High Performance Computing Model for Processing Meteorological Data in Cluster System

Li Dequan, Wang Ying, Xiong Anyuan, Tinghuai Ma

1 National Meteorological Information Centre (NMIC), China Meteorological Administration (CMA), 10086 Beijing, China {lidq, cdcjk, xay}@cma.gov.cn
2 School of Computer & Software, Nanjing University of Information Science & Technology, 210044, Nanjing, China thma@nuist.edu.cn
doi:10.4156/jcit.vol6. issue4.11

Abstract

Meteorological observational data is collected from many stations around the country at the same time, and immediately submitted to the national meteorological data center-NMIC. NMIC aggregates national data and carries out effective processing and rapid service. This paper describes a high performance computing model in cluster environment to process massive meteorological data, which employs parallel computing and distributed computing frameworks focused on task division, task scheduling management, load balancing methods on various types of meteorological data in the Master-Slave operation environment. Within implementation of this architecture, high ability to efficiently handle large amount of content and good usability and stability of the data management and data service are achieved.

Keywords: Parallel Computing, Distributed Computing, Meteorological Data, Cluster, Scheduling Management

1. Introduction

Meteorological data (i.e., winds, air temperature, atmospheric stability, mixed layer heights, etc.) is the atmospheric science data with typical time-series and space properties. Meteorological data has the following characteristics: source complexity of a wide range of diverse formats, different forms, such as huge amount of data. Directed by China Meteorological Administration (CMA), an integrated atmospheric observation system has been set up in China, incorporating sky-, space- and ground-based subsystems, covering relatively complete weather elements, including geographically balanced station sites [1-2]. There are three levels in the national observation system: observational stations, data departments of province, national data center. The meteorological observation data are collected and submitted by three levels of the business structure level by level. Observational data come from stations around the country and transfer to the host provincial departments. After aggregated by the provincial departments, data are uploaded to the national data center. The National Meteorological Information Center (NMIC) of CMA has such responsibilities as the national meteorological data center for meteorological data collection, processing, storage, archive, and services of weather forecasting and researches and so on. Based on CMA’s standards, all meteorological data are divided into 14 categories. These data can be widely used for various types of meteorological service systems: weather forecast, climate prediction, weather modification, drought and flood monitoring and forecasting, thunderstorm and Lightning prevention, agro-meteorology and eco-meteorology, climate resource exploitation, etc.

The file number and capacity of data received in the national data center for each day are very large [3]. For example, NMIC receives tens of millions of files and about 600 GB each day (information of parts is shown in Table 1). On the other hand, most of meteorological data are observed with a certain frequency in schedule. In other words, different stations usually observe at the same time, while uploading and immediately to provide data service to the weather analysis system for forecasting and forecasters. The whole process requires a high timeliness. This manipulation will inevitably has cyclical peak, which require fast, high-performance computing to meet the requirements. In particular, some data would increase more than 50% of the amount in special some period such as the Flood Season [4].
Table I shows data statistics for NMIC to handle. The table only chooses two typical data: weather radar data and surface data, and four typical observation times such as 0, 6, 12, 18 hours.

<table>
<thead>
<tr>
<th>Time</th>
<th>Surface data File number</th>
<th>Surface data Capacity</th>
<th>Radar data File number</th>
<th>Radar data Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0Z</td>
<td>78748</td>
<td>25.7 MB</td>
<td>33732</td>
<td>~1GB</td>
</tr>
<tr>
<td>6Z</td>
<td>78748</td>
<td>25.7 MB</td>
<td>33732</td>
<td>~1GB</td>
</tr>
<tr>
<td>12Z</td>
<td>80149</td>
<td>26.1 MB</td>
<td>33732</td>
<td>~1GB</td>
</tr>
<tr>
<td>18Z</td>
<td>78748</td>
<td>25.7 MB</td>
<td>33732</td>
<td>~1GB</td>
</tr>
</tbody>
</table>

a. Z means the universal time (Greenwich Mean Time)

When the observational data submitted to NMIC, the data must complete a series of processes before service. The processing includes format inspection, quality control, Data transmission, data storage and so on. Therefore, faced with such huge number of observation files and high real-time service support requirements, high performance computing is introduced to improve processing efficiency [4-12].

2. Two processing modes for different meteorological data

There are two typical cases of meteorological data processing from its own characteristics. One type is often with the small number of files, but each file has a great capacity, such as weather radar data, satellite data, etc. Such data processing needs complex operations and long time consuming. Furthermore, such data files are suitable for dividing a single data into several partitions to execute parallel processing (based on some specific parallel algorithm). The other type of data is the small size for single file while significant number of all files. Such data is suitable for distributed processing of multiple data files in the same time.

Correspondingly, there are two processing frameworks as follows:

2.1. SDMP (Single-Data and Multi-stream Processing) Mode

The conditions for applying SDMP mode is as follows.
- Large amount of capacity of input data (small number of files).
- Data processing method is very complicated and with complex algorithms for large processing overhead.
- Process (including algorithms) can be divided into many independent parts.
The main process of the SDMP mode is shown in Fig. 1.
2.2. MDMP (Multiple-data and Multi-stream Processing) Mode

The conditions for applying MDMP mode is as follows.
- Large amount of files and small file size (single file is general 50MB or less).
- Process relatively simple, lower complexity to algorithms.
- The process can not be split into several independent parts.

The main process of the MDMP mode is shown in Fig. 2.

3. Methods

Many previous works process large amounts of data through the grid and parallel computing algorithms [13-16]. This paper focuses on the two modes and discusses their applications in meteorological data processing field. The MDMP mode is suitable to handle large amount of regular meteorological report packets. However, the SDMP mode is suitable to create the mosaic of many weather radars (the data required for the radar network can be considered as a whole input data). As is known to all, the mosaic of weather radar can conquer the limit of single radar’s range and is one of important service products for short-range forecast and detecting and monitoring the severe weather [17-18].
The running environment is the cluster system within the Master-Slave framework. Scheduling control will take effect in the Master-Slave parallel computing environment. Master-Slave structure is simple and easy to maintain.

3.1. Task scheduling management

When a new task (a business task often contain a number of data packages to manipulate) requests to the business management system for scheduling new business works, the business management system sends a request message to the master node. The master node receives a request, generates a new task and adds it into the task queue. The scheduling control system in the master node picks up a task from the task queue, and separates the task into several independent jobs at first, thus depending on the load balancing calculation, assigns jobs to the corresponding slave nodes to perform job calculations. The overall process is shown in Figure 3.

The master node sorts the priority weight of every task in the queue, and determines which task is first run. Through the job scheduling, jobs within a task are distributed to different computing nodes in the system.

![Figure 3. Process for a task](image)

3.2. Dynamic Load Balancing Weight Calculation

The dynamic load balancing weight calculation is used to solve the load balancing problems between nodes’ settlement and distribution of jobs from one task.

Task in this paper refers to a work need for data processing, including business process description (data processing). For example, on every hour in a day to deal with the data uploaded from automatic weather station in all provinces; or to handle four mosaics of participating network consist of 158 weather radars each hour.

Job refers to executable units in each slave node. Combined with characteristics of parallel algorithms, the task handles the data according to their characteristics and divides it into the same executable units, which are unrelated to the specific business. For example: on the automatic mission
weather station data processing tasks can be divided into many jobs (executable units) with measure of regions or reports, or the mosaic mission of 158 radars be divided into hundreds of different jobs with regional and network layers. Each job has only a small part of the overall task.

A task consists of multiple jobs, and each job has a job weight. The sum of all job weights in a task is equal to this task weight. The task weight $W_t$ is:

$$W_t = \sum_i \omega_i$$  \hspace{1cm} (1)

Where $\omega_i$ is the $i$th job weight in the task.

Each slave computing node has a weight threshold to limit the workload of the node. Calculated the sum of job weights for all running jobs in each slave node, the sum value is first compared with the weight threshold value. When the slave computing node accepts a job, the sum value will be refreshed by adding job weight to current total weights to form a new value. Correspondingly, when the current total weight value are more than the weight threshold value, the slave node will no longer receive new jobs until the current resources are released.

A job with its weight $\omega_0$ can be assigned to slave computing node $s$ whose job weight threshold is $\omega_\theta$, by the following formula to determine.

$$F_{assign}(s) = \begin{cases} 
0, & \text{if } \sum_i \omega_i > \omega_\theta \\
1, & \text{otherwise}
\end{cases}$$  \hspace{1cm} (2)

Where $F_{assign}(s)$ is a Boolean function to determine whether this job can be assigned to slave node $s$. $0$ means it can be assigned, and $1$ means not to be assigned.

4. Results and discussion

The above models have been implemented in the real-time meteorological data processing system. Faced to large amount of different types of data with very different characteristics, the meteorological data processing system need not only employs distributed computing (processing regular meteorological report packets), but also requires high complexity of parallel computing (radar network processing) for a large amount of data. Furthermore, because of the great differences with processes and algorithm between distributed computing and parallel computing applications, the design must be targeted.

Based on efficiency considerations, the main parts of meteorological data processing system including the main program, algorithm library, the algorithm plug-ins are developed with C / C++ language. However, some algorithms are written in FORTRAN, which can improve resource utilization because interfaces between C and FORTRAN are more efficient.

Various configurations and task description of the meteorological data processing system are written using LUA scripting language. LUA can be used as an embedded script to the host program written in C / C++ language.

In addition, the data processing system uses MPI as the basis for distributed and parallel computing in order to significantly simplify construction of the distributed and parallel computing environment, and improve the stability of whole system.

This model in the real application process has also considered the fault treatment strategy to respond to possible failures. When one slave node was crash or had hardware failure, the master node will pick up its tasks and distribute its tasks to other nodes based on load balancing strategy. When the master node’s crashes lead to failure, one slave node which has the shadow of the master program will consults with other slave nodes and create a temporary master program, while the temporary master node reduces its load threshold to ensure its stability.
5. Conclusions and Future Works

In this paper, a high performance computing model in cluster environment is proposed to offer service of high efficiency and high quality to real-time meteorological data processing, which employs parallel computing and distributed computing framework. This model consists of many technical contributions, including task division, task scheduling management, load balancing methods, and so on. Although the model is proposed for real-time meteorological data, it can be generalized to many other types of data which requires high performance and real-time computing. As result, within implementation of this architecture, real-time meteorological data processing system significantly improved the capacity and emergency treatment to effectively respond to high-intensity tasks burden in flood season temporarily. Additionally, good business and better usability and stability are achieved.

Further work involves analysis, design and implementation of a well-targeted load distribution mechanism that will enable different scheduling strategy selections to different types of meteorological data. Moreover, in the MDMP model, aimed to ensure that each node attain roughly equal capacity , adding more sophisticated strategy supports for task assignment within a balanced layout according to the sub-classification for each type of meteorological data.

6. Acknowledgments

This work was supported by the project of China Integrated Meteorological Information Service System (CIMISS) in China Meteorological Administration.

7. References