DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

MODELING AND ANALYZING MARINE DATA USING DATA MINING TECHNIQUES

by
Derya BİRANT

May, 2006
İZMİR

MODELING AND ANALYZING MARINE DATA USING DATA MINING TECHNIQUES

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylul University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Computer Engineering, Computer Engineering Program

by
Derya BİRANT

May, 2006
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"MODELING AND ANALYZING MARINE DATA USING DATA MINING TECHNIQUES"** completed by **DERYA BİRANT** under supervision of **PROF. DR. ALP KUT** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Alp KUT

Supervisor

Assoc. Prof. Dr. Yalçın ÇEBİ

Committee Member

Asst. Prof. Dr. Reyat YILMAZ

Committee Member

Asst. Prof. Dr. Şen ÇAKIR

Jury Member

Assoc.Prof.Dr.Vladimir RADEVSKI

Jury Member

Prof. Dr. Cahit HELVACI

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to express my sincere appreciation first and foremost to my advisor, Prof. Dr. Alp KUT, for his strong support, patience, valuable insights and encouragement, not only in bringing this research work to a successful completion, but also in all aspects of my academic life. He devoted his time and energy to improve this thesis despite his busy schedule.

I extend my thanks to the members of my committee, Assoc. Prof. Dr. Yalçın ÇEBİ, and Asst. Prof. Dr. Reyat YILMAZ for their useful comments and suggestions during my study.

I would like to thank the staff of the Institute of Marine Sciences and Technology at Dokuz Eylul University for sharing their knowledge with me and for their helps in obtaining database. I also want to thank Ufuk DEMİR and Mehmet Harun GÜLEN for their support during the development of the prototype program.

Lastly, I would like to express my special gratitude to my parents and my husband, Kökten Ulaş BİRANT. They have always been with me to help and to support.

Derya BİRANT

MODELING AND ANALYZING MARINE DATA USING DATA MINING TECHNIQUES

ABSTRACT

The research presented in this thesis is an interdisciplinary work that combines computer science and marine science. It provides new computer based approaches, techniques and technologies for (i) modeling, collecting, archiving marine data, (ii) analyzing and mining marine data by using data mining techniques and (iii) visualizing marine data. It presents my efforts on the collecting physical, biological, chemical marine data, some explanations about the visualization of marine data on the map, my works on the construction of decision trees to classify physical marine data. This thesis introduces two new data mining algorithms: one is for clustering spatio-temporal data and the other is for spatio-temporal outlier detection in data warehouses. It also proposes a new approach: web service-based parallel clustering which includes the parallel execution of web services for discovering clusters in large data warehouses.

In addition to new clustering algorithm, this thesis also presents the validation and evaluation of the clustering results of this clustering algorithm. It shows the mathematical quality and reliability of the clustering results by using a cluster validation technique. It also presents the sensitivity analysis of the new clustering algorithm to the input parameters.

Keywords : Data Mining, Knowledge Discovery, Spatio-Temporal Databases, Clustering, Outlier Detection, Decision Tree, Data Warehouse

VERİ MADENCİLİĞİ TEKNİKLERİNİN KULLANILARAK DENİZ VERİLERİNİN MODELLENMESİ VE ANALİZİ

ÖZ

Bu tezde sunulan çalışma bilgisayar bilimleri ve deniz bilimlerini birleştiren disiplinler arası bir çalışmadır. Bu tez (i) deniz verilerinin modellenmesi, toplanması, arşivlenmesi (ii) deniz verileri üzerinde veri madenciliği tekniklerinin kullanılması ve analizlerin yapılması (iii) deniz verilerinin görselleştirilmesi için yeni bilgisayar tabanlı yaklaşımlar, teknikler ve teknolojiler sağlamaktadır. Bu tez fiziksel, kimyasal ve biyolojik deniz verilerinin toplanması için yaptığım çabaları, deniz verilerinin haritalar üzerinde görselleştirilmesi üzerine bazı açıklamaları, fiziksel deniz verileri için oluşturulan karar ağaçlarını sunmaktadır. Ayrıca iki tane yeni veri madenciliği algoritmasını tanıtmaktadır: bunlardan bir tanesi konumsal-zamansal verilerin kümelenmesi için, diğeri ise veri ambarlarında konumsal-zamansal sıra dışı verilerin tespiti içindir. Bu tez ayrıca yeni bir yaklaşımda önermektedir: web servis tabanlı paralel kümeleme. Bu yöntem büyük veritabanlarında kümelerin keşfedilmesine yönelik web servislerin paralel olarak çalıştırmasını öngörmektedir.

Bu tez yeni kümeleme algoritmasına ek olarak bu algoritma ile elde edilen kümeleme sonuçlarının doğruluğunu ve geçerliliğini değerlendirir. Bir küme doğrulama tekniği kullanarak matematiksel kalitesini ve güvenilirliliğini gösterir. Bu tez aynı zamanda yeni kümeleme algoritmasının girdi değerlerine olan duyarlılık analizini de sunmaktadır.

Anahtar sözcükler : Veri Madenciliği, Bilgi Keşfi, Konumsal-Zamansal Veritabanları, Kümeleme, Sıradışı Verilerin Tespiti, Karar Ağacı, Veri Ambarı

ABBREVIATIONS

AVHRR	Advanced Very High Resolution Radiometer
CTD	Conductivity Temperature and Depth
DBSCAN	Density Based Spatial Clustering of Applications with Noise
GDI	Graphic Device Interface
IMST	Institute of Marine Sciences and Technology
KDD	Knowledge Discovery in Databases
MARDEX	MARine Data EXplorer
MCSST	Multi-Channel Sea Surface Temperature
MPI	Message Passing Interface
NASA	National Aeronautics and Space Administration
NOAA	National Oceanic and Atmospheric Administration
PVM	Parallel Virtual Machine
RS Index	R-Squared Index
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
ST-DBSCAN	Spatio-Temporal DBSCAN
UDDI	Universal Description, Discovery and Integration
WSDL	Web Service Description Language

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

1.1 General

This thesis is an interdisciplinary work which combines computer science and marine science. It is useful for marine science by providing new computer based technologies for archiving, analyzing, and visualizing marine data. The new computer based technologies developed in this thesis are generally related with the topics:

Data Modeling: Establishing new standardized methods of collecting and publishing marine data, modeling and constructing a marine data center for scientific researches and analyses. (Section 1.1.1)

Data Mining: Applying data mining algorithms on marine data to extract useful marine-related knowledge. (Section 1.1.2)

Web Services and Parallel Computing: Applying web service-based approaches for parallel computing on marine data. (Section 1.1.3)

In addition, this thesis also contains Data Warehousing, Web Programming and Data Visualization issues.

1.1.1 Data Modeling

In many marine institutes in Turkey, data are stored in various machines of the staff in different formats and they are not all published. Data are distributed disorderly and data accessibility are unsatisfactory for most potential users. In this study, in order to overcome these difficulties, a marine data center was modeled and constructed to make the data more readily available to marine scientists. Data accumulation and data acquisition procedures were identified by applying

preprocessing operations. Preprocessing operations are necessary to arrange raw data into a common format and to clean inconsistent, incorrect and missing data. In this study, a system was also developed for remote publishing of data. In addition, an efficient indexing mechanism was used to ensure fast access to data. A regular backup mechanism was also provided for the protection of the data. In this system, the data catalog is quite broad, containing information on many data types. The data collections can be extended easily when new data arrive.

1.1.2 Data Mining

Data Mining is the non-trivial extraction of implicit, previously unknown, and potentially useful information from large databases. (Cios et al., 1998) This definition indicates that data mining process is assumed to be nontrivial because it contains many necessary steps such as data preparation, search for patterns, knowledge retrieval and refinement etc. In order to discover the valid information, the database should be large.

Main goal of data mining is to extract knowledge from data and to predict outcomes of future situations. Knowledge discovery from marine data is a very promising subfield of data mining because increasingly large volumes of marine related data can be collected from satellites and need to be analyzed. In marine science, data mining is useful for understanding of the seawater characteristics. For example, clustering which is one of the major data mining methods can be used to obtain a number of clues about how the physical properties of the water are distributed in a marine environment. Similarly, outlier detection technique in data mining can be used to detect rare events and exceptional cases in a marine environment.

Seawater data involve both spatial and temporal dimensions. Knowledge discovery process for spatio-temporal data is more complex than for the non-spatial and non-temporal data because spatio-temporal data mining algorithms have to consider the spatial and temporal neighbors of objects to extract useful knowledge. In

addition to marine science, data mining algorithms designed for spatio-temporal data can also be used in many applications such as geographic information systems, medical imaging, and weather forecasting.

1.1.2.1 Data Mining Techniques in this Thesis

The basic data mining techniques are classification, clustering, association rule analysis, outlier detection (anomaly detection), regression analysis, and time series analysis. This thesis focuses on clustering, outlier detection and classification techniques.

- Clustering: A cluster is a collection of data objects that are *similar* to one another within the same cluster and are dissimilar to the objects in other clusters. Clustering is the process of identifying clusters embedded in the data. A measure of similarity can be computed between pairs of observations, between a single observation and previously formed clusters, or between two previously formed clusters. As shown in Figure 1.1, clustering algorithms can be categorized as: *partitioning methods*, *hierarchical methods*, *density-based methods, grid-based methods*, and *model-based methods*. (Han and Kamber, 2001) This approaches are described in chapter two. This study focuses on density-based clustering algorithms.

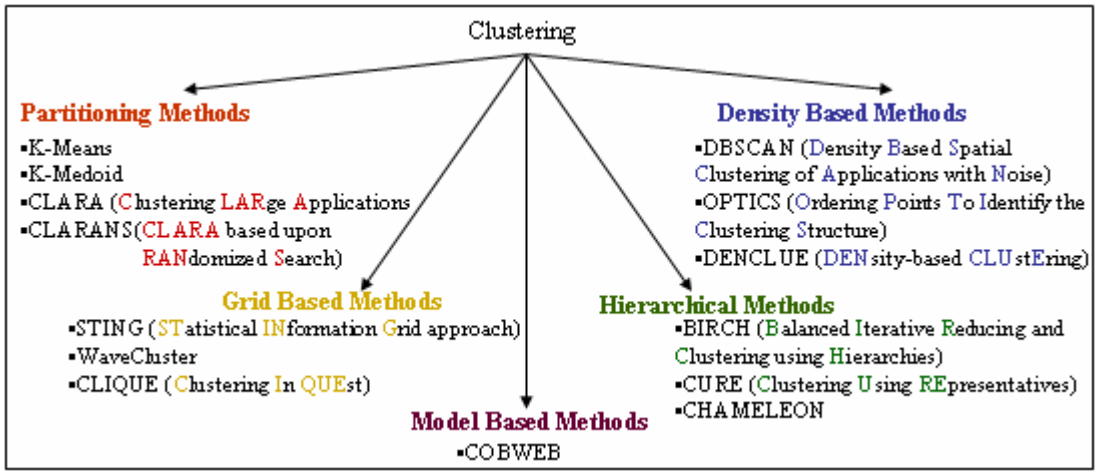


Figure 1.1 The categories of clustering algorithms

- Outlier Detection: It is the process of searching for unexpected discoveries, rare events and extreme deviations from an average value. Outlier data object does not comply with the general behavior of the data. Outliers are considerably dissimilar from the remainder of the data. It is usually considered as noise or exception. In data mining, this technique is quite useful in fraud detection, rare events analysis, and medical analysis. It can also be used in numerous applications, including discovery of criminal activities in e-commerce, network intrusion detection, weather prediction, video surveillance, pharmaceutical research, unusual or unknown astronomical objects or phenomena. As shown in Figure 1.2, the existing approaches to outlier detection can be classified into five categories: *distribution-based*, *clustering-based*, *depth-based*, *distance-based*, and *density-based* (Papadimitriou and Faloutsos, 2003) (Kovács et al., 2004). These approaches are described in chapter three.

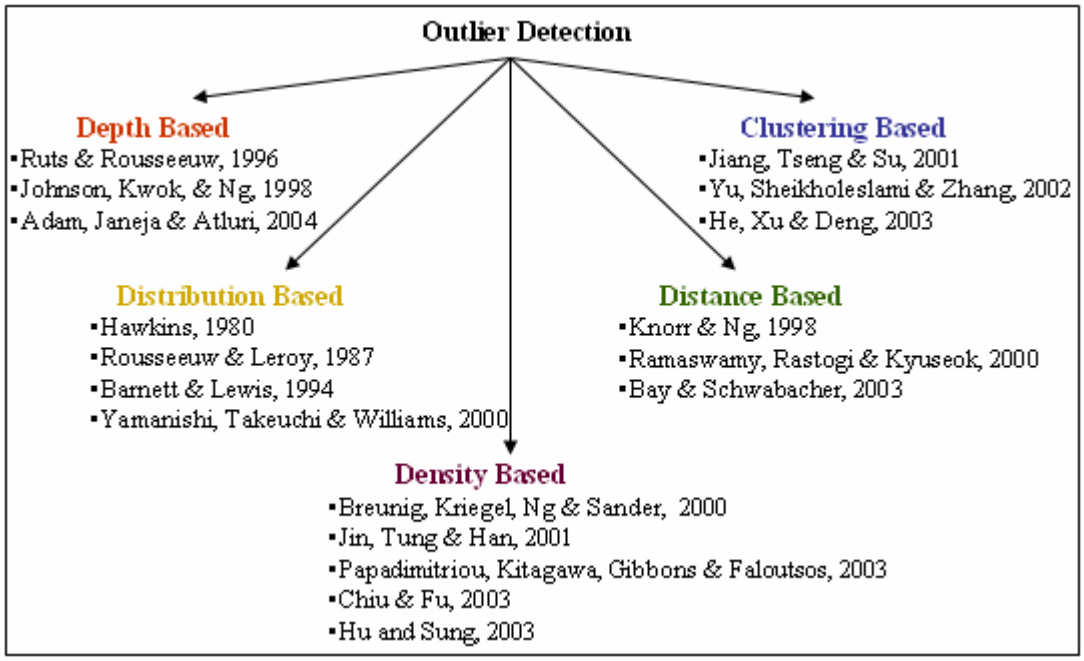


Figure 1.2 The categories of outlier detection algorithms

- Classification: Basic classification techniques used in data mining are decision trees, Bayesian classification, neural network approach and genetic algorithms.

This thesis focuses on decision tree technique. A decision tree defines groups by selecting breakpoints for the variables. For example, in the analysis of the usage of sport cars, a decision tree may determine that results divide on the age of the drivers [less than 30, over 30]. In a decision tree, an internal node denotes a test on an attribute, branch represents an outcome of the test, and leaf nodes represent class labels or class distribution.

A decision tree generation consists of two phases: tree construction and tree pruning. In tree construction phase, all the training examples are at the root and they are partitioned recursively based on selected attributes. In tree pruning phase, branches that reflect noise or outliers are identified and removed. After the construction of a decision tree, it is possible to classify an unknown sample by following the tree.

1.1.2.2 Knowledge Discovery Process

Knowledge Discovery from Databases (KDD) includes several steps for finding useful patterns in data (Figure 1.3). Data mining is only a particular step in the process of knowledge discovery. Other steps can be summarized as: (Seidman, 2001)

- Data Integration: Combining multiple data sources such as from Database, Data Cubes, Flat Files, etc.

- Selection: Creating a target dataset or a data sample on which knowledge discovery will be performed.

- Preprocessing: Correcting inconsistent data, deciding on strategies for getting missing data, smoothing out noisy data.

- Transformation: Finding features to represent the data depending on the aim of the application, reducing number of variables under construction, converting data into common format, and transforming data into new format.

- Data Mining: Choosing an appropriate data mining algorithm according to the purpose of the model, deciding on the parameters of the algorithm, applying this data mining algorithm on transformed data.

- Interpretation and Knowledge Presentation: Interpreting the extracted patterns including visualization of these patterns and translating into terms understandable by users.

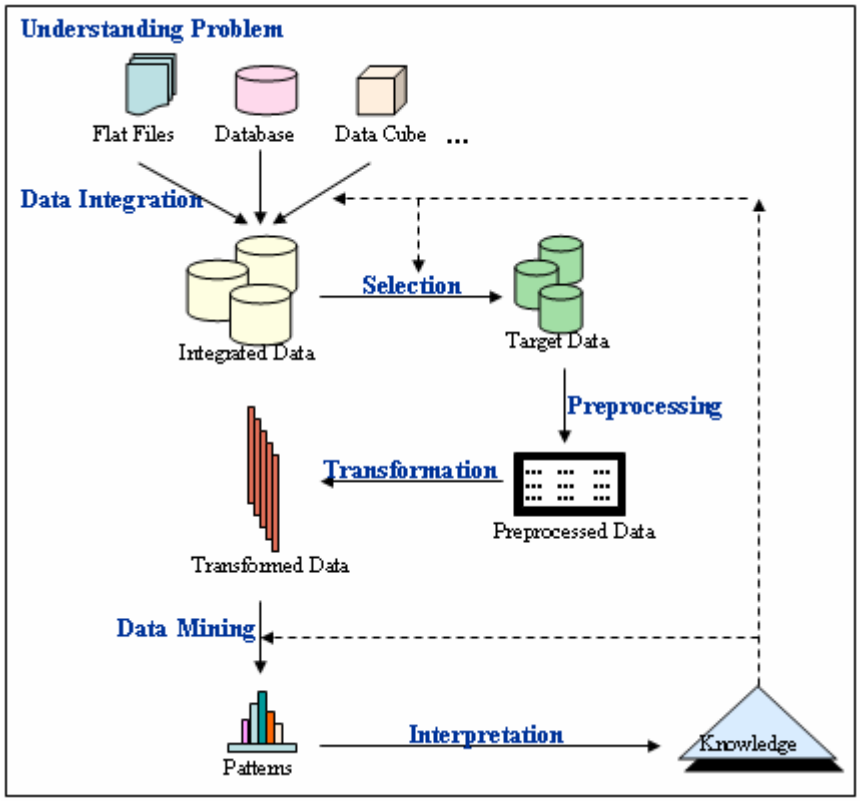


Figure 1.3 The steps of knowledge discovery process

At the beginning of the KDD process, the analyst should search and understand prior knowledge about the application domain and the aim of the application.

1.1.3 Web Services and Parallel Computing

Processing large datasets is troublesome and time consuming. The need for more CPU power is driven researchers to take advantage of the parallel paradigm. By applying parallel processing techniques, both processing time and response time can be reduced. However, writing a parallel program requires some special knowledge and experience about a parallel programming interface such as MPI (Message Passing Interface) or PVM (Parallel Virtual Machine). It is necessary to learn the syntax of the commands to provide messaging between machines. In order to overcome these difficulties, a new approach was proposed in this thesis: web service-based parallel computing. This approach includes the parallel execution of web services. Each web service is responsible for processing one portion of data, so the work load is distributed over several web services. The purpose of the study is to allow users to satisfy their parallel processing needs without the knowledge of any parallel programming interface such as MPI or PVM. This approach is useful to simplify the steps of developing parallel processing programs.

1.2 The Purpose of the Thesis

This thesis is processed for two different types of purposes: (i) purposes for marine science and (ii) purposes for computer science.

From marine science point of view, the first purpose was to overcome the failure of existing archiving activities in the Institute of Marine Sciences and Technology (IMST) at Dokuz Eylul University. Previously, data was stored in various machines of the staff in different formats and they are not all published. Data accessibility was unsatisfactory for most potential users. It had to be unified and made easily accessible to promote marine information technology. In order to make the data more readily available to users, data accumulation and data acquisition procedures had to be identified by applying pre-processing operation. Pre-processing operation was necessary to arrange raw data into a common format and to clean inconsistent, incorrect and missing data. For these purposes, a marine data center was modeled

and constructed. The primary goal of marine data center is scientific researches and analyses.

Another purpose was to construct an interactive system to access, process and present data contained within new constructed data center. The system had to involve not only the actual dispensing of data but also its display in an easily understood format. For example, data representation of oceanic conditions such as water temperature, current salinity, and water depth are more easily understood when presented through a pictorially rich graphical user interface as opposed to lists and tables.

Another objective of this study was to allow users to monitor collected marine data without learning a database tool, a programming language or a query language. Marine scientists are not required to know the details of the internal data storage format and they are not required to have programming experience. The users should easily observe marine data via a user-friendly interface. Data supply with the Internet is also indispensable for processing marine data because in this way, the users don't need to install any software. They will be able to monitor collected marine data through the Internet browser from any location in the world at any time.

From Computer Science point of view, the first intention was to develop new approaches related with the topic data mining. We have introduced two new data mining algorithms which have many advantages over existing data mining algorithms. Useful marine information can be extracted and many unexpected marine information can be discovered by applying these new data mining algorithms on collected marine data. Before applying a data mining tool, the collected data have to be undergone some preprocessing and filtering procedures and have to be stored in a data warehouse. In literature, there are too few studies for mining marine data.

Another intention in this thesis was to introduce a new design and implementation techniques related with web services. We have proposed a new approach: web service-based parallel computing. (Kut and Birant, 2004) The purpose of the

approach is to allow users to satisfy their parallel programming needs without knowledge of any parallel programming tool.

One of the purposes of this thesis was the providing of new computer based technologies for visualization of marine data. Because the existing visualization tools have some disadvantages. Firstly, some of them are windows applications, so it is necessary to install the program to all clients. Secondly, some of the existing tools use external graphic libraries which make the execution of the program slower. Thirdly, they are generally too expensive. In order to overcome these disadvantages, an algorithm was developed for drawing and painting the maps according to the stored data within the database and for showing physical, chemical and biological values within the database on the map. The visualization tool was developed on web environment.

1.3 Thesis Organization

The thesis consists of nine chapters.

Chapter 2 presents our works on clustering technique in data mining. Firstly, it gives the preliminaries and basic concepts of density-based clustering algorithms. Then, it describes the drawbacks of existing density-based clustering algorithms and our efforts to overcome these problems. It explains our algorithm in detail and presents performance evaluation of the algorithm. It also presents two applications that were implemented to demonstrate the applicability of our algorithm to real world problems. Finally, this chapter shows and discusses the data mining results.

Chapter 3 presents our works on outlier detection technique in data mining. Firstly, it describes the existing approaches related with the problem of outlier detection. Then, it explains our algorithm to detect spatio-temporal outliers. Using a real-world dataset, it presents an application to demonstrate our solution and shows the data mining results.

Chapter 4 presents our works on classification technique in data mining. It presents the decision trees constructed for analyzing physical parameters of marine data by using SQL Server Analysis Services.

Chapter 5 firstly explains the concept of sensitivity analysis and then presents the sensitivity analysis of our algorithm to the input parameters. The results presented in this chapter show the sensitivity degree of the algorithm to parameter settings.

Chapter 6 includes a new approach which was proposed to allow users to satisfy their parallel processing needs without the knowledge of any parallel programming interface such as MPI (Message Passing Interface) or PVM (Parallel Virtual Machine). It shows how web service-based parallel clustering can be done to quickly obtain information.

Chapter 7 shows the validation and evaluation of the clustering results of the new algorithm, ST-DBSCAN. It presents the mathematical quality and reliability of the clustering results by using a cluster validation technique.

Chapter 8 presents the capabilities of developed visualization tool which is able to monitor collected data through the Internet browser from any location in the world at any time.

Chapter 9 concludes with a summary and identification of key contributions and main findings of this thesis. It also addresses the possible avenues of further research based on this work.

CHAPTER TWO
CLUSTER ANALYSIS

2.1 Overview

Clustering is one of the major data mining methods for knowledge discovery in large databases. It is the process of grouping large data sets according to their *similarity*. Cluster analysis is a major tool in many areas of engineering and scientific applications including data segmentation, discretization of continuous attributes, data reduction, outlier detection, noise filtering, pattern recognition and image processing. In the field of Knowledge Discovery in Databases (KDD), cluster analysis is known as an unsupervised learning process, because there is no priori knowledge about the data set.

Most studies in KDD (Halkidi et al., 2001) focus on discovering clusters in ordinary data (non-spatial and non-temporal data), so they are impractical to use for clustering spatio-temporal data. Spatio-temporal data refers to data which is stored as temporal slices of the spatial dataset. Knowledge discovery from spatio-temporal data is a very promising subfield of data mining because increasingly large volumes of spatio-temporal data are collected and need to be analyzed. The knowledge discovery process for spatio-temporal data is more complex than for non-spatial and non-temporal data. Because spatio-temporal clustering algorithms have to consider the spatial and temporal neighbors of objects in order to extract useful knowledge. Clustering algorithms designed for spatio-temporal data can be used in many applications such as geographic information systems, medical imaging, and weather forecasting.

Spatio_temporal data is indexed and retrieved according to spatial and time dimensions. A time period attached to the spatial data expresses when it was valid or stored in the database. So a temporal database may support valid time, transaction time or both. Valid time denotes the time period during which a fact is true with

respect to the real world. Transaction time is the time period during which a fact is stored in the database. This study focuses on valid time aspect of temporal data.

This chapter presents a new density-based clustering algorithm ST-DBSCAN which was developed to cluster spatio-temporal data. This new algorithm is based on the algorithm DBSCAN (*Density Based Spatial Clustering of Applications with Noise*) (Ester et al., 1996). In DBSCAN, the density associated with a point is obtained by counting the number of points in a region of specified radius around the point. Points with a density above a specified threshold are constructed as clusters. Among the existing clustering algorithms, we have chosen the DBSCAN algorithm because it has the ability to discover clusters of arbitrary shape, such as linear, concave, and oval. Furthermore, in contrast to some clustering algorithms, it does not require the pre-determination of the number of clusters. DBSCAN has proven its ability to process very large databases (Aoying & Shuigeng, 2000; Ester et al., 1996; Ester et al., 1998a).

The DBSCAN algorithm was improved in three important ways. First, unlike the existing density-based clustering algorithms, the new algorithm can cluster spatio-temporal data according to its non-spatial, spatial and temporal attributes. Second, DBSCAN cannot detect some noise objects when clusters of different densities exist. Our algorithm solves this problem by assigning to each cluster a *density factor*. Third, the values of border objects in a cluster may be very different to the values of border objects on the opposite side of the cluster, if the non-spatial values of neighbor objects have little differences and the clusters are adjacent to each other. Our algorithm solves this problem by comparing the average value of a cluster with the new incoming values.

2.2 Related Works and Basic Concepts

2.2.1 Clustering Algorithms

The problem of clustering can be defined as follows:

Definition 2.1 *Given a database of n data objects $D=\{o_1, o_2,...,o_n\}$. The process of partitioning D into $C=\{C_1,C_2,,...,C_k\}$ based on a certain similarity measure is called clustering, C_i's are called clusters, where $C_i \subseteq D$, (i =1,2,...,k), $\bigcap_{i=1}^{k} C_i = \emptyset$ and $\bigcup_{i=1}^{k} C_i = D$.*

Clustering algorithms can be categorized into five main types (Han and Kamber, 2001): *Partitional*, *Hierarchical*, *Grid-based*, *Model-based* and *Density-based* clustering algorithms.

In *Partitional* clustering algorithms, cluster similarity is measured in regard to the mean value of the objects in a cluster, center of gravity, (K-Means (MacQueen, 1967)) or each cluster is represented by one of the objects of the cluster located near its center (K-Medoid (Vinod, 1969)). K is an input parameter for these algorithms, unfortunately it is not available for many applications. In addition, these algorithms do not identify arbitrarily shaped clusters due to their reliance only on the distance between points. CLARANS (Ng & Han, 1994) is an improved version of K-Medoid algorithm for mining in spatial databases.

Hierarchical clustering algorithms such as CURE (Guha et al., 1998), BIRCH (Zhang et al., 1996) produce a set of nested clusters organized as a hierarchical tree. Each node of the tree represents a cluster of a database *D*. The tree can be constructed from leaves to root (*agglomerative approach*) or from root to leaves (*divisive approach*). In contrast to partitioning algorithms, hierarchical algorithms do not require the number of clusters, *k*, as an input. However, they need a termination condition, e.g., the desired number of clusters or a distance threshold for merging. In addition, they are suitable for meaningful taxonomies; e.g., animal kingdom.

Grid-based clustering algorithms are based on multiple-level grid structure on which all operations for clustering are performed. STING (STatistical INformation Grid approach) (Wang et al., 1997) explores statistical information stored in the grid cells, such as the mean, standard deviation, maximum, and minimum values. WaveCluster (Sheikholeslami et al., 1998) applies wavelet transformation for

clustering analysis. In CLIQUE (Clustering In QUEst) (Agrawal et al., 1998) algorithm, the clustering process starts at single-dimensional subspaces and grows upwards to higher dimensional ones.

In Model-based clustering algorithms, a model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other. They are often based on the assumption that the data are generated by a mixture of underlying probability distributions. They generally attempt to optimize the fit between the data and some mathematical model. COBWEB (Fisher, 1987) is a conceptual learning algorithm that performs probability analysis and takes concepts as a model for clusters.

The *Density-based* notion is a common approach to clustering. Density-based clustering algorithms are based on the idea that objects which form a *dense* region should be grouped together into one cluster. They use a fixed threshold value to determine *dense* regions. They search for regions of high density that are separated by regions of lower density in a feature space.

Density-based clustering algorithms such as DBSCAN (Ester et al., 1996), OPTICS (Ankerst et al., 1999), DENCLUE (Hinneburg & Keim, 1998), and CURD (Ma et al., 2003) are to some extent capable of clustering databases. (Qian & Zhou, 2002) One drawback of these algorithms is that they capture only certain kinds of noise objects when clusters of different densities exist. Furthermore, they are adequate if the clusters are distant from each other, but not satisfactory when clusters are adjacent to each other. The detailed description of these problems and our solutions are given in section 2.3.

In our study, we have chosen the DBSCAN algorithm, because it has the ability in discovering clusters with arbitrary shape such as linear, concave, oval, etc. Furthermore, in contrast to some clustering algorithms, it does not require the pre-determination of the number of clusters. DBSCAN has been proven in its ability of processing very large databases (Aoying & Shuigeng, 2000) (Ester et al., 1998a).

2.2.2 Basic Concepts

DBSCAN was designed to discover arbitrary-shaped clusters in any database *D*, and at the same time it can distinguish noise objects. More specifically, DBSCAN accepts a radius value *Eps* (ε) based on a user-defined distance measure and a value *MinPts* for the minimum number of points that should occur within the *Eps* radius. Some concepts and terms to explain the DBSCAN algorithm are defined as follows:

Definition 2.2 *(Neighborhood). Neighborhood is determined by a distance function (e.g., the Manhattan Distance or Euclidean Distance) for two points p and q, denoted by dist(p,q).*

Definition 2.3 *(Eps-neighborhood). The Eps-neighborhood of a point p is defined as* $\{q \in D \mid dist(p,q) \leq Eps\}$.

Definition 2.4 *(Core Object). A core object refers to a point such that its neighborhood of a given radius (Eps) contains at least a minimum number (MinPts) of other points.* (*Figure 2.2)*

Definition 2.5 *(Directly Density-Reachable). An object p is directly density-reachable from the object q if p is within the Eps-neighborhood of q, and q is a core object. (Figure 2.1.a)*

Definition 2.6 *(Density-Reachable). An object p is density-reachable from the object q with respect to Eps and MinPts if there is a chain of objects p1, ...pn, p1=q and pn=q such that pi+1 is directly density-reachable from pi with respect to Eps and MinPts, for* $1 \leq i \leq n$, $pi \in D$. *(Figure 2.1.b)*

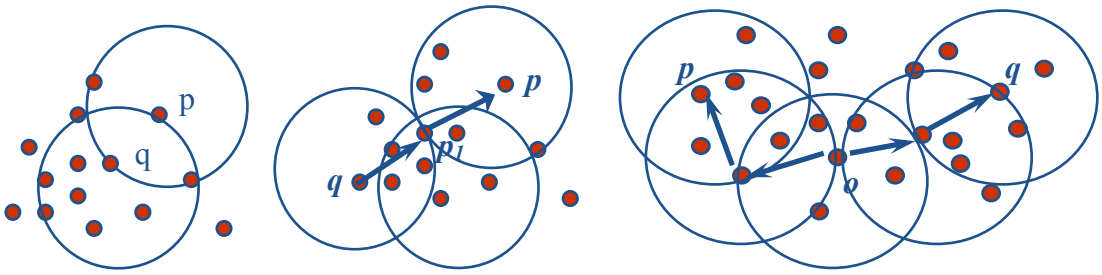
Definition 2.7 *(Density-Connected). An object p is density-connected to object q with respect to Eps and MinPts if there is an object* $o \in D$ *such that both p and q are density-reachable from o with respect to Eps and MinPts. (Figure 2.1.c)*

Definition 2.8 *(Density-Based Cluster). A cluster C is a non-empty subset of D satisfying the following "Maximality" and "Connectivity" requirements:*

1) ∀ p,q:if q∈C and p is density-reachable from q with respect to Eps and MinPts, then p∈C.

2) ∀ p,q∈C: p is density-connected to q with respect to Eps and MinPts.

Definition 2.9 *(Border Object). An object p is a border object if it is not a core object but is density-reachable from another core object.* (*Figure 2.2)*



(a) (b) (c)

Figure 2.1 Basic concepts and terms: (a) p is directly density-reachable from q (b) p is density-reachable from q (c) p and q are density-connected to each other by o

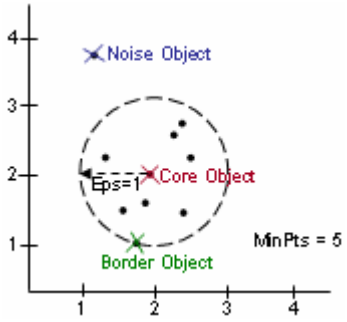


Figure 2.2 Border, Core and Noise object

The algorithm starts with the first point *p* in database *D*, and retrieves all neighbors of point *p* within distance *Eps*. If the total number of these neighbors is greater than *MinPts*, i.e if *p* is a core object, a new cluster is created. The point *p* and its neighbors are assigned into this new cluster. Then, the algorithm iteratively

collects the neighbors within distance *Eps* from the core points. The process is repeated until all of the points have been processed.

2.2.3 DBSCAN Based Algorithms

In the literature, the DBSCAN algorithm has been used in many studies. For example, the other popular density-based algorithm OPTICS (Ordering Points To Identify the Clustering Structure) (Ankerst et al., 1999) is based on the concepts of the DBSCAN algorithm and identifies nested clusters and the structure of clusters.

Incremental DBSCAN (Ester et al., 1998) algorithm is also based on the clustering algorithm DBSCAN and is used for incremental updates of a clustering after insertion of a new object to the database and deletion of an existing object from the database. Based on the formal notion of clusters, the incremental algorithm yields the same results as the non-incremental DBSCAN algorithm.

The SDBDC (Scalable Density-Based Distributed Clustering) (Januzaj et al., 2004) method also uses DBSCAN on both local sites and global site to cluster distributed objects. In this method, DBSCAN is first carried out on each local site. Then, based on these local clustering results, cluster representatives are determined, and then based on these local representatives, the standard DBSCAN algorithm is carried out on the global site to construct the distributed clustering. This study proposes the usage of different *Eps*-values for each local representative.

Wen et al. adopted DBSCAN and Incremental DBSCAN as the core algorithms of their query clustering tool. They used DBSCAN to cluster frequently asked questions and the most popular topics on a search engine. Spieth et al. applied DBSCAN to identify solutions for the inference of regulatory networks. Finally, SNN density-based clustering algorithm (Tan et al., 2005) is also based on DBSCAN and it is applicable to high-dimensional data consisting of time series data of atmospheric pressure at various points on the earth.

2.3 Problems of Existing Approaches

2.3.1 Problem of Clustering Spatio-temporal Data

To determine whether a set of points is *similar* enough to be considered a cluster, we need a distance measure dist(i, j) that tells us how far apart points i and j are. The most common distance measures used are the Manhattan distance, Euclidean distance, and Minkowski distance. Euclidean distance is defined in Equation 2.1.

$$\text{dist}(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{in} - x_{jn})^2} \quad (2.1)$$

where $i=(x_{i1}, x_{i2},..., x_{in})$ and $j=(x_{j1}, x_{j2},..., x_{jn})$ are two n-dimensional data objects. For example, the Euclidean distance between the two data objects A(1, 2) and B(5, 3) is 4.12.

The DBSCAN algorithm uses only one distance parameter, *Eps*, to measure *similarity* of spatial data with one dimension. To support two dimensional spatial data, we propose two distance metrics, *Eps1* and *Eps2*, to define the *similarity* by a conjunction of two density tests. *Eps1* is used for *spatial values* to measure the closeness of two points geographically. *Eps2* is used to measure the *similarity* of *non-spatial values*. For example, *A(x1, y1)* and *B(x2, y2)* are two points (spatial values), *t1, t2* (*DayTimeTemperature, NightTimeTemperature*) and *t3, t4* are four temperature values of these points respectively (non-spatial values). In this example, *Eps1* is used to measure the closeness of two points geographically, whereas *Eps2* is used to measure the *similarity* of temperature values. If *A(x1,y1,t1,t2)* and *B(x2,y2,t3,t4)* are two points, *Eps1* and *Eps2* are calculated by the formulas in Equation 2.2.

$$\begin{aligned} \text{Eps1} &= \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \\ \text{Eps2} &= \sqrt{(t1 - t2)^2 + (t1 - t2)^2} \end{aligned} \quad (2.2)$$

The existing clustering algorithms are generally impractical to use for clustering spatio-temporal data. In order to support temporal aspects, spatio-temporal data is first filtered by retaining only the temporal neighbors and their corresponding spatial values. Two objects are temporal neighbors if the values of these objects are observed in consecutive time units such as consecutive days in the same year or in the same day in consecutive years.

2.3.2 Problem of Identifying Noise Objects

From the view of a clustering algorithm, noise is a set of objects not located in clusters of a database. More formally, noise can be defined as follows:

Definition 2.10 *(Noise). Let C_1 ,...,C_k be the clusters of the database D. Then the noise is the set of points in the database D not belonging to any cluster C_i, where i = 1,...,k i.e. noise = $\{p \in D \mid i: p \notin C_i\}$*

Existing density-based clustering algorithms (Han et al., 2001) produce meaningful and adequate results under certain conditions, but their results are not satisfactory when clusters of different densities exist. To illustrate this point, consider the example given in Figure 2.3. This is a simple dataset containing 52 objects. There are 25 objects in the first cluster C_1, 25 objects in the second cluster C_2, and two additional noise objects o_1 and o_2. In this example, C_2 forms a denser cluster than C_1; in other words, the densities of the clusters are different. The DBSCAN algorithm identifies only one noise object o_1 because approximately for every object p in C_1, the distance between the object p and its nearest neighbor is greater than distance between o_2 and C_2. For this reason, we can't determine an appropriate value for the input parameter *Eps*. If the *Eps* value is less than the distance between o_2 and C_2, some objects in C_1 are assigned as noise object. If the *Eps* value is greater than the distance between o_2 and C_2, the object o_2 is not assigned as a noise object.

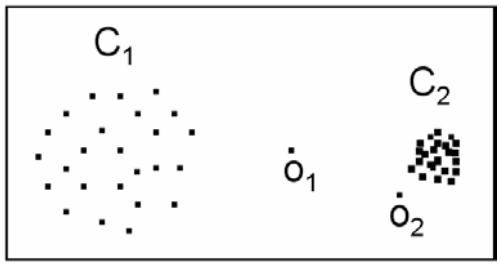


Figure 2.3 Example data set containing clusters with different densities

The DBSCAN algorithm is not satisfactory when clusters of different densities exist. To overcome this problem, we propose two new concept: *density factor* and *density distance*. We assign to each cluster a *density factor*, which is the degree of the density of the cluster.

Definition 2.11 *(Density_Distance). Let density_distance_max of an object p denote the maximum distance between the object p and its neighbor objects within the radius Eps. Similarly, let density_distance_min of an object p denote the minimum distance between the object p and its neighbor objects within the radius Eps.*

(i) density_distance_max (p) = max { dist(p,q) | q ∈ D ∧ dist(p,q) ≤ Eps }

(ii) density_distance_min (p) = min { dist(p,q) | q ∈ D ∧ dist(p,q) ≤ Eps }

The density_distance of an object p is defined as density_distance_max (p) / density_distance_min (p).

Density factor of a cluster can be defined as follows.

Definition 2.12 *(Density_Factor). The density_factor of a cluster C is defined in Equation 2.3.*

$$\text{density\_factor}(C) = 1/\left[\frac{\sum_{p \in C} \text{density\_distance}(p)}{|C|}\right] \tag{2.3}$$

The *density factor* of a cluster C captures the degree of the density of the cluster. If *C* is a "loose" cluster, density_distance_min would increase and so the *density*

distance in definition 11 would be quite small, thus forcing the *density factor* of *C* to be quite close to 1. Otherwise, if *C* is a "tight" cluster, density_distance_min would decrease and so the *density distance* in definition 11 would be quite big, thus forcing the *density factor* of *C* to be quite close to 0.

2.3.3 Problem of Identifying Adjacent Clusters

The existing density-based clustering algorithms (Kolatch, 2001) are adequate if the clusters are distant from each other, but not satisfactory when clusters are adjacent to each other (Figure 2.4). If the values of neighbor objects have small differences, the values of border objects in a cluster may be very different to the values of other border objects on the opposite side. In other words, the value of a border object may be very different to the value of the border object furthest away from it because small value changes in neighbors can cause big value changes between starting points and ending points of a cluster. However, cluster objects should be within a certain distance from the cluster means. We solve this problem by comparing the average value of a cluster with the new incoming value. If the absolute difference between Cluster_Avg() and Object_Value is bigger than the threshold value, $\Delta\epsilon$, then the new object is not appended to the cluster. Cluster_Avg() refers to the average or mean value of the objects contained in the cluster and Object_Value refers to the non-spatial value of the object such as the temperature value of a location.

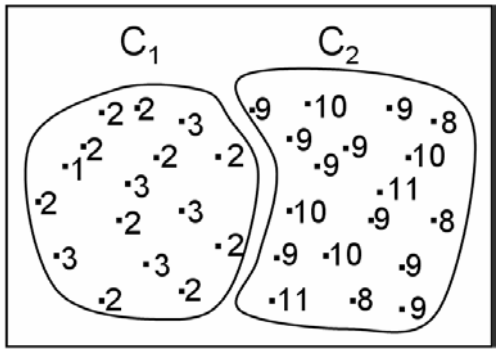


Figure 2.4 Example data set containing adjacent clusters

2.4 ST-DBSCAN Algorithm

2.4.1 The Description of the Algorithm

Whereas the DBSCAN algorithm needs two inputs, the new algorithm, ST-DBSCAN, requires four parameters *Eps1*, *Eps2*, *MinPts*, and $\Delta\epsilon$ because of the extensions described in section 3. *Eps1* is the distance parameter for spatial attributes (latitude and longitude) and *Eps2* is the distance parameter for non-spatial attributes. A distance metric such as the Euclidean, Manhattan or Minkowski distance metric can be used for *Eps1* and *Eps2*. *MinPts* is the minimum number of points within the *Eps1* and *Eps2* distances of a point. If a region is dense, then it should contain more points than *MinPts*.

In (Ester et al., 1996), a simple heuristic is presented which is effective in many cases to determine the parameters *Eps* and *MinPts*. The heuristic suggests *MinPts* ≈ *ln(n)* where *n* is the size of the database and *Eps* must be picked depending on the value of *MinPts*. The first step of the heuristic method is to determine the distances to the *k-nearest neighbors* for each object, where *k* is equal to *MinPts*. Next, these *k-distance values* should be sorted in descending order and then the threshold point which is the first "valley" of the sorted graph is determined as shown in Figure 2.5. The value of the *Eps* parameter should be selected as less than the distance defined by the first valley. The last parameter $\Delta\epsilon$ is used to prevent the discovering of combined clusters because of the small differences in non-spatial values of the neighboring locations.

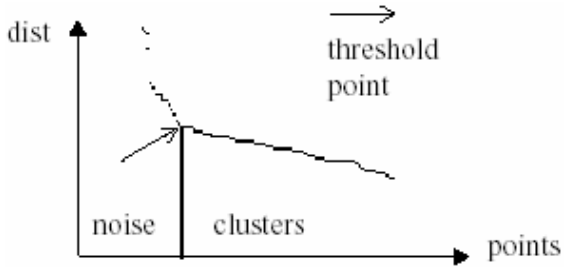


Figure 2.5 Sorted distance graph for a sample database (Ester et al., 1996)

The algorithm starts with the first point *p* in database *D* and retrieves all points density-reachable from *p* with respect to *Eps1* and *Eps2*. If *p* is a core object (see Definition 2.4), a cluster is formed. If *p* is a border object (see Definition 2.9), no points are density-reachable from *p* and the algorithm visits the next point of the database. The process is repeated until all the points have been processed.

As shown in Figure 2.6, the algorithm starts with the first point in database *D* (i). After processing this point, it selects the next point in *D*. If the selected object does not belong to any cluster (ii), the *Retrieve_Neighbors* function is called (iii). A call of *Retrieve_Neighbors(object, Eps1, Eps2)* returns the objects that have a distance less than the *Eps1* and *Eps2* parameters to the selected object. In other words, the *Retrieve_Neighbors* function retrieves all objects *density-reachable* (see Definition 2.6) from the selected object with respect to *Eps1, Eps2,* and *MinPts*. The result set forms the *Eps-Neighborhood* (Definition 2.3) of the selected object. *Retrieve_Neighbours(object, Eps1, Eps2)* is equal to the intersection of *Retrieve_Neighbours(object, Eps1)* and *Retrieve_Neighbours(object, Eps2)*. If the total number of returned points in the *Eps-Neighborhood* is smaller than the *MinPts* input, the object is assigned as noise (iv). This means that the selected point has not enough neighbors to be clustered. The points marked as noise may be changed later, if they are not directly density-reachable (Definition 2.5) but they are density-reachable (see Definition 2.6) from some other point of the database. This happens for border points of a cluster.

If the selected point has enough neighbors within distances *Eps1* and *Eps2*, i.e. if it is a core object, then a new cluster is constructed (v). Then all directly density-reachable neighbors of this core object are also marked as new cluster label. Next, the algorithm iteratively collects density-reachable objects from this core object by using a stack (vi). The stack is necessary to find density-reachable objects from directly density-reachable objects. If the object is not marked as noise or it is not in a cluster, and the difference between the average value of the cluster and the new incoming value is smaller than $\Delta\epsilon$, it is placed in the current cluster (vii). After

processing the selected point, the algorithm selects the next point in *D* and algorithm continues iteratively until all the points have been processed.

```
Algorithm ST_DBSCAN (D, Eps1, Eps2, MinPts, Δε)
   // Inputs:
       // D={o1, o2, ..., on}  Set of objects
       // Eps1 : Maximum geographical coordinate (spatial) distance value.
       // Eps2 : Maximum non-spatial distance value.
       // MinPts : Minimum number of points within Eps1 and Eps2 distance.
       // Δε : Threshold value to be included in a cluster.
   // Output:
       // C={C1, C2, … Ck} Set of clusters

 Cluster_Label = 0

 For i=1 to n                                        //(i)
   If oi is not in a cluster Then                    //(ii)
      X=Retrieve_Neighbors(oi , Eps1, Eps2)          //(iii)

      If |X| < MinPts Then
            Mark oi as noise                         //(iv)
      Else                              //construct a new cluster (v)
         Cluster_Label = Cluster_Label + 1

         For j=1 to |X|
           Mark all objects in X with current Cluster_Label
         End For

         Push(all objects in X)                      //(vi)

         While not IsEpmty()
            CurrentObj = Pop()
            Y= Retrieve_Neighbors(CurrentObj, Eps1, Eps2)

            If |Y| >= MinPts Then
               ForAll objects o in Y                 //(vii)
                  If (o is not marked as noise or it is not in a cluster) and
                     |Cluster_Avg() - o.Value| <= Δε Then
                     Mark o with current Cluster_Label
                     Push(o)
                  End If
               End For
            End If
         End While
      End If
   End If
 End For
End Algorithm
```

Figure 2.6 ST_DBSCAN algorithm

When the algorithm searches the neighbors of any object by using the *Retrieve_Neighbors* function (line iii in the algorithm), it takes into consideration both spatial and temporal neighborhoods. The non-spatial value of an object such as a temperature value is compared with the non-spatial values of spatial neighbors and

also with the values of temporal neighbors (previous day in the same year, next day in the same year, and the same day in other years). In this way, non-spatial, spatial and temporal characteristics of data are used in clustering when the algorithm is applied to the table that contains temporal values, as well as spatial and non-spatial values.

If two clusters C_1 and C_2 are very close to each other, a point p may belong to both, C_1 and C_2. In this case, the point p must be a border point in both C_1 and C_2. (Figure 2.7) The algorithm assigns point p to the cluster discovered first.

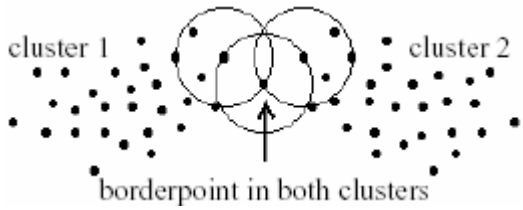


Figure 2.7 Overlap of two clusters

2.4.2 Performance Evaluation of the Algorithm

The average runtime complexity of the DBSCAN algorithm is O(n*logn), where n is the number of objects in the database. Our modifications do not change the runtime complexity of the algorithm. DBSCAN has proven its ability to process very large databases. The paper (Ester et al., 1998a) shows that the runtime of other clustering algorithms such as CLARANS (Ng & Han, 1994), DBCLASD (Xu et al., 1998) is between 1.5 and 3 times the runtime of DBSCAN. This factor increases with increasing size of the database.

As in all databases, fast access to raw data in spatio-temporal databases depends on the structural organization of the stored information and the availability of suitable indexing methods. A well designed data structure can facilitate the rapidl extraction of the desired information from a set of data, and suitable indexing methods can quickly locate single or multiple objects. (Abraham & Roddick, 1999) Well known spatial indexing techniques include Quadtrees (Samet, 1990), R-Trees

(Guttman, 1984), X-Tree (Böhm et al., 2000) and others, see (Guting, 1994) for an overview. An *R-Tree* is a spatial indexing technique that stores information about spatial objects such as object identifiers and the Minimum Bounding Rectangles (MBRs) of the objects or groups of objects**.** Each entry of a leaf node is of the form *(R, P)* where *R* is a rectangle that encloses all the objects that can be reached by following the node pointer *P*. In this study, an improvement to the R-Tree indexing method was used to handle spatio-temporal information. In R-Tree, some nodes were created for each spatial object and they were linked in temporal order. During the application of the algorithm, this tree is traversed to find the spatial or temporal neighbor objects of any object. Two objects are temporal neighbors if the values of these objects are observed on consecutive time units such as consecutive days in the same year or in the same day in consecutive years.

In addition to the spatial index structure, some filters should also be used to reduce the search space for spatial data mining algorithms. These filters allow operations on neighborhood paths by reducing the number of paths actually created. They are necessary to speed up the processing of queries.

2.5 Application

In order demonstrate the applicability of our algorithm to real world problems; we developed two data mining applications by using a spatio-temporal data warehouse. The task of clustering is to discover regions that have *similar* seawater characteristics. The aim of the clustering is obtain a number of clues about how the physical properties of the water are distributed in a marine environment. The first application discovers the regions that have *similar* sea surface temperature values. In the second application, the goal is to identify spatially based partitions that have *similar* sea surface height residual values.

Figure 2.8 shows the structure of the system. In the visualization part of the study, remotely sensed data on the historical extent of marine areas were used in a spatial metrics analysis of the geographical form of countries and islands. User-friendly

interfaces were developed, allowing relatively inexperienced users to operate the system. Special functions were developed for data integration, data conversion, querying, visualization, analysis and management. marine environmental data (e.g. sea surface temperature, wave height values, bathymetric data), from a variety of sources, were integrated as coverages, grids, shape-files, and tables.

The process of KDD involves several steps such as data integration and selection, data preprocessing and transformation, data mining, and the evaluation of the data mining results. Our efforts at each step are described below.

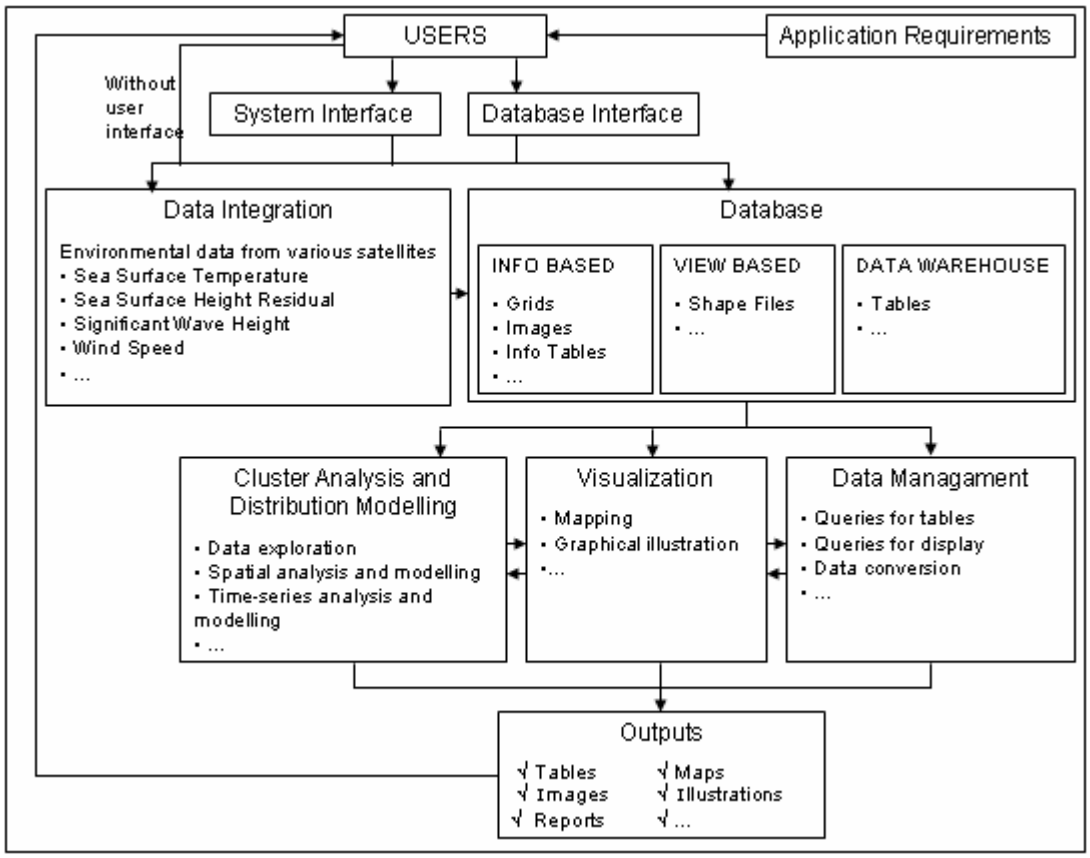


Figure 2.8 Schematic diagram of the system

2.5.1 Data Integration and Selection

We designed a data warehouse that contains information about four seas: the Black Sea, the Marmara Sea, the Aegean Sea, and the eastern part of the Mediterranean Sea. These seas surround the countries Turkey to the north, west, and

south; Greece to the east, south, and west; and Cyprus. The geographical coordinates of our work area are 30^{o} to 47.5^{o} north latitude and 17.0^{o} to 42.5^{o} east longitude.

As shown in Figure 2.9, the data model contains a central fact table, STATIONS, which interconnects the tables: Sea_Surface_Temperature, Sea_Surface_Height, Wave_Height, and Sea_Winds. The data size is approximately 8 GB. In the data warehouse, the dimensions are time and space. The time dimension can be grouped into Year, Month, and Day. Similarly, the space dimension can be grouped into StationID, RegionID, and ClusterID. The first column, StationID, identifies the geographic location of the monitoring station. The column, RegionID, identifies the name of the sea (Black Sea, Marmara Sea, Aegean Sea, or Mediterranean Sea). The last column, ClusterID, identifies a particular cluster of stations that have similar characteristics.

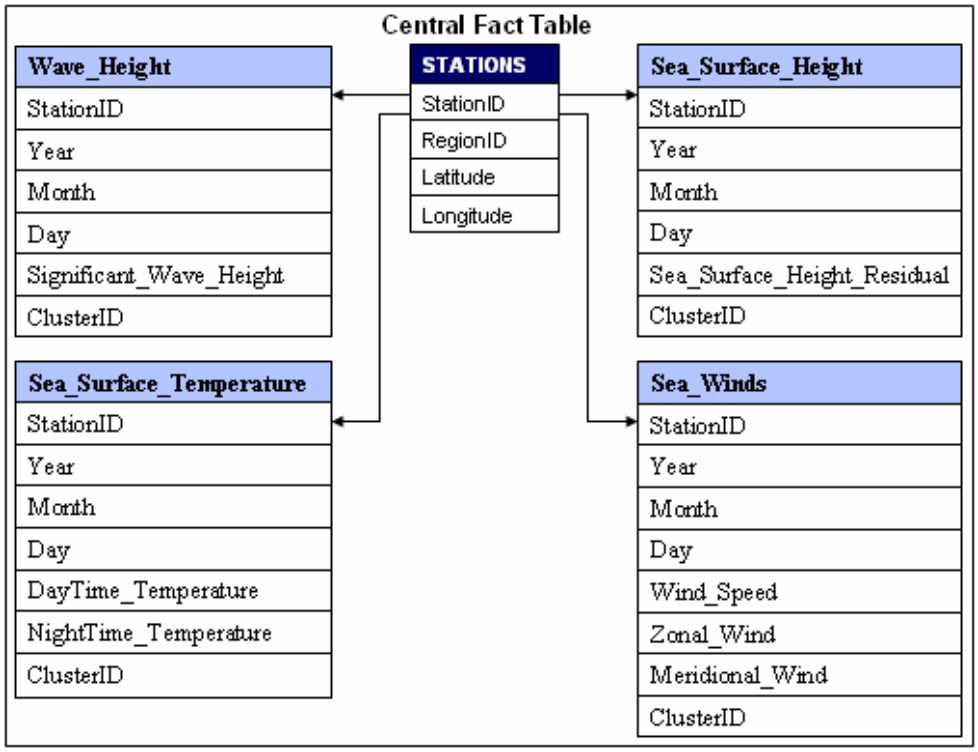


Figure 2.9 General overview of data warehouse

The Sea_Surface_Temperature table contains weekly daytime and nighttime temperature records from 2001 to 2004. The data was provided by National Oceanic and Atmospheric Administration (NOAA) satellites. (NASA, 2003b). It contains

approximately 1.5 million rows. Data in the Sea_Surface_Height table were provided by the Topex/Poseidon satellite (NASA, 2003c) and were collected over five-day periods between 1992 and 2002. The Wave_Height table contains significant wave height values that were collected over ten-day periods between 1992 and 2002. Similar to the significant sea surface height values, the significant wave height values were also provided by the Topex/Poseidon satellite. The Sea_Winds table contains information about wind speed, zonal wind and meridional wind. The data were measured daily between 1999 and 2004 and were provided by the QuikSCAT satellite (NASA, 2003d).

2.5.2 Data Preprocessing

Satellite data generally contain false information and sometimes several values can be missing. In this study, missing values were filled with the average of adjacent object values. The missing values were generally located at the coasts of the Aegean Sea. Because the Aegean coast is extremely indented with numerous gulfs and inlets.

The maps derived from the NOAA-AVHRR (Polar Orbiting Advanced Very High Resolution Radiometer) are used to compute Sea Surface Temperatures (SSTs) by applying the Multi-Channel Sea Surface Temperature algorithm (MCSST). The latest version of this algorithm uses the following formula in the calculation of the SST:

$$SST = a * T4 + b * (T4 - T5) * Tf + c * (\sec(q) - 1) * (T4 - T5) - d \qquad (2.4)$$

where q is the satellite zenith angle or the incidence angle of the incoming radiation based on the horizontal plane of the satellite, T4 and T5 are the brightness temperatures from AVHRR channels 4 and 5, respectively. Tf is a first-guess SST estimate (obtained from the 1 km MCSST AVHRR mosaic SSTs), and a, b, c, and d are empirically derived coefficients (Walton et al., 1998). These coefficients are predetermined by comparing AVHRR radiance values to temperature measurements taken from moored and drifting buoys. For example, the nighttime and daytime equations for NOAA-14 Satellite are:

$$\begin{aligned}\text{Daytime SST} &= .9506 * T4 + .0760 * (T4 - T5) * Tf + .6839 * (\sec(0) - 1) * (T4 - T5) - 258.096 \\ \text{Nighttime SST} &= .9242 * T4 + .0755 * (T4 - T5) * Tf + .6040 * (\sec(0) - 1) * (T4 - T5) - 250.428\end{aligned} \quad (2.5)$$

The Sea Surface Height Residual values are calculated by the formula in Equation 2.6.

$$\text{SSHR} = \text{SSH} - \text{MSS} - \text{Tide Effects} - \text{Inverse Barometer} \quad (2.6)$$

where SSHR is the Sea Surface Height Residual value, SSH is the Sea Surface Height value, and MSS is the Mean Sea Surface height value. The residual sea surface is defined as the sea surface height minus the mean sea surface and minus known effects, i.e., tides and inverse barometer effects.

2.5.3 Data Transformation

Data transformation is an important aspect of data preprocessing to enable the dataset to assure the accuracy of the model. Data transformation, in terms of data mining, is the process of changing the form or structure of existing attributes. Data transformation involves converting data into a single common format acceptable to the data mining methodology. One of the most common forms of data transformation used in data mining is the normalization of the attributes. Data normalization consists in scaling the attribute values of the data into a specified range, such as -10 to 10. It is also known as range normalization. The most widely used normalization methods are z-score normalization and min-max normalization. Z-score normalization normalizes numerical attributes using the mean and standard deviation computed from the data. This method works well in cases when you do not know the actual minimum and maximum of your input data or when you have outliers that have great effect on the range of the data. It maps a value v of attribute A to v' by the following formula:

$$v' = \frac{v - mean_A}{stand\_dev_A} \tag{2.7}$$

In min-max normalization, attribute values are normalized to lie in a fixed interval given by the minimum and maximum values. It performs a linear transformation on the original data into the specified interval. The advantage of this method is that it preserves all relationships of the data values exactly. It does not introduce any potential bias into the data. It maps a value *v* of attribute *A* to *v'* in the range [new_min_A, new_max_A]) by the following formula:

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A \tag{2.8}$$

In the case studies, temperature values have to be normalized because the temperature value of the same location changes in winter season (e.g. 9ºC) and summer season (e.g.20ºC). We normalized the daytime and nighttime temperature attribute values of SST table in the range [1, 24] and sea surface height values of SSH table in the range [-14, +14]. We applied min-max normalization on the oceanographic data in the data warehouse by using following function:

```
Double min_max_norm (Double a, Double MinA, Double MaxA, Double
NewMinA, Double NewMaxA) {
      Return (a - MinA)/(MaxA - MinA)*(NewMaxA - NewMinA) + NewMinA;
}
```

2.5.4 Spatio-temporal Data Mining

In this step of the study, our clustering algorithm is applied two times to discover the spatio-temporal distributions of two physical parameters. The first application uses Sea_Surface_Temperature data to find the regions that have similar sea surface temperature characteristics. The input parameters are designated as *Eps1=3*, *Eps2=0.5*, and *MinPts=15*. The second application uses Sea_Surface_Height data to

find regions that have similar sea surface height residual values and the input parameters are designated as *Eps1=3*, *Eps2=1*, and *MinPts=4*.

2.5.5 The Evaluation of Data Mining Results

The example data warehouse contains weekly daytime and nighttime temperature records that were measured at 5340 stations between 2001 and 2004 and these stations are shown in Figure 2.10.a as black dots. The spatial distribution of temperature in surface water (30-47.5°N and 17-42.5°E) is shown in Figure 2.10.b. Table 2.1 also shows the clustering results. In the table, the first column identifies the cluster, the second column presents the number of objects in each cluster, the third and sixth columns display the average values of objects in each cluster. Other columns show the minimum and maximum values in each cluster.

Table 2.1 Clustering results for sea surface temperature data

Cluster ID	Num. of Obj.	AVG Day Temp.	MIN Day Temp.	MAX Day Temp.	AVG Night Temp.	MIN Night Temp.	MAX Night Temp.
1	63	2.74	2.09	4.59	2.71	1.00	4.55
2	92	10.05	9.60	11.10	9.67	8.39	10.75
3	1306	13.70	11.40	16.20	13.41	9.44	15.75
4	90	16.35	15.75	16.65	16.25	15.30	16.54
5	3061	20.14	16.35	22.5	20.04	16.20	22.35
6	349	14.35	13.25	15.75	14.01	13.10	15.65
7	279	22.73	21.50	24.00	22.67	21.45	23.60

In the results, each cluster has data points that have similar sea surface temperature characteristics. Cluster number 1 is bordered by Ukraine and Russia; this region is the coldest area. Cluster number 2, bordering Romania and Ukraine, is the second coldest area. The seawater temperatures of other parts of the Black sea are similar to those of the Marmara Sea. Cluster number 4 covers the north of the Aegean Sea. Cluster number 5 forms a great single cluster covering the eastern Mediterranean. The temperature values of the stations in Cluster 6 also have similar characteristics. Cluster number 7 is the hottest region, because it is the closest area to the equator. In winter, C5 and C7 clusters can be marked as one cluster, because they cannot be distinguished very clearly. In summer, C6 cluster becomes smaller.

Many factors can affect this distribution of seawater temperature. The temperature varies both attitudinally and depth-wise in response to changes in air-sea interactions. Heat fluxes, evaporation, river in flow, and the movement of water and rain all influence the distribution of seawater temperature.

The Topex/Poseidon satellite provides sea surface height residual data as a two-dimensional grid separated by one degree in latitude and longitude. So SSHR values stored in the database are available at 134 stations shown in Figure 2.11.a as black dots. The clusters obtained by using the Sea_Surface_Height table are showed in Figure 2.11.b. Table 2.2 also shows the clustering results.

Table 2.2 Clustering results for sea surface height residual data

ClusterID	Num.of Obj.	AVG(Height)	MIN(Height)	MAX(Height)
1	9	4.53	3.80	5.40
2	8	12.60	10.90	14.00
3	6	7.58	7.00	8.50
4	11	11.41	9.69	13.00
5	4	1.30	1.20	1.50
6	17	-11.21	-14.00	-9.40
7	6	-0.91	-1.40	-0.40
8	7	2.51	1.80	3.80
9	29	-0.86	-6.00	3.50
10	32	-4.31	-5.90	-2.90

In the results, each cluster has data points that have similar sea surface height residual values. Clusters C1, C2, C3 and C4 are located in the Black Sea and the cluster C7 is located in the Aegean Sea. The rest of the clusters are located in the Mediterranean Sea. Many factors contribute to changes in sea surface height including sea eddies, temperature of the upper layer of seawater, tides, sea currents, and gravity.

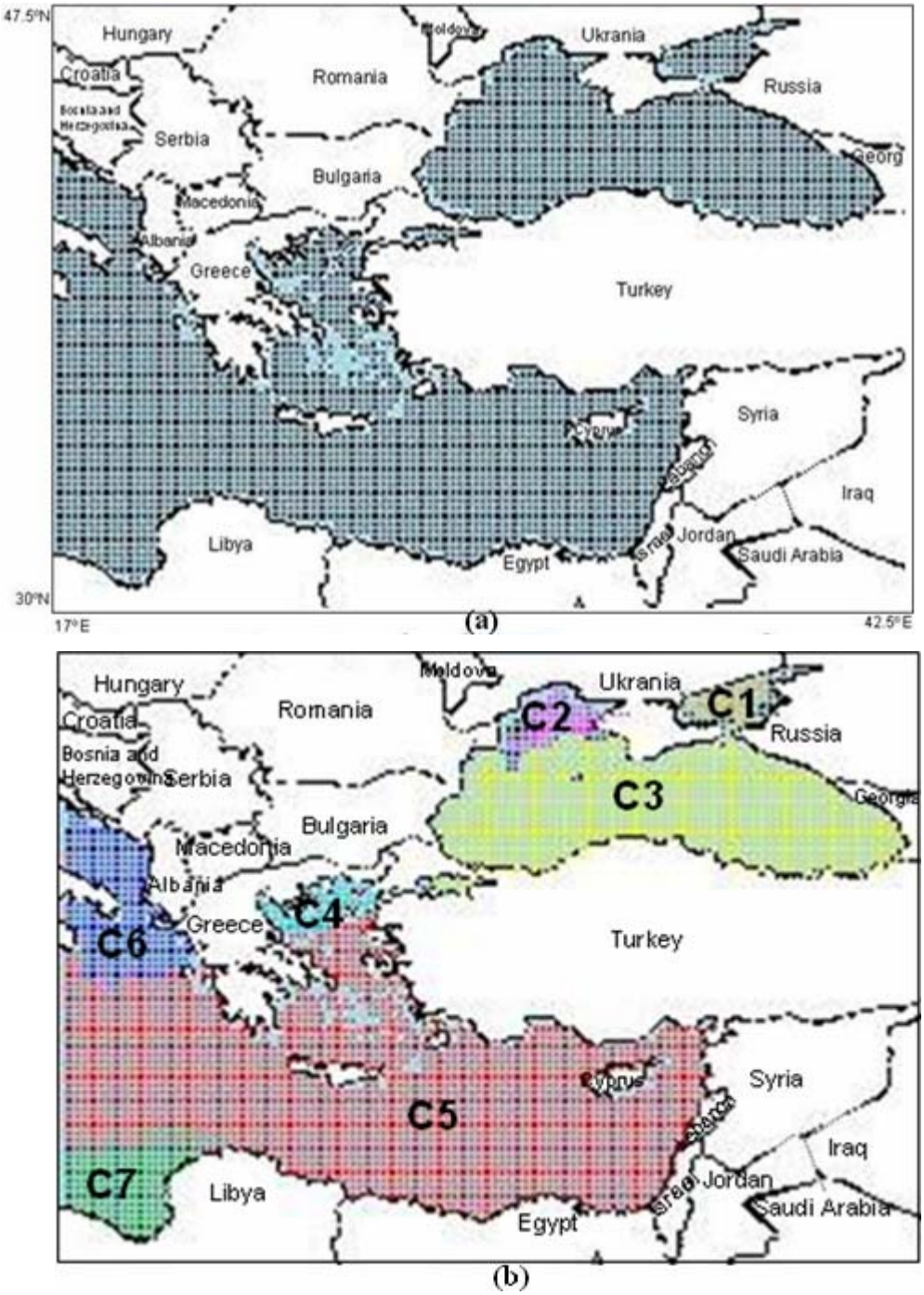


Figure 2.10 (a) The locations of 5340 stations (b) The results of cluster analysis on sea surface temperature data

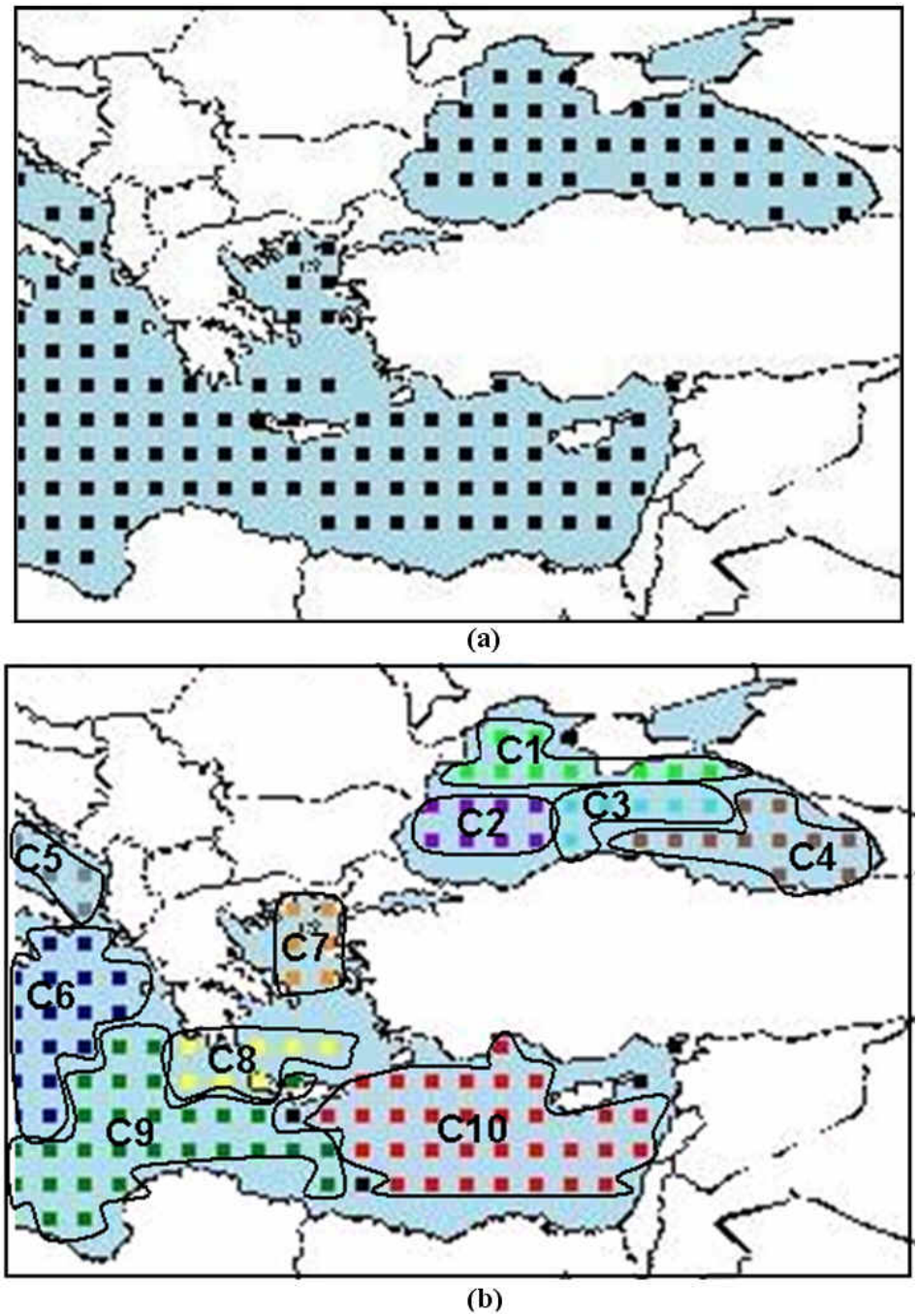


Figure 2.11 (a) The locations of 134 stations (b) The results of cluster analysis on sea surface height residual data

2.6 Distances in Cluster Analysis

In density-based clustering algorithms, a distance measure is used to determine whether a set of points is *similar* enough to be considered a cluster or not. The distance measures satisfy the following mathematic requirements:

1. Dist(i, j)>=0. Distance is a nonnegative number.
2. Dist(i, i) = 0. The distance of an object to itself is 0.
3. Dist(i, j) = Dist(j, i). Distance is symmetric.
4. Dist(i, j) ≤ Dist(i, h) + Dist(h, j). The triangle inequality.

The most common distance measures used are Manhattan distance, Euclidean distance, and Minkowski distance.

Manhattan distance is defined as:

$$Dist(i, j) = |x_{i1}-x_{j1}| + |x_{i2}-x_{j2}| + \cdots + |x_{in}-x_{jn}| \quad (2.9)$$

Euclidean distance is defined as:

$$Dist(i, j) = \sqrt{|x_{i1}-x_{j1}|^2 + |x_{i2}-x_{j2}|^2 + \cdots + |x_{in}-x_{jn}|^2} \quad (2.10)$$

where $i=(x_{i1}, x_{i2},..., x_{in})$ and $j=(x_{j1}, x_{j2},..., x_{jn})$ are two n-dimensional data objects.

Minkowski distance is a generalization of both Euclidean distance and Manhattan distance. It is defined as

$$Dist(i, j) = (|x_{i1} - x_{j1}|^p + |x_{i2} - x_{j2}|^p + \cdots + |x_{in} - x_{jn}|^p)^{1/p} \quad (2.11)$$

where *p* is a positive integer. It represents the Manhattan distance when p = 1 and the Euclidean distance when p = 2.

Figure 2.12.a shows two data objects A(1, 2) and B(5, 3) as an example. While the Manhattan distance between the two is 5, the Euclidean distance between the two is 4.12, and the Minkowski is equal to 4.02. However, they are equal to each other, when data objects are A(3.07, 2) and B(4.25, 2). (Figure 2.12.b)

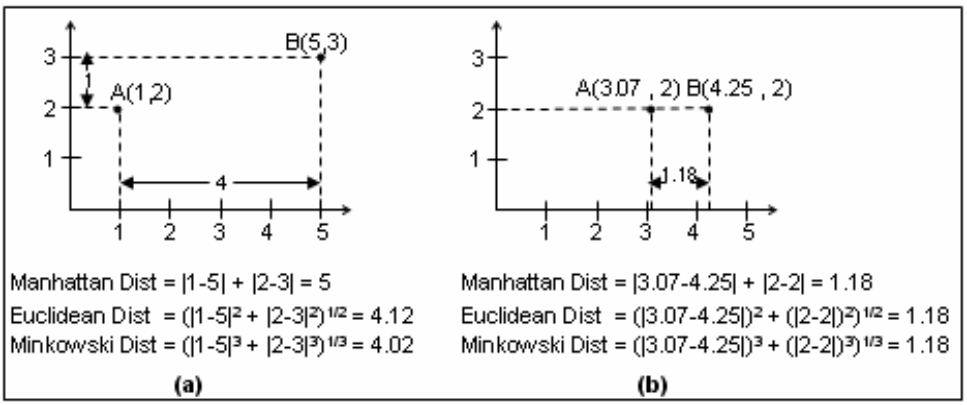


Figure 2.12 Manhattan, Euclidean and Minkowski (with order 3) Distances between two data objects

In this study, firstly, we analyzed sea surface temperature data to obtain the clusters by using Euclidean measure. The analysis results are shown in Figure 2.9. Then we also applied the same algorithm on the same data but in this case we used a different distance measure, Manhattan Distance and Minkowski Distance. By this way, we tried to answer the questions: “the results will be change when we use another distance measure or not”, “the results are sensitive to the distance formula or not”.

Table 2.3 summarizes the results. When the order of the formula increases, the number of noise objects decreases. When Euclidean distance was used to measure distance, 100 noise objects detected by the algorithm. However, the number of noise objects increased to 128 when Manhattan distance was used. There is no any change in the number of clusters. In addition, there is no so many changes in the shape of the clusters.

Table 2.3 Comparison of clustering results with different distance measures

Distance Measure	**Order**	**Formula**	**Num. of Noise Obj.**	**Num. of Clusters**
Manhattan Distance	1	$\|x_{i1}-x_{j1}\| + \ldots + \|x_{in}-x_{jn}\|$	128	7
Euclidean Distance	2	$(\|x_{i1}-x_{j1}\|^2 + \ldots + \|x_{in}-x_{jn}\|^2)^{1/2}$	100	7
Minkowski Distance	3	$(\|x_{i1}-x_{j1}\|^3 + \ldots + \|x_{in}-x_{jn}\|^3)^{1/3}$	91	7

CHAPTER THREE
OUTLIER DETECTION

3.1 Overview

Spatio-temporal databases are growing very rapidly, both in size and in number. This condition results in an increasing need for knowledge discovery in spatio-temporal databases. Most studies in KDD (Knowledge Discovery in Databases) focus on finding the common patterns. However, finding the outliers (rare events or exceptional cases) may be more interesting and useful than finding the common patterns.

Outliers can be defined as observations which appear to be inconsistent with the remainder of the dataset. They deviate too much from other observations. *Outlier detection* is a data mining technique like classification, clustering, and association rules.

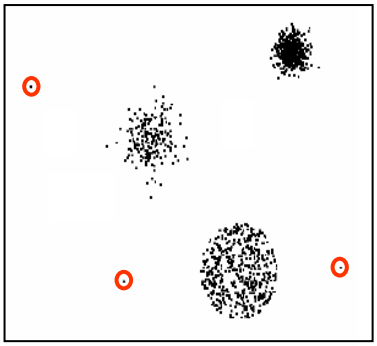


Figure 3.1 Example outliers

A *Spatial Outlier* (S-Outlier) is an object whose non-spatial attribute value is significantly different from the values of its spatial neighbors. Recently, a few studies have been conducted on *spatial outlier detection* for large datasets. (Breunig et al., 2000; Papadimitriou and Faloutsos, 2003; Shekhar et al., 2003) However, most of these studies don't consider temporal aspects. *Temporal outlier detection* should also be considered in many applications such as geographic phenomena based applications. A *Temporal Outlier* (T-Outlier) is an object whose non-spatial attribute

value is significantly different from those of other objects in its temporal neighborhood. It represents a time location which deviates too much from its temporal neighbors.

Our study combines S-Outlier and T-Outlier definitions to define a *Spatio-Temporal Outlier* (ST-Outlier) to be an object whose non-spatial attribute value is significantly different from those of other objects in its spatial and temporal neighborhood. For many applications, identification of ST-Outliers can lead to the discovery of unexpected, interesting, and implicit knowledge.

This study focuses on the question how ST-Outliers can be detected. It presents a new outlier detection algorithm which is based on the DBSCAN clustering algorithm since clustering is a basic method for spatial outlier detection. In contrast to the existing outlier detection algorithms, new algorithm has the ability of discovering outliers according to the non-spatial, spatial and temporal values of the objects. It proposes a three-step approach to detect spatio-temporal outliers in large databases. These steps are clustering, checking spatial neighbors, and checking temporal neighbors. From the viewpoint of a clustering algorithm, outliers are objects not located in any cluster. Furthermore, if a cluster is significantly different from other clusters, the objects in this cluster might be potential S-Outliers.

3.2 Outlier Detection Approaches

The existing approaches to outlier detection can be classified into five categories: *distribution-based*, *clustering-based*, *depth-based*, *distance-based*, and *density-based* (Kovács et al., 2004) (Papadimitriou and Faloutsos, 2003).

Distribution-based approaches use standard statistical distribution. They deploy some standard distribution model (e.g. Normal, Poisson, etc.) and recognize as outliers those points which deviate from the model. (Barnett & Lewis, 1994) However, for many KDD applications, the underlying distribution is unknown. Often a large number of tests are required in order to decide which distribution model fits

the arbitrary dataset best, if any. Fitting the data with standard distributions is costly, and may not produce satisfactory results.

Clustering-based approaches detect outliers as by-products (Jain et al., 1999). Some clustering algorithms such as CLARANS (Ng & Han, 1994), DBSCAN (Ester et al., 1996) (Ester et al., 1998a), CURE (Guha et al., 1998) have the capability of handling exceptions. However, since the main objective of the clustering algorithms is to discover clusters, they are not developed to optimize outlier detection.

Depth-based approaches are based on computational geometry and compute different layers of k-d convex hulls. (Johnson et al., 1998) (Ruts & Rousseeuw, 1996) Outliers are more likely to be data objects with smaller depths. However, in practice, this technique becomes inefficient for large datasets (k>=4). Depth-based approach is also applied for spatial outlier detection. (Adam et al., 2004)

Distance-based methods use a distance metric to measure the distances among the data points. (Knorr & Ng, 1998; Knorr et al., 2000) Problems may occur if the parameters of the data are very different from each other in different regions of the data set.

Density-based approach was proposed by M. Breunig, et al. (Breunig et al., 2000). This method assigns a Local Outlier Factor (LOF) to each sample based on their local neighborhood density. Samples with high LOF value are identified as outliers. The neighborhood is defined by using *MinPts* parameter.

One drawback of the existing methods is that they don't consider temporal aspects. In this study, we propose a ST-Outlier detection algorithm to overcome this disadvantage. Our algorithm combines the advantages of the *clustering-based* and *density-based* approaches.

3.3 ST-Outlier Detection Algorithm

In our algorithm, a three-step approach is proposed to identify the spatio-temporal outliers. As shown in Figure 3.1, these steps are: clustering, checking spatial neighbors, and checking temporal neighbors.

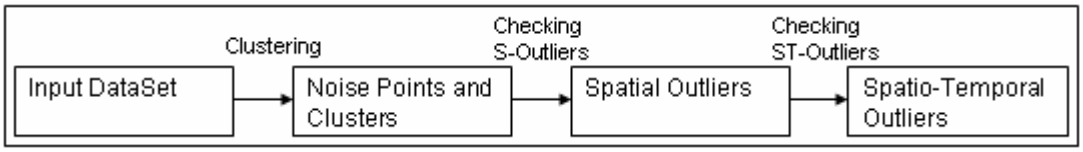


Figure 3.2 The steps of ST-Outlier detection algorithm

3.3.1 Clustering

Clustering is a basic method to detect potential S-Outliers. From the viewpoint of a clustering algorithm, potential outliers are objects not located in any cluster. Furthermore, if a cluster is significantly different from other clusters, the objects in this cluster might be potential S-Outliers.

A clustering algorithm should satisfy three important requirements: (i) discovery of clusters with arbitrary shape (ii) good efficiency on large databases and (iii) some heuristics to determine the input parameters. DBSCAN algorithm satisfies all these requirements. But it doesn't consider temporal aspects and it can't detect some outliers when clusters have different densities. In order to overcome these disadvantages, we developed ST-DBSCAN clustering algorithm (section 2.4). As the first step, this algorithm was applied to detect potential S-Outliers. Then checking_spatial_neighbors and checking_temporal_neighbors functions were implemented to check potential S-Outliers and potential ST-Outliers.

3.3.2 Checking Spatial Neighbors

From the viewpoint of a clustering algorithm, potential outliers are objects not located in any cluster. In the previous step, potential outliers were detected when the

data was clustering. In this step, these potential outliers are checked to verify whether these objects are actually S-outlier or not. During the verification, the background knowledge (the characteristic) of the data is required. If no prior-knowledge about the data available, some methods such as neural network can be applied to handle it. Furthermore, if a cluster is significantly different from other clusters, the objects in this cluster might be S-Outliers. Thus this step also checks all clusters identified in the previous step to decide whether the cluster is S-Outlier or not. The formula used to verify S-Outliers is defined in Definition 3.1.

Definition 3.1. *Given a database of n data objects D={o1, o2,…,on}. Assume that the object o is detected as potential outlier in clustering. The average value of the spatial neighbors of o within Eps1 radius is defined as*

$$A \overset{\text{def}}{=} \frac{o_{\text{neighbor1}} + o_{\text{neighbor2}} + \ldots + o_{\text{neighbor m}}}{m} \tag{3.1}$$

where m is the number of spatial neighbors of o within Eps1 radius and the standard deviation for the object o is defined as

$$\sigma = \sqrt{V} \tag{3.2}$$

where

$$V \overset{\text{def}}{=} \frac{(o_{\text{neigh1}} - A)^2 + (o_{\text{neigh2}} - A)^2 + \ldots + (o_{\text{neigh m}} - A)^2}{m}. \tag{3.3}$$

The object o is classified as an S-Outlier if it is outside the interval [L,U] (i.e., if either o < L or o > U), where

$$L \overset{\text{def}}{=} A - k_0.\sigma \tag{3.4}$$

$$U \overset{def}{=} A + k_0.\sigma$$

and $k_0 > 1$ is some pre-selected value.

3.3.3 Checking Temporal Neighbors

This step checks the temporal neighbors of the S-Outliers identified in the previous step. Two objects are temporal neighbors if the values of these objects are observed in consecutive time units such as consecutive days in the same year or in the same day in consecutive years. During the application of the algorithm, a tree is traversed to find the temporal neighbor objects of any object.

In order to support temporal aspects, S-Outliers are compared to other objects of the same local area but in different times. In comparison operation, spatio-temporal data is first filtered by retaining only the temporal neighbors and their corresponding values. If the characteristic value of a S-Outlier does not have significant differences with its temporal neighbors, this is not a ST-Outlier. Otherwise, it is confirmed as a ST-Outlier. The formula used to detect ST-Outliers is similar to formula defined in Definition 3.1. In this case temporal neighbors are checked instead of spatial neighbors.

3.4 Application

This section demonstrates how our algorithm detects ST-Outliers by using a real-world dataset. The purpose of the application is to detect rare events and exceptional cases related with sea waves in years between 1992 and 2002. The application tries to find unusual sea waves in Turkish seas.

3.4.1 Dataset Description

In the application, we used Wave-Height table of the data warehouse defined in section 2.5.1. Dataset has been provided from Topex/ Poseidon Satellite as a 6.2 km

spacing along-track grid. Topex/Poseidon data are released by NASA and CNES. Wave heights are measured in meters. Significant wave height from TOPEX is calculated from altimeter data based on the shape of a radar pulse after it bounces off the sea surface. A calm sea with low waves returns a sharply defined pulse whereas a rough sea with high waves returns a stretched pulse. The significant wave height is the average height of the highest one-third of all waves in a particular time period.

Dataset has approximately six million rows of record. It has the columns: StationID, RegionID, Year, Month, Day of the record, Latitude of the station, Longitude of the station, WaveHeight value, and ClusterID. The first column, StationID, identifies the geographic location of monitoring station. The column, RegionID, identifies the name of the sea (Black Sea, Marmara Sea, Aegean Sea, or Mediterranean Sea). The last column, ClusterID, identifies a particular cluster of stations.

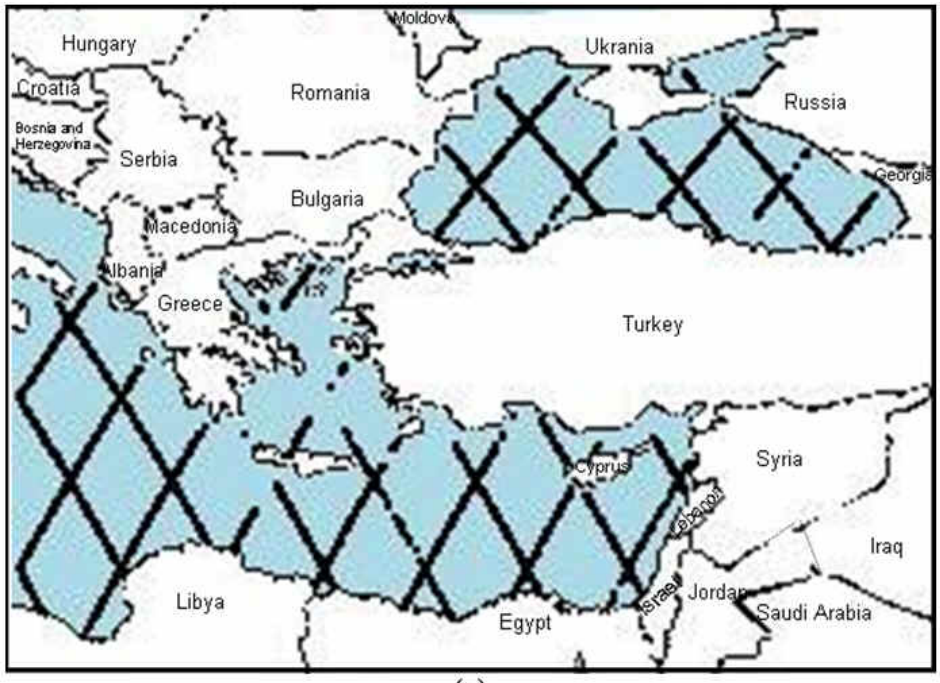
3.4.2 Implementation Details and Results

During the implementation, first, we clustered the dataset to find the regions that have similar sea wave height characteristics. The input parameters are designated as *Eps1=1*, *Eps2=0.25*, and *MinPts=15*. These values for the input parameters are determined by using the heuristics given in (Ester et al., 1996). Significant wave height values stored in database were collected at 1707 stations shown in Figure 3.3.a as black dots. Figure 3.3.b shows clustering result obtained using of the dataset measured between 1992 and 2002. Each cluster has data points that have similar significant wave height values.

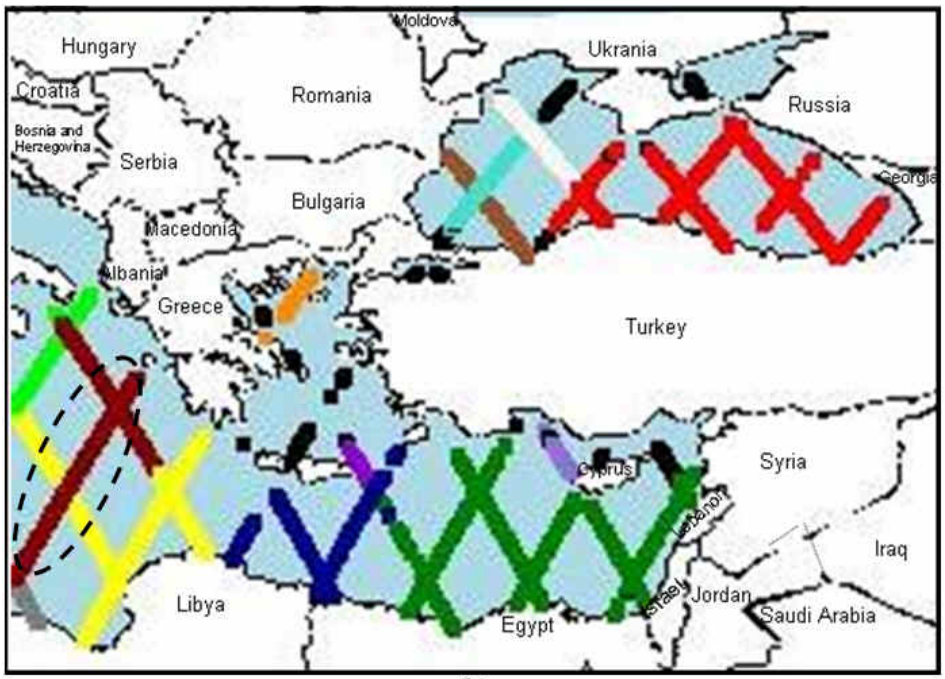
Second, we checked all noise objects and all clusters identified in the previous step to determine S-Outliers. The region circled in dashed lines in Figure 3.3 has significantly high wave height values on January 28, 2001. This region has high wave height values. So it contains S-Outliers.

Third, we checked the temporal neighbors. We compared the wave height values of S-Outliers with other data points of the same location but in different times. Figure 3.4 shows the wave height values of the same region in different years. We detected that the region circled in dashed lines have too extreme wave height values in 1998. This region has the wave height values approximately 6 meters. But, in other years, wave height values of this region are not too high. For this reason, the objects in this region are confirmed as ST-Outliers.

Many factors can affect wave heights such as wind speed, the water depth, the transfer and direction of energy, tsunami. For example, water depth can produce larger waves, especially when the depth of water is less than 1/2 the wavelength.



(a)



(b)

Figure 3.3 (a) The locations of 1707 stations (b) The results of cluster analysis on significant wave height values

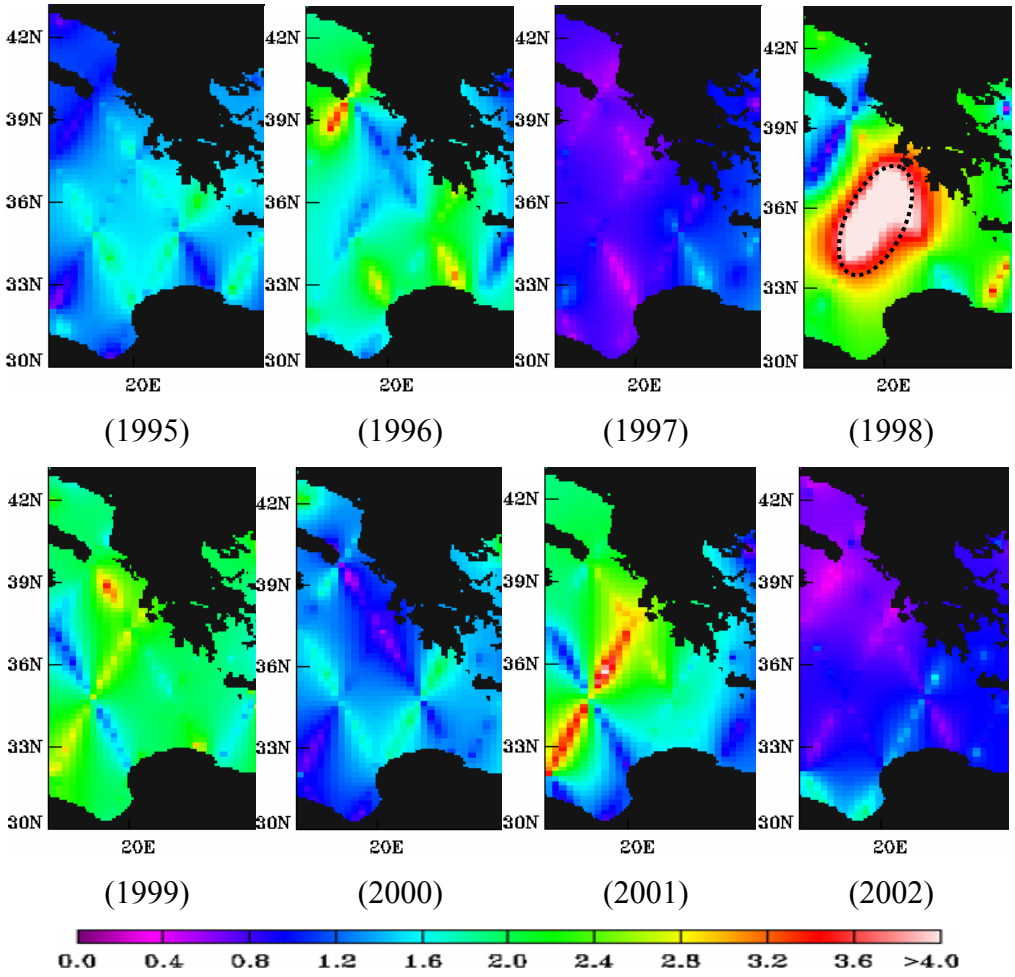


Figure 3.4 The wave height values of the same region in the same day, but in different years

CHAPTER FOUR
CLASSIFICATION

4.1 Overview

A decision tree is one of the most popular methods for discovering meaningful patterns and classification. This chapter presents the decision trees constructed for physical marine data by using SQL Server Analysis Services. The purpose of the study is to describe the structure of seawater characteristics and relationships between physical parameters such as seawater temperature, salinity, density, pressure etc. and to simplify these relationships in order to enable general statements about classes of parameters.

SQL Server Analysis Service has the capability to mine relational and multidimensional data warehousing, OLAP, or external OLE DB data. New models can be created by using *mining model* wizard. SQL Server 2000 Analysis Service includes two data mining algorithms namely *Decision Tree* and *Clustering*.

4.2 Decision Trees

Decision trees define groups by selecting breakpoints for the variables. In a decision tree, an internal node denotes a test on an attribute, branch represents an outcome of the test, and leaf nodes represent class labels or class distribution. A decision tree algorithm analyzes the data and creates a repeating series of branches until no relevant branches exist. After the construction of a decision tree, it is possible to classify an unknown sample by following the tree.

4.3 Explanation about Dataset

The physical parameters obtained from the Institute of Marine Sciences and Technology (IMST) at Dokuz Eylul University is used for the decision tree implementation. These physical parameters are:

- Water Temperature (^{o}C)
- Salinity (psu)
- Conductivity (mS / cm)
- Density (Kg / m^3)
- Pressure (db - decibars)
- Sound Velocity (m/s)
- Potential Temperature (^{o}C)
- Light Transmission (%)
- Ph
- Dissolved Oxygen (mg / lt)
- Dissolved Oxygen Concentration (mg / lt)
- Beam Attenuation Coefficient

These parameters are measured by SBE 911 CTD (Conductivity Temperature and Depth) System in K.Piri Reis research vessel. The dataset contains physical parameters of 40 stations in Izmir Gulf in 2001. Data is available not only for the sea surface but also for each meter.

4.4 Creating Mining Models

In SQL Server Analysis Manager, a new model can be created by using Mining Model wizard. When creating a new model, it is necessary to select a data source, a case table or tables, a data mining technique (algorithm), a case key column, the input and prediction columns. It joins the tables if multiple tables are chosen. Figure 4.1 shows a screen shot of analysis manager which includes six mining models created for conductivity, density, dissolved oxygen concentration, light transmission, ph, and sound velocity parameters of a marine environment.

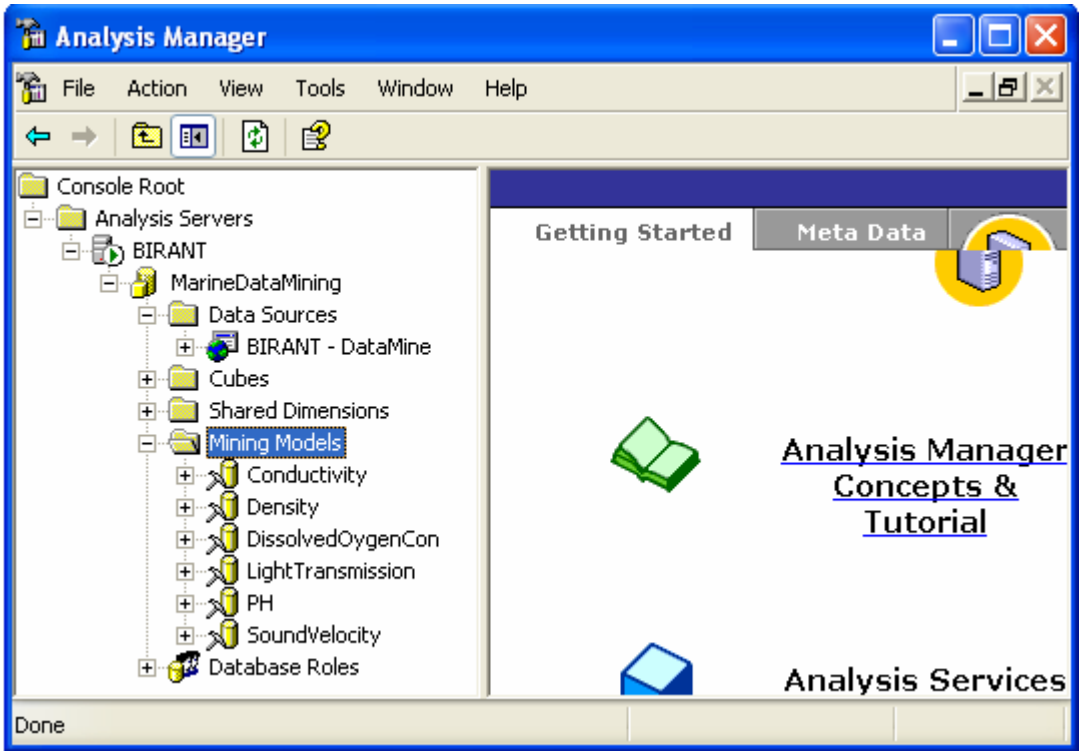


Figure 4.1 Mining models created by using SQL Server Analysis Manager

4.4.1 The Concept of Density

Density response to changes in temperature and salinity. A decrease in temperature results in an increase of density, a decrease in salinity, on the other hand, produces a density decrease. (Emery et al., 2005) As shown in Figure 4.2, in the ocean, the effect of the temperature decrease is stronger than the effect of the salinity decrease, so the density increases with depth.

- *Surface Ocean* (Mixed Layer) (0-200m) - well mixed and homogeneous, low density
- *Pycnocline* (200-1000m) - temperature, salinity, density change rapidly
- *Deep Ocean* (>1000m) - cold (<5°C) and dense

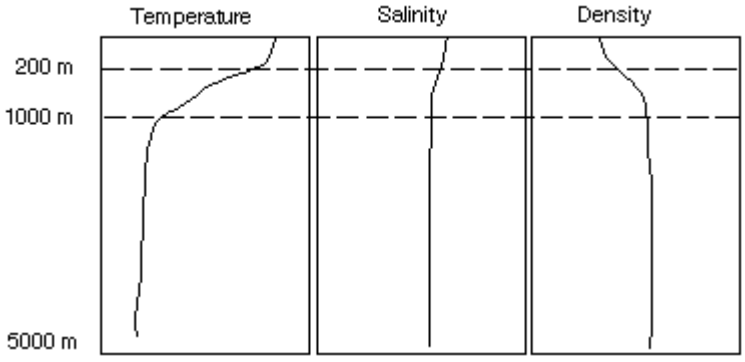


Figure 4.2 Changes of temperature, salinity and density with depth (Lecture Notes - Earth Sciences 1, 2003)

The colors in the decision trees give you a feel for the density of the cases in the nodes. The darker color, the higher percentage of cases corresponding to that value in the attribute is in that node. When you click on the "*All*" node, as shown in Figure 4.3, the *Attributes* pane tells you that 734 of 1388 cases (%52,88) fall into 28,8-28,9 category, 521 cases (%37,54) fall into 28,7-28,8 category, and 133 cases (%9,58) fall into 28,6-28,7 category.

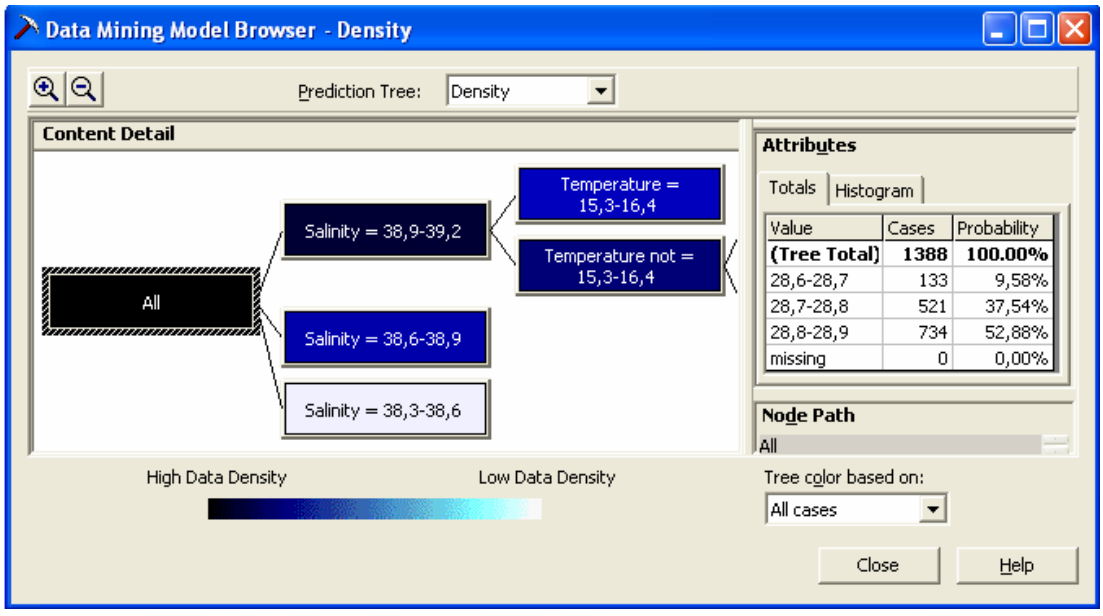


Figure 4.3 Decision tree for density analysis

The results presented in Figure 4.4 show that a decrease in salinity produces a density decrease. The results presented in Figure 4.5 show that a decrease in temperature results in an increase of density.

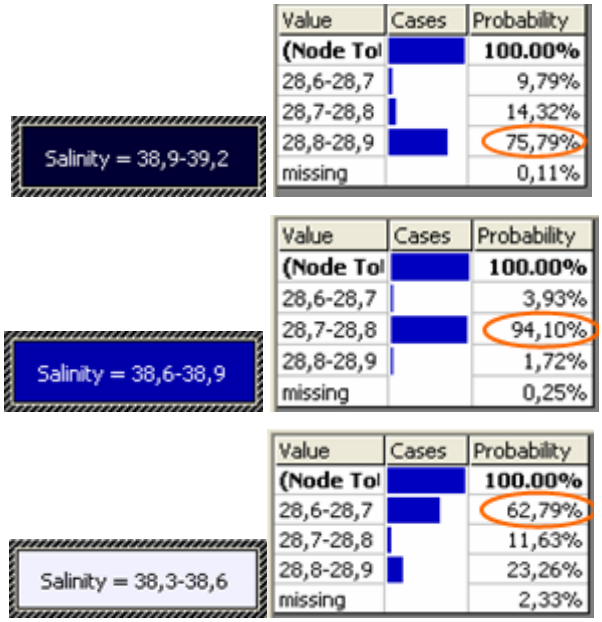


Figure 4.4 The density characteristics with different salinity values

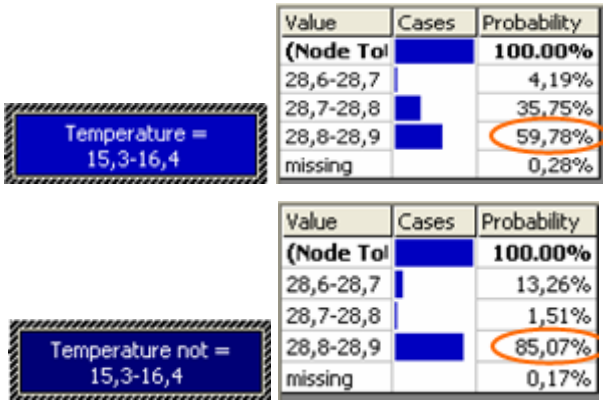


Figure 4.5 The density characteristics with different temperature values

4.4.2 The Concept of Conductivity

The conductivity of sea water depends on the number of dissolved ions per volume (i.e. salinity) and the mobility of the ions (i.e. temperature and pressure). Its units are mS/cm (milli-Siemens per centimeter). Conductivity increases with a salinity increase, a temperature increase, and a depth (i.e. pressure) increase. In most practical oceanographic applications the change of conductivity is dominated by temperature. (Emery et al., 2005) The data mining results related with conductivity

are showed in Figure 4.6 and Figure 4.7. The results support this relationship between temperature, salinity and conductivity.

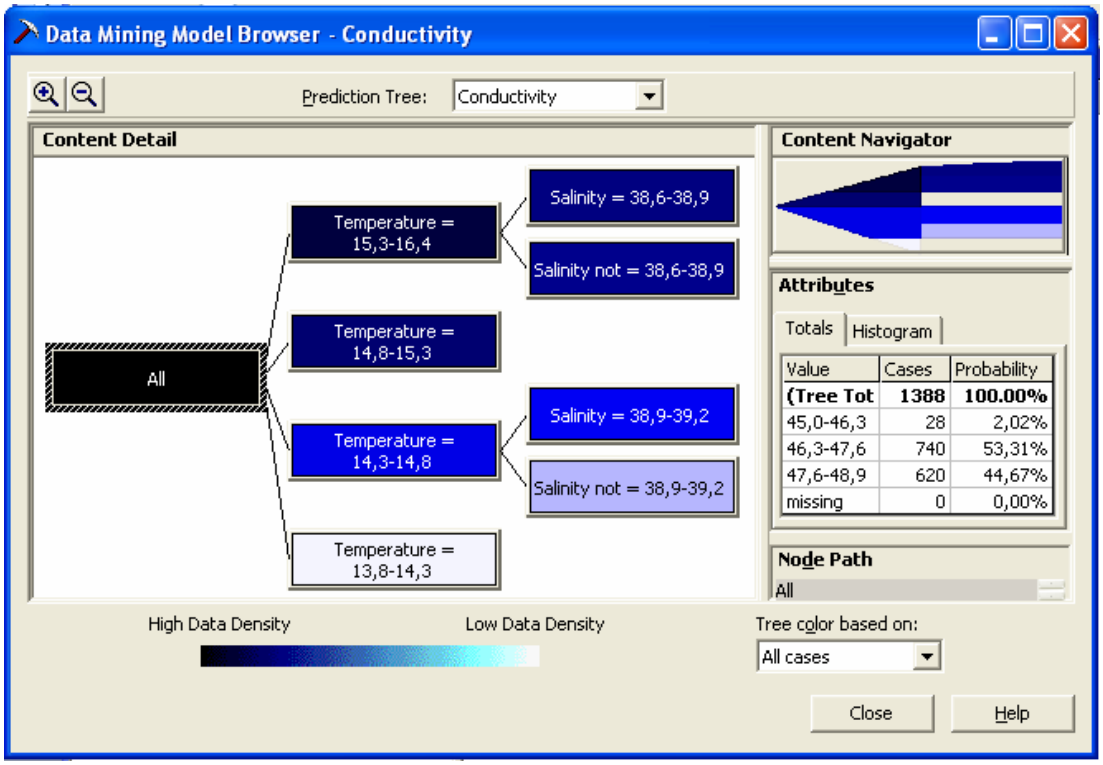


Figure 4.6 Decision tree for conductivity analysis

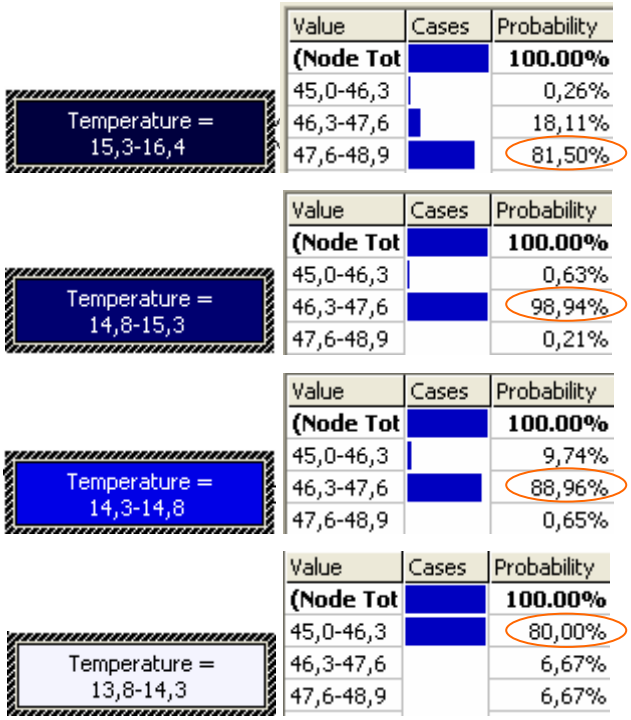


Figure 4.7 The conductivity characteristics with different temperature values

4.4.3 The Concept of Sound Velocity

The sound speed is a function of temperature T, salinity S, and pressure p and varies between 1400 m s^{-1} and 1600 m s^{-1}. In the open ocean, it is influenced by the distribution of T and p but not much by S. It decreases with decreasing T, p and S. The combination of the variation of these three parameters with depth produces a vertical sound speed profile with a marked sound speed minimum at intermediate depth. Temperature decreases rapidly in the upper kilometer of the ocean and dominates the sound speed profile. In the deeper regions (below the top kilometer or so), the temperature change with depth is small and sound velocity is determined by the pressure increase with depth. Vertical changes of salinity are too small to have an impact; but the average salinity determines whether sound velocity is low (if the average salinity is low) or high (if the average salinity is high) on average. (Emery et al., 2005) Figure 4.8 shows the data mining result for sound velocity. This result supports this relationship between temperature, salinity, conductivity and sound velocity.

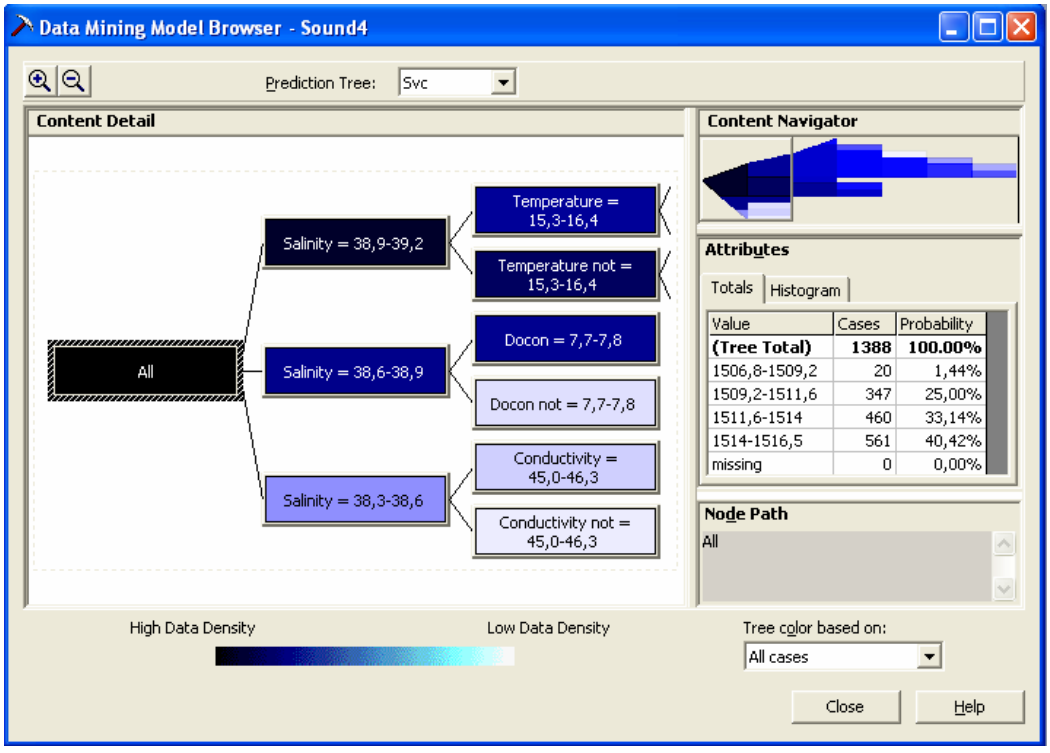


Figure 4.8 Decision tree for sound velocity analysis

4.4.4 The Concept of Dissolved Oxygen Concentration

The physical process that affects dissolved oxygen concentrations is the relationship between water temperature and gas saturation. Cold water can hold more of any gas, in this case oxygen, than warmer water. Warmer water becomes "saturated" more easily with oxygen. As water becomes warmer it can hold less and less dissolved oxygen. (Michaud, 1991) Figure 4.9 shows the data mining result for dissolved oxygen concentration.

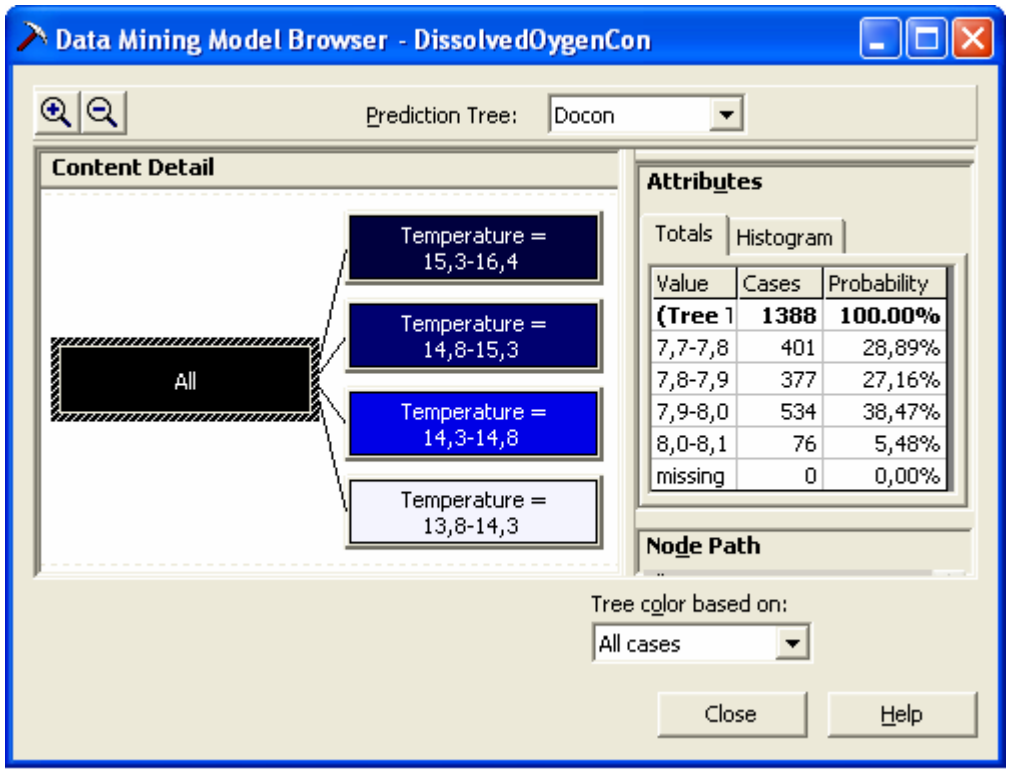


Figure 4.9 Decision tree for dissolved oxygen concentration analysis

4.4.5 The Concept of PH

The pH of a sample of water is a measure of the concentration of hydrogen ions. The pH scale ranges from 0 to 14. A pH of 7 is considered to be neutral. Substances with pH of less that 7 are acidic; substances with pH greater than 7 are basic. PH levels decrease due to the increase of water depth. As the water depth increases, less Oxygen is produced, because of the lesser amounts of algae in the water. More

Carbon Dioxide is present, because of microbial digestion, thus, creating an acidic environment. (Freshwater et al., 2002) Figure 4.10 shows the data mining result for PH profile.

.

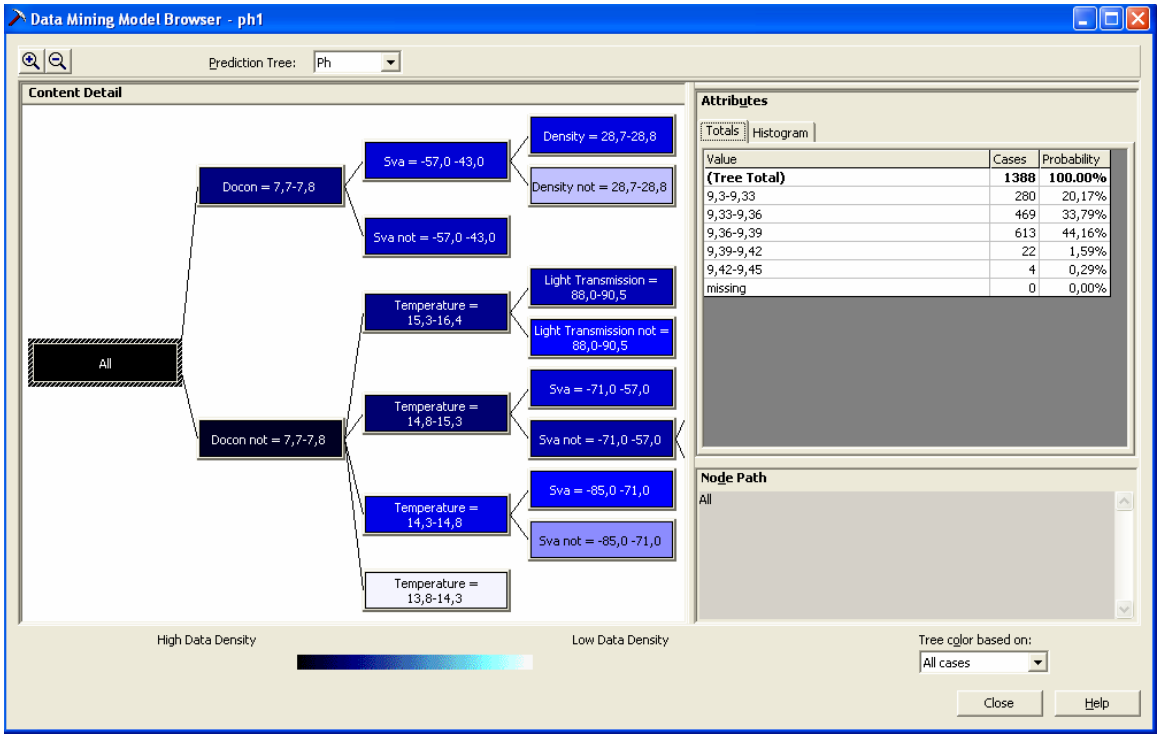


Figure 4.10 Decision tree for PH analysis

4.4.6 The Concept of Light Transmission

The absorption of solar heat energy is only effective in the topmost layers of the oceans. With depth, the longer wavelengths of visible light (red, orange, yellow) are absorbed. At depths greater than about 100 meters, only short wavelengths of visible light are transmitted (green, blue, violet). (Lecture Notes - Geology 117: The Oceans, 2003) Figure 4.11 shows that the value of light transmission is between 88,0 and 90,5 with the probability of 98,93, if Beam Attenuation Coefficient is between 1,0 and 1,4, and Sound Velocity is between 1511,6 and 1514.

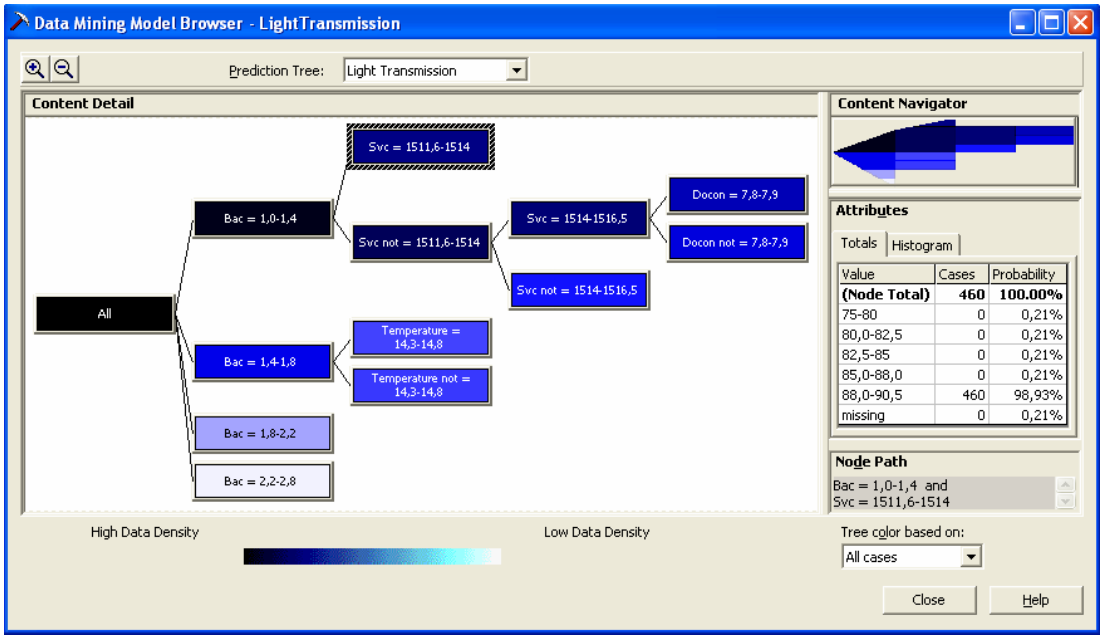


Figure 4.11 Decision tree for light transmission analysis

Figure 4.12 shows a screen shoot of the *Dependency Network Browser* for the transmission light property. *Dependency Network Browser* is used to view the dependencies and relationships among objects in a data-mining model. Dependency is indicated by arrows. The direction of predictability is indicated by arrowheads and by the color-coding of the nodes. As you move the slider to the left of the window down, *Dependency Network Browser* removes the weakest dependencies one at a time. Figure 4.12 reveals that the strongest predictor of light transmission is *Beam Attenuation Coefficient* because the last arrow on the screen is for *Beam Attenuation Coefficient* attribute. This result is not unexpected result. Because another way of reporting transmissivity measurement is the beam attenuation coefficient, which is related to perent transmission as: percent transmission = (100%) exp(-cz), where c is the beam attenuation coefficient in units of meter -1, and z is the length of the light path in meters. (Magoon, 1995)

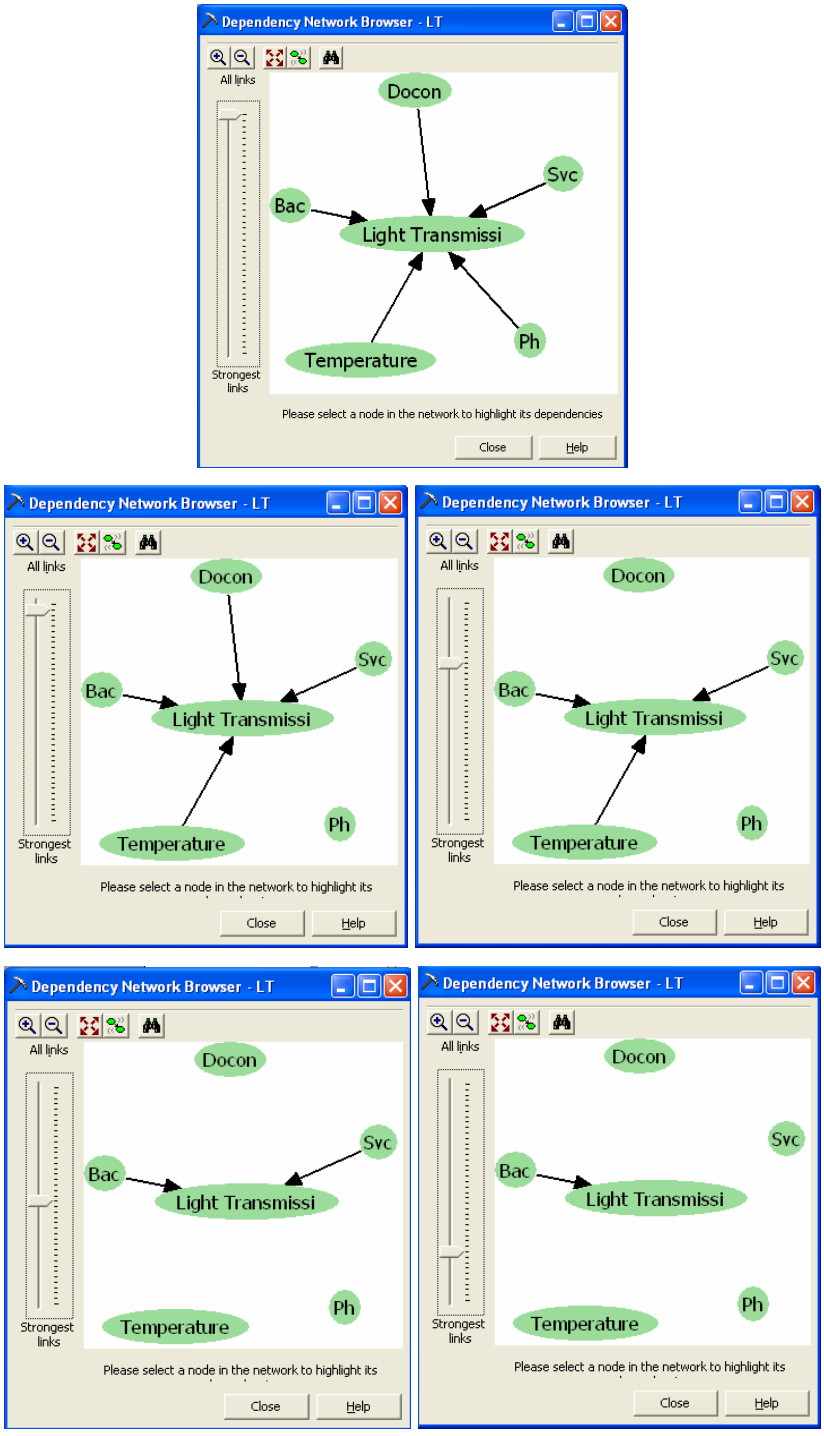


Figure 4.12 Dependency Network Browser for light transmission analysis

CHAPTER FIVE
SENSITIVITY ANALYSIS

5.1 Overview

Sensitivity analysis is used to determine how a given model output depends upon the input parameters. It is an important method for checking the quality of a given model, as well as a powerful tool for checking the robustness and reliability of its analysis. This chapter presents the sensitivity of the parameters of new clustering algorithm, ST-DBSCAN. The results presented in this chapter show the sensitivity degree of the algorithm to parameter settings.

5.2 Sensitivity Analysis of the Parameters

A sensitivity analysis is the process of varying model input parameters over a reasonable range and observing the relative change in model response. The purpose of the sensitivity analysis is to demonstrate the sensitivity of the model simulations to uncertainty in values of model input data. The sensitivity analysis helps to identify parameters that strongly affect model output. (Saltelli et al., 2000)

Most sensitivity analyses involve changing one parameter at a time. A sensitivity analysis is conducted by changing each parameter value by +/- 10% and +/-50% (Toxopeus, A.G., 1997). The sensitivity analysis indicates a *robust* model when small changes in input parameters result in small changes in model output (Swartzman and Kaluzny, 1987). If a small change in a parameter results in relatively large changes in the outcomes, the outcomes are said to be sensitive to that parameter. This may mean that the parameter has to be determined very accurately or that the alternative has to be redesigned for low sensitivity.

5.3 The Parameters in ST-DBSCAN Algorithm

In order to determine whether a set of points is *similar* enough to be considered a cluster or not, we need a distance parameter dist(i, j) that tells how far points i and j are. This distance parameter is used by a distance formula such as Manhattan, Euclidean, or Minkowski distance formula. In this study, Euclidean distance defined in Equation 2.1 was used.

DBSCAN algorithm accepts a radius value *Eps* based on a user defined distance measure and a value *MinPts* for the number of minimal points that should occur within *Eps* radius. It uses only one distance parameter *Eps* to measure similarity of data with one dimension. In order to support two dimensional data, we propose two distance metrics, *Eps1* and *Eps2*, to define the *similarity* by a conjunction of two density tests.

Our algorithm ST-DBSCAN requires three basic parameters: *Eps1*, *Eps2*, and *MinPts*. *Eps1* is the distance parameter for spatial attributes and it is used for spatial values to measure the closeness of two points geographically. *Eps2* is the distance parameter for non-spatial attributes and it is used to measure the similarity of non-spatial values. *Eps1* and *Eps2* parameters are calculated by the formulas in Equation 2.2. The last basic parameter, *MinPts*, is the minimum number of points within *Eps1* and *Eps2* distance of a point. If a region is dense, then it should contain more points than *MinPts* value.

In (Ester et al., 1996), a simple heuristic is presented which is effective in many cases to determine the parameters *Eps* and *MinPts*. The heuristic suggests $MinPts \approx ln(n)$ where n is the size of the database and *Eps* must be picked depending on the value of *MinPts*. The first step of the heuristic method is to determine the distances to the *k-nearest neighbors* for each object, where *k* is equal to *MinPts*. Then these *k-distance values* should be sorted in descending order. Then we should determine the threshold point which is the first “valley” of the sorted graph. We should select *Eps* to less than the distance defined by the first valley.

5.4 Sensitivity Analysis of the Parameters in ST-DBSCAN Algorithm

Most sensitivity analyses involve changing one parameter at a time. A sensitivity analysis was conducted by changing each parameter value by +/- 10% and +/-50%. We used Sea_Surface_Temperature table as an example table which is describing in section 2.5.1. According to the heuristic defined in section 5.2, the input parameters should be assigned as *Eps1=3*, *Eps2=0.5*, and *MinPts=15*. Table 5.1, Table 5.2 and Table 5.3 show the results of sensitivity analysis of *Eps1*, *Eps2,* and *MinPts* parameters respectively. Figure 5.1, Figure 5.3, and Figure 5.5 show the sensitivity results for *Eps1*, *Eps2*, and *MinPts* parameters as bar charts. Figure 5.2, Figure 5.4, and Figure 5.6 present the sensitivity of the algorithm to the *Eps1*, *Eps2* and *MinPts* parameters with respect to number of noise objects and number of clusters, while Figure 5.7 presents all of them.

According to the results of the tests, even there is a huge variation of *Eps1* parameter; there is a little influence on the results. The number of noise objects continuously increases when the value of *Eps1* parameter decreases. But there is no any change in the number of noise objects when the value of *Eps1* parameter increases. Furthermore, there is nearly no change in the number of clusters. So Eps1 parameter has little influence on the algorithm output. The results show that *Eps2* parameter is the most sensitive parameter. Decreasing in the value of the *Eps2* parameter greatly increases the number of noise objects and the number of clusters. The output of the algorithm is very much depending on the reliability of this parameter. The results show that the output of the algorithm is little sensitive to changes in the *MinPts* parameter values. The output is lowly sensitive to big changes in the *MinPts* parameter values.

Table 5.1 Sensitivity analysis results for *Eps1* parameter

Parameter Change	Eps1	Eps2	MinPts	Num. of Clusters	Num. of Noise Obj.	C1	C2	C3	C4	C5	C6	C7
0 %	3.0	0.5	15	7	100	63	92	1306	90	3061	349	279
- 10 %	2.7	0.5	15	7	101	63	92	1306	90	3060	349	279
- 20 %	2.4	0.5	15	7	103	63	92	1305	90	3060	348	279
- 30 %	2.1	0.5	15	7	108	63	92	1305	83	3058	348	283
- 40 %	1.8	0.5	15	7	114	63	92	1303	83	3055	347	283
- 50 %	1.5	0.5	15	7	134	63	92	1294	81	3045	345	286
+ 10 %	3.3	0.5	15	7	95	63	92	1311	90	3061	349	279
+ 20 %	3.6	0.5	15	7	95	63	92	1311	90	3061	349	279
+ 30 %	3.9	0.5	15	7	95	63	92	1311	90	3077	349	263
+ 40 %	4.2	0.5	15	7	95	63	92	1311	183	3115	293	188
+ 50 %	4.5	0.5	15	6	95	63	91	1312	234	3270	275	0

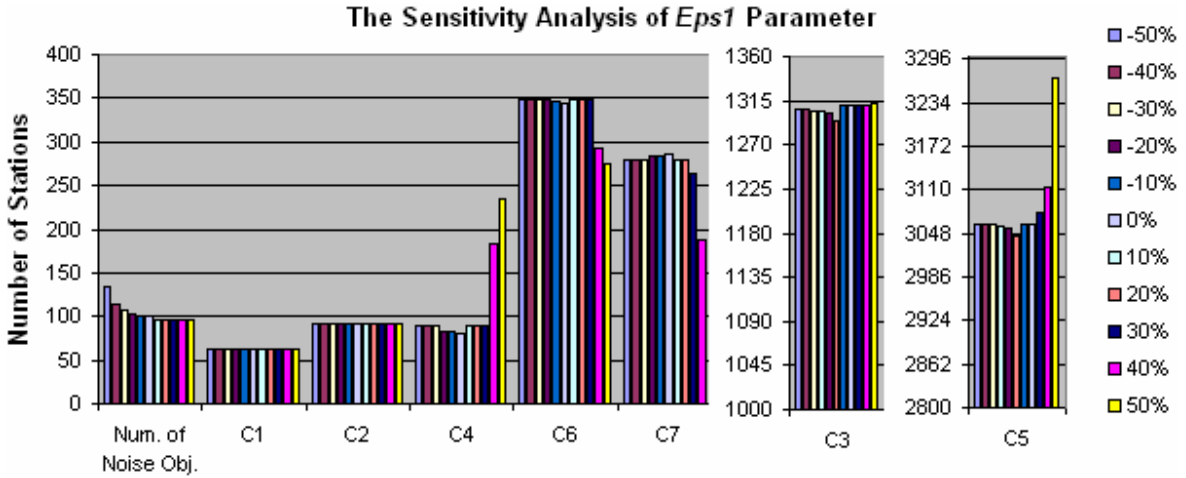


Figure 5.1 The sensitivity test results for Eps1 parameter

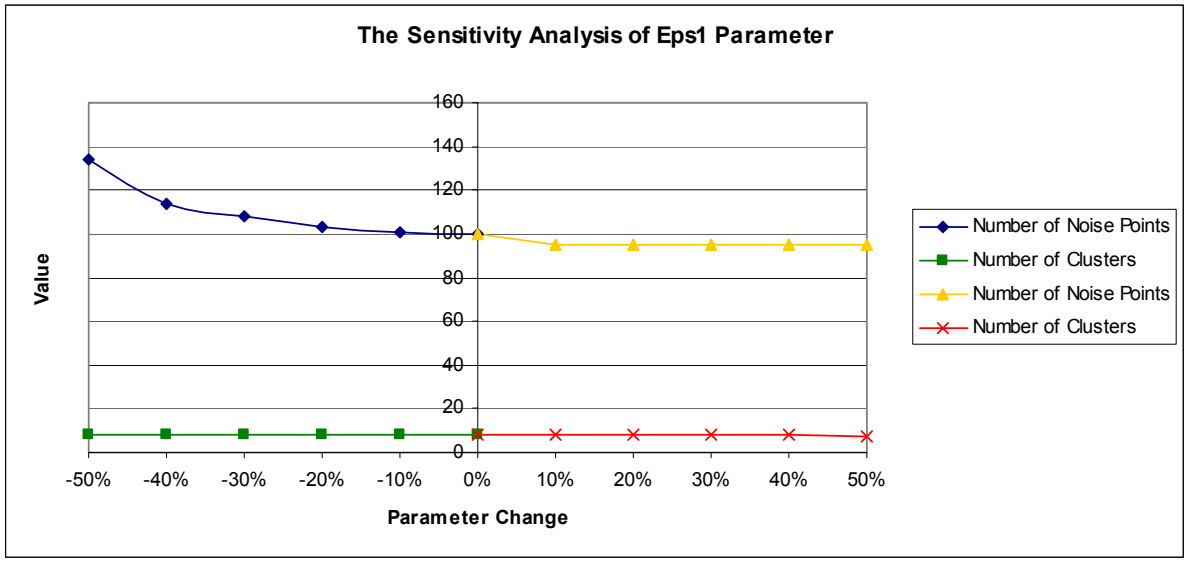


Figure 5.2 The sensitivity of *Eps1* parameter with respect to number of noise objects and number of clusters

Table 5.2 Sensitivity analysis results for *Esp2* parameter

Parameter Change	Eps1	Eps2	MinPts	Num. of Clusters	Num. of Noise Obj.	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13
0 %	3	0.5	15	7	100	63	92	1306	90	3061	349	279	0	0	0	0	0	0
- 10 %	3	0.45	15	8	128	59	89	1290	75	3052	327	304	16	0	0	0	0	0
- 20 %	3	0.4	15	9	177	58	80	1248	71	3042	316	310	23	15	0	0	0	0
- 30 %	3	0.35	15	9	182	58	80	1240	71	3037	316	315	26	15	0	0	0	0
- 40 %	3	0.3	15	11	377	39	75	1048	64	2967	277	345	26	40	26	56	0	0
- 50 %	3	0.25	15	13	424	21	39	1041	62	2832	233	425	52	45	26	60	32	48
+ 10 %	3	0.55	15	6	82	64	91	1309	0	3099	365	330	0	0	0	0	0	0
+ 20 %	3	0.6	15	6	79	64	90	1313	0	3099	365	330	0	0	0	0	0	0
+ 30 %	3	0.65	15	5	72	67	0	1349	0	3122	370	360	0	0	0	0	0	0
+ 40 %	3	0.7	15	5	69	68	0	1350	0	3122	370	361	0	0	0	0	0	0
+ 50 %	3	0.75	15	5	65	69	0	1350	0	3122	370	364	0	0	0	0	0	0

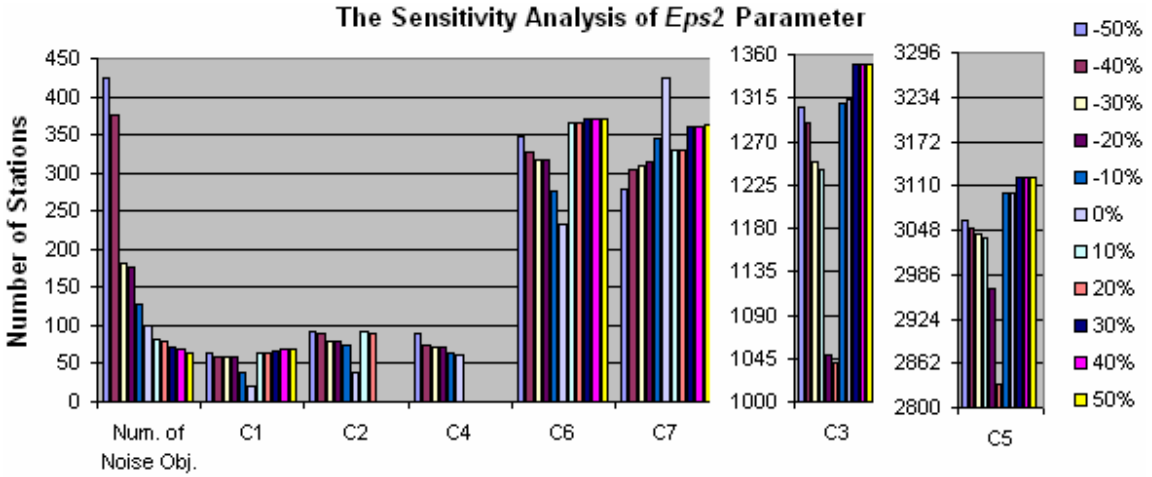


Figure 5.3 The sensitivity test results for *Eps2* parameter

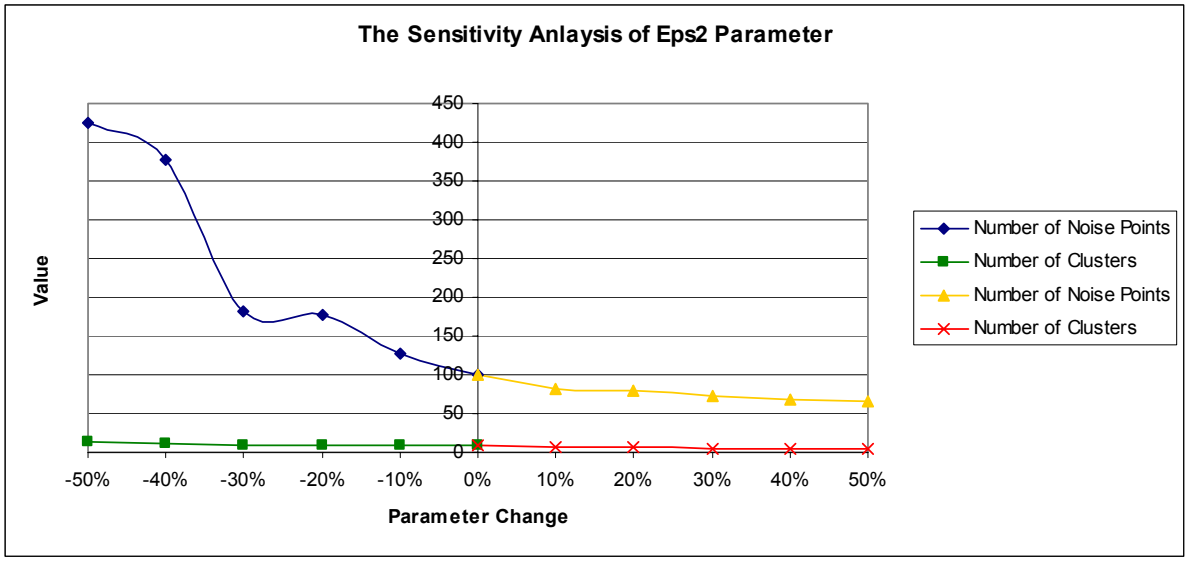


Figure 5.4 The sensitivity of *Eps2* parameter with respect to number of noise objects and number of clusters

Table 5.3 Sensitivity analysis results for *MinPts* parameter

Parameter Change	Eps1	Eps2	MinPts	Num. of Clusters	Num. of Noise Obj.	C1	C2	C3	C4	C5	C6	C7
0 %	3	0.5	15	7	100	63	92	1306	90	3061	349	279
- 10 %	3	0.5	13.5	7	100	63	92	1306	90	3061	349	279
- 20 %	3	0.5	12	7	93	63	97	1307	90	3062	349	279
- 30 %	3	0.5	10.5	7	71	65	103	1319	90	3063	351	278
- 40 %	3	0.5	9	7	70	63	104	1321	90	3063	351	278
- 50 %	3	0.5	7.5	7	68	65	104	1321	90	3063	351	278
+ 10 %	3	0.5	16.5	7	105	63	92	1306	85	3061	349	279
+ 20 %	3	0.5	18	7	108	63	91	1305	85	3060	349	279
+ 30 %	3	0.5	19.5	6	128	66	90	1305	0	3059	348	344
+ 40 %	3	0.5	21	6	136	66	82	1305	0	3059	348	344
+ 50 %	3	0.5	22.5	6	148	67	74	1301	0	3057	348	345

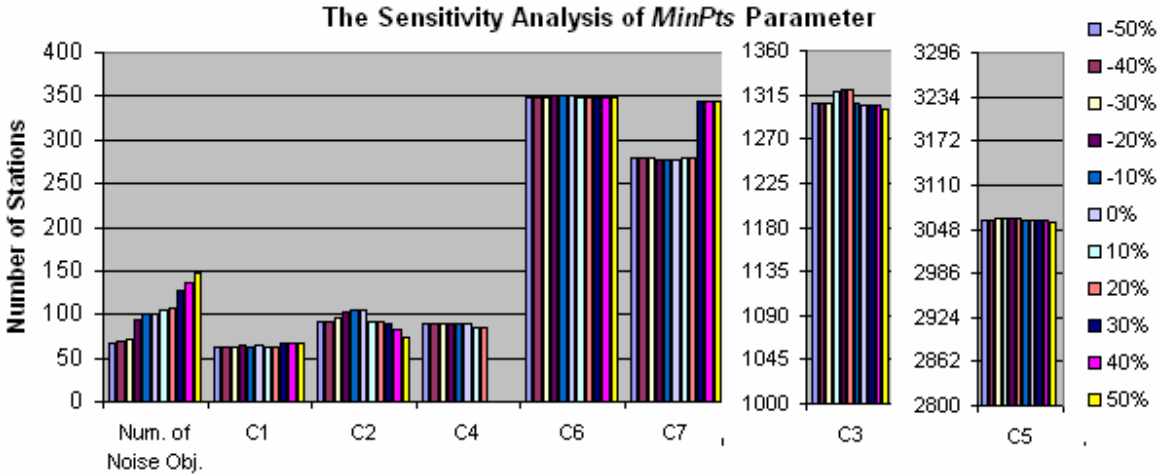


Figure 5.5 The sensitivity test results for *MinPts* parameter

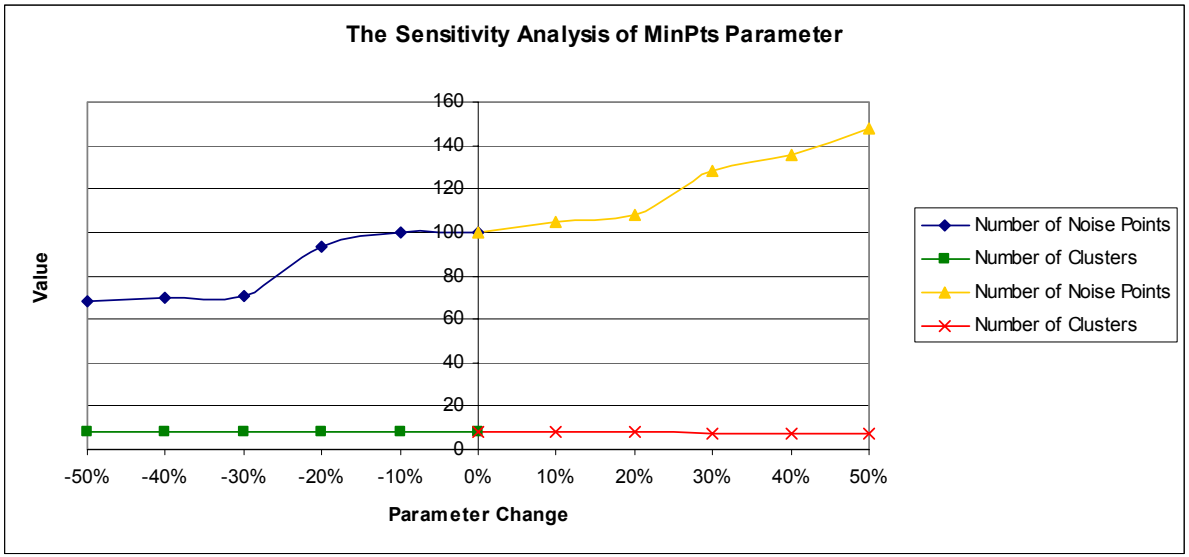


Figure 5.6 The sensitivity of *MinPts* parameter with respect to number of noise objects and number of clusters

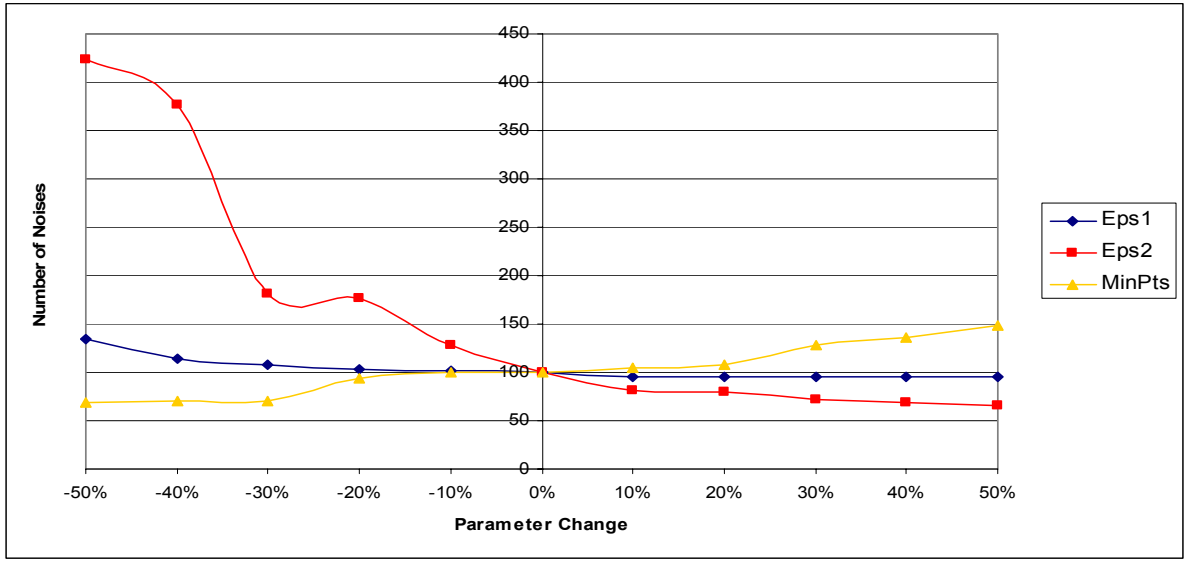


Figure 5.7 The sensitivity test results for *Eps1*, *Eps2*, and *MinPts* parameters

CHAPTER SIX
WEB SERVICE-BASED PARALLEL CLUSTERING

6.1 Overview

Clustering task in large databases generally requires too long processing time when executing in sequentially. For this reason, we need parallel processing solutions to decrease processing time in practice. In this study, we present a new approach to decrease both processing time and response time: Web Service-Based Parallel Clustering. This approach includes the parallel execution of web services for discovering clusters in large databases. Each web service is responsible for clustering one portion of data, so the work load is distributed over several web services. The purpose of the study is to allow users to satisfy their clustering needs without the knowledge of any parallel programming interface such as MPI (Message Passing Interface) or PVM (Parallel Virtual Machine). Instead of using any parallel programming interface, we propose the usage of web services which is useful to simplify the steps of developing parallel clustering programs.

A *web service* is an accessible application that can be automatically discovered and invoked in a distributed web environment. There are various technologies behind this achievement: WSDL (Web Service Description Language), UDDI (Universal Description, Discovery and Integration), and SOAP (Simple Object Access Protocol). These technologies are standards to describe, publish, discover, and use web services (Bosworth, 2001)(Roy and Ramanujan, 2001). However, they do not provide a support for the composition of multiple web services. Requests to multiple web services are issued separately, but a set of requests cannot be grouped into a single process flow (Mikalsen et al., 2001). For this reason, an additional technology is required for the execution of multiple web services.

Traditionally, multiple web services are invoked and run in sequential order, although they are located in different machines. The concurrent execution of N web

services is possible and it calls for specific mechanisms such as multithreading. A process can create one or more threads to execute a portion of the program code associated with the process. To establish concurrent execution, the master application has to create a thread for each web service.

6.2 Advantages of Web Service-Based Parallel Clustering

By the way of executing web services in parallel, the work load is distributed over several web services running on different machines. Each web service executes the same clustering algorithm. Each web service is responsible for processing one portion of data. Thus, the processing time and response time are both decreased by doing the necessary steps in parallel. There are many other reasons as to why we propose Web Service-Based Parallel Clustering:

- Writing a parallel program requires some special knowledge and experience about a parallel programming interface such as MPI or PVM. But the concepts behind web services are very easy to understand. Therefore the need for parallel clustering can be provided without the knowledge of any parallel programming interface syntax. Web services can be useful to simplify the steps of developing parallel programs.

- Web services are generally located on different machines. Therefore, more than one web services can be easily executed at the same time. Web services should not have temporal dependency on each other. The capability of parallel execution can be provided by using multi-thread technology.

- Web services are language-independent. It is possible for web services to be written in any programming language. Web services can be supported on different types of clients, such as Java, Cell-phones, Microsoft .NET, and Advanced Artificial Intelligence Environments.

- Web services are also platform-independent interfaces that allow communication with other applications using standard technologies and standard protocols, such as HTTP, SOAP and XML. By using standard technologies, web services allow programmers to develop flexible integration solutions that can evolve as needs change.

- The system becomes easily scalable with just adding new nodes with new web services.

6.3 General Structure of the Model

In the model, a master application invokes a set of web services that process the data concurrently and return partial results to the master application after they have finished processing. Data can be imported from different sources such as databases, data warehouse or data marts. The clustering results are accessible via web interfaces. Figure 6.1 illustrates the way of web service-based parallel clustering.

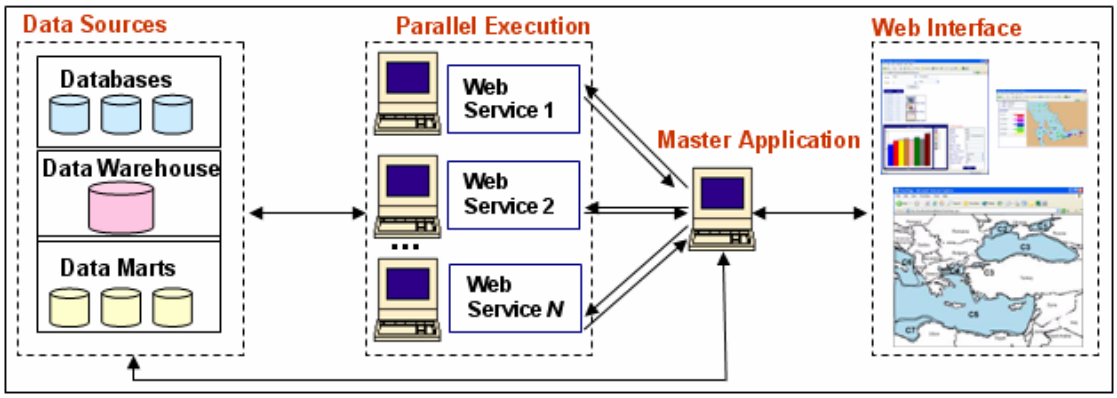


Figure 6.1 The general structure of the model

Data access should be authenticated through a concept of user/user group and right management. In particular, access rights can also be granted not only for the data, but also for the functions on the data, such as import, export, query etc.

6.4 Parallel Execution of Web Services

Web service-based parallel clustering application consists of two steps as shown in Figure 6.2. In the first step, threads are created and started for each Web Service. Join method causes a calling procedure to wait either until a thread is done or until the call times out if a timeout is specified. After all web services have finished processing, Merge_Results procedure is called to merge all partial results.

```
Dim Tasks As New WebServiceClass
 Dim Thread1 As New System.Threading.Thread(AddressOf Tasks.Task1)
 Dim Thread2 As New System.Threading.Thread(AddressOf Tasks.Task2)

Tasks.StrArg = "Some Arg" ' Set an argument
Thread1.Start()    ' Start the new thread.
Thread2.Start()    ' Start the new thread.
Thread1.Join()     ' Wait for thread 1 to finish.
Thread2.Join()     ' Wait for thread 2 to finish.

Merge_Results()
```

(a)

```
Class WebServiceClass
    Friend StrArg As String
    Friend RetVal As Boolean

    Sub Task1()
        Dim DS1 As New DataSet
        Dim a = New WebReference1.Service1
        DS1 = a.ST_DBSCAN_Clustering_Algorithm()
        RetVal = True     'Set a return value
    End Sub

    Sub Task2()
        Dim DS2 As New DataSet
        Dim a = New WebReference2.Service1
        DS2 = a.ST_DBSCAN_Clustering_Algorithm()
        RetVal = True     'Set a return value
    End Sub
End Class
```

(b)

Figure 6.2 (a) Threads are created and started for each web service. When the parallel execution of all web services is completed, all data portions are completely handled on a new dataset by applying a merge algorithm. (b) Two web services are invoked in parallel

The first step is followed by a parallel invocation of *N* web services running on different servers. Each web service is responsible for clustering one portion of data and each web service executes the same clustering algorithm, ST-DBSCAN, which is described in section 2.4. It is possible to increase the number of web services by creating and, starting new threads and by adding new invocation lines for new web services.

The average time is assumed to obtain a response from any web service *1* through *N* to be *S* time units. If *N* web services are executed in sequential order, the result can be obtained after *S***N* time units. However, if *N* web services are executed in parallel, the result can be obtained after approximately *S* time units. In parallel execution, the time spent waiting for all *N* web services to respond is approximately equal to the application's overall response time. In actuality, the time to execute *N* web services takes MAX $\{S_i\}$, where S_i is the time it takes to execute web service *i* and $i = 1, \ldots, N$. Because the application's overall response can only be obtained after all *N* web services have responded (Menasce, 2004).

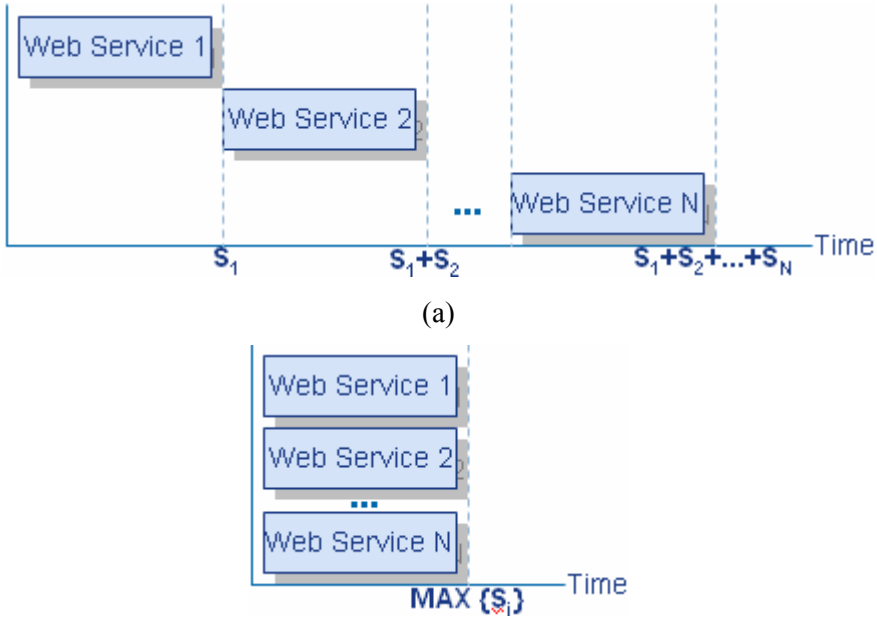


Figure 6.3 (a) The concurrent execution of web services (b) The parallel execution of web services

6.5 Possible Communication Problems

In our model, the communication between web services is not needed. The communication between master application and web services must only be provided. Because a master application invokes a set of web services and then collects and combines the partial results after web services have finished processing. Web services are invoked for execution remotely, not locally. For this reason, possible network failures should be raised depending on communication between master machine and remote machines. Each request of master machine must be responded within the maximum expended time period. A web service cannot be executed for several reasons such as network connection problems, service disconnected for maintenance, or service overloaded (Maamar et al., 2003). For these reasons, it is necessary to provide a reliable message delivery as the ability to guarantee reliable communication between web service client applications and web services.

In order to increase the quality of Web Service-Based Parallel Clustering, the web services should be well-defined, observable, and guaranteed. However, remote machine and network communication based failures may occur unexpectedly. Handling these kinds of errors, a special messaging mechanism must be provided. Basically a return value can be used for detecting the web service's end of job state. If an error happens, the return value will not be set correctly and this will cause a timeout condition for the related thread. However, the related task of the unreachable web service can be executed with re invoking another machine's web service.

For many parallel applications, a high-performance network is essential for efficiency. At present, neither the latency nor the bandwidth across the network is satisfactory. For this reason, high speed network and high aggregate network bandwidth are important criterions for efficient Web Service-Based Parallel Clustering. One solution is to exploit the communication capabilities of networks of workstations. Another possibility is to carry out Web Service-Based Parallel Clustering on a cluster of local workstations or on an intranet connecting all

workstations inside a company. Local clients on local hosts are generally guaranteeing fast communication.

6.6 Application

In order to compare clustering with single processing unit and with parallel processing architecture, the clustering algorithm described in chapter 2 was applied on the seawater temperature values in the data warehouse which is also described in chapter 2. The task of clustering is to discover regions that have similar seawater temperature characteristics.

The case study has been implemented with three web services on three PCs. PCs are linked together using a 100 Mb/s Ethernet network. Each of the process nodes features the following configuration: 2.6 GHz Pentium IV processor; 512 MB of main memory; 80 GB of storage capacity. The first process node discovers *similar* regions in the Black Sea and the Marmara Sea by applying our clustering algorithm. While the second node discovers clusters in the Aegean Sea, the third node works on data related to the Mediterranean Sea. The input parameters of the algorithm designated as *Eps1=3*, *Eps2=0.5*, and *MinPts=15*.

The bandwidth of the network didn't cause a bottleneck that would degrade the performance of the system. The level of traffic that takes place as a result of the exchange of communications between *master application* and *slave web services* is quite small.

We run the same clustering algorithm on three machines by using three web services. As shown in Figure 2.7, the first web service discovered the clusters C1, C2, and C3 in Black Sea, the second web service discovered the clusters C4, and C5 in Aegean Sea, and the third web service discovered the clusters C5, C6, and C7 in Mediterranean Sea. Whereas clustering time takes S time units when we execute only one machine, it takes approximately S/3 time units when we execute three web services in parallel for clustering. So, clear reduction was seen in computation time

of the algorithm when we run several web services in parallel. When larger datasets are used to be clustered, the differences are more apparent.

CHAPTER SEVEN
CLUSTER VALIDATION

7.1 Overview

One of the most important issues in cluster analysis is the evaluation of clustering results to validate the outcome of a clustering method. Cluster validation is the assessment of the quality and the correctness of the outcome of a cluster analysis. Most cluster analysis methods generate a clustering in all datasets without checking the accuracy of cluster structure. Often, different cluster analysis methods generate different clustering on the same data and it has to be decided which one is the optimal, if any. We define, here, the term "optimal" clustering scheme as the outcome of running a clustering algorithm that best fits the inherent partitions of the data set. Furthermore, if the clustering algorithm parameters are assigned an improper value, the clustering method may result in a partitioning scheme that is not optimal for the specific data set leading to wrong decisions.

The correctness of clustering algorithm results is verified using appropriate criteria and techniques. (Halkidi et al., 2001) In this study, the clustering results of the new algorithm ST-DBSCAN is justified by using RS (R-Squared) cluster validity index.

7.2 Cluster Validation Methods in Data Mining

The questions like "how many clusters are there in the data set?", "does the resulting clustering scheme fits our data set?", "is there a better partitioning for our data set?" call for clustering results validation and are the subjects of methods discussed in the literature. They aim at the quantitative evaluation of the results of the clustering algorithms and are known under the general term *cluster validity* methods.

The problems of deciding the number of clusters better fitting a data set as well as the evaluation of the clustering results has been subject of several research efforts (Halkidi et al., 2001; Zaiane et al., 2002). Basic principles for cluster validation can be distinguished as three approaches: external criteria, internal criteria and relative criteria.

The external and internal criteria approaches are based on statistical tests and their major drawback is their high computational cost. Moreover, the indices related to these approaches aim at measuring the degree to which a data set confirms an a-priori specified scheme. They generally use Monte Carlo methods to evaluate whether the clustering is significantly different from chance. These methods are clearly very expensive in processing time and only tell us that the clustering result is not pure chance.

The Relative Criteria does not involve statistical tests but attempts to evaluate among several results arising from different parameter settings. Here the basic idea is the evaluation of a clustering structure by comparing it to other clustering schemes, resulting by the same algorithm but with different parameter values. In this study, we focus on this approach. In this approach, the most common used validity indices are RS (R-Squared) index, CD (Centroid Distance) index and SD (Separation Distance) index (Halkidi et al., 2001). In this study, the clustering results of the new algorithm ST-DBSCAN is justified by using RS cluster validity index.

7.3 Visual Cluster Validation

Since clustering algorithms define clusters that are not known a prior information, the clustering results require some kind of evaluation in most applications. Visualization of the data set is a crucial verification of the clustering results. The visual method allows the user to visually examine the clusters created with a clustering algorithm and determine the genuine clusters found.

In a case study, we used visual cluster validation method to interactively compare our results and the visual cluster results. Figure 7.1 verifies our clustering method visually. While Figure 7.1.a shows the Topex/Poseidon satellite map, Figure 7.1.b shows our clustering result obtained by the usage of the wave height dataset measured on January 28, 2001. The goal of this application is to discover regions that have similar wave height values. In this application, we assigned input parameters as *Eps1=1*, *Eps2=0.25*, and *MinPts=15*.

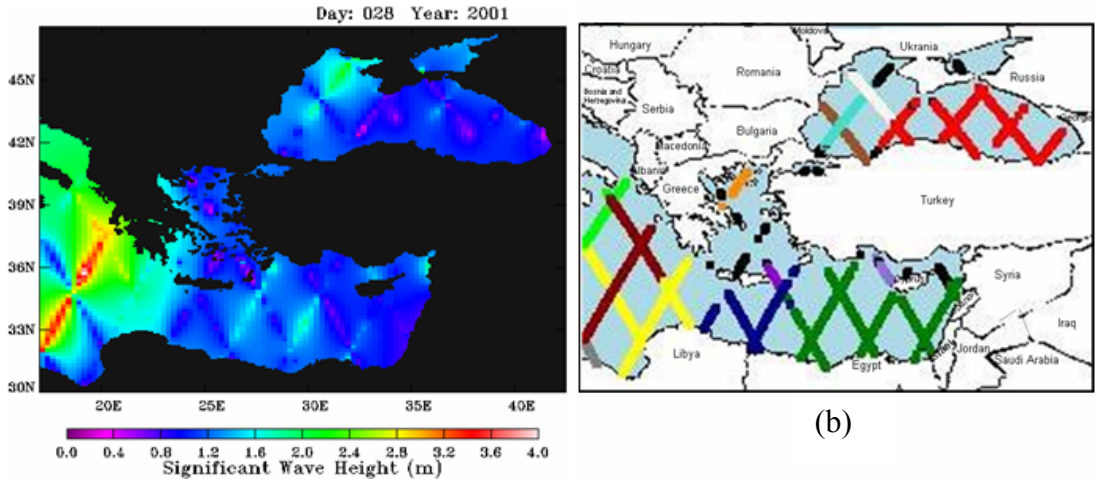


Figure 7.1 Visual cluster validation (a) Satellite map (b) The map which shows the clustering results

7.4 Cluster Validation with RS Index

In order to validate the clustering results, some validity indexes can be used such as RS index, CD index, and SD validity index. In this study, we used RS index which may be considered as a measure of the degree of difference between clusters. Furthermore, it measures the degree of homogeneity between groups. RS is the ratio of SS_b to SS_t, where SS means sum of squares, SS_b (Eq. 7.1) is a measure of difference between clusters, SS_w (Eq. 7.2) is a measure of difference within clusters and SS_t (Eq. 7.3) is equal to $SS_b + SS_w$.

$$SS_b = \sum_{j=1}^{p} n_j (\bar{x}_j - \bar{x})^2 \qquad (7.1)$$

$$SS_w = \sum_{j=1}^{p} \sum_{i=1}^{n_j} (x_{ij} - \overline{x}_j)^2 \quad (7.2)$$

$$SS_t = \sum_{j=1}^{p} \sum_{i=1}^{n_j} (x_{ij} - \overline{x})^2 \quad (7.3)$$

where p is number of clusters, n_j is the number of observations in the cluster j, $\overline{x}$ is the overall mean (the sum of all the observations divided by the total number of observations), $\overline{x}_j$ is the mean of the cluster j, and x_{ij} is the i^{th} observation in the cluster j.

For example, assume that there are three clusters with the values C1={2, 4, 3, 4, 5, 6}, C2={9, 10, 10, 7, 8, 10}, and C3={4, 5, 6, 3, 7, 5}. The total of all values is 108, the overall mean ($\overline{x}$) is 108/18=6 and the means of C1, C2, and C3 are 4, 9, and 5 respectively. So SS_b, SS_w, and SS_t are calculated as follows:

$SS_b = 6 * (4-6)^2 + 6 * (9-6)^2 + 6 * (5-6)^2 = 84$

$SS_w = (2-4)^2 + (4-4)^2 + (3-4)^2 + (4-4)^2 + (5-4)^2 + (6-4)^2 + (9-9)^2 + (10-9)^2 + (10-9)^2 + (7-9)^2 + (8-9)^2 + (10-9)^2 + (4-5)^2 + (5-5)^2 + (6-5)^2 + (3-5)^2 + (7-5)^2 + (5-5)^2 = 28$

$SS_t = (2-6)^2 + (4-6)^2 + (3-6)^2 + (4-6)^2 + (5-6)^2 + (6-6)^2 + (9-6)^2 + (10-6)^2 + (10-6)^2 + (7-6)^2 + (8-6)^2 + (10-6)^2 + (4-6)^2 + (5-6)^2 + (6-6)^2 + (3-6)^2 + (7-6)^2 + (5-6)^2 = 112$

$RS = SS_b / SS_t = 84 / 112 = 0.75$

The values of RS range between 0 and 1. In case that the value of RS is zero (0) indicates that no difference exists among groups. On the other hand, when RS equals 1 there is an indication of significant difference among groups. Larger RS implies the better quality. Figure 7.2 shows the screen shot of the program which is developed to present the evaluation and validation of our clustering results. As shown in Figure

7.2, RS validity index of our clustering results given in chapter 2 is 0.91. So the difference between clusters discovered by new method is quite high and the homogeneity within the clusters is quite low. Thus the mathematical quality and reliability of our clustering results are quite good.

Daytime Sea Surface Temperature

Overall Mean ($\bar{x}$) 17.83

	# of Obj.	Means ($\bar{x}_j$)	SS_b	SS_w
C1	63	2,74	14345.61	5.01
C2	92	10.05	5568.61	10.01
C3	1306	13.70	22276.31	1583.53
C4	90	16.35	197.14	4.52
C5	3061	20.14	16333.80	5068.24
C6	349	14.35	4226.53	162.45
C7	279	22.73	6698.79	59.95

SS_b(Between Groups)	69646.79
SS_w(Within Gorups)	6893.71
SS_t(Total) = SS_b+ SS_w	76540.5
RS Index (SS_b/ SS_t)	0.91

Calculate

NightTime Sea Surface Temperature

Overall Mean ($\bar{x}$) 17.68

	# of Obj.	Means ($\bar{x}_j$)	SS_b	SS_w
C1	63	2,72	14099.50	4.66
C2	92	9.68	5888.00	44.80
C3	1306	13.42	23700.77	2119.62
C4	90	16.56	112.90	20.78
C5	3061	20.05	17193.33	4854.70
C6	349	14.02	4675.06	273.38
C7	279	22.68	6975.00	31.25

SS_b(Between Groups)	72644.56
SS_w(Within Gorups)	7349.19
SS_t(Total) = SS_b+ SS_w	79993.75
RS Index (SS_b/ SS_t)	0.91

Calculate

Figure 7.2 The evaluation and validation of the clustering results

CHAPTER EIGHT
VISUALIZATION

8.1 Overview

The last step of a complete knowledge discovery process is the representation and visualization of extracted patterns. In order to present our data mining results, we developed a visualization tool. The visualization tool is based on web technology. The users can use the system by opening an internet browser, without installing any program. So they can use it from any place such as from home, an office, or a hotel.

In this thesis, a high-level software, named by MARDEX (MARine Data EXplorer), is also provided to process and present physical, chemical, and biological marine data. It is useful for the electronic publication of data on the internet. It includes some graphical representations such as bar, line and pie charts. The objective of this application is to allow users to monitor and to query marine observations easily. By using this application, users can monitor collected marine data without learning a database tool, a programming language or a query language. Users are able to query data with an online interface and to retrieve filtered data or summary data to get information on their works.

8.2 The Visualization of Data Mining Results

The visualization tool developed for representing data mining results mainly consists of two steps: drawing of the map by taking geographical coordinates of any place and displaying the selected data on the map.

8.2.1 Map Drawing

In the visualization tool, the map of any place (sea, ocean, city, country, etc.) is dynamically drawn by taking the geographical coordinates from a database or a text file. GDI+ library is used for the drawing of the maps. GDI stands for Graphic

Device Interface. The GDI version of .NET is called GDI+. GDI+ optimizes many of the capabilities of GDI and also provides additional features. Map drawing process mainly contains three steps: (i) obtaining geographical coordinates, (ii) calculation of margins, and (iii) usage of drawing objects.

- Obtaining Geographical Coordinates: This step is the process of getting geographical X and Y coordinates of the place from a text file or a database. We find the coordinates of all countries of the world from the web address (SURFER, 2003) as blanking (*.bln) files.

- Calculation of Margins: Before the drawing operation, it is necessary to find the maximum and minimum coordinates for the healthy drawing. In addition, it is necessary to determine suitable width and height values of the Bitmap Object. In other case, some parts of the map should be invisible.

Usage of Drawing Objects: In Microsoft .NET platform, the *System.Drawing* namespace provides basic GDI+ functionality. It contains the definition of basic classes such as Brush, Pen, Graphics, Bitmap, Font etc. The Graphics class plays a major role in GDI+ and contains methods for drawing to the display device.

Figure 8.1 shows the maps of Izmir Gulf and Turkey drawn by the visualization tool. Figure 8.2 shows an example BLN file which includes the geographical coordinates of Izmir Gulf. The first line, 542, means that following 542 lines express one closed area. After 542 lines, the number 69 indicates that the following 69 line express another closed area, big island. In addition, the following 15 lines form the third closed area, small island.

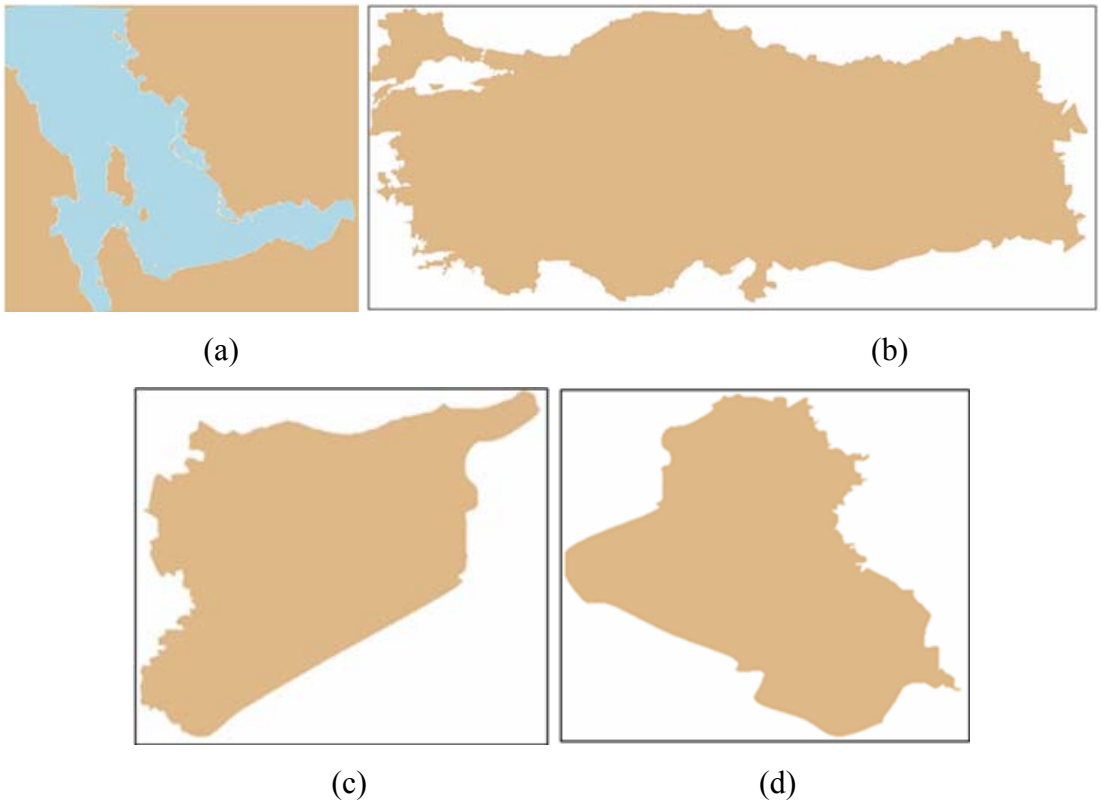


Figure 8.1 The maps of (a) Izmir Gulf (b) Turkey (c) Syria (d) Iraq drawn by the visualization tool

```
542 0
26.50   38.6586
26.5011 38.6558
26.5067 38.6525
26.5133 38.6483
...
69 1
26.6983 38.5481
26.7022 38.5433
26.7056 38.5419
26.705  38.5442
...
15 1
26.7089 38.7031
26.7119 38.7
26.7164 38.6956
26.7197 38.6931
...
```

Figure 8.2 An example file which includes geographical coordinates of Izmir Gulf

A part of program code for map drawing is shown in Figure 8.3. Firstly two procedures *Read_Geographical_Coordinates* and *Find_Min_Max* are called to read geographical coordinates of the place from a file or from a database and to find minimum and maximum X, Y coordinates. Then a bitmap object and a graphic object is created with suitable height and width values. Then the background color of the graphic object, the color of the pen object, and the color of closed area are determined. After that the closed areas are drawn continuously by reading how many lines will form this area and by reading the geographical coordinates of these lines. *Separation* function separates X and Y coordinates which are originally in the same line and then returns them in PointF type. X and Y coordinates are separated by determining blank characters. *objGraphics.DrawLine* and *objGraphics. FillClosedCurve* functions are used to draw map. BoolValue variable is used to draw closed areas with different colors. Finally the drawing is saved as a JPEG (Joint Photographic Experts Group) picture file.

```
... 'Variable Definitions

Read_Geographical_Coordinates()

Find_Min_Max()

x_margin = max_x - min_x
y_margin = max_y - min_y

Dim objBitmap As New Bitmap(x_margin, y_margin)
Dim objGraphics As Graphics = Graphics.FromImage(objBitmap)

objGraphics.Clear(Color.LightBlue)
              'or Color.White

Dim p As New Pen(Color.Black, 2)
Dim pn, pnex As PointF

Dim Color1 As New System.Drawing.SolidBrush(System.Drawing.Color.
                                                    BurlyWood)
Dim Color2 As New System.Drawing.SolidBrush(System.Drawing.Color.
                                                    LightBlue)
                                                'or Color.White
Dim BoolValue As Boolean = False

For i = 1 To Number_of_Closed_Areas
  str = Take_New_Value()   'Read number of lines
  number = CInt(str)-1
  Dim CurvePoints(number) As PointF
  str = Take_New_Value()   'Read geographical coordinate(X and Y)
  pnex = Seperation()
  CurvePoints(0) = pnex

  For j = 1 To number
    str = Take_New_Value()  'Read geographical coordinate (X and Y)
    pn = Seperation()
    CurvePoints(j) = pn
    objGraphics.DrawLine(p, pn, pnex)
    pnex = pn
  Next

  If BoolValue = False Then
    objGraphics.FillClosedCurve(Color1, CurvePoints)
  Else
    objGraphics.FillClosedCurve(Color2, CurvePoints)
  End If

  BoolValue = True
Next

objBitmap.Save("c:\map.jpg",System.Drawing.Imaging.ImageFormat.Jpeg)
Image1.ImageUrl = "c:\map.jpg"
```

Figure 8.3 A part of map drawing code

8.2.2 Displaying Data on the Map

Displaying data on a map can be sometimes more understandable and preferable than the displaying data in a table. Because pictures allow humans easily to memorize and understand. Three steps are followed for displaying data on the map: obtaining data to represent, editing of color legend, and drawing objects on the map.

- Obtaining Data to Represent: It is the process of getting selected data from the database. The dataset is filtered according to the user selected criterions. For example, as shown in Figure 8.4, the user can select a time interval or the type of the data such as water temperature, salinity, conductivity, density, etc.

- Editing of Color Legend: It is the process of determination of colors according to the values in the data set. Data are grouped and color-coded. The colors are chosen to correspond to some data intervals.

Drawing Objects on the Map: It is the process of drawing some shapes on some locations of the map to represent the data. For example, as shown in Figure 8.5, a rectangular can be drawn for each object with corresponding color. In this figure, water temperature values of Izmir Gulf in October 2001 are shown on the map. The color tone should be determined for each data value. A part of program code for this process is showed in Figure 8.6. Each rectangular is drawn and filled by using *objGraphics.FillRectangle* function. *Calculate_Color_Tone* function is used to determine the corresponding color of the current data value. Finally the drawing is saved as a jpeg picture file.

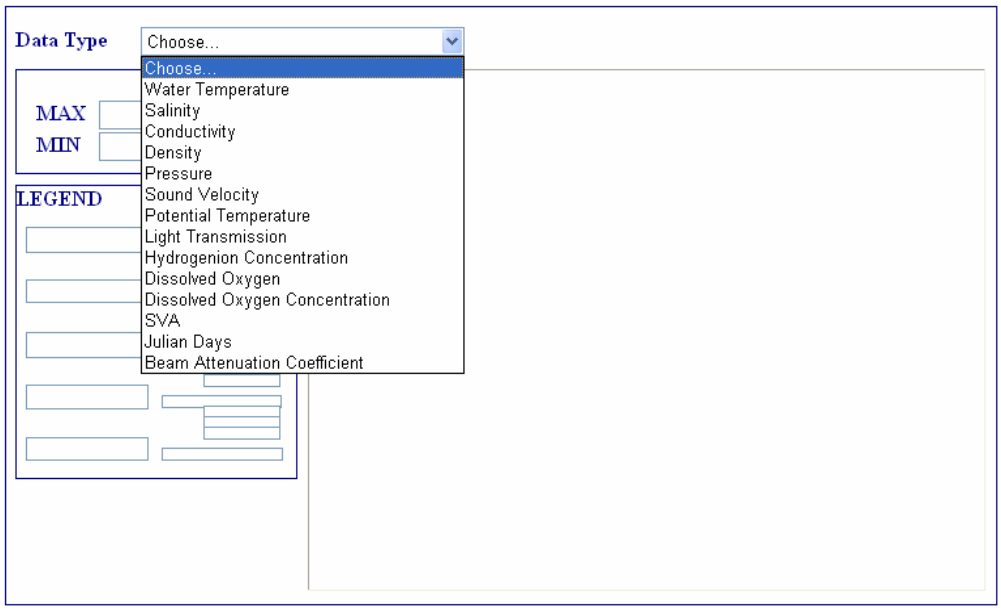


Figure 8.4 Getting data according to the user selection

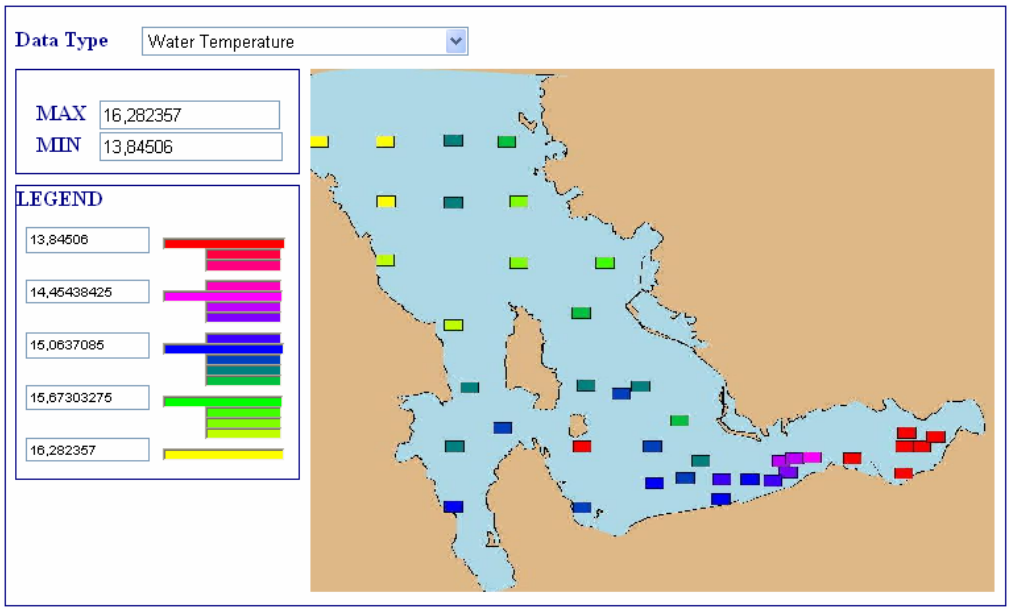


Figure 8.5 Water temperature of Izmir Gulf in October 2001

```
... 'Drawing Map Code (Figure 7.3)

... 'Variable Definitions

Dim mydataset As New Dataset
Mydataset = Retrieve_Data_From_Database()

min = Find_Min_Value_of_Dataset
max = Find_Max_Value_of_Dataset

increment = (max - min) / 4

Textbox1.Text = min
Textbox2.Text = min + increment
Textbox3.Text = min + increment * 2
Textbox4.Text = min + increment * 3
Textbox5.Text = max

For i = 0 To Number_of_Stations - 1
   Data_Value = Mydataset.Tables(0).Rows(i).Item(0)
   DrawColor=Calculate_Color_Tone(Data_value)

   pn.X =Read_X_Coordinate_of_Station()
   pn.Y =Read_Y_Coordinate_of_Station()

   objGraphics.FillRectangle(DrawColor, pn.X, pn.Y, 3, 3)
Next


objBitmap.Save("c:\map2.jpg",System.Drawing.Imaging.ImageFormat.Jpg)
Image1.ImageUrl = "c:\map2.jpg"
```

Figure 8.6 Code for displaying data on the map

8.3 MARDEX Program

In this thesis, the software, named by MARDEX (MARine Data EXplorer), is provided for physical, chemical, and biological data processing and presentation. By using developed tool, marine scientists can monitor and query collected marine data without learning a database tool, a programming language or a query language. This study can be useful by establishing and developing some data processing techniques and user friendly presentation formats.

8.3.1 Program Capabilities

MARDEX was designed to be flexible and easy-to-use. Users are not required to know the details of the data storage format and are not required to have any programming experience. The tool includes some graphical representations such as bar, line and pie charts. The features of the graphics can be modified easily, and they can be stored as picture files on disk for later use.

MARDEX can manage very large data collections and run on widely available hardware platform. It includes the active involvement through all operational phases, including data retrieving, processing, and visualizing. It also provides a security mechanism for preventing unauthorized persons to enter the system. MARDEX is a web based application so it provides independence of the working platform and program installation. Figure 8.7 shows login page and the main page of the program. The login page provides access to the web application for users with a valid User Id and Password. The main page includes a menu for physical, chemical, biological or mixed analyses. Mixed queries running on physical, chemical and biological characteristics of a marine environment are fundamental for effective management of the marine researches, and the development, support and management of marine industry.

After the selection of a data type, the user should select a data category, a table, and some columns in this table. Figure 8.8 shows an example screen shot which appears on the screen if the user selects physical data type, CDT system data category, IzmirGulf_01_2001 table. After the selection of the columns, the data in the table and a sub menu appears on the screen as shown in Figure 8.9. If the user selects chart drawing item in the sub menu, the graphical representation of the data is shown on the screen as a bar, pie, or line chart. (Figure 8.10, Figure 8.11) The user can easily switch between these types by pressing related buttons. The features of the charts such as chart title, legend position, legend width etc. can also be modified easily, and they can be stored as picture files on disk for later use.

Figure 8.7 Some screen shots from MARDEX program

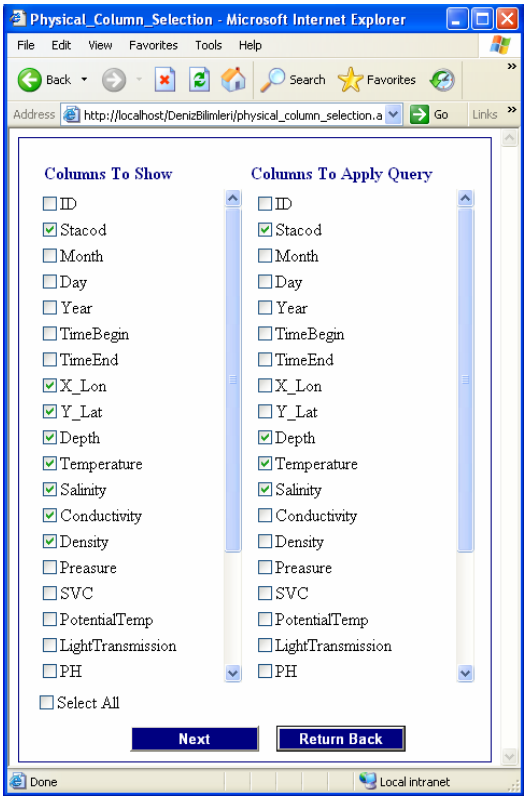


Figure 8.8 Selection of physical parameters to show and to apply a query

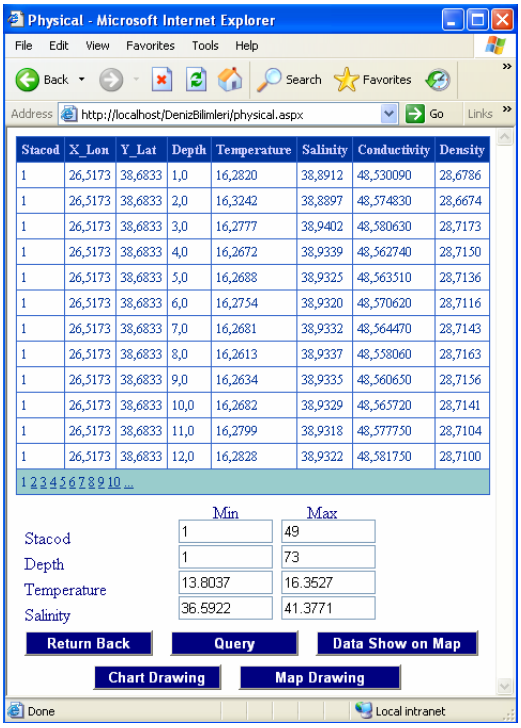


Figure 8.9 The window that includes a sub menu

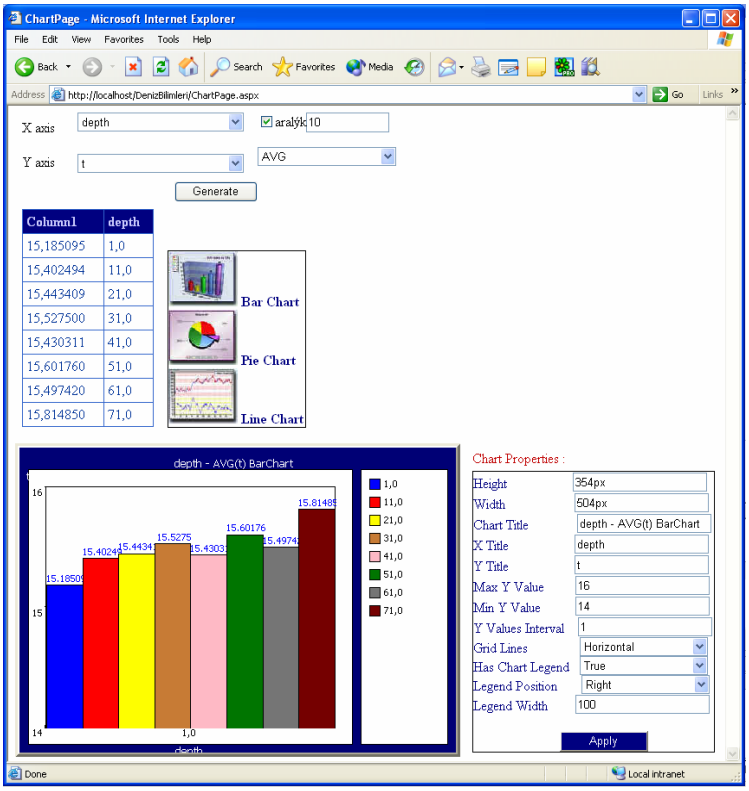


Figure 8.10 Bar chart representation of the data

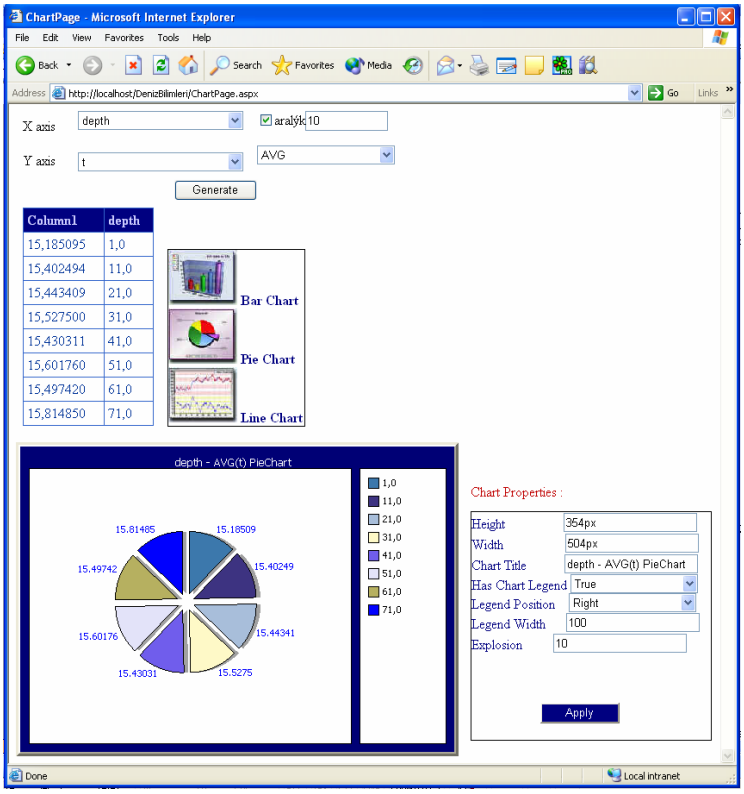


Figure 8.11 Pie chart representation of the data

8.3.2 Data Categories Supported by MARDEX Program

Three basic data categories are supported by MARDEX program: physical, biological and chemical. (Figure 8.12) Physical data type is divided into three groups: CDT system data, flow meter data and seiche data, and rainfall-temperature data. CTD (Conductivity Temperature and Depth) systems in research vessels measure some physical seawater characteristics such as temperature, salinity, density, pressure PH, conductivity, light transmission, sound velocity etc. Flow meter (current meter) data is also provided in the system. Flow meters are used widely in the industry by researchers and marine coastal consultants to measure seawater currents. Parameters typically recorded in current meter data comprise east-west and north-south velocity components, current direction, water temperature for each current meter and the type of the current meter. In order to measure seiche values, researches generally use a seicchi disk which is a black and white colored disk and usually 20-30 cm in diameter. This disc is used to measure depth of water clarity by lowering into a water body. Clarity of the water is measured as the depth at which the circle can no longer be seen. Seiche values relating the depth at which the disk disappears are measured empirically. Changes in rainfall temperature can also be observed by MARDEX. This type of data includes the average of measures taken in four time periods (0 pm-6 pm, 6pm-12 pm, 0 am-6 am, 6 am-12 am) every day.

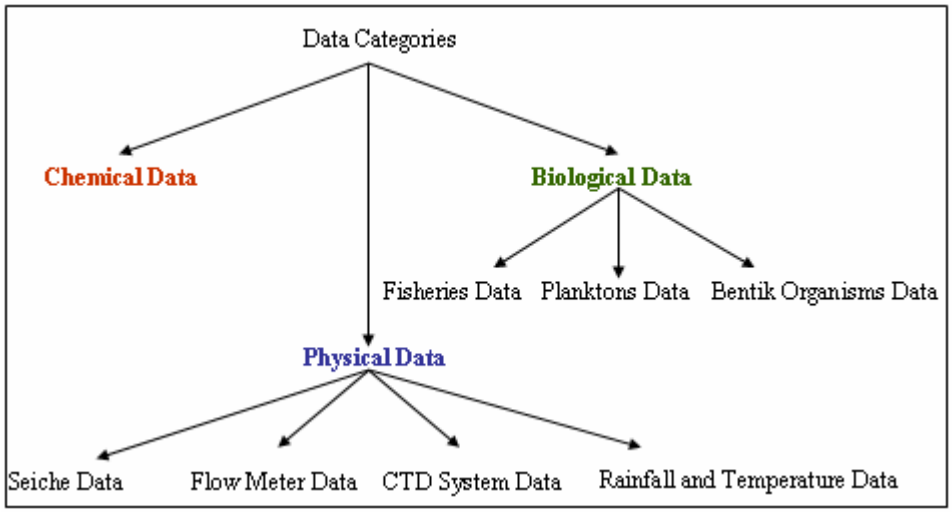


Figure 8.12 The current data categories in MARDEX system

The scientists analyze the chemical parameters in aqueous samples. The sea can be chemically variable in both space and time, so scientists are involved in making time-series measurements. Chemical marine data includes some parameters such as silicate, chlorine, ortho phosphorus, faecal coliforms, tris phosphine oxide etc. Data is available not only for the sea surface but also for each meter.

Marine biology data involves many records for seawater animals, freshwater animals, coastal water animals and marine plants. An illustration of each animal is accompanied by details of their class, order, size, family, scientific name, range, diet, conservation status, habitat and similar properties. By using MARDEX program, scientist are able to carry out their researches on three areas: fisheries data, planktons (living creature with one or more cells) and bentik organisms (organisms living in marine mud).

CHAPTER NINE
CONCLUSIONS

9.1 Conclusions and Future Works

This study meets the need of interdisciplinary research. It combines computer science and marine science by providing new computer based techniques and technologies for modeling, collecting, archiving, analyzing, mining and visualizing marine data.

This thesis covers designing and constructing a data center needed by marine scientists to fulfill the specific mission objectives. It also includes new algorithms, techniques and approaches for analyzing, mining and visualizing marine data. It provides the necessary background to fully understand the clustering, outlier detection and decision tree techniques in data mining.

This thesis contains necessary issues and criteria in developing marine data management projects, and helps determine the direction of future efforts in marine data management. It takes in consideration the needs of marine researches, current topics in marine information technology, and the deficiencies of existing data management projects in marine science.

In the thesis, highly beneficial and satisfying applications are presented to demonstrate the applicability of new algorithms to real world problems. In most of the applications, a data warehouse is used, having 8 GB size and including sea surface temperature, sea surface height residual, significant wave height, and sea wind speed values of Turkish seas between years 1992 and 2002. In some applications, physical parameters of Izmir Gulf provided by Institute of Marine Sciences and Technology at Dokuz Eylul University is used, such as salinity, density, pressure, light transmission, conductivity, ph, dissolved oxygen concentration, sound velocity etc.

One of the practical results of this study is that the clustering algorithm introduced in this thesis can be applied on spatio-temporal data, in addition to ordinary data. Experimental results demonstrate that the algorithm appears to be very promising when spatio-temporal data is used to be clustered. The algorithm has three input parameters: *Eps1*, *Eps2* and *MinPts*. According to the sensitivity analysis results of the algorithm, even there is a huge variation of *Eps1* and *MinPts* parameters, there is a little influence on the output. There is no any change in the number of noise objects when the value of these parameters increase. Furthermore, there is nearly no change in the number of clusters. The results also show that only *Eps2* parameter has considerable influence on the algorithm output.

Another practical results of this study is that the outlier detection algorithm introduced in this thesis can be applied on spatio-temporal data to detect rare events and exceptional cases. It consists of three steps: clustering, checking spatial neighbors to identify spatial outliers, and checking temporal neighbors to identify spatio-temporal outliers. Our algorithm can be used not only for analyzing marine data but also for carrying out many researches such as for analyzing astronomical data to detect unusual objects, for filtering out high frequency noise in acoustic data, for detecting extreme events in weather and climate and for performing image processing.

The last step of a complete knowledge discovery process is the presentation and visualization of extracted patterns. In order to present our data mining results, we developed a visualization tool which consists of two steps: drawing of the map by taking geographical coordinates of any place and displaying the selected data on the map.

Working on large scale datasets requires much CPU and memory power, and sufficient algorithms. During this study, different algorithms were developed to cluster large dataset. It was seen that, some existing clustering algorithms are too slow for this kind of operation because of the general nature of the algorithm. By using a specific and suitable algorithm, lots of time can be gained, and more detailed

analysis can be done. In addition to the suitable algorithm, fast processing of data depends on the structural organization of the stored information and the availability of suitable indexing methods. While a well designed data structure can facilitate to rapidly extract the desired information from a set of data, suitable indexing methods can provide to quickly locate single or multiple objects. Well known spatial indexing techniques include Quadtrees, R-Trees, and X-Tree. In this study, an improvement version of R-Tree indexing method was used to handle spatio-temporal information. Some nodes in R-Tree were created for each spatial object and linked them in temporal order. During the application of the algorithm, this tree is traversed to find the spatial or temporal neighbor objects of any object. In addition to spatial index structure, some filters should also be used to reduce the search space for data mining algorithms. These filters allow the operations on neighborhood paths by reducing the number of paths actually created. They are necessary to speed up the processing of queries.

In addition to the structural organization of the stored information and the availability of suitable indexing methods, parallel processing can be used to provide fast processing of data. Writing a parallel program requires some special knowledge and experience about a parallel programming interface such as MPI or PVM. It is necessary to learn the syntax of the commands to provide messaging between machines. In order to overcome these disadvantages, a new approach was proposed in this thesis: web service-based parallel computing. This approach includes the parallel execution of web services. Each web service is responsible for processing one portion of data, so the work load is distributed over several web services. The purpose of the study is to allow users to satisfy their parallel processing needs without the knowledge of any parallel programming interface such as MPI or PVM. This approach is useful to simplify the steps of developing parallel processing programs.

In the future, the data warehouse size can be increased by adding new features released by NASA such as sea currents which is provided by OSCAR satellite. The cluster validation techniques other than RS can be applied to justify the number of

clusters in a clustering result because the result of clustering needs to be validated in most applications.

REFERENCES

Abraham, T., & Roddick, J.F. (1999). Survey of spatio-temporal databases, *GeoInformatica*, *3* (1), 61-99.

Adam, N.R., Janeja, V.P., & Atluri, V. (2004). *Neighbourhood based detection of anomalies in high dimension spatio-temporal sensor datasets*, ACM Symposium on Applied Computing, Nicosia Cyprus, pp. 576-583.

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). *Automatic subspace clustering of high dimensional data for data mining applications*, in: Proceedings of ACM-SIGMOD International Conference on Management of Data, Seattle, WA, pp. 94-105.

Ankerst, M., Breunig, M.M., Kriegel, H-P., & Sander J. (1999). *OPTICS: Ordering points to identify the clustering structure*, in: Proceedings of ACM SIGMOD International Conference on Management of Data, Phila-delphia PA, pp. 49-60.

Aoying, Z., & Shuigeng Z. (2000). Approaches for scaling DBSCAN algorithm to large spatial database. *Journal of Computer Science and Technology*, *15* (6), 509-526.

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data* (3rd ed.). New York: John Wiley.

Bosworth, A. (2001). *Developing web services*. in: Proceedings of IEEE 17th International Conference on Data Engineering, Heidelberg, Germany, pp. 477 – 481.

Böhlen, M.H., Jensen, C.S., & Scholl, M.O. (Ed.). (1999). *Spatio-temporal database management*. Berlin: Springer-Verlag

Böhm, C., Berchtold, S., Kriegel, H-P., & Michel, U. (2000). Multidimensional index structures in relational databases, *Journal of Intelligent Information Systems (JIIS)*, *15* (1), 51-70.

Breunig, M.M., Kriegel, H-P., Ng, R., & Sander, J. (2000). *LOF: Identifying density-based local outliers*, ACM SIGMOD Int. Conf. on Management of Data, Dallas, TX; 2000, p. 93-104.

Cios, K. J., Pedrycz, W, & Swiniarski, R. W., (1998), *Data mining methods for knowledge discovery*, Boston: Kluwer Academic Publishers.

Emery, W.J., Talley, L.D., & Pickard, G.L. (2005). *Descriptive physical oceanography* (7th ed.). Elsevier Publication.

Ester, M., Kriegel, H-P., Sander, J., & Xu, X. (1996). *A density-based algorithm for discovering clusters in large spatial databases with noise*, in: Proceedings of Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, pp. 226-231.

Ester, M., Kriegel, H-P., Sander, J., & Xu, X. (1998a). Clustering for mining in large spatial databases, *KI-Journal (Artificial Intelligence), Special Issue on Data Mining*, *12* (1), 18-24.

Ester, M., Kriegel, H-P., Sander, J., Wimmer, M., & Xu, X. (1998b). *Incremental clustering for mining in a data warehousing environment*, in: Proceedings of International Conference on Very Large Databases (VLDB'98), New York, USA, pp. 323-333.

Fisher. D. (1987). Knowledge acquisition via incremental conceptual clustering, *Machine Learning*, *2* (2), 139–172.

Freitas, A.A. (2002).Data Mining and Knowledge Discovery with Evolutionary Algorithms. Berlin: Springer.

Freshwater, J., Oxenford, B., & White, S. (2002). *Is the pH level affected by the depth of water in the MVMS retention pond?*. Department of Biology at Kenyon College, Retrieved May 20, 2003, from http://biology.kenyon.edu/HHMI/middle_school_2002/pH/index.htm

Guha, S., Rastogi, R., & Shim, K. (1998). *CURE: An efficient clustering algorithms for large databases*, in: Proceeding ACM SIGMOD International Conference on Management of Data, Seattle, WA, pp. 73-84

Guting, R.H. (1994). An introduction to spatial database system, *VLDB Journal*, *3* (4), 357-399.

Guttman, A. (1984). *R-trees: A dynamic index structure for spatial searching*, in: Proceedings of ACM SIGMOD Int. Conf. on Management of Data, Boston, Massachusetts, pp. 47-57.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On clustering validation techniques*, Journal of Intelligent Information Systems*, *17* (2-3), 107-145.

Han, J., & Kamber, M. (2001). *Data mining concepts and techniques*, CA San Francisco: Morgan Kaufmann Publishers, pp. 335-391.

Han, J., Kamber, M., & Tung, A.K.H. (2001). *Spatial clustering methods in data mining: A survey*. In H. Miller and J. Han (eds.), Geographic Data Mining and Knowledge Discovery, Taylor and Francis, London.

Hinneburg, A., & Keim, D.A. (1998). *An efficient approach to clustering in large multimedia databases with noise*, in: Proceedings of 4th International Conference on Knowledge Discovery and Data Mining, New York, NY, pp. 58-65.

Jain, A., Murty, M., & Flynn, P. (1999). Data clustering: A review, *ACM Computing Surveys*, *31* (3), 264-323.

Januzaj, E., Kriegel, H-P., & Pfeifle, M. (2004). Scalable density-based distributed clustering, *Lectures Notes in Computer Science*, *3202*, 231-244.

Johnson, T., Kwok, I., & Ng, R. (1998). *Fast computation of 2-dimensional depth contours*, in: Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining, New York, pp. 224-228.

Knorr, E.M., & Ng, R.T. (1998). *Algorithms for mining distance-based outliers in large datasets*, in: Proc. 24th Int. Conf. Very Large Data Bases, New York, NY, p. 392-403.

Knorr, E.M., Ng, R.T., & Tucakov, V. (2000). Distance based outliers: algorithms and applications, *Journal: Very Large Data Bases*, *8* (3-4), 237-253.

Kolatch, E. (2001). *Clustering algorithms for spatial databases: A survey*. Retrieved Septermber 10, 2003, from http://bit.kuas.edu.tw/~cscheng/research/paper/SpatialClustering.pdf

Kovács, L., Vass, D., & Vidács, A. (2004). *Improving quality of service parameter prediction with preliminary outlier detection and elimination*, in: Proc. 2nd Int. Workshop on Inter-Domain Performance and Simulation, Budapest, Hungary, p. 194-199.

Kut, A., & Birant, D. (2004). *An approach for parallel execution of web services*, in: Proceedings of IEEE International Conference on Web Services, San Diego, California, USA.

Lecture Notes - Earth Sciences 1 - Lecture 9: Seawater Physical Properties. (n.d.). Retrieved June 10, 2003, from http://ic.ucsc.edu/~jzachos/eart1/Notes/

Lecture Notes - Geology 117: The Oceans - Lecture 15: Properties of sea water, (n.d.). Retrieved July 14, 2003, from http://ijolite.geology.uiuc.edu/02SprgClass/geo117/

Ma, S., Wang, T.J., Tang, S.W., Yang, D.Q., & Gao, J. (2003). *A new fast clustering algorithm based on reference and density*, in: Proceedings of International Conference on Web-Age Information Management, Lectures Notes in Computer Science, Springer 2762, pp. 214-225.

Maamar, Z., Sheng, Q. Z., & Benatallah B. (2003). *Interleaving web services composition and execution using software agents and delegation*, in: Workshop on Web Services and Agent-based Engineering, Melbourne, Australia.

MacQueen, J. (1967). *Some methods for classification and analysis of multivariate observations*, in: Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297.

Magoon, O.T. (1995). *Coastal zone management dictionary*. United States Army Coastal Engineering Research Center, Washington.

Menasce, D.A. (2004). Response-Time Analysis of Composite Web Services, *IEEE Internet Computing*, *8* (1), 90-92.

Michaud, J.P. (1991). *A citizen's guide to understanding and monitoring lakes and streams*. Washington State Department of Ecology, Publications Office, Publication No 94-149, Olympia, Washington, pp. 73.

Mikalsen, T., Rouvellou, I., & Tai, S. (2001). *Reliability of composed web services from object transactions to web transactions*, in: Workshop on Object-Oriented Web Services, Tampa, Florida, USA.

National Aeronautics and Space Administration (NASA). (2003a). *PO.DAAC Ocean ESIP Tool (POET)*. Retrieved January 20, 2003, from http://poet.jpl.nasa.gov/

National Aeronautics and Space Administration (NASA). (2003b). *NAVOCEANO Satellite – Sea Surface Temperature Data Web Site*. Retrieved January 20, 2003, from http://podaac.jpl.nasa.gov/navoceano_mcsst/

National Aeronautics and Space Administration (NASA). (2003c). *Topex/Poseidon Satellite – Sea Surface Height and Sea Wave Height Data Web Site*. Retrieved January 28, 2003, from http://podaac.jpl.nasa.gov/woce/

National Aeronautics and Space Administration (NASA). (2003d). *QuikSCAT Satellite – Sea Winds Data Web Site*. Retrieved February 21, 2003, from http://podaac.jpl.nasa.gov/quikscat/

Ng, R.T., & Han, J. (1994). *Efficient and effective clustering methods for spatial data mining*, in: Proceedings of 20th International Conference on Very Large Data Bases, Santiago, Chile, pp. 144-155.

Qian, W.N., & Zhou, A.Y. (2002). Analyzing popular clustering algorithms from different view-points, *Journal of Software*, *13* (8), 1382-1394.

Papadimitriou, S., & Faloutsos, C. (2003). *Cross-outlier detection*, in: Proc. 8th International Symposium on Spatial and Temporal Databases, Greece, 199-213.

Roy, J., & Ramanujan, A. (2001). Understanding web services, *IEEE IT Professional*, *3* (6), 69-73.

Ruts, I., & Rousseeuw, P. (1996). Computing depth contours of bivariate point clouds, *Journal of Computational Statistics and Data Analysis*, *23* (1996), 153-168.

Saltelli, A., Chan, K., & Scott, E. M. (2000). *Sensitivity analysis*, John Wiley & Sons, Chichester, England.

Samet, H. (1990). *The design and analysis of spatial data structures*, MA: Addison-Wesley.

Seidman, C. (2001). *Data mining with SQL server 2000*, US: Microsoft Press.

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). *WaveCluster: A multi-resolution clustering approach for very large spatial databases*, in: Proceedings of International Conference on Very Large Databases (VLDB'98), New York, USA, pp. 428-439.

Shekhar, S., Lu, C-T., & Zhang, P. (2003). A unified approach to detecting spatial outliers, *GeoInformatica, Kluwer Academic Publishers*, *7* (2), 139-166.

Spieth, C., Streichert, F., Speer, N., & Zell, A. (2005). *Clustering based approach to identify solutions for the inference of regulatory networks*, in: Proceedings of the IEEE Congress on Evolutionary Computation, Edinburgh, UK.

SURFER Data files, Retrieved June 22, 2003, from http://www.goldensoftware.com/

Swartzman, G.L., & Kaluzny, S.P. (1987). *Ecological simulation primer*, New York: MacMillan Publishing.

Tan, P-N., Steinbach, M., & Kumar, V. (2005). *Introduction to data mining*, New York: Addison-Wesley.

Toxopeus, A.G. (1997).*Spatial and temporal modeling for sustainable management of tropical rainforest*, in: Proceedings of the International Conference on Geo-Information for Sustainable Land Management, Enschede, The Netherlands.

Vinod, H. (1969). Integer programming and the theory of grouping, *Journal of the American Statistical Association*, *64* (326), 506-519.

Walton, C.C., Pichel, W.G., & Sapper, J.F. (1998). The development and operational application of nonlinear algorithms for the measurement of sea surface temperatures with the NOAA polar-orbiting environmental satellites, *Journal of Geophysical Research*, *103* (C12).

Wang, W., Yang, J., & Muntz, R. (1997). *STING: A statistical information grid approach to spatial data mining*, in: Proceedings of 23rd International Conference on Very Large Data Bases (VLDB), pp. 186-195.

Wen, J-R., Nie, J-Y., & Zhang, H-J. (2002). Query clustering using user logs, *ACM Transactions on Information Systems*, *20* (1), 59-81.

Xu, X., Ester, M., Kriegel, H-P., & Sander, J. (1998). *A distribution-based clustering algorithm for mining in large spatial databases*, in: Proceedings of IEEE International Conference on Data Engineering, Orlando, Florida, pp. 324-331.

Zhang, T., Ramakrishnan, R., & Linvy, M. (1996). *BIRCH: An efficient data clustering method for very large databases*, in: Proceeding ACM SIGMOD International Conference on Management of Data, pp. 103-114.

Zaiane, O.R., Foss, A., Lee, C.-H., & Wang, W. (2002). On data clustering analysis: Scalability, constraints and validation, *Lecture Notes in Artificial Intelligence 2336*, 28-39.