A Distributed Memory Implementation of the Regional Atmospheric Model PROMES

Juan E. Garrido[1], Enrique Arias[1], Diego Cazorla[1], Fernando Cuartero[1], Iván Fernández[2], Clemente Gallardo[2] *

Abstract— **This paper describes the parallelization process of the code PROMES, which represents a regional atmospheric model developed by some of the authors. The parallel code, called PROMESPAR, has been carried out under a distributed platform (cluster of PCs) and using Message Passing Interface (MPI) communication subroutines.**

Keywords: Regional atmospheric model, parallelization, message passing interface

1 Introduction

Climate change induced by human activities is one of the topics to which more attention is devoted to scientific research today. This is due, not only by the great complexity involved in the processes affecting the climate, but also to the threat involved in the serious impact that occurs on the economics and the environment in many parts of the planet. Three or four decades ago, it was believed that the oceans would be able to absorb the pollutants emitted by human activities; but today, maritime degradation is undeniable. Even more recently, the idea that humanity could induce a change in climate was a hypothesis that received little scientific support. However, there is now a broad consensus among scientists, about the evidence of anthropogenic climate change and the need for better knowledge about likely developments in the following decades.

To simulate the climate, we use numerical models reproducing the main processes occurring in the five components of the climate system: Atmosphere, hydrosphere, geosphere, and biosphere, and the exchange of mass and energy between them. The results obtained by the models are evaluated and compared with the observed features of the climate in recent decades. Once it is found the quality level of the climate model is correct, we apply it to simulate potential changes in the climate, considering various scenarios of anthropogenic emissions of greenhouse gases and aerosols. Since this information, we can deduce the potential impact of climate change produced in such a hypothesis.

The history of weather forecasting is intimately associated to development of high performance and parallel computing [9].

Is in the early stage of 1922, when L. F. Richardson provides a vision of how to partition the large amount of computation required in this task, by using thousands of computers. [1].

However, is in later forties, when the first steps towards the use of computers in weather forecasting were done. This beginning was made by von Neumann, Charney and his colleagues in the computer ENIAC and its successors. The work done by these researchers was so important that, thereafter, it was considered the numerical weather prediction methods as a whole discipline, and that was the origin of the establishment of national prediction centres. In fact, today the major supercomputing centres tend to focus on such tasks.

While the first steps in the weather prediction were bearing fruit, it was thought to apply the same methodology in predicting the a longer term, not only predicting changes in the atmosphere (weather) but also in the global system time (climate change).

Since the forties, there was a dramatic improvement in numerical methods, algorithms and computer technology, as well as physical models and science related with weather and climate.

In fact, scientists working with models of climate and weather are the main users of parallel platforms. However, it is necessary not only to have a platform, but also the parallel algorithms suited to these platforms to exploit the full potential of resources. Scientists in these areas were the first to make effective use of machines with segmented architecture, such as IBM 360/195, Cray 1, Cyber 205, Cray YMP and Cray 90.

*This work has been supported by National Project CGL2007-66440-C04-03. Instituto de Investigación en Informática de Albacete [1] and Instituto de Ciencias Ambientales [2], University of Castilla-La Mancha, Avda. España s/n,02071-Albacete, Spain, email and telephone number: JuanEnrique.Garrido@uclm.es +34-967-599200 Ext. 2690, Enrique.Arias@uclm.es +34-967-599200 Ext. 2497, Diego.Cazorla@uclm.es +34-967-599200 Ext. 2545, Fernando.Cuartero@uclm.es +34-967-599200 Ext. 2478, Ivan.Fernandez@uclm.es +34-967-599200 Ext. 5761, Clemente.Gallardo@uclm.es +34-967-599200 Ext. 5461. Fax number: +34-967-599224 [1], +34-925-268840 [2].

Until nineties, it was not made a serious attempt to develop an operational parallel model. Americans were the first to combine the efforts of experts in meteorology with experts in high performance computing, included in the program of High Performance Computing and Communications (HPCC), in a more ambitious program such as the Computer Hardware, Advanced Mathematics and Model Physics (CHAMMP), a program of the U.S. Department of Energy. The result was the development of a set of models to use scalable parallel computer systems.

Thanks to the parallelization of weather prediction models, it is provided to scientists the ability to deal with longer simulations, to increase the spatial resolution, etc.

Throughout the last decade, several parallel approaches have been developed. Among them, we remark [3] the based on vectorial multiprocessors such as CCM2 and its scalable variants [2, 12, 20], massively parallel computers (adaptation from the spectral model of the National Meteorology Centre) [19], distributed memory multiprocessors [18] (integrated prediction system) and passing messages [15].

Since 1995 until now, it has been followed different ways in the application of parallelism to the weather prediction. These paths have led us to new versions of the above mentioned models (i.e. the last version of the CAM model called CCM [7]), to applications in our area of interest, such as GRID technology (IrisGRID [4] or CrossGrid [5]), the apparition of Climateprediction.net [6] program, to adaptations of different codes to the most powerful machine of the moment [17, 13, 14] and implementations of meteorological aspects such as weather data assimilation or transposing multidimensional vectors [10].

Special mention deserves the MM5 fifth generation mesoscale model [8].It is relevant because it is the model used as reference by Promespar (its implementation is carried out taking into account only part of the comprehensive scheme conforming the full model).

Designed to work with high resolution (higher than 5km), the MM5 consists on a model with very sophisticated physical parameterizations schemes, but needing a huge computational power. It was developed by the University of Pennsylvania (PSU) and the National Center for Atmospheric Research (NCAR) in the United States.

The MM5 model, running on a parallel distributed memory platform, with massively parallel processor (MPP), networks of workstations, etc., is called MM90 [16]. This code was implemented in Fortran 90, using a communication library developed at Argonne National Laboratory called RSL, library that corresponded to the ones provided by the seller (NX for Intel Paragon, or MPL for IBM SP2), or MPI for other platforms. MM90 is the successor of MM5 implementation of massively parallel machine called MPMM [11].

The paper is organized as follows. Section 2 introduces the regional atmospheric model PROMES, and in Section 3 the parallelization of PROMES is presented. The experimental results are outlined in Section 4. Finally, the conclusions and future work are commented in Section 5.

2 The regional atmospheric model PROMES

PROMES is a regional atmospheric model developed by some of the authors and presented in [1]. In particular, PROMES is a mesoscale forecast model over which several physical phenomena which act on the atmosphere are parametrized modifying its conditions and behaviour. Ir becomes evident that due to the fact that the model is represented by a set of equations, as bigger the number of physical parameters to parametrized as complex its resolution; and obviously its accuracy. The complexity on the solution makes necessary the used of parallel platforms to solver the problem in order to obtain the results in a reasonable time.

Figure 1 shows the physical parameters that are modelled on PROMES.

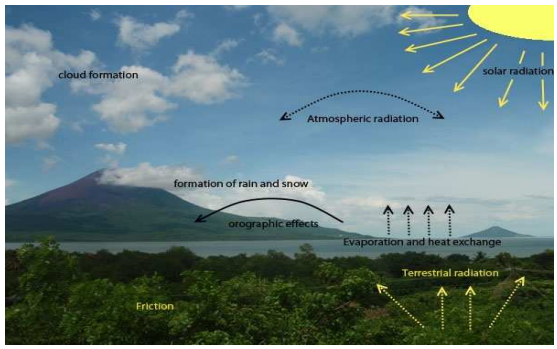


Figure 1: Physical parameters modelled at PROMES

In order to make the computations easier, the model divides the zone to be studied on a set of vertical columns, each one with the atmosphere behaviour in an instant of time. This division is known as grid of calculus and it is shown on Figure 2.

Finally, and overview of the structure of PROMES code is shown in Figure 3.

3 PROMESPAR: a distributed memory implementation of PROMES

As it was previously commented, in order to obtain a very accuare solution in a reasonable time, it is necessary the use of parallel platforms. In this paper, a distributed memory implementation of PROMES code, called PROMESPAR, is presented.

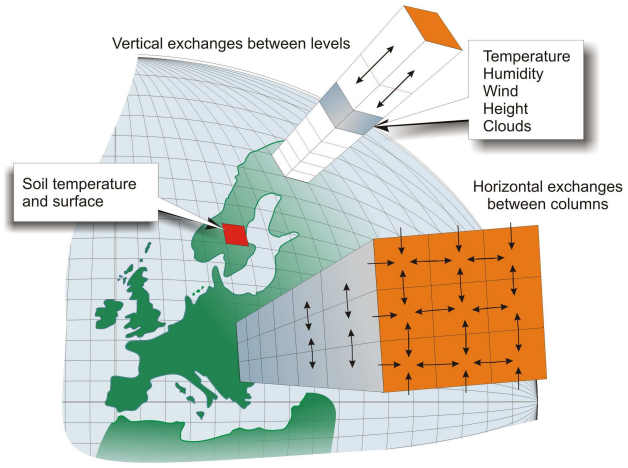


Figure 2: Grid of calculus

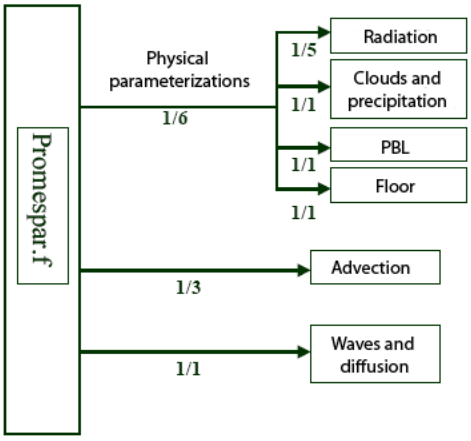


Figure 3: General squeme of PROMES code

La parallelization of PROMES consists on dividing the domain on a set of subdomains getting out the work to carried out into the different processors (see Figure 4). Once the domain has been divided the processors just exchange the frontier information.

In order to obtain an equally load balancing, a constrain is applied to the size of the subdomain and the number of processor to be used. This constrain is given by equation 2

$$ProcXBlockSize = (\frac{OrXmatSize}{XsizeProc}) \pm XBorderSize \quad (1)$$

$$ProcYBlockSize = (\frac{OrYmatSize}{YsizeProc}) \pm YBorderSize \quad (2)$$

where $ProcXBlockSize$ and $ProcYBloclSize$ mean the size of blocks for each processor at X or y coordinate,

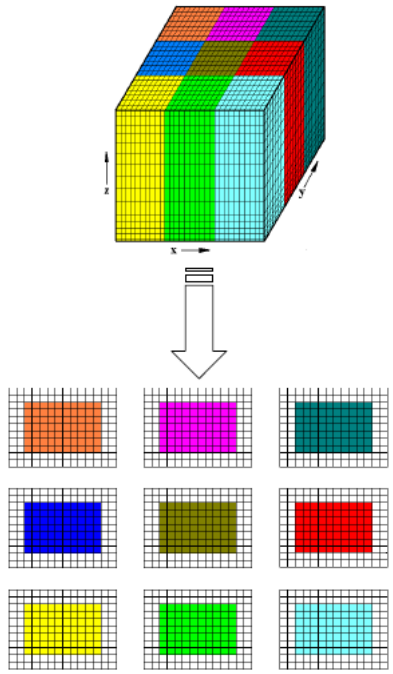


Figure 4: Squeme of spliting the domain into subdomains

respectively, which is computed from the original dimension of the matrix ($OrXmatSize$ and $OrYmatSize$) and the number of processors by each coordinate ($XsizeProc$ and $YsizeProc$), and taking into account the boundary conditions ($XBorderSize$ and $YBorderSize$).

However, processor 0 has additional tasks due to the fact that it acts as master reading initial conditions, boundary values for the domain, etc from files

In any case, the good load balancing could be affected mainly by two factors:

- **Static imbalance**. Those processors whose subdomains contain maritima zones have less computational load. This circunstance is due to the fact that the computations needed for solving the forecasting model are simplest in this kind of cells (some physical phenomena as the effect of orography, heat exchange with the masses of plants, etc are not taken into account).
- **Dynamic imbalance**. This kind of imbalence is devoted by the initial conditions. For instance, the effect of solar radiation could vary if a cloudy day or a sunny day is considered. These effects are unpredictable. However, other effects as the solar radiation during the night are predictable.

Figure 5 shows the different libraries considered in the implementation of PROMESPAR, all used under FOR-

TRAN programming language. In particular, the following libraries have been considered:

- MPI: Messing Passing Interface use for communications purpose. This library supports the comunication between the different processors of the distributed memory platform.
- NETCD: NetCDF (network Common Data Form) is a set of software libraries and machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.
- IOPSL: Library for input/output operations with meteorological data.
- Other physical libreries: computation of solar radiation, heat exchange ground-atmosphere, etc.

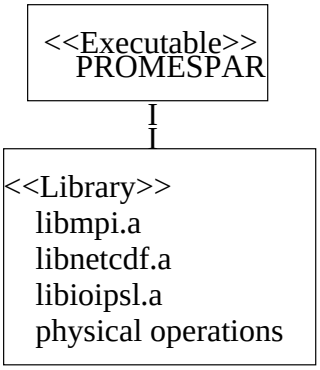


Figure 5: Components squeme of PROMESPAR

Figure 6 represents the workflow of the parallel implementation of PROMES, PROMESPAR.

The workflow in Figure 6 is followed by each processor, and the barriers on Figure 6 mean communication or synchronization taks amount the different processors.

4 Experimental results

The experimental results have been obtained taken into account 24 hours of simulation. The distributed memory implementation has been run into a cluster of PCs with 16 Intel processors at 1.8GHz, each one with 512 MB of main memory and interconnected by a Myrinet Network using NFS file system.

The performance obtained in the parallel implementations are evaluated in terms of:

- Execution time: Time spent in order to solve the problem.

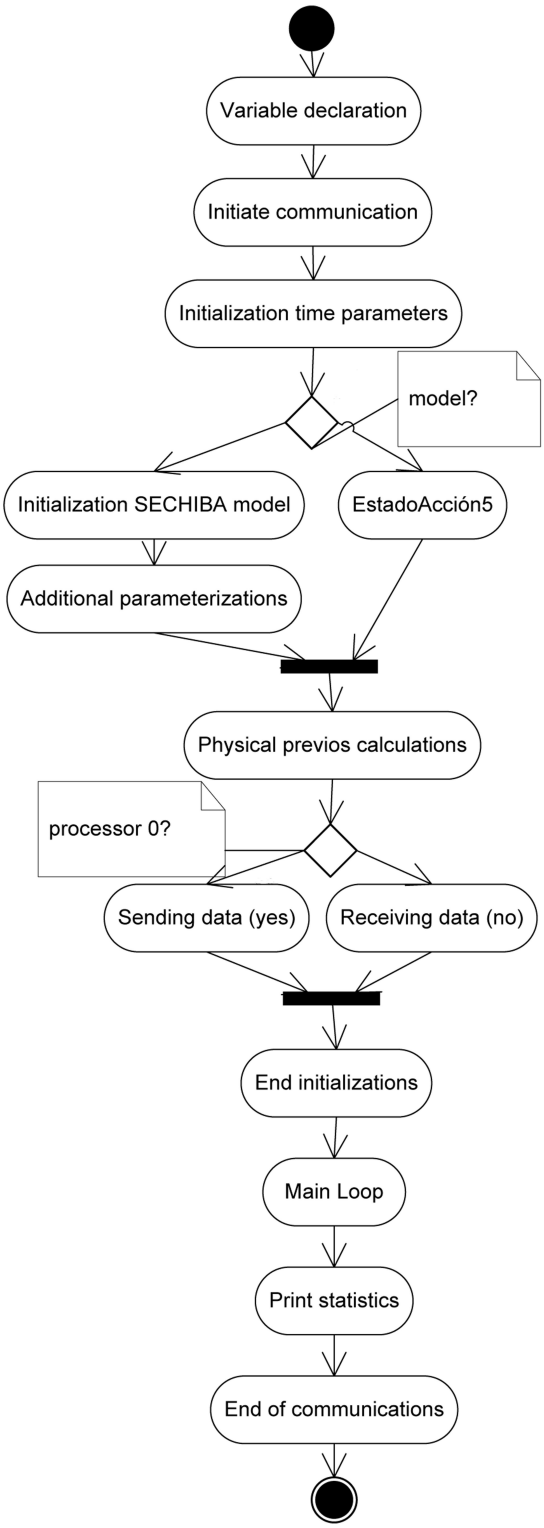


Figure 6: Workflow of PROMESPAR

- Speed-up: The ratio of the time taken to solve a problem on a processor to the time required to solve the same problem on a parallel computer with p identical processors.

- Efficiency: A measure of the fraction of time for which a processor is usefully employed; it is defined as the ratio of the speed-up to the number of processors.

Most time consuming has been spent at main loop where are contained the most caomputational cost operations. In particular, apart from send and receive operations for communication purpose, physical operations are invoqued. These operations are shown at Figures 3 and 6.

The experimental results considered in this section take into account a 24 hour simulation, which is equivalent to carry out 2881 iterations of main loop.

Figures 7, 8 and 9 show the results of the previous experiment (24 hour simulation) in terms of execution time, speed-up and efficiency.

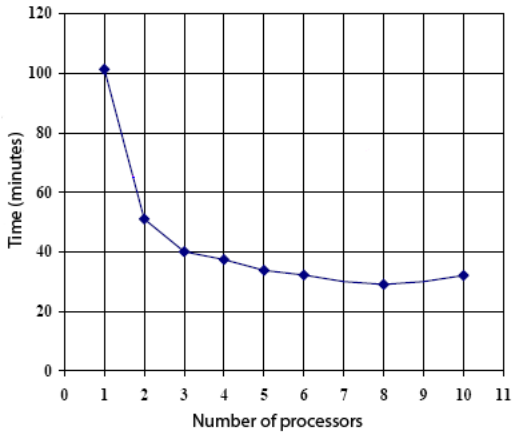


Figure 7: Execution time of PROMESPAR

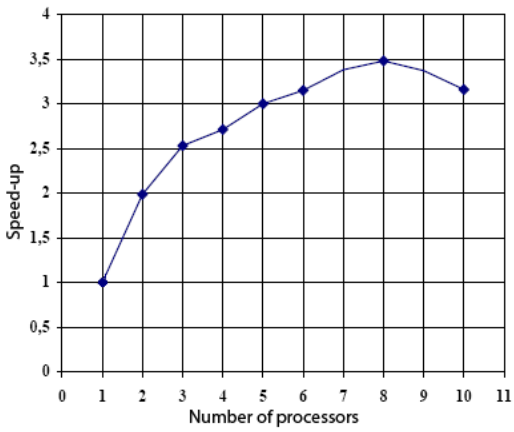


Figure 8: Speed-up of PROMESPAR

From the experimental results, te main conclusion is that the best results, in terms of execution time has been obtained considering 8 processors. However, in terms of speed-up and efficiency best results are obtained for 2 processors. This is a normal circunstance due to the influence of the communications. However, for this particular applications the main goal is to reduce the execution time.

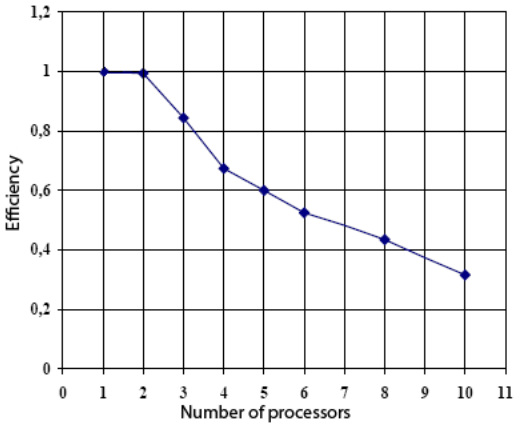


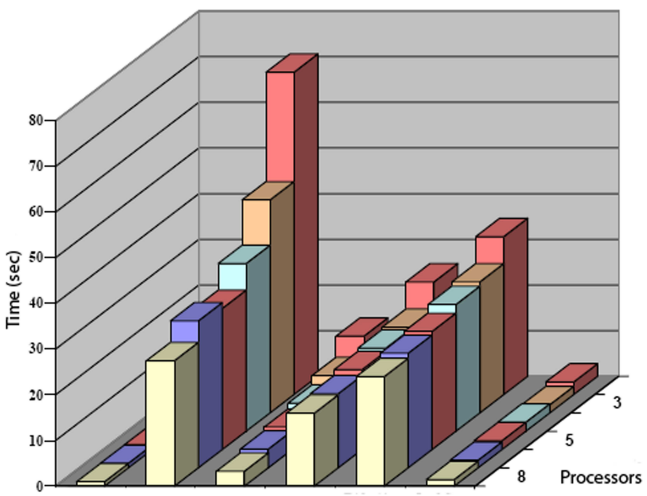
Figure 9: Efficiency of PROMESPAR

As it was previously commented, the most time consuming of PROMESPAR code is spend on main loop. Figure 10 show a detailled study of the time spend on main loop. It is possible to observe that *fisicapal*, *Coriolis* and *Difusion* functions spent the most quantity of time, and obviously the parallelization approach allows to reduce this execution time, overall from one to two processors. Anyway, the reduction of execution time results quite good.

5 Conclusion

PROMES is a mesoscale regional atmospheric model developed by some of the authors of this paper. However, due to the high time consuming by PROMES code and the necessity of having more accurate results, both circunstances justify the used of parallelism. In this paper, a distributed memory implementation of the regional atmospheric model PROMES has been carried out. This parallel implementation is called PROMESPAR.

The experimental results show a dramatically execution time reduction by means of the use of a parallel platform considering the same configuration that the original PROMES code. These results leads to think that either longer or more accurate simulations could be carried out spending the same time, or more complex models could be considered. In fact, the authors are extending PROMES code in order to be able of making climate change studies. Climate change studies consider 100 years simulations spending, obviously, lot of time and then if the researchers want to provide conclusions from these studies the use of parallelism becames essential.



	Antes_f	Fisicapal	Advecc.	Coriolis	Difusión	Int&Sta
8	1	27	3	16	24	1,44216
6	1	32,17	4,11579	16,2341	25	1,66534
5	1,04169	31	4,91797	17,5079	25,8628	1,9383
4	1,23075	37	6,15263	18,2263	27,7531	1,98593
3	1	46,6334	8,16989	18,7399	29	1,79524
2	1,96858	70,5729	12,783	24,6558	34,614	2,75723

Measuring points

Figure 10: Execution time of the main loop of PROMES-PAR for an hour simulation

Acknowledgments

The authors would like to thank to the Madrid supercomputer and visualization center (CESVIMA) to be able to use the supercomputer known as MAGERIT and for the valuable technical support to Jaime Pinto Rodriguez.

References

[1] M. Castro, C. Fernández, M. A. Gaertner. Descrition of a meso-scale atmospheric numerical model. *Mathematics, climate and environment*, 230–253, 1993.

[2] David L. Williamson Byron A. Boville James J. Hack, James M. Rosinski and John E. Truesdale. Computational design of the ncar community climate model. *Parallel Computing*, (21):1545–1569, 1995.

[3] Special Issue on Parallel Computing in Climate and Weather Modeling *Parallel Computing*, 21, 10, 1995.

[4] `http://www.irisgrid.es`. Spanish GRID Iniciative to coordinate both scientific and academic research groups.

[5] `http://www.eu-crossgrid.org/`. The primary objective of the CROSSGRID Project is to further extend the Grid environment to a new category of applications of great practical importance, and into 11 new European countries.

[6] `http://www.climateprediction.net/`. Climateprediction.net is a distributed computing project to produce predictions of the Earth's climate up to 2080 and to test the accuracy of climate models.

[7] Cam: Community atmosphere model. `http://www.ccsm.ucar.edu/models/atm-cam`.

[8] J. Dudhia D. O. Grell and D. R. Stuffer. A description of the fifth-generation penn state/ncar mesoscale model (mm5). Ncar/tn-398+str, National Center for Atmosphere Research, Boulder, Colorado, 1994.

[9] John Drake and Ian Foster. Introduction to the special issuue on parallel computing in climate and weather modelling. *Parallel Computing*, (21):1539–1544, 1995.

[10] Yun He and Chris H. Q. Ding. Mpi and openmp paradimgs on cluster of smp architectures: the vacancy traking algorithm for multi-dimensional array transposition. In *Proceedings of the IEEE/ACM SC2002 Conference*, 2002.

[11] R. Nanhyndiah S. Hammond J. Michalakes, T. Canfield and G. Grell. Parallel implementation, validation and performance of mm5. In *Proceedings of the Sixth ECMWF Workshop on the Use of Parallel Processors in Meteorology, World Scientific, River Edge, New Jersey*, pages 266–276, 1995.

[12] John Michalakes Brian Toonen John Drake, Ian Foster and Patrick Worley. Design and performance of a scalable parallel community climate model. *Parallel Computing*, (21):1571–1591, 1995.

[13] H. Kanamaru Cui Y. H. Juang Kanamitsu, M. Parallel implementation of the regional spectral atmospheric model. Pier energy-related environmental research cec-500-2005-016, Scripps Institution of Oceanography, University of California at San Diego, and National Oceanic and Atmospheric Administration for the California Energy Commission, 2005.

[14] Takashi ABE Hirofumi Sakuma Keiko Takahashi, Akira Azami and Tetsuya Sato. Developing coupled ocean-atmospheric global climate model for the earth simulator and its computational/physical validation. *NEC Res. and Develop*, 44(1), 2003.

[15] P. G. Eltgroth W. P. Dannevik C. R. Mechoso J. D. Farrara M. F. Wehner, A. A. Mirin and J. A. Spahr. Performance of a distributed memory finite difference atmospheric general circulation model. *Parallel Computing*, (21):1655–1675, 1995.

[16] J. Michalakes. Mm90: A scalable parallel implementation of the penn state/ncar mesoscale model (mm5). 1997.

[17] N. L. Miller. Recent advances in regional climate modelling and climate change analyses of extreme heat. Pier energy-related environmental research cec-500-2005-016, Lawrence Berkeley National Laboratory, for the California Energy Commission, 2005.

[18] L. Isaksen G. Robinson G. Mozdzynsky S. R. M. Barros, D. Dent and F. Wollenweber. The ifs model: A parallel production weather code. *Parallel Computing*, (21):1621–1638, 1995.

[19] J. G. Sela. Weather forecasting on parallel architectures. *Parallel Computing*, (21):1639–1654, 1995.

[20] John M. Dennis Steven Hammond, Richard D. Loft and Richard K. Sato. Implementation and performance issues of a massively parallel atmospheric model. *Parallel Computing*, (21):1593–1619, 1995.