Efficient Information Extraction from Hyperspectral Imagery Using Networks of Workstations

Antonio Plaza, Javier Plaza, David Valencia and Pablo Martínez
Neural Networks and Signal Processing Group
Department of Computer Science, University of Extremadura
Avda. de la Universidad s/n, E-10071, Cáceres, Spain
aplaza@unex.es

Abstract— **The rapid development in space and computer technologies has made possible to store a large amount of remotely sensed image data, collected from heterogeneous sources. In particular, NASA is continuously gathering imagery data with hyperspectral sensors such as the Airborne Visible-Infrared Imaging Spectrometer (AVIRIS) or the Hyperion imager aboard Earth Observing-1 (EO-1) spacecraft. The development of efficient techniques for transforming the massive amount of collected data into scientific understanding is critical for space-based Earth science and planetary exploration. Heterogeneous networks of workstations are a very promising cost-effective parallel computing architecture. Unlike traditional homogeneous parallel platforms, heterogeneous architectures are composed of processors running at different speeds. This heterogeneity results in distributed-memory parallel computing systems created from commodity components that can satisfy specific computational requirements for the Earth and space sciences community. This paper explores techniques for mapping hyperspectral image analysis algorithms onto heterogeneous networks of workstations. Important aspects in algorithm design such as portability, reusability and scalability are illustrated by using homogeneous and heterogeneous parallel computing facilities at NASA's Goddard Space Flight Center and, European Center for Parallelism of Barcelona, and University of Extremadura in Spain. Hyperspectral image data from the AVIRIS data repository is used in experiments, which reveal that heterogeneous networks of workstations are a source of computational power that is both accessible and applicable to obtaining results quickly enough for practical use in information extraction applications from hyperspectral imagery.**

Keywords- Hyperspectral imaging, Parallel algorithms, Heterogeneous computing, Spatial/spectral analysis.

I. INTRODUCTION

The rapid development of space and computer technologies has made possible to store a large amount of image data, collected from heterogeneous sources. In particular, NASA is continuously gathering imagery data with earth observing sensors. The automation of techniques for transforming collected data into scientific understanding is critical for space-based earth science and planetary exploration with onboard scientific data analysis. Advances in sensor technology have led to the development of hyperspectral instruments capable of collecting hundreds of images, corresponding to different wavelength channels, for the same area on the surface of the Earth. A chief hyperspectral sensor is the NASA's Jet Propulsion Laboratory Airborne Visible-Infrared Imaging Spectrometer (AVIRIS) system, which currently covers the wavelength region from 0.4 to 2.5 μm using 224 spectral channels, at a nominal spectral resolution of 10 nm. On other hand, the Hyperion hyperspectral imager aboard NASA's Earth Observing-1 (EO-1) spacecraft has been NASA's first hyperspectral imager to become operational on-orbit. It routinely collects images hundreds of kilometers long with 220 spectral bands from 0.4 to 2.5 μm. In the near future, the use of hyperspectral sensors on satellite platforms will produce a nearly continual stream of multidimensional data, and this expected high data volume would demand fast and efficient means for storage, transmission and analysis.

While developments in hyperspectral imaging hold great promise for Earth science image analysis, they create new processing challenges. In particular, the price paid for the wealth spatial and spectral information available from hyperspectral sensors is the enormous amounts of data that they generate. As a result, analysis techniques are often computationally tedious, and require lengthy durations to calculate desired quantities. Several applications exist for hyperspectral data where the desired information must be calculated in real-time or near real-time. That is the case of military applications, where a commonly pursued goal is the detection of either full pixel or subpixel targets, often associated to hostile weaponry, camouflage, concealment, and/or decoys. Another example is the detection and tracking of natural disasters such as forest fires, oil spills, and other types of chemical contamination, where timely classification is highly desirable.

Parallel processing can help to tackle large remotely sensed data sets and to get reasonable response times in complex analysis scenarios [1]. Today's RISC microprocessor systems and personal computers (PCs) have computational speeds above one gigaflop (billions of floating point operations per second). Heterogeneous networks of based on these architectures are a very promising distributed-memory parallel paradigm [2], which offers offers the possibility of performance in hundreds of gigaflops at low cost, and memory capacity sufficient for detailed hyperspectral studies. Unlike traditional homogeneous parallel platforms, heterogeneous architectures are composed of processors running at different speeds. Often, this heterogeneity is not even planned, but arises simply due to the march of technology over time and computer market sales and trends.

0-7803-9050-4/05/$20.00 ©2005 IEEE.

In this paper, we explore techniques for mapping hyperspectral image analysis algorithms onto heterogeneous networks of workstations. The paper is structured as follows. Section II briefly describes a sequential morphological processing algorithm that will serve as our case study throughout the paper. Section III develops a parallel version specifically designed for heterogeneous platforms. In Section IV, we assess the parallel performance of the algorithm by drawing comparisons between its efficiency on a heterogeneous cluster of workstations with the efficiency achieved by a homogeneous version of the same algorithm on Thunderhead, a (homogeneous) massively parallel computer at NASA's Goddard Space Flight Center. Performance data on an SGI Origin 2000 computer are also given for comparison. Finally, Section 5 concludes with some remarks.

II. Sequential Morphological Processing Algorithm

This section briefly develops a morphological processing algorithm for analysis and classification of hyperspectral image data. The algorithm will be used as a case study throughout the paper, as a representative algorithm of integrated spatial/spectral approaches, i.e., algorithms that take into account both the spatial and spectral information of the data in simultaneous fashion. This algorithm has been thoroughly described before [3], and we will not expand on its detailed implementation here. Instead, we provide relevant information on how to extend morphological operations to hyperspectral image data, an issue that indeed represents the most important computational requirement of the algorithm.

Let us denote by $\boldsymbol{f}$ a hyperspectral defined on an N-D space, where N is the number of channels or bands. The main goal of extended morphological operations is to impose an ordering relation in terms of spectral purity in the set of pixel vectors lying within a spatial search window (structuring element), designed by B, defined in advance. In order to do so, we define a cumulative distance between one particular pixel $\boldsymbol{f}(x,y)$, where $\boldsymbol{f}(x,y)$ denotes an N-D vector at discrete spatial coordinates $(x,y) \in Z^2$, and all the pixel vectors in the spatial neighborhood given by B (B-neighborhood) as:

$$D_B[\boldsymbol{f}(x,y)] = \sum_i \sum_j \mathrm{SAM}[\boldsymbol{f}(x,y), \boldsymbol{f}(i,j)] \quad (1)$$

where (i,j) refers to spatial coordinates in the B-neighborhood and SAM is the spectral angle mapper, defined as follows:

$$\mathrm{SAM}(\boldsymbol{f}(x,y), \boldsymbol{f}(i,j)) = \cos^{-1}\left(\boldsymbol{f}(x,y) \cdot \boldsymbol{f}(i,j) / \|\boldsymbol{f}(x,y)\| \|\boldsymbol{f}(i,j)\|\right) \quad (2)$$

As a result, $D_B[\boldsymbol{f}(x,y)]$ is given by the sum of SAM scores between $\boldsymbol{f}(x,y)$ and every other pixel vector in the B-neighborhood. Based on the distance above, the extended erosion of $\boldsymbol{f}$ by B selects the B-neighborhood pixel vector that produces the minimum value for D_B:

$$(\boldsymbol{f} \Theta B)(x,y) = \left\{\boldsymbol{f}(x+i', y+j'), (i',j') = \arg\min_{(i,j)} \{D_B[\boldsymbol{f}(x+i, y+j)]\}\right\} \quad (3)$$

where the argmin operator selects the pixel vector is most highly similar, spectrally, to all the other pixels in the B-neighborhood. On other hand, the extended dilation of $\boldsymbol{f}$ by B selects the B-neighborhood pixel vector that produces the maximum value for D_B:

$$(\boldsymbol{f} \oplus B)(x,y) = \left\{\boldsymbol{f}(x-i', y-j'), (i',j') = \arg\max_{(i,j)} \{D_B[\boldsymbol{f}(x+i, y+j)]\}\right\} \quad (4)$$

where the argmax operator selects the pixel vector that is most spectrally distinct to all the other pixels in the B-neighborhood. The proposed morphological algorithm is based on the calculation of a morphological eccentricity index (MEI) at each pixel location in the scene as follows [3]:

$$\mathrm{MEI}(x,y) = \mathrm{SAM}[(\boldsymbol{f} \oplus B)(x,y), (\boldsymbol{f} \Theta B)(x,y)] \quad (5)$$

The resulting scores can be then used for a variety of applications in hyperspectral imaging, ranging from pure pixel (endmember) identification to mixed pixel characterization and classification. In the following section, we provide a parallel approach to compute MEI scores at a pixel level.

III. Parallel Implementation

This section describes a heterogeneous parallel processing framework for hyperspectral image analysis, which makes use of the algorithm outlined in section II as a particular case study. Before providing an overview of the proposed parallel algorithm, an important issue in algorithm design such as data partitioning is first discussed.

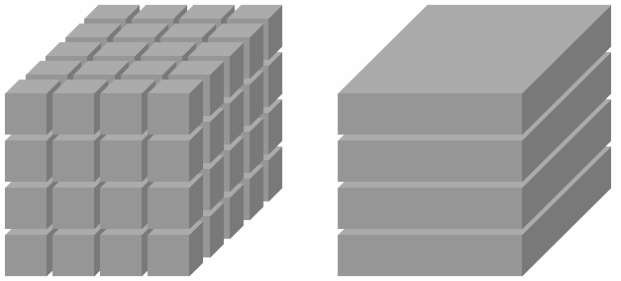

Figure 1. Spectral-domain (left) and spatial-domain data partitioning (right).

A. Data Partitioning

A major requirement for efficient parallel algorithms on distributed memory systems is finding a decomposition that minimizes the communication between the processors. Domain decomposition techniques provide the greatest flexibility and scalability in parallel image processing. Two types of partitioning can be exploited in multi/hyperspectral image analysis algorithms: spectral-domain partitioning and spatial-domain partitioning. Spectral-domain partitioning subdivides the volume into small cells or sub-volumes made up of contiguous spectral bands, and assigns one or more sub-volumes to each processor [see Fig. 1(left)]. With this model, each pixel vector is split amongst several processors and the communication cost for the proposed processing algorithm is enormous, thus preventing scalable implementations. In order

to achieve load balance and to exploit parallelism as much as possible, a spatial-domain partitioning approach was adopted in our parallel framework [4]. The volume is divided into slabs as depicted in Fig. 1(right). It should be noted that, in spectral-domain partitioning, the calculations made for each pixel vector need to originate from several processors, and thus require intensive inter-processor communication.

B. Implementation details

Before describing our parallel algorithm, we should point out that an important issue in neighborhood-based image processing applications such as convolution or mathematical morphology is that additional inter-processor communications are required when the structuring element computation needs to be split amongst several different processing nodes. However, if redundant information such as a scratch line is added to one of the adjacent partitions to avoid accesses outside image domain, as illustrated in Fig. 2, then boundary data no longer need to be exchanged between neighboring processors. It is clear at this point that the scratch line would introduce redundant computations, since the intersection between the two involved partitions would be non-empty. However, we have experimentally tested that redundant computation-based solutions are more effective that data exchange-based solutions in hyperspectral imaging [4], mainly due to the large volume of data that need to be exchanged between processors in the latter.

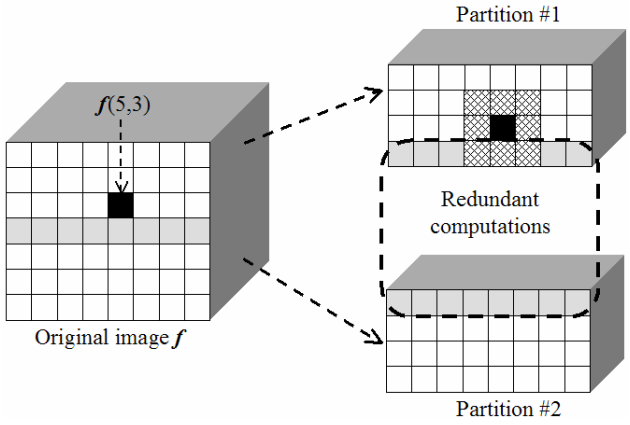

Figure 2. Redundant computations to reduce inter-processor communication.

Below, a redundant computation-based parallel algorithm designed to be run in heterogeneous computing platforms is provided. The main purpose of the algorithm is to provide a mechanism to slice the available data into chunks, so that the total execution time is minimized. For this purpose, there is a need to load-balance the workloads of p participating heterogeneous resources so that each processor P_i will accomplish a share α_i of the total workload W, where $\alpha_i \geq 0$ for $1 \leq i \leq p$ and $\sum_{i=1}^{p} \alpha_i = 1$. Therefore, a desired goal is to find a set of optimal values for the set $\{\alpha_i\}_{i=1}^{p}$. Below, we provide a step-by-step description of the proposed algorithm, where the input parameters are a hyperspectral image, $\boldsymbol{f}$, and a structuring element, B, that will be used for the construction of morphological operations.

1. Obtain necessary information about the parallel system, including the number of available processors, p, each processor's identification number, $\{P_i\}_{i=1}^{p}$, and processor cycle-times, $\{w_i\}_{i=1}^{p}$.
2. Using B and the information obtained in step 1, determine the total volume of information, R, that needs to be replicated from the original data volume, V, in order to avoid inter-processor communication. It should be noted that the total workload W to be handled by the algorithm is given by $W = V + R$.
3. Set $\alpha_i = \left\lfloor \frac{(p/w_i)}{\sum_{i=1}^{p}(1/w_i)} \right\rfloor$ for all $i \in \{1, \cdots, p\}$.
4. For $m = \sum_{i=1}^{p} \alpha_i$ to $(V+R)$ do begin:

   4.1. Find $k \in \{1, \cdots, p\}$, $w_k \cdot (\alpha_k + 1) = \min\{w_i \cdot (\alpha_i + 1)\}_{i=1}^{p}$.

   4.2. Set $\alpha_k = \alpha_k + 1$.
5. Use the resulting $\{\alpha_i\}_{i=1}^{p}$ to obtain a set of p spatial-domain heterogeneous partitions of $(V+R)$, and send its corresponding partition to each processor P_i along with B. The sequential algorithm in section II is executed in parallel at each processor.
6. Collect all the individual results $\{\text{MEI}_i\}_{i=1}^{p}$ provided by each processor P_i, and merge them together to form a final image $\text{MEI} = \bigcup_{i=1}^{p} \{\text{MEI}_i\}$.

It should be noted that a homogeneous version of the algorithm above can be obtained by replacing step 3, so that $\alpha_i = p/w_i$ for all $i \in \{1, \cdots, p\}$, where w_i is a constant communication speed between each processor pair. Performance data for the parallel algorithm are given in the following section.

IV. Experimental Results

This section provides an assessment of the effectiveness of the proposed parallel algorithm. The section is organized as follows. First, we provide an overview of the parallel computing architectures used for evaluation purposes. Second, performance data are given.

A. Parallel Computing Architectures

We have experimented with three clusters of workstations. The first one is a small-scale heterogeneous network of 16 different SGI, Solaris and Linux workstations, and four communication segments. The communication network of the heterogeneous platform consists of four relatively fast homogeneous communication segments interconnected by three slower communication links. Although this is a simple architecture, it is also a quite typical and realistic one as well.

The second parallel computing architecture used in experiments is a distributed-memory homogeneous system (SGI Origin 2000) located at European Center for Parallelism of Barcelona (CEPBA). The system used in experiments is composed of 64 MIPS R10000 processors at 250 MHz (each one with 4 MB of cache memory) and 12 GB of main memory, interconnected via 1.2 Gbps communication network. Finally, in order to test the algorithm on a larger-scale parallel platform, we have also implemented the algorithm on Thunderhead, a Beowulf cluster located at NASA's Goddard Space Flight Center. It is composed of 256 dual Linux workstations at 2.4 GHz, interconnected through 1.2 Gbps Myrinet network.

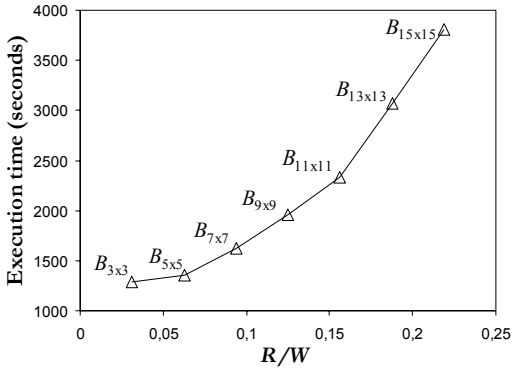

Figure 3. Execution times in a 16-processor fully heterogeneous network.

B. Performance Analysis

The parallel algorithm in section III-B was applied to a 145x145-pixel hyperspectral scene collected by AVIRIS, using seven different square-shaped structuring elements, i.e., B_{3x3}, B_{5x5}, B_{7x7}, B_{9x9}, B_{11x11}, B_{13x13} and B_{15x15}. The data set was acquired over the well-known Indian Pines region, a mixed forest/agricultural test site. The data set represents a very challenging classification problem due to the presence of mixed pixels. Extensive ground-truth information is available for the area, which is available online, along with ground-truth information (http://dynamo.ecn.purdue.edu).

For illustrative purposes, Fig. 3 plots the execution times in seconds of the parallel algorithm on the heterogeneous cluster as a function of the ratio R/W, where R is the amount of redundant information introduced by the structuring element, and W is the total workload. It should be taken into account that the most appropriate structuring element in terms of computational cost-performance ratio was B_{11x11}, which resulted in 90.13% accuracy obtained by a classification algorithm based on the proposed morphological processing [3]. In order to explore the scalability and portability of the algorithm to massively parallel computing platforms (mainly homogeneous in nature), Fig. 4 shows the speedups achieved by the parallel algorithm and its homogeneous version over a single-processor run of the sequential algorithm in section II on Thunderhead, as a function of the number of processors. B_{11x11} was the considered structuring element size. Fig. 4 reveals that both the homogeneous and heterogeneous algorithm achieved very similar performance. Also, scalability results obtained in the Origin system were very similar to those addressed in Fig. 4, but the number of processors was much smaller (only 16 processors were available to us at the time of measurements). Regarding near real-time requirements, experiments revealed that processing times obtained in the Origin system generally prevented near real-time exploitation of the data, mainly due to the limited number of processors. However, the utilization of 36 processors in the Thunderhead system allowed the proposed parallel algorithm to accomplish very high (above 90%) classification scores in about 60 seconds. The same scores could be obtained in less than 10 seconds when 64 processors were used. Although the measured processing times seem to indicate that near real-time data exploitation is possible, further experimentation is required in order to extrapolate the above results to additional hyperspectral data sets.

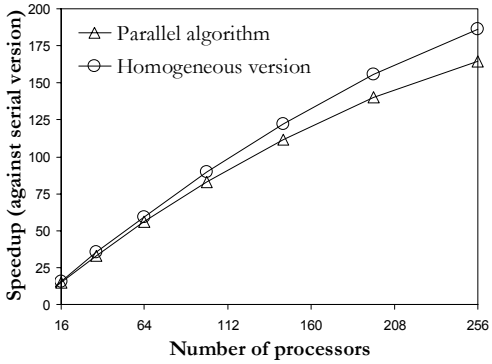

Figure 4. Measured speedups in a 256-processor massively parallel network.

V. CONCLUSIONS

This paper provided an investigation of parallel techniques to extract relevant information from hyperspectral image data sets. For this purpose, networks of computers are a cost-effective way of exploiting this sort of parallelism in remote sensing applications. It has been shown that parallel computing at the massively parallelism level, supported by message passing, provides a unique framework to extract information in near real-time and with adequate reliability in a remote sensing environment. The proposed parallel framework is particularly suitable for data mining applications that previously looked to be too computationally intensive for practical applications due to immense files and data archives common to remote sensing problems. Combining this readily available computational power with the new sensor instruments may introduce major changes in the systems used by NASA and other agencies for exploiting earth and planetary remotely sensed data.

REFERENCES

[1] F. J. Seinstra, D. Koelma and J. M. Geusebroek, "A software architecture for user transparent parallel image processing," *Parallel Computing*, vol. 28, pp. 967-993, 2002.

[2] A. Lastovetsky, *Parallel computing on heterogeneous networks*. Wiley-Interscience: Hoboken, NJ, 2003.

[3] A. Plaza, P. Martinez, R. Perez, J. Plaza, "A new approach to mixed pixel classification of hyperspectral imagery based on extended morphological profiles," *Pattern Recognition*, vol. 37, pp. 1097-1116, 2004.

[4] D. Valencia, A. Plaza and P. Martínez, "On the use of cluster computing architectures for implementation of hyperspectral algorithms," *Proc. IEEE Symp. Computers and Communications*, Cartagena, Spain, 2005.