ADVANCES IN ATMOSPHERIC SCIENCES, VOL. 20, NO. 1, 2003, PP. 159 163

Distributed Parallelization of a Global Atmospheric Data Objective Analysis System

ZHAO Jun[1] (赵 军), SONG Junqiang[1] (宋君强) and LI Zhenjun[2] (李振军)

[1] *College of Computer Science, National University of Defense Technology, Changsha* 410073

[2] *Meteorological Center of Airforce, Beijing* 100843

(Received March 6, 2002; revised December 9, 2002)

ABSTRACT

It is difficult to parallelize a subsistent sequential algorithm. Through analyzing the sequential algorithm of a Global Atmospheric Data Objective Analysis System, this article puts forward a distributed parallel algorithm that statically distributes data on a massively parallel processing (MPP) computer. The algorithm realizes distributed parallelization by extracting the analysis boxes and model grid point latitude rows with leaped steps, and by distributing the data to different processors. The parallel algorithm achieves good load balancing, high parallel efficiency, and low parallel cost. Performance experiments on a MPP computer are also presented.

Key words: distributed parallelization, analysis box, data distribution, objective analysis

P4 A

1. Introduction

Medium range numerical weather prediction is a typical kind of high-performance computation problem. It is of great importance to the objectivity, quantification, and automation of weather prediction and the improvement of weather forecast veracity. It predicts a future atmospheric state by analyzing the atmospheric state on initial and boundary conditions, and solving the atmospheric basic equation system. The objective analysis system is the main part of the data assimilation system and it combines the information from the first guess field, typically a six-hour forecast valid at the analysis time, and the observed quantities, taking error estimates into account (Pfaendtner et al. 1995, da Silva et al. 1995).

In this paper, the objective analysis system is based on the optimal interpolation (OI) algorithm introduced by Gandin (von Laszewski, 1996). It uses a statistical process based on mean square minimization to obtain the missing values for the computational grid (Mu et al., 2002). It takes a first guess for the fields and observations. Then, for each grid point, actual observations are used in order to obtain an analysis increment, based on the weighted sum of all observations in the vicinity of the point.

The methods to parallelize the objective analysis system based on the OI algorithm were proposed by Baker et al. (1987), Isaksen (1992), and von Laszewski et al. (1994), using locking mechanisms and dynamic load balancing, respectively. But these methods occupy massive resources on a MPP computer, so they are not appropriate for such a computer.

This article presents a distributed parallel computation strategy of the objective analysis system on a MPP computer (Tanenbaum, 1995). A static load balancing method based on MPI (Message Passing Interface Forum, 1994) will be used in this strategy. First, the sequential algorithm of a Global Objective Analysis System will be introduced briefly. Then, the distributed parallel algorithm will be presented. And last, the parallel efficiency of the parallelized objective analysis system will be discussed.

2. Sequential algorithm

2.1 *Optimal interpolation method*

The optimal interpolation method has been described in numerous textbooks and papers (e. g., Baker et al., 1987; Daley, 1991), so it will only be introduced briefly here.

The basic interpolation equations are illustrated below, in which A represents a random scalar, E the root-mean-square error of this scalar, and subscripts i, p, o, and t the interpolation value, prediction value, observation value, and true value, respectively.

$$\frac{A_{k,\mathrm{i}} - A_{k,\mathrm{p}}}{E_{k,\mathrm{p}}} = \sum_{n=1}^{N} w_{kn} \frac{A_{n,\mathrm{o}} - A_{n,\mathrm{p}}}{E_{n,\mathrm{p}}}$$

*E-mail: frank611@163.net

Denote

$$\alpha_{n,\mathrm{o}} = \frac{A_{n,\mathrm{o}} - A_{n,\mathrm{t}}}{E'_{n,\mathrm{o}}},$$
$$\alpha_{n,\mathrm{p}} = \frac{A_{n,\mathrm{p}} - A_{n,\mathrm{t}}}{E'_{n,\mathrm{p}}},$$
$$\alpha_{k,\mathrm{i}} = \frac{A_{k,\mathrm{i}} - A_{k,\mathrm{t}}}{E_{k,\mathrm{i}}},$$
$$\varepsilon_{n,\mathrm{o}} = \frac{E_{n,\mathrm{o}}}{E_{n,\mathrm{p}}}, \qquad (1)$$
$$\varepsilon_{k,\mathrm{i}} = \frac{E_{k,\mathrm{i}}}{E_{k,\mathrm{p}}}.$$

Assuming that the correlation between prediction error and observation error is zero, and minimizing the expected variance of the normalized interpolation error to ascertain the optimal weight of each piece of observation data, we get the optimal interpolation equation,

$$\frac{A_{k,\mathrm{i}} - A_{k,\mathrm{p}}}{E_{k,\mathrm{p}}} = \mathbf{B}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{P}_k = \mathbf{C}^{\mathrm{T}}\boldsymbol{P}_k,$$

where $\boldsymbol{M} = \boldsymbol{P} + \boldsymbol{O} = (\langle\alpha_{m,p}, \alpha_{n,\mathrm{p}}\rangle + \varepsilon_{m,o}\langle\alpha_{m,o}, \alpha_{n,o}\rangle \varepsilon_{n,o})$, $\mathbf{B}$ is the vector of normalized increment value $(A_{n,o} - A_{n,p})/(E_{n,\mathrm{p}})$, and P_k vector of the prediction error correlation $\langle\alpha_{k,p}, \alpha_{n,\mathrm{p}}\rangle$.

2.2 *Box trees*

In the objective analysis system, a box method is utilized, and the following presents the process of constructing box trees.

(a) Construct base box trees

Divide the earth's surface into boxes of approximately equal area. For example, there may be 32 rows of base boxes whose sides are 5.625 longitude-latitude degrees between the two poles. For each base box, specify its maximal and minimal data-selecting areas. In the minimal data-selecting area, count the amount of data in each vertical level.

(b) Generate sub-box trees

If the amount of data in the minimal data-selecting area exceeds the allowable maximal matrix dimension (a predefined constant, such as 451), divide the base box into four sub-boxes of equal area. For each new box, specify its maximal and minimal data-selecting areas. Repeat the process until the amount of data in the minimal data-selecting area of a box is smaller than the allowable maximal matrix dimension. The generated new boxes constitute the sub-box trees.

In the objective analysis system, computation of box-trees is the computation of each leaf of the box-trees, that is, the final analysis boxes.

2.3 *Sequential algorithm in the objective analysis system*

The objective analysis system consists of several modules which include initial values, inverse Legendre transform of the background field, observation data processing scan, mass and wind field analysis, humidity analysis, and direct spectral transform of the analysis field. The mass and wind field analysis module includes four main steps:

(a) Create super-observations

(b) Check data and eliminate errors

The algorithm is illustrated as follows:

```
create box-trees;
for each final analysis box do
    calculate observation-observation error correlation matrix;
    invert the error correlation matrix;
    calculate observation-grid-point error correlation matrix;
    calculate analysis errors;
end for
```

(c) Evaluate analysis coefficient

The algorithm is as follows:

```
create box-trees;
for each final analysis box do
    calculate observation-observation error correlation matrix;
    solve the linear equation system;
end for
```

(d) Calculate grid-point analysis value

The algorithm is as follows:

```
for each grid-point latitude row do
    find all influencing boxes;
    calculate observation-grid-point error correlation matrix;
    calculate virtual temperature increments;
    calculate the analysis values of u and v;
    calculate the logarithms of surface pressure;
end for
```

The humidity analysis module is similar to the mass and wind field analysis module, only differing in the analysis variable relative humidity.

3. Parallel strategy

In section 2.3, the sequential algorithm of the objective analysis system was briefly introduced. After analyzing components of the run-time of the sequential programs, we find that the run-time of the mass and wind field module and the humidity module takes more than 90% of the total run-time, and the proportion increases with an increase in the amount of observational data. Nearly all of the run-time (about 95%) of the two modules is spent on data checking, evaluating the analysis coefficient, and calculating the grid-point analysis value. Therefore in order to decrease the cost of parallelization, only these modules will be parallelized.

The load balancing problem is very important in parallelization. It determines the means of data distribution. An analysis box is the computation unit of

the data checking module and analysis coefficient evaluating module. If each analysis box is independent, the computation can be parallelized. The computation unit of the grid-point analysis value evaluating module is a grid-point latitude row, and the generated grid-point analysis values are saved according to latitude rows, so this module can be parallelized according to latitude rows. The respective parallelizations of these modules are introduced below.

3.1 *Parallelization of data checking module and analysis coefficient evaluating module*

As stated above, the computation unit of the two modules is an analysis box, so they can be parallelized by distributing a part of the total collection of analysis boxes to each processor. The problem is how to distribute the analysis boxes to balance the load on each processor.

There are two methods of data distribution, static distribution and dynamic distribution. Dynamic distribution assigns tasks according to the actual computation state of each processor. This method can obtain a good load balancing, but the computer code is complicated and occupies many resources, which are very massive for a MPP computer. Therefore, the static data distribution will be adopted in the process of parallelization.

We have studied the distribution of observational data on the earth, and found that it is very uneven: the distribution is thick in the continents and thin in the oceans and at the poles. But the distribution in the continents is regular: the atmospheric data is thickly distributed mainly in America, Europe, and Asia. The difference in the amount of data between two neighboring analysis boxes among the box trees constructed in the data checking module and analysis coefficient evaluating module is very small. This means that the computation load in two neighboring analysis boxes is almost the same. So by selecting the analysis boxes with leaped steps in the thick data region we can balance the load on each processor. For the thin data region, since the data in the analysis boxes is scarce, the computation load in these boxes is very light and has little influence on the load balancing of the whole system.

The parallel algorithm in the data checking module is as follows:

Let N_b be the total number of analysis boxes and N_P the total number of processors.

broadcast the super observation data to each processor;

construct box trees on each processor for data checking (the computing cost of box trees is less than the communication cost);

for processor=1 **to** N_p **do**

for box=processor **to** N_b, stepsize=N_p **do**

calculate observation-observation error correlation matrix;

invert the error correlation matrix;

calculate observation-grid point error correlation matrix;

calculate analysis errors;

end for (boxes)

end for (processors)

gather the computing results on each processor by means of reducing and combining, then broadcast the whole result to each processor.

The parallel algorithm in the analysis coefficient evaluating module is as follows:

construct box trees on each processor for analysis coefficient evaluating (the computing cost of box trees is less than the communicating cost);

for processor=1 **to** N_p **do**

for box=processor **to** N_b, **stepsize**=N_p **do**

calculate observation-observation error correlation matrix;

solve the linear equation system;

end for (boxes)

end for (processors)

gather the computing results on each processor by means of reducing and combining, then broadcast the whole result to each processor.

In this way, the data checking module and analysis coefficient evaluating module are parallelized. After testing, we find that the loads on each processor are approximately balanced, with the load difference being about 5%. The communication cost is not massive and it increases very slowly with the increase in the number of processors. So the communication has little influence on the efficiency of parallelization.

3.2 *Parallelization of grid-point analysis value calculating module*

The computation unit of this part is the model grid-point latitude row. Since the process of each latitude row is independent, the earth is divided according to grid-point latitude rows and different processors work on different latitude rows. Because the amount of data in different latitude rows is different, computations of each latitude row are not equivalent. So the earth cannot be divided simply into several latitude bands. Since the distribution of global atmospheric data is relatively concentrated, latitude rows can be selected with leaped steps so that each processor can process a latitude row with heavy or light computation in turn. In this way, the load on each processor can be balanced.

The parallel algorithm to calculate grid-point analysis values is as follows:

Let N_r be the total number of latitude rows.

for processor=1 **to** N_p **do**

for row=processor to N_r, **stepsize**=N_p **do**

find all influencing boxes in a row;

calculate observation-grid point error correlation matrix based on the initial value of surface air pressure;

calculate virtual temperature increments;

calculate the analysis value of u, v;

calculate the logarithms of surface pressure in the row;

end for (rows)

end for (processors)

gather the computing results on each processor to transmit to No. 1 processor, reducing and combining them.

The length of the final computation result array is a constant and does not change with an increase in the number of processors. Furthermore, when the number of processors increases, the increasing proportion of communication is very small, so the total communication will not exceed the computing result array, thus communication has little influence on the parallel efficiency.

4. Analysis of results and conclusion

Table 1 shows the load balancing state of an non-balancing parallel algorithm that sequentially selects analysis boxes. Table 2 shows the load balancing state of the balancing parallel algorithm. The number of processors is eight. The time when the data is gathered is 1200 UTC 15 July, 1999. Modules 1, 2 and 3 represent the data checking module, analysis coefficient evaluating module, and grid-point analysis value calculating module, respectively.

From the tables, we can see that an uneven data distribution leads to a poor load balancing. A good load balancing can be achieved by using the parallel algorithm presented in this article.

Table 3 shows the speedup and parallel efficiency of this balancing parallel algorithm in the parallelizing data checking module, analysis coefficient evaluating module, and grid-point analysis value calculating module in the cases of 1, 2, 4, 8, and 16 processors.

Two groups of data are selected to test the parallel efficiency of the parallelized global atmospheric data objective analysis system on a high-performance MPP computer. The two groups of data are gathered in different seasons and at different times, so they are representative. The two times are 1800 UTC on 15 October, 1998 and 1200 UTC on 15 July, 1999. There are 11271 reports in the analysis observation file for the first group of data, and 13381 reports for the second group.

From the above, we see the load is approximately in balance, the parallel efficiency is high, and the parallel method is scalable within a limited number of processors. At the same time, the parallel efficiency in the first group of data is lower than that in the second because the amount of atmospheric data in the first is smaller than in the second. In the case that the amount of atmospheric data increases rapidly in the future, the computation loads of the three modules will increase rapidly. Then more processors can be used. The communication load will increase, but the proportion to the total load is very small. So, good parallel efficiency can still be obtained. The distributed parallel method presented in this paper can be easily scaled to a distributed memory parallel system with 32 or 64 processors. In other words, it has good scalability.

Table 1. Load balancing state of a non-balancing parallel algorithm

	CPU 0	CPU 1	CPU 2	CPU 3	CPU 4	CPU 5	CPU 6	CPU 7	Max/min
Module 1	8.139	11.209	6.365	4.559	2.334	1.504	**0.167**	**19.164**	114.75
Module 2	6.673	10.182	**10.454**	7.991	6.427	3.673	**0.578**	3.135	18.09
Module 3	**113.695**	104.196	80.855	53.713	29.487	12.351	3.868	**2.138**	53.18

Table 2. Load balancing state of the balancing parallel algorithm

	CPU 0	CPU 1	CPU 2	CPU 3	CPU 4	CPU 5	CPU 6	CPU 7	Max/min
Module 1	6.630	**6.972**	**6.405**	6.450	6.584	6.917	6.660	6.572	1.089
Module 2	6.781	6.421	6.695	6.541	**6.860**	**6.319**	6.800	6.438	1.086
Module 3	56.854	**56.554**	57.079	57.338	57.298	**57.387**	56.928	56.986	1.015

Table 3. Speedup and parallel efficiency of the balancing parallel algorithm

Number of processors	1800UTC 15 Oct. 1998		1200 UTC 15 Jul. 1999	
	Speedup	Efficiency	Speedup	Efficiency
1	1	100%	1	100%
2	1.8412	92.06%	1.89	94.50%
4	3.6376	90.94%	3.7044	92.61%
8	6.8568	85.71%	6.9008	86.26%
16	12.832	80.20%	13.232	82.70%

REFERENCES

Baker, W. E., S. C. Bloom, John S. Woollen, Mark S. Nestler, Eugenia Brin, Thomas W. Schlatter, Grant W. Branstator, 1987: Experiments with a three-dimensional statistical objective analysis scheme using FGGE data. *Mon. Wea. Rev.*, **115**(1), 272–296.

Daley, R., 1991: *Atmospheric Data Analysis*, Cambridge University Press, 457pp.

Isaksen, L., 1992: Parallelizing the ECMWF optimum interpolation analysis. *Proceedings of the fifth ECMWF workshop on the use of parallel processors in meteorology*, (Eds. M. Ghil, R. Sadourny and J. Südermann, Singapore, 240–249.

Message Passing Interface Forum, 1994: A Message-Passing Interface Standard. (*draft obtainable at ftp://info.mcs.anl.gov/ pub/mpi*). Version 1.0.

Mu M., Duan W. S., Wang J. C., 2002: The predictability problems in numerical weather and climate prediction. *Advances in Atmospheric Sciences*, **19**, 191–204.

Pfaendtner, J., S. Bloom, D. Lamich, M. Seablom, M. Sienkiewicz, J. Stobie, and A. da Silva, 1995: Documentation of the Goddard Earth Observing System (GEOS), Data Assimilation System-Version 1. NASA Technical Memorandum 104606, Vol.4, NASA GSFC Data Assimilation Office, Greenbelt, Maryland, Jan. 1995.

da Silva, A. , J. Pfaendtner, J. Guo, M. Sienkiewicz, and S. E. Cohn, 1995: Assessing the effects of data selection with DAO's physical-space statistical analysis system. *Proceedings of the International Symposium on Assimilation of Observations in Meteorology and Oceanography*, (Tokyo, Japan, World Meteorological Organization, 273 278.

Tannenbaum, A. S. 1995: *Distributed Operating Systems*. Prentice Hall, 648pp.

von Laszewski, G., Mike Seablom, Miloje Makivic, Peter Lyster, Sanjay Ranka, 1994: Design issues for the parallelization of an optimal interpolation algorithm, 4th Workshop on Parallel Processing in Athmospheric Science, Edinburgh, UK.

von Laszewski, G., 1996: A parallel data assimilation system and its implications on a metacomputing environment, Ph.D, thesis, Syracuse University, 204pp.

全球气象资料客观分析系统的分布式并行化

赵 军 宋君强 李振军

摘 要

对已有的串行算法进行并行化，是一项很困难的工作。通过对全球气象资料客观分析系统串行算法的研究，提出了在MPP高性能计算机上的一种静态分配数据的分布式并行算法。该算法通过间隔选取分析盒子和模式格点纬圈行，将数据分配给不同的处理机实现分布式并行。该并行算法负载平衡好，并行效率高，而且并行化代价较低，具有良好的可扩展性。最后，给出了并行算法的性能测试结果。

关键词：分布式并行，分析盒子，数据分配，客观分析