Mining Association Rules in Medical Image Data Sets

by

Adepele Williams

A Thesis
Submitted to the Faculty of Graduate Studies
in Partial Fulfillment of the Requirements
for the Degree

Master of Science

Department of Computer Science
University of Manitoba
Winnipeg, Manitoba, Canada

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION PAGE

Mining Association Rules in Medical Image Data Sets

BY

Adepele Williams

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirements of the degree of Master of Science

Abstract

The goal of this thesis is to develop an efficient association-rule mining algorithm that is suitable for large data sets with long patterns. The FP-growth algorithm is a recent Association Rule Mining (ARM) technique which efficiently extracts knowledge, such as associative patterns between attribute values of large data sets, due to its highly compact data representation and pattern finding scheme.

The proposed algorithm, the Partitioned FP-growth (PFP-growth) algorithm, involves the use of parallel processing techniques to the FP-Growth algorithm to reduce the processing bottleneck that arises when extremely large data sets are mined sequentially. The test data for the proposed algorithm is extracted from medical images (mammograms) which is a typical example of such large data sets.

Experiments show that the PFP-growth algorithm improves on the mining efficiency of the FP-growth algorithm in segments prone to processing bottlenecks by achieving between 23.20% to 45.07% speed up, indicating a positive contribution with the use of parallel techniques. Also, processing speeds show that the PFP-growth algorithm scales well with the number of records mined. The results are a set of association rules that provide a framework for an image classifier. Classifying new images with the image classifier indicates a detection accuracy of approximately 80.36%.

List of Publications

1. S. A. Ehikioya and A. Olukunle, "Mining of Association Rules in Medical Image Data Sets", *Supplement to the Journal of Digitial Imaging of the 20th Symposium for Computer Applications in Radiology (SCAR 2003)*, Vol. 16, Boston, Massachusetts, USA, June 7-10, 2003, pp. 2-3.

2. S. A. Ehikioya and A. Olukunle, "On the Mining of Association Rules in Medical Image Data Sets", Nagib Callaos, Baoyu Zheng, and Firoz Kaderali (Editors), *The 6th World Multiconference on Systemics, Cybernatics and Informatics (SCI-2002)*: Volume V – Computer Science I, Orlando, Florida, USA, July 14 - 18, 2002, pp. 17-22.

3. A. Olukunle and S. A. Ehikioya, "A Fast Algorithm for Mining Association Rules in Medical Image Data", *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE'02)*, Winnipeg, Manitoba, Canada, May 12-15, 2002, pp. 1181-1187.

COMMITTEE SIGNATURE PAGE

This dissertation was presented

by

Williams, Adepele

It was defended on

August 19, 2003

and approved by

(Signature) _____

Thesis Supervisor

Sylvanus A. Ehikioya, Ph.D. (Computer Science)

(Signature) _____

Commitee Member

Christel Kemke, Dr. rer. nat. (Computer Science)

(Signature) _____

Commitee Member

Jose Rueda, Ph.D. (Electrical Engineering)

(Signature) _____

Chair of Oral Examination

John Bate, Ph.D. (Computer Science)

Dedication

This thesis is dedicated to the Omniscience God who is the source of all wisdom.

Acknowledgements

My heartfelt gratitude goes to my thesis supervisor Dr. S. A. Ehikioya for giving me the opportunity to work with him and providing me selfless fatherly advice and support throughout my masters degree program. I also like to express my deep appreciation to Dr. S. A. Ehikioya for the financial support I enjoyed during my program which was provided through his Natural Sciences and Engineering Research Council (NSERC) research grant.

I am exceedingly grateful to the members of my thesis examining committee, Dr. Christel Kemke, Dr. Jose Rueda and Dr. S. A. Ehikioya for agreeing to serve on the committee, despite the short notice and their busy schedules. Many thanks also to the staff, faculty and all my friends, at the Computer Science Department, University of Manitoba. Your friendship and dedication made a big difference.

To my parents and sisters, thank you for sacrifices you made. You are the best! I deeply appreciate my darling husband Adesoji, my major cheer leader. Thank you for believing in me.

I am also indebted to Nike, Tayo, Moses, Sola, Wura, Lanre, Sumbo and all members of the Winners fellowship, who made my stay in Winnipeg memorable. I am grateful to all who in one way or the other contributed towards the success of my masters degree program. How I wish could name you all. Thank you everyone!

Finally, to God with whom nothing is impossible. You keep on looking out for me. I owe it all to You.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

Data Mining, also called Knowledge Discovery in Databases (KDD), is an active research area in databases that involves the extraction of implicit, previously unknown, and potentially useful information (knowledge) from vast stores of data [40, 43]. Various data mining techniques, such as neural networks [11], clustering techniques [6], decision trees [39], bayesian classifiers [37], and k-means methods [6], have been successfully applied in domains where large amounts of data need to be analyzed in order to provide useful information for decision support. Most data mining techniques are often concerned with how to efficiently mine these large data sets within time and memory constraints. Recent research [18] also focus on data mining algorithms that can harness the parallel processing power that is available on some systems today.

Association Rule Mining (ARM) is a standard data mining technique that seeks to discover information (rules) between the attribute values in large data sets. A data set is a structured collection of data, such as records in a database. An attribute value is a piece of data that represents an attribute of a record. For example, a data set associated with a real-world Employee entity, might have attributes such as ID, name, age, and sex. A record in that data set could be represented with attribute values {*675456, Smith, 39, male*}, which represent a unique employee and correspond to the set of attributes {*ID, name, age,*

sex}.

ARM offers a simple and transparent approach to data mining that makes it appealing and thus, it has been widely applied to a variety of domains [18]. Another key feature of ARM is its combinatorial nature, which makes it useful for the discovery of patterns that appear in subsets of all attributes. A pattern is a set of attribute values that occur frequently together in records. For example, there could be a {*39, male*} pattern in a data set, implying that these two attribute values tend to occur together in records within that data set. Various ARM algorithms [1, 2, 3, 7, 17, 40, 45] aim at efficiently extracting all relevant attribute values in a database and finding interesting rules between them, based on user and problem-defined frequency thresholds. A rule, e.g., $39 \Rightarrow male$, is an implication between two (or more) attribute values, that expresses the likelihood of them occurring together in a given data set. A number of these algorithms apply heuristics that require multiple database scans and iterative rule generation methods, thus limiting the size of databases that can be efficiently mined. Most existing ARM algorithms are based on the Apriori ARM algorithm (see Subsection 2.2.2), which generates memory intensive intermediate patterns in its search for rules.

The Frequent Pattern (FP)-Growth algorithm [17] is a recent ARM algorithm that has enjoyed wide acceptance [18, 47] because it requires a maximum of two database scans and also avoids generating memory intensive intermediate patterns. The FP-Growth algorithm achieves efficiency through the use of a Frequent Pattern (FP)-tree, which is a complete and highly compact representation of the relevant attribute values in the transaction data and the patterns between them. The FP-Growth algorithm has proven to be significantly more efficient than apriori-like algorithms [18] in mining large databases that have a large number of relevant attribute values. The patterns formed between attribute values of such databases which have a large number of relevant attribute values are often long and frequent, hence the name, **Frequent Pattern**-Growth algorithm.

Medical images play a significant role in medical diagnosis. In situations that require mass image screening or an early detection of subtle and varied symptoms, computer aided diagnosis has proven to be useful in increasing the efficiency and effectiveness of medical

diagnosis [46]. Digital mammograms have been of particular interest, since they have been asserted as the most reliable source of an early detection of breast cancer [38]. Each medical image object (a segmented region of interest within a medical image) has a large number of attribute values whose pattern of occurrence can be analyzed to obtain rules to build classifier systems, which are used to distinguish between normal and abnormal mammograms.

However, an algorithm that can efficiently mine medical images must be fast, amenable to long patterns and large data sets. In addition, the algorithm must be scalable to parallel implementation to suit systems that can provide this additional processing benefit and also systems that have images distributed over a network.

This thesis proposes and implements a novel and an efficient ARM algorithm that is suitable for medical image data sets. An initial discussion of this algorithm is available in [12, 13, 32]. The proposed parallel association rule mining algorithm, the Partitioned Frequent Pattern (PFP)-Growth algorithm, is tested on a set of digital mammograms to determine its suitability for large databases with long patterns. Furthermore, we compare our implementation results with those obtained using the FP-growth algorithm. The PFP-growth algorithm performs significantly better at the task of mining medical images.

1.2 Motivation

This research is motivated by the need for an efficient ARM algorithm that is suitable for large data sets with long patterns. Han, Pei and Yin [17] have shown that the FP-growth algorithm is efficient for mining data sets with long patterns. However, for extremely large databases, the Frequent Pattern-tree structure could get large enough to constitute a memory bottleneck. The reasoning motivating this research is that if the entire database, T, is partitioned into N partitions, and an FP-tree of size T/N is built for each partition, the proposed algorithm will be scalable for much larger databases. It should also be amendable to distributed data mining.

Useful information for medical diagnosis can be extracted from the large volume of medical images generated today. A modern hospital generates more than 1 terabyte of

image data per year [16]. Large data sets that contain life-dependent information need accurate and efficient data mining techniques. However, it is also essential in the medical domain that the technique being applied is simple. This requirement is important because physicians that apply the extracted knowledge tend to accept self-explanatory techniques, which makes decision-making more transparent [31]. Association rule algorithms suit these stated requirements. However, previous ARM algorithms have been limited in the volume of images that they can efficiently mine due to their memory-intensive, intermediate rule generation stages. The FP-growth algorithm has not yet been applied to mining medical images. The PFP-growth algorithm is an extension of the FP-growth algorithm, which in addition to the compact nature of the latter, is scalable to large data sizes. This scalability feature motivates the need to apply the PFP-growth algorithm to medical image mining to meet the simplicity and efficiency needs of this domain.

1.3 Problem Definition

The problem of mining association rules can be summarized in two tasks. The first involves a search for items (attribute values) in the database whose frequency exceeds a user-defined threshold (the minimum support). Such sets are referred to as *large itemsets.* The second task involves discovering all large itemsets (patterns) whose frequency exceeds a second user-defined threshold (the minimum confidence). These patterns are interpreted to form the association rules.

Since the introduction of ARM in 1993 [2], a variety of association rule algorithms have been proposed, with the aim of performing efficiently these two major tasks involved in mining large data sets. The various association rule algorithms differ in the approach taken to search for large itemsets and discovering rules from these sets. Most of the existing association rule algorithms require an intermediate stage of processing, while searching for large itemsets.

Apriori-like algorithms, for instance, generate *candidate itemsets*, a search space for large itemsets, which grows exponentially with the number of items. These intermediate data

sets often become huge, resulting in a bottleneck in processing. Bottlenecks in processing limit the size of data and length of patterns that can be mined to the available main memory, which in most cases is insufficient for the available data [20]. In addition, the required multiple scans of the entire database constitute a bottleneck as blocks of data are transferred from disk to memory and back.

A number of ARM algorithms have been proposed to solve these problems [1, 3, 4, 40, 45]. However, since these algorithms are based on the Apriori scheme, the original database still gets expanded during the intermediate mining stage, making it infeasible for huge data sets like medical images. The FP-growth algorithm, another solution, provides a compact representation using an FP-tree and drastically reduces the input/output bottlenecks by scanning the entire database twice. In the first scan, it counts the large items. In the second scan, it builds the FP-tree. However, if the database is very large and has long patterns, such as an image database, the FP-tree could exceed the size of the main memory, resulting in a memory bottleneck.

1.4 Brief Outline of the Proposed Solution

The FP-growth algorithm provides a more compact data representation and efficient rule mining procedure than other ARM algorithms developed prior to it. However, when the database is extremely large and consists of long patterns, the tree structure the FP-growth algorithm uses for pattern representation could still exceed main memory size, resulting in a processing bottleneck.

This thesis solves the FP-tree size problem, which occurs when the database is large, by partitioning the database and building multiple smaller trees, one in each partition. The patterns obtained in each tree can be merged by a summation of counts. This implies that a larger number of images can be mined at a faster speed, since all the FP-trees are built independently of one another. This solution is called the PFP-growth algorithm.

The PFP-growth algorithm proposed in this thesis also achieves computational gains over the FP-growth algorithm by avoiding trimming and sorting of the original database

prior to the tree-building phase. Instead, a temporary array is used for the representation of each record's item that appears on the header table (ordered list of large itemsets) and is also present in the transaction. The contents of this array is used to plot the tree path for that record.

The PFP-growth algorithm is applied to a data set that contains statistical measures extracted from digital mammograms. A total of 322 digital mammograms are used, each of which is made up of 64 sub-images. Four measures, with a total of 36 possible values each, are extracted from a total of 20,421 sub-images.

The database obtained provides a typical case study of a large data set characterized by long patterns, while also providing a possible solution to the medical image-mining problem described in the previous section.

1.5 Scope of Work

The scope of work covered in this thesis is as follows:

1. Present the design of the Partitioned FP-growth ARM algorithm.
2. Describe the data preprocessing (medical image mining) stages which lead to a segmentation of the image objects.
3. Provide details on the extraction of statistical measures from image objects. These measures constitute the attributes of the image database.
4. Mine the image database with the basic FP-growth algorithm and the PFP-growth algorithm.
5. Discuss a performance analysis of the PFP-growth algorithm in comparison to the FP-growth algorithm.

1.6 Contributions

The contributions of this thesis are the design of a parallel FP-growth based ARM algorithm and its application to medical images. The compact representation of data used in this algorithm, as opposed to the Apriori-like algorithms used in other research, enables the efficient mining of large medical image data sets.

In addition, the PFP-growth algorithm uses compact tree-building heuristics, which makes the tree building stage more computationally efficient in terms of execution time. Furthermore, partitioning the image database records further increases the scalability of the mining process, and makes it possible to mine distributed and possibly heterogeneous images. For instance, if the images to be mined are in various locations and of different formats, they can be preprocessed seperately, though they must be represented with the same set of attributes in different partitions.

1.7 Organization

The organization of the rest of this thesis is as follows:

- *Chapter 2: Literature Review.* This chapter provides a discussion of related research in the application of association rules to medical images. It includes the definition of association rules, a description of key ARM algorithms, and related work in this area. Also, the image mining process is briefly discussed.

- *Chapter 3: Algorithm Specification and Implementation.* This chapter presents the proposed algorithm, using pseudo-code, and a detailed description of its phases. This chapter provides a comparison of the PFP-growth algorithm and the FP-growth based algorithms described in Chapter 2. This chapter also provides a brief description of the implementation environment.

- *Chapter 4: Experimental Results.* This chapter presents and explains the experimental results obtained. It also includes a performance analysis of the proposed algorithm

using results obtained for various data sizes and number of processing units, especially in comparison to the sequential FP-growth algorithm.

- *Chapter 5: Discussion and Conclusions-* This chapter summarizes the research findings and includes suggestions for future work to improve and enhance the PFP-growth algorithm's suitability for mining medical images and large data set in other domains.

Chapter 2

Literature Review

2.1 Background

Advances in data acquisition and processing techniques have enabled most organizations accumulate huge databases of routine business data, which can be mined and then applied for strategic and intelligent decision support. Data mining is applied in areas like financial forecast [26], census [18], consumer profiling [7], marketing strategy [7] and decision support [18], and recently in medical diagnosis [27].

Today, ARM is one of the most widely used data mining techniques [20]. Several ARM algorithms have been proposed, the majority of which require multiple passes over the entire database to be mined. Multiple passes result in poor response times and a high input/output overhead when the database is very large. In addition, the majority of ARM algorithms are based on a key algorithm, called the Apriori algorithm [20], which is characterized by a memory-intensive intermediate stage, the generation of *candidate itemsets.*

The Frequent Pattern-Growth algorithm solves the multiple pass and memory intensive stage problems of the Apriori-like algorithms because it requires only two database passes, in addition to representing patterns with a compact FP-tree. A number of variants of the basic FP-Growth algorithm have been proposed [28, 47]. This chapter discusses these variants and their main features in detail.

The motivation to mine medical images stems from the need to achieve high efficiency

and effectiveness in medical diagnosis. For images, such as digital mammograms, which are difficult to read but can be used to detect early symptoms of cancer [46], it is important to find rules that can be applied to aid disease detection. Antonie et al. [5] show that ARM algorithms can be useful in mining medical images. The research by Antonie *et al.* is limited by the use of an ARM algorithm (Apriori) whose search strategy generates a large search space, which can constitute a processing bottleneck with large image sets, considering the size of each image record and the sequential nature of the mining algorithm used. Related work on the use of ARM on medical images [46, 35] is also discussed in Section 2.4.1.

To fully present the research problem and its solution, the basic tasks of ARM, the limitations of Apriori-based algorithms, the FP-Growth algorithm and its recent extensions are reviewed. This chapter also presents the medical image preprocessing stages, using digital mammograms as a case study. This background in image preprocessing provides the basis on which the results are discussed and points out why medical images require specific algorithms that can handle their peculiarities.

2.2 Association Rule Mining

Initially, ARM was applied to basket data to identify the set of items that are most often purchased together [40]. Basket data typically consist of data relating to supermarket transactions, such as items, price, quantity and date of transaction. Therefore, most of the terminology used in ARM is derived from market basket analysis. In ARM, the records of a database are referred to as *transactions* while attribute values, which are analogous to articles in a shopping cart, are called *items*. Association rules express the different patterns that underlie a data set. The patterns association rules express do not often include the entire set of attributes. [43].

An association rule is an implication expressed as $X \Rightarrow Y$. It is generally interpreted as: "If X occurs, it is most likely that Y also will occur". X is called the *antecedent* while Y is referred to as the *consequent* of the rule. Formally, X and Y are defined as follows [2]:

Let $I = \{i_1, i_2, ...i_n\}$ be a set of *items*.

Let $T = \{t_1, t_2, ...t_n\}$ be a database containing subsets of transactions (records).

A set X is called a k-itemset (i_k) if $X \subseteq I$ and $k = |X|$.

A transaction, $t_k \in T$, supports itemsets $X \subseteq I$ and $Y \subseteq I$, if $i_k \subseteq T$ holds, where $X \neq \emptyset$, $Y \neq \emptyset$ and $X \cap Y = \emptyset$.

Often, a large number of association rules are derived in transactional databases. Only a small fraction of these are of enough interest for knowledge discovery in a given application. Therefore, it becomes necessary to define a measure of interestingness, which serves to remove uninteresting rules from the set of rules. Two measures of interestingness used in ARM are *support* and *confidence* [18].

The *support* (accuracy) of an association rule is the fraction of transactions in the database for which the rule is true. The *confidence* (validity) of a rule is the fraction of transactions in the set of transactions containing the antecedent that also contain the consequent. Support is a measure of statistical significance while confidence corresponds to a rule's strength [2]. Minimum Support (MS) and Minimum Confidence (MC) are user-defined minimum values of support and confidence, respectively, that a rule must have in order to be considered as interesting. These constraints are defined below.

A rule, $X \Rightarrow Y$, holds and is considered interesting if the support,

$$S = \frac{Number\ of\ transactions\ containing\ X \cup Y}{Total\ number\ of\ transactions} \geq MS$$

and the confidence,

$$C = \frac{Number\ of\ transaction\ which\ rule\ X \Rightarrow Y\ predicts\ correctly}{Total\ Number\ of\ transactions\ in\ the\ data\ containing\ itemset\ X} \geq MC$$

The rule, $X \Rightarrow Y$, implies that when a transaction t_k contains X, it most likely, also contains Y.

In discovering association rules, the first task involves a search for itemsets in the database whose frequency exceeds the user-defined minimum support. Such sets are referred to as *large itemsets.* Finding all the large itemsets is a non-trivial task, which involves searching a huge search space that forms a lattice [3]. The itemset lattice refers to the space

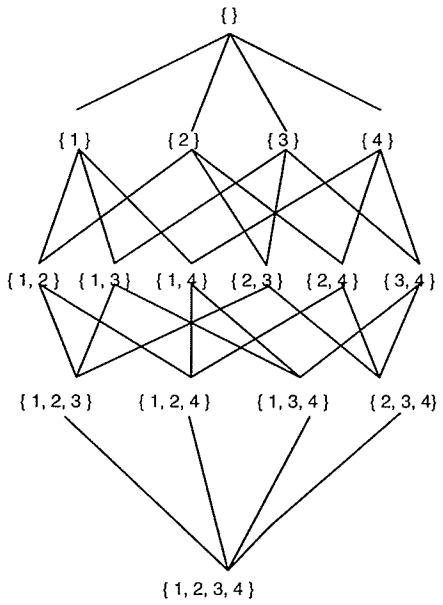

Figure 2.1: Lattice for a 4-itemset (c.f. [45])

of all itemsets that can exist in the transaction database. For instance, for a k-itemset, there can exist 2^k itemsets in the lattice [45]. This search space is illustrated in figure 2.1. The search space for itemsets grows exponentially with the number of itemsets in a database. Thus, it is impractical to search for the frequency of all subsets of I to determine whether they are large or not.

The second task, which is easier, involves discovering all rules that describe the patterns between these large itemsets that exceed a user-defined threshold (the minimum confidence). For instance, if RST is a large itemset, the confidence values for any of the rules $RS \rightarrow T$, $RT \rightarrow S$ and $ST \rightarrow R$ needs to exceed the minimum confidence before it can be considered as a valid rule. ARM algorithms differ solely by the search and counting heuristics employed to discover rules. A major emphasis in developing ARM algorithms is the use of efficient heuristics to obtain all potential large itemsets while searching the minimum possible number of small itemsets.

Counting heuristics are applied to obtain the frequency of itemsets right after a search strategy has been applied to select the potential large itemsets (candidate itemsets). The aim of counting heuristics is to ascertain that the frequency counts of potential large item-

sets exceed the minimum support using the minimum number of database passes. Two major methods exist [21]. The first involves directly counting the occurrences of candidate itemsets through the use of counters. The counter is increased whenever one of the candidate itemsets is contained in a transaction. The second approach involves counting the support of candidates indirectly, by allocating a transaction identification, TID (X_{tid}) to every transaction containing the itemset X. The support of a candidate, $C = X \cup Y$, can be obtained by intersecting the TIDs of sets X and Y, i.e., $C_{tid} = X_{tid} \cap Y_{tid}$ [21].

2.2.1 The Basic Association Rule Algorithm (AIS)

The AIS algorithm [2] is the first ARM algorithm introduced. AIS is an acronymn for the names of the developers: **A**grawal, **I**mielinski and **S**wami. This algorithm searches for large itemsets by checking all itemsets in the itemset lattice. The AIS approach is impractical for a large value of itemsets, especially when only a small fraction of the whole lattice is frequent. Furthermore, rules in AIS are limited to single item consequents, i.e., a rule of the form $X \rightarrow Y$, where Y is a single itemset. The AIS algorithm is described in [2] as shown below:

AIS Algorithm
Input = Transaction dataset and minimum support value
1) L_1 = large 1-itemsets ;
2) **for**($k = 2$; $L_{k-1} \neq \emptyset$; k++) **do begin**
3) $C_k = \emptyset$;
4) **forall** large itemsets $l_t \in T$ **do begin**
5) $L_t = subset(L_{k-1}, t)$; *//Large itemsets in t*
6) **forall** large itemsets $l_t \in L_t$ **do begin**
7) C_t = 1-extensions of l_t contained in t; *// Candidates contained in t*
8) **forall** candidates $c \in C_t$ **do**
9) **if** ($c \in C_k$) **then**
add 1 to the count of c in the corresponding entry in C_k
else
add c to C_k with a count of 1
10) **end**
11) $L_k = c \in C_k \mid c.count \geq MS$
12) **end**
Output = $\bigcup_k L_k$; //Union of all large itemsets

The AIS algorithm has a number of limitations. The most outstanding limitation is

its memory-intensive search space for large itemsets, which grows exponentially with the number of items in I. The AIS algorithm requires that the support of each of the subsets of I be computed to determine if it is a large itemset or not. Each candidate itemset X and Y needs to be checked (counted) against the transaction database to crosscheck if it indeed belongs to the set of large itemsets. Furthermore, this algorithm requires multiple passes over the entire transaction database, which is impractical for large data sets. For instance, K passes are required for a database with $K = |I|$. This causes a bottleneck when applying the AIS algorithm [1].

2.2.2 The Apriori Algorithm and Its Extensions

Various association rule algorithms have been proposed [1, 3, 4, 40, 45] after the introduction of the AIS algorithm, in a bid to solve its memory and I/O intensive problems. The foundation of most known ARM algorithms, whether sequential or parallel, is the Apriori algorithm [3, 47]. The Apriori algorithm is an improvement over the AIS algorithm. It employs the downward closure property of itemset support for pruning the lattice. This implies that it prunes (discards) itemsets whose subsets are not large (frequent) to avoid counting their support.

For example, all k-itemsets and their subsets that constitute a $(k+1)$-itemset must be frequent if the $(k+1)$-itemset will be frequent. If any sub-itemset of a $(k+1)$-itemset is not frequent the $(k+1)$-itemset can be pruned (ignored in the search) without missing any large itemset. This pruning reduces the search space. Pruning the search space greatly reduces the number of passes required. The Apriori algorithm employs a breath-first-search strategy, [10] i.e., it searches for all large k-itemsets before it begins the search for large $(k+1)$-itemsets and counts the number of occurrences of candidates using counters [21]. The Apriori algorithm is described in [3] as shown:

Apriori Algorithm
Input = Transaction data set and minimum support
1) L_1 = large 1-itemsets;
2) **for** ($k = 2; L_{k-1} \neq \emptyset; k++$) **do begin**
3) C_k = *Apriori-gen*(L_{k-1}) ; *//Generate C, the set of Candidate (potential) itemsets*
4) **forall** transactions $t \in T$ **do begin**

5) $C_t = Subset(C_k, t)$; // *Large (k-1)-itemsets contained in t*
6) **forall** candidates $c \in C_t$ **do**
7) $c.count++$;
8) **end**
9) L_k = c $\in C_k$ such that c.count $\geq$ Minimum Support
10) **end**
11) **Output** = $\bigcup_k C_k$; //Union of all sets C_k

The Apriori-gen Function

Apriori-gen(L_{k-1}) ;
To generate the set of all large k-itemsets from L_{k-1},
join L_{k-1} and L_{k-1} such that,
$c_1 = (i_i, i_2, ...i_{k-1})$ and $c_2 = (j_i, j_2....j_{k-1})$ are joined if
$i_p = j_p$ for $1 \leq p \leq k-2$;
c = $(i_i, i_2, ...i_{k-1}, j_{k-1})$; //New candidate **c** is of this form
Add **c** to candidate itemsets

The Apriori-gen function generates new potentially large (candidate) itemsets from itemsets that are already confirmed to be large. For instance, two large 3-itemsets, $c_1 = (a, b, c)$ and $c_2 = (a, b, d)$, generate a candidate set of 4-itemset, $\mathbf{c} = (a, b, c, d)$.

The Apriori algorithm is more efficient in both generating and counting candidate itemsets because it determines apriori the possible large itemsets. The number of database passes is reduced because the large itemsets identified in the previous pass are used to generate candidate itemsets for the next pass. A major drawback of the Apriori algorithm is that it can be computationally expensive when there are many large itemsets. For instance, to generate a pattern of length 100, approximately 10^{30} candidate itemsets are generated [18]. Thus, the Apriori algorithm is suitable for nominal data sets with small combinations of itemsets.

Many extensions of the Apriori algorithm exist. The common factor they share is that they generate candidate itemsets, while searching for large itemsets. The variants differ in the use of heuristics to reduce the number of candidtate itemsets and improve the counting scheme. This section briefly describes four major Apriori-based algorithms. A detailed survey of these algorithms is provided by Hipp *et al.* [20].

1. **The AprioriTID Algorithm**

The AprioriTID algorithm [3] differs from the Apriori algorithm in its representation

of each transaction with a unique transaction identifier, the **TID**. It initially uses the Apriori-gen function (see the Apriori alogrithm) to search for candidate itemsets. The occurence of a candidate itemset is represented with a TIDlist, defined as the list of the TIDs of all transactions in which the itemset occurs. The important gain in the AprioriTID algorithm lies in the fact that after the first pass, it intersects TIDs of generated candidate sets to count supports. The benefit of this approach is that scanning the entire database is avoided at each pass other than the first pass. However, in very large databases, memory bottlenecks still occur at the candidate generation stage. The AprioriHybrid algorithm [3] is yet another extension of the Apriori algorithm, proposed by the developers of the AprioriTID algorithm. The AprioriHybrid algorithm is a combination of the heuristics used in the Apriori and the AprioriTID algorithms.

2. **The Dynamic Itemset Counting (DIC) Algorithm**

   The DIC algorithm [7] provides an alternative to Apriori itemset generation (Apriori-gen function). Candidate itemsets are dynamically generated, counted and deleted as transactions are read, reducing the strict distinction between counting and generating candidates [20]. The DIC algorithm relies on the fact that for an itemset to be frequent, all of its subsets must also be frequent. In addition, $k + 1$ candidate itemsets are dynamically generated within the same pass, as soon as its k-item subset achieves minimum support. The principal gain in this algorithm is a reduction in the number of database passes required and candidate itemsets. For instance, a 3-item database which requires 3 passes of the database by the Apriori algorithm would require approximately 1.5 passes using the DIC algorithm [7]. This algorithm, however, still generates enough candidate itemsets to render it inefficient for extremely large data sets.

3. **The Eclat Algorithm**

   The Eclat algorithm [45] derives its name from its use of **E**quivalence classes to **C**luster the set of potentially maximal frequent itemsets (a superset of all candidate itemsets),

which forms a subset **LAT**tice. The subset lattice is searched using a depth-first search (DFS) algorithm [10]. The search begins with the element having the maximum support and extends to the next element in a sorted order. This approach is based on the intuition that the larger the support of an itemset, the more likely it is to be part of a larger itemset. However, the DFS implies that candidates cannot be pruned on the basis of their subset support [21].

The Eclat algorithm requires a decomposed storage structure [22] that is represented as a list of items rather than a list of transactions. Each item has a list of all the transactions containing that item. The support of itemsets is determined through the use of TID-list intersections, thus requiring only one scan of the database [20]. A notable drawback is that intersecting 1-itemset TID-lists to obtain longer itemsets can be very expensive, and thus is impractical for very large data sets.

4. **The Partition Algorithm**

The Partition algorithm [40] is a variant of the Apriori algorithm. The Partition algorithm is an attempt to resolve the problem of finding large candidate itemsets in sequential Apriori-like algorithms by partitioning the transaction database and processing the partitions in parallel. The Partition algorithm splits the transaction database into several non-overlapping parts, which can be stored in memory. These partitions are treated independently, making the Partition algorithm suitable for parallelization. The partition size is chosen in such a way that all the TID-lists of each partition fit into main memory. This reduces I/O and memory overhead. The partition algorithm requires two scans. In the first pass, it generates the set of all potential large itemsets (any itemset locally frequent in a partition) using a breadth-first search [10] strategy. The second pass checks that itemsets that are locally frequent are also globally frequent. The partition technique uses set intersections (i.e., tid-lists)to determine rules.

A number of variants of the Partition algorithm exist. They differ in the approach used to process the data in parallel [4]. Their major limitation lies in the use of the

Apriori-gen function which is used to mine partitions indepedently, and candidate itemsets may still grow to an unreasonable size.

2.2.3 The Frequent Pattern Growth Algorithm

The Frequent Pattern (FP)-growth algorithm [17] is a new ARM technique that eliminates the need for intermediate candidate itemsets by representing the entire database and associative patterns with a compact tree-like structure, called the FP-tree. An FP-tree consists of nodes, which represent the frequent items in the transaction database and their corresponding frequencies. Two nodes are linked by an edge, which shows the patterns between them, in a descending order of frequency. Thus, a path in an FP-tree describes the frequency at which items in that path occur together in the transaction database, and the pattern in which they do.

The FP-Growth algorithm has three phases. In the first phase, the entire transaction database is scanned and the frequencies of all items are obtained. All items with frequencies below a user-defined minimum frequency (the minimum support) are pruned. Next, a header table is created. A header table consists of all items with frequencies above the minimum support, arranged in descending order of frequency. In the second phase, the transaction database is scanned again to generate the FP-tree, based on the position of the items on the header table. The FP-tree is a highly condensed representation of the large items in the transaction data. Items that are present in the header table (frequent items) are mapped to a tree-like structure, based on their position in the table. The frequency of an item in a path is increased by one each time it is encountered in the database. The FP-tree is used to derive the support values of all frequent itemsets in their respective patterns.

The third phase involves concatenating paths in the tree (patterns) and mining them to obtain association rules. All the possible frequent patterns that contain a frequent item, F, can be obtained by concatenating the prefix paths of the tree (a list of items that occur higher up in a path before the item of interest), starting from the nodes in which F occurs. This process is referred to as constructing the conditional pattern base. Nodes in the pattern, which are less than the minimum support, are excluded. The pattern base is used

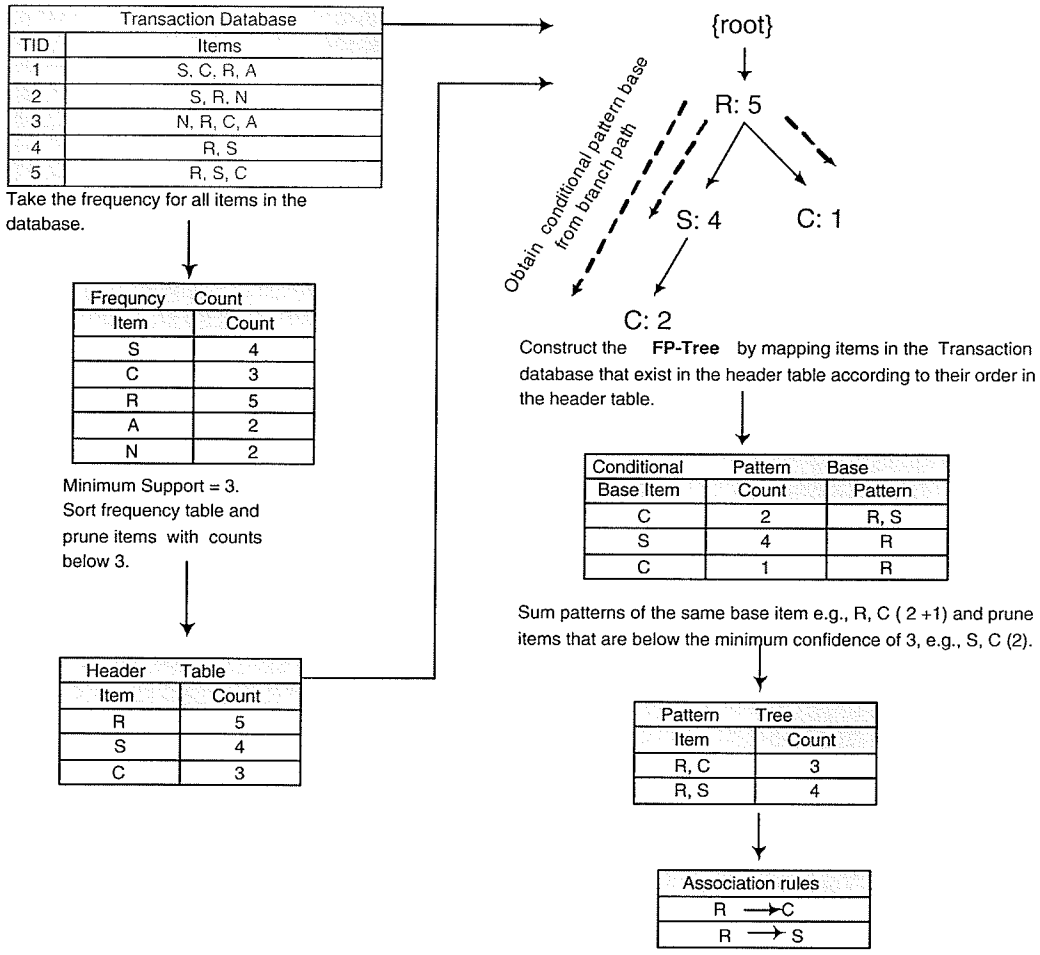

Figure 2.2: The FP-Growth Algorithm : An Example

to construct conditional FP-trees for each item, which can be mined recursively to obtain all possible patterns that include this item. Association rules are derived from the item patterns.

Figure 2.2 illustrates how the FP-Growth algorithm works using a 5-item database which consists of 5 transactions. The FP-Growth algorithm is described in [17] as shown below:

The FP-Growth Algorithm:
FP-Tree construction

L = Large 1-itemsets in T ; // *Scan Database to find large 1-itemsets*
Sort L in support descending order
Create root of tree Tr; // *This is a null node*
For each transaction, t in T **do** ; //*t represents a transaction*
Select and sort $[p \mid P]$; // *a list of the frequent items in transaction, according to // the order of L (p is the first item and P is the remaining list).*
call insert_tree($[p \mid P]$, Tr) //*begin to insert each list at root node, Tr*

Procedure insert tree (list P, node Tr)
if P has an item, p, represented as child node N, such that N. $item = p.item$
then increment N's count by 1
else create new node N, count N = I;
let node N be linked to Tr;
if $P - p \neq \emptyset$ **then**
call insert tree ($P - p$,N) // *call this function recursively till list is empty*

Procedure FP-Growth(Tree, α)
if Tree contains a single path P **then**
for each combination (denoted as β) of the nodes in the path P
generate pattern $\beta \cup \alpha$ with support $\geq$ minimum support of nodes in β ;
endfor
else for each a_i in the header of tree
{
generate pattern $\beta = a_i \cup \alpha$ with support $a_i.support$;
construct β conditional pattern base and then $\beta's$ conditional FP-tree, $Tree_\beta$;
if $Tree_\beta \neq \emptyset$ **then**
call FP-Growth ($Tree_\beta, \beta$);
}

This algorithm is suitable for large databases containing long patterns because the size of the FP-tree is usually highly compact and much smaller than the original database. However, for extremely large data sets with long patterns, the pattern tree could still exceed main memory size, resulting in a processing bottleneck. The FP-Growth algorithm is useful in domains with low support and a high number of transactions. By avoiding candidate itemset generation and the counting of candidates that may turn out to be small, the FP-Growth algorithm eliminates repeated database scans.

2.3 The Image Mining Process

2.3.1 Definition of Terms

This section defines some of the keywords used in image processing to provide a background for their use in subsequent subsections.

- *Pixel:* The word pixel is formed from a combination of the words "**pic**ture" and "element". A pixel represents the smallest unit/element of a digitized image.

- *Spatial Resolution:* The spatial resolution of an image is the size of the square region

(window) of conventional image film that each picture element (pixel) represents. It is measured in microns (10^{-6}metres) per pixel. For instance, a digitized image with a spatial resolution of 50 microns per pixel is considered a finer representation than a 200 micron per pixel image.

- *Optical Intensity and Gray Level Resolution:* The optical intensity of a picture element (pixel) is a measure of the ratio of the known intensity of light (the photon flux density) passed through the pixel location to the intensity of the light beam that gets transmitted off. The grey level resolution of a pixel is directly proportional to its optical density and it serves as a basis for the values of most pixel level measures such as variance, skewness and kurtosis.

2.3.2 A Background on Images

A 2-dimensional digital image f(x,y), is a group of gray level intensities (also called amplitudes) that occupy a space $\{(x, y) \mid x = 1..., m; y = 1, ..., n\}$. Each gray level intensity in each location represents a picture element (*pixel*). However, an image region can also be described by measures representing the intensities and locations of the group of pixels that it contains. The group description of pixels stems out of the need to reduce dimensionality. For instance, a 256 x 256 pixel image would need a higher level of representation in order to aid its classification [29].

Texture, the main descriptor for digital images, can be defined by the local statistics of pixel grey levels. Texture can be defined as a measure of the range of gray level intensities and the frequency and location of their occurrence in an image [9]. Textual measures can be either statistical or structural (spatial) in nature [19]. Statistical (first order) measures, also called Gray Level Histogram Moments, include averages, standard deviation and high order statistics of intensity [9]. Examples of high order statistics are variance, skewness and kurtosis. Statistical measures describe the pixel values that comprise an image region.

Structural (second order) descriptors, also called Spatial Gray-Level Dependence Matrices, show the spatial dependencies or interaction between intensity values of groups of pixels. Example of structural descriptors include angular second moment, correlation, entropy, etc.

When ranges of image descriptors are distinct enough, they are mapped into features [19]. For instance, a large value of a measure could indicate a certain type of texture. Such a measure is referred to as a **feature**. Examples of textural features include fineness, coarseness, smoothness and granulation [19]. These nouns represent some gray level descriptor measure and spatial interactions between them. However, there are no standard values for each of these features. They are often relative within a set of images, because they vary with the resolution and scanning technique used to produce each set of images. The purpose of image mining is another deciding factor for the use of features. For instance, in medical applications, feature extraction might be geared towards solving a mass screening problem. Then, feature definitions will require less detail and features will be easier to extract than features required for diagnosis or treatment planning [29].

2.3.3 Mining Digital Mammograms

A massive amount of medical images are produced and archived by hospitals everyday [35]. Kalman *et al.* [25] and Woods [44] argue that applying data mining techniques to these images could improve the accuracy and efficiency of medical diagnosis, especially in mass screening scenarios where the possible presence of disease symptoms need to be found for further investigation [46].

Digital mammograms are two-dimensional images that show the structure of the breast [25]. Digital mammograms are the most reliable method for the early detection of breast cancer, which is a necessary precursor for a high survival rate [5]. The high occurence of breast cancer cases among women makes the mining of digital mammograms a worthwhile research problem. A trained radiologist can reliably identify suspicious cancer areas, although with a false positive (abnormal images wrongly labeled as normal) rate of 10 to 31% [25]. The issue of specificity (i.e., fewer missed cases) is important since the chances (rate) of surviving breast cancer is strongly dependent on its early detection [46]. The accuracy of detecting breast cancer symptoms increases with a second read by a trained radiologist. In situations where there is a huge number of images to be read, computer-aided methods can be used to complement the radiologist in selecting suspicious images which can later

be investigated in detail [29]. A comprehensive study on the benefits of computer-aided detection of abnormalities in mammograms can be found in [42].

Malignant (cancerous) breast tumors, which is the major manifestation of breast cancer, can show in the form of microcalcification (deposits of calcium), spiculated masses, circumscribed masses, ill-defined masses, asymmetry and architectural distortion lesions [46]. A microcalcification is an accumulated deposit of calcium between 0.1 to 0.5 millimeters in diameter, and it is often identified by radiologist as a spot of high optical intensity (see Subsection 2.3.1). When about 3 to 5 microcalcifications exist within a square centimeter region, they are collectively referred to as a cluster of microcalcifications. A detailed study is available in [44].

A breast region with any of the major symptoms of malignant breast tumors (i.e., locally invasive and destructive growth) is said to be **malign**. Radiologists look for different sets of clues such as a distinct circular border within the breast tissue, a change in tissue density and the star shape of a spiculated lesion when detecting any of these symptoms [44]. Likewise, in computer-aided diagnosis, the mining process differs slightly (at the feature extraction stage) according to the nature of the problem to be solved. Images that do not contain any of these symptoms are classified as **normal** images. Images with indications of abnormality considered as mild and harmless growth, though having the character of an illness, are classified as **benign** [38]. This section discusses briefly the detailed processes involved in mining digital mammograms for the purposes of distinguishing between the image classes: normal, malign and benign.

All processes that precede the stage where actual image features (see Section 2.3.2) are mined are collectively referred to as the preprocessing stage. The preprocessing stages can be roughly classified into the data reduction, data integration, data enhancement, and feature extraction stages. Techniques used in each of these stages sometimes overlap.

2.3.4 Data Reduction

Reduction techniques aim at deriving a smaller data set size, while maintaining the integrity of the original data. Reduction is done at the initial preprocessing stage, using segmentation

techniques, which identify and extract regions of interest (ROIs). In digital mammograms, reduction is achieved using a cropping operation [44] which clips away large amounts of background pixels. Background pixels often have the lowest gray level intensity values in an image [33] and can constitute as much as 50% of the image. Cropping involves setting a gray level threshold at which pixels with lower level values are set to a value of 0, all other pixels are set to 1. Areas with a gray level of 0 are then clipped off. The benefit of this operation is a significant reduction of the noise level, storage and processing requirements for image processing.

2.3.5 Data Normalization

Data from various sources are integrated using normalization techniques, which transform the different image formats to a common format [44]. Images that are of varied ranges in size and gray levels must be scaled to the same range. Transformation standards are often determined by the nature of the image application. Microcalcifications, for instance, are often as minute as 100 microns or less. An image that is digitized at 200 microns per pixel will need to be normalized to a finer resolution, e.g., 50 microns per pixel, if microcalicifications will be accurately detected.

2.3.6 Data Enhancement

Data cleaning techniques aim at further reducing noise in the data. ROIs are enhanced so that clearly distinguished measures can be extracted. Data enhancement techniques are often geared towards specific applications. A widely used enhancement technique is histogram equalization [24], which increases the range of gray level values, so the contrast between various parts of an ROI is further enhanced.

2.3.7 Feature Extraction

After medical images have been cleaned and enhanced, the image data is further reduced using feature extraction techniques. Feature extraction techniques aim at selecting only features that are of interest while removing attributes that are not interesting for the purpose

of the image mining experiment. There are numerous statistical measures that can be used to characterize images. Selecting interesting image measures reduces the order of complexity of processing and increases the accuracy of mining. Interesting measures are used to generate image features. Image features are then analyzed to generate relevant rules for the nature of the associations desired, based on domain knowledge. This stage increases the overall efficiency of the system.

Enhanced images are segmented into sub-images that represent smaller objects/regions. Sub-images are used so that each ROI is as internally homogenous as possible and have minimal variation of statistical measures. Two segmentation approaches are applicable in the processing of mammograms. One approach is to equally divide the segmented image into equal quarters, and further divide each quarter into quarters, resulting in 16 sub-images per segmented image [5, 46]. Another approach involves spot segmentation which selects high intensity pixels and groups together neighbouring selected pixels which exceed a predefined area size to form a segmented region of suspicion [44].

Statistical measures are extracted from (pixels that make up) these sub-images and averages of these values represent the statistical measure for that object. Additionally, numerical attributes are transformed into discrete intervals which can be represented in a binary transaction database as present or absent. The basic properties of a pixel are its optical intensity, α , (see Section 2.3.1), and its position (x, y), measured by its location in the x and y direction of a 2-dimensional plane. All other measures are derived from these basic properties. The basic descriptive statistics of 2-dimensional images that have proven useful in tumor detection, collectively referred to as **Gray Level Histogram Moments** [5, 9, 25, 29, 46] are defined below:

$$\textit{Mean:} \qquad \mu = \sum_{k=1}^{N} f_k p_f(f_k) \tag{2.1}$$

$$\textit{Variance:} \qquad \sigma^2 = \sum_{k=1}^{N} (f_k - \mu)^2 p_f(f_k) \tag{2.2}$$

$$\textit{Skewness:} \qquad \mu_3 = \frac{1}{\sigma^3} \sum_{k=1}^{N} (f_k - \mu)^3 p_f(f_k) \tag{2.3}$$

$$\textit{Kurtosis:} \qquad \mu_4 = \frac{1}{4}\sum_{k=1}^{N}(f_k - \mu)^4 p_f(f_k) - 3 \tag{2.4}$$

where N is the number of gray levels in the mammogram, f_k is the kth gray-level.
n is the total number of pixels in the segmented region, p_k is the number of pixels with a gray level value f_k. The value $p_f(f_k)$ is defined as $\frac{p_k}{N}$.

The mean, μ, of an image region is an average of the gray level intensities of pixels in a region. The mean intensity of an image region is used in conjunction with other measures as a base value for characterizing the distribution of gray level intensities within a region.

The variance, σ^2, is a measure of the uniformity of optical intensity values of pixels that make up an image region. An image histogram is a plot of the frequency of the occurence of each optical intensity against that intensity value. Typically medical images, represented with 8 bits per pixel have an intensity of range of 0-255. A high-contrast image has a wide histogram indicating widely varying intensity values.

The skewness is a measure of the departure from symmetry about the mean gray level. The mean gray level is plotted with a value of 0. Positive skewness describe an asymmetric distribution with more positives values while negative skewness describe an asymmetric distribution tending towards more negative values.

The kurtosis is a measure of the relative peakness or flatness of the intensity distribution of an image, with respective to a normal distribution (bell curve). Positive kurtosis describes a relatively peaked distribution. Negative kurtosis describes a relatively flat distribution. The skewness and kurtosis of a region describe the shape of a region's histogram, and so provide a measure of its constrast [29].

2.4 Related Work

To give a complete description of research related to this thesis, this section approaches the discussion on related work from two directions. First of all, we give a description of image mining research in which ARM algorithms are used. Secondly, we describe research efforts to improve on the performance of the FP-Growth algorithm in general.

2.4.1 Image Mining Research

Ordonez and Omiecinski [33] prove the feasibility of the use of ARM on image data sets using simple geometrical shapes. The data mining objective is to obtain rules for the presence of particular objects in 2-dimensional colour images containing multiple objects. In addition, there is an emphasis on mining image content without the use of auxiliary domain knowledge. This research applied the Partition algorithm [40]. Recall the Partition algorithm is a fast (and parallel) implementation of the Apriori algorithm that is more efficient than sequential Apriori-like algorithms because of its parallel generation of candidate itemsets. Each image is represented as a collection of blobs.

A blob is a 2-dimensional ellipse representing a relatively homogenous region of an image with respect to colour and texture. In this research, image preprocessing consists of segmentation of images into blobs, object identification and record creation, creation of auxiliary (sub) images objects, and algorithm application. Measures extracted from each sub-image (image attributes) are treated as numerical values. Examples of numerical values are continuous numbers such as 245.56, 34.89 and 98.43. The numerical values are partitioned into intervals, which are then indexed. These indexes are used to generate rules. Results indicate that image mining is feasible using the ARM approach and simple rules can be obtained from a few simple objects. Ordonez and Omiecinski [33] use an Apriori-like algorithm on an image database, which is initially partitioned. However, candidate items sets are still generated in each partition. The generation of candidate itemsets limits the mining efficiency that is obtained. The authors also observed that better results could be obtained by occasional human intervention and applying domain knowledge.

Perrizo *et al.* [36] use ARM to mine satellite and remotely-sensed images, using the P-ARM (Association Rule Mining using P-Trees) algorithm. The P-ARM algorithm is an Apriori-like algorithm because it generates candidate itemsets in its search for large itemsets. However, it involves a preliminary stage in which a tree-like structure, called the P-tree (Peano Count Tree), is used to represent each image bit by bit. The P-tree differs from the FP-tree because each P-tree holds a bit value of an attribute in an image. Therefore, an 8-bit pixel format will have 8 P-trees that can be merged using the logical

AND operation to obtain the support for that attribute of the image. The P-tree structure avoids subsequent scans of the image database after an initial scan.

In both [33] and [36], the size of candidate itemsets that are generated limits the number of image data sets that these algorithms can efficiently mine. Our approach differs significantly from these techniques because it avoids candidate generation. It is particularly important that we do not generate candidate itemsets because medical images typically have an extremely large number of (attribute) values that can characterize them, and the candidate itemsets they generate limit the number of images that can be mined and the efficiency of the mining process.

The first attempt to apply ARM to medical images (digital mammograms) in [5] proves its feasibility. The same data set is mined using both the Apriori association rule algorithm and artificial neural networks. Generating association rules is much faster than training a neural network [5, 9]. Therefore, in addition to its simplicity, using ARM is more efficient, compact and consistent for mining medical images.

Antonie, Zaiane and Coman [5] focus on constructing an image classifier (normal, benign and malign)from association rules generated. An accuracy of 69.11% was achieved using the Apriori ARM technique. Four statistical measures; mean, variance, variance and kurtosis (see Section 2.3.7) are extracted from each image region obtained by sub-dividing each segmented breast image into 16 parts. The data consists of 322 digital mammograms taken from the Mammographic Image Analysis Society (MIAS) [30]. The MIAS collection has gradually become a standard for use in mammography mining research [5], and thus has provided a basis for comparison of results. There are currently 57 copies of the database in use in 11 countries [30]. The MIAS database consists of 208 normal images, 63 benign and 51 malign images, totaling 322 images.

An extension of the research in [5] is available in [46]. Here, an accuracy of 81.2% is achieved in classification, which is a result of improved image pre-processing techniques. Specifically, measures of only abnormal quadrants were extracted in abnormal images and supporting information about the images, such as breast position and tissue type, are excluded from each transaction since they proved to mislead the classification.

We note that though the research in [46] is limited to first order statistical images measures (extracted from 16 sub-divisions of each image), which fall short of the characteristic use of first and second order statistics for standard medical image representation [9, 25, 44], experiments in [46] still achieved a significantly high classification accuracy of 81.2% in comparison to experiments done in [9] where a classification accuracy of 78.15% is achieved with the full set of 14 measures. Furthermore, Christoyianni in [9] obtained similar detection accuracy values when first order statistics and second order statistics (referred to as spartial gray level dependance matrix) measures are extracted from the same set of images.

In both [5, 46], the huge size of candidate itemsets that is generated using the Apriori algorithm limits its scalability. Thus, the Apriori algorithm and its extensions may not be efficient or even feasible for larger medical image databases, which are characteristic of most hospitals today. To investigate the use of ARM in medical images further, an algorithm that can handle large data sets efficiently is required. The FP-Growth algorithm has not yet been applied to medical image mining.

2.4.2 Frequent Pattern-Growth Mining Research

Some variations of the FP-Growth algorithm, which aim at improving its performance, or specializing its application on a specific type of data set, have been proposed. We provide an overview of these variations of the FP-Growth algorithm in order to make a distinction from the partitioned FP-Growth algorithm proposed in this thesis.

Zaiane *et al.* [47] propose a fast parallel association rule-mining algorithm without candidacy generation called the **M**ultiple **L**ocal **F**requent **P**attern **T**ree (MLFPT) algorithm. This algorithm is implemented on a 64 processor SGI 2400 Origin machine using 50 million market basket data. Each record has at least 12 items. Similar to the Partitioned Frequent Pattern (PFP)-Growth algorithm proposed in this thesis, the transaction database is also horizontally partitioned into approximately equal partitions amongst the processing units. An FP-tree is constructed for each partition. However, the PFP-Growth algorithm eliminates the process of pruning non-frequent items in the transaction database and does not sort the transactions, according to the ordering of the header table, before constructing the

trees. Eliminating these stages provides an advantage of the PFP-Growth algorithm over the MLFPT algorithm proposed by Zaiane *et al.* [47], without any loss in efficiency.

Another significant difference between the MLFPT algorithm and the PFP-Growth algorithm lies in the mining stage. In the MLFPT algorithm large items (whose frequency exceed the minimum support) are equally allotted processors. For instance, an item, A, represented as node A in the various trees could be assigned to processing unit 1, which recursively mines its conditional pattern base regardless of which processing unit the item path was constructed in. The PFP-Growth algorithm mines each tree within the processing unit that creates it, making the various trees independent of one another. This is an important gain when privacy-related concerns are considered in a distributed network, and also results in a significant reduction in communication between processing units.

Li *et al.* [28] propose a sequential method, **C**lassification based on **M**ultiple class **A**ssociation **R**ules (CMAR), that extends the FP-Growth algorithm by using a CR-tree. A CR-tree represents the frequent items and their patterns, also represents the sharing of items between rules. The CMAR algorithm builds a classifier whose rules are selected based on the *weighted* χ^2 correlation between them, confidence and database coverage in order to avoid over fitting and bias in small data sets. It should be noted, however, that tree-building is achieved with the FP-Growth tree, and CR-tree is only applied to efficiently store and retrieve mined association rules. The association rules obtained provide a framework that is used to build an image classifier. Experiments show that CMAR based classifier achieves a higher prediction accuracy when compared to classifiers based on two standard classification algorithms, the C4.5 and the CBA (Classification Based Associations). Li *et al.* [28] performed these experiments on a 600MHz Pentium PC with 128M of main memory using 26 data sets from the University of California, Irvine (UCI) machine learning database repository.

CMAR represents the rules obtained in a compact form, for compact storage and retrieval, especially in data sets that generate a large number of rules. Thus, the CMAR approach differs from the PFP-Growth approach in two areas: its sequential nature, and its objective, which is to extend the FP-Growth algorithm for compact rule representation.

Chapter 3

Algorithm Specification and Implementation

This chapter focuses on the methods and techniques used in this thesis. This chapter includes a detailed description of the PFP-Growth algorithm using a variety of techniques, such as pseudo-code, UML class diagrams, activity flowchart and a step-wise illustrative example. This chapter also provides a flavour of the implementation environment including the hardware and software requirements and a description of the image mining system flow, using a collaboration diagram. In addition, this chapter presents a description of the mammography image collection and the pre-processing techniques applied to them. It concludes with a detailed description of the data mining and algorithm evaluation techniques applied.

3.1 The Partitioned Frequent Pattern-Growth Algorithm

3.1.1 Description

The PFP-Growth algorithm consists of three phases. In the first phase, the entire database is horizontally partitioned into n approximately equal parts, where $n + 1$ is the number of processing units. One of the processing units is designated as the master. All others (n of them) are referred to as slaves. Each processing unit does a frequency count of all items

in its partition of the database. Local counting of item frequency is especially desirable in cases where the database to be mined is distributed in nature. Each processing unit sends the frequency count of each item in its partition to the master processing unit. The master processing unit then sums the local counts for each item to calculate the global count of each item. The master processing unit constructs a header table consisting of items whose global count is at least equal to the user-specified minimum support (see Section 2.2) in descending order of their global frequencies. This phase ends with the master processing unit sending a copy of the header table to each of the n slave processing units.

In the second phase, each processing unit builds a local FP-tree using the global header table created in phase one, similar to the tree building algorithm used in the FP-tree algorithm. However, the entire database is not pruned and sorted in order of the FP-tree, such as was done in [17]. The partitioned FP-Growth algorithm avoids the database pruning and sorting stage through the use of a temporary header-sized structure that is used to store the current transaction to be included in the tree.

Each slave processing unit generates its conditional pattern bases from its local FP-tree using similar techniques to those used in the FP-Growth algorithm. All pattern bases belonging to each tree are generated by the same processing unit that built the tree. This is a major deviation from the MLFPT algorithm. All the pattern bases from all trees are then sent to the master processing unit. The master processing unit merges all patterns that are exactly alike (i.e., have the same itemset) by their summing counts.

The third and final stage of mining involves obtaining conditional FP-trees from the conditional pattern bases. For each base item (item at the bottom of the pattern), X, the counts of all items that make up all pattern bases in which X is a base item are summed. For an item Q, whose count is below the minimum confidence (i.e., it does not participate in enough rules), item Q is pruned from the conditional pattern tree for X. All pattern trees are recursively mined to generate a set of association rules. Figure 3.1 is an activity flowchart description of the PFP-Growth algorithm while Figure 3.2 provides an illustrative example using the algorithm. The pseudo-code of the algorithm follows.

The Partitioned Frequent Pattern (FP)-Growth Algorithm

// The first phase: (a) Obtain frequency counts

1) Divide Transaction Database into n partitions for n number of processing units e.g. threads
2) **Read** Min supp // User-defined minimum support for large item sets.
3) **for** $n = 1$ to N **do begin** //Where N is the total number of transactions
4) **Procedure Item count** ; //count 1-item sets in database
5) **for** $j = 1$ to X ; // Where X is the total number of items
6) **if** $(I(ij) == 1)$ // i.e, is present in this binary database
7) $S_x^G + +$; // S_x^G = count or global support for item X
8) **endif**
9) **endfor**
10) **endfor**
11) **Output:** Frequency of all items in the local partition, S_x^L = local support for item X

//The first phase: (b) Create the header table in master processing unit
12) **for** $i = 1$ to n **do begin** // Where n is the total of slave processing units
13) $(S_x^G = S_x^G + S_x^L)$ // sum up local count for each item X as global count S_x^G
14) **endfor**
15) **for** each item x, **if** $(S_x^G < Min.Supp.)$ **then**
16) $Itemset = Itemset - x$ // Prune all items that are not large
17) **sort** Itemset //in descending order of S_x^G
18) HeaderTable = Itemset // HeaderTable now contains large items, i.e., $S_x^G \geq Minsupp$

//The second phase: Construct FP-trees
19) **Concurrently process** $i = 1$ to n **do begin** // Where n is the number of slave processing units
20) Construct FP-$tree_i$ in each Partition i
21) Create root of FP-$tree_i$ (R_i)
22) $R_i = null$;
23) **for** j = 1 to T **do begin** // Where T is the number of transactions in partition i
24) **for all** x: x$\in j \mid x \in HeaderTable \Rightarrow$
25) $HeadTemp[n] \leftarrow x$ //temporary header-sized array
26) **for** $k = 0$ to p //where p+1 is the number of items on the header table
27) $currentnode = R_i$;
28) **if** $(HeadTemp[k] \neq 0)$
29) **if**(current node has child headTemp[k])
30) $currentnode.child[k].count + +$;
31) **else** create current node.child[k] ; current $node.child[k].count = 1$;
32) **endif**;
33) $currentnode = currentnode.child[k]$;
34) **endif**
35) **endfor**
36) endfor

The third phase: Mine FP-trees to obtain FP-conditional pattern bases. Send tree paths to master processing unit.

Concurrently process:

//Input: FP-$tree_i$ at each Partition i, β_i^G
37) **for** $k = 1$ to K //Where K is the number of leaf nodes in FP-$tree_i$

```
//Mine the global conditional FP-tree,β_i^G in each partition i
38)      basenode = node_k ; //basenode stores the leaf node of every path
39)         currentnode = basenode ;
40)       while (current node ≠ R_i ) //stop when you get to the root
41)          Store pattern : currentnodeand(basenode.count) ;
42)          currentnode = currentnode.parent
43)       endwhile
44)       basenode = basenode.parent;
45)    endfor
46)    send all pattern bases to master processing unit.
47)  end concurrency
48)  Add up all patterns with same basenode k, by adding counts of same items.
49)  Remove any item x, with node_x.count ≤ Mincon − 1
50)  Write rules between 2 or more items in pattern P, with lower items implying upper items
     in the path.
51)  end
```

3.1.2 Order of Magnitude Analysis

According to [8], the order of an algorithm A is $f(n)$ if there exist constants c and n_0 such that the algorithm A will take no more than $c * f(n)$ units of time when solving a problem of size $n \geq n_0$. The order of algorithm A is denoted as $O(f(n))$.

The PFP-Growth algorithm basically consists of $2 * n$ read operations, $2 * n$ traversals of a linked list (tree-building and mining) and $2 * n/p$ sum operations (partition merging) for a data set of n records and p processing units. Observing that the read and sum operations are of the order n, the task of analyzing the order of the PFP-Growth algorithm is simplified to the analysis of the order of transversing a linked list. A linked list of n items is transversed as follows [8]:

```
Node current = head;                    //one assignment
while (current ≠ null){                 //n +1 comparisons
Current.getItem());                     //n writes
current.setNext(current.getNext());     //n assignments
}//endwhile
```

The total number of operations is evaluated as $n + 2$ assignments, $n + 1$ comparisons and n write operations. Therefore the execution time of the transversal of a linked list is said to be proportional to n. Based on the above analysis the order of the FP-Growth is

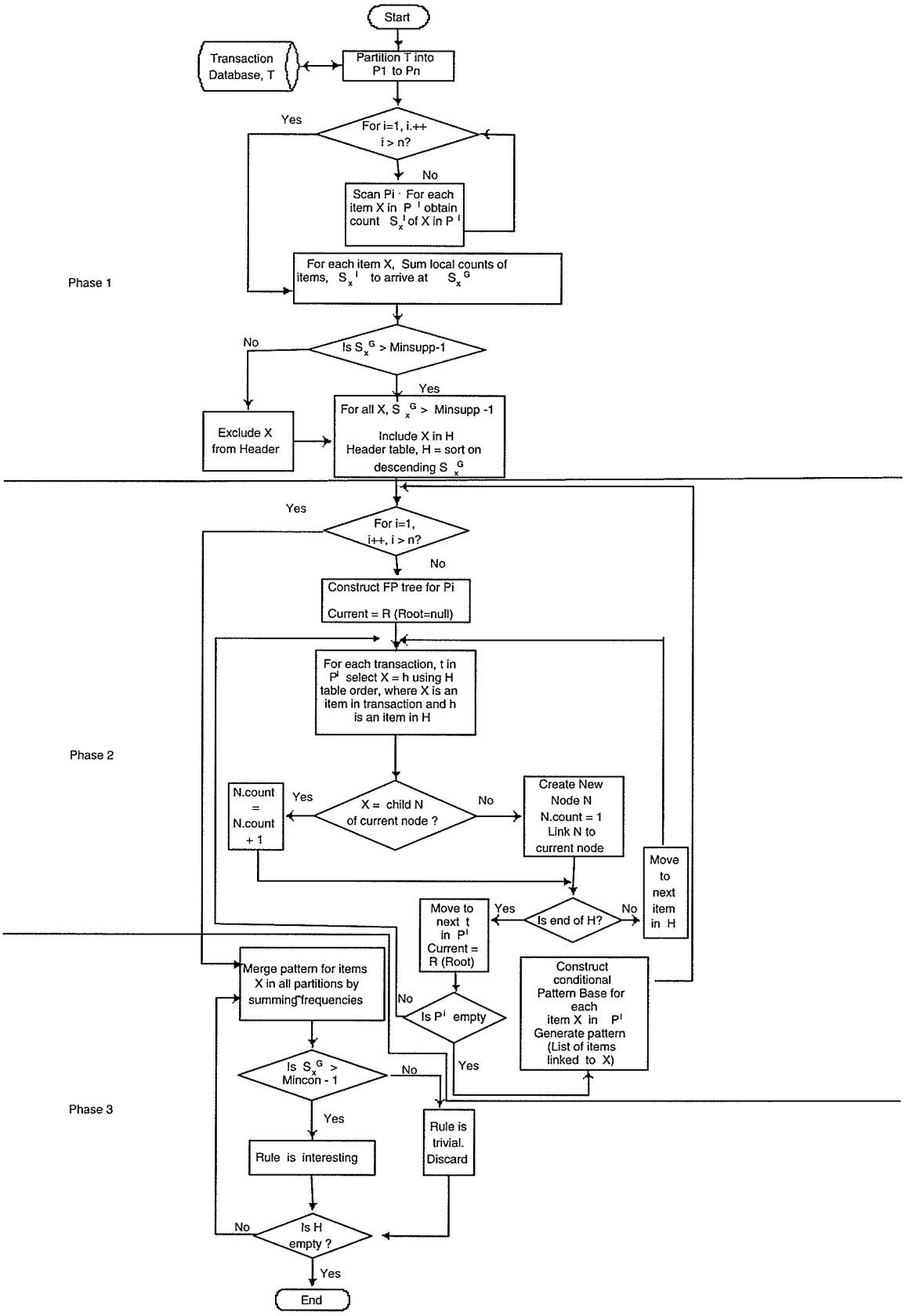

Figure 3.1: The Partitioned Frequent Pattern-Growth Algorithm: Activity Flowchart

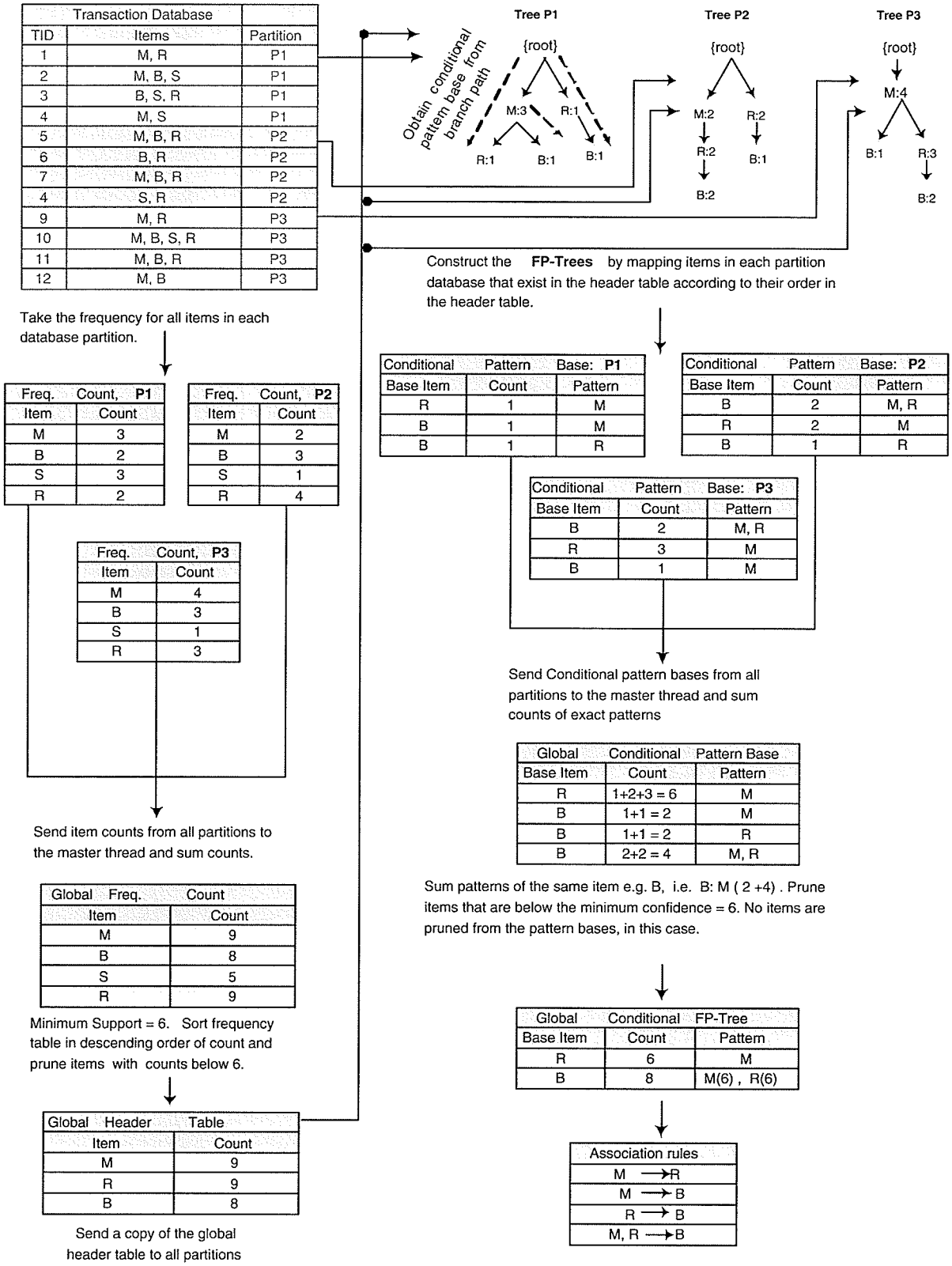

Figure 3.2: The Partitioned Frequent Pattern-Growth Algorithm: An Example

$O(f(n))$, where n is the number of records in the data set to be mined. However, execution time will be proportional the number of items that make up the header table (which the longest possible link list), as shown by the analysis of the transversal of a linked list.

3.2 Algorithm Implementation

The PFP-Growth algorithm was implemented in Java (version 1.4.1 API specification) on a Microsoft Windows 2000 Professional platform. The classes used and their interactions are shown in a UML class diagram in Figure 3.3. The images were pre-processed using the public domain NIH image program, *imagej* [24].

Approximately 65MB of memory is required per image for the major pre-processing techniques, at an average speed of 8 million pixels per second [24], though approximately 322 MB is required to store the digital images. The image database is mined on a 350 MB memory network drive. Figure 3.4 shows the entire image mining system using a UML collaboration diagram. Detailed descriptions of the various stages are given in subsequent sections of this chapter.

3.3 Image Aquisition

The Mammographic Image Analysis Society (MIAS) focuses on the understanding of mammograms (X-rays of the breast) [30]. MIAS maintains a database of digital mammograms consisting of images from the UK National Breast Screening Programme.

The MIAS collection has been digitized to 200 micron pixel edge with a Joyce-Loebl scanning microdensitometer. This device provides an optical density range of 0 to 3.2. All images are padded/clipped at 1024 x 1024 pixels. Each pixel is represented with an 8-bit binary word.

Each image includes a radiologist's 'truth'-markings, which specify the locations of abnormalities if present. Truth-markings are provided in form of a tuple of x, y and r values, where x and y are coordinate values of the centre of a circle of radius r, whose area completely includes the region of abnormal tissue. Prior to the preprocessing stage, each

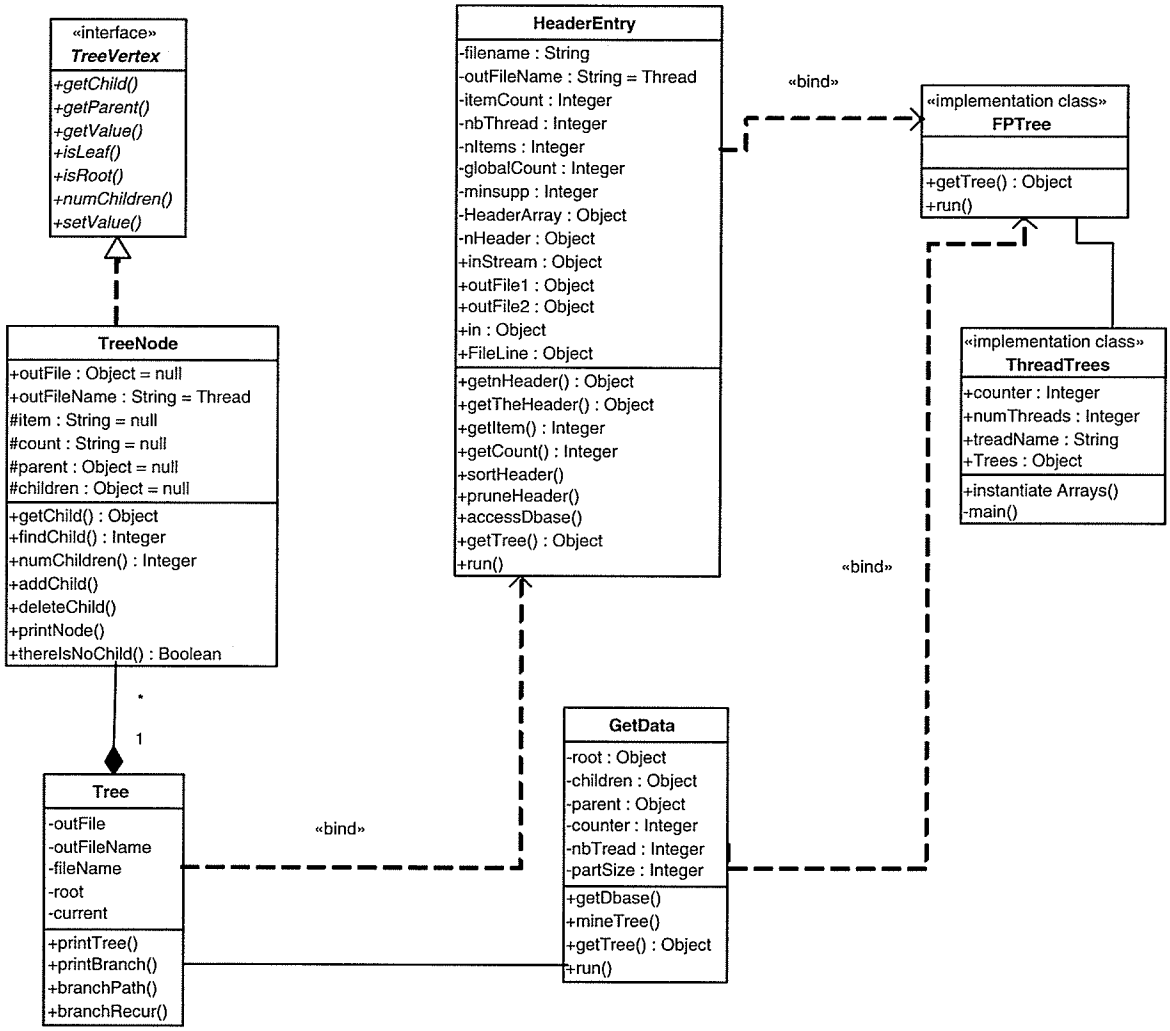

Figure 3.3: UML Class Diagram Representation of the Partitioned Frequent Pattern-Growth Algorithm

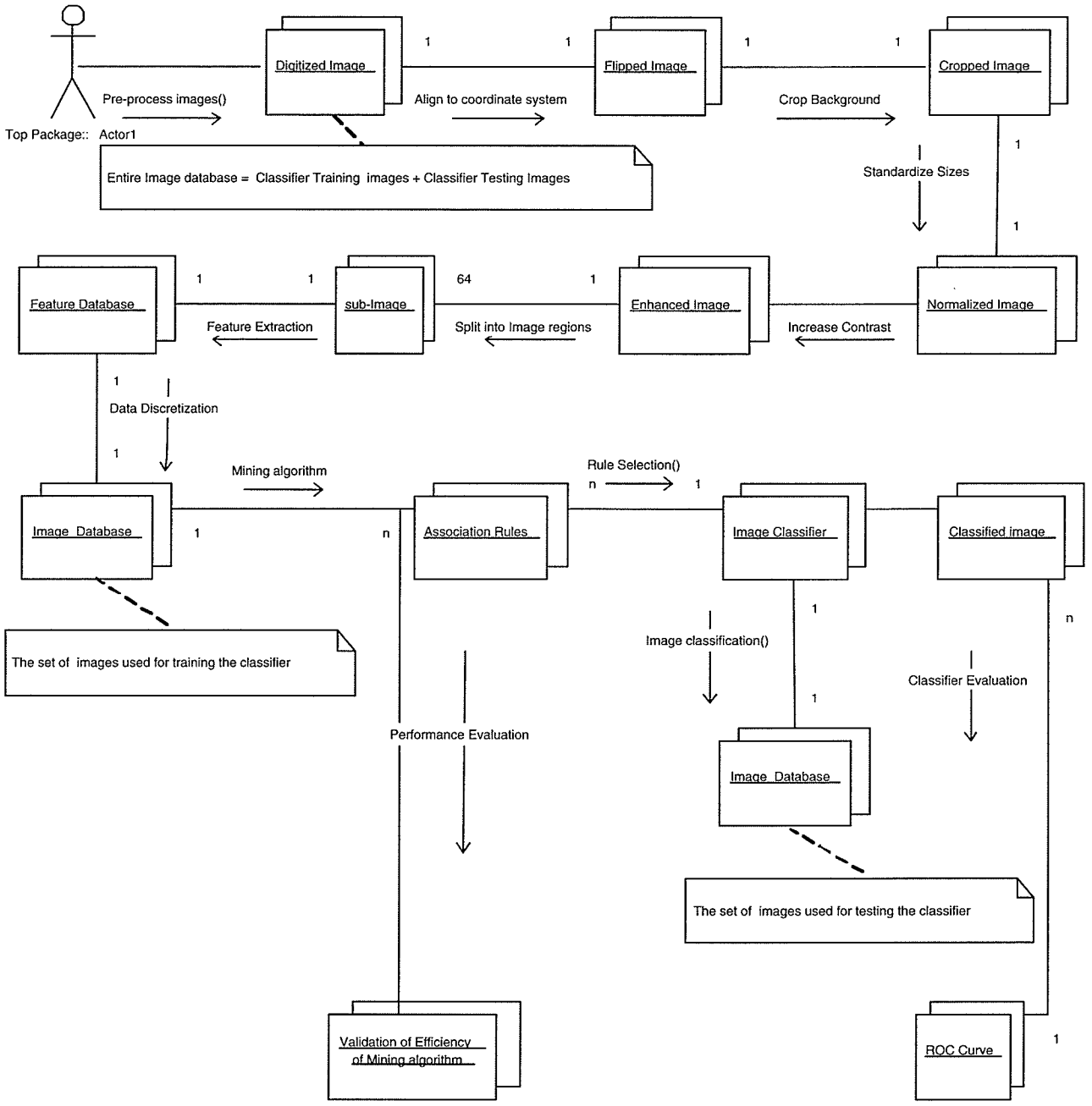

Figure 3.4: UML Representation (Collaboration) of the Image Mining System

abnormal image is marked as shown in Figures 3.5 and 3.6, using the x, y and r values.

For this thesis, I selected the MIAS database because it is freely available and has been widely used in 11 countries by 57 different institutions, thus providing a basis for comparison of results with related research.

Spot segmentation techniques basically locate pixels with high intensity values and groups together high intensity pixels that are close together into internally homogenous regions. Each region is different from its neighbours. All regions whose sizes disqualify them to be potential tumor spots, i.e., they are too large or too small, are eliminated [44]. The spot segmentation technique is often used for the detection of microcalcifications in mammograms. Microcalcifications are a form of cancer cells that are formed by deposits of calcium in breast tissue. They are usually identified by bright spots of high intensity [44]. In this thesis, however, we aim at applying the association rules to categorizing images into 3 image classes: normal malign and begin. Malign and bengin tissues are characterized by various symptoms, one of which could be microcalcifications. Research that focuses on classification of mammograms into these classes has segmented images into small, even sub-regions from which image descriptors are extracted. The image descriptors are used as attribute values for each sub-image in the image mining stage.

3.4 Image Pre-processing

Digital images have to undergo a number of pre-processing steps before statistical measures can be extracted from them. In this research, each digitized image is flipped in order to align it with the x, y coordinate orientation used in the image analysis program. For the mini MIAS database, the flipping process is a 180^0 rotation along the y-axis. I marked the abnormal regions on all abnormal images based on the x, y and r values supplied with the MIAS image collection. The rest of the image preprocessing steps can be roughly classified into image reduction or image enhancement techniques.

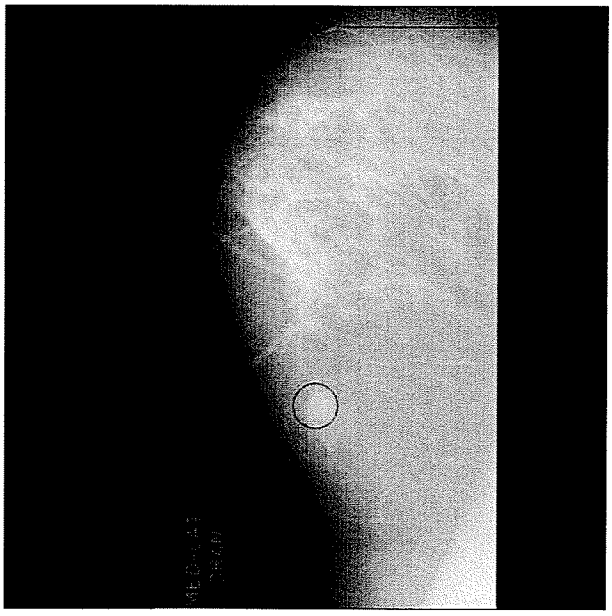

Figure 3.5: Original Image

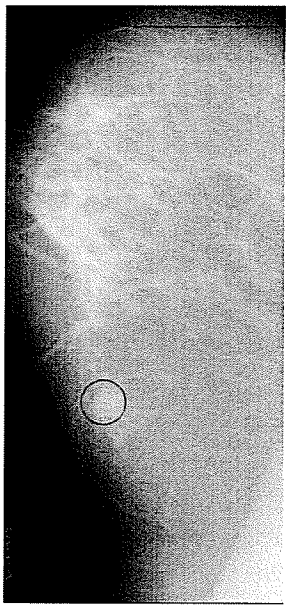

Figure 3.6: Cropped Image

3.4.1 Image Reduction

Each flipped image is a 1024 x 1024 pixel image, which consists of a breast tissue surrounded by background pixels. In some cases, 50% of the entire image can be background pixels. Background pixels have the lowest intensity in an image histogram (pixel level representation of image gray levels). Thus, the background is cropped (chopped off) by vertically and horizontally cutting off parts of each image that have mean gray level values beyond a specified threshold gray level value. Pixels with gray level values greater than or equal to the lower threshold and less than or equal to the upper threshold are regarded as areas of interest while pixels beyond the range are discarded as part of the background. Sections of extremely bright pixels (often outside the breast tissue) form the border of the mammogram film and so can be disregarded. This operation is important because it eliminates a lot of background noise which distorts the image database. Figure 3.6 shows an example of a reduced image.

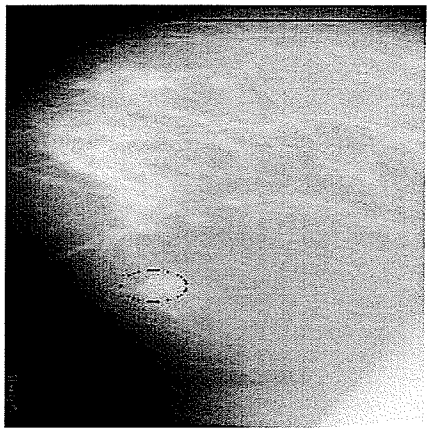

Figure 3.7: Normalized Image

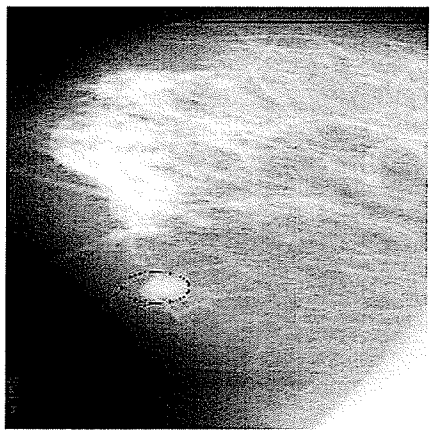

Figure 3.8: Enhanced Image

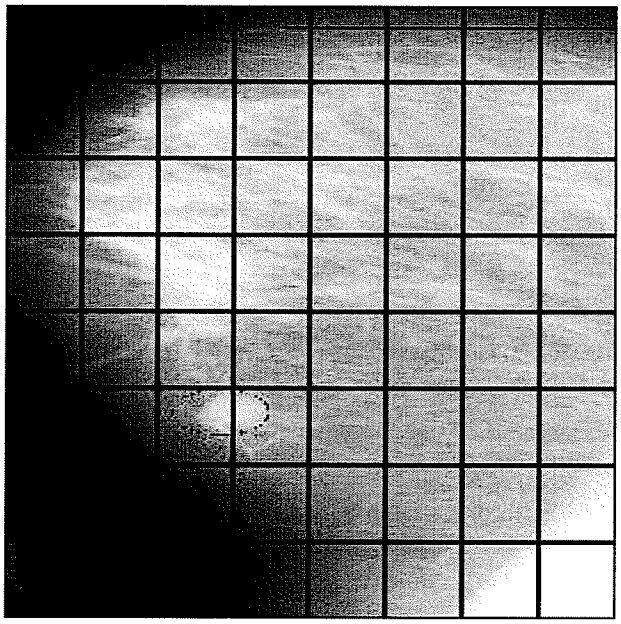

Figure 3.9: Image After Segmentation into Sub-images

3.4.2 Image Enhancement

The cropped images have varied sizes because the area clipped off from each image varies. Therefore, a normalization step is performed to standardize the areas of all images. All (x, y) coordinate values are mapped within a 0 to 255 range. Figure 3.7 shows an example of a normalized image. The normalized images are then enhanced using histogram equalization techniques to increase the gray level contrast between regions of different textures. The histogram equalization algorithm employed in the *imagej* program accentuates images by re-assigning the intensity values of pixels in an input image, $A(x, y)$, using a non-linear transformation function f to produce an output image $B(x, y)$ that has the same distribution of pixels as the input image. The tranformation function for digital images [14] has the form.

$$f(D_A) = max(0, roundD_M * n_K/N^2) - 1 \tag{3.1}$$

where D_A is the intensity density level of image A, D_M is the minimum intensity level, which is often 0, n_k is the number of pixels at intensity level K or less, and N is the number of image pixels. Figure 3.8 shows an example of a enhanced image. Minute details in the breast tissue are accentuated.

3.4.3 Feature Extraction and Selection

To observe variations in regional texture values such that abnormal regions which range from 3 to 197 pixels are detected, each image is split into 64 equal sub-images of 32 x 32 pixels, as shown in Figure 3.9. For each sub-image, four statistical measures associated with texture are extracted. These values are selected based on findings in [44] and [46], which indicate that these statistical measures are the best attribute set for classifying digital mammograms from the MIAS database for the purpose of microcalcification detection. All measures are derived from the basic pixel gray level values and spatial values. The formula from which each measure is derived and its interpretation is discussed in Section 2.3.7.

3.5 Image Mining and Rule Generation

The set of statistical measures extracted from each sub-image makes up a database transaction. The database is divided into two equal parts, one used for training and one for testing the classifier, consisting of an approximately equal proportion of sub-images from abnormal and normal images. A sub-image with 20% or more of its area occupied by a marked abnormality is considered abnormal, otherwise it is said to be normal. The sub-image transactions in the training data set have an additional attribute which specifies each image as normal, benign or malign. The test data set is not mined for association rules, but is used in later stages to evaluate the accuracy of the image classifier.

3.5.1 Database Partitioning and Mining

The training transaction database is mined with the PFP-Growth algorithm. The database is horizontally partitioned amongst a varied number of processing units, at different runs, to observe the effect of parallelization. To measure the speed-up achieved by parallel processing in comparison to the sequential FP-Growth algorithm, I carried experiments with the same data set using 2, 4 and 8 processors.

3.5.2 Rule Generation

Each image region is represented by values of each of the descriptors i.e., each region has a mean, variance, kurtosis and skewness value. These values are discretized into ranges, which reflect the frequency of their occurrence in the entire image data set. For instance, the mean descriptor could be mapped into 4 ranges, say *0-80*, *81-160* and *161-255* average intensity values per region. These three sets could have distinct characteristics associated with each set and so they can be given qualitative attribute names; low, medium and high. In the training data set, each image also has in addition to its descriptor a pre-known class, i.e., normal, benign or malign.

The attributes values of the mean, variance, skewness and kurtosis attributes are discretized into intervals whose ranges are determined by the distribution of values for each

Descriptor	Range	Feature	Interpretation
Mean	0-20	$M0$	Minimum Level
Mean	21-40	$M1$	Low Dim
Mean	41-60	$M2$	Mid Dim
Mean	61-80	$M3$	High Dim
Mean	81-100	$M4$	Dim Grey
Mean	101-120	$M5$	Average Grey
Mean	121-140	$M6$	High Grey
Mean	141-160	$M7$	Bright Grey
Mean	161-180	$M8$	Extreme Grey
Mean	181-200	$M9$	Low Bright
Mean	201-220	$M10$	Mid Bright
Mean	221- 240	$M11$	High Bright
Mean	241-260	$M12$	Maximum Level

Table 3.1: Mean Intensity Measures and Features

attribute. Tables 3.1 to 3.4 show the class intervals and attribute names to which the ranges of attribute values are mapped.

Using tables 3.1 to 3.4, a subimage, I, with attribute values $Mean = 110$, $Variance = 250$, $Skewness = 0$ and $Kurtosis = 15$ is represented in the database as $I\{110, 250, 0, 15\}$ and described as $I\{$*Average Gray, Close to Mean, Average, Averagely Peaked*$\}$. The training data set consists of an equal amount of normal, benign and malign sub-images. The entire database is not trained, in a bid to satisfy the condition that if a record is used in the training set, it must not be used for testing the classifier. This condition must be met if the experiment to test the accuracy of the rules will be fair. A total of 132 malign, 116 benign and 20,174 normal sub-images make up the database. Therefore, normal sub-images, malign sub-images and benign sub-images in a ratio of about 150:1:1. I observed that malign and benign classifying rules where not represented in the rule set when I mined the entire

Descriptor	**Range**	**Feature**	**Qualitative Values**
Variance	0-50	$V0$	Closest to Mean (Mean =0)
Variance	51-100	$V1$	Closer to Mean
Variance	101-500	$V2$	Close to Mean
Variance	501-1000	$V3$	Slightly far from Mean
Variance	1001-2000	$V4$	Far from Mean
Variance	2001-4000	$V5$	Further from Mean
Variance	4001-15000	$V6$	Furthest from Mean

Table 3.2: Variance Measures and Features

Descriptor	**Range**	**Feature**	**Qualitative Values**
Skewness	-32 to -10	$S0$	High Negative
Skewness	-9 to -2	$S1$	Mid Negative
Skewness	-1	$S2$	Low Negative
Skewness	0	$S3$	Average
Skewness	1	$S4$	Low Positive
Skewness	2 to 9	$S5$	Mid Positive
Skewness	10 to 32	$S6$	High Positive
Kurtosis	-2	$S7$	Flat

Table 3.3: Skewness Measures and Features

Descriptor	Range	Feature	Qualitative Values
Kurtosis	-1	$K0$	Less flat
Kurtosis	-0	$K1$	Normal Distributed
Kurtosis	1	$K2$	Peaked- Normal
Kurtosis	2	$K3$	Slightly Peaked
Kurtosis	3 - 20	$K4$	Averagely Peaked
Kurtosis	21 - 100	$K5$	Quite Peaked
Kurtosis	101 - 500	$K6$	Very Peaked
Kurtosis	501 - 1019	$K7$	Extreme Peak

Table 3.4: Kurtosis Measures and Features

data set because their supports were too low to be considered as valid rules. Lowering the minimum support will result in almost all attribute values qualifying as large itemsets. The rules formed from all attribute sets will be meaningless for classification purposes. Thus, the training transaction database is selected and it comprises of 66 malign and 58 benign subimages (half the abnormal set) and 60 randomly selected normal sub-images. The output of mining the training transaction database is a set of association rules which specify the set of measures that strongly occur within the normal, benign and malign image classes.

3.5.3 Building an Image Classifer

The association rules obtained from the mining stage are used to build an image classifier system. We use a rule selection algorithm similar to that used by Zaiane *et al.* [46] to have a basis for classifier accuracy comparisons with the work by Zaiane *et al.* [46], in which the Apriori association rule algorithm is applied on the MIAS mammography collection.

The algorithm for rule selection is described in [46] as show below:

Rule Trimming Scheme

Start Input (A, The set of association rules with rule frequency, $f >=$ Minimum Confidence)
for each image class , C_i, $\in A$, order the set of association rules, A_i such that:
A rule R_1 in A_i is ranked higher than a rule R_2 in A_i **if**
1) R_1 is a superset of R_2
//if R_1 contains all the attributes in R_2 in addition to other attributes

2) R_2 has a higher confidence value than R_1,
3) **if** R_2 and R_1 has equal confidence values, **then** R_2 has a higher support value

a) Remove from A_i, each rule R_1, that has a superset R_2 in A_i
b) Remove from A_i, each rule R_1 that exist in the set of association rules, A_j of another image class C_j *//i.e., remove all conflicting rules*

Stop

Trimming the rule set to increase will increase the accuracy of the image classifier, since some rules will mislead the classification process. An illustration of the rule selection algorithm is as follows:

1. First, remove all rules which are a subset of another rule. For example $M2, V3, K5 \Rightarrow Normal$ is a subset of $M2, V3, S6, K5 \Rightarrow Normal$. Therefore the first rule is removed.

2. Next, remove all rules that contradict each other. For instance $M2, V3, K5 \Rightarrow Normal$ is a contradiction of $M2, V3, K5 \Rightarrow Benign$, and so both rules are removed.

3.5.4 Performance of the Image Classifier

The test data set is used to test the validity (accuracy) of these rules. If an image region is categorized by the rule set as being malign, and the region is actually malign (based on prior knowledge from radiologists truth markings), then the image region is said to be rightly classified. A confusion matrix is a standard method of evaluating the performance of data mining algorithms that depend on user-defined thresholds [41]. The rows of the matrix represent the actual classes of objects which is assumed correct. The columns of the matrix represent the predicted (by the classifer) classes of objects. The components of a confusion matrix are defined as :

1. True Positives (TP): Cases that are actual positives and are predicted as being positive by the classifier.

2. True Negatives (TN): Cases that are actual negatives and are predicted as being negative by the classifier.

3. False Positives (FP): Cases that are actual negatives and are predicted as being positive by the classifier.

| **Predicted Negati**

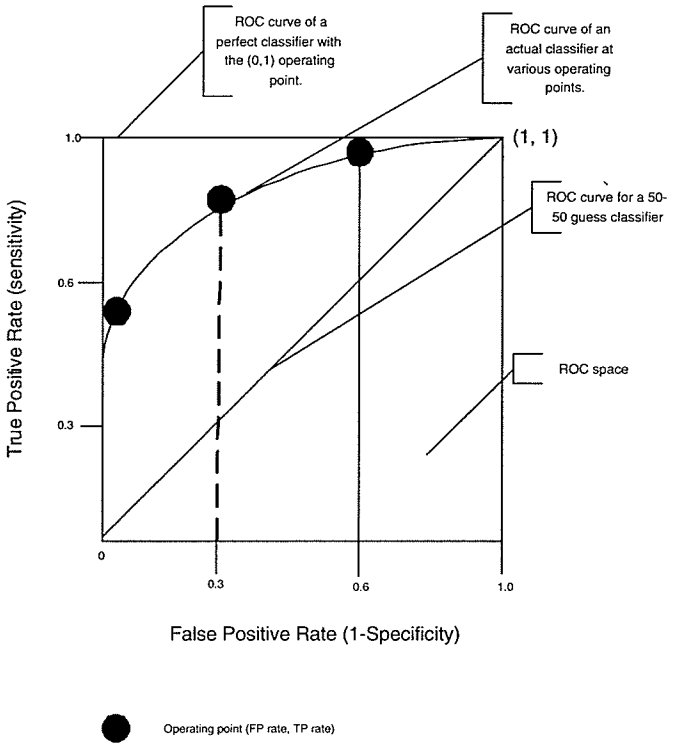

Figure 3.10: ROC Curve at Various Operating Points

processing. The ROC curve is a measure of the performance, at various operating conditions, of a receiver (human or mechanical) at assigning cases into distinctive classes [41]. The ROC curve shows the tradeoff between sensitivity (i.e. detecting all abnormal cases accurately) and specificity (i.e. classifying normal cases accurately) values at different operating points. Operating points (in this thesis, minimum support and confidence values) are user-defined criteria with which tradeoffs between specificity and sensitivity can be adjusted, depending on the use of the classifier. The upper rightmost point on an ROC plot is $(1, 1)$. It is always shown even though such an operating point does not naturally exist.

The area under an ROC function (AUC) provides threshold-dependent classifiers (such as an association rule-based classifier) with a single measure of overall accuracy which is not dependent upon a specific user-defined threshold [41]. The area under an ROC curve indicates a measure of the ability of the classifier to accurately classify data. If the AUC has a value of 0.9, it means that each time a random case is classified, there is a 90% chance

of classifying that case accurately.

When a classifier is based on a $50-50$ classification chance, its ROC curve is plotted as a 45-degree diagonal and its AUC is 0.5. Figure 3.10 shows a graph for a $50-50$ chance ROC curve, and an actual ROC curve for a classifier, plotted at 3 operating points. Sensitivity and specificity values range between 0 and 1. Sensitivity is the accuracy of classification among positive cases and while specificity is the accuracy of classification among negative cases. Here, a negative case is taken to a case that has indications of cancer, i.e. an abnormal case. A perfect classifier has a false positive rate of 0 (i.e., no abnormal case is wrongly classified as being normal) and a true positive rate of 1 (i.e., all the normal cases are rightly classified as being normal). The ROC curve of a perfect classifier is plotted a a straight horizontal line from the point (0, 1) to the upper rightmost point (1,1), and so it has an AUC that fills the ROC space. Therefore, when a classifier's curve is close to the left-hand border and top-border of the ROC curve i.e., close to the curve of the perfect classifier, it is regarded as very accurate.

The accuracy of the image classifier is calculated as the area under the ROC curve. The AUC is approximated using the trapezoidal rule [41], which is given as:

$$\textit{Area:} \qquad AUC = \frac{H}{2}(a_1 + a_n + 2 * (\sum_{i=2}^{i=n-1} a_i)) \tag{3.7}$$

H is the width of equal columns into which the AUC is divided, i.e., the maximum width of 1.0 is divided into equal intervals. The value y_i is the height of the i^{th} column, which can be projected from the ROC curve to y-axis value. y_1 and y_n refer to the height of the first and last column respectively.

Chapter 4

Experimental Results

This chapter presents the results of the image mining process which includes a set of association rules and an analysis of the performance of the PFP-Growth algorithm using various numbers of processors. This chapter also presents an evaluation of the accuracy of the image classifier.

4.1 Association Rules

The result of mining the training data set with the PFP-Growth algorithm is a set of association rules, which describe the co-occurrence of certain image measures with the various image classes: normal, benign and malign. Not all rules are useful. The rule selection scheme introduced in Section 3.5.3 removes all rules that state contradictory implications and rules that are subsets of other rules.

The minimum support value of a mining experiment is directly proportional to the number of attributes in the rule set and the size of the rule set. For instance, when the minimum support value is set at 40% of the data size, more attributes are included in the rule set compared to when the minimum support is set to 60% of the data size. A small rule set (with a high minimum support) is strict in its classification requirements, and therefore very specific, i.e., a lower false positive rate. A large rule set (generated with a low minimum support) has more participating attributes than a small rule set, which implies

Rule No.	Association Rule	Comment
1	$M6 \Rightarrow Normal$	contradicts rule 3 (malign)
2	$M8$, $K3 \Rightarrow Normal$	superset of rule 5 (normal)
3	$M8$,$V3 \Rightarrow Normal$	useful rule
4	$K5 \Rightarrow Normal$	contradicts rule 5 (benign)
5	$K3 \Rightarrow Normal$	subset of rule 2 (normal
1	$M10$, $V4 \Rightarrow Benign$	useful rule
2	$M5$, $K1 \Rightarrow Benign$	useful rule
4	$K5 \Rightarrow Benign$	useful rule
3	$V1$, $K1 \Rightarrow Benign$	useful rule
5	$S3$, $K5 \Rightarrow Benign$	contradicts rule 4 (normal)
1	$K2 \Rightarrow Malign$	useful rule
2	$V3$, $S3$, $K2 \Rightarrow Malign$	superset of rule 4, 5 (malign)
3	$M6$, $V6 \Rightarrow Malign$	contradicts rule 1 (normal)
4	$V3$, $K2 \Rightarrow Malign$	subset of rule 2 (malign)
5	$V3$, $S3 \Rightarrow Normal$	subset of rule 2 (malign)

Table 4.1: Association Rules at 40% support and 10% Confidence

that the criteria for classification into a particular image set is less strict and classification is more sensitive, i.e., a higher true positive rate. However, with a large rule set, the specificity of classification is low since attributes which classify one class can also show up in another class. Therefore, there is a trade off between sensitivity and specificity.

In the experiments for this thesis, I attempted to achieve a balance of maximal sensitivity and specificity and thereby obtain a wholesome view of the classifier by varying the minimum support and confidence values. When the minimum support value is low (i.e., sensitivity is high), a balance is made by applying a high minimum confidence value, thereby improving the specificity at the rule selection stage. I used a support range of $40\% - 25\%$ and a confidence range of $10\% - 40\%$ because at lower or upper limits of support and confidence,

Rule No.	Association Rule	Comment
1	$V2, S5 \Rightarrow Normal$	useful rule
2	$K3 \Rightarrow Normal$	subset of rule 3 and 4 (normal)
3	$V2, K3 \Rightarrow Normal$	superset of rule 2 (normal)
4	$S5, K3 \Rightarrow Normal$	superset of rule 2 (normal)
1	$M10, S3 \Rightarrow Benign$	useful rule
2	$K1 \Rightarrow Benign$	useful rule
3	$V6, K1 \Rightarrow Benign$	contradicts rule 3 (malign)
4	$S3, K1 \Rightarrow Benign$	useful rule
1	$K2 \Rightarrow Malign$	useful rule
2	$V3, K2 \Rightarrow Malign$	useful rule
3	$M6, V6 \Rightarrow Malign$	contradicts rule 3(benign)

Table 4.2: Association Rules at 35% support and 20% Confidence

Rule No.	Association Rule	Comment
1	$M8, V2, K3 \Rightarrow Normal$	useful rule
2	$M4, K3 \Rightarrow Normal$	contradicts rule 2, 4 (malign)
3	$M3, V2 \Rightarrow Normal$	contradicts rule 1 (normal)
1	$V3, K1 \Rightarrow Benign$	useful rule
2	$S7, K1 \Rightarrow Benign$	useful rule
3	$S7 \Rightarrow Benign$	useful rule
1	$S3, K2 \Rightarrow Malign$	useful rule
2	$V4, S3 \Rightarrow Malign$	contradicts rule 2 (normal)
3	$M3 \Rightarrow Malign$	contradicts rule 3(normal)
4	$M4, S2 \Rightarrow Malign$	contradicts rule 2 (normal)

Table 4.3: Association Rules at 30% support and 30% Confidence

Rule No.	Association Rule	Comment
1	$M5$, $V6$, $K3 \Rightarrow Normal$	useful rule
2	$S2$, $K2 \Rightarrow Normal$	contradicts rule 4 (malign) at $\{30, 30\}$
3	$M2$, $V2 \Rightarrow Normal$	contradicts rule 4 (benign)
4	$M8$, $S5$, $K3 \Rightarrow Normal$	useful rule
1	$M10$, $K1 \Rightarrow Benign$	useful rule
2	$S7$, $K2 \Rightarrow Benign$	contradicts rule 2 (normal)
3	$M10$, $S7 \Rightarrow Benign$	useful rule
4	$M2 \Rightarrow Benign$	contradicts rule 1(normal)
1	$V3$, $K5 \Rightarrow Malign$	contradicts rule 4 (normal at 40, 10)
2	$S3$, $K4 \Rightarrow Malign$	useful rule
3	$V3$, $S3 \Rightarrow Malign$	useful rule
4	$S3 \Rightarrow Malign$	subset of rule 3 (malign)

Table 4.4: Association Rules at 25% support and 40% Confidence

Cut off	Association Rule	Comment
40% support & 10% Confidence	$M8$, $V3$, $K3 \Rightarrow Normal$	
40% support & 10% Confidence	$M10$, $V4$, $K1 \Rightarrow Benign$	
40% support & 10% Confidence	$V3$, $S3$, $K2 \Rightarrow Malign$	
35% support & 20% Confidence	$V2$, $S5$, $K3 \Rightarrow Normal$	
35% support & 20% Confidence	$M10$, $S3$, $K1 \Rightarrow Benign$	
35% support & 20% Confidence	$V3$,$K2 \Rightarrow Malign$	
30% support & 30% Confidence	$M8$, $V2$, $K3 \Rightarrow Normal$	
30% support & 30% Confidence	$V3$, $S7$, $K1 \Rightarrow Benign$	
30% support & 30% Confidence	$S3$, $K2 \Rightarrow Malign$	
25% support & 40% Confidence	$M8$, $S5$, $K3 \Rightarrow Normal$	
25% support & 40% Confidence	$M10$, $S7$, $K1 \Rightarrow Benign$	
25% support & 40% Confidence	$V3$, $S3$, $K4 \Rightarrow Malign$	

Table 4.5: Final Set of Association Rules After Rule Trimming

too few or too many rules are generated, i.e., this is the optimal range for support and confidence limits for the image data set. An operating point, such as minimum support limit = 40%, and minimum confidence limit = 10% is denoted as a point $\{40, 10\}$. A sample set of association rules obtained from mining a training set at varying minimum support and minimum confidence values is listed in Tables 4.1 to 4.4.

Based on the rule trimming scheme, a number of rules are eliminated. For example, in Table 4.1 *rule* 5 (normal) is a subset of *rule* 2 (normal), and so *rule* 5 can be trimmed from the rule set. Also, in Table 4.3, *rule* 3 (normal) contradicts *rule* 3 (malign), and so both rules are removed from the rule set.

The resultant set of rules obtained at each operating point (minimum support and minimum confidence) is applied to classify the test data set. At each operating point, I construct a confusion matrix (see Section 3.5.4) which is shown in Tables 4.8 to 4.11.

Processors	10210 records	20421 records	40842 records	81684 records
1	2473.20	8834.80	11067.00	16119.40
2	1883.40	6086.40	6681.38	11668.10
4	1575.70	4482.10	5454.70	9184.15
8	1381.88	3455.42	4066.00	7992.92

Table 4.6: Execution Time of Processors using the PFP-Growth Algorithm

4.2 Performance Evaluation

4.2.1 Scalability with number of Processing Units

To validate the efficiency of the PFP-Growth algorithm, I mined an image data set with the sequential FP-Growth algorithm (1 processor) and then compared it with the execution times (in milliseconds) obtained from the parallel algorithm with various numbers of processors (2, 4, 8). Initially, the experiments were carried out using Java threads. The mining results with threads achieves no speed up. In fact, the processing time increases with an increase in the number of threads. This is as a result of the fact that Java threads use a many to one model, i.e., they all share the same processor and simply use a time-sharing scheme so that they appear to run simultaneously.

Therefore, I repeated the experiments using a client/server model, such that all the processing on slave threads were allotted to client processors on different nodes in a shared memory local area network. Each node in the network has an Intel Pentium III processor. The master thread ran on a server node.

The results of the client/server experiment is shown in Table 4.6 and plotted as a graph of execution time versus number of processors in Figure 4.1. The processing speed obtained for each set is an average of execution times for 5 experiments at each processor scale.

The client/server model, using 2 processors achieved an average speed up of 45.07% over the sequential FP-Growth algorithm with 1 processor. However, the speed up decreases with an increase in number of processors. The calculation of speed up is given by Equation 4.1

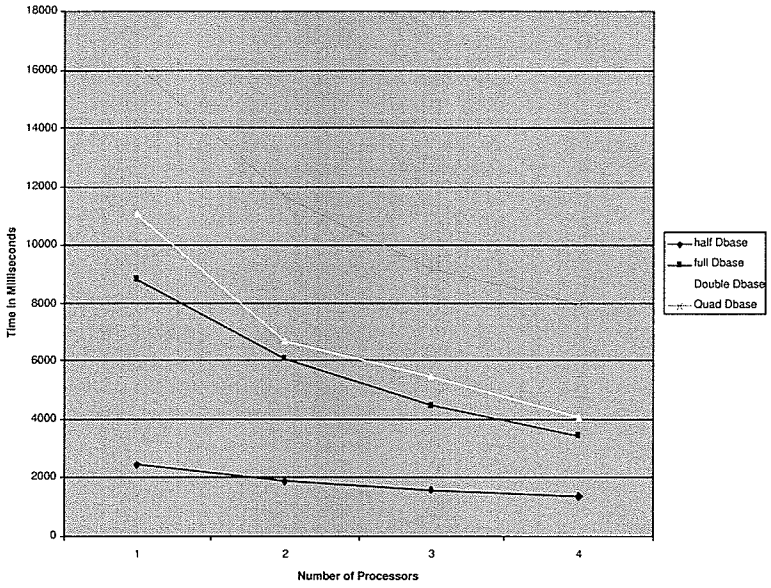

Figure 4.1: Performance of the PFP-Growth Algorithm

$$\textit{Speed-up:} \qquad S = (1 - \frac{T_A}{T_B}) * 100\% \tag{4.1}$$

where T_A and T_B are the execution times of two experiments A and B respectively. Experiment A is said to achieve a percentage speed up of S in comparison to experiment B. Experiment A is regarded to be more efficient than experiment B, in terms of execution time, if $S > 0$.

Number of Records	**1 to 2 procs.**	**2 to 4 procs.**	**4 to 8 procs.**
10210	31.32	19.52	14.03
20421	45.16	35.79	29.71
40842	65.64	22.49	34.15
81648	38.15	27.05	14.9
Av. Speed up (%)	**45.07**	**26.21**	**23.20**

Table 4.7: Speed Up of Processors using the PFP-Growth Algorithm

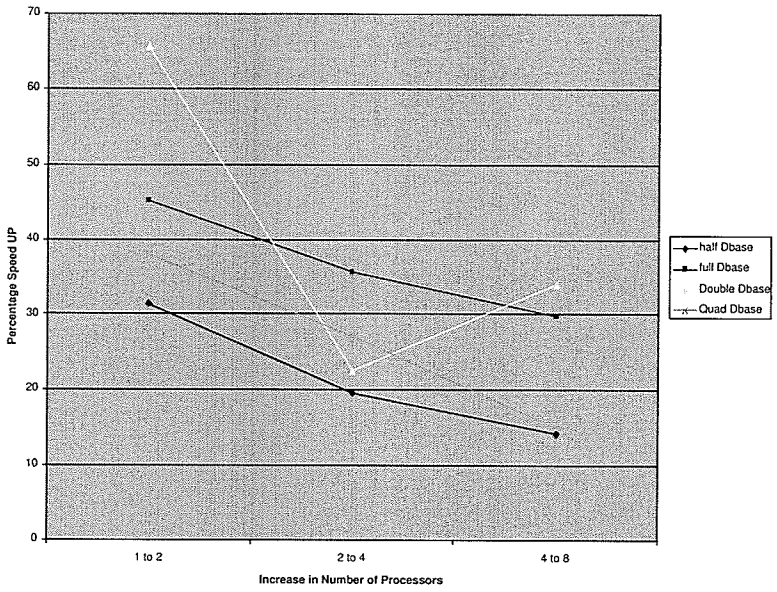

Figure 4.2: Processor Speed Up

4.2.2 Scalability with Data Size

In one experiment, I varied the number of sub-image records in order to assess the scalability of the PFP-Growth algorithm with respect to data size. Figure 4.3 shows the processing speeds obtained. The experimental results indicate that there are processing time gains as the number of records in the data set increases. This gain in processing time suggests that the PFP-Growth algorithm is well suited to larger data sizes.

4.3 Classifier Results

For mammogram classification, the optimal operating point is considered to be the point at which the false positive rate is lowest, i.e., specificity is highest [44]. From the experiments, specificity is highest at operating point {40, 10}, with support and confidence rates of 40% and 10% respectively. Table 4.8 show the confusion matrix at operating point {40, 10}.

At the {40, 10} operating point, the image data set with 4 processors resulted in a set of 8 association rules, shown in Table 4.3. These rules to classify the test data set. Image records are assigned into classes based on what the rule set predicts they would be.

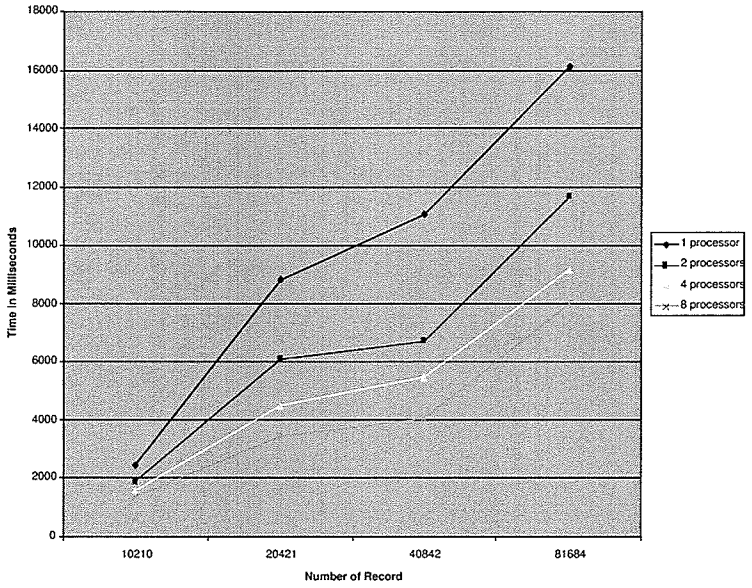

Figure 4.3: Performance at varying Data Sizes

Predicted Classes →	**Benign**	**Malign**	**Normal**	**Total**
Actual Benign	TN = 49	TN = 7	FP= 2	58
Actual Malign	TN = 11	TN = 53	FP= 2	65
Actual Normal	FN = 10	FN = 4	TP = 46	60

Table 4.8: Confusion Matrix at operating point {40, 10}

Confusion matrixes (Tables 4.8 to 4.11) show the classification results.

The true positive rate (i.e., sensitivity) and the false positive rate (1-specificity) at operating point {40, 10} , are calculated using Equations 3.4 and 3.2 respectively. The accuracy of the classifier is calculated using the area under the ROC curve, rather than with the accuracy formula, to provide an overall summary of the performance of the classifier.

$$\textit{True Positive Rate: } TPRate = \frac{TP}{TP + FN} = \frac{46}{46 + (4 + 10)} = \frac{46}{60} = 0.767 \qquad (4.2)$$

Predicted Classes →	**Benign**	**Malign**	**Normal**	**Total**
Actual Benign	TN = 35	TN = 6	FP= 17	58
Actual Malign	TN = 12	TN = 42	FP= 11	65
Actual Normal	FN = 5	FN = 3	TP = 52	60

Table 4.9: Confusion Matrix at operating point {35, 20}

Predicted Classes →	**Benign**	**Malign**	**Normal**	**Total**
Actual Benign	TN = 26	TN = 5	FP= 27	58
Actual Malign	TN = 5	TN = 36	FP= 24	65
Actual Normal	FN = 3	FN = 2	TP = 55	60

Table 4.10: Confusion Matrix at operating point {30, 30}

$$\textit{False Positive Rate: } FPRate = \frac{FP}{TN + FP} = \frac{2+2}{(49+7+11+53)+(2+2)} = \frac{4}{124} = 0.032 \tag{4.3}$$

$$Specificity = 1 - FPrate = 1 - 0.032 = 0.968 \tag{4.4}$$

The confusion matrixes at other operating points are shown in Tables 4.9 to 4.11.

Predicted Classes →	**Benign**	**Malign**	**Normal**	**Total**
Actual Benign	TN = 20	TN = 3	FP= 35	58
Actual Malign	TN = 24	TN = 4	FP= 37	65
Actual Normal	FN = 0	FN = 2	TP =58	60

Table 4.11: Confusion Matrix at operating point {25, 40}

{Min. Support, Min. Confidence}	Specificity	FP Rate	TP Rate
N/A, N/A	1.000	0.000	0.000
40, 10	0.968	0.032	0.767
35, 20	0.772	0.228	0.867
30, 30	0.585	0.415	0.917
25, 40	0.415	0.585	0.967
N/A, N, A	0.00	1.000	1.000

Table 4.12: Specificity and Sensitivity Rates at Various Operating Points

4.3.1 Detection Accuracy

The percentage of false positives (i.e., cancerous sub-images wrongly classified as normal) is 3.2% at the $\{40, 10\}$ operating point. The classifier is highly accurate if it has a low false positive percentage..

The area under the ROC curve shown in Figure 4.4 is evaluated with the trapezoidal rule in Equation 3.7 using the TP Rate values (on the y-axis) in Table 4.12, and equal widths of 0.2 units. The AUC value obtained indicates a detection accuracy of 80.36% as shown in Equation 4.5.

$$\textit{Area:} \quad AUC = \frac{0.2}{2}(0+1+2*(0.767+0.867+0.917+0.967)) = 0.1\{8.036\} = 0.8036 \quad (4.5)$$

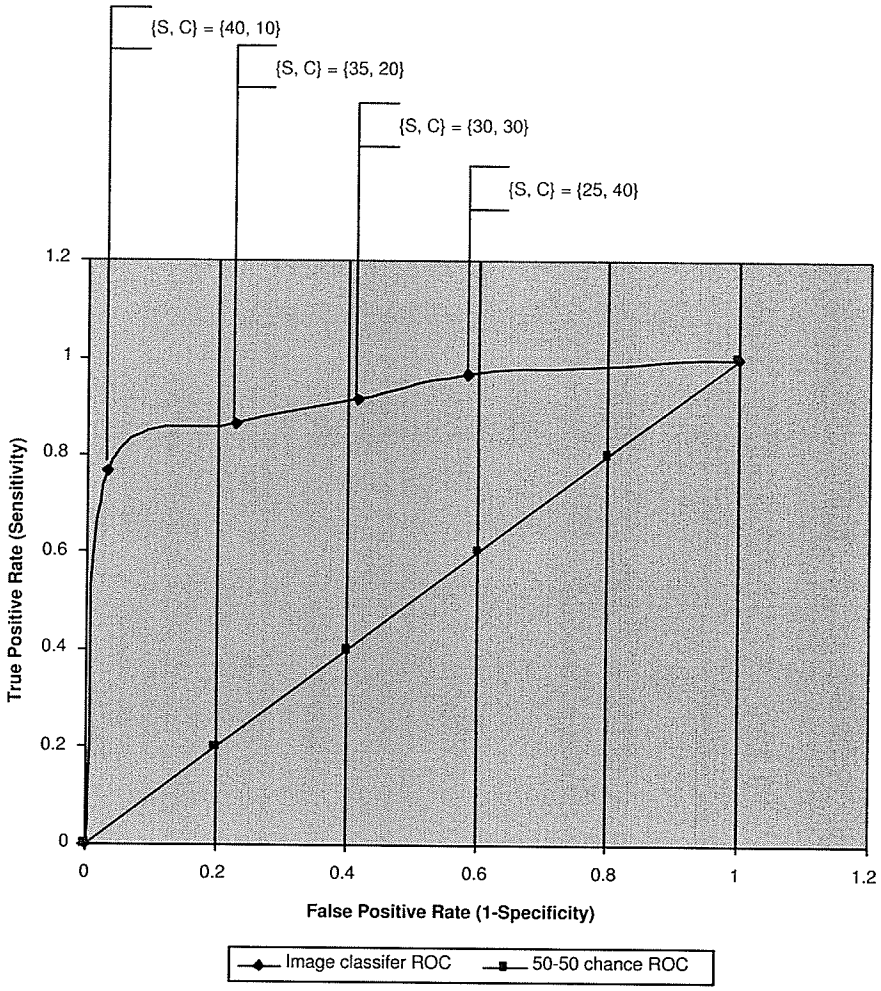

Figure 4.4: ROC curve for PFP-Growth based Image Classifier

Chapter 5

Discussion and Conclusions

This chapter presents a summary and discussion of the experimental results obtained in this thesis, and provides suggestions for future work in this area.

5.1 Summary of Findings

In this thesis, we presented the design of the PFP-Growth algorithm and applied it to a medical image data set. The PFP-Growth algorithm is an extension of the FP-Growth association rule algorithm that applies parallel heuristics in a bid to achieve an improved performance with large data sets that have long patterns. The medical images are obtained from the MIAS database, which comprises of 322 1024 x 1024 digital mammograms. Each image is preprocessed and segmented into 64 sub-images, each of which is represented by a set of image features. The feature set for each sub-image is stored as a database record. The image database was then mined using the PFP-Growth algorithm. Initially, results did not indicate a speed up with an increase in processing units. This was because each processing unit is a Java thread which shares a single processor with all other threads and just uses a time-slicing scheme. However, when these threads are allocated to different processors in a shared memory network, they achieve between 23.20% to 45.07% speed up. The improvement was particularly good within the phases that previously caused processing bottlenecks, such as the tree-building phase. The result of mining the database is a set of

association rules that is used as a framework for an image classifier. The classifier showed in a set of experiments a detection accuracy of 80.36%

5.1.1 Effect of Parallelism

The aim of parallelism in the PFP-Growth algorithm is to increase available processing power and to reduce processing bottlenecks that occur when mining large data sets based on the FP-growth algorithm.

Bottlenecks can occur within the tree building phase (when the FP-tree is very large) and the pattern mining phase if patterns are very long. These phases have been parallelized in the implementation of the PFP-Growth algorithm. Experimental results indicate processing speed ups with an increase in the number of processors. As the number of processors increase from 1-2, 2-4 and 4-8 the speed up reduces gradually from 45.07% to 26.21% and then to 23.20% (see Table 4.7). It is reasoned that communication overheads increase with the number of processors and so speed up reduces. In addition, there are 2 merging phases within the PFP-Growth algorithm (merging of frequency counts and merging of FP-trees) right after which all processors wait for the master processor to give a global feedback before proceeding to the next phase. The wait for a global feedback indicates that speed of the previous phase to the feedback is determined by the speed of the slowest unit.

5.1.2 Effect of Data Size

Large databases have often constituted a problem for data mining algorithms. Thus, an algorithm which provides gains with increases in data size is highly desirable. The PFP-Growth algorithm in Figure 4.3 shows a steady, proportionate increase in execution time with increases in the numbers of records. However, as the data sizes get larger (see Figure 4.2), there is a drop in the amount of speed up that is achieved. I reason that, the decrease in speed up could be a result of losses in memory efficiency as memory requirements increase.

5.2 Future Work

Possible extensions to this thesis include:

1. The PFP-Growth algorithm has two merging points: after the building of the local trees and after mining local patterns, where each slave processing units has to wait for a global feedback from the master processing unit, before they can continue. Therefore the execution time in these phases is the speed of the slowest processor. To improve processing efficiency, the PFP-growth algorithm can be enhanced by decentralizing feedback at these merging points. One possibility will be to run all the phases of FP-Growth mining in each processing unit and then send the rules to the master processor. Rule selection is performed by the master processor. It would be interesting to ascertain if the classifier will have the same accuracy or even an improved accuracy compared to the PFP-Growth algorithm.

2. Our mining set consists of 322 mammograms from the MIAS mammography collection. The MIAS collection is the largest publicly available single image collection. Other image collections can be integrated in future research to obtain a larger mining set. It is reasoned that a larger mining set will generate more reliable rules and possibly increase the accuracy of the image classifier. Detection accuracy might also be increased if additional image features that will not mislead classification can be extracted. In addition, this research can also be extended to images in other domains such as geological and geographical maps used for detecting oil rich regions. These images are are easily available in large collections.

3. Databases today are often distributed in nature. It will be of interest to further extend the PFP-Growth algorithm to handle distributed data sets. The algorithm can also be further enhanced to accommodate concerns that arise in distributed domains, such as data security and distribution issues. An improved algorithm can, for instance, mine heterogenous image formats in a distributed network. It is significant and interesting to study the constraints that a distributed environment will place on mining asso-

ciation rules. Some of these constraints include workload balancing, communication minimization, and synchronization of data across multiple sites.

References

[1] C. Aggrawal and P. Yu "Mining Large Itemsets for Association Rules", *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, Vol.2, No.1, 1998, pp. 23-31.

[2] R. Aggrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases", *Proceedings of the ACM SIGMOD Conference on Management of Data*, Washington D.C., May 1993, pp. 207-216.

[3] R. Aggrawal and R. Srikant, "Fast Algorithms for Mining Association Rules", *Proceedings of the 20th International Conference of Very Large Data Bases (VLDB)*, Chile, 1994, pp. 487- 499.

[4] R. Aggrawal and J. Shafer, "Parallel Mining of Association Rules", *IEEE Transactions in Knowledge and Data Engineering*, Vol. 8, No. 6, December 1994, pp. 962-969.

[5] M. Antonie, O. Zaiane, and A. Coman, "Application of Data Mining Techniques for Medical Image Classification", *Proceedings of the Second International Workshop on Multimedia Data Mining in conjunction with ACM SIGKDD Conference (MDM/KDD2001)*, San Francisco, U.S.A., August 26, 2001, pp. 94-101.

[6] P. Berkin, "Survey of Clustering Data Mining Techniques", *Accrue Software, Inc.*, 2001. Available at: *http : //citeseer.nj.nec.com/berkhin02survey.html*

[7] S. Brin, R. Motwani, J. Ullman, and S. Tsur, "Dynamic Itemset Counting and Implication Rules for Market Basket Data", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, May 1997, Tucson, Arizon, U.S.A. pp. 255 - 264.

[8] F. Carrano and J. Prichard, *"Data Abstraction and Problem Solving with Java: Walls and Mirrors"*, Addison Wesley, Boston, U.S.A., 2001, pp. 371-380.

[9] I. Christoyianni, E. Dermatas, and G. Kokkinakis, "Neutral Classification of Abnormal Tissue in Digital Mammography using Statistical Features of the Texture ", *IEEE International Conference on Electronics Circuits and Systems (ICECS'99)*, Pafos, Cyprus, September 1999, pp. 117-120.

[10] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *"Introduction to Algorithms"*, Second Edition, The MIT Press, Cambridge, Massachusetts, U.S.A., 2001, pp. 532, 540-546.

[11] M. Craven and J. Shavlik, "Using Neural Networks for Data Mining", *Future Generation Computer Systems Special Issue on Data Mining,* Vol. 13, pp.211-229, 1998.

[12] S. A. Ehikioya and A. Olukunle, "Mining of Association Rules in Medical Image Data Sets", *Proceedings of the 20th Symposium for Computer Applications in Radiology (SCAR 2003)*, Boston, Massachusetts, USA, June 7-10, 2003.

[13] S. A. Ehikioya and A. Olukunle, "On the Mining of Association Rules in Medical Image Data Sets", Nagib Callaos, Baoyu Zheng, and Firoz Kaderali (Editors), *Proceedings of the 6th World Multiconference on Systemics, Cybernatics and Informatics (SCI-2002): Volume V – Computer Science I*, Orlando, Florida, USA, July 14 - 18, 2002. pp. 17 - 22.

[14] B. Fisher, S. Perkins, A. Walker, and E. Wolfart, "Hypermedia Image Processing Reference", Department of Artificial Intelligence, University of Edinburgh, U.K., 1994. Available at: *http : //www.cee.hw.ac.uk/hipr/html/hipr$_t$op.html*

[15] D. Forsyth, J. Malik, M. Fleck, H. Greenspan, T. Leung, S. Belongie, C. Carson, and C. Bregler, "Finding Pictures of Objects in Large Collections of Images", *Technical Report*, U.C. Berkeley, CS Division, 1997.

[16] S. Ghebreab, "Information Retrieval and Exploration in Mining Medical Image Collections", *VISIM Workshop, Call for Papers*, Uretch Netherlands, 2001. Available at: http://www.science.uva.nl/research/isis/VISIM/

[17] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation", *Proceedings of SIGMOD-2000*, Dallas, U.S.A., May 2000, pp. 1-12.

[18] J. Han and M. Kamber, *"Data Mining: Concepts and Techniques"*, Morgan Kaufmann Press, San Francisco, U.S.A., 2001, pp. 225-277.

[19] R. Haralick, "Statistical and Structural Approaches to Texture", *Proceedings of the IEEE*, Vol. 67, No. 5, May 1979, pp. 786-804.

[20] J. Hipp, V. Guntzer, and G. Nakhaeizadeh, "Algorithms for Association Rule Mining: A General Survey and Comparison", *Proceedings of ACM SIGKDD*, Volume 2, Issue 1, July 2000, pp. 58-64.

[21] J. Hipp, V. Guntzer, and G. Nakhaeizadeh, "Deriving a Superior Algorithm by Analyzing Todays Approaches", *4th European Symposium on Principles of Data Mining and Knowledge Discovery*, 2000, pp. 78-87.

[22] M. Holsheimer, M. Kersten, H. Mannila, and H. Toivonen, "A Perspective on Databases and Data Mining", *1st International Conference on Knowledge Discovery and Data Mining (KDD-95)*, Montreal, Canada, August 1995, pp. 150-155.

[23] M. Houtsma and A. Swami, "Set-oriented Mining of Association Rules" *Proceedings of ACM SIGMOD-93*, 1993, pp. 207-216.

[24] "ImageJ, Version 1.29: National Institutes of Health (NIH) Image Program", *U.S. National Institutes of Health*, 2002. Available at: http://rsb.info.nih.gov/nih-image/.

[25] B. Kalman, S. Kwasny, and W. Reinus "Diagnostic Screening of Digital Mammograms using Wavelets and Neural Networks to Extract Structure", *Technical Report 98-20*, Department of Computer Science, Washington University, St. Loius, U.S.A. 1998.

[26] B. Kovalerchuk and E. Vityaev, *Data Mining in Finance*, Kluwer Acedemic Publishers, 2000 pp. 9-10.

[27] N. Lavrac, "Data Mining in Medicine: Selected Techniques and Applications", *Proceedings of the Second International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, March 1998, pp. 11-31.

[28] W. Li, J. Han, and J. Pei, "CMAR: Accurate and Efficient Classification based on Multiple Class-Association Rules", *Proceedings of IEEE 2001 International Conference on Data Mining (ICDM 2001)*, November to December, 2001, pp. 396-376.

[29] M. Loew, "Feature Extraction", J. Fitzpatrick and M. Sonka (Editors), *Handbook of Medical Imaging, Vol.2: Medical Image Processing and Analysis* SPIE Press, Vol. PM80, Washington, U.S.A., June 2000. pp. 9-22 (Available at: http://www.seas.gwu.edu/ medimage/feature.pdf).

[30] J. Suckling, J. Parker, D. Dance, S. Astley, I. Hutt, C. Boggis. I. Ricketts, E. Stamatakis, N. Cerneaz, S. Kok, P. Taylor, D. Betal, and J. Savage "The Mammographic Image Analysis Society Digital Mammogram Database", *Exerpta Medica. International Congress Series 1069*, 1994, pp.375-378.

[31] D. Nigrin and I. Kohane, "Data Mining by Clinicians", *American Medical Informatics Association(AMIA), Annual Symposium*, Orlando, Florida, USA, November 1998.

[32] A. Olukunle and S. A. Ehikioya, "A Fast Algorithm for Mining Association Rules in Medical Image Data", *Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering (CCECE'02)*, Winnipeg, Manitoba, Canada, May 12-15, 2002, pp. 1181-1187.

[33] C. Ordonez and E. Omiecinski, "Discovering Association Rules Based on Image Content", *Proceedings of IEEE Advances in Digital Libraries Conference (ADL 99)*, Baltimore, MD, USA, May 1999, pp. 38- 49.

[34] C. Ordonez, E. Omiecinski, and N. Ezquerra, "A Fast Algorithm to Cluster High Dimensional Basket Data", *IEEE ICDM 2001 Conference*, San Jose, CA, U.S.A, 2001.

[35] C. Ordonez, C. Santana, and L. de Braal, "Discovering Interesting Association Rules in Medical Data", *Proceedings of ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, Dallas, Texas, USA, May 2000, pp. 78-85

[36] W. Perrizo, Qin Ding, Qiang Ding, and A. Roy, "On Mining Satellite and other Remotely Sensed Images", *Proceedings of ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, May 2001, pp. 33-40.

[37] G. Ridgeway and D. Madigan, "Bayesian Analysis of Massive Datasets via Particle Filters", *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Canada, July 2002, pp. 5-13 Available at : http://www.i-pensieri.com/gregr/bayesdm.shtml

[38] RadiologyInfo, "Mammography", *Radiological Society of North America Inc. (RSNA)*, May 22, 2002. (Available at: http://www.radiologyinfo.org/content/mammogram.htm)

[39] S. Safavian and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology", *IEEE Transactions on Systems, Man, and Cybernetics* Vol. 21, No.3, 1991, pp. 660-674.

[40] A. Savasere, E. Omiecinski, and S. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases", *Proceedings of the 21st International Conference on Very Large Databases (VLDB'95)*, Zurich, Switzerland, September 1995, pp. 432-444.

[41] T. Tape, "Interpreting Diagonistic Tests", *University of Nebraska Medical Centre*, July, 2001. (Available at : http://gim.unmc.edu/dxtests/ROC2.htm)

[42] C. Vyborny, "Can Computers Help Radiologists Read Mammograms", *Radiology*, Vol. 191, 1994, pp.315-317.

[43] E. Witten, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Koufmann Publishers, 2000 pp. 65-67, 105-108.

[44] K. Woods, *Automated Analysis Techniques for Digital Mammography*, Ph.D. Thesis, Computer Science and Engineering Department, University of South Florida, December 1994.

[45] M. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, "New Algorithms for Fast Discovery of Association Rules", *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD97)*, California, USA, August 1997, pp. 283-286.

[46] O. Zaiane, M. Antonie, and A. Coman, "Mammography Classification by an Association Rule-Based Classifier", *Proceedings of the 3rd Intl. ACM SIGKDD Workshop on Multimedia Data Mining(MDM/KDD'2002)*, in conjunction with the 8th ACM SIGKDD, (Edmonton), Alberta, Canada, 17-19 July 2002, pp. 62-69.

[47] O. Zaiane, Z. El-Hajj, and P. Lu, "Fast Parallel Association Rule Mining without Candidacy Generation", *Proceedings of IEEE 2001 International Conference on Data Mining (ICDM 2001)*, November to December, 2001, pp. 665-668.