IMAGE SEGMENTATION WITH CYCLIC LOAD BALANCED PARALLEL FUZZY C - MEANS

by

MOGANA VADIVELOO

Thesis submitted in fulfillment of the requirements for the degree of Master of Science

December 2010

DECLARATION

Name**: MOGANA VADIVELOO**

Matric No**: PCOM 0053 / 09**

Faculty**: SCHOOL OF COMPUTER SCIENCES**

Thesis Title**: IMAGE SEGMENTATION WITH CYCLIC LOAD BALANCED PARALLEL FUZZY C - MEANS**

I hereby declare that this thesis in I have submitted to **SCHOOL OF COMPUTER SCIENCES** on **December 2010** is my own work. I have stated all the references used for the completion of my thesis.

I make this declaration with the believe that what is stated in this declaration is true and the thesis as forwarded is free from plagiarism as provided under Rule 6 of the Universities and University Colleges (Amendment) Act 2008, University Science Malaysia Rules (Student Discipline) 1999.

I conscientiously believe and agree that the University can take disciplinary actions against me under Rule 48 of the Act should my thesis be found to be the work or ideas of other persons.

Students Signature: ……………………….. Date:

Acknowledgement of receipt by: ……………………. Date:

ACKNOWLEDGEMENT

Firstly, I would like to thank the Malaysian Government for sponsoring my study. I would like to acknowledge and extend my heartfelt gratitude to both of my supervisors, Prof. Rosni Abdullah and Assoc. Prof. Mandava Rajeswari for their vital encouragement and invaluable support assistance.

I dedicate this dissertation to both my parents, who moulded me into the person that I am today. Lastly my grateful thanks to my fellow lab mates especially to Miss. Anusha and Mr. Ahmad Adel Abu Shareha.

Thank you very much.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SEGMENTASI IMEJ MENGGUNAKAN STRATEGI PENGIMBANGAN BEBAN SIKLIK DALAM ALGORITMA FUZZY C – MEANS SELARI

ABSTRAK

Fuzzy C - Means (FCM) ialah satu daripada algoritma pengelompokan yang paling kerap digunakan dalam segmentasi imej. Dalam algoritma FCM, suatu piksel boleh menjadi sebahagian daripada dua atau lebih pengelompokan dan mampu mengekalkan lebih banyak maklumat daripada imej dan yang paling penting, dapat mengelakkan optima setempat. Namun begitu, FCM melibatkan penggunaan komputer yang lebih intensif apabila mensegmen set data imej yang besar dan dalam kes seperti ini, algoritm FCM selari telah digunakan. Versi selari ini juga diimplementasi untuk meminimumkan kos komunikasi selain untuk mengurangkan kos perkomputeran. Untuk menjadikan pendekatan ini berkesan dan untuk meningkatkan kecekapan pemprosesan selari, strategi pengimbangan beban siklik dalam seni bina multiteras telah digunakan dalam kajian ini. Kajian ini menunjukkan peningkatan yang signifikan pada prestasi algoritma FCM selari. Strategi pengimbangan beban siklik yang digunakan dalam algoritma FCM selari telah memberikan keputusan yang baik dan menghasilkan kelajuan serta kecekapan yang memuaskan terutamanya apabila saiz pengelompokan adalah besar berbanding dengan algoritma FCM selari yang sedia ada.

IMAGE SEGMENTATION WITH CYCLIC LOAD BALANCED PARALLEL FUZZY C – MEANS.

ABSTRACT

Fuzzy C - Means (FCM) is one of the most popular clustering algorithm that has been used in image segmentation. It allows one piece of pixels to belong to two or more clusters and able to retain more information from the image and most importantly avoids local optima. However when FCM works with large image data set, it becomes more computationally intensive, and in such a case parallel FCM has been implemented. This parallel version was also implemented to minimize the communication cost other than to reduce the computational cost. In order to make this approach effective and to increase the efficiency of parallel processing, the cyclic load balancing strategy on multicore architecture was deployed in this research. The work presents a significant improvement to the performance of the parallel FCM algorithm. The cyclic load balancing strategy deployed in parallel FCM algorithm scales well and presents good speed up and efficiency especially when the size of number clusters is large as compared to the existing parallel FCM algorithm.

CHAPTER 1

INTRODUCTION

1.0 Introduction

Image segmentation is the process of partitioning image into desired regions according to some image property such as image intensity. One of the most popular techniques used for the segmentation problem is clustering. In clustering the goal is to produce coherent clusters of pixels. All the pixels in a cluster are as similar as possible with respect to some image property such as pixel color, texture, or intensity. However the adjacent clusters are significantly different with respect to the same image property. The goal of this image segmentation is to represent a meaningful representation and easy analyzation. Medical imaging, tumors locating, diagnosis are the examples of some applications of image segmentation.

There are many variants of data clustering techniques used for image segmentation such as K – Means, Fuzzy C – Means (FCM), ISODATA, but however FCM has been very popularly used to solve the image segmentation problem since last decades (Shen and Li, 4 June 2009). FCM is a method of clustering that allows one piece of pixels to belong to two or more clusters (Shen and Li, 4 June 2009). The introduction of fuzziness for the belongingness of each image pixels makes the FCM clustering algorithm able to retain more information from the image and most importantly to avoid local optima (Shen and Li, 4 June 2009).

However when FCM works with large image data set, it becomes computationally intensive (Wang et al, 2008). Thus the sequential version of FCM becomes very inefficient. Due to that, the parallel version has been implemented and the nature of the clustering algorithm makes it suitable to be parallelized as well (Wang et al, 2008).

In the parallel version of implementation the data sets are divided equally among the available number of threads or processors, as to calculate the location of cluster centers simultaneously (Modenesi et al, 2008). However this approach requires high communication cost due to the need of frequent interaction among threads or processors (Modenesi et al, 2008). As to minimize the communication cost, implementation has been done by distributing the partitions of clusters over threads or processors making the threads or processors in charge calculating the partitions and at the end the master thread or processor indicates the result of the best partition of clusters (Modenesi et al, 2008). In order to make this approach effective, a load balancing strategy is proposed in order to ensure the threads or processors receiving the balanced charge of partitions to process as to minimize the total time of the parallel processing.

This chapter gives some background information on digital image and its properties. It will also state the problem statement, objective, and the scope of the research. The motivation and contribution of the research will also be presented in this chapter. Finally the research methodology is also discussed in this chapter.

1.1 Background Material

The data that will be used throughout the research implementation is discussed in this section. The sequential FCM algorithm, the various types of implementations of parallel FCM algorithms and also the load balancing strategies are discussed in Chapter 2.

Image Vision perspective.

Throughout the research, the data sets that will be used are digital images. The following sub – sections are the explanations about the digital images.

1.1.1 Image

By image, we shall understand the usual intuitive meaning, and an example might be the image on the human eye retina or the image captured by TV camera. These example of images are two – dimensional (2D) (Vaclav Hilavac et al, 1998). The image can be modeled by a continuous function of two or three variables. Arguments are coordinates *(x, y)* in a plane, while if images change in time a third variable *t* might be added (Vaclav Hilavac et al, 1998).

1.1.2 Digital Image

A digital image consists of picture elements with finite size. These pixels carry information about the brightness of a particular location in an image (Vaclav Hilavac et al, 1998). Usually pixels are arranged into a rectangular sampling grid. Such a digital image is represented by a two – dimensional matrix whose elements are integer numbers.

The following explains the properties of a digital image: (Vaclav Hilavac et al, 1998)

A. Pixel - represents the elemental part of an image.

B. Pixel adjacency - a two pixels are called 4 – neighbors if they have distance D_4 = 1 from each other. Whilst 8 – neighbors are two pixels with D_8 = 1.

C. Regions - consisting of several adjacent pixels.

D. Border of a region – represents the set of pixels within the region that have one more neighbor outside.

E. Edge – local property of a pixel and its immediate neighbor. It is a vector given by a magnitude and direction.

F. Color – represents a property of enormous importance to human visual perception. Hardware will generally deliver color via RGB model (referring to red, green, and blue), thus particular pixel may have associated with it a three – dimensional vector (r,g,b) which provides the respective color intensities.

1.1.3 Image Quality

An image might be degraded during the capture, transmission, or processing and the measures of image quality can be used as the degree of degradation (Vaclav Hilavac et al, 1998).The quality required naturally depends on the purpose for which the image is used. There are two methods to access image quality and that are subjective and objective (Vaclav Hilavac et al, 1998).

1.1.4 Noise in images

Real images are often degraded by some random errors. This degradation is usually known as noise. Noise can occur during image capture, transmission, or processing and may be dependent on, or independent of, image content (Vaclav Hilavac et al, 1998).

1.2 Problem Statement

When FCM works with a very large data set of image, a large number of computations of distance between the cluster centers and each corresponding pixels are required. The computational time will be very high (Modenesi et al, 2007) and as to reduce the computational time, the parallel FCM has been implemented (Rahmi et al, 2004). Usually in parallel implementation the data sets are divided among the respective threads or processors to calculate the location of cluster centers concurrently (Murugavalli and Rajamani, December 2006). However this implementation requires high communication cost due to the frequent interaction among the threads or processors (Modenesi et al, 2008).

As to reduce the communication cost, another parallel approach has been implemented, and in this parallel FCM the partitions of clusters are distributed over threads or processors making the threads or processors in charge calculating the partitions and at the end the master thread or processor indicates the result of the best partition clusters (Modenesi et al, 2008). However the minimization of the total time cost is still a question in this approach, as the threads or processors are not receiving the balance charge of partitions to process. This occurs since the respective partitions require different processing time respectively.

Therefore the main problem here is to solve the question of minimizing the total time cost and maximizing the parallel processing efficiency of FCM algorithm when parallelization is done over the number of clusters.

1.3 Objectives

The objectives of this research are:

1. To study the cyclic load balancing strategy as to improve the performance of parallel FCM algorithm.
2. To compare the performance of both the parallel FCM algorithm and the load balanced parallel FCM algorithm.

1.4 Scope

This research is focused on designing and implementing a load balancing strategy on parallel FCM clustering algorithm. Our main aim is to minimize the total time cost and also to increase the efficiency of parallel processing by deploying the load balancing strategy.

1.5 Motivation

Parallel FCM algorithm was chosen because naturally FCM produces a better and more useful result in clustering the corresponding image pixels (Murugavalli and Rajamani, December 2006) (Shen and Li, 4 June 2009). Moreover when the algorithm has been parallelized, it becomes very efficient compared to the sequential version. This is because the computational time is minimized in parallel FCM when working with a very large data set of image. (Murugavalli and Rajamani, December 2006) (Wang et al,2008)

However this parallel version of FCM causes a high communication cost due to the need of frequent interaction among threads or processors ones (Modenesi et al, 2008). To overcome this problem, distribution of the partitions of clusters over threads or processors has been done ones (Modenesi et al, 2008). However the total time cost is still not minimized as the threads or processors are not receiving the balanced charge of partitions of clusters to process as the respective partitions require different processing time respectively.

Therefore to overcome the problem, a load balancing strategy has been proposed. This strategy will be developed in shared memory architecture because shared memory architecture has less communication compared to distributed memory architecture (Calvin and Snyder, 2009).

1.6 Contribution

The main contributions in this research are:

1. Improve the parallel FCM algorithm's performance with a load balancing strategy.
2. Implementation of load balancing strategy on shared memory computing architecture.

1.7 Research Methodology

The flow of the pre research procedures is shown in the flow chart below.

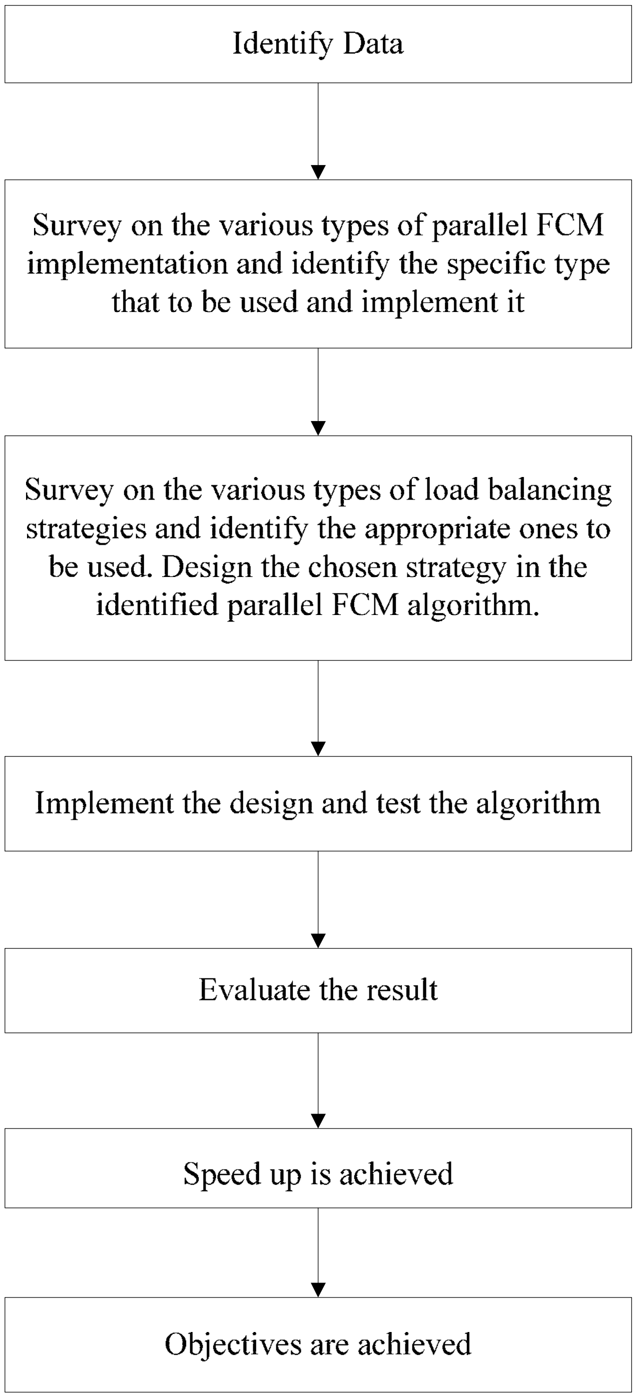
Identify Data
Survey on the various types of parallel FCM implementation and identify the specific type that to be used and implement it
Survey on the various types of load balancing strategies and identify the appropriate ones to be used. Design the chosen strategy in the identified parallel FCM algorithm.
Implement the design and test the algorithm
Evaluate the result
Speed up is achieved
Objectives are achieved

Figure 1.1 Flowchart of the pre research procedures.

The explanation of the pre research procedures are as follows:

A. Data Identification

Digital images will be used as the data set for parallel FCM algorithm. The images would be in grayscale. Both large and small sizes of images will be used to evaluate the algorithms' performances.

B. Parallel FCM Algorithm Identification and Implementation

In this step, the appropriate parallel FCM algorithm will be identified. There are variants of parallel FCM exist, and in such a case the most appropriate ones is selected to be implemented.

C. Identification and Designation of Load Balancing Technique in Parallel FCM

There are various types of load balancing strategies exist, the most appropriate one is chosen. In parallel FCM, there are different levels of classes and functions. The most suitable class or function level will be chosen as to design the identified load balancing technique. Both identifying the most appropriate class function level and designing the identified load balancing technique are very important as these will determine the objectives of the research.

D. The Design Implementation and Algorithm Testing

The design is implemented on the identified parallel algorithm and the testing is done using the appropriate hardware and software specifications.

E. The result evaluation

The obtained result is evaluated on the basis of certain factors. Those factors are the execution time, speed up and efficiency of both the balanced and unbalanced algorithms. If the performance of the balanced algorithm is higher than the unbalanced ones, then the objectives of this research are met.

1.8 Organization of Thesis

This chapter described the terminology that will be used in this thesis. It also stated the problem statement, objectives, and scope. Motivation of parallel FCM algorithm also has been highlighted. Finally the contribution and the research methodology of this research were also stated.

In Chapter 2, the introduction to clustering in image segmentation will be discussed. This follows by the discussion on sequential FCM algorithm and also clusters validity functions in FCM. Later on the background of parallel computing method will also be discussed and follows by the related work. Finally the load balancing and summary of the chapter will be covered.

In Chapter 3, the research procedures to implement the load balancing strategy in parallel FCM algorithm will be discussed. The sections consist of data identification, implementation of the identified parallel FCM algorithm and also designing the chosen load balancing technique in the identified parallel FCM algorithm. Later it focuses on the implementation and program testing and result evaluation. Finally the summary of the chapter will be covered.

While in Chapter 4, the performance evaluation of the parallel FCM and load balanced parallel FCM will be discussed. Results and discussions on the performance of both algorithms will be presented.

Finally in Chapter 5 the conclusion of the research and the future work will be covered.

CHAPTER 2

LITERATURE REVIEW

2.0 Introduction

The chapter consists of 6 sections that are Section 2.1 presents the Introduction to Clustering in Image Segmentation whilst section 2.2 describes the sequential FCM algorithm. In section 2.3 the Background of Parallel Computing Methods is discussed. Whilst in section 2.4 the Related Works are presented. In section 2.5 the Load Balancing Strategies are discussed and finally the Summary of this chapter is presented.

2.1 Introduction to Clustering in Image Segmentation

Clustering is a technique to group data items into a finite set of categories (clusters) and to summarize a dataset according to similarities among its objects (Hruschka et al, March 2009). The main aim in clustering is to maximize both the homogeneity within each cluster and the heterogeneity among different clusters. This technique is commonly used in image segmentation. Pixels of the corresponding image are segmented into clusters according to the same image property such as the pixel intensity.

There are 3 types of clustering techniques and that are overlapping, partitional, and hierarchical clustering (Hruschka et al, March 2009) (Ujjwal and Saha, 2010). The partitional and hierarchical clustering is related to each other in that a hierarchical clustering is a nested sequence of partitional clustering (Hruschka et al, March 2009).

In both clustering techniques, they represent hard partition data sets into a different number of mutually disjoint subsets. As for an example, a hard partition of a dataset X = $\{x_1, x_{2,}\ldots, x_n\}$ (where $x_j : j = 1,\ldots., n$ is the n^{th} dimensional feature or attribute vector) is a collection of $C = \{C_1, C_2,\ldots.., C_k\}$ (where k is a non – overlapping data subsets) where $C_i \neq \square$ (no null clusters) such that $C_1 \cup C_2 \cup \ldots. C_k = X$ and $C_i \cap C_m = \square$ for $i \neq m$ (Hruschka et al, March 2009). Example of popular hard partition clustering algorithm is K – Means.

When this condition of mutual disjunction ($C_i \cap C_m = \square$ for $i \neq m$) is relaxed, then it is known as the overlapping clustering. Overlapping clustering produces data partitions that can be either soft (each object fully belongs to one or more clusters) (Hruschka et al, March 2009) or fuzzy (each object belongs to one or more clusters to different degree) (Hruschka et al, March 2009). Fuzzy C – Means (FCM) is classified as a fuzzy partition clustering algorithm. Figure 2.1 shows an example of fuzzy clusters where H_1 and H_2 represent the data partitions while F_1 and F_2 are the overlapping clusters. The diagram shows that the data partition 4, 5, 6, and 7 belong to the overlapping clusters, F_1 and F_2.

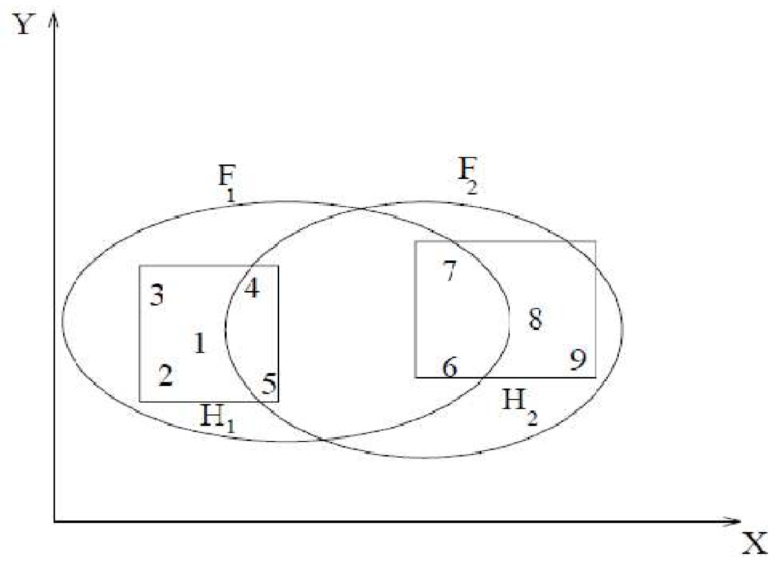

Figure 2.1 Fuzzy Clusters

(Jain, Murty, and Flynn, September 1999)

2.2 Fuzzy C - Means Algorithm (FCM)

Fuzzy C – Means (FCM) was developed by Dunn (1973) and was improved by Bezdek (1981) (Shen and Li, 4 June 2009). It is one of the most popular clustering algorithms used in image segmentation. The FCM is an iterative optimization that minimizes the cost function defined as follows:

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{j=1}^{n} U_{ij}^{m} d_{ij}^{2} \tag{1}$$

Equation (1) is the optimization function that has two main components U, V.

U_{ij} is the membership function of pixel x_j whereas m is a constant that represents the fuzziness value of the resulting partitions of clusters that to be formed; $m \in (1, \infty)$.

d_{ij} is the distance function between the pixel x_j and the cluster center v_i that normally addressed using Euclidean distance, $d_{ij} = ||x_j - v_i||$

$\sum_{i=1}^{c} U_{ij} = 1, \quad 1 \leq j \leq n$

The summation of membership function to each cluster of pixel x_j is 1.

$$0 < \sum_{j=1}^{n} U_{ij} < n, \quad 1 \leq i \leq c \tag{2}$$

$\sum_{i=1}^{c} \sum_{j=1}^{n} U_{ij} = n$

The objective function J_m in (1) is minimized when pixels nearby the cluster center v_i of corresponding clusters are assigned higher membership value, while lower membership values are assigned to pixels far from the cluster centers v_i (Zhou and Schaefer, 2009). The membership function and cluster centers are iteratively updated by:

$$U_{ij} = \left[\sum_{k=1}^{c} \left(\frac{d_{ij}^2}{d_{kj}^2} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{3}$$

$$v_i = \frac{\sum_{j=1}^{n} U_{ij}^m \; x_j}{\sum_{j=1}^{n} U_{ij}^m} \tag{4}$$

The following describes the steps in FCM: (Shen and Li, 4 June 2009) (Hruschka et al, March 2009).

a. The number of fuzzy clusters, c to be formed is initialized by the user.
b. The cluster centers, v_i in equation (4) is initialized by assigning a random value

c. At each iterations of k, the membership function, U_{ij} in equation (3) is calculated with the cluster centers, v_i in equation (4)

d. Later, the cluster centers, v_i in equation (4) are recomputed and updated.

e. When the overall difference in the cluster centers, v_i in equation (4) between the current and previous iteration is smaller than a given threshold value, ε the algorithm stops. Otherwise it returns to (b).

The complexity of FCM is O (k.n. c^2. s) (Hruschka et al, March 2009) where k is the number of iterations; n is the dimension of the data set; c is the number of clusters; and s is the number of attributes (as for example the image property such as the pixel intensity).

2.2.1 Cluster validity functions

Cluster validity functions are used in order to obtain a quantitative comparison. Generally in FCM there are two types of cluster validity functions that are often used. There are the fuzzy partition and feature structure (Shen and Li, 4 June 2009).

The representative functions for the fuzzy partition are partition coefficient, v_{pc} developed by Bezdek (1974) and partition entropy, v_{pe} by Bezdek (1975). They are calculated as below: (Shen and Li, 4 June 2009).

$$v_{pc} = \frac{1}{n} \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{2} \tag{5}$$

$$v_{pe} = \frac{-1}{n} \sum_{1=1}^{c} \sum_{j=1}^{n} \left(u_{ij} \log u_{ij}\right) \tag{6}$$

These validity functions are to show that the partition with the less fuzziness means better performance. The best clustering is achieved when the value in equation (5) is maximal and the value in equation (6) is minimal Shen and Li, 4 June 2009).

On the other hand the representative function for the feature structure is Xie – Beni's, V_{xb} by Xie and Beni (1991). It is defined as: (Shen and Li, 4 June 2009).

$$V_{xb} = \frac{\sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{2} \, ||x_j - v_i||^2}{n(min_{i \neq k}\{ ||v_k - v_i||^2)} \tag{7}$$

This validity function shows that the best clustering is achieved when the result generates samples that are compacted within one cluster and samples that are separated from different clusters. Minimizing V_{xb} in equation (7) leads to best clustering (Shen and Li, 4 June 2009).

PBM index by Parkhira (2003) is also another type of cluster validity functions (Pakhira et al, 2003) (Modenesi et al, 2007). The PBM index is defined as:

$$\text{PBM}(c) = (1/K.\ E_1/E_c.\ D_c)^2 \tag{8}$$

The factor E_1 is the sum of the distances of each pixel to the geometric center of the n data set. This factor is independent on the number of clusters and is defined as: (Pakhira et al, 2003)

$$E_1 = \sum_{t=1 \ldots n} d(x(t), n_0) \tag{9}$$

The factor E_c is the sum of within cluster distances of c clusters and is defined as follows: (Pakhira et al, 2003)

$$E_c = \sum_{t=1 \ldots n} \sum_{i=1..c} U_i(t) d(x(t), w_i)^2 \tag{10}$$

and D_c is the maximum separation of each pair of clusters defined as: (Pakhira et al, 2003)

$$D_c = max_{i\,, j=1..c}(d(w_i\,, w_j)) \tag{11}$$

The best clustering is achieved when the value of PBM index is maximal (Modenesi et al, 2007).

2.3 Background of Parallel Computing Methods

Parallel computing method is a technique used to increase algorithms performance and to complement with the increasing advancement of the hardware technology. In this section the discussion are focused on the architecture, decomposition technique, and the granularity of parallel computing.

2.3.1 Architecture

There are two main architectures in parallel computing and those are shared memory architecture and distributed memory architecture.

In the shared memory architecture, there is more than one processor or core in a Central Processing Unit (CPU) that shares one main memory. The number of processor or core could be two (dual), four (quad), between 8 and 16 (multi) and more than 16 many (Calvin and Snyder, 2009). There are three parallel application programming interface (API), and that are PThread, Java Thread and OpenMP. These API could be used to implement shared memory models. By using PThread and Java Thread parallel programmers have more control on choosing parts need to be parallelized in the algorithms compared to OpenMP (Calvin and Snyder, 2009). Figure 2.2 shows shared memory architecture.

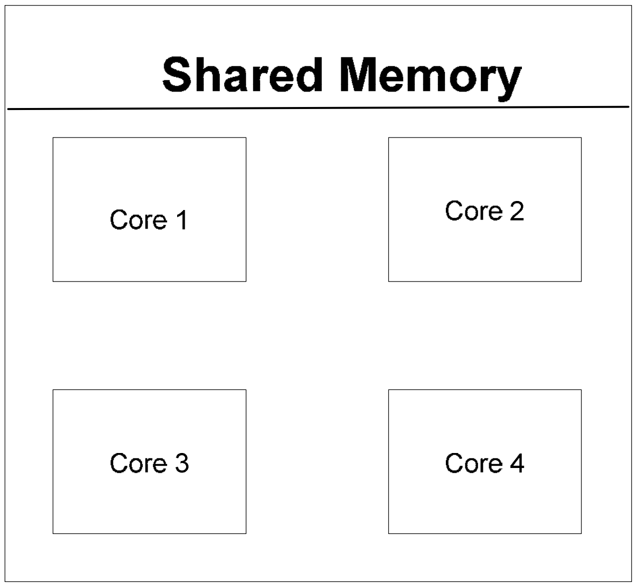

Figure 2.2 Shared Memory Architecture

Whilst in distributed memory architecture, processors are physically separated and they have their own memory (Calvin and Snyder, 2009). The communication between processors is through the interconnection network. Therefore distributed memory architecture has more communication between the processor compared to shared memory architecture (Calvin and Snyder, 2009). As for this reason, programmer must minimize the communication and must maximize the computation in distributed memory architecture (Calvin and Snyder, 2009). There are two parallel applications programming interface (API) for distributed memory architecture and that are Parallel Virtual Machine (PVM) and Message Passing Interface (MPI). PVM and MPI have the same functions, but in term technical support, MPI is better because MPI is the improved version of PVM (Calvin and Snyder, 2009). Figure 2.3 shows distributed memory architecture.

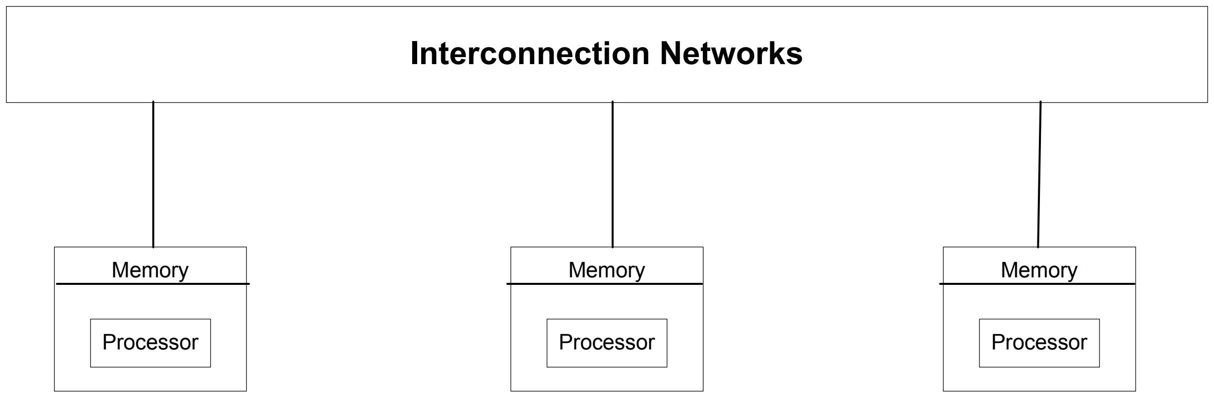

Figure 2.3 Distributed Memory Architecture

2.3.2 Methodical Design of Parallel Algorithm

Foster (1995) proposed a methodology that organizes the design process into four distinct stages that are partitioning, communication, agglomeration, and mapping. The first and most important step is to design and develop concurrent and scalable algorithm and the last phase fully focus on the locality and performances – related issues.

A. **Partitioning**

There is four basic partitioning or decomposition techniques used in parallel computing area. There are bit level decomposition, instruction decomposition, data decomposition, and task decomposition (Foster, 1995). However the most widely used decomposition techniques are data decomposition and task decomposition.

i. Data Decomposition

Data decomposition is a parallel decomposition technique that parallelism applied by performing the same operation to different data at the same time. The parallelism growth increases with the size of the data (Calvin and Snyder, 2009). There are many ways to decompose data into partitions to be distributed. The first way is one dimensional data distribution which includes block distribution and cyclic distribution (Foster, 1995). The second one is two dimensional data which includes block – block distribution, block – cyclic distribution and cyclic – block distribution (Foster, 1995).

ii. Task Decomposition

Task decomposition is a parallel decomposition technique that parallelism is applied by performing different operation at the same time or simultaneously. The number of task is fixed for parallelize, therefore, the parallelism is not scalable (Calvin and Snyder, 2009). As for example, if the tasks to parallelize are three tasks but the creation of threads is four threads. It means that one thread will be unusable and idle that leads to overhead. It will contribute to low performance.

iii. Instruction Decomposition

Instruction decomposition is a parallel decomposition technique that parallelism is applied by performing number of instructions that can be performed during a single clock (Calvin and Snyder, 2009). The parallelism is a set of instruction that do not depend on each other in execution.

iv. Bit Level Decomposition

Lastly, is the bit level decomposition, it is more or less similar to instruction decomposition technique and the oldest technique among the rest. The parallelism is based on the increasing of the processor word size (Calvin and Snyder, 2009).

B. Communication

Task communication depends on the problem itself. Sometimes, the problem can be decomposed in multiple tasks that can be executed in parallel with no intercommunication between them. On the other hand, some complex application shared their data and this involves intercommunication between the threads or processors (Foster, 1995). Generally, the communication depends in the information flow and task coordination during the partitioning stage. The communication pattern can be determined by the problem nature and the method of decomposition (Foster, 1995). The four well – known communication patterns in parallel application are local / global, structure / unstructured, static / dynamic, and synchronous / asynchronous respectively (Foster, 1995).

C. Agglomeration

In this stage, the tasks and communication structure defined above are evaluated in term of performance requirements and implementation costs. The tasks are grouped into sizes that will improve the performance or to minimize costs (Foster, 1995).

D. Mapping

The main responsibility of the mapping is to give each task to a thread or processor in such a way that it can achieve the maximum utilization of the CPU power and minimum cost of the inter – communication of the processors (Foster, 1995)

2.3.3 Granularity

The granularity is determined by the frequency of interaction between the threads or processors. There are two main type of granularity that is either coarse grain or fine grain (Calvin and Snyder, 2009). If the interaction of threads or processors is infrequent, then it is known as the coarse grain as shown in Figure 2.5, whilst if the interaction is frequent then it is known as fine grain as illustrated in Figure 2.4.

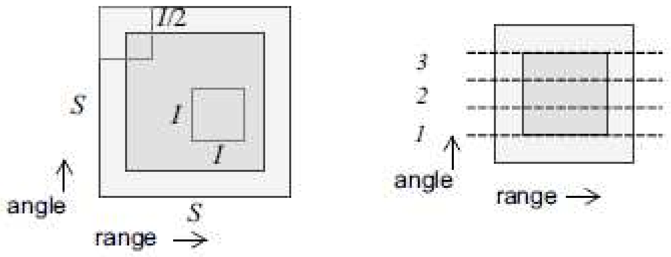

Figure 2.4 Fine granularities of image segmentation and the segmented blocks assigned, to the respective processors or threads (Ahlander et al, 2007).