6-2004

A Parallel Fuzzy C-Mean algorithm for Image Segmentation

Shahram Rahimi
Southern Illinois University Carbondale, rahimi@cs.siu.edu

M. Zargham
Southern Illinois University Carbondale

A. Thakre
Southern Illinois University Carbondale

D. Chhillar
Southern Illinois University Carbondale

A Parallel Fuzzy C-Mean Algorithm for Image Segmentation

S. Rahimi, M. Zargham, A. Thakre, D. Chhillar
Department of Computer Science
Southern Illinois University
Carbondale, IL 62901, USA
{rahimi, mehdi, athakre, divyac}@cs.siu.edu

Abstract - This paper proposes a parallel Fuzzy C-Mean (FCM) algorithm for image segmentation. The sequential FCM algorithm is computationally intensive and has significant memory requirements. For many applications such as medical image segmentation and geographical image analysis that deal with large size images, sequential FCM is very slow. In our parallel FCM algorithm, dividing the computations among the processors and minimizing the need for accessing secondary storage, enhance the performance and efficiency of image segmentation task as compared to the sequential algorithm.

I. INTRODUCTION

Clustering is the process of grouping a data set in a way that the similarity between data within a cluster is maximized while the similarity between data of different clusters is minimized [2]. Clustering is used for pattern recognition in image processing, and usually requires a high volume of computation. This high volume computation requires considerable amount of memory which may lead to frequent disk access, making the process inefficient. With the development of affordable high performance parallel systems, parallel algorithms may be utilized to improve performance and efficiency of such tasks.

To recognize a given pattern in an image various techniques can be utilized, but in general two broad categories of classifications have been made: unsupervised techniques and supervised techniques. In unsupervised classification methods, data items that are to be clustered are not pre-classified while in supervised clustering the data points are pre-classified. One of the well-known unsupervised algorithms that has been applied to many applications is fuzzy c-mean (FCM) [1]. FCM is sensitive to initial guess with respect to speed and stability [6]. The extensive computation and memory requirement needed for executing FCM is a big hurdle which we have tried to overcome in this paper.

The rest of the paper is organized as follow. Section 2, gives a short overview of fuzzy c-mean algorithm for pattern recognition. In Section 3, image segmentation and the need for pattern recognition in image processing is described. Section 4, presents the parallel FCM algorithm for image clustering and its implementation issues. In section 5, performance of the algorithm is evaluated and discussed. Finally, section 6 concludes the paper.

II. FUZZY C-MEAN ALGORITHM

Fuzzy c–mean algorithm is one of the best known fuzzy clustering algorithms which is classified as constrained soft clustering algorithm [1]. A soft clustering algorithm finds a soft partition of a given data set by which an element in the data set may partially belong to multiple clusters. Moreover, there is a constraint on the function that the membership degree of a point in all the clusters adds up to 1.

Consider X to be a set of n data points, $X=\{x_0, x_1, ..., x_{n-1}\}$ and P to be a set of k clusters such that $P=\{c_0, c_1, ..., c_{k-1}\}$. Each of the data points such as x_i may belong to one or more clusters depend on its degree of membership. Point x_i belongs to cluster c_j as long as its degree of membership to c_j, produced by membership function $\mu_{c_j}(x_i)$, is more than zero.

Clustering a data set requires finding the center of each cluster and deciding to which cluster each point belongs to. In FCM to find the center of a cluster, the sum of the distance between points in the cluster and its center is used as criterion. The criteria is represented by an objective function, *J*, which needs to be minimized with respect to P, a fuzzy c-partition of the data set, and V, a set of K prototypes for cluster centers (v_j):

$$J_m(P,V)=\sum_{j=1}^{K}\sum_{i=1}^{n}(\mu_j(x_k))^m \| x_i - v_j \|^2 \qquad (1)$$

The formula incorporates fuzzy membership degree μ_c and an additional parameter *m*, as a weighted exponent for the fuzzy membership. Parameter *m* is the value that determines the degree to which partial members of cluster affect the clustering results. x_i is the value of the data point under consideration and v_j is the center of cluster C_j. At the beginning of the process V is initialized with some prototype values that get updated during the process.

The function J is minimized by using the following equations for updating the membership degrees and V iteratively until $|V_i - V_{i-1}| < \epsilon$.

$$\mu_{cl}(x_i) = \frac{1}{\sum_{j=1}^{K}\left(\frac{\| x_i - v_l \|^2}{\| x_i - v_j \|^2}\right)^{\frac{1}{m-1}}} \quad (2)$$

Where $1 \leq l \leq k$ and $1 \leq i \leq n$

$$v_l = \frac{\sum_{i=1}^{n}(\mu_{cl}(x_i))^m x_i}{(\mu_{cl}(x_i))^m} \quad (3)$$

It can be analyzed that these two equations carry heavy computational load for large data sets such as high resolution pictures which requires availability of large memory [2]. Unavailability of enough memory for the computation may cause the utilization of slower storage unites which in turn causes further slow down of the process. The parallel algorithm given in section 3 seeks to reduce the computational load and increase main memory availability by dividing the computation and data among multiple processes. This will not only speed up the process, but also ensure that the computation can be done in the main memory, thus reduces disk access during the process.

III. Image Segmentation

Extraction of useful information from an image is called image analysis. The most important step in image analysis is image segmentation [7]. By segmenting the image various patterns of interest may be discovered. In the area of pattern recognition and image processing, unsupervised clustering is often used to perform the task of "segmenting" an image [1]. The pixels on an image are partitioned in regions that correspond to different objects or different faces of object in the image [1]. It has been found that gray scale feature values offers better discernable results than the RGB counterpart [8]. For the purpose of this paper and for proof of concept, a color to gray scale conversion scheme was devised according to the following straightforward mapping equation:

$$Y = 0.3\,R + 0.6\,G + 0.1\,B \quad (4)$$

Y denotes the pixel values of the gray scale image. The conversion of RGB to grey scale reduces the data size by a factor of 3 since only Y is used as the pixel attribute for clustering instead of RGB.

The results obtained by applying FCM on a grey scale image file need to be defuzzified to obtain the resultant output image file. The defuzzification process begins by transforming the final membership function matrix back to image by deciding to which cluster each pixel belongs. The decision on which cluster a pixel belongs to is made based on the winning μ_{ci}, i.e. selecting the cluster for which the pixel has the highest degree of membership. If a pixel belongs to ci, then it will be painted with the color code of cluster *i*; the resulting image will be partitioned by color-coded regions. This reversed mapping process forms the defuzzification task.

The next step is to map these color-coded regions to the original image clusters. Color-to-image mapping is an arbitrary process arranged at one's discretion. In this implementation, we colored the darker colored pixel cluster by white color and the lighter colored pixels cluster by black color and finally the pixel in the original file is painted as black or white depending on the membership function into the output file[3].

IV. Parallel Fuzzy C-Mean Algorithm

Improving the performance of the FCM based image segmentation by distributing computation and main memory usage was the main objective for design and implementation of this parallel FCM application. This algorithm is designed for an OSCAR cluster using SPMD model and Message Passing Interface (MPI).

This parallel FCM algorithm divides the image pixels equally among the processors so that each processor handles n/p data points (n is the total number of pixels and p is the number of processors involved in the computation). The fuzzy membership function (2) is distributed among the processors and is used for the calculation of the degree of membership, $\mu_{c_j}(x_l)$, only for the local data set. By dividing the data set among p processors the likelihood of carrying out the computation only on the main memory of the processors and without the need to access secondary storage will be increased. This enhances the performance and efficiency as compared to FCM algorithm.

We now describe the algorithm as follows. For given(X, C, m, ε), where $X = \{x_0, x_2, \ldots\ldots\ldots x_{n-1}\}$ is a set of n data points, $C = \{c_0, c_2, \ldots\ldots\ldots c_{k-1}\}$ is the set of k clusters to be formed, m is a weight that determines the degree to which partial members of a cluster affect the clustering results, and *E* (a vector of k ε) indicates the precision of the results, we have $V = \{v_0, v_2, \ldots\ldots\ldots v_{k-1}\}$, a set of k centers of the clusters and $M = \{\mu_0(x_0), \ldots, \mu_0(x_{n-1}), \ldots, \mu_{k-1}(x_0), \ldots, \mu_{k-1}(x_{n-1})\}$, a vector of membership degrees for each point to each cluster. The algorithm will take the following steps:

<u>On the initiating processor:</u>

1. Divide *X* into n/p subsets and send each subset to a participating processor using a one to all personalized broadcast.

2. One to all broadcast Vector *V*.

On each participating processor:

3. calculate:

$$A[j,l] = \mu_{c_i}(x_l) = \frac{1}{\sum_{j=1}^{k}\left(\frac{\| x_l - v_i \|^2}{\| x_l - v_j \|^2}\right)^{\frac{1}{m-1}}} \quad (5)$$

Where $0 \le i \le k-1$, $\frac{n}{p} \times P_{id} \le l < \frac{n}{p} \times (P_{id}+1)$, and $0 \le P_{id} \le p-1$. $A[j,l]$ is a member of matrix A, partial membership matrix.

4. calculate:

$$Q[l] = \sum_{j=1}^{n/p} (A[j,l])^m \quad (6)$$

and form vector Q from Q[l] for $0 \le l \le n/p$. Vector Q would be partial denominator of the equation that calculates V in step

5. calculate:

$$L = A^m \times \begin{bmatrix} x_{n.Pid/p} \\ x_{(n.Pid/p)+1} \\ \vdots \\ x_{\frac{n}{p}\times(Pid+1)} \end{bmatrix} \quad (7)$$

where L is a vector of k values which form the partial numerator of the equation that calculates V in step 7.

6. Perform all reduce operation for L and Q to form final numerator L^* and final denominator Q^*.

7. for $0 \le i \le k\text{-}1$ calculate:

$$V[i] = \frac{L^*[i]}{Q^*[i]} \quad (8)$$

8. If $V(current) - V(previous) \ge E$ then go to step 2. If the difference between previous and new clusters' center vector is less than E then stop.

This algorithm was implemented on a Red Hat based cluster with nine nodes implemented using OSCAR (Open Source Cluster Application Resource) software package. Pentium IV Linux box with 1 GB memory was used as the head node while the other 8 nodes were identical Pentium 2 GHz Linux boxes each with 750 MB memory. In our program, we used C programming language and Massage Passing Interface (LAM-MPI). Next paragraph provides some more details about our implementation.

In our program, before the start of parallel FCM process, color images are converted to grey scale image (Fig. 1). The conversion of the color image to gray is actually a filtering pre-stage that drops unnecessary information from the image for low resolution segmentation tasks. We did not parallelize the implementation of this step.

To divide the image file among p processors, it is cut horizontally into equal blocks of h/p rows where h presents number of the rows in the image. The image height is not always an integral multiple of the number of processors. The remaining rows cannot be greater than n – 1, therefore they are divided among the processors starting with the processor having the lowest ID. For example, if the image size is 270 and there are 4 processors each processor gets 67 rows and 2 rows are left, so processor 0 and processor 1 each get 1 extra remaining row. In addition, the weight component (m) was set to 2. After the completion of the computation of the centers of the clusters and final degree of membership matrix, accordance to the parallel algorithm discussed above, each processor defuzzifies its local data and sends it back to the initiating processor. The initiating processor, in turn, writes the received data into the output file to form the segmented image.

V. Performance Evaluation and Discussion

In this section, we report the performance evaluation of the parallel FCM algorithm against a sequential FCM, both executed on our OSCAR cluster (mentioned earlier). Parallel and sequential FCM algorithms were used to segment three different images sized from 155KB to 12 MB to a black and a white cluster (Fig. 2).

The first image (image 1) was 155KB, 270 × 200 pixels, 24 bit RGB image. For this image, which is a fairly small one, the speed up of the parallel FCM program with 2 processors was about 1.1 which was increased by increasing the number of the processors to no more than 2.2 for seven processors and then slightly declined for 8 and 9 processors. The decrease of speed up for more number of processors (after 7) was due to the small size of the image that caused the communication cost to exceed the computation cost. Next, a larger size image, with 800×600 pixels and size of 1.37MB, was used (image2). The sequential algorithm took 12.41 seconds to segment the image, while the parallel algorithm with 9 processors took only 1.55 seconds giving a speedup of 8. We then segmented an image of dimensions 2580×1720 pixels and of size 12.6 MB (image3). For a single processor the image was segmented in 115.80 sec, while the parallel algorithm with 9 processors took only 12.90 sec. Therefore, a speedup of 8.97 was achieved, which indicates an efficiency factor close to 1. This speed up efficiency factor is due to reduced disk access and parallel computation on image data. The speed up gained by using the parallel FCM algorithm (with different number of processors) for segmentation of the above images is depicted in Fig. 3.

As the results in Fig. 3 reflect, the speedup obtained for images over 1MB of size justifies the use of the parallel FCM algorithm. Moreover, for oversized images this algorithm could achieve super liner speed up which reflects the decrease of disk access.

Fig. 1 Original color image (Left) and gray scale converted image (right).

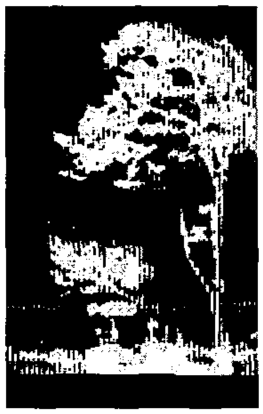
Fig. 2 Conversion of Gray scale image (left) to segmented black and white image (right).

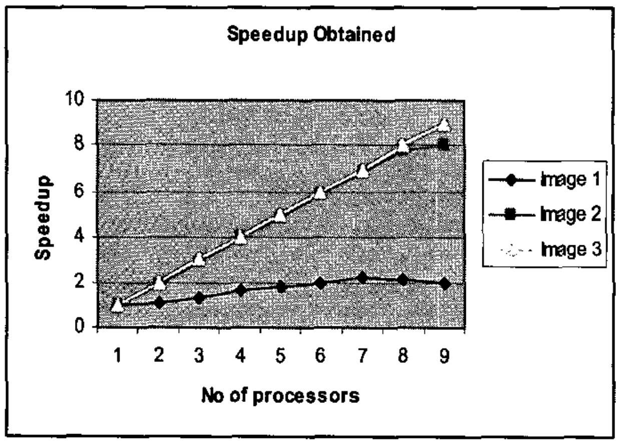

Fig. 3 Speed up of parallel FCM algorithm for clustering images 1, 2, and 3 to black and white.

VI. Conclusion

We provided a scalable parallel algorithm to improve the performance of fuzzy c-mean algorithm for image segmentation. The algorithm was implemented on an OSCAR cluster with nine nodes and achieved significant performance improvements for images with over 1MB of size. Decreasing the number of disk access, parallel FCM algorithm could even achieve super liners speed up. Practical applications for this algorithm include medical image segmentation and geographical image analysis among many others. Future works can attempt to use this algorithm for practical purposes in particular applications.

References

[1] J. Yen and R. Langari, *Fuzzy logic,* Prentice Hall, chapter13, 1999.

[2] T. Kwok, K. Smith, S. Lozano, and D. Taniar, "Parallel fuzzy c-means clustering for large data sets," In Burkhard Monien and Rainer Feldmann, editors, *EUROPAR02*, volume 2400 of *LNCS*, pp. 365-374, 2002.

[3] S. Chuai-Aree, C. Lursinsap, P. Sophatsathit, and S. Siripant, "Fuzzy c-mean: A statistical feature classification of text and image segmentation method," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 9, no.6, pp. 661-671, 2001.

[4] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact, well separated clusters," *Journal of Cybernetics*, vol. 3, pp. 32-57, 1973.

[5] D. Judd, P. K. McKinley, and A. K. Jain, "Large-scale parallel data clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 871-876, August 1998.

[6] N.A. Mohamed, M.N. Ahmed, and A. Farag, "Modified fuzzy c-mean in medical image segmentation," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, Piscataway, NJ USA, 1999, vol.6, pp.3429-3432.

[7] R. C. Gonzalez, R. E. Woods, *Digital image processing*, Pearson Education, 2002.

[8] L. O. Hal, A. M. Bensaid, L. P. Clarke, R.P. Velthuizen, M. S. Silbiger,and J.C. Bezdek, "A comparison of neural network and fuzzy clustering techniques in segmenting magnetic resonance images of the brain," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp 672 -682, Sept. 1992.

[9] H. R. Tizhoosh, "Fuzzy image processing: potentials and state of the art," in *Proceedings of 5th International Conference on Soft Computing*, Iizuka, Japan, October 16-20, vol. 1, pp. 321-324, 1998.