Journal of Computer Science 3 (3): 130-133, 2007
ISSN 1549-3636
© 2007 Science Publications

Adapting the LMF Temporal Splining Procedure From Serial to MPI/Linux Clusters

[1]Sajia Akhter, [1]Ipshita Sarkar, [1]Kazi Golam Rabbany, [1]Nahid Akter
[1]Shamim Akhter, [2]Yann Chemin and [2]Honda Kiyoshi
[1]Department of Computer Science, American International University-Bangladesh
Banani, Dhaka, Bangladesh
[2]RS&GIS Field of Study, School of Engineering and Technology (SET)
Asian Institute of Technology (AIT), Thailand

Abstract: Remote Sensing (RS) provides images over large areas such as provincial or country level. During the last 20 years, it plays a vital role for developing many complex applications. However, RS image includes data noises due to influence of haze or cloud especially in the rainy season. It is thus necessary to remove the noise to recover the real ground variations of information studied. Local Maximum Fitting (LMF) is a combined procedure, which helps to remove the noisy data. When dealing with sufficiently large and such complex processing with RS data, single computers time processing extends to unacceptable limits. Such as, to remove the noise from RS image with 146 bands, 38 rows and 37 columns which mean 146 x 38 x 37= 205276 pixels, the LMF procedure requires 26 minutes approximately. So, 1000 x 1000 Remote Sensing Image with 146 bands is required approximately two weeks. It is necessary to reduce the time constraint and make the LMF process executable in a suitable time limit. This study deals with the design and implementation of a distributed LMF procedure. Especially, inside the LMF procedure, a consecutive amount of pixels (pixels in a column) is processed for each row. This behavior is used in this study to make the LMF parallel. For processing the RS image (146 bands, 38 rows and 37 columns), the execution time reduces to 16.13 minutes by adding distributed computing to the program (37 columns distributed to 3 computers).

Key words: LMF, MPI, remote sensing, distributed computing, cluster computers

INTRODUCTION

RS image is always useful for monitoring, prediction and management. One problem with RS image is distortion of data due to noise causing the clouds, hazes etc. It results lack of data due to contamination by clouds (slightly low NDVI: Normalized Difference Vegetative Index). The Local Maximum Fitting (LMF) algorithm that was developed by ACT-JST[1,2] is a combination of the time series filtering and the functional fitting technique, which removes the effect of clouds, hazes and other atmospheric effects from time series data of each pixel and extracts the seasonal change pattern of the ground. Under LMF procedure, a consecutive amount of pixels (pixels in a column) is processed for each row. This behavior seems interesting to make the LMF parallel.

Cluster and Grid computing are two most commonly used parallel computing, which provide increased computing capabilities. A cluster is a type of parallel and distributed processing system, which consists of a collection of interconnected stand-alone computers working together as a single, integrated computing resource. Under the assumption that cluster style computing will remove computational time constraints for LMF process, a parallel LMF procedure for remote sensing images can be considered. The Kasasert University cluster computer (http://magi.cpe.ku.ac.th) is used for this experiment.

The three basic objectives are covered in this study. (i) Propose an overall implementation scheme for Distributed LMF procedure. (ii) Analyze the accuracy of the RS image processed with Distributed LMF procedure. (iii) Analyze the execution time of serial and Distributed LMF procedure.

LMF architecture: LMF is the method to obtain the maximum value A of four points before continuous time-series data and the maximum value B of four points after continuous time-series data in order to give the minimum values of both A and B[3,4] (Fig. 1).

Corresponding Author: Dr. Honda Kiyoshi, RS&GIS Field of Study, School of Engineering and Technology (SET) Asian Institute of Technology (AIT), Thailand

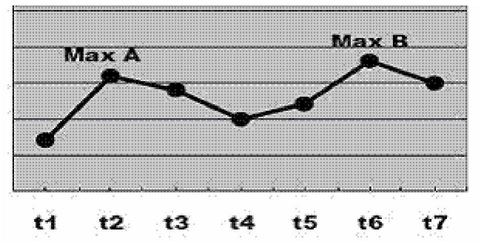


Fig. 1: Local maximum fitting

Maximum A = MAX (t1, t2, t3, t4), Maximum B = MAX (t4, t5, t6, t7), t4 = MIN (Maximum A, Maximum B), t1 - t7 = NDVI at each time point, t4'= Value of t4 after filtering. So, the basic equation for filtering process is Local Maximum Filter:

$$d'_t = \text{Min}\begin{bmatrix}\text{Max}\left(d_{t-w+1}, d_{t-w+2}, \wedge, d_t\right), \\ \text{Max}\left(d_t, d_{t+1}, \wedge, d_{t+w+1}\right)\end{bmatrix} \qquad (1)$$

d_t : Observed data at time t , W : filter window ,

d'_t : Modified data at time t .

Now, the Fitting Model is:

$$f_t = c_0 + c_1 t + \sum_{t=1}^{N} c_{2t} \sin\left(\frac{2\prod ki\, t}{M}\right) + c_{2t+1} \cos\left(\frac{2\prod ki\, t}{M}\right) (2)$$

ci: Co– efficient(s), t: time, N: Number of time series data, M: Number of data for 1 cycle, e.g. M=36, as 36 image/year, Ki: As a periodic function, by assuming that six periods (1 year, half year, 4 months, 3 months, 2 months, 1 month) might be used e.g. {1, 2, 3, 4, 6, 12}, C_{1t}: trend. These six periodic functions are adopted at the initial step. Equation (2) is converted to a sine curve function.

$$f_1 = c_D + c_1 t + \sum_{i=1}^{n}\left\{A_i \sin\left(\frac{2\prod k_i t}{M} + x_i\right)\right\} \qquad (3)$$

Where, A_i is amplitude and x_i is phase lag of sine curve. In this study, we use these A_i and x_i parameters calculated from initial step of LMF processing. In LMF processing, to remove the effects of clouds, hazes and system noises, the time-series filtering and the fitting processing are repeated until the optimum result functions[1] are obtained.

Serial LMF procedure: The whole serial LMF procedure is divided into three parts. (i) Pre LMF, (ii) LMF procedure, (iii) Post LMF.

Remote Sensing Image is a combination of different bands (date wise sorted), which is considered as a 3-D matrix. 46 different bands are available which mean one single pixel containing 46 different values. As GIS and RS software works with only a single row of a RS image at a time, so each row of the 3-D matrix

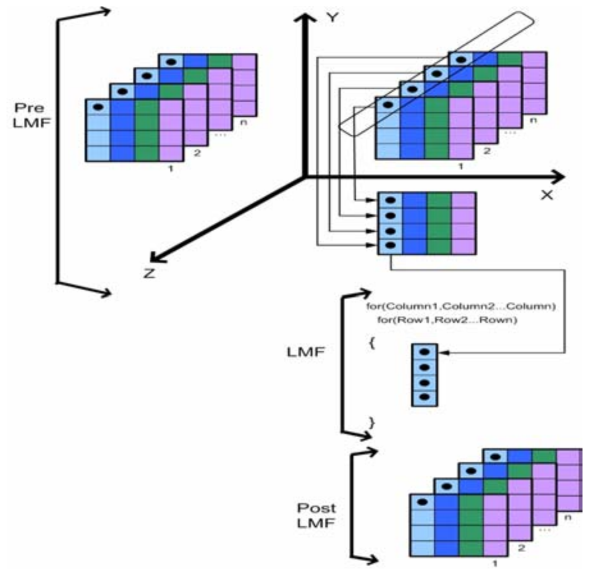


Fig. 2: Serial LMF procedure

is converted into a 2-D matrix. Each column of that 2-D matrix contains different band values for each pixel. In Fig. 2, N Column= X-axis, N Lines= Y-axis, N Bands= Z-axis. This whole process is called pre-LMF. Every time each column of that 2-D matrix, executes the LMF method.

```
For (Col 1, Col2….Col N)
    For (Row1, Row2…Row N)
        LMF ( , , ,)
```

The calculated and processed pixel values are again written to the same column of the 2-D matrix and then placed in the 3-D matrix. This process after executing LMF is called Post-LMF.

Proposed architecture for distributed LMF procedure: To make the whole LMF procedure parallel the pre-LMF process is remain same. Only, LMF procedure is broken down for parallelism. LMF process will work parallel by executing more then one column of that 2-D matrix at the same time. To serve this procedure a cluster implementation methodology is required.

In the cluster computing technology, there are a master and several slaves. Where master will do the Pre LMF process then transfer the pixels among the available slaves. Then each slave will execute the LMF process for its pre-defined columns. Thus several columns are parallally executed at the same time by their corresponding salves. Thereafter the processed pixels will be sent to the Master again. Master will do the Post-LMF. Thus the whole RS image without noisy data will be provided.

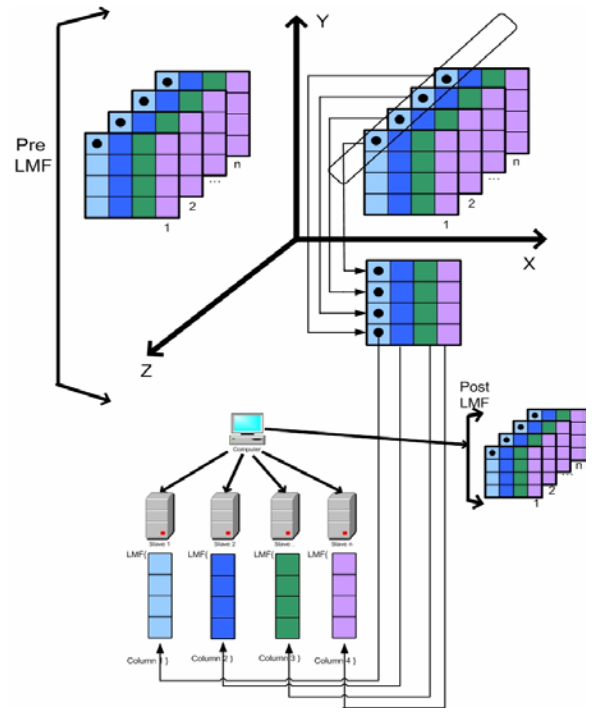
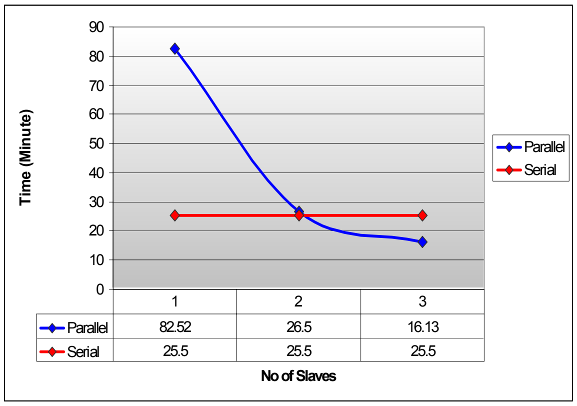


Fig. 3: Distributed LMF procedure

Implementation and result analysis: FORTRAN 77 is used to implement parallel LMF. This computation consists of four processors, each working on local data. MPI, subroutines in FORTRAN is used to communicate between processors. The master processor is responsible for pre LMF and post LMF. For pre LMF, master splits the RS image. It reads the image from data.bsq. Then for every row i, a 2-D matrix KPIX [No of Bands][No of columns] is created. KPIX (k,j) denotes the pixel value of kth image. The pseudo code is given bellow for pre LMF.

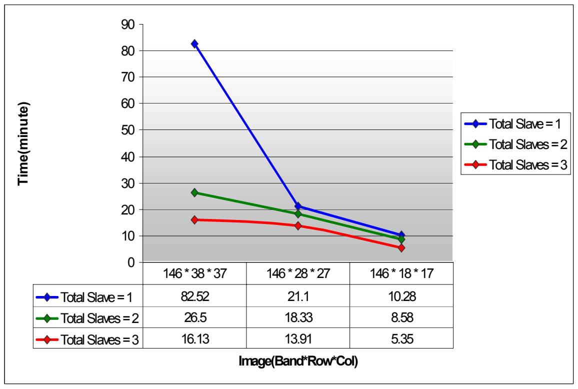
```
Do i ← 1 to No_Of_Rows
   CALL GTBQ2F (i, No_Of_Rows, No_Of_Columns, starting_index_of_band, No_Of_Bands, KPIX)
   Do j ← 1 to No_Of_Columns
      Do k ← 1 to No_Of_Bands
         KPIXP (k) ← KPIX (k, j)
```

Here, GTBQ2F (, ,) procedure is responsible for calculating KPIX[][] for each row. After that for each column j, master generates a 1 dimensional matrix KPIXP [No of Bands] and sends it to each slave for calculating LMF procedure. Let assume, there are 10 columns and 3 salves are available. Slave 1 will get column 1, 4, 7, 10; slave 2 will get column 2, 5, 8; slave 3 will get column 3, 6, 9. Master also sends other necessary data with KPIXP [] array. Each slave executes LMF procedure for their defined column and



	1	2	3
Parallel	82.52	26.5	16.13
Serial	25.5	25.5	25.5

Fig. 4: Execution time for serial and parallel



	146 * 38 * 37	146 * 28 * 27	146 * 18 * 17
Total Slave = 1	82.52	21.1	10.28
Total Slaves = 2	26.5	18.33	8.58
Total Slaves = 3	16.13	13.91	5.35

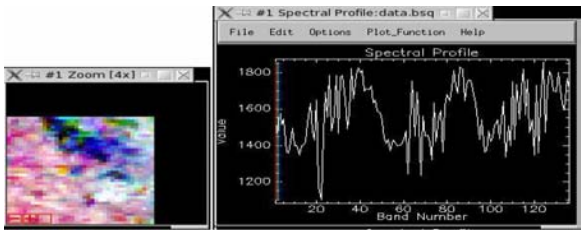
Fig. 5: Image vs. execution time

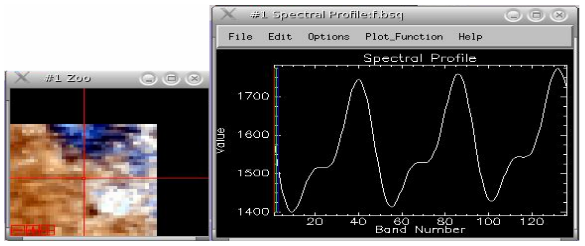
returns the pixel value to the master after removing distortion. Then Master has to regenerate the RS image using post LMF procedure. For each row and column it gets the band value from slaves and saves it to PIX [No of row][No of band][No of column] array.

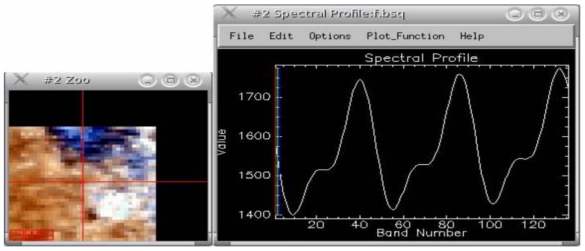
```
Do i ←1 to No_Of_Rows
   Do j←1 to No_Of_Columns
      DO k ← 1, NBANDS
         PIX (i, k, j) ← KPIXP (k)
```

The output image without distortion is in PIX [][][] array. This array is saved to f.bsq.

Figure 4 represents the execution time comparison between the Serial and Parallel LMF. In parallel LMF procedure, it seems that to increase the slave numbers will reduce the execution time. However, when there will be only one slave, the serial LMF execution time is smaller then the parallel LMF execution time. This was happened because of the communication and data-transferring overhead (82.52-25.5 =57.02 minutes) between slave and master (slave only runs the LMF process and master executes the PRE LMF and POST LMF procedure). Due to the inversely proportional behavior between the processor number and the


#1 Zoom [4x]
#1 Spectral Profile:data.bsq
File Edit Options Plot_Function Help
Spectral Profile
Value
Band Number
Fig. 6: Input image


#1 Zoo
#1 Spectral Profile:f.bsq
File Edit Options Plot_Function Help
Spectral Profile
Value
Band Number
Fig. 7: Output image for serial implementation


#2 Zoo
#2 Spectral Profile:f.bsq
File Edit Options Plot_Function Help
Spectral Profile
Value
Band Number
Fig. 8: Output image for parallel implementation

execution time, the communication time will decrease rapidly for successive increment of the number of processors. Additionally, the parallel LMF execution time also depends on the image size (Fig. 5). For smaller image size, the execution time remains same with the increased processor numbers. However, for larger image size, the parallel execution time is decreased when processor number is increased. It concludes that parallel LMF will work better, when the execution time in each slave will be higher then the communication time.

In the present experiment, the following (left portion of Fig. 6) Remote Sensing Image (data.bsq) is used as input for both serial and parallel running code. The right portion of Fig. 6 shows the spectral resolution profile for the input distorted image.

The Fig. 7 and 8 show the output images generated by serial and parallel LMF procedure. In this figure the spectral resolution profile for both images (right portion of Fig. 7 and right portion of Fig. 8) are analyzed, which seems that the values are same. It concludes that the present parallel LMF program running successfully with almost 100% accuracy.

CONCLUSION AND RECOMENDATIONS

Serial LMF is successfully re-implemented parallel in cluster computers and provides a time optimization method for agriculture monitoring with Remote Sensing data. Parallel LMF takes approximately 26 minutes to process the data with a file size of 205276 pixels and it gives 100% accurate result. Increased processor number will provide a better timing. However, processor number can be increased as long as every time the execution time of a single salve is greater than the communication time of that slave.

In FORTRAN 77, in case of real value transferring, it can transfer correctly only 60% of actual data. To overcome this problem, every time double size of data is to be transferred and working out with only 50% data, which increases the communication time. However, converting the whole procedure from FORTRAN 77 to C++, the real value transfer problem can be removed.

REFERENCES

1. Sawada, H., Y. Sawada, I. Nagatani and M. Anazawa, 2001. Proceeding for the 1st regional seminar on geo-informatics for Asian eco-system management.
2. Sawada, H. and Y. Sawada, 2002. Modeling of vegetation seasonal change based on high frequency observation satellite. Environmental Information Science Papers. Vol. 16.
3. Andres, L., W.A. Salas and D. Skole, 1994. Fourier analysis of multi -temporal AVHRR data applied to a land cover classification. Intl. J. Remote Sensing, 15: 1115-1121.
4. Azzali, S. and M. Menenti, 2000. Mapping vegetation-soil-climate complexes in southern Africa using temporal Fourier analysis of NOAA-AVHRR NDVI data. Intl. J. Remote Sensing, 21: 973-996.