model is composed of three components: LIB, GSM and RUN. LIB contains utilities, model libraries and climatological/constant fields. Model libraries are machine dependent and need to be built once for a particular machine. GSM contains model source code files which are used to create model executables. For different resolution and/or different

modeling options, we need to build the model executables separately. RUN contains the necessary scripts to run the model and store the model outputs. The parallelization strategy used and the portability details of SFM are explained in the following sub sections.

2.1 Parallelization strategy used in SFM

The basic strategy underlying the implementation of SFM to a parallel computing platform is to provide the flexibility to allow the same code to run on sequential, shared memory parallel as well as on distributed memory parallel machines. This will be achieved by preprocessing the code before it is compiled on different machines. It uses 2-D domain decomposition method to partition the data in both spectral and grid space [3].

SFM uses spherical harmonics based global spectral method for horizontal discretization of state variables. For the spectral method, the data distribution is based on the ease of computations without communication. It requires the entire array in one of the three dimensions (X or Y or Z) to reside in one processor to perform computations without communication. For example, fast Fourier Transform requires all the arrays in the X-direction to reside in one processor, but arrays in Y and Z directions can be separated into different processors. Similarly, Fourier summation in the Y-direction requires that the entire array in Y-direction must reside in one processor, but arrays in X and Z directions can be in separate processors [3].

The physical process calculations require all the variables at all levels for each grid point, so that arrays in the Z-direction need to be in the same processor but arrays in the X and Y directions can be separated onto different processors. This array distribution requires that entire arrays be rearranged into different configurations before the computational operations. This transpose method is named 2-Dimensional decomposition because one of the dimensions is fixed but the other two are distributed. This method has been widely used in many global spectral models [7]. 2-D decomposition is flexible in the choice of number of processors, almost any number can be chosen, in practice. As long as the number of processors is not a prime number, this works, but the efficiency may be affected for multi-way node machines, where inter-node communication tends to be faster than intra-node communication [7].

2.2 Portability details

As mentioned in section 2.1, SFM can run on multiple platforms with single and/or multiple shared memory machines. It can also run on a massively parallel processor (MPP) machine using Message Passing Interface (MPI). The code was designed carefully in such a way that reproducibility of the computation for shared memory and MPP computers is guaranteed in all processor configurations. Two-dimensional decompositions made the model flexible to run on any number of processors available, except prime number. This model can run on multiple platforms ranging from Cray, SGI, T3E, SUN and IBM-SP and has been demonstrated to be very efficient [3]. This model is being used by Indian Meteorological Department (IMD) for forecasting Indian summer monsoon [10]. So, it is a suitable application that can efficiently utilize the power of Grid computing paradigm which is heterogeneous in nature.

3. The GARUDA Grid

GARUDA Grid is an aggregation of resources comprising of compute clusters, mass storage and scientific instruments distributed across the Indian Territory.

3.1 Core components of GARUDA Grid

In this section, we describe the components of GARUDA Grid [Fig. 1]. GARUDA is based on Service-Oriented Architecture (SOA), comprises of - a set of core system components that provide system-wide services and a set of common interface definitions that resources or services may implement in order to provide users with familiar and consistent interfaces to build their applications.

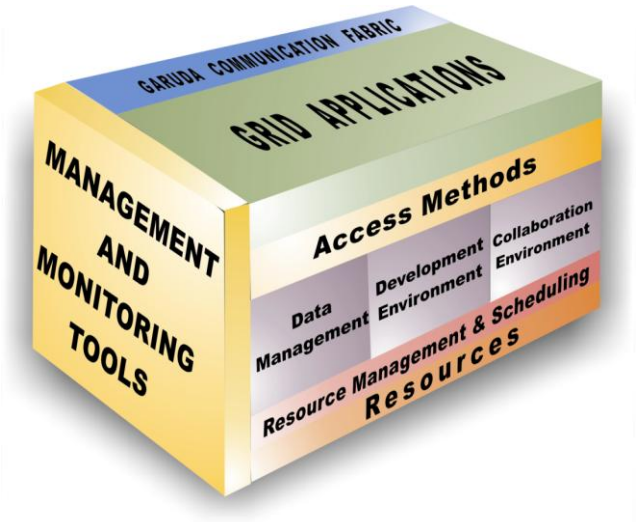

Fig. 1: The Core components of GARUDA Grid Infrastructure

The core system components of the GARUDA include:

- Communication fabric - Dependent on National Knowledge Network (NKN), connectivity through high - speed communication fabric [8].
- Resources - Heterogeneous and distributed computing resources, pooling of compute and storage resources and special devices provided by C-DAC and its partners [9].
- GARUDA Information Service - keeps track of distributed GARUDA resources.
- Security (with Authentication and Authorization service) - VOMS for Virtual organization and MyProxy for certificate management.
- Job Management - Web Portal, Command line interface, Workflow tools and PSE for the job management interface. It deals with data movement, scheduling, reservation and accounting of jobs.
- Access methods
    a) GARUDA Access Portal (GAP) acts as a GUI interface
    b) Command line Interfaces (CLI)

Now, we describe the GARUDA Grid from a user's point of view, which is shown in [Fig. 2], in the following sub-section.

3.2 Schematic representation of GARUDA Grid – A user's view

In a HPC cluster, a user is accustomed to local cluster environments, where all resources are homogeneous and access to them is done through a unique user account. In a Grid environment, resources are distributed and heterogeneous in nature. In order to provide the

transparent access to these distributed heterogeneous resources, Grid technology uses some services called middleware that aggregates heterogeneous resources and present them as a single homogeneous system.

The most important part of Grid middleware is centralizing the management of all the distributed resources [Fig 2]. There are two main services, authentication & authorization (using X509 IGCA Certificate and Proxy) and information (managing resource information and status). Here, IGCA stands for "Indian Grid Certification Authority", which issues the X509 certificate to the user. These basic services are used by other services (e.g. the data management and Job execution services). These resource-specific services rely on the authentication and information services to make their decisions. In order to provide all these services middleware user tools are installed in the user interface.

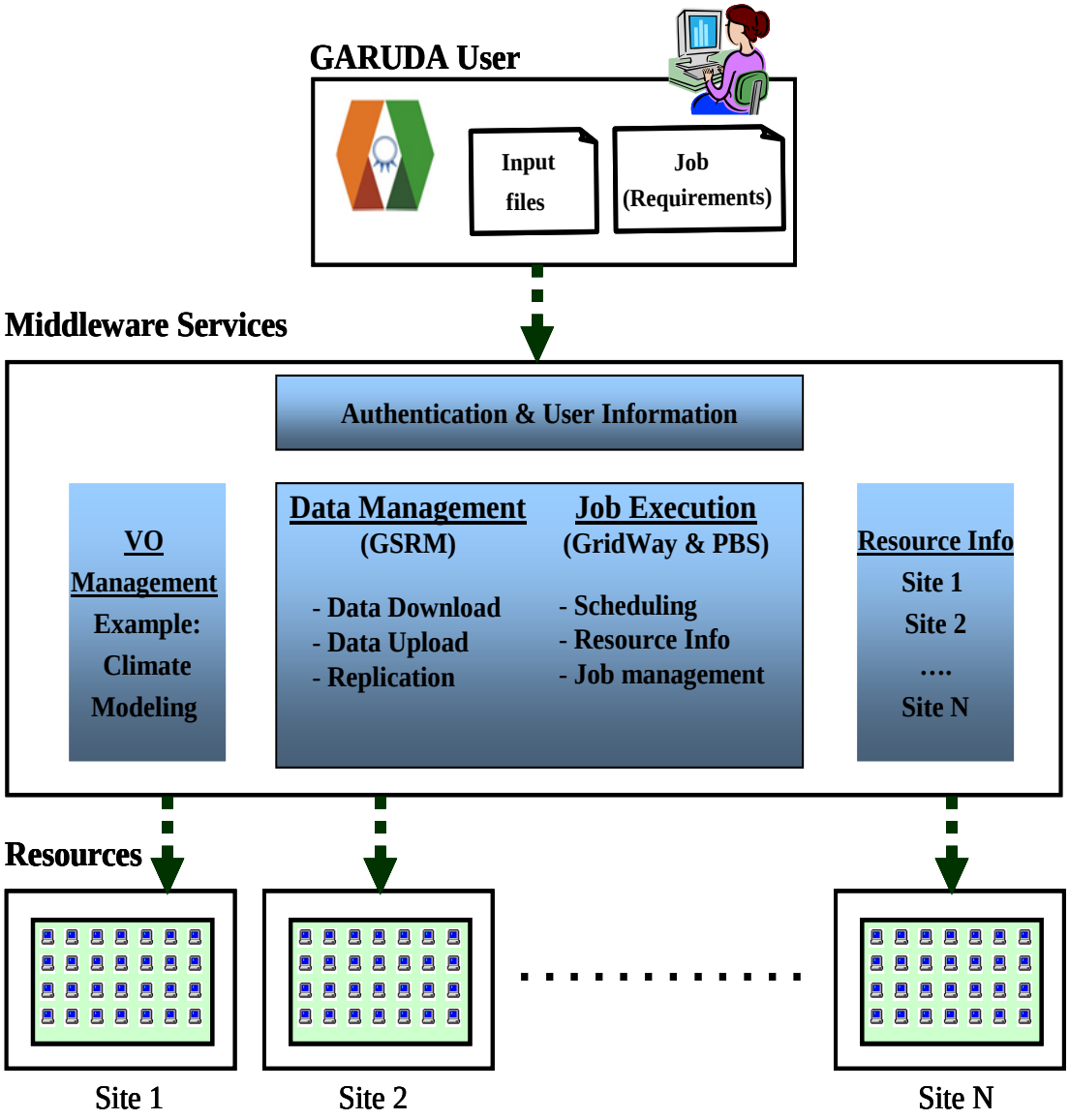

Fig 2: The user view of GARUDA Grid – A schematic representation

GARUDA Grid users are organized as Virtual Organizations (VO) (e.g. ClimateModelling), where they register their IGCA certificates. A VO is just an entity that maintains a list with the certificates of all the users that belong to it along with their roles and groups. The VO is queried by the resources in order to determine if a user can access it or not.

From the user's point of view, the job submission process to the Grid is same as the job submission process to a local cluster or a supercomputer i.e. the user fills or prepares a job template with the job requirements and the executable to be run and submits the job to a queue using the middleware user tools (GARUDA Access Portal or Command Line Interface). Job scheduling is done using a meta-scheduler called GridWay.

The storage and access to data are done through a virtual file system called GARUDA Storage Resource Manager (GSRM). The data can be replicated and distributed to different sites and the GSRM can select the particular copy to be used for a particular execution according to, for instance, proximity to the execution node. The user can transfer/download files to/from the GARUDA Grid though GridFTP. After the execution of the job user can post-process and visualize the data using a tool called GARUDA Visualization Gateway (GVG).

4. Implementation of SFM on GARUDA

Initially, the low resolution configuration of SFM (T62, equivalent to 200Km x 200Km physical grid) was implemented on GARUDA Grid on five compute clusters as a prototype. Here, 'T62' stands for spherical harmonic expansion truncated at wave number 62 and it uses triangular truncation. After the successful completion of these prototype experiments, we implemented the high resolution configuration of the model (T320, equivalent to 40Km x 40Km physical grid) on GARUDA Grid and conducted many experiments to understand the scalability of the application. Later, we have designed a framework to do the ensemble forecasting experiments using high resolution configuration of the model across the clusters of GARUDA Grid. Both the configurations of the model are compiled using the C compiler "gcc-v4.0" and the FORTRAN compiler "mpiifort" of Intel MPI. The clusters used for these experiments are shown in Fig 3. The following sub-sections describe each of these experiments in detail.

Clusters	Processor	Speed	Mem.	CPUS /Node
Bangalore	Intel Xeon	3.16 GHz	16 GB	8
Hyderabad	Intel Xeon	3.16 GHz	16 GB	8
Chennai	Intel Xeon	3.16 GHz	16 GB	8
Pune	Intel Xeon	2.95 GHz	64 GB	16
Ahmedabad	AMD Opteron	2.50 GHz	64 GB	16

Fig. 3: The clusters used to do the ensemble experiments

4.1 Prototype experiments

The model was configured for 3-days forecast with 30 minutes time step, to run on 8 processors (1 Node x 8 Processors). Several runs were made on each of the five clusters and the results were validated by comparing them with the results obtained under the project "Seasonal Prediction of the Indian Monsoon (SPIM)" [10]. The results compared well. After validating the above runs, the model was configured to forecast the Indian summer monsoon season that spans from June to September, using the same number of processors. Several experiments were conducted on five clusters of GARUDA and performance variations were observed. These observations were communicated to GARUDA Grid operational community to fine-tune the configuration of the clusters to improve the performance. The details are given in section 5.

4.2 Ensemble forecast experiments of high resolution SFM

Initially, the model was configured for a 1-day forecast with 2 minutes time step, to run on 64 processors (8 nodes x 8 processors). Many runs were made on five clusters of GARUDA and the porting of the high resolution configuration of SFM was validated by comparing the results obtained over Grid to those of cluster. We also configured and compiled the model to run it using 128 processors and 256 processors on every cluster. These configurations are used to run the model on 128 and 256 processors to understand the scalability of the model. Later, a framework was developed to do the ensemble experiments across the clusters on GARUDA. The methodology followed to develop this framework is explained below.

4.2.1 Methodology

The model was ported on five clusters and on each cluster it can run with a different ensemble member. To do this, a workflow wrapper was developed using the existing middleware services to organize and manage the execution of the model without modifying its source code. The data management was controlled by the GARUDA Storage Resource Manager (GSRM) and the job submission and monitoring was handled by the GridWay meta-scheduler. The post-processing and the visualization of the output data was done using GARUDA Visualization Gateway (GVG) where the visualization tool "Grid Analysis and Display System (GrADS)" is integrated.

The execution of SFM-T320 on Grid is divided into two parts, the first part is sequential, in which we do the model configuration, preprocessing and creating the necessary intermediate files for the model forecast. The second part is the forecast which runs in parallel. The framework to conduct ensemble experiments across the clusters on GARUDA is shown in Fig. 4. The following is the stepwise procedure to do the ensemble experiments using the above framework.

a) Login to the Job submission Node (GridFS) and check the Grid proxy information.
b) Create a job parameters file which will have details about the job, like the list of ensemble members, number of processors and the list of initial/input conditions.
c) Enter the list of ensemble members using which the model need to be run.
d) Check the resources available on each compute cluster and give the ranking to the clusters according to the availability of the resources.

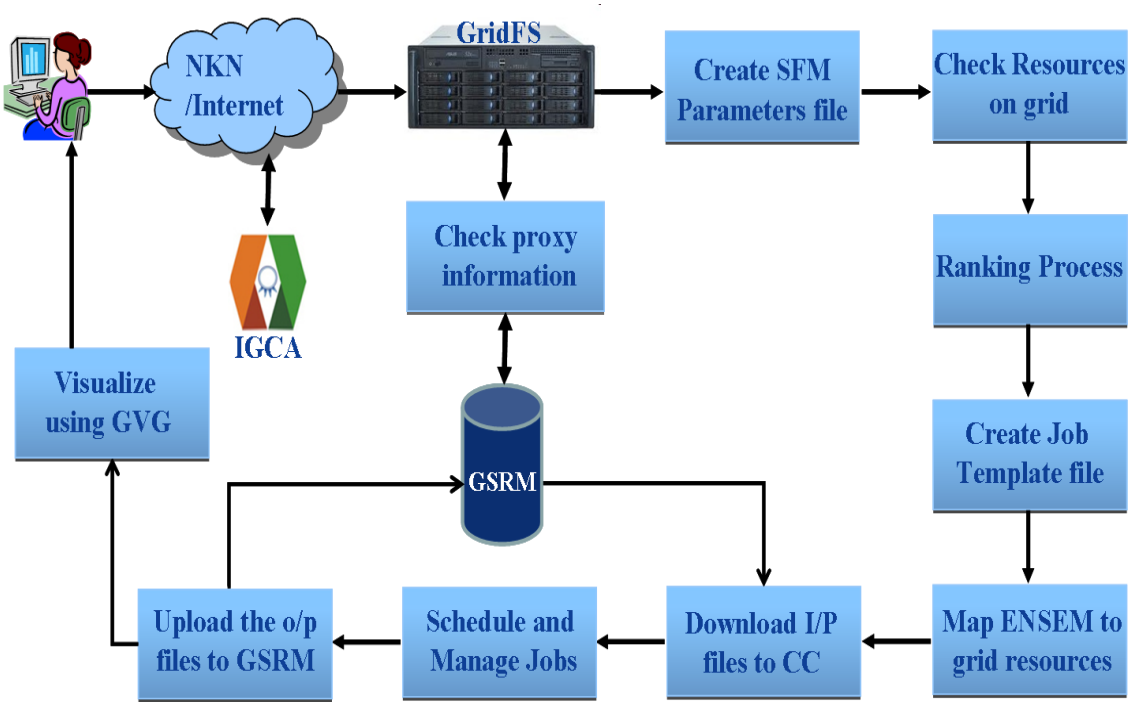

Fig. 4: The framework to conduct the ensemble experiments across the clusters on GARUDA (I/P – Input files, O/P – Output files, CC – Compute Cluster, ENSEM – Ensemble member)

e) Create a job template file which will be used by GridWay meta-scheduler.
f) Get the input files to the corresponding compute cluster from GSRM.
g) Schedule the jobs according to the rank given to a cluster. On each cluster the job will be scheduled using local resource manager called Portable Batch System (PBS). The first ensemble member experiment will go to the cluster having rank 1.
h) Monitor the jobs using GridWay meta-scheduler. If a job fails, resubmit the job automatically with the same ensemble member as the input. If it fails for second time then it aborts. User intervention is required here to trace and fix the problem by using run logs. Then user should resubmit the job.
i) Upload the output files to distributed storage of GARUDA using GSRM.
j) Visualize the output data using GrADS which is integrated in GVG.

This framework is developed using shell scripts. Few changes are made to the model run scripts to integrate them with the workflow wrapper to conduct ensemble experiments.

5. Discussions & Results

In order to compare the efficiency of the Grid with traditional compute clusters, the low resolution configuration of the model was simulated using one node of an independent cluster which is located at Bangalore in India (1 node with 2 x Quad core Xeon @ 3.16 GHz totaling 8 cores and 16 GB of memory). The performance metrics of the experiment on this independent

cluster and the experiment done on the same cluster using Grid tools were observed. Significant performance degradation was noticed when migrating from cluster to Grid. In case of 3-days forecast experiment the total turnaround time of the model run on Grid (2m 35s) is almost double the total turnaround time of the cluster (1m 13s). If we increase the problem size to 165 days forecast, then the total turnaround time of the model run on Grid (75m 46s) is almost 20% more than that of the independent cluster (63m 16s). The reason for the fall-off of performance in both the cases is investigated and was found out that this is due to the extra time taken by the stage-in & stage-out processes. We also observed the variations in the performance of different clusters on the Grid [Table 1], which were due to several factors including the CPU speed, wall time spent in queue, MPI libraries, the differences in bandwidth and errors during the execution. These observations are used by GARUDA operational community to fine-tune the clusters of GARUDA to improve the performance.

Table 1: Performance metrics of SFM-T62-Seasonal forecast experiment using 8 cores on each of the five clusters of GARUDA. T1 and T2 are the total turnaround time before and after tuning the clusters.

	Pune	**Bangalore**	**Chennai**	**Hyderabad**	**Ahmedabad**
Processor speed	2.93 GHz	3.16 GHz	3.16 GHz	3.16 GHz	2.5 GHz
Processor family	Intel Xeon	Intel Xeon	Intel Xeon	Intel Xeon	AMD Opteron
T1	74m 46s	75m 46s	191m 37s	191m 20s	273m 38s
T2	74m 46s	75m 46s	81m 37s	77m 20s	92m 38s

Later, we conducted experiments with the high resolution SFM using 64, 128 and 256 processors on each of the clusters to generate 1-day forecast to understand the scalability of the application. We observed 80% gain in performance when we increase the number of processors from 64 to 128, and 40% gain from 128 to 256. It shows that the high resolution model can scale up to 256 processors. After this, ensemble experiments are conducted using the workflow wrapper.

The ensemble experiment consists of numerous runs, which include 1-day forecast runs and seasonal forecast runs. It was conducted on different clusters which are geographically distributed, and we executed the model using 64 cores on each of the clusters. Each ensemble run took around 30 min to complete the 1-day forecast on each of the clusters and requires around 2.1 GB of disk space for output files. After validating the porting of the model, using 1-day forecast output, we conducted ensemble runs for seasonal forecast. A seasonal run generates the forecast for the months of June, July, August and September. This period is called Indian summer monsoon season. A seasonal run using a single ensemble member needs around 27 GB of disk space for its output data and it requires around 80 hrs of wall clock time to complete the run. Total CPU time of all these runs was more than 8000 hours i.e. around 1 year. Using the five clusters of GARUDA Grid with 64 processors each, model ensemble runs could be completed within 1 month. This shows that typical climate applications can harness the power of Grid computing.

We present few of the results obtained from SFM-T320 by running it on GARUDA Grid infrastructure. In the following graph [Fig. 5], the top panel shows the ensemble mean rainfall of Indian summer monsoon season of 1987 and the bottom panel shows the ensemble mean rainfall of Indian summer monsoon season for the year 1988. We are interested in assessing SFM whether it is capable of simulating droughts and excess rainfall years. Excess monsoon rainfall occurred in 1988, whereas drought occurred in 1987. SFM is capable of simulating these extremes i.e. drought or excess rainfall seasons.

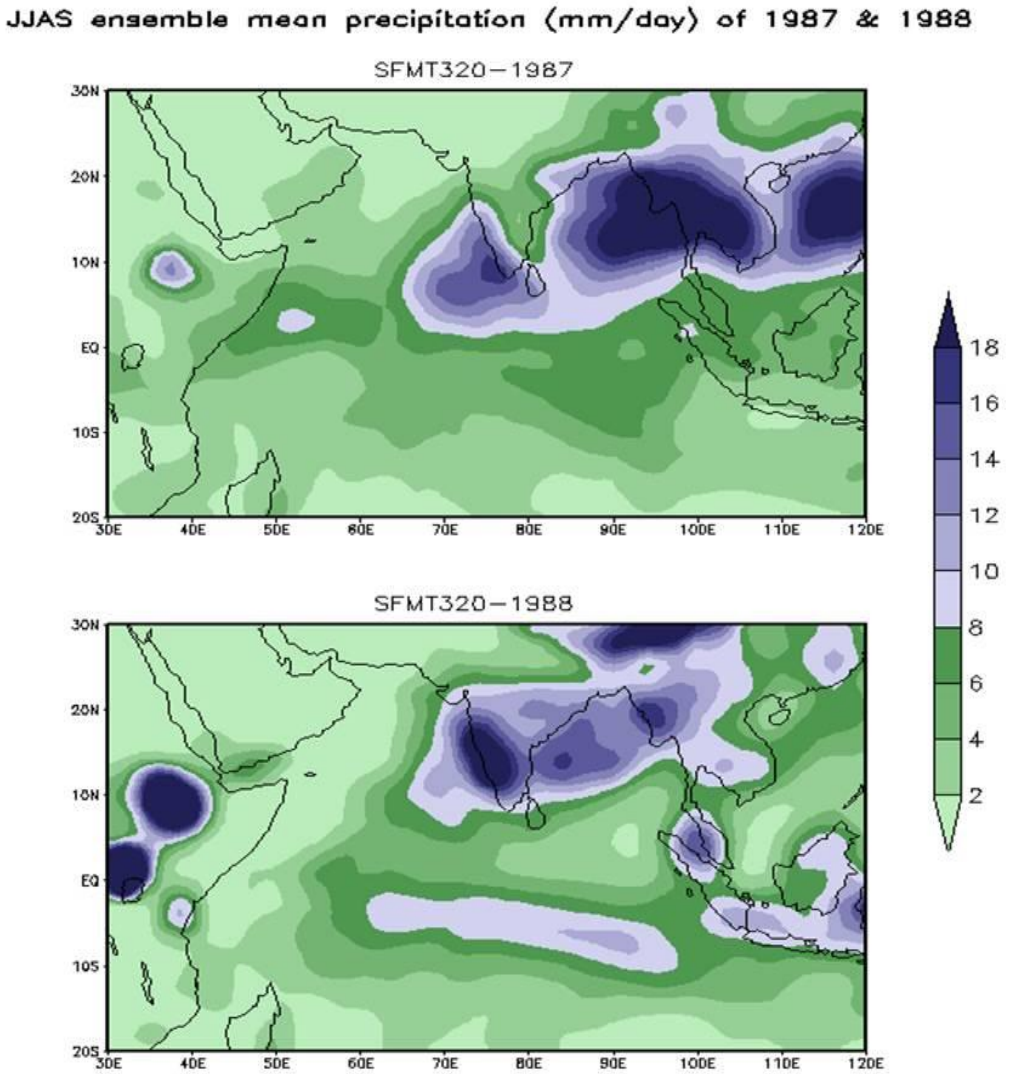

Fig 5: The ensemble mean rain fall of summer monsoon season of India for the years 1987 and 1988, generated using SFM-T320

The experience gained from conducting ensemble experiments on GARUDA Grid gave us a perspective on the desirable features of a Grid middleware for easier uptake to Grid computing by the climate modeling community. The typical challenges of climate models are – huge number of parallel computations, long runs lasting several hours to several days depending on

the type of experiment and large volumes of output. For example, to compute the climatology for the period of 1984-2011, i.e. 28 years, using six ensemble members using SFM-T320, one would require 4.5 TB of disk space and around 560 days of wall clock time if the model is run on a single cluster using 64 processors (8 nodes, each node with 2 x Quad core Xeon @ 3.16 GHz and 64 GB memory). To address these challenges, Grid middleware should have the following features.

- Capability to handle the issues related to non-uniform and small disk quotas that are available on the remote clusters of the Grid.
- Capability to identify the defective jobs as early as possible to avoid the wasting of processor time.
- Capability to transfer the output data files to distributed storage system which is available on Grid during the experiment which will avoid the accumulation of huge data on remote clusters.

6. Conclusions

Our experience in conducting the ensemble forecast experiments using a climate application called SFM on GARUDA Grid infrastructure is presented in this paper. It has been shown that ensemble forecasting is a suitable application from climate modeling domain which can harness the power of Grid computing paradigm. A framework was developed to conduct the ensemble forecast experiments. This framework gave a foundation on which a reliable Grid environment for climate applications, which are time critical, can be build.

This experience with deploying the climate modeling application on the Grid suggests the inclusion of additional capabilities to the middleware .viz. capability to handle the small disk quotas on remote clusters, capability to identify the defective jobs as early as possible to avoid the wasting of CPU time and capability to transfer huge data at run time from remote clusters to distributed storage system etc. The suggested features to the middleware will help to deal with the complexities in a better manner and minimize the instabilities.

Also, many of the performance variations noticed in the experiments were attributable to the heterogeneous nature of the resources in the Grid. With the contribution of this paper we will further enhance our framework to deal with the complexity and instability of the Grid infrastructure which can help the climate community to port their complex applications on Grid with more comfort.

Acknowledgments

We would like to thank GARUDA Grid Operations and Administration (GGOA) team and National PARAM Supercomputing Facility (NPSF) team of C-DAC for their timely support to conduct experiments on GARUDA. We also thank GARUDA Storage Resource Management team (GSRM) for providing disk space.

References

[1] M. Leutbecher and T. N. Palmer, *"Ensemble forecasting"*, Journal of Computational Physics, Vol. 227, pp 3515-3539, 2008

[2] V. Fernadez etal, *"Benefits and requirements of grid computing for climate applications. An example with the community atmospheric model"*, Journal of Environmental and Modelling software, Vol. 269, pp 1057-1069, 2011.

[3] Masao Kanamitsu etal, "NCEP dynamical seasonal forecast system 2000", Bulletin of the American Meteorological Society, Vol. 83, pp 1019-1037, 2002.

[4] John M. Lewis, "Roots of Ensemble Forecasting", Monthly Weather Review, Vol. 133, pp 1865-1885, 2005.

[5] The GARUDA India National Grid Initiative, http://www.garudaindia.in/.

[6] The ECPC seasonal prediction system, http://ecpc.ucsd.edu/projects/G-RSM/docs/INT/

[7] J. G. Sela, "Weather forecasting on parallel architectures", Journal of parallel computing, Vol. 21, Issue 10, pp 1639-1654, ISSN 0167-8191, 10.1016/0167-8191(95)00039-1, October 1995.

[8] The National Knowledge Network, http://nkn.in/

[9] The GARUDA partner's information, http://www.garudaindia.in/html/partner_info.aspx

[10] Sulochana Gadgil and J. Srinivasan, "Seasonal prediction of the Indian monsoon", Current Science, Vol. 100, No. 3, pp 343-353, 2011.