FINDING STRONG, COMMON AND DISCRIMINATING CHARACTERISTICS OF CLUSTERS FROM THEMATIC MAPS

by

YiQing Yu

B.Sc., HoHai University, 1988

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

July 1996

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Computer Science

The University of British Columbia

2366 Main Mall

Vancouver, Canada

V6T 1Z4

Date: July 16, 1996.

Abstract

The goal of the thesis is to discover knowledge on large spatial databases. Specifically, it discusses the problem of extracting patterns and characteristics of clusters from thematic maps. For instance, a characteristic of an expensive housing cluster may be that the average household income is over 100,000 dollars. Three key issues are addressed in this thesis. The first issue is how to measure the interest/utility values of characteristics. In order to accommodate different kinds of thematic maps, two measures are proposed and analysed: one based on entropy, and the other on standard deviation. Both measures satisfy all the desirable properties, and work effectively in practice. The second issue is how to extract patterns from multiple clusters. Two pattern extraction operations are defined. The *common*() operation is able to find the common characteristics among multiple clusters. The *different*() operations is capable of discovering the characteristics which distinguish one cluster from another. The third issue is how to compute characteristics utility measures and pattern extraction operations efficiently. Four different methods are proposed and evaluated for the computation of utility measures. Complexity and experimental results indicates that a technique based on isothetic rectangle intersections is the most efficient, outperforming all the other techniques such as a technique based on R-tree technique. For the problem of how to extract patterns of multiple clusters efficiently. Two different methods for pattern extraction are evaluated. The technique based on isothetic rectangle intersections again outperforms the technique based on R-tree, and can extract patterns from hundreds of thematic maps in seconds of CPU time.

Table of Contents

List of Tables

List of Figures

Acknowledgement

I would like to express my sincere appreciation and gratitude to my superior Dr. Raymond Ng for his guidance and encouragement which make this thesis possible. Thanks to Dr. Alan Wagner for his careful reading of this thesis and valuable comments.

I would also like to acknowledge the Department of Computer Science for providing the research environment. Thanks to Dr. Raymond Ng and Dr. Craig Boutilier for providing financial support in the form of research assistantship during my graduate studies.

Final thank goes to Christos Faloutsos and his group at the University of Maryland for providing the R-tree code.

Chapter 1

Introduction

1.1 Data Mining

1.1.1 Demand for Data Mining

It has been estimated that the amount of information in the world doubles every 20 months [1]. The corporate, government and scientific communities are overwhelmed with an influx of data that is routinely stored in databases. For example, even simple transactions, such as a telephone call, the use of a credit card, or a medical test, are typically recorded in a computer. This flood of raw data contains interesting and useful information. For instance, by analysing the database of a supermarket, one may discover certain knowledge such as people who buy whole wheat bread tend to buy 1% milk. This discovered knowledge may help in making marketing decisions and other purposes. This capacity of inducing hidden knowledge is beyond the scope of traditional database management systems (DBMSs).

Traditional DBMSs not only provide procedures for storing, accessing and modifying the data, but also offer simple operators for the deduction of information, e.g., inferring information that is a logical consequence of the information in the database. For example, the join operator, when applied to two relational tables where the first administrates the relation between employees and departments and the second describes the relation between departments and managers, infers a relation between employees and managers. Although all DBMSs support deduction of information, none supports induction.

Data mining aims to find hidden *patterns* that can be induced from a database. Such patterns can be formulated as a rule to predict the value of an attribute in terms of other attributes. For example, from the employee-department table and the department-manager table mentioned above, it might be inferred that each employee has a manager.

The need for data mining has been widely recognised and it is even ranked one of the most important topics of database research for the 90's. A combination of business and research interests has produced increasing demands for, as well as increased activity to provide, tools and techniques for mining hidden knowledge in databases. For instance, several banks have derived better loan approval and bankruptcy prediction methods by using patterns discovered in loan and credit histories. IBM has used data mining techniques to predict defects during the assembly of disk drives. American Airlines searches frequent flyer database to find its better customers, targeting them for specific marketing promotions. Some of the data mining applications will be briefly introduced in the next chapter.

1.1.2 Tasks of Data Mining

Although there are numerous data mining applications, they perform the following four classes of tasks: dependency analysis, class identification, concept description and deviation detection [2].

Analysing dependencies among data represents a fundamental class of discoverable knowledge. A dependency exists between two items if the value of one item can be used to predict the value of the other under certain probability. There have been many studies

conducted on dependency analysis [3][4][5], and it has been applied to many areas, such as astronomy [6], and the stock market [7].

Class identification is to group data into meaningful classes. This problem has been widely investigated in AI and Statistics literatures [8][9]. There are numerous clustering algorithms ranging from the traditional methods of pattern recognition [10] and mathematical taxonomy [11], to the conceptual clustering techniques developed in machine learning [12]. In the next chapter, a clustering algorithms [16], which is related to this thesis, will be introduced.

Concept description is to summarise interesting qualities about a class [13]. There are two broad types of intentional descriptions: *characteristic* and *discriminating*. A characteristic description describes what the data in a class share in common. A discriminating description describes how two or more classes differ. The methods for performing concept description will be discussed in this thesis.

Deviation detection is to find extreme patterns, such as anomalous instances that do not fit into standard classes. The main applications of deviation analysis are to filter out patterns which are significant enough to be of interest [14].

1.2 Spatial Data Mining

Recent studies on data mining have extended the scope of data mining from alphanumeric data to spatial data. This section is dedicated to introduce knowledge discovery on spatial databases.

1.2.1 Motivations for Spatial Data Mining

Spatial data are data related to *space*. The space of interest can be, for example, a geographic space such as the surface of the earth, or a man-made space like the layout of a VLSL design. Just like what happened to conventional data types, i.e., alphanumeric data, there have also been huge amount of spatial data accumulated from satellites, medical equipment, etc. For instance, earth observation satellites are expected to generate one terabyte (10^{15} bytes) of data every day. At a rate of one picture each second, it would take a person several years just to look at the pictures generated in one day. This motivates the study and development of knowledge discovery mechanisms for spatial data.

Spatial data mining is the extraction of interesting spatial patterns, general relationships between spatial data and non-spatial data, and other data characteristics not explicitly stored in spatial databases. Besides general challenges faced by data mining tasks, such as that the discovered knowledge should accurately portray the contents of the database, there are extra requirements on the spatial database mining algorithms. For example, efficiency is crucial for spatial database mining algorithms, due to the inherent complications of the spatial data types and spatial access methods. These issues will be discussed in Chapter 2.

Spatial data mining has been a very active research area in recent years. There have been valuable studies on spatial data mining, ranging from spatial association rules induction, to spatial data clustering, spatial data generalisation, etc. While the detailed discussion will be conducted in Chapter 2, the following is a brief introduction of the two research works which motivated the study of this thesis.

1.2.2 Applications of Spatial Data Mining

According to the classification discussed in Section 1.1.2, the work developed by W. Lu, et al. [15] can be categorised into concept description. A generalisation-based knowledge discovery mechanism is proposed to perform spatial merge and generalisation on spatial data. Concept hierarchies must be available in advance in order to guide the spatial merge and generalisation. This causes the following two problems. The first is that the quality of discovered knowledge relies crucially on the appropriateness of the hierarchy. The second is for many applications, it is very difficult to know *a priori* which hierarchy would be appropriate.

In [16], R. T. Ng and J. Han pointed out the problems with [15], and proposed a clustering algorithm CLARANS for finding interesting clusters on spatial data. The clusters produced by the clustering algorithms possess a very tight spatial characterisation. For example, the clustering structure found by CLARANS may lead to such a discovery that 80% of all mansions in Greater Vancouver have either a mountain or river view.

1.3 Problem Definition

Although CLARANS answers the question of what the clusters are, it fails to give the characteristic description of the clusters. While this problem can be solved by generalising the non-spatial attributes of the clusters as proposed in [15], again, it imposes the restriction that there exists pre-constructed concept hierarchies. In order to avoid the same problems bought by concept hierarchies , this thesis aims to provide concept description of the clusters without any dependence on concept hierarchies.

One way of finding the characteristics of the clusters is to correlate the clusters with various statistical census data. Census data are released and updated frequently by statistical and polling agencies such as Statistics Canada. Census data are excellent source for knowledge discovery, because they almost cover every aspect of the society such as population, education, occupation, dwelling, income and so on. Since census data is always related to certain geographic region, it is presented in the form of a *thematic map*, which is the map showing the spatial distribution of some theme. For example, the theme of the map shown in Appendix A is "the percentage of female labour force in managerial and administrative occupations".

Hence, this thesis works on discovering the characteristics of clusters through thematic maps. Furthermore, the clusters may have certain characteristics common to all clusters, or some characteristics discriminating from one with another. Methods are designed to extract common and discriminating characteristics of the clusters. The objective of this thesis is to solve the following three main problems.

Q1. How to measure the characteristics of a cluster?

Since there might be thousands of thematic maps, Measures should be designed to select out the maps which represent the characteristics of a cluster. How to define the measures is crucial. The definitions of measures determines whether the characteristics of a cluster captured by the measures are accurate or not. Also they set the foundation for the pattern extraction, which is explained in Q2. The problem of how to define the measures is not trivial,

due to the fact that there are different types of thematic maps, and different maps may have different units.

Q2. How to extract the common and discriminating characteristics from the multiple clusters?

Measures should be defined to perform concept description task, i.e. discovering the common and discriminating characteristics of the multiple clusters. Similar to the measures required in Q1, the measures performing concept description task should also be carefully studied, in order to obtain "good" results, i.e. the maps truly reflecting the common or different characteristics of the clusters.

Q3. How to provide scalable computation of the measures?

In order to study the characteristics of the clusters extensively, it is desirable to process as many maps as possible. The number of maps may grow from hundreds to thousands. This gives rise to the problem of how to provide scalable computations for the measures mentioned in Q1 and Q2. This problem is complicated by the fact that spatial indexing technique is not readily applicable, because the thematic maps usually do not have accompanying spatial indices.

1.4 Contributions

This thesis proposed the following solutions to the problems presented above.

A1. There are two kinds of thematic maps, namely ordinal and nominal thematic maps. While the details of the thematic maps will be introduced in Chapter 3, briefly speaking, that whether a thematic is ordinal or nominal depends on whether the census data described by the thematic map is ordinal or nominal. The existence of two different maps imposes the following two requirements. The first requirement is that two distinct measures are needed for these two different kinds of maps, due to the different nature of the two kinds of maps. The second requirement is that the two measures should uniformly deal with a mixture of two different kinds of maps, i.e. the results of the two measures should be normalised. After thoroughly analysing the desired properties of the measures, two measures based on statistical techniques are proposed. One is a entropy-based measure for nominal maps, and the other is a standard-deviation-based measure for ordinal maps. The two measures works together providing a uniform framework for knowledge discovery in the thematic maps.

A2. Although the measures performing concept description task deal with multiple clusters, it is desirable that the definition of the characteristics for multiple clusters should be consistent with the one for a single cluster. Following this principle, two measures, *common*() and *different*(), are proposed for discovering the common and discriminating characteristics of the clusters respectively. The experiments show that the common and discriminating characteristics extracted by the two measures are effective.

A3. The main purpose of spatial indexing technique is to support spatial selection. Since the thematic maps usually do not have accompanying indices, there is no obvious way to provide efficient algorithms for the measures mentioned above. One way to solve the problem is to build an index for each map at run time. As complexity analysis and experimental results show, building an index at run time is not an efficient method. While algorithm IR, which takes advantage of spatial object approximations, outperforms the algorithm based on spatial indexing and other proposed algorithms. IR provides efficiency with the ability of processing hundreds of maps in seconds.

1.5 Outline of Thesis

Related works are introduced in chapter 2. Thematic maps and the measures of cluster characteristics are discussed in chapter 3. The two operators for exacting common and discriminating characteristics of the clusters are introduced in chapter 4. The algorithms of computing the characteristic measures are presented and analysed in chapter 5. The experimental evaluations are conducted in chapter 6. The conclusion and future work are discussed in chapter 7.

Chapter 2

Related Work

2.1 Data Mining and Its Relationships with Other Disciplines

Machine learning technologies and statistical methods have been extensively used in knowledge discovery in database. The following two subsections will briefly discuss the relationships of knowledge discovery with machine learning and statistical methods.

2.1.1 Machine Learning

Machine learning, a field in artificial intelligence, is the automation of inductive learning processes [17]. During the learning phrase, the cognitive system observes its environment and recognises similarities among objects and events in this environment. It groups similar objects in classes and constructs rules that predict the behaviour of the inhabitants of such a class. Two types of learning techniques are of special interest: supervised learning and unsupervised learning. In supervised learning, an external teacher defines classes and provides the cognitive system with examples for each class. The system has to discover common properties in the examples for each class. This technique is also known as learning from examples. In unsupervised learning, the system has to discover the classes itself, based on common properties of objects. Hence, this technique is known as learning from observation.

Machine learning sets the foundation for data mining, because the tasks of data mining and machine learning are very similar except that for data mining the learning in done on a database. In despite of their similarities, the methods of machine learning could not be directly applied to data mining for the following reasons. Unlike machine learning which uses a small set of carefully selected laboratory data, data mining deals with real-world databases which are dynamic, incomplete, noisy, and much larger than typical machine learning data sets. Therefore, it is harder to discover descriptions in such an environment than in the ideal conditions found in machine learning. In addition, the size of the database makes most learning algorithms ineffective in the general case.

2.1.2 Statistical Methods

Statistical techniques also play an important role in data mining, especially in spatial data mining. There are large number of algorithms in statistics which offer a strong possibility of detecting generalised information from databases. Until now, statistical spatial analysis has been one of the most common techniques for analysing spatial data [18]. Statistical analysis encompasses an expanding range of methods which address many different spatial problems, such as image enhancement, pattern recognition and spatial clustering.

Although statistics provides the theoretical foundation for the problems of data analysis, a purely statistical approach is not enough. The main reason is that the results of statistical analysis are difficult to interpret, due to the fact that statistics is totally data driven, excluding the use of available *domain knowledge*, which can be used in all aspects of automated

discovery to make the discovered results more understandable. And also, the existing statistical algorithms become inefficient in performing data mining tasks when large data sets are involved.

2.2 Background Work in General Data Mining

Data mining tasks can be categorised into the following four classes, according to the data types they study: (1) temporal data mining; (2) transactional data mining; (3) alphanumeric data mining; and (4) spatial data mining. There have been many excellent studies on knowledge discovery in database. The following are the some representative research conducted on the first three categories. And the research on spatial data mining will be introduced in the next section.

The data mining task performed in [19] is on time-series databases. The purpose of [19] is to find time-sequences that are similar to a given sequence or to find all pairs of similar sequences. This capability has many applications in business and scientific fields. For example, this technique can be used to identify companies with similar patterns of growth. The work in [19] is extended from the studies in [20] and [21]. In [20], an indexing structure was proposed for fast similarity searches over time-series databases, assuming that the data sequences and query sequence were of the same length. The Discrete Fourier Transform (DFT) is used to map a time sequence to a frequency domain. This study was generalised in [21], where data sequences now could be of different lengths and the query sequence could be smaller than any of the data sequences. In [19], a new model of time sequence similarity is

proposed to address the limitations of [20] and [21]. One of the improvements is that the new model allows the amplitude of one of the two sequences to be scaled by any suitable amount and its offset is adjusted appropriately. Its matching system was applied to the U. S. mutual funds data and discovered several interesting matches. For example, it could find funds in the same category that have similar price behaviour .

Another kind of data mining tasks is performed on transaction data. Retail organisation are now able to collect massive amount of transaction data. Such data typically consists of the transaction date and other information associated with the transaction. The motivation of data mining on transaction data is to discover interesting patterns in this kind of data. For example, the study reported in [22] focuses on the problem of mining *sequential patterns* from a database, which stores video rental information. An example of a sequential pattern given in the paper is that customers typically rent "Star Wars", then "Empire Strike Back", and then "Return of the Jedi" in a video rental store. Three algorithms have been proposed to find sequential patterns. The algorithms guarantee that all sequential patterns of interest are discovered and have scale-up properties to the number of transactions on a customer sequence as well as the number of items on a transaction.

DBLEARN [23] is a system that was developed to perform data mining tasks on a relational database. DBLEARN integrates a machine learning paradigm with set-oriented database operations. The fundamental part of the system is the notion of *a concept hierarchy* which is used to control data generalisation. An example of a concept hierarchy is the

following: {biology, chemistry, computing,...}⊂science, {literature, music,...}⊂art, {science, art}⊂ANY(major). The system uses relational tables as knowledge structures. A tuple can be viewed as a logical formula, formed by the conjunction of its attribute-value pairs. A table, i.e., a set of tuples, can thus be viewed as a disjunction of these conjuncts. Hence, the starting point in the search space is a set of tables, and the aim is to generalise these tables for the concepts defined by the users. For example, if all the students majoring in biology and chemistry are excellent, it can be generalised that all students majoring in science are excellent. The system constructs probabilistic rules to deal with noisy data, but there is no technique to deal with missing attribute values.

2.3 Background Work in Spatial Data Mining

While all the research discussed in Section 2.2 studies non-spatial data, the following research, which studies spatial databases, is of special interest to this thesis.

The study reported in [15] is the extension of system DBLEARN [23] from a relational database to a spatial database. It developed a generalization-based knowledge discovery mechanism which performs spatial merge and generalisation on spatial data. Concept hierarchies must be available in advance to perform the spatial merge and generalisation. For instance, given regional temperature data, e.g., Harrison is 65.7°c, and Hope is 63.3°c, and high-level concept of temperature, e.g., moderately hot if temperature is between 50°c and 70°c, the generalisation of spatial data can be done by clustering spatial data objects according to their regions, e.g., Harrison and Hope are merged into south region of British Columbia. The

corresponding non-spatial attributes are merged until they reach the desired concept level, e.g., the temperature of Harrison and Hope are merged into a high-level concept of temperature which is moderately hot. The final result of generalisation for the above example can be that the temperature of south region of British Columbia is moderately hot.

As pointed out in [16], the method proposed in [15] suffers from a serious problem. It is difficult to know *a priori* which hierarchy will be appropriate. As a matter of fact, discovering this hierarchy may itself be one of the reasons to apply spatial data mining. The following example from [16] illustrates this problem. Suppose a spatial data mining request is to be performed on all the expensive houses in Greater Vancouver. A default spatial hierarchy to use may be the one that generalises streets to communities and to cities. However, if some of the expensive houses are spatially located along something such as a river that runs through many communities and cities, then the default spatial hierarchy would generate a statement such as the expensive houses are scattered in all the cities, and fails to explore the true characteristic that expensive houses are along a nature scene. In order to deal with this problem, a clustering algorithm CLARANS was proposed in [16].

Motivated by existing statistical clustering algorithms, CLARANS is based on randomised search to find the best clusters. A cluster is represented by its *medoid*, which is the most centrally located object within the cluster. CLARANS starts with a random node, and checks a specified number of neighbouring nodes. If it can not find a better node which has a lower cost, then the current node is declared to be a "local" minimum. CLARANS repeats to

search for other local minima, until a specified number of local minima are found. In [16], two algorithms, SD(CLARANS) and NSD(CLARANS), are proposed to enhance the data mining ability of [15] and [23]. In algorithm SD(CLARANS), CLARANS is first applied to find clusters, then DBLEARN is used to find the generalised characteristics of clusters through non-spatial attributes. While in algorithm NSD(CLARANS), DBLEARN is first performed to find a number of generalised tuples, then CLARANS is applied to find the clusters in the those generalised tuples. In the final step, the clusters are checked to determine whether they overlap with each other, if so, they can be merged, causing the corresponding tuples further generalised.

In [34], Ester et. al. developed a focusing technique which is able to extend CLARANS to cluster objects residing on disks. In [32], Zhang et. al. presented an algorithm, BIRCH (Balanced Iterative Reducing and Clustering) to improve the efficiency of clustering algorithms, including CLARANS.

As mentioned before, although CLARANS answers the question of what the clusters are, it does not determine the characteristics of these clusters . The study reported in [35] was intended to solve this problem. In [35], two tasks are achieved. First, it developed an algorithm CRH to discover the relationships between CLARANS clusters and geographic features, such as schools, golf courses, etc. Second, it proposed an algorithm GenCom, making use of concept hierarchies, to derive commonalities among the clusters. The work in [35] is

extended in [36], where an additional algorithm GenDis is provided, which is to find the features which discriminate one cluster from another.

Although the problem of discovering the characteristics of clusters is solved in [35][36], the methods requires a concept hierarchy. The solution proposed in this thesis is to discover the characteristics of the clusters by correlating the location of the clusters with thematic maps without the use of a concept hierarchy. A detailed description will be given in Chapter 3 and 4.

2.4 Spatial Data and Its Access Methods

Spatial database systems have become more and more important for industry and research. There are two fundamental problems associated with a spatial database system. The first problem is how to model real-life spatial objects, such as mountains and rivers. The second problem is how to efficiently access spatial data in a spatial database.

There are two major methods of representing spatial objects: *raster* and *vector*. In a raster representation, an object is represented by the pixels it occupies. In a vector representation, an object is specified by its geometry. For example, in a raster representation, a mountain is represented by all the pixels it covers. While in a vector one, it is specified as sequence of points at the boundary.

There have been many studies on the design of efficient spatial access methods. Most of the research show that efficient spatial access can be achieved by spatial indexing. Spatial

indexing organises space and the objects in such a way that only parts of the space and a subset of the objects need to be considered. Among the various indexing structures, the most well-known spatial indexing structures are Quadtree [37] and R-tree [28]. While Quadtree is used for raster data, which is usually applied in image processing area, the interest of this thesis is on R-tree and its variants.

An R-tree is a generalisation of a B-tree [30] to higher dimensions. The R-tree, shown in Figure 1, is a height-balanced tree, with each of its leaf node containing a point to a data object, and each internal node containing an *isothetic rectangle*, a rectangle encompassing all the rectangles in the level below. An R-tree has an upper and lower bound for the number of children of an interior node. The lower bound prevents the degeneration of trees and leads to an efficient storage utilisation. The upper bound can be derived from the fact that each tree node corresponds to exactly one disk page. Furthermore, the sibling nodes, i.e., nodes whose parent node is identical, may correspond to overlapping intervals.

The property of the R-tree facilitates the insertion and deletion of data objects, but it may lead to performance losses during search operations. In the case of range searches, the number of nodes to be inspected tends to be higher with overlaps. This problem led to the development of those techniques, R-tree variants, to minimise the overlap [38], the R^+-tree [39], where no overlaps are allowed unless the intervals are data intervals, and R*-tree [40], the most efficient member in the R-tree family. R*-tree requires fewer disk accesses for queries and less storage space for indexing structures than any other variants.

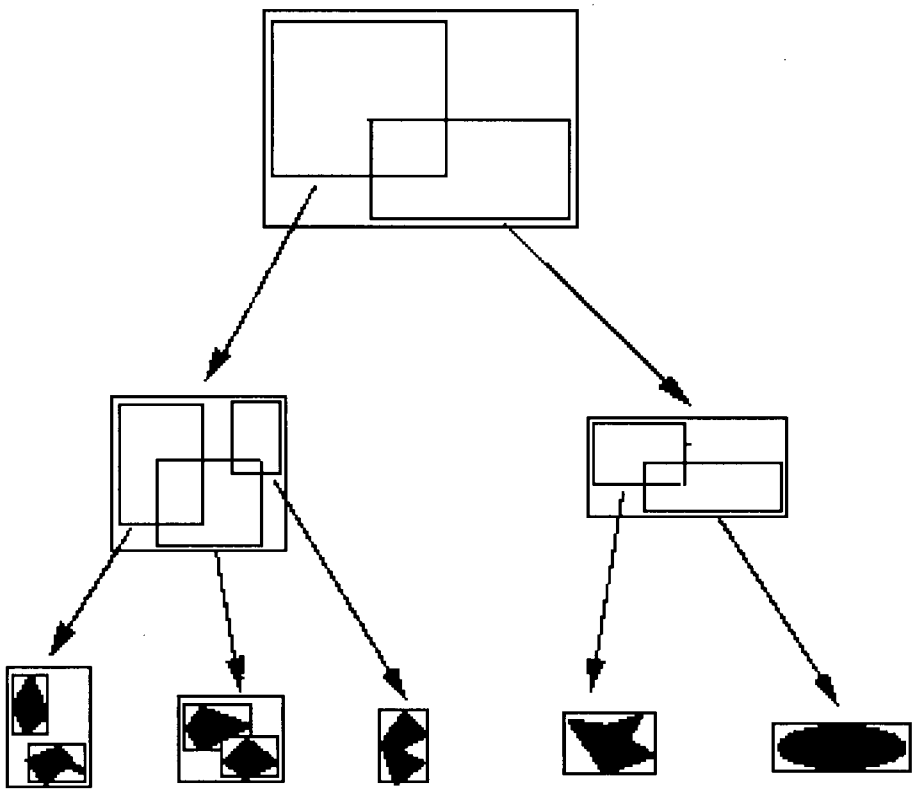


Figure 1 An R-tree with data in the leaves

Although using a spatial indexing structure is beneficial for general tasks performed on spatial objects, for particular tasks, spatial indexing may not be the most efficient method. For example, the divide-and-conquer algorithm developed in [41], is efficient in finding all pairwise intersections in a set of isothetic rectangles. The time required by the algorithm is $O(n\log n+k)$[1], where n is the number of given objects and k is the number of reported pairs. While in the case of R-tree method, the time of a tree construction is $O(n\log n)$, and the time for searching the intersected rectangles for n objects takes an additional $O(n\log n)$ steps.

[1] All logarithms in this thesis log to base 2.

Chapter 3

Characteristic Utility Measures

The main emphasis of this chapter is on how to discover the characteristics of a cluster though a large number of thematic maps. The variety of maps makes this problem complicated. As shown in the following section, the domain of the maps can be different, and even within the same domain, the unit of the maps can be different. In order to accommodate the various maps, different measures are needed to calculate the characteristics of a cluster.

3.1 Basics of Thematic Maps

"A map is a graphic representation of the *milieu*" [24], where milieu is used to broadly include all aspects of the physical and cultural environment. The variety of maps is extremely wide. In general, maps may be classified as either *reference maps*, or *thematic maps*. Reference maps are used to display both natural and man-made objects from geographical environment, such as the maps of landscape. A thematic map, as its name implies, explores the distribution of a certain subject, such as age, income, educational level, etc. Every thematic map is composed of two essential elements: a *base map* and a *thematic overlay*.

The base map defines the geographical region covered by the map. For example, the base map shown in Appendix A is the St. John's region of Canada. In order to represent a thematic map in a computer, the base map is regarded as a collection of polygons, each of which represents a sub-region of the base map. The polygons are not necessarily convex.

The thematic overlay describes the thematic data related to the base map. The thematic overlay is related to the base map by associating each polygon with a *label*, which is the thematic data of the corresponding sub-region. It is obvious that many polygons may have the same label. For instance, the thematic overlay in Appendix A describes the distribution of the labour force. And there are altogether seven labels in this map.

In general, the label of a thematic map are elements of either the *ordinal* or *nominal* domain. A domain is nominal if the elements in the domain are unordered. A domain is ordinal if the elements of the domain can be ordered. For example, the domain of political parties is nominal, while the domain of household income is ordinal.

An ordinal thematic map can be further categorised as either *single-value* or *interval*. A thematic map is single-value if each label of the map consists of a single value of the domain. In an interval thematic map, each label is a "sub-range" of the domain. For example, the labels of a single-value map describing education level can be "6years, 12years, 16years, 20years". The labels of an interval map describing household income can be "≤$20K, $20-40K, $40-60K, ≥$100K".

There are two kinds interval ordinal maps: *open range* and *closed range*. The range of an interval map is open if there is at least one label whose range boundary is undefined. The range of an interval is closed if the boundaries of all the labels are closed. While the map shown in Appendix A is an example of a closed range thematic map, the above example of household

income is an open range map, because it has two labels,"≤$20K" and "≥$100K", whose range boundary are not defined:

3.2 Characteristics of Clusters

A cluster, produced by some clustering algorithms such as CLARANS, is a collection of 2-D points. The points of a cluster are located in one or more polygons of the map. Since each polygon has a label, each point is correlated to a label.

The *characteristics* of a cluster are those maps where the distributions of the cluster's points are *centralised* with respect to the labels. The more centralised the distributions, the stronger the characteristic of the cluster will be. Suppose there is a thematic map which displays the information of household income. There are five labels in the map, which are {$10K, $20K, $30K, $40K, $50K}. The following three examples will explain the definition of the characteristics of the clusters.

Example 1.

If all the points have the same labels, this distribution is most centralised and the map is a characteristic of the cluster. On the other hand, if all the points of a cluster are evenly distributed in all five labels, i.e., 20% of points is $10K, 20% of the points is $20K, etc., this distribution is most chaotic, so that this map is not a characteristic of the cluster.

Example 2.

Consider the following two cases: in the first, 100% of the points are labelled 50K; in the second, 50% of the points are labelled \$30K, and 50% are labelled \$50K". It is desirable to say that the map is a stronger characteristic of the cluster in the first case than the second one, because the distribution of the cluster in the first case is more centralised than the one in the second case.

Example 3.

Compare the following two situations: the first is "50% of the points are labelled \$40K and 50% are labelled \$50K", the second is "50% of the points are labelled \$10K, and the rest are labelled \$50K". Because the domain of this map is ordinal, there exists an ordering between the elements. Thus, since \$40K is closer to \$50K than \$10K to \$50K, the distribution of the cluster in the first case is more centralised than the one in the second case. So, it is desirable to say that the map is a stronger characteristic of the cluster in the first case than the second one. Note that if the domain of the map is nominal, i.e., there is not ordering between the elements, the utility value for these two situations should be the same.

In order to facilitate the description of the relationship between a cluster *Cl* and a thematic map *M*, the following notation is defined:

A characteristic set, denoted by *chr*(*Cl, M*), is a collection of triplets: $<PO_1, L_1, p_1>$, ..., $<PO_n, L_n, p_n>$, where:

- $PO_1, \ldots, PO_n$ are polygons in M;
- $L_1, \ldots, L_n$ are the labels of $PO_1, \ldots, PO_n$ respectively and
- for all $1 \leq i \leq n$, p_i is the fraction of points in Cl that are contained in PO_i, $0 \leq p_i \leq 1$, and $(p_{1+} \ldots p_n) \leq 1$.

3.3 Utility Measures

In order to discover the maps which are the characteristics of a cluster, measures are needed to calculate the distribution of the cluster with respect to the labels of a thematic map. The measures assign a *utility value* to each of the thematic maps. The higher the utility value of a thematic map, the stronger the map is a characteristic of the cluster. These measures are referred to as *utility measures*.

The essential requirement for a utility measure is that it should correctly reflect the definition of the characteristics of a cluster. This implies that the utility value of a thematic map calculated by the measures should correctly reflect the phenomena described in section 3.2.

The measures should also work with all the categories of maps discussed in Section 3.1, namely nominal maps, single-value maps, open range and closed range interval maps.

Furthermore, even within the same category, different maps may have different units. It is impossible to have a single utility measure for all maps. Different measures should be proposed to suit different kinds of maps. In addition, the utility values assigned by the measures should be normalised, so that the utility values of the different maps can be compared with each other.

In the next few sections, utility measures are proposed and analysed for each kind of thematic maps.

3.4 An Entropy-based Measure for Nominal Thematic Maps

In information theory, entropy is a measure of the disorder of a system. The definition of entropy is following [25]: for a random variable $I(x_i)$, its expected value is:

$$E\{I(x_i)\} = H(X) = \sum_{i=1}^{M} p(x_i) \log p(x_i) \qquad (1)$$

where $p(x_i)$ is the probability of system in ith state, M is the number of states of the system. $H(X)$ is the entropy of the probability distribution of $p(x_i)$, where $\sum_{i=1}^{M} p(x_i) = 1$.

From the definition, entropy has the following properties [33]:

1. $H(X) \geq 0$.
2. $H(X) = 0$ if and only if all of the probabilities are zero except for one, which must be unity.
3. $H(X) \leq \log M$.

4. $H(X) = \log M$ if and only if all the probabilities are equal so that $p(x_i)=1/M$ for all i.

As a moderate modification to the standard entropy measure, the following is the proposed entropy-based measure.

Definition 1. Let the Characteristic set be $\{< PO_1, L_1, p_1>, \ldots, <PO_n, L_n, p_n>\}$, the entropy-based measure is:

$$CU^{ent} = 1 - \frac{\sum_i p_i * \log p_i}{\log(|D|)} \qquad (2)$$

where D is the domain of the thematic map, $|D|$ is the cardinality of D.

From the properties of entropy, the following are true about CU^{ent}:

1. $0 \le CU^{ent} \le 1$.
2. $CU^{ent} = 1$ if and only if $n=1$ and $p_1=1$;
3. $CU^{ent} = 0$ if and only if $n = |D|$ and for all $1 \le i \le n$, $p_i=1/n$.

If the characteristic set is $\{<PO_1, L_1, 1>\}$, it means that all the points in the cluster correspond to the same label L_1. Thus, as discussed in Section 3.2, this utility value of the map should have the highest value, one. If the characteristic set is $\{< PO_1, L_1, 1/n>, \ldots, <PO_n, L_n, 1/n>\}$, which corresponds to the most chaotic distribution of the points, the utility value deserves to be the lowest, which is zero. Thus, the measure correctly reflects the phenomenon described in Example 1 of Section 3.2. It can be easily verified that the measure also correctly depicts the phenomenon in Example 2. For Example 3, the utility values for both cases are the

same. As discussed in Example 3, this result is correct of nominal maps, but fails for ordinal maps.

Thus, the entropy-based measure is applicable only for nominal maps. In order to deal with the problem with the ordinal maps, the following standard-deviation-based measure (referred to as STD-based measure hereafter) is proposed.

3.5 A STD-Based Measure for Ordinal Thematic Maps

As mentioned in Section 3.1, there are two kinds of ordinal thematic maps: single-value and interval. In this section, a STD-based measure is proposed to suit the single-value maps. How to deal with interval maps will be discussed latter.

The standard deviation [26] is the most frequently used measure of variability. It can be considered as the average of the absolute deviations of observations from the mean. The standard deviation of n variables v_i is:

$$\sigma = \sqrt{\sum_{i=1}^{n} (v_i - \mu)^2 * p_i} \qquad (3)$$

where p_i is the probability of variables v_i, and μ is the mean of the n variables, which is $\sum_{i=1}^{n} v_i * p_i$.

From the definition, the standard deviation has the following properties:

1. $\sigma \geq 0$.
2. $\sigma = 0$ if and only if the probabilities are zero except for one, which must be unity.
3. $\sigma \leq (\max(v_i) - \min(v_i)) / 2$ [2].

As a moderate modification to the standard deviation measure, the following is the proposed standard deviation measure.

Definition 2. Let the characteristic set be $\{< PO_1, L_1, p_1>, \ldots, <PO_n, L_n, p_n>\}$. For all $1 \leq i \leq n$, L_i consists of a single value v_i. The STD-based measure is:

$$CU^{std} = \begin{cases} 1 - \dfrac{\sqrt{\sum_i (v_i - \mu)^2 * p_i}}{[\max(D) - \min(D)] / 2} & \text{if } \max(D) \neq \min(D) \\ 1 & \text{if } \max(D) = \min(D) \end{cases} \quad (4)$$

where D is the domain of the thematic map.

From the property of standard deviation, the following are true about CU^{std}:

1. $0 \leq CU^{std} \leq 1$.
2. $CU^{std} = 1$ if and only if n=1, max(D)=min(D).
3. $CU^{std} = 0$ if and only if n=2, v_1=min(D), v_2=max(D), and $p_1 = p_2 = 0.5$.

[2] From the definition of STD, the most chaotic situation happens when the points evenly distributed in the two extremes. In this case, the value of STD is (max(v_i)-main(v_i))/2.

The second property occurs when all the cluster points give a uniform value. In this case, the map is assigned to the highest utility value, one. The third property corresponds to the most "chaotic" situation, where the utility value is zero. For Example 2 and Example 3 discussed in Section 3.4, the utility value for the first case is now higher than the value for the second case. Thus, this STD-measure is applicable to ordinal maps.

3.6 A STD-Based Measure for Interval Thematic Maps

So far, the utility measure defined in Equation (4) can only be applied to ordinal thematic maps whose labels consist of single values. The labels of interval thematic maps are numerical ranges. Techniques for closed and open range interval maps will be discussed in the next two sub-sections.

3.6.1 Closed Range Interval Maps

Given a characteristic set $\{< PO_1, L_1, p_1>, \ldots, <PO_n, L_n, p_n>\}$ where L_i is the range $[a_i, b_i]$, the problem is to find appropriate single value v_i to represent the mean and standard deviation of $[a_i, b_i]$. There exists a standard strategy to generalise standard deviation from single values to range values [26]. The standard strategy is to treat all the values v_i as if they were the average value of the range, i.e., (a+b)/2. More specifically, the following adjustment is needed to adapt Equation (4):

Definition 3. Given a characteristic set $\{< PO_1, L_1, p_1>, \ldots, <PO_n, L_n, p_n>\}$, for all closed interval labels L_i:

$$v_i = \frac{a_i + b_i}{2} \qquad (5)$$

where L_i is the closed range of $[a_i, b_i]$ for all $1 \leq i \leq n$.

Thus the combination Equation (5) with Equation (4) is the utility measure for a closed range interval maps.

Aside: An improved estimate can be obtained by subtracting the *Sheppard's correction* from the standard deviation as calculated by the above standard strategy [26]. However, the Sheppard's correction is not included for the following two reasons. The first reason is that Sheppard's correction is defined for ranges with equal width which is not always true for the thematic maps. The second reason is that the Sheppard's correction may not be applicable for distributions that are not normal.

3.6.2 Open Range Interval Maps

The open range interval maps are the maps whose two end-points of the labels are not specified. For example, "≥100K" may be one end of the labels for the map describing average household income. Heuristic algorithms will be proposed to estimate the average value of an open range map.

Since a majority of phenomena described by thematic maps is normally distributed [27], it is reasonable to assume that the *observed* distribution of a thematic map is a normal distribution. The observed distribution of a map can be easily obtained by using the area covered by each label in the map.

Once the observed distribution is available, the following heuristic algorithm can be used to estimate the average of an open range label of the form '≤c'.

Heuristic Algorithm 1 Let X denote the random variable of the observed distribution and Z denote a standard-normal random variable.

1. Set the mean μ of X to be the median of the observed distribution.
2. Set the standard deviation of X to:

$$\sigma = \frac{iq}{1.35} \qquad (6)$$

where iq denotes the interquartile range of the observed distribution

3. From the observed distribution, find the probability p such that $p = prob(X \leq c)$.
4. By using the standard normal curve, find Z_p such that:

$$prob(Z \leq Z_p) = p/2 \qquad (7)$$

5. Set the average value v_c of the open range label '≤c' to:

$$v_c = \begin{cases} \mu - Z_p * \sigma & \text{if } 0 \leq \mu - Z_p * \sigma \leq c \\ D & \text{otherwise} \end{cases} \qquad (8)$$

where D denotes a reasonable value that should be used, if the estimated value $\mu - Z_p * \sigma$ is inappropriate.

For a normal distribution, its mean is exactly the same as its median. This explains Step (1) of the algorithm. The interquartile range is used to find the standard deviation σ. The interquartile is defined as the difference between the first quartile Q_1 and the third quartile Q_3, where $prob(X \leq Q_1)=0.25$ and $prob(X \leq Q_3)=0.75$ [26]. For a standard normal distribution, it is true that $prob(Z \leq (\mu - 0.675\sigma)) = 0.25$ and $prob(Z - (u + 0.675\sigma)) = 0.75$. In other words, the interquartile range is the same as $(\mu + 0.675\sigma) - (\mu - 0.675\sigma) = 1.35\sigma$. This explains Equation (6). Step (3) is to find the percentile that corresponds to $X \leq c$. The average of the label $X \leq c$ is approximated by the median of the label, which is the value that splits the percentile corresponding to $X \leq c$ in two halves. In a standard normal curve, this value is Z_p as defined in Equation (7). In Equation (8), the average value v_c is obtained by translating the Z_p value from the standard normal distribution to the actual normal distribution with the parameters μ and σ. If the value of $\mu - Z_p * \sigma$ is negative or is larger than c, then v_c is simply set to D which is a reasonable estimate, such as $c/2$.

This following is an example. Consider the average household income example again. Suppose the first quartile, the mean/median and the third quartile are 40K, 60K and 80K respectively. Then σ is 29.6K. Suppose further that $prob(income \leq 20K)=0.1$. From the standard normal curve, the Z_p value is found to be 1.64. Thus, the average value of v_c is estimated to be 11.5K.

A similar algorithm, Heuristic Algorithm 2, can be used to estimate the average value of the open label '$\geq c$', The modifications to Heuristic Algorithm 1 are Step (3), (4) and (5). The following is Heuristic Algorithm 2:

Heuristic Algorithm 2 Let X denote the random variable of the observed distribution and Z denote a standard-normal random variable.

1. Set the mean μ of X to be the median of the observed distribution.
2. Set the standard deviation of X to:

$$\sigma = \frac{iq}{1.35} \qquad (10)$$

   where iq denotes the interquartile range of the observed distribution

3. From the observed distribution, find the probability p such that $p = prob(X \geq c)$.
4. By using the standard normal curve, find Z_p such that:

$$prob(Z \leq Z_p) = 1 - p/2 \qquad (11)$$

5. Set the average value v_c of the open range label '$\leq c$' to:

$$v_c = \begin{cases} \mu + Z_p * \sigma & \text{if} \quad c \leq \mu + Z_p * \sigma \\ D & \text{otherwise} \end{cases} \qquad (12)$$

where D denotes a reasonable value that should be used, if the estimated value $\mu + Z_p * \sigma$ is inappropriate.

The explanation for Heuristic Algorithm 2 is omitted for brevity, due to the fact that the principles of the two algorithms are just the same. Although the observed distribution assumed in the algorithms is not the same as the distribution of the characteristic set, the experimental results discussed in Chapter 5 show that the algorithms work well in practice.

3.7 Summary

In this chapter, the characteristics of a cluster and its computation are discussed. The characteristics of a clusters are defined to be those maps where the distribution of the cluster are centralised. The utility value of a thematic map is the measure of the centrality of the distribution, i.e., the strongness of the characteristic of cluster. The higher the utility measure, the stronger the map is a characteristic of the cluster. In order to accommodate different kinds of maps, two basic measures are proposed to compute the utility values of the maps. An entropy-based measure works for nominal maps, while a standard-deviation-based measure is applied to ordinal maps. Some extra steps should be taken before employing the STD-based measure to interval maps. The results of the two utility measures are comparable, because the utility value produced by the two measures are all in the range of [0,1]. Thus, all the requirements imposed to the utility measures are all satisfied.

Chapter 4

Pattern Extraction Operations

In the previous chapter, the utility measures have been discussed. In this chapter, two pattern extraction operations *common*() and *different*() will be introduced. The operation *common*($Cl_1,\ldots, Cl_n$) is to find strong characteristics that are common to all n clusters $Cl_1,\ldots, Cl_n$. The operation *different*(Cl_1, Cl_2) is to find the characteristics that distinguish Cl_1 from Cl_2.

4.1 Find Common Characteristics of Multiple Clusters

The *common*() operation is intended for the extraction of common characteristics exhibited by multiple clusters. But if only one cluster is provided as input to the operation, the operation can be used to select maps whose utility values exceeded a user-defined threshold. In this section, the operation on one cluster will be first discussed.

4.1.1 Finding Strong Characteristics of One Cluster

As defined in Chapter 3, a characteristic set is defined as $chr(Cl, M)=\{<PO_1, L_1, p_1>,\ldots, <PO_n, L_n, p_n>\}$. As shown in Equation (2) and Equation (4), the polygons $PO_1,\ldots PO_n$ are not required in the computation of utility values. Here, a new notion is introduced to simplify the notation of $chr(Cl, M)$ by the following two steps. The first is to remove the notation of the polygons, the second is to group classes that have the same label together. Thus, the new notation, *chr_sum*(Cl,M), summarises the information carried in $chr(Cl,M)$ into the form: $\{<D_1,q_1>,\ldots<D_m,q_m>\}$, where for all $1\leq j \leq m \leq n$ and $1\leq k \leq n$:

- $group_j=\{k \mid C_k=D_j\}$, and
- $q_j=\sum_{k\in\ group_j} p_k$.

For example: If $chr(Cl,M)=\{<PO_1, [0, 50K], 0.2>, <PO_2, [50K, 80K], 0.3>, <PO_3, [0, 50K], 0.5>\}$, then $chr\_sum(Cl,M)$ is the set $\{<[0, 50K], 0.7>, <[50K, 80K], 0.3>\}$.

The end-users may determine whether the characteristic of a cluster exhibited in a map is strong or not by providing a threshold. If the utility value of a map exceeds the threshold, the map is considered to be a strong characteristic of the cluster. Thus, the *common*() operation takes two inputs: a cluster *Cl*, and a user-defined threshold *ITh*, which is in the range of [0,1]. The output is the set $\{chr\_sum(Cl,M) \mid M$ is a thematic map and the characteristic-utility of $chr\_sum(Cl,M) \geq ITh\}$. From now on, *M* is said to be a strong characteristic of *Cl* if the utility value of $chr\_sum(Cl,M)$ exceeds the threshold *ITh*.

The following is a sample run of the $common(Cl_1, ITh)$ operation on a collection of thematic maps. The input threshold is 0.9.

** There are 2 maps that passed the threshold 0.9. The maps are arranged in the descending order of their utility values.

* Map 1
* Content of map: Average gross rent
* Utility: 1.000
* Distribution:
  100% of all points are in range [$700, $900].

* Map 2
* Content of map: Unemployment rate
* Utility: 0.938
* Distribution:
  90% of all points are in [5%, 7%].
  10% of all points are in [9%, 11%].

Figure 2: Finding Strong Characteristics of One Cluster

4.1.2 Finding Strong Characteristics Common to Multiple Clusters

The strong characteristics common to all input clusters have two meanings. The first one is that if a map is a strong characteristic of all the clusters, it is a strong characteristic of each cluster. In other words, only the maps representing strong characteristics for each and every cluster will be processed by $common(Cl_1, \ldots Cl_n)$. A map M is discarded if there exists a cluster Cl_i such that the characteristic-utility of $chr\_sum(Cl_i,M)$ is below the user-defined threshold ITh.

The second meaning is that a map is a strong characteristic common to all input clusters. Although there are many ways to define the common characteristic of multiple clusters, in this thesis, it is defined as the map which is a strong characteristic of the cluster formed by merging all clusters together. The benefit of such a definition is that it is consistent with the definition of the characteristic for a single cluster.

To merge the clusters, first all classes with the same label in each $chr\_sum(Cl_i,M)$ are grouped together by adding up their percentages weighted by the number of points in the clusters. Next, the percentages acquired from the first step are re-scaled so that they sum to 100.

More specifically, let the characteristic summaries for all n clusters be: $chr\_sum(Cl_1, M) = \{<D_{1,1}, q_{1,1}>,\ldots,<D_{1,m_1}, q_{1,m_1}>\},\ldots, chr\_sum(Cl_n, M)=\{<D_{n,1}, q_{n,1}>,\ldots,<D_{n,m_n}, q_{n,m_n}>\}$. The combined characteristic for all clusters, denoted by $com\_chr\_sum(\{Cl_1,\ldots,Cl_n\}, M)$, is the set: $\{<D_1, q_1>,\ldots,<D_m, q_m>\}$, where for all $1\le j\le m$, $1\le u\le n$, and $1\le v\le m_u$:

- $group_j=\{<u,v> \mid D_{u,v}=D_j\}$, and
- $q_j=(\sum_{<u,v>\in group_j} q_{u,v} * s_u) / (\sum_{u=1}^{n} s_u)$

where $s_1,\ldots s_n$ represent the number of points in cluster $Cl_1,\ldots,Cl_n$ respectively.

The following example illustrates the merging procedure. Suppose

- $chr\_sum(Cl_1, M)=\{<C_1, 0.8>, <C_2, 0.2>\}$,
- $chr\_sum(Cl_2, M)=\{<C_1, 0.1>, <C_2, 0.6>, <C_3, 0.3>\}$, and
- $chr\_sum(Cl_3, M)=\{<C_2, 0.4>, <C_3, 0.6>\}$.

Assume that the three clusters Cl_1, Cl_2 and Cl_3 have 10, 20 and 30 points respectively. Then $com\_chr\_sum(\{Cl_1,Cl_2,Cl_3\}, M)$ is $\{<C_1, (0.8*10+0.1*20)/(10+20+30)>, <C_2, (0.2*10+0.6*20+0.4*30)/(10+20+30)>, <C_3,(0.3*20+0.6*30)/(10+20+30)>\}$.

Having computed the combined characteristic, the appropriate utility measure can be used. For the example above, if the classes C_1, C_2, C_3 are nominal, then Equation (2) of the entropy-based measure should be used. Otherwise Equation (4) of the standard-deviation-based measure, combined with Equation (5) if necessary, should be applied.

In determining whether a combined characteristic is strong or not, there is no particular reason why *ITh* for individual characteristics can not be used. Nevertheless, it is more flexible to allow the end-users providing a different threshold, *CTh*, for combined characteristics. Thus, formally speaking, a map M is a strong characteristic of all n clusters, if it satisfies the following conditions:

- for all $1 \leq u \leq n$, the utility of $chr\_sum(Cl_u, M) \geq ITh$, and
- the utility of $com\_chr\_sum(\{Cl_1, \ldots, Cl_n\}, M) \geq CTh$.

Figure 3 shows the output of a sample run of the $common(Cl_1, Cl_2)$ operation, with the individual threshold, ITh=0.8, and the combined threshold, CTh=0.6. Two maps are the common characteristics in this example. they are displayed in descending order of their combined utility values.

** There are 2 map that passed the individual threshold 0.8 and the
** combined threshold 0.6.

Map 1
Content of Map: Annual Household Income
Combined Utility: 0.8
Distribution of the first cluster:
100% of all points are in [$20,000, $50,000].
Distribution of the second cluster:
90% of all points are in [$20,000, $50,000].
10% of all points are in [$50,000, $10,000].

Map 2
Content of Map: Average Education Level
Combined Utility: 0.65
Distribution of the first cluster:
30% of all points are in [6years, 12years].
70% of all points are in [12years, 16years].
Distribution of the Second Cluster:
75% of all points are in [12years, 16years].
25% of all points are in [16years, 21years].

Figure 3: Finding Common Characteristics of Two Clusters

4.2 Finding Discriminating Characteristics of Two Clusters

The *different*(Cl_1, Cl_2) operation finds characteristics of Cl_1 and Cl_2 that distinguish one cluster from the other. In order to be a discriminating characteristic of two clusters, a map should satisfy the following two requirements. First, the map must be a strong characteristic of both clusters. The second one is that the strong characteristics derived from the same thematic map for both clusters are *sufficiently different*. For nominal and ordinal maps, the meanings of sufficient different are not the same, which will be given in the next section.

For maps with nominal domains, because there is no ordering relationships between the classes, it is enough to say that two clusters are sufficiently different if the two sets of class labels for the two clusters do not intersect. More formally, if the characteristic summaries of

the two clusters are: $chr\_sum(Cl_1, M)=\{<C_1, q_1>,\ldots,<C_m, q_m>\}$, and $chr\_sum(Cl_2, M)=\{<D_1, p_1>,\ldots,<D_k, p_k>\}$, a nominal map M is a strong discriminating characteristic of Cl_1 and Cl_2, if

- for $u=1$ or 2, the utility of $chr\_sum(Cl_u,M)\geq ITh$, and
- $\{C_1,\ldots,C_m\}\cap\{D_1,\ldots,D_k\}=\varnothing$, i.e., the sets of class labels do not intersect,.

For ordinal maps, requiring the non-overlapping of the two sets of class labels may not be sufficient. For example, suppose M is the thematic map about average household income with the domain from 10K to 200K. Let $chr\_sum(Cl_1, M)$ be {<[80K, 90K], 0.9>, <[90K, 100K], 0.1>}, and $chr\_sum(Cl_2, M)$ be {<[70K, 75K], 1>}. The set of class labels for the first cluster corresponds to the range [80K, 100K], while the set of labels for the latter corresponds to [70K, 75K]. Even though the two ranges do not overlap, it is questionable whether the two ranges are sufficiently different, given that the domain is the much larger range of [0,200K].

The above example suggests that for ordinal domains, it is reasonable to consider two factors in determining whether two ranges are sufficiently different. They are the *gap* between the two ranges, and the size of the domain of the map. As for the definition of the gap, there are at least two options. One way is to define the gap by the end-points of the ranges that correspond to the sets of class labels. The problem with this definition is that it ignores the distribution of the cluster points in the ranges. Thus, similar to the use of standard deviation in defining utility values, the gap is determined by the difference between the mean values of the cluster points. Formally, if the characteristic summary is $\{<C_1, p_1>,\ldots,<C_m, p_m>\}$, then the mean value is $\mu=\sum_i p_i * v_i$, where v_i the value of class C_i. For a given ordinal map M

with domain D, and two clusters Cl_1 and Cl_2 with means μ_1 and μ_2 respectively, the discriminating utility of M for Cl_1 and Cl_2 is given by:

$$DU = \frac{|\mu_1 - \mu_2|}{\max(D) - \min(D)} \qquad (13)$$

The numerator give the gap between the ranges, while the denominator specifies the size of the domain. The larger the ratio between the gap and the size of the domain, the stronger the discriminating characteristic will be.

The end-users may determine whether a map is a discriminating characteristic or not by providing a gap threshold, *GTh*. Thus, an ordinal map M is a strong discriminating characteristic of Cl_1, Cl_2, if it satisfies the following three conditions:

- for $u = 1$ or 2, the utility of $chr\_sum(Cl_u, M) \geq ITh$,
- $\{C_1,\ldots,C_m\} \cap \{D_1,\ldots,D_k\} = \varnothing$[3] , i.e., the sets of class labels do not intersect, and,
- $DU \geq GTh$, i.e., the discriminating utility of M for Cl_1, Cl_2 exceeds a gap threshold *GTh*.

where DU is defined in Equation (13) and $0 \leq GTh \leq 1$.

Figure 4 shows the output of a sample run of the different(Cl_1,Cl_2) operation, with the individual threshold 0.9 and the gap threshold 0.6. Average gross rent is the only discriminating characteristic in this example. If multiple discriminating characteristics are found, they are displayed in descending order of their discriminating utility values.

[3] If C_i and D_j are intervals, the formula should be $(\cup C_i) \cap (\cup D_j) = \varnothing$, where $1 \leq i \leq m$, $1 \leq j \leq k$.

```
**      There are 1 map that passed the individual threshold 0.9 and the
**      discriminating threshold 0.6.

Map 1
Content of Map:         Average gross rent
Discriminating Utility: 0.65
Distribution of the first cluster:
        100% of all points are in [$500, $700].
Distribution of the second cluster:
        90% of all points are in [$1200, $1400].
        10% of all points are in [$1000, $1200].
```

Figure 4: Finding Discriminating Characteristics of Two Clusters

4.3 Summary

In this chapter, two operations for pattern extraction have been defined. *Common*() extracts the common characteristics of a cluster set. A map is a common characteristic of the multiple clusters, if it is a strong characteristic for each of the clusters and the cluster formed by merging the clusters together. *Different*() extracts the discriminating characteristics of two clusters. A map is a discriminating characteristic of the two clusters, if the map is a strong characteristic of each of the two clusters, and the characteristics of the two clusters shown in the map are sufficiently different.

Chapter 5

Computation of Utility Measures and Pattern Extraction

In the previous chapters, the methods of computing the characteristics of clusters and the pattern extraction operations were discussed. In this chapter, the problem of how to efficiently compute these methods will be discussed.

5.1 Computation of Utility Measures

According to the two utility measures defined in Chapter 3, for a characteristic set $chr(Cl, M)=\{<PO_1, L_1, p_1>,\ldots,<PO_n, L_n, p_n>\}$, the key point in computing the measures is to calculate the percentage p_i of points in a cluster Cl that are contained in polygon PO_i. In other words, the problem is to find all the polygons that contain at least one point of the cluster. Note, the polygons are not necessarily convex.

Retrieving the desired polygons is typically a spatial selection problem which could be efficiently computed by a spatial indexing method, such as R-trees [28]. However, in practice, thematic maps are not likely to have accompanying indices. In the absence of spatial indices, the problem encountered here is how to compute the measures efficiently. In the remainder of this chapter, four different methods are proposed and analysed.

5.2 An algorithm Based on Pointwise Containment

In order to find all the polygons that contain at least one point of a cluster, the most obvious way is to check every point of the cluster with every polygon in a map for containment. This idea is reflected in the following algorithm, algorithm PC (standing for "Point Containment").

Algorithm PC

1. For each point *pt* in a cluster *Cl*,
2. For each polygon *PO* in map *M*,
3. If point-in-polygon(*pt*, *PO*) is true, increment the counter for *PO*.

In PC, the procedure point-in-polygon(*pt*, *PO*) [31] is to test whether the given point *pt* is contained in a specified polygon *PO*. If *PO* is a non-convex polygon with k edges, the time complexity of the above procedure is $O(k)$ [31]. Thus, if there are n points in cluster *Cl* and m polygons in map *M*, the computational complexity of PC is $O(kmn)$.

5.3 An Algorithm Based on Convex Hull Intersections

In PC, if the number of polygons that overlap with *Cl* is much less than the total number of polygons in a map, then most of the computations of point-in-polygon(*pt*, *PO*) are unnecessary. In this section, algorithm CH (standing for "Convex Hull") is proposed to improve the efficiency of PC by applying a filtering step before testing every point in a cluster.

First, a convex hull is constructed to contain all the points in a cluster. Those polygons whose convex hulls do not intersect with the convex hull containing the points of a cluster can be safely removed, because a polygon's convex hull always contains the polygon. Algorithm CH is the following:

Algorithm CH

1. Compute the convex hull H_{cl} of cluster Cl, and initialise set S to empty.
2. For each polygon PO in map M:
   (a) Compute its convex hull H_{po};
   (b) If convex-hull-intersect(H_{cl}, H_{po}) is true, add PO to S.
3. Apply PC to the polygons in S.

Step (1) of CH uses $O(n\log n)$ steps to compute the convex hull of n points [31]. Similarly, Step (2a) takes $O(k\log k)$ for each polygon with k edges. In Step (2b), convex-hull-intersect() takes time linear to the number of edges of the hulls in the worst case [31]. Thus, the complexity of Step (2b) is $O(k+n)$. By combining Step (2a) and (2b), the time complexity of Step (2) is $O(mn+km+mk\log k)=O(mn+mk\log k)$, where m is number of polygons in M. Finally, if β_H is the fraction of polygons that overlap with H_{cl}, then, as shown in Section 5.1.1, the complexity of Step (3) is $O(\beta_H kmn)$.

It follows from the above analysis, the complexity difference between CH and PC is $O(n\log n+mn+mk\log k)$ to $O((1-\beta_H)kmn)$. For larger n or smaller β_H, the time for computing PC increases faster than that for computing CH. Indeed, the experimental results shown in Chapter 6 show that CH outperforms PC in those cases.

5.4 An Algorithm Based on Isothetic Rectangle Intersections

Although the filtering step is effectual, the disadvantages of CH is that convex hull operations are not trivial to compute. Another common approach is to approximate a polygon by its *Minimum Bounding Rectangle* (MBR), rather than its convex hull. The Minimum Bounding Rectangle of a spatial object is the smallest axis-parallel rectangle which encloses the spatial object. The motivation of performing MBR approximation is that, from the complexity point of view, it is more efficient to deal with rectangles than convex hulls.

The only difference between the following algorithm IR and CH is that an isothetic rectangle is used instead of its convex hull.

Algorithm IR

1. Compute the isothetic rectangle B_{cl} of cluster Cl, and initialise set S to 0.
2. For each polygon PO in map M:
   (a) Compute its isothetic rectangle B_{po}.
   (b) If isothetic-rectangle-intersect(B_{cl}, B_{po}) is true, add PO to S.
3. Apply PC to the polygons in S.

In RT (standing for "Isothetic Rectangles"), it takes $O(n)$ to compute the isothetic rectangle of n points. Similar, Step (2a) takes $O(k)$ for each polygon with k edges. Due to the simplicity of isothetic rectangles, the procedure isothetic-rectangle-intersect() only takes constant time $O(1)$. Thus, the combined time complexity of Step (2) is $O(km)$, where m is the number of polygons in M. Finally, similar to Step (3) of CH, the complexity of Step (3) of IR is $O(n+km+\beta_B kmn)$, where β_B is the fraction of polygons that overlap with B_{cl}.

The complexity of IR with CH can be compared step by step. In Steps (1) and (2a), it is $O(n)$ and $O(k)$ for IR versus $O(nlogn)$ and $O(klogk)$ for CH. In Step 2(b), it is $O(1)$ for IR versus $O(k+n)$ for CH. The only part that CH can outperform IR is Step (3), where the difference is between the fraction β_H and β_B. Given any set of points, it can be shown that the isothetic rectangle always contains the convex hull. In other words, it is necessary that for any polygon PO, if the convex hull H_{cl} and H_{po} overlap, then their corresponding isothetic rectangles B_{cl} and B_{po} must also overlap. The converse is not true though. Thus, it is necessary that $\beta_H \leq \beta_B$. But unless β_B is significantly larger than β_H, IR is expected to be more efficient than CH.

5.5 An Algorithm Based on R-Tree Searches

The disadvantage of all the algorithms discussed so for is that all the polygons in a map must be checked for possible intersection with the given cluster. This problem can be solved if there is spatial indexing available.

The following algorithm, RT (standing for "R-Tree), is proposed to take advantage of the R-tree method. Due to fact that there is no prepared indexing for the thematic maps, R-trees must be constructed for each map during the run-time.

Algorithm RT

1. For each polygon *PO* in Map *M*
   (a) Compute its isothetic rectangle, and
   (b) Insert the rectangle into a R-tree.
2. Compute the isothetic rectangle B_{cl} of cluster *Cl*. Retrieve all isothetic rectangles in the R-tree that overlap with B_{cl}, Let all the polygons that correspond to these isothetic rectangle be the elements of a set *S*.
3. Apply *PC* to the polygons in *S*.

In Step (1), it takes $O(k)$ to find a isothetic rectangle of a polygon with k vertices. Thus, for m polygons, Step (1a) takes $O(km)$. For Step(1b), since there are m isothetic rectangle to be inserted, a total time of $O(m\log m)$ is required. Thus, Step (1) takes $O(km+m\log m)$. Step (2) takes $O(n)$ to construct the isothetic rectangle of *Cl*. The complexity of search for all the required isothetic rectangles from the tree depends on whether multiple paths of the tree are needed. For simplicity of analysis, suppose only one path is needed, in which case the search takes $O(\log m)$. Finally, if β_B is the fraction of polygons whose isothetic rectangle overlap with that of the cluster, then the complexity of Step(3) is $O(\beta_B kmn)$. Hence, the complexity of RT is $O(n+km+m\log m+\beta_B kmn)$.

From the complexity analysis shown above, although RT saves time on spatial searching, which takes $O(\log m)$ time, the construction of the R-tree is costly, requiring $O(m\log m)$. If the R-tree will be searched repeatedly, it is possible that the time saved by the subsequent searches exceeds the complexity of building a R-tree.

5.6 Brief Summary

A comparison of the complexity of each of the four algorithms is shown in Table 1. As discussed in Section 5.3, as long as β_H is small, and n is large, the complexity of CH is less then PC. As for the comparison between IR and CH, as illustrated in Section 5.4, IR is more efficient than CH, unless β_B is significantly larger than β_H. It is clear that IR outperforms RT from the complexity point of view. The key is that although IR computes the intersection between each polygon and a cluster, checking whether two isothetic rectangles intersect or not is a constant time operation. In contrast, in RT, inserting the isothetic rectangles into the index takes $O(m\log m)$.

Algorithm	Time Complexity
PC	$O(kmn)$
CH	$O(\beta_H kmn+mklogk+mn+nlogn)$
RT	$O(\beta_B kmn+mk+mlogm+n)$
IR	$O(\beta_B kmn+mk+n)$

Table 1: Complexities of the Computations of Utility Measures

5.7 Computation of Two Pattern Extraction Operations

As mentioned in Section 5.5, the initial cost of the R-tree construction can be amortised, if the index can be re-used in the following searches. This is exactly the case for pattern extraction operations, where a map is searched by several clusters. Although IR is the best for the computation of the characteristic-utility measures, it is questionable whether IR will still be superior to RT in the computation of pattern extraction. In order to investigate this problem, algorithms based on both IR and RT are proposed in this section.

5.7.1 Algorithms for Common()

The following algorithm Com(*IR*), based on IR, computed the strong common characteristics of n clusters.

Algorithm Com(IR)

1. For each Map M;

(a) Set u to 1.

(b) Call IR.

(c) If the utility is less than *ITh*, go to Step 1.

(d) Otherwise, if $u<n$, increment u by 1 and go to Step(1b).

(e) Otherwise, compute $Common(\{Cl_1,\ldots Cl_n\}, M)$.

(f) if the utility of $Common(\{Cl_1,\ldots Cl_n\}, M\} \geq CTh$, output this value.

The purpose of Step(1c) is to exclude a map M as soon as there is a cluster whose characteristic-utility measure for M is below the threshold *ITh*. No further processing of the

map *M* with the rest of the clusters is necessary. The higher the threshold value *ITh*, the higher the percentage of maps are filtered.

For RT, two modifications, Step 1(a) and 1(b), are needed to the above algorithm. The modification are made, because for each map, once the R-tree is created, each of all *n* cluster can use the same tree.

Algorithm Com(RT)

1. For each Map *M*;

   (a) Call Step (1) of RT. Set *u* to 1.

   (b) Call Step (2) and (3) of RT.

   (c) If the utility is less than *Ith*, go to Step 1.

   (d) Otherwise, If $u<n$ increment *u* by 1 and go to Step(1b).

   (e) Otherwise, compute $Common(\{Cl_1,\ldots Cl_n\}, M)$.

   (f) If the utility of $Common(\{Cl_1,\ldots Cl_n\}, M\} \geq CTh$, output this value.

There are two main factors that influence the performance comparison of Com(RT) and Com(IR). The first is the number of the clusters *n*. The run-time of Com(IR) linearly increases with *n*. The run-time of Com(RT) only partially increases with *n*, because Step(1a) of Com(RT) only executes once. However, how large *n* should be in order for Com(RT) to outperform Com(IR) remains a question. The second factor is the value of threshold *ITh*. When *ITh* is high, a map is likely be eliminated before it will be examined by all the clusters. This situation is disadvantageous for Com(RT), because the R-tree will not be fully used. The effort of *ITh* on Com(RT) is also a question. The experimental results to be presented in Chapter 6 sheds light on these questions.

5.7.2 Computation of Discriminating Characteristics of Two Clusters

It is easy to modify both Com(IR) and Com(RT) to compute discriminating characteristics of two clusters. The key difference is instead of computing the combined characteristic, the algorithm should compute the discriminating utility as defined in Section 4.3.

The following algorithm, based on IR, computes of the discriminating characteristics of two clusters.

Algorithm Diff(IR)

1. For each Map *M*;

(a) Set u to 1.

(b) Call IR.

(c) If the utility is less than *ITh*, go to Step 1.

(d) Otherwise, if $u<2$, increment u by 1 and go to Step(1b).

(e) Otherwise, compute *different*($\{Cl_1, Cl_2\}$, *M*).

(f) if the utility of *different*($\{Cl_1, Cl_2\}$, *M*) $\geq GTh$, output this value.

An algorithm based on RT is not proposed, because it only involves two clusters in the computation of *different*(Cl_1, Cl_2). In the next chapter, experimental results will show that since the efficiency of IR is usually several times than of RT, it is impossible for Diff(*RT*) to outperform Diff(*IR*).

5.8 Summary

In this chapter, the algorithms of how to compute the utility measures have been proposed. From the complexity analysis, IR is the best among the four algorithms with respect to efficiency. For the computation of pattern extraction operations, algorithms based on both IR and RT are proposed, which of them is more efficient will be discussed in the next chapter.

Chapter 6

Experimental Evaluation

In this chapter, the following algorithms are experimentally evaluated: the two heuristic algorithms, the four algorithms for computing characteristic sets, and the two algorithms for pattern extractions. All the experiments were carried out in a time-sharing SPARC-LX workstation running SunOS 5.5. The reported execution figure, obtained by the UNIX *time* command, is the time averaged over 15 runs.

6.1 Details of Thematic Maps

In order to make the experimental results as realistic as possible, thematic maps, provided by Statistics Canada of Greater Vancouver area in 1990, were used. Among the available 90 categories/theme of census, 50 were randomly chosen for the purpose of experiments, including household income, unemployment rate, average value of dwellings, education level, etc. As illustrated in Chapter 5, the run-times of the algorithms are linearly proportional to the number of maps. Thus, to test for the run-time scalability, there is no need to try with thousands of maps, it is sufficient to obtain the average run-time of a map, as long as the maps represent a typical mix.

As discussed in Chapter 3.1, the base map of a thematic map is regarded as a collection of polygons which represent the sub-regions of the map. Greater Vancouver region is divided into 88 sub-regions by Statistics Canada. Thus, the number of polygons of a Vancouver thematic map is 88. In order to represent a typical mix of the maps, one third of the maps are

reconstructed by merging neighbouring polygons with the same class labels together, so that they contain around 20 polygons if possible. And another one third of the maps are processed using the same merging method, so that they contain around 50 polygons if possible. Thus, of the 50 thematic maps, one third contain 20 polygons, one third contain 50 polygons and the rest have 88 polygons. Note that although the maps in a typical geographical information system (GIS) may easily contain tens of thousands of "topographical" polygons, a typical thematic map contains fewer polygons, e.g., at most 88 polygons in Vancouver thematic maps.

Due to the constraint of the census data domain, all of the 50 thematic maps belong to open or closed range numerical maps. The number of classes for all the maps varies from 5 to 20.

Another implementation detail for the maps is that, though the polygons do not overlay in a thematic map, it is common to encounter the situation that a polygon is completely contained in another polygon, which is called a complex polygon. In the implementation, to simplify the computation, the complex polygons are always divided into simple polygons which contain no holes.

6.2 Details of Clusters

Although in practice, the clusters should be generated by clustering algorithms, such as CLARANS, for the purpose of extensive experimentation, clusters are a mixed collection of various sizes, densities and locations.

The size of a cluster is described by an area ratio, which is the actual size of the cluster divided by the actual size of the region covered by the map. The area ratio 0.01 represents an area of roughly 9 blocks by 9 blocks for the Vancouver region. The area ratio conducted in the experiments ranges from 0.01 to 0.1. Area ratio is an important parameter because the larger the area ratio, the more likely that a larger number of polygons are intersected with the cluster.

The density of a cluster measures the average number of points in the cluster within a 4 block by 4 block area. The density of the clusters conducted in the experiments ranges from 1 to 5. Multiplying the area ratio with density gives the total number of points in the clusters.

To diminish the influence of the location of the clusters to the experimental results, for each cluster, 3 experiments are done at 5 different locations of a map. And the average of these experimental results is taken to be the run-time of the cluster.

6.3 Validity of Open Class Average Value Approximation

In Chapter 3, Heuristic Algorithm 1 and 2 were proposed to estimate the average value of open range classes. The assumption of the algorithms is that there is an underlying normal distribution. In this experiment, the validity of this assumption is evaluated.

The thematic map that describes the average value of dwellings is chosen for the evaluation. The average value of dwellings for each of the 88 polygons of the map is obtained for the purpose of comparison with the estimation. The thematic has 5 classes with an open class of the form "≤c".

mean	371,297
interquartile range	278,078
standard deviation	205,984
p=prob(value≥580K)	0.106
Zp	1.62

Table 2 Observed distribution

Table 2 gives the key figures from the observed distribution. Thus, by Equation (8), the estimated average of the open range class is 704K. The actual average, computed based on the exact statistical figures, is 686K. Furthermore, to evaluate the effect of the estimated average on the final utility value, as defined by Equation (4), experiments are done with clusters with varying percentages of points in the open range class. Table 3 gives the results of the three percentages.

	20%	50%	90%
using estimated average	0.746	0.683	0.81
using actual average	0.76	0.7	0.82

Table 3 Comparison of actual and estimated values

The utility values obtained from the estimated average (704,000) are lower than those based on the actual average (684,000). This is because the estimated average is higher than the

actual average, and the points in the cluster that are not in the open range class are all from classes with lower values. In any case, the difference between the actual and estimated utility values is below 2.5 percent, showing that the heuristic algorithm provides valid approximations.

6.4 Efficiency of Computing Characteristic Sets

In this series of experiments, the four algorithms for computing characteristic sets are implemented for the purpose of performance comparison. The computational geometry procedures, namely point-in-polygon(), convex-hull-generation(), convex-hull-intersect(), and isothetic-rectangle-intersect(), are implemented based on [31]. The code for the R-tree implementation are developed and kindly provided by Christos Faloutsos and his group at the University of Maryland, College Park. Though the R-tree code may not be as optimised as other implementations such as the R^*-tree, given that a thematic map typically does not have a huge number of polygons, the conclusions drawn based on the R-tree code should still be valid.

For the purpose of extensive studies, experiments are conducted on clusters with various sizes and densities. As mentioned before, the area ratio of a cluster to a map gives the geographical size of a cluster. And the density of clusters together with the area ratio gives the total number of points in a cluster. Figure 5 shows the CPU time taken by the four algorithms on low density clusters whose area ratios range from 0.01 to 0.1. And Figure 6 shows the CPU time taken on the high density clusters with the same area ratio range. The CPU time in the two figures is the average of 50 maps.

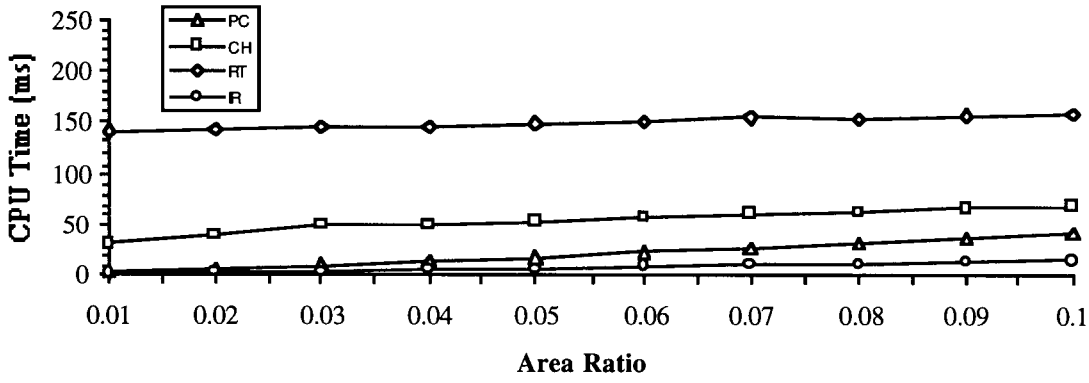


Figure 5. Low Density Clusters

In Figure 5, the density of the cluster is 1. Figure 5 shows that the run-times of all algorithms are not too dependent on the cluster sizes. The reason is that even if the area ratio increases, the actual number of points does not increase much due to the low density value. Figure 5 shows that the algorithm IR is at least 3 times faster than RT.

In Figure 6, the density value is 5. The run-time of all algorithms do increase with the growing size of the clusters. PC is the most affected. But in any case, IR still outperforms the other algorithms.

Figure 5 and 6 clearly show that characteristic sets can be computed very efficiently. For instance, for medium sized clusters, i.e., ratio=0.05, of low density, the time taken by IR for one map is about 10 milliseconds. For medium-sized clusters of high density, the time taken is about 30 milliseconds.

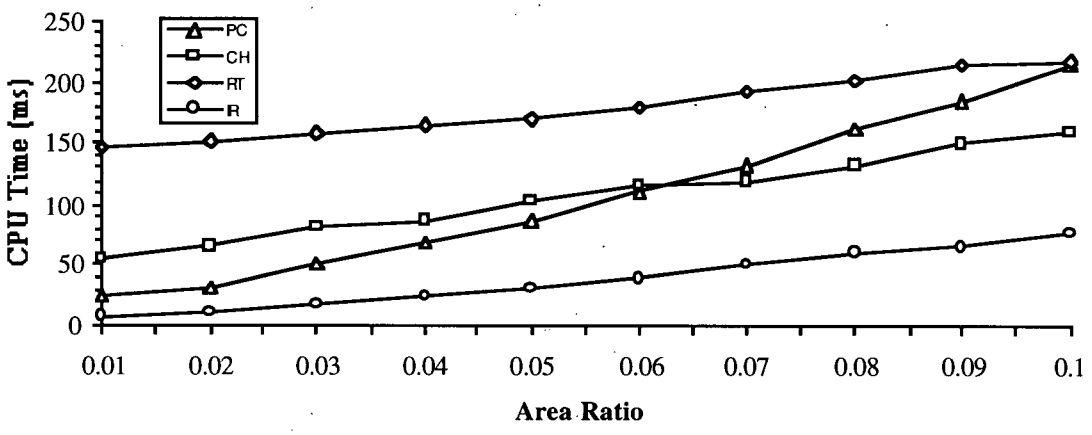


Figure 6. High Density Clusters

6.5 Efficiency of Pattern Extraction Operation

In this series of studies, experiments are done on Com(IR) and Com(RT). The number of clusters ranges from 1 to 6. Figure 7 and 8 show the CPU time for the two algorithms with a high and low threshold value input respectively. The CPU time in the two figures is the average of 50 maps.

Figure 7 shows the situation where 10% of all thematic maps are strong characteristics for all the 6 clusters. The run-time of both Com(IR) and Com(RT) flatten off as the number of clusters increases. This is because each cluster represents a filter that removes non-qualified maps. Thus, as n becomes larger and larger, the extra effort incurred in increasing the number of clusters from n to n+1 becomes smaller and smaller. This trend is significant because it implies that it will take very large values of n for Com(RT) to approach Com(IR). For n=6 clusters, Com(RT) requires 3 times as long to compute $common(Cl_1,...Cl_n)$ as Com(IR) does.

Even when the percentage of maps passing threshold is much higher, like 60% in Figure 8, Com(RT) still takes twice as much time as Com(IR) for 6 clusters. It can be concluded that Com(RT) may come closer to Com(IR), only when n is very large.

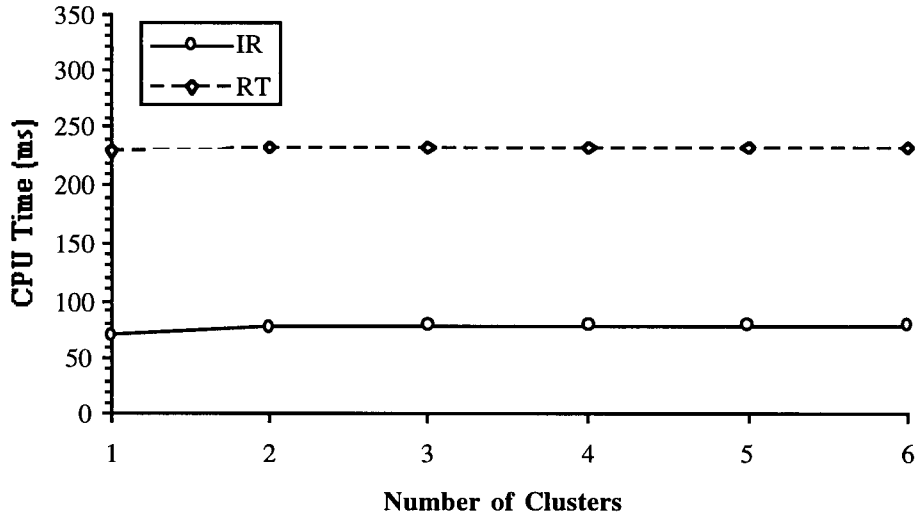


Figure 7. Large *ITh*: 10% Passing Rate

Figure 8 also shows that strong common characteristics can be found very efficiently. Even with a small passing rate such as 10%, it only takes about 70 ms per map for 6 clusters

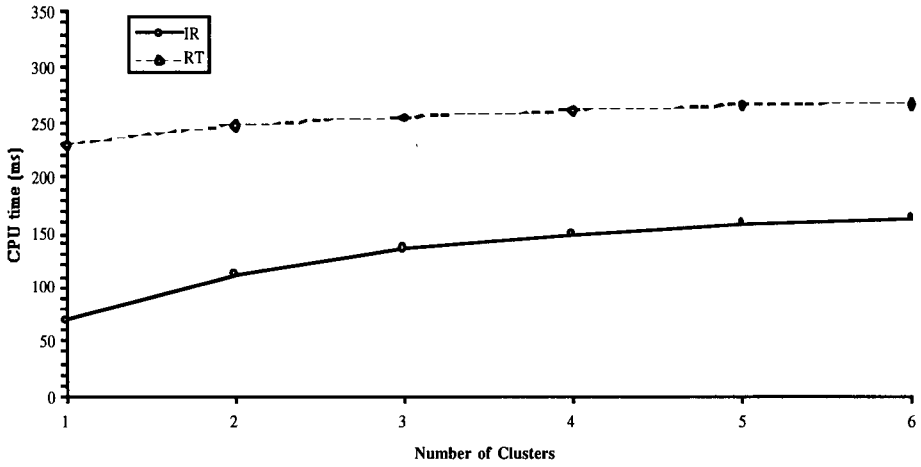


Figure 8. Very Small *ITh*: 60% Passing Rate

6.6 Summary

In this chapter, it has been shown that the heuristic algorithms provide valid approximations. Experiments also show that IR is a promising algorithm. It is able to provide efficient computation for both characteristic sets and pattern extraction operations, even when processing thousands of maps.

Chapter 7

Discussions

7.1 Summary

This thesis studies how to discover strong, common and discriminating characteristics of clusters. Previous works solved this problem by relying on pre-constructed concept hierarchies. Since concept hierarchies have their own limitations, this thesis aimed to solve the problem by correlating clusters with statistical census data. Census data are excellent sources for knowledge discovery, because they cover a variety of information such as population, education, occupation, dwelling, income and so on. Census data usually exist in the form of thematic maps. Thus, this thesis works on discovering strong, common and discriminating characteristics of clusters from thematic maps. Specifically, there are three main issues addressed in the thesis.

The first issue is how to discover those maps which represent the characteristics of a cluster. Mainly, there are two different kinds of thematic maps, ordinal and nominal thematic maps. Ordinal maps include single-value and interval maps. Interval maps can further be classified into open and closed range maps. In order to accommodate different kinds of maps, two basic measures are proposed to assign utility values to the thematic maps. The higher the utility measure, the stronger the map is a characteristic of the cluster. An entropy-based measure works for nominal maps, while a standard-deviation-based measure is applied to single-value ordinal maps. Some extra steps should be taken before employing the STD-based

measure to interval ordinal maps. The utility values produced by the two measures are normalised in the range of [0,1], so that the results from different maps are comparable.

The second issue is to define measures which perform concept description task, i.e. discovering the common and discriminating characteristics of the multiple clusters. Though there could be many ways to define them, the concern is that the definitions of common and discriminating characteristics for the multiple clusters should be consistent with the one for individual clusters. Therefore, the common characteristics of the clusters are specified to be those maps, which are not only strong characteristics of each of the clusters, but also strong characteristics of the cluster formed by merging the clusters together. This specification indicates that the utility measure for a common characteristic is exactly the same as the one for individual clusters. An operation, *common*(), is defined to carry out this specification. In order for a map to be a discriminating characteristic of two clusters, it must be a strong characteristic of each of the two clusters, and characteristics which the two clusters displayed in the map should be "sufficiently different". The operation *different*() measures the difference of characteristics exhibited by the two clusters.

The last issue is how to provide scalable computations for the above tasks. It is obvious that the more the maps are processed, the better the characteristics of the clusters will be captured. Usually, there are hundreds and thousands of maps needing to be handled. Four algorithms have been proposed for computing the characteristics of the clusters. Both complexity analysis and experimental results show that algorithm IR, which takes advantage of

spatial object approximations, outperforms the algorithm based on spatial indexing and other proposed algorithms. IR is capable of processing a hundred of maps in about 5 seconds. To extract patterns efficiently, algorithm Com(IR) and Com(RT) are proposed, which are based on IR and RT respectively. Experimental results show that Com(IR) always outperforms Com(RT).

7.2 Future Work

7.2.1 Efficiency Improvement

Although the algorithm IR, provides satisfactory computational efficiency, it still has some drawbacks. The first is that all the polygons in the space must be searched for possible intersection with a given cluster. The spatial indexing method, R-tree, is an obvious way to fix this problem, but is inapplicable for thematic maps. The reason is that a thematic map usually does not contain a large number of polygons for the cost of R-tree construction to be amortised.

The study reported in [37] explores a spatial join method that dynamically constructs index trees, called *seeded tree*, at join time. Seeded trees are R-tree-like structures, and are divided into the seed levels and the grown levels. The characteristics of the input data sets are utilised to build the seeded levels. A tree is dynamically constructed on the basis of the seed level.

It is possible that the seeded tree method is more efficient than algorithm IR. Since all the thematic maps are based on a common base map, constructing a common seed level for all the maps is possible. Unlike R-tree where the entire tree is constructed for each of the maps, only grown levels will be constructed in the case of a seeded tree. This could dramatically reduce the construction time.

The second problem with algorithm IR is that it is a memory-based method. It assumes that all thematic maps are all stored in main memory. This assumption may not be valid, if the number of maps is extremely large. Also this drawback of IR can be solved by the seeded tree, which is a disk-based method.

7.2.2 Effectiveness Improvement

This thesis answered the question of what the characteristics of clusters are. But it would be more interesting to give the answer of why the clusters with such characteristics are there. Usually, the characteristics of a cluster is highly related to its geographical circumstances. For example, a cluster which has the characteristic of high income may close to a park. Ng and Knorr [35] [36] have investigated algorithms to determine the relationship with the clusters and their geographical surrounding. By combining their work with the study of this thesis, the comprehensive properties of the clusters can be discovered.

References

[1] W. J. Krawley, G. Piatesky-Shapiro and C. J. Matheus, "Knowledge Discovery in Database: An Overview", in *Knowledge Discovery in Databases*, AAAI/MIT, 1991.

[2] C. J. Matheus, P. K. Chan and G. Piatesky-Shapiro, "Systems for Knowledge Discovery in Databases", in *IEEE Transaction on Knowledge and Data Engineering*, Vol. 5, No. 6, Dec. 1993.

[3] J. Pearl and T.S. Verma, "A Theory of Inferred Causation," in *Proc. of the 2nd Intl. Conference on Principles of Knowledge Representation and Reasoning*, pp441-452, 1991.

[4] R. Sriknat and R. Agrawal, "Mining Generalised Association Rules", in *Proc. of the 21st VLDB Conference*, Zurich, Switzerland, 1995.

[5] W. Ziarko, "The Discovery, Analysis, and Representation of Data Dependencies in database", in *Knowledge Discovery in Databases*, AAAI/MIT, 1991.

[6] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor and D. Freeman, "AutoClass: A Bayesian Classification System", *in Proc. of the 5th Intl. Conference on Machine Learning*, pp54-64, San Mateo, CA, 1988.

[7] P. Beard, "Automated Arbitrage Expert System Developed: It Outperformed S&P's 500 in First Quarter", in *AIWeek*, pp1-3, Vol. 6, No. 13, 1989.

[8] D. J. Lubinsky, "Discovery form Databases: A Review of AI and Statistical Techniques", *AT&T Bell Laboratories*, Holmdel, New Jersey, June 1989.

[9] G. Piatesky-Shapiro and W. J. Frawley (Editors), *Proc. of IJCAI-89 Workshop on Knowledge Discovery in Databases*, Detroit, Michigan, August 1989.

[10] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[11] G. Dunn and B. S. Everitt, *An Introduction to Mathematical Taxonomy*, MIT, 1982.

[12] D. Fisher, M. Pazzani, and P. Langley (Editors), *Concept Formation: Knowledge and Experience in Unsupervised Learning*, San Mateo, CA, 1991.

[13] G. Piatesky-Shapiro and C. J. Matheus, "Knowledge Discovery Workbench: An exploratory environment for discovery in business database", in *Workshop Notes from the 9th National Conference on Artificial Intelligence: Knowledge Discovery in Databases*, Anaheim, CA, July 1991.

[14] D. B. Lenat. "On Automatic Scientific Theory Formation: A Case Study Using the AM Program", in *Machine Intelligence*, Vol. 9, New York, Halsted, 1977.

[15] W. Lu, J. Han and B. C. Ooi, "Discovery of General Knowledge in Large Spatial Databases", in *Proc. of Far East Workshop on Geographic Information Systems*, pp 275-289, Singapore, 1993.

[16] R. T. Ng and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining", in *Proc. of the 20th VLDB Conference*, Santiago, Chile, 1994.

[17] M. Holsheimer and A. P. J. M Siebes, "Data Mining: The Search for Knowledge in Databases", in *Report CS-R9406*, January 1994.

[18] S. Forheringham and P. Rogerson (Editors), *Spatial Analysis and GIS*, Taylor & Francis, 1994.

[19] R. Agrawal, K. Lin, H. S. Sawhney and K. Shin, "Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases", in *Proc. of the 21th VLDB Conference*, Zurich, Switzerland, 1995.

[20] R. Agrawal, C. Faloutsos and A. Swami, "Efficient similarity search in sequence databases", in *Proc. of the 4th Intl. Conference on Foundations of Data Organisation and Algorithms*, Chicago, 1993.

[21] C. Faloutsos, M. Ranganathan and Y. Manolopulos, "Fast subsequence matching in time-series database", in *Proc. of ACM SIGMOD Conference on Management of Data*, May 1994.

[22] R. Agrawal and R. Srikant, "Mining Sequential Patterns", in *Proc. of Intl. Conference on Data Engineering*, Taiwan, March 1995.

[23] J. Han, Y. Cai and N. Cercone, "Knowledge Discovery in Databases: an Attribute-Oriented Approach, in *Proc. of the 18th VLDB*, pp.547-559, 1992.

[24] B. Dent, *Cartography: Thematic Map Design*, Wm. C. Brown, 1990.

[25] R. M. Gray, *Entropy and Information theory*, Springer-Verlag, New York, 1990.

[26] R. S. Witts, *Statistics*, Harcourt Brace Jovanovich Colledge, 1993.

[27] D. J. Cuff and M. T. Mattson, *Thematic maps: Their Design and Production*, Methuen, New York, 1982.

[28] R. Guttman, "R-TREES: A Dynamic Index Structure For Spatial Searching", in *Proc. of ACM SIGMOD Int. Conference on Management of Data*, pp47-57, Boston, MA, 1984.

[29] H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley, 1990.

[30] R. Bayer and E. McCreight, "Organisation and Maintenance of Large Ordered Indices", in *Proc. of ACM DISFIDET Workshop on Data Description and Access*, Houston, Texas, Nov. 1970.

[31] J. O'Rourke, *Computational Geometry in C*, Cambridge University, New York, 1994.

[32] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRTH: an Efficient Data Clustering Method for Very Large Database", to appear in *Proc. of ACM SIGMOD Intl. Conference on Management of Data*, Montreal, Canada, June 1996.

[33] Fink, *Electronic Engineer's Handbook*, McGraw-Hill, 1975.

[34] M. Ester, H. P. Kriegel and X. Xu, "Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification", in *Proc. of the 4th Intel. Symp. on Large Spatial Databases*, Portland, Maine, August 1995.

[35] E. M. Knorr and R. T. Ng, "Finding Aggregate Proximity Relationships and Commonalities in Spatial Data Mining", to appear in *IEEE Transactions on Knowledge and Data Engineering*.

[36] E. M. Knorr and R. T. Ng, "Extraction of Spatial Proximity Pattern by Concept Generalisation", in *Knowledge Discovery in Database*, AAAI/MIT, 1996.

[37] Ming-Ling Lo and C. V. Ravishankar, "Spatial Joins Using Seeded Trees", in *Proc. of ACM SIGMOD on Management of Data*, Minneapolis, Minnesota, May 1994.

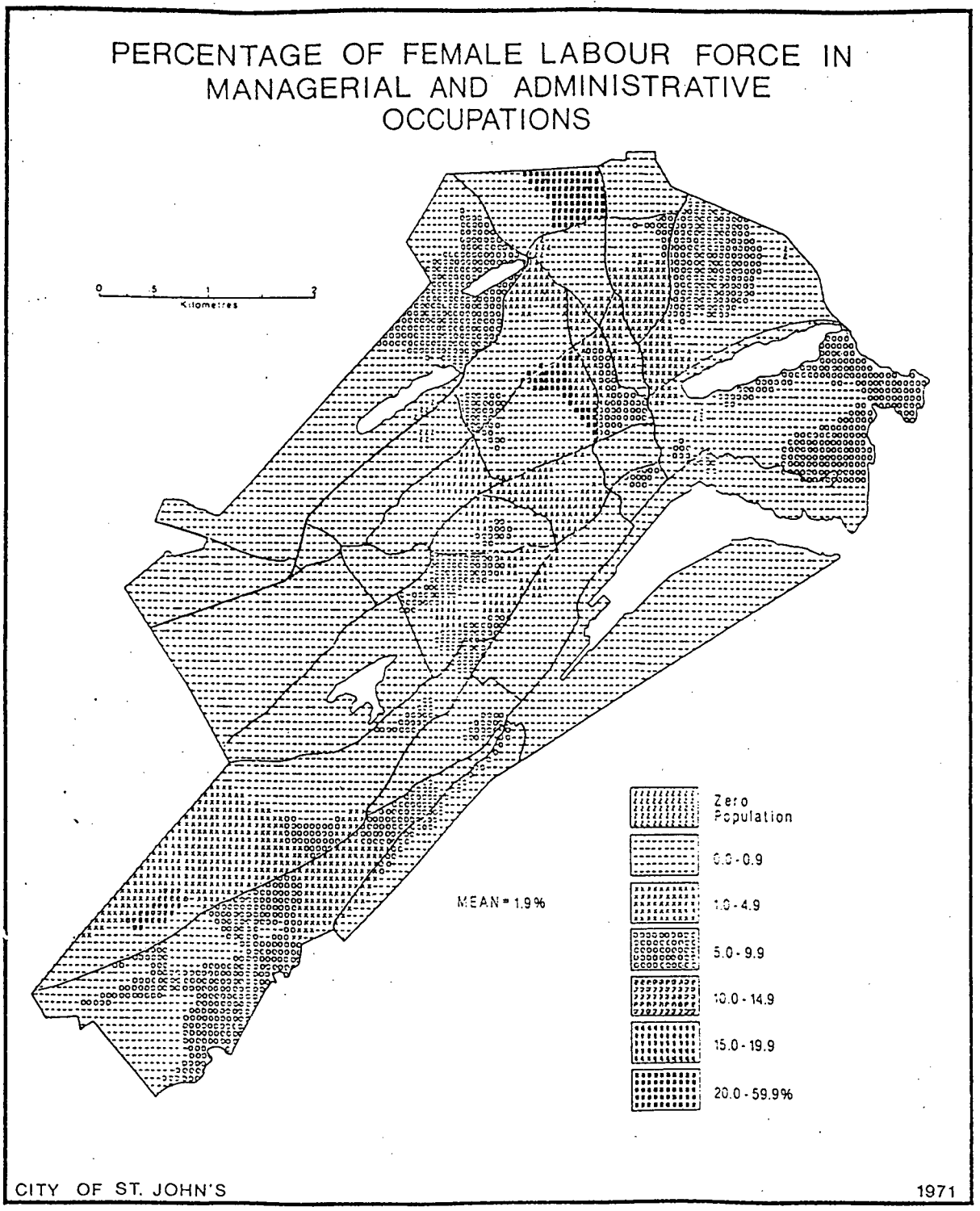
[38] N. Roussopoulos and D. Leifker, "Direct Spatial Search on Pictorial Databases Using Packed R-trees", in *Proc. of ACM SIGMOD Conference on Management of Data*, 1985.

[39] T. Sellis, N. Roussopoulos and C. Faloutsos, "The R^+-tree: A Dynamic index for multi-dimensional objects", in *Proc. of the 13th Intl. Conference on Very Large Data Bases*, pp507-51, Brighton, 1987.

[40] N. Beckmann, H.-P. Kriegel, R. Schneider and B. Seeger, "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles", in *Proc. of ACM SIGMOD Conference on Management of Data*, pp322-331, May 1990.

[41] R. H. Guting and D. Wood, "Finding Rectangle Intersections by Divide-and-Conquer", in *IEEE Transactions on Computers*, Vol. c-33, No. 7, July 1984.

Appendix A


PERCENTAGE OF FEMALE LABOUR FORCE IN MANAGERIAL AND ADMINISTRATIVE OCCUPATIONS
0 5 1 3
Kilometres
MEAN = 1.9%
Zero Population
0.0 - 0.9
1.0 - 4.9
5.0 - 9.9
10.0 - 14.9
15.0 - 19.9
20.0 - 59.9%
CITY OF ST. JOHN'S
1971