Scientific Research Publishing

Journal of Geoscience and Environment Protection, 2017, 5, 75-83
http://www.scirp.org/journal/gep
ISSN Online: 2327-4344
ISSN Print: 2327-4336

Study on Massive Vegetation Data Processing of FY-3 Based on RAM (h)

Manyun Lin, Xiangang Zhao, Cunqun Fan*, Lizi Xie, Lan Wei

National Satellite Meteorological Center, Beijing, China
Email: *fancq@cma.gov.cn

How to cite this paper: Lin, M.Y., Zhao, X.G., Fan, C.Q., Xie, L.Z. and Wei, L. (2017) Study on Massive Vegetation Data Processing of FY-3 Based on RAM (h). *Journal of Geoscience and Environment Protection*, **5**, 75-83.
https://doi.org/10.4236/gep.2017.54007

Received: December 21, 2016
Accepted: April 22, 2017
Published: April 25, 2017

Abstract

The vegetation data of the Fengyun meteorological satellite are segmented according to the latitude and longitude, and can be written into 648 blocks. However, the vegetation data processing efficiency is low because the data belongs to massive data. This paper presents a data processing method based on RAM (h) for Fengyun-3 vegetation data. First of all, we introduce the Locality-Aware model to segment the input data, then locate the data based on geographic location, and finally fuse the independent data based on geographical location. Experimental results show that the proposed method can effectively improve the data processing efficiency.

Keywords

Meteorological Satellite, Vegetation Data, RAM (h), Massive Data Processing

1. Introduction

The Fengyun-3 (FY-3) satellite is a near-polar solar synchronous orbit satellite, in which the high-speed instrument generates a data file every five minutes. It can obtain up to 288 five-minute data in 24 hours. The vegetation data of FY-3 are widely used in the fields of environment, geography and agriculture. The current vegetation data are written into 648 blocks, which are divided by the latitude and longitude of the ten degrees, the size of 4000 × 4000 pixels. Due to the large amount of vegetation data, the corresponding data I/O amount and calculation amount are very considerable, resulting in the low efficiency of vegetation data processing. At present, it has made good research results for the data processing research. The originality of this study [1] comes with an introduction of linguistic values articulated in terms of component failure possibilities in order to qualitatively assess basic event failure possibilities treated as inputs of the proposed model and generate basic event failure probabilities as its outputs. The paper [2] deals with several data integration and data processing issues that frequently occur within this context. To this end, the data processing workflow within the PROFILE project is presented, a multi-OMICS project that aims on the identification of novel biomarkers and the new therapeutic targets for severely important. The paper [3] comprehensively reviews the development of GPS surveys and their applications, and GPS data processing. Different from most reviews in GPS research, this paper provides a detailed and systematic comparison between different methods from the trip identification to mode and purpose detection, introduces the methods that researchers and planners are currently using, and discusses the pros and cons of those methods. The paper [4] summarizes the basics of pulsed thermal nondestructive testing including theoretical solutions, data processing algorithms and practical implementation. Typical defects are discussed along with 1D analytical and multi-dimensional numerical solution. Special emphasis is focused on the defect characterization by the use of inverse solutions. The paper [5] proposes an upgrade version MGMR++ to eliminate GPU memory limitation and a pipelined version, PMGMR, to handle the Big Data challenge through both CPU memory and hard disks. MGMR++ is extended from MGMR with flexible C++ templates and CPU memory utilization, while PMGMR fine-tuned the performance through the latest GPU features such as streams and Hyper-Q as well as hard disk utilization.

Based on the above research, a data processing method based on RAM (h) for Fengyun-3 vegetation data is proposed. The parallel processing of RAM (h) is used to block the vegetation data according to the latitude and longitude. First of all, we introduce the Locality-Aware model to segment the input data, then locate the data based on geographic location, and finally fuse the independent data based on geographical location.

2. Parallel Method Based on RAM (h)

The data read and write performance can be described and estimated using RAM (h) (RAM with h-level Memory Hierarchies) parallel computing model. Generally speaking, in this multi-layer storage system, the farther away from the processor, the greater the storage capacity, the slower the memory access speed; on the contrary, the closer to the processor, the smaller the storage capacity, but the faster the storage access.

The RAM (h) parallel computing model is based on the RAM (Random Access Machine) model, which extends the data storage, where h is the number of system memory layers. The core of the RAM (h) model is based on two assumptions:

(1) Data in the model use block as the basic unit of transmission between the various storage layers.

(2) In the beginning of the program, the required data stored in the storage layer farthest away from the processor. Only in the need of calculation it will be transferred to the nearest layer of the processor layer by layer. In this process, each layer of storage will retain a copy of the data until the end of the program

or due to mapping conflicts or insufficient storage capacity is replaced by other data. Subsequently, if the data is to be accessed repeatedly, the data is extracted from the storage layer at which it resides closest to the processor. The cost of each store access is therefore determined by the number of layers of data residing nearest the processor and the data reuse rate (locality of data access). The lower the number of layers or the higher the data reuse rate, the smaller the cost, and vice versa the greater the cost.

For a data block with storage layer h and length l, the cost of the first access can be expressed as

$$C(l)=\left\lceil \frac{l}{b(h-1)} \right\rceil c(h)+\left\lceil \frac{l}{b(h-2)} \right\rceil c(h-1)+\cdots+\left\lceil \frac{l}{b(0)} \right\rceil c(1). \tag{1}$$

where $\lceil\ \rceil$ is the round-up operation; l is the length of the data block to be obtained; $b(i)$ ($0 \le i \le h$) is the unit data block size of the i-th storage layer, in general, there are $b(i-1)\le b(i)$; $c(i)$ ($1 \le i \le h$) is the time required for the (i-1)th layer to read and write the unit data block of the size $b(i-1)$ from the i-th layer, in general, there are $c(i-1)\le c(i)$.

Because in a given computing environment, $c(i)$ is constant. Analysis RAM (h) model data access cost formula, we should reduce the cost of data access $c(l)$ value.

Based on the above method analysis, the core idea of this paper is: Starting from the ten-degree block, the original operation of the global data is divided into 648 separate tasks concurrently. At the same time, data segmentation technology based on the Locality-Aware model is used for data block reading and writing. Each task will be the corresponding five minutes of data fusion for the final ten-degree block data. All tasks are scheduled and managed by the dynamic task scheduling strategy based on data dependency, and finally generates all the target data.

3. Input Data Segmentation Method Based on Locality-Aware

Considering that blade cluster and parallel file system adopt multi-layer storage mechanism (including register, cache and buffer), good data access locality can greatly improve the application performance. Especially in the process of parallel program implementation involving massive data processing, high-performance data reading and writing is a very important part of program design. How to operate these data rationally is the key to determine the performance of parallel algorithm and program.

The basic principle of the Locality-Aware model is to filter all the input data, and to filter the same or related data to improve the locality of data access. he LARD (Locality-Aware Request Distribution) algorithm is the representative algorithm for this model-distributing the same content request to the same server that provides the relevant content so that the content requested by the client is found in the cache of the server as much as possible. The LARD algorithm maintains a mapping table of service contents and backend service node sets for fast retrieval.

In the blade cluster environment, a single blade node available memory size is very limited. And there are a large number of invalid data in the input data file for the current processing task. In other words, when the program is only using part of the data in the file to process, the importance of local data access principle is highlighted.

Because of the large amount of input data and the need for repeated read, so hard splitting the input data (that is to split into multiple independent small files) generated by the cost of time and space is unbearable, also not necessary. A feasible and feasible scheme is to determine the dependency between the current task and the input data file from the order of file composition and data storage, and realize the soft segmentation of the file by marking the effective data block position in the original file. This method is to achieve the logical division of the input data, but also to avoid taking up additional storage space, the resulting time overhead is entirely affordable. Input data pre-processing of the core processing link is the projection look-up table file and five minutes of data for positioning analysis. The following will be the detailed description from the projection look-up table positioning and positioning of five minutes.

3.1. Location of Projection Look-up Table Files

Using the row and column offset and row and column values in the file header information of the projection look-up table, the ID number set of the ten-degree block corresponding to the projection look-up table file can be estimated. Determine all ID numbers in the collection:

Locate the projection look-up table file is to find the projection look-up table file with a ten-degree block coincidence area of the location of the upper left corner of the region and the value of the overlap area. As shown in Figure 1, if the location of the lower right corner of the blue area of information, you need to calculate the location of the A point and x, y value. While the location of point

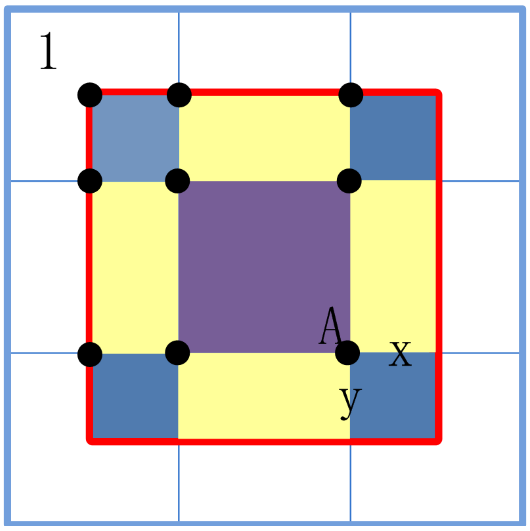

Figure 1. Projection look-up table file and ten degrees block coincidence diagram.

A is the key to the calculation, when the value of point A is obtained, the value of x, y is also derived from the ranks of A value. The calculation of the value of point A is divided into two steps.

(1) Calculate the logical offset of point A.

The logical offset of point A is the actual geographical position of point A relative to the global block of ten-degree. Calculating the logical offset of point A requires first obtaining the row and column offsets (top_left_offset_x, top_left_offset_y) in the header information of the projection look-up table in which point A is located. Assuming that the blue square in Figure 1 is a global block of ten-degree and the red square is a projection look-up table file, the offset of the upper left corner of the red square is the row and column offset in the header information of the projection look-up table file, using the Formula (1) to calculate the position of point A.

$$\begin{cases} \text{logic\_x} = \text{top\_left\_offset\_x} + i \\ \text{logic\_y} = \text{top\_left\_offset\_y} + j \end{cases}$$

where, logic_x, logic_y is the required row and column logical offset of point A, and there are:

$i <$ top_left_offset_x/BLOCK_SCANS.

$j <$ top_left_offset_y/BLOCK_SAMPLES.

BLOCK SCANS and BLOCK SAMPLES as the default value, i, j overlapping part of the beginning of the block number of ten-degree starting block offset.

(2) Calculate the actual offset value of point A.

After obtaining the logical offset value of point A, it is necessary to further convert it to the actual offset value, easy to read the projection look-up table. The actual offset value is the actual "coordinate" value of point A in the projection look-up table, calculated as formula:

$$\begin{cases} \text{real\_x} = \big((\text{logic\_x} - \text{top\_left\_offset\_x}) + (\text{logic\_y} - \text{top\_left\_offset\_y}) \\ \qquad\qquad * \text{data\_pixels}\big) / (\text{data\_lines} * 2) \\ \text{real\_y} = \Big(\big((\text{logic\_x} - \text{top\_left\_offset\_x}) + (\text{logic\_y} - \text{top\_left\_offset\_y}) \\ \qquad\qquad * \text{data\_pixels}\big)/2\Big) \% \ \text{data\_lines} \end{cases}.$$

Data_lines, data_pixels is the row and column value of the projection look-up table. For the formula for real_x, the expression before division is the logical offset of point A to be converted. Assuming that all data in the projection look-up table file has a number, identify the data in the file belongs to the first few data, the division before the expression is to find the point number, divisible by 2 times the projection table file line value, because the projection look-up table file is a two-tier structure. The value of real_y is the need for the projection look-up table file (data_lines) for modulo operation, where the first two is to calculate the number of tables on a layer of data.

3.2. Positioning of Five-Minute Data

The positioning of data for five minutes depends on the data values in the projection look-up table file. After the data in the projection look-up table file is read and the invalid data is removed, the maximum and minimum values of the lines of the five-minute segment data corresponding to the ten-degree block data are calculated, respectively. The data corresponding to the minimum value of the column ranks is the first position for reading the data of the five-minute segment. Equation (3) is used to calculate the row and column values of the five-minute segment of the portion to be read.

$$\begin{cases} \mathrm{MVI\_x} = \max(\mathrm{lines}) - \min(\mathrm{lines}) \\ \mathrm{MVI\_y} = \max(\mathrm{pixels}) - \min(\mathrm{pixels}) \end{cases}.$$

MVI_x and MVI_y are the row and column values of five minutes of data to be read.

4. Independent Data Fusion Based on Geographic Location

Data fusion adopts the ten-degree block as the basic data processing unit. Each ten-degree block corresponds to several projection look-up table files and five-minute data files, and there are duplicate coverage situations among the multiple sets of data. In this regard:

First, the data mapping relation is loaded, and the mapping table of the corresponding projection look-up table file and the data of five minutes are established for the effective ten-degree block to be processed. And then cyclically read the mapping table in a number of groups of data and processing until the loop read the mapping table. And finally to complete the block ten-degrees block processing. In this process, through the above-mentioned input and output data segmentation technology and data block read and write technology, making the task read only with the current ten-degree block-related input data. So different ten-degree of block production process is independent of each other, and does not affect each other, and is conducive to efficient operation in a cluster environment.

One set of data processing flow is as follows: firstly, according to the positioning data information, the corresponding partial data of the projection look-up table file is loaded, and according to the read result of the projection look-up table file. The data for five minutes are to locate the location of the results of the use of five minutes to read the corresponding part of the data. Then, the data of five minutes is processed, and if there is duplication with the original data, it is determined whether to replace the original data according to the data selection mechanism. If you need to replace, replace the existing data with existing data, otherwise do not replace the original data. Finally, the end of this group of data is processing. The flow chart of independent data fusion based on geographical location is shown in Figure 2.

Wherein for a single projection look-up table file or five-minute segment data, the portion corresponding to the ten-degree block is only a portion of the file data. To avoid reading in redundant data, the processor completes a single ten-degree block processing task by reading the mapping file, selectively loading the

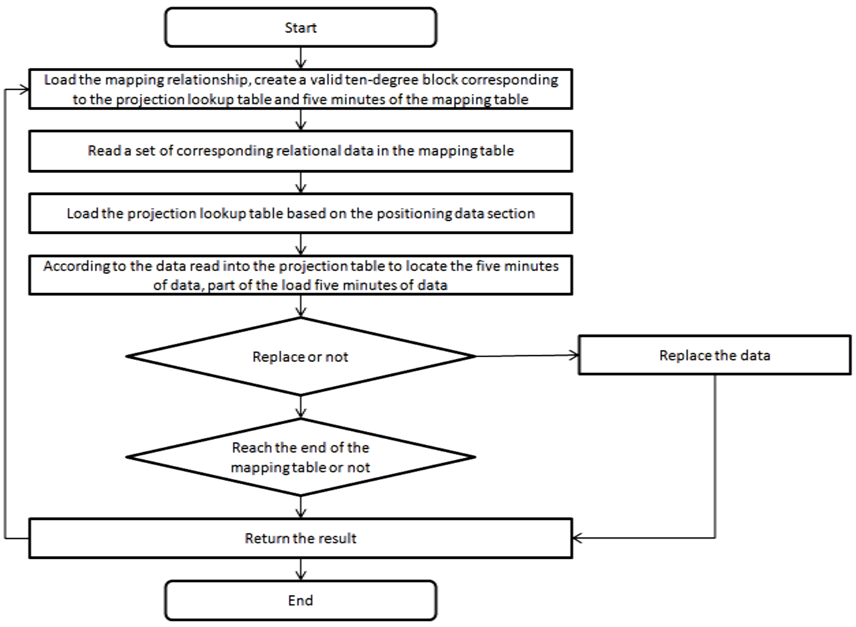
Start
Load the mapping relationship, create a valid ten-degree block corresponding to the projection lookup table and five minutes of the mapping table
Read a set of corresponding relational data in the mapping table
Load the projection lookup table based on the positioning data section
According to the data read into the projection table to locate the five minutes of data, part of the load five minutes of data
Replace or not
Replace the data
Reach the end of the mapping table or not
Return the result
End

Figure 2. Multi-group data fusion flow chart.

corresponding contents of the projection look-up table file and the data of five minutes in a selective manner. In this way, it can effectively reduce the amount of data I/O, but also a corresponding reduction in memory use, making a single multi-core computing nodes parallel processing possible.

5. Experiment Analysis

5.1. Experimental Configuration

The experimental configuration shown in Table 1.

In the experiment, two kinds of data processing methods were designed and compared. The two methods are as follows:

MVID_Load All: in order to block ten blocks for the processing unit, in turn read with the ten block-related data and five minutes of the projection look-up table file, read all the data files. The resulting ten-degree block product data file is finally generated.

MVID_Load Part: in order to block ten blocks for the processing unit, in turn read with the ten block-related data and the five-minute projection look-up table file, read as a way to read only the data file with the ten block-related content. The resulting ten-degree block product data file is finally generated.

5.2. Analysis of Results

Partial loading and full loading refer to MVID_Load Part and MVID_Load All, respectively, when the program runs the data loading strategy. MVID_Load Part part of the program is loaded in accordance with the ten-degree block and the projection look-up table and five minutes of data corresponding to the relationship between the only load projection look-up table and five minutes of the part of the data. The MVID_Load All program is fully loaded to the entire block of

Table 1. Experimental configuration.

Project	Model	Configuration	Quantity
Management node	HP BL460c G6	CPU: 2 * X5670 6 cores 2.93 RAM: 48 GB Hard Disk: 2 * 500 GB 15 k rpm SAS 1 * 2 port InfiniBand HCA	1
Computing node	HP BL460c G6	CPU: 2 * X56706 cores 2.93 RAM: 24 GB Hard Disk: 2 * 500 GB 15 k rpm SAS 1 * 2 port InfiniBand HCA	8
IB switch	HP BLc 4X QDR IB Switch	32 port QDR InfiniBand Switch	1
FC switch	HP Storage Works SN6000c Fibre Channel Switch	48 port FC Switch	1
IBRIX server	HP X9320 FileServer	N/A	2
Disk array	HP StorageWorks P2000 G3 SFF Modular Smart Array	24*HP 72 GB 6 G SAS 15 K 2.5 inch DP ENT HDD	12
IO accelerator	Fusion IO ioDriver Duo	640 GB	2

the projection look-up table and five minutes of data that are all stored in memory.

Performance analysis is mainly from the overall time-consuming, CPU utilization, memory usage and average I/O rate of four indicators to compare.

6. Summary

In this paper, a new data processing method based on RAM (h) was proposed to deal with the low efficiency of data processing in FY-3. First of all, we introduce the Locality-Aware model to segment the input data, then locate the data based on geographic location, and finally fuse the independent data based on geographical location. Based on the proposed algorithm, the experimental verification is carried out. Experimental results show that the proposed method can effectively improve the data processing efficiency. The research of vegetation data processing can provide some references for the data processing of other products of FY-3.

Acknowledgements

The work presented in this study is supported by National High-tech R & D Program (2011AA12A104).

References

[1] Purba, J.H., Lu, J., Zhang, G., *et al.* (2014) A Fuzzy Reliability Assessment of Basic Events of Fault Trees through Qualitative Data Processing. *Fuzzy Sets & Systems*, **243**, 50-69.

[2] Kohl, M., Megger, D.A., Trippler, M., *et al.* (2014) A Practical Data Processing Workflow for Multi-OMICS Projects. *Biochimica Et Biophysica Acta*, **1844**, 52-62.

[3] Shen, L. and Stopher, P.R. (2014) Review of GPS Travel Survey and GPS Data-Pro-

cessing Methods. *Transport Reviews*, **34**, 316-334.
https://doi.org/10.1080/01441647.2014.903530

[4] Vavilov, V.P. and Burleigh, D.D. (2015) Review of Pulsed Thermal NDT: Physical Principles, Theory and Data Processing. *Ndt & E International*, **73**, 28-52.

[5] Jiang, H., Chen, Y., Qiao, Z., *et al.* (2015) Scaling up MapReduce-Based Big Data Processing on Multi-GPU Systems. *Cluster Computing*, **18**, 369-383.
https://doi.org/10.1007/s10586-014-0400-1