SEMANTICS ORIENTED SPATIAL TEMPORAL DATA MINING FOR WATER RESOURCE DECISION SUPPORT

by

Guangyan Huang

BSc. and MSc. (Southwest Petroleum University, China) 1999 and 2002

A dissertation submitted in fulfillment
of the requirements for the degree of

Doctor of Philosophy

in the

School of Engineering & Science, Faculty of Health, Engineering & Science

of the

VICTORIA UNIVERSITY, AUSTRALIA

October 2011

DEDICATION

To my parents.

DOCTOR OF PHILOSOPHY DISSERTATION DECLARATION

"I, *Guangyan Huang*, declare that the PhD thesis entitled *Semantics Oriented Spatial Temporal Data Mining for Water Resource Decision Support* is no more than 100,000 words in length including quotes and exclusive of tables, figures, appendices, bibliography, references and footnotes. This thesis contains no material that has been submitted previously, in whole or in part, for the award of any other academic degree or diploma. Except where otherwise indicated, this thesis is my own work".

Signature [redacted] **Date** 13 Nov. 2011

ABSTRACT

Semantics Oriented Spatial Temporal Data Mining for Water Resource Decision Support

Guangyan Huang

Doctor of Philosophy in Computer Science

Victoria University, Australia

Water resource management is becoming more complex and relies heavily on computer software processing to help data queries for common and rare patterns for analyzing critical water events. For example, it is vital for decision makers to know if certain types of water quality problems are isolated (e.g. rare) or ubiquitous (e.g. common) and whether the conditions are changing spatially or temporally for a proper management plan. Already known spatiotemporal water quality data analysis methods are generally based on statistical techniques and spatiotemporal patterns are recognized manually or semi-manually; thus they cannot handle a large number of water data efficiently, automatically and in detail. Besides, state-of-the-art spatiotemporal data mining algorithms cannot directly satisfy mining water patterns efficiently and accurately due to uncertainty and heterogeneity problems in water quality datasets.

This thesis aims to automatically detect spatiotemporal common and rare patterns by significantly addressing the uncertainty and heterogeneity in water quality data, in order to enhance the accuracy and efficiency of common and rare pattern mining models underpinning many of the water resource management strategies and planning decisions. Therefore, we propose two novel semantics-oriented mining methods: the Correcting Imprecise Readings and Compressing Excrescent Points (CIRCE) method and the Exceptional Object Analysis for Finding Rare Environmental Events (EOAFREE) method. The CIRCE method resolves uncertainty problems in retrieving common patterns based on spatiotemporal semantic points, such as inflexions. The EOAFREE method tackles the heterogeneity problem by summarizing raw water data into a water quality index, that is, water semantics, in discovering rare patterns. We demonstrate the efficiency and effectiveness of the two methods by using simulation and real world datasets, and then implement them in a Semantics-Oriented Mining Application for Detecting Water Quality Events (SOMAwater) prototype system, which is used to query spatiotemporal common and rare patterns for a real world water quality dataset of 93 sites in 10 river basins in Victoria, Australia from 1975 to 2010.

ACKNOWLEDGMENTS

I would like to thank my principal supervisor, Professor Yanchun Zhang, for fully supporting me throughout the course of my doctoral program at Victoria University and for patiently guiding, pushing and encouraging me to focus on conducting high level research. At the same time, I would like to thank my associate supervisor, Dr. Jing He, who is also been very supportive. Her leadership in project work is proved to be one of my best experiences at Victoria University.

I would like to acknowledge the support of the ARC Linkage Project "Data Enhancement, Integration and Access Services for Smarter, Collaborative and Adaptive Whole-of-Water-Cycle Management" that provided a scholarship for my doctoral program.

I would like to thank Associate Professor Xun Yi, who gave me constructive advice about my candidature. I would like to thank Professor Yuan Miao, who discussed stream data processing methods with me. I would like to thank Associate Professor Hao Shi, who gave me encouragement at the beginning of the candidature. I would like to thank Dr. Guandong Xu. I would like to thank Dr. Jiangang Ma and Dr. Yanan Hao as well as all the members and visitors of the Center for Applied Informatics (CAI).

PUBLICATIONS

Refereed Journal and Conference Articles:

[1] **G. Huang**, Y. Zhang, J. He and J.-L. Cao, Fault Tolerance in Data Gathering Wireless Sensor Networks, *The Computer Journal*, 54(6), pp. 976-987, 2011.(**ERA A***).

[2] **G. Huang**, Y. Zhang, J. He and Z. Ding, Efficiently Retrieving Longest Common Route Patterns of Moving Objects by Summarizing Turning Regions, *Proc. of PAKDD2011* (Part I, LNAI 6634), ShenZhen, China, 24-27 May, 2011, pp.375-386, Springer, Heidelberg.(**ERA A**).

[3] J. He, Y. Zhang, **G. Huang** and J.-L. Cao, A Smart Web Service based on the Context of Things, *ACM Trans. on Internet Technology*. (Accepted in July, 2011) (**ERA A**)

[4] J. He, Y. Zhang, Y. Shi and **G. Huang**, Domain-Driven Classification Based on Multiple Criteria and Multiple Constraint-Level Programming for Intelligent Credit Scoring. *IEEE Trans. Knowl. Data Eng.* 22(6): 826-838, 2010.(**ERA A**)

[5] J. He, Y. Zhang, **G. Huang** and C. Pang, A Novel Time Computation Model Based on Algorithm Complexity for Data Intensive Scientific Workflow Design and Scheduling, *Concurrency and Computation: Practice and Experience*, 21(16): 2070-2083, 2009.(**ERA A**)

[6] J. He, Y. Zhang, **G. Huang** and J. Cao, Exceptional Object Analysis for Finding Rare Environmental Events from Water Quality Datasets, *Neurocomputing Journal*. (Accepted in April, 2011) (**ERA B**)

[7] Z. Ding and **G. Huang**, Real-Time Traffic Flow Statistical Analysis Based on Network-Constrained Moving Object Trajectories, *Proc. of the 20th International Conference on Database and Expert Systems Applications (DEXA'09)*, pp. 173-183, Linz, Austria, 31 Aug. - 4 Sep., 2009.(**ERA B**)

Submitted Articles:

[1] J. He, Y. Zhang and **G. Huang**, CIRCE: Correcting Imprecise Readings and Compressing Excrescent Points for Querying Common Patterns in Uncertain Sensor Streams, *Information System*. (The first revision submitted in Oct. 2011) (**ERA A***)

Contents

List of Figures

List of Tables

Chapter I

Introduction

"Good tools are prerequisite to the successful execution of a job."

–Confucius (*Analects of Confucius*)

Water resource management is becoming more complex and relies heavily on computer software processing to help data queries (e.g. queries of common and rare patterns for analyzing critical water events). For example, it is vital for decision makers to know if certain types of water quality problems are isolated (e.g. rare) or ubiquitous (e.g. common) and whether the conditions are changing spatially or temporally is essential for a proper management plan. Already known spatiotemporal water quality data analysis methods are generally based on statistical techniques and spatiotemporal patterns are recognized manually or semi-manually; thus, they cannot handle a large number of water data efficiently and in detail. State-of-the-art spatiotemporal data mining algorithms cannot directly satisfy mining water quality data efficiently and accurately due to uncertainty and heterogeneity problems in real world water quality datasets.

This dissertation provides semantics-oriented mining methods for automatically detecting spatiotemporal common patterns and rare patterns from water datasets, which address critical issues associated with improving the quality and completeness of the water data and negotiate the uncertainty and heterogeneity of water data in order to enhance the accuracy and efficiency of data processing models (e.g. common and rare pattern mining) underpinning many of the water resource management strategies and planning decisions.

I.1 Background and Motivations

Water is the most important catalyst for human development [Mat02]. Rivers, as a typical type of water resource, are the prime factors controlling the global water cycle. **River Water Pollution** due to urbanization [VV01] [KLL08] [SLJ+02] [KLK+07] [Pra05] has become a critical issue that must be mitigated to provide the suitability of water to sustain various uses or processes (e.g. drinking, irrigation, industry etc.). Water quality can be defined as a range of variables [Pra05] related to certain levels of physical, chemical or biological characteristics of water. Water quality differs by location (spatial factor) and season (temporal factor) [Pra05].

With the advanced tools (e.g. sensors [GQZe08] [CJ08] [VM10] and Geographical Information Systems (GIS) [KJHK] [GMFC02] [MGR05]), frameworks [ATL+05] [TNGA09] [SR08] and protocols [U.S11] for continuously and closely monitoring the environmental parameters related to water resources, we can capture abundant physical chemical water

quality data (such as PH, temperature, dissolved oxygen, total phosphate, nitrates, turbidity, total dissolved solids etc.) to analyze the spatial and temporal variation in river water quality [RA09]. However, water resource management will be more complex in the future worldwide and relies heavily on computer software processing [Mat02], since the decision makers may not be engineers or water resource domain experts and must be fed the right information (or useful knowledge)[Mat02] through data queries (e.g. queries of common or rare patterns for analyzing critical water events). For example, knowing if certain types of water quality problems are isolated (e.g. rare) or ubiquitous (e.g. common) [KLL08] and whether the conditions are changing spatially or temporally is essential for a proper management plan [KLL08] [Cha08]. A response to these queries faces three challenges:

- accuracy challenge: the discrete sampled water data, which varies widely in completeness, quality, scale, scope, reliability and metadata quality due to the many independently developed data sources and models [eW05] decreases the accuracy of data queries.

- efficiency challenge: a large number of excrescent data exist in the continuous generated water data, which waste storage and reduce the efficiency of data queries.

- integrating heterogeneous raw data challenge: historical water data are provided by different organizations and collected by different equipment over a long historical period where the collecting technologies vary.

Therefore, this dissertation aims to significantly address critical issues associated with improving the quality and completeness of water data and negotiating the uncertainty and heterogeneity of water data in order to enhance the accuracy and efficiency of data processing models (e.g. common and rare pattern mining for detecting critical water events) underpinning many of the water resource management strategies and planning decisions.

I.2 Challenges in Discovering Common and Rare Patterns from Spatiotemporal Water Data

Data sources include sensors, manually captured monitoring data, legacy databases and derived/predicted data (from models). It includes point and non-point sources, numerical data, temporal and spatial data and remote sensing satellite images. The data varies widely in completeness, quality, scale, scope, reliability and metadata quality. Continuous sensor sampled water data or a long history of manually captured monitoring water data are often recorded as a series of discrete points in databases [GBEe00], where useful knowledge can be achieved through queries. We define water data as follows:

DEFINITION 1. ***Water data*** *is a kind of spatiotemporal data that is sampled at chosen places to provide water-related physical parameters (e.g., PH, temperature, dissolved oxygen, total phosphate, nitrates, turbidity, total dissolved solids etc.) that change with time for a part or a whole of water bodies (e.g. rivers, lakes, wetlands etc.). Generally, we denote water data as a 4-tuple: (location, time, water parameter type, water parameter value).*

One of the greatest challenges facing querying useful knowledge from water data is rapid, seamless integration of the many independently developed data sources and models [eW05]. The current approach involves manual data mapping and tuning by domain experts, which is extremely tedious, time-consuming, non-scalable, inflexible and a bottleneck as the complexity, size and scope of the data and models grow. The aim of this dissertation is to improve the speed, rigour and adaptability of water management decisions - by focussing on services that will improve the quality, completeness, relevance and interpretability of the data being used in the models, such as data mining models, underpinning many of these decisions [AGBT06].

To support efficient and accurate queries on water data, we must resolve uncertainty problems when retrieving common patterns (e.g. Longest Common Curves (LCC)) and tackle heterogeneity problems for discovering rare patterns (e.g. water pollution). A combination of approaches will be used to improve data quality and completeness. Although advanced ontology-based semantic mapping and syntactic and structural mapping techniques can be employed to enable integration of disparate data sources [PZHZ07] [oI07] [SZZ06], we prefer to resolving uncertainty and heterogeneity problems automatically. It will also facilitate improved coordination of activities and decisions through faster, easier exchange of data and information between the current agencies involved in water management.

I.2.1 Uncertainty Problems in Retrieving Common Patterns from Spatiotemporal Data

We focus on uncertainty problems and other data quality problems, such as significant errors in datasets, are beyond the discussion range of this thesis. Even if every discrete point is correct, the discrete points on the time series are uncertain- that is, it is not exactly like a continuous stream since some critical points are missing due to the limited capabilities of sensing equipment and database servers [MdB04]. On one hand, a large amount of redundant information exists and thus storage resources are wasted, which also reduces the efficiency of the user query. On the other hand, missing critical points make the user query inaccurate. We call this **Uncertainty due to Discrete Sampling** (DS Uncertainty). Another uncertainty problem is that sensor readings of the same situation cannot be repeated exactly when we record them at different times or use different sensors, since different sampling errors exist; we call it **Uncertainty due to Sampling Error** (SE Uncertainty). For example, we cannot record the highest temperature of each day by sampling the temperature at a fixed time tick, since the peak values of different days are reached at different times; though the difference may be very small in two consecutive days.

Uncertainty due to Discrete Sampling and Uncertainty due to Sampling Error in the water data decrease both the efficiency and accuracy of querying common patterns. For example, we must determine whether two values are the same by tolerating their difference in an error bound instead of exactly matching with each other.

I.2.2 Heterogeneity Problem in Discovering Rare Patterns

Detecting when and where water quality events, such as pollution events, happen from spatiotemporal data faces heterogeneity problems. The first type of heterogeneity problem relates to heterogeneous raw water data with various data qualities; that is, historical water data are provided by different organizations and collected by different equipment over a long historical period where the collection technologies vary. For example, the collected water parameters are different from site to site; and there is a different sampling frequency for different sites (or different months).

The second type of heterogeneity problem is that it is difficult to detect rare patterns from data with different water quality value ranges by using statistical analysis. For example, we cannot find rare events (e.g., water pollution) directly by using a specific threshold (e.g., "poor" water quality of the river). Instead, spatiotemporal variations of the water data are more useful. Also, we can 'learn' some abstract rules from the historical data but cannot directly achieve normal values as the threshold for exception analysis. This means statistical analysis is invalid in detecting exceptional objects from the water quality data, which we explain as follows. Rare environmental events are generally unusual, relative to the normal patterns of behavior of an environmental body (e.g., a river) [KJBB09].

The simplest and the most straightforward approach to detect rare events is to explore exception analysis which identifies whether an attribute or measure value belongs to or does not belong to a specific list of values. One limitation of this approach is that it requires

knowledge of the normal value or what is anomalous [MSN]. Although we can "learn" the normal values from historical data and then detect events that indicate departures from the norm [KJBB09], the learnt knowledge may be out-of-date, since the environmental situation is changing with time. For example, it is unreasonable to use the average value of several sampling locations to denote the water quality of a whole river. Another limitation of the straightforward approach is that it is only valid when rare environmental events directly lead to an abnormal value. But it is invalid when rare environmental events produce a normal value since the whole environmental system (e.g. a river water system) can bear a pollution event for an extended period due to the following factors.

- First, daily water flow varies greatly in different seasons or in different rivers. A pollution event may not instantly change the water quality of the whole river that has large amount of water flow in season, since the pollution may be flushed away; although a pollution event may persist for a long time and reduce the river water quality eventually.

- Second, different rivers have a various range of water quality from "excellent" to "very poor", where water quality is higher, the pollution event is harder to detect. For example, if the water quality of a river is "excellent", it may take a long time for a pollution event to change the water quality into a "poor" state. This means we are not aware of the harm of this pollution event from the beginning, for example, when a factory

drains waste water into a nearby river for an extended period of time.

I.3 Solutions: Semantics-Oriented Mining Methods for Analysis of Water Quality Events

We use data mining methods to discover semantic points, such as inflexions detected from spatiotemporal curves, to remove uncertainties; then we simplify the spatiotemporal curves by using semantic place IDs and change the problem of mining uncertain spatiotemporal curves into mining common patterns from certain ID sequences. Note that semantic points also can help find missing points on the spatiotemporal curves.

Also, we take advantage of the water quality index, water semantics defined by domain experts, to unify heterogenous water data curves. Then we define a new concept of water quality changes to remove the geographical differences between any two sites and the temporal differences between any two years and thus we can discover common or rare patterns among different data sequences at different sites for different years.

By using these semantics (e.g., summarized semantic points on the curves and water quality index semantics), we develop three methods: MicPasts and CIRCE for tackling the uncertainty problems in the process of retrieving common patterns, and EOAFREE for resolving the heterogeneity problems in the processing of detecting rare events in the water quality data curves. Finally, we implement these three methods into an SOMAwater prototype system for the application of detecting water quality common and rare events.

I.3.1 Mining Common Patterns based on Spatio-Temporal Semantics (MicPasts)

In this dissertation, we propose a novel Mining Common Patterns based on spatiotemporal Semantics (MicPasts) method to tackle the problem of Uncertainty due to Sampling Error (SE Uncertainty). To resolve the SE Uncertainty problem, the MicPasts method summarizes the original data streams by using inflexions, then discovers semantic places on the spatiotemporal curves by grouping close inflexions into the same cluster, and then changes the uncertain data streams into exact sequences of cluster IDs. Thus, the problem of mining common patterns is mapped into the traditional problem of mining sequential patterns. Particularly, to help query common patterns directly on exact ID sequences, a Discovering Implicit Semantic Places (DISP) procedure is developed to discover the implicit semantic regions.

I.3.2 Correcting Imprecise Readings and Compressing Excrescent Points (CIRCE)

We then extend the MicPasts method to a novel CIRCE method by providing a novel Correcting Imprecise Readings and Compressing Excrescent points (CIRCE) core algorithm to tackle the Uncertainty due to Discrete Sampling problem. The CIRCE core algorithm aims to find inflexions from a single incomplete data stream. It uses a Detecting Inflexions and Computing Missing Inflexions (DICMI) algorithm for detecting local inflexions including missing inflexions and then compressing an original sensor stream by a sequence of

inflexions. Also, a novel Angle-DP algorithm is developed for finding critical points from inflexions and then for compressing an inflexion sequence generated by the DICMI algorithm further into a critical point sequence.

I.3.3 Mining Rare Patterns based on Water Quality Semantics

Critical events are generally rare events. There are two classes of rare event detection methods: application-specific and general rare event detections. However, already known application-specific rare event detection methods may not be suitable for detecting rare events from water quality data. Besides, although data mining methods, such as clustering algorithms, which do not force every data instance to belong to a cluster can be used to generate some data instances that could not be grouped into any cluster as rare events, the disadvantage of such techniques is that they are not optimized to find rare patterns, since the main aim of the underlying clustering algorithm is to find clusters.

We develop a novel Exceptional Object Analysis for Finding Rare Environmental Events (EOAFREE) method. Particularly, a general Improved Exceptional Object Analysis based on Noises (IEOAN) algorithm is created to distinguish those data objects (or data points) that cannot be grouped into any clusters as exceptional objects. Interestingly, opposite to the already known Principal Component Analysis (PCA) that ranks principal components, our IEOAN ranks exceptional objects. The EOAFREE method preprocesses heterogeneous spatiotemporal data through exploring water semantics and defines spatiotemporal changes

instead of the water quality value itself as the input of the IEOAN algorithm to remove the geographical differences between any two sites and the temporal differences between any two years.

I.3.4 Application: Semantics-Oriented Mining Application for Detecting Water Quality Events (SOMAwater) Prototype System

Finally, we implement the CIRCE method (which includes the MicPasts method for resolving Uncertainty due to Sampling Error problem and the CIRCE core algorithm for tackling Uncertainty due to Discrete Sampling problem) and the EOAFREE method into a SOMAwater prototype system to query longest common curve (LCC) patterns and rare patterns to support decision making for the water resources management domain. We illustrate querying LCC patterns and several rare patterns (e.g., continual, simultaneous, seasonal and global rare patterns) from water quality data by using extensive examples. We use a massive experimental study to demonstrate the CIRCE method for querying LCC patterns and to demonstrate the EOAFREE method by querying spatiotemporal rare patterns.

I.4 Contributions of This Dissertation

The findings of this work include three aspects. First, our CIRCE method [HZHa] which includes both the MicPasts method (also called LCRTurning in our paper in [HZHD11]) and the CIRCE core algorithm, as well as our IEOAN algorithm [HZHb], are general solutions proposed for mining common and rare patterns from spatiotemporal data. The MicPasts

method discovers semantic regions by summarizing inflexions. The CIRCE core algorithm is developed for improving the quality and compressing the volume of spatiotemporal data. The IEOAN algorithm is proposed for discovering and ranking exceptional objects in a spatiotemporal database. Second, the EOAFREE method [HZHb] is developed especially for mining water quality data in order to detect critical water quality events. Third, we implement the CIRCE method and the EOAFREE method in a Semantics-Oriented Mining Application for Water Pollution Event Detection (SOMAwater) prototype system to discover common patterns, especially Longest Common Curves (LCC) and rare patterns (e.g. continual, simultaneous, seasonal and global rare patterns) from water quality data, and to support decision making for effective water management. Compared to manual and semi-manual spatiotemporal pattern analysis, SOMAwater can help decision makers to focus only on the valuable datasets and discover spatiotemporal water patterns efficiently and automatically, as well as in more detail.

I.5 Dissertation Structure

The rest of this dissertation is concerned with mining common and rare patterns based on semantics in water quality data for detecting water quality events. The relations between the major chapters are shown in Fig. I.1. In Chapter II, we introduce preliminary knowledge and compare our methods with related work. In Chapter III, we present a novel MicPasts method. In Chapter IV, we extend the MicPasts method to a novel CIRCE method by developing

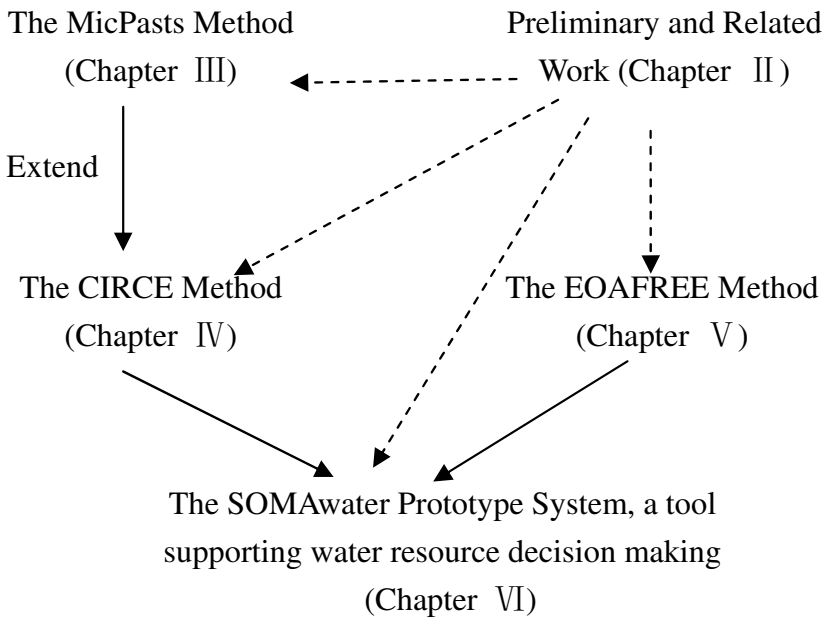


Figure I.1: Dissertation Structure.

a CIRCE core algorithm especially for improving the quality of a single stream data. In Chapter V, we present a novel EOAFREE method. In Chapter VI, we implement the CIRCE method (including the MicPasts method and the CIRCE core algorithm) and the EOAFREE method (including the IEOAN algorithm) into a SOMAwater prototype system for water resource decision support by detecting longest common curves and rare events (e.g., water pollution and positive factor for water quality improvement) from water quality data. In Chapter VII, we summarize the work of this dissertation and mention some directions for future research.

Chapter II

Preliminary and Related Work

"If you know your counterparts and know yourself, you need not fear the result of a hundred battles."

–Sun Tzu (*The Art of War*)

In this chapter, we first introduce preliminary work in spatiotemporal data mining, then we present state-of-the-art related work from three aspects: mining spatiotemporal common and rare patterns, as well as the data analysis method to support decision making in water resource management domain. Meanwhile, we compare our CIRCE method, EOAFREE method and SOMAwater prototype system with the related work to point out our contribution.

II.1 Preliminary Work in Spatio-Temporal Data Mining

It has been estimated that 80% of the available datasets have spatial components [FG01] and are often related to temporal aspects [EMT08]. Mining spatio-temporal data is complex

in terms of both the mining efficiency and the complexity of patterns that can be extracted from spatio-temporal datasets [RL01] [Mit10]. Water quality data are typical spatiotemporal data streams denoted by time series curves. In this section, we introduce basic techniques that are widely used in mining general spatiotemporal datasets as preliminary knowledge.

These techniques are also adopted as a basis to develop our new techniques for mining spatiotemporal water quality datasets. First, enormous volumes of spatiotemporal stream data bring a great challenge for storage, transmission and computation and thus compression techniques for water data curves are vital. Many critical points detection and data compression methods for line simplification are widely accepted and used within digital cartography. Of them, the Douglas-Peucker (DP) algorithm [DP73] is mathematically and perceptually superior [VW91]. In this dissertation, we modify and improve the DP algorithm to make it suitable for compressing water data curves, considering that some critical points may be missing in the real world application. We show that water data curves can be simplified by the same methods used for simplifying spatial curves, particularly as the time series based water data curves have no self-intersections and thus satisfy the criteria of many line simplification methods well. Also, we discover common patterns using clustering methods and detect rare patterns by using opposite methods, that is, to find exceptional objects that cannot be grouped into any clusters. The Density Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is an interesting clustering algorithm that can recognize both common patterns (e.g. clusters) and rare patterns (e.g. noises). Finally, after compressing

and clustering, the water data streams become concise, without any uncertainty. We then change the problem of retrieving common curves into the problem of finding exact common subsequence patterns that can be resolved by already known mature techniques.

Therefore, to help understand the content in this dissertation, in this section, we present preliminary knowledge about the Douglas-Peucker line-simplification algorithm, the DBSCAN clustering algorithm and the mining algorithm for common subsequence patterns.

II.1.1 Douglas-Peucker Line-Simplification Algorithm

Given a continuous curve denoted by a function, we can compute all the inflexions through derivations. Given a discrete curve denoted by a sequence of points, we can use the DP algorithm to find the inflexions (or the turning points). The main idea of the DP algorithm in [Whi85] is given as follows.

- *Step 1*: The first point and the last point of one trajectory are chosen as the anchor point and float point, respectively.

- *Step 2*: For intermediate points, the cut point is the point with the maximum perpendicular distance, which is greater than a pre-defined threshold, λ, to the line connecting anchor and float points.

- *Step 3*: Then the cut point becomes the new float point for the first segment and the anchor point for the second segment. If no cut point exists, it stops.

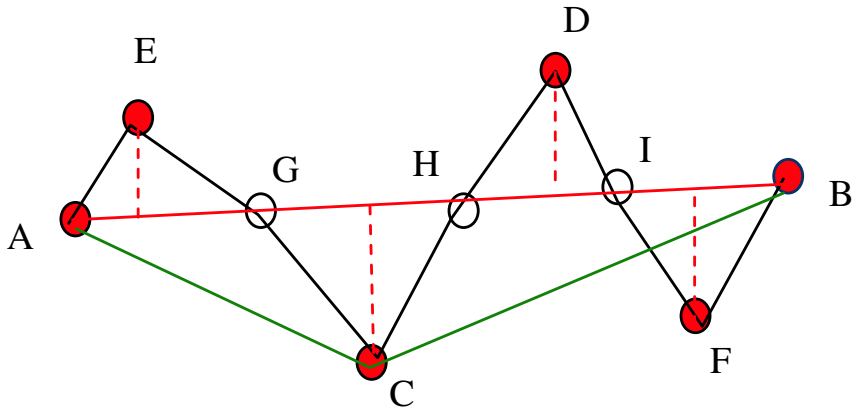


Figure II.1: An example of the DP algorithm.

- *Step 4*: This procedure is recursively repeated for both segments.

The complexity of the most efficient implementation of DP is $O(NlogN)$, where N is the number of points on a single curve provided in [HS92].

An example discrete curve "AEGCHDIFB" is shown in Fig. II.1. First, 'A' and 'B' are the anchor point and float point, respectively. Then 'C' is a cut point. For 'AC', 'E' is a cut point. For 'CB', D is a cut point. Finally, for 'DB', 'F' is a cut point. All the critical points detected by the DP algorithm are denoted by dots and other points denoted by circles can be omitted.

II.1.2 DBSCAN Clustering Algorithm

The DBSCAN algorithm is designed to discover clusters of arbitrary shape as well as to distinguish noise [SEKX98]. The original DBSCAN (eps, $MinPts$) algorithm is provided in [EKSX96]. We summarize an implementation version of DBSCAN according to our work

in [HHD07] [HHZS07] [HHD08a] as follows:

- *Step 1*. Build neighbour lists of each object. The neighbours of object q must satisfy criteria that neighbours are in the neighbourhood circle area with location of q as the centre and eps as the radius.

- *Step 2*. Build a set of core objects, I. The object, which has more than $MinPts$ neighbors, is marked as a core object.

- *Step 3*. All core objects are marked unused. For each unused core object p, put p and p's neighbors into cluster $class\_id$ and mark the object as being used. Any core object r in cluster $class\_id$ will recruit r's neighbors into cluster $class\_id$ and the used objects are marked.

- *Step 4*. Noise objects are those that are not used and are assigned into a special noise cluster.

The complexity of DBSCAN is $O(NlogN)$, where N is the number of points being clustered. If the points are organized as a grid, the complexity of DBSCAN is $O(N)$ [SEKX98]. Two parameters: eps and $MinPts$ control the clustering output according to applications. Given the data structure in Fig. II.2 [HHD08a], the time spent by DBSCAN is $T(N, Eps) = N(\pi eps^2 + qsort(\pi eps^2))$, where $qsort(\cdot)$ denotes the time for running $qsort$ function to quick sort the maximum number of points in the neighbor lists.

struct _Neighbor	struct _DataPoint
{	{ bool core_tag;
float distance;	int class_id;
int ID;	bool used_tag;
};	int x;
	int y;
	int ID;
	int numNeighbor;
	_Neighbor NeighborList[π *Eps*^{2}]; // π =3.14
	} DataPoints[*n*];

Figure II.2: Data Structure of DBSCAN Algorithm.

An example is shown in Fig. II.3. Suppose $MinPts = 4$. Start at core object, q, p is a neighbor of q, r is a neighbor of the core object, p, and thus r is also added into q's cluster (e.g. r and q are density-connected). Object r's neighbor, t, cannot be added into q's cluster, since r is not a core object.

II.1.3 Mining Common Subsequences

A suffix tree is an efficient data structure for retrieving common sequences from multi sequences [McC76]. We firstly introduce the suffix tree as follows. If $S =< t_1, t_2, ...t_i, ... t_n >$ is a string, then $T_i =< t_i, t_{i+1}, ... t_n >$ is the suffix of S that starts at position i.

The suffix tree for a string, S, with length n is defined as a tree such that (in [Dan97]):

- the paths from the root to the leaves have a one-to-one relationship with the suffixes of S,

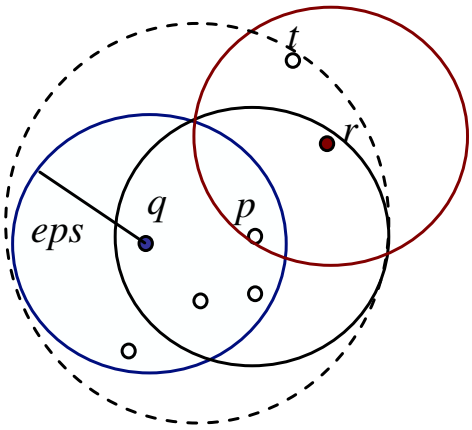


Figure II.3: Density-connection concept in DBSCAN.

- edges spell non-empty strings,
- and all internal nodes (except perhaps the root) have at least two children.

An example of two sequences: "ABCD" and "BCDE" is shown in Fig. II.4. If min_sup=2, then "BCD" and "CD" are common sequences. We also build a list of supports on the nodes of suffix tree. The algorithm retrieving all common sequences based on the suffix tree is given in Fig. II.5.

II.2 Spatio-Temporal Data Mining for Common Patterns

In this section, we present a survey of mining spatiotemporal common patterns. Uncertainty due to Discrete Sampling and Uncertainty due to Sampling Errors are major challenges in the process of mining common patterns from spatiotemporal datasets. To the best of our

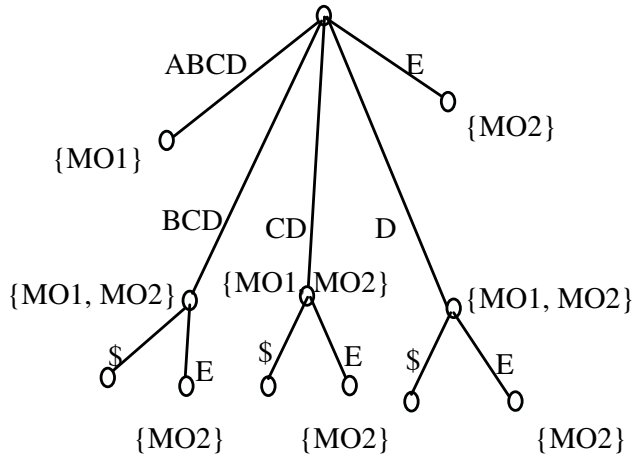


Figure II.4: Suffix tree for retrieving all Common Sequences from multiple sequences.

Algorithm 1. Mining Common Sequences.

Procedure SuffixTree_CS (*STree*)
1. *flag*=0;
2. **While** (*STree*!=**Null**)
3. **If** ($N_{MO} \geq min\_sup$) **then /***N_{MO} is the number of moving objects on *STree*.*/
   flag=1;
 3a. **If** *STree* is leaf **then**
   CSset ← String from root to *STree*;
 3b.**ElseIf** (!**SuffixTree_ CS** (*STree.firstChild*)) **then**
   CSset ← String from root to *STree*;
   Endif
4. *STree*=*STree.next*;
   EndWhile
5. **Return** *flag*;

Figure II.5: Retrieving common subsequences.

knowledge, no related work resolves the Uncertainty due to Discrete Sampling problem (e.g., correcting the missing readings) while compressing. So, we only present the already known work that tackles Uncertainty due to Sampling Errors. We classify the solutions to overcome Uncertainty due to Sampling Errors into two classes: non-semantic-based methods and semantic-based methods for querying common pattern on spatiotemporal data streams,

One non-semantic-based method is to determine whether two points are the same or not by tolerating a bounded error. Then, we can determine whether two trajectories are the same or not by checking whether the spatial difference at every timestamp is in the bounded error [JYZ+08] [CMC05]. Given two trajectories, it is easy to retrieve LCR patterns with $min\_sup = 2$ using the non-semantic-based method. But given n trajectories, how to efficiently retrieve LCR patterns with $min\_sup = m$? A MC2 (clustering moving cluster) algorithm in [KMB05] provides a solution, that is, to cluster points at every timestamp, taking the points in the same cluster as in a bounded error. This actually changes the problem of clustering 3-D points (2-D location, time) of trajectories into the problem of clustering 2-D locations. The MC2 algorithm checks the number of common moving objects between any two clusters in two consecutive timestamps. Given two cluster sets: $c1$ (at $T1$) and $c2$ (at $T2$), if $|c1 \cap c2| > k$, then the common objects moving together between $[T1, T2]$ are a moving cluster. However, the MC2 algorithm runs at every timestamp and thus is time-consuming. A CuTS (Convoy Discovery using Trajectory Simplification) algorithm has been proposed in [JYZ+08] to improve the MC2 algorithm for retrieving convoy patterns.

First, CuTS retrieves coarse candidate results by querying on Douglas-Peucker (DP) [DP73] compressed trajectories; that is to cluster direct line segments between two critical points other than clustering location points at every timestamp. The coarse searching screens a large volume of points to be checked further and thus increases the efficiency. Then, CuTS checks the points on the candidate results by using MC2 to ensure high accuracy. To achieve the longest convoy, CuTS connects the sub-patterns on consecutive direct line segments together. However, due to missing critical inflexions of original data streams, one long patterns may be split into two sub-patterns and thus CuTS is not suitable in retrieving the longest patterns from data streams with Uncertainty due to Discrete Sampling problem.

Another non-semantic-based method is to determine if two polygon line segments are the same or not by testing whether they are in the same bounded rectangle region. Then, a common trajectory is a sequence of common polygon line segments. In [CMC05], first, a trajectory is split into a sequence of pieces based on the DP algorithm [DP73] and each piece is compressed by direct line segments to reduce the enormous volume of data [MdB04] [Tha89]. Then it groups similar direct line segments on DP-based trajectories into clusters. Suppose L_1 and L_2 are two direct line segments that have been grouped into the same cluster. Different from [JYZ$^+$08], the method in [CMC05] checks whether the original polygon line segments of L_1 comply with L_2, if so, L_1 is defined as close to L_2. Then a mean line segment of a cluster, $\overrightarrow{L}$, as a central line segment is used to define a rectangle region with a width threshold. Only if L_1 and L_2 are not only similar but also close to $\overrightarrow{L}$, they are supporters of

this region. If the number of supporters is greater than a threshold min_sup, the rectangle region is a frequent one. Given IDs to rectangle regions to simplify trajectories further, the problem is mapped into the problem of mining Sequential PAttern (SPA). However, clustering direct line segments is complex and inefficient as our experiments in Chapter III will demonstrate.

In the semantic-based methods in [GNPP07], it inputs some already known regions of interest (RoI) to determine whether different locations are at the same RoI. There are three types of RoIs introduced in [GNPP07]: candidate places such as restaurants, shops etc; popular square regions visited by at least 10% of the objects; and crossroads where more than 50% of objects change their direction. Since the predefined RoIs are limited, [GNPP07] dynamically defines RoIs by popular region, a rectangular region with a certain density of location instances. A popular region in [GNPP07] is discovered by processing the locations in the whole dataset. However, the definition of popular region by the certain density of accumulated locations cannot be discovered efficiently if the trajectories are sampled by regular timestamps, not only because the volume of location datasets is huge but also because large numbers of meaningless locations may hinder finding the real popular regions.

Furthermore, retrieving patterns efficiently from trajectories based on trajectory simplification (e.g. the DP algorithm), as was done in [JYZ+08] and [CMC05], brings a big challenge for accuracy, because the DP algorithm cannot correct the imprecise inflexions. Accuracy is often measured by two metrics: false-positive rate (the percentage of incorrect

patterns wrongly identified as positive) and false-negative rate (the percentage of correct patterns that have not be retrieved). False-positive results often lead to an exponential explosion of LCR pattern mining and thus reduce the mining efficiency [YCL+06]. The number of false-positive results can be reduced by post-checking such as in [JYZ+08] and [GNPP07]. However, evaluating false-negative rates is more difficult in many applications and in related work, such as those in [JYZ+08], [GKS07], [GNPP07] and [CMC05], no evaluation is undertaken to measure false-negative rates.

In order to retrieve LCR patterns efficiently, our MicPasts method, also called LCR-Turning algorithm published in [HZHD11], adopts a semantic-based method and uses the semantic points (or turning points) to simplify the original trajectories. If the split points are in the same cluster, then they are denoted by the same cluster ID. Thus, we can map the problem of mining LCR patterns from uncertain data streams mainly into the problem of mining Longest Common Sequences (LCS) [Mai78] [BHR00] [Gre03] [MP80] [SA96] [HS77] from exact region IDs. Also, we only cluster direct line segments without further validation on the original trajectories since a sequence of semantic turning region IDs abstract the trajectories with high precision.

The CIRCE method extends the MicPasts method and improves the accuracy of querying LCR patterns further by providing a novel CIRCE core algorithm for resolving the Uncertainty due to Sampling Errors problem. Although it is mathematically and perceptually superior for compressing curves, the DP algorithm cannot achieve minimal number of ver-

tices [LZ11]. Thus, the CIRCE core algorithm (including Angle-DP) outperforms the DP algorithm, since it corrects missing inflexions to resolve Uncertainty due to Discrete Sampling problem while summarizing the curves into less critical points by developing a new Angle-DP. Therefore, we achieve more accurate but less semantic points to compress data streams. Moreover, our CIRCE method makes querying common patterns directly on compressed data available by developing a DISP procedure to discover implicit semantic places from compressed data.

We also present a survey of applying spatiotemporal data mining to water data. In [PCE08], a data mining method is used to generate a classification model based on association rules to classify water data. In [SWMW09], k-means clustering and graph partitioning algorithms are used to discover patterns or connections among different variables or different spatial locations within a data field. In [PAN05], Principal Component Analysis (PCA) together with variance and time series analysis is used to study inter-correlation between pollutants and ions. Our CIRCE method (or MicPasts) focuses on discovering longest common curve patterns that are used to group similar locations and seasons of the water quality data into clusters to assist decision making and thus we are interested in spatiotemporal common patterns instead of other knowledge, such as relation rules between different water variables.

II.3 Spatio-Temporal Data Analysis for Rare Event Detections

We classify existing algorithms for rare event detection from spatiotemporal data into two classes: application-specific and general rare event detection.

Most existing rare event detection research either focuses on a particular application domain or on a single research area [CBK09]. In [KJBB09] and [MKB09], uncertainty quantification-based techniques are used to detect salinity events within oceanographic data, particularly, a composite of the static threshold method and dynamic uncertainty quantification-based techniques is proved to improve the event detection precision. A method to more accurately detect tornadoes is developed by using Support Vector Machines (SVMs) [TTR03]. In comparison with other detection methods, such as neural networks and radial basis function networks, SVMs are found to be more effective in tornado detection. Statistical signal processing techniques are applied to event detection in wireless sensor networks [JGS07]. Specifically, they use Principal Component Analysis (PCA) to build a model of observed environmental phenomena that captures daily and seasonal trends within the sensor measurements. The divergence between sensor measurements and model predictions is used as an indicator of discrete events within the data stream. [Kou06] uses abnormal spatial pattern recognition to locate extreme meteorological events such as water pollution incidents. These rare event detection methods that are only suitable for a specific application domain may not be suitable for detecting rare events from water quality data.

General rare event detection algorithms may be more useful. For example, several clustering algorithms, such as DBSCAN [EKSX96], ROCK [GRS00], Shared Nearest Neighbor (SNN) clustering [ESK03] and Findout algorithm [YQLZ02], that do not force every data instance to belong to a cluster generate some data instances that could not be grouped into any cluster. These anomaly data instances are rare events. However, the disadvantage of such techniques is that they are not optimized to find rare patterns, since the main aim of the underlying clustering algorithm is to find clusters [CBK09]. Therefore, in this dissertation, we develop a general novel Improved Exceptional Object Analysis based on Noise (IEOAN) algorithm by taking advantage of DBSCAN clustering to detect rare patterns. Experimental study shows that our IEOAN algorithm is far more efficient than directly recursively using DBSCAN clustering algorithm to detect rare events. Interestingly, opposite to the already known Principal Component Analysis (PCA) that ranks principal components, our IEOAN ranks exceptional objects.

The most related work for rare event detection is a sensor data mining model in [VM10] for monitoring continuously changing statio-temporal water data. In this data mining model, two criteria, the threshold value or the probability of outstanding from the set of average value, are used to extract outlier and semantic measures are used for the data mining methods such as classification and association rule. Our EOAFREE method is different: first, we use water semantics to fuse multiple raw water data curves into one water quality index curve; then we develop an IEOAN algorithm based on DBSCAN clustering to detect exceptional

objects (the same as outlier) from water quality index data. We not only discover recurring and sequential patterns that are discovered in [VM10], also we define a seasonal pattern as a novel temporal pattern. Besides, we not only analyze spatial similarity like in [VM10], also we define spatial patterns, such as global rare patterns.

II.4 Spatiotemporal Water Quality Data Analysis to Support Decision Making

In [RA09], multivariate statistical techniques like hierarchical cluster analysis (HCA) and principle component analysis (PCA) were used to segregate and examine the water quality data. Multivariate statistics is considered to be an appropriate and efficient tool [QMH07] [SKSB06] [SMS+05]. HCA is used to study the spatial variation in water quality. GIS is used to visualise spatial data, for example, geographical information related to water quality [KKE+]. In [WXSY06], a statistics and generalized regression neural network model is proposed for analyzing detailed spatial distribution of water quality. In [KLK+07] and [KLL08], the spatial variation and temporal variation of the water quality are analyzed. However, none of these methods detect common or rare water quality patterns.

In [CSM+08], the point sources of pollution due to lack of facilities for appropriate treatment of domestic and industrial sewage are identified and detected by assessing water quality variables through manual observation of the water quality curves, such as temperature, conductivity, PH, dissolved oxygen and etc. However, manual pattern recognitions

cannot analyze a large number of stream data efficiently. In this thesis, we adopt Water Quality Index (WQI), a standard measurement used by domain experts in the water quality management field, to detect water quality changes. Our aim is to develop efficient methods to automatically detect common and rare patterns. We can easily validate the effectiveness of the experimental results by manually checking WQI change.

In [EGBD06], Multiple Criteria Decision Analysis (MCDA) is used in water quality management, where the results of action for the sustainable improvement of water quality are evaluated. It used standard levels of water quality parameters to define economic and ecological criteria for decision making. Our SOMAwater prototype system can assist in defining more detailed standard levels of water quality parameters by considering spatiotemporal differences and thus improve the efficiency to replace the manual comparison of the spatiotemporal differences by experts in water quality field.

In [HFK08], knowledge-based expert systems, such as rule-based reasoning and case-based reasoning systems are incorporated to assist decision making. For instance, a rule-based reasoning system is used to select appropriate data and a case-based reasoning system uses knowledge gained through similar cases. Our SOMAwater prototype system can help experts to choose the right datasets and to discover common patterns of similar cases automatically and thus release experts from manual or semi-manual analysis.

In [GMFC02], spatial decision-making based on the Geographic Information System (GIS) are used in the water management domain to manage the complex decision prob-

lems emerging due to the increasing complexity of sustainable spatial development. Our SOMAwater prototype system can distinguish where common or rare water events happen while marking the event location on the river basin maps; this actually offers a potential means to decision makers.

In [MGR05], information technologies, such as data availability and ease of use, provide solutions for decision making systems. In this dissertation, our solutions for removing data uncertainty and data heterogeneity as well as automatically discovering high level knowledge from raw water data are more than just providing a user-friendly interface for decision makers.

In [Liu04], the Geographic Information System (GIS) is used to display the spatially and dynamically varied water data and an expert system is provided for decision making, but only statistical analysis is provided. Also, in [Cha08], spatial or temporal patterns of water quality trends of 118 sites in the Han River basin of South Korea are given; but only a statistical analysis method is used without any new knowledge discovery. Our SOMAwater prototype system can discover spatiotemporal common and rare patterns while grouping similar data into clusters; this is more useful for decision making.

Chapter III

The MicPasts Method

"Originally there is no path in this world, but when there are many who have walked upon it, then a path came into being."

–Xun Lu (*Hometown*)

In this chapter, we provide a new Mining Common Patterns based on Spatial Temporal Semantics (MicPasts) method. The MicPasts method is a general method that can discover common patterns for any curves, including spatial curves (trajectories of moving objects) and water quality data curves. Since many algorithms for mining common patterns are originally provided for spatial curves, there are more tools (e.g. simulators) for spatial trajectories, which can be used to validate the effectiveness of a new mining algorithm. Besides, spatial curves are more complex than water quality data curves. We will show in Chapter VI that the mining algorithm for spatial curves can be used for water quality data curves. Therefore, in this chapter, we present the main idea of our MicPasts method for general curves and then demonstrate its effectiveness by using spatial curves. A part of the content of this chapter

has been published in [HZHD11].

As we mentioned in Chapter I, the continuous sensor stream data or historically accumulated data for water quality are often recorded as a series of discrete points. However, sensor readings or manual readings for the same physical situation cannot be repeated exactly when we record them at different times or use different sensors since different sampling errors exist. Even though already known algorithms exist, they are not accurate and efficient. Therefore, the MicPasts method is vital to improve the accuracy and efficiency of the mining process. To help the user understand the MicPasts method, we use moving object datasets in this chapter to explain it and demonstrate its effectiveness.

The structure of this chapter is organized as follows. We present the overview of the MicPasts method in Section 1 and model the basic problem in Section 2. Then a Mining LCR patterns based on turning regions (LCRTurning) algorithm, an implemented version of the MicPasts method for trajectories of moving objects, is provided in Section 3, while the performance of the proposed algorithm is evaluated in Section 4. Finally, Section 5 summarizes this chapter.

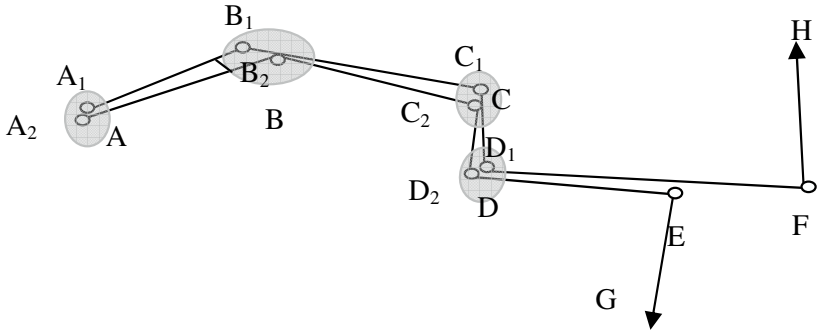
III.1 Overview of The Mining Common Patterns based on Spatiotemporal Semantics (MicPasts) Method

Continuous stream data are often recorded as a series of discrete points in a database from which knowledge can be retrieved through queries. However, querying stream curves faces

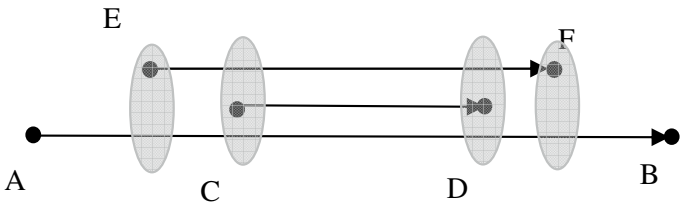
two challenges: the efficiency challenge due to a large number of useless points on stream curves and the accuracy challenge due to the fact that sensor or manual readings for the same physical situation cannot be repeated exactly when we record them at different times or use different monitoring methods since different sampling errors exist. The MicPasts method addresses the problem of mining Longest Common Curve (LCC) patterns efficiently and accurately.

For spatial curves, an LCC pattern is the longest common route (LCR) pattern. Mining LCR patterns is one of the fundamental issues in mining trajectories of moving objects. LCR patterns are the longest LSPs (Long, Sharable Patterns) [GP09]. Thus, to retrieve LCR patterns from spatial curves, the detailed challenges are: (1) the efficiency challenge: the sampling interval can be very small to ensure the accuracy of trajectories but this approach generates a large number of useless points on trajectories [JYZ+08] [GNPP07] [CMC05]; and (2) the accuracy challenge: as a trajectory of a moving object is generally represented by a sequence of discrete locations sampled with an interval, the different trajectory instances along the same route may be denoted by different sequences of points (location, timestamp) [CMC05] [JYZ+08] [GKS04] [GKS07] [LDHK11]. To tackle the above two challenges, we propose a novel mining algorithm for LCR patterns based on popular Turning regions (LCRTurning) and explain its effectiveness using examples in Fig. III.1. We classify LCR patterns into two classes: Polygon line based LCR (P-LCR) and Direct line based LCR (D-LCR), as shown in Fig. III.1 (a) and (b), respectively. We observe that the turning points,

where objects change their directions, are always critical and thus we abstract each trajectory by using a sequence of turning points. In Fig. III.1 (a), two trajectories are simplified as $A_1B_1C_1D_1FH$ and $A_2B_2C_2D_2EG$, by using turning points. Then, we group all the turning points on both simplified trajectories into different clusters based on their spatial proximity; examples are cluster 1: $\{A_1, A_2\}$, cluster 2: $\{B_1, B_2\}$, cluster 3: $\{C_1, C_2\}$ and cluster 4: $\{D_1, D_2\}$. We define a popular turning region to be an area that encloses at least min_sup (minimal number of supports) points in a cluster and assume $min\_sup = 2$; examples include regions A, B, C and D. Finally, we unify the two trajectories as two strings: $ABCDFH$ and $ABCDEG$. By the above three steps, the problem of mining LCR from trajectories is mapped into the traditional problem of mining Longest Common Subsequences (LCS) from strings. Obviously, the common string is $ABCD$ in this example. Then we refine $ABCD$ by $ABCDE$ and $ABCDE$ is a P-LCS pattern in this case. Meanwhile, D-LCR patterns are discovered by another method. For example, the moving object MO1 travels from A to B in Fig. III.1 (b), where MO1 also passes the other four regions: E, C, D and F, but these four points are not recorded in the trajectory of MO1. So, we retrieve common direct line segments to find D-LCR patterns (e.g. EF in Fig. III.1 (b)). Therefore, the LCRTurning algorithm simplifies trajectories by turning points to remove a large number of useless points and then discovers popular turning regions to simplify trajectories further; these tackle the efficiency challenge. Moreover, it tackles the accuracy challenge by unifying simplified trajectories based on popular turning regions.



(a) Polygon line based LCR pattern. .



(b)Direct line based LCR pattern.

Figure III.1: Two classes of LCR patterns.

Our LCRTurning algorithm is different from already existing algorithms and can retrieve LCR efficiently and accurately and we explain this as follows. In [CMC05], the Douglas-Peucker (DP) algorithm [DP73] is used to simplify the trajectories, which tackles the efficiency challenge. We not only validate that the DP algorithm is effective in finding turning points in a single trajectory, our proposed algorithm also considers the global popularity of a turning region to simplify trajectories further. Moreover, we discover LCR patterns mainly through clustering turning points other than mainly clustering direct line segments in [CMC05]. In [JYZ+08], the CuTS (Convoy Discovery using Trajectory Simplification) algorithm is provided, which firstly retrieves coarse candidate results based on DP-simplified trajectories and then validates candidates by clustering locations at every timestamp. The coarse searching screens a large volume of points to be checked further to increase efficiency, but it did not evaluate the accuracy of CuTS since the correct patterns were unknown. In this chapter, to evaluate both false positive rates and false negative rates of our LCRTurning algorithm, we develop a benchmark method to denote a trajectory by using the intersections and every trajectory along the same route can be detected exactly by the sequences of intersections. Note that our LCRTurning algorithm does not depend on any road network information. In [GNPP07], the popular regions visited frequently by moving objects are used to dynamically define regions of interests (RoIs) and then help discover trajectory patterns. However, the popular regions cannot be discovered precisely because the large number of meaningless locations may lead to finding false popular regions. Also, if we use

intersections as RoIs and define a neighborhood of each intersection to help remove useless non-intersection locations on sampled original trajectories to tackle the accuracy challenge, it is time consuming to retrieve common patterns since there are a huge number of intersections; our experimental results validate this. The efficiency of our LCRTuring algorithm is validated by an experimental study based on various sizes of moving objects datasets.

III.2 Problem Definition

We formally define basic terms below and then model the problem.

A model where points are sampled in a constant interval (e.g. 30 seconds once) is widely used to sample trajectories due to its simpleness. We adopt this model and define a trajectory as follows:

DEFINITION 2. A trajectory of moving object is a sequence of points $S =< p_1, ..., p_u, ..., p_k >$, where $p_u = (x_u, y_u, t_u)$, and t_u $(u = 0..k)$ is a timestamp for a snapshot, $\forall_{0 \leq u < k}$, $t_u < t_{u+1}$, $t_{u+1} - t_u = \tau$ (τ is a constant), while (x_u, y_u) is a 2-D location.

DEFINITION 3. $d_{lp}(p, AB)$ is the minimal distance between a point, p, and any point on a direct line segment, AB.

Then, we detect critical points from original points to simplify a trajectory. We take inflexions (or turning points), defined as follows, as critical points.

DEFINITION 4. A turning point (or an inflexion) is a point, $p_i = (loc_i(x_i, y_i), t_i)$, on a trajectory $S =< p_1, ..., p_u, ..., p_k >$, which satisfies one of the two conditions: (1) i=1 or k for

the start or end points of the trajectory; (2) $d_{lp}(loc_i, loc_b loc_a) > \lambda$ (λ is a distance threshold), where $2 \leq b < i < a \leq k-1$, and p_b and p_a are inflexions; any other point, p_j ($b < j < a$), a point on the original trajectory between p_b and p_a satisfies $d_{lp}(loc_j, loc_b loc_a) \leq \lambda$.

DEFINITION 5. A **turning point simplified trajectory** is a sequence of turning points $TP =< p_1, ...p_i, ...p_w >$, where $p_i = (x_i, y_i, t_i)$ is the i^{th} turning point.

DEFINITION 6. Interest region. Let a set of *inflexion-based trajectories* be $\Sigma = \{TP_1$, ... TP_u, ... $TP_m\}$ in a period of $[T1, T2]$, where $T1$ and $T2$ are timestamps. Let A be a 2-D region area with arbitrary shape, which encloses a set of points: $F = \{Q_1, ..., Q_i, ...Q_c\}$, where $Q_i = (loc_i, t_i) \in TP_1 \cup ...TP_u... \cup TP_m$ and $c = |F|$.

- (1) A is a **(Explicit) Popular turning region**, if $c \geq min\_sup$.
- (2) A is an **Implicit popular turning region**, if $c < min\_sup$ but $(c + b) \geq min\_sup$, where b is the number of direct line segments (on different trajectories) stabbing A.

Both explicit and implicit popular turning region are interest regions.

We also call **Explicit/Implicit popular turning regions** in Definition 6 as **Explicit/Implicit Semantic Places/Regions**.

DEFINITION 7. Longest Common Route pattern (LCR pattern). Let a sequence of *interest regions* be $\zeta =< r_1, ..., r_i, ..., r_n >$ ($n \geq 2$ and $2 \leq i \leq n$), which is visited by a set of moving objects, $MOset = \{MO_1, MO_2$, ...,$MO_k\}$. ζ is a Longest Common Route pattern, if the following criteria are satisfied:

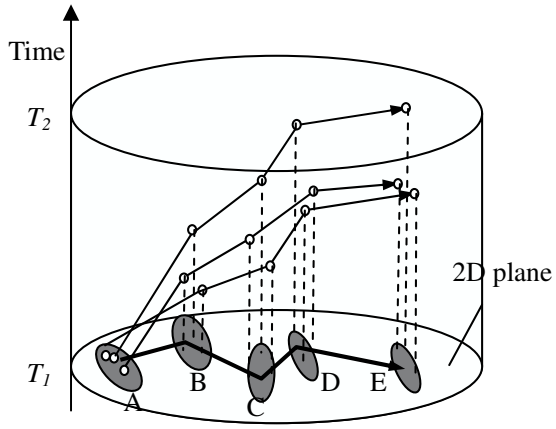


Figure III.2: An example of an LCR pattern with popular turning regions on a 2D plane.

- (1) $k \geq min\_sup$;
- (2) k is the maximum value, no any other moving objects $MO_i \notin MOset$ visit the whole route: ζ;
- (3) ζ is the longest route visited by the whole set: $MOset$.

An example of popular turning regions for a LCR pattern is shown in Fig. III.2, given min_sup=3. Note that this LCR includes three route instances (pieces of trajectories) of different moving objects. The points on the route instances are turning points, which project within different popular turning regions on the 2D plane. We may give an ID to each popular turning region.

DEFINITION 8. A Polygon line based LCR (P-LCR) pattern is an LCR pattern which includes at least two popular turning regions.

DEFINITION 9. A Direct line based LCR (D-LCR) pattern is an LCR pattern which includes at most one popular turning region and at least one implicit popular turning region.

DEFINITION 10. Given N sequences, an LCS is the longest subsequence with at least min_sup supports, $2 \leq min\_sup \leq N$.

DEFINITION 11. d_{ll} is the minimal distance between any two points p and q, where $p \in AB$ and $q \in CD.$, that is, d_{ll}=min ($d_{pl}(p, CD)$), where $p \in AB$ and d_{pl} is defined in Definition 3.

Definition 3 and Definition 11 are based on the similar definitions in [JYZ$^+$08]. Then, the problem of mining LCR patterns is modeled as follows. We first detect *turning points* (Definition 4) by the DP algorithm and a single trajectory is denoted by a *turning point simplified trajectory* (Definition 5). We then group all the *turning points* in a set of trajectories into clusters and *turning points* in the same cluster are taken as in the same *popular turning region* (Definition 6). Thus, a trajectory can be abstracted using a sequence of *popular turning region* IDs. Note that we only mark region IDs on the *turning points* but keep the original locations in order to compute the distance between region IDs. Therefore, we can discover *P-LCR patterns* (Definition 8) by mining *LCS* (Definition 10). Finally, we discover *implicit popular turning regions* (Definition 6). Based on d_{lp} (Definition 3), we compute d_{ll} (Definition 11) of two direct line segments. Two direct line segments can be grouped into the same cluster, if d_{ll} is in a bounded error and the angle between the two direct line segments is also in another bounded error. So, for each direct line segment cluster, we discover

the *implicit popular turning regions* through analyzing the overlapping part of the direct line segments. This is the process by which we retrieve *D-LCR patterns* (Definition 9). We prove that *P-LCR patterns* and *D-LCR patterns* compose the whole set of *LCR patterns* (Definition 7) in Theorem 1.

THEOREM 1. *Given a* P-LCR pattern *set* S_{PLCR}*, a* D-LCR pattern *set* S_{DLCR} *and an LCR pattern set* S_{LCR} *for the same input trajectories, then they satisfy: (1)* $S_{PLCR} \cap S_{DLCR} = \Phi$*; (2)* $S_{LCR} = S_{PLCR} \cup S_{DLCR}$*.*

Proof. (1) We use a pair of interest region IDs, P, to denote a *D-LCR pattern* and a sequence (no less than two) of interest region IDs , H, to denote a *P-LCR pattern*. $P \in S_{DLCR}$ and $H \in S_{PLCR}$. We firstly insert the implicit popular turning regions of P into H, and achieve denoted by H', if P is included in a piece of segment of H and any of P's ID are not in H yet. There are two cases:

Case 1: a *D-LCR pattern* is fully included in a *P-LCR pattern*, that is $H' \supseteq P$ (this means P is a subsequence of H'), since a *P-LCR pattern* includes no less than two popular turning region IDs and a *D-LCR pattern* includes at most one popular turning region IDs. If $H' \neq P$, then P is not the longest common route and thus P is not an LCR; this also means $P \notin S_{DLCR}$. Therefore, $H' = P$, a *D-LCR pattern* is a subsequence of a *P-LCR pattern*; in this case, we neglect this *D-LCR pattern*.

Case 2: two lists of moving objects, $MOSet1$ and $MOSet2$ support H' and P, respectively and a part of a *D-LCR pattern* is included in a *P-LCR pattern*, that is $H' \cap P \subset P$

or $(P - H') \cap H' = \Phi$. If $MOSet1$=$MOSet2$, then this produces a paradox: moving objects that support both H' and P not only changed direction but also went straight without direction change. Therefore, $MOSet1 \neq MOSet2$ and $H' \cap P = \Phi$. This proves Theorem 1(1).

(2) According to Definition 8 and 9, $S_{PLCR} \subseteq S_{LCR}$ and $S_{DLCR} \subseteq S_{LCR}$. Thus, $S_{PLCR} \cup S_{DLCR} \subseteq S_{LCR}$. Suppose, in addition to P-LCR patterns and D-LCR patterns, there is another type of pattern, say an X pattern, which belongs to LCR pattern. Thus, $S_{PLCR} \cup S_{DLCR} \cup S_X = S_{LCR}$ and $S_X \cap (S_{PLCR} \cup S_{DLCR}) = \Phi$. According to Definition 8, a P-LCR pattern includes m popular turning regions, $m \geq 2$. According to Definition 9, a D-LCR pattern includes n popular turning regions, $0 \leq n < 2$. Suppose an X pattern includes k popular turning regions. $S_X \cap (S_{PLCR} \cup S_{DLCR}) = \Phi$, $0 \leq k < 2$ and ($k < 0$ or $k \geq 2$), and thus $k < 0$. This is not correct, since $k \geq 0$. So, $S_X = \phi$. This proves Theorem 1 (2). □

Without losing generality, a Longest Common Route (LCR) actually is a kind of spatial Longest Common Curve (LCC) and we will formally define LCC, P-LCC and D-LCC in Chapter VI. The flowchart of the MicPasts method is shown in Fig. III.3.

The MicPasts method includes four main steps:

- *Step 1. Detect Turning Points by using the DP algorithm on a single curve.*
- *Step 2. Discover Semantic Places based on the DBSCAN clustering algorithm from turning points of multiple curves.*

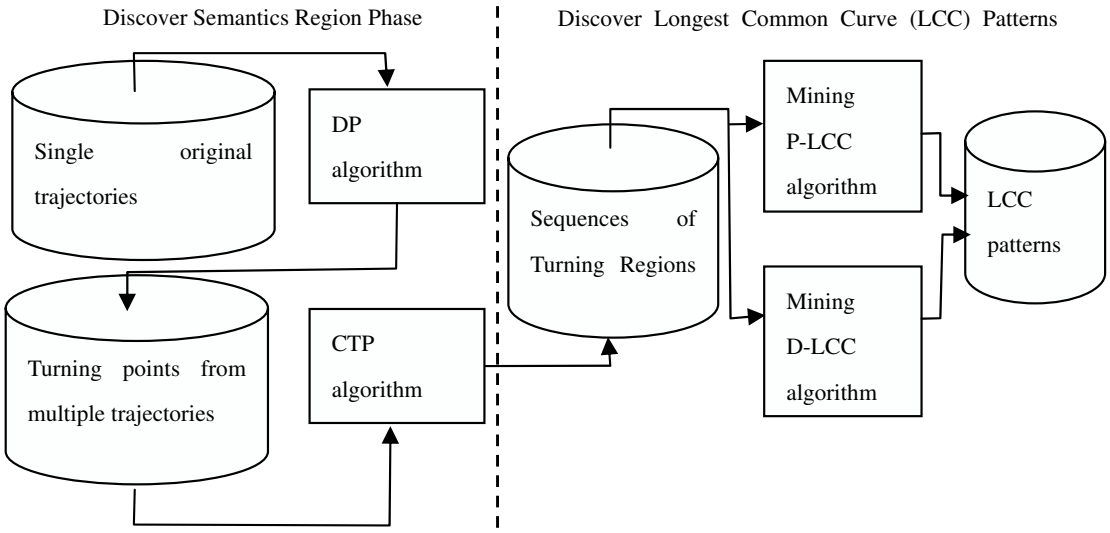


Figure III.3: Flowchart of MicPasts method.

- *Step 3. Discover Implicit Semantic Places from turning points of multiple curves.*

- *Step 4. Discover Longest Common Curve Patterns.*

III.3 The MicPasts Method

In this section, we present the details of the MicPasts Method.

III.3.1 Discovering Semantic Places

According to Definition 4, the DP algorithm [DP73] [Whi85] that is introduced in Chapter II is suitable to detect turning points (or critical points) for trajectory simplification.

The real entity value that changes with time in the physical world would be recorded, for example, by sensor readings. But different sensors (or the same sensor working at different times) may generate different readings for the same real entity value due to random sam-

pling errors and thus makes querying on sensor streams uncertain. Since random sampling errors are generally bounded, we can group a set of critical points that actually are readings for the same entity value into a cluster and use the expected value of the reading set as a semantic place (e.g., given an ID) to denote the real entity value. The advantage of using the expected value is twofold: first, it unifies uncertain sensor readings; second, it reduces the error between the sensor reading and the real entity value.

Clustering is regarded as the segmentation of a heterogeneous population into a number of more homogeneous subgroups and finding groups in a data set by some natural criterion of similarity [HHZS07]. Thus, the clustering methods are useful to group turning points for computing semantic places. The DBSCAN algorithm [EKSX96] is a popular clustering algorithm since it can discover clusters with arbitrary shape, so we develop a Clustering Turning Points (CTP) algorithm based on the DBSCAN algorithm to group turning points into turning regions. Then trajectories are denoted by sequences of popular turning region IDs.

We show in Fig. III.4 that using semantic places to summarize stream curves helps to reduce data volume considerably.

III.3.2 Discovering the Implicit Semantic Places (DISP) Procedure

In this section, we present the DISP procedure. The main purpose of the DISP procedure is to discover implicit semantic places and the algorithm is given in Fig. III.5. The DISP

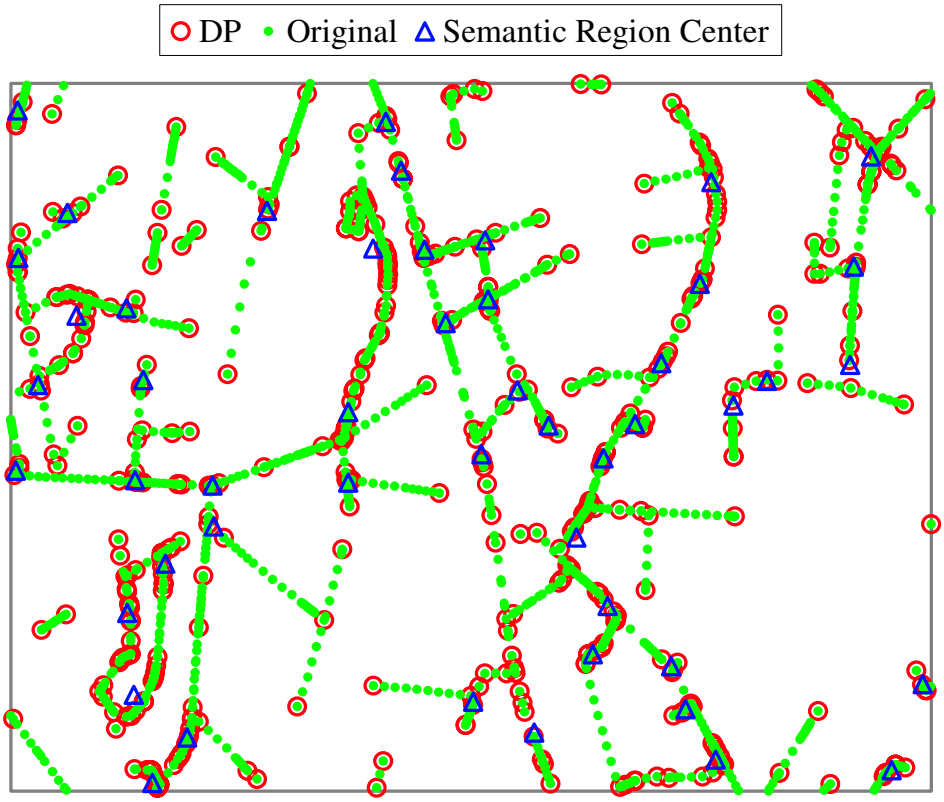


Figure III.4: Example of reducing data volume: From original points, to turning points and then to semantic place centers.

Algorithm 1. Discovering Implicit Semantic Places. Given a set of direct line segments, *DLSet*, the procedure DISP (*DLSet*) detects all the implicit semantic places.

Procedure **DISP** (*DLSet*)
1. *Cluster_DLS*←**DL_DBSCAN***(DLSet, DL_angle, DL_eps, MinPts)*;
2. **For** *each cluster i* in *Cluster_DLS*
   CutSet ←**DIP**(*Cluster_DLS(i)*); /* Algorithm 2 */
3. **Return** *CutSet*;

Figure III.5: Discovering implicit semantic places algorithm.

procedure comprises two sub-algorithms: a DL_DBSCAN algorithm (DL denotes Direct Line segments) for clustering direct line segments based on the DBSCAN algorithm and a Discovering Implicit Points (DIP) algorithm.

In the DL_DBSCAN algorithm, there are two criteria to determine the similarity between two different direct line segments:

- Spatially close to each other;
- Similar direction.

We use an intuitive method to compute the distance between two direct line segments, d_{ll} (Definition 11). First, we compute the distance between a point and a line segment, d_{lp} (Definition 3), as follows:

$$d_{lp}(A, CD) = \begin{cases} a & (\alpha < 90^o, \beta > 90^o), \\ b & (\alpha > 90^o, \beta < 90^o), \\ h & (\alpha \leq 90^o, \beta < 90^o) \\ 0 & (A \in CD) \\ a & (\alpha = \beta = 0, a < b) \\ b & (\alpha = \beta = 0, b < a) \end{cases} . \quad \text{(III.3.1)}$$

where $h = 2\sqrt{s(s-a)(s-b)(s-c)}/c$, $\alpha = \angle ACD$, $\beta = \angle ADC$, $a = |AC|$,$b = |AD|$, $c = |CD|$ and $s = (a+b+c)/2$. Then, d_{ll} can be computed as follows:

$$d_{ll} = \begin{cases} 0 & (Intersected) \\ min(d_{lp}(A,CD),d_{lp}(B,CD),d_{lp}(C,AB),d_{lp}(D,AB)) & (Other) \end{cases} . \quad \text{(III.3.2)}$$

We define the angle between two consecutive direct line segments as follows:

$$f(u, v) = arccos(\frac{u \bullet v}{|u| * |v|}), f \in [0, \pi]. \quad \text{(III.3.3)}$$

where $\pi = 3.1415926$, u and v are two vectors, and $|u|$ and $|v|$ denote the lengths of the two vectors, respectively.

Given three consecutive points on a trajectory: A, B and C, according to Eq. III.3.3, we can compute an angle $\angle ABC = f(\overrightarrow{AB}, \overrightarrow{BC})$. There are two important parameters: DL_angle and DL_eps in DL_DBSCAN. DL_angle is the threshold angle between two consecutive direct line segments and DL_eps is the threshold of distance between two direct line segments.

Algorithm 2. DIP Algorithm. Given a cluster of direct line segments, the procedure DIP (*Cluster_DLS[i]*) outputs implicit semantic places.

Procedure **DIP** (*Cluster_DLS[i]*)

1. *avAngle* ←Average direction of *Cluster_DLS[i]*;
2. **For** each direct line segment *j* in *Cluster_DLS[i]*
   2a. *RDLSCluster[j]*←**Rotate**(*avAngle*, *Cluster_DLS[i][j]*);
   2b. *CutSet* ← *RDLSCluster[j].x1'*;
   2c. *CutSet* ← *RDLSCluster[j].x2'*;
3. **Qsort**(*CutSet*);
4. Remove redundancy from *CutSet*;
5. **For** each point *j* in *CutSet*

5a. **For** each direct line segment *k* in *Cluster_DLS[i]*

If (*[CutSet[j],CutSet[j+1]* ∈ *[RDLSCluster[k].x1', RDLSCluster[k].x2']*)

CutSupport[j]+=Cluster_DLS[i][j].nSupport;

5b. **If** point j has no less than *min_sup* support
ISPset←point j; /* j is an implicit semantic place*/

6. **Return** *ISPset*;

Figure III.6: DIP algorithm.

We then present the main idea of the DIP algorithm in Fig. III.6. Given n clusters generated by DL_DBSCAN that are denoted by Φ_1 , Φ_2, ..., Φ_n and $\Phi = \{\Phi_1 \cup \Phi_2 \cup .....\Phi_n \cup \Phi_{noise}\}$. First, the average direction, $\overrightarrow{v}$, of each cluster of direct line segment is computed, then we rotate the axes so that the X axis is parallel to $\overrightarrow{v}$. Most related work that handles a cluster of direct line segments is given in [LHW07] for achieving representative trajectory. Different from them, we rotate a cluster of direct line segments to help discover cuts (or implicit semantic places). Given a direct line segment cluster: $RDLSCluster = \{AB, CD, FE\}$, where $A(x1, y1)$, $B(x2, y2)$, $C(x3, y3)$, $D(x4, y4)$, $E(x5, y5)$ and $F(x6, y6)$ in $X - Y$ coordinate, we can compute $A'(x1', y1')$, $B'(x2', y2')$, $C'(x3', y3')$, $D'(x4', y4')$, $E'(x5', y5')$ and $F'(x6', y6')$ in $X' - Y'$ coordinate by Eq. III.3.4.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix}. \tag{III.3.4}$$

Thus, $x6' < x1' < x3' < x5' < x4' < x2'$ as shown in Fig. III.7. For each cluster of direct line segments as Fig. III.7 shows, we find all the cut points and sort them. The sorted cuts are $< x6', x1', x3', x5', x4', x2' >$, so there are 5 cut sections: $[x6', x1']$, $[x1', x3']$, $[x3', x5']$, $[x5', x4']$ and $[x4', x2']$. For each cut section $[xi', xj']$ included in any line segments in $RDLSCluster$, the number of supports are counted. Note that one direct line segment may have more than one support. If a cut with supports no less than min_sup, then the cut is an implicit semantic place. For example, if $min\_sup = 1$, then all the cuts are implicit semantic places.

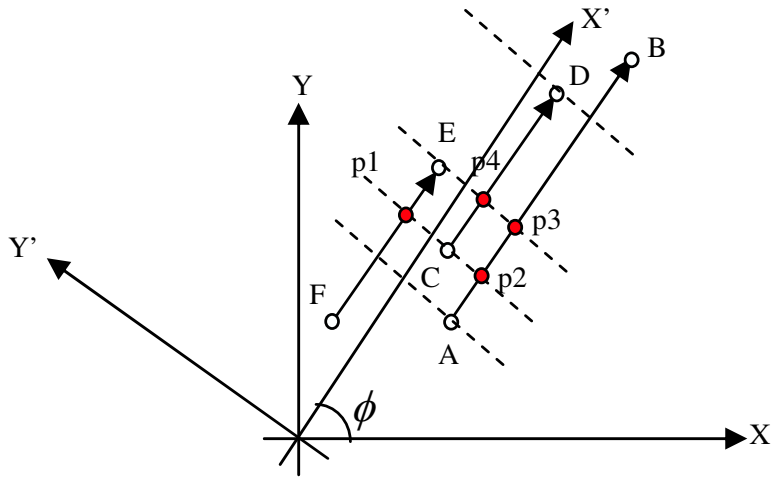


Figure III.7: Cut points in a cluster of direct line segments.

III.3.3 Discovering Longest Common Curve Patterns

Mining Longest Common Subsequences

We present an algorithm for mining LCS in Fig. III.8. First, a suffix tree is built with a list of moving objects on the nodes of the suffix tree at line 1. After retrieving all common sequences by the procedure $SuffixTree\_CS$ (see Figure II.5) at line 2, we remove subsequences at line 3 by the procedure $Remove\_SubSeq$ to only leave Longest Common Sequences patterns.

Refining Coarse LCR Patterns Using DISP Procedure

Fig. III.1 (a) and (b) shows two typical cases of using the DISP procedure to retrieve implicit semantic places and thus to refine the coarse LCR patterns or to retrieve LCR patterns

Algorithm 3. Mining LCS. Given a set of sequences of semantic places: S_{PTR}, the procedure LCS(S_{PTR}) retrieves LCS.

Procedure **LCS** (S_{PTR})
1. $STr \leftarrow$ Build suffix tree for S_{PTR};
2. $S_{CS} \leftarrow$ **SuffixTree_CS***(STr);*
3. $S_{LCS} \leftarrow$ **Remove_SubSeq** (S_{CS});

Figure III.8: Mining LCS algorithm.

that do not comprise LCS. We present the process of using the DISP procedure to complete finding all the exact LCR patterns as follows.

- (1) We first retrieve the direct line segments that are connected to the two ends of an LCS into two sets: $EndLineSet1$ and $EndLineSet2$. Then we use $EndLineSet1$ (or $EndLineSet2$) as the input of the DISP procedure to achieve implicit semantic places and choose the implicit semantic place with maximum distance to the end to extend the LCS pattern to a full LCR pattern. Note that extensions may happen at both ends of the LCS. Fig. III.9 shows an example of extending an LCS to a P-LCR.

- (2) Then, we remove the direct line segments that support the already retrieved LCR patterns and put the rest of the direct line segments into a set: $DirectLineSet$. We use $DirectLineSet$ as the input of the DISP procedure to achieve implicit semantic places and those two implicit semantic places with maximum distance is a new LCR pattern. Fig. III.1 (b) shows an example for retrieving a D-LCR.

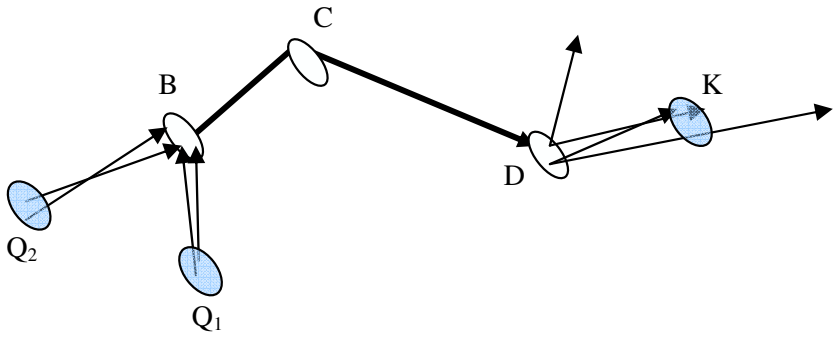


Figure III.9: Example of extending an LCS: "BCD" to achieve a P-LCR.

We notice that the implicit semantic places cannot connect two LCS patterns into one longer LCS pattern. For example, suppose we have retrieved two LCS patterns: $ABCD$ and FGH and we extend them to $ABCDE$ and $E'FGH$, where E and E' are two implicit semantic places discovered by the DISP procedure, then $E \neq E'$, supposing both $ABCDE$ and $E'FGH$ are supported by an object list $\{MO1, MO2, MO3\}$. If $E = E'$, there are two cases:

- (1)D, E and F are on a direct line since E is an implicit semantic place. But $ABCDFGH$ can be retrieved by an LCS and E is useless. Thus, $E \neq E'$.

- (2)D, E and F are not on a direct line; that is, E is an inflexion. E should be detected as a semantic place by the CTP algorithm. But E is an implicit semantic place. Thus, $E \neq E'$.

III.4 Performance Evaluations

In our experiments, we used a Network-based Generator of Moving Objects [Bri02] to generate various sizes of moving object datasets, where we chose an open road map of Oldenburg (a city in Germany) as the network input in Fig. III.10. Each moving object has a sensor to sample locations. A set of parameters are used to control the datasets of moving objects, such as the number of moving objects, the number of classes of moving objects, the maximum time (the number of timestamps), report probability (e.g. if a report is sent every two timestamps, the probability is 50%) and maximum speed (the sum of the x- and y-extension of the data space is divided by div for determining the maximum speed of the objects). Note that there may be $M = M1 + M2$ moving objects at the beginning, where $M1$ is for internal and $M2$ is for external objects, and $m = m1 + m2$ (for internal and external objects respectively) new moving objects generated dynamically may be added into the network. Also, the speed is limited on the edge, when the number of objects exceeds the specified capacity [Bri03]. We set report probability of 100%, 10 classes of moving objects at the beginning and 5 classes during the process, $M1 = 1000$, $M2 = 10$, $m1 = 5$, $m2 = 0$ and the maximum speed is determined by $div = 250$. Then, we change the value of maximum timestamps to produce various volumes of datasets as shown in Table III.1.

We introduce three metrics to evaluate the accuracy of retrieved pattern results. Firstly, we define three concepts similar to the definitions in our work [HZSH10]: TP (True Positive)- the true (or correct) pattern length retrieved; FP (False Positive)- the false (or wrong) pattern

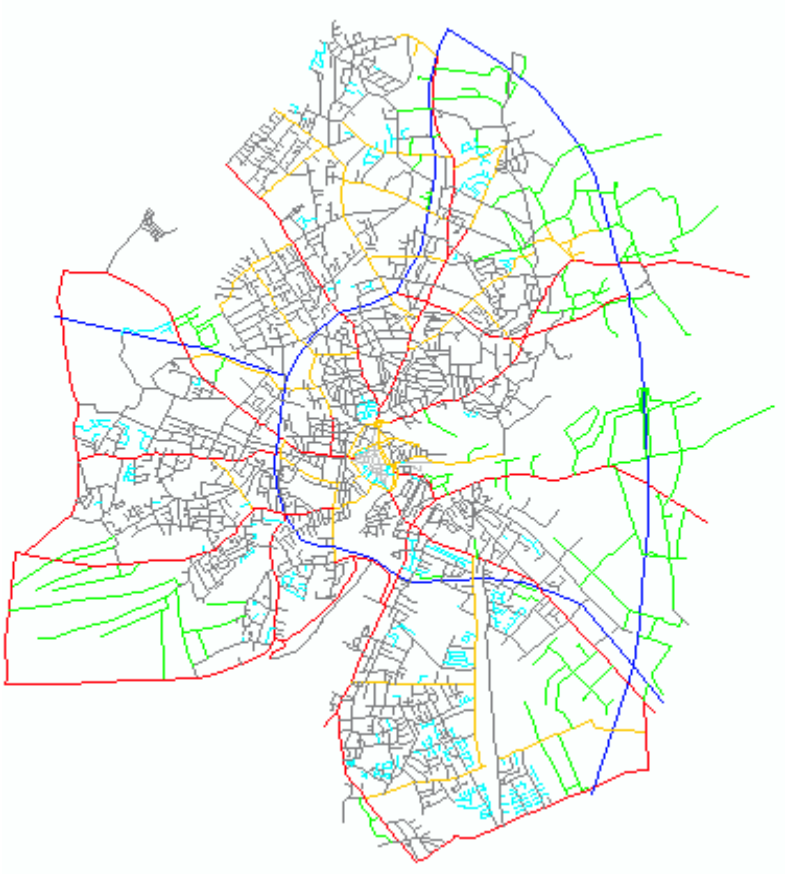


Figure III.10: Road network of Oldenburg city.

Table III.1: Moving Objects Datasets.

Parameter	Test 1	Test 2	Test 3	Test 4	Test 5
Max. Timestamps	60	120	180	240	300
Number of MOs	1,295	1,600	1,900	2,200	2,500
Number of Points	65,012	121,833	179,314	230,082	264,348
Data size (Mb)	3.2	6.2	9.2	11.7	13.8
Total pattern length Benchmarks(Meter)	24470	54509	81502	103613	117839

length retrieved; FN (False Negative)- the true (or correct) pattern length that is not retrieved. Then, the three metrics are:

$$False\ Positive\ Rate = \frac{FP}{TP + FP}, \tag{III.4.1}$$

$$False\ Negative\ Rate = \frac{FN}{TP + FN}, \tag{III.4.2}$$

$$Total\ False\ Rate = \frac{FP + FN}{FP + TP + FN}, \tag{III.4.3}$$

The total pattern length retrieved equals ($TP + FP$) and the benchmark pattern length equals ($TP + FN$).

Several parameters of our algorithms are given in Table III.2. We set $DL\_eps = eps$ and $MinPts = 2$. We now briefly describe our experimental setup. Given the trajectories exactly denoted by intersections provided by the simulator, we ran the MPLCR algorithm to retrieve a set of longest common subsequences of intersections supported by at least min_sup moving objects as benchmarks (shown in Table III.1), which were used to measure the accuracy of our proposed methods. We studied three cases of proposed algorithms: DP2, DP5 and DP10, where DPi denote using DP ($\lambda = i$) to abstract trajectories as input for the LCR-Turning algorithm (e.g. CTP, MPLCR and MDLCR). First, we studied the optimal eps and optimal DL_angle by setting $min\_sup = 2$. Then, we observed the scalability based on datasets with various data sizes shown in Table III.1. We adopted the same optimal values of

Table III.2: Parameters of LCRTurning Algorithm.

Parameter	**Explanation**	**Parameter**	**Explanation**
Angle_eps	Threshold angle in *CL_DBSCAN*	λ	Pre-defined distance threshold in *DP*
DL_Angle	Threshold angle in *DL_DBSCAN*	*eps*	Threshold distance in *DBSCAN*
DL_eps	Threshold of d_{ll} in *DL_DBSCAN*	*min_sup*	Threshold number of supports for a pattern.
MinPts	In three clustering algorithms: *DBSCAN*, *CL_DBSCAN* and *DL_DBSCAN*.		

eps and DL_angle in the scalability test. We use false positive rate and false negative rate to measure the accuracy of patterns. Given the benchmark total pattern length, L', and the total retrieved pattern length, $L = L_c + L_e$, where L_c is the correct retrieved pattern length and L_e is the error retrieved pattern length. The false positive rate is L_e/L' and false negative rate is $(L' - L_c)/L'$. All experiments were conducted on an IBM Laptop with Microsoft Windows XP, Genuine Intel 1.83GHz CPU and 512MB main memory. We implemented the proposed algorithms mainly by using C++, and the suffix tree based retrieving LCS algorithm was implemented by Java. We changed all the lengths into meters by timing 30.92 meters in the following experiment results; this was the simplest way to achieve rough length represented by meters between locations (longitude, latitude). The parameter values depend on the applications.

III.4.1 Optimal eps and DL_angle

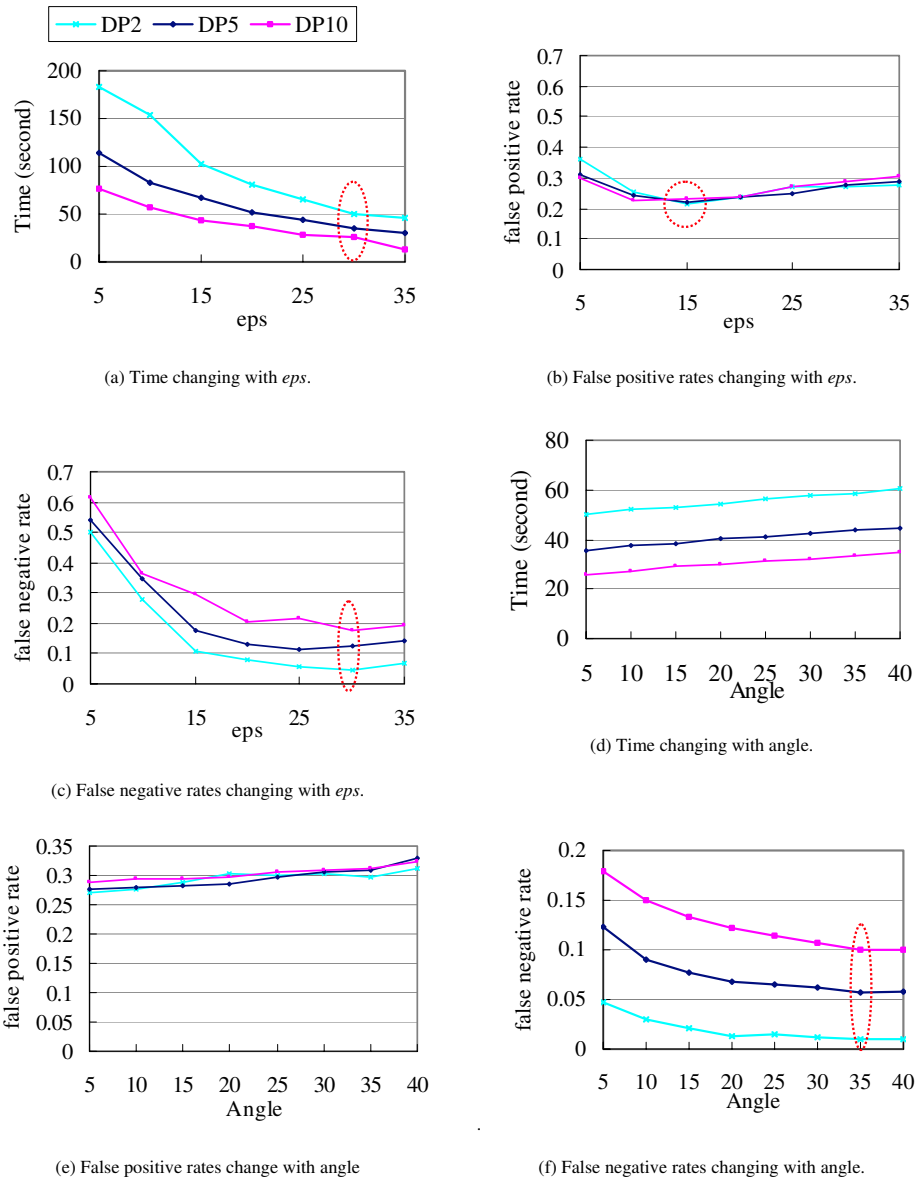
Given $L\_Angle = 5^o$, $L\_eps = eps$ and min_sup=2, we ran our proposed algorithms on dataset Test 1. The time spent on retrieving LCR patterns is plotted in Fig. III.11 (a); it shows that the greater the value of eps, the less time spent on running algorithms DP2,

DP5 and DP10. DP10 performed more efficiently than both DP2 and DP5, and DP5 was more efficient than DP2. We analyze the accuracy of patterns shown in Fig. III.11(b)(c). The whole trend of the three algorithms in Fig. III.11 (b) is that the lowest values of the false positive rate are all at eps=15. Also, DP2 performs better than DP10 and DP5, with the lowest false negative rate (4.7%) at eps=30 as shown in Fig. III.11 (c). Overall, the best effectiveness and false negative rate is achieved at eps=30, while the best false positive rate is achieved at eps=15. Since all three algorithms performed more efficiently at eps=30 than at eps=15, eps=30 was the optimal one with the best effectiveness.

Given $L\_eps = eps = 30$, we studied the optimal DL_angle. We plotted time changing with angles in Fig. III.11(d), where DP2 consumes more time than DP5 and DP10. DP2 achieves the best false negative rate, near to zero at $L\_Angle = 35$ in Fig. III.11 (f). Fig. III.11 (f) also shows that the greater the angle, the lower the value of the false negative rate, while Fig. III.11 (e) shows that the value of the false positive rate is increased with the angle. Overall, DP2 is more accurate but less efficient. The best false positive rate and the best time efficiency are achieved at $L\_Angle = 5$ and the best false negative rate is achieved at $L\_Angle = 35$.

III.4.2 Efficiency and Accuracy

Based on the results in Section 4.1, we set eps=30 and $L\_Angle = 35^o$ to evaluate the efficiency and accuracy of our algorithms in various dataset sizes.



(a) Time changing with *eps*.

(b) False positive rates changing with *eps*.

(c) False negative rates changing with *eps*.

(d) Time changing with angle.

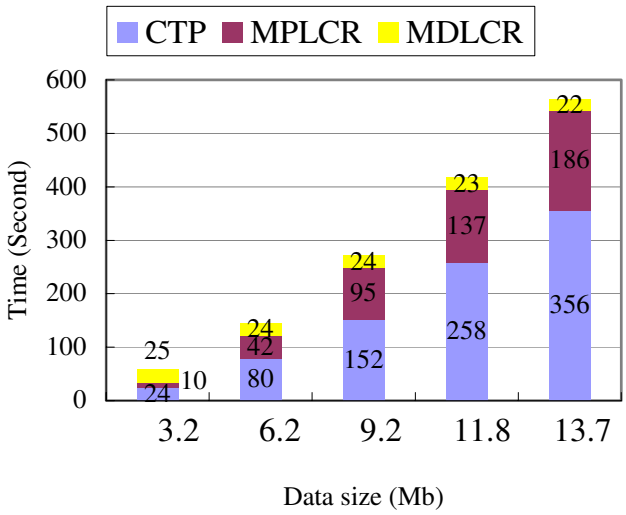
(e) False positive rates change with angle

(f) False negative rates changing with angle.

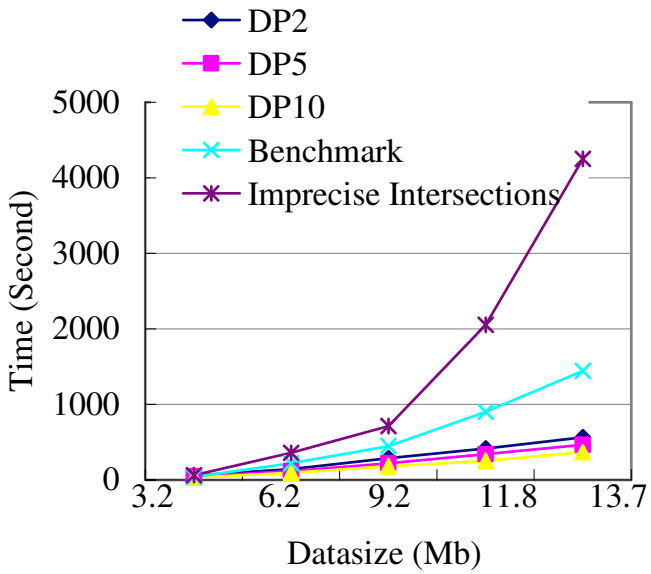
Figure III.11: Analysis of optimal parameters.

Our proposed algorithms are more efficient than both Benchmark and Imprecise Intersections methods as shown in Fig. III.12 (b). In the Imprecise Intersections method, where the intersections provided by the simulator are not repeated exactly, we use the CTP algorithm with $eps = 0.1$ to group imprecise intersections into region clusters and mine LCS of region IDs. The Imprecise Intersections method is not efficient, since a huge number of imprecise intersections make the CTP algorithm time-consuming. The time costs of DP2, DP5 and DP10 are all nearly linear, while the time cost of Benchmark and Imprecise Intersections climbed quickly with increased data size, up to around four times and twelve times respectively, spent on DP10. The Imprecise Intersections method, which has the worst time efficiency, also validates that using turning regions to retrieve LCR patterns is efficient. Fig. III.12 (a) shows in DP2, the time spent on CTP is increased around 14 times from 24s at data size of 3.2Mb to 356s at 13.7Mb, and the time spent on MPLCR is increased around 18 times from 10s to 186s, while the time spent on MDLCR is unchanged at around 24s. The time spent on CTP is around twice that spent on MPLCR.

The accuracy of our proposed algorithms is shown in Fig. III.13(a)(b). When data size is increased, the false positive rate stays in the range of 30%-35%, and the false negative rate is lower, always less than 15%. Fig. III.13(c) shows that the correct lengths of D-LCR patterns are nearly the same in various data sizes, which also only constitute a very small part of the total lengths. The correct length of P-LCR patterns is the major part: from 86% at data size=3.2Mb to 98% at data size =13.7Mb. Although we evaluate our proposed algorithm by

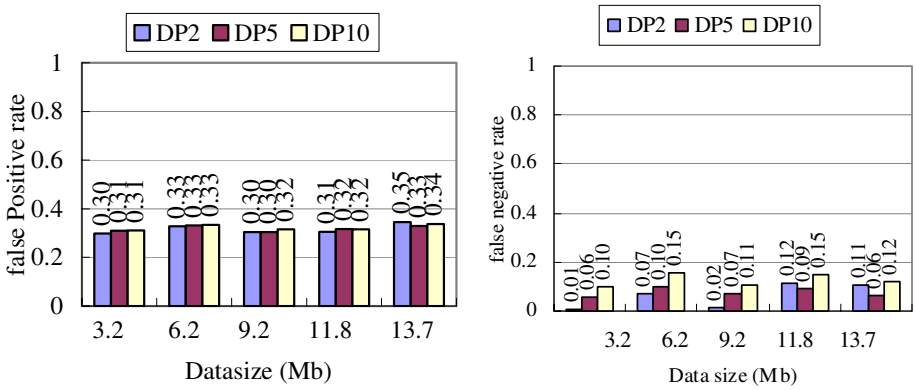


(a) Time distribution.

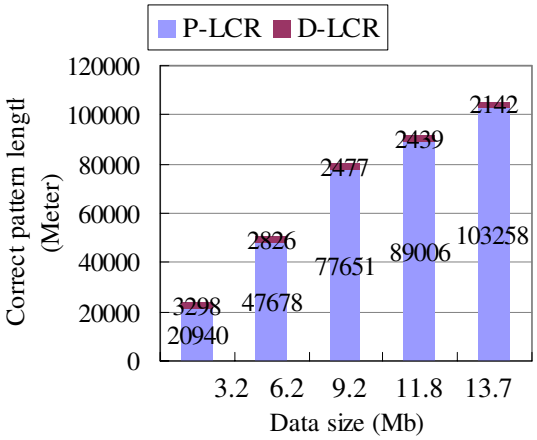


(b) Time efficiency.

Figure III.12: Time Efficiency.



(a) False Positive Rate. (b) False Negative Rate.



(c) P-LCR vs. D-LCR.

Figure III.13: Accuracy.

using simulated traffic moving objects, the LCRTurning algorithm can be used widely on any trajectories of moving objects that may not follow a road network, since turning regions are discovered automatically from trajectories.

III.5 Summary

In this chapter, we have proposed a novel MicPasts method to retrieve longest common curves. Particularly, we implement an LCRTurning algorithm based on the MicPasts method to retrieve longest common route patterns from moving object trajectories. The LCRTurning algorithm has wide applications for any trajectories that may not follow a road network. Experimental results validated that the LCRTurning algorithm is more efficient than the algorithms that retrieve LCR patterns by using intersections while achieving reasonable accuracy.

Chapter IV

The CIRCE Method

"In this world nothing can be said to be certain, except death and taxes."

–Benjamin Franklin (*The Works of Benjamin Franklin*)

We have mentioned in the previous chapter that continuous sensor or manually-captured data streams are often recorded as a series of discrete points in a database from which knowledge can be retrieved through queries. In this chapter, we focus on tackling the uncertainty problems in mining data streams. Two classes of uncertainties inevitably happen in data streams: DS Uncertainty (Uncertainty due to Discrete Sampling) and SE Uncertainty (Uncertainty due to Sampling Error) that we present as follows. Even if every discrete point is correct (actually, we can use many methods to ensure this, including our work in [HZHC11] to improve the data quality in the gathering process), the discrete data streams is uncertain, that is, it is not exactly like the continuous stream since some critical points are missing due to the limited capabilities of the sensing equipment and the database server. Also, sensor readings for the same situation cannot be repeated exactly when we record them at different

times or use different sensors since different sampling errors exist. Both DS and SE Uncertainties reduce the efficiency of querying common patterns. Already known algorithms generally only resolve Uncertainty due to Sampling Errors (SE Uncertainty).

Recall that in Chapter III, we have proposed a MicPasts method to resolve the Uncertainty due to Sampling Errors problem. The MicPasts method summarizes the original data streams by using inflexions, then groups close inflexions into the same cluster, and thus changes the uncertain data streams into exact sequences of cluster IDs. Particularly, to help query common patterns directly on exact ID sequences, a DISP procedure that is developed in the MicPasts method (see Chapter III) is used to deduce the implicit common regions.

In this chapter, we extend MicPasts method to develop a novel Correcting Imprecise Readings and Compressing Excrescent Points (CIRCE) Method to tackle both Uncertainty due to Discrete Sampling (DS Uncertainty) and Uncertainty due to Sampling Error (SE Uncertainty). To resolve the Uncertainty due to Discrete Sampling, a novel CIRCE core algorithm is developed in the CIRCE method to correct the missing points while compressing the original data streams. The experimental study based on various sizes of stream datasets validates that the CIRCE core algorithm is more efficient and more accurate than a counterpart DP algorithm to compress data streams by using ID sequences. Also, the application for querying Longest Common Route patterns validates the effectiveness of the CIRCE method.

The structure of this chapter is organized as follows. Firstly, we present an overview of the CIRCE method in Section 1. Then, the CIRCE method is provided in Section 2. In

Section 3, we introduce querying common patterns based on the CIRCE method. The performance of the proposed method is evaluated in Section 4 and finally, Section 5 summarizes this chapter.

IV.1 Overview of the Correcting Imprecise Readings and Compressing Excrescent Points (CIRCE) Method

With the advances in satellite, RFID, GPS, wireless and video technologies, stream database systems that manage a time series of sensor readings are becoming increasingly more available. Continuous data streams are often recorded as a series of discrete points in databases [GBEe00], where useful knowledge can be achieved through queries. Querying stream databases has a wide range of applications, such as monitoring locations of moving objects (flocks [GKS07], vehicles [HLT10] [DH09] [JYZ+08] [GNPP07], cloud cluster [LHY04], fleets [LHLG08]), surveillance of environmental physical parameters (e.g. temperature [CKP07] etc.) and automatic controlling of robots through vision, sound and radio sensors. However, the data streams are inevitably uncertain and data mining over uncertain data streams has become a hot research topic [ZGZ09]. Different from simple queries (e.g., average (sum) query and nearest neighbor query) from uncertain data streams in [CKP07] and clustering uncertain data streams [ZGZ09], we are more interested in querying common patterns from multiple streams. Typical examples of common patterns are convoy [JYZ+08], spatiotemporal sequential pattern [CMC05], trajectory pattern [GNPP07] and longest com-

mon route (LCR) pattern [HZHD11], explained as follows. *Convoy* is defined in [JYZ+08] as a group of objects which travel together. *Spatiotemporal sequential pattern* [CMC05] is a Sequential PAttern (SPA) of route segments, where each route segment is visited by at least min_sup (minimal number of supports) objects. *Trajectory pattern* [GNPP07] is a set of objects traveling a common route with similar sequences of durations, where a route is denoted by a sequence of popular regions. *Longest common route (LCR) pattern* [HZHD11] is a route visited by the same list of more than min_sup moving objects, where the route is denoted by a sequence of turning points. They all require data mining over uncertain data streams.

However, the following two types of uncertainties in data streams impact the efficiency and accuracy of queries for the above mentioned common patterns.

- *Uncertainty due to Discrete Sampling* (DS Uncertainty). Even if every discrete point is correct, the discrete points on the time series is uncertain, that is, it is not exactly like the continuous stream since some critical points are missing due to the limited capabilities of the sensing equipment and the database server [MdB04]. On one hand, a large amount of redundant information exists and thus storage resources are wasted, which also reduces the efficiency of the user query. On the other hand, missing critical points makes the user query inaccurate.

- *Uncertainty due to Sampling Error* (SE Uncertainty). Sensor readings of the same

situation cannot be repeated exactly when we record them at different times or use different sensors. The trajectories of different moving objects [CMC05] are different, even if the objects move along the same route. For example, two moving objects passed the same region at time $t0$, however, the GPS sensors record two different location values: $r1(loc$(-37.6934, 144.7931), $t0)$ and $r2(loc$(-37.6935, 144.7931), $t0)$.

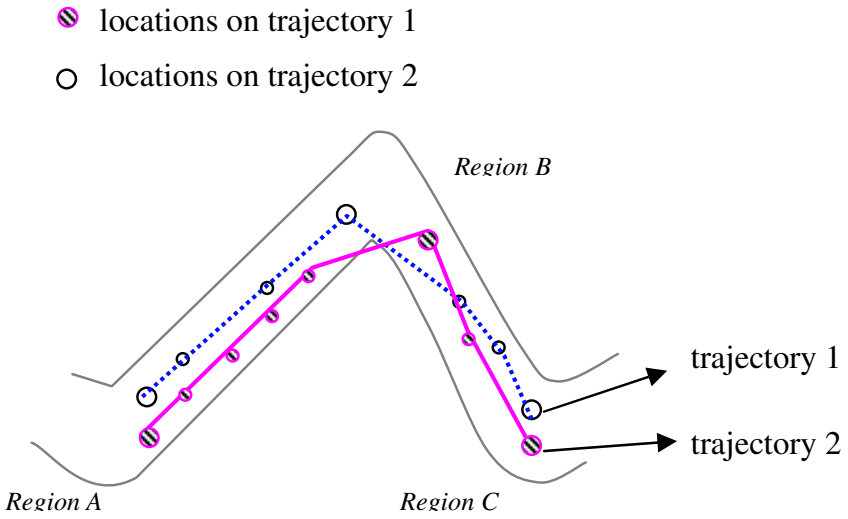
One solution to overcome Uncertainty due to Sampling Errors is to tolerate a bounded error [JYZ$^+$08]. That is, the Euclid distance between r_1 and r_2 (in above example) is 0.0001, and we suppose that it is smaller than a threshold, $\varepsilon = 0.0002$, so, r_1 and r_2 are taken as being in the same place. But to determine whether two trajectories are the same, it needs to check if every location pairs are in the bounded error and thus, queries of common patterns directly on original data streams are not efficient. To improve query efficiency, a scheme is developed to split original trajectories through inflexions (or turning points) and then to use a direct line segment between two consecutive inflexions to summarize the original trajectory piece and finally to determine that two line segments are the same if they are in a bounded rectangle [CMC05]. Another solution to overcome uncertainty is to discover semantic locations (e.g. popular regions such as intersections) and summarize original trajectories by using semantic location IDs [HZHD11][GNPP07].

However, a limitation of these methods is that it cannot tackle the Uncertainty due to Discrete Sampling problem. That is, a common pattern may not be found since it is inevitable that some inflexions are missing and thus queries are not accurate. An example is given in

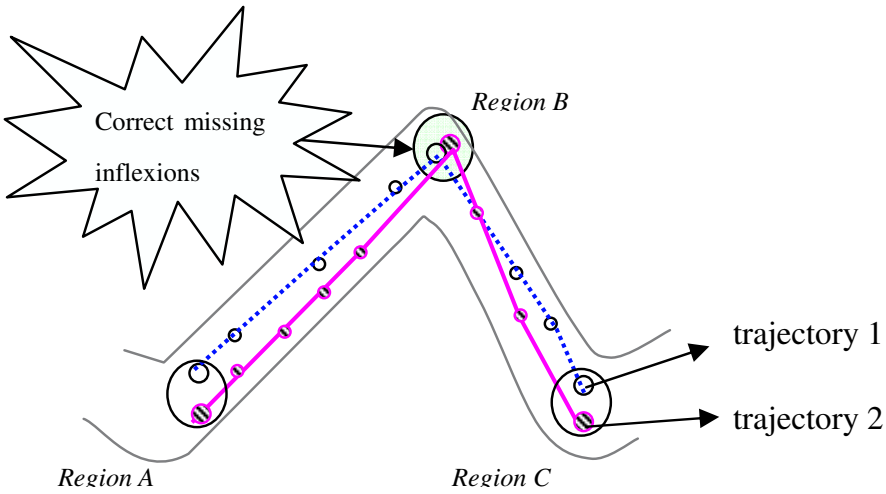
Fig. IV.1 (a).

In this chapter, we propose a novel Correcting Imprecise Readings and Compressing Excrescent points (CIRCE) method to resolve both Uncertainty due to Discrete Sampling (DS Uncertainty) and Uncertainty due to Sampling Error (SE Uncertainty) problems, which aims to achieve efficient and accurate queries of common patterns from data streams. We now present the main idea of the CIRCE method. In the CIRCE method, we first develop a CIRCE core algorithm which comprises two main procedures: the Detecting Inflexions and Computing Missing Inflexions (DICMI) procedure and the Angle-DP procedure. The DICMI procedure detects local inflexions including missing inflexions based on four consecutive points on original data streams. The Angle-DP procedure compresses an inflexion sequence generated by the DICMI procedure further into a critical point sequence. Then, the same as in the MicPasts method, we group inflexions into clusters, take the inflexions in the same cluster as the same region to overcome Uncertainty due to Sampling Errors, and finally use sequences of exact cluster IDs to compress data streams. Thus, common patterns can be queried using query methods developed for exact data, such as querying Longest Common Subsequences (LCS). Moreover, we take advantage of the efficient Discovering Implicit Semantic Places (DISP) procedure in the MicPasts method to ensure the accuracy of querying common patterns directly from cluster ID sequences.

The CIRCE core algorithm is one of the major contributions of this chapter, since it corrects the missing points as shown in Fig. IV.1 (b). This is different from already known



(a) A common pattern is not detected by already known methods due to missing inflexions.



(b) The missing inflexions are deduced by the CIRCE method and thus the common route pattern is detected correctly.

Figure IV.1: Effectiveness of the CIRCE method.

algorithms for data stream compressing that only can compress redundant data but cannot correct missing data. Fig. IV.1 (b) shows that correcting missing inflexions helps improve query accuracy. Meanwhile, Fig. IV.1 (a) shows that a common pattern is not detected by already known methods due to missing inflexions. The most related work, the Douglas-Peucker (DP) [DP73] algorithm compresses the trajectories to reduce the enormous volume of data [MdB04] [Tha89] [CMC05], but it only removes uncritical points and it cannot correct missing points. Moreover, experimental study of this chapter demonstrates that the CIRCE core algorithm-based query of common patterns is more accurate and efficient than the DP-based query. Interestingly, by correcting the missing inflexions, the CIRCE method uses less inflexions to compress a data stream, and thus achieves higher efficiency to group less inflexions into clusters. Compared to queries on original data streams, the other contributions of our CIRCE method are: (1) improving query quality; and (2) realizing highly efficient queries. In the experimental study, we take querying longest common route (LCR) patterns from various sizes of stream datasets as an example to validate the accuracy and efficiency of our CIRCE method. Note that the concepts of inflexions and turning points can be used interchangeably in this chapter.

IV.2 The CIRCE Method

In this section, we first present the framework of the CIRCE method and then introduce the CIRCE core algorithm.

IV.2.1 Framework of the CIRCE Method

The main purpose of the CIRCE method is to remove the uncertainty of the data streams in the following aspects: first, the missing critical points are deduced and thus the error between sensor readings and the real entity values is reduced; second, different sensor readings for the same entity value are grouped into the same cluster and are unified by a semantic place ID to reduce the query uncertainty; third, a large number of noncritical points (e.g. excrescent points) are also uncertain due to random sample errors, so removing them makes the data streams more clear and thus improves the query efficiency and accuracy. The CIRCE method comprises the following main steps:

- Step 1: A CIRCE core algorithm is developed to first detect inflexions including missing inflexions and then to compress excrescent points by replacing the original data streams using a direct line segment between two consecutive inflexions in a bounded error.

- Step 2: Discovering the semantic places algorithm is to remove uncertainty of data streams by using semantic places, which cluster inflexions based on DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [EKSX96] in order to discover semantic places and then reduce the uncertainty by unifying the data streams based on semantic places.

- Step 3: A Discovering Implicit Semantic Places (DISP) procedure is proposed to sup-

port efficient and accurate queries of common patterns from CIRCE-compressed data streams.

In Step 1, the missing inflexions are corrected and the excrescent points are compressed on a single data streams. In Step 2, semantic places are discovered from the whole set of data streams and a group of inflexions that actually denote the same semantic place are unified and thus the data volume is compressed further and the uncertainty of data streams is reduced further. In Step 3, when users query on the data streams denoted by semantic places, the DISP procedure helps to recover implicit semantic places and to make query results more accurate.

In the CIRCE method, we use the same terms as in the MicPasts method in Chapter III but redefine turning points (or inflexions) as follows:

DEFINITION 12. Turning point (or inflexion) is a critical point, $P_i = (e_i, t_i)$ and e_i is n-dimension point ($n = 1, 2, ...$), on a trajectory, $S =< p_1, ..., p_u, ..., p_k >$, that satisfies one of the two conditions: (1) $i = 1, k$. (2) The angle between $\overrightarrow{p_a p_i}$ and $\overrightarrow{p_i p_b}$ satisfies $Angle(\overrightarrow{p_a p_i}, \overrightarrow{p_i p_b}) > \varepsilon$ ($2 \leq b < i < a \leq k - 1$), where ε is an angle threshold, p_b and p_a are inflexions; any other non-inflexion point, p_j ($b < j < a$) satisfies $Angle(\overrightarrow{p_a p_j}, \overrightarrow{p_j p_b}) \leq \varepsilon$.

Also, the discovering semantic places algorithm in step 2 and the DISP procedure in step 3 are the same as those in the MicPasts method.

IV.2.2 CIRCE Core Algorithm

The CIRCE core algorithm aims to find inflexions in single data streams, which comprises two steps:

- Step 1: Use a Detecting Inflexions and Computing Missing Inflexions (DICMI) algorithm for detecting local inflexions including missing inflexions. Then, compress an original data stream by a sequence of inflexions.

- Step 2: Provide a novel Angle-DP algorithm for finding critical points from inflexions. Then, compress an inflexion sequence generated by the DICMI algorithm further into a critical point sequence.

The CIRCE core algorithm compresses a single data stream by the above two steps. In Step 1, excrescent points that are not inflexions are removed and missing inflexions are deduced. In Step 2, some inflexions may be removed since local inflexions may not be real inflexions from a global view. We present the two sub-algorithms: the DICMI algorithm and the Angle-DP algorithm as follows.

Detecting Inflexions and Computing Missing Inflexions (DICMI) Algorithm

We develop a novel DICMI algorithm to detect local inflexions and at the same time correct imprecise readings by computing missing inflexions based on local consecutive points.

We first discuss how many consecutive points are suitable to deduce missing inflexions. To determine whether a point, p_1, is a local turning point or not, we need at least one of p_1's

direct precursors, p_0, and one of p_1's direct successors, p_2. If the angle between $\overrightarrow{p_0p_1}$ and $\overrightarrow{p_1p_2}$ is greater than an angle threshold, then p_1 is a local turning point. If p_1 is missing, we need at least two of p_1's direct precursors: p_0 and $p_{0'}$, and two of p_1's direct successors: p_2 and $p_{2'}$ to determine whether p_1 is a local turning point by testing whether the angle between $\overrightarrow{p_0p_{0'}}$ and $\overrightarrow{p_2p_{2'}}$ is greater than an angle threshold. If we consider that there are more than two consecutive precursors (or successors) of p_1, which may not be strictly on a direct line, we can choose any two points on each side of the missing inflexion (e.g., p_1) to approximate the direct line within bounded deviation error; however we have not achieved better experimental results than four consecutive points based DICMI. Therefore, we choose four (consecutive local points) to deduce a missing inflexion. Note that Angle-DP algorithm (in Fig. IV.5) will check whether the local inflexions are also global inflexions and thus resist a local sharp direction change as shown in Fig. IV.6 (c).

The simplest case is that only one turning point is missing. But, even if multiple (> 1) consecutive turning points are missing, we can always deduce one missing turning point. This is reasonable and we explain it as follows. The main reason for missing turning points is that the objects move too fast and maybe pass one or multiple turning points in a constant sampling interval, since these consecutive turning points are too close to each other. When this happens, we only deduce one missing turning point to replace multiple missing turning points. This actually would not impact the common patterns we retrieved. As shown in Fig. IV.2, P_1 and $P_1^{'}$ are two consecutive missing inflexions, and V is a virtual inflexion deduced

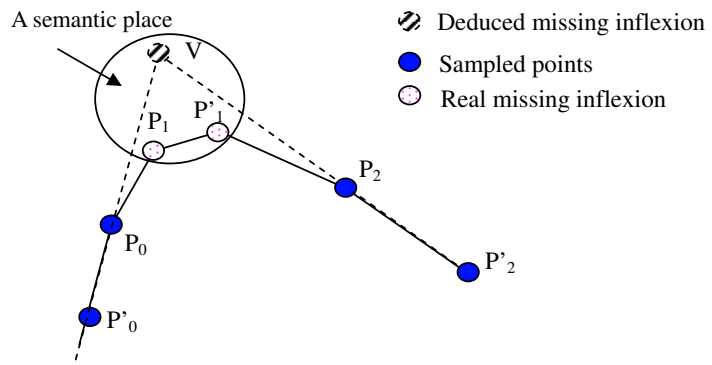


Figure IV.2: Multiple turning points are missing.

based on four consecutive points: P_0, $P_0^{'}$, P_2 and $P_2^{'}$. The area in the large circle is coarsely used to represent a semantic place, where close consecutive inflexions are grouped into the same cluster; detailed method to discover semantic place see Section III.3.1. If multiple consecutive turning points are missing, we use one virtual missing turning point (e.g. V) to represent them all (e.g. P_1 and $P_1^{'}$). So, deducing one missing inflexion is enough to help find a semantic place.

The DICMI algorithm is given in Algorithm 1. Let any four successive points on a trajectory be: A (p_1), B (p_2), C (p_3) and D (p_4). In Fig. IV.4 (a), M (p) is a missing inflexion between $\overrightarrow{AB}$ and $\overrightarrow{CD}$. $\alpha = \angle MBC$ and $\beta = \angle MCB$ are computed at *line 2b* in Algorithm 1. Suppose angle error tolerance is ε, the inflexions are detected at *line 2c*, *line 2d* and *line 2f*. In the other cases (at *line 2e*) as shown in Fig. IV.4 (b) and (c), no missing inflexion exists.

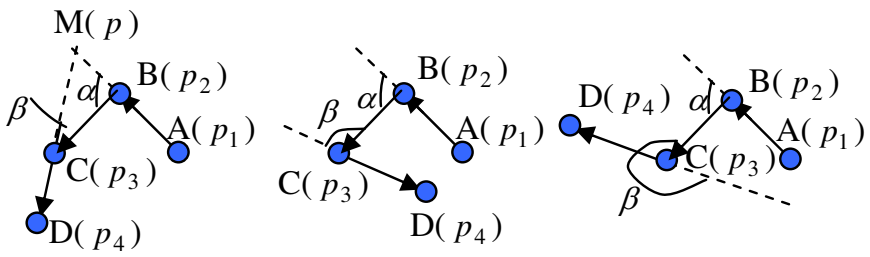
We define the angle between two consecutive direct line segments as follows:

Algorithm 1. Detecting Inflexions and Computing Missing Inflexions (DICMI). Given a sequence of points on a sensor stream: $P =< P_i , P_{i+1},..., P_j >$, where P_k ($e_k (x_k , y_k)$, t_k) ($k \in [i, j]$), the procedure DICMI(P , ε , *i*, *j*) detects the inflexions on P , while deducing the missing inflexions.

Procedure **DICMI** (P , ε , *i*, *j*)

1. P_i and P_j are inflexions;
2. **For** *k*= *i* **To** *j*-3 **Step** 2
   2a. $A= P_k$; $B= P_{k+1}$; $C= P_{k+2}$; $D= P_{k+3}$;
   2b. $\alpha = f(\overrightarrow{AB}, \overrightarrow{BC})$; $\beta = f(\overrightarrow{BC}, \overrightarrow{CD})$;
   2c. **If** ($\beta \le \varepsilon$ and $\alpha > \varepsilon$) B is an inflexion;
   2d. **Else If** ($\beta > \varepsilon$ and $\alpha \le \varepsilon$) C is an inflexion;
   2e. **Else If** ($\alpha \le \varepsilon$ and $\beta \le \varepsilon$) no inflexion;
   2f. **Else If** ($\alpha > \varepsilon$ and $\beta > \varepsilon$)
      2f Ⅰ **If** (($\alpha + \beta$)< π)
         M=*ComputeMissingInflexion* (α , β , *B*, *C*);
         /* *M*(P (e (*x*, *y*), *t*)) is a missing inflexion */
         If (*M*!=**Null**)
         M is an inflexion;
         Else
         B and *C* are inflexions;
      2f Ⅱ **Else**
         B and *C* are inflexions;

Figure IV.3: Detecting inflexions and computing missing inflexions.



(a) $\alpha + \beta < \pi$ (b) $\alpha + \beta \geq \pi$ (case 1) (c) $\alpha + \beta \geq \pi$ (case 2).

Figure IV.4: Relationship among four successive points on a data stream.

$$f(u, v) = arccos(\frac{u \bullet v}{|u| * |v|}), f \in [0, \pi]. \tag{IV.2.1}$$

where $\pi = 3.1415926$, u and v are two vectors, and $|u|$ and $|v|$ denote the lengths of the two vectors, respectively.

The main idea of the *computeMissingInflexion* procedure for computing $M(p(e, t))$ at line 2fI in Algorithm 1 is given as follows. In $\triangle BMC$, we have

$$\frac{|MB|}{sin\beta} = \frac{|MC|}{sin\alpha} = \frac{|BC|}{sin(\pi - \alpha - \beta)}. \tag{IV.2.2}$$

Let $a = \frac{|BC|}{sin(\pi-\alpha-\beta)}$, $e = (x, y)$ can be computed by 2-tuple functions as follows:

$$\begin{cases} (x - x_2)^2 + (y - y_2)^2 = a^2 sin^2\beta \\ \\ (x - x_3)^2 + (y - y_3)^2 = a^2 sin^2\alpha \end{cases} \tag{IV.2.3}$$

If there is no solution to Eq. IV.2.3, then M(p) does not exist. If there are two solutions, the one which satisfies $\overrightarrow{AB}$ and $\overrightarrow{BM}$ is on the same straight line as the result. The length between two successive point is computed by Euclidean distance. The average speed $\overrightarrow{v}$ can

Algorithm 2. Angle-DP. Given a sequence of locations P =< P_i , P_{i+1} ,..., P_j >,the procedure Angle-DP(P , α , i, j) simplifies the subsequence from P_i to P_j .

Procedure **Angle-DP**(P , α , i, j)

1. Find the vertex P_k , which satisfies $\theta = \pi - \angle P_i P_k P_j$ is the maximum angle;
2. **If** $\theta > \alpha$ **then** /* Split at P_k recursively. */

3a. **Angle-DP**(P , α , i, k);

3b. **Angle-DP**(P , α , k, j);

Else

4. Output($\overrightarrow{P_i P_j}$);/*Use $\overrightarrow{P_i P_j}$ in the approximation*/

Figure IV.5: Angle-DP algorithm.

be computed by

$$\overrightarrow{v} = \frac{|MB| + |MC|}{t_3 - t_2}. \tag{IV.2.4}$$

Thus, $t = t_2 + |MB|/\overrightarrow{v}$.

The DICMI algorithm is an online algorithm since it detects existing inflexions and computes missing inflexions based on four consecutive points on a single data stream.

Angle-DP Algorithm

We provide a novel Angle-DP algorithm that can be fully controlled by the angle as shown in Fig. IV.5 (Algorithm 2). We assume that the trajectories are simple without self-intersections as in DP.

We have presented the most related work, the DP algorithm [DP73], in Chapter II. The difference between Angle-DP and DP is that the split points in Angle-DP make the simplified trajectories in a bounded direction-based deviation error but split points in DP simplify trajectories in a bounded distance-based deviation error. We explain the concepts of direction-based deviation error and distance-based deviation error as follows. In Fig. IV.6 (a), if we use AB to simplify trajectory ACB, then $\theta = \pi - \angle ACB$ is a **direction-based deviation error** brought by simplification. Then $d_{lp}(C, AB)$ in Fig. IV.6 (a) is a **distance-based deviation error** brought by simplification, if we use AB to simplify trajectory ACB.

There are three advantages of Angle-DP. First, the output of the DICMI algorithm is the input of the Angle-DP algorithm. Because DICMI only detects inflexions locally, it cannot resist local sharp direction change as shown in Fig. IV.6 (c); Angle-DP can resolve this problem. Also, one limitation of using the DP algorithm to detect turning points is that some points may be falsely taken as turning points; an example is shown in Fig. IV.6 (a), where the direction-based deviation error is very small but the distance-based deviation error is great. Angle-DP can overcome this limitation of DP. Thirdly, another limitation of the DP algorithm is that it cannot choose the inflexions with smallest direction-based deviation error; an example is shown in Fig. IV.6 (b); we prove that Angle-DP can achieve inflexions with the smallest direction-based deviation error in Theorem 2.

THEOREM 2. *$Angle - DP$ can achieve inflexions more accurately than DP, by choosing the turning points with the smallest direction-based deviation errors.*

Proof. In Fig. IV.6(b), given a piece of trajectory around an inflexion: $P =< P_1, P_2, ..., P_7 >$, let $\theta_1 = \pi - \angle P_1P_4P_5$ and $\theta_2 = \pi - \angle P_7P_5P_4$, where $\pi = 3.1415926$. Let $l_1 = d_{lp}(P_4, P_1P_7)$ and $l_2 = d_{lp}(P_5, P_1P_7)$. Suppose $\theta_2 < \theta_1$. We determine which one: P_4 or P_5 is better to be a turning point. There are two cases: (1) if $l_2 < l_1$, then DP chooses the same inflexion as Angle-DP does, since $\theta_2 < \theta_1$; (2) if $l_2 \geq l_1$, then DP chooses an inflexion different from the one Angle-DP chooses. So, we only need to prove that Angle-DP chooses inflexions with the smallest direction-based deviation errors in the second case. In the second case, we focus on discussion of $l_2 \geq \ell \geq l_1$ (ℓ is the distance threshold) since other conditions can be discussed in the same way.

If we choose P_5 as a turning point, then we use P_1P_5 to simplify $P_1P_2P_3P_4P_5$ and the direction-based deviation error is θ_1. If we choose P_4 as a turning point, then we use P_7P_4 to simplify $P_7P_6P_5P_4$ and the direction-based deviation error is θ_2. Let ε be the angle threshold and there are three cases:

(1) If $\theta_2 < \varepsilon < \theta_1$, then Angle-DP detects P_4 as a turning point with the smallest direction-based deviation error: θ_2. DP detects P_5 as a turning point if $l_2 \geq \ell \geq l_1$ and the direction-based deviation error is θ_1 (not the smallest).

(2) If $\theta_2 < \theta_1 < \varepsilon$, then Angle-DP detects no turning point. DP detects P_5 as a turning point if $l_2 \geq \ell \geq l_1$ and thus θ_1 is the direction-based deviation error; but θ_1 is not the smallest direction-based deviation error. Also, this may be a false inflexion as shown in Fig. IV.6 (a).

(3) If $\varepsilon < \theta_2 < \theta_1$, then P_4 and P_5 are both turning points, and thus the direction-based deviation error equals zero (e.g. the smallest). DP also detects P_5 as a turning point if $l_2 \geq \ell \geq l_1$, but it cannot identify P_4 as a turning point; thus the direction-based deviation error equals θ_2, which is not the smallest, compared to zero.

□

According to Theorem 2, *Angle-DP* is better than DP for detecting turning points. The simplification of trajectories with the smallest direction-based deviation error is very important, since it can help cluster direct line segments precisely.

IV.3 Querying Common Patterns from Uncertain Data Streams Based on the CIRCE Method

In this section, we first introduce our CIRCE package based on the CIRCE method and then we present querying Longest Common Route patterns from data streams as a typical example to explain how to use the CIRCE package to query common patterns.

IV.3.1 The CIRCE Package

Based on the CIRCE method, we provide a CIRCE Package to help query common patterns. Fig. IV.7 shows that the CIRCE package changes a query on uncertain data streams into a query on exact semantic data streams. First, the uncertainty of the original data streams is removed in two steps: (1) the CIRCE core algorithm corrects the missing points and re-

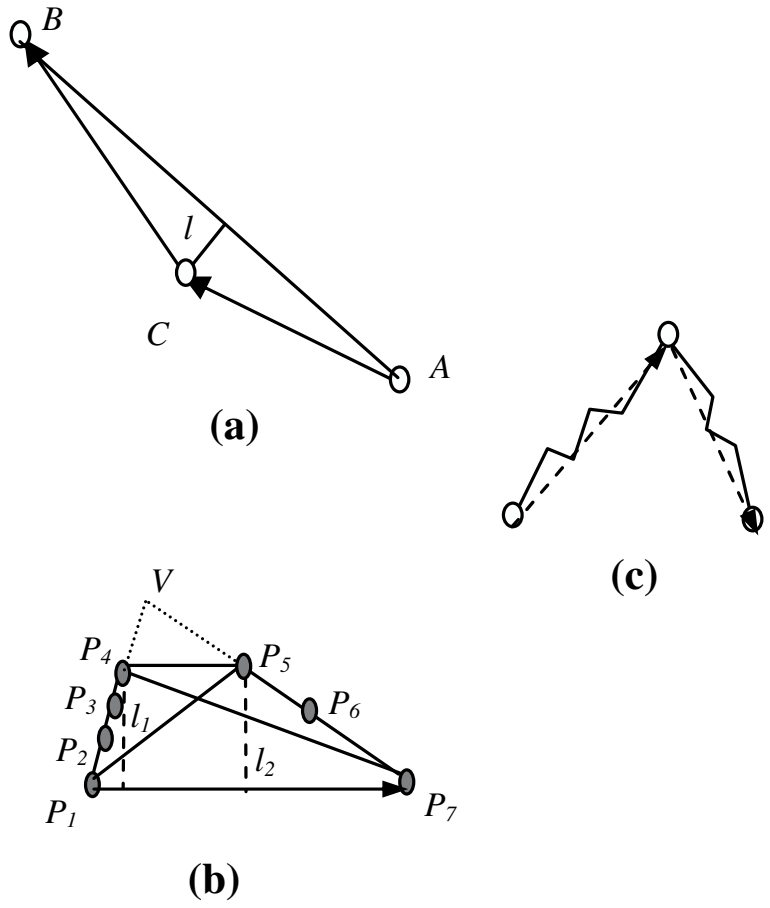


Figure IV.6: Three advantages of Angle-DP to discover turning points: (a) to avoid false turning point; (b) to choose inflexions with the smallest direction-based deviation error; (c) to resist local sharp direction change.

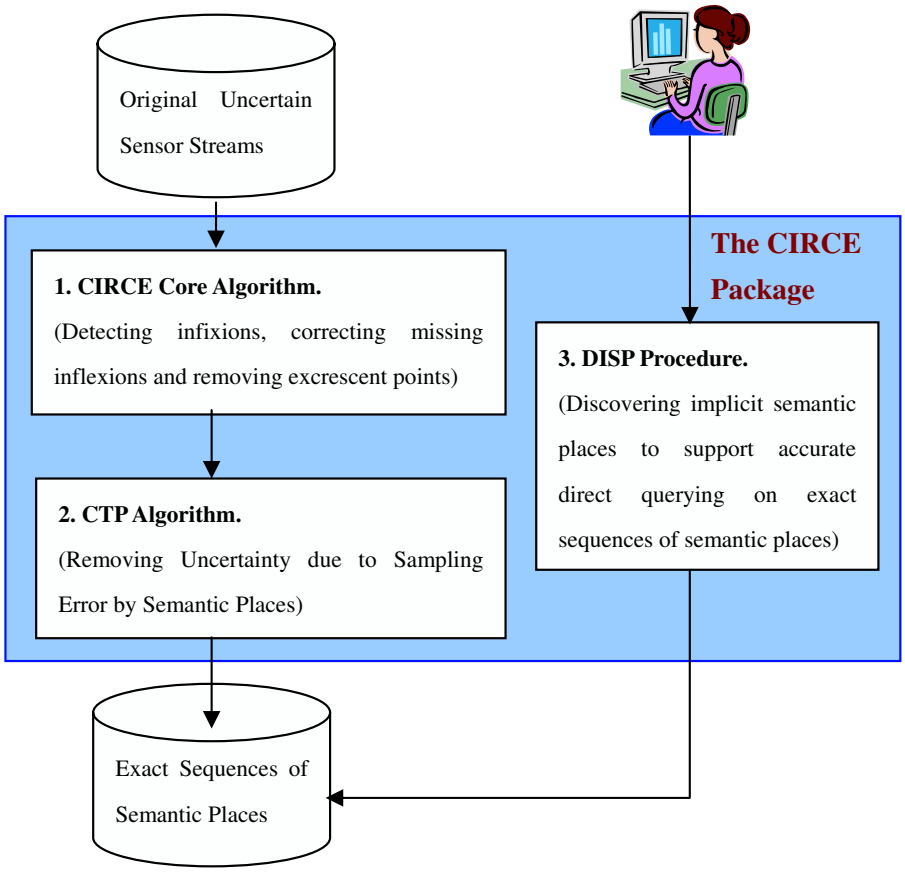


Figure IV.7: Querying common patterns from uncertain data streams supported by the CIRCE package.

moves excrescent points; (2) the discovering semantic places algorithm (or CTP algorithm in MicPasts) removes Uncertainty due to Sampling Errors by semantic places. So, the uncertain original data streams are preprocessed into exact data streams denoted by sequences of semantic places. Querying coarse common patterns can be achieved directly from exact semantic data streams, since many already known methods can satisfy this kind of exact query. But for a more accurate query, the DISP procedure in the CIRCE package is ready to refine the coarse query results.

IV.3.2 Application: Query of Longest Common Route Patterns

A typical application of our CIRCE method is to query LCR patterns directly from data streams. According to the definition of LCR patterns in Chapter III, an LCR pattern is a sequence of interest regions (e.g. semantic places and implicit semantic places). We explain semantic places and implicit semantic places in Fig. III.1. Given two trajectories in Fig. III.1 (a) and three trajectories in Fig. III.1 (b), where all the trajectories are simplified by the CIRCE core algorithm and let $min\_sup = 2$, then A, B, C and D are semantic places and E is an implicit semantic place in Fig. III.1 (a) and E, C, D and F are implicit semantic places in Fig. III.1 (b). We give a tag with semantic place value to each inflexion on a single data stream to help the query. Thus, an exact semantic data stream is a sequence of inflexions with a semantic tag. Note that implicit semantic places are not tagged. Therefore, executing the query of LCR patterns from the exact semantic data streams includes two steps: mining Longest Common Subsequences for retrieving coarse LCR patterns and refining the coarse result by the DISP procedure of the CIRCE package. For example, in Fig. III.1 (a), we can retrieve the longest common subsequence: $ABCD$ as a coarse LCR pattern and then refine this pattern by using the DISP procedure to achieve the exact LCR pattern: $ABCDE$. Another example is in Fig. III.1 (b), where no LCS is discovered but we find EF is a LCR pattern denoted by a sequence of implicit semantic places by using the DISP procedure.

IV.4 Performance Evaluations

In our experiments, we use the application of querying LCR patterns to validate the efficiency and accuracy of our CIRCE method.

IV.4.1 Experimental Setup

We use the same moving objects datasets and accuracy metrics in Table III 1 in Chapter III. Most of the parameters in the CIRCE method are the same as those in the LCRTurning algorithm in Table III 2. ε is specially for Angle-DP algorithm in the CIRCE method.

We now briefly describe the experimental setup for evaluating the CIRCE method. We also use the LCRTurning algorithm (mining algorithm for LCR patterns based on turning regions) that was in our work in [HZHD11] as a counterpart for our CIRCE method. The only difference between the CIRCE method and the LCRTurning algorithm is that we use the CIRCE core algorithm to replace the DP algorithm to compress each single data stream. The major aim of this experiment is to validate that the effectiveness of our CIRCE core algorithm is better than that of the DP algorithm to compress the data stream, thereby validating that the CIRCE method outperforms the LCRTurning algorithm. Before comparing the CIRCE method and the LCRTurning algorithm, we first adjust and find the optimal parameters of them. Several parameters for both methods are given in Table IV.1. Generally, we set $DL\_eps = eps = a$ (e.g. $a \times 30.92$ meters) and $MinPts = 2$. We change all the lengths into meters by timing 30.92 meters in the following experiment results; this is a

Table IV.1: Benchmarks based on information of intersections (Unit: meter).

min_sup	**Total pattern length**
2	24470
3	18687
4	14060
5	10705

min_sup=2	**Total pattern length**
Test1	24470
Test2	54509
Test3	81502
Test4	103613

simple way to achieve rough length represented by meters between locations (longitude, latitude). We have found in [HZHD11] that the greater the DL_Angle, the worse the total false rate. So we set a suitable lower value: $DL\_Angle = 5$ for all of the experiments. We studied $DP\lambda$ (λ=0.1, 0.5, 1.0) and $CIRCE\varepsilon$ (ε=0.05, 0.1, 0.2) to find the optimal λ and optimal ε that can achieve the best accuracy, while we also search for the optimal eps. Then, we conduct two scalable tests by changing min_sup and $datasize$ respectively to validate that the best CIRCE method (with optimal eps and ε) performs better than the best LCRTurning algorithm (with optimal eps and λ) in terms of accuracy and efficiency. Note that in the accuracy evaluation process, the correct length of an LCR pattern retrieved (e.g., TP) is the length of the LCR, where all the supporters (moving objects) must visit the same sequences of intersections.

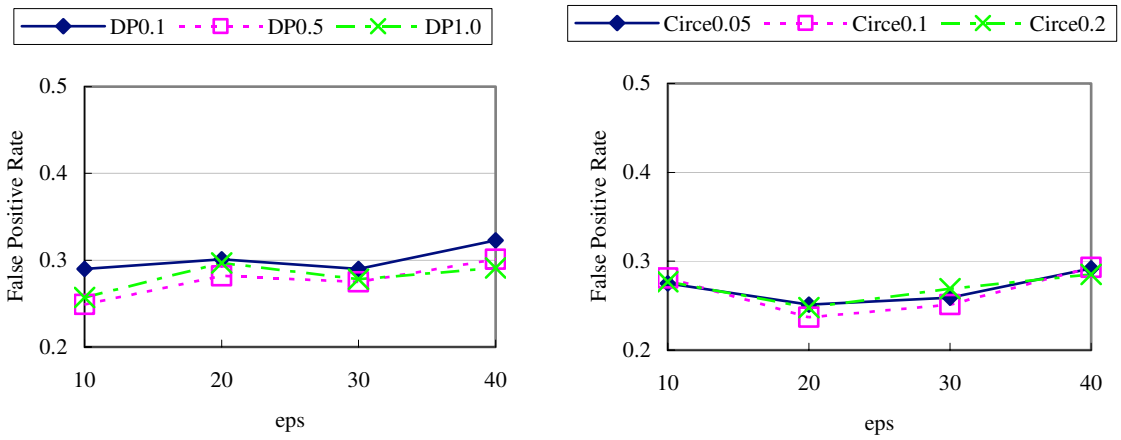
All experiments were also conducted on an IBM Laptop with Microsoft Windows XP,

Genuine Intel 1.83GHz CPU and 512MB main memory. We implemented the proposed algorithms mainly by using C++, and the algorithm based on the suffix tree to retrieve LCS was implemented by Java.

IV.4.2 Optimal Parameters

In this subsection, we choose optimal eps and optimal ε (or optimal λ) in the CIRCE method (or the LCRTurning algorithm) based on three accuracy metrics: false positive rate, false negative rate and total false rate, shown in Fig. IV.8, Fig. IV.9 and Fig. IV.10, respectively.

First, we analyze the results of LCRTurning based on DP to determine the optimal eps. We can see from Fig. IV.8 (a) that the optimal (lowest) false positive rate is achieved at $eps = 10$, but the optimal false negative rate is achieved at $eps = 30$ as shown in Fig. IV.9 (a). $eps = 10$ is not the optimal value, since LCRTurning achieves the worst false negative rate at $eps = 10$. Actually, LCRTurning performs best in terms of total false rate at $eps = 30$ as shown in Fig. IV.10 (a), thus, $eps = 30$ is the optimal value for LCRTurning. Then, we also determine the optimal eps based on the results of the CIRCE method. Fig. IV.8 (b) shows that the optimal false positive rate is achieved at $eps = 20$, but the optimal false negative rate is achieved at $eps = 40$ as shown in Fig. IV.9 (b). $eps = 40$ is not the optimal value, where the CIRCE method achieves worst false positive rate. $eps = 20$ is also not the optimal value, where the second worst false negative rate is achieved. It is interesting to find



(a) LCRTurning based on DP.

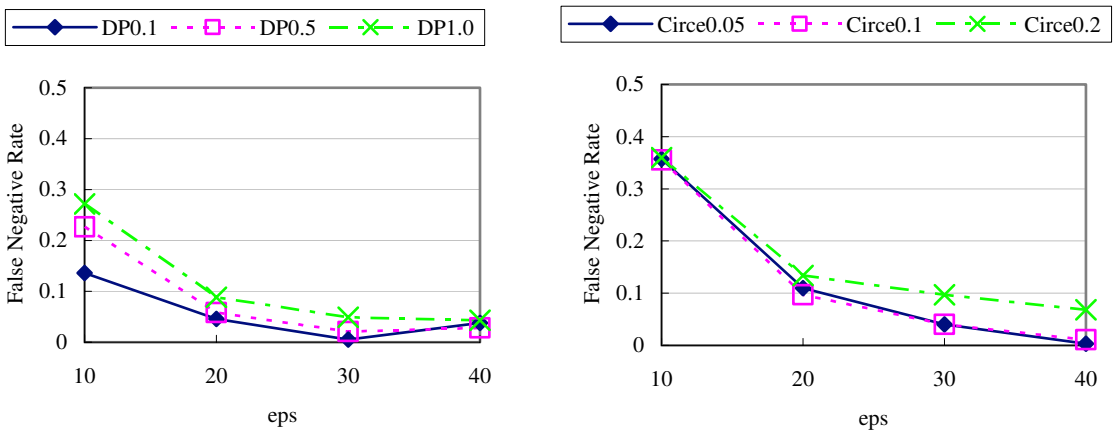
(b) The CIRCE Method.

Figure IV.8: False positive rate.

that the CIRCE method also achieves the optimal total false rate at $eps = 30$. Therefore, $eps = 30$ is the optimal value for both methods.

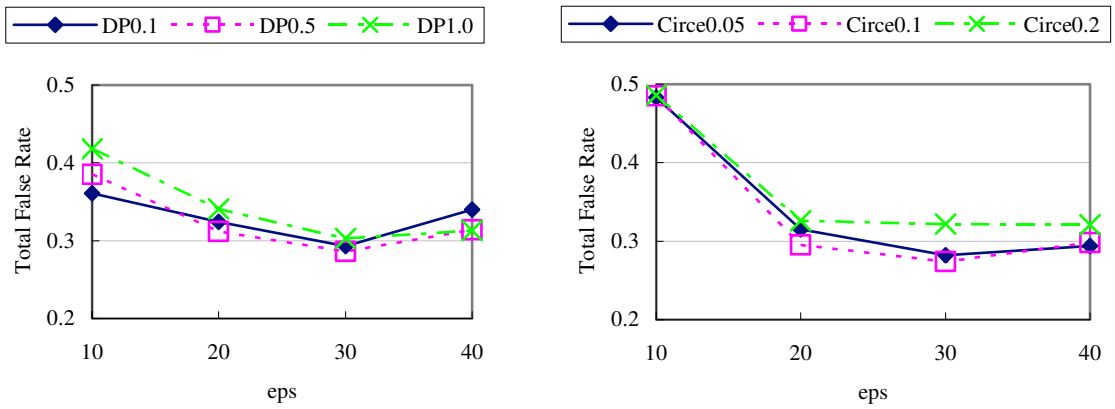
In the same way, we determine optimal λ and ε based on the lowest total false rates. Fig. IV.10 (a) and (b) show that $DP0.5$ and $CIRCE0.1$ achieve the optimal total false rates at $eps = 30$. It is reasonable that both optimal λ and ε are not at maximum or minimum values. The smaller the λ (or the ε) is (e.g. removing less points on data streams), the longer total correct pattern length is retrieved. That is, the lowest false negative rate is achieved at the lowest λ (or ε), as shown in Fig. IV.9. It is also very interesting that both $DP0.5$ and $CIRCE0.1$ perform better than other cases in terms of false positive rate as shown in Fig. IV.8.

We also analyze the time efficiency of $DP0.5$ and $CIRCE0.1$ in Fig. IV.11 (a) and (b). The trend is obvious that the lower the λ (or the ε), the more efficient is the LCRTurning



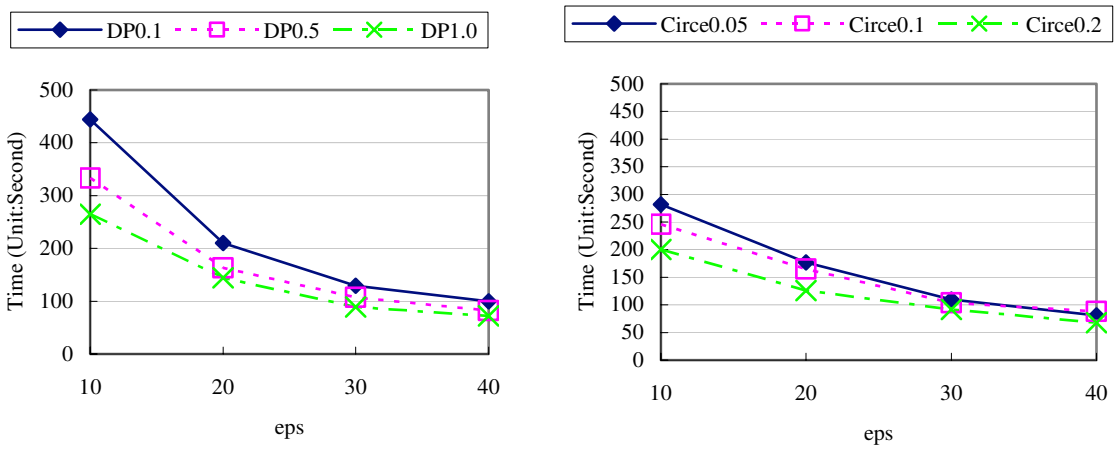
(a) LCRTurning based on DP. (b) The CIRCE Method.

Figure IV.9: False negative rate.



(a) LCRTurning based on DP. (b) The CIRCE Method.

Figure IV.10: Total false rate.



(a) LCRTurning based on DP.

(b) The CIRCE Method.

Figure IV.11: Time analysis.

algorithm (or the CIRCE method). Also, the greater is the eps, the more efficient are both methods. Since the execution time difference brought by λ (or the ε) is very small, both methods have a reasonable execution time at $eps = 30$.

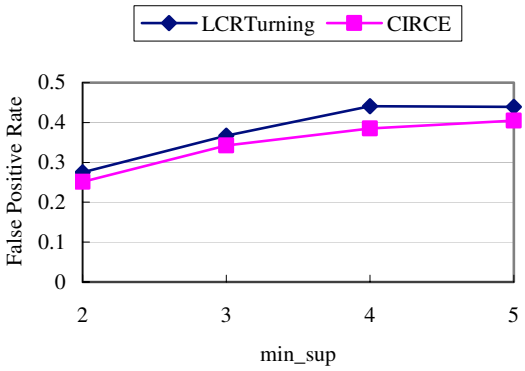
IV.4.3 Accuracy

In this subsection, we compare the CIRCE method (e.g., $CIRCE0.1$) with the LCRTurning algorithm (e.g., $DP0.5$) in two scalable tests (e.g. changing with min_sup and data sizes) and set $eps = 30$. In the min_sup scalable test, we use the dataset of test1. In the datasize scalable test, we let $min\_sup = 2$.

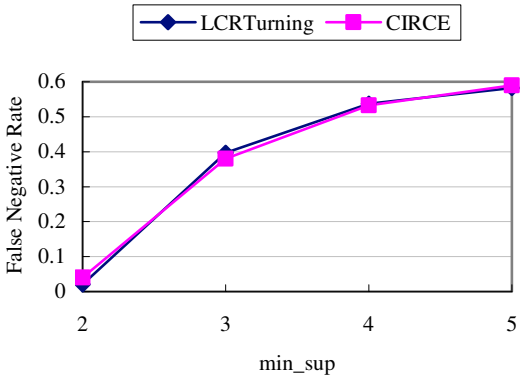
Fig. IV.12 (a) shows that CIRCE performs better than LCRTurning in terms of false positive rates in all cases (e.g. min_sup=2, 3, 4 and 5). At the same time, CIRCE achieves nearly the same false negative rate as LCRTurning in all cases of min_sup. That is, the

CIRCE method retrieves less wrong pattern length than the LCRTurning algorithm while returning the same correct pattern length. This is very important for users who generally do not want to look for correct results among a huge volume of rubbish. Besides, the CIRCE method excels the LCRTurning algorithm in terms of total false rates. Overall trends of all the three graphs in IV.12 show that the greater the min_sup, the greater are the three accuracy metrics. This means the query results become less accurate when the min_sup increases. In other words, we retrieve more wrong patterns and miss more real patterns when the min_sup is greater. This is inevitable in querying common patterns from uncertain data streams. Suppose the average correct ratio of a single data stream compressed by the CIRCE core algorithm (or DP algorithm) is $\mu = 90\%$. Let $min\_sup = 2$, the correct ratio of the retrieved common pattern is $(1-\mu)^{min\_sup} = 81\%$. But let $min\_sup = 5$, the correct ratio decreases to $(1 - \mu)^{min\_sup} = 59\%$. Therefore, the high total false rate does not mean that the correct ratio of the data streams compressed by CIRCE core algorithm (or DP algorithm) is low. Thus, we can compute the average correct ratio, η, achieved by CIRCE core (or DP) by using $\eta = \sqrt[min\_sup]{(1 - \varsigma)}$, where ς is a total false rate. If we take the numbers in Fig. IV.12 (c) as an example to compute the correct ratio, we may find that the correct ratio is not changed obviously with min_sup.

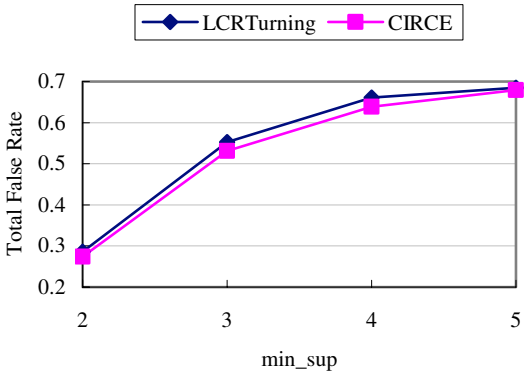
The overall trends in Fig. IV.13 are that when the data size is larger, the three accuracy metrics are slightly greater. But the changes are very small. Thus, our CIRCE method outperforms the LCRTurning algorithm in terms of false positive rates and total false rates



(a) False Positive Rate.



(b) False Negative Rate.



(c) Total False Rate.

Figure IV.12: Accuracy changed with min_sup.

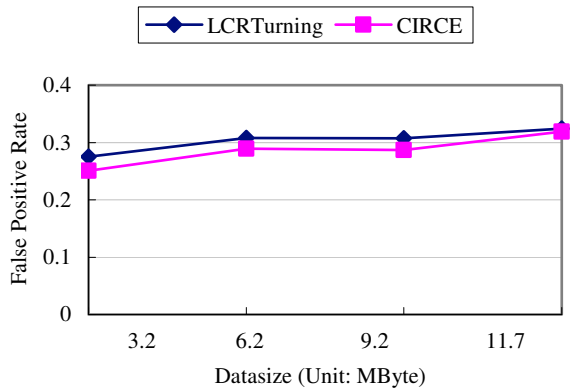
and performs nearly the same as the LCRTurning algorithm in false negative rates.

In summary, the two scalable tests validate that our CIRCE method achieves more accurate query results than the LCRTurning algorithm dose.

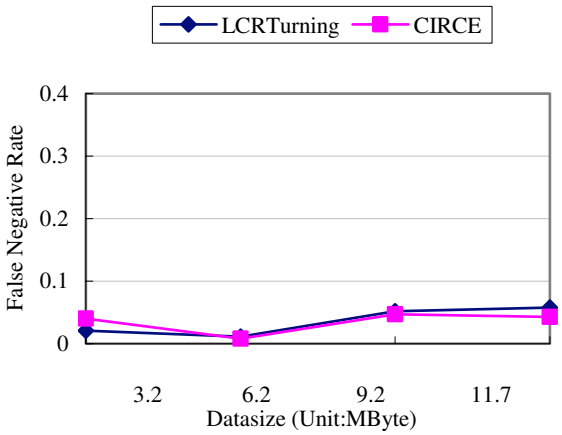
IV.4.4 Time Efficiency

In this subsection, we analyze the execution time of the two methods for different sizes of datasets. We set $min\_sup = 2$.

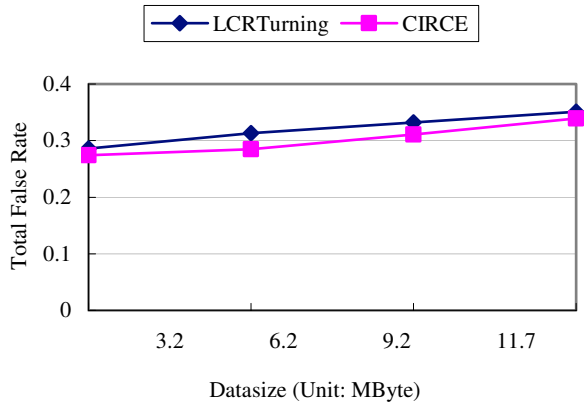
Fig. IV.14 (b) shows that the overall trend is that the CIRCE method executes noticeably faster than the LCRTurning algorithm. Moreover, the greater the data size, the better the CIRCE method performs than the LCRTurning algorithm. There are three procedures that consume the majority of time: CTP, DISP and MLCS (Ming LCS). We use DP to denote LCRTurning. The time distribution is shown in Fig. IV.14 (a). We can see from Fig. IV.14 (a) that CTP occupies the major part of time in the two methods, especially, when the data size is greater. For example, the time spent by DP_CTP is 96% of the total time spent by the LCRTurning algorithm and the time spent by $Circe_CTP$ is 86% of the total time spent by the CIRCE method at data size of 11.7MBytes. The reason that the CIRCE method is more efficient than the LCRTurning algorithm is that the CIRCE core algorithm can compress the original data streams by using less inflexions but more accurate inflexions than the DP algorithm. The advantage of the CIRCE core algorithm in correcting missing inflexions helps to reduce the number of inflexions needed to compress the streams.



(a) False Positive Rate.

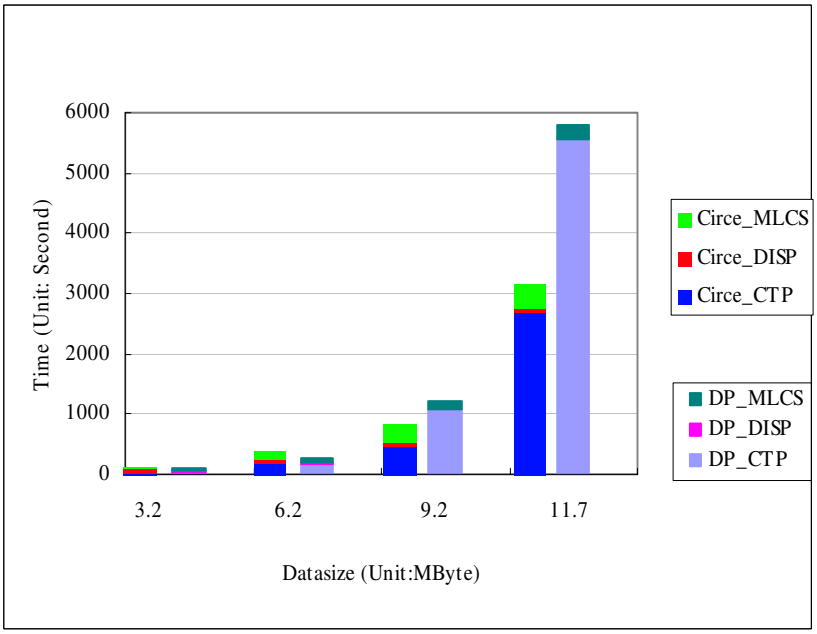


(b) False Negative Rate.

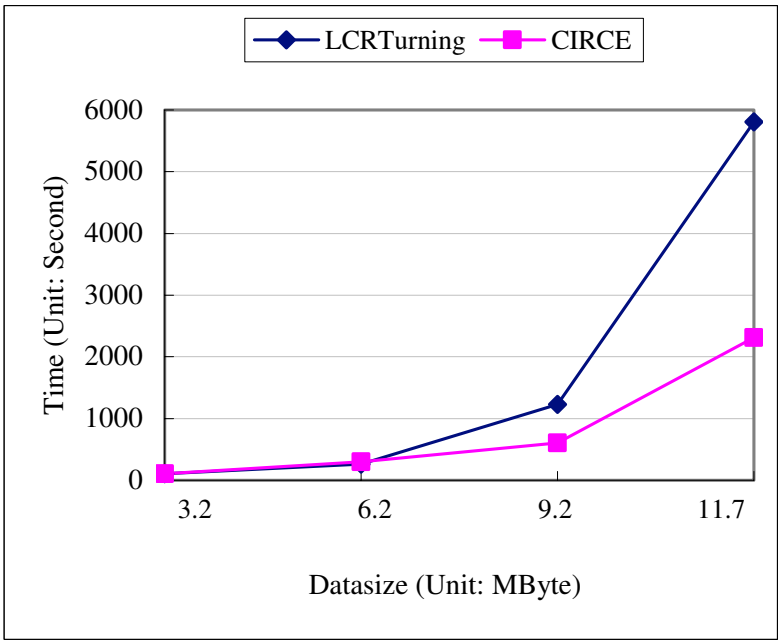


(c) Total False Rate.

Figure IV.13: Accuracy changed with data sizes.



(a) Time Distribution.



(b) Total Time.

Figure IV.14: Time efficiency.

IV.5 Summary

This chapter has extended the MicPasts method in Chapter III and proposed a novel CIRCE method to enhance the efficiency and accuracy of quering common patterns from uncertain data streams. The major contribution of the CIRCE method is the ability to tackle Uncertainty due to Sampling Error (SE Uncertainty) and Uncertainty due to Discrete Sampling (DS Uncertainty). Uncertainty due to Sampling Errors problem has been tackled by the MicPasts method. To resolve the Uncertainty due to Discrete Sampling problem, a novel CIRCE core algorithm was developed in the CIRCE method to correct the missing points while compressing the original data streams. The experimental study based on various sizes of data stream datasets validates that the CIRCE core algorithm is more efficient and more accurate than a counterpart DP algorithm to compress the data streams by using ID sequences. To resolve the Uncertainty due to Sampling Errors problem, the CIRCE method adopts the same technique in MicPasts method to summariz the original data streams by using inflexions, then groups close inflexions into the same cluster, and thus changes the uncertain data streams into exact sequences of cluster IDs. Particularly, to help query common patterns directly on exact ID sequences, the CIRCE method takes advantage of the DISP procedure in MicPasts to deduce the implicit common regions. Also, the application for querying Longest Common Route patterns validates the effectiveness of the CIRCE method.

Chapter V

The EOAFREE Method

"The road to excess leads to the palace of wisdom."

–William Blake (*The Proverbs of Hell*)

This chapter proposes a novel Exceptional Object Analysis for Finding Rare Environmental Events (EOAFREE) method. A typical application is to find water pollution events from water quality datasets. The major contribution of our EOAFREE method is that it proposes a general Improved Exceptional Object Analysis based on the Noises (IEOAN) algorithm to cluster objects, and then distinguishes those data objects (or data points) that cannot be grouped into any clusters as exceptional objects. Interestingly, opposite to the already known Principal Component Analysis (PCA) that ranks principal components, our IEOAN ranks exceptional objects. Another contribution is that it provides an approach to preprocess heterogeneous real world data through exploring domain knowledge. That is, it defines changes instead of the water quality data value itself as the input of IEOAN algorithm to remove the geographical differences between any two sites and the temporal differences between any

two years. The effectiveness of our EOAFREE method is demonstrated by a real world application - that is, to detect water pollution events from the water quality datasets of 93 sites distributed in 10 river basins in Victoria, Australia between 1975 and 2010.

The structure of this chapter is organized as follows. In Section 1, we present an overview of the EOAFREE method. In Section 2, we present a framework of the EOAFREE method. Then, we preprocess water quality data in Section 3 and Section 4. In Section 5, we provide novel algorithms for exceptional object analysis. In Section 6, we utilize a real world water quality dataset to validate our method. In Section 7, we summarize this chapter.

V.1 Overview of the Exceptional Object Analysis for Finding Rare Environmental Events (EOAFREE) Method

Rare event detection is very vital, since the earth's environment can be extremely violent and early warnings can impend natural disasters within the affected regions. For example, Hurricane Ike devastated the city of Galveston, Texas. Due to the influence of early detection and warning systems, the majority of the populace was safely evacuated prior to hurricane landfall [MSN]. In the same way, it is also necessary to continuously monitor river water by water utilities to detect purity and potential contaminants. The earth's environment changes with time, as a result of the forces of nature. It is the activity of humans (e.g., urbanization) that negatively impacts the environment and causes unusual environmental events for river water. Excluding the human factor, the environmental river water will eventually and pre-

dictably change, this change being generally global. We assume that the earth's environment will be resilient for hundreds and thousands of years, thus normal environmental events that exhibit common and predictable trends should be the norm. However, exceptional events brought about by humans should be the minority; otherwise, the environment will soon be totally destroyed.

Our goal in this chapter is to find rare environmental events (e.g. water pollution events) from water quality datasets; that is, to find when and where rare environmental events happen. We define rare environmental events as follows:

DEFINITION 13. ***A Rare Environmental Event*** *is an event that is unpredictable, based on the natural environment system and is different from common environmental changes.*

As they are unpredictable, rare environmental events should be detected as early as possible to minimize their negative impacts. In this chapter, we take water pollution detection from water bodies (e.g. rivers and lakes) as a typical example for detecting rare environmental events, since water bodies are one of the most important environment components.

To detect water pollution events, we need to collect data from the water. Thanks to modern advanced tools and sophisticated protocols [ATL+05], [U.S11], [TNGA09], we are able to closely monitor water bodies and collect water data. Water quality data is typical water data, which includes water quality-related physical parameters, such as PH, temperature, dissolved oxygen, total phosphate, nitrates, turbidity, total dissolved solids etc. We use water data to denote water quality data throughout this chapter. However, analyzing water data to

detect when and where the water pollution events happen faces heterogeneity problems.

The first type of heterogeneity problem relates to heterogeneous raw water data with various data qualities, that is, historical water data are provided by different organizations and collected by different equipment over a long historical period where the collecting technologies vary. For example, the collected water parameters are different from site to site and there is a different sampling frequency for different sites (or different months).

The second type of heterogeneity problem is that it is difficult to detect rare patterns from data with different water quality value ranges by using statistical analysis. For example, we cannot find rare events (e.g., water pollution) directly by using a specific threshold (e.g., "poor" water quality of the river). Instead, spatiotemporal variations of the water data are more useful. Also, we can 'learn' some abstract rules from the historical data but cannot directly achieve the normal values as the threshold for exception analysis. This means statistical analysis is invalid in detecting exceptional objects from the water quality data, which we explain as follows. Rare environmental events are generally unusual, relative to the normal patterns of behavior of an environmental body (e.g. a river) [KJBB09].

The simplest and the most straightforward approach to detect rare events is to explore exception analysis which identifies whether an attribute or measure value belongs to or does not belong to a specific list of values. One limitation of this approach is that it requires knowledge of the normal value or what is anomalous [MSN]. Although we can "learn" the normal values from historical data and then detect events that indicate departures from the

norm [KJBB09], the learnt knowledge may be out-of-date, since the environmental situation is changing with time. For example, it is unreasonable to use the average value of several sampling locations to denote the water quality of a whole river. Another limitation of the straightforward approach is that it is only valid when rare environmental events can directly lead to an abnormal value. But it is invalid when rare environmental events produce a normal value, since the whole environmental system (e.g. a river water system) can bear a pollution event for an extended period due to the following factors.

- First, daily water flow varies greatly in different seasons or in different rivers. A pollution event may not instantly change the water quality of the whole river that has a large amount of water flow in season, since the pollution may be flushed away; although a pollution event may persist for a long time and reduce the river water quality eventually.

- Second, different rivers have a various range of water quality from "excellent" to "very poor"; where water quality is higher, the pollution event detection is harder. For example, if the water quality of a river is "excellent", it may take a long time for a pollution event to change the water quality into a "poor" state. This means we are not aware of the harm of this pollution event from the beginning, for example, when a factory drains waste water into a nearby river for an extended period of time.

Meanwhile, already known data mining algorithms cannot directly apply to detect and

rank exceptional objects. There are two classes of rare event detection methods: application-specific and general rare event detections. However, already known application-specific rare event detection methods may not be suitable for detecting rare events from water quality data. Besides, clustering algorithms, such as DBSCAN [EKSX96], ROCK [GRS00], Shared Nearest Neighbor (SNN) clustering [ESK03] and Findout algorithm [YQLZ02], which do not force every data instance to belong to a cluster can be used to generate some data instances that could not be grouped into any cluster as rare events. But the disadvantage of such techniques is that they are not optimized to find rare patterns, since the main aim of the underlying clustering algorithm is to find clusters [CBK09].

In this chapter, we provide a novel EOAFREE method to satisfy our goal. Our EOAFREE method has the following advantages:

- It provides an approach to explore domain knowledge to pre-process real world data to remove the heterogeneous data differences brought by different organizations and different collected technologies. That is, we use a unified water quality index to denote water quality instead of multiple different water quality parameters.

- It defines water quality changes instead of water quality value itself as the input of the data mining algorithm in order to overcome the limitation of statistical analysis, since both the geographical differences between any two sites and the temporal differences between any two years are removed.

Site	Date	Water parameters						
		nitrates	total phosphate	temperature	turbidity	PH	dissolved oxygen	total dissolved solids
Avoca Site 1	June 2009	√	√	√	√	√	√	√
Avoca Site 1	Oct. 2003	√	√	√	√	√	√	×
Campaspe Site 2	Dec. 2007	×	×	√	√	√	√	√

Table V.1: Example of heterogeneous water data.

- It proposes two Exceptional Object Analysis (EOA) algorithms to cluster objects based on the objects' water quality changes, and then distinguishes those data objects (or data points) that cannot be grouped into any clusters as exceptional objects. These algorithms are general for rare (or exceptional) object detection. Interestingly, opposite to the already known Principal Component Analysis (PCA) that ranks principal components, our EOA algorithms rank exceptional objects. To the best of our knowledge, no related work exists to rank exceptional objects.

Also, the effectiveness of our EOAFREE method is demonstrated by a real world application - that is, to detect water pollution events from the water quality datasets of 93 sites distributed in 10 river basins in Victoria, Australia between 1975 and 2010 [Dat].

V.2 Framework of the EOAFREE Method

In this section, we present the framework of our proposed EOAFREE method.

The EOAFREE method comprises three steps:

- **Step 1: Preprocessing raw data by using domain knowledge.** We unify heterogeneous water quality datasets by summarizing multiple water quality parameters (e.g.

PH, temperature, dissolved oxygen, total phosphate, nitrates, turbidity, total dissolved solids etc.) into one Water Quality Index (WQI), a standard index for evaluating water quality, segment time series of water data by years and interpose missing data.

- **Step 2: Defining water quality forward changes.** We define water quality forward change of each water data object (or point), which describes the difference between the WQI value of the current month and the WQI value of the next consecutive month and the difference between the WQI value of the current month and the WQI value of the same month in the next consecutive year.

- **Step 3: Exceptional Object Analysis.** We cluster water data objects based on the differences of the objects' forward changes and distinguish those objects (or points) that cannot be grouped into any clusters as exceptional objects.

V.3 Preprocess Raw Water Data

In this section, we preprocess raw water data, including to unify heterogeneous data based on water semantics, that is, water quality index, and partition time series data.

V.3.1 Unify Heterogeneous Water Data Using Water Quality Index

The heterogeneous water data are produced due to historical factors, such as different collecting organizations, different types of the collecting equipment and different levels of collecting technologies and thus the data quality varies. We give an example of heteroge-

Factor	**Weight**
Dissolved oxygen	0.17
Fecal coliform	0.16
PH	0.11
Biochemical oxygen demand	0.11
Temperature change	0.10
Total phosphate	0.10
Nitrates	0.10
Turbidity	0.08
Total dissolved solids	0.07

Table V.2: Water Quality Factors and Weights.

neous water quality data in Table V.1. In Table V.1, the water data at Avoca Site 1 include 7 water parameters in June 2009 but 6 water parameters in Oct. 2003, missing data of total dissolved solids. Also, there are only 5 water parameters in the dataset of Campaspe Site 2 in Dec. 2007.

Thus, to unify heterogeneous water data, we adopt the method in [NSF] to calculate the water quality index. When test results from fewer than all nine measurements are available, the relative weights are preserved for each factor and the total is scaled so that the range remains 0 to 100. Note that, to ensure the data quality, we compute the WQI only when the number of water parameters is no less than 4; otherwise we set the WQI as "unknown" in this chapter. The water quality factors and weights are listed in Table V.2.

The 100-point index can be divided into several ranges, corresponding to the general descriptive terms shown in Table V.3.

An example of computing WQI based on multiple water parameters is shown in Table

Range	Quality
90-100	Excellent
70-90	Good
50-70	Medium
25-50	Poor
0-25	Very Poor

Table V.3: Water Quality Index Legend.

Site	Date	Water Parameters	Value	WQI
East Gippsland Cann river (west branch) @Weeragua (Site No. 221201)	June-2009	Dissolved oxygen	10.4	69 (M)
		PH	7	
		Nitrates	0.003	
		Temperature change	7.8	
		Total dissolved solids	51	
		Total phosphate	0.007	
		Turbidity	1.1	

Table V.4: Example of computing WQI.

V.4. There are 7 water parameters at Weeragua of East Gippsland Cann river (west branch) (Site 221201) sampled in June 2009. We first compute WQI of each original parameter value in Table V.4. Then we combine them into one WQI value through

$$WQI = \sum_{i=1}^{n} (\frac{W_i \times f_i(para_i)}{\sum W_i}) \tag{V.3.1}$$

where n is the number of parameters, W_i is the weight of parameter $para_i$ that can be achieved in Table V.2, f_i is the function (curve) of the i^{th} parameter according to the method in [NSF]. In this case, the WQI is 69, denoting 'Medium' quality.

Site Number	Year	Seasonal Partition (Months from Jan. to Dec.)											
		1	2	3	4	5	6	7	8	9	10	11	12
Avoca 408202	1994	55	55	55	58	57	57	60	58	57	56	55	53
	1995	52	76	49	49	-	58	46	57	55	56	46	57
	1996	77	53	55	55	58	59	58	56	53	54	55	54
	1997	53	54	54	48	58	58	48	50	54	61	61	61

Note: "-" denotes unknown/missing.

Table V.5: Example of seasonal partitions.

V.3.2 Seasonally Partitioned Time Series of Water Data

Since the environment changes with the seasons, we partition the time series of water data by "year".

DEFINITION 14. ***A Seasonal Partition of Water Data*** *is a sequence of 12 WQIs for the 12 months from January to December in a year at a site.*

For example, a river basin, Avoca, comprises 4 sites and there is a sequence of 24 seasonal partitions for each site from 1976 to 2009. An example of season partitions at site 408202 in the Avoca river basin is shown in Table V.5.

V.4 Water Quality Changes and Exceptional Objects

In this section, we define water quality changes based on the result of preprocessed data on seasonal partitions of water data without data missing. We have discussed in Section I that the values of WQIs at different sites are obviously different: some sites show "Excellent" water quality at most times while other sites generally have "Poor" water quality. The same

problem happens to the water data over two different periods of time. Thus, it is difficult to compare the water data at different sites and at different times. To overcome this difficulty and to detect an exceptional event at the time it occurs, we define two types of water quality changes. The first is to describe the instant "forward" change trend of the WQI value in a given month as follows:

DEFINITION 15. ***Water Quality Change*** *is a 2D value: (H-change, V-change), where H-change is the difference between the WQI value of the current month and the WQI value of the next consecutive month and V-change is the difference between the WQI value of the current month and the WQI value of the same month in the next consecutive year.*

The advantage of water quality changes is that the geographical differences between any two sites and the temporal differences between any two years have been removed and thus we can discover common (or frequent) patterns among different data sequences at different sites for different years. Moreover, water quality changes are useful to help detect events, since water pollution events happen at the time when water quality is being changed.

Based on the concept of water quality change, we define exceptional objects as follows:

DEFINITION 16. ***Similarity***. *Given two water quality changes:* $p(H_1, V_1)$ *and* $q(H_2, V_2)$*, the difference between two water quality changes,* ϖ*, is given by*

$$\varpi = \alpha|H_1 - H_2| + \beta|V_1 - V_2|(\alpha + \beta = 1), \tag{V.4.1}$$

where α *is the weight of H-change and* β *is the weight of V-change. Generally, we set*

$\alpha = \beta = 0.5$. *If* $\varpi \leq \lambda$, *where* λ *is a given threshold, then the two objects have similarity (are similar).*

DEFINITION 17. ***Strict Exceptional Objects***. *Given an object set* R, *a strict exceptional object* $x(x \in R)$ *is not similar to any other objects in* R.

DEFINITION 18. ***Relaxed Exceptional Objects***. *Given an object set R and a relaxed exceptional object set* R_e, *a relaxed exceptional object* $x(x \in R)$ *is not similar to any other objects in* $R - R_e$.

We give some properties of exceptional objects as follows:

LEMMA 1. *Given two strict exceptional objects:* A *and* B, A *is not similar to* B.

LEMMA 2. *If object* A *is a strict exceptional object, object* B *is similar to object* C *and object* B *is not similar to* A, *then object* A *is not similar to object* C.

Water Pollution Events often happen at the points where those exceptional objects whose H-change and V-change are both negative are detected. Taking advantage of noises that cannot be grouped into any clusters in the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [EKSX96], [HHD08a], [HHD08b], we develop an innovative Exceptional Object Analysis (EOA) algorithm to rank the exceptional objects. Not only the cited DBSCAN [EKSX96], but also other algorithms efficient in finding notices, such as ROCK [GRS00], Shared Nearest Neighbor (SNN) clustering [ESK03] and Findout algorithm [YQLZ02], can be applied in our EOA algorithm.

V.5 Exceptional Object Analysis

In this section, we first introduce a novel EOAN algorithm in Section 5.1, then present an improved EOAN algorithm in Section 5.2 and discuss the extendibility of our proposed algorithms in Section 5.3.

V.5.1 Exceptional Objects Analysis based on Noises (EOAN)

We develop a Detecting Exceptional Objects based on Noises (DEON) algorithm shown in Algorithm 1 by modifying an implemented version of DBSCAN clustering in our work [HHD08a].

Algorithm 1. Detecting Exceptional Objects based on Noises

Function DEON (R, eps, MinPts, R_e) **Input**: An object set, *R*, with object data format: (*siteNumber, date, H-change, V-change*) *eps*, *MinPts* **Output**: An exceptional object set, R_e .
Step 1. Build neighbour lists of each object. The neighbours of object *q* (*siteNumber, date, H-change, V-change*) must satisfy the criteria that the locations of neighbours are in the neighbourhood circle area with (*H-change, V-change*) as the centre and *eps* as the radius. Initialize all objects in R as "unused". **Step 2.** Build a set of core objects, *I*. The object which has greater than *MinPts* neighbors is marked as a core object. **Step 3.** For each unused core object *p*, put *p* and *p*'s neighbors into cluster class_ id and mark the object p as "used". Any core object *r* in cluster *class_ id* will recruit *r*'s neighbors into cluster *class_ id* and the used objects are marked. **Step 4.** Exceptional objects (or noises) are those objects that are not used.

Figure V.1: Detecting Exceptional Objects based on Noises.

Generally, we set $MinPts = 1$. The algorithm complexity of DBSCAN is $O(n * lnn)$

[EKSX96] and thus the algorithm complexity of DEON is $O(n * lnn)$, where $n = |R|$.

DEON is a basic algorithm to detect exceptional objects. We can recursively run DEON by setting different values of eps to produce different classes of exceptional objects and the Exceptional Objects Analysis based on Noises (EOAN) algorithm is shown in Algorithm 2.

Algorithm 2 shows that DEON repeats several times and thus the algorithm complexity of EOAN is $O(m * n * lnn)$, where $n = |W|$ and m is the number of repeat times. An example result of EOAN is shown in Fig. V.3. The original dataset, W, are all of the objects shown in Fig. V.3. We set rank 1 to original data and the rank 1 dataset is $R1 = R$. Then we discover Rank 2- Rank 7 datasets, denoted by $R2$ - $R7$, respectively. We observe that some objects are given more than one rank and we generally use the highest rank to distinguish the data. Finally, we group objects into different clusters by their highest rank. Thus, every object belongs to a cluster. In Fig. V.3, we may be interested in Rank 3 - Rank 7 objects (marked in the figure) and thus neglect the remaining Rank 1 and Rank 2 objects. Note that two objects may be at the same location in Fig. V.3.

V.5.2 Improved EOAN Algorithm

We improve Algorithm 2 and develop a new Improved EOAN algorithm in Algorithm 3. In the improved EOAN algorithm, although the DEON function is also repeated several times, the number of objects that are input to the DEON function is reduced greatly using Line 5 in Algorithm 3. Initially, all the objects in W are set to rank 1. After the first running

Algorithm 2. Exceptional Objects Analysis based on Noises (EOAN)

Function EOAN (W) **Input**: An object set, *W*, object $x \in W$ having a data format: (*siteNumber, date,* *H-change, V-change, rank*), where *x.rank*=1 **Output**: Each object $x \in W$ is set a rank.	
1	*R*=*W*; *eps*=2; *MinPts=1*; *Rank*=2;
2	**While** ($\|R_e\|$>0)// $\|R_e\|$ is the number of objects in the set R_e.
3	**DEON** (*R, eps, MinPts,* R_e);
4	**For** each object $x \in R_e$ *x.rank*=*Rank*; **End For**
5	*eps*=*eps*+λ ; // λ is the step length.
6	*Rank*++;
7	**End While**

Figure V.2: Exceptional Objects Analysis based on Noises (EOAN).

of DEON, $R = W$ and thus each object $x \in R_e$ is set to rank 2. Then, in the second running of DEON, it only clusters the noise set R_e produced by the first running of DEON. If a noise, p, produced in the second time is not similar to any objects with rank 1, then p is set to rank 3. The procedure keeps running in this way until $|R| = 0$ when no noise is produced any more. The improved EOAN algorithm achieves the same result as that produced by EOAN but is far more efficient than the EOAN algorithm.

V.5.3 Discussions

The above algorithms are general to find relaxed exceptional objects (see Definition 18). According to Definition 17, we can use Step 1 in Algorithm 1 to find strict exceptional

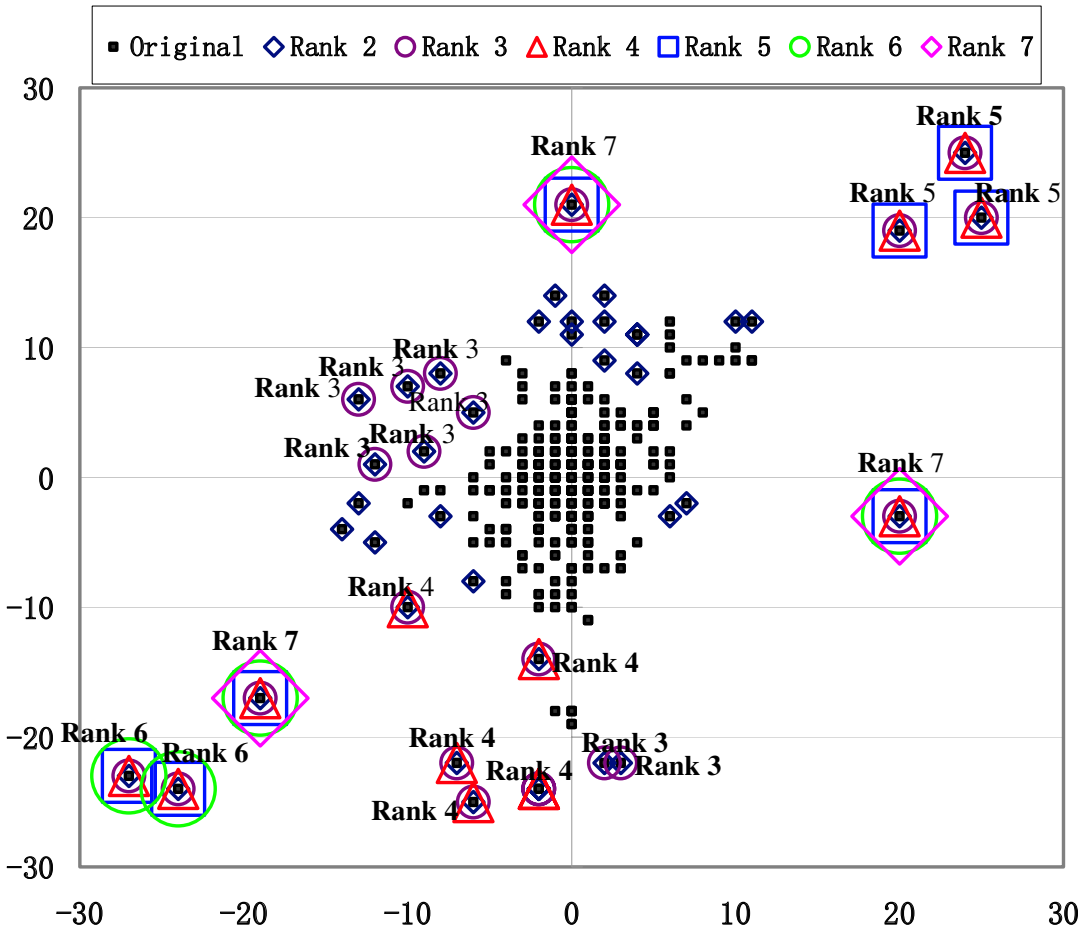


Figure V.3: An Example Result of EOAN.

Algorithm 3. Improved EOAN (IEOAN)

Function IEOAN (W) **Input**: An object set, W, object $x \in W$ having a data format: (*siteNumber, date, H-change, V-change, rank*). Set initial *rank*=1. **Output**: Each object $x \in W$ is set a rank.		
1	$R=W$; *eps*=2; *MinPts=1*; *Rank*=2;	
2	**While** ($\|R\|>0$)// $\|R\|$ is the number of objects in the set R.	
3	**DEON** (R, *eps, MinPts,* R_e);	
4	**For** each object $x \in R_e$	
	4.1	**If** ($R=W$) **then**
	4.2	*x.rank=Rank*;
	4.3	**Else If** not ($\exists$ p $\in$ x.neighborlist and p.cluster_id $\neq$ 0) **then**
	4.4	*x.rank=Rank*;
	End For	
5	$R= R_e$	
6	*eps=eps*+ λ ; // λ is the step length.	
7	*Rank*++;	
8	**End While**	

Figure V.4: Improved EOAN (IEOAN).

objects that have no neighbors. We assume that the common trend is unknown. This means the objects on the border may not be exceptional objects, though it is true in some cases.

V.6 Experimental Study

V.6.1 Application Background and Motivations

Work in [Dat] and [oSE04] reports the environment quality of the river basins in Victoria, Australia based on hydrology, physical form, streamside zone, water quality and aquatic life. Our work is different, since we focus on analyzing water quality and detecting water quality related events, such as pollution events, from water quality data. In addition, we consider more detailed water quality data, for example, at least 3 sites in each river basin, to learn about water quality-related common patterns for detecting pollution events at the beginning of when they happen.

We selected water quality datasets of 93 sites at 10 river basins: Avoca, Barwon, Broken, Bunyip, Campaspe, Corangamite, East Gippsland, Glenelg, Goulburn and KiewWa in Victoria, Australia between 1976 and 2010 from the Victorian water resources data warehouse [Dat]. The distribution of the 10 river basins is shown in Fig. V.5. There are a total of 7 water quality parameters: PH, temperature, dissolved oxygen, total phosphate, nitrates, turbidity and total dissolved solids in the datasets.

In this application, the raw water quality data are heterogeneous, provided by different organizations and have different data quality. There are less than 7 water quality parameters

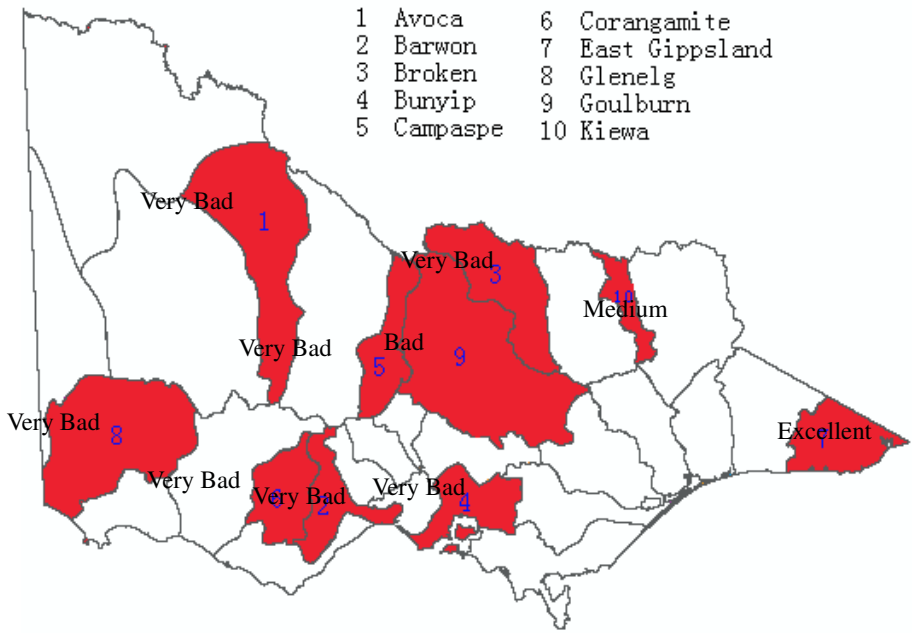


Figure V.5: Selected 10 River Basins in Victoria, Australia. The environmental quality (from excellent to very bad) at each basin is evaluated in 2004 ISC Report [oSE04].

available in some years for a site, with various numbers of collection times for different months. Therefore, we first preprocess water quality data using the method described in Section 3 as follows. We compute the water quality index (WQI) for a month by using 4-7 water quality parameters to ensure the data quality, since not all the sites collected all of the 7 water quality parameters. Then we segmented the time series water quality data into seasonal partitions by year and thus one seasonal partition is a sequence of 12 WQIs for the 12 months from January to December in a year.

We briefly analyze the goals of this application. The first goal of this application is to study whether water quality change (Definition 15) is effective to detect water pollution events. Two proposed EOA algorithms aim to discover exceptional objects by using the

unsupervised clustering method, since no standard evaluation criteria can distinguish exceptional objects from normal ones. Also, the efficiency of the data mining algorithm is critical, since the target datasets that comprise more than 30 years' water quality data for 93 sites are very huge. Therefore, another goal of this application is to contrast the efficiency of the proposed EOAN and the Improved EOAN algorithms.

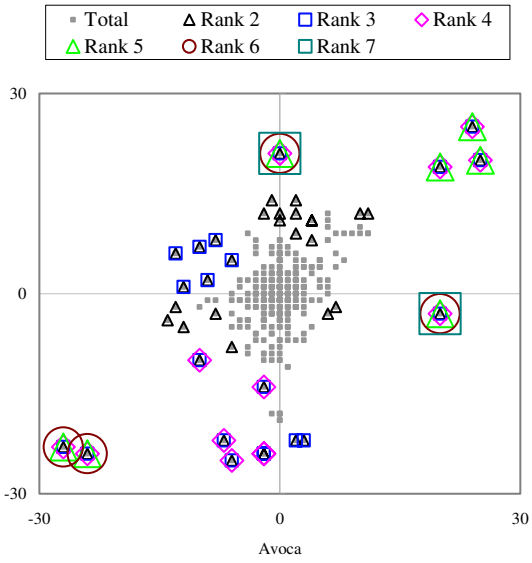
We implemented all the experimental algorithms in Microsoft VC++6.0 and Excel (VB-script), which were run on a Windows XP 1.83GHZ cpu with 512 Mbyte of RAM. We set $MinPts = 1$ to find the strict exceptional objects in the experiment and then change eps in the range [U.S11], [KJBB09] to find strict exceptional objects with rank 2 to rank 7, respectively. To find the relaxed exceptional objects, we set $MinPts \geq 2$.

V.6.2 Exceptional Objects Ranking

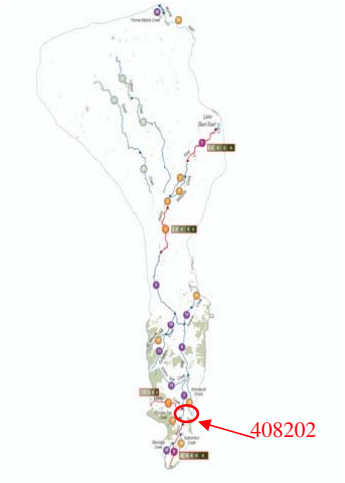
We plot all of the points (H-change, V-change) of water quality objects on a 2D plane and build a water quality change map for each river basin. Fig. V.6 (a) -V.15 (a) show water quality change maps, on which exceptional objects are marked. A common trend is that the exceptional objects are always located at the borders of the water quality change maps of the 10 river basins, while most are normal objects and located at the center of the maps. Also, we observed from Fig. V.6 (a) -V.15 (a) that the higher the rank of exceptional objects, the farther they are from the map centers and thus the exceptional objects with the highest rank (Rank 7) are the farthest away from the map centers. Fig. V.6 (c) -V.15 (c) show the details

of the exceptional objects, which point out where and when the rare events happen. Note that both pollution events and water quality improvement events are included. Also, we mark the rare event locations on the Basin Maps in Fig. V.6 (b) -V.15 (b) according to the site name in Fig. V.6 (c) -V.15 (c). The basin maps are provided by the 2004 ISC Report [oSE04] in [Dat] and the water qualities of each area in 2004 have been marked.

We then validate the effectiveness of the EOAFREE method by explaining the detected exceptional objects in 10 river basins based on the basins' environmental quality map provided in the ISC 2004 report. Note that the metric (H-change, V-change) denotes the forward change trends. For example, the first row in Fig. V.6 (c) means that WQI of March 1994 is 55, the same as the WQI of February 1993 since H-change equals 0; while WQI of February 1995 is 76, since V-change is 21. Fig. V.6 (a) shows an exceptional object distribution map from Rank 2 to Rank 7. Although we only list the details of rank 7 objects in Fig. V.6 (c), other ranked exceptional objects can be used for many purposes according to users' requirements. For example, if the user is interested in analyzing pollution factors, then two rank 6 exceptional objects at the bottom left corner are more important than other objects. Also, if the users focus on looking for the reasons for water quality improvement, the three rank 5 exceptional objects at top right corner are more useful. We can see from Fig. V.6 (c) that two rare events (rank 7) happened at the same site: Avoca River @ Amphitheatre (408202) in two different months: Feb. 1994, Dec. 1995 and Dec. 2009 in Avoca Basin. Also, this site is marked on the map in Fig. V.6 (b).



(a) Exceptional Object Analysis.



(b) Avoca Basin Map in ISC 2004 Report.

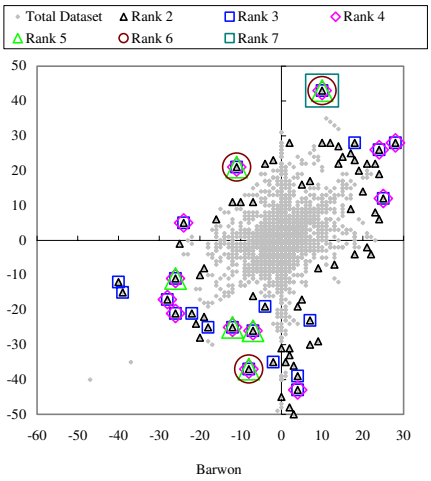
Basin	Site	Site Name	Time	H-change	V-change	WQI
Avoca	408202	Avoca River @ Amphitheatre	Feb-94	0	21	55
	408202	Avoca River @ Amphitheatre	Dec-95	20	-3	57

(c) Rare Events (Where and When).

Figure V.6: Avoca.

Another feature of the exceptional object distribution map is to show different change trends, for example, to answer questions of which change range is greater: the V-change or H-change and which trend is the major: the negative change or the positive change. In Fig. V.7 (a), we can see that the vertical change range [-50, 50] is greater than the horizonal change range [-40, 30]; this means the water quality changes are greater in consecutive years than in consecutive months in the same year. But a different change trend is shown in Fig. V.8 (a), where both H-change and V-change are in the range of [-40, 50]. For the second question, we can see in Fig. V.7 (a) that 12 exceptional objects from rank 2 to rank 7 are negative since they are located at the bottom left corner while only 5 exceptional objects are positive at the top right corner. But Fig. V.8 (a) shows an opposite trend: 8 exceptional objects are at the bottom left corner while 12 exceptional objects are at the top right corner. This trend also is evidenced by rank 7 exceptional objects listed in Fig. V.8 (c) with 5 positive events (both H-change and V-change are positive) and only 1 negative event (both H-change and V-change are negative). We also can observe the spatial distribution of the four sites where rare events happened in V.8 (b). The fact that they are scattered along the Broken River while they happened from 1978 to 2009 may denote the whole trend of the Broken River basin in the past years is positive.

There are 5 river basins that have a balanced negative change and positive change in the exceptional object distribution maps: Bunyip in Fig. V.9 (a), Campaspe in Fig. V.10 (a), East Gippsland in Fig. V.12 (a), Glenelg in Fig. V.13 (a) and Goulburn in V.14 (a). The other



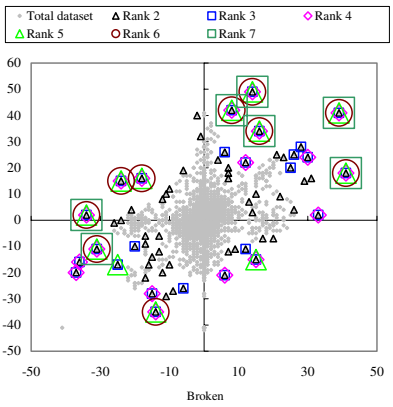
(a) Exceptional Object Analysis.

(b) Barwon Basin Map in ISC 2004 Report.

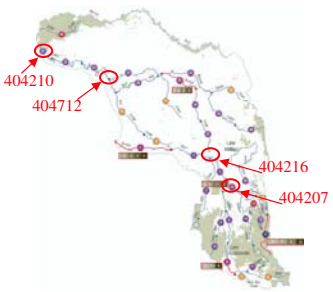
Basin	Site	Site Name	Time	H-change	V-change	WQI
Barwon	233224	Barwon River @ Ricketts Marsh	Apr-93	10	43	38

(c) Rare Events (Where and When).

Figure V.7: Barwon.



(a) Exceptional Object Analysis.

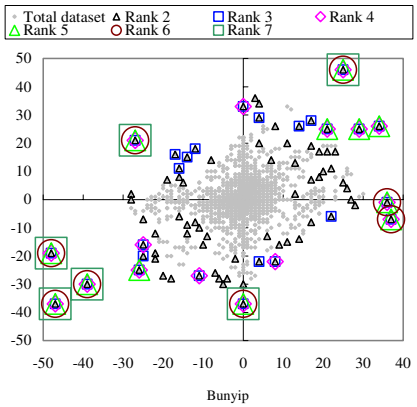


(b) Broken Basin Map in ISC 2004 Report.

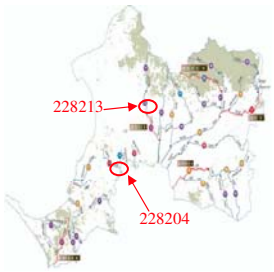
Basin	Site	Site Name	Time	H-change	V-change	WQI
Broken	404207	Holland Creek @ Kelfeera	Dec-78	41	18	39
	404210	Broken Creek @ Rices Weir (Affra Unit)	May-81	-31	-11	69
	404210	Broken Creek @ Rices Weir (Affra Unit)	Feb-88	39	41	29
	404210	Broken Creek @ Rices Weir (Affra Unit)	Mar-88	-34	2	68
	404210	Broken Creek @ Rices Weir (Affra Unit)	May-88	16	34	36
	404216	Broken River @ Goorambat (Casey Weir H. Gauge)	Jan-09	14	49	44
	404712	Muckatah Depression Drain @ Numurkah Outfall	Mar-09	8	42	52

(c) Rare Events (Where and When).

Figure V.8: Broken.



(a) Exceptional Object Analysis.



(b) Bunyip Basin Map in ISC 2004 Report.

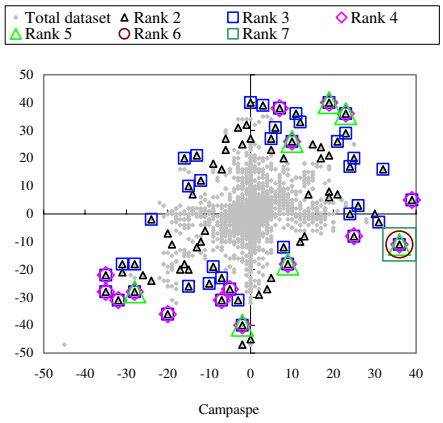
Basin	Site	Site Name	Time	H-change	V-change	WQI
Bunyip	228204	Dandenong Creek @ Dandenong	Feb-86	-48	-19	74
	228204	Dandenong Creek @ Dandenong	Jan-87	25	46	30
	228204	Dandenong Creek @ Dandenong	Mar-87	-27	21	55
	228204	Dandenong Creek @ Dandenong	Jan-88	0	-37	76
	228204	Dandenong Creek @ Dandenong	Mar-88	-47	-37	76
	228213	Bunyip River @ Iona	Sep-78	-39	-30	75

(c) Rare Events (Where and When).

Figure V.9: Bunyip.

two: Corangamite in Fig. V.11 (a) and Kiewa Fig. V.15 (a) have the same trend as Barwon in Fig. V.7 (a) in that negative change is the majority.

Table V.6 provides the numeric summary of exceptional objects in Fig. V.6-V.15. Another interesting discovery is while the total number of objects is greater, the percentage of exceptional objects is smaller. In Table V.6, for example, only 1.3% of the Goulburn dataset were exceptional objects, while, 4.7% of the Avoca dataset were exceptional objects. We also notice that there are more exceptional objects located at the top right and bottom left corners of each map. This means H-change and V-change of the exceptional objects are often of the common trend: either both are positive or both are negative. To satisfy our goal to find water pollution events, we are interested in those exceptional objects whose H-change



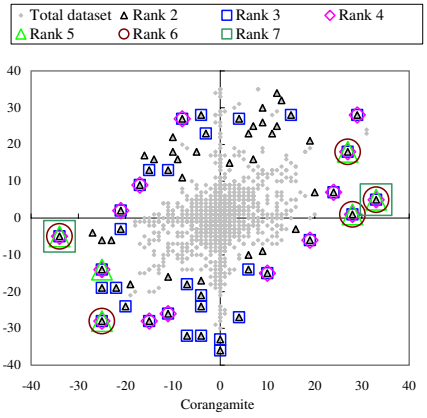
(a) Exceptional Object Analysis.

(b) Campaspe Basin Map in ISC 2004 Report.

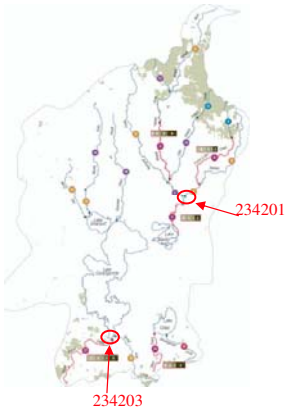
Basin	Site	Site Name	Time	H-change	V-change	WQI
Campaspe	406202	Campaspe River @ Rochester	Jul-09	36	-11	55

(c) Rare Events (Where and When).

Figure V.10: Campaspe.



(a) Exceptional Object Analysis.

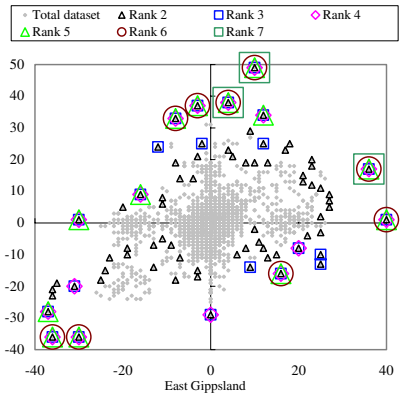


(b) Corangamite Basin Map in ISC 2004 Report.

Basin	Site	Site Name	Time	H-change	V-change	WQI
Corangamite	234201	Woady Yaloak River @ Cressy (Yarima)	Aug-04	-34	-5	64
	234203	Pirron Yallock Creek @ Pirron Yallock (Above HWY BR.)	Apr-81	33	5	42

(c) Rare Events (Where and When).

Figure V.11: Corangamite.



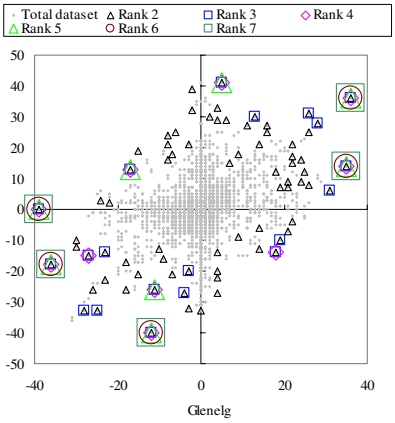
(a) Exceptional Object Analysis.

(b) East Gippsland Basin Map in ISC 2004 Report.

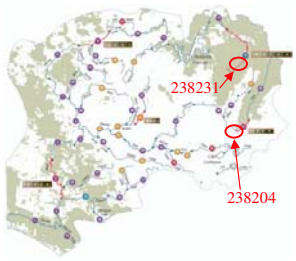
Basin	Site	Site Name	Time	H-change	V-change	WQI
East Gippsland	221212	Bemm River @ Princes Highway	Mar-77	4	38	44
	221212	Bemm River @ Princes Highway	Feb-78	36	17	46
	221212	Bemm River @ Princes Highway	Jul-78	10	49	40

(c) Rare Events (Where and When).

Figure V.12: East Gippsland.



(a) Exceptional Object Analysis.

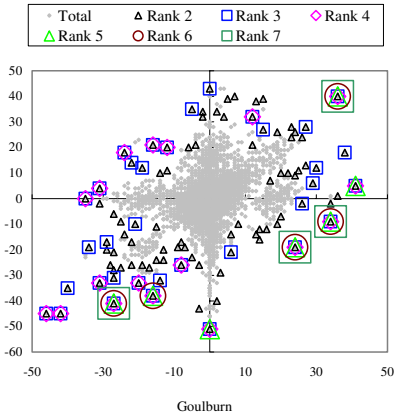


(b) Glenelg Basin Map in ISC 2004 Report.

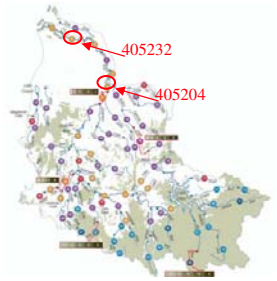
Basin	Site	Site Name	Time	H-change	V-change	WQI
Glenelg	238204	Wannon River @ Dunkeld	Dec-86	-36	-18	68
	238231	Glenelg River @ Big Cord	Nov-81	-39	0	65
	238231	Glenelg River @ Big Cord	Aug-82	-12	-40	65
	238231	Glenelg River @ Big Cord	Aug-83	35	14	25
	238231	Glenelg River @ Big Cord	Dec-08	36	36	58

(c) Rare Events (Where and When).

Figure V.13: Glenelg.



(a) Exceptional Object Analysis.

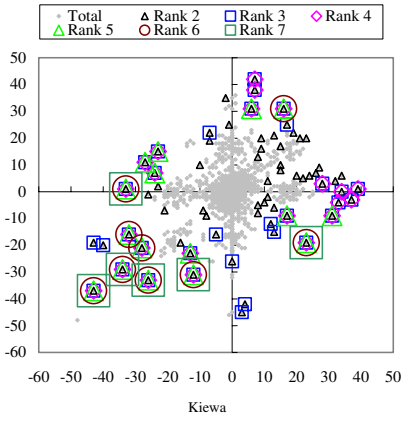


(b) Goulburn Basin Map in ISC 2004 Report.

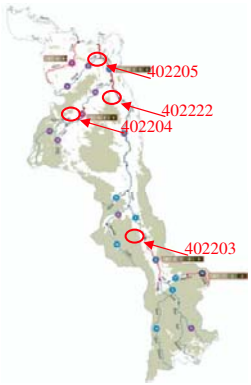
Basin	Site	Site Name	Time	H-change	V-change	WQI
Goulburn	405204	Goulburn River @ Shepparton	Jun-78	-27	-41	71
	405204	Goulburn River @ Shepparton	Dec-79	34	-9	43
	405232	Goulburn River @ McCoy Bridge	Feb-88	36	40	35
	405232	Goulburn River @ McCoy Bridge	Jul-09	24	-19	62

(c) Rare Events (Where and When).

Figure V.14: Goulburn.



(a) Exceptional Object Analysis.



(b) Kiewa Basin Map in ISC 2004 Report.

Basin	Site	Site Name	Time	H-change	V-change	WQI
Kiewa	402203	Kiewa River @ Mongans Bridge	Aug-79	-43	-37	88
	402204	Yackandandah Creek @ Osbornes Flat	Feb-84	-33	1	61
	402204	Yackandandah Creek @ Osbornes Flat	May-88	-34	-29	82
	402204	Yackandandah Creek @ Osbornes Flat	May-91	-26	-33	84
	402205	Kiewa River @ Bandiana	Jul-09	23	-19	65
	402222	Kiewa River @ Kiewa (Main Stream)	Dec-79	-12	-31	76

(c) Rare Events (Where and When).

Figure V.15: Kiewa.

River Basin Name	Number of sites	Total Number of Objects	Number of exceptional objects						
			Total	Rank2 (*eps*=2)	Rank3 (*eps*=3)	Rank4 (*eps*=4)	Rank5 (*eps*=5)	Rank6 (*eps*=6)	Rank7 (*eps*=7)
Avoca	3	924	43 (4.7%)	24	6	5	5	1	2
Barwon	12	3096	70 (2.3%)	48	9	7	3	2	1
Broken	11	2712	63 (2.3%)	37	8	6	2	3	7
Bunyip	10	1896	76 (4%)	50	10	4	4	2	6
Campaspe	7	2724	81 (3%)	38	27	9	6	0	1
Corangamite	8	1824	62 (3.4%)	28	19	9	2	2	2
EastGippsland	7	2556	72 (2.8%)	50	6	3	4	6	3
Glenelg	12	3648	74 (2%)	54	10	2	3	0	5
Goulburn	21	7020	94 (1.3%)	59	17	11	2	1	4
Kiewa	5	1704	66 (3.9%)	33	10	7	7	3	6

Table V.6: Exceptional Objects Ranking Summary.

and V-change are both negative and we discuss this in Section 6.4.

V.6.3 Time Efficiency

In this subsection, we compare EOAN and Improved EOAN algorithms in terms of time efficiency.

DEON is the core function called by EOAN and has two important parameters: $MinPts$ and eps. Since DEON repeats several times with increased eps to process the same dataset in EOAN , we analyze the execution time of each run. We run DEON with $eps = 2 - 7$, then we achieved rank 2-7 exceptional objects, respectively. Fig. V.16 shows the time spent on executing DEON in the EOAN algorithm. We can see that the overall trend of the execution time is increased with the increasing eps.

The execution time of EOAN is the sum of the total execution time of running DEON with $eps = 2 - 7$ and thus is very huge as shown in Fig. V.16.

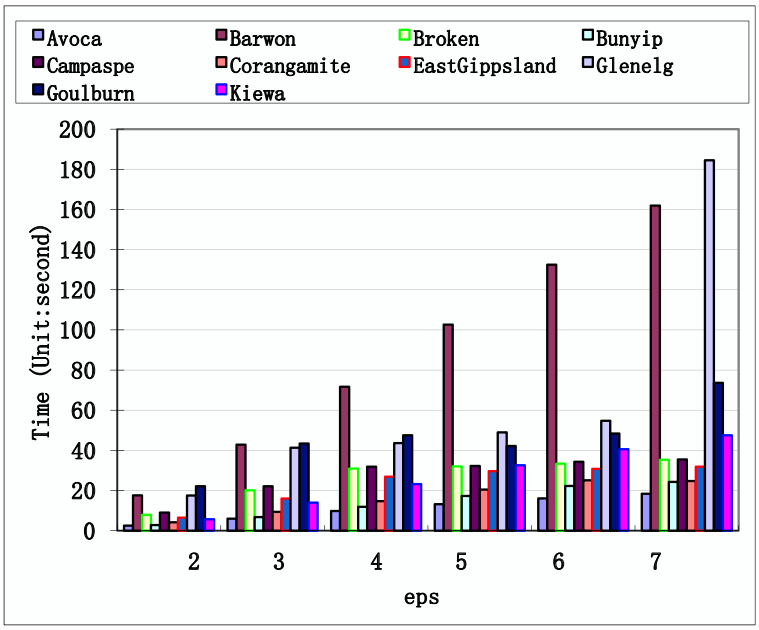


Figure V.16: Time Analysis of DEON in EOAN Algorithm.

As shown in Fig. V.17, Improved EOAN performs far better than EOAN, because Improved EOAN reduces the data size greatly for each DEON function running, as shown in Fig. V.18. Therefore, the execution time is reduced from several hundreds of seconds to several ten seconds. This is critical for processing a huge volume of water data accumulated over many years.

V.6.4 Exceptional Water Pollution Events

In this subsection, we show the effectiveness of our proposed method to detect exceptional water pollution events. We assume Rank 7 exceptional objects with both negative H-change and V-change are the points where water pollution events happen. Table V.7 shows water pollution events detected in 6 river basins. There are no water pollution events in

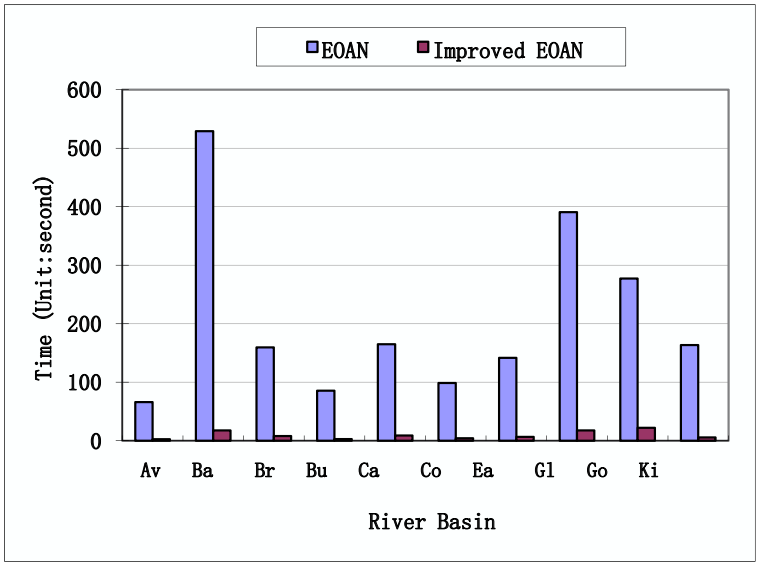


Figure V.17: Time Efficiency: EOAN vs. Improved EOAN.

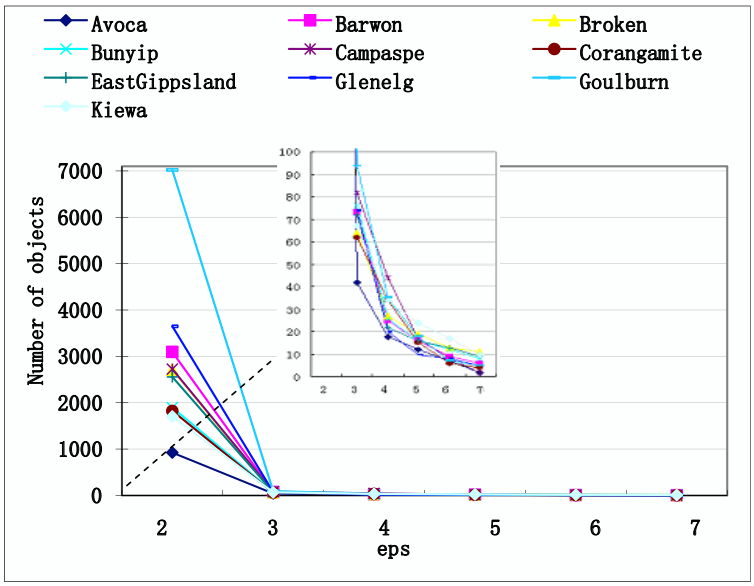


Figure V.18: Data size reduced greatly for each running of DEON in Improved EOAN.

Avoca, Barwon, Campaspe and East Gippsland in this case. It is possible to detect common patterns between the water pollution events. For example, in Kiewa, there are 4 water pollution events at three sites, two happening at at Yackandandah Creek at Osbornes Flat (site 402204) in May 1988 and May 1991. We can test a rule that water pollution events always happen in May at Yackandandah Creek at Osbornes Flat (site 402204). Also, in 1979, two water pollution events happened at two different sites: Kiewa River @ Mongans Bridge (402203) and Kiewa River @ Kiewa (Main Stream) (402222); we may ask the question whether something happened in 1979 in the Kiewa basin that is related to the two events. In Bunyip, two water pollution events happened at Dandenong Creek @ Dandenong (site 228204) in Feb. 1986 and March 1988, two very close months, and one happened at site 228213 in Sep. 1978.

Also, we may observe that Kiewa with medium environmental quality suffered the most water pollution events. Also, Bunyip and Glenelg with very poor environmental quality suffered the second and third most water pollution events. Meanwhile, East Gippsland with excellent environmental quality escaped water pollution entirely. Avoca, Barwon and Campaspe with very poor environmental quality may improve gradually without any water pollution events.

Basin	**Environmental Quality** (in 2004 ISC Report*)	**Site**	**Date**	**WQI**	**H-change**	**V-change**
Broken	Very Poor	404210	1981-5	69	-31	-11
Bunyip	Very Poor	228213	1978-9	75	-39	-30
		228204	1986-2	74	-48	-19
		228204	1988-3	76	-47	-37
Corangamite	Very Poor	234201	2004-8	64	-34	-5
Glenelg	Very Poor	238231	1982-8	65	-12	-40
		238204	1986-12	68	-36	-18
Goulburn	Poor	405204	1978-6	71	-27	-41
Kiewa	Medium	402203	1979-8	88	-43	-37
		402222	1979-12	76	-12	-31
		402204	1988-5	82	-34	-29
		402204	1991-5	84	-26	-33
Avoca	Very Poor	No water pollution events.				
Barwon	Very Poor					
Campaspe	Very Poor					
East Gippsland	Excellent					

* http://www.vicwaterdata.net/

Table V.7: Water pollution events (Rank 7).

V.7 Summary

In this chapter, we have proposed a novel EOAFREE method to detect rare environmental events. Detecting water pollution events from water quality data is a typical case. The core Improved EOAN algorithm in our EOAFREE method is a general solution to detect exceptional objects and to rank the exceptional objects, opposite to the already known Principal Component Analysis (PCA) that ranks principal components. In the real world application, that is, to detect water pollution events from water quality datasets of 93 sites distributed in 10 river basins in Victoria, Australia between 1975 and 2010, we summarize several experiences to use our EOAFREE method: (1) domain knowledge is required to preprocess data; (2) different applications may have different exceptional objects distribution in change maps, though in this application, exceptional objects are located at the borders of the map; (3) the EOAFREE method can automatically rank the exceptional objects from a huge volume of datasets and provide a useful way to help decision makers focus on further analyzing a small critical range of exceptional objects.

Chapter VI

SOMAwater for Water Resource Decision Support

"Every marvel is the strength which isn't resisted outside; or a kind of order with wisdom inside."

– Qiuyu Yu (*A Voyage West*)

In this chapter, we develop a Semantics-Oriented Mining Application for Detecting Water Quality Events (SOMAwater) prototype system based on the CIRCE method for discovering common patterns and the EOAFREE method for detecting rare patterns. The CIRCE method resolves Uncertainty Problems in retrieving common patterns from spatiotemporal data, while the EOAFREE method tackles the heterogeneity problem in Discovering Rare Patterns. The SOMAwater prototype system aims to significantly address critical issues associated with improving the quality and completeness of the water data and negotiating the heterogeneity of water data in order to enhance the accurate and efficient data processing models (e.g. common and rare pattern mining for detecting critical water events) underpin-

ning many of the water resource management strategies and planning decisions.

The structure of this chapter is organized as follows. In Section 1, we present the overview of the SOMAwater prototype system. We query spatiotemporal common patterns in Section 2 and find spatiotemporal rare patterns in Section 3. Finally, we summarize this chapter in Section 4.

VI.1 Overview of The SOMAwater Prototype System

Water is the most important catalyst for human development [Mat02]. Rivers, as a typical type of water resource, are the prime factors controlling the global water cycle. **River Water Pollution** due to urbanization [VV01] [KLL08] [SLJ+02] [KLK+07] [Pra05] becomes a critical issue that must be mitigated to provide the suitability of water to sustain various uses or processes (e.g. drinking, irrigation, industry etc.). Water quality can be defined as a range of variables [Pra05] related to certain levels of physical, chemical or biological characteristics of water. Water quality differs by location (spatial factor) and season (temporal factor) [Pra05].

With the advanced tools (e.g. sensors [GQZe08] [CJ08] [VM10] and Geographical Information Systems (GIS) [KJHK] [GMFC02] [MGR05]), frameworks [ATL+05] [TNGA09] [SR08] and protocols [U.S11] for continuously and closely monitoring the environmental parameters related to water resources, we can capture abundant physical chemical water data (such as PH, temperature, dissolved oxygen, total phosphate, nitrates, turbidity, total

dissolved solids etc.) to analyze the spatial and temporal variation in river water quality [RA09]. However, water resource management will be more complex in the future, worldwide, and relies heavily on computer software processing [Mat02], since the decision makers may not be the engineers and water resource domain experts and must be fed with the right information (or useful knowledge) [Mat02] through data queries (e.g. queries of common or rare patterns for analyzing critical water events). For example, knowing if certain types of water quality problems are isolated (e.g. rare patterns) or ubiquitous (e.g. common patterns) [KLL08] and whether the conditions are changing spatially or temporally is essential for a proper management plan [KLL08] [Cha08]. Responses to these queries face two challenges: (1) uncertainty problems of discrete water quality data curves; and (2) heterogeneity problems of water quality data.

We have introduced the CIRCE method to tackle the uncertainty problems in Chapter IV and presented the EOAFREE method to resolve the heterogeneity problems in Chapter V. Also, we have evaluated the performance of the CIRCE method in terms of efficiency and accuracy by using trajectories of moving objects, while demonstrating the effectiveness of the EOAFREE method by real world water quality datasets. Now, we focus on how to use these two methods in the SOMAwater prototype system to query common and rare water pollution events to support high level decision making in the water resource management field.

The framework of the SOMAwater prototype system is shown in Fig. VI.1. The SO-

MAwater prototype system is a data mining tool specially for water quality data. So, different from other data mining tools, the SOMAwater prototype system first preprocess raw water data. We can see that in the CIRCE package, the CIRCE core module is used to remove Uncertainty due to Discrete Sampling, while the CTP module and the DISP procedure are used to remove Uncertainty due to Sampling Errors. Also, in the EOAFREE package, the water quality index and change-based heterogeneity removing module are used to get rid of heterogeneity. Note that semantics are vital to assist data preprocessing; the CTP module and the DISP procedure are used to find spatiotemporal semantic regions, while water quality index is a kind of water semantics. After preprocessing, the water data become clean and unified to be processed by the LCS mining module to discover common patterns and the IEOAN (Improved EOAN) module to detect rare patterns. These two modules also can be used to generally process any data.

Based on Chapter III, Chapter IV and Chapter V, in this chapter, we focus on introducing the inside structure that organizes the CIRCE package and the EOAFREE package into the SOMAwater prototype system and the application aspects to support water resource decision making.

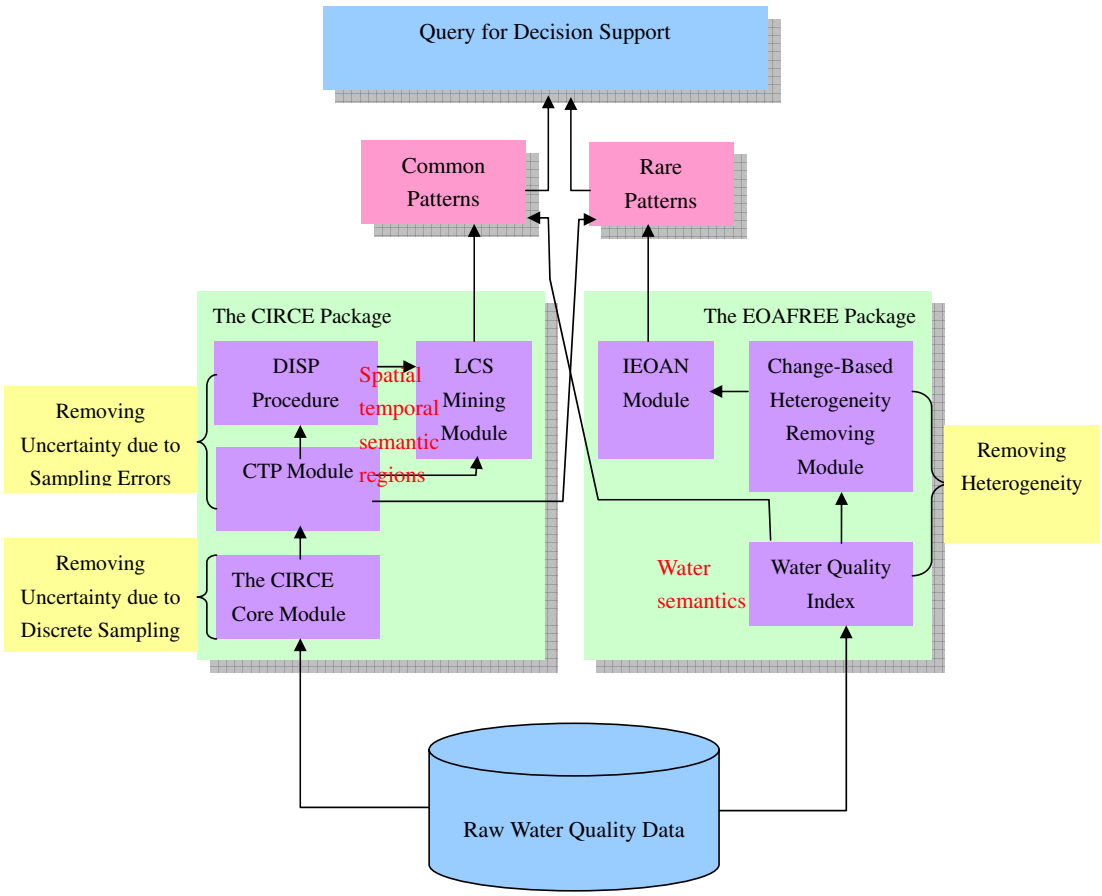


Figure VI.1: The Framework of The SOMAwater Prototype System.

VI.2 Querying Spatio-Temporal Common Patterns for Water Resource Decision Support

VI.2.1 Water Common Patterns

In Chapter III, we have defined a trajectory of a moving object as a sequence of points denoted by $(location, timestamp)$. Here, we take a Water Instance Object (WIO) denoted by $WIO\ (parameter, site, year)$ as a 'moving object', then we define a curve of this WIO as follows:

DEFINITION 19. A curve of a water instance object is a sequence of points $S =< p_1, ..., p_u, ..., p_k >$, where $p_u = (paraValue_u, t_u)$, t_u $(u = 0..k)$ is a timestamp when $paraValue$ is sampled, $\forall_{0 \leq u < k}$, $t_u < t_{u+1}$.

We note that a trajectory of a moving object is a 3D sequence while a curve of a water instance object is a 2D sequence. Similar to Longest Common Route (LCR) patterns, we are interested in Longest Common Curve patterns. We first introduce how to calculate the distance (or difference) between two curves of water instance object. Similar to our work [HHD08b] that emulates how the frog's eyes detect difference, we detect the difference between two curves by emulating human's eyes to detect the visual curves' difference. In Fig. VI.2, suppose there are two curves: S_1S_2 and S_3S_4. Let P_1 denote the closed area of AS_1S_2B and let P_2 denote the closed area of AS_3S_4B. The area in the hatched area denotes the value

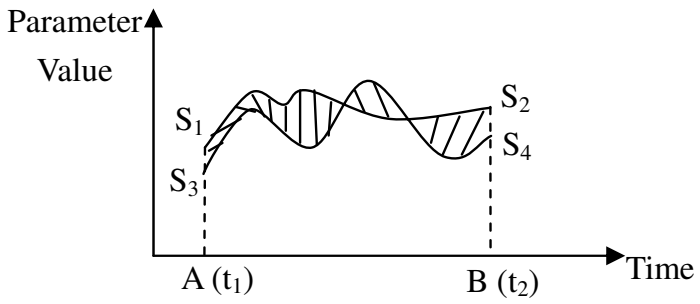


Figure VI.2: Measuring difference between two curves.

of the distance (or difference) between S_1S_2 and S_3S_4, given as follows:

$$diff(S_1S_2, S_3S_4) = \frac{\int_{t=t_1}^{t_2}(|f_{S_1S_2}(t) - f_{S_3S_4}(t)|)dt}{t_2 - t_1}. \quad \text{(VI.2.1)}$$

Thus, we define similar curves as follows:

DEFINITION 20. Given two curves: f_{c_1} and f_{c_2}, if they satisfy the following two criteria, then they are similar.

(1) according to Eq. VI.2.1, $diff(c_1, c_2) < \tau_1$, where τ_1 is a threshold of the difference area in a unit time; and

(2) $|f_{c_1}(t) - f_{c_1}(t)| < \tau_2$, where τ_2 is the threshold of the differences between two values at the same time, t.

Then, based on the difference between curves, we define Longest Common Curve patterns as follows:

DEFINITION 21. A Longest Common Curve (LCC) patterns is a group of k curves that satisfies the following criteria:

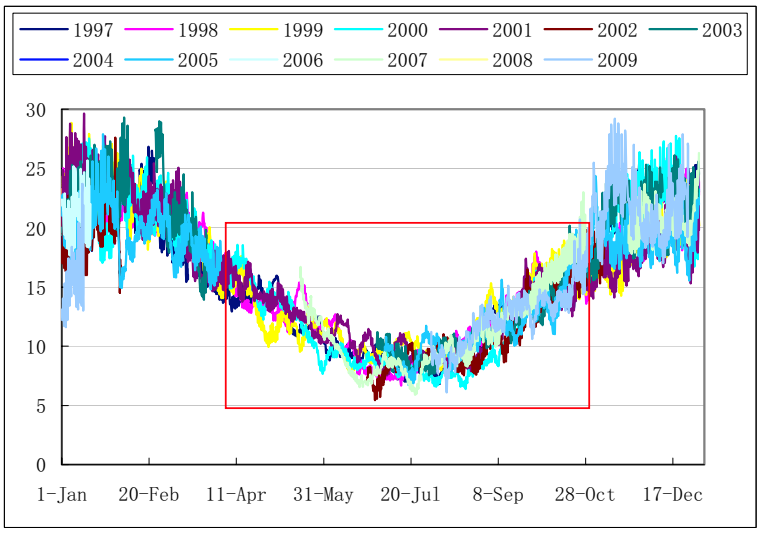


Figure VI.3: A Longest Common Curve Pattern in time span: [April,October] in temperature curves of Avoca Basin, Australia.

(1) each curve is sampled in the same time span $[t_1, t_2]$;

(2) any two curves are similar to each other;

(3) $k \geq min\_sup$ and

(4) this LCC pattern is not a subsequence of any other LCC patterns.

Fig. VI.3 shows an example of a Longest Common Curve pattern in time span: [April, October] in the temperature curves of Avoca Basin, Australia, over 13 years from 1997 to 2009.

We call interesting regions as semantic inflexions and popular turning regions/implicit turning regions as popular semantic inflexions/implicit semantic inflexions in this chapter for summarizing water stream curves and define the following concepts.

DEFINITION 22. A Polygon line based LCC (P-LCC) pattern is an LCC pattern which

includes at least two semantic inflexions.

DEFINITION 23. A Direct line based LCC (D-LCC) pattern is an LCC pattern which includes at most one popular semantic inflexion and at least one implicit popular semantic inflexion.

P-LCC patterns and D-LCC patterns compose the whole set of LCC patterns. We neglect the proving process, since it is similar to Theorem 1 in Chapter III.

DEFINITION 24. A Spatial LCC (S-LCC) pattern is an LCC pattern that is supported by a group of Water Instance Objects denoted by $WIO\ (Para, Site, Year)$, where $Para$ and $Year$ are the same and only $Site$s are different.

DEFINITION 25. A Temporal LCC (T-LCC) pattern is an LCC pattern that is supported by a group of Water Instance Objects denoted by $WIO\ (Para, Site, Year)$, where $Para$ and $Site$ are the same and only $Year$s are different.

DEFINITION 26. A Spatio-Temporal LCC (ST-LCC) pattern is both an S-LCC pattern and a T-LCC pattern.

VI.2.2 Discovering Longest Common Curves (LCC)

To simplify the problem of checking theoretically the similarity of two curves based on Definition 20, we learn the mechanism of retrieving longest common route (LCR) patterns in Chapter III and Chapter IV and thus summarize a curve by a sequence of semantic regions. Therefore, we slightly modify the CIRCE method and then apply it to discover LCC patterns.

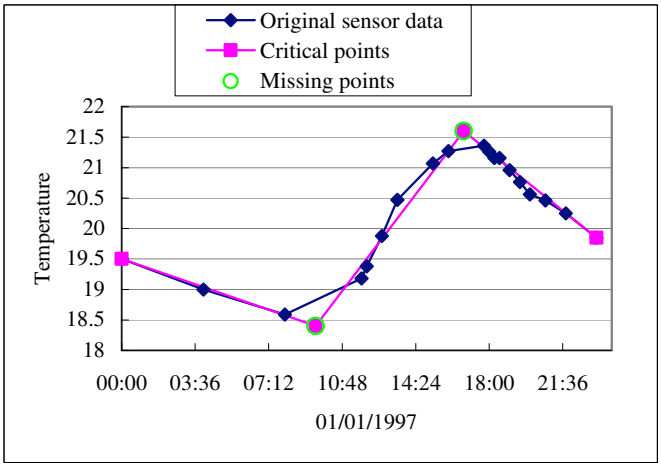


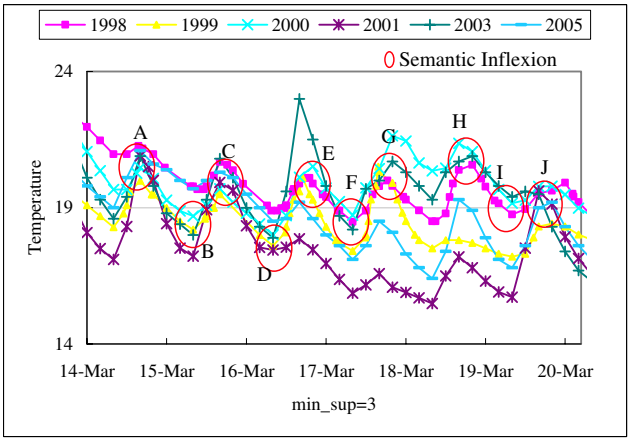
Figure VI.4: Missing highest (lowest) points on a temperature curve at Coonooer, Avoca River, Australia.

The CIRCE Core Algorithm for LCC Patterns

A nature rule about river temperature is that the time of reaching the highest (lowest) value in a day is at a relatively fixed region within a tolerant error; these fixed regions are just like turning regions in trajectories of moving objects. Also, the CIRCE algorithm is useful to correct highest (lowest) points and compress temperature curves in a day by using the critical points (or turning regions). Fig. VI.4 illustrates an example of missing data in Jan. 1, 1997. We can take advantage of the CIRCE core algorithm to correct the missing critical points.

The MicPasts Method for LCC Patterns

We adopt the MicPasts method to discover semantic inflexions. An example is shown in Fig. VI.5. In this case of temperature curves, given $min\_sup = 3$, there are two LCC patterns: $ABCD$ supported by a WIO list of $\{1999, 2000, 2003\}$ and HIJ supported by a



1998 (A, -, C, -, E, F, G, -, H, I, J)
1999 (A, B, C, D, E, -, G, -, -, -, -)
2000 (A, B, C, D, E, F, -, -, H, I, J)
2001 (A, -, C, D, -, -, -, -, -, -, -)
2003 (A, B, C, D, -, F, G, -, H, I, J)
2005 (A, -, C, -, E, -, -, -, -, -, J)

Figure VI.5: Temporal Semantic Inflexions Detected by MicPasts.

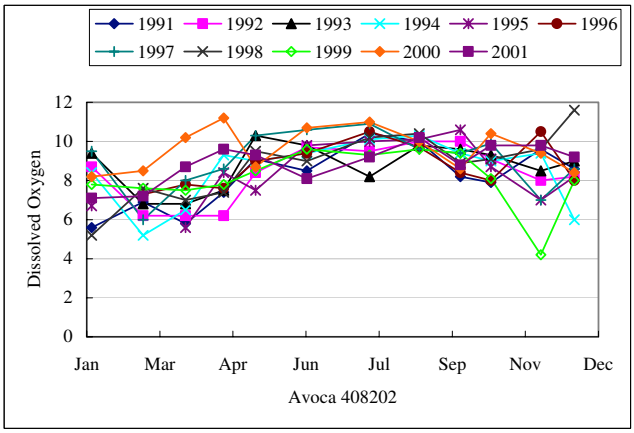
WIO list of $\{1998, 2000, 2003\}$. Also, we can use the DISP procedure in MicPasts to detect implicit semantic inflexions to extend a P-LCC pattern to an LCC pattern. Since the supports are years, these are also two temporal LCC patterns.

We also give more curves of different water quality parameters (e.g. Dissolved Oxygen, Turbidity, PH, Nitrates, Total Phosphorus and WQI) to illustrate temporal LCC patterns from Fig. VI.6 to Fig. VI.11 by using the MicPasts Method. We use 12 month IDs: J (Jan.), F (Feb.), M (March), A (April), Y (May), N (June), L (July), U (Aug.), S (Sep.), O (Oct.), V (Nov.) and D (Dec.) to denote an LCC pattern. Every year curve is denoted by 12 points, each for a value of a month. In some months, there is more than one sampling point, while in others, there is only one sampling point. For the first cases, we use average values of

a month. Note that the sampling points in the same month but in different years may be sampled at different dates. Thus, we also get the average date for an average value. Since the parameter values and sampling dates are different in a month, we use our CTP clustering algorithm to group points into clusters. Those points that are in the same cluster are denoted by a month ID, as a semantic place.

We can detect LCC patterns automatically by using the CIRCE method. Here, we also introduce a simple method to determine whether a curve is an LCC or not. Taking Dissolved Oxygen curves in Fig. VI.6 as an example and given $min\_sup = 3$, we want to determine whether the whole curve is an LCC. For January, 1991 and 1998 are excluded. In February, 1994 and 2000 are excluded. In March, 2001 and 2000 are excluded. In April, 1992 is excluded. In June, 1997 and 2000 are excluded. In July, 1993 is excluded. Then in November, 1999 is excluded. In December, 1994 and 1998 are excluded. Thus, there is only 1996 and 1997 support the whole curve and thus the whole curve is not an LCC, since $2 < min\_sup$. So, we only achieve a shorter LCC pattern: "JFMAYNLUSO", with a support list of $\{1996, 1997, 1999\}$.

Using the same method, we have found several complete curves that are LCC patterns. For example, the whole Turbidity curves show an LCC pattern: "JFMAYNLUSOVD", which are supported by $\{1993, 1994, 1996, 1997, 1999, 2000\}$ as shown in Fig. VI.7, the whole PH curves show an LCC pattern: "JFMAYNLUSOVD", which are supported by $\{1991, 1994, 1995, 1997, 1998, 1999, 2000\}$ in Fig. VI.8 and the whole WQI curves show an

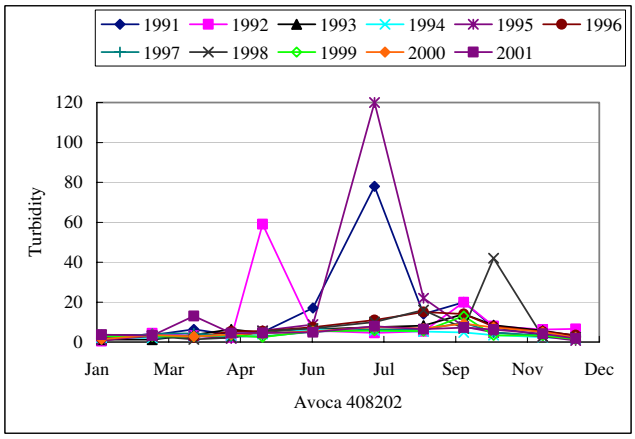


LCC Pattern: JFMAYNLUSO, {1996, 1997, 1999}.

Figure VI.6: LCC patterns in Dissolved Oxygen Curves.

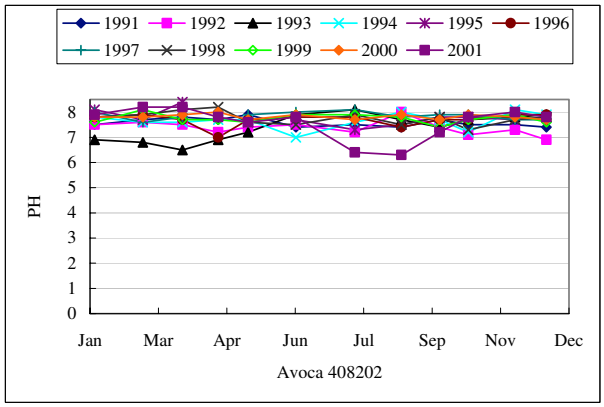
LCC pattern: "JFMAYNLUSOVD", which are supported by $\{1994, 1998, 1999, 2000, 2001\}$ in Fig. VI.11. We also detect shorter LCC patterns, such as in Fig. VI.8, where LCC pattern "AYNLUSOV" with a support list of $\{1992, 1993, 1996\}$ is not the subsequence of "JFMAYNLUSOVD" with a support list of $\{1991, 1994, 1995, 1997, 1998, 1999, 2000\}$, since they have different support lists. More shorter LCC patterns include "JFMAY" with $\{1998, 2000, 2001\}$, "NL" with $\{1998, 1999, 2000\}$ and "OVD" with $\{1999, 2000, 2001\}$ in Fig. VI.9, as well as "FMA" with $\{1998, 1999, 2000\}$ and "NL" with $\{1998, 2000, 2001\}$ in Fig. VI.10.

We then give another example of a spatial LCC pattern in Fig. VI.12, where "ABCDE-FGHIJKLMNOPQRST" is a S-LCC pattern, supported by a site list of $\{$ $Avoca408202$, $Avoca408203$, $Avoca408204$ $\}$.



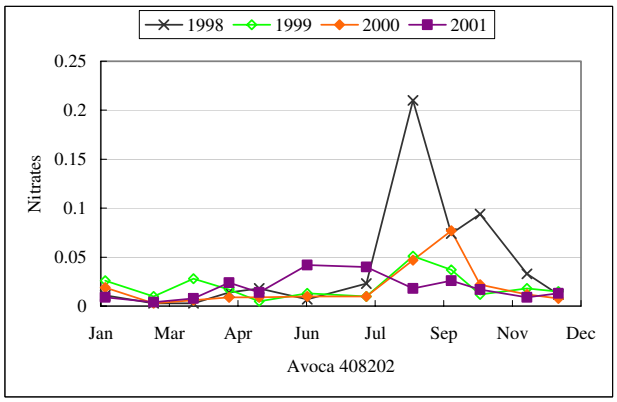
LCC Pattern: JFMAYNLUSOVD, {1993, 1994, 1996, 1997, 1999, 2000}.

Figure VI.7: LCC patterns in Turbidity Curves.



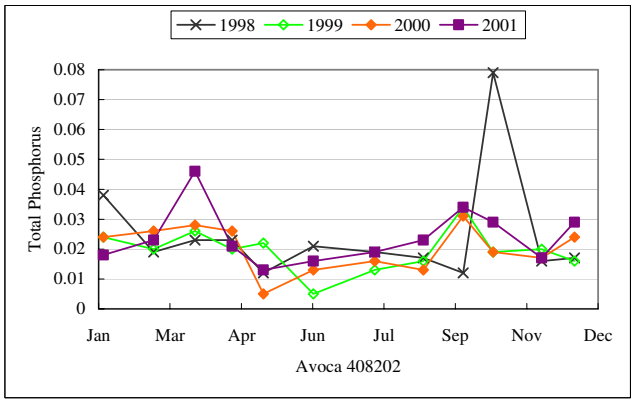
LCC Pattern: JFMAYNLUSOVD, {1991, 1994, 1995, 1997, 1998, 1999, 2000}.
AYNLUSOV, {1992, 1993, 1996}.

Figure VI.8: LCC patterns in PH Curves.



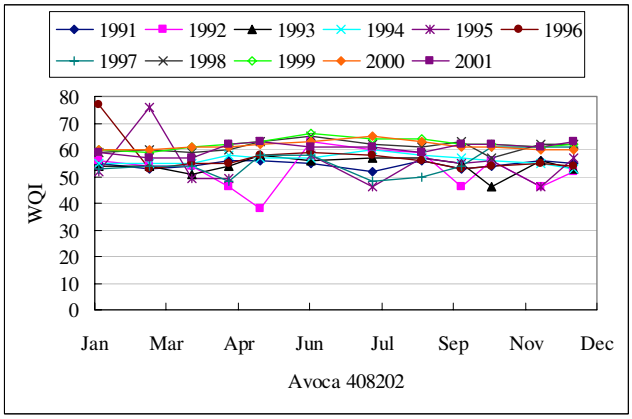
LCC Pattern: JFMAY, {1998, 2000, 2001}
NL, {1998, 1999, 2000}
OVD, {1999, 2000, 2001}

Figure VI.9: LCC patterns in Nitrates Curves.



LCC Pattern: FMA, {1998, 1999, 2000}
NL, {1998, 2000, 2001}

Figure VI.10: LCC patterns in Total Phosphorus Curves.



LCC Pattern: JFMAYNLUSOVD, {1994, 1998, 1999, 2000, 2001}.

Figure VI.11: LCC patterns in WQI Curves.

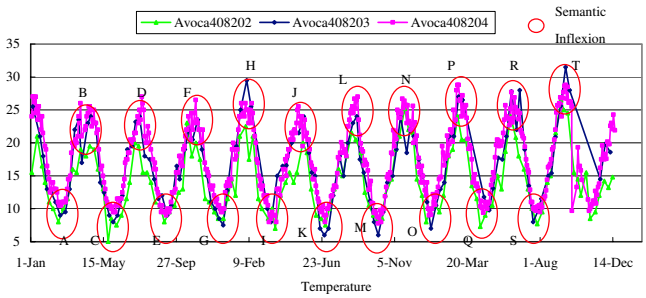


Figure VI.12: Spatial Semantic Inflexions Detected by MicPasts.

Given discrete curves, Eq. VI.2.1 can be changed into

$$diff(S_1S_2, S_3S_4) = \frac{\sum_{t=t_1}^{t_2} |f_{S_1S_2}(t) - f_{S_3S_4}(t)|}{t_2 - t_1}. \quad \text{(VI.2.2)}$$

It is actually difficult to detect whether two curves are similar or not, since two discrete point pairs on two curves may not be exactly sampled at the same time. So the CIRCE method is a solution to detect LCC patterns. First, the CIRCE method splits the whole curves into several segments by using semantic inflexions. Then, it ensures every segment of the curves satisfy the two similar criteria in Definition 20, because the cluster area is very small, compared to the distance between two consecutive semantic inflexions.

Different from trajectories of moving objects, curves of water instance objects have the following distinct characteristics:

- (1) curves of water instance objects do not have self-intersections. So, the DP algorithm and the Angle-DP algorithm can be used correctly.

- (2) the first similar criteria (e.g., accumulated error is bounded) in Definition 20 is necessary to ensure the results are the same as the results determined by a human's eyes. Also, computing accumulated error for a curve is simpler than that for a trajectory.

Also, the CIRCE method has several advantages for finding LCC patterns. First, Angle-DP removes the zigzag between two critical points. Second, the critical points grouped in the same cluster are very close to each other; this is the basis to make a segment between

two semantic inflexions satisfy the two similar criteria in Definition 20. Third, the problem of finding uncertain LCC patterns is mapped onto the problem of finding LCS patterns.

VI.2.3 Experimental Study for Massive Dataset

The massive dataset includes 126,752 points of 6 parameters at 93 sites in 10 river basins between 1975 and 2010. We use the CIRCE method to find LCC patterns. The CIRCE core algorithm can correct some missing critical points, if there are sampled points near to the missing points. Note that the CIRCE core algorithm is only valid when the gap due to the missing data is not too wide; thus, those which have no any sampled point at all in a whole month are out of our control.

We now briefly introduce the experimental setup. We first split the massive dataset into pieces of year curves. Each year curve includes 12 months' values for each water quality parameter. That is, there are 12 points on a year curve, where a point is denoted by (WIO, month, value) and WIO (Water Instance Object) is denoted by $WIO(parameter, site, year)$. An LCC pattern is a sequence of consecutive timestamps (e.g. months or dates) supported by a list of more than min_sup WIOs. We suppose $min\_sup = 3$. First, given m sampling points in a month, if $m = 1$, the only point is the critical point. Otherwise, we use the CIRCE core algorithm to detect the critical point in a month. Then, we group critical points into clusters based on the DBSCAN clustering algorithm. The points that are in the same cluster satisfy two similarity criteria: (1) the span of the two timestamps in a bounded

Parameter	Number of Curves	Total Number of Points
Dissolved Oxygen	2243	22,834
Temperature	2323	24,451
Turbidity	2315	24,116
PH	2355	24127
Nitrates	1634	14,661
Total Phosphorus	1822	16,563

Table VI.1: Curves.

threshold; (2) the difference of two values is bounded in a threshold. We set a cluster ID for each cluster as a semantic place ID and use the cluster IDs to replace the values in order to format a year curve by a sequence of cluster IDs. Finally, we discover LCSs from a number of year curves as the LCC patterns. Note that we can retrieve P-LCC first and then refine a coarse P-LCC by discovering implicit semantic places based on the DISP procedure. If a D-LCC pattern is too short, it is neglected. We classify WIO by water quality parameters and thus there are 6 groups of LCC patterns based on 6 parameters. The basic information of 6 types of curves is summarized in Table VI.1.

We then present some results of 6 types of LCC patterns as shown in Table VI.2 - Table VI.7. By using LCC patterns, the WIOs with sites and years are actually grouped into clusters (e.g., support lists). We can analyze the other common spatiotemporal information about the sites and years to explain why they behave the same as the LCC patterns. This automatic retrieval of LCC patterns offers a potential means to decision makers.

In this experimental study, the parameters of the CIRCE method for mining LCC patterns are the same as in Table III.2, together with ε, a parameter of the CIRCE core algorithm.

We set $\varepsilon = 0.1$ and the other main parameters are marked below the tables. The LCC patterns are denoted by a sequence of consecutive months with support number and list. For example, in Table VI.2, the first row with LCC: "1:2:3" means the Dissolved Oxygen curves of three WIOs: (site 233224, 1996), (site 238228, 1989) and (site 402205, 2006) are common in January, February and March. Also, we observe that for the same LCC, there are different support lists. For instance, LCC: "2:3:4" has 5 support lists. This means the LCC patterns between February and April are in 5 different clusters. According to the parameter value ranges, we set $eps = 0.1$ for Dissolved Oxygen, Temperature, Turbidity and PH and set $eps = 0.001$ for Nitrates and Total Phosphorus. $MinPts = 2$ for all six parameters. Generally, if the users like to get longer LCC patterns, they can decrease min_sup. For example, the max LCC pattern length of Dissolved Oxygen in Table VI.2 is 3 months with $min\_sup = 3$, while all the max LCC pattern lengths of the other 5 water parameters is greater than 4 with $min\_sup = 2$. We do not list LCC patterns with pattern length shorter than 3 or 4, since there are too many of them. Through Table VI.2 - Table VI.7, we can easily get any result with $min\_sup > 2$.

These LCC pattern results are very useful. First, it provides detailed spatiotemporal seasonal common patterns of each water parameter. At a glance of these results in Table VI.2 - Table VI.7, we can learn that there are a greater number of LCC patterns of PH and temperature than those of Dissolved Oxygen, Turbidity, Nitrates and Total Phosphorus, if we set the pattern length greater than 3. Also, we see from Table VI.4 and Table VI.7 that

LCC of Dissolved Oxygen	Sum	Support List (Site, Year)
1:2:3	3	{(233224,96), (238228,89), (402205,06)}
2:3:4	3	{(221212,89), (404206,80), (405219,85)}
	3	{(221201,98), (402205,90), (402222,95)}
	3	{(233224,81), (238205,92), (405202,85)}
	3	{(233218,87), (238205,98), (405202,79)}
	3	{(404216,89), (404224,00), (405200,01)}
3:4:5	3	{(405227,86), (405264,06), (405264,95)}
	3	{(404224,06), (405204,84), (408202,86)}
	3	{(221201,82), (221211,98), (233215,96)}
	3	{(221208,91), (234201,94), (402222,81)}
	3	{(233224,81), (404207,97), (405251,04)}
4:5:6	3	{(405209,92), (405214,01), (405251,80)}
	3	{(221210,99), (233215,96), (405251,05)}
	4	{(233214,93), (404207,95), (405204,80), (405234,98)}
	3	{(221212,89), (404206,06), (405231,80)}
	3	{(402222,02), (402222,81), (405200,89)}
	3	{(228207,97), (402203,81), (405205,98)}
	3	{(221201,82), (221207,85), (221210,86)}
	3	{(405204,04), (406202,04), (406207,98)}
	3	{(221208,99), (405219,00), (406215,00)}
	3	{(238206,85), (238206,98), (238224,98)}
	3	{(221207,98), (405264,81), (405264,89)}
	3	{(5654,92), (22000,96), (404224,95)}
5:6:7	3	{(404224,06), (405204,85), (405237,94)}
	3	{(228203,93), (233228,05), (404207,03)}
	3	{(238231,92), (404206,01), (404224,97)}
6:7:8	3	{(233228,05), (234203,94), (404207,03)}
	3	{(221210,01), (238223,86), (402223,96)}
7:8:9	3	{(404206,80), (405209,93), (405231,98)}
	3	{(404206,08), (404206,91), (405209,86)}
	3	{(233218,85), (404207,00), (405219,96)}
8:9:10	3	{(404224,01), (405202,82), (405209,81)}
	3	{(402222,02), (405203,01), (405214,96)}
	3	{(233218,91), (238223,03), (238223,88)}
	3	{(228213,93), (404224,93), (405251,78)}
	3	{(221211,99), (238204,94), (238206,94)}
	3	{(221210,87), (402204,05), (405212,79)}
	3	{(402204,08), (402205,02), (406215,79)}
9:10:11	3	{(221210,03), (234203,85), (405237,93)}
	6	{(404224,01), (404224,94), (405202,82), (405234,82), (405251,94), (406235,89)}
	3	{(238223,04), (238223,79), (238223,88)}
	3	{(405200,89), (405203,02), (405209,81)}
	3	{(238224,94), (238231,87), (404207,84)}
	3	{(238202,83), (238204,01), (238208,99)}
10:11:12	3	{(402222,01), (404206,91), (406202,94)}
	3	{(221201,83), (238206,83), (405232,08)}
	3	{(221211,83), (233218,91), (405219,05)}
	3	{(233214,01), (402203,80), (408202,01)}
	4	{(233224,99), (238223,79), (238223,88), (405214,01)}

eps=0.1, MinPts=2, min_sup=3 and pattern length>=3.

Table VI.2: LCC of Dissolved Oxygen.

LCC of Temperature	Sum	Support List (site, year)
1:2:3:4	2	{(238208,84), (405202,77)}
1:2:3:4:5:6:7:8	2	{(5678,92), (5681, 92)}
2:3:4:5	2	{(405212,86), (405234,87)}
	2	{(221201,96), (406207,95)}
3:4:5:6	2	{(234201,02), (404207,94)}
	2	{(233214,90), (405202,90)}
3:4:5:6:7:8	2	{(234203,90), (405240,92)}
	3	{(5654,92), (5678,92), (5681,92)}
4:5:6:7	2	{(238231,87), (405214,86)}
	2	{(233218,93), (405212,79)}
	2	{(221211,98), (233214,89)}
	2	{(402222,89), (405200,77)}
	2	{(233228,02), (405251,94)}
	2	{(238205,96), (404210,86)}
	2	{(221212,82), (405212,96)}
4:5:6:7:8	2	{(234203,90), (405240,92)}
	2	{(234201,89), (404210,92)}
5:6:7:8	2	{(233224,77), (404207,89)}
	2	{(233215,88), (406202,92)}
	2	{(402204,97), (405214,85)}
	2	{(233215,80), (238202,95)}
	2	{(405202,87), (406214,78)}
	2	{(234201,89), (404210,92)}
	2	{(52900,97), (238223,96)}
	4	{(5654,92), (5678,92), (5681,92), (228209, 90)}
	2	{(233211,94), (405232,93)}
	2	{(402205,04), (405209,93)}
	2	{(5678,86), (238224,91)}
5:6:7:8:9	2	{(233218,87), (405246,87)}
	2	{(405234,84), (405234,86)}
6:7:8:9	2	{(402204,89), (405204,89)}
	2	{(404207,89), (404214,83)}
	2	{(238223,95), (238228,00)}
	2	{(238231,86), (404207,92)}
	2	{(228207,93), (405240,91)}
	2	{(405200,94), (405204,86)}
	2	{(405231,90), (406207,92)}
	2	{(234203,02), (405232,93)}
	2	{(406214,80), (406224,80)}
	2	{(238204,88), (405212,93)}
	4	{(233211,87), (233218,87), (404207,96), (405246,87)}
	2	{(233214,99), (402222,81)}
	2	{(404210,92), (405204,80)}
	2	{(228209,89), (233218,79)}
6:7:8:9:10	2	{(221201,92), (221212,90)}
7:8:9:10	3	{(52900,85), (238228,77), (405214,95)}
	2	{(404214,95), (405204,85)}
	2	{(405205,84), (405209,86)}
	2	{(405202,78), (406202,86)}
	2	{(233214,80), (405212,91)}
	2	{(221201,92), (221212,90)}
	2	{(238231,97), (405218,84)}
	2	{(233215,81), (238224,98)}
7:8:9:10:11	2	{(221201,78), (406214,92)}
8:9:10:11	2	{(221201,78), (406214,92)}
	2	{(52900,85), (405214,79)}
	2	{(233214,85), (402222,81)}
	2	{(406202,79), (406207,78)}
	2	{(233228,97), (405219,98)}
	2	{(233211,84), (405202,78)}
	2	{(405234,84), (406215,91)}
	2	{(405202,86), (405232,89)}
	2	{(238204,98), (406207,77)}
9:10:11:12	2	{(406202,81), (406214,94)}

eps=0.1, MinPts=2, min_sup=2 and pattern length >=4

Table VI.3: LCC of Temperature.

LCC of Turbidity	Sum	Support List (Site, Year)
1:2:3:4	2	{(233218,99), (405202,90)}
	2	{(238206,80), (238208,09)}
2:3:4:5	2	{(233215,08), (238224,01)}
	2	{(221207,77), (405264,06)}
2:3:4:5:6	2	{(221210,93), (238236,96)}
	2	{(221207,92), (405219,99)}
3:4:5:6	2	{(221210,93), (238236,96)}
	2	{(221207,92), (405219,99)}
	2	{(405258,96), (405264,87)}
	2	{(221209,80), (234608,92)}
	2	{(221201,95), (405258,92)}
3:4:5:6:7	2	{(5654,96), (5678,96)}
4:5:6:7	2	{(221207,82), (238237,06)}

eps=0.1, MinPts=2, min_sup=2 and pattern length>=4

Table VI.4: LCC of Turbidity.

Turbidity and Total Phosphorus curves show less seasonal patterns spatially and temporally, since the number of supports for a LCC pattern is almost 2.

We can also detect some rare events by setting a very great pattern length. For example, in Table VI.5, two WIOs: (site 238202, 2007) and (site 238228, 2007) are supports for the longest LCC pattern (a whole year pattern). Another example is “6:7:8:9:10:11:12”, which is supported by 14 WIOs, with the greatest number of supports. But the point real means to decision makers is that site 234608 supports this LCC pattern of PH for 9 years: 1985, 1993, 1995, 1995, 1997, 1999, 2002, 2005 and 2007. Also, the same site: 238231 supports LCC pattern of Nitrates “8:9:10:11:12” for consecutive 5 years: from 1990 to 1994 as shown in Table VI.6.

LCC of PH	Sum	Support List (Site, Year)
1:2:3:4:5:6:7	2	{(238202,92), (238228,97)}
	2	{(221207,03), (405232,81)}
1:2:3:4:5:6:7:8	2	{(234200,88), (405214,85)}
	2	{(238224,87), (238224,99)}
1:2:3:4:5:6:7:8:9	2	{(233218,97), (234201,88)}
1:2:3:4:5:6:7:8:9:10:11:12	2	{(238202,07), (238228,07)}
1:3:4:5:6:7:8	3	{(221201,79), (221209,79), (405218,77)}
2:3:4:5:6:7:8	2	{(238202,92), (238228,91)}
	5	{(234201,78), (234201,97), (238202,01), (402222,82), (406213,99)}
	3	{(233215,98), (238228,96), (238228,98)}
	2	{(238202,96), (238205,07)}
	3	{(233215,02), (233215,08), (233215,88)}
	2	{(404218,96), (406202,84)}
2:3:4:5:6:7:8:9	2	{(233218,97), (234201,88)}
	2	{(404218,06), (405212,78)}
	2	{(234200,79), (402222,81)}
2:3:4:5:6:7:8:9:10	2	{(22000,92), (234606,97)}
2:3:4:5:6:7:8:9:10:11	2	{(404219,95), (405204,81)}
2:3:4:5:6:7:8:9:10:11:12	2	{(238202,07), (238228,07)}
3:4:5:6:7:8:9	9	{(234200,79), (234201,79), (234201,80), (238202,88), (238228,84), (238228,88), (402222,81), (404219,95), (405204,81)}
	7	{(5654,02), (405204,80), (405214,85), (405232,07), (405240,78), (406202,97), (406215,77)}
3:4:5:6:7:8:9:10	2	{(238228,97), (406213,08)}
3:4:5:6:7:8:9:10:11	3	{(402222,81), (404219,95), (405204,81)}
3:4:5:6:7:8:9:10:11:12	2	{(238202,07), (238228,07)}
4:5:6:7:8:9:10	2	{(402203,80), (402204,82)}
	6	{(234201,05), (402203,81), (402204,81), (405234,82), (406207,04), (406214,01)}
4:5:6:7:8:9:10:11:12	2	{(238202,07), (238228,07)}
	3	{(234606,93), (234608,02), (234608,85)}
5:6:7:8:9:10:11	3	{(5254,95), (238228,94), (406207,77)}
	9	{(221210,96), (233217,05), (234201,05), (234201,98), (238202,05), (238223,88), (405212,78), (406207,04), (406215,01)}
	2	{(5678,96), (233215,04)}
	2	{(405200,05), (405205,99)}
5:6:7:8:9:10:11:12	2	{(402223,92), (405264,92)}
	4	{(5254,02), (233218,05), (233224,84), (405214,84)}
	2	{(238202,07), (238228,07)}
6:7:8:9:10:11:12	4	{(5254,02), (233218,05), (233224,84), (405214,84)}
	5	{(402222,81), (404219,95), (405219,79), (406215,02), (406215,77)}
	2	{(402223,92), (405264,92)}
	9	{(3361,88), (221211,79), (238202,98), (238206,97), (238228,98), (402203,81), (402204,81), (404219,97), (406207,05)}
	14	{(22000,95), (22000,99), (234606,05), (234606,93), (234606,99),(234608,02),(234608,05),(234608,07), (234608,85),(234608,93),(234608,94), 234608,95), (234608,97), (234608,99)}
	2	{(22000,02), (22000,85)}
	2	{(221209,79), (406202,07)}

eps=0.1, MinPts=2, min_sup=2 and pattern length>=7

Table VI.5: LCC of PH.

LCC of Nitrates	**Sum**	**Support List (Site, Year)**
2:3:4:5:6	2	{(238224,08), (238231,95)}
	2	{(234203,91), (409216,93)}
2:3:4:5:6:7:8:9:10:11:12	3	{(238231,91), (238231,92), (238231,94)}
4:5:6:7:8:9:10:11:12	4	{(238231,91), (238231,92), (238231,93), (238231,94)}
8:9:10:11:12	7	{(238231,90), (238231,91), (238231,92), (238231,93), (238231,94), (406214,94), (409204,97)}

eps=0.001, MinPts=2, min_sup=2 and pattern length >=5.

Table VI.6: LCC of Nitrates.

LCC of Total Phosphorus	Sum	Support List (Site, Year)
1:2:3:4	2	{(238228,89), (238228,94)}
3:4:5:6	3	{(221210,10), (238231,97), (405205,08)}
	2	{(238236,92), (405258,92)}
	2	{(238224,99), (405205,94)}
3:4:5:6:7:8:9	2	{(238231,92), (238231,93)}
4:5:6:7	2	{(238236,93), (405214,91)}
	2	{(238236,08), (406215,98)}
	2	{(405214,06), (405219,06)}
	2	{(52900,96), (234608,96)}
	2	{(408202,93), (408202,94)}
	2	{(221207,07), (238237,05)}
	2	{(405203,01), (405219,94)}
	2	{(238231,00), (238231,05)}
4:5:6:7:8	2	{(234608,92), (408202,95)}
4:5:6:7:8:9	2	{(238231,92), (238231,93)}
5:6:7:8	2	{(52900,91), (408202,91)}
7:8:9:10	2	{(234608,85), (234608,99)}
	2	{(238231,08), (238231,91)}
8:9:10:11	3	{(221210,04), (221210,91), (238231,07)}
	2	{(405264,95), (408202,94)}
	2	{(221210,97), (405258,92)}
9:10:11:12	2	{(405203,93), (405231,90)}
	2	{(405205,96), (405231,94)}

eps=0.001, MinPts=2, min_sup=2 and pattern length>=4.

Table VI.7: LCC of Total Phosphorus.

VI.2.4 Applications of LCC Patterns

In summary, LCC patterns have extensive applications. We can query LCC patterns to achieve a group of supporters; this is very useful. We can recursively use the CTP algorithm in the CIRCE method to analyze spatial similarity of WIOs in the same LCC support list. Or, we can go on to find seasonal rare patterns in an LCC support list by using the EOAFREE method. Moreover, two more interesting application scenes of the LCC patterns are to predict the future trends and to estimate the missing points. For example, suppose a user in June 2010, given already known six points of a turbidity curve from January to June in 2010, we can query an LCC pattern with year 2010 as one of the supports for this half a year's curve, and then choose the most similar year, say 1998 in the same site, from the support list as the counterpart to predict the trend of 2010's curve in July or onwards. Also, suppose the data in July or onwards are missing, we can use the same method to estimate the missing points.

VI.3 Finding Spatio-Temporal Rare Patterns for Water Resource Decision Support

In Chapter V, we focus on evaluating the time efficiency of our EOAFREE method. Also, we have evaluated the effectiveness of the EOAFREE method by using the pollution events (negative rare events) that have been found. Actually, for decision makers, both negative and positive rare events are critical. The positive rare events (or factors) that increase the water quality are very useful, since we can learn useful experiences from the rare positive

events and then popularize their effectiveness to cure pollution in other sites. Therefore, in our SOMAwater prototype system, we provide several query functions to enhance the information to assist water resource decision making.

VI.3.1 Water Rare Patterns

Given a set of rare patterns, denoted by $R = \{R_1, R_2, ...R_k\}$, where R_i ($i \in [1, k]$) is a rare pattern denoted by a 5-tuple $(site_i, time_i, H-change_i, V-change_i, WQI_i)$, we define several concepts about rare patterns as follows:

DEFINITION 27. A Positive Rare Pattern is a rare pattern in which $H - change_i > 0$ and $V - change_i > 0$.

DEFINITION 28. A Negative Rare Pattern is a rare pattern in which $H - change_i < 0$ and $V - change_i < 0$.

DEFINITION 29. A Month-Decrease Rare Pattern is a rare pattern in which $H-change_i < 0$ and $V - change_i > 0$.

DEFINITION 30. A Year-Decrease Rare Pattern is a rare pattern in which $H-change_i > 0$ and $V - change_i < 0$.

These four rare patterns are classified by $H - change$ and $V - change$. Positive rare patterns are very useful for evaluating pollution control effectiveness, such as wastewater treatment and in-stream purification system [SLJ+02], while negative rare patterns are used

to detect the pollution events that we have presented in Chapter V. Month-Decrease rare patterns are so called since the decrease of water quality is related to seasons. Year-Decrease rare patterns may also be evidence to support some positive rare patterns happening in the last year.

Then, according to the applications, there are following interesting rare events:

DEFINITION 31. A Continual Rare Event is a sequence of consecutive months in a year of (years for the same month) that is supported by a group of rare patterns happening at the same site.

DEFINITION 32. A Simultaneous Rare Event is a month supported by a group of rare patterns happening at multiple sites.

DEFINITION 33. A Global Rare Event is an exceptional object that is detected by the IEOAN algorithm, given the input as the rare pattern set R.

DEFINITION 34. A Seasonal Rare Event is a group of rare patterns related to a month and rare patterns happening in any year and at any sites that can support this pattern.

The above four types of rare events are interesting for decision makers. Continual rare events generally indicate that some factors decrease or increase the water quality continuously, and if this factor does not disappear, the water quality continues to change. For example, if an oil tank leakage decreases the water quality of a nearby river, we may detect a continual rare pattern. Simultaneous rare events generally are more interesting than the rare

patterns happening in a single site; they are useful evidence to help analyze the real reasons for changing water quality. We can analyze common patterns based on the other characteristics of those sites to find the reasons. Not all local rare patterns that are discovered at a local site are global rare patterns, since the same rare patterns may happen at different sites and from a global view, they are not exceptional. Seasonal rare events are helpful to analyze the relationship between seasons and rare events.

VI.3.2 Rare Pattern Queries

We take an example set of rare patterns to explain how to query the three types of rare patterns. The rare patterns detected in the experiments in Chapter V based on exceptional ranking 7 are shown in Table VI.8. Here, we list all the negative, positive, Month-Decrease and Year-Decrease rare patterns.

Querying Continual Rare Patterns

From Table VI.8, the continual rare patterns are shown in Fig. VI.13. Fig. VI.13 (a) and (b) show a continual rare pattern in consecutive months in a year and a continual rare pattern in consecutive years for a month, respectively.

Querying Simultaneous Rare Patterns

From Table VI.8, the simultaneous rare patterns are shown in Fig. VI.14. Querying simultaneous rare patterns involves two functions: (1) to find the common factors that cause homogenous rare patterns, such as the positive rare patterns in Fig. VI.14 (b) and Year-

Basin	Site	Time	H-change	V-change	WQI
Avoca	408202	Feb-94	0	21	55
	408202	Dec-95	20	-3	57
Barwon	233224	Apr-93	10	43	38
Broken	404207	Dec-78	41	18	39
	404210	May-81	-31	-11	69
	404210	Feb-88	39	41	29
	404210	Mar-88	-34	2	68
	404210	May-88	16	34	36
	404216	Jan-09	14	49	44
	404712	Mar-09	8	42	52
Bunyip	228204	Feb-86	-48	-19	74
	228204	Jan-87	25	46	30
	228204	Mar-87	-27	21	55
	228204	Jan-88	0	-37	76
	228204	Mar-88	-47	-37	76
	228213	Sep-78	-39	-30	75
Campaspe	406202	Jul-09	36	-11	55
Corangamite	234201	Aug-04	-34	-5	64
	234203	Apr-81	33	5	42

Basin	Site	Time	H-change	V-change	WQI
East Gippsland	221212	Mar-77	4	38	44
	221212	Feb-78	36	17	46
	221212	Jul-78	10	49	40
Glenelg	238204	Dec-86	-36	-18	68
	238231	Nov-81	-39	0	65
	238231	Aug-82	-12	-40	65
	238231	Aug-83	35	14	25
	238231	Dec-08	36	36	58
Goulburn	405204	Jun-78	-27	-41	71
	405204	Dec-79	34	-9	43
	405232	Feb-88	36	40	35
	405232	Jul-09	24	-19	62
Kiewa	402203	Aug-79	-43	-37	88
	402204	Feb-84	-33	1	61
	402204	May-88	-34	-29	82
	402204	May-91	-26	-33	84
	402205	Jul-09	23	-19	65
	402222	Dec-79	-12	-31	76

Table VI.8: Rare Patterns (Ranking 7).

Broken	404210	**Feb-88**	39	41	29
	404210	**Mar-88**	-34	2	68

(a) A continual rare pattern in consecutive months in a year.

Glenelg	238231	**Aug-82**	-12	-40	65
	238231	**Aug-83**	35	14	25

(b) A continual rare pattern in consecutive years for a month.

Figure VI.13: Continual Rare Patterns.

Goulburn	**405204**	Dec-79	43	34	-9
Kiewa	**402222**	Dec-79	76	-12	-31

(a) Year-Decrease vs. Negative Rare Patterns.

Broken	**404210**	Feb-88	29	39	41
Goulburn	**405232**	Feb-88	35	36	40

(b) Positive Rare Patterns.

Broken	**404210**	Mar-88	68	-34	2
Bunyip	**228204**	Mar-88	76	-47	-37

(c) Month-Decrease vs. Negative Rare Patterns.

Broken	**404210**	May-88	36	16	34
Kiewa	**402204**	May-88	82	-34	-29

(d) Positive vs. Negative Rare Patterns.

Campaspe	**406202**	Jul-09	55	36	-11
Goulburn	**405232**	Jul-09	62	24	-19
Kiewa	**402205**	Jul-09	65	23	-19

(e) Year-Decrease Rare Patterns.

Figure VI.14: Simultaneous Rare Patterns.

Decrease rare patterns in Fig. VI.14 (e); (2) to compare factors that cause heterogenous rare patterns, such as the Year-Decrease vs. negative rare patterns in Fig. VI.14 (a), Month-Decrease vs. negative rare patterns in Fig. VI.14 (c) and positive vs. negative rare patterns in Fig. VI.14 (d).

Querying Seasonal Rare Patterns

We also can query seasonal rare patterns from Table VI.8. The results are shown in Fig. VI.15. We can observe from Fig. VI.15 that most of the rare patterns happen in December and February, while the least rare patterns happen in October, September, November and June. Also, negative rare patterns (38%) and positive rare patterns (40%) are the majority, being around 78% of the total number of rare patterns. The experimental study in Chapter

Month	Total Rare Patterns	Negative	Positive	Month-Decrease	Year-Decrease
Jan.	3	1	2	0	0
Feb.	6	1	4	1	0
Mar.	5	1	2	2	0
Apr.	2	0	2	0	0
May	4	3	1	0	0
Jun.	1	1	0	0	0
Jul.	4	0	1	0	3
Aug.	4	3	1	0	0
Sep.	1	1	0	0	0
Oct.	0	0	0	0	0
Nov.	1	1	0	0	0
Dec.	6	2	2	0	2
Sum	37	14	15	3	5

Figure VI.15: Seasonal Rare Patterns.

V has validated the effectiveness of using negative rare patterns to detect pollution events. Here, we say that the positive rare patterns are useful to help learn about discovering anti-pollution events.

Querying Global Rare Patterns

Querying global rare patterns is to study the relationship between the global exceptional water pollution events for 93 sites and the local exceptional water pollution events for each site. As we believe "exception" is a comparative concept, exceptional objects in one site's dataset may not be exceptional in a global dataset and we aim to find some general rules for environmental rare event detection applications. Different from querying local rare patterns, we develop a procedure to query global rare patterns, which includes the following steps:

Step 1: Use the EOAFREE method for a river basin, i, and detect the exceptional objects as rare patterns that are put into a rare pattern set: R_i.

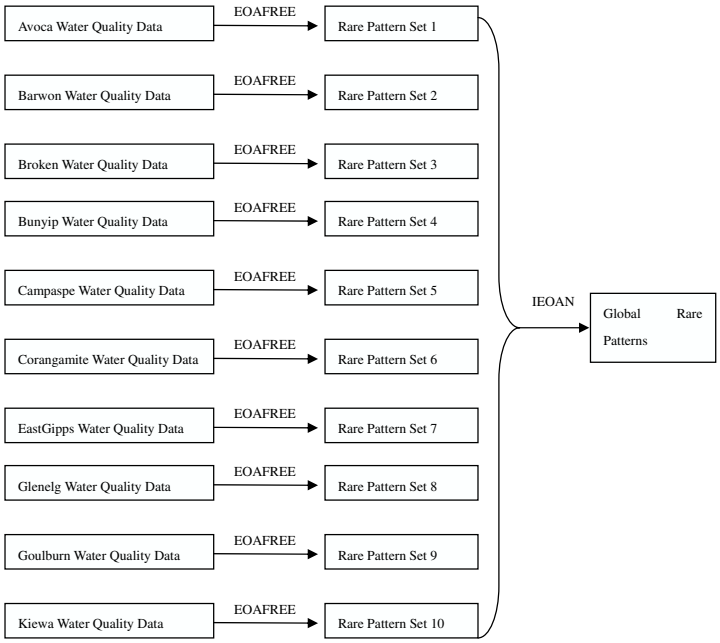


Figure VI.16: Flowchart of Detecting Global Rare Patterns.

Step 2: Combine all of k rare pattern sets for k river basin into one rare pattern set: $R = R_1 \cup R_2 ... \cup R_k$.

Step 3: Use the IEOAN algorithm again to detect and rank global rare patterns from R.

A flowchart is shown in Fig. VI.16, where local rare patterns are found by the EOAFREE method firstly in 10 river basins and then the IEOAN algorithm runs again to detect global rare patterns.

The global rare patterns for these 10 river basins are shown in Table VI.9 and the distribution of (H-change, V-change) is shown in Fig. VI.17. We can see from Table VI.9 that there are 13 global rare patterns, being 35% of total 37 local rare patterns.

Basin	Site	Time	WQI	H-change	V-change
Avoca	408202	Feb-94	55	0	21
Avoca	408202	Dec-95	57	20	-3
Broke	404210	May-88	36	16	34
Bunyip	228204	Feb-86	74	-48	-19
Bunyip	228204	Jan-87	30	25	46
Bunyip	228204	Mar-87	55	-27	21
Bunyip	228204	Jan-88	76	0	-37
Corangamite	234203	Apr-81	42	33	5
Glenelg	238204	Dec-86	68	-36	-18
Glenelg	238231	Aug-82	65	-12	-40
Goulburn	405204	Jun-78	71	-27	-41
Kiewa	402204	May-91	84	-26	-33
Kiewa	402222	Dec-79	76	-12	-31

Table VI.9: Global Rare Patterns (Ranking 7).

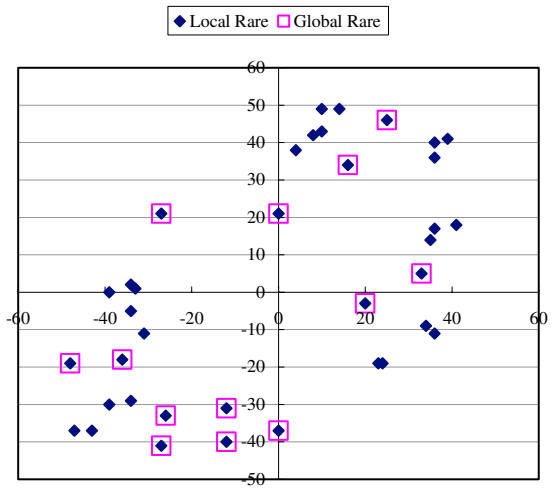


Figure VI.17: Results of Global Rare Patterns.

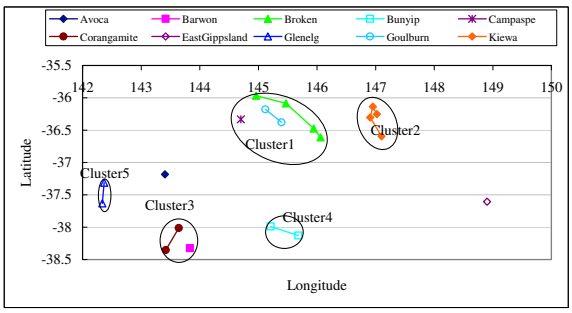


Figure VI.18: Rare Pattern Clusters.

Querying Spatial Similarity

Although we naturally group rare patterns by river basins, spatial similarity analysis is also important. For example, we group all of the rare patterns in Table VI.8 into clusters based on the DBSCAN clustering algorithm as shown in Fig. VI.18. Rare patterns in the same river basin naturally form a cluster, such as 4 rare patterns in Kiewa in Cluster 2, 2 rare patterns in Glenelg in Cluster 5 and 2 rare patterns in Bunyip in Cluster 4. Meanwhile, using spatial similarity analysis, we also merge rare patterns in different river basins into one cluster. For example, 4 rare patterns in Broken, 2 rare patterns in Goulburn and 1 rare pattern in Campaspe are grouped into Cluster 1 and 2 rare patterns in Corangamite and 1 rare pattern in Barwon are grouped into Cluster 3. Each of Avoca and East Gippsland has 1 rare pattern separated naturally. We can use the methods to detect common and rare patterns intra a spatial cluster, the same as we have done intra a river basin.

VI.4 Summary

In this chapter, we have presented the SOMAwater prototype system that aims to query Longest Common Curve (LCC) patterns and rare patterns to support decision making. LCC patterns and rare patterns of water quality curves are important, since they can help decision makers focus only on the valuable datasets and discover spatiotemporal water patterns efficiently and automatically, as well as in more detail. The main techniques adopted in the SOMAwater prototype system are the CIRCE method in Chapters III and IV and the EOAFREE method in Chapter V. Note that the efficiency and accuracy of the CIRCE method and the EOAFREE method have been validated in Chapter III, Chapter IV and Chapter V. This chapter focuses on illustrating querying LCC patterns and several rare patterns (e.g., continual,simultaneous, seasonal and global rare patterns) from water quality data by using extensive examples. We have conducted a massive experimental study based on a water quality dataset of 93 sites distributed in 10 river basins in Victoria, Australia between 1975 and 2010 for querying LCC patterns while taking advantage of the experimental results in Chapter V to classify rare patterns further.

Chapter VII

Conclusions

VII.1 Summary of Contributions

Water is becoming a highly valued resource, not only in Australia, but in many countries around the world. This value is driving the need to improve the precision and efficiency of water resource management practices. With the help of advanced tools (e.g. sensors) and protocols for collecting water resource data, water data can be collected by closely monitoring the environmental parameters related to water resources. Already known spatiotemporal water quality data analysis methods are generally based on statistical techniques and spatiotemporal patterns are recognized manually or semi-manually; thus they cannot handle a large number of water data efficiently and in detail. State-of-the-art spatiotemporal data mining algorithms cannot directly satisfy the efficient and accurate mining of water quality data due to uncertainty and heterogeneity problems in the real world water quality dataset.

This dissertation have addressed several vital issues associated with improving the precision and efficiency of querying critical events from spatiotemporal datasets being used in the

models underpinning many of the water management strategies and planning decisions, that is, to improve data quality and completeness by identifying and correcting data inaccuracies in datasets, to remove redundant data points to compact the data volume and to detect critical events such as river pollution. These are also general problems in the spatiotemporal data mining field that we present from two aspects. The first problem is to discover common patterns from an uncertain, even imprecise, and huge volume of spatiotemporal data. However, already known algorithms for processing spatiotemporal data only can compress data but cannot correct the imprecise critical points on a data stream at the same time. The second problem is to detect critical events from a heterogeneous spatiotemporal database. Critical events are generally rare events. Although many data mining algorithms are developed for finding common patterns, few can efficiently detect rare patterns by analyzing exceptional objects in the spatiotemporal database.

We have developed two semantics-oriented methods: the CIRCE method which includes the MicPasts and the CIRCE core algorithm for mining common patterns and the EOAFREE method for mining rare patterns from water quality dataset. We also implement the two methods into a SOMAwater prototype system to satisfy the goal of improving the precision and efficiency of querying common and rare patterns from spatiotemporal datasets to support better decision making. First, a MicPasts method explored semantics related to space and time that are discovered in this dissertation through mining the original spatiotemporal data to help remove Uncertainty due to Sampling Error (SE uncertainty). Then, we

extended the MicPasts method to the CIRCE method by replacing the DP algorithm by a novel CIRCE core algorithm to remove Uncertainty due to Discrete Sampling (DS Uncertainty) of the original data in order to improve the precision of the data and compact the data volume. Meanwhile, a novel Exceptional Object Analysis for Finding Rare Environmental Events (EOAFREE) method was provided to detect rare events (such as water pollutions). The EOAFREE method comprises three major steps. First, it provides an approach to explore semantics based on domain knowledge to unify heterogeneous real world data that are collected by different organizations. For example, a water quality index can be used to denote water quality instead of multiple different water quality parameters. Second, it defines changes in a data object in a spatiotemporal database by considering both the difference between consecutive points on the same stream and the difference between two data objects on the two streams at the same season in two consecutive years. Third, a proposed Improved Exceptional Object Analysis based on Noises (IEOAN) algorithm clusters objects based on the objects' changes, and then distinguishes those data objects (or data points) that cannot be grouped into any clusters as exceptional objects. Interestingly, opposite to the already known Principal Component Analysis that ranks principal components, IEOAN ranks exceptional objects.

The findings of this work include following three aspects:

- First, the CIRCE method [HZHa] including both the MicPasts method (also called LCRTurning in our paper in [HZHD11]) and the CIRCE core algorithm, as well as

the IEOAN algorithm [HZHb], are general solutions proposed for mining common and rare patterns from uncertain spatiotemporal data. The MicPasts method discovers semantic places by summarizing turning points. The CIRCE core algorithm is developed for improving the quality and compressing the volume of spatiotemporal data. The IEOAN algorithm is proposed for discovering and ranking exceptional objects in a spatiotemporal database.

- Second, the EOAFREE method [HZHb] is developed especially for mining water data in order to detect critical water events (e.g. water pollution).

- Third, the SOMAwater prototype system that is built on the aforementioned novel methods and algorithms can be used to automatically discover spatiotemporal common patterns (e.g., Longest Common Curve patterns) and rare patterns (e.g. continual, simultaneous, seasonal and global rare patterns) from water quality data and to support decision making for effective water management. Compared to manual and semi-manual spatiotemporal pattern analysis, SOMAwater can help decision makers to focus only on the valuable datasets and discover spatiotemporal water patterns efficiently and automatically, as well as in more detail.

VII.2 Future Work

We have introduced the mechanism (based on the CIRCE method and the EOAFEE method) and the whole graph of our SOMAwater prototype system that are used to support

decision making in the water resources management field. In our future work, we will extend it to a more general data mining tool and support more applications from the following aspects:

- We will extend the Detecting Inflexions and Computing Missing Inflexions (DICMI) algorithm in the CIRCE core to support more methods to compute the missing inflexions.

- We will provide interfaces to embed more clustering algorithms, such as K-Means [HW79], in Clustering Turning Points (CTP) module.

- We will provide services to query more common patterns, such as Sequential PAttern (SPA) etc.

- We will provide a general interface to embed more clustering algorithms, including those that do not force every data instance to belong to a cluster, such as ROCK [GRS00], Shared Nearest Neighbor (SNN) clustering [ESK03] and Findout algorithm [YQLZ02], into the IEOAN module.

Bibliography

[AGBT06] E. G. Abal, P. F. Greenfield, S.E. Bunn, and D F. Tarte. Healthy Waterways: Healthy Catchments - An Integrated Research/Management Program to Understand and Reduce Impacts of Sediments and Nutrients on Waterways in Queensland. In *Proc. Of APWeb'06/e-Water'06, LNCS, Springer*, 2006.

[ATL+05] R. Ackland, K. Taylor, L. Lefort, M. Cameron, and J. Rahman. Semantic Service Integration for Water Resource. In *ISWC2005 (LNCS 3729)*, pages 816–828, 2005.

[BHR00] L. Bergroth, H. Hakonen, and T. Raita. A Survey of Longest Common Subsequence Algorithms. In *SPIRE'00*, pages 39–48, 2000.

[Bri02] T. Brinkhoff. A Framework for Generating Network-based Moving Objects. *GeoInformatica*, 6(2):153–180, 2002.

[Bri03] T. Brinkhoff. Generating Traffic Data. *Bulletin of the Technical Committee on Data Engineering, IEEE Computer Society*, 26(2):19–25, 2003.

[CBK09] V. Chandola, A. Banerjee, and V. Kumar. Anomaly Detection: A Survey. *ACM Computing Surveys*, 41(3), 2009.

[Cha08] Heejun Chang. Spatial Analysis of Water Quality Trends in the Han River Basin, South Korea. *Water Research*, 42:3285–3304, 2008.

[CJ08] M. Choi and J. M. Jacobs. Temporal Variablility Corrections for Advanced Microwave Scanning Radiometer E (AMSR-E) Surface Soil Moisture: Case Study in Little River Region, Georgia, U.S. *Sensors*, 8:2617–2627, 2008.

[CKP07] R. Cheng, D. V. Kalashnikow, and S. Prabhakar. Evaluation of Probabilistic Queries Over Imprecise Data in Constantly-Evolving Environments. *Information Systems*, 32:104–130, 2007.

[CMC05] H. Cao, N. Mamoulis, and D. W. Cheung. Mining Frequent Spatio-Temporal Sequential Patterns. In *ICDM'05*, pages 82–89, 2005.

[CSM+08] M. A. Cerqueira, J. F. Silva, F. P. Magalhaes, F. M. Soares, and J. J. Pato. Assessment of Water Pollution in the Antua River Basin (Northwestern Portugal). *Environ. Monit. Assess*, 142, 2008.

[Dan97] G. Dan. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. USA: Cambridge University Press, 1997.

[Dat] Victorian Database. Victorian Water Resources Data Warehouse. http://www.vicwaterdata.net/.

[DH09] Z. Ding and G. Huang. Real-Time Traffic Flow Statistical Analysis Based on Network Constrainted Moving Object Trajectories. In *DEXA2009*, pages 173–183, 2009.

[DP73] D. H. Douglas and T. K. Peucker. Algorithms for the Reduction of the Number of Points Required to Represent A Digitized Line or Its Caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

[EGBD06] R. S. Iliev E. G. Bournaski, L. M. Kirilov and I. Diadovski. Decision Support for Water Quality Management. In *Proc. of International Conference on Computer Systems and Technologies - CompSysTech'06*, 2006.

[EKSX96] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *SIGKDD'96*, pages 226–231, 1996.

[EMT08] C. Elena, B. Michela, and K. Tahar. Mining Spatio-temporal Data at Different Levels of Detail. In *Proc. of the 11th AGILE International Conference on Geographic Information Science (AGILE2008), Girona, Spain, 5-8 May*, 2008.

[ESK03] L. Ertoz, M. Steinbach, and V. Kumar. Finding Topics in Collections of Documents: A Shared Nearest Neighbor Approach. *Clustering and Information Retrieval*, 2003.

[eW05] eWater 2005 Workshop. eWater 2005 Workshop, 2005. http://www.itee.uq.edu.au/ eii/workshop/ewater.php.

[FG01] U. M. Fayyad and G. G. Grinstein. Introduction. In *Information Visualization in Data Mining and Knowledge Discovery, Los Altos, CA: Morgan Kaufmann*, pages 1–17, 2001.

[GBEe00] R. H. Guting, M. H. Bohlen, M. Erwig, and etc. A Foundation for Representing and Querying Moving Objects. *ACM Transactions on Database System*, 25:1–42, 2000.

[GKS04] J. Gudmundsson, M. V. Kreveld, and B. Speckmann. Efficient Detection of Motion Patterns in Spatio-Temporal Data Sets. In *GIS'04, Nov. 12-13, Washington, DC, USA*, 2004.

[GKS07] J. Gudmundsson, M. V. Kreveld, and B. Speckmann. Efficient Detection of Patterns in 2D Trajectories of Moving Points. *GeoInformatica*, 11:195–215, 2007.

[GMFC02] C. Giupponi, J. Mysiak, A. Fassio, and V. Cogan. Towards a Spatial Decision Support System for Water Resource Management: MULINO-DSS 1st Release. In *the 5th AGILE Conference on Geographic Information Science (Palma, Spain, April 25-27)*, 2002.

[GNPP07] F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli. Trajectory Pattern Mining. In *SIGKDD'07*, pages 330–339, 2007.

[GP09] G. Gidofalvi and T. B. Pedersen. Mining Long, Sharable Patterns in Trajectories of Moving Objects. *GeoInformatica*, 13(1):27–55, 2009.

[GQZe08] M. Gao, Z. Qin, H. Zhang, and etc. Remote Sensing of Agro-droughts in Guangdong Province of China Using MODIS Satellite Data. *Sensors*, 8:4687–4708, 2008.

[Gre03] R. I. Greenberg. Bounds on the Number of Longest Common Subsequences. In *CoRR cs. DM/0301030*, 2003.

[GRS00] S. Guha, R. Rastogi, and K. Shim. ROCK: A Robust Clustering Algorithm for Categorical Attributes. *Information Systems*, 25(5):345–366, 2000.

[HFK08] K. W. Hipel, L. Fang, and D. M. Kilgour. Decision Support Systems in Water Resources and Environmental Management. *Journal of Hydrologic Engineering*, 2008.

[HHD07] G. Huang, J. He, and Z. Ding. Automatic Generation of Traditional Style Painting by Using Density-Based Color Clustering. In *ICDM Workshops 2007*, pages 41–44, 2007.

[HHD08a] G. Huang, J. He, and Z. Ding. Inter-Frame Change Directing Online Clustering of Multiple Moving Objects for Video-Based Sensor Networks. In *Web Intelligence/IAT Workshops 2008*, pages 442–446, 2008.

[HHD08b] G. Huang, J. He, and Z. Ding. Wireless Video-based Sensor Networks for Surveillance of Residential Districts. In *APWeb'08*, pages 154–165, 2008. Video Sensor Network.

[HHZS07] J. He, G. Huang, Y. Zhang, and Y. Shi. Cluster Analysis and Optimization in Color-based Clustering for Image Abstract. In *ICDM Workshops'07*, pages 213–218, 2007.

[HLT10] J. Han, Z. Li, and L. A. Tang. Mining Moving Object, Trajectory and Traffic Data. In *DASFAA2010*, pages 485–486, 2010.

[HS77] J. W. Hunt and T. G. Szymanski. A Fast Algorithm for Computing Longest Common Subsequences. *Communications of the ACM*, 20(5):350–353, 1977.

[HS92] J. Hershberger and J. Snoeyink. Speeding up the Douglas-Peucker Line-Simplification Algorithm. In *Proc. 5th SDH, vol. 1, Charleston, South Carolina, USA*, pages 134–143, 1992.

[HW79] J. A. Hartigan and M. A. Wong. A k-means Clustering Algorithm. *Applied Statistics*, 28:100–108, 1979.

[HZHa] J. He, Y. Zhang, and G. Huang. CIRCE: Correcting Imprecise Readings and Compressing Excrescent Points for Querying Common Patterns in Uncertain Sensor Streams. *Information System (The first revision made in Oct. 2011)*.

[HZHb] J. He, Y. Zhang, and G. Huang. Exceptional Object Analysis for Finding Rare Environmental Events from Water Quality Datasets. *Neurocomputing Journal (Accpeted in April 2011)*.

[HZHC11] G. Huang, Y. Zhang, J. He, and J. Cao. Fault Tolerance in Data Gathering Wireless Sensor Networks. *The Computer Journal*, 54(6):976–987, 2011.

[HZHD11] G. Huang, Y. Zhang, J. He, and Z. Ding. Efficiently Retrieving Longest Common Route Patterns of Moving Objects By Summarizing Turning Regions. In *PAKDD2011, Shenzhen, China, 24-27 May*, pages 375–386, 2011.

[HZSH10] J. He, Y. Zhang, Y. Shi, and G. Huang. Domain-Driven Classification Based on Multiple Criteria and Multiple Constraint-Level Programming for Intelligent Credit Scoring. *IEEE Trans. Knowl. Data Eng.*, 22(6):826–838, 2010.

[JGS07] A Terzis J. Gupchup, R. Burns and A Szalay. Model-based Event Detection in Wireless Sensor Networks. In *Proceedings of the Workshop on Data Sharing and Interoperability on the World-Wide Sensor Web (DSI)*, 2007.

[JYZ+08] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. Shen. Discovery of Convoys in Trajectory Databases. In *VLDB'08*, pages 1068–1080, 2008.

[KJBB09] M. C. Kerman, W. Jiang, A. F. Blumberg, and S. E. Buttrey. Event Detection Challenges, Methods and Applications in Natural and Artificial Systems. In *The 14th International Command and Control Research and Technology Symposium (Washington, DC, June 15-17)*, 2009.

[KJHK] J. Kjelds, T. Jacobsen, J. Hughes, and J. Krejcik. Decision Support Tools for Integrated Water Resources Management. International Congress on River Basin Management.

[KKE+] D. Koutsoyiannis, G. Karavokiros, A. Efstratiadis, N. Mamassis, A. Koukouvinos, and A. Christofides. A Decision Support System for the Management of the Water Resource System of Athens. *Physics and Chemistry of the Earth*, 28(14).

[KLK+07] P. R. Kannel, S. Lee, S. R. Kanel, S. P. Khan, and Y.-S. Lee. Spatial-Temporal Variation and Comparative Assessment of Water Qualities of Urban River System: A Case Study of the River Bagmati (Nepal). *Environ. Monit. Assess.*, 129:433–459, 2007.

[KLL08] P. R. Kannel, S. Lee, and Y.-S. Lee. Assessment of Spatial-Temporal Patterns of Surface and Ground Water Qualities and Factors Influencing Management Strategy of Groundwater System in An Urban River Corridor of Nepal. *Journal of Environmental Management*, 86:595–604, 2008.

[KMB05] P. Kalnis, N. Mamoulis, and S. Bakiras. On Discovering Moving Clusters in Spatiotemporal Data. In *SSTD'05*, pages 364–381, 2005.

[Kou06] Y. Kou. *Abnormal Pattern Recognition in Spatial Data*. PhD thesis, Virginia Polytechnic Institute and State University, 2006.

[LDHK11] Z. Li, B. Ding, J. Han, and R. Kays. Swarm: Mining Relaxed Teporal Moving Object Clusters. In *Proc. of the VLDB Endowment*, volume 3, pages 723–734, 2011.

[LHLG08] J.-G. Lee, J. Han, X. Li, and H. Gonzalez. TraClass: Trajectory Classification Using Hierarchical Region-Based and Trajectory-Based Clustering. In *VLDB'08*, 2008.

[LHW07] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory Clustering: A Partition-And-Group Framework. In *SIGMOD'07*, pages 593–604, 2007.

[LHY04] Y. Li, J. Han, and J. Yang. Clustering Moving Objects. In *SIGKDD'04*, pages 617–622, 2004.

[Liu04] C. W. Liu. Decision Support Systems for Management ground water resources in the Choushui River Alluvial in Taiwan. *Journal of The American Water Resources Association*, 2004.

[LZ11] Y. Li and E. Zhong. A New Vector Data Compression Approach for WebGIS. *Geo-spatial Information Science*, 14(1):48–53, 2011.

[Mai78] D. Maier. The Complexity of Some Problems on Subsequences and Supersequences. *J. ACM*, 25(2):322–336, 1978.

[Mat02] J. I. Matondo. A Comparison Between Conventional and Integrated Water Resources Planning and Management. *Physics and Chemistry of the Earth*, 27:831–838, 2002.

[McC76] E. M. McCreight. A Space-Economical Suffix Tree Construction Algorithm. *Journal of the ACM*, 23(2):262–272, 1976.

[MdB04] N. Meratnia and R. A. de By. Spatiotemporal Compression Techniques for Moving Point Objects. In *EDBT'04*, pages 765–782, 2004.

[MGR05] J. Mysiak, C. Giupponi, and P. Rosato. Towards the Development of a Decision Support System for Water Resource Management. *Environmental Modelling and Software*, 20:203–214, 2005.

[Mit10] T. Mitsa. *Temporal Data Mining*. Chapman and Hall/CRC, 2010.

[MKB09] A. Blumberg M. Kerman, W. Jiang and S. Buttrey. The Application of A Quantile Regression Meta Model for Salinity Event Detection Confirmation Within New York Harbor Oceanographic Data. *Journal of Operational Oceanography*, 2(1):49–70, 2009.

[MP80] W. J. Masek and M. S. Paterson. A Faster Algorithm for Computing String-Edit Distances. *Journal of Computer and System Sciences*, 20(1):18–31, 1980.

[MSN] MSNBC. Crews Fan Out in Texas to Search for Ike Victims, September 14, 2008. http://www.msnbc.msn.com/id/26637482/.

[NSF] NSF. Calculating NSF Water Quality Index. http://water-research.net/watrqualindex/.

[oI07] School of ITEE. Infrastructure for Large-scale Data Resource Sharing and An Environmental Case Study. Technical report, The University of Queensland, 2007.

[oSE04] Department of Sustainability and Environment. Index of Stream Condition: The Second Benchmark of Victorian River Condition. Technical Report 2004 ISC report, Department of Sustainability and Environment, 2004.

[PAN05] Z. Polkowska, A. Astel, and J. Namieoenik. Studies on Intercorrelation between Selected Persistent Organic Pollutants (POPs) and Ions in Sea Water from the Coastal Zone of the Baltic Sea, Gdansk Bay Region. *Polish Journal of Environmental Studies*, 14(5):613–630, 2005.

[PCE08] G. C. Pereira, R. Coutinho, and N. F. F. Ebecken. Data Mining for Environmental Analysis and Diagnostic: A Case Study of Upwelling Ecosystem of Arraial do Cabo. *Brazillan Journal of Oceanography*, 56(1):1–12, 2008.

[Pra05] B. Pradhan. Water Quality Classification Model in the Hindu Hush-Himalayan Region: The Bagmati River in Kathmandu Valley, Nepal. In *ICIMOD2005*, 2005.

[PZHZ07] D. Pullar, J. F. Zhang, J. Hunter, and X. Zhou. Integrative Environmental Queries Using Geospatial Web Services. In *Proc. of Conference on Spatial Information Theory (COSIT'07)*, 2007.

[QMH07] A. Quadir, R. N. Malik, and S. Z. Husain. Spatio-temporal Variations in Water Quality of Nullah AikCtributary of the River Chenab. *Pakistan. Environ. Monit. Assess.*, 140, 2007.

[RA09] N. Raj and P. A. Azeez. Spatial and Temporal Variation in Surface Water Chemistry of A Tropical River, the River Bharathapuzha, India. *Current Science*, 96(2):245–251, 2009.

[RL01] J. F. Roddick and B. G. Lees. Paradigms for Spatial and Spatio-Temporal Data Mining. In *Geographic Data Mining and Knowledge Discovery, Taylor and Francis*, pages 33–50, 2001.

[SA96] R. Srikant and R. Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. In *Proceeding of 5th Intl. Conf. Extending Database Technology*, 1996.

[SEKX98] J. Sander, M. Ester, H. P. Kregel, and X. Xu. Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.

[SKSB06] B. B. Nayak S. K. Sundaray, U. C. Panda and D. Bhatta. Multivariate Statistical Techniques for the Evaluation of Spatial and Temporal Variations in Water Quality of the Mahanadi RiverCestuarine System (India) C A Case Study. *Environ. Geochem. Health*, 28, 2006.

[SLJ+02] M. Y. Song, K. Y. Lee, S. H. Jang, C. W. Lee, C. H. Park, J. Y. Kim, and I. S. Kim. Comprehensive Water Quality Management Plan for the Imjin River Basin. In *5th International Conference on Hydro-Science and Engineering(ICHE2002)*, 2002.

[SMS+05] K. P. Singh, A. Malik, S. Sinha, V. K. Singh, and R. C. Murthy. Estimation of Source of Heavy Metal Contamination in Sediments of Gomti River (India) Using Principal Component Analysis. *Water, Air, Soil Pollut*, 166, 2005.

[SR08] R. O. Strobl and P. D. Robillard. Network Design for Water Quality Monitoring of Surface Freshwaters: A Review. *Journal of Environmental Management*, 87(4):639–648, 2008.

[SWMW09] J. Sukharev, C. Wang, K.-L. Ma, and A. T. Wittenberg. Correlation Study of Time-Varying Multivariate Climate Data Sets. In *Proc. of the 2009 IEEE Pacific Visualization Symposium*, pages 161–168, 2009.

[SZZ06] Y. Shu, J. F. Zhang, and X. Zhou. A Grid-enabled Architecture for Geospatial Data Sharing. In *Proc. of the 2006 IEEE Asia-Pacific Conference on Ser-*

vices Computing (APSCC'06), 12-15 December 2006, Guangzhou, Guangdong, China., pages 369–375, 2006.

[Tha89] K. Thapa. Data Compression and Critical Points Detection Using Normalized Symmetric Scattered Matrix. *Autocarto*, 9:78–89, 1989.

[TNGA09] I. T. Telci, K. Nam, J. Guan, and M. M. Aral. Optimal Water Quality Monitoring Network Design for River Systems. *Journal of Environmental Management*, 90(10):2987–2998, 2009.

[TTR03] H. Ince T. Trafalis and M. Richman. Tornado Detection with Support Vector Machines. In *ICCS2003*, pages 289–298, 2003. LNCS.

[U.S11] U.S.G.S. National Field Manual for the Collection of Water-Quality Data: U. S. Geological Survey Techniques of Water-Resources Investigations, 2011. Book 9, Chaps. A1-A9.

[VM10] S. Veleva and K. Mitreski. Data Analysis of Spatio-Temporal Sensor Data as A Contribution to the Model Analysis for Water Resources. In *BALWOIS 2010 (Ohrid, Macedonia, May 25-29)*, 2010.

[VV01] O. Varis and P. Vakkilainen. China's 8 Challenges to Water Resources Management in the First Quarter of the 21st Century. *Geomorphology*, 41:93–104, 2001.

[VW91] M. Visvalingam and J. D. Whyatt. The Douglas-Peucker Algorithm for Line Simplification: Re-evaluation through Visualization. *Computer Graphics Forum*, 9(3):213–228, 1991.

[Whi85] E. R. White. Assessment of Line Generalization Algorithms Using Characteristic Points. *The American Cartographer*, 12:17–27, 1985.

[WXSY06] H. Wang, L. Xu, S. Sun, and S. X. Yang. Spatial Distribution of Water Quality Based on Generalized Regression Neural Network. In *Proc. of the 2006 IEEE International Conference on Information Acquisition, Aug. 20- 23, Weihai, Shandong, China*, 2006.

[YCL+06] J. X. Yu, Z. Chong, H. Lu, Z. Zhang, and A. Zhou. A False Negative Approach to Mining Frequent Itemsets from High Speed Transactional Data Streams. *Inf. Sci.*, 176(14):1986–2015, 2006.

[YQLZ02] J. X. Yu, W. Qian, H. Lu, and A. Zhou. Finding Centric Local Outliers in Categorical/Numerical Spaces. *Knowledge and Information Systems*, 9(3):309–338, 2002.

[ZGZ09] C. Zhang, M. Gao, and A. Zhou. Tracking High Quality Clusters Over Uncertain Data Streams. In *ICDE'09*, pages 1641–1648, 2009.