

High Performance Computing for Image Classification ¹

Rama Chellappa ²
S. Krishnamachari ²

Larry Davis ³
Jerry Sobieski ⁵
Rahul Parulekar ²

David Harwood ⁵
John Townshend ⁴

Institute for Advanced Computer Studies (UMIACS)
University of Maryland
College Park, MD 20742

Abstract

We present the overall goals of our research program on the application of high performance computing to remote sensing applications, specifically applications in land cover dynamics. This involves developing scalable and portable programs for a variety of image and map data processing applications, eventually integrated with new models for parallel I/O of large scale images and maps. Here, we focus on our research in classification of remotely sensed images, describing new parallel algorithms for classification using Markov Random Fields and for classifying segmentations of images using a multistage decision procedure.

1 Introduction

We describe the results of our research program on the application of high performance computing to the analysis of remotely sensed imagery. The specific application area that we are focusing on, initially, is generating maps of the world's tropical rain forest during the past three decades. This is currently being done using manually labor intensive techniques, and is representative of a wide variety of remote sensing applications requiring nearly global data analysis.

Our research is conducted at three levels:

1. At the **science level** we are developing new models for fundamental science problems such as atmospheric correction, determining ground reflectivity from imagery, mixture modeling, image feature extraction, image pixel and region classification, and spatial data structures.
2. At the **algorithms level** we design and develop portable and scalable parallel implementations of our algorithms and data structures, and test them - first in terms of their scientific merit, and then in terms of their parallel scalability and speed - on large image and map data sets. This paper will

focus on our research at this level, describing our work on image classification using both Markov Random Field models, and more structural approaches that employ multi level decision procedures to classify regions in a hierarchical segmentation of a multispectral image.

3. At the **high performance computing level** we are developing new tools and techniques for supporting applications such as image processing and spatial data handling on parallel machines. Our research here focuses on object oriented parallel programming and on new models and techniques for parallel I/O of large image and map data structures.

2 Scientific background

Understanding land cover dynamics is one of the most important challenges in the study of global change. Many of these changes take place at very fine scales (less than 1 km cell size), and require the analysis of high resolution satellite images for accurate measurement. Databases of land cover dynamics are essential for global carbon models, biogeochemical cycling, hydrological modeling and ecosystem response modeling. These needs have been recognized in a number of international programs such as the International Geosphere Biosphere Program (IGBP), the World Climate Research Program and the International Satellite Land Surface Climatology Project.

Some aspects of global change modeling depend on relatively coarse resolution data sets as in the case of General Circulation Models, where resolutions as coarse as 250 km cell size are typical. Even in these situations, reliable parameterizations of sub-grid scale variability require detailed knowledge of what is happening at very fine scales. The need for the analysis of fine resolution data sets arises from the low spatial autocorrelation of land surface properties. Even in areas with near natural land cover, much of the spatial variability can exist at the finest scales, largely because of the influence of local terrain variability on vegetation response for a given set of climate characteristics. Human-induced changes are the source of many of the finest scale alterations, and are often the most signifi-

¹This work was supported in part by the National Science Foundation under Grant #ASC 9318183.

²UMIACS and Department of Electrical Engineering

³UMIACS and Department of Computer Science

⁴Department of Geography

⁵UMIACS

cant drivers of global changes. Currently we have very poor data on where these changes are occurring and what are the resultant land cover properties of the altered land.

As an example of the need to consider such local changes, the LANDSAT Pathfinder project is generating global detailed databases of the whole of the Earth's tropical moist forest to reduce major uncertainties in the global carbon budget. Without the use of fine resolution LANDSAT data, it has been demonstrated conclusively that data on deforestation cannot be derived with sufficient accuracy. This task is being carried out for only roughly 10% of the Earth's surface and is largely conducted through visual analysis of the data. Currently we do not have the algorithms and computational resources available that might allow us to match human interpreters in terms of accuracy.

Our ultimate goal is the automated generation of land cover change for the globe on an annual basis at a resolution of 30 meters. or better, using advanced algorithms for extraction of information from multi-sensor, multi-spectral, multi-temporal, multi-resolution data sets. The need for this integrated approach arises from the need to use different data sets sensitive to different properties. Moreover, they will have to undergo substantial radiometric and geometric pre-processing to prepare them for data extraction using advanced algorithms.

3 Markov Random Field Model Based Classification

In this section we present a multiresolution Gauss Markov random field (**GMRF**) scheme applied to classification. GMRF models have been extensively used to model, synthesize and classify textures [3]. MRF models provide a convenient tool to characterize prior beliefs about various image features and to combine prudently the apriori knowledge and information contained in the observed data. One of the drawbacks of MRFs is that the associated optimization schemes are iterative and computationally expensive. On the other hand, all the computations involved are local and MRF algorithms are inherently amenable to parallel implementation. Multiresolution schemes are often used to reduce the computational burden [5], [6], [4]. We present a multiresolution GMRF model based on *local conditional distribution invariance* approximation. The scheme involves the following steps:

1. Different classes in the image are modeled by GMRF processes. The parameters of the GMRF processes for all classes are obtained from a small training area using a maximum likelihood estimator. The training area used for each class is a rectangle of dimensions about 30x30 which represents less than 0.3% of the total image size.
2. The image at fine resolution is subsampled to obtain lower resolution data. The parameters of the GMRF processes characterizing the classes at lower resolutions are obtained by the local conditional distribution invariance method presented in this section.

3. Classification is performed first at the coarsest resolution while usually the convergence at the coarser resolutions is much faster. The classification result obtained at a coarse resolution is propagated to the next immediate higher resolution and so on, until the image at fine resolution is classified.

Let $\Omega^{(0)} = \{(i, j) : 0 \leq i \leq M - 1, 0 \leq j \leq M - 1\}$ be a lattice. Let $X^{(0)}$ represent a random vector, obtained by ordering the random variables on the two-dimensional lattice $\Omega^{(0)}$, through a row-wise scan. The elements of $\Omega^{(0)}$ are indexed by s , where $s = (s_1, s_2)$. Let $X^{(0)}$ be modeled by a GMRF; then the joint probability density function of $X^{(0)}$ can be written as follows:

$$P^{(0)}(X^{(0)} = x) = \frac{\exp\{-\frac{1}{2}x^T[\Sigma^{(0)}]^{-1}x\}}{(2\pi)^{\frac{M^2}{2}}(\det\Sigma^{(0)})^{\frac{1}{2}}} \quad (1)$$

where $\Sigma^{(0)}$ is the covariance matrix of $X^{(0)}$. In another equivalent representation, the process $X^{(0)}$ can be written in terms of a non-causal interpolative representation.

$$X_s^{(0)} = \sum_{r \in \eta^{(0)}} \theta_r^{(0)}(X_{s+r}^{(0)} + X_{s-r}^{(0)}) + e_s^{(0)}$$

where $e_s^{(0)}$, is zero mean, spatially correlated, Gaussian noise with variance $[\sigma^{(0)}]^2$. Hence a GMRF process can be completely characterized by the set of parameters $(\underline{\theta}, \sigma^2)$.

Let $\Omega^{(k)}$ represent the lattice obtained by subsampling (or subsampling followed by local averaging) $\Omega^{(0)}$, k times. It can be shown that the GMRFs lose Markovianity on resolution transformation. However, if the lower resolution data are modeled by the exact non-Markov Gaussian processes, conventional optimization techniques based on Markov properties cannot be applied. We present a Markov approximation based on minimizing the Kullback-Leiber distance between the local conditional distributions, such that, if $P^{(k)}(X^{(k)})$ is the non-Markov pdf at the k th resolution then a GMRF approximation $P_g^*(k)(X^{(k)})$ can be obtained such that, for $r \in \eta^{(k)}$, where $\eta^{(k)}$ is the neighbor set,

$$P_g^*(k)(X_s^{(k)}/X_{s+r}^{(k)}) = \arg \min_{P_g^{(k)}} D[P^{(k)}(X_s^{(k)}/X_{s+r}^{(k)}) \parallel P_g^{(k)}(X_s^{(k)}/X_{s+r}^{(k)})],$$

where $D(\cdot|\cdot)$ is the Kullback-Leibler distance.

The parameters $(\underline{\theta}^*(k), [\sigma^{*(k)}]^2)$ corresponding to $P_g^*(k)(X^{(k)})$ are obtained as follows:

$$\underline{\theta}^*(k) = \arg \min_{\underline{\alpha}} E_{P^{(k)}} [X_s^{(k)} - \sum_{r \in \eta^{(k)}} \alpha_r (X_{s+r}^{(k)} + X_{s-r}^{(k)})]^2$$

and using the $\hat{\theta}^{*(k)}$ obtained, we can estimate the $[\sigma^{*(k)}]^2$ as,

$$[\sigma^{*(k)}]^2 = E_{P^{(k)}}[X_s^{(k)} - \sum_{r \in \eta^{(k)}} \theta_r^{*(k)}(X_{s+r}^{(k)} + X_{s-r}^{(k)})]^2.$$

The classification problem involves assigning a label, say from $(1, 2, \dots, V)$ to the sites on the lattice, where the label stands for the class to which the site belongs to. Different classes in an image are modeled by GMRF models with different parameters represented by $(\hat{\theta}(v), \sigma^2(v))$, $v \in \{1, 2, \dots, V\}$. The label process is modeled by an MRF.

$$P(L = l) = \frac{1}{Z} \exp(\beta \sum_{s \in \Omega} U(l_s))$$

where, $U(l_s)$ is the number of neighbors that have the same label as l_s .

Now given the intensity process, the label process can be estimated by minimizing a suitable criterion. We have used the iterated conditional mode (ICM) method, a greedy algorithm that converges to a local maxima. The ICM solution is obtained by:

$$\begin{aligned} \max_{L_s} P(L_s/L_r, X_s, X_r) = \\ \min_{L_s} \log(\sigma(L_s = v)) - \beta U(L_s = v) \\ + \frac{1}{2} [x_s - \sum_{r \in \eta} \theta_r(L_s = v)(x_{s+r} + x_{s-r})]^2. \end{aligned}$$

Figure 1 shows a section of a thematic mapper data in one band and Figure 2 shows the corresponding classification map obtained using the multiresolution scheme for the classes river, forest, deforestation and regrowth, with an interior correct classification of 86.8%.

As a first step towards developing a parallel implementation of this algorithm, we studied a simpler, multiresolution algorithm that constructs what is called a Laplacian pyramid. In order to support the implementation of this code, we recently extended the multiblock PARTI system [8] to support the diagonal neighbor communications needed to develop typical local operations in image processing (most scientific computations are performed using only horizontal and vertical adjacencies).

The algorithm is essentially the one described in the paper by Burt and Adelson [2], with a few differences. The following is a brief summary of the algorithm.

The first step is the building of a pyramid of low-pass filtered versions of the image, obtained by applying a convolution operator at every alternate point. In this implementation the operator at every such point was chosen as the locally symmetric 5×5 kernel function that resembles the Gaussian function as was suggested in [2] with the peak value of the Gaussian kernel chosen as 0.4.

The Laplacian pyramid of images is then obtained by subtracting each pixel at level $i+1$ of the Gaussian pyramid from their corresponding parent pixel at level

i , where the resolution decreases with increasing i . In this case there are four parents at a higher resolution for each child at an immediately lower resolution.

The algorithm was implemented in sequential C code and then the parallel version of the code was developed using the multiblock PARTI library. Apart from providing computational speedup, the library facilitated efforts to try out different distributions of the image among the processors, such as distribution along each dimension or along both dimensions, with minimal changes to the code.

As the algorithm requires the operation of a 5×5 kernel over selected points in the image, communication is required between processors in the form of local copies of data in adjacent blocks that are stored in internal ghost cells. The multiblock PARTI library handles this by building a schedule and performing the requisite data movement. Communication between consecutive levels was turned into local, on-processor, copies by aligning each level exactly above its predecessor so that a simple relationship could be used to access corresponding pixels in adjacent levels during the construction of the Laplacian pyramid.

The algorithm was run on both a CM5 and an Intel Paragon. Here, we present only the results on the Paragon - an extended discussion can be found in [7]. The results are illustrated in Tables 1 to 2. These tables show, as a function of number of processors and data distribution model, the computation, communication and scheduling time for the pyramid algorithm for a 768×768 and 1536×1536 image, respectively. The three data distribution schemes - BB, B* and *B - correspond to a square block, row and column distribution of the array. The three components of the running time correspond to the time required by multiblock PARTI to set up the communication schedules for message passing between processors (scheduling), the time required to move pixel values between processors to complete neighborhoods at processor boundaries (communication) and the time taken to compute the Laplacian pyramid. Five runs were made for each of the 9 cases tabulated for the CM5 and the Paragon machines. The maximum value of the total time, which is the sum of the communication and computation times, over all the processors is obtained for each run and the median of these five values is used as the final result. The corresponding values of the schedule building time, the datamove time and the actual computation times are also part of the final result. Total time (excluding I/O) for the sequential run on a Sparc-10 was 15.47 seconds for the 768×768 image, and was 63.03 seconds for the 1536×1536 image.

We make the following observations:

1. The computation times scaled well on both the Paragon and the CM5.
2. The scheduling time increases slightly as we move to more processors, but would stabilize as larger processor sets are employed. Generally, the time it takes to set up the communication schedule depends on the number of processors that a given processor must communicate with.

Distribution	No. of Processors	Total Time	Schedule Time	Datamove Time	Computation Time
BB	8	0.983	0.042	0.048	0.894
	16	0.552	0.051	0.052	0.449
	32	0.361	0.045	0.091	0.225
B*	8	0.969	0.028	0.050	0.891
	16	0.538	0.029	0.063	0.446
	32	0.337	0.050	0.063	0.223
*B	8	0.989	0.036	0.047	0.906
	16	0.556	0.040	0.054	0.462
	32	0.343	0.043	0.060	0.234

Table 1: Results for Intel Paragon (768x768 image), in sec.

Distribution	No. of Processors	Total Time	Schedule Time	Datamove Time	Computation Time
BB	8	3.724	0.040	0.079	3.604
	16	1.946	0.040	0.099	1.807
	32	1.096	0.064	0.128	0.903
B*	8	3.726	0.027	0.087	3.611
	16	1.941	0.045	0.095	1.801
	32	1.062	0.055	0.106	0.901
*B	8	3.743	0.028	0.081	3.633
	16	1.961	0.044	0.081	1.835
	32	1.077	0.045	0.097	0.936

Table 2: Results for Intel Paragon (1536x1536 image), in sec.

- The communication time increases as we increase the number of processors, since the proportion of pixels that must be moved between processors increases as the number of processors increases, given a constant image size.

We observe that these results are obtained using essentially the same code running on both the Paragon and the CM5 (the only changes being those required to communicate with the file systems for image I/O), indicating the portability of the codes developed. Finally, in early versions of the program we observed that there were processor outliers in terms of timing (taking factors of ten times longer than other processors). We traced this to problems with image I/O and the underlying file systems of the machines, with some processors experiencing long delays in reading their part of the image from the parallel file system. While we did nothing to correct for this (simply inserting a synchronization point after the initial image reads to localize the problem) our experience does indicate the need for improved I/O subsystems on machines such as these.

4 Region classification

In this section we describe an approach to image classification that involves first segmenting an image into regions, and then classifying the regions using a two stage decision process. We first describe the basic approach, and then discuss issues in parallelizing the classification algorithms. Results of applying the classification algorithm to a piece of the LANDSAT

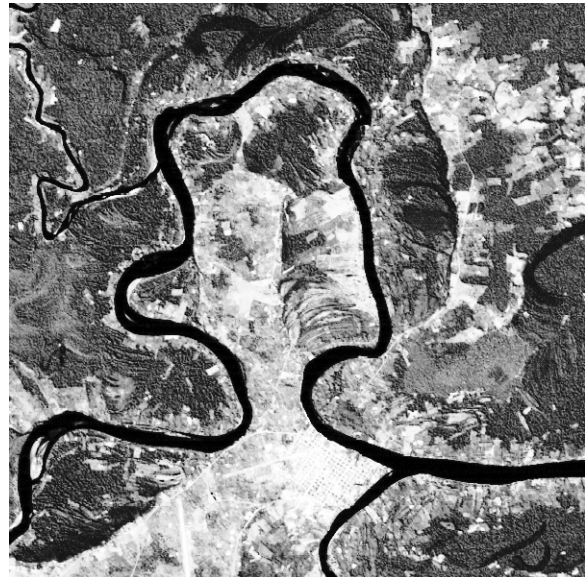


Figure 1: TM Data

scene used in the previous section for which we have extensive ground truth are then presented.

Our approach to image classification involves, in principle, four steps:

1. The multispectral image is first processed by a nonlinear filter called a symmetric neighborhood filter (SNF). This filter is an iterative local image analysis algorithm that sharpens the boundaries between regions while reducing the variability within regions. It is designed to retain both small and thin regions.
2. The enhanced image is segmented, hierarchically, into a set of regions using a connected component style of analysis. Each region is represented using a subset of its first and second order histograms (frequency distributions) of its interior pixels (those pixels not adjacent to pixels from other regions). Based on the results of a training phase, these histograms might be for single bands or pairs of bands.
3. Each region is tentatively classified by comparing one of its histograms of against the corresponding histogram for each class obtained during a training phase.
4. What we ordinarily discover during training is that when a region is misclassified, it is almost always the case that the second best choice for the region is the correct one. Therefore, a second stage of classification is performed in which, based on the initial classification of each region, a histogram of that region is compared to histograms of the initially chosen class and the mostly likely confused class. The histogram chosen will, generally be different for different classes. As an example, all regions might be initially classified by comparing their joint band3-band4 histograms against the band3-band4 histograms obtained during training for each class. Then, in the second stage, any region labeled as forest (for example), which might most frequently be confused with grassland, might have its band4-band5 histogram compared to those for forest and grassland to obtain a final classification based on interior pixels.

In fact, it is neither necessary nor desirable to compute the histograms of the individual regions during step 3 of the algorithm for the following two reasons:

- First, there may be many regions in the image, requiring a significant amount of storage for the complete set of histograms.
- Second, it is possible to directly compute the log probability that any region comes from a given class by replacing the (integer array) class histograms computed during training with (floating point array) class log probability arrays. Then, as the pixels in a given region are scanned, their log probabilities of belonging to any given class

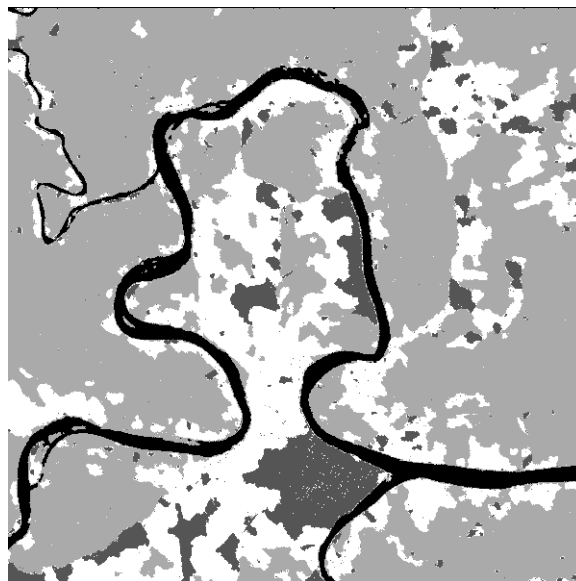


Figure 2: Classification

are summed up. This requires only storing one floating point variable per region per class, at the expense of additional floating point operations (compared to computing the actual histograms and then multiplying the frequencies by the log probabilities afterwards).

In this paper we will not address the issues raised by either the construction or classification of a hierarchical segmentation (in which each image pixel might belong to many regions at different levels of the hierarchy but which, obviously, belongs to only one ground category). We are developing a parallel C++ code for constructing the hierarchical segmentation using object-oriented versions of the CHAOS run time support libraries developed at the University of Maryland [1]. We focus, instead, on issues in parallelizing the classification step applied to an arbitrary level of the pyramid, corresponding to a fixed partition of the image pixels into regions. There are two possible ways to parallelize the classification step:

1. By region, assigning a subset of the regions in the segmentation to each processor.
2. By location, using standard techniques for dividing the image into nonoverlapping blocks.

The former approach has the disadvantage that we must either replicate the image in every processor or suffer large communications costs in moving pieces of image from processor to processor so that the processor associated with any region can collect the statistics over the entire spatial extent of the region. The second approach requires us to maintain a table in each processor as large as the total number of regions

in the image (since we do not know which regions in the segmentation belong to any element of the spatial partition), and then combining these tables over the entire processor set, after a local processing stage, using a simple logarithmic merging algorithm. Additionally, the second approach can be implemented using the multi-block PARTI run time support system developed at the University Maryland, resulting in both efficient and portable code.

The classification accuracy of this approach is very high, especially for large regions of the segmentation. The algorithm was applied to the image in Figure 1. We used as a training set the regions in the manually prepared classification map and additionally used that map as the given segmentation. While, in practise, one would classify an automatically segmented image (from which the models would also be acquired), the class map used as a segmentation presented a serious challenge to the classifier because of misregistration between the class map boundaries and true class boundaries (resulting from both errors in the gross hand classification of the data and in registration of the original class maps which were constructed in a different image coordinate system than the final class map). This both pollutes the estimated models and, potentially, leads to classification errors for small and thin regions. Furthermore, the current classification model treats interior and border pixels identically, even though we know that the spectral distributions of border pixels are different than those for interior pixels due to mixing effects, for example. Finally, only the first stage of the two stage classification algorithm is reported here. Nevertheless, the classifier obtained an overall classification rate of over 93%.

The algorithm was run using from 4 to 32 processors of the CM5. In all cases, the time taken to combine the probability tables from the processors was the same - .03 seconds. The computation time to construct the table on individual processors scaled as one would expect, taking 1.86 seconds on the 4 processor configuration and .23 seconds on the full 32 processors. Current research involves implementing the second stage of classification, as well as adding a special analysis for border pixels, which can be mixed and misaligned due to segmentation errors.

References

- [1] G. Agrawal, A. Sussman, and J. Saltz, "Compiler and Runtime Support for Structured and Block Structured Applications," in *Proceedings Supercomputing '93*, pp. 578-587, IEEE Computer Society Press, Nov. 1993. An extended version available as University of Maryland Technical Report CS-TR-3052 and UMIACS-TR-93-29.
- [2] P. Burt and E. Adelson, "The Laplacian Pyramid as a Compact Image Code," *IEEE Transactions on Communications*, Vol. COM-31, No. 4, pp. 532-540, 1983.
- [3] R. Chellappa, "Two-dimensional Discrete Gaussian Markov Random Field Models for Image Processing," in *Progress in Pattern Recognition* (L. N. Kanal and A. Rosenfeld, eds.), pp. 79-112, Elsevier, 1985.
- [4] F. S. Cohen and D. B. Cooper, "Simple Parallel Hierarchical and Relaxation Algorithms for Segmenting Noncausal Markovian Random Fields," *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol. 9, pp. 195-219, March 1987.
- [5] B. Gidas, "A Renormalization Group Approach to Image Processing," *IEEE Trans. Patt. Anal. Mach. Intell.*, Vol. 11, No. 2, pp. 164-180, 1989.
- [6] S. Lakshmanan and H. Derin, "Gaussian Markov Random Fields at Multiple Resolutions," in *Markov Random Fields: Theory and Applications* (R. Chellappa, ed.), pp. 131-157, Academic Press, 1993.
- [7] R. Parulekar, L. Davis, R. Chellappa, J. Saltz, A. Sussman, and J. Townshend, "High performance computing for land cover dynamics," in *International Conference on Pattern Recognition*, 1994.
- [8] A. Sussman, J. Saltz, R. Das, S. Gupta, D. Mavriplis, R. Ponnusamy, and K. Crowley, "PARTI Primitives for Unstructured and Block Structured Problems," *Computing Systems in Engineering*, Vol. 3, No. 1-4, pp. 73-86, 1992. Papers presented at the Symposium on High-Performance Computing for Flight Vehicles, December 1992.