

Multi-Purpose Boundary-Based Clustering on Proximity Graphs for Geographical Data Mining

by

Ickjai Lee

BSc., Computer Science, Pusan National University, Korea, 1993.

Grad. Dips., Spatial Science, Curtin University of Technology, Australia, 1997.

BSc. (Hons), Spatial Science, Curtin University of Technology, Australia, 1998.

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy



School of Electrical Engineering and Computer Science
The University of Newcastle
Callaghan NSW 2308
Australia

© Ickjai Lee, February 2002

Certificate of Originality

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree at any other University or Institution.

(Signed) _____
Ickjai Lee

Approval

Name: Ickjai Lee
Degree: Doctor of Philosophy
Thesis Title: Multi-Purpose Boundary-Based Clustering on
Proximity Graphs for Geographical Data Mining

Examining Committee: Dr. Vladimir Estivill-Castro (Supervisor)
The University of Newcastle
Australia

Dr. Alan Murray (Co-supervisor)
The Ohio-State University
U.S.A.

Dr. Harvey J. Miller (External Examiner)
The University of Utah
U.S.A.

Dr. John F. Roddick (External Examiner)
Flinders University of South Australia
Australia

Dr. Brian G. Lees (External Examiner)
The Australian National University
Australia

Acknowledgements

I would like to express my sincere thanks to my supervisor, Associate Professor Vladimir Estivill-Castro, for invaluable feedback, enthusiasm, tireless reviewing, encouragement, collaboration and willingness to provide assistance. His support towards my participation in international conferences, and commitment to an environment with adequate equipment and facilities is appreciated. Without his support, it would have been impossible to complete this thesis. I also would like to thank my co-supervisor Alan Murray for his wise advice and support.

I would like to thank Jenifer Purdi, Manager of Statistical Services from the Queensland Police Service, for providing real datasets. Also, I would like to give special thanks to the Australian Research Council (ARC) who provided a Ph.D. scholarship through the ARC large project titled “Clustering for Knowledge Discovery, Pattern Spotting and Exploratory Analysis in Spatial Databases” with chief investigator Vladimir Estivill-Castro and project id A49918061.

I also wish to thank all members of Data Mining and Knowledge Discovery Research Group in the University of Newcastle, in particular Jianhua Yang, Stephan Chalup and Rodolfo Torres-Velazquez, for fruitful discussions and collaboration. Appreciation also includes all members of Machine Learning Journal Club for discussions on current issues and on technical advances in the area of machine learning.

I would like to thank the faculty, staff and fellow postgraduates in the School of Electrical Engineering and Computer Science for their hospitality and kindness. They provided a wonderful atmosphere where I felt comfortable to work. They were highly supportive and helpful.

I acknowledge the support of my parents, Kangsu Lee and Bokryun Cho, and my sisters and brothers. They understand little what this is but are proud nonetheless.

Their endless love has also made this thesis possible.

Last but not least, I am deeply grateful to my wife, Kyungmi Lee, and my daughter, Eojin Lee. They gave me inspiration and focus to complete this thesis and kept me working hard.

To my wife, **Kyungmi (Joanne)**, and
my daughter, **Eojin (Jinny)**.

Publications arising from this thesis

International Journals with full paper refereed

1. Vladimir Estivill-Castro and Ickjai Lee [41]. “Argument Free Clustering via Boundary Extraction for Massive Point-data Sets.” *Computers, Environments and Urban Systems*, Pergamon, Elsevier Science, 26(4):315-334, 2002. ISSN 0198-9715.
2. Vladimir Estivill-Castro and Ickjai Lee [42]. “Multi-Level Clustering and its Visualization for Exploratory Spatial Analysis.” *Geoinformatica*, Kluwer Academic Publishers, 6(2):123-152, 2002. ISSN 1384-6175.

International conferences and workshops with full paper refereed

1. Vladimir Estivill-Castro and Ickjai Lee [36]. “AMOEBA: Hierarchical Clustering Based on Spatial Proximity Using Delaunay Diagram.” In P. Forer, A. G. O. Yeh, and J. He, editors, *Proceedings of the 9th International Symposium on Spatial Data Handling (SDH)*, pages 7a.26-7a.41, Beijing, China, 2000.
2. Vladimir Estivill-Castro and Ickjai Lee [37]. “AUTOCLUST+: Automatic Clustering of Point-Data Sets in the Presence of Obstacles.” In J. F. Roddick and K. Hornsby, editors, *Proceedings of the International Workshop on Temporal, Spatial and Spatio-Temporal Data Mining (TSDM)*, in conjunction with the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), Lecture Notes in Artificial Intelligence 2007, pages 133-146, Lyon, France, 2000. Springer-Verlag. ISBN 3-540-41773-7.
3. Vladimir Estivill-Castro, Ickjai Lee, and Alan Murray [43]. “Criteria on Proximity Graphs for Boundary Extraction and Spatial Clustering.” In D. Cheung,

- Q. Li, and G. Williams, editors, *Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, Lecture Notes in Artificial Intelligence 2035, pages 348-357, Hong Kong, China, 2001. Springer-Verlag. ISBN 3-540-41910-1.
4. Ickjai Lee [88]. “MESCAT: Multi-purpose Exploratory Spatial Clustering Analysis Toolkit.” In J. Fong and M. K. Ng, editors, *Proceedings of the International Workshop on Mining Spatial and Temporal Data*, in conjunction with the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pages 112-121, Hong Kong, China, 2001. City University of Hong Kong. ISBN 962-442-187-0.
 5. Ickjai Lee and Vladimir Estivill-Castro [89]. “Effective and Efficient Boundary-based Clustering for Three-dimensional Geoinformation Studies.” In H. Lu and S. Spaccapietra, editors, *Proceedings of the 3rd International Symposium on Cooperative Database Systems for Advanced Applications (CODAS)*, pages 87-96, Beijing, China, 2001. IEEE Computer Society. ISBN 0-7695-1128-7.
 6. Vladimir Estivill-Castro and Ickjai Lee [40]. “Fast Spatial Clustering with Different Metrics and in the Presence of Obstacles.” In W. G. Aref, editor, *Proceedings of the 9th International Symposium on Advances in Geographic Information Systems (ACM-GIS)*, pages 142-147, Atlanta, GA, 2001. ACM Press. ISBN 1-58113-443-6.
 7. Ickjai Lee and Vladimir Estivill-Castro [90]. “Polygonization of Point Clusters through Cluster Boundary Extraction for Geographical Data Mining.” In D. Richardson and P. v. Oostrom (editors), *Proceedings of the 10th International Symposium on Spatial Data Handling (SDH)*, pages 27-40, Ottawa, Canada, 2002. Springer-Verlag. ISBN: 3-540-43802-5.

International conferences and workshops with abstract refereed

1. Vladimir Estivill-Castro and Ickjai Lee [38]. “AUTOCLUST: Automatic Clus-

- tering via Boundary Extraction for Mining Massive Point-Data Sets.” In J. Abraham and B. H. Carlisle, editors, *Proceedings of the 5th International Conference on Geocomputation*, Kent, UK, 2000. Geocomputation CD-ROM: GC049. ISBN 0-9533477-2-9.
2. Vladimir Estivill-Castro and Ickjai Lee [39]. “Data Mining Techniques for Autonomous Exploration of Large Volumes of Geo-referenced Crime Data.” In D. V. Pullar, editor, *Proceedings of the 6th International Conference on Geocomputation*, Brisbane, Australia, 2001. Geocomputation CD-ROM: estivillcastro.pdf. ISBN 1864995637.

Notation

Mathematical symbols

- R - a study region.
- P - a set of points within R .
- n - the number of points in P .
- $\|P\|$ - the number of points in P .
- $O(f(n))$ - a function bounded from above by $f(n)$.
- $\Omega(f(n))$ - a function bounded from below by $f(n)$.
- $\Theta(f(n))$ - both $O(f(n))$ and $\Omega(f(n))$.
- p_i - the i -th point in P .
- C_i - the i -th cluster in P .
- m_i - the mean of C_i .
- $dis(p_i, p_j)$ - dissimilarity between p_i and p_j .
- SE - the squared-error criterion.
- $|e|$ - the length of edge e .
- $d(p_i)$ - the number edges incident to point p_i .
- $E[d(p_i)]$ - an expected value of $d(p_i)$.
- $N(p_i)$ - a set of edges incident to point p_i .
- $Local_Mean(p_i)$ - the mean length of edges incident to point p_i .
- $Local_St_Dev(p_i)$ - the standard deviation of the lengths of edges in $N(p_i)$.

- $Mean_St_Dev(P)$ - the average of $Local_St_Dev(p_i)$ values for all p_i in P .
- G - a graph with vertices and edges.
- m - a control value.
- l - a scaling factor.
- $m \times Mean_St_Dev(P)$ - a tolerance value.
- $e_{i,j}$ - an edge connecting two points p_i and p_j .
- $\Re$ - set or real numbers.
- $\Re^2$ - bi-dimensional Euclidean space.
- $\Re^3$ - three-dimensional Euclidean space.
- $CC[p_i]$ - the connected component of point p_i in P .
- λ - intensity.
- L - a line segment defining an obstacle.
- s - the number of line segments that determine a set of obstacles.
- t - the total number of Delaunay edges removed from the Delaunay diagram.
- $VD(P)$ - the Voronoi diagram of P in the Euclidean metric.
- $DT(P)$ - the Delaunay triangulation of P in the Euclidean metric.
- $DD(P)$ - the Delaunay diagram of P in the Euclidean metric.
- L_p - the p -th Minkowski metric.
- L_1 - the Manhattan metric.
- L_2 - the Euclidean metric.
- L_∞ - the Dominance metric.

- $VD_{L_p}(P)$ - the Voronoi diagram of P in the p -th Minkowski metric.
- $DD_{L_p}(P)$ - the Delaunay diagram of P in the p -th Minkowski metric.
- $GG(P)$ - the Gabriel graph of P .
- $RNG(P)$ - the relative neighborhood graph of P .
- $GT(P)$ - the greedy triangulation of P .
- $MST(P)$ - the Minimum Spanning Tree of P .
- $d - DG(P)$ - the d -distance neighbor graph of P .
- $k - NN(P)$ - the k -nearest neighbor graph of P .
- $k - UNN(P)$ - the undirected k -nearest neighbor graph of P .
- $k - CSG(P)$ - the k -cone spanner graph of P .
- $k - UCSG(P)$ - the undirected k -cone spanner graph of P .
- $CH(P)$ - the convex hull of P .
- X - the antecedent in an association rule.
- Y - the consequent in an association rule.
- M - the number of rows in a table.
- N - the number of columns in a table.
- CwC - Confidence with Clusters.
- SwC - Support with Clusters.

Acronyms

- BSM - Boundary Shape Matching.

- CDA - Confirmatory Data Analysis.
- CF - Clustering Feature.
- CF-Tree - Clustering Feature Tree.
- CGAL - Computational Geometry Algorithms Library.
- CSAR - Clustered Spatial Association Rule.
- CSR - Complete Spatial Randomness.
- EDA - Exploratory Data Analysis.
- EM - Expectation Maximization.
- FBI - Federal Bureau of Investigation.
- GDM - Geographical Data Mining.
- GIS - Geographic Information Systems.
- GITA - Geo-spatial Information & Technology Association.
- GPS - Global Positioning System.
- KDD - Knowledge Discovery in Databases.
- LBS - Location-Based Service.
- LEDA - Library of Efficient Data Types and Algorithms.
- MAUP - Modifiable Areal Unit Problem.
- NCGIA - National Center for Geographic Information and Analysis.
- NRMA - National Roads and Motorists' Association.
- PDAs - Personal Digital Assistants.
- SDMS - Spatial Database Management Systems.

- UML - Unified Modeling Language.
- VDA - Visual Data Analysis.

Contents

List of Figures	xviii
List of Tables	xxii
1 Introduction	1
1.1 Geographical data mining	1
1.2 Motivation	5
1.3 Contributions	7
1.4 Outline of the thesis	9
2 Related Work	12
2.1 Local clustering <i>vs.</i> global clustering	13
2.2 Dynamic model clustering <i>vs.</i> static model clustering	15
2.3 Fuzzy clustering <i>vs.</i> crisp clustering	15
2.4 Hierarchical clustering <i>vs.</i> partitioning clustering	16
2.4.1 Hierarchical clustering	17
2.4.2 Partitioning clustering	22
2.5 Density-based clustering	26
2.6 Graph-based clustering	35
2.7 Model-based clustering	37
2.8 Remarks	38
3 Multi-Purpose Exploratory Spatial Clustering	40
3.1 Requirements for spatial clustering	40

3.2	Traditional peak-inference clustering fails	48
3.2.1	Sparse clusters adjacent to high-density clusters	49
3.2.2	Multiple chains between clusters	52
3.2.3	Closely located high-density clusters	53
3.3	Multi-purpose exploratory boundary-based spatial clustering (a framework)	55
3.4	Remarks	57
4	Short-Long Clustering Criteria	58
4.1	Delaunay diagrams for spatial clustering	58
4.1.1	Modeling points with Delaunay triangulations	59
4.1.2	Avoiding ambiguities in Delaunay triangulations	61
4.1.3	Qualities of Delaunay diagrams for spatial clustering	63
4.2	Short-Long clustering criteria	65
4.2.1	Working principle	65
4.2.2	Intuition and definitions for point-centric statistics	67
4.2.3	A three-phase Short-Long clustering algorithm	70
4.3	Control value m	75
4.3.1	One standard deviation of the mean	75
4.3.2	Determination of the control value m	78
4.3.3	The control value m as an exploratory tool	82
4.4	Time complexity analysis	83
4.4.1	Short-Long clustering only requires $O(n \log n)$ time	83
4.4.2	CPU time comparison	84
4.5	Performance evaluation	85
4.5.1	Clustering quality with stochastic datasets	86
4.5.2	Clustering quality with synthetic datasets	87
4.5.3	Clustering quality with real datasets	89
4.6	Remarks	90
5	Short-Long Clustering with Various Structural Models	92
5.1	In the presence of obstacles	93

5.1.1	Obstacles emerge in many spatial settings	93
5.1.2	Detour distance and detour path	97
5.1.3	Algorithms of obstacle Short-Long clustering	98
5.1.4	Time complexity analysis	106
5.1.5	Performance evaluation	107
5.1.6	Discussion	109
5.2	Minkowski metrics	112
5.2.1	Structural models in the Minkowski metric	113
5.2.2	Determination of the control value m	114
5.2.3	Obstacle Short-Long clustering in Minkowski metrics	117
5.2.4	Time complexity analysis	120
5.2.5	Performance evaluation	120
5.3	With other proximity graphs	123
5.3.1	Proximity graphs are explicit models of neighborhoods	124
5.3.2	Performance evaluation	129
5.4	Remarks	134
6	Short-Long Clustering in Higher Dimensions	135
6.1	Benefits from three-dimensional clustering	136
6.2	Determination of the control value m	138
6.2.1	The control value m for three-dimensional Delaunay diagrams	138
6.2.2	The control value m for undirected k -nearest neighbor graphs	140
6.3	Time complexity analysis	142
6.3.1	With three-dimensional Delaunay diagrams	142
6.3.2	With undirected k -nearest neighbor graphs	143
6.4	Performance evaluation	143
6.4.1	Clustering with three-dimensional Delaunay diagrams	144
6.4.2	Clustering with undirected k -nearest neighbor graphs	145
6.5	Remarks	149
7	Multi-level Short-Long Clustering	151
7.1	Multi-level decompositions	151

7.2	Intuition and algorithms	154
7.3	Definitions	157
7.4	Visualization with weighted dendrogram	159
7.5	Time complexity analysis	160
7.6	Performance evaluation	161
7.7	Remarks	163
8	Cluster Polygonization for Multivariate Associations	166
8.1	Cluster polygonization	166
8.1.1	Shapes of clusters	167
8.1.2	Cluster polygonization algorithm	169
8.1.3	Time complexity analysis	175
8.1.4	Performance evaluation	175
8.1.5	Applications	177
8.2	Mining multivariate positive associations using cluster polygonization	181
8.2.1	Spatial association rules	181
8.2.2	Algorithms for mining multivariate positive associations	183
8.2.3	Performance evaluation	190
8.2.4	Discussion	195
8.3	Remarks	196
9	Conclusions and Future Work	198
9.1	Summary	198
9.2	Implementation issues	201
9.3	Immediate avenues for further research	202
	Bibliography	206

List of Figures

1-1	Generic schema for spatial-dominant generalization	5
2-1	Local clustering and global clustering	14
2-2	Fuzzy clustering and crisp clustering	16
2-3	BIRCH's tree structure	19
2-4	An overview of CHAMELEON	21
2-5	An overview of DBSCAN	28
2-6	An overview of GAM	30
2-7	An overview of STING	31
2-8	An overview of CLIQUE	34
3-1	Neighbor modeling of point data with raster and vector approaches .	42
3-2	Data that illustrates the importance of relative proximity in spatial clustering	45
3-3	A sparse cluster adjacent to high-density clusters	51
3-4	Multiple chains between clusters	52
3-5	Closely located high-density clusters	54
3-6	Framework of exploratory boundary-based spatial clustering using Template-Method Pattern	55
4-1	The modeling and structuring of point data with the Voronoi diagram and Delaunay triangulation	60
4-2	Two possible triangulations when four points are cocircular	62
4-3	Illustration of the characteristics of border and chain points	66
4-4	The three-phase Short-Long clustering procedure	71

4-5	Distributions (as histograms) of lengths of Delaunay edges for CSR and for datasets exhibiting clusters	77
4-6	Expected profile of edge lengths in proximity graphs.	78
4-7	Expected profiles of point-centric statistics with CSR datasets and datasets exhibiting clusters without noise	79
4-8	Profiles of point-centric statistics for Delaunay diagrams in the Euclidean metric with CSR dataset and dataset exhibiting clusters . . .	81
4-9	Comparison of CPU times	85
4-10	Clustering results of stochastic datasets	86
4-11	Clustering results of synthetic datasets	88
4-12	Clustering results of real datasets	90
5-1	Impact on clustering results when incorporating obstacles	94
5-2	Locational optimization problem in the presence of obstacles when the number of clusters is unknown	97
5-3	Detour distance and detour path	98
5-4	Voronoi regions change in the presence of obstacles, but neighborhood information may not change in the constrained Voronoi diagram . . .	101
5-5	Short-Long clustering in the presence of obstacles with a real dataset from Sydney, Australia	108
5-6	Short-Long clustering in the presence of obstacles with synthetic datasets	110
5-7	Real data from the city of Brisbane illustrating unsuitability of the Euclidean metric	112
5-8	Neighborhoods defined by Voronoi diagrams in Minkowski metrics . .	114
5-9	Profiles of point-centric statistics for Delaunay diagrams in the Manhattan metric and the Dominance metric with CSR dataset	115
5-10	Profiles of point-centric statistics for Delaunay diagrams in the Manhattan metric and the Dominance metric with dataset exhibiting clusters	116
5-11	Delaunay diagrams in the Minkowski metric with data exhibiting various types of obstacles	118
5-12	Clustering with Minkowski metrics in the presence of obstacles	122
5-13	Subgraphs of the Delaunay triangulation	125
5-14	A comparison between the Delaunay triangulation and the greedy triangulation	126

5-15	Illustration of k -nearest neighbor graphs and k -cone spanner graphs .	127
5-16	Clustering a dataset exhibiting a sparse cluster near high-density clusters using various proximity graphs	130
5-17	Clustering a dataset exhibiting multiple chains that link clusters using various proximity graphs	131
5-18	Clustering a dataset exhibiting closely located high-density clusters using various proximity graphs	132
5-19	Example where clustering results with the Delaunay diagram and the greedy triangulation differ significantly	132
6-1	Illustration of positive-clusters	137
6-2	Illustration of a negative-cluster	137
6-3	Use of the three-dimensional Delaunay diagram to analyze three-dimensional data exhibiting no clusters	139
6-4	Use of the three-dimensional Delaunay diagram to analyze three-dimensional data exhibiting clusters	140
6-5	Use of undirected 16-nearest neighbor graph to analyze three-dimensional data exhibiting no clusters	141
6-6	Use of undirected 16-nearest neighbor graph to analyze three-dimensional data exhibiting clusters	142
6-7	Use of the three-dimensional Delaunay diagram to cluster a dataset exhibiting clusters in the absence of noise	144
6-8	Use of the three-dimensional Delaunay diagram to cluster a dataset exhibiting clusters in the presence of noise	145
6-9	Use of 8-nearest neighbor graph to cluster a dataset exhibiting clusters in the absence of noise	146
6-10	Use of 16-nearest neighbor graph to cluster a dataset exhibiting clusters in the absence of noise	146
6-11	Use of 8-nearest neighbor graph to cluster a dataset exhibiting clusters in the presence of noise	148
6-12	Use of 16-nearest neighbor graph to cluster a dataset exhibiting clusters in the presence of noise	148
7-1	Point dataset representing an inherent hierarchical data structure . .	153
7-2	Effect of two approaches to delineate multi-level clusters: reconstruction <i>vs</i> recuperation	155

7-3	Multi-level Short-Long clustering applied to a real dataset	162
7-4	The dendrogram of single-linkage hierarchical clustering using the Minimum Spanning Tree	164
8-1	Cluster polygonization through cluster boundary extraction	169
8-2	Cluster boundary extraction when intra-cluster edges are external Delaunay edges	172
8-3	Cluster boundary extraction when intra-cluster edges are internal Delaunay edges	173
8-4	Oriented cluster boundaries in synthetic datasets exhibiting clusters .	176
8-5	Polygons of clusters in synthetic datasets exhibiting clusters	177
8-6	Choropleth maps with cluster polygons as a base map	179
8-7	Cluster reasoning	180
8-8	Mining multivariate associations: a vertical-view approach <i>vs</i> a horizontal-view approach	184
8-9	An example of the vertical-view approach with 4 cells for mining multivariate association rules	185
8-10	An example of the vertical-view approach with 16 cells for mining multivariate association rules	187
8-11	Real datasets for mining multivariate associations with the horizontal-view approach	192
8-12	Illustration of the horizontal-view approach with real datasets	193
9-1	Graphical user interface of Short-Long clustering towards mining associations	202

List of Tables

2.1	Comparison of well-known clustering methods.	39
4.1	Comparison of properties of the Delaunay diagram for clustering. . .	64
5.1	Comparison of point-centric statistics in the four proposed clustering methods in the presence of obstacles	105
5.2	Delaunay edge removal when obstacles appear in Minkowski metrics .	120
5.3	Characteristics of proximity graphs.	128
8.1	A 4×4 Boolean table.	186
8.2	A 16×4 relational table.	187
8.3	Clustered spatial association rules discovered by the horizontal-view approach	194

Abstract

With the growth of geo-referenced data and the sophistication and complexity of spatial databases, data mining and knowledge discovery techniques become essential tools for successful analysis of large spatial datasets. Spatial clustering is fundamental and central to geographical data mining. It partitions a dataset into smaller homogeneous groups due to spatial proximity. Resulting groups represent geographically interesting patterns of concentrations for which further investigations should be undertaken to find possible causal factors.

In this thesis, we propose a spatial-dominant generalization approach that mines multivariate causal associations among geographical data layers using clustering analysis. First, we propose a generic framework of multi-purpose exploratory spatial clustering in the form of the Template-Method Pattern. Based on an object-oriented framework, we design and implement an automatic multi-purpose exploratory spatial clustering tool. The first instance of this framework uses the Delaunay diagram as an underlying proximity graph. Our spatial clustering incorporates the peculiar characteristics of spatial data that make space special. Thus, our method is able to identify high-quality spatial clusters including clusters of arbitrary shapes, clusters of heterogeneous densities, clusters of different sizes, closely located high-density clusters, clusters connected by multiple chains, sparse clusters near to high-density clusters and clusters containing clusters within $O(n \log n)$ time. It derives values for parameters from data and thus maximizes user-friendliness. Therefore, our approach minimizes user-oriented bias and constraints that hinder exploratory data analysis and geographical data mining.

Sheer volume of spatial data stored in spatial databases is not the only concern. The heterogeneity of datasets is a common issue in data-rich environments, but left open by exploratory tools. Our spatial clustering extends to the Minkowski metric in the absence or presence of obstacles to deal with situations where interactions between spatial objects are not adequately modeled by the Euclidean distance. The genericity

is such that our clustering methodology extends to various spatial proximity graphs beyond the default Delaunay diagram. We also investigate an extension of our clustering to higher-dimensional datasets that robustly identify higher-dimensional clusters within $O(n \log n)$ time. The versatility of our clustering is further illustrated with its deployment to multi-level clustering. We develop a multi-level clustering method that reveals hierarchical structures hidden in complex datasets within $O(n \log n)$ time. We also introduce weighted dendrograms to effectively visualize the cluster hierarchies.

Interpretability and usability of clustering results are of great importance. We propose an automatic pattern spotter that reveals high level description of clusters. We develop an effective and efficient cluster polygonization process towards mining causal associations. It automatically approximates shapes of clusters and robustly reveals asymmetric causal associations among data layers. Since it does not require domain-specific concept hierarchies, its applicability is enhanced.

Chapter 1

Introduction

This chapter presents the need for data mining, especially in geo-referenced datasets, and summarizes the motivation behind this thesis. Then, it summarizes its contributions and outlines its contents.

1.1 Geographical data mining

Data collection had long been one of the most expensive procedures in Geographic Information Systems (GIS) [20]. Now, this data gathering process has become more accurate, faster, easier and cheaper due to the advances in data acquiring technologies such as Global Positioning System (GPS), digital remote sensing, aerial photography, automatic or semi-automatic scanners and bar-code readers. Thus, GIS has shifted from a data-poor and computation-poor to a data-rich and computation-rich environment [102, 103]. The immensity of digital spatial data stored in GIS or Spatial Database Management Systems (SDMS) turns spatial analysis into a complex task that far exceeds the capability of human analysis. This large volume of spatial data greatly demands intelligent, efficient and effective automatic or semi-automatic pattern spotters that suggest highly likely plausible hypotheses.

In such data-rich environments, conventional statistical analysis is neither suitable nor applicable [81, 102, 103, 125]. In data-poor environments, the hypothesis space is relatively small and limited. Thus, comprehensive search for possible hypotheses is computationally feasible. That is, emphasis is on evaluating suggested hypotheses. Traditionally, generating hypotheses is based on visual analysis and classical statistical analysis guided by the experts' background knowledge and experience. This guided analysis generates hypotheses near expected patterns. Therefore, this approach is unable to suggest totally unexpected patterns since it is not data-driven. As the volume of digital data grows explosively, the hypothesis space becomes huge, and thus the number of possible patterns is very large. Some patterns may lie beyond the experts' imagination. In data-rich environments, the emphasis must be placed on generating highly likely plausible hypotheses rather than evaluating them. Data mining is a suitable technology that summarizes massive volumes of data and suggests a reduced number of valuable patterns. Data mining re-examines the process of scientific formulation of knowledge [123]. The generic cycle of hypothesis formulation, experimental design, data collection and hypothesis refutation finds alternative paths, since much of the data is collected without a preconceived hypothesis [111, 123].

Although data mining is sometimes regarded as a synonym for Knowledge Discovery in Databases (KDD), we adhere here to well-established definitions:

“KDD is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data” [48].

KDD is a series of processes involving several interactive and iterative steps that may require quantitative inference and suggest unexpected patterns. The discovered unseen patterns are beneficial, understandable and applicable to new datasets with some degree of certainty. The KDD process consists of three main steps: pre-mining, data mining and post-mining. Pre-mining selects a subset of data to mine, cleans this subset by removing redundancy and incomplete data, and transforms it to a suitable format for the algorithms at hand. Data mining is the core step in KDD.

The application of pattern location algorithms is an exploratory technology that suggests highly likely plausible hypotheses as opposed to a confirmatory technology that evaluates hypotheses. Post-mining interprets and evaluates discovered patterns, possibly performing some validation. Thus, data mining enlarges the possibilities and complements Visual Data Analysis (VDA) and Exploratory Data Analysis (EDA) while post-mining corresponds to Confirmatory Data Analysis (CDA) in traditional interactive and iterative data analysis [10].

Geographical Data Mining (GDM) is data mining applied to geo-referenced data-sets. GDM is also often referred as spatial data mining in the data mining community [82, 125]. Spatial data mining is broader than GDM in the sense that ‘spatial’ is broader than ‘geographical’. The space in spatial data mining is a metric space, but the space in GDM is a metric and geographical space. Thus, GDM must consider the peculiar nature of geoinformation, but this is not always the case in spatial data mining. In particular, this thesis focuses on GDM that deals with geo-referenced datasets. In this thesis, unless otherwise noted, ‘spatial’ will correspond with ‘geographical’ and ‘geo-spatial’.

Openshaw [113] defines GDM as follows:

“GDM is a special type of data mining that seeks to perform similar generic functions as conventional data mining tools, but modified to take into account the special features of geoinformation, the rather different styles and needs of analysis and modeling relevant to the world of GIS, and peculiar nature of geographical explanation.”

This definition emphasizes the importance and uniqueness of geographical space. GDM must consider the peculiar characteristics of geoinformation that make space special in order to detect geographically interesting patterns that are either descriptive or predictive [61].

GDM techniques (see [122] for an updated bibliography of temporal, spatial, and spatio-temporal data mining research) can be classified according to tasks and meth-

ods as follows [47, 48, 61, 97, 103]:

- Spatial clustering: identifying a set of spatial aggregations with similar characteristics that summarize and describe the data (descriptive pattern);
- Spatial characterization: generalizing the data to find compact descriptions (descriptive pattern);
- Spatial deviation detection: finding outliers that deviate from spatial aggregations (descriptive pattern);
- Spatial classification: assigning data to one of a set of predefined classes (predictive pattern);
- Spatial trend detection: detecting changes and trends along a spatial dimension (predictive pattern);
- Spatial association: finding interesting dependencies among attributes (predictive pattern).

The first three are descriptive pattern mining tools that summarize and characterize the data, while the last three are predictive pattern mining tools that draw inferences from the data for predictions.

GDM, as a quantitative study of geo-referenced phenomena, explicitly focuses on the spatial extent, including location, distance, interaction and spatial arrangement [7, 10]. Thus, spatial-dominant generalization, where identifying spatial clusters is a fundamental and core operation, is more important than aspatial-dominant generalization [81, 82, 125]. In spatial-dominant generalization, clustering works as a summarization or generalization operation on the spatial component of the data. Later, an agent (computer or human) can explore associations between layers (themes or coverages) [39, 44, 80, 130, 133, 137]. In fact, other layers are explored to attempt to find associations that may explain the concentrations found by the clustering algorithm. This is illustrated in Figure 1-1.

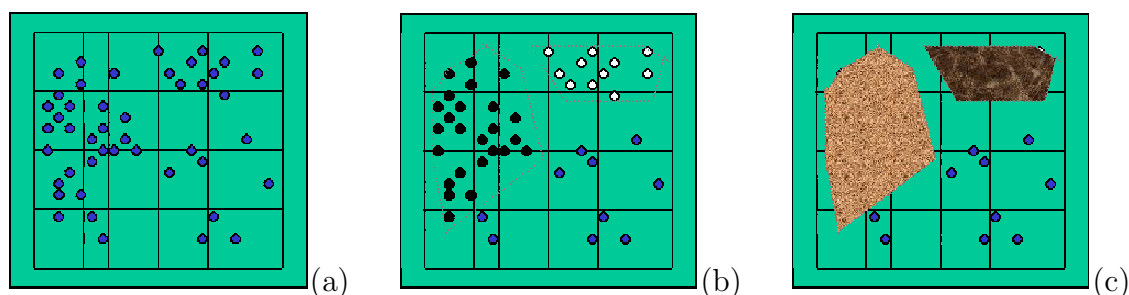


Figure 1-1: Generic schema for spatial-dominant generalization; (a) Selecting a target layer; (b) Clustering the target layer; (c) Searching for high level description.

Since clustering is used in this approach to extract point patterns in one layer towards association analysis between layers, clustering results must be of high-quality. Otherwise, spatial associations between layers are undetected. Thus, spatial clustering is of particular importance in spatial-dominant generalization.

This thesis investigates fast and robust spatial-dominant generalization that is based on spatial clustering and spatial association analysis. Spatial clusters are spatial concentrations or aggregations. We investigate other geographical layers to find possible causal factors by association mining analysis. In this thesis, spatial clustering focuses on point data, but spatial association deals with point, line and area data.

1.2 Motivation

There are many indicators of the exponential growth of spatial databases associated with GIS. GIS represents today one of the more mature solutions in the enterprise arena. For the most part, customer satisfaction is at an all-time high. The Geo-spatial Information & Technology Association (GITA) publishes an annual study (the geo-spatial technology report) that with respect to 1998 seems to confirm this claim. The report surveys organizations that use geo-spatial technology and tracks its status. The

evidence is that customers and applications for GIS continue to grow. Moreover, GIS technology has been strengthening its roles in the private sector. While the original niche for GIS was in local and federal government, science, and resource management, now GIS is commercially recognized as a powerful business tool for decision making. This comes down to all aspects of government and private enterprise having to manage information that is associated with a geographical location.

Robertson [121] indicates that interactive field-based GIS and Internet mapping applications will continue to enhance the availability of GIS applications, and they will continue to expose the technology to common users. According to Robertson's figures the global GIS market is expected to expand from \$3.2 billion (USD) in 1997 to more than \$5.5 billion in 2002, with an additional \$1.5 billion attributed to the high-resolution remote sensing industry, of which aerial imagery is the largest contributor. McKee [100] points out how geo-spatial information on the WEB is getting bigger and easier to obtain and mentions the expansion and need for data mining based analysis. Even late in 2001, after the so called "slow-down", reports [146] indicate that GIS and geo-technologies are still growing, and that they will serve as a strategic advantage to core markets. These views indicate that many companies that serve specific vertical markets view GIS as a means to broaden their influence and increase their ability to add value to their customers.

This can only fuel more data collection, and more pressure for efficient and effective data analysis. GIS is called to complement traditional roles of data mining in customer relationship management like customer satisfaction, targeted marketing, and customer retention [68]. This relationship to customers and the geographical world is going to be examined by the emerging Location-Based Service (LBS) market. LBS technology aims to provide specific, targeted information to users, based on each specific user's location at any time. While this technology creates an environment in which spatial information can be delivered to consumers via digital devices such as lap-tops, palm-tops, Personal Digital Assistants (PDAs) or cell phones, it can produce even more geo-referenced data.

Some infrastructures initiatives reflect the growing availability of geo-referenced data in electro-magnetic media. Many involve major collective efforts in many of the traditional areas of GIS, but incorporating even richer data environments. An example is the integration of spatial infrastructure initiatives that have been progressing for a few years at the state or regional level (see for example, the Georgia spatial data infrastructure [<http://www.gis.state.ga.us/>] or the Queensland spatial infrastructure [<http://www.qsiis.qld.gov.au/>]), at the national level (see for example, the survey of national and regional spatial data infrastructure activities around the globe [<http://www.spatial.maine.edu/~onsrud/GSDI.htm>] and at the international level (see for example, projects by European umbrella organizations for geographic information [<http://www.eurogi.org/>])).

This explosive growth in data cannot be matched by the current capabilities for analysis available in GIS technology. That is, the advances in our technology for collecting, managing and communicating geo-referenced data far exceed the technological capability to analyze it. This abundance of data without adequate analysis tools has already been identified as the constructor of “data-rich but information-poor environments”. These environments are moving to the spatial databases associated with GIS. The motivation of this thesis is to alleviate this trend by enhancing the exploratory analysis capabilities of GIS.

1.3 Contributions

The core objective of this thesis is to develop an effective and efficient paradigm for GDM, emphasizing the exploratory nature of KDD while incorporating the special characteristics of spatial data. To this end, we assume a rich environment of point data layers and line and area layers in which it is important to minimize the need for user-supplied arguments, but the system is capable of automatically producing a descriptive and meaningful set of patterns. We describe the patterns as spatial clusters, but we also provide an approach to obtain indicative descriptions for the

patterns as hierarchies of clusters and as spatial associations between layers.

Therefore, this thesis provides several advances:

1. An analysis of the modeling alternatives available for representing spatial proximity on point data. We postulate the Delaunay diagram as a concise representation of proximity and neighborhood information that encodes all necessary locality information. The Delaunay diagram data structure is manageable for large datasets because it can be computed efficiently and it requires linear storage and logarithmic time for updates. The argument-free nature of Delaunay diagrams not only fosters the postulate of “letting the data speak by themselves” [112], but suits data-rich environments where tuning arguments is computationally expensive and undesirable.
2. The design of a general clustering criterion based on proximity graphs whose computational time requirements are linear on the size of the underlying proximity graph. The clustering criterion is argument-free but can be turned into an exploratory tool with user interaction to explore the fragility and rigidity of clusters.
3. The design and implementation of a clustering algorithm that uses the clustering criterion to obtain clusters in complex scenarios that include non-convex clusters, clusters in the presence of noise, clusters of different sizes, clusters of arbitrary shapes, clusters of heterogeneous densities, clusters connected by multiple linear links, sparse clusters near to high-density clusters and closely located high-density clusters.
4. The incorporation of the peculiar characteristics of spatial data to clustering. The proposed clustering algorithm uniquely captures spatial dependence and considers spatial heterogeneity. That is, it treats each point as equally important and combines relative proximity and absolute proximity to measure similarity between points. It efficiently combines local interactions and global interactions to automatically detect globally significant spatial concentrations.

Thus, this approach is able to report geographically interesting clusters that traditional algorithms fail to identify.

5. Extensions of the clustering algorithm to Minkowski metrics (the Euclidean, the Manhattan and the Dominance) in the presence of obstacles for two- or three-dimensional datasets. The extensions not only allow for the efficient computation of clustering in various spatial contexts, but facilitate “what-if” analysis.
6. The design of the clustering criterion and the clustering algorithm using the Template-Method Pattern. This allows extensions of the clustering algorithm to other contexts.
7. The incorporation of the clustering algorithm as a divisive hierarchical (multi-level) clustering algorithm for the development of cluster hierarchies and the introduction of weighted dendrograms to effectively visualize the cluster hierarchies.
8. The design and implementation of a cluster polygonization algorithm. It does not demand costly parameter-tuning but automatically approximates shapes of clusters. Cluster polygonization enhances the readability and usability of clusters.
9. The development of algorithms for the derivation of spatial association rules towards the exploratory description and characterization of spatial clusters.

1.4 Outline of the thesis

The remainder of this thesis is organized as follows.

Chapter 2 presents background material on clustering. This chapter surveys well-known families of clustering methods and discusses their advantages and disadvantages for GDM. The clustering algorithms reviewed are those common amongst the data mining community and the GIS community.

Chapter 3 presents detailed motivation for the development of multi-purpose exploratory spatial clustering. This chapter discusses special characteristics of geoinformation that spatial clustering must consider to identify geographically interesting patterns of concentrations. Then, it explains the problems caused by ignoring the special characteristics, with examples, and illustrates the drawbacks in traditional clustering approaches with those examples. This chapter also presents a framework for our multi-purpose exploratory boundary-based spatial clustering.

Chapter 4 presents the detailed working principle of our multi-purpose exploratory boundary-based spatial clustering. It discusses modeling issues of point data and demonstrates that the Delaunay diagram robustly and unambiguously models spatial adjacency of point data. Then, this chapter presents and explains the detailed algorithmic procedure of our clustering method on how to quantify the Delaunay diagram and how to discriminate inter-cluster edges from intra-cluster edges from the proximity graph. Experimental results are also presented to confirm the effectiveness (quality of clustering) and efficiency (time requirements suitable for GDM) of our clustering.

Chapter 5 extends our clustering to various structural models defining different spatial proximity information. It first examines our clustering when the structural model is obstructed by the presence of obstacles. In this chapter, we introduce four generic approaches to effectively and efficiently process obstacles for clustering. Next, this chapter investigates the extension of our clustering when the structural model is affected by the metric used. In particular, it examines several instances of the Minkowski metric. Finally, this chapter investigates the flexibility of our clustering to various well-known proximity graphs. These extensions make our exploratory clustering suitable for “what-if” analysis and allow the user to quickly explore clustering in different structural models.

Chapter 6 discusses the extension of our approach to three and even higher dimensions based on the discussion made in Chapter 5. The extension is based on the higher dimensional Delaunay diagram and the undirected k -nearest neighbor graph. This

chapter demonstrates that our clustering with the higher dimensional Delaunay diagram consistently produces quality clustering in higher dimensions but its efficiency decreases as dimensions become higher. This chapter also shows that clustering with the undirected k -nearest neighbor graph produces similar quality of clustering but CPU requirements remain subquadratic for higher dimensions. The performance is evaluated analytically and experimentally.

Chapter 7 extends the proposed clustering method to hierarchical clustering that recursively decomposes and reveals multi-level cluster hierarchies within subquadratic time. In order to effectively visualize the cluster hierarchy, a new tree diagram is introduced. The diagram includes weights and thus significantly reduces dendrogram size and enhances readability and usability.

Chapter 8 describes a post-clustering method that automatically mines multivariate associations among layers. In this chapter, we introduce a cluster polygonization algorithm that extracts shapes of clusters and transforms clusters (points) to regions (areas). Then, we present multivariate association mining algorithms based on the cluster polygonization that do not require concept hierarchies nor domain-specific knowledge.

Chapter 9 concludes this thesis with a brief summary, discussions of implementation issues and future work.

Chapter 2

Related Work

Clustering is a series of processes grouping a set of point data, $P = \{p_1, p_2, \dots, p_n\}$ in some study region R , into smaller homogeneous clusters with similar characteristics so that inter-cluster similarity is minimized and intra-cluster similarity is maximized. Numerous clustering approaches have been suggested within the data mining, spatial databases, image processing, statistics and GIS communities. Different clustering methods have been shown to offer different capabilities, different domains of applicability and different computational requirements. Clearly, no particular clustering method has been shown to be superior to all its competitors in all aspects.

This chapter surveys families of clustering techniques and reviews well-known clustering methods proposed to efficiently handle large databases in the data mining community and the GIS community. This chapter outlines their working principles and discusses their pros and cons for the purpose of GDM.

2.1 Local clustering *vs.* global clustering

Clustering determines to which cluster each point in P belongs. That is, it is a determination process of the membership of each point in P to a cluster. If the local neighborhood of a point is the only factor affecting the membership of the point, the determination process is a local clustering approach. Otherwise, it is a global clustering approach. That is, the distribution of P affects the membership of each point in global clustering methods while the distribution of local neighbors is the primary factor in local clustering methods. In local clustering methods, the membership of a point is invariable unless changes occur in the neighborhood of the point. However, in global clustering methods, the membership must be reassessed when changes occur in R . Thus, for incremental clustering, global clustering methods typically require at least linear time unless they use a sophisticated indexing (hashing) method. On the other hand, local clustering methods work well for incremental clustering since they only need to check the limited local neighborhood of where a change occurs. Thus, they are fast for incremental clustering, which is attractive in data-rich environments where total reclustering is extremely expensive. As a result of this, several incremental clustering approaches [125, 142] are proposed based on local clustering methods.

However, the use of local information in local clustering approaches brings about several drawbacks for GDM. The use of only local information contradicts Tobler's proposition now recognized as the 'First Law of Geography' [93] stating "everything is related to everything else, but near things are more related than distant things" [135]. Thus, local clustering methods may miss spatial concentrations that convey potentially important spatial meaning. Special care is required when they are used for spatial data. Second, local clustering methods are not able to automatically report globally significant concentrations in the context of R since they only consider local information. Third, although they are fast for incremental clustering, resulting clusters may be non-informative after some updates. This is because the updates can substantially change the distribution of P , thus resulting clusters do not prop-

erly represent concentrations in the new distribution of P after the updates. When this happens, local clustering approaches need to tune parameters to find informative clusters in the new distribution. However, this tuning is also laborious and expensive in data-rich environments.

Figure 2-1 illustrates the difference between local clustering and global clustering. Figure 2-1(a) depicts a dataset before updates while Figure 2-1(b) displays it after adding 8 new points. The shaded area in Figure 2-1(a) indicates the local neighborhood of p_1 . Note that, there exist many ways of defining local neighborhoods. This will be further discussed in Section 5.3. In this case, we use 4-nearest neighboring. In local clustering methods, changes in this neighborhood of p_1 only affect the membership of p_1 . For instance, addition or deletion of any point in the shaded region in Figure 2-1(a) will affect the membership of p_1 . Since the updates occur outside of the region as shown in Figure 2-1(b), the membership of p_1 is invariable. Thus, if points $(p_1, p_2, p_3, p_4$ and $p_5)$ belong to the same cluster before the updates, they will belong to the same cluster after the updates. However, in the new distribution of P , the five points do not seem to have the same membership. There are four new spatial aggregations represented in different tones of greys in Figure 2-1(b). Local clustering methods require parameter-tuning to detect these newly aggregated clusters. On the other hand, global clustering methods do not necessitate parameter-tuning since they account for the distribution of P after each update.

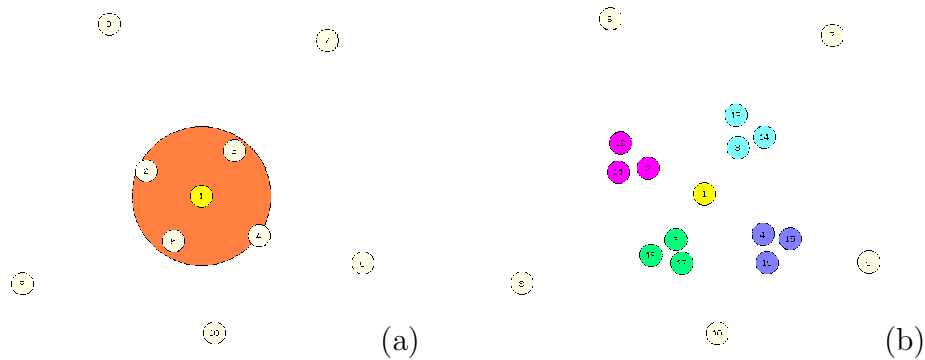


Figure 2-1: Local clustering and global clustering: (a) Before updates ($\|P\| = 10$); (b) After updates ($\|P\| = 18$).

2.2 Dynamic model clustering *vs.* static model clustering

To measure the similarity (equivalently dissimilarity) between two groups is an important process for clustering. Based on the similarity, clustering assigns similar groups to the same cluster and allocates dissimilar groups to distinct clusters. In dynamic model clustering methods, the similarity measure is based on a dynamic model that varies with points. The similarity of two points is measured in relation to the neighborhoods of the two. That is, the absolute similarity between the two is normalized in proportional to the neighborhoods. On the other hand, an absolute model is used in static model clustering. The absolute model is globally invariable.

One simple example of static model clustering is to remove edges in graph-based clustering that are greater than a given global threshold. Thus, edges only shorter than or equal to the global threshold represent intra-cluster interactions. Here, all intra-cluster edges must be shorter than any inter-cluster edge. If the absolute cut-off value varies with points, then the clustering approach becomes a dynamic model clustering method. In this case, intra-cluster edges are not necessarily shorter than inter-cluster edges. Some intra-cluster edges may be longer than inter-cluster edges. One major drawback of static model clustering is that it fails to find relatively low peaks, but seeks only global high peaks. Thus, it is not suitable to applications where low peaks are also of importance. Other problems will be further discussed in Section 3.1.

2.3 Fuzzy clustering *vs.* crisp clustering

Fuzzy clustering [72] (sometimes called non-crisp clustering) [45, 69] allocates each point to multiple clusters with degrees of membership while crisp clustering [23] (sometimes called hard clustering [72]) assigns each point in P to a single cluster.

That is, in fuzzy clustering, clusters are not mutually exclusive, but overlap. Figure 2-2 explains this. Two crisp clusters H1 (p_1, p_2 and p_3) and H2 (p_4, p_5 and p_6) denoted by rectangles are mutually exclusive. On the other hand, two fuzzy clusters F1 and F2 denoted by ellipses overlap.

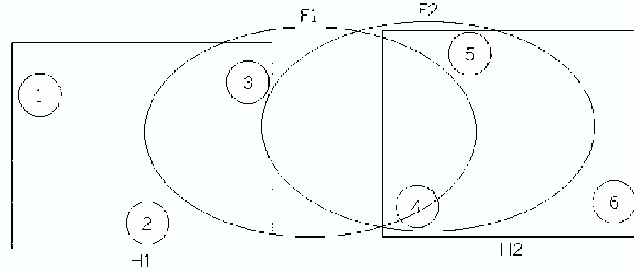


Figure 2-2: Fuzzy clustering and crisp clustering ($\|P\| = 6$).

Fuzzy clustering is especially useful when there is no clear boundary between clusters. Typically, degrees of membership are real values from zero to one. The main strength of fuzzy clustering is in its convertibility to crisp clustering. It is easily converted to crisp clustering by assigning each point to the cluster exhibiting the greatest degree of membership. The most well-known approach is the fuzzy c -means algorithm [17]. Logically, the algorithmic procedure is the same as k -means clustering except it needs to compute the $n \times k$ membership matrix and uses degrees of membership to compute new centroids (means). The membership matrix plays a great role in clustering results. Thus, to define the membership matrix is an important issue in fuzzy clustering [72].

2.4 Hierarchical clustering *vs.* partitioning clustering

A categorization of various clustering methods into hierarchical clustering and partitioning clustering is a well-known distinction [33, 71, 72]. Hierarchical cluster-

ing [36, 58, 59, 76, 141, 142, 151] creates a nested decomposition of P (thus, it is called multi-level clustering) while partitioning clustering [19, 22, 26, 31, 33, 35, 75, 108, 111] produces only one partition of P (thus, it is called single-level clustering).

2.4.1 Hierarchical clustering

In hierarchical clustering, the nested decomposition is represented by a tree diagram (dendrogram) that can either be top-down (divisive) or bottom-up (agglomerative). The top-down approach starts with P in a single cluster representing the root in a dendrogram and it is successively partitioned into subclusters until a termination condition is met. The bottom-up approach is logically the opposite of the top-down approach. In the bottom-up approach, each point in P constitutes a distinct cluster at the beginning. The agglomerative approach successively merges points until all points in P belongs to the same cluster. The merge is based on some similarity measures. Five widely used similarity measures are single-linkage, complete-linkage, average-linkage, mean-linkage and Ward's method. These are defined as follows where C_i , C_j and C_k are clusters of P , C_k is the union of C_i and C_j , p and p' are points in P , n_i is the number of points in C_i , and m_i is the mean of C_i .

- Single-linkage: $S_{sing}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} \{dis(p, p')\}$.
- Complete-linkage: $S_{comp}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} \{dis(p, p')\}$.
- Average-linkage: $S_{aveg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} \{dis(p, p')\}$.
- Mean-linkage: $S_{mean}(C_i, C_j) = dis(m_i, m_j)$.
- Ward's method: $S_{ward}(C_i, C_j) = \sum_{p \in C_k = C_i \cup C_j} dis(p, m_k)^2 - (\sum_{p \in C_k} dis(p, m_k))^2$.

The most popular dissimilarity measure for dis is the Euclidean distance. Namely, the closer the points in space, the more similar the points. The single-linkage approach [131] merges clusters that contain the closet pair of points. Thus, it is robust

to non-globular clusters. However, resulting clusters tend to be long and elongated. That is, clusters connected by narrow linear links are merged into the same cluster. This is the major drawback of this approach. The complete-linkage approach merges clusters that have the closet farthest pair of points. Thus, this approach tends to find relatively compact and globular clusters. Note that, the merge in the single-linkage and the complete-linkage approaches is solely based on only the two extreme pairs. The average-linkage approach attempts to incorporate more information to the merge. This approach computes the average of dissimilarities between clusters. Here, every point within a cluster is equally important. Thus, this approach is extremely sensitive to noise points and outliers. The mean-linkage approach is a hybrid of the extreme value approaches (the single-linkage and the complete-linkage) and the average-linkage approach. The mean-linkage approach considers all points in a cluster when it computes the mean, but uses only the mean for the merge. Since means are the representatives of clusters, resulting clusters tend to be globular, large clusters tend to be broken into meaningless groups and parts belonging to close neighboring clusters tend to be merged. The Ward's method optimizes the within-groups sum of squares. This method merges the two clusters resulting in the smallest increase in the within-groups sum of squares. Thus, this approach tends to partition P into globular clusters with similar sizes.

One advantage of hierarchical clustering is that it does not necessitate the number of clusters as an input argument. It allows the user to choose the best decomposition of P from the dendrogram. However, where to stop partitioning or merging must be specified instead. This termination condition is not easy to determine. Another drawback of this approach is that it is rigid in the merge or split process [61]. This rigidity may cause erroneous clustering when the merge or split process is not optimal. The computational inefficiency of this clustering is another drawback. Algorithms of this type may easily require of $O(n^2)$ [107] time, since they typically need $O(n)$ merging or splitting operations where the decision on what to split or merge is proportional to the current number of parts. In addition, dendograms become very large with large datasets and their expert interpretation becomes extremely complex and challenging.

Now, we discuss the details of several clustering methods.

BIRCH

BIRCH [151] is a hierarchical clustering method that uses compact summary statistics instead of large datasets themselves to improve efficiency. Two underlying components are the Clustering Feature (CF) and the Cluster Feature Tree (CF-Tree). A $CF = (n, \overrightarrow{LS}, SS)$ is a triplet summarizing the information about a cluster. In this triplet, n is the number of points in the cluster, $\overrightarrow{LS}$ is the linear sum of the n points and SS is the square sum of the n points. BIRCH first scans a dataset and incrementally builds an initial CF-Tree. The CF-Tree is a height-balanced tree requiring two parameters. These are a branching factor B and a threshold TH . The branching factor defines the maximum number of entries in non-leaf nodes. Figure 2-3 depicts a CF-tree structure where b denotes the bottom level in the tree and L denotes the maximum number of entries in leaf nodes. In this tree structure, each non-leaf node has exactly B entries and each leaf node has at most L entries. Here, all entries in leaf nodes must satisfy the threshold TH . That is, the radius of each entry in a leaf node must be less than TH . These two parameters influence the size of CF-tree and thus the effectiveness of clustering.

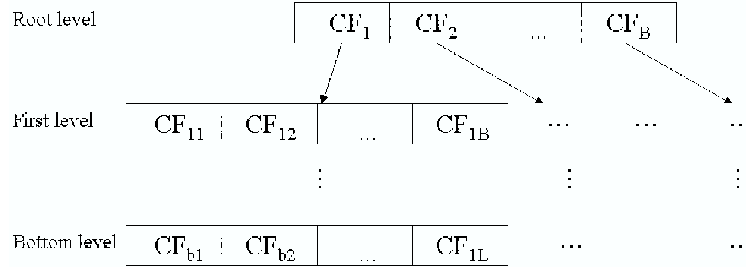


Figure 2-3: BIRCH's tree structure.

When an entry is inserted into the CF-tree, BIRCH recursively descends from the root level to the bottom level by selecting the closest child node at each level. When it reaches to a leaf node at the bottom level, it searches for the closest leaf entry. If the closest leaf entry can absorb the new entry without violating the threshold condition then the CF of this entry is updated and those of its parents are subsequently updated.

If the closet leaf entry can absorb the new entry, then the new entry will be added to the leaf. If the size of the leaf node after insertion exceeds L , then the node is split. Once a complete CF-tree is constructed, then BIRCH groups leaf nodes to find clusters.

In BIRCH, the use of compact summary statistics may degrade the quality of clustering. Also, not every point is treated equally. In addition, BIRCH favors globular clusters since it uses the radius to measure similarity. It is also dependent on the probabilistic model used to collect statistics.

CURE

CURE [58] is a bottom-up hierarchical clustering method that improves the merge process using multiple representative points. Thus, it is seen as a hybrid between the single-linkage and the mean-linkage. In each level of clustering, CURE first chooses a constant number of well scattered points for each cluster as representative points. The representative points approximate the physical shape and geometry of the cluster. Next, CURE shrinks the representative points toward the mean of the cluster by a factor α that varies from 0 to 1. The shrinking causes the representative points lying further away from the mean to move more toward the mean while the representative points lying closer to the mean to shift less. This makes CURE robust to outliers. After shrinking, CURE merges two clusters exhibiting the closet pair of representative points belonging to different clusters. The shrinking factor α controls the compactness of representative points and shapes of clusters. CURE becomes similar to mean-linkage clustering when $\alpha = 1$ while it becomes similar to single-linkage clustering when $\alpha = 0$. Thus, the use of larger values of α favors compact clusters while the use of smaller values favors clusters of arbitrary shapes.

Despite the flexibility and effectiveness of CURE, it suffers from high computational cost. It requires $O(n^2 \log n)$ time for higher dimensions and $O(n^2)$ time for lower dimensions where n is the number of points. To handle massive datasets, CURE exploits a combination of random sampling and partitioning. It first draws a random sample and then segments the sample into the user-specified number of partitions.

CURE proceeds to segment each partition into partial clusters and eliminate outliers. Finally, CURE merges the partial clusters in each partition to find the user-supplied number of clusters.

CHAMELEON

CHAMELEON [76] is a bottom-up hierarchical clustering method that explores dynamic modeling based on a proximity graph. The key feature of this approach is that it employs a combination of partitioning and merging. It partitions the proximity graph and merges the pair of most similar clusters (connected components) based on the relative inter-connectivity and the relative closeness of the two. Thus, it employs a dynamic model to measure the similarity between clusters.

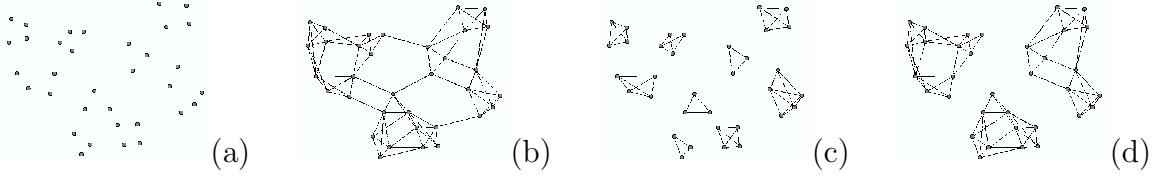


Figure 2-4: An overview of CHAMELEON: (a) A dataset ($\|P\| = 34$); (b) Undirected 4-nearest neighbor graph; (c) Partition of the graph; (d) Merged partitions.

Figure 2-4 illustrates the working principle of CHAMELEON. It first constructs the k -nearest neighbor graph as an underlying proximity graph. Figure 2-4(b) illustrates the 4-nearest neighbor graphs where edges are undirected. Then, it partitions the graph to find initial subclusters such that the *edge-cut*, the sum of weights (inversely proportional to lengths) of edges that straddle subclusters, is minimized. Figure 2-4(c) depicts initial partitions. Finally, CHAMELEON merges the most similar subclusters based on the two similarity measures: relative inter-connectivity and relative closeness. The relative inter-connectivity between two clusters C_i and C_j ($RI(C_i, C_j)$) is the absolute inter-connectivity between the two clusters normalized with respect to the internal inter-connectivity of the two. That is,

$$RI(C_i, C_j) = \frac{|EC_{\{C_i, C_j\}}|}{\frac{|EC_{C_i}| + |EC_{C_j}|}{2}}, \quad (2.1)$$

where $EC_{\{C_i, C_j\}}$ denotes the *edge-cut* between clusters C_i and C_j and the internal inter-connectivity EC_{C_i} is the weights sum of edges that partition C_i into two roughly equal parts. The relative closeness ($RC(C_i, C_j)$) is the absolute closeness between C_i and C_j normalized with respect to the internal closeness of the two clusters. That is,

$$RC(C_i, C_j) = \frac{\overline{S}_{EC_{\{C_i, C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\overline{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i|+|C_j|}\overline{S}_{EC_{C_j}}}, \quad (2.2)$$

where $\overline{S}_{EC_{\{C_i, C_j\}}}$ is the average weight of $EC_{\{C_i, C_j\}}$ and $\overline{S}_{EC_{C_i}}$ is the average weight of the EC_{C_i} . CHAMELEON uses a function that combines $RI(C_i, C_j)$ and $RC(C_i, C_j)$ to find the best pair to merge. The function is $RI(C_i, C_j) * RC(C_i, C_j)^\beta$, where β is a user-specified parameter. CHAMELEON gives a higher importance to the relative closeness if $\beta > 1$ while it gives a higher importance to the relative inter-connectivity if $\beta < 1$.

The use of a dynamic model makes CHAMELEON detect clusters of different densities. However, it requires laborious parameter-tuning for the best k in k -nearest neighboring. Once points are assigned to the same cluster after the initial partitioning, then they always belong to the same cluster regardless of the merge process. Thus, careful tuning is required for the best k and the best partitioning of the underlying graph.

2.4.2 Partitioning clustering

Partitioning clustering aims at segmenting the n points of P into a set of k clusters. Typically, k is an input argument, thus it is also called k -clustering [46]. In general, k -clustering attempts to detect k optimal clusters by an iterative relocation method based on an optimization criterion function. That is, this algorithm starts with a partition into k clusters and then iteratively optimizes the criterion function by reassigning some points to their nearest representatives of clusters. The optimization function that measures cohesiveness signals the best partition. However, the need to use approximation algorithms (hill-climbers) arises because optimizing the objective

function is NP-Hard [54] or even intractable. Thus, k -clustering finds a local optima as opposed to a global optima. This suboptimality is the major problem of k -clustering.

A well-known optimization function is the squared-error criterion (SE) defined as follows, where $rep[P_i]$ denotes the representative of cluster P_i and $|p - rep[P_i]|$ denotes the distance between p and $rep[P_i]$.

$$SE = \sum_{i=1}^k \sum_{p \in P_i} |p - rep[P_i]|^2. \quad (2.3)$$

Thus, the aim of k -clustering is to minimize SE . Depending on the type of representative, the effectiveness and efficiency of k -clustering differ. The k -means clustering approach [95] uses the mean (centroid) as the representative of a cluster. It initially chooses arbitrary k centroids and assigns each point in P to its nearest centroid. After that, it computes the mean of each cluster and reassigns each point in P to its nearest newly computed mean. The k -means clustering approach keeps computing the means and reassigning until the set of representatives converges. The k -means clustering approach is relatively efficient since it converges fast. It requires $O(nkr)$ time, where n is the number of points, k is the number of clusters and r is the number of iterations. Typically, k and r are much smaller than n in large datasets, thus this approach is of particular interest in data-rich environments. However, k -means clustering has several drawbacks. First, it is vulnerable to noise points and outliers [62]. Second, the centroids may be geographically non-informative. For instance, the center of houses may fall within a lake or the center of suburbs may be within a river. Third, it is not applicable to categorical datasets where the mean of a cluster is not defined.

The k -modes clustering approach [70] extends the k -means clustering approach to categorical datasets. It replaces the means of clusters with the modes and uses different dissimilarity measures to update the modes. It computes the total number of mismatches in the corresponding attribute categories. Here, the smaller the number of mismatches is, the more similar the two objects are. Expectation Maximization (EM) clustering [26] is another generalization of k -means clustering, but in contrast

to k -means clustering, it allows non-crisp membership with a probability function. That is, there is no hard boundary between clusters. Thus, each point in P belongs to k clusters with different probabilities of membership. One widely used representation of the probability function is the mixture model using Gaussian probability distributions. Each Gaussian distribution generates a cluster and P is the mixture of such clusters. The k -harmonic means clustering method [150], that is also a variant of k -means clustering, attempts to overcome the sensitivity of k -means clustering to the initialization of centers. It computes the harmonic average of each cluster (that is the reciprocal of the arithmetic average of the reciprocals of the numbers in a cluster) and uses it as an optimization function. Logically, the same algorithmic procedure of k -means clustering can be used to these generalizations. And, they all converge to a local optima.

The k -medoids clustering method [35, 77, 108] differs from the k -means clustering method in the sense that the representative of a cluster is the member of the cluster. More importantly, the Equation 2.3 can be replaced by the absolute error [35]. This medoid is typically close to the centroid of the cluster. Note that, medians are a robust estimator of location in the statistical sense, but means are not. Thus, representatives in k -medoids clustering are more informative than those in k -means clustering. In addition, k -medoids clustering is less sensitive to noise and outliers than k -means clustering. Algorithms for k -medoids clustering iteratively exchange one of the medoids with a non-medoid to improve the optimization function. This exchange needs an exhaustive search to find the best candidate, which is prohibitive in data-rich environments. Thus, k -medoids clustering is typically inefficient and some hill-climbing iterations [77, 108] require quadratic time.

Regardless of the type of representative, k -clustering shares some common drawbacks. Although several attempts [35, 77, 128] have been made to find the number of clusters, typically k -clustering requires k as an input argument that usually remains constant during the course of iterations. These type of approaches implicitly assume the data comes from a probabilistic mixture model. This not only puts artificial constraint

(how many clusters are there) on data, but introduces user biases. Second, since the points are assigned to the nearest representative, clusters are convex-shaped. Han and Kamber [61] emphasize that the discovery of clusters of arbitrary shapes is of importance for further analysis. Third, large discrepancies in size or density cause big clusters to pass undetected since their points are assigned to many meaningless fragments.

CLARANS

CLARANS [108] is a k -medoids clustering method that is based on a random search heuristic. It first chooses k arbitrary medoids. And then, it randomly chooses one medoid from the k medoids and exchanges the medoid with another randomly chosen non-medoid to examine if the exchange improves SE . If SE is improved, then CLARANS replaces the medoid with the non-medoid and repeats the random search. If a number of exchanges does not improve SE , then the current k medoids are assumed to be a local minimum.

CLARANS requires two input parameters to prune the search space. They are *num-local* and *maxneighbor*. The former represents the number of local minima obtained while the latter represents the number of exchanges between one of the medoids and one of the non-medoids. The use of higher values in the parameters results in better quality of clustering but requires more expensive computational cost. Thus, the user needs to tune values of these parameters to find a satisfactory clustering result within a given time allowance. However, this tuning is very expensive for CLARANS since it requires quadratic time for each clustering. Other problems with CLARANS have been highlighted in the literature [44].

A generalization of CLARANS to discover clusters in the presence of obstacles was recently proposed [139, 140]. COE (Clustering with Obstacle Entities) attempts to optimize SE in the presence of obstacles where the distance is measured by the shortest Euclidean distance avoiding obstacles. It computes *micro-clusters* that compactly summarize the original dataset and performs the clustering on these summarized groups to enhance time efficiency. However, this lowers the quality of clustering and

still suffers from expensive time requirements. We further contrast this method with our approach in Section 5.1.6.

2.5 Density-based clustering

Density-based clustering [1, 6, 33, 111, 127, 129, 141, 142] searches for regions of high density and differentiates them from regions of low density. In general, this approach first defines a region with an arbitrary shape (typically, circle or grid) and then computes the density of the region. If the density of the region is greater than or equal to a user-specified density threshold, then the region belongs to a cluster. Otherwise, it becomes noise. Thus, density-based clustering is able to detect clusters of arbitrary shapes. However, it requires to specify the shape of the region, the size of the region and the density threshold in advance.

The circle (vector-like modeling) and the grid (raster-like modeling) are the most widely used shapes. Depending on the shape of the region, the effectiveness and efficiency of clustering will differ. In general, circle-based clustering [6, 33, 111] produces better clustering, but it is more expensive in terms of time efficiency than grid-based clustering [1, 127, 129, 141, 142]. This is because circles do not tessellate R with mutually exclusive and collectively exhaustive regions, but grids do. Thus, circle-based clustering needs to explore more regions than grid-based clustering to check if they satisfy the user-specified density threshold.

On the other hand, the efficiency of grid-based clustering is dependent on the granularity of the grid. This is mainly because grid-based clustering uses statistical summaries of grids for clustering operations instead of individuals. Since the number of grids is typically much smaller than the number of points in data-rich environments, this approach is fast when the grid is coarse. However, the use of statistical summaries and coarse tessellation lowers both quality and accuracy of clustering results. Also, points have unequal importance unless the cell size is fine enough to capture individ-

uals, and in this case, the grid is of no use. Thus, it is difficult to determine the best resolution in grid-based clustering as it is with raster-like modeling.

Besides its ability to detect clusters of arbitrary shapes, density-based clustering is robust to linear links (chains) that connect clusters. This is because regions around the chains are not dense (otherwise, they are not linear chains) even if points on the chains are closely located. However, it suffers from several drawbacks. Density-based clustering uses ambiguous models of spatial proximity, thus the effectiveness of this approach is heavily dependent on the resolution. Another major problem of this approach is that it focuses on high peaks. That is, it is able to identify high-density clusters, but unable to detect low peaks forming sparse clusters that are also of importance in spatial analysis [111]. This problem will be further discussed in Section 3.2.

DBSCAN

DBSCAN [33] is a density-based clustering method that attempts to locate high-density areas. The underlying idea of this approach is that the neighborhood of a point in a cluster contains at least a minimum number of points *MinPts*. The neighborhood of a point is defined by a given radius (*Eps*) and is called the *Eps*-neighborhood of the point. If the *Eps*-neighborhood of a point p_1 contains more than the *MinPts*, then the point p_1 is called a *core point*. A point p_k becomes *density-reachable* from the *core point* p_1 if there is a list of points $p_1, p_2, \dots, p_k$ such that p_{i+1} is within the *Eps*-neighborhood of p_i for $1 \leq i \leq k$. Since all points within a cluster are *density-reachable* from any *core point* within the cluster, DBSCAN first chooses a *core point* and then expands it by searching for all *density-reachable* points from the *core point* to detect a cluster. The set of points that do not belong to any cluster is noise. DBSCAN keeps searching and expanding until no point is left.

Figure 2-5 explains this. If $MinPts = 4$ is used, then p_1 becomes a core point since the *Eps*-neighborhood of p_1 contains 5 points (p_2, p_3, p_4, p_5 and p_6). The point p_{15} is *density-reachable* from p_1 since there is a list of points $p_1, p_6, p_8, p_{10}, p_{13}, p_{15}$ such that the successor is within the *Eps*-neighborhood of the predecessor. However, p_{16} is not

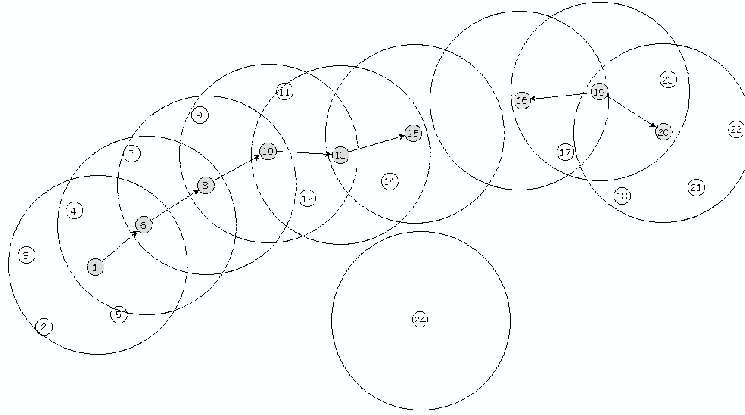


Figure 2-5: An overview of DBSCAN ($\|P\| = 24$).

density-reachable from p_1 since none of points within the Eps -neighborhood of p_{16} is *density-reachable* from p_1 . Thus, p_{16} belongs to a different cluster. If p_{19} is selected as the next core point, then points from p_{16} to p_{23} are *density-reachable* from p_{19} . Thus, they form another cluster. The point p_{24} that does not belong to any other cluster becomes noise.

DBSCAN uses R*-trees [12] to efficiently retrieve *density-reachable* points and to prune the search space. It requires subquadratic time if the parameter Eps is small enough so that balls of radius Eps retrieve a constant number of points on average. DBSCAN is able to detect clusters of arbitrary shapes, which is an improvement over k -clustering algorithms. However, it is problematic when clusters exhibit different densities since it uses a global density threshold. Thus, it inherits the problems of static model clustering. In addition, it is a local clustering method. Thus, it ignores the First Law of Geography. DBSCAN requires two parameters Eps and $MinPts$ to be specified prior to clustering. Clustering results are heavily dependent on the values of parameters that are difficult to determine.

OPTICS [6] is an extension of DBSCAN. It attempts to overcome the difficulties of parameter-tuning and single-level partitioning of DBSCAN by producing an augmented ordering of P . OPTICS reveals the hierarchical density structure P that is

mostly independent from the choice of the two parameters Eps and $MinPts$. Thus, it overcomes the sensitivity of DBSCAN to the parameters and assists the user to explore the complex hierarchical structure of P .

GAM

GAM [111] is one of the most well-known density-based clustering methods in spatial analysis. It is very similar to DBSCAN in the sense that both measure density based on the circle and report high-density regions as clusters. The main difference between the two is that DBSCAN uses only one user-specified circle, but GAM uses a series of circles with different radii. The use of several circles for density measure makes GAM less biased but more computationally expensive. The basic algorithm of GAM is as follows.

1. Define a starting (minimum) circle radius, a maximum circle radius, a radial size increment and a degree of overlap.
2. Generate an initial grid lattice covering R so that circles of current radius located on grid intersections overlap by the desired amount.
3. Generate circles of current radius for all grid intersections.
4. Retrieve the number of points for each circle and apply some ‘significance’ test such as the Poisson test or the Monte Carlo test.
5. Keep the result if significant.
6. Repeat Step 4 and Step 5 until all circles have been processed.
7. Increase the current circle radius by the given radial size increment. If the increased circle radius is less than the given maximum circle radius, then return to Step 3.
8. Identify clusters with the significant circles.

One interesting characteristic of GAM is that it allows circles to be overlapped to the given degree. The main reason for this is to minimize locations being missed out from the significance test. However, the possibility to overlap increases the number of circles to test and some points belonging to multiple circles are tested several times. This requires computationally intensive search that is unaffordable in data-rich environments. Figure 2-6 explains the working principle of GAM. Figure 2-6(a)

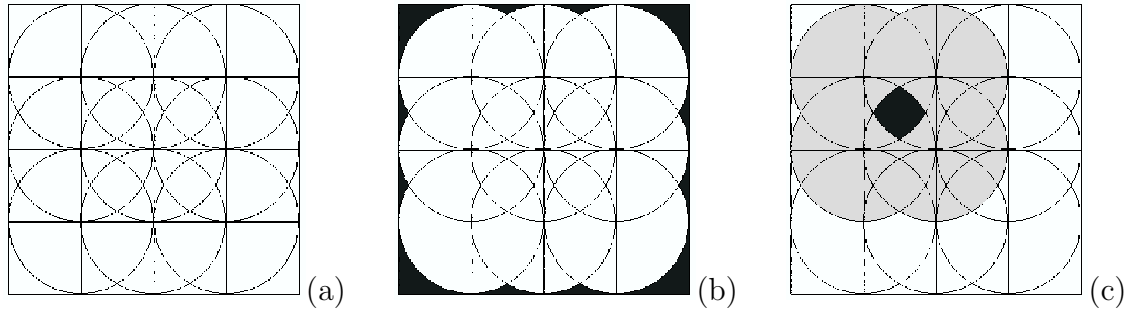


Figure 2-6: An overview of GAM: (a) A 4×4 grid and 9 circles; (b) Locations being missed out; (c) An example of locations multiply tested.

illustrates Step 2 and Step 3. The enclosing rectangle is decomposed into a 4×4 grid and 9 circles are located on grid intersections overlapping to a certain degree. GAM tests the significance of the density for each circle. Although circles overlap, some points in the shaded area in Figure 2-6(b) are left out from the significant test. On the other hand, some points in the dark region in Figure 2-6(c) are tested 4 times. Thus, locations are not regarded equally.

STING

STING [141] is a grid-based clustering method that uses a hierarchical structure to efficiently detect regions of high-density. The hierarchical structure consists of several levels of different resolution. Each cell in a certain level is decomposed into 4 regular grids in the next level similar to the quadtree data structure.

Figure 2-7 illustrates the hierarchical structure consisting of 4 levels. The root level containing only one cell shown in Figure 2-7(a) corresponds to R . The cell is decomposed into 4 subcells in the next level (the first level) shown in Figure 2-7(b). The

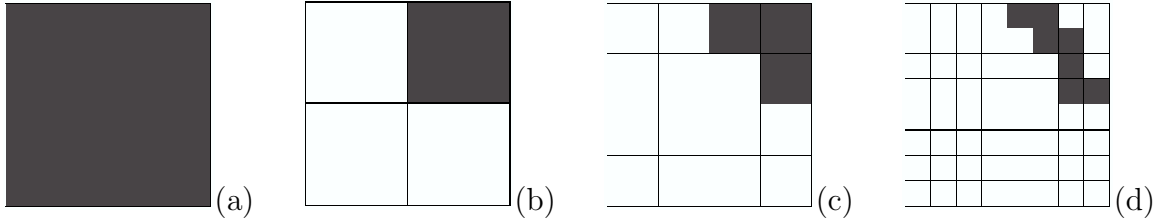


Figure 2-7: An overview of STING: (a) The first level (root level or top level); (b) The second level; (c) The third level; (d) The fourth level (bottom level).

subcells in the first level are successively decomposed in the next level as shown in Figure 2-7(c). Each cell in the hierarchical structure stores summary statistics including the number of points, the mean of all values, the standard deviation of all values, the minimum value, the maximum value and the type of distribution of points in this cell. These summary statistics represent each cell and they are used to measure if the density of the cell is significant or not. STING builds the hierarchical structure from the bottom level to the top level (however, it will search for high-density regions from the top to the bottom to prune search space). It first goes through P once in order to calculate the summary statistics associated with each cell at the bottom level. Once computed, the statistics of parent cells in the next higher level are computed from those of 4 children cells in the current level. Thus, building the structure requires O/ng time where n is the number of points and g is the total number of cells in all levels in the hierarchical structure (note that, g grows at least quadratically with the depth of the structure in 2D and it grows cubically in 3D).

Once the hierarchical structure is computed, STING uses a top-down approach to efficiently find multi-level clusters. If the summary statistics of cells on a higher level satisfy user-specified conditions, then these cells are marked as *relevant* otherwise as *not relevant*. Instead of looking at all the children of cells in the next lower level, STING only examines the children of the *relevant* cells in the previous level to prune the search space. STING keeps doing this until the bottom level is reached. Figure 2-7 illustrates an example. Here, we assume that the shaded cells are marked as *relevant* at each level and we start examining from the top level. STING looks at the 4 children

of the only cell in the top level in the second level since the cell in the top level is *relevant*. In the third level, STING examines only the 4 children of the top-left cell in the second level. Similarly in the bottom level, only cells whose parent are *relevant* are examined. Eventually, shaded regions at each level represent clusters.

Although STING is appealing in terms of efficiency, it suffers from the drawbacks of grid-based clustering as mentioned earlier in this section. In addition, it inherits the problems of local and static model clustering. Thus, STING fails to detect clusters of different densities. The crudeness of the shapes of resulting clusters makes STING less attractive for further association analysis where effectiveness is of importance. As depicted in Figure 2-7, clusters only have axis-parallel boundaries. One other problem of STING is that it does not use information in the neighboring cells.

WaveCluster

WaveCluster [129] is a grid-based hierarchical clustering method based on wavelet transforms. The main idea of this approach is that the multidimensional spatial points can be represented in a d -dimensional feature space where a feature vector represents the d numerical attributes of a point. In the feature space, boundaries of clusters correspond to the high frequency parts while clusters themselves correspond to the low frequency parts. WaveCluster applies a wavelet transform (convolving the low pass filter) to the d -dimensional feature space at least d times to identify the low frequency parts.

The algorithmic procedure of WaveCluster is given below.

1. Represent a dataset in a feature space.
2. Tessellate the feature space with regular units and assign each point in the dataset to corresponding units.
3. Apply a wavelet transform to each unit on the feature space.
4. Find connected components in the transformed feature space.

5. Label units that are included in any connected component in the transformed feature space.
6. Map points in the original feature space to the labeled units in the transformed feature space.

WaveCluster is similar to STING in some aspects. First, they are both grid-based hierarchical clustering. The granularity is an important factor that affects the effectiveness of clustering and impacts computational efficiency. A series of different unit sizes results in multi-level clustering. They are robust to noise and outliers and are able to detect clusters of arbitrary shapes. Both require linear time when the number of units in the finest level is much smaller than the number of points. They ignore the First Law of Geography and use summary statistics. However, WaveCluster differs from STING in several points. First, boundaries of clusters detected by WaveCluster are not necessarily axis-parallel. This is because grids are only used for tessellating the original data space and clustering is performed in the transformed feature space. Second, WaveCluster incorporates neighborhood information of a unit when it determines if the unit is dense enough or not. In particular, the low-pass filters used in Step 3 are designed to incorporate neighborhood information.

CLIQUE

CLIQUE [1] is a grid-based hierarchical clustering method that is particularly designed for detecting subspace clusters in high-dimensional datasets. This approach is of particular interest in the fact that densities around clusters are very low in high-dimensional space. Thus, full-dimensional dense clusters hardly exist. Traditional density-based clustering methods that focus on full-dimensional dense clusters are not well-suited to such situations.

CLIQUE searches for subspace clusters with a bottom-up approach that exploits a monotonicity property with respect to dimensionality to prune search space. The monotonicity property states that if a collection of points is a cluster in a k -dimensional space, then the collection of points is also part of a cluster in any $(k-1)$ -dimensional

projections of this space. CLIQUE first determines 1-dimensional dense units that satisfy a user-supplied density threshold. And then, it proceeds level by level. Once $(k-1)$ -dimensional dense units are determined, then CLIQUE generates candidates for k -dimensional dense units. CLIQUE keeps detecting subspace clusters until it reaches the highest dimension.

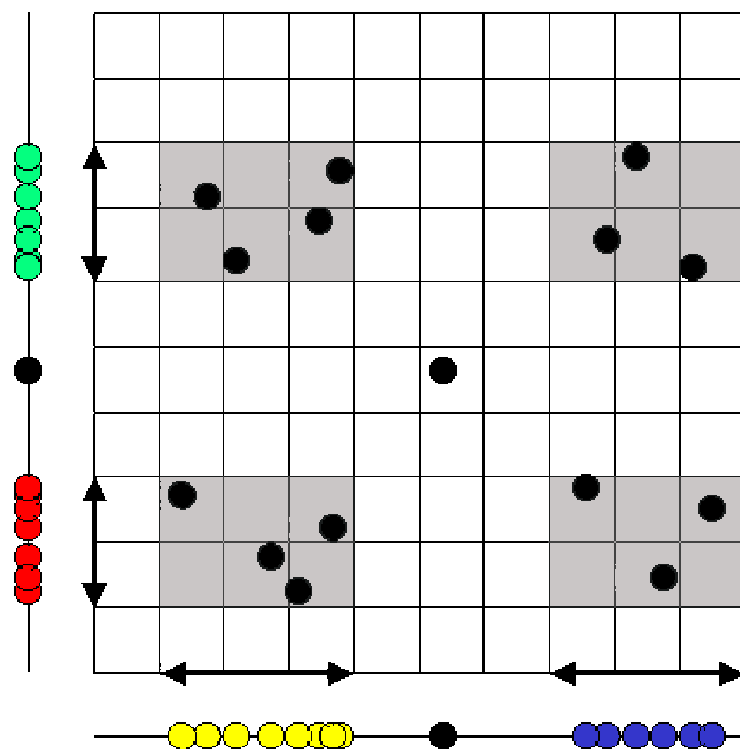


Figure 2-8: An overview of CLIQUE with a 10×10 grid ($\|P\| = 15$).

Figure 2-8 illustrates the working principle of CLIQUE with a 2-dimensional dataset. CLIQUE first tessellates the data space with a regular grid and finds 1-dimensional subspace clusters. In this example, there are 4 1-dimensional subspace clusters. Two of them are found when the dataset is projected to the x -axis while the other two are found when the dataset is project to the y -axis. According to the monotonicity property, CLIQUE examines only 2-dimensional units whose 1-dimensional units are dense to find 2-dimensional clusters. The candidates for the 2-dimensional search are the shaded areas in Figure 2-8. The monotonicity property makes CLIQUE efficient

in high-dimensional spaces. In this example, only 24 2-dimensional cells need to be examined instead of 100. However, this simplicity may cause the degradation of effectiveness. Also, it suffers from the problems of density-based clustering in that clusters with less density than the threshold are not detected. It is also extremely sensitive to the granularity parameter in the grid.

2.6 Graph-based clustering

Graph-based clustering [31, 36, 75, 76, 149] has gained popularity as an alternative to density-based clustering methods since it works from an underlying proximity graph that models spatial interactions between points. The observations (points) are assigned to vertices and a discrete relation is encoded, making two related vertices connected with an edge. Thus, edges incident to a point connect the point to its nearby neighbors. In this abstract structure, a *cut-point* is a vertex whose removal disconnects the graph and a *bridge* is an edge whose removal disconnects the graph [63]. Connected components are the most elementary clusters. Edges have a weight corresponding to the geographical distance with “distance decays effect” between the associated end-vertices.

Graph-based clustering first constructs a proximity graph. Once the proximity graph is constructed, all clustering operations are performed on the graph. Naturally, the aim is to extract the edges whose end-points belong to the same cluster. Typically, graph-based clustering removes uninteresting edges based on a certain criterion function and then computes connected components with remaining edges to reveal clusters. One weakness of graph-based clustering is its vulnerability to multiple chains. Removal of *cut-points* or *bridges* will detect clusters linked by a single chain. However, clusters linked by multiple bridges are undetected by the removal. This will be further discussed in Section 3.2.2. The second problem is that most graph-based clustering methods are edge-centric. Since clustering is to group points, edge-centric clustering may introduce unexpected results. Third, graph-based clustering requires

extra time to construct an underlying proximity graph. Similar to density-based clustering, graph-based clustering is likely to be peak-inference clustering since it focuses only on shorter edges. Thus, sparse clusters linked by relatively homogeneous edges may remain undetected.

However, graph-based clustering has some promising aspects in terms of effectiveness and efficiency. In terms of effectiveness, it is able to detect non-convex clusters and is robust to noise points. In terms of efficiency, it is typically linear in the number of edges in an underlying proximity graph. Thus, if the number of edges is not excessive but sufficient, then graph-based clustering is computationally efficient.

SMTIN

SMTIN [75] is a graph-theoretic approach that performs local clustering and static model clustering. It is an intuitive method that uses the Delaunay triangulation as an underlying proximity graph to capture spatial proximity and to measure the similarity between points. Practically, the working principle of this approach is the same as the one dimensional clustering of Delaunay edges [31]. That is, once the Delaunay triangulation is constructed, then the clustering becomes the matter of classification of Delaunay edges into two groups: inter-cluster edges and intra-cluster edges. Delaunay edges whose lengths are greater than a given global threshold are inter-cluster edges. These longer edges represent uninteresting interactions and indicate that the two end-points of each edge belong to distinct clusters. Thus, they are removed from the underlying graph. Eventually, connected components of the remaining graph represent clusters. This approach is easy to implement and is able to detect clusters of arbitrary shapes. However, this approach suffers from the use of a static model as a threshold. Thus, it is difficult to detect clusters of different densities. In addition, this approach suffers from the edge-centric nature of clustering. Recently, a node-centric extension [32] of this approach is proposed to improve the quality of clustering.

Zahn's clustering

Zahn [149] proposes a graph-theoretic clustering method that uses a dynamic model. This approach first constructs the Minimum Spanning Tree of P , denoted by $MST(P)$,

to capture similarity in perceptual groupings of P and removes inconsistent edges from $MST(P)$ to detect clusters. If the length of an edge $e_{i,j} = (p_i, p_j) \in MST(P)$ is significantly larger than the average of lengths of nearby edges on both end-points of $e_{i,j}$, then the edge becomes an inconsistent edge. That is, the criterion for the determination if $e_{i,j}$ is consistent or inconsistent is purely depending on relative proximity and varies over R . Thus, this approach belongs to dynamic model clustering and it may be able to detect clusters of different densities. However, this notion of inconsistent edges works well with datasets where separations between clusters are apparent and densities of clusters varies smoothly.

For complex situations, Zahn introduces several case-specific methods. Here, the user is required to use a specific heuristic to handle each particular situation. For instance, in the case of touching clusters, the user is required to compute the path of the $MST(P)$ that includes the largest number of edges in the $MST(P)$ (called the *diameter*). This is used to locate a connecting part between the touching clusters. For clusters of different densities, the heuristic is different. The user is required to compute a histogram of edge lengths to find a separation between a dense cluster and a sparse cluster and subsequently required to recompute the MST of the sparse cluster. Thus, Zahn's approach demands domain knowledge and apriori knowledge about P to select a proper heuristic. However, it is a difficult task to choose an adequate heuristic when the distribution of P is unknown. Thus, although this approach provides some promising aspects, it is laborious and not well-suited for data-rich environments.

2.7 Model-based clustering

Graph-based clustering is the closest to descriptive and non-parametric statistics. These approaches are closer to exploratory analysis since no probabilistic model is imposed in the data. Grid-based clustering is originally non-parametric in that it is an exploratory analysis based on the histogram. However, most algorithms use some parametric statistics over hierarchical data structure for computational efficiency.

When we use parametric statistics, an explicit parametric model is fit to the data to find clusters.

Model-based clustering algorithms [11, 22, 26, 49, 50, 66, 118] are widely adopted within the statistics community. Here, the strong assumption is that a mixture of underlying probability distributions generates the dataset and each component represents a different cluster. This assumption makes model-based clustering confirmatory rather than exploratory. Typically, a hierarchical technique or a partitioning technique is used to obtain clusters from these models. In the former [11, 49, 118], a criterion function (like the Maximum Likelihood or Minimum Message Length) is used to compare models to merge at each stage in the construction of hierarchies. In the latter, points are grouped to maximize the likelihood of the mixture model. Here, deriving optimal partitions from these models is usually an elusive task [118]. The EM clustering method [22, 26] is a good example for the latter. Since the algorithmic procedure of model-based clustering typically follows either hierarchical clustering or partitioning clustering, model-based clustering algorithms inherit drawbacks of hierarchical clustering or partitioning clustering discussed earlier in this chapter.

2.8 Remarks

This chapter reviews families of clustering approaches and surveys well-known clustering methods that are able to deal with large spatial databases. In addition, this chapter discusses their strengths and weaknesses for the purpose of GDM. A summary of properties for well-known clustering methods is given in Table 2.1.

Table 2.1: Comparison of well-known clustering methods.

	global <i>vs.</i> local	dynamic model <i>vs.</i> static model	hierarchical <i>vs.</i> partitioning	explicit density <i>vs.</i> implicit density
BIRCH	local	static	hierarchical	implicit
CURE	global	static	hierarchical	implicit
CHAMELEON	local	dynamic	hierarchical	implicit
CLARANS	global	static	partitioning	implicit
DBSCAN	local	static	partitioning	explicit
GAM	local	static	partitioning	explicit
STING	local	static	hierarchical	explicit
WaveCluster	local	static	hierarchical	explicit
CLIQUE	local	static	partitioning	explicit
SMTIN	local	static	partitioning	implicit
Zahn	local	dynamic	partitioning	implicit
EM	global	dynamic	partitioning	explicit

Chapter 3

Multi-Purpose Exploratory Spatial Clustering

In this chapter, we present detailed motivation for multi-purpose exploratory spatial clustering. First, in Section 3.1, we discuss special characteristics of geoinformation. These are characteristics brought over by the domain and the nature of the data and spatial clustering must account for them. Then, Section 3.2 discusses the drawbacks of traditional peak-inference clustering approaches. Ignoring the special characteristics of geoinformation causes problems. Such drawbacks are illustrated with several examples where peak-inference clustering approaches fail to detect some interesting patterns and spatial concentrations. In Section 3.3, we propose a framework for our multi-purpose exploratory boundary-based spatial clustering.

3.1 Requirements for spatial clustering

Spatial clustering partitions geo-referenced point data into smaller homogeneous subgroups with similar characteristics due to contiguity (spatial proximity). These geo-referenced datasets encapsulate the inherent and peculiar characteristics of spatial

data what makes them special [7, 8]. These special characteristics of geoinformation require specialized clustering methods that incorporate the geo-referenced nature in order to identify high-quality patterns of spatial concentrations (and not only the spatial nature). Special geo-characteristics are:

- *Consideration of spatial dependence.*
- *Unambiguous modeling of spatial adjacency.*
- *Consideration of local interactions and global interactions.*
- *Relative proximity rather than absolute proximity.*
- *Non-stationary analysis.*
- *Exploratory rather than confirmatory (data-driven rather than argument driven).*
- *Versatility (multi-purpose).*
- *Consideration of background information.*

Now, we explain these issues in details.

- *Consideration of spatial dependence:*

Spatial dependence is a well-known special characteristic of geo-referenced data expressed in the First of Law of Geography [93]. Anselin discussed spatial dependence along with spatial heterogeneity [8]. His proposition saying “the phenomenon where locational similarity is matched by value similarity” [9] emphasizes the importance of spatial dependence among nearby neighbors. This spatial dependence is in opposition to the principle (or assumption) of independence underlying traditional statistics. For spatially correlated datasets, spatial clustering must consider spatial dependence. This is required to identify geographically aggregated clusters.

- *Unambiguous modeling of spatial adjacency:*

We indicated before that spatial clustering is strongly related to the discrete relation $is_nearby_neighbor \subset P \times P$ since nearby neighbors tend to form spatial clusters. Thus, spatial clustering becomes the matter of modeling and capturing spatial adjacency and proximity. In order to capture spatial dependence between nearby neighbors, we need to concisely define who the neighbors are and how far they are. Traditional raster-like modeling and vector-like modeling do not adequately handle the spatial adjacency of discrete point data [4, 36, 55]. More seriously, they constitute argument-dependent neighboring approaches that produce different neighboring information depending on argument values. These arguments are the cell size for raster-like modeling, and the number of neighbors and the radius for vector-like modeling (for instance, k -nearest neighboring and d -distance neighboring) [43]. Spatial clustering should avoid ambiguity and inconsistency in modeling spatial adjacency of point data as much as possible. Clustering results should not be brittle to those ambiguities. First, because user-supplied arguments inhibit the possibility of exploratory derivation of arguments from the data. Second, for data-rich environments, finding best values for arguments in argument-dependent modeling is time consuming.

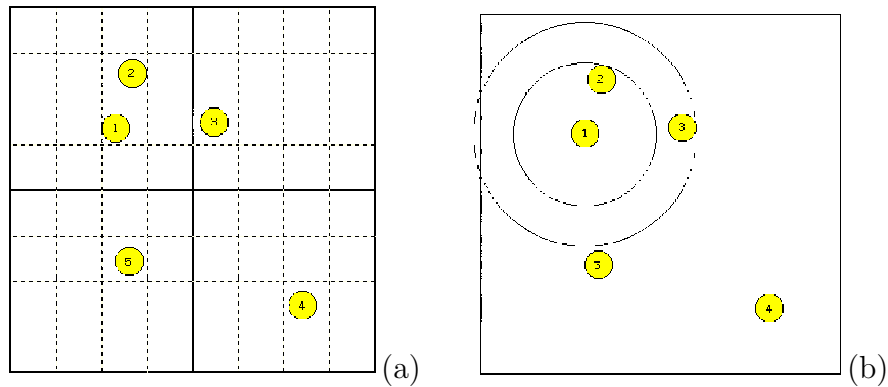


Figure 3-1: Neighbor modeling of point data with raster and vector approaches ($\|P\| = 5$): (a) The raster approach; (b) The vector approach.

Figure 3-1 illustrates the ambiguity caused by raster and vector modeling. The first source of ambiguity in the raster model is the shape of pixels and the pattern of neighbors. Each point shown in Figure 3-1(a) could have 4-connectivity neighbors (considering N, S, E and W cells) or 8-connectivity neighbors (considering N, S, E, W, NE, NW, SE and SW cells). If we consider 4-connectivity, the points p_1 and p_2 are considered as neighbors, but the points p_4 and p_5 are not. However, the adjacency relationship also heavily depends on the size of cell. In Figure 3-1(a), if the cell size (volume) becomes 16 times bigger (considering now solid lines, rather than dotted lines), then the points p_4 and p_5 become neighbors in the bigger cell size.

Similarly, it is a difficult task to define topological relationships of point data with vector models. This is because points do not have extent, but only have location. Thus, points do not exhibit topological relationships such as “adjacent”, “touch” or “intersect” unless they coincide. Often, metric relationships such as d -distance concepts are used to overcome this problem. Points lying within a certain distance are regarded as neighbors. This works in some sense. However, this adjacency relationship still presents inconsistencies similar to the problems mentioned about the raster model. For instance, two points p_1 and p_3 are not neighbors when a shorter distance (smaller circle in Figure 3-1(b)) is applied, but they become adjacent when a longer distance (larger circle) is used.

- *Consideration of local interactions and global interactions:*

When there is no interaction or no correlation among the phenomena registered at data points, the points do not form clusters. However, interactions result in data points that may either attract or repulse one another. Both attraction and repulsion will likely produce either clustered or regular distributions (note that physical models of attraction and/or repulsion are used for producing nice graph drawings [29]). If two points represent phenomena that attract each other, then locations for events of the phenomena become closer. The two data points belong to the same cluster. In graph drawings, if all end-points of

interactions repel one another equally, then the resulting distribution is regular. Inversely, strong attraction among a certain number of points may bring about isolated points, those too far become unattracted. Also, powerful repulsion causes clusters [36]. All this implies that isolated points (noise and other data points not belonging to a cluster) are not only generated by repulsion with cluster members, but also by attraction among other points in the dataset.

It is now obvious that both clusters and noise points are generated not only by interactions between their neighbors, but also other members in the dataset. A combination of local influence and global influence is the effect that shapes clusters. The behavior of spatial phenomena is the result of a combination of both local interactions (interactions confined to nearby neighbors) and global interactions (all the interactions within R) [10, 135]. Spatial clustering must take these both interactions into account. Thus, geo-referenced clustering should not be as crisp as local clustering approaches.

- *Relative proximity rather than absolute proximity:*

Spatial heterogeneity implies that each location in R has intrinsic uniqueness [10]. That is, conditions vary from place to place. Statistical properties including mean, variance and covariance are dependent on absolute location and have different interpretations over R . Although relative locations (distances) are the same, their relative interpretations may differ.

Figure 3-2 depicts a MST with 9 points. The two points, p_1 and p_6 , are both equidistant from p_5 . Clustering based on absolute distance will have the same grounds to place p_1 and p_6 in a cluster with p_5 . However, the length of edge $e_{1,5} = (p_1, p_5)$ seems to be relatively long with respect to the length of edge $e_{5,6} = (p_5, p_6)$ because of other points that create an heterogeneous perspective of the study region. Thus, p_5 is likely to belong to the same cluster as p_6 , but not to the cluster of p_1 . Although the lengths of the edges $e_{1,5}$ and $e_{5,6}$ are equal, they have different relative interpretations (relative to other points in R). Now, it is clear that relative proximity is more important than absolute proximity in

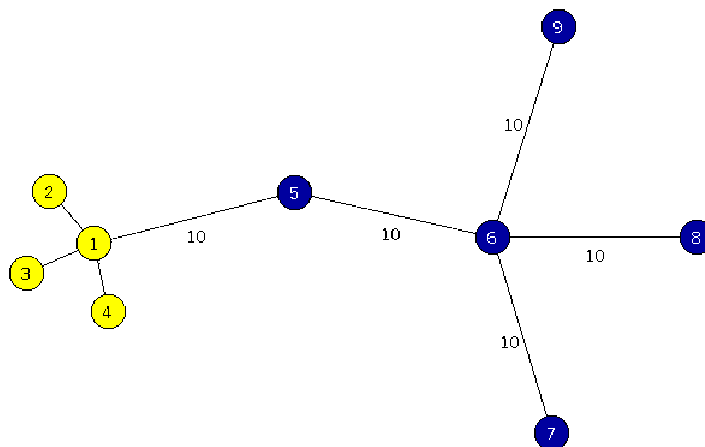


Figure 3-2: Data that illustrates the importance of relative proximity in spatial clustering ($\|P\| = 9$).

geo-referenced settings and thus in spatial clustering. Thus, spatial clustering must incorporate both relative locations and absolute locations to satisfy spatial heterogeneity.

- *Non-stationary analysis:*

The geographical property of non-stationarity also implies that local variations at one place are not the same as variations elsewhere. When analyzing a point p_i and the region surrounding it, considering local interactions and global interactions deals with how much to take into account those points close to p_i and how much to take into account those points far from p_i . Relative proximity is about considering distances in relation to other points. Non-stationarity means that these decisions can not remain static across the study region and that is somehow the dataset has to dictate how to account for them dynamically as the focus of analysis varies in the study region.

An example of stationary analysis is the use of global values as thresholds for spatial clustering. This not only ignores local variations, but disregards spatial heterogeneity. As a consequence, static model clustering approaches fail to identify locally aggregated sparse concentrations that are also of importance as

well as high-density concentrations. For instance, let us consider the dataset in Figure 3-2. Static model clustering approaches will report either one cluster (p_1, p_2, p_3 and p_4) while declaring all other points noise if global thresholds are set to a smaller value than 10, or alternatively one big cluster grouping all the points into a single cluster when thresholds are set to a greater value than 10. Static model clustering ignoring spatial heterogeneity is unable to report two distinct groups (dense and sparse (p_5, p_6, p_7, p_8 and p_9)) simultaneously. Thus, spatial clustering must have the properties of dynamic model clustering in order to belong to non-stationary analysis incorporating relative proximity and spatial heterogeneity.

- *Exploratory rather than confirmatory (data-driven rather than argument driven):*

Since clustering is to identify spatial clusters without apriori knowledge, we do not know how many clusters exist, where they are, how close they are and how big they are prior to clustering. These information should be derived from data, which underpins EDA and the principle to “let the data speak for themselves” [56, 112]. Openshaw [112] stresses that constraints and preconditions should be minimized in EDA and GDM. Hypotheses should be uncovered from the data, not from the user’s prior knowledge and/or assumptions. Minimizing the need to adjust arguments reduces the risk of tampering with the analysis tools. It also promises higher user friendliness. Traditional clustering methods share a need for user-specified arguments and prior knowledge to produce their best results. Such information needs are supplied as density threshold values in density-based clustering, merge or split conditions in hierarchical clustering, number of parts in partitioning clustering, prior probabilities and assumptions about the distribution of continuous attributes within classes in model-based clustering, or edge-cut thresholds in graph-based clustering. This parameter-tuning is extremely expensive and inefficient for huge datasets because it demands pre-processing and/or several trial and error steps. The need to find best-fit arguments in wide-spread semi-automatic clustering is not the only concern, slight changes of values of arguments produce totally different

clustering results. This sensitivity to user-supplied arguments raises serious questions regarding the validity of results.

- *Versatility (multi-purpose):*

The popular way of modeling real-world spatial data is McHarg's multi-layer view of the world [99]. In this view, each geographical layer captures something unique to it. Thus, many heterogeneous types of features are stored in GIS including 2-dimensional point data, 3-dimensional point data, obstacle data (rivers, mountains or political boundaries). There is also a diversity in scales or contexts, such as data from urban areas where the Manhattan metric is more applicable and data on a large scale where the Euclidean metric is more suitable. In data-rich environments, GIS consists of hundreds of layers that may contain thousands or millions of points. A sophisticated versatile spatial clustering that is generic to much of this diversity is in demand for such data-rich environments to handle many heterogeneous clustering settings such as clustering 2-dimensional point data in the presence of river obstacles in the Manhattan metric and grouping 3-dimensional point data in the absence of obstacles in the Euclidean metric.

- *Consideration of background information:*

Since there is high spatial and temporal correlation between geographical layers [93], incorporating background information to clustering may produce unexpected interesting patterns. A classical approach is the consideration of a layer of population-at-risk but an obstacle layer is also a possible candidate for background information. The former is well-known in CDA and suited for data-poor environments [16, 96, 111]. In CDA, a high concentration of disease is not considered as a cluster when it overlays on a densely populated city. A high concentration of disease cases where there is low population-at-risk, but a source of pollution, is seen as a genuine cluster. Thus, clusters are explained even before they are formed. Users generate hypotheses for confirmation, and as such, this approach requires thorough domain knowledge to select all possi-

ble background information that describes the population-at-risk to detect and confirm genuine clusters. Thus, this approach may fail to explore some possible associations and may not discover unexpected patterns. Note that, the use of improper population-at-risk layers produces invalid and false clusters. More seriously, it is an expensive and rather difficult task to select a proper population-at-risk layer in data-rich environments where there are many other layers to be explored as the possible population-at-risk. EDA and GDM regard the high concentration of disease as a cluster and look for possible causal factors. Then, the clusters are explained by automatic or semi-automatic means. In our illustration, some concentrations of disease will be explained by the concentration of population in a large city and others by the presence of pollutants. To the user, one may appear as an interesting association while the other seems not interesting (depending on the amount of background knowledge from the user).

Incorporating obstacles as background information has recently attracted attention in GDM [37, 40, 139, 140]. In many spatial settings, obstacles (rivers, mountains, lakes and political boundaries) prevent traversing the straight path between two points, and thus influence interactions among objects.

3.2 Traditional peak-inference clustering fails

One reason for the richness of clustering is that, clusters are at least in part in the eye of the beholder as much as there are inductive principles [23]. Any result of a clustering algorithm is a hypothesis that explains the data in some way. It is a selection among a family of models for the best fit of the model to the data. All clustering algorithms optimize an induction principle [147] that defines the family of models and the criteria by which a model is to be regarded as the one that best fits the data.

For GDM as an exploratory tool in data-rich environments, it is important to detect

and suggest as many as possible concentrations exhibiting similar characteristics in order not to miss patterns. Also, spatial clustering is a starting point to the analysis for further association explorations, and as such it must deliver high-quality clusters.

Although there exist many different clustering methods, none of the approaches satisfies the combination of the requirements discussed in the previous section. Since each method ignores some of the special characteristics of spatial data, they may be inapplicable to geographical space and may fail to detect geographically interesting groups. We illustrate this with several examples.

3.2.1 Sparse clusters adjacent to high-density clusters

Clustering approaches focus on finding relatively high-density areas (density-based), merging closer groups (hierarchical and model-based), assigning to closer prototypical representatives (partitioning and model-based) or detecting shorter or stronger interactions (graph-based). All typically focus on global peaks, thus they miss relatively sparse clusters adjacent to high-density clusters. This is because they are stationary clustering approaches ignoring spatial heterogeneity and relative proximity.

Consider the dataset shown in Figure 3-3(a). Most global peak-inference clustering algorithms report either

- three very dense clusters ($C1$, $C2$ and $C3$ in Figure 3-3(b)) when user-supplied arguments combine to produce very high-density clusters, or
- one big cluster ($C23$ in Figure 3-3(c)) and one dense cluster ($C1$), when user-supplied arguments are tuned to relax the criterion function in order to detect the sparse cluster ($C4$ in Figure 3-3(d)).

These algorithms are not able to detect the three high-density clusters and the sparse cluster (Figure 3-3(e)) simultaneously. Among the well-known clustering approaches

reviewed in Table 2.1, static model clustering approaches (BIRCH, CURE, DBSCAN, GAM, STING, WaveCluster, CLIQUE and SMTIN) belong to this case.

They are inclined to report the big cluster (C_{23} in Figure 3-3(c)) as one cluster because the intra-cluster distance of cluster C_4 (sparse cluster) is greater than the inter-cluster distance between the high-density clusters (C_2 or C_3) and the sparse cluster (C_4). We explain this with DBSCAN. Fixing the value of $MinPts$ to 4 and then tuning the other threshold Eps as originally suggested for the method, we see that DBSCAN detects three high-density clusters with $Eps = 50$, but leaves the sparse C_4 region undetected as shown in Figure 3-3(b). As we increase the value of Eps , some points within the sparse cluster C_4 are merged with the high-density clusters (C_2 and C_3), breaking the sparse cluster into many meaningless subclusters. Finally, when the value of Eps is large enough ($Eps = 150$) so all the points within the sparse cluster are in the same cluster, then the sparse cluster and the high-density clusters are all merged into a single cluster as shown in Figure 3-3(c).

Apart from the three high-density clusters, the sparse cluster is also important for further correlation investigation between themes. If we assume that points represent crime incidents around cities and surrounding suburbs, then these mere density peak-inference clustering is unable to find correlation between crime incidents and suburbs by missing the distinction between the sparse cluster and the high-density clusters. Often, this phenomenon is found in real data, such as big cities having more crime incidents surrounded by suburbs recording relatively less.

Another static model clustering approach, CLARANS, produces even worse results. Figure 3-3(f) depicts the clustering results from k -medoids clustering. Not only is the sparse cluster missed out, but clusters of arbitrary shapes are partitioned into meaningless subgroups.

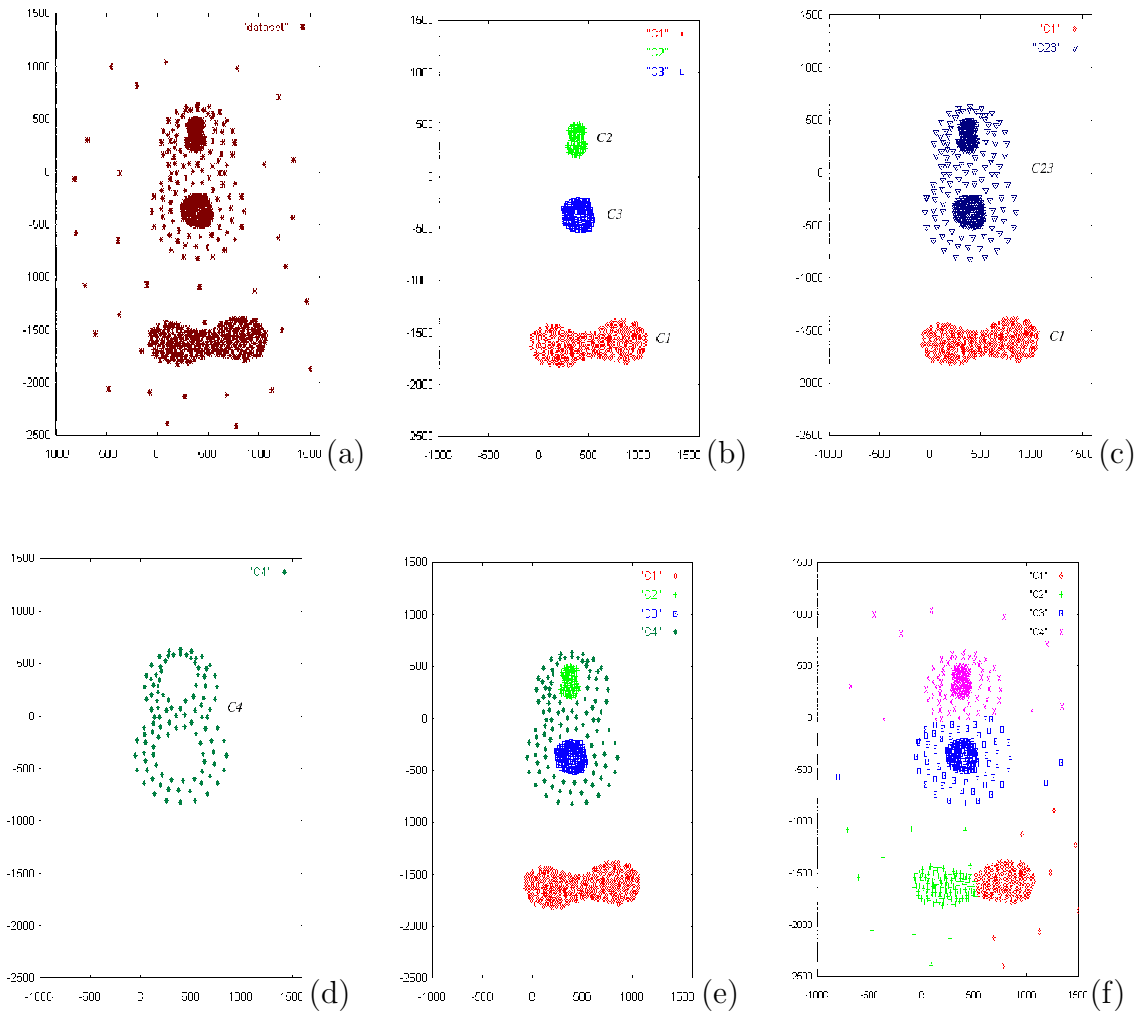


Figure 3-3: A sparse cluster adjacent to high-density clusters: (a) A dataset ($\|P\| = 600$); (b) Three high-density clusters; (c) One big cluster and one high-density cluster; (d) One sparse cluster; (e) Four clusters; (f) k -medoids clustering (4 clusters).

3.2.2 Multiple chains between clusters

As discussed in Section 2.4.1, hierarchical clustering with single-linkage works well for non-globular clusters but suffers from chains (small number of points that linearly link clusters). Chains may be of interest for some studies, but hinder cluster detection in data-rich environments [58]. Problems also arise for other strategies when clusters are connected only at chains. Well-known approaches for this attempt is to find all *bridges* and *cut-points* [149]. This works when only one chain connects two clusters.

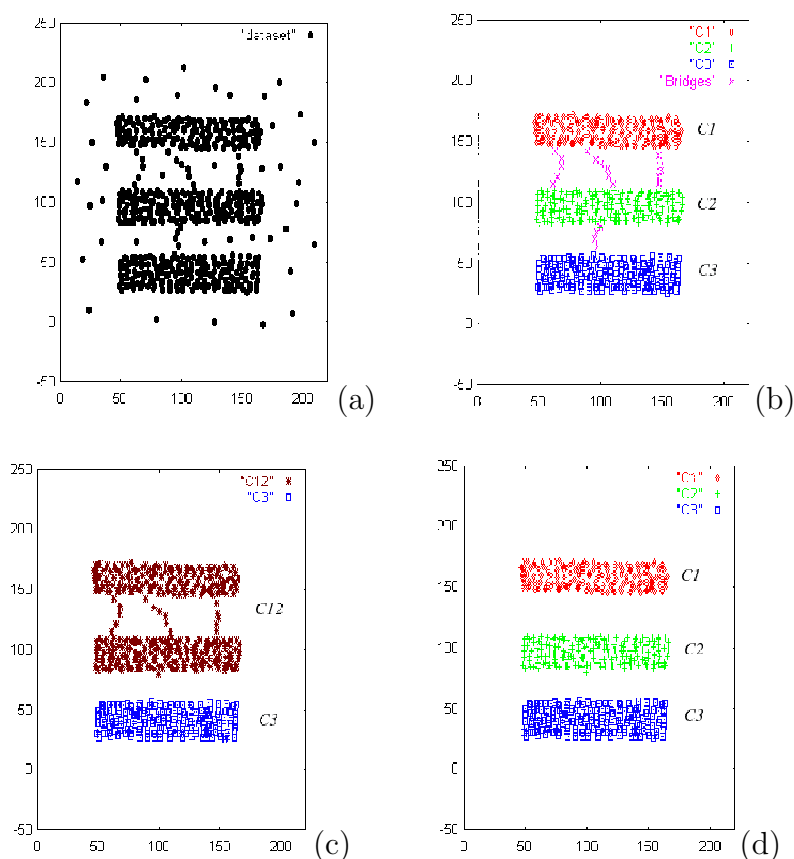


Figure 3-4: Multiple chains between clusters: (a) A dataset ($\|P\| = 555$); (b) Three clusters and chains; (c) One big cluster and one small cluster; (d) Three clusters.

Consider the data in Figure 3-4(a). Figure 3-4(d) shows three clusters ($C1$, $C2$ and

$C3$). Figure 3-4(b) shows that there are three path-like chains connecting two clusters ($C1$ and $C2$) while one connecting two clusters ($C2$ and $C3$). There are no *bridges* or *cut-points* between clusters $C1$ and $C2$ while there are between clusters $C2$ and $C3$. Therefore, the approach using *bridges* and *cut-points* fails to detect two clusters linked by multiple chains, and thus it generates two clusters $C12$ and $C3$ as shown in Figure 3-4(c).

In this example, points on chains are closely located. That is, distances to nearby neighbors from points on chains are as close as intra-cluster distances or even closer. Thus, using a global cut-off threshold to detect chains used in SMTIN, removing *bridges* or *cut-points* used in Zahn's clustering and partitioning nearest neighboring used in CHAMELEON are unable to detect these three clusters shown in Figure 3-4(d) simultaneously.

3.2.3 Closely located high-density clusters

Figure 3-5 illustrates another example of problems caused by static model clustering. Four high-density clusters and one sparse cluster seem to be obvious in Figure 3-5(a). Figure 3-5(d) highlights four high-density clusters with labels $C1$ to $C4$ and one sparse cluster with label $C5$. However, ignoring the importance of relative proximity and using only global arguments, it is very difficult to obtain a reply from a clustering system that suggests these five clusters simultaneously. Settings of user-supplied arguments alternate between replies that

- detect four high-density clusters ($C1$, $C2$, $C3$ and $C4$) leaving the sparse cluster ($C5$) out, perhaps as noise (refer to Figure 3-5(b)), and
- detect three high-density clusters ($C1$, $C2$ and $C34$) and one sparse cluster ($C5$) as illustrated in Figure 3-5(c).

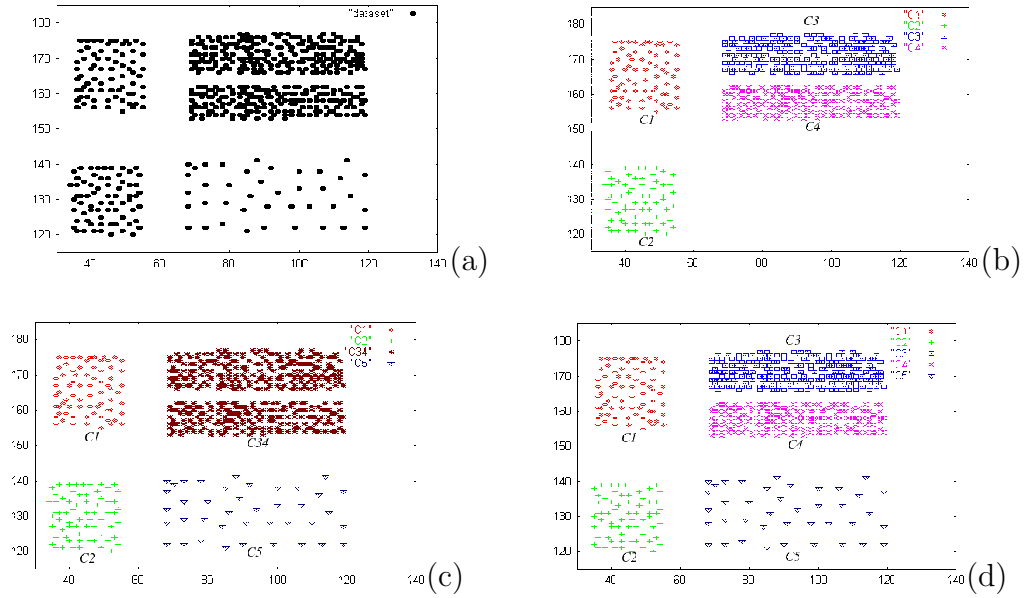
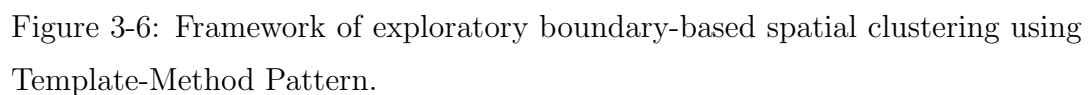


Figure 3-5: Closely located high-density clusters: (a) A data set ($\|P\| = 570$); (b) Four high-density clusters; (c) Three high-density clusters and one sparse cluster; (d) Five clusters.

The former is the result of arguments slightly of tune. For example, closer merge condition (BIRCH and CURE), shorter cut-off criterion function (SMTIN), higher density parameter or smaller size of kernel (DBSCAN, GAM, STING, WaveCluster and CLIQUE). For instance, DBSCAN with parameters $MinPts = 4$ and $Eps = 4$ has the output shown in Figure 3-5(b). By relaxing the global arguments, the sparse cluster now can be identified. But typically, because the separation between $C3$ and $C4$ is smaller than the intra-cluster distance in cluster $C5$, the two are combined into one high-density cluster $C34$ (refer to Figure 3-5(c)). This is the output of DBSCAN with parameters $MinPts = 4$ and $Eps = 7$. Other grid-based algorithms are also very sensitive to the size of the grid used to define the threshold for density and will suffer from the same problem.

Figure 3-6 depicts the UML ¹ framework of our multi-purpose exploratory boundary-based spatial clustering using the Template-Method Pattern [52, Section 5]. In fact,



¹UML stands for Unified Modelling Language and is a notation to present object-oriented structures, designs and artifacts.

between clusters. The procedure is a template method that can be instantiated to fit several purposes according to the concrete situation of the data, like 2D or 3D.

Because this is a framework (in the sense of Object Orientation), it admits extensions by user-supplied application-specific components. For example, our implementation of methods discussed in this thesis does not include the incorporation of the population-at-risk as background information to clustering because of the reasons we discussed in the previous section, but the interested user could plug in the necessary code and the entire scheme would then work with this variant.

Our exploratory spatial clustering allows the user to select a Minkowski metric that best describes P . Users can explore different proximity graphs and can incorporate sets of obstacles that block interactions between points in different ways. The spatial clustering method considers the characteristics of geoinformation and satisfies the requirements of spatial clustering discussed in this chapter. Thus, it is able to detect quality spatial aggregations including clusters of various densities, clusters of arbitrary shapes, clusters of different sizes, non-convex clusters, clusters with noise points, sparse clusters near high-density clusters, clusters linked by multiple chains and closely located high-density clusters. Figure 3-3(e), Figure 3-4(d) and Figure 3-5(d) illustrate clusters detected by the multi-purpose exploratory boundary-based spatial clustering method. These results exhibit more homogeneous distributions (characteristics) within clusters and thus our approach is more informative and precise. In addition, the versatile clustering approach can handle various spatial clustering settings such as 2-dimensional or 3-dimensional point data in different Minkowski metrics, both in the absence and presence of obstacles.

Further ahead, we will detail and illustrate examples of proximity graphs and the algorithm that carries out the analysis. However, it is important to emphasize that we have managed to separate the modeling issues from the algorithmic analysis issues. As a result, the algorithm is versatile, working in many different contexts essentially without modification. The algorithm can be coded in a template independently of the application and domain where the spatial clustering approach is to be performed.

3.4 Remarks

Bailey and Gatrell [10] remark “space can make a difference”. Data mining promises more precise and meaningful output with the incorporation of spatial dimensions. For instance, ‘crime incidents of urban areas are increasing and those of rural areas are decreasing’ is more precise and meaningful than ‘crime incidents are increasing or decreasing’. GIS organizes spatial data along with many layers that contain massive volume of heterogeneous data. In such data-rich environments, versatile exploratory spatial clustering is essential for GDM in order to explore various spatial settings and in order to discover valuable patterns. In this chapter, we discussed the special characteristics of geoinformation that spatial clustering must consider and provided a generic framework for versatile spatial clustering. Details of our clustering approach will be provided in subsequent chapters.

Chapter 4

Short-Long Clustering Criteria

This chapter describes our multi-purpose exploratory boundary-based clustering approach. We also refer to it as Short-Long clustering in this thesis for reasons that will be apparent soon. Section 4.1 discusses modeling issues of point data. Here, we show that the Delaunay diagram robustly and unambiguously defines spatial adjacency of point data. It also determines a succinct proximity graph in 2D, the Delaunay triangulation. Proximity graphs encode the modeling aspects required for Short-Long clustering. Then, in Section 4.2, we provide intuition and definitions before we discuss algorithms for our clustering method. Our method is essentially autonomous, however, it is possible to adjust a control parameter. Following this, in Section 4.3, we discuss the exploratory nature of our clustering and introduce methods that facilitate exploratory analysis using the control parameter. Finally, Section 4.4 presents time complexity analysis while Section 4.5 reports on performance evaluations.

4.1 Delaunay diagrams for spatial clustering

The modeling and structuring of the geo-referenced data is strongly related to the analysis functionality of GIS [116]. Thus, modeling, structuring and spatial analysis

are inter-related and play major roles for the success of informative mining of spatial data.

4.1.1 Modeling points with Delaunay triangulations

Besides geographical location, topological information is important to GIS, especially to spatial data analysis. Topology describes the relationship between neighboring spatial objects, for example *near_to*, *contain*, *touch* or *adjacent_to*. In practice, topological relationships reduce redundancy and enhance the integrity of spatial databases. Proper modeling of spatial proximity provides useful topological information for spatial data analysis. Neighbors and adjacency can be defined in obvious ways by topological relationships such as *intersect* or *meet* (detecting intersections or common boundaries between objects). Unfortunately, points neither intersect nor meet, unless they overlap. One possible way of capturing spatial proximity of point data is to perform point-to-area transformations. Now, the topological relationship *meet* can be applied. Thus, two points are considered as neighbors if their corresponding areas share a common boundary. A widely adopted point-to-area transformation is to assign every location in R to the nearest p_i in P . This creates regions in R . The regions are mutually exclusive regions except for their boundaries and collectively such regions are exhaustive [110]. Under the Euclidean metric, the resulting tessellation is the well-known Voronoi diagram denoted by $VD(P)$. As a consequence, two points are neighbors if and only if their territories (Voronoi regions) share a common Voronoi boundary. In 2D, the explicit representation of this *is_nearby_neighbor* relation is another tessellation (the Delaunay triangulation denoted by $DT(P)$). It is the graph dual of $VD(P)$ and can be obtained by connecting two neighboring points in $VD(P)$. Under the assumption that no more than 3 cocircular points occur in P , the proximity graph $DT(P)$ unambiguously defines spatial adjacency.

Figure 4-1 demonstrates the unambiguous modeling by the Voronoi diagram in the Euclidean metric. The Voronoi diagram approach has a number of merits over con-

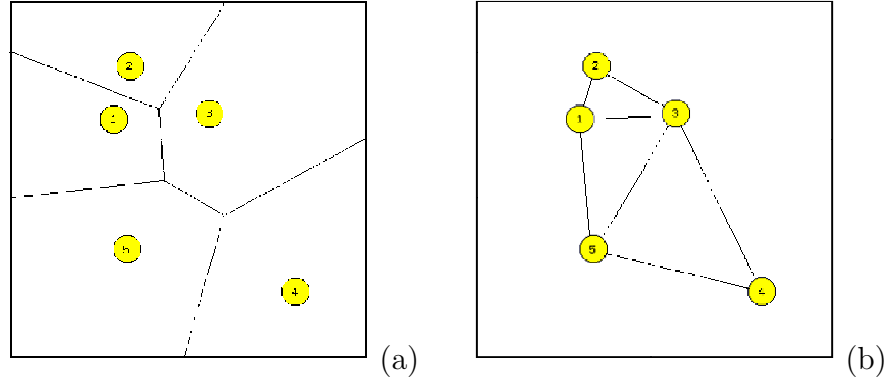


Figure 4-1: The modeling and structuring of point data with the Voronoi diagram and Delaunay triangulation ($|P| = 5$, the configuration of points is the same as in Figure 3-1): (a) The Voronoi diagram; (b) The Delaunay triangulation.

ventional data models. First, the dual Delaunay triangulation represents adjacency information explicitly. In 2D, the Delaunay triangulation can be constructed in only $O(n \log n)$ time. The structure is succinct; the number of edges is linear in the number of points and the expected number of edges incident to a certain point is limited to a constant value. In addition, the triangulation is as equilateral as possible, so that the unexpected effect of long elongated edges can be minimized. Given these reasons, the Delaunay triangulation is a very solid candidate for extracting clustering information among points. In addition, the Delaunay triangulation is a spanner [78]. A graph G is a spanner if it encodes all proximity information approximately. That is, for any pair of points p_i and $p_j \in P$, the shortest path along G from p_i to p_j has a length competitive to (approximates within a bounding constant) the path between p_i and p_j . For instance, the sum of the length of Delaunay edge $e_{2,3}$ and the length of Delaunay edge $e_{3,4}$ in Figure 4-1(b) approximates the Euclidean distance between p_2 and p_4 .

The complete graph on P models and captures all the interactions within P . However, it has $\Theta(n^2)$ size. This is excessive for data mining. $DT(P)$ approximates the complete graph on P with linear size. This is not excessive and remains informative. Thus,

Delaunay triangulations robustly capture and quantify spatial dependence between points in P , which underpins the First Law of Geography discussed in Section 2.1. Argument-dependent modeling methods such as k -nearest neighbors and d -distance neighbors are generally not spanners since they can be disconnected. With k much smaller than n , the k -nearest neighbor is a linear size proximity graph. But for more precise proximity information, a larger value of k is necessary. Similarly with proximity graphs like d -distance neighbors. The trade-off is precise modeling of proximity information versus space requirements. As k and d increase, these proximity graphs become closer to the complete graph on P . Thus, the use of larger values of k and d causes a computational burden while the use of smaller values may ignore informative interactions between points.

4.1.2 Avoiding ambiguities in Delaunay triangulations

In a proximity graph, points are connected by edges if and only if they seem to be close by some proximity measure [92]. If this simple rule is applied to clustering, two points belong to the same cluster if they are connected by a small enough edge in the proximity graph. Analysis of Delaunay edges is very promising for clustering because a Delaunay edge not only indicates that its end-points are close to each other, but also that there are no other points in the vicinity of the end-points. This is referred as the empty circumcircle criterion [110]. Two Voronoi neighbors define an edge of a triangle of the Delaunay triangulation, the circumcircle of which is empty of any other points in the dataset. Thus, this accounts for spatial heterogeneity and relative measurement of separation. These triangles are not affected by insertions/deletion of points far away but get affected by nearby updates, thus this modeling is dynamic in the extent of the study region.

However, use of the Delaunay triangulation requires removing some ambiguities. In the Delaunay triangulation, three points determine a Delaunay triangle if its circumcircle does not contain any other points in its interior. However, when more than

three points lie on a circumcircle (where all other points in the dataset are outside the circle), a unique triangulation can not be defined. Whilst several possible alternatives of selecting Delaunay edges result in a valid dual of the Voronoi diagram, a problem arises for the purposes of analyzing edges and proximity.

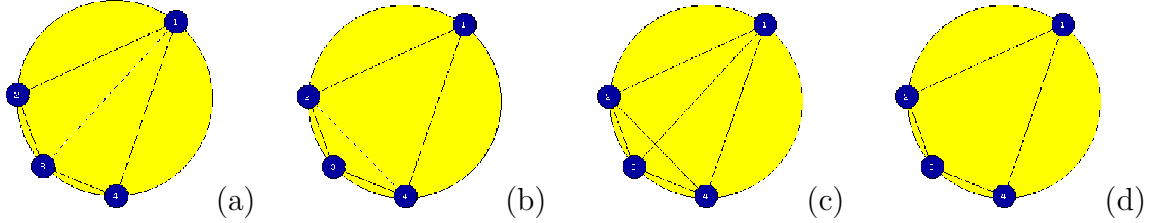


Figure 4-2: Two possible triangulations when four points are cocircular ($\|P\| = 4$): (a) Triangulation with longer edge; (b) Triangulation with shorter edge; (c) Delaunay graph with all possible edges; (d) Delaunay diagram with no ambiguous edges.

Consider Figure 4-2. There exist two possible triangulations when four points are cocircular. Proximity information and summary statistics are not the same in the two different cases. Two points p_2 and p_4 are not considered as neighbors in Figure 4-2(a), but they are in Figure 4-2(b). On the other hand, two adjacent points p_1 and p_3 in Figure 4-2(a) become unrelated in Figure 4-2(b). Furthermore, the sum (and thus, the average) of the lengths of Delaunay edges, the degrees (the number of edges incident to a point) and the standard deviation vary between the two cases. Accepting any of the valid triangulations over the other alternatives would artificially introduce structure not really existing in the data. There are two remedies to overcome this problem. First, all possible edges are taken into account as depicted in Figure 4-2(c). In this case, the number of edges is no longer linear to the number of points. Moreover, it causes intersection between edges (removing the property that the Delaunay triangulation is a planar map) resulting in a complex structure to implement. The second alternative removes diagonals from the Delaunay triangulation as illustrated in Figure 4-2(d). This is the Delaunay diagram [101], denoted by $DD(P)$, and makes the embedded planar map simple. If a face is not a triangle, it is because its points are cocircular. Thus, it eliminates the statistical side effects caused by triangles when

cocircularity happens. This Delaunay diagram, constructed by removing all ambiguous diagonals in the Delaunay triangulation, is the source of edge proximity analysis in this thesis.

4.1.3 Qualities of Delaunay diagrams for spatial clustering

The Delaunay diagram has a number of unique characteristics and offers at least three direct avenues towards spatial features for clustering. These are perimeter (the sum of length of edges in a face), angle and Delaunay edge with length. Researchers [35] have successfully derived the number of clusters using the perimeter. Analyzing perimeters of Delaunay triangles is particularly intuitive because points determine a Delaunay face (interior) if and only if its circumcircle does not include any other points in the dataset. Thus, when a perimeter is small, the points in the face must be close together and belong to the same cluster. Because Delaunay triangles maximize the smallest angle, if the perimeter is large, it is because some points in the face is not in a common cluster. Unfortunately, analyzing angles or analyzing edges is not as intuitive because neither angles nor edges directly determine the Delaunay diagram. However, as features of a triangulation, they encode significant proximity information. The angle is widely used in point pattern analysis to decide whether a set of points is randomly distributed, regularly distributed or exhibiting clusters [18].

The edge (and its length) is used to detect arbitrary shapes of clusters without requiring the number of clusters [31, 36, 75]. Using edge analysis has a couple of advantages over the analysis of perimeter and angle measurements for clustering. It is simpler than the perimeter and the angle. Each edge represents an interaction between a pair of points while three points are required to determine each perimeter and angle (and which point to add to the pair for the analysis is not immediate). Thus, the perimeter and angle are closer to the analysis of triangles in the Delaunay triangulation. This complicates the definition of a discrete relation of proximity between a pair of points. Secondly, the length of an edge inversely represents the strength of

interaction between points (recall the distance decays effect [10]). For instance, in Figure 4-2(a) the pair of points (p_2 and p_3) is a stronger association (relation) than the pair of points (p_1 and p_2). Thus, edges encode proximity and density. Table 4.1 depicts the characteristics of these three properties.

Table 4.1: Comparison of properties of the Delaunay diagram for clustering.

	perimeter	angle	edge with length
considers empty circumcircle property	for intuitive justification	no explicit consideration	no explicit consideration
previously used for	finding the number of clusters	point pattern analysis	finding clusters of arbitrary shapes
structure of a discrete relation	more complex (3 points)	more complex (3 points)	simple and explicit (2 points)
the strength of proximity	not able to represent	not able to represent	able to represent (inversely)

In fact, Delaunay edges consider implicitly the empty circumcircle property as well as many other relative proximity properties because the Delaunay triangulation includes many other proximity graphs (like the Gabriel graph, the relative neighbor graph and the MST). Thus, we argue that Delaunay edges with their lengths are suitable features for edge-removal analysis to obtain clusters.

This approach has gained popularity. The simplest example is again the MST or single-linkage hierarchical clustering. The intuition of edge-removal approaches is similar. One-dimensional classification of edges results in clusters. However, the projection of high-dimensional data (2- or 3-dimensional) to 1-dimensional data loses relative and absolute locational information and thus ignores spatial heterogeneity. More seriously, the edge-removal approaches are edge-centric. Note that, clustering is a point-centric (node-centric) process. Edges alone contain less information than geo-referenced points. An edge only has information about two end-nodes and their absolute closeness while a point regarded as a node of the planar map has information about its nearby neighbors and their closeness to the node. Thus, the edge-centric approaches are clustering methods with less information. Another drawback of these

previous ideas is that they use directly Delaunay triangulations and return inconsistent results in the presence of cocircularity. This kind of static approach is not able to detect local variations, and again, fails to identify clusters in situations like those illustrated in Section 3.2.

4.2 Short-Long clustering criteria

Boundaries are informative to identify clusters since the two sides of a boundary have different characteristics. The Short-Long clustering method is a boundary-based clustering method. The main principle behind boundary-based clustering is to extract cluster boundaries from proximity graphs that model discrete point data. The Short-Long clustering approach based on the Delaunay diagram attempts to locate globally significant cluster boundaries in the Delaunay diagram. Local neighboring information suggests candidates for cluster boundaries and global variation information determines if suggested candidates are globally significant or not. The local neighboring information includes local adjacency and density information around each point that vary over R . Thus, Short-Long clustering is a node-centric dynamic method.

4.2.1 Working principle

Sharp changes in point density are informative of cluster boundaries since the two sides of a cluster boundary have different densities (one is dense while the other is sparse). These sudden alternations separate points into similar subgroups and thus they form cluster boundaries. These cluster boundaries occur where there is high discrepancy (great variability) among the lengths of incident edges on a vertex p in the Delaunay diagram. We call border points those points in the proximity graph that are on cluster boundaries. These border points have two different types of incident edges: short edges and long edges. Short edges typically connect points within a cluster. Edges between points in the same cluster are called intra-cluster edges. Long edges

straddle between clusters or between a cluster and noise (or outliers). Edges between points in different clusters are inter-cluster edges. These relatively close and remote neighbors to border points contribute to the characterization of cluster boundaries. Interestingly, border points share some of these characteristics with points on chains. These are called chain points. They also have both short edges linking to other neighboring chain points or border points and long edges connecting to noise points. We will later characterize chain points.

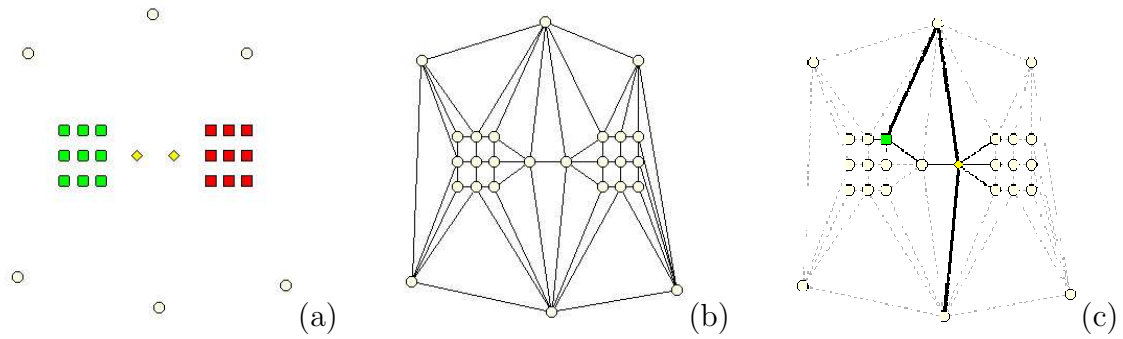


Figure 4-3: Illustration of the characteristics of border and chain points: (a) A dataset ($\|P\| = 26$); (b) $DD(P)$; (c) Incident edges to a border point and a chain point.

As an example, the dataset shown in Figure 4-3(a) seems to have two clusters linked by two chain points. Figure 4-3(c) highlights a border point (rounded rectangle) and a chain point (rhombus) with incident edges in solid lines. Both have exceptionally long edges in thick solid lines and exceptionally short edges in thin solid lines. As discussed in Section 3.2.2, chains connect two different spatial aggregations and obstruct cluster detection. In order to differentiate these chain points from border points, Short-Long clustering permanently removes exceptionally long edges and temporarily removes exceptionally short edges. This removal makes some border points and chain points having only exceptionally long and short edges isolated. Thus, clusters linked by multiple chains are identified. Later, Short-Long clustering performs connected component analysis to recuperate all short edges that are intra-cluster edges (typically incident to border points) and to permanently remove chains.

4.2.2 Intuition and definitions for point-centric statistics

There are several measures for describing dispersion. Undoubtedly, the standard deviation and its square, the variance, are the most valuable and popular [144]. Thus, Short-Long clustering detects points whose lengths of incident edges exhibit unusually large standard deviation. In order to precisely define this notion, we introduce the following definitions. The graph $DD(P)$ is a planar map that contains vertices (nodes) and edges. Here, nodes represent data points and edges connect pairs of nearby neighbors. For a point $p_i \in P$ (a vertex of $DD(P)$), the neighborhood $N(p_i)$ is the set of Delaunay edges incident to point p_i .

Definition 4.2.2.1 (Point-centric statistic 1) We denote by $Local_Mean(p_i)$ the mean length of edges in $N(p_i) = \{e_j | p_i \text{ is incident in } e_j\}$. That is,

$$Local_Mean(p_i) = \sum_{e \in N(p_i)} |e| / \|N(p_i)\| = \sum_{j=1}^{d(p_i)} |e_j| / d(p_i), \quad (4.1)$$

where $d(p_i)$ denotes the number of Delaunay edges incident to p_i (the degree of p_i in the graph theory sense), and $|e|$ denotes the length of Delaunay edge e .

Definition 4.2.2.2 We denote by $Local_St_Dev(p_i)$ the standard deviation of the lengths of edges in $N(p_i)$. That is,

$$Local_St_Dev(p_i) = \sqrt{\sum_{j=1}^{d(p_i)} (Local_Mean(p_i) - |e_j|)^2 / d(p_i)}. \quad (4.2)$$

Firstly, Short-Long clustering needs to quantify relative proximity of incident edges to each $p_i \in P$. Edges e_j incident to a certain point p_i could be potentially classified into inter-cluster edges if

$$|e_j| > Local_Mean(p_i) + l \times Local_St_Dev(p_i), \quad (4.3)$$

where l is a scaling factor. If the factor l scales globally, it will not apply local and non-stationary information but global information to all points. This is because

$Local_St_Dev(p_i)$ increases as $Local_Mean(p_i)$ increases. However, a point p_i with a smaller value of $Local_Mean(p_i)$ is likely to be a point lying within a cluster (an inner point) — while if a point p_j exhibits a larger value, it is more likely to be noise. Thus, we need to vary the value l in order to incorporate the relative tendency. Namely, a smaller value of $Local_Mean(p_i)$ requires a larger value of l while a larger value of $Local_Mean(p_i)$ requires a smaller value of l .

More seriously, $Local_Mean(p_i)$ and $Local_St_Dev(p_i)$ are statistical indicators based on only $d(p_i)$ length values. This is a very small sample, because $E[d(p_i)] = 6$ in Delaunay diagrams [110]. Thus, we now introduce measures that incorporate global and local interactions, and in fact, this measure will result in tests that make a decision on each edge using the data of all $\Theta(n)$ edges in $DD(P)$.

Definition 4.2.2.3 (Point-centric statistic 3) *We denote by $Mean_St_Dev(P)$ the average of the $Local_St_Dev(p_i)$ values as we let p_i be each point in P . That is,*

$$Mean_St_Dev(P) = \sum_{i=1}^n Local_St_Dev(p_i)/n. \quad (4.4)$$

Here, n is the number of points in P . Note that, a value of edge length spread like $Local_St_Dev(p_i)$ is large or small only in relation to the global value reflected in $Mean_St_Dev(P)$. Thus, we introduce an indicator of the relative size of $Local_St_Dev(p_i)$ with respect to $Mean_St_Dev(P)$.

Definition 4.2.2.4 *We let $Relative_St_Dev(p_i)$ denote the ratio of $Local_St_Dev(p_i)$ and $Mean_St_Dev(P)$. That is,*

$$Relative_St_Dev(p_i) = Local_St_Dev(p_i)/Mean_St_Dev(P). \quad (4.5)$$

$Relative_St_Dev(p_i)$ represents the required relative deviation. The value of $Relative_St_Dev(p_i)$ is less than 1 for a point p_i that locally exhibits less spread of lengths of its incident edges than the generality of P , while it is greater than 1 for a point p_i

that locally exhibits more variability. The inverse of $Relative_St_Dev(p_i)$ fulfills the role required for the scaling factor l , thus it is replaced as follows.

$$l = m \times Relative_St_Dev(p_i)^{-1} = m \times Mean_St_Dev(P) / Local_St_Dev(p_i), \quad (4.6)$$

where m is a control value.

If we replace Equation 4.6 in Equation 4.3, then we get the following derivation.

$$\begin{aligned} |e_j| &> Local_Mean(p_i) + m \times Relative_St_Dev(p_i)^{-1} \times Local_St_Dev(p_i) \\ &= Local_Mean(p_i) + m \times \frac{Mean_St_Dev(P)}{Local_St_Dev(p_i)} \times Local_St_Dev(p_i) \\ &= Local_Mean(p_i) + m \times Mean_St_Dev(P). \end{aligned} \quad (4.7)$$

Thus, edges that satisfy Equation 4.7 are defined as globally long edges. That is, edges incident to p_i are globally long edges if their lengths exceed the tolerance of $m \times Mean_St_Dev(P)$. Thus, $m \times Mean_St_Dev(P)$ is referred as a tolerance value.

With this tool, we propose to discriminate edges into those unusually short, those unusually long and those that seem to be proper intra-cluster edges.

Definition 4.2.2.5 *An edge $e_j \in N(p_i)$ belongs to $Short_Edges(p_i)$ if and only if the length of e_j is less than $Local_Mean(p_i) - m \times Mean_St_Dev(P)$. That is,*

$$\begin{aligned} Short_Edges(p_i) = \\ \{e_j \in N(p_i) \mid |e_j| < Local_Mean(p_i) - m \times Mean_St_Dev(P)\}. \end{aligned} \quad (4.8)$$

Definition 4.2.2.6 *An edge $e_j \in N(p_i)$ belongs to $Long_Edges(p_i)$ if and only if the length of e_j is greater than $Local_Mean(p_i) + m \times Mean_St_Dev(P)$. That is,*

$$\begin{aligned} Long_Edges(p_i) = \\ \{e_j \in N(p_i) \mid |e_j| > Local_Mean(p_i) + m \times Mean_St_Dev(P)\}. \end{aligned} \quad (4.9)$$

Definition 4.2.2.7 *Other_Edges(p_i)* denotes the subset of $N(p_i)$ whose edges are greater than or equal to $Local_Mean(p_i) - m \times Mean_St_Dev(P)$, and less than or equal to $Local_Mean(p_i) + m \times Mean_St_Dev(P)$. That is,

$$Other_Edges(p_i) = N(p_i) - (Short_Edges(p_i) \cup Long_Edges(p_i)). \quad (4.10)$$

Note that, for each p_i , the edges in $N(p_i)$ are partitioned into exactly 3 subsets: $Short_Edges(p_i)$, $Long_Edges(p_i)$ and $Other_Edges(p_i)$. However, an edge linking p_i with p_j may belong, for example, to both $Short_Edges(p_i)$ and $Other_Edges(p_j)$.

The Short-Long clustering criteria are formalized by Equation 4.8, Equation 4.9 and Equation 4.10. These criteria utilize both local trend and global trend. In particular, $Local_Mean(p_i)$ represents inverse local strength and $Mean_St_Dev(P)$ denotes the global degree of variation in R . Note that, $Local_Mean(p_i)$, $Local_St_Dev(p_i)$ and $Mean_St_Dev(P)$ are point-centric statistics and two of which vary with p_i . Thus, all the Short-Long clustering criteria are not static, but dynamic over R .

4.2.3 A three-phase Short-Long clustering algorithm

The Short-Long clustering method consists of three phases. Phase $\mathcal{I}$ classifies incident edges into $Short_Edges(p_i)$, $Long_Edges(p_i)$ and $Other_Edges(p_i)$ for each p_i . Phase $\mathcal{I}$ also removes all edges in $Short_Edges(p_i)$ and $Long_Edges(p_i)$ for each point p_i in P . This produces rough boundaries for clusters, perhaps with some chains and perhaps isolating some points. For each point p_i , Phase $\mathcal{II}$ recovers edges in $Short_Edges(p_i)$ as long as they do not merge connected components having more than one element, if this happens, it may re-establish the connected components by removing some $Other_Edges(p_i)$. Finally, Phase $\mathcal{III}$ looks for inter-cluster edges in a local region of each p_i . Figure 4-4 shows the operation of these phases in a dataset with 3 spatial aggregations. The Delaunay diagram is presented in Figure 4-4(a), while Figure 4-4(b) shows the conclusion of Phase $\mathcal{I}$. Application of the second phase results in

Figure 4-4(c). Data points isolated are attached back. Phase *III* breaks the links between the two left clusters. Figure 4-4(d) depicts the final result.

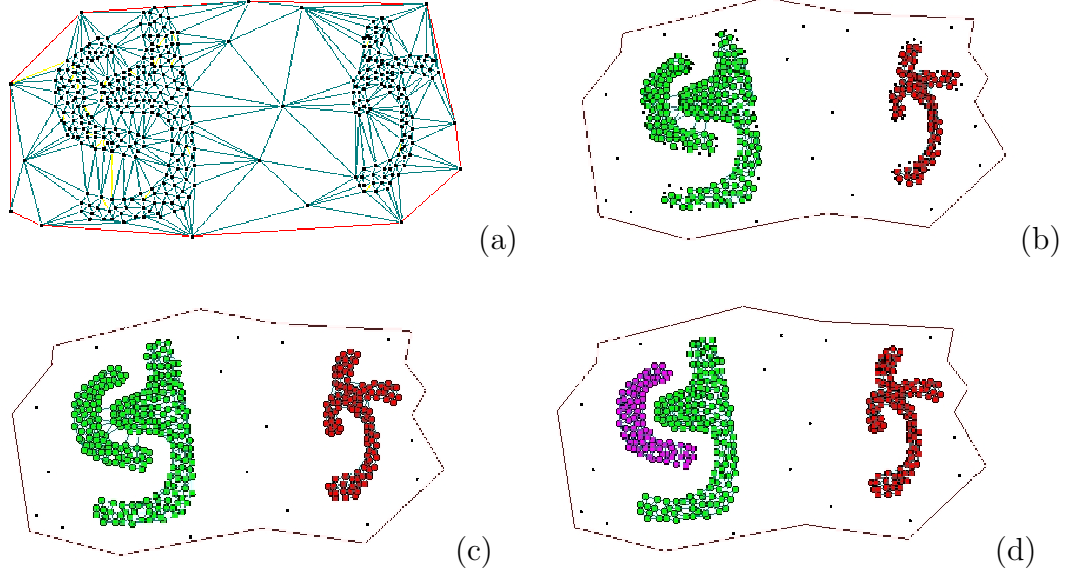


Figure 4-4: The three-phase Short-Long clustering procedure ($m = 1$): (a) Delaunay diagram ($\|P\| = 391$); (b) After Phase *I* (initial 2 clusters); (c) After Phase *II* (2 clusters with refinement); (d) After Phase *III* (3 clusters).

Phase *I*: Finding rough cluster boundaries

The two tasks of Phase *I* are:

1. Permanently remove $Long_Edges(p_i)$ for each $p_i \in P$.
2. Temporary remove $Short_Edges(p_i)$ for each $p_i \in P$.

The main objective of this phase is to extract initial rough boundaries by eliminating inter-cluster edges. Our earlier discussion justifies in detail the removal of all edges in $Long_Edges(p_i)$ from the Delaunay diagram. They simply are too long and are merging two distinct clusters. We stress that some edges $e_{i,j} = (p_i, p_j)$ in $Short_Edges(p_i)$

where p_j is in the same cluster is temporary removed and to be recuperated in the next phase to distinguish border points from chain points.

Phase $\mathcal{II}$: Refining clusters by restoring and re-attaching

The goal of this phase is to restore the edges $e_{i,j} = (p_i, p_j)$ in $Short_Edges(p_i)$ where p_i and p_j are in the same cluster. Also, the goal is to remove some $Other_Edges(p_i)$ that are creating paths that may result in chains or links between border points and noise points. Thus, the first step of Phase $\mathcal{II}$ is to compute connected components and label each point p_i with its connected component $CC[p_i]$. If a connected component has more than one element, it is called *non-trivial*. If p_i is the only point in a connected component, then the point p_i is isolated and called *singleton*. In this case, $CC[p_i]$ is not a cluster.

To introduce the processing of Phase $\mathcal{II}$, we bring attention to the following observation.

Observation 4.2.3.1 *Let $p_i \in P$ be such that $Other_Edges(p_i) \neq \emptyset$. Then, all edges $e \in Other_Edges(p_i)$ connect p_i to the same non-trivial connected component.*

Using our notation, this means that if two edges $e_{i,j} = (p_i, p_j)$ and $e_{i,k} = (p_i, p_k)$ are in $Other_Edges(p_i)$, then $CC[p_j] = CC[p_k]$. The observation follows trivially from the fact that p_j and p_k are connected by the path $e_{i,j}$ and $e_{i,k}$. Now, we recuperate $Short_Edges(p_i)$, and in some cases, revise the connected component for p_i . If all edges in $Short_Edges(p_i)$ suggest unanimously a connected component, p_i is integrated to that suggestion. If there are two edges in $Short_Edges(p_i)$ suggesting different connected components, the largest is adopted. While $Other_Edges(p_i)$ participate in the creation of the connected components, now they delegate precedence to the suggestion by $Short_Edges(p_i)$ (which in the large majority of the cases is the same suggestion). The component swap typically occurs on isolated border points that have

no *Other_Edges* (for example, the isolated points on the left of the bottom cluster in Figure 4-4(b)). Details of this process are as follows.

1. For each point p_i , we perform the following steps if $Short_Edges(p_i) \neq \emptyset$ and $\exists e_{i,j} = (p_i, p_j) \in Short_Edges(p_i)$ satisfying that $CC[p_j]$ is a non-trivial connected component.
 - (a) Determine a candidate connected component C for p_i .
 - i. If there are two edges $e_{i,j} = (p_i, p_j)$ and $e_{i,k} = (p_i, p_k)$ in $Short_Edges(p_i)$ with $CC[p_j] \neq CC[p_k]$, then
 - A. Compute, for each edge $e_{i,j} = (p_i, p_j) \in Short_Edges(p_i)$, the size $\|CC[p_j]\|$ and let $MX = \max_{e_{i,j}=(p_i, p_j) \in Short_Edges(p_i)} \|CC[p_j]\|$.
 - B. Let C be the class label of the largest connected component (if there are two different connected components with cardinality MX , we let C be the one with the shortest edge to p_i).
 - ii. Otherwise, let C be the label of the connected component to which all edges $e \in Short_Edges(p_i)$ connect p_i .
 - (b) If the edges in $Other_Edges(p_i)$ connect p_i to a connected component different from C , remove them. Note that,
 - because of Observation 4.2.3.1 above, when this case applies, all edges in $Other_Edges(p_i)$ are removed, and
 - only in this case, will p_i swap non-trivial connected components.
 - (c) Restore all edges $e \in Short_Edges(p_i)$ that connect p_i to C .
2. Re-compute connected components.
3. For each singleton p_i ($CC[p_i] = 1$), we restore all $Short_Edges(p_i)$ if $\forall (e_{i,j} = (p_i, p_j)) \in Short_Edges(p_i)$, $CC[p_j]$ is the same connected component C .

Thus, Phase $\mathcal{II}$ incorporates points that are isolated in singleton connected components to non-trivial connected components if and only if they are very close (they have

at least one *Short_Edges* connected to a non-trivial connected component) in Step 1. Singletons whose incident edges connected to another singletons are re-attached in Step 3. Phase *II* also re-assigns a point p_i to C' although it is linked to a cluster C by *Other_Edges*(p_i) if there are *Short_Edges*(p_i) that indicate membership to $C' \neq C$. Thus, Phase *II* refines cluster boundaries.

Phase *III*: Detecting second-order inconsistency

After the first two phases, the current subgraph of $DD(P)$ is the result of local analysis with regard to global trend. To complete the analysis, the immediate neighborhood is extended slightly. We extend the notion of neighborhood of a vertex p_i in a graph G as follows.

Definition 4.2.3.2 *The k -neighborhood $N_{k,G}(p_i)$ of a vertex p_i in graph G is the set of edges of G that belong to a path of k or less edges starting at p_i .*

The third phase now computes *Local_Mean*(p_i) for each p_i in P for the 2-neighborhood. This is defined as follows.

Definition 4.2.3.3 (Point-centric statistic 4) *We denote by $Local_Mean_{k,G}(p_i)$ the mean length of edges in $N_{k,G}(p_i)$. That is,*

$$Local_Mean_{k,G}(p_i) = \sum_{e \in N_{k,G}(p_i)} |e| / \|N_{k,G}(p_i)\|.$$

For each point p_i , this phase simply removes all edges in $N_{2,G}(p_i)$ that are long edges. That is, all $e \in N_{2,G}(p_i)$ such that $|e| > Local_Mean_{2,G}(p_i) + m \times Mean_St_Dev(P)$. Thus, Phase *III* removes unusually long edges in the 2-neighborhood, which reduces inter-cluster dissimilarity and elevates intra-cluster similarity.

4.3 Control value m

As indicated in Definition 4.2.2.5 and Definition 4.2.2.6, the tolerance value, $m \times \text{Mean_St_Dev}(P)$, determines a range of acceptable heterogeneity. If the tolerance value grows, then most edges are considered as homogeneous. That is, less number of edges will likely belong to *Short_Edges* and *Long_Edges*. Thus, less number of edges will be removed in Phase *I* and Phase *III*. Namely, relatively heterogeneous lengths of edges around p_i are preserved. Points tend to be linked to their neighbors and thus close clusters likely merge into the same cluster. If the tolerance value drops, then exactly the opposite occurs. The tolerance value does not affect $\text{Mean_St_Dev}(P)$. This is a value dependent only on the dataset P . Thus, the control value m becomes an exploratory tool. This section investigates and provides a good initial suggestion for the value of m .

4.3.1 One standard deviation of the mean

Slightly more than two-thirds of the values in the normal distribution lie within one standard deviation of the mean [144]. That is, relatively homogeneous values concentrate around the mean and spread within one standard deviation of the mean. The values within this range are relatively close to the mean and thus exhibit a certain level of consistency (homogeneity). On the other hand, the values outside this range are relatively away from the mean and suggest inconsistency (heterogeneity). Thus, one standard deviation of the mean is a solid classifier in the normal distribution.

Short-Long clustering analyzes the lengths of edges in $DD(P)$ with a judging range (in the form of a statistical test) that combines $\text{Local_Mean}(p_i)$ and $\text{Mean_St_Dev}(P)$. We investigate the distribution of $\text{Local_Mean}(p_i)$, and compare and contrast what proportion lies around a ball of radius $\text{Mean_St_Dev}(P)$ centered on the mean of $\text{Local_Mean}(p_i)$. We show that, across different settings, qualitatively the distribution has the same structure. We instantiate two datasets, one of which is CSR (Complete

Spatial Randomness) [28] and the other exhibiting well-identifiable concentrations. These validation sets have the same number of points and scale (the same study region). These are shown in Figure 4-5.

Delaunay diagrams minimize skinny elongated triangles and maximize equilateral triangles. In CSR, the proximity to equilateral triangles is better since no point is particularly far away from others. Thus, the distribution of $Local_St_Dev(p_i)$ values exhibits a concentration around $Mean_St_Dev(P)$ as depicted in Figure 4-5(c). Most $Local_St_Dev(p_i)$ values vary within a range of radius $Mean_St_Dev(P)$. In addition, Figure 4-5(d) shows the tendency of $Local_Mean(p_i)$ values as a histogram that displays a concentration around their observed mean. More than four-fifths of the $Local_Mean(p_i)$ values lie within $Mean_St_Dev(P)$ of their mean. Thus, most points in R exhibit similar local strengths that concentrate around the mean. The distributions are close to a bell-shaped distribution (only one mode). Frequency tends to decrease rapidly as $Local_Mean(p_i)$ goes away from the mean.

Border points exhibit great variability in datasets that exhibit clusters (for example, see Figure 4-5(e)). These border points boost $Mean_St_Dev(P)$. Thus, the concentration of small $Local_St_Dev(p_i)$ values of inner points is moved and skewed to the left. This is also the case in the distribution of $Local_Mean(p_i)$ values as depicted in Figure 4-5(h). Note that, the presence of noise points will not dramatically change the distribution, but will remain similar. Although the distribution of $Local_Mean(p_i)$ values is skewed, we have a similar number of $Local_Mean(p_i)$ values within a ball of radius $Mean_St_Dev(P)$ of their mean. This is illustrated for a dataset exhibiting clusters in Figure 4-5(h). Thus, considering a radius equal to $Mean_St_Dev(P)$ of the mean of $Local_Mean(p_i)$ includes most homogeneous $Local_Mean(p_i)$ values regardless of shapes of distributions in Delaunay diagrams. Due to the maximization of proximity to equilateral triangles of Delaunay diagrams, it is expected that a radius equal to $Mean_St_Dev(P)$ centered at the mean of $Local_Mean(p_i)$ also includes most homogeneous lengths of $N(p_i)$. Thus, the use of 1 as the control value (m) in the tolerance value ($m \times Mean_St_Dev(P)$) is desirable to preserve globally homogeneous

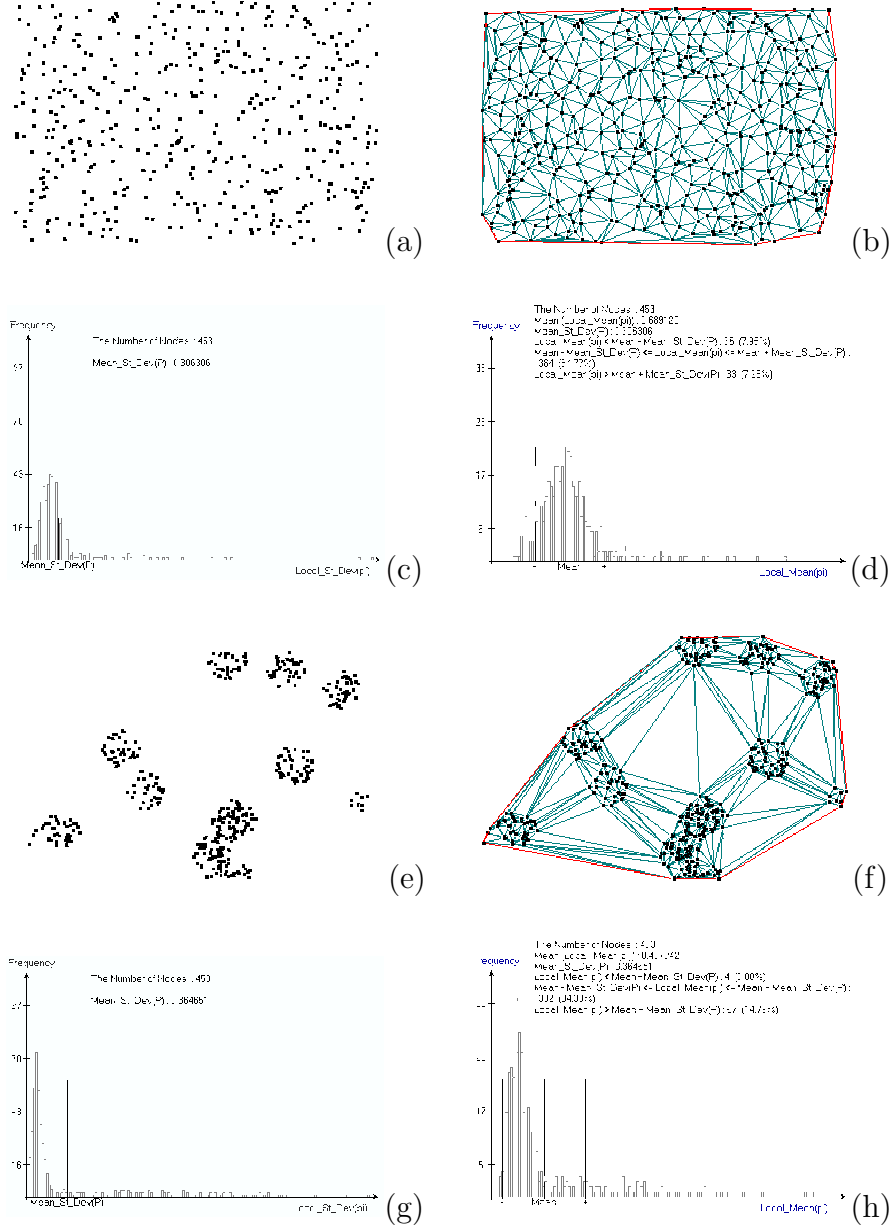


Figure 4-5: Distributions (as histograms) of lengths of Delaunay edges for CSR and for datasets exhibiting clusters: (a) CSR dataset ($\|P\| = 453$); (b) $DD(P)$; (c) The distribution of $Local_St_Dev(p_i)$ values in $DD(P)$; (d) The distribution of $Local_Mean(p_i)$ values in $DD(P)$; (e) Dataset with clusters ($\|P'\| = 453$); (f) $DD(P')$; (g) The distribution of $Local_St_Dev(p_j)$ values in $DD(P')$; (h) The distribution of $Local_Mean(p_j)$ values in $DD(P')$.

lengths of edges and remove heterogeneous lengths of edges.

4.3.2 Determination of the control value m

Our argumentation above should convince that a control value of $m = 1$ is sufficient to obtain robustly clusters using the Short-Long analysis of edges. But, some users may wish to explore profiles of $Local_Mean(p_i)$ values and $Local_St_Dev(p_i)$ values, and their relative spread to the value of $Mean_St_Dev(P)$. Comparing these values also provides some ground for the setting an initial value for m . The tolerance value defines an acceptable range for homogeneous edges. Thus, if there exists a clear distinction between inner points and border points, then the distinction will provide a robust indicator for the acceptable range.

Consider for a moment the premise of partition-based clustering on proximity graphs that there are two classes of edges, namely, intra-cluster edges that are short and inter-cluster edges that are long. Under this premise, sorting the edges by their lengths and plotting the lengths in increasing order would result in a profile like Figure 4-6.

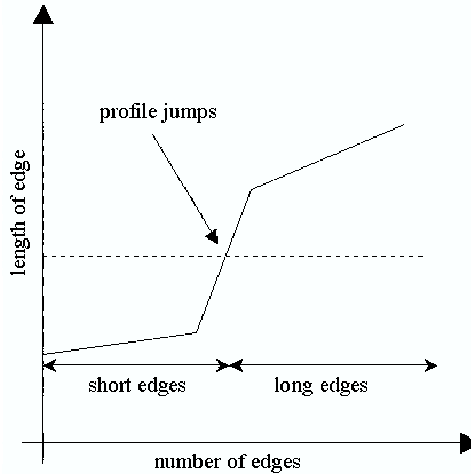


Figure 4-6: Expected profile of edge lengths in proximity graphs.

Namely, there would be a point at which the lengths of edges jump because the intra-

cluster edges are finished and now lengths of inter-cluster edges show up. This global static model clustering approach would require only to identify this profile jump to achieve its goal. It then would find the dotted horizontal line in Figure 4-6 and would remove all edges longer than this threshold. However, we have argued in Chapter 2 and Chapter 3 that a dynamic model and global clustering approach is needed. We have also argued in Section 4.1.3 that an approach based on the regional dispersion around a point p_i is more informative than the mere lengths of edges that ignore such spatial dispersion. Thus, we visualize our point-centric statistics in a similar plot to Figure 4-6. The control value m regulates the position of the dotted line as a factor that scales $Mean_St_Dev(P)$. However, the values that are displayed in sorted order are $Local_Mean(p_i)$ values and their corresponding $Local_St_Dev(p_i)$ values for confirmation. Again, a jump in the profile (a sudden increase on the values) is what indicates the classification between unusually short edges and long edges.

We illustrate this with Figure 4-7(a) and Figure 4-7(b). Figure 4-7(a) illustrates typ-

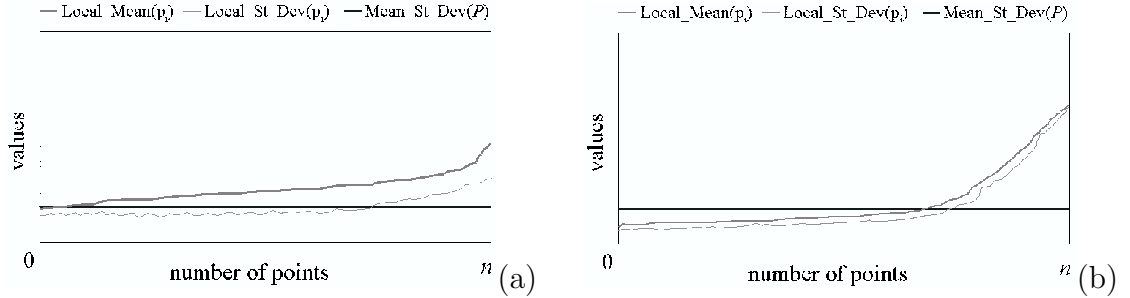


Figure 4-7: Expected profiles of point-centric statistics with CSR datasets and datasets exhibiting clusters: (a) Expected profiles of point-centric statistics with CSR datasets; (b) Expected profiles of point-centric statistics with datasets exhibiting clusters without noise.

ical profiles of point-centric statistics with CSR datasets while Figure 4-7(b) displays typical profiles of point-centric statistics with datasets exhibiting clusters without noise points. Figure 4-7(a) and Figure 4-7(b) plot all of $Local_Mean(p_i)$ values and $Local_St_Dev(p_i)$ values and compare these values to the value of $Mean_St_Dev(P)$. Here, the values of $Local_Mean(p_i)$ are sorted in increasing order. The value of

$Mean_St_Dev(P)$ appears as a horizontal line, since this is a global statistic. Figure 4-7(a) and Figure 4-7(b) depict that $Local_St_Dev(p_i)$ fluctuates but grows as $Local_Mean(p_i)$ increases. This is natural, because the same neighborhood of points at smaller or larger scale will scale $Local_St_Dev(p_i)$ and $Local_Mean(p_i)$. Thus, greater $Local_Mean(p_i)$ is more likely to have greater $Local_St_Dev(p_i)$.

For CSR datasets, typical profiles of $Local_Mean(p_i)$ values and $Local_St_Dev(p_i)$ values are likely to display constant slope (no sudden jump in the profile) as shown in Figure 4-7(a). This is because no particular sets of points are far away from others, thus most points exhibit similar point-centric statistics. In particular, it is expected that most points exhibit small values of $Local_St_Dev(p_i)$ due to this homogeneity. It is also expected that several border points¹ exhibit relatively large $Local_Mean(p_i)$ values and $Local_St_Dev(p_i)$ values. This is because a border point p_i may have both relatively short Delaunay edges (connecting p_i to inner points) and relatively long edges (connecting p_i to another border points) incident to them. However, a distinction between inner points and border points in CSR datasets is not expected to exist in the same sense as it will be in datasets exhibiting clusters. There is no distinction between inner points and border points in CSR datasets. Because $Mean_St_Dev(P)$ is the observed mean of $Local_St_Dev(p_i)$, it should be central with respect to the $Local_St_Dev(p_i)$. The observed mean is an estimator of location or central tendency. This is what we observe in Figure 4-7(a). We also observe no sudden jump in the profile.

For datasets exhibiting clusters without noise, both $Local_Mean(p_i)$ values and $Local_St_Dev(p_i)$ values are expected to raise rapidly after being essentially flat (refer to Figure 4-7(b)). This reflects the discrepancy in inner points and border points. Here, the value of $Mean_St_Dev(P)$ is expected to cut through where the values of $Local_St_Dev(p_i)$ sharply rise and is expected to provide a distinction between inner points and border points. Thus, the value of $Mean_St_Dev(P)$ robustly behaves as the tolerance value. It is also expected to be the case in datasets exhibit-

¹For a CSR dataset P encapsulated in R , border points are likely to be points lying on the convex hull of P since all points in P tend to belong to a single cluster.

ing intermediate distributions (datasets exhibiting clusters with noise) between CSR datasets and datasets exhibiting clusters without noise. This is because the value of $Mean_St_Dev(P)$ averages $Local_St_Dev(p_i)$ values and thus it is expected to be smaller than $Local_St_Dev(p_i)$ values of inner points and larger than $Local_St_Dev(p_i)$ values of border points. Thus, in intermediate distributions, the value of $Mean_St_Dev(P)$ is expected to robustly scale Equation 4.3 and expected to provide a solid initial value for the tolerance value. Consequently, $m = 1$ is used as a default value in Short-Long clustering in the Euclidean distance, unless otherwise noted.

Now, we examine this discussion with datasets shown in Figure 4-5. Figure 4-8(a)

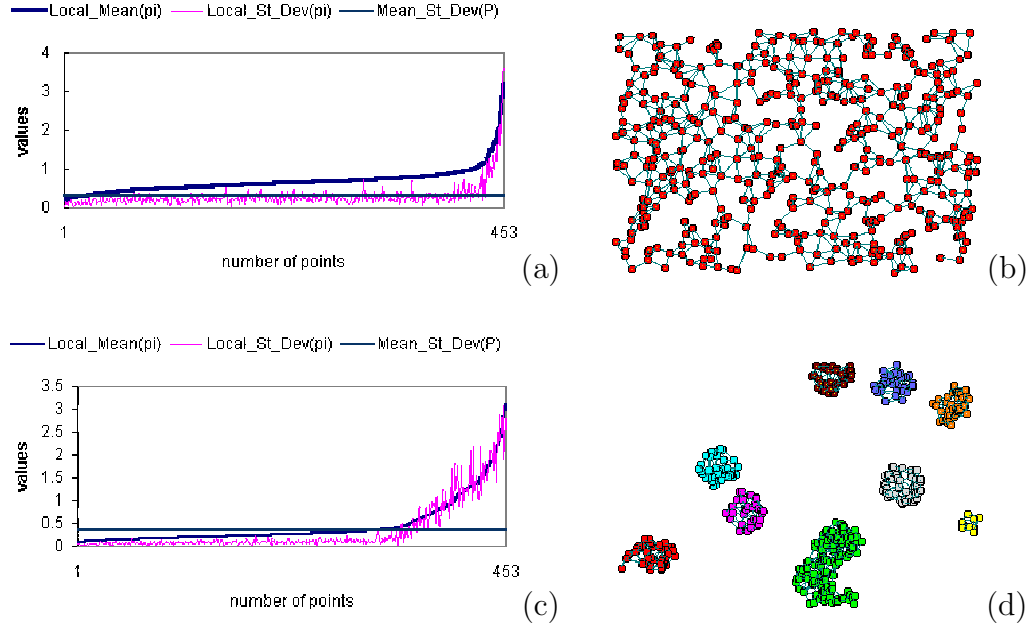


Figure 4-8: Profiles of point-centric statistics for Delaunay diagrams in the Euclidean metric with the CSR dataset and the dataset exhibiting clusters shown in Figure 4-5: (a) Profiles of point-centric statistics with the CSR dataset shown in Figure 4-5(a); (b) Clustering result of the CSR dataset (1 cluster); (c) Profiles of point-centric statistics with the dataset exhibiting clusters shown in Figure 4-5(e); (d) Clustering result of the dataset exhibiting clusters.

and Figure 4-8(c) plot profiles of point-centric statistics for the CSR dataset shown in

Figure 4-5(a) and for the dataset with clusters shown in Figure 4-5(e), respectively. In both Figure 4-8(a) and Figure 4-8(c), the value of $Mean_St_Dev(P)$ is almost greater than the smallest value of $Local_St_Dev(p_i)$ for inner points and is less than the largest value of $Local_St_Dev(p_i)$ for border points. Thus, $Mean_St_Dev(P)$ robustly behaves as the tolerance value in these examples. Short-Long clustering puts all the points into the same cluster and suggests no particular aggregation for the CSR dataset with $m = 1$ as shown in Figure 4-8(b). On the other hand, with $m = 1$, Short-Long clustering reports 9 clusters for the dataset exhibiting clusters (see Figure 4-8(d)). These clustering results demonstrate the robustness of the default control value.

4.3.3 The control value m as an exploratory tool

The control value m serves as an exploratory tool. Larger values of m widen the acceptable range of *Other_Edges*, thus they reduce the number of *Short_Edges* and *Long_Edges*. This decreases the number of edges removed in Phase *I* and Phase *III*. This eventually causes relatively close clusters to merge into the same cluster. The user may increase the value of m to explore which clusters are growing fast in terms of their sizes or which clusters are going to merge. Conversely, smaller values of m result in clusters with more homogeneous lengths of edges. Clusters tend to be broken into smaller subclusters unless edge lengths are homogeneous enough. Eventually, the use of smaller values m identifies relatively high-density clusters and homogeneous clusters, but declares relatively sparse clusters as noise and heterogeneous clusters as potential outliers. The user may reduce the value of m to find breakable or vulnerable regions in clusters where they are about to split.

Recall that Chapter 3 offered 3 challenging datasets in Figure 3-3, Figure 3-4 and Figure 3-5. With the control value $m = 1$, we obtain the best results. Namely, Short-Long clustering reports 4 clusters for the dataset in Figure 3-3(a), identifies 3 clusters for the dataset in Figure 3-4(a) and detects 5 clusters for the dataset in Figure 3-5(a). These clustering results are shown in Figure 3-3(e), Figure 3-4(d) and

Figure 3-5(d), respectively. Short-Long clustering leaves sparse clusters undetected with the use of smaller values of m than 1. It suggests 3 high-density aggregations for the dataset in Figure 3-3(a) and 4 concentrations for the dataset in Figure 3-5(a). These clustering results are shown in Figure 3-3(b) and Figure 3-5(b), respectively. Similarly, Figure 3-3(c) and Figure 3-5(c) depict clustering results with the use of larger values of m than 1. Thus, Short-Long clustering not only can reproduce the same groupings suggested by traditional peak-inference clustering approaches, but promises more exploratory capability for identifying cluster merging and discovering potential splitting risk.

4.4 Time complexity analysis

Along with effectiveness, efficiency is an important factor for clustering in data-rich environments. Subquadratic time complexity is essential for clustering algorithms in GDM. The Short-Long criterion is applicable to clustering of very large geo-referenced data since it requires $O(n \log n)$ time.

4.4.1 Short-Long clustering only requires $O(n \log n)$ time

Short-Long clustering requires $O(n \log n)$ time to construct the Delaunay diagram. Once this model of proximity is available, Short-Long clustering performs the remaining work in linear time. The computation of $Local_Mean(p_i)$ and $Local_St_Dev(p_i)$ takes constant time, since retrieving incident edges for p_i requires $O(1)$ time (a constant number of operations). Recall that in 2D, the Delaunay diagram has $\Theta(n)$ edges. Now, the calculation of $Mean_St_Dev(P)$ is bounded by twice the number of edges and, therefore, the edge elimination in Phase $\mathcal{I}$ requires $O(n)$ time. To compute connected components only requires $O(n)$ time, since this is linear in the sum of both number of edges and number of points. Each test to restore $Short_Edges(p_i)$ in Phase $\mathcal{II}$ is bounded by $d(p_i)$ and thus, the total time for Phase $\mathcal{II}$ is again bounded

by a constant time; the number of edges (also linear). Similarly, the local subgraph around p_i in Phase *III* has size $O(d(p_i)^2)$. Because in the Delaunay diagram we have $E[d(p_i)] = 6$, the total execution in this phase requires time proportional to the sum of the degrees of all vertices, which is $O(n)$. Consequently, the time complexity of Short-Long clustering is dominated by computing the Delaunay diagram.

4.4.2 CPU time comparison

While Short-Long clustering requires $O(n \log n)$ time, we are interested in verifying experimentally that the constants under the O notation are small. This is to be expected, since the dominating $O(n \log n)$ time in Short-Long clustering is the computation of the Delaunay diagram. Everything else requires $O(n)$ time.

We compare Short-Long clustering with AMOEBA [36] and DBSCAN [33]. Both AMOEBA and DBSCAN also require $O(n \log n)$ time and have been shown to be very efficient, (small constant under the O notation). We generated 7 types of datasets based on a simple mixture model. The number of points varies from 1,000 to 100,000. Each dataset has 10 randomly distributed cluster centers within the unit square $[0, 1] \times [0, 1]$ and 10 percent of the total number of points are just noise points (selected at random within the unit square). Since Short-Long clustering, AMOEBA and DBSCAN require spatial proximity modeling (Delaunay diagram for the first two and R*-tree for DBSCAN), the run time comparison shown in Figure 4-9 includes the time for constructing their respective data structures. All the experiments were performed on a 550MHz Pentium *III* workstation with 128MB memory.

Figure 4-9 demonstrates that Short-Long clustering outperforms AMOEBA with respect to CPU-time by a small margin. Not surprisingly, slight changes of the parameter value (Eps) in DBSCAN results in totally different clustering results and different CPU-time consumption. DBSCAN outperforms Short-Long clustering with particular parameter values ($Eps = 0.01$ and $MinPts = 4$). However, these values are set to maximize efficiency and do not guarantee reasonable clustering. DBSCAN requires

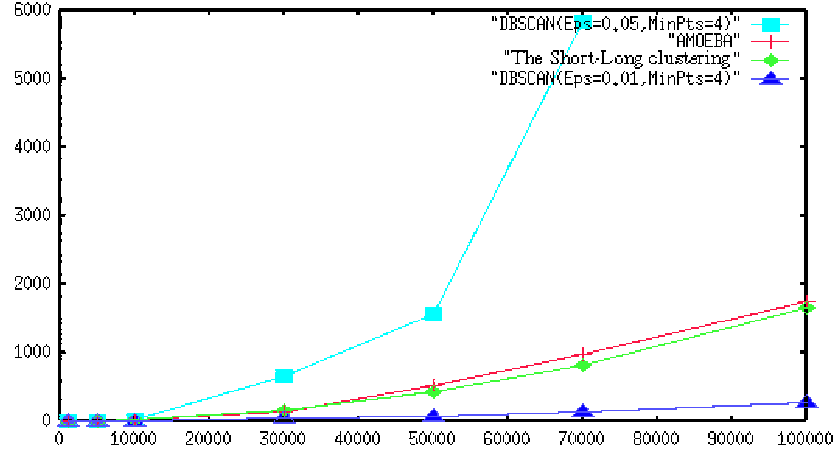


Figure 4-9: Comparison of CPU times (measurements in seconds).

costly parameter-tuning to produce its best clustering (it still fails to find clusters in situations like those in Section 3.2). Note that, with parameter values $Eps = 0.05$ and $MinPts = 4$ (as suggested by authors, $MinPts$ is fixed to 4) DBSCAN's time consumption increases rapidly as the size of data grows. Also, incorrect parameters for DBSCAN (to large Eps) can make the time requirements quadratic. Short-Long clustering is argument free. Thus, Short-Long clustering will easily outperform the use of AMOEBA, DBSCAN or other clustering methods that need parameter-tuning. The experimental results shown in Figure 4-9 include the time requirements for I/O and the clustering process. These experimental results illustrate that Short-Long clustering is efficient in terms of time complexity and scalable to large datasets.

4.5 Performance evaluation

This section presents results of experiments with synthetic datasets and real datasets that illustrate remarkably the robustness of Short-Long clustering. For the datasets used in this thesis, we report only clusters with size greater than 1% of the total number of points, unless otherwise indicated.

4.5.1 Clustering quality with stochastic datasets

Three datasets shown in Figure 4-10 are generated within a rectangle $[0, 15] \times [0, 10]$ under CSR. Since the data is filling the space with random points, we expect one single cluster. The intended (generation) intensity (λ) for dataset $\mathcal{I}$, $\mathcal{II}$ and $\mathcal{III}$ is 1.0, 5.0 and 10.0 while the estimated (sample) λ is 1.07, 5.02 and 9.65, respectively.

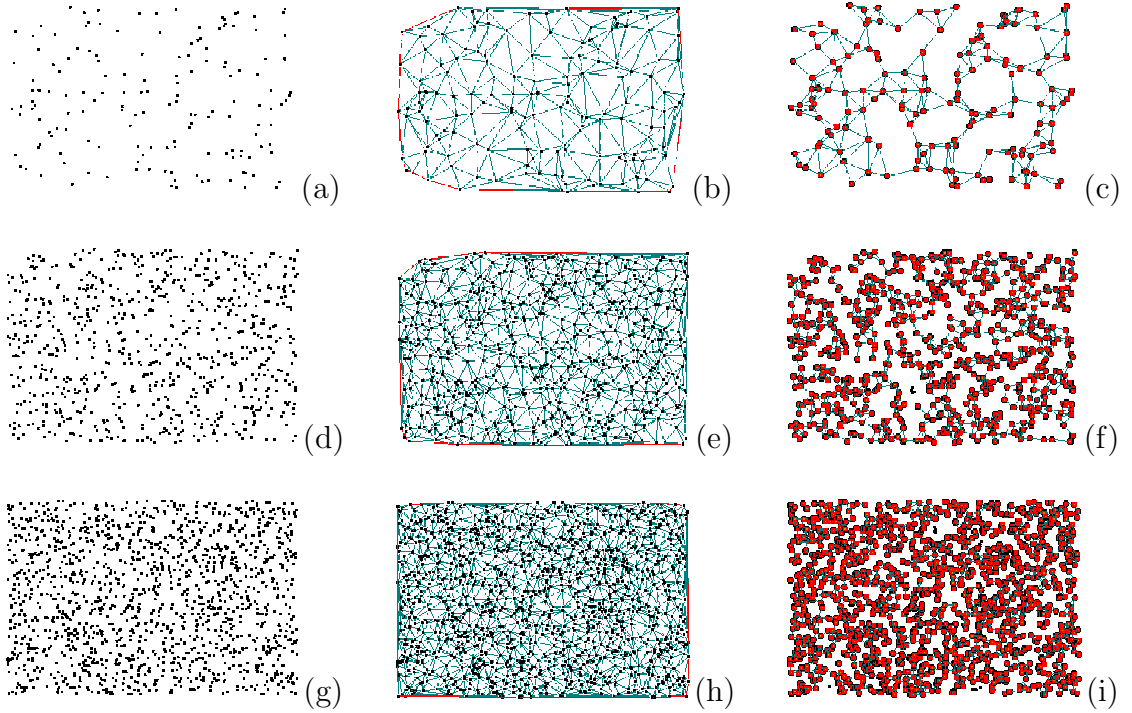


Figure 4-10: Clustering results of stochastic datasets: (a) CSR dataset $\mathcal{I}$ ($\|P\| = 160$); (b) Delaunay diagram of CSR dataset $\mathcal{I}$; (c) Clustering of CSR dataset $\mathcal{I}$; (d) CSR dataset $\mathcal{II}$ ($\|P\| = 753$); (e) Delaunay diagram of CSR dataset $\mathcal{II}$; (f) Clustering of CSR dataset $\mathcal{II}$; (g) CSR dataset $\mathcal{III}$ ($\|P\| = 1447$); (h) Delaunay diagram of CSR dataset $\mathcal{III}$; (i) Clustering of CSR dataset $\mathcal{III}$.

The results of applying our clustering method to these datasets are as follows. All the points in CSR dataset $\mathcal{I}$ are placed in the same cluster as illustrated in Figure 4-10(c). A big cluster constitutes more than 98% of points in the CSR dataset $\mathcal{II}$ while 99% in

the CSR dataset *III* are also placed in a single group. Short-Long clustering reports neither spatial concentrations nor localized excesses in the CSR datasets. Thus, the default control value robustly works for stochastic datasets.

4.5.2 Clustering quality with synthetic datasets

The data shown in Figure 4-11(a) consists of 12 clusters of various shapes and different densities. Note that, many are not convex. Short-Long clustering successfully detects all of them as depicted in Figure 4-11(b). The data in Figure 4-11(c) is the result of synthetic data generation aimed at creating situations where peak-inference clustering methods typically fail. Section 3.2 explains these types of situations. The clustering obtained with the Short-Long clustering approach demonstrates its robustness. Short-Long clustering correctly identifies two clusters connected by multiple chains in the top-left corner. Short-Long clustering has no difficulty in detecting a cross-shaped, but sparse cluster around a high-density cluster at the top-right corner of the dataset. Again, this mimics real world cases like densely populated city surrounded by slightly less dense suburbs. Short-Long clustering successfully reports the two high-density clusters separated by a twisting narrow gap (at the bottom-left of the dataset). This mimics real world situations where a river separates two highly populated areas. Short-Long clustering correctly reports the ring-shaped cluster and the surrounded island-shaped cluster at the bottom-right of that dataset. Figure 4-11(d) shows clusters detected by Short-Long clustering.

The datasets in Figure 4-11(e) and Figure 4-11(g) belong to CHAMELEON's benchmark [76] for which most other methods fail to report all concentrations. Regardless of the challenge posed by different sizes, different densities, and different separations (inter-cluster distances), Short-Long clustering correctly identifies 8 clusters and 6 clusters as shown in Figure 4-11(f) and Figure 4-11(g), respectively. Some of these clusters have very little space between them, but Short-Long clustering still detects these boundaries.

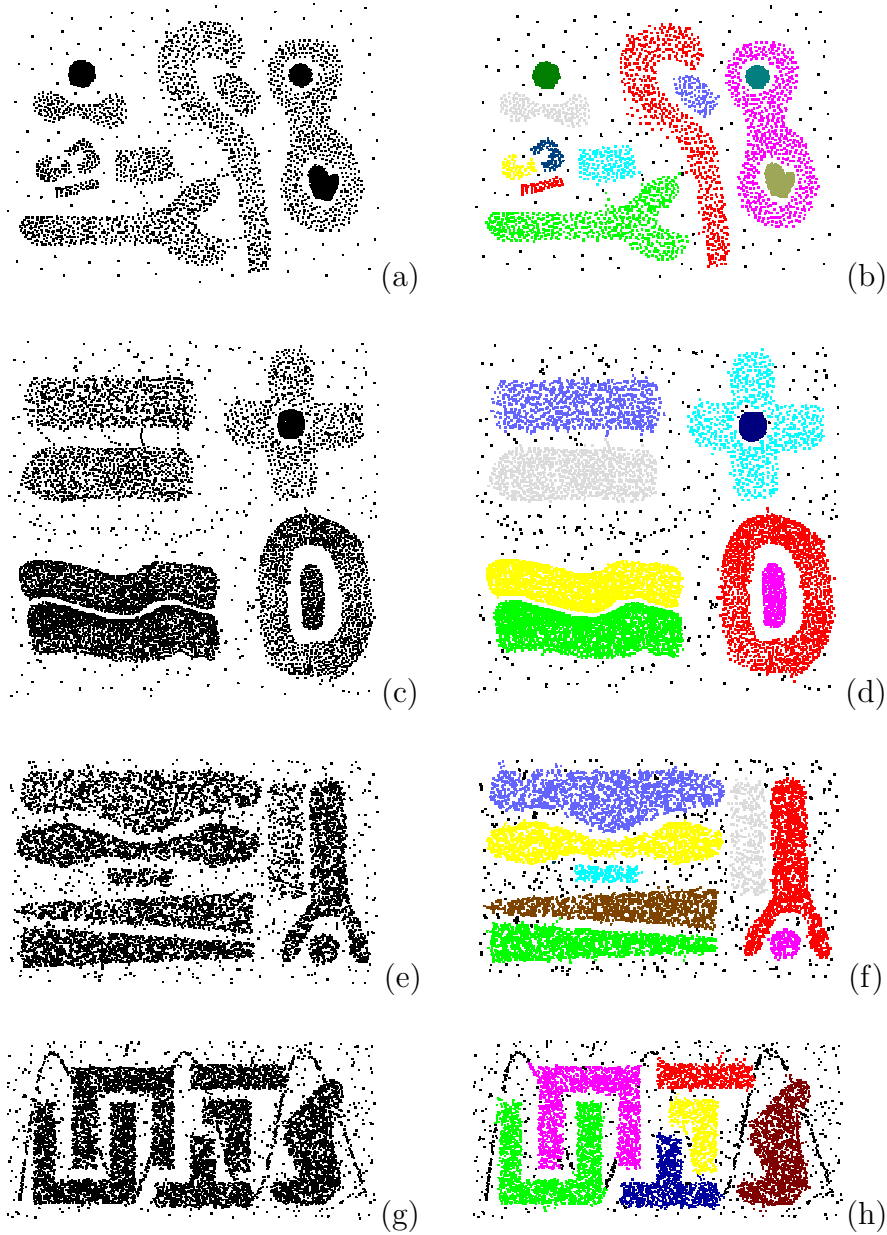


Figure 4-11: Clustering results of synthetic datasets: (a) Synthetic dataset $\mathcal{I}$ ($\|P\| = 3250$ and exhibiting 12 clusters); (b) Clustering of synthetic dataset $\mathcal{I}$ (results in 12 clusters); (c) Synthetic dataset $\mathcal{II}$ ($\|P\| = 8000$ exhibiting 8 clusters); (d) Clustering of synthetic dataset $\mathcal{II}$ (results in 8 clusters); (e) Synthetic dataset $\mathcal{III}$ ($\|P\| = 8000$ exhibiting 8 clusters); (f) Clustering of synthetic dataset $\mathcal{III}$ (results in 8 clusters); (g) Synthetic dataset $\mathcal{IV}$ ($\|P\| = 8000$ and exhibits 6 clusters); (h) Clustering of synthetic dataset $\mathcal{IV}$ (results in 6 clusters).

Although these 4 datasets exhibit different distributions, the default control value successfully identifies globally significant cluster boundaries and suggests high level spatial concentrations. This means that the control value is resistant to types of distributions. This is expected since the default control value is not set by the user as it is the case in traditional clustering approaches, but learned from data. Data determines the point-centric statistics ($Local_Mean(p_i)$, $Local_Mean(p_i)$ and $Mean_St_Dev(P)$) and the point-centric statistics determines the default control value.

4.5.3 Clustering quality with real datasets

The datasets displayed in Figure 4-12 are real datasets of Queensland in Australia. Figure 4-12(a) displays centers for living areas in Queensland and the Short-Long clustering results. In the bottom-right of the dataset, Brisbane, the capital city of Queensland, creates one large cluster (incorporating 70 percent of the living areas). Short-Long clustering detects this and simultaneously, it finds three relatively smaller clusters along the coast which correspond to big cities such as Cairns, Atherton and Townsville. Inland areas have no clusters. This highlights the pattern of urban settlement in coastal areas and around capital cities.

The dataset displayed in Figure 4-12(b) represents landing grounds including aerodromes and airports. This dataset does not seem to have any significant cluster. Landing grounds seem to be scattered all over Queensland. Short-Long clustering groups 90 percent of them into a large cluster that covers almost all Queensland. Short-Long clustering also detects two small clusters in the southern portion of Brisbane. Note that, visual inspection reveals that there is less density for landing grounds surrounding the concentrated living areas near Brisbane. Creating the two other clusters South and East of Brisbane. Densely populated urban areas are not likely to have enough space for many airports. Conversely, sparsely populated inland areas are likely to have enough space for small aerodromes.

Figure 4-12(c) illustrates a dataset for national parks in Queensland. National parks

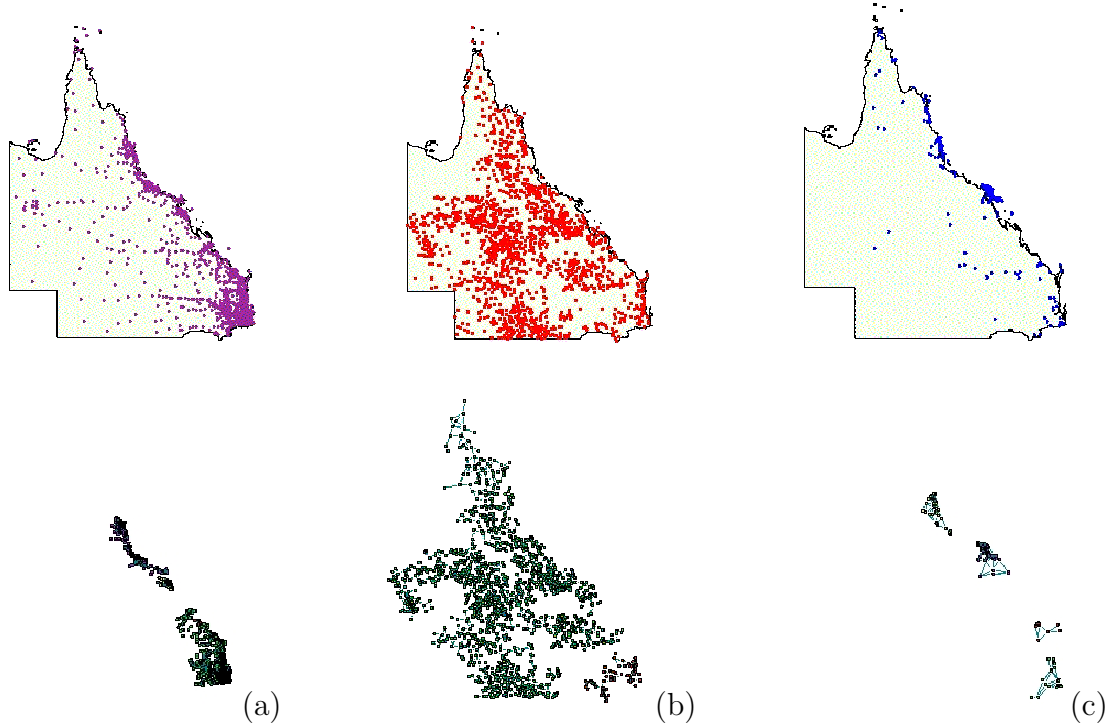


Figure 4-12: Clustering results of real datasets: (a) Living areas ($\|P\| = 1837$) and Short-Long clustering grouping into 4 clusters; (b) Landing grounds ($\|P\| = 1579$) and Short-Long clustering grouping into 3 clusters; (c) National parks ($\|P\| = 227$) and Short-Long clustering grouping into 4 clusters.

are generally located along the coast and the *Great Barrier Reef*. Short-Long clustering successfully derives clusters directly overlying the major cities, urban areas and coastal areas. It finds two high-density clusters including more than half of national parks on the outer margin of Australia's continental shelf. Two less dense clusters are detected around urban areas.

4.6 Remarks

This chapter provides the algorithmic schema for Short-Long clustering and discusses the efficiency and effectiveness of our approach with a series of performance evalu-

ations. Unlike classification, clustering is an example of unsupervised learning that does not rely on user-supplied predefined information [61]. We show that the automatic nature of Short-Long clustering minimizes constraints and bias posed by the user and suggests many possible high-quality clusters regardless of types of distributions. Short-Long clustering incorporates the special characteristics of spatial data discussed in Section 3.1, thus it is able to identify various spatial aggregations. In particular, we can discover heterogeneous sizes, different densities or arbitrary shapes, in the absence or presence of noise. Further, Short-Long clustering makes clear distinction between border points and chain points, thus it is able to detect clusters linked by multiple chains. In addition, this approach is insensitive to the order of input, thus it generates consistent results. We also demonstrate the efficiency of our approach and scalability to large datasets.

Chapter 5

Short-Long Clustering with Various Structural Models

Adjacency information explicitly defines *is_nearby_neighbor* relations in a topological space. This topological information forms a structural model used for graph-based clustering. Once relations are defined, they are preserved by topological transformations (homeomorphism) of the space [148]. However, there are several ways to define adjacency information and thus to build a structural model. The previous chapter discusses Short-Long clustering with a structural model defined by the Delaunay diagram in the Euclidean metric. This structural model assumes that there is a straight path between two neighboring points. It would be limiting if the user was forced to use this structural model only. In some spatial settings, adjacency information based on the Euclidean distance is neither suitable nor applicable.

This chapter investigates extensions of Short-Long clustering to various structural models. The extensions make Short-Long clustering exploratory and suitable for “what-if” analysis that enables the user to quickly explore clustering in different settings. Section 5.1 introduces structural models where interactions between neighbors (Delaunay edges) are obstructed by obstacles. It presents Short-Long clustering in

the presence of obstacles (referred here as obstacle Short-Long clustering). Different metric information results in different structural models. Section 5.2 extends the Euclidean metric case of the previous chapter to all Minkowski metrics. In Section 5.3, we compare and contrast Short-Long clustering with various proximity graphs that encode different adjacency information.

5.1 In the presence of obstacles

Conventional clustering methods use the Euclidean distance to measure spatial proximity and thus to define a structural model. This assumes that there exists a straight path between two nearby points in P . However, this assumption is unacceptable in applications where obstacles (rivers, lakes, political boundaries or mountains) prevent traversing the straight path between the two points. Here, obstacles are not traversable. Thus, clustering in the presence of obstacles requires a new structural model that considers distance through the shortest path avoiding obstacles and thus defines informative adjacency resulting from obstacles.

5.1.1 Obstacles emerge in many spatial settings

This subsection provides two examples where clustering in the presence of obstacles emerge and motivates the need of incorporating obstacles.

Association analysis when obstacles exist

Spatial clustering is the starting point of many exploratory analyses including spatial-dominant generalization as discussed in Chapter 1. Here, clustering is to find point concentrations in a particular layer towards association analysis among layers. Thus, spatial clustering must ensure high-quality clusters, otherwise spatial associations

among layers remain undetected. A simple synthetic example in Figure 5-1 illustrates this point.

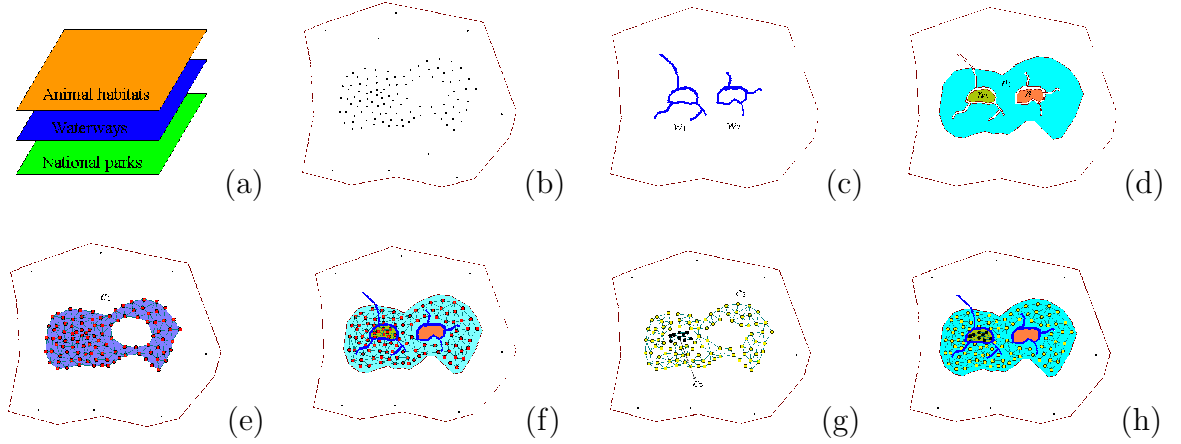


Figure 5-1: Impact on clustering results when incorporating obstacles: (a) Three geographical layers; (b) Animal habitats ($\|P\| = 113$); (c) Obstacles (two waterways across national parks); (d) Three national parks; (e) Clustering result of the “Animal habitats” layer ignoring obstacles; (f) Overlay of the “National parks” layer and the “Waterways” layer with clustering results ignoring obstacles; (g) Clustering result accounting for obstacles; (h) Overlay of the “National parks” layer with clustering result accounting for obstacles.

Figure 5-1 illustrates the enhanced cluster reasoning and interpretability by accounting for an obstacle layer. Figure 5-1(a) shows three geographical layers: “Animal habitats”, “Waterways” and “National parks”. Figure 5-1(b) shows point data in the “Animal habitats” layer. Figure 5-1(c) depicts data in the layer labeled “Waterways” that consists of major rivers and streams. Figure 5-1(d) shows data in the “National parks” layer consisting of three areas administered as different national parks. Two of them (island-like national parks indicated as n_2 and n_3) are surrounded by the waterways. Spatial-dominant generalization of the “Animal habitats” layer without accounting for obstacles would proceed as follows.

1. Cluster the “Animal habitats” layer.

2. Overlay clustering result with the “Waterways” layer to check for association.
3. Overlay clustering result with the “National parks” to check for association.
4. Overlay clustering result with both the “Waterways” and the “National parks” layers to check association.

Without considering obstacles, the clustering of “Animal habitats” results in a large cluster c_1 with only one hole (refer to Figure 5-1(e)). Then, the overlays with other layers (Figure 5-1(f)) do not find a close enough match. It is very unlikely that any interesting association will be found. That is, neither of the national parks explains the cluster (note that n_3 explains only the boundary of the hole, but not the outside boundary). As polygonized areas, each national park is significantly different from the cluster c_1 and an autonomous system (agent, algorithm or miner) will not flash out this association.

Alternatively, the “Waterways” layer can be incorporated as obstacles.

1. Cluster the “Animal habitats” layer accounting for the “Waterways” layer as an obstacle layer.
2. Overlay clustering result with the “National parks” layer to check for association.

If the waterways are incorporated into the analysis as crisp obstacles, then the clustering produces two groups c_2 and c_3 as illustrated in Figure 5-1(g). The autonomous system will find significant matches between the polygonized area of cluster c_2 and national park n_1 and also between cluster c_3 and national park n_2 . These associations will be suggested by the system and could form the bases of further EDA or CDA. This oversimplified example suggests that clustering in the presence of obstacles contributes to find the high level description of clusters and provides a new avenue of exploration, namely “why are no animal habitats found in the national park n_3 ”

given that it is also surrounded by major waterways as n_2 , but n_2 does exhibit a concentration of habitats?”.

Facility location in the presence of obstacles

The second illustration of the need to incorporate obstacles in the clustering process is in the context of facility location. Typically, k -partitioning clustering algorithms solve an optimization problem that is equivalent to a facility location problem [44]. For example, when medians are used as representatives in k -partitioning clustering, clusters are encoded by the selection of representatives as each point belongs to the cluster of its nearest representative. In this case, clusters minimize the absolute error between a point and its representative or equivalently the expected distance to a facility (as facilities are placed at the representatives).

Existing k -partitioning algorithms are most useful when the number of facilities (the facility location literature uses p for the number of representatives, but $p = k$ in k -partitioning clustering) is known in advance. Searching for p (or k) multiplies the complexity for the potential values of p and favors larger p since $p = n$ is an absolute minimum. However, EDA may bring a pleasant balance between the cost of the number of facilities with the actual cost of the location solution. In other words, the number of facilities may be decided after careful investigation of the data.

Consider the points in Figure 5-2(a) representing locations of houses and a scenario where local government is planning to locate public mail boxes. An analysis will aim at maximizing convenience and accessibility while optimizing the number of mail boxes. In this region, a river runs across. Algorithms that do not take this into consideration (for example, plain Short-Long clustering) suggests two clusters (Figure 5-2(b)). However, this is not a good choice as many people are forced to make long detours. On the other hand, 4 clusters in the region (Figure 5-2(c)) avoid long detours and represent spatial aggregations in the presence of obstacles. This simple example also emphasizes the importance of incorporating obstacles to clustering.

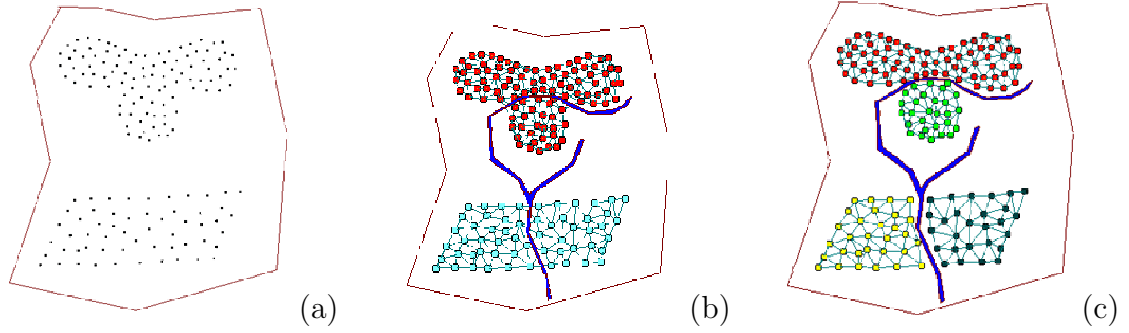


Figure 5-2: Locational optimization problem in the presence of obstacles when the number of clusters is unknown: (a) Houses ($\|P\| = 155$); (b) Clustering result in the absence of obstacles (2 clusters); (c) Clustering result in the presence of obstacles (4 clusters).

5.1.2 Detour distance and detour path

In a neighborhood defined by Delaunay diagrams, each Delaunay edge represents a relationship that indicates two end-points are relative neighbors. Strong interactions (short edges) indicate two end-points more likely belong to the same cluster while weak ones (long edges) imply end-points belong to distinct clusters. But, in the presence of obstacles, interactions and paths are blocked by obstacles. A Delaunay edge that intersects the defining lines of obstacles no longer suffices to link its end-points as members of the same cluster even if it is very short.

The criteria to evaluate now depend also on the length of edges on the *detour path* (and the transparency associated to the obstacle). We use *detour path* for the shortest path between two points in the Delaunay diagram where the edges on the path are Delaunay edges that avoid all obstacles. This is a path in a graph embedded in the plane. In contrast, *detour distance* is the length of the shortest path in the plane between two points.

Figure 5-3 illustrates the distinction. In Figure 5-3(a), thick solid lines represent the path whose length constitutes the detour distance. The detour path for the same

pair of points is highlighted in Figure 5-3(b). Dotted edges represent Delaunay edges that traverse an obstacle. Note that, the lines determining the detour distance are not edges in the Delaunay diagram. That is, the lines do not represent proper interactions between points in the absence of obstacles. The detour distance is approximated by the length of existing Delaunay edges (detour path), since the Delaunay diagram is a spanner as discussed in Section 4.1. Thus, two end-points of a Delaunay edge traversing some obstacles would not belong to the same cluster if Delaunay edges on a detour path for such Delaunay edge are long.

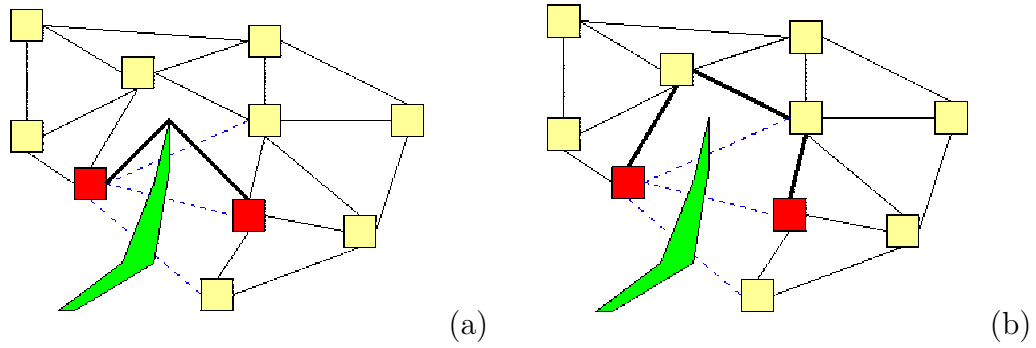


Figure 5-3: Detour distance and detour path: (a) Detour distance (thick solid lines); (b) Detour path (thick solid lines).

5.1.3 Algorithms of obstacle Short-Long clustering

Obstacle Short-Long clustering allows four policies to incorporate the effect of obstacles. The four proposed methods correspond to levels of transparency in the obstacles. The participation of obstacles into the analysis is implemented as degrees of participation (or removal) into the construction of the Delaunay diagram or into the computation of the two core statistics in the Short-Long clustering criteria (the local cohesion $Local_Mean(p_i)$ and the global degree of variation $Mean_St_Dev(P)$).

To show these degrees of inclusion of obstacles, we summarize obstacle Short-Long clustering in pseudo-code next.

Algorithm **Obstacle Short-Long clustering**

Input: A set P of points, a set B of obstacles and the control value m ;

Output: A graph $outG$ with intra-cluster edges;

1) **begin**

2) $\left\{ \begin{array}{l} \text{[v1]} \ G \leftarrow \text{BuildGeneralizedDelaunayDiagram}(P, B); \\ \text{[v2]} \ G \leftarrow \text{BuildDelaunayDiagram}(P); \end{array} \right.$

3) $Mean_St_Dev(P) \leftarrow \text{ComputeMeanStDev}(G)$;

4) **do**

5) $outG \leftarrow \text{ThreePhaseClustering}(G, m, Mean_St_Dev(P))$;

6) $\text{VisualizeClusteringResult}(outG)$;

7) $m \leftarrow$ a new control value;

8) **until** (the user stops)

9) **end**

The process labeled `ComputeMeanStDev()` (Step 3) calculates the global variation indicator $Mean_St_Dev(P)$. The process labeled `ThreePhaseClustering()` (Step 5) represents the three-phase clustering for Short-Long clustering discussed in Section 4.2.3, which requires the proximity graph, the control value and the global variation value. The process labeled `VisualizeClusteringResult(outG)` (Step 6) visualizes clustering results with the output graph ($outG$).

We have indicated that the second step has two versions with a version identifier. One or the other version (exclusively) in Step 2 is to be used. This will define a starting proximity graph. We will refer to Approach 1 as an instantiation of obstacle Short-Long clustering that uses Step 2.v1 to determine the initial proximity graph. Alternatively, we refer to Approach 2 as the use of Step 2.v2 to build the initial proximity graph. Now, the step labeled `BuildGeneralizedDelaunayDiagram()` returns a generalized Delaunay diagram that considers the obstacles and where distances are computed as detour distances to define Voronoi regions (the shortest-path Voronoi diagram). This point-to-area transformation step will be fully aware of the crisp obsta-

cles. However, the step labeled `BuildDelaunayDiagram()` computes an ordinary Delaunay diagram of the point data. This step alone simply ignores the obstacles. Thus, Approach 2 requires an obstacle handling process (referred as `ProcessObstacles()`). The routine `ProcessObstacles()` incorporates the obstacles into the template by modifying the Delaunay diagram with the removal of all Delaunay edges that traverse some obstacles. Instead, detour paths replace the Delaunay edges obstructed by the obstacles as discussed in Section 5.1.2. The `ProcessObstacles()` can be placed in one of three possibilities that differently define the two core statistics.

1. Between Step 2 and Step 3 (Approach 2a).
2. Between Step 3 and Step 4 (Approach 2b).
3. Between Step 5 and Step 6 (Approach 2c).

We proceed now to describe and discuss the four approaches that result from the proposed template.

1. Constrained Voronoi diagram based approach (Step 2.v1).

This approach defines a structural model with the consideration of obstacle in the proximity structure. For the construction of the generalized Voronoi diagram, the metric used evaluates the distance between two points p_i and p_j as the length of the shortest path in the plane between the two points that does not cut any obstacle (see [110] for details). Thus, topological information and geometrical information are determined by the location, size and every aspect of the obstacles (as well as the metric used to measure the length of segments of the path that does not touch obstacles). Thus, computing the proximity graph is more laborious. Although this approach constructs a Voronoi diagram with obstacles and the resulting generalized Voronoi diagram is typically different from the ordinary Voronoi diagram without obstacles, the resulting proximity graph may not be informative in some cases. In fact, when obstacles are small

with respect to Delaunay triangles, even if obstacles block straight lines between two points, the proximity graph may be the same.

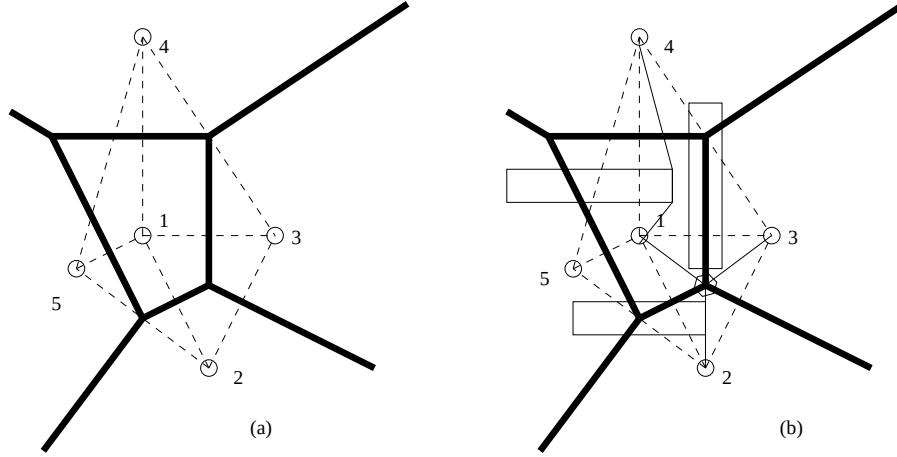


Figure 5-4: Voronoi regions change in the presence of obstacles, but neighborhood information may not change in the constrained Voronoi diagram ($\|P\| = 5$): (a) The Voronoi diagram and Delaunay diagram in the absence of obstacles; (b) The Voronoi diagram and Delaunay diagram with three rectangular obstacles.

We illustrate this in Figure 5-4. Figure 5-4(a) shows 5 data points. The Voronoi regions are indicated by dark lines, while the Delaunay diagram is shown in dotted lines. Figure 5-4(b) shows the same configuration but with 3 rectangular obstacles. We do not show the constrained Voronoi diagram but the edges of the proximity graph (the dotted lines) remain despite they intersect the obstacles. This is counterintuitive since p_1 is almost surrounded by the obstacles. However, the point equidistant to p_1 , p_2 and p_3 remains equidistant to these points despite the obstacles, and thus in the boundary of the constrained Voronoi regions. Similarly, there is still a path to get from p_1 to p_4 , and any position along this path has p_1 or p_4 as their nearest point. So, this path must go over the boundary of the constrained Voronoi regions for p_1 and p_4 . So, p_1 and p_4 remain as neighbors.

Thus, because the proximity graph derived from the constrained Voronoi di-

agram is ‘as close as possible’ to the ordinary Delaunay diagram [24, 110], omitting the procedure `ProcessObstacles()` is likely to be the application of `ThreePhaseClustering()` to the ordinary Delaunay diagram, which may look like plain Short-Long clustering. The additional complexity in computing the generalized Voronoi diagram may not pay off. In the algorithm above, this approach may be more effective with the inclusion of `ProcessObstacles()`, but in this latter case the approaches described below (Approach 2 and its variants) are faster for rather equivalent results.

This approach considers the neighboring information derived directly from the obstacles. This seems to incorporate the influence of obstacles into the proximity modeling. However, we see that users may be interested in exploring and clustering a target layer with respect to various settings. For instance, they may want to inspect a clustering of the target layer with rivers as obstacles, later with mountains as obstacles, and further, to test the clusters by overlaying waterways, rivers and lakes as obstacles. This very important exploratory spatial analysis would be computationally costly because the entire proximity graph is revised for each layer that might be used as obstacles. For each obstacle layer, we need to compute the constrained Voronoi diagram. Thus, this approach is not efficient for “what-if” analysis. It will not return results quickly enough for decision making and mining massive spatial databases. Another aspect that is of concern is that the statistic $Mean_St_Dev(P)$ is an indication of global variation in the dataset P and not on the obstacles. But, under this approach, it will vary significantly with the type of obstacle considered.

2. Edge removal based approach (Step 2.v2).

In this approach, we compute the ordinary Voronoi diagram without obstacles and derive the ordinary Delaunay diagram. And then, we remove Delaunay edges that intersect obstacles using the procedure `ProcessObstacles()`.

- (a) Computing $Mean_St_Dev(P)$ after processing obstacles.
(`ProcessObstacles()` between Step 2 and Step 3.)

This approach removes Delaunay edges first with the edge removal procedure `ProcessObstacles()` and then calculates the global variation indicator $Mean_St_Dev(P)$ after the edge removal. Thus, the obstacles influence the global variation indicator. Finally, we apply `ThreePhaseClustering()` to the modified Delaunay diagram.

Users shall consider some features of this alternative. One aspect is that, this method is not efficient for “what-if” analysis, although it incorporates an association between layers when computing $Mean_St_Dev(P)$. Recall that this is also the case with Approach 1. In addition, this alternative has several drawbacks. First, it is still slightly more expensive in terms of computational efficiency than the two approaches we describe next. This is because here, we must compute $Mean_St_Dev(P)$ for each new set of obstacles. Second, clustering results are sometimes too sensitive to the types of obstacles. Typically, this approach can not separate closely located clusters. This is because most obstacles are located between clusters or on sparse regions (where obstacles intersect relatively long Delaunay edges). Thus this approach computes a value for $Mean_St_Dev(P)$ that is smaller than what is actually needed. As a consequence, less edges belong to $Short_Edges(p_i)$ and $Long_Edges(p_i)$. Therefore, less number of edges are removed in Phase *I* and Phase *III* of Short-Long clustering procedure. This eventually causes closely located clusters to merge into the same cluster.

- (b) Computing $Mean_St_Dev(P)$ before processing obstacles.
(`ProcessObstacles()` between Step 3 and Step 4.)

This approach is the same as Approach 2a except it calculates the global variation indicator $Mean_St_Dev(P)$ before removing Delaunay edges intersecting overlaid obstacles. This approach obtains a global indicator of variation in P that is not affected by obstacles, but allows obstacles to change the local topological information (incident edges). Thus, the inverse local strength indicator $Local_Mean(p_i)$ is affected by obstacles and

this new local indicator is used in the three-phase clustering.

This alternative is efficient for clustering a target layer with different sets of obstacles since we do not have to compute the Delaunay diagram and $Mean_St_Dev(P)$ for each set of obstacles. Once the Delaunay diagram is built, removal of Delaunay edges that traverse obstacles requires $O([s + t] \log n)$, where s is the number of line segments that determine the obstacles and t is the total number of edges removed from the Delaunay diagram. Typically, s and t are much smaller than n unless the set of obstacles is extremely complex. Once we have the Delaunay diagram and $Mean_St_Dev(P)$ computed, clustering with respect to a new set of obstacles can be obtained in logarithmic time. Thus, this approach is suitable for exploratory data analysis and “what-if” analysis. Another good aspect of Approach 2b is that $Mean_St_Dev(P)$ is now invariable regardless of types of obstacles considered in clustering. Namely, $Mean_St_Dev(P)$ in the presence of obstacles is the same as one in the absence of obstacles. Thus, this approach is able to automatically detect globally significant (within the context of R and P) spatial concentrations irrespective of types of obstacles. Thus, we can compare and contrast clustering in the absence of obstacles with clustering in the presence of obstacles with the same global variation indicator.

- (c) Removing Delaunay edges after computing spatial clusters.

(`ProcessObstacles()` between Step 5 and Step 6.)

We compute spatial clusters with total disregard for obstacles. When the results with obstacles are to be presented, obstacles are used to cut, divide or separate clusters. Clusters are built with a geometry that is oblivious to obstacles.

This last approach is the most efficient computationally if one considers that it is very likely that obstacles are actually different geographical themes and they are stored in different layers. This last approach computes the clusters independently of obstacles (and thus, the computational effort

for clustering depends only on the size n of the dataset P). It depends on the complexity of obstacles only later when an overlay of clusters and obstacles is being performed; however, this is very fast because the data in P has been summarized to clusters and the complexity of obstacles is usually much smaller than n . Thus, exploration with alternative sets of obstacles becomes very fast (the point clustering needs to be computed only once). However, this approach incorporates obstacles only as separators of clusters and not as elements that affect the neighboring information of the point data.

Each approach offers valid modeling for settings with obstacles. The user may have application specific information to prefer one above the other. For example, does housing distribute along a rail track or does the rail track cut and separate an apparent cluster of houses? The distinction may be historical, if the rail track was there before many of the houses, which because of the presence of a station locates the cluster along the track. Or it may be that the need for a fast train has developed the train line after the cluster of houses, effectively splitting a cluster.

Nevertheless, each decision to model the geometry, and to allow its impact on the topology will result in slightly different results. With obstacle Short-Long clustering, the scenarios of different geometries can all be explored and contrasted. They offer different geometric interpretations and they offer different computational trade-offs.

Table 5.1: Comparison of point-centric statistics in the four proposed approaches in the presence of obstacles.

	Approach 1	Approach 2a	Approach 2b	Approach 2c
$Local_Mean(p_i)$	changed	changed	changed	not changed
$Mean_St_Dev(P)$	changed	changed	not changed	not changed

Table 5.1 summarizes the differences brought into the two statistics by the four proposed approaches. Approach 1 and Approach 2a have different values of $Local_Mean(p_i)$ and $Mean_St_Dev(P)$ in the presence of obstacles from those in the

absence of obstacles. Thus, they are “tightly-coupled” approaches. However, different values of $Mean_St_Dev(P)$ for different sets of obstacles complicate exploratory data analysis. Also, the computational cost of Approach 1 makes it less attractive. Approach 2c is a “loosely-coupled” approach. The presence of obstacles does not change the two statistics. Obstacles are only used to cut clusters. Approach 2b is seen as a hybrid approach of Approach 2a and Approach 2c. In this approach, the global variation is invariant to sets of obstacles. We believe this approach is the most suitable for exploratory data analysis and “what-if” analysis. Thus, we set Approach 2b as default because the evaluation of global interactions is considered with all Delaunay edges even those intersect obstacles, since this is what allows the discovery of density estimates over R . But when analyzing the neighbors of a data point, obstacles are considered. An edge intersecting an obstacle is not an edge that justifies its end-points are neighbors.

5.1.4 Time complexity analysis

Since `BuildGeneralizedDelaunayDiagram()` requires to construct the constrained Voronoi diagram for every set of obstacles, Approach 1 is more expensive than Approach 2. Short-Long clustering requires $O(n \log n)$ time as discussed in Section 4.4. Removal of edges that traverse obstacles requires $O([s + t] \log n)$, where s is the number of line segments that determine the obstacles and t is the total number of edges removed from the Delaunay diagram as discussed in Approach 2b. This is because for each line-segment L defining an obstacle, its end-points can be located within the Delaunay diagram in $O(\log n)$ expected time. Then, the planar structure of the Delaunay diagram can be navigated along the line-segment L collecting edges to remove in $O(\log n)$ expected time. Typically, s and t are much smaller than n resulting in $O(\log n)$ time to update the proximity information with respect to a set of obstacles unless the set of obstacles is extremely complex. Consequently, obstacle Short-Long clustering (in default mode) only requires $O(n \log n + [s + t] \log n)$ time. The time complexity of obstacle Short-Long clustering is dominated by computing the Delau-

may diagram, and in default mode, this does not depend on obstacles. This means that once we have the Delaunay diagram, it can be stored and analysis with clustering can be obtained in $O(n)$ time for new sets of obstacles.

5.1.5 Performance evaluation

In this section, we evaluate the performance of obstacle Short-Long clustering. We compare and contrast clustering results of the four proposed approaches.

Obstacle Short-Long clustering with a real dataset

To illustrate the contrast with the alternatives of incorporating obstacles into Short-Long clustering, we use a real dataset from Sydney, Australia (refer to Figure 5-5(a)). The study region has rivers and is surrounded by the ocean.

The point dataset that is the target of clustering is presented in Figure 5-5(b). Figure 5-5(c) illustrates the clustering result without obstacles that is just one cluster. Obstacles can be incorporated towards the analysis for EDA. Unfortunately, if obstacles are incorporated as Approach 1, the constrained Voronoi diagram computes distances between the points that are very similar for many cases as if there were no obstacles. This is due to the pathways over the rivers through roads and bridges. Thus, once again, the result is just one cluster as in Figure 5-5(c). The last approach (Approach 2c) results in only two clusters (one labeled with $\triangle$ and $\blackplus$, and the other with $+$, $\star$, $\diamond$ and $\oplus$ in Figure 5-5(d)). This is because only the rivers going East-West cut a North section of the data. Approaches that compute the ordinary Voronoi diagram but use obstacles to remove edges of the Delaunay diagram produce much better results. The result of Approach 2b is shown in Figure 5-5(d). Note that, the North section is shown as two clusters (one labeled with $\triangle$ and the other with $\blackplus$) although there is no obstacle between these two clusters. This is because the inverse local strength is changed due to the presence of obstacles which makes the separa-

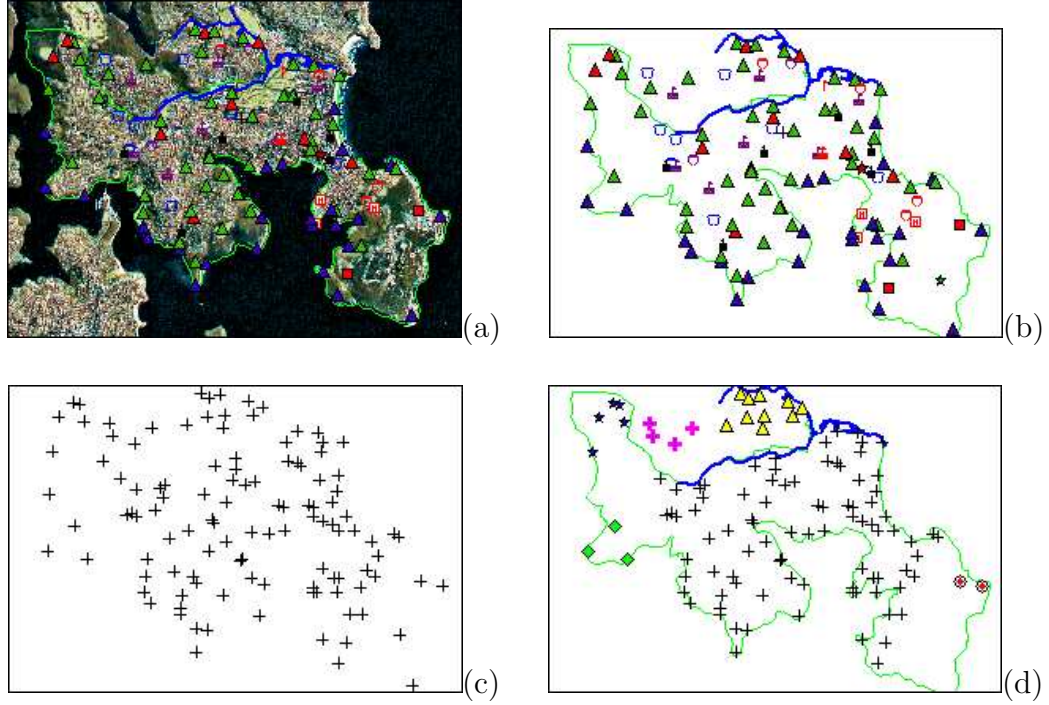


Figure 5-5: Obstacle Short-Long clustering with a real dataset from Sydney, Australia: (a) The study region has rivers and surrounded by the ocean; (b) Point features ($\|P\| = 115$); (c) Clustering result of plain Short-Long clustering (1 cluster); (d) Clustering result of obstacle Short-Long clustering (6 cluster and 4 objects are isolated).

tion between the two clusters globally significant. Similarly on the region south of the East-West rivers. We actually have four clusters (labeled with $+$, $\star$, $\diamond$ and $\oplus$). The geography of the coastline makes these relevant subgroups. In the light of the obstacles, we see the North-West cluster labeled with $\star$ is indeed relatively far from other points, similarly the points in the east peninsula labeled with $\oplus$, as well as the points in the west peninsula labeled with $\diamond$. Note that, we have removed points that are placed in a cluster by themselves (singletons).

Obstacle Short-Long clustering with synthetic datasets

To illustrate that obstacle Short-Long clustering produces robust results in the presence of noise and obstacles with complex scenarios, we present clustering results on several synthetic datasets that have previously used as benchmarks in Section 4.5.2. Experimental results reported here are produced with Approach 2b.

Figure 5-6(a), (c), (e) and (g) show synthetic obstacles. Figure 5-6(b), (d), (f) and (h) depict results of obstacle Short-Long clustering on the datasets shown earlier in Figure 4-11. These figures use the synthetic obstacles. Figure 5-6(b) depicts 18 spatial aggregations for the synthetic dataset $\mathcal{I}$ (Figure 4-11(a)) in the presence of the waterways obstacle (Figure 5-6(a)). The presence of a range of mountains obstacle (Figure 5-6(c)) in the synthetic dataset $\mathcal{II}$ (Figure 4-11(c)) results in 14 spatial clusters as shown in Figure 5-6(d). Approach 2b has no difficulties with the CHAMELEON's benchmark as shown in Figure 5-6(f) and Figure 5-6(h). It suggests 14 and 13 spatial aggregations in the presence of obstacles, respectively. As illustrated, heterogeneous cluster sizes, different cluster densities and various cluster shapes do not hinder Short-Long clustering in the absence or presence of obstacles.

5.1.6 Discussion

In this section, we provide a comparison of obstacle Short-Long clustering with COE [139, 140]. For our best knowledge, these are the only methods for clustering large geo-referenced point data in the presence of obstacles.

COE is a modification of the k -partitioning algorithm named CLARANS [108] that embodies a randomized heuristics to solve the k -median problem under the assumption that the number k of clusters is known. Obstacle Short-Long clustering has several direct advantages over COE. The first advantage is that obstacle Short-Long clustering uses the Euclidian distance and not the square of the Euclidian distance. Using the square of the Euclidian distance is typically motivated by considerations

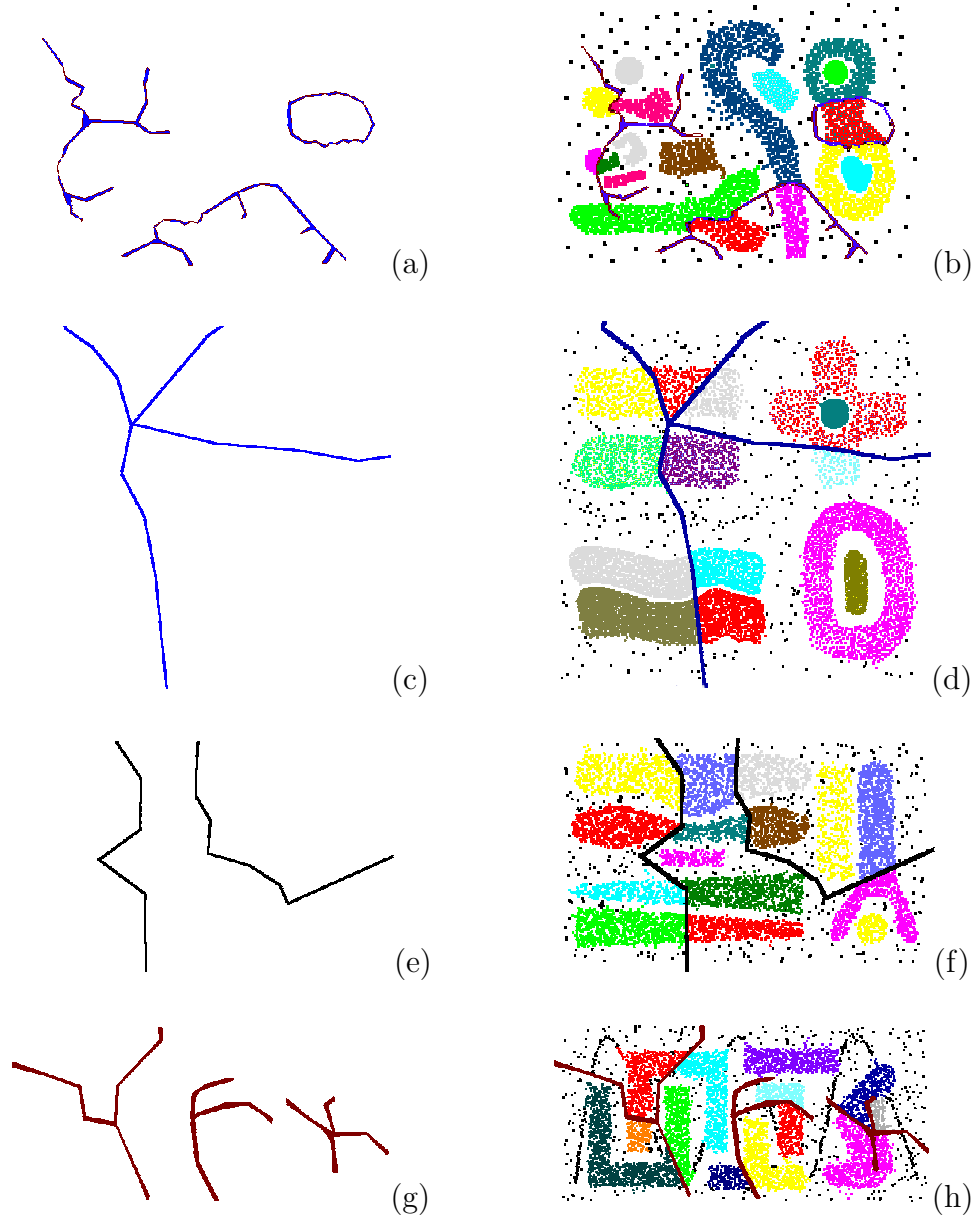


Figure 5-6: Obstacle Short-Long clustering with the synthetic datasets those shown earlier in Figure 4-11: (a) Waterways; (b) Clustering of the synthetic dataset $\mathcal{I}$ with the waterways (18 clusters); (c) A range of mountains; (d) Clustering of the synthetic dataset $\mathcal{II}$ with the range of mountains (14 clusters); (e) Rivers; (f) Clustering of the synthetic dataset $\mathcal{III}$ with the rivers (14 clusters); (g) Political boundaries; (h) Clustering of the synthetic dataset $\mathcal{IV}$ with the political boundaries (13 clusters).

like speeding the implementation by a constant factor and/or by an inductive principle of least-squares (as is the case with k -means [34]). However, using squares of the Euclidian distance in the modeling makes the approach extremely sensitive to outliers and noise [35, 104, 124]. A second advantage of obstacle Short-Long clustering over COE is that COE needs to experiment with different values for the number of clusters while obstacle Short-Long clustering does not. This is a significant benefit, both in CPU-time and in user-analysis time. As third advantage, obstacle Short-Long clustering operates with all the data and does not require the preprocessing of COE to form mini-clusters. This construction of mini-clusters introduces numerical error and biases to the grouping. Thus, the quality of the clustering is reduced and the exploratory capability is limited in COE. Fourth, COE uses a randomized interchange heuristic to approximate the optimum set of k representatives for the clusters. This heuristics is a very poor hill-climber that can not guarantee even local optimality and that sacrifices significantly the quality of the clustering for speed in the optimization. Much better variants of interchange heuristics are well-known from the literature of facility location, and they have been applied to medoid-based clustering [44]. Fifth, because of the representative-based nature of COE, the algorithm has a bias for convex clusters, and in particular for spherical clusters. This is probably acceptable when statistical justifications allow to assume that the data has a probabilistic distribution from a mixture model of exponential distributions. However, in the presence of obstacles, this assumption will certainly be very difficult to sustain. Obstacle Short-Long clustering is not restricted to convex clusters as discussed in Section 5.1.5.

One thing in which obstacle Short-Long clustering (especially, Approach 1 and Approach 2a) and COE are similar is that they are a “tightly-coupled” approach to including obstacles into the clustering. That is, rather than computing the clustering ignoring obstacles and then cutting the resulting groups by obstacles, obstacles are used to modify parameters (the distance or neighboring information) when considering the cluster. In the case of COE, there is no real explanation about the “loosely-coupled” approach, specially since this is more efficient and produces similar results as the “tightly-coupled” approaches with COE. On the other hand, obstacle Short-

Long clustering not only includes the “loosely-coupled” approach (Approach 2c), but provides the hybrid Approach 2b that is more efficient than the “tightly-coupled” approaches (Approach 1 and Approach 2a) and more effective than the “loosely-coupled” approach (Approach 2c).

5.2 Minkowski metrics

Different metrics result in different structural models. Different clustering results are derived from different structural models. Short-Long clustering in the absence or presence of obstacles discussed so far assumes the Euclidean metric where the amount of separation is based on the existence of a linear path between pairs of points. In many spatial settings, the Euclidean distance is neither realistic nor applicable. For example, in urban geography, the Manhattan distance better approximates real world situations [84] or the Dominance distance may be more applicable. Even these distances are obstructed by obstacles. Figure 5-7 illustrates this with real data from the city of Brisbane. Streets demonstrate that the Manhattan metric is more realistic and applicable to this case. Moreover, the Brisbane river obstructs paths traveling between p_1 and p_2 in the Manhattan metric.



Figure 5-7: Real data from the city of Brisbane illustrating unsuitability of the Euclidean metric.

We instantiate two metrics of the Minkowski metric in this section, the Manhattan metric and the Dominance metric. We illustrate the application of Short-Long clustering with the Manhattan metric and the Dominance metric for mining patterns in urban geography or situations where paths only run horizontally and vertically (grid-like paths). This extension makes it possible to compare and contrast clustering P with different metrics. Our exploratory tool allows users to plug in the metric that they desire, because the software is designed in this way. Recall the Template-Method Pattern for our framework in Figure 3-6.

5.2.1 Structural models in the Minkowski metric

Adjacency information captured by the Delaunay diagram differs according to the underlying metric. Let $d_{L_p} : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}$ denote the p -th Minkowski metric defined by

$$d_{L_p}(p_i, p_j) = (|p_{ix} - p_{jx}|^p + |p_{iy} - p_{jy}|^p)^{1/p} \quad (p \in \mathbb{R} \cup \{\infty\}, \text{ and } p \geq 1). \quad (5.1)$$

If $p = 1$, then Equation 5.1 becomes

$$d_{L_1}(p_i, p_j) = |p_{ix} - p_{jx}| + |p_{iy} - p_{jy}|, \quad (5.2)$$

which is called the Manhattan metric, the taxicab metric, the city-block metric or the rectilinear metric [84, 110]. The Minkowski metric becomes the Euclidean metric when $p = 2$. If $p = \infty$, then the Minkowski metric becomes

$$d_{L_\infty}(p_i, p_j) = \max\{|p_{ix} - p_{jx}|, |p_{iy} - p_{jy}|\}, \quad (5.3)$$

which is called the dominance metric, the maximum metric or the supremum metric [73, 110]. We denote by $VD_{L_p}(P)$ the Voronoi diagram and denote by $DD_{L_p}(P)$ the Delaunay diagram in the Minkowski metric L_p ¹.

¹In this thesis, the Euclidean metric is the default metric space. Thus, the notation of the Euclidean metric (L_2) can be omitted throughout this thesis unless confusion arises. For example, $DD(P)$ can be used for $DD_{L_2}(P)$ and $d(p_i, p_j)$ can be used for $d_{L_2}(p_i, p_j)$.

Figure 5-8 illustrates differences between $VD_{L_p}(P)$ and $DD_{L_p}(P)$ for the three metrics under study. Not only are Voronoi regions of p_1 (shaded areas) different, but neighborhoods vary with the choice of metric. In Figure 5-8(a), the Voronoi region of

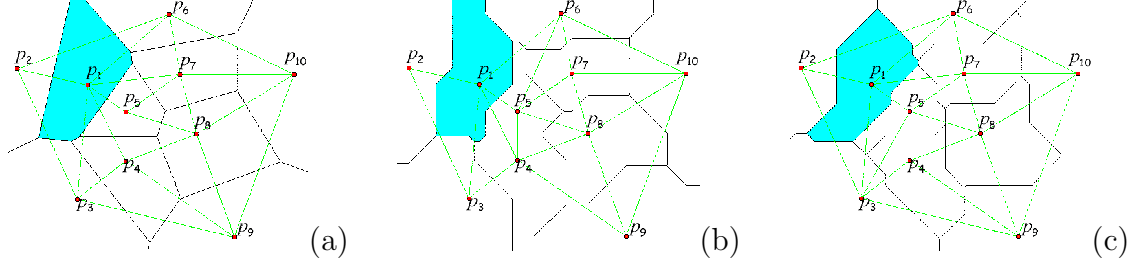


Figure 5-8: Neighborhoods defined by $VD_{L_p}(P)$ and $DD_{L_p}(P)$ ($\|P\| = 10$): (a) $VD(P)$ and $DD(P)$; (b) $VD_{L_1}(P)$ and $DD_{L_1}(P)$; (c) $VD_{L_\infty}(P)$ and $DD_{L_\infty}(P)$.

p_1 in $VD(P)$ consists of six Voronoi edges, thus p_1 has six neighbors (from p_2 to p_7). These incident edges to p_1 appear explicitly in the dual $DD(P)$. Figure 5-8(b) and Figure 5-8(c) show that p_1 has five neighbors in $VD_{L_1}(P)$ and $VD_{L_\infty}(P)$, respectively. Obviously, they are not the same. The point p_4 is a neighbor of p_1 while p_7 is not in $VD_{L_1}(P)$. However, p_7 is a neighbor of p_1 while p_4 is not in $VD_{L_\infty}(P)$.

5.2.2 Determination of the control value m

Profiles of $Local_Mean(p_i)$ values and $Local_St_Dev(p_i)$ values, and their relative spread to the value of $Mean_St_Dev(P)$ also provide intuitive guidance for the initial value of m in the Manhattan metric and in the Dominance metric. This is similar to the case for the Euclidean metric as discussed in Section 4.3.2. We use the same datasets used in Section 4.3 for illustration purposes. Again, the reader can extrapolate to other datasets from these extreme datasets.

Figure 5-9 depicts $DD_{L_1}(P)$ and $DD_{L_\infty}(P)$ with the CSR dataset shown earlier in Figure 4-5(a). It also displays profiles of point-centric statistics for $DD_{L_1}(P)$ and $DD_{L_\infty}(P)$. Figure 5-9(b) shows no sharp increase in the profiles of $Local_Mean(p_i)$

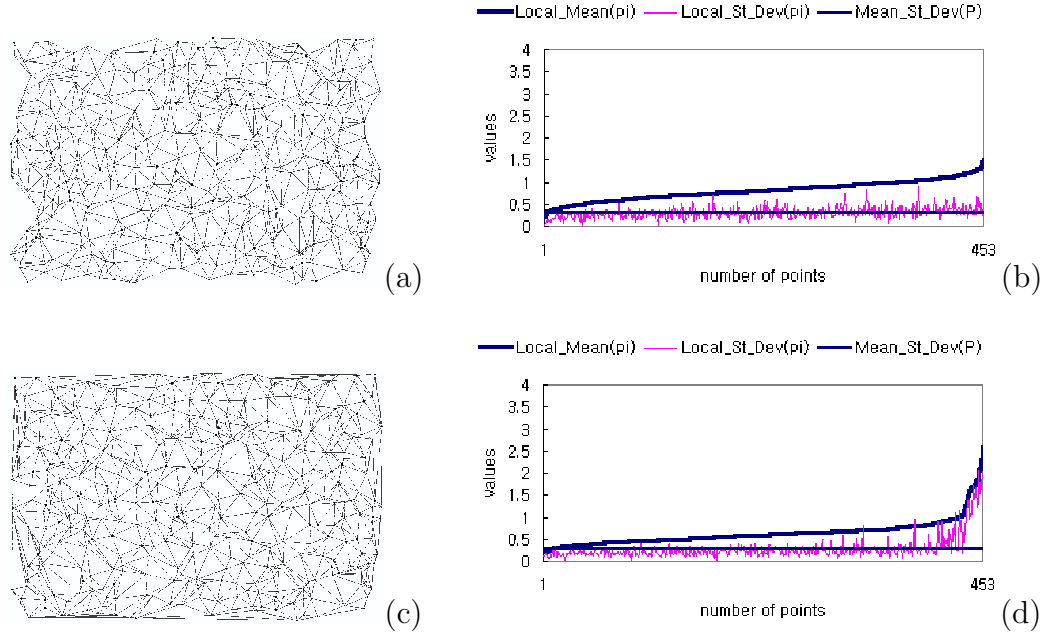


Figure 5-9: Profiles of point-centric statistics for $DD_{L_1}(P)$ and $DD_{L_\infty}(P)$ with the CSR dataset (P) shown earlier in Figure 4-5(a): (a) $DD_{L_1}(P)$; (b) Profiles of point-centric statistics for $DD_{L_1}(P)$ with the CSR dataset; (c) $DD_{L_\infty}(P)$; (d) Profiles of point-centric statistics for $DD_{L_\infty}(P)$ with the CSR dataset.

values and $Local_St_Dev(p_i)$ values in the Manhattan metric (L_1). That is, lengths of Delaunay edges incident to p_i exhibit relative homogeneity in the Manhattan metric. In particular, $Local_St_Dev(p_i)$ values of border points (border of R) are almost similar to those of inner points. That is, there is no distinction between inner points and border points in $Local_St_Dev(p_i)$ values. This is expected in $DD_{L_1}(P)$ since relatively long incident edges to border points are not considered as neighbors in $VD_{L_1}(P)$ and are not encoded in $DD_{L_1}(P)$. Outer boundaries of $DD_{L_1}(P)$ shown in Figure 5-9(a) illustrate this. However, this is not the case with $DD_{L_2}(P)$ and $DD_{L_\infty}(P)$ (refer to Figure 4-5(b) and Figure 5-9(c)). Border points in $DD_{L_2}(P)$ and $DD_{L_\infty}(P)$ exhibit slightly larger $Local_St_Dev(p_i)$ values than inner points. Since there is no distinction between inner points and border points in $Local_St_Dev(p_i)$ values, the mean of $Local_St_Dev(p_i)$ values robustly scales Equation 4.3 and suggests a solid initial value for the tolerance value. Experiments show that the use of $Mean_St_Dev(P)$

for CSR datasets in the Manhattan metric does not detect any significant spatial aggregations but it places all points in the same cluster.

Profiles in Figure 5-9(d) are very similar to ones in Figure 4-8(a). Thus for the same reason discussed in Section 4.3.2, $Mean_St_Dev(P)$ can be used as the tolerance value. Experimental results show that the use of $Mean_St_Dev(P)$ for CSR datasets in the Dominance metric places almost all points in the same cluster in most cases. This demonstrates the robustness of the use of $m = 1$ in CSR datasets in Minkowski metrics.

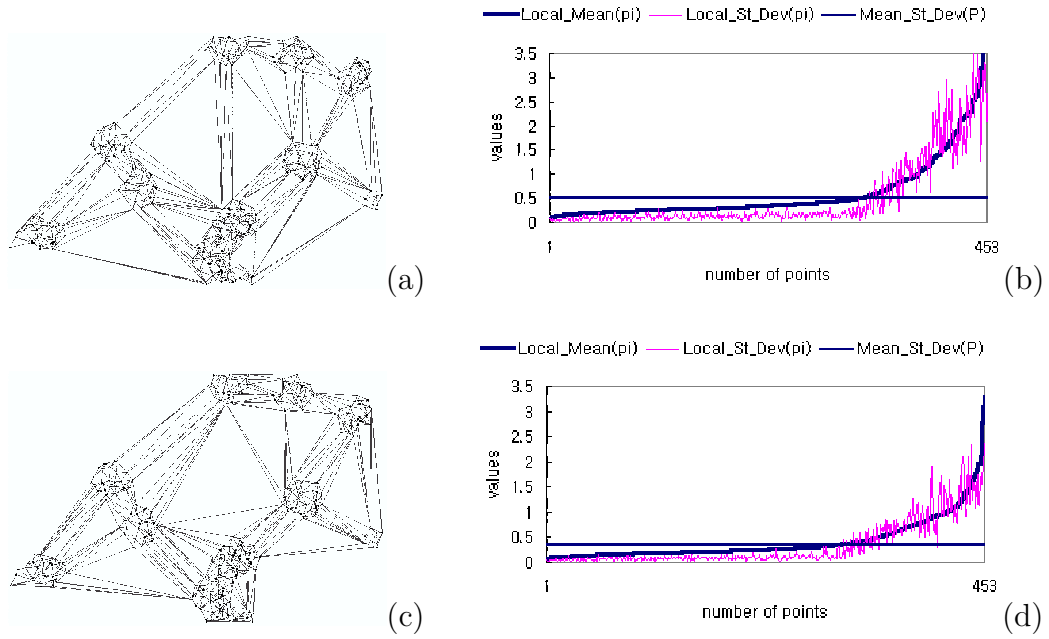


Figure 5-10: Profiles of point-centric statistics for $DD_{L_1}(P)$ and $DD_{L_\infty}(P)$ with the dataset exhibiting clusters (P) shown earlier in Figure 4-5(e): (a) $DD_{L_1}(P)$; (b) Profiles of point-centric statistics for $DD_{L_1}(P)$ with the dataset exhibiting clusters; (c) $DD_{L_\infty}(P)$; (d) Profiles of point-centric statistics for $DD_{L_\infty}(P)$ with the dataset exhibiting clusters.

Figure 5-10 displays $DD_{L_1}(P)$ and $DD_{L_\infty}(P)$ with the dataset exhibiting clusters shown earlier in Figure 4-5(e). It also depicts profiles of point-centric statistics for $DD_{L_1}(P)$ and $DD_{L_\infty}(P)$. Delaunay diagrams in Figure 4-5(f), Figure 5-10(a) and

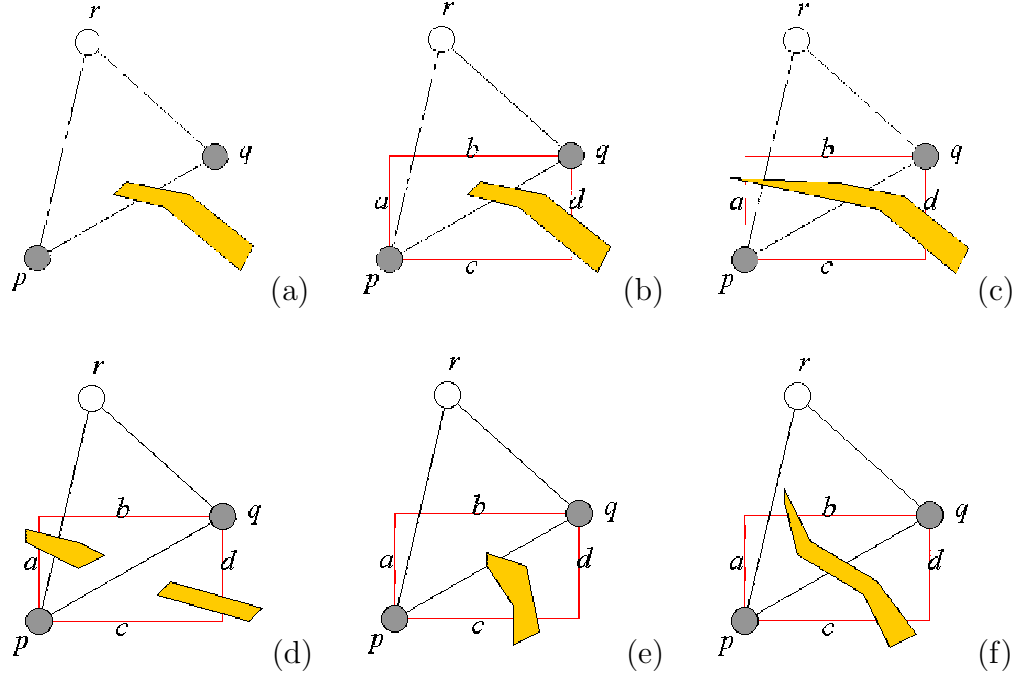
Figure 5-10(c) illustrate different adjacency information according to different metrics. However, profiles in Figure 5-10(b) and Figure 5-10(d) are very similar to one in Figure 4-8(c). The value of $Mean_St_Dev(P)$ clearly separates inner points from border points in these figures. That is, $Mean_St_Dev(P)$ clearly indicates a sharp increase in the profile of $Local_St_Dev(p_i)$ values. Thus, the control value m is set to 1 as default for datasets exhibiting clusters in Minkowski metrics.

5.2.3 Obstacle Short-Long clustering in Minkowski metrics

As illustrated in Figure 5-7, obstacles obstruct the Manhattan distance and the Dominance distance. This section discusses obstacle Short-Long clustering with these metrics. Approach 1 in these metrics is straightforward. The edge removal approach (Approach 2) that removes edges to extract the topological information in the presence of obstacles allows for variants according to geometric models of how obstacles block paths. That is, the user can also specify the interaction between obstacles and paths. This interaction is used to infer the topological information from the geometric information. We illustrate now this issue with the Manhattan metric and the Dominance metric. This issue is not apparent in the Euclidean metric.

If a Delaunay edge $e_{p,q}$ intersects an obstacle (for example, Figure 5-11(a)), then the Delaunay edge is simply removed. This models that if the direct line from p to q is unfeasible because of obstacles, then the edge representing the *is_nearby_neighbor* topological relation must disappear. Now, the obstacle has different effects in the geometry of the Manhattan distance for p and q . Note that, the obstacle shown in Figure 5-11(a) does not intersect all Manhattan paths². In particular, ab is not intersected but the path cd . This is depicted in Figure 5-11(b). In fact, there exist many Manhattan paths between p and q . However, in this first illustration we consider Manhattan paths composed of two line segments (ab and cd) as depicted in Figure 5-11(b) (note that, *obstacle blocking* between two points p and q in the

²Manhattan paths between points p_i and p_j are paths with a set of axis-parallel line segments ($e_1, e_2, \dots, e_k$) where $\sum_{l=1}^k d(e_l)$ is the same as $d_{L_1}(p_i, p_j)$.

Figure 5-11: $DD_{L_p}(P)$ with data exhibiting various types of obstacles.

Manhattan distance may be defined by another researcher as an obstacle that blocks all Manhattan paths). Our geometric model reflects situations where data is not too sparse and where Delaunay edges link two closely located neighbors where not many Manhattan paths (axis-parallel streets) exist between them. More formally, it is up to the user to decide what topological information to be extracted from a geometric setting. In a situation like Figure 5-11(b), where an obstacle intersects the Delaunay edge $e_{p,q}$ and the Manhattan path cd ; but the obstacle does not interfere the other Manhattan path ab , one first answer is to decide that the obstacle does not completely prevent traversing Manhattan paths between points p and q . Then, on this intuition, the edge $e_{p,q}$ is not removed for clustering in the Manhattan metric despite it intersects the obstacle. However, when an obstacle intersects the two Manhattan paths (as in Figure 5-11(c)), the edge $e_{p,q}$ is removed. Another interesting example is shown Figure 5-11(d). Despite two obstacles do not intersect $e_{p,q}$, they prevent traversing the two Manhattan paths between p and q . The user may opt to decide that in this geometric setting the edge $e_{p,q}$ is to be removed.

Note that for other metrics, the user is also in need of defining the extraction of topological information from the geometric setting. In L_∞ , $|p_x - q_x|$ is the Dominance distance in the examples shown in Figure 5-11. Thus, two line segments b and c are dominant paths. The two obstacles in Figure 5-11(d) do not block the two dominant paths. Thus, the user may consider that $e_{p,q}$ still represents a proper interaction in the presence of obstacles in L_∞ . An obstacle in Figure 5-11(e) blocks one of the dominant paths namely c , but does not obstruct b . On the other hand, an obstacle in Figure 5-11(f) completely intersects both dominant paths. Thus, we are inclined to suggest that the Delaunay edge $e_{p,q}$ in Figure 5-11(e) stays, while in Figure 5-11(f) (from $DD_{L_\infty}(P)$) the edge is to be removed.

Thus, we allow users to incorporate their extraction of geometric to topological information according to how it best fits their application. What we are proposing goes beyond a clustering algorithm for spatial data. Because Short-Long clustering sits at the topological level and is linear in the topological information, we are actually presenting a framework for clustering with respect to many metrics and many geometric interpretations of obstacle blocking. In fact, our software is designed in the spirit of object-oriented design, design patterns and frameworks, where our clustering only requires the user to supply components that decide if an edge is to be included in the topological model or not (as in the Template-Method Pattern [86]). Users can provide their own components that encapsulate the geometry of the metric and the geometry of obstacle blocking.

Table 5.2 summarizes how we have suggested to apply edge removal for the data shown in Figure 5-11 for the metrics under study, but we insist that it is the user who may incorporate more ways. The exploration of these alternative definitions and the contrasting of clustering results bring insights into the geographical phenomena.

Table 5.2 shows that obstacle Short-Long clustering with L_2 will be the most inclined to separate points in the presence of obstacles while obstacle Short-Long clustering with L_∞ will be the most resistant. The definitions we have provided for the geometric interaction of obstacles only preserve the Delaunay edge $e_{p,q}$ in $DD_{L_2}(P)$ for the

Table 5.2: Delaunay edge removal when obstacles appear in Minkowski metrics for the data shown in Figure 5-11.

	$e_{p,q}$ in (b)	$e_{p,q}$ in (c)	$e_{p,q}$ in (d)	$e_{p,q}$ in (e)	$e_{p,q}$ in (f)
the Euclidean metric	removed	removed	preserved	removed	removed
the Manhattan metric	preserved	removed	removed	preserved	removed
the Dominance metric	preserved	preserved	preserved	preserved	removed

obstacles shown in Figure 5-11(d). On the other hand, they only remove $e_{p,q}$ in the Dominance metric for the obstacle depicted in Figure 5-11(f).

5.2.4 Time complexity analysis

Lee [87] has proposed subquadratic algorithms for computing $VD_{L_p}(P)$ and $DD_{L_p}(P)$. Since the number of edges in $DD_{L_p}(P)$ is linear [87], the three-phase Short-Long clustering procedure is linear. Thus, Short-Long clustering with Minkowski metrics in the absence of obstacles requires $O(n \log n)$ time. Note that, the component that decides on the extraction of topological information (preservation or removal of a Delaunay edge) requires only constant time. For example, for each Manhattan Delaunay edge the determination for $e_{p,q}$ tests 4 segments for intersection in the presence of obstacles. And, it tests 2 segments for each Dominance Delaunay edge. Thus, the time required for clustering in the presence of obstacles in L_1 and L_∞ still remains in $O(n \log n + [s + t] \log n)$. Still, the complexity of computing $DD_{L_p}(P)$ dominates the process, thus once proximity graphs are constructed, Short-Long clustering with Minkowski metrics in the absence or presence of obstacles requires only linear time.

5.2.5 Performance evaluation

Due to the apparent cluster boundaries in the synthetic datasets shown in Figure 4-11, clustering results of plain Short-Long clustering in the Manhattan metric and the

Dominance metric are very similar to those in the Euclidean metric. However, this is not expected in the presence of obstacles as discussed in Section 5.2.3. Obstacle Short-Long clustering in the Dominance metric is likely to produce less number of clusters than the other two metrics. We illustrate this with real datasets from urban areas of Brisbane, Australia that do not have clear cluster boundaries.

Figure 5-12 illustrates datasets including obstacles. Figure 5-12(a) shows 415 major living areas in inland Brisbane suburbs. Suburbs appear as shaded polygons. Figure 5-12(b) depicts waterways (light blue in thin solid lines) and the Brisbane river (dark blue in thick solid lines) also with the shaded areas of suburbs. Figure 5-12(c) overlays these two layers. Visual inspection indicates that living areas are aligned along the river despite there are no clear cluster boundaries. The topological information disregarding obstacles appears in the next row of subfigures. Figure 5-12(d) is the topological information derived as the dual of the Voronoi diagram for the Euclidean distance. Figure 5-12(e) is the topological information for L_1 while Figure 5-12(f) is the topological information for L_∞ . Clustering ignoring obstacles results in essentially one large cluster. At first sight, all metrics in the absence of obstacles produce essentially the same result. A cluster that reflects the elongated form of the suburbs. Figure 5-12(g), (h) and (i) in L_2 , L_1 and L_∞ show each one cluster that is not exactly the same despite there does not exist a big difference. The cluster in L_1 is the most compact one. The other two seem to have a slight peninsula on the North-West.

By considering waterways obstacles, the clustering method reports more differences. Clustering in L_2 is depicted in Figure 5-12(j). In this case, clustering detects 4 spatial aggregations in the south of the Brisbane river and 1 cluster in the north. The one in the north has left out the peninsula referred above and is a cluster that is more elongated North to South. Using L_1 , similar clusters are obtained in the south as shown in Figure 5-12(k). One big cluster in the north of the Brisbane river demonstrates the main difference between clustering results in L_1 and L_2 . This North cluster incorporates the peninsula, and it is elongated East-West. Clearly reflecting

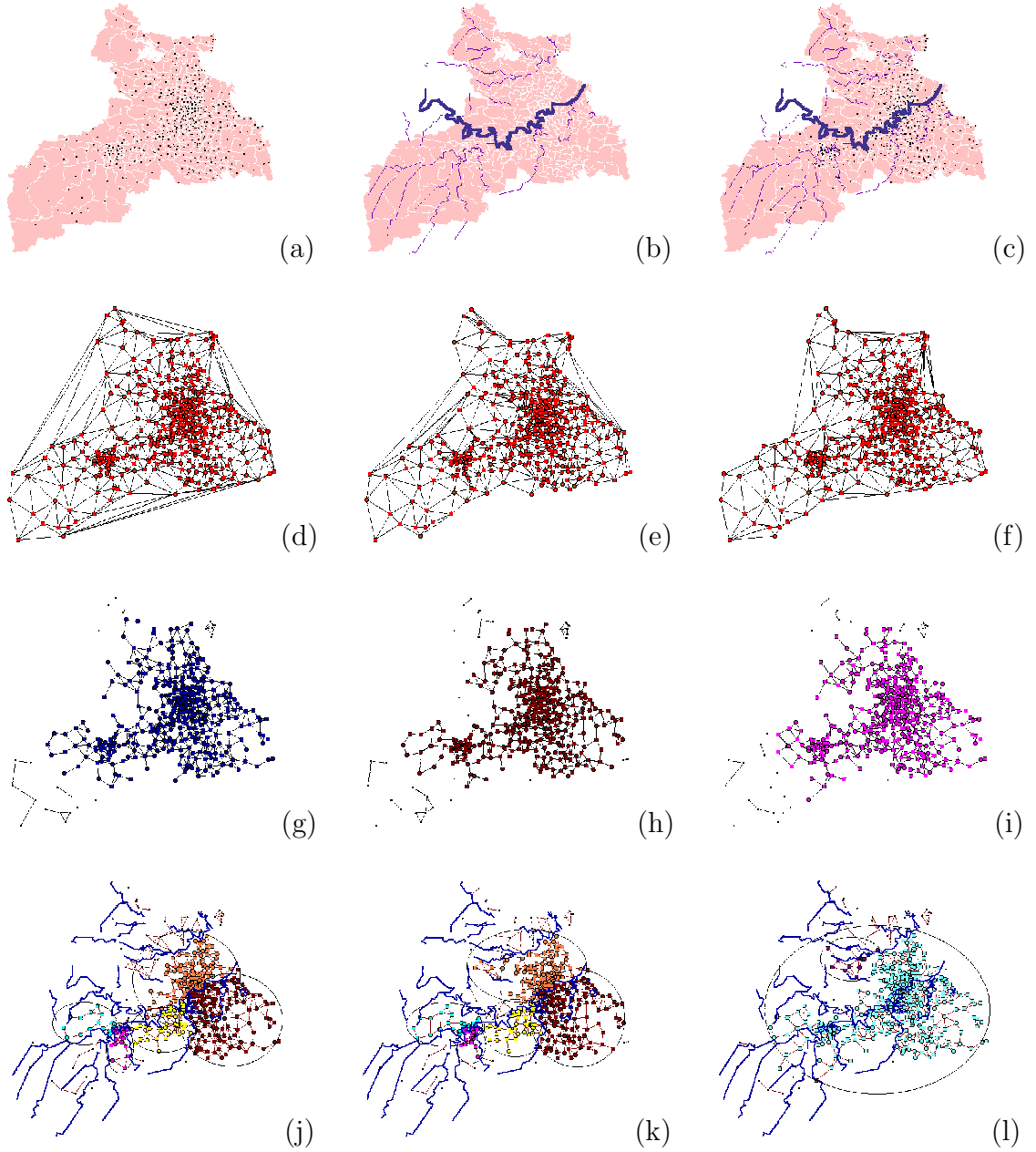


Figure 5-12: Clustering with Minkowski metrics in the presence of obstacles ($\|P\| = 415$, clusters less than 3% of $\|P\|$ are not highlighted): (a) Major living areas around Brisbane suburbs in Australia; (b) Waterways including Brisbane river; (c) Living areas with waterways; (d) $DD_{L_2}(P)$; (e) $DD_{L_1}(P)$; (f) $DD_{L_\infty}(P)$; (g) Clustering of $DD_{L_2}(P)$; (h) Clustering of $DD_{L_1}(P)$; (i) Clustering of $DD_{L_\infty}(P)$; (j) Clustering of $DD_{L_2}(P)$ with waterway obstacles (5 clusters); (k) Clustering of $DD_{L_1}(P)$ with waterway obstacles (5 clusters); (l) Clustering of $DD_{L_\infty}(P)$ with waterway obstacles (2 clusters).

that in the North the waterways travel North-West. This is why the big cluster in L_1 shown in Figure 5-12(k) spreads horizontally while the cluster in L_2 depicted in Figure 5-12(j) spreads rather vertically.

Figure 5-12(l) illustrates the effect of obstacles in L_∞ . One small sparse cluster and one large dense cluster are detected. That is, the peninsula has been separated into the small sparse cluster located in the North-West outskirts of Brisbane and away from the Brisbane river namely Samford Valley, Camp Mountain and Wights Mountain suburbs near to the Manorina national park. The large cluster is spread around urbanized suburbs along the Brisbane river. Waterways do not completely block all the dominant paths of the large cluster because they are not as dense as the points. Thus, the exploration with different metrics highlights the interplay of obstacles. Again, this real data demonstrates that clusters in L_∞ are the most unbreakable with obstacles. However, the broken small cluster highlights the contrast on the other two metrics.

5.3 With other proximity graphs

In graph-based clustering, proximity graphs capture and store adjacency information along with edges and nodes. Thus, neighborhoods differ from underlying proximity graphs. This section discusses the extension of Short-Long clustering with other well-known proximity graphs. In particular, the Delaunay diagram is linear in size for 2D but quadratic in size for 3D. However, it is noteworthy that some other proximity graphs (k -nearest neighbor graphs and k -cone spanner graphs) are linear in size for all dimensions. Thus, the main goal of this section is to compare and contrast clustering results using different proximity graphs in the application of Short-Long clustering. Then, to provide guidelines to have exploration tools for very large spatial datasets, where the user may trade the argument free use of the Delaunay diagram for an argument driven exploration of the data with other proximity graphs.

5.3.1 Proximity graphs are explicit models of neighborhoods

Due to the succinctness of the Delaunay triangulation, its subgraphs are widely used for representing spatial proximity of point data. Note that, the Delaunay diagram is a subgraph of the Delaunay triangulation. The Gabriel graph of P , denoted by $GG(P)$, is another widely used representation of spatial closeness [51, 92, 98]. $GG(P)$ has an edge $e_{i,j} = (p_i, p_j)$ if and only if

$$d^2(p_i, p_j) < d^2(p_i, p_k) + d^2(p_j, p_k), \text{ for all } p_k \in P, k \neq i, j. \quad (5.4)$$

In other words, two points $p_i, p_j \in P$ are neighbors if and only if all other points in $P - \{p_i, p_j\}$ lie outside the circle having p_i and p_j as antipodal points [92]. Matula and Sokal [98] believe that Gabriel graphs provide sufficient but not excessive interconnections. Note that, the proximity information in Delaunay triangulations is an explicit function of three data points, but of only two points for Gabriel graphs (of course, they are implicitly dependent on the other points being outside the circles proposed in the definition). However, the issue is that Gabriel graphs may lose some proximity information due to the simplification.

The relative neighborhood graph of P , denoted by $RNG(P)$, is based on the notion of “relative close” neighbors [136]. Two points $p_i, p_j \in P$ form an edge in $RNG(P)$ if and only if

$$d(p_i, p_j) \leq \min \max\{d(p_i, p_k), d(p_j, p_k)\}, \text{ for all } p_k \in P, k \neq i, j. \quad (5.5)$$

Namely, an edge $e_{i,j} = (p_i, p_j)$ explicitly represents *is_nearby_neighbor* if and only if all other points in $P - \{p_i, p_j\}$ lie outside of the intersection of two circles: the circle with center at p_i and radius $d(p_i, p_j)$ and the circle with center at p_j and the same radius [110]. Thus, they are neighbors if they are relatively close enough as they are to any other point. These sound notions of spatial proximity make $RNG(P)$ as well as $GG(P)$ reasonable candidates for spatial point pattern analysis.

The $MST(P)$ has long been used for single-linkage clustering. Namely, $MST(P)$ is a connected subgraph with minimum total length and with vertex set P (in fact,

$MST(P)$ is not unique, there may be several Minimum Spanning Trees for the same dataset P . Since $MST(P)$ is a subgraph of $RNG(P)$ which in itself is a subgraph of $GG(P)$. And $GG(P)$ is a subgraph of $DT(P)$ [110], each encodes less proximity information along this family. The simplicity of $MST(P)$ makes its local statistical properties vulnerable to small changes. More seriously, $MST(P)$ does not reflect local optimization information like $DT(P)$, $GG(P)$ or $RNG(P)$, but rather global optimization. Thus, it ignores to a larger degree local variations.

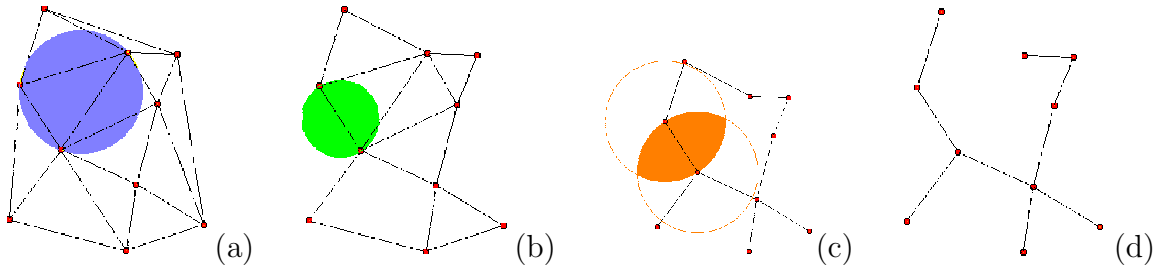


Figure 5-13: Subgraphs of the Delaunay triangulation ($\|P\| = 10$): (a) $DT(P)$; (b) $GG(P)$; (c) $RNG(P)$; (d) $MST(P)$.

Figure 5-13 shows the four graphs on the same dataset and illustrates their alternative definitions. As shown in Figure 5-13(a), a shaded circumcircle that does not include any other points determines a facet of Delaunay triangulations. Similarly, a shaded circle in Figure 5-13(b) does not contain any other points in its interior, besides the two end-points of the diameter. So, these are neighbors to each other in this proximity graph. The intersection of two circles in Figure 5-13(c) is an empty lune (shaded area), thus a straight line between the two center points forms a relative neighborhood edge for the relative neighborhood graph.

The greedy triangulation of P , denoted by $GT(P)$, is another type of spatial tessellation implicitly encoding proximity. $GT(P)$ is obtained by repeatedly inserting the shortest edge that does not intersect any of the previously inserted edges. Since $DT(P)$ and $GT(P)$ are two well-known triangulations that approximate the open problem known as Minimum Weight Triangulation [85], they try to minimize the total length of all the edges. A *greedy* edge $e_{i,j} = (p_i, p_j)$ represents spatial proximity

in the sense that $e_{i,j}$ is the strongest (shortest) interaction that does not interfere previously chosen stronger (shorter) interactions. Figure 5-14 contrasts $DT(P)$ and $GT(P)$. The graph $DT(P)$ has a global optimization property in that the smallest angle is as large as possible. Thus, while all the triangles in $DT(P)$ maximize the smallest angle, some triangles in $GT(P)$ have wide angles and some have very narrow angles.

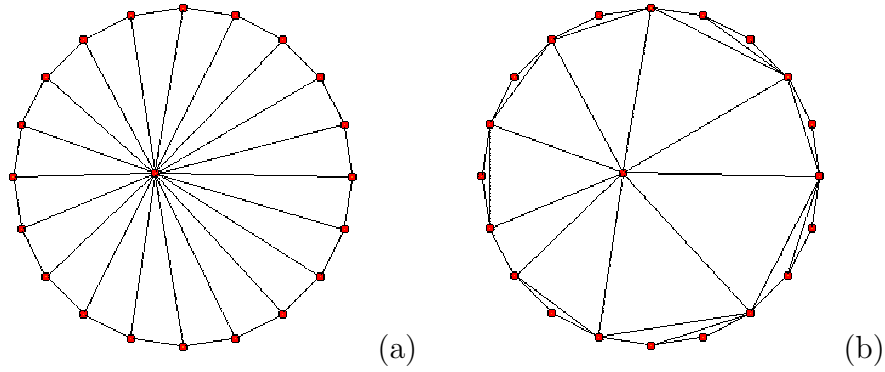


Figure 5-14: A comparison between $DT(P)$ and $GT(P)$ ($\|P\| = 21$):
(a) $DT(P)$; (b) $GT(P)$.

Another popular way of modeling spatial proximity is to simply use a metric and the corresponding distance concept (this is the implicit philosophy between methods for spatial clustering in data mining like DBSCAN [33]). Namely, points are considered as neighbors if and only if they lie within a certain distance d . The corresponding proximity graph is denoted by $d-DG(P)$. This notion seems to be reasonable and equivalent weight is assigned to every point. Several density-based clustering approaches [33, 111] use these d -distance neighbors. However, this has a number of critical problems. First, this proximity is not argument-free. A value for the argument d is required. Thus, parameter-tuning is essential to find a best-fit value, and this may depend heavily on the dataset. Further, globally setting the value of d ignores spatial heterogeneity. In addition, neighbors may not be visible. This makes visibility, clustering in the presence of obstacles and monitoring analysis difficult.

An alternative is to assign the same number of neighbors to each point, namely k -nearest neighbors, denoted by $k\text{-}NN(P)$. This has been used in spatial data mining in CHAMELEON [76] and could be seen as a variant of d -distance neighboring. Here, d is not globally static, but rather dynamic over R . Let p_k be the k -th nearest neighbor to p_i , then the circle with center p_i and radius $d(p_i, p_k)$ contains a fixed number of at least k points (except for p_i) in its interior or on its boundary. Directed edges from p_i to k nearest points in this circle are included in the proximity graph. All points in P are assigned a globally fixed number k of neighbors. The uniformity of the number of neighbors again ignores spatial heterogeneity. In addition, the use of k -nearest neighbors represents non-symmetric neighboring [4]. This happens where p_i is a neighbor of p_j , but p_j is not a neighbor of p_i . This is inappropriate for graph-based clustering where two connected points belong to the same cluster. This could potentially be somewhat alleviated by making every directed edge undirected (referred as $k\text{-}UNN(P)$). But still, visibility is hard to incorporate.

One variant overcoming this invisibility is the family of k -cone spanner graphs of P , denoted by $k\text{-}CSG(P)$. The main idea is that it captures the nearest neighbors in pre-specified k directions or cones of visibility. Thus, k -nearest neighbors, each the nearest in each of k directions. Making directed edges in $k\text{-}CSG(P)$ undirected results in undirected k -cone spanner graphs of P , denoted by $k\text{-}UCSG(P)$.

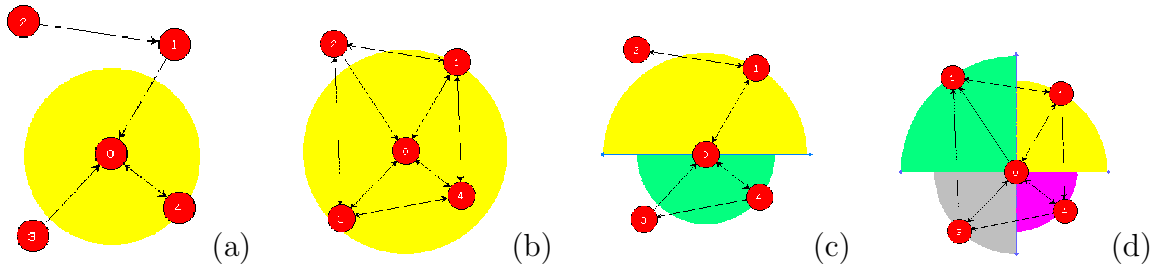


Figure 5-15: Illustration of k -nearest neighbor graphs and k -cone spanner graphs ($\|P\| = 5$): (a) 1-nearest neighbor graph; (b) 3-nearest neighbor graph; (c) 2-cone spanner graph; (d) 4-cone spanner graph.

As depicted in Figure 5-15, $k\text{-}NN(P)$ and $k\text{-}CSG(P)$ are argument-dependent (need

a value for k) and directed (some edges are not symmetric). In Figure 5-15(b), the shaded disk with center at p_0 and radius $d(p_0, 3rd\ nearest\ neighbor\ (p_1))$ contains 3 points in $P - \{p_0\}$ in its interior or on its boundary. These are the 3 neighbors of p_0 in the 3-nearest neighbor graph. Figure 5-15(c) illustrates a 2-cone spanner graph. Thus, each point has at most two neighbors. For instance, a point p_0 has p_1 and p_4 as neighbors. The point p_1 is the nearest point to p_0 in the cone $[0, \pi)$, while p_4 is the closet in $[\pi, 2\pi)$. Also, p_1, p_2, p_3 and p_4 are the 4 neighbors of p_0 in each quadrant in the 4-cone spanner graph shown in Figure 5-15(d). Thus, each cone of orientations (shaded area corresponding to a quadrant) does not contain any other closer point.

Table 5.3: Characteristics of proximity graphs.

	<i>DD</i>	<i>GG</i>	<i>RNG</i>	<i>MST</i>	<i>GT</i>
Symmetry	✓	✓	✓	✓	✓
Planar tessellation	✓	×	×	×	✓
Planar graph	✓	✓	✓	✓	✓
Spatial heterogeneity	✓	✓	✓	✓	✓
Neighbor visibility	✓	✓	✓	✓	✓
Scale-independence	✓	✓	✓	✓	✓
Argument-free	✓	✓	✓	✓	✓
Variable number of neighbors	✓	✓	✓	✓	✓
Local optimization	✓	✓	✓	×	×

	<i>d-DG</i>	<i>k-NN</i>	<i>k-UNN</i>	<i>k-CSG</i>	<i>k-UCSG</i>
Symmetry	✓	×	✓	×	✓
Planar tessellation	×	×	×	×	×
Planar graph	×	×	×	✓	✓
Spatial heterogeneity	×	×	×	×	×
Neighbor visibility	×	×	×	✓	✓
Scale-independence	×	✓	✓	✓	✓
Argument-free	×	×	×	×	×
Variable number of neighbors	✓	×	✓	✓	✓
Local optimization	✓	✓	✓	✓	✓

Table 5.3 summarizes the characteristics of the proximity graphs discussed above. Interestingly, the neighboring of $d-DG(P)$ is scale-dependent, which is different from the others. The graph $k-NN(P)$ is the only proximity graph having a fixed number of

neighbors among the graphs discussed in this section. While the number of neighbors in $k\text{-CSG}(P)$ is bounded by k , not every point has k neighbors (for example, points on the convex hull may have empty cones and thus, less than k neighbors). Only, $DD(P)$ and $GT(P)$ are planar tessellations, while only $d\text{-DG}(P)$, $k\text{-NN}(P)$ and $k\text{-UNN}(P)$ are not planar graphs. The graphs $MST(P)$ and $GT(P)$ are not derived from local optimization, but rather from global criteria. In two dimensions, they all have linear size, however, when we move up to 3D this is not the case. The graphs $k\text{-NN}(P)$, $k\text{-UNN}(P)$, $k\text{-CSG}(P)$ and $k\text{-UCSG}(P)$ are attractive as we move to higher dimensions since they require $O(n \log n)$ time and $O(n)$ space (when k is small with respect to n so that it can be regarded as a constant in data-rich environments).

5.3.2 Performance evaluation

Section 3.2 discusses some examples where typical peak-inference clustering fails to detect interesting spatial aggregations. This section evaluates experimental results of Short-Long clustering with the proximity graphs discussed in the previous section on those examples.

Figure 5-16 shows experimental results with the dataset (Figure 3-3) that exhibits heterogeneous densities, arbitrary shapes and different sizes while Figure 5-17 shows experimental results with the dataset (Figure 3-4) where narrow chains connect clusters. Figure 5-18 shows experimental results with the dataset (Figure 3-5) where narrow gaps appear between high-density clusters. Among the subgraphs of $DT(P)$, the results of $GG(P)$ approximate those of $DD(P)$. Regardless of heterogeneous distributions, Short-Long clustering with $GG(P)$ consistently identifies quality clusters in these examples as shown in Figure 5-16(a), Figure 5-17(a) and Figure 5-18(a). Not surprisingly, $MST(P)$ produces the worst result. This is illustrated in Figure 5-16(c), Figure 5-17(c) and Figure 5-18(c). The main reason for this is that $MST(P)$ has lost proximity information in its $n - 1$ interconnections. Note that, $DD(P)$ has more edges ($3n - 6$ at most) and encodes far more complete spatial proximity. This

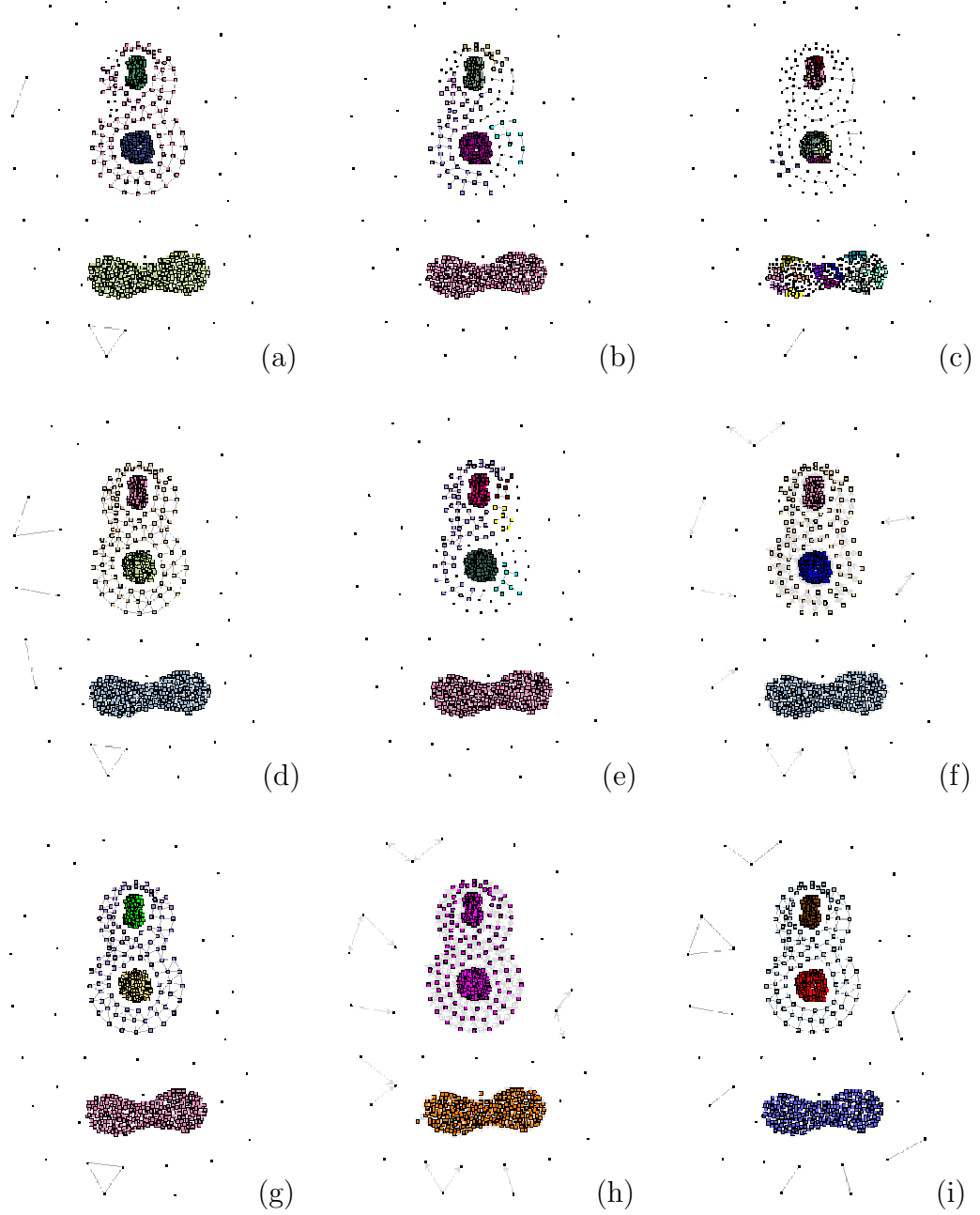


Figure 5-16: Clustering of the dataset shown in Figure 3-3: (a) Result with $GG(P)$ (4 clusters); (b) Result with $RNG(P)$ (7 clusters); (c) Result with $MST(P)$ (18 clusters); (d) Result with $GT(P)$ (4 clusters); (e) Result with $d-DG(P)$ ($d = 102$, 7 clusters); (f) Result with $k-NN(P)$ ($k = 5$, 4 clusters); (g) Result with $k-UNN(P)$ ($k = 5$, 4 clusters); (h) Result with $k-CSG(P)$ ($k = 4$, 2 clusters); (i) Result with $k-UCSG(P)$ ($k = 4$, 4 clusters).

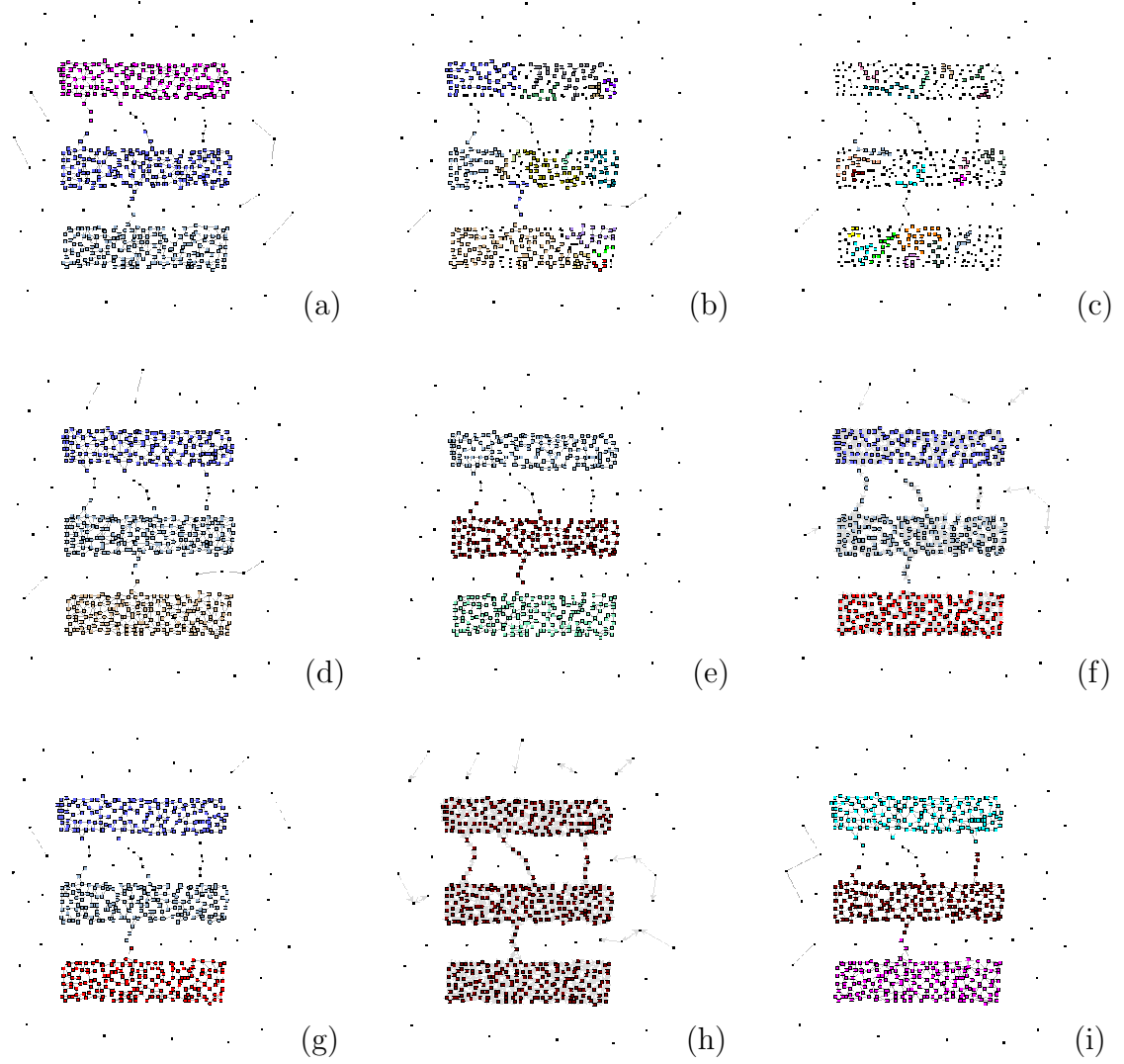


Figure 5-17: Clustering of the dataset shown in Figure 3-4: (a) Result with $GG(P)$ (3 clusters); (b) Result with $RNG(P)$ (17 clusters); (c) Result with $MST(P)$ (22 clusters); (d) Result with $GT(P)$ (3 clusters); (e) Result with $d-DG(P)$ ($d = 8$, 3 clusters); (f) Result with $k-NN(P)$ ($k = 5$, 3 clusters); (g) Result with $k-UNN(P)$ ($k = 4$, 3 clusters); (h) Result with $k-CSG(P)$ ($k = 4$, 1 cluster); (i) Result with $k-UCSG(P)$ ($k = 4$, 3 clusters).

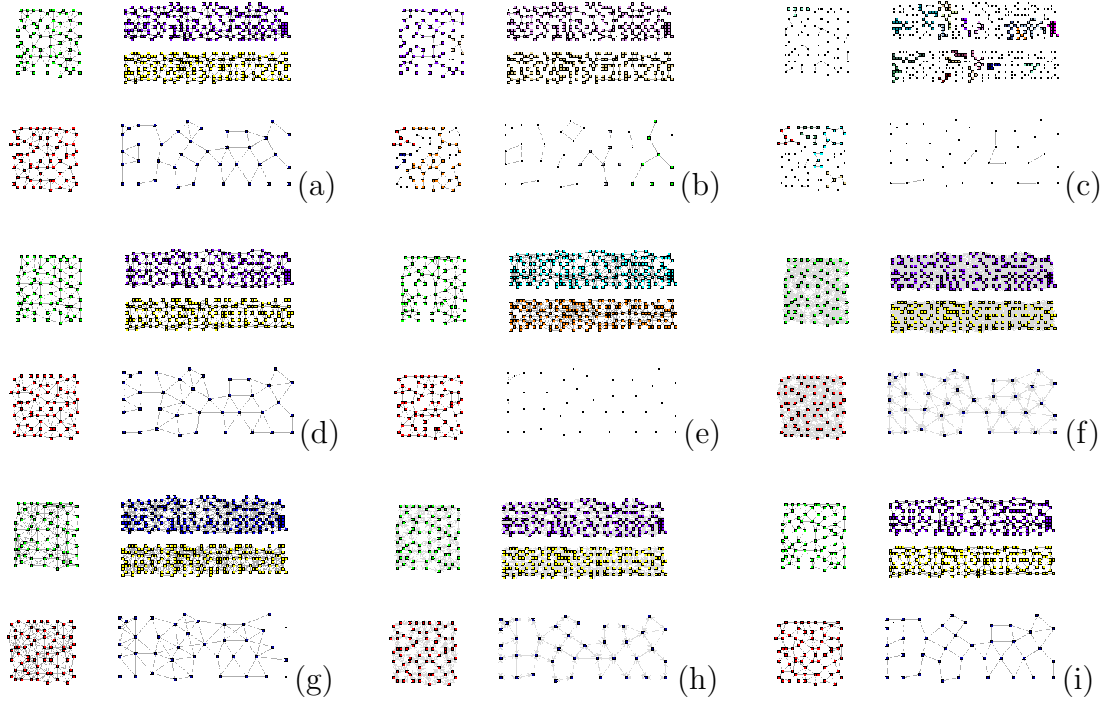


Figure 5-18: Clustering of the dataset shown in Figure 3-5: (a) Result with $GG(P)$ (5 clusters); (b) Result with $RNG(P)$ (9 clusters); (c) Result with $MST(P)$ (25 clusters); (d) Result with $GT(P)$ (5 clusters); (e) Result with $d-DG(P)$ ($d = 4$, 4 clusters); (f) Result with $k-NN(P)$ ($k = 12$, 5 clusters); (g) Result with $k-UNN(P)$ ($k = 12$, 5 clusters); (h) Result with $k-CSG(P)$ ($k = 4$, 5 cluster); (i) Result with $k-UCSG(P)$ ($k = 4$, 5 clusters).

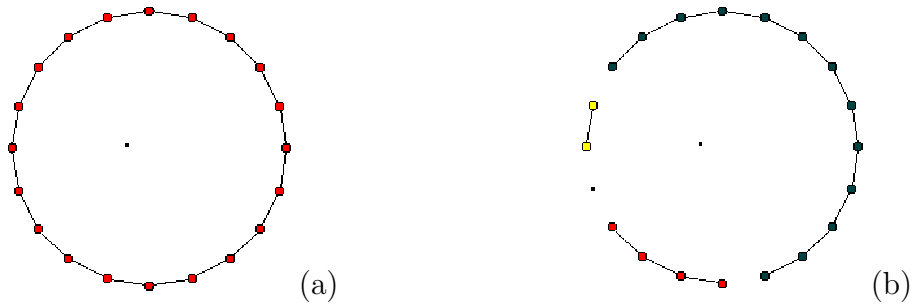


Figure 5-19: Example where clustering results with $DT(P)$ and $GT(P)$ differ significantly (P is the same as in Figure 5-14): (a) Result with $DT(P)$; (b) Result with $GT(P)$.

simple proximity information makes $MST(P)$ extremely vulnerable to small variations, and it is not surprising that $MST(P)$ is known to be fast but also known to be a fragile clustering method. Intermediate quality is obtained with $RNG(P)$ as shown in Figure 5-16(b), Figure 5-17(b) and Figure 5-18(b). The clustering results with $GT(P)$ (Figure 5-16(d), Figure 5-17(d) and Figure 5-18(d)) are as good as those with $DD(P)$. As we discussed earlier, these two graphs are approximations to the Minimum Weight Triangulation. Thus, they may be similar in some cases, and when this happens, resulting clustering is naturally very similar. However, $GT(P)$ may be radically different in some cases, exhibiting the same type of problems as single-linkage clustering. It creates greedily long paths of artificial chains using short edges. Figure 5-19 shows that the two graphs exhibit a great level of dissimilarity resulting different clustering results. Here, $GT(P)$ fails to detect the circular cluster as one.

The arguments d and k have to be tuned by the user for Short-Long clustering with the argument-dependent proximity graphs. This argument-tuning may enable the user to explore datasets with different values of arguments, but is costly in GDM. We experiment with different values of arguments and what we believe the best results are presented in Figure 5-16, Figure 5-17 and Figure 5-18. Typically, larger values of arguments cause closely located clusters to merge into the same cluster while smaller values leave sparse clusters fragmented. This is because large values increase inter-connectivity and smaller values decrease inter-connectivity. However, Short-Long clustering exhibits relatively little sensitivity with respect to the values of arguments. This will be further discussed in Section 6.4.2.

The Short-Long clustering method with $d-DG(P)$ overcomes the difficulties caused by multiple chains (Figure 5-17(e)), but fails to identify clusters with different densities (Figure 5-16(e) and Figure 5-18(e)). This is expected since DBSCAN [33] (that is based on this type of proximity representation) is unable to simultaneously find clusters with different densities. Experimental results with $k-NN(P)$ (Figure 5-16(f) Figure 5-17(f) and Figure 5-18(f)) and its counterpart $k-UNN(P)$ (Figure 5-16(g) Figure 5-17(g) and Figure 5-18(g)) demonstrate the robustness of the Short-Long

clustering criteria. Short-Long clustering has no difficulties with these graphs in these examples where challenges imposed by various types of clusters. This is also the case with $k\text{-UCSG}(P)$ (Figure 5-16(i) Figure 5-17(i) and Figure 5-18(i)). However, $k\text{-CSG}(P)$ is sensitive to heterogeneous densities and narrow chains.

These experiments reveal that Short-Long clustering not only detects quality clusters consistently with argument-free graphs, $DD(P)$, $GG(P)$ and $GT(P)$, but works robustly with argument-dependent proximity graphs, $k\text{-NN}(P)$, $k\text{-UNN}(P)$ and $k\text{-UCSG}(P)$. This extendability offers more exploratory functions to the user. Recall that typical peak-inference clustering fails to identify quality clusters in these examples as discussed in Section 3.2.

5.4 Remarks

The problem of spatial clustering is a matter of identifying neighbors (building neighborhoods) and quantifying their relative closeness or remoteness (establishing a criterion function). Thus, clustering can be viewed as a two-step process: data preprocessing and data grouping. Data preprocessing structures raw input data and produces a clustering schema. The schema will guide data grouping.

This chapter fixes the second process to Short-Long clustering and investigates its applicability to various structural models that arise from incorporating obstacles, different metric information and various structural proximity models. Experiments demonstrate that plain and obstacle Short-Long clustering consistently report quality spatial aggregations in various clustering settings. This extendability enables the user to explore different clustering settings and to select the best one for the data under investigation.

Chapter 6

Short-Long Clustering in Higher Dimensions

Geographical data includes locational dimension, attribute and time [25]. Short-Long clustering in the previous chapters deals with point data with two-dimensional location information (x and y or easting and northing). In some situations, three-dimensional data models better real-world phenomena [14]. For instance, underground and marine geoinformation studies and even astronomy demand clustering techniques capable of dealing with three-dimensional data due to their inherent volumetric nature. Furthermore, the rapid development of 3D GIS (for instance, Voxel Analyst ¹, ArcView 3D Analyst ², VULCAN ³ and Surpac Vision ⁴) is demanding clustering algorithms for exploratory and data mining studies identifying volumetric clusters.

This chapter investigates the extension of Short-Long clustering method to three dimensions ($P \subset \mathbb{R}^3$) that include a measurement z or elevation in a third dimension.

¹<http://www.intergraph.com>

²<http://www.esri.com>

³<http://www.maptek.com>

⁴<http://www.surpac.com.au>

We limit discussions to the Euclidean metric in this thesis, but extensions to the Minkowski metric can be performed as in Section 5.2. As discussed in Section 5.3, $k\text{-UNN}(P)$ (the symmetric closure of $k\text{-NN}(P)$) robustly identifies quality cluster boundaries in various situations (refer to experimental results in Section 5.3.2). This graph is of particular interest in three dimensions, since it requires $O((n+k)\log n)$ time and $O(kn)$ space while three-dimensional $DD(P)$ requires $O(n^2)$ time and space. In this chapter, we discuss three-dimensional Short-Long clustering with the default $DD(P)$ and $k\text{-UNN}(P)$.

First, Section 6.1 discusses benefits from three-dimensional clustering by incorporating additional location information. Then, we investigate the method to determine the initial value of the exploratory argument m with the three-dimensional Delaunay diagram and $k\text{-UNN}(P)$ in Section 6.2. After that, we analyze time complexity and clustering results in Section 6.3 and Section 6.4, respectively.

6.1 Benefits from three-dimensional clustering

Incorporating additional location information in geo-referenced data enables us to distinguish positive-clusters (clusters in higher dimensions, but not in lower dimensions) and negative-clusters (clusters in lower dimensions, but not in higher dimensions).

For example, Figure 6-1 illustrates positive-clusters. Three-dimensional points projected into the xy -plane do not seem to exhibit any spatial concentrations. However, projections to the yz -plane, the zx -plane or the xyz -view reveal that there are clusters in $\mathbb{R}^3$. Thus, GIS may miss spatial aggregations embedded in 3D when manipulating data only on the xy -plane. In general, to determine the 2D projection to reveal a 3D cluster may demand probabilistic assumptions in order to apply parametric statistics (techniques like factor analysis). This alternative will step out from exploratory analysis. On the other hand, Figure 6-2 shows an example of a negative-cluster. The projection of a set of points appears to reveal a cluster that is inexistent in higher

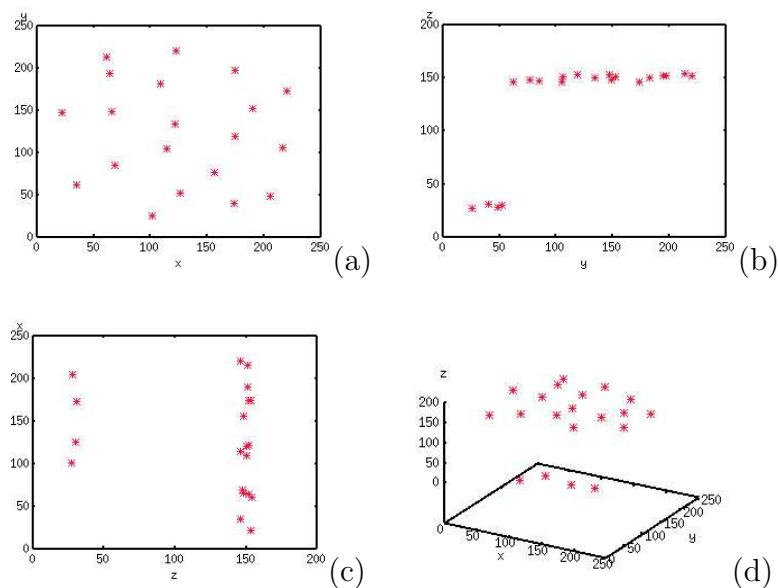


Figure 6-1: Illustration of positive-clusters ($\|P\| = 20$): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view.

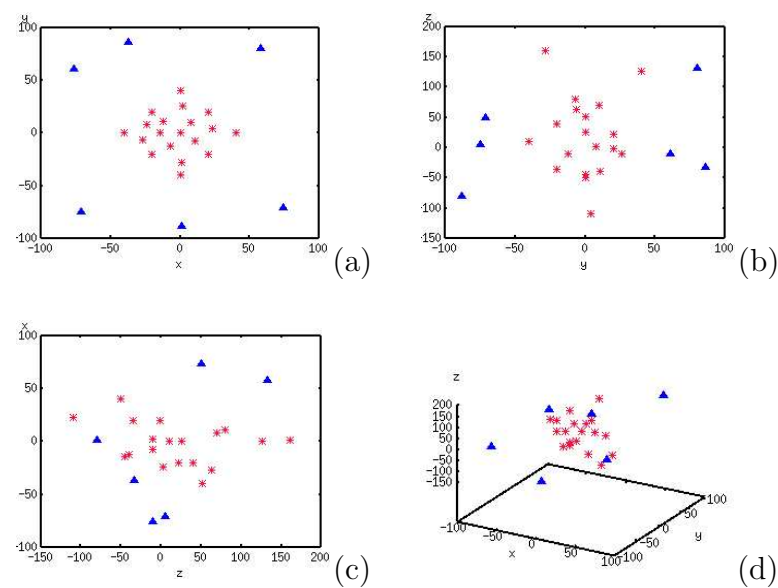


Figure 6-2: Illustration of a negative-cluster ($\|P\| = 25$): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view.

dimensions. Points grouped to the same cluster in Figure 6-2(a) do not preserve this membership in $\mathbb{R}^3$. Improperly reported clusters can cause incorrect interpretation and may mislead further explorations. Thus, spatial clustering must be able to handle three-dimensional data and even higher-dimensional data including attributes and/or time.

6.2 Determination of the control value m

We now present an evaluation of initial determination of the control value m in the two proximity graphs ($DD(P)$ and $k-UNN(P)$) under study. We will show that the heuristic applies uniformly to these proximity graphs in three dimensions. This evaluation will also provide a justification for the default value of m . The evaluation is experimental. We consider two datasets (one does not exhibit spatial clusters while the other does). These datasets are generated for illustration purpose within the unit cube $[0, 1] \times [0, 1] \times [0, 1]$. Figure 6-3 depicts the first dataset, referred as 3D-CSR (for 3-dimensional CSR). The second dataset will be called 3D-DwC (for 3-dimensional Data with Clusters). This is shown in Figure 6-4. 3D-DwC was generated from 5 randomly selected seeds within the unit cube. Points belonging to clusters are generated around the seeds within randomly chosen radii. However, 5% of the points are noise — they are uniformly drawn in the unit cube.

6.2.1 The control value m for three-dimensional Delaunay diagrams

Figure 6-3(b) and Figure 6-4(e) show the trend of the point-centric statistics when the proximity graph is the three-dimensional Delaunay diagram. Figure 6-3(b) is for 3D-CSR data while Figure 6-4(e) is for 3D-DwC.

For 3D-CSR, we see in Figure 6-3(b) that $Local_Mean(p_i)$ and $Local_St_Dev(p_i)$ cover

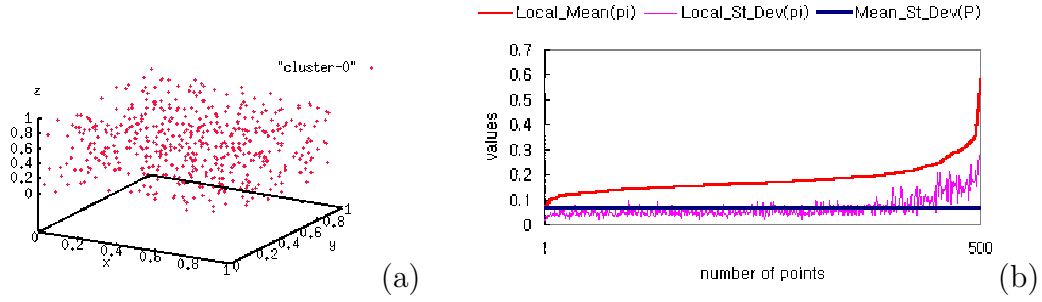


Figure 6-3: Use of the three-dimensional Delaunay diagram to analyze three-dimensional data exhibiting no clusters ($\|P\| = 500$): (a) The xyz -view; (b) Profiles of point-centric statistics for the three-dimensional Delaunay diagram with 3D-CSR.

a range with no significant features similar to the two-dimensional counterpart as shown in Figure 4-8(a). That is, lengths of edges incident to p_i exhibit relative homogeneity since no particular points are far away from others in CSR datasets. Values of $Local_St_Dev(p_i)$ fluctuate closely to the value $Mean_St_Dev(P)$. Also, the value $Mean_St_Dev(P)$ is smaller than almost all values of $Local_Mean(p_i)$ for CSR datasets (refer to Figure 4-8(a) for 2D and Figure 6-3(b) for 3D). Here, the value of $Mean_St_Dev(P)$ is greater than $Local_St_Dev(p_i)$ values of inner points. However, it is less for $Local_St_Dev(p_i)$ values of border points that are artificially introduced by the bounded study region. Thus, $Mean_St_Dev(P)$ robustly behaves as the tolerance value in three-dimensional Delaunay diagrams. Experimental results show that Short-Long clustering places all the points into the same volumetric cluster and suggests no particular aggregation for the three-dimensional CSR dataset with $m = 1$.

On the other hand, for the dataset with clusters (3D-DwC), Figure 6-4(e) shows that both, $Local_Mean(p_i)$ and $Local_St_Dev(p_i)$, increase sharply after being essentially flat for about two-third of the data points. Here, $Mean_St_Dev(P)$ clearly indicates the sharp increase in $Local_St_Dev(p_i)$ values. This ensures the value of $Mean_St_Dev(P)$ as the tolerance value for 3D-DwC. With the use of $m = 1$, Short-Long clustering identifies 5 volumetric clusters as shown in Figure 6-4(d). Interestingly, the noise (5%) creates a certain degree of homogeneity in the lengths of edges incident to noise

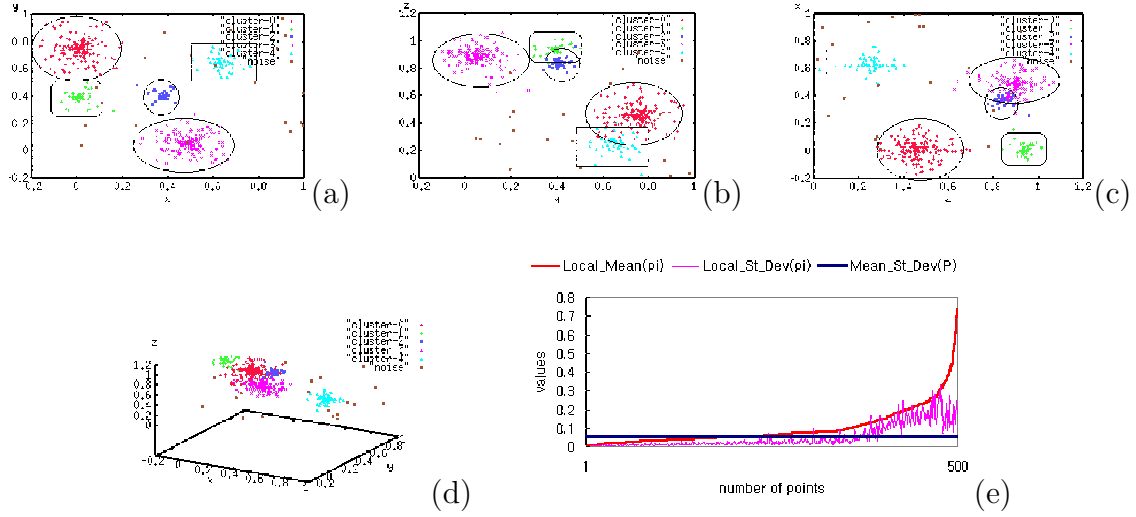


Figure 6-4: Use of the three-dimensional Delaunay diagram to analyze 3D-DWC ($\|P\| = 500$): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for the three-dimensional Delaunay diagram with 3D-DWC.

points. Thus, a decrease is observed in the plot for $Local_St_Dev(p_i)$ as depicted in Figure 6-4(e). This phenomenon is unlikely in Figure 4-8(c) and Figure 6-7(e) without noise.

Thus, we can induce that setting the value of m to 1 in the three-dimensional Delaunay diagram robustly identifies globally significant clusters for three-dimensional datasets with or without clusters.

6.2.2 The control value m for undirected k -nearest neighbor graphs

We contrast datasets 3D-DWC and 3D-CSR, but now the underlying graph is k -UNN(P). There is no rapid increase in $Local_St_Dev(p_i)$ values for 3D-CSR (Figure 6-5(b)). The values of $Local_St_Dev(p_i)$ fluctuate slightly in the proximity of the value of $Mean_St_Dev(P)$ as the values of $Local_Mean(p_i)$ grow. Here, $Local_St_Dev(p_i)$

values exhibit a profile that is essentially flat and the value of $Mean_St_Dev(P)$ represents a global trend in $Local_St_Dev(p_i)$ values. Similarly, m is set to 1 as default and the use of $Mean_St_Dev(P)$ for $k-UNN(P)$ as the tolerance value does not report any spatial concentrations, but rather assigns all the points into the same cluster.

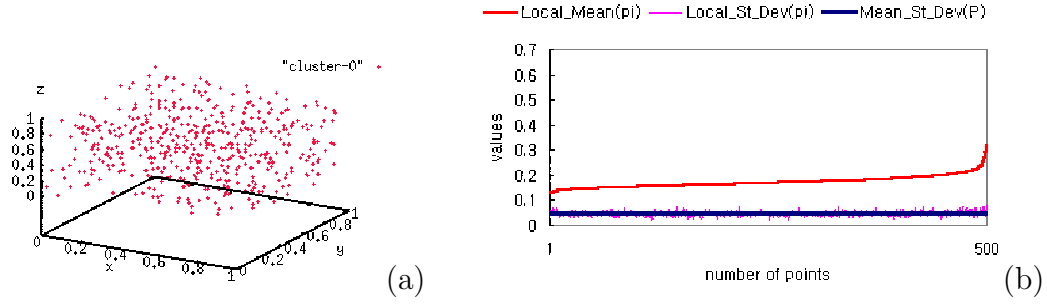


Figure 6-5: Use of 16- $UNN(P)$ to analyze 3D-CSR (P is the same as in Figure 6-3): (a) The xyz -view; (b) Profiles of point-centric statistics for 16- $UNN(P)$ with 3D-CSR.

Figure 6-6(e) shows that the distinction between inner points, and border and noise points in the values of $Local_St_Dev(p_i)$ for 16- $UNN(P)$ with 3D-DwC. The value of $Mean_St_Dev(P)$ still provides a solid distinction. The plots in Figure 6-6 show the results of clustering 3D-DwC with the $k-UNN(P)$ variant when m is set to 1. There is essentially no discernible difference between the two clustering results displayed in Figure 6-6 (the clustering obtained with $k-UNN(P)$) and Figure 6-4 (the clustering obtained with $DD(P)$), which demonstrates that Short-Long clustering is robust to the use of different families of proximity graphs in three dimensions.

The alternation of families of proximity graphs is not as smooth as imagined from the figures. In fact, we have facilitated this alternation by carefully selecting the value of k . Since the expected number of incident edges to a point in the three-dimensional Delaunay diagram is $48\pi^2/35 + 2$, which is approximately 15.5 [110], the value of k is set to 16 in the analysis. Sensitivity to the value of k will be further discussed in Section 6.4. As discussed in this section, the value $m = 1$ is set as default for $DD(P)$ and $k-UNN(P)$ regardless of types of distributions of three-dimensional data, unless

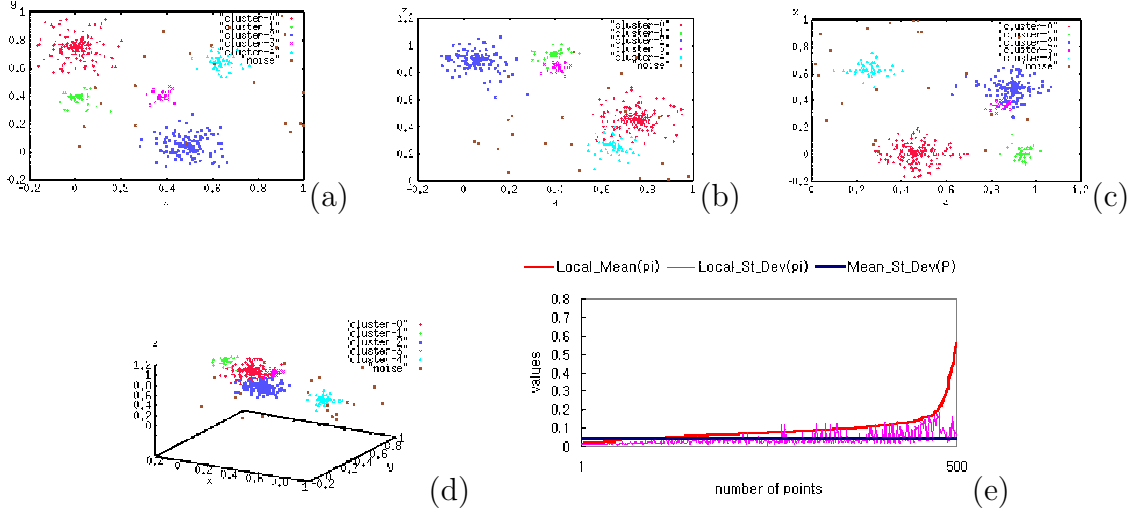


Figure 6-6: Use of 16-UNN(P) to analyze 3D-DwC (P is the same as in Figure 6-4): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for 16-UNN(P) with 3D-DwC.

otherwise noted.

6.3 Time complexity analysis

Consider a proximity graph G with $\|E\|$ edges. Then, the time complexity for the three-phase Short-Long clustering procedure with a generic proximity graph is $O(\|E\|)$ as discussed in Section 4.4.1.

6.3.1 With three-dimensional Delaunay diagrams

The construction of the three-dimensional Delaunay diagram requires $O(n^2)$ and the size of the Delaunay diagram is not bounded by $O(n)$ [109, 115]. Klee [79] shows that the number of f -faces of the Voronoi diagram for n points in a d -dimensional space is

$O(n^{\min\{d+1-f, \lceil \frac{d}{2} \rceil\}})$ for $0 \leq f \leq d$. Thus, the number of 2-faces of the Voronoi diagram for three-dimensional space is $O(n^2)$. Namely, the number of edges in the three-dimensional Delaunay diagram is quadratic. Thus, three-dimensional Short-Long clustering for three-dimensional Delaunay diagrams requires $O(n^2)$ time. However, researchers [13] have shown how to construct spanner graphs of size $O(n)$ that model the Delaunay diagram closely by the insertion of $O(n)$ data points. These spanners can be constructed in $O(n \log n)$ time making them suitable proximity graphs to be used with the Short-Long clustering method in 3D.

6.3.2 With undirected k -nearest neighbor graphs

The construction of k -nearest neighbor graph needs $O((n+k) \log n)$ time in 3D. And, the size of k -nearest neighbor graph is bounded by $O(kn)$. Now, for the applications of large spatial datasets in GIS, k can be presumed constant. We can obtain k -UNN(P) by making all edges undirected. The symmetry of the adjacency matrix can be used to compactly store k -UNN(P) in less space than the corresponding k -NN(P), which is linear anyway. Thus, three-dimensional Short-Long clustering for k -UNN(P) requires $O(n \log n)$ time.

6.4 Performance evaluation

This section summarizes experimental results with two datasets to reflect the benefits of Short-Long clustering in three dimensions, in particular, robustness to noise, clusters of different sizes, clusters of various densities and clusters of heterogeneous shapes. Figure 6-7 and Figure 6-8 show the datasets. These were also generated within the unit cube $[0, 1] \times [0, 1] \times [0, 1]$ based on a simple mixture. The first dataset named 3D-DS $\mathcal{I}$ (3-dimensional DataSet $\mathcal{I}$) has no noise (Figure 6-7) but the second named 3D-DS $\mathcal{II}$ (3-dimensional DataSet $\mathcal{II}$) exhibits 10% noise.

6.4.1 Clustering with three-dimensional Delaunay diagrams

Regardless of absence or presence of noise points, values of $Local_Mean(p_i)$ and $Local_St_Dev(p_i)$ exhibit a sharp increase as shown in Figure 6-7(e) and Figure 6-8(e). Again, a decrease is observed in the plot for $Local_St_Dev(p_i)$ values due to 10% noise for 3D-DSII. The value of $Mean_St_Dev(P)$ cuts through the plot of $Local_St_Dev(p_i)$ values when the change of magnitude in the derivative occurs. Thus, Short-Long clustering with the three-dimensional Delaunay diagram correctly identifies cluster boundaries and detects volumetric clusters in the absence and presence of noise points. Figure 6-7 depicts 5 volumetric aggregations in the absence of noise while Figure 6-8 shows 5 volumetric aggregations in the presence of noise.

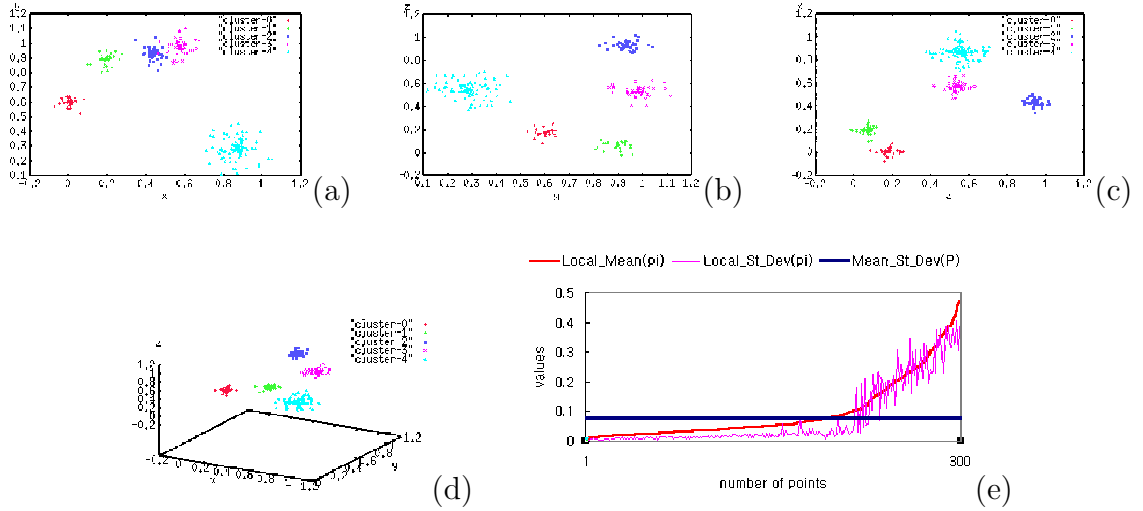


Figure 6-7: Use of $DD(P)$ to cluster 3D-DSI ($\|P\| = 300$): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for $DD(P)$ with 3D-DSI.

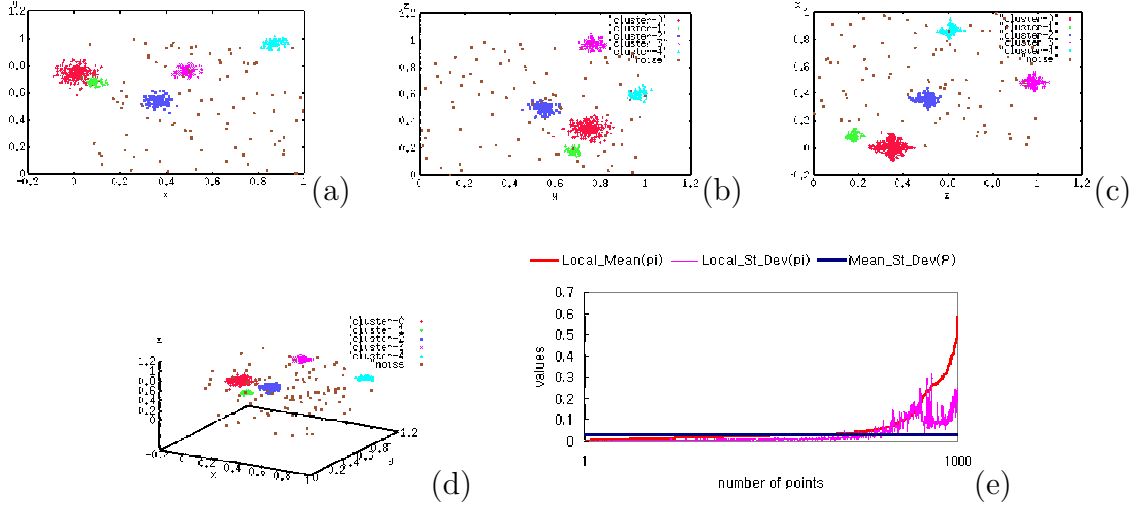


Figure 6-8: Use of $DD(P)$ to cluster 3D-DSII ($\|P\| = 1000$): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for $DD(P)$ with 3D-DSII.

6.4.2 Clustering with undirected k -nearest neighbor graphs

The proximity graph $k\text{-UNN}(P)$ is argument-specific. A change of the value of k results in a rather different proximity graph. Thus, we evaluate the clustering results with $k = 8$ and $k = 16$ in order to analyze the sensitivity of clustering with respect to the values of k .

Note that, the plots in Figure 6-9(e) and Figure 6-10(e) exhibit similar patterns. Firstly, values of $Local_St_Dev(p_i)$ in $8\text{-UNN}(P)$ and $16\text{-UNN}(P)$ exhibit the same variability, fluctuating slightly around $Mean_St_Dev(P)$. This is because for the spatially aggregated data without noise, all the points have neighbors close to them in k -nearest neighbor modeling. Secondly, $Local_Mean(p_i)$ is far greater than $Local_St_Dev(p_i)$ and thus it is greater than $Mean_St_Dev(P)$ for most points. This results in narrow acceptable range for $Other_Edges(p_i)$. Thus, few points slightly apart from clusters are left as noise. Note that, clustering with the three-dimensional Delaunay diagram groups those points. In the Delaunay diagram, Delaunay edges

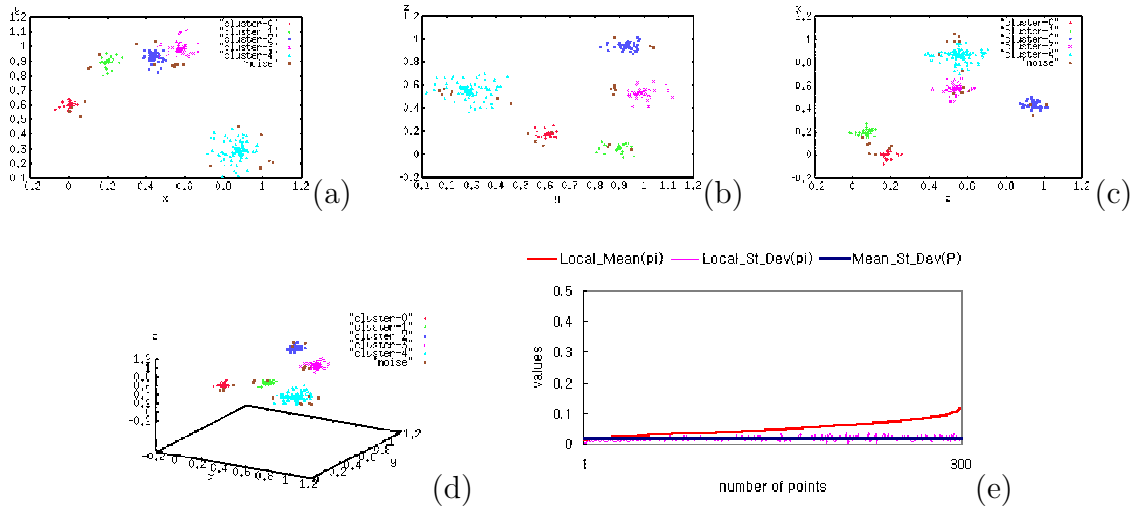


Figure 6-9: Use of 8- $UNN(P)$ to cluster 3D-DSI (P is the same as in Figure 6-7): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for 8- $UNN(P)$ with 3D-DSI.

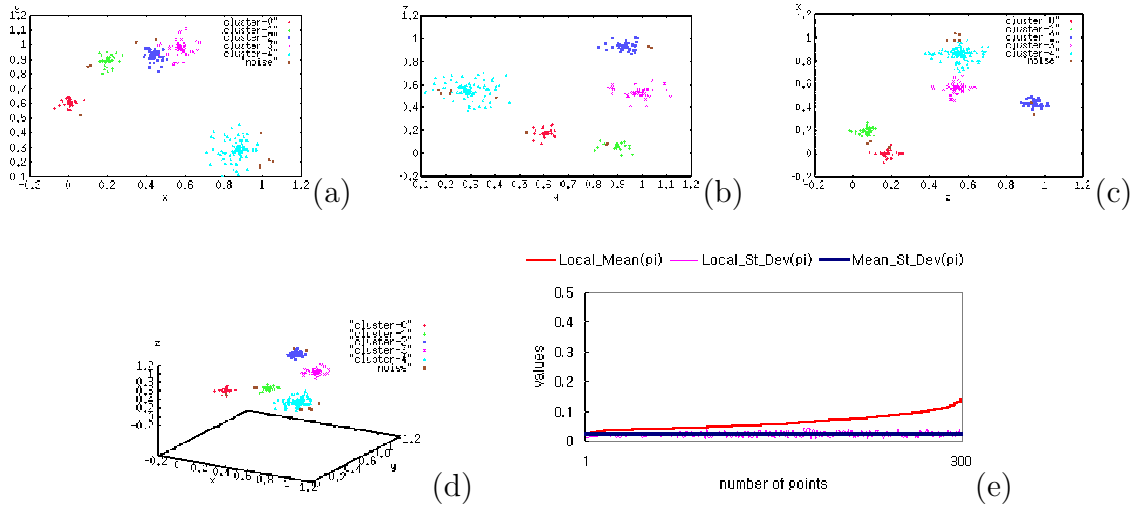


Figure 6-10: Use of 16- $UNN(P)$ to cluster 3D-DSI (P is the same as in Figure 6-7): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for 16- $UNN(P)$ with 3D-DSI.

between clusters increase $Mean_St_Dev(P)$. The increase widens the acceptable range for $Other_Edges(p_i)$ enough to group points lying slightly apart from clusters. Thirdly, $Local_Mean(p_i)$ grows steadily.

Figure 6-9 and Figure 6-10 show that Short-Long clustering with the default control value for $k-UNN(P)$ variants identifies 5 spatial concentrations regardless of changes in values for k . Figure 6-9 shows clusters detected by $8-UNN(P)$ while Figure 6-10 depicts clusters detected by $16-UNN(P)$, respectively. This demonstrates the robustness of three-dimensional Short-Long clustering. This multi-purpose boundary-based clustering is not sensitive to variations of the user-supplied value of k (as long as k is not extreme, for example $k = 1$ would result in poor clustering). This is in contrast to traditional clustering as discussed in Section 3.1. Since three-dimensional Short-Long clustering derives the point-centric statistics from the underlying proximity graph (and not from users), it is data-driven and robust to changes in values for k . Thus, Short-Long clustering preferably infers clusters from data, rather than from the user's parameter settings.

Again, the two plots shown in Figure 6-11(e) and Figure 6-12(e) show similar patterns. However, these patterns are different from those shown in Figure 6-9(e) and Figure 6-10(e), but similar to that shown in Figure 6-8(e). The values of $Local_Mean(p_i)$ and $Local_St_Dev(p_i)$ increase quickly indicating the distinction between noise points and points belonging to clusters. The value of $Mean_St_Dev(P)$ robustly indicates the distinction. Thus, Short-Long clustering identifies 5 volumetric clusters with $k-UNN(P)$ variants with the control value $m = 1$ and in the presence of noise. Three-dimensional Delaunay diagrams produce good quality clusters in the absence and presence of noise while $k-UNN(P)$ works better in the presence of noise. This is because incident edges to noise will reasonably increase $Mean_St_Dev(P)$ in $k-UNN(P)$.

In summary, experiments show three-dimensional Short-Long clustering based on proximity graphs (the three-dimensional Delaunay diagram and the $k-UNN(P)$) is robust and able to detect clusters in the presence of noise, clusters of different densities

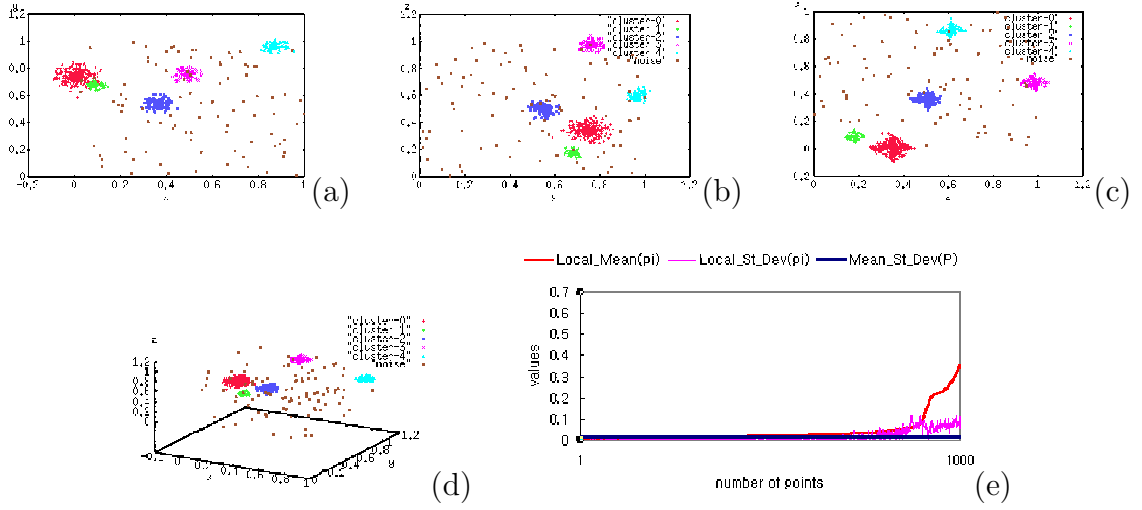


Figure 6-11: Use of 8- $UNN(P)$ to cluster 3D-DSII (P is the same as in Figure 6-8): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for 8- $UNN(P)$ with 3D-DSII.

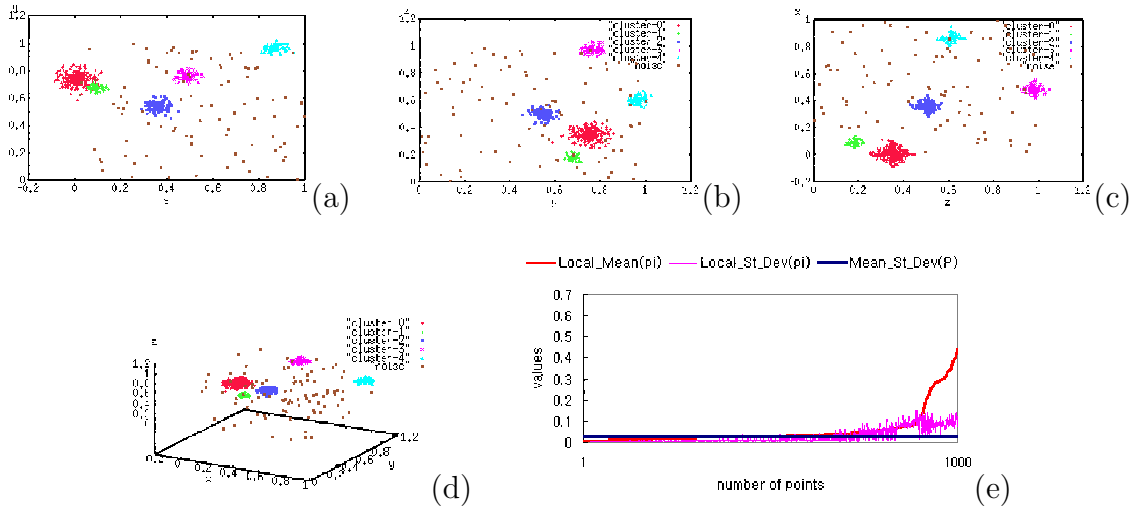


Figure 6-12: Use of 16- $UNN(P)$ to cluster 3D-DSII (P is the same as in Figure 6-8): (a) Projection into the xy -plane; (b) Projection into the yz -plane; (c) Projection into the zx -plane; (d) The xyz -view; (e) Profiles of point-centric statistics for 16- $UNN(P)$ with 3D-DSII.

and clusters of different sizes. Further, the clustering method is not restricted to globular clusters, but able to identify clusters of arbitrary shapes.

6.5 Remarks

Spatial clustering approaches have previously focused on two dimensions. Ignoring a third dimension results in imprecise clustering for 3D data. Geoinformation studies, such as minefields in underground databases, fishes in marine databases and stars in astronomy databases need three-dimensional clustering to identify complex volumetric clusters. Clustering algorithms from the data mining literature, although applicable in high dimensional spaces produce imprecise results for complex geo-referenced clusters due to the ignorance of the special characteristics of geoinformation discussed in Section 3.1.

This chapter investigates Short-Long clustering for three-dimensional point data based on discussions made in Section 5.3. Experimental results demonstrate that the argument-free modeling for three-dimensional point data, Delaunay diagram, encodes sufficient proximity information. Thus, it is best-suited for Short-Long clustering where precise proximity information is crucial and high-quality of clusters are essential. However, Short-Long clustering with the three-dimensional Delaunay diagram requires $O(n^2)$ time and space, which will be problematic for mining massive datasets. Section 5.3 discusses the applicability of Short-Long clustering to the large family of proximity graphs including $k\text{-}UNN(P)$ to obtain satisfactory clustering. The $k\text{-}UNN(P)$ is subquadratic in time and linear in space for any dimensions if we assume k is constant, which is the case in data mining. This makes it popular in higher dimensions.

Experiments in Section 5.3.2 demonstrate that Short-Long clustering with $k\text{-}UNN(P)$ robustly identifies various types of clusters and approximates Short-Long clustering with $DD(P)$. Not surprising, three-dimensional Short-Long clustering with $k\text{-}$

$UNN(P)$ identifies quality volumetric clusters. Most surprisingly, three-dimensional Short-Long clustering with k - $UNN(P)$ generates consistent quality clustering results with little sensitivity to the variation of k values. This is clearly better than previous traditional semi-automatic clustering where different clustering results are obtained for slightly modified user-supplied arguments. Thus, three-dimensional Short-Long clustering reduces exploration time to find best-fit values for arguments.

This chapter now opens the possibility to extend Short-Long clustering to higher dimensions that incorporate attributes and/or time. Short-Long clustering with k - $UNN(P)$ only requires $O(n \log n)$ time for higher dimensions (the dimension parameter appears as a constant under the O notation).

Chapter 7

Multi-level Short-Long Clustering

This chapter discusses hierarchical decompositions of data with Short-Long clustering (referred as multi-level Short-Long clustering). First in Section 7.1, we provide background on multi-level Short-Long clustering. This aims at revealing the cluster hierarchy of datasets. Then, we propose two multi-level clustering algorithms in Section 7.2. In Section 7.3, we introduce definitions to explain the working principle of multi-level Short-Long clustering. In Section 7.4, we introduce a new visualization tree that robustly reveals the cluster hierarchy. In this section, we show that traditional visualization approaches are neither suitable nor applicable for data-rich environments and claim that our approach better explains the cluster hierarchy and is suitable for GDM. Section 7.5 shows the computational requirements of multi-level Short-Long clustering. It requires subquadratic time and linear space. In Section 7.6, we illustrate the effectiveness of our clustering.

7.1 Multi-level decompositions

In the previous chapters, we have considered single-level clustering (referred as single-level Short-Long clustering). However, one of the unique characteristics of geo-

referenced data is its generic hierarchical complexity. Clusters may contain several numbers of subclusters. These subclusters may consist of smaller subclusters. This kind of hierarchy can be easily found in the real world due to its hierarchical nature. For instance, a big city is naturally surrounded by several small cities and may contain a high-density center. Each small city may contain a number of towns. Therefore, a robust exploratory clustering method should be able to report all possible clusters and to reveal a cluster hierarchy to help users explore and examine patterns at different geographical scales. Because of the hierarchical nature of geo-referenced data, partitioning an initial set of points into mutually exclusive and collectively exhaustive groups with similar and homogeneous characteristics is insufficient to understand its complex structure. Instead, multi-level (hierarchical) clustering is more informative in some spatial settings [125].

Figure 7-1 illustrates this. Intuitively, we can recognize that there are three large-scale clusters in Figure 7-1(a): a cluster of circles, a cluster of triangles and a cluster of rectangles. Each large-scale cluster is composed of small-scale subclusters. For instance, the cluster of circles consists of three subclusters while the cluster of triangles consists of two. A weighted tree graph in Figure 7-1(b) reveals this hierarchy. With the aid of this tree graph, users not only recognize the total number of clusters, but also grasp easily the hierarchical structure of the data. Single-level clustering may produce either three large-scale clusters or nine small-scale clusters for the dataset. In either case, some structural information is lost. In the former case, nine small-scale clusters can not be found. Hence, a later search for associations may miss a relationship. If we assume that points represent Asian shops, then we may miss that the regular shapes are the result of street boundaries forming blocks (Figure 7-1(c)). In the case of finding only the high level clusters, the analysis will miss the correct relationship between Asian shops and suburbs boundaries (Figure 7-1(d)). Single-level Short-Long clustering reports the three large-scale clusters with $m = 1$, thus misses the nine small-scale clusters. Each cluster is of potential interest. Therefore, missing a cluster can cause incomplete analysis.

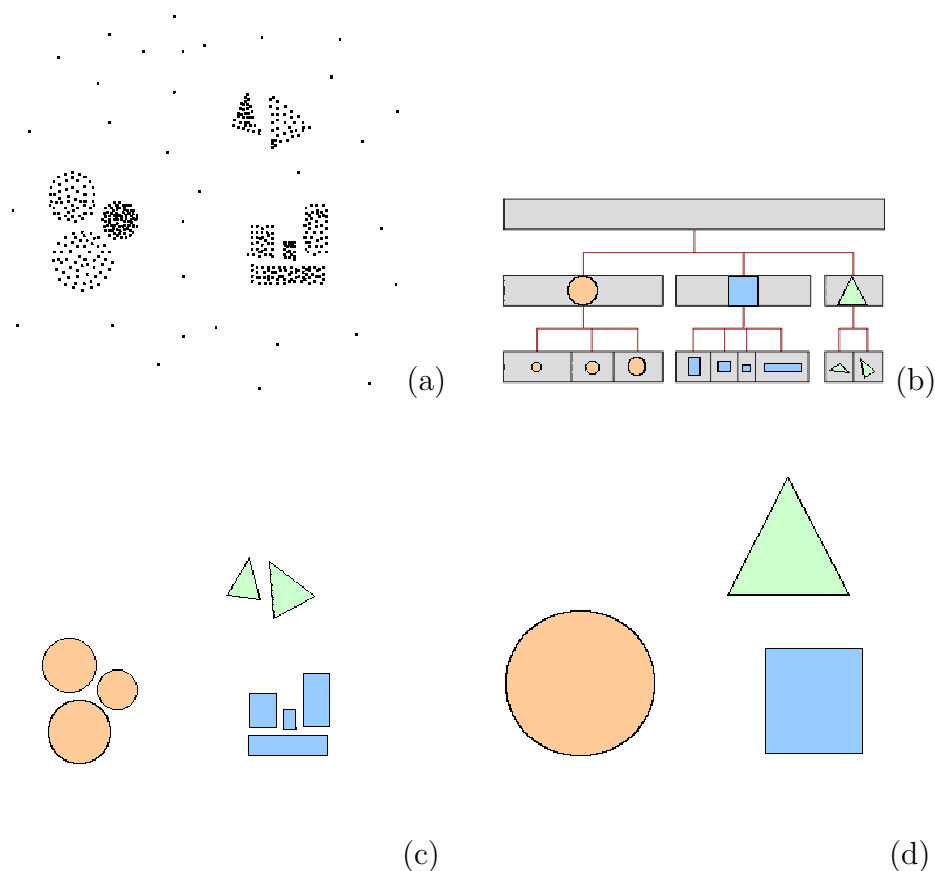


Figure 7-1: Point dataset representing an inherent hierarchical data structure:
 (a) Point dataset representing Asian shops ($\|P\| = 460$); (b) A weighted dendrogram explaining the cluster hierarchy; (c) Boundary data representing blocks;
 (d) Boundary data representing suburbs.

On the other hand, multi-level Short-Long clustering produces thirteen clusters for the dataset (the thirteen shaded rectangles in Figure 7-1(b)). These are: a single cluster forming the whole dataset, three large-scale clusters and nine small-scale clusters. Now, we can successfully explore and navigate the dataset with the aid of a cluster hierarchy (Figure 7-1(b)). Now, these multi-level clusters could be tested for possible associations.

7.2 Intuition and algorithms

An intuitive notion of what is a cluster (at least for graph-based clustering) is usually based on connected components as discussed in Section 2.6. These connected components may be further decomposed into subclusters in hierarchical clustering methods unless they are homogeneous. In traditional hierarchical clustering [58, 59, 76, 141, 142, 151], user-supplied global arguments determine termination or merge conditions. However, these arguments are not easily determined without prior knowledge that hinders exploration [33]. Multi-level Short-Long clustering overcomes this difficulty. As discussed in Section 4.3, Section 5.2.2 and Section 6.2, single-level Short-Long clustering does not detect any spatial concentrations for CSR datasets with $m = 1$ in 2D or 3D in the Minkowski metric, but rather assigns P to the same cluster. Thus, the termination condition is met in multi-level Short-Long clustering when a set of points does not exhibit any spatial aggregations, but rather exhibits point-centric statistics similar to CSR. Therefore, multi-level Short-Long clustering does not require a termination condition from users, but automatically derives it from data.

We now propose and discuss two approaches for multi-level Short-Long clustering.

1. Reconstruction based approach.

This approach first applies single-level Short-Long clustering to the whole dataset P to extract globally significant clusters. And then, it recomputes the Delaunay diagram for each cluster and recursively applies single-level Short-Long clustering to the cluster. That is, P is decomposed into subsets $(C_1, C_2, \dots, C_k)$ and noise after the initial clustering. For each subset C_i ($1 \leq i \leq k$), this approach reconstructs $DD(C_i)$ and applies single-level Short-Long clustering to $DD(C_i)$.

Although this approach provides reasonable grounds for multi-level clustering, it has several drawbacks. First, it requires extra time for recomputing the Delaunay diagram for each cluster. Second, recomputing $DD(C_i)$ is not affected by the distribution of points $p \in P \setminus C_i$. Since the cluster C_i is a subset of P , it

is correlated to every point in P . Third, recomputing independent $DD(C_i)$ may cause some non-informative interactions (edges) that do not exist in $DD(P)$. Especially, this is the case when the cluster C_i contains other clusters. Some edges in $DD(C_i)$ may go over the surrounded clusters. This is illustrated in Figure 7-2. Let C_i be the “8-like” sparse cluster. Figure 7-2(a) depicts $DD(P)$. Figure 7-2(b) shows a subgraph of $DD(P)$ after the initial clustering. $DD(C_i)$ illustrated in Figure 7-2(c) contains Delaunay edges running across the dense clusters. These edges are non-informative in $DD(P)$. In order to overcome these drawbacks, we propose another approach that is based on valid Delaunay edges recuperation.

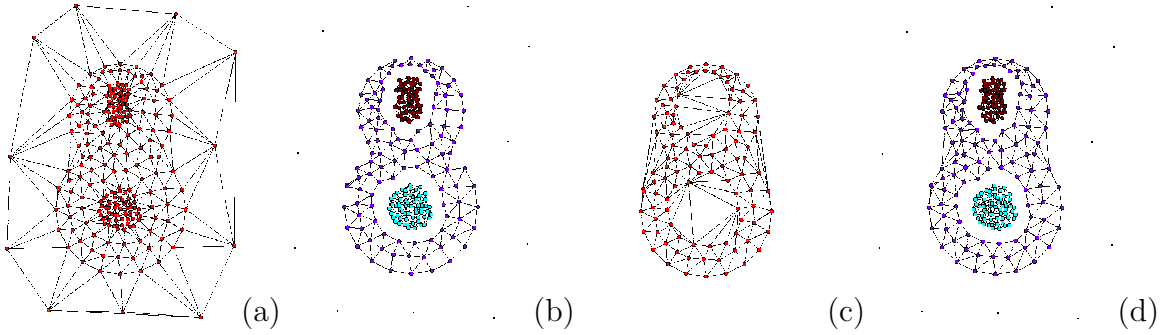


Figure 7-2: Effect of two approaches to delineate multi-level clusters: reconstruction *vs* recuperation ($\|P\| = 250$): (a) A dataset with $DD(P)$; (b) After single-level Short-Long clustering (3 clusters); (c) Reconstruction of the Delaunay diagram of a sparse cluster; (d) Recuperation of the Delaunay edges of the sparse cluster.

2. Recuperation based approach.

Similar to the previous approach, it initially applies single-level Short-Long clustering to P to extract significant clusters in the context of R . Since each edge $e_{i,j} = (p_i, p_j)$ represents a valid interaction when $CC[p_i] = CC[p_j]$ after the clustering, this edge must be recuperated for the next level of clustering. Thus, this approach recuperates all Delaunay edges in $DD(P)$ when their two end-points belong to the same cluster after the clustering. Once these intra-

cluster edges are recuperated, then single-level Short-Long clustering is applied to each connected components. Figure 7-2(d) depicts a recuperated subgraph of $DD(P)$ after the clustering.

To summarize these algorithms, we provide multi-level Short-Long clustering in pseudo-code.

Algorithm Multi-level Short-Long clustering

```

Input: A graph  $G$  of  $P$  and the control value  $m$ ;
Output: Multi-level clusters with a cluster hierarchy;
1) begin
2) WriteCluster( $G$ );
3)  $Mean\_St\_Dev(P) \leftarrow \text{ComputeMeanStDev}(G)$ ;
4)  $outG \leftarrow \text{ThreePhaseClustering}(G, m, Mean\_St\_Dev(P))$ ;
5) ComputeConnectedComponents( $outG$ );
6) for each connected component  $CC$ 
7)   if ( $\frac{\|CC\|}{\|P\|} \geq NoiseRatio \ \&\& \ \|CC\| \neq \|G\|$ )
8)   then
9)     {
10)      [v1]  $subG \leftarrow \text{BuildDelaunayDiagram}(CC)$ ;
10)      [v2]  $subG \leftarrow \text{RecuperateDelaunayDiagram}(CC)$ ;
11)      Call Multi-level Short-Long clustering with the  $subG$ ;
12)    }
13) end for
14) end

```

The algorithm for multi-level Short-Long clustering is based on recursion. The initial input of the algorithm is $DD(P)$. The process labeled **WriteCluster**(G) reports all points in G as a cluster. The process labeled **ComputeConnectedComponents**($outG$) (Step 5) computes connected components of the output graph $outG$. The label

NoiseRatio (Step 7) represents the default noise ratio. Note that, the default noise ratio is set to 1% as mentioned in Section 4.5. Step 9 has two versions with a version identifier. One or the other version is to be used. The reconstruction based approach uses Step 9.v1 while the recuperation based approach uses Step 9.v2. The process labeled `BuildDelaunayDiagram(CC)` builds a new Delaunay diagram with points in connected component CC while the process `RecuperateDelaunayDiagram(CC)` recuperates intra-cluster edges in CC . Based on discussion made earlier in this section, the recuperation based approach is set as default and subsequent sections will be based on this approach.

7.3 Definitions

The default divisive clustering process starts at the top level (level zero). Each level labels edges as either “active” or “passive”. Active edges at level k are intra-cluster edges in that level while passive edges at level k are inter-cluster edges in that level. Only active edges are used for split to create subclusters in the next level. Here, a cluster C_i (recall that it is a connected component) is passive at level k if an attempt to subdivide it fails. That is, the application of single-level Short-Long clustering to C_i does not find any subclusters (C_i remains a connected component).

The following definitions formalize the discussion above to present the working principle of multi-level Short-Long clustering.

Definition 7.3.0.1 *Given a level k in the hierarchy, a Delaunay edge $e_{i,j} = (p_i, p_j) \in DD(P)$ is a passive edge in the level k if and only if $CC[p_i] \neq CC[p_j]$ in that level.*

Passive edges are removed from the proximity graph for the next level, since they are of no interest any more. Thus, they are no longer used for further clustering. For instance, edges between the large-scale clusters are eliminated in Figure 7-1(a) after the first level of clustering. These edges removed are passive edges in the first level

(level 1). These passive edges represent weak interactions that are not significant in that level.

Definition 7.3.0.2 *Given a level k in the hierarchy, a Delaunay edge $e_{i,j} = (p_i, p_j) \in DD(P)$ is an active edge in the level k if and only if $CC[p_i] = CC[p_j]$ in that level.*

Active edges and points incident to them form a new proximity graph in each level of the hierarchy. The newly created proximity graph is a subgraph of the previous graph and is used for detecting subclusters. For instance, all edges between the small-scale clusters within the same large-scale cluster in Figure 7-1(a) are active edges in the first level. Some of them will be passive edges in the next level, while others still remain active. Edges surviving from the previous level represent strong interactions influencing the formation of subclusters.

Definition 7.3.0.3 *A path in the proximity graph of a given level k is an active path if and only if every edge in the path is an active edge in the level k .*

Definition 7.3.0.4 *Given a level k , a cluster is a passive cluster in the level k if and only if there still exists an active path between any pair of points in the cluster at level $k+1$ (after a further splitting process of multi-level Short-Long clustering).*

A splitting process of multi-level Short-Long clustering is the classification of edges into interesting (formally active edges) and uninteresting (passive edges). Passive clusters are the leaves of the clustering hierarchy (for instance, small-scale clusters in Figure 7-1(b)) because their points are in one connected component before and after the splitting process. Actually, because of the statistical considerations in multi-level Short-Long clustering, there is no large difference among the strength of interactions amongst points in a passive cluster. That is, passive clusters exhibit point-centric statistics similar to CSR.

Definition 7.3.0.5 *Given a level k , a cluster is an active cluster in the level k if and only if there does not exist an active path between all pairs of points in the cluster at level $k+1$ (after a further splitting process of multi-level Short-Long clustering).*

Here, a cluster is considered to have significantly varying internal interactions. The edge split is considered to generate meaningful subclusters. Thus, further split is required to reveal the structure of active clusters.

7.4 Visualization with weighted dendrogram

Dendrograms display the hierarchy of clusters in hierarchical clustering [5]. However, as the number of point increases, the depth of the tree becomes longer. This not only reduces readability, but requires more space. In addition, it is difficult to visualize the proportion of noise at each level. Moreover, the relative size of a cluster is not easily recognized. It is also hard to compare the significance of either intra-level clusters or inter-level clusters. For these reasons, dendrograms do not represent well the hierarchical structure revealed by multi-level Short-Long clustering. These drawbacks are resolved by incorporating weights to clusters in dendrograms. Weights are the number of points in a cluster. In this weighted diagram, rectangles represent clusters. An example is given in Figure 7-1(b). The length of a rectangle in the weighted diagram is proportional to the corresponding weight. We call these diagrams Pendrograms¹. These Pendrograms will illustrate when early stops (passive clusters that exhibit point-centric statistics similar to CSR) occur for EDA. Also, recall that a group of points whose size is greater than 1% of the total number of points is considered as a cluster in this thesis (Section 4.5). Thus, all passive clusters and some active clusters (whose subclusters have their sizes less than the default noise ratio) form leaves in Pendrograms. With the aid of Pendrograms, users can easily

¹Pendrogram denotes a weighted dendrogram in this thesis; from the Latin pendro (meaning to be weighed) and dendrogram.

count both the total number of clusters and the number of intra-level clusters at each level.

For instance, Figure 7-1(b) suggests 13 clusters in total for the dataset shown in Figure 7-1(a), 1 cluster that includes the whole dataset, 3 clusters in the first level and 9 clusters in the second level. Analogies and comparisons between clusters can easily reveal the relative significance of clusters. The proportion of noise creates a gap (or mismatch) at each level, which equally separates intra-level clusters. As depicted in Figure 7-1(b), noise is significant after multi-level Short-Long clustering of the first level (a logical division into active edges and passive edges). Thus, clusters in the first level are separated with a certain gap. On the other hand, splits of the large-scale (first level) clusters generate no noise. Therefore, the small-scale (second level) clusters are presented without gaps. Pendrograms have a number of merits over traditional dendrograms. They enhance readability. In addition, they suggest the distribution of interactions within clusters. Interactions in the small-scale clusters are considered as homogeneous in the dataset shown in Figure 7-1(a), since they are leaves in the Pendrogram. Meanwhile, interactions within the large-scale clusters are to be considered heterogeneous, since they have children. Further, Pendrograms make it possible to compare or contrast between intra-level clusters or between inter-level clusters. Obviously, these are much simpler and illustrative than dendrograms. An experimental comparison will be presented in Section 7.6.

7.5 Time complexity analysis

First note that, the preprocessing to create the proximity graph is not replicated at each level. Multi-level Short-Long clustering repeatedly calls itself until the termination condition is met. For time complexity analysis, we examine first the total time complexity at each level of the hierarchy. At the root level of the clustering hierarchy, the input number of edges is $O(n)$. At each level of hierarchy, we have a remaining proximity graph that is bounded by $O(n)$. Thus, multi-level Short-Long clustering

requires $O(n)$ time at each level. The structure of the splitting process of multi-level Short-Long clustering is similar to that of binary search, removing a constant fraction of the edges at each level. The Short-Long clustering criteria have a main component that is the local mean of the current active edges incident to each point at this level. Since the mean is an estimator of the median, the criterion of $Other_Edges(p_i)$ (Definition 4.2.2.7) is very likely to be close to the median of the lengths of active edges incident to p_i at the current level. Thus, the expected depth (the number of levels) is expected to be $O(\log n)$. Consequently, the time complexity of the proposed clustering is $O(n \log n)$ expected time. This compares extremely favorably with other clustering methods for spatial clustering and specially other hierarchical clustering algorithms that require $O(n^2)$ time [107].

7.6 Performance evaluation

This section analyzes experimental results of multi-level Short-Long clustering with a real dataset that exhibits a generic hierarchical structure. Experimental results will highlight the merit of multi-level Short-Long clustering over the single-level counterpart.

We use living areas around Queensland in Australia as our first dataset. Despite of the complex nature of the real dataset, multi-level Short-Long clustering discovers all possible clusters hidden in the dataset. It successfully derives clusters directly overlying the major cities, suburbs and urban area in the dataset. For instance, a cluster identified in the first level as shown in Figure 7-3(b), directly overlays the main urban area. Note that, single-level Short-Long clustering only reports the first level of clusters. The active cluster is further partitioned into 5 clusters in the next level. Figure 7-3(c) depicts these 5 second level clusters. Among them, four are active clusters and one is a passive cluster. Active clusters are further split into subclusters in the next level. This cluster hierarchy is illustrated in Figure 7-3(j). The Pendrogram helps users navigate and explore the hierarchy of the dataset with

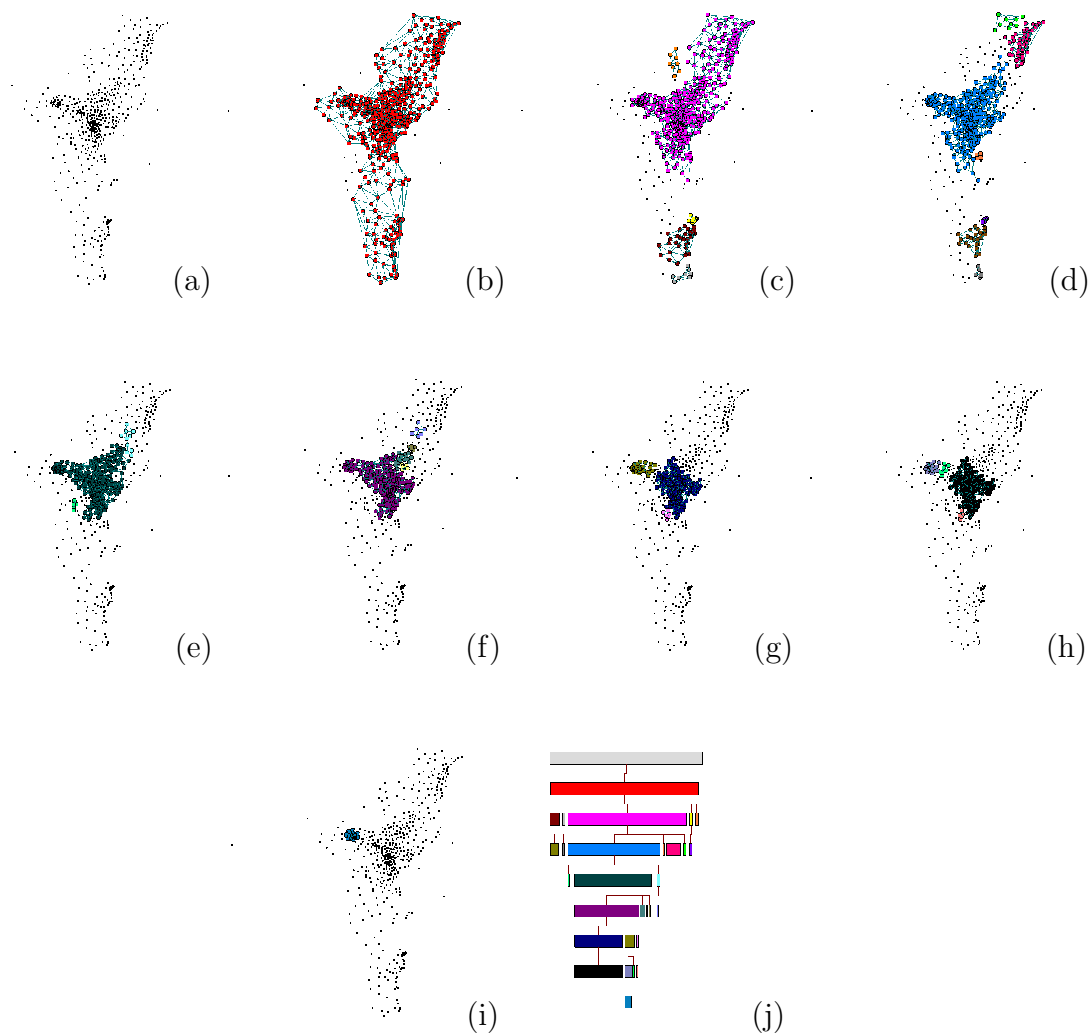


Figure 7-3: Multi-level Short-Long clustering applied to a real dataset ($\|P\| = 494$): (a) Living areas around Queensland in Australia; (b) First level clustering (1 cluster); (c) Second level clustering (5 clusters); (d) Third level clustering (7 clusters); (e) Fourth level clustering (3 clusters); (f) Fifth level clustering (5 clusters); (g) Sixth level clustering (3 clusters); (h) Seventh level clustering (4 clusters); (i) Eighth level clustering (1 cluster); (j) Pendrogram.

ease. An application is designed and implemented in such a way that users can simply click over any rectangle (cluster) in the Pendrogram to highlight its corresponding cluster. Typically, users are more interested in relatively stronger clusters. Clusters with more points are easily selected by simply choosing relatively larger rectangles in the Pendrogram.

This dataset also allows us to illustrate the advantages of our hierarchical Pendrograms over the results of traditional hierarchical clustering and dendrograms. The depiction and interpretation of dendrograms is difficult [5, 64]. Dendrograms do not identify groups, and fail to summarize or generalize the dataset (being as large as the original input). Their results are difficult to incorporate into a data mining process, like spatial-dominant generalization. Figure 7-4 depicts the dendrogram of single-linkage for the small dataset shown in Figure 7-3 (only 494 data points). Even eliminating branches with 1% or less points, the dendrogram does not reveal anything about the data. The dendrogram here was produced with the gnu-licensed package R ². There is no doubt that the Pendrogram shown in Figure 7-3(j) is much more readable and informative.

7.7 Remarks

Multi-level clusterings report all possible clusters within P that may convey significance for further explorations. In this chapter, we propose an effective and efficient divisive multi-level clustering that robustly reveals a cluster hierarchy. This section demonstrates the benefits of our approach over traditional hierarchical clustering approaches. Multi-level Short-Long clustering does not require a termination condition, but rather derives it from data. If a subset exhibits point-centric statistics similar to CSR, multi-level Short-Long clustering leaves it as a single cluster. That is, it declares the subset as a passive cluster. This is the termination condition used in multi-level Short-Long clustering. Note that, traditional hierarchical clustering re-

²For program and documentation see www.R-project.org.

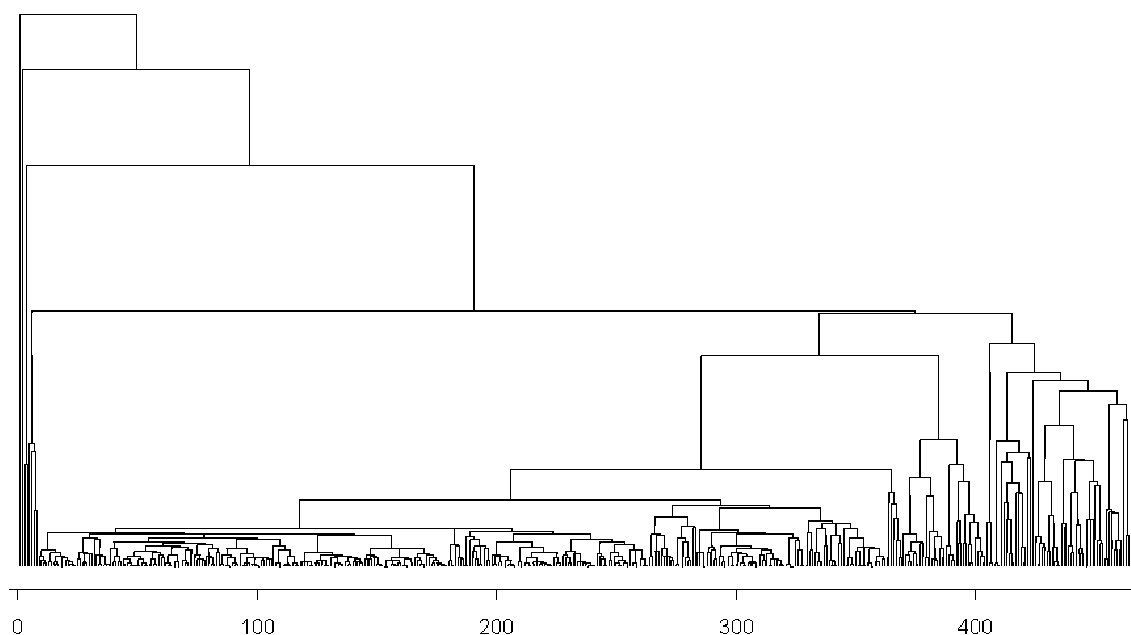


Figure 7-4: The dendrogram of single-linkage hierarchical clustering using the $MST(P)$ (P is the same as in Figure 7-3).

quires a user-supplied merge or termination condition such as a desired number of clusters or the distance between the two closet clusters as discussed in Section 2.4.1. Thus, clustering is very much guided by the user in traditional hierarchical clustering methods.

The other well-known problem of hierarchical clustering is its non-scalability. Typically, hierarchical clustering algorithms require quadratic time [107] and suffer from several trial-and-error steps to examine and evaluate a good number of clusters that are laborious and inefficient [61]. On the other hand, multi-level Short-Long clustering is well-suited for EDA and GDM in data-rich environments since CPU time requirements remain subquadratic and does not necessitate the laborious trial-and-error steps to determine the best termination condition.

Visualization of cluster hierarchy revealed by hierarchical clustering is another difficult task. This chapter discusses the drawbacks of dendrogram and proposes a variant

of dendrogram called Pendrogram. The weight associated dendrogram enhances readability, conveys more information, requires a small space and is suitable for data-rich environments.

Chapter 8

Cluster Polygonization for Multivariate Associations

Clustering provides answers for “where” and suggests leads into “why” for post-clustering explorations. However, post-clustering processes have attracted less attention than clustering itself despite of their importance. This section investigates cluster polygonization into mining multivariate associations using cluster analysis. In Section 8.1, we discuss possibilities of extracting shapes of clusters and propose an automatic cluster polygonization algorithm. In Section 8.2, we develop mining multivariate associations techniques based on the cluster polygonization. Here, we emphasize that our approaches do not necessitate concept hierarchies. These hierarchies require domain-specific knowledge, but our approaches utilize cluster structures revealed from data.

8.1 Cluster polygonization

One popular way of defining topological relationships between spatial objects is to use the set-oriented relationships based on boundary and interior [119]. Intersection

between interiors and boundaries of spatial objects defines all possible relationships (meets, overlaps, inside and covers which are known as the 4-intersection relationships [148]) in the Euclidean plane. Since intersection is a robust operation for defining relationships, it is a solid candidate for detecting associations. Thus, we base our multivariate association mining on intersection. Point and line spatial objects will be transformed to area objects. A line spatial object can be easily transformed to an area spatial object by “buffering” operation that creates area containing locations within a given range [148]. Short-Long clustering has summarized P into a set of clusters. Thus, what is required is a fast and robust cluster-to-area transformation for the proposed multivariate association mining. In this section, we propose an automatic cluster polygonization algorithm that is well-suited for mining massive datasets.

8.1.1 Shapes of clusters

The first process of cluster polygonization is to identify shapes of clusters (equivalently boundaries). If shapes of clusters for P match with a particular feature data [80], then clusters exhibit a high positive association with the feature. However, it is difficult to extract the shape and still there is no method that can be considered as an absolute winner since most approaches are application-dependent [110].

The smallest bounding rectangle is an intuitive way of representing the shape of a cluster $C_i \subset P$. However, it is too crude to capture details of the shape of C_i . The convex hull $CH(C_i)$ of C_i is another simple method. Since the $CH(C_i)$ is the smallest convex set containing C_i , it can be a natural choice. In addition, it is unique for C_i (this is also the case with the smallest bounding rectangle representation). Fast and unique models for shape extraction are useful for data mining where parameter-tuning is laborious and time consuming. However, again the $CH(C_i)$ is too crude to capture details of the shape of C_i unless the points are all arranged in a convex shape (and then, the vertices of $CH(C_i)$ coincide with C_i , which does not help much in summarizing C_i). The α -shape [30] (this is a generalization of $CH(C_i)$) overcomes

the crudeness of $CH(C_i)$. The value of the real number α controls the desired level of detail. The set of α values leads to a whole family of shapes capturing from crude to fine shapes of C_i . Boundary Shape Matching (BSM) [80] uses the α -shape to explore associations among layers.

Although the α -shape allows users to explore the spectrum of shapes of points, several problems arise when the α -shape is used for detecting spatial cluster boundaries in data-rich environments.

1. It is difficult to determine the best value α that produces a neither too crude nor too fine shape [80].
2. Several trial-and-error steps are necessary for tuning the value of α . In data-rich environments, this is laborious and time consuming.
3. If P contains k clusters, then we need to tune α for as many as k times in order to find the best shape of each cluster. Since clusters in P require different values for α , the independent manipulation of values of α for clusters belonging to the same layer not only increases required time but incorporates human bias.
4. The α -shape of a cluster $C_i \subset P$ is not affected by the distribution of points in $P \setminus C_i$. Clusters are truly the result of the peculiarities in the entire distribution sampled by P . Thus, although a point in $P \setminus C_i$ does not belong to the cluster C_i , it has some effect on the shape of C_i . Note that, shape is a matter of sets of points and everything is related to everything else in geographical settings [135] as discussed in Section 3.1.

In this section, we propose an automatic boundary extraction process for clusters in point data and extend it to cluster polygonization. In contrast to the α -shape approach, our approach minimizes the need for parameter-tuning. Instead, it derives the shape of a cluster C_i from both the spread of points in itself and the spread of points in $P \setminus C_i$. It approximates shapes of clusters using the Delaunay diagram.

8.1.2 Cluster polygonization algorithm

A cyclic list of edges determines the polygon of a cluster that discriminates internal points of the cluster (points lying within the cluster or on boundaries) from external points (points lying outside of the cluster). Short-Long clustering delivers P with cluster identifiers, but this is not enough for cluster polygonization analysis. We need to extract sets of cyclic lists of edges that define polygons for clusters. To do this, we repeatedly merge triangles and remove edges shared by triangles.

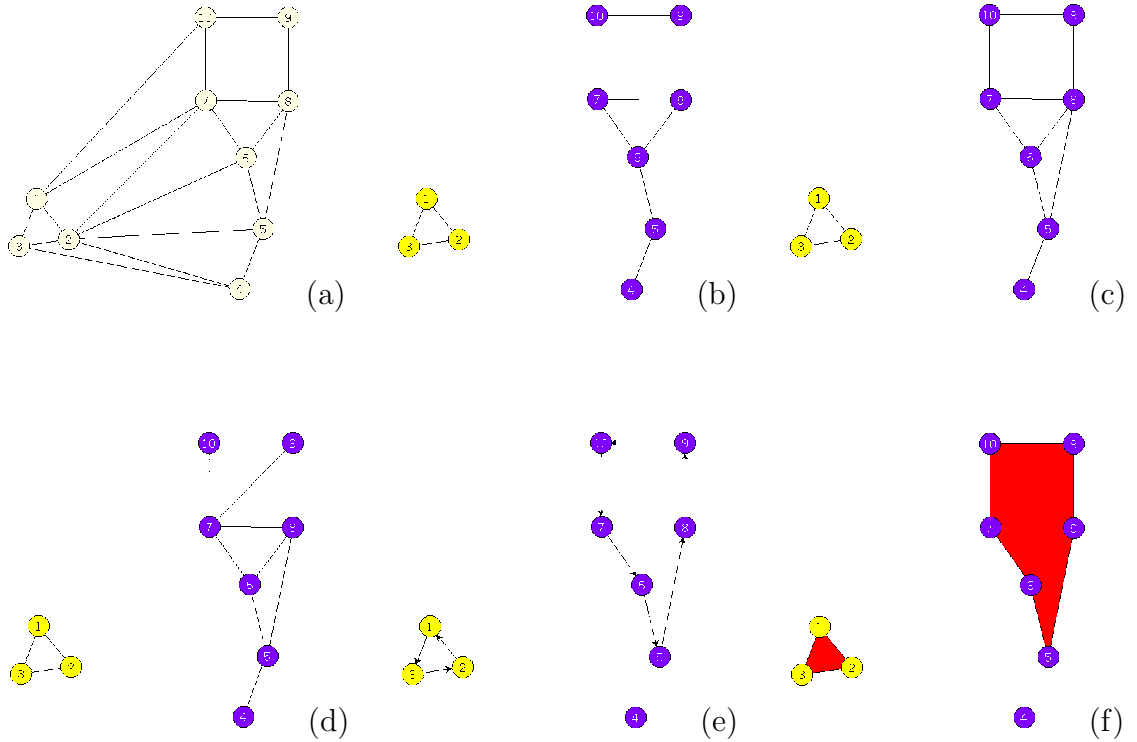


Figure 8-1: Cluster polygonization through cluster boundary extraction ($|P| = 10$) (a) A set P of points with $DD(P)$; (b) Short-Long clustering results (2 clusters); (c) Recuperating intra-cluster edges in $DD(P)$; (d) Triangulating faces resulting from cocircularity; (e) Extracting directed cluster boundaries; (f) Polygons for the clusters.

First, we recuperate all intra-cluster edges by reconnecting every edge $e_{i,j} = (p_i, p_j)$ satisfying $CC[p_i] = CC[p_j]$ after Short-Long clustering. This is the same as the recuperation process for multi-level Short-Long clustering discussed in Section 7.2. Figure 8-1(c) depicts a result of the recuperation process while Figure 8-1(b) illustrates a result after Short-Long clustering. The recuperated subgraph approximates shapes of clusters.

Since $DD(P)$ is the underlying graph of our study, faces after the recuperation are not necessarily triangles. Figure 8-1(c) depicts an example $\langle p_7, p_8, p_9, p_{10} \rangle$. These faces that are not triangles are temporarily triangulated (some edges are added) to make our cluster polygonization algorithm easy and simple. For instance, either $e_{7,9}$ or $e_{8,10}$ can be added for the face $\langle p_7, p_8, p_9, p_{10} \rangle$. Note that, these artificially added edges are removed in the later process (merging triangles) thus they do not affect shapes of clusters. Figure 8-1(d) depicts triangulation of non-triangle faces.

Now, we can merge triangles to extract cluster boundaries. The triangle merge process produces directed edges that eventually result in sets of cyclic lists of edges. However, polygons may contain holes. Thus, a polygon may require more than one cyclic list of edges. In our approach, sets of cyclic lists of edges, that are in counterclockwise ordering, are external cluster boundaries defining inner areas as cluster regions while sets of cyclic lists of edges, that are in clockwise ordering, are internal cluster boundaries defining holes within corresponding cluster regions. Thus, the interior of the polygon is always to the left of the directed edge.

However, incomplete *dead ends* [20] need to be removed in the triangle merge process since no areas are associated with them. Note that, a dead end $e_{4,5}$ depicted in Figure 8-1(d) is removed and does not appear after the triangle merge process as shown in Figure 8-1(e). Finally, polygons for clusters can be derived as depicted in Figure 8-1(f).

Now, we provide the details of the cluster polygonization algorithm. It consists of four phases and is as follows.

Algorithm **The Short-Long cluster polygonization procedure**Input: A subgraph G of $DD(P)$ labeled with cluster identifiers;

Output: Polygons of clusters;

- 1) **begin**
- 2) $intraG \leftarrow \text{RecuperateDelaunayDiagram}(G)$;
- 3) $triangulatedG \leftarrow \text{TriangulateFaces}(intraG)$;
- 4) $boundaryG \leftarrow \text{TheTriangleMerge}(triangulatedG)$;
- 5) $\text{ClusterPolygonization}(boundaryG)$;
- 6) **end**

The process labeled **RecuperateDelaunayDiagram** (Step 2) is simple and explained earlier in this section. It recuperates intra-cluster edges in $DD(P)$ after Short-Long clustering and returns the recuperated graph ($intraG$) (see Figure 8-1(c) for example). The process labeled **TriangulateFaces** takes $intraG$ as an input graph and produces $triangulatedG$ as an output. It triangulates non-triangle faces in $intraG$ resulting from cocircularity (see Figure 8-1(d) for example). Let a non-triangle face in $intraG$ consist of k points $(p_1, p_2, \dots, p_k)$ and let the $j - i + 1$ points $(p_i, p_{i+1}, \dots, p_j)$ for $1 \leq i \leq j \leq k$ have the same cluster identifier. Then, the triangulation adds edges $e_{i,i+2}, e_{i,i+3}, \dots, e_{i,j-1}$.

The process labeled **TheTriangleMerge** (Step 4) takes $triangulatedG$ as an input graph and produces $boundaryG$ as an output graph. It removes dead ends, and extracts and orients cluster boundary edges (see Figure 8-1(e) for example). Since the Delaunay triangulation is a planar subdivision into triangles, each edge in the Delaunay triangulation belongs to either one Delaunay triangle (external Delaunay edge or hull edge [110]) or to two (internal Delaunay edge [110]). Because $triangulatedG$ is a subgraph of planar subdivision into triangles, each edge in $triangulatedG$ belongs to none (dead end), to one Delaunay triangle or at most to two. For each intra-cluster edge $e_{a,b} = (p_a, p_b) \in triangulatedG$, the **TheTriangleMerge**($triangulatedG$) proceeds as follows.

- Simple case: The intra-cluster edge $e_{a,b}$ is an external Delaunay edge or hull edge. That is, the edge can have at most one incident triangle. Figure 8-2 illustrates this case.
 1. Subcase: Triangle in a cluster — The third node p_c of the triangle $\langle p_a, p_b, p_c \rangle$ has the same cluster identifier as nodes p_a and p_b . Then, edge $e_{a,b}$ is labeled as a boundary edge and oriented (because *triangulatedG* is a planar embedding) such that p_c is on its left (the interior of the triangle) and the exterior is on its right. Figure 8-2(a) illustrates this subcase.
 2. Subcase: No triangle — The third node p_c of the triangle $\langle p_a, p_b, p_c \rangle$ does not have the same cluster identifier as p_a and p_b . Then, $e_{a,b}$ is not labeled as a boundary edge (it will be removed). That is, $e_{a,b}$ is not placed in the output of this process because it is a dead end. Figure 8-2(b) illustrates this subcase.

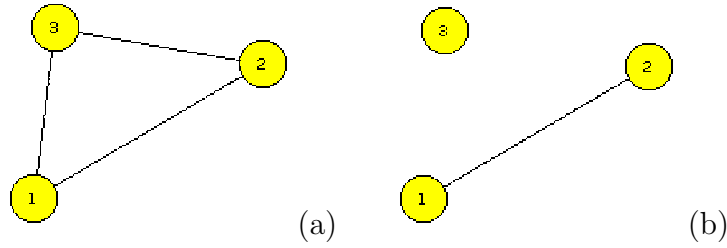


Figure 8-2: The two subcases of the simple case with $p_a = p_1$, $p_b = p_2$ and $p_c = p_3$ in *triangulatedG*: (a) Triangle in a cluster; (b) No triangle.

- Complex case: The intra-cluster edge $e_{a,b}$ is an internal Delaunay edge. That is, the edge can have two incident triangles (the most possible). Figure 8-3 illustrates this complex case.
 1. Subcase: Two triangles in a cluster — The third nodes p_c and p_d of triangles $\langle p_a, p_b, p_c \rangle$ and $\langle p_a, p_b, p_d \rangle$ are both in the same connected component as p_a and p_b . Then, $e_{a,b}$ is removed (it will not be in the output as a boundary

edge since clearly it is inside the quadrilateral $\langle p_a, p_c, p_b, p_d \rangle$ that is inside the cluster. Figure 8-3(a) illustrates this subcase.

2. Subcase: Triangle in a cluster — One of the third nodes p_c or p_d is in the cluster of p_a and p_b , but the other is not. Without loss of generality, we assume p_c is in the cluster of p_a and p_b (recall that $CC[p_a] = CC[p_b]$) while p_d is not (i.e. $CC[p_d] \neq CC[p_a]$). Then, edge $e_{a,b}$ is a boundary edge since the triangle $\langle p_a, p_b, p_c \rangle$ is in the cluster but the triangle $\langle p_a, p_b, p_d \rangle$ is not. Then, $e_{a,b}$ is labeled as a boundary edge and oriented in such a way that p_c is on its left (the interior) and p_d is on its right (the exterior). Figure 8-3(b) and (c) illustrates this subcase.
3. Subcase: No triangle — Both, p_c and p_d are not in the same cluster as p_a and p_b . Then, $e_{a,b}$ is removed because it is a dead end. Figure 8-3(d) illustrates this subcase.

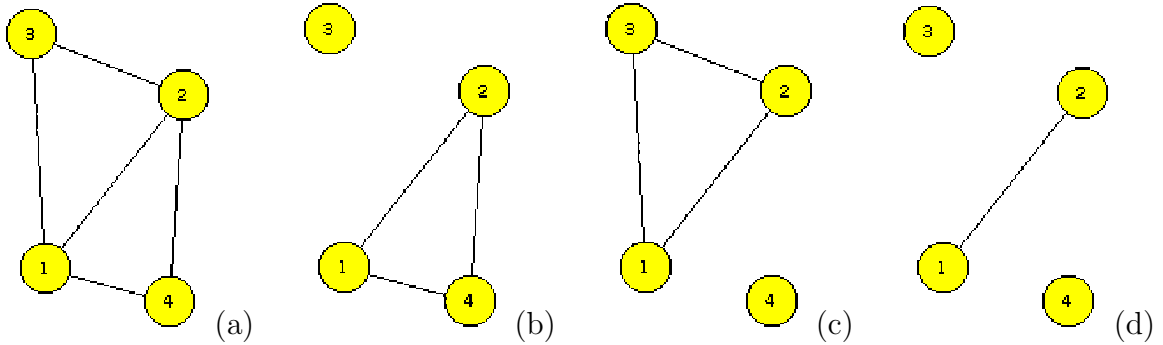


Figure 8-3: The subcases of the complex case with $p_a = p_1$, $p_b = p_2$, $p_c = p_3$ and $p_d = p_4$ in *triangulatedG*: (a) Illustration of the first complex subcase, two triangles in a cluster; (b) and (c) are the second subcase triangle in a cluster; (d) No triangle.

After the **TheTriangleMerge** process, *external cluster boundaries* (defining bounded regions) are always in counterclockwise ordering whilst *internal cluster boundaries* (defining unbounded regions) are always in clockwise ordering. Cluster polygonization constructs a list of edges such that traversing the list of edges corresponds to

navigating along the boundary of a cluster. The list will be circular, eventually returning to the same node. A polygon with holes will be made of several of these lists. We now prove that cluster polygonization is always possible from the output of the **TheTriangleMerge** process (*boundaryG*). The following lemma proves that *boundaryG* is Eulerian and it is now a matter of traversing the graph within the planar embedding to extract the circular (cyclic) lists of edges (equivalently nodes) that constitute polygons.

◇ **Lemma 8.1** *For every node p_a attached to an oriented edge e in *boundaryG*, the following holds.*

- *The degree of p_a is even.*
- *The indegree of p_a equals the outdegree of p_a .*
- *It is possible to alternate the incoming edges and the outgoing edges either clockwise or counterclockwise in the planar embedding representing the output of the **TheTriangleMerge** process (*boundaryG*).*

Proof. The output of the **TheTriangleMerge** process is equivalent to a union of disjoint triangles (the triangles share edges but not their interiors). Note that, when the **TheTriangleMerge** process removes intra-cluster edges, it performs the union of two regions which are the union of triangles and thus the new region is also the union of triangles. The point p_a must belong to a triangle in the union because it is attached to an oriented edge e . Let T be the sequence of triangles clockwise around the point p_a . This sequence T of triangles corresponds to a sequence of edges E . These edges are all incident to p_a with e oriented. Without loss of generality, assume e is an incoming edge and that the sequence $E = \langle e, e_1, \dots, e_k \rangle$ is the sequence of edges in the triangulation incident to p_a when traveling clockwise around p_a . Because e is incoming, the triangle determined by $\langle e, p_a, e_1 \rangle$ must be exterior. If e_1 is oriented, then it must be outgoing and the triangle $\langle e_1, p_a, e_2 \rangle$ must be interior. If e_1 is not

oriented, it is because the next triangle $\langle e_1, p_a, e_2 \rangle$ is exterior. Continuing in this way, we see that the triangles in T alternate between a few that are interior and a few that are exterior. In any case, when the triangles swap from interior to exterior, the edge incident to p_a is incoming and when the swap back the edge must be outgoing. But the sequence of triangles around node p_a is finite, so when completing the clockwise traversal we see that the lemma is satisfied.

8.1.3 Time complexity analysis

Since the number of edges and the number of faces in $DD(P)$ are linear in P [110], The `RecuperateDelaunayDiagram` and `TriangulateFaces` processes are bounded by $O(n)$ time. The `TheTriangleMerge` process tests if every intra-cluster edge e in *triangulatedG* is a boundary edge. It performs constant work for each edge. Thus, this is linear as well. Therefore, the cluster polygonization algorithm requires linear time for Short-Long clustering.

8.1.4 Performance evaluation

We present results from experiments with the synthetic datasets shown earlier in Chapter 4. Experiments with real datasets will appear in Section 8.2.

Figure 8-4 illustrates that our `TheTriangleMerge` process automatically derives oriented cluster boundaries. Cluster boundaries reveal shapes of clusters more easily than the datasets themselves and clustering results of the datasets. Figure 8-4(a) depicts that our approach successfully derives boundaries of clusters that contain other clusters. The “8-like” cluster consists of three lists of edges; one of them in counterclockwise ordering represents external cluster boundaries and two of them in clockwise ordering represent internal cluster boundaries. Figure 8-4(b), Figure 8-4(c) and Figure 8-4(d) demonstrate that our approach has no difficulty with clusters containing holes, clusters of arbitrary shapes, clusters of different densities and clusters

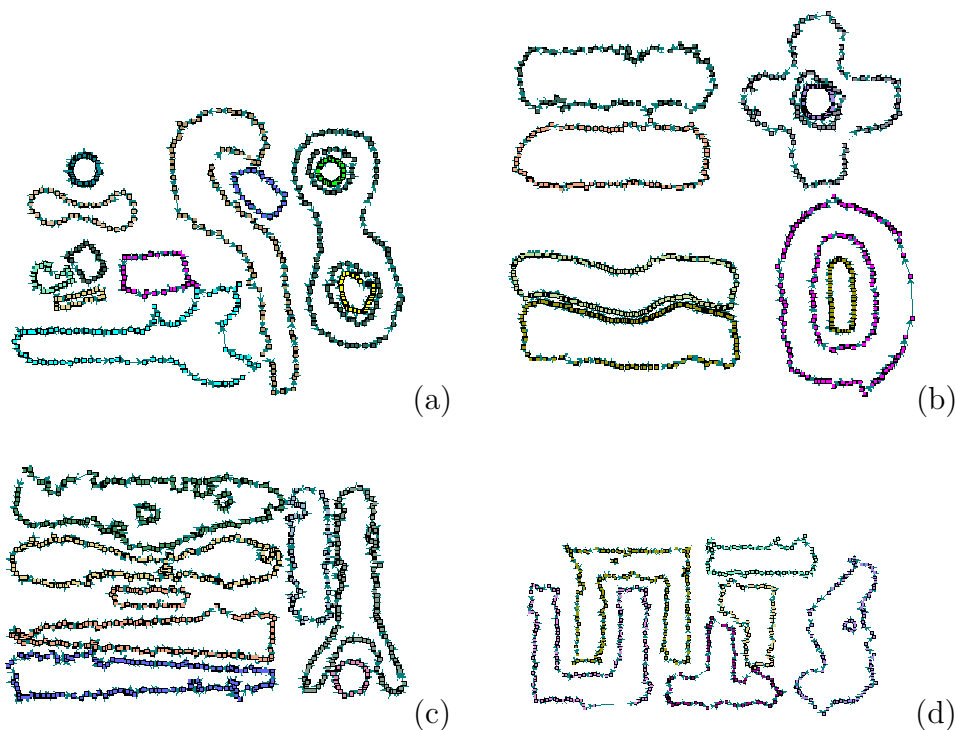


Figure 8-4: Oriented cluster boundaries in the datasets shown earlier in Figure 4-11; (a) Oriented cluster boundaries in the synthetic dataset $\mathcal{I}$ (Figure 4-11(a)); (b) Oriented cluster boundaries in the synthetic dataset $\mathcal{II}$ (Figure 4-11(c)); (c) Oriented cluster boundaries in the synthetic dataset $\mathcal{III}$ (Figure 4-11(e)); (d) Oriented cluster boundaries in the synthetic dataset $\mathcal{IV}$ (Figure 4-11(g)).

of heterogeneous sizes. In terms of efficiency, it does not require parameter-tuning but it is automatic. In terms of effectiveness, it produces quality cluster boundaries by considering the distribution of all points in P . Thus, our approach is well-suited for data-rich environments.

Figure 8-5 depicts corresponding cluster polygons after the `ClusterPolygonization` process. Here, shaded regions represent cluster polygons and holes become visually apparent.

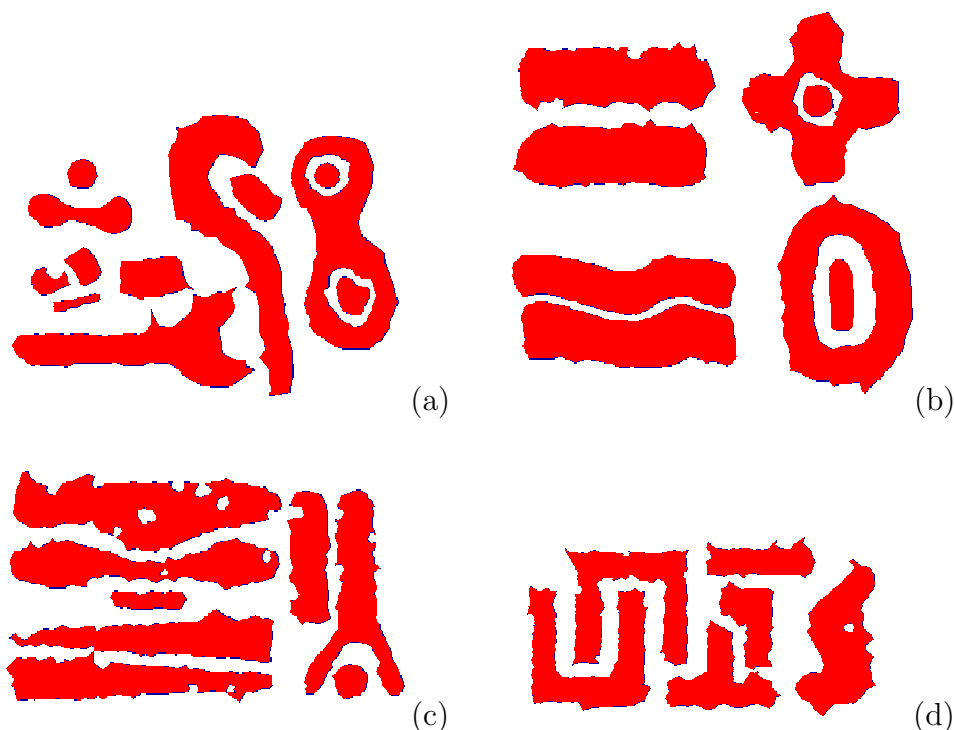


Figure 8-5: Polygons of clusters in the datasets shown in Figure 4-11; (a) Polygons of clusters of the synthetic dataset $\mathcal{I}$; (b) Polygons of clusters of the synthetic dataset $\mathcal{II}$; (c) Polygons of clusters of the synthetic dataset $\mathcal{III}$; (d) Polygons of clusters of the synthetic dataset $\mathcal{IV}$.

8.1.5 Applications

In this section, we provide several applications of the cluster polygonization algorithm.

Modeling points with clusters

Although we have discussed several modeling methods using proximity graphs in Section 5.3, their defined neighbors are confined to a local region. For instance, points are neighbors when their corresponding Voronoi regions touch and the expected number of neighbors in $\mathbb{R}^2$ is 6 [110] in Voronoi modeling. In some spatial settings, we need large scales of neighboring. For instance, two points are considered as neighbors if

they exhibit similar characteristics although their locally defined “territories” (corresponding areas resulting from point-to-area transformations) do not touch. Clustering is a possible solution for this local neighbor limitation. Thus, points within the same cluster are regarded as neighbors (since similar concentration occurs around them). Regions that do not belong to any other cluster regions are assigned to a default region (a noise region) for noise points. Note that, the noise region may be disconnected. That is, the noise region may include several disconnected regions. Thus, every location in space is assigned to one of cluster regions or alternatively to the noise region. This constitutes a partition of the plane into regions. The resulting regions by cluster polygonization are mutually exclusive and collectively exhaustive. That is, cluster polygonization produces a spatial tessellation.

Choropleth mapping with cluster polygons

One of the difficulties of visualizing point data is to manage privacy. Another problem with point data is that theoretically, the probability of an event at a point is zero. More intuitively, even a point is information about an area, but what is the region for which the information applies? In order to resolve the privacy problem or to make inferences on regions rather than locations, point data are often aggregated based on polygonal maps. For example, one way of visualizing point data is to use choropleth maps. These maps display point data related to a specific topic with respect to a boundary map such as political or administrative area map. The display of a choropleth map fills polygons of the reference areas with colors or grey tones according to densities (the number of points per unit of area).

One major flaw of choropleth mapping is that its visual impression is heavily dependent on the base map under use. Dent [27] warns that distributions of continuous geographical phenomena are not governed by political or administrative subdivisions. Thus, choropleth mapping of points with political or administrative layers is less informative. In addition, the Modifiable Areal Unit Problem (MAUP) [114] can arise in analysis due to the use of different polygonal regions as base maps in choropleth

mapping. Using the set of cluster regions of P as a basic area map overcomes these problems of traditional choropleth map, since boundaries of cluster regions are derived from the distribution of P . Thus, it minimizes artificial constraints on choropleth mappings, which is important in GDM.

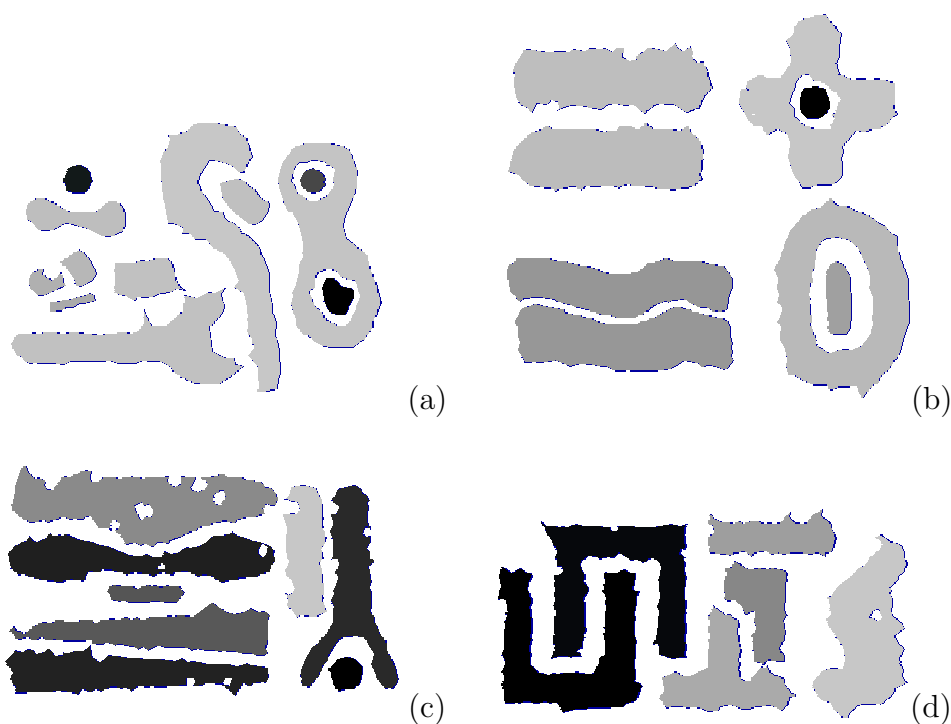


Figure 8-6: Choropleth maps with cluster polygons as a base map; (a) A choropleth map of the synthetic dataset $\mathcal{I}$; (b) A choropleth map of the synthetic dataset $\mathcal{II}$; (c) A choropleth map of the synthetic dataset $\mathcal{III}$; (d) A choropleth map of the synthetic dataset $\mathcal{IV}$.

Figure 8-6 displays choropleth maps for the datasets discussed in Section 8.1.4. Here, the densest cluster is in black (RGB(0,0,0)) and the sparsest cluster is in light grey (RGB(200,200,200)). Intermediate clusters are colored between the tones of grey in proportion to their respective densities. Figure 8-6(a) shows that clusters are either high density or low density. Thus, we can easily find three high concentrations. Figure 8-6(b) indicates that densities among clusters do not have a global pattern. However, clusters surrounded by other clusters and two closely located clusters ex-

hibit relatively high densities. Figure 8-6(c) reveals a hierarchy of cluster densities that is easily understandable. The denser clusters of the synthetic dataset *III* (Figure 4-11(e)) are randomly mixed with other clusters. Since this spread is not easily recognized from Figure 4-11(f), the corresponding choropleth map (Figure 8-6(c)) is more informative. Visual inspection of the original datasets (Figure 4-11) is insufficient to report relatively dense or sparse clusters. In Figure 4-11(g), one can hardly see any difference in density. The dataset seems too large for such visual inspection. However, the choropleth map in Figure 8-6(d) clearly indicates that two clusters on left-hand side are relatively dense. A pattern that shows density decreasing from left to right is now clearly visible.

Cluster reasoning

Clustering for data mining is to summarize the distribution of P in order to suggest a manageable number of patterns of concentrations. Thus, clusters are representatives of the phenomena recorded with locations in P and suggesting interesting groups. Several approaches [44, 80, 130, 137] have been proposed to reason about clusters. Boundaries of clusters and representatives of clusters (medoids or means) are the most popular candidates for reasoning with clusters. However, these candidates are summaries. They constitute only partial information about clusters, thus we need special care when we use these information for cluster reasoning.

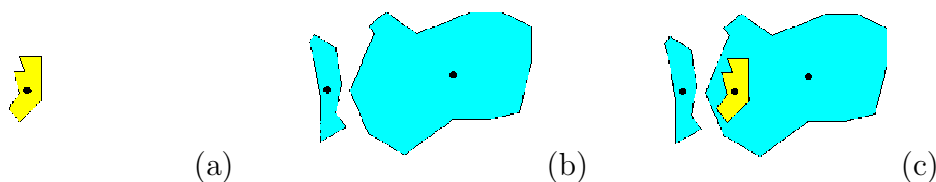


Figure 8-7: Cluster reasoning: (a) Layer 1; (b) Layer 2; (c) Overlay of Layer 1 and Layer 2.

Figure 8-7 illustrates the problem of using such partial information. Consider two layers shown in Figure 8-7(a) and Figure 8-7(b). Layer 1 displayed in Figure 8-7(a)

has a small cluster while Layer 2 depicted in Figure 8-7(b) has two clusters (one small and one large). Black dots represent medoids of clusters. Although the small cluster in Layer 1 has a high association with the large cluster in Layer 2 (since the small cluster lies within the large cluster), traditional approaches fail to detect this association. Cluster reasoning analysis using boundary matching [80] does not succeed in detecting this correlation, since boundaries of the small cluster in Layer 1 does not match with those of the large cluster in Layer 2. This is shown in Figure 8-7(c). Similarly, cluster reasoning analysis using medoids [44] is unable to report this association, since the medoid of the small cluster in Layer 1 is closer to the medoid of the small cluster in Layer 2 than that of the large cluster in Layer 2.

We can detect this relationship by computing the intersection area of cluster regions. The intuition of this approach is that two clusters exhibit a high association if most phenomena in the first cluster take place where concentrations of phenomena in the second cluster occur. The small cluster in Layer 1 intersects with the large cluster in Layer 2, while it does not intersect with the small cluster in Layer 2. Thus, it is more associated with the large cluster in Layer 2 rather than the small one.

8.2 Mining multivariate positive associations using cluster polygonization

8.2.1 Spatial association rules

The mining of association rules has been a powerful tool for discovering correlations among massive transactional databases [2, 3, 60, 74, 117, 126, 138]. An association rule is an expression in the form of $X \Rightarrow Y(c\%)$, where X is the antecedent and Y is the consequent, X and Y are sets of items in transactional databases, and $X \cap Y = \emptyset$. It is interpreted as “ $c\%$ of data that satisfy X also satisfy Y ”. Here, a table summarizes a set of records. The table has a number of rows corresponding

to transactions and a number of columns corresponding to items. The value of an item for a given record is “1” if the item is in the transaction, “0” otherwise. From the table, we mine association rules that correlate the presence of a set of items with another set of items. Each rule has an associated *support* and *confidence* defined as follows:

$$\text{support} \approx \text{Pr}[X \cap Y], \quad (8.1)$$

$$\text{confidence} \approx \frac{\text{Pr}[X \cap Y]}{\text{Pr}[X]}. \quad (8.2)$$

The support is an estimate for the ratio of transactions that satisfy both X and Y to the number of transactions in databases. The confidence is an estimate for the conditional probability of Y given X . Since users are interested in large support and high confidence (strong rules [2, 61]), two thresholds (minimum support and minimum confidence) are used for pruning rules to find strong association rules. The aim of mining association rules is to compute all association rules satisfying user-specified minimum support and minimum confidence constraints.

Recently, Koperski and Han [81, 83] discuss mining spatial association rules that reveals associations between spatial objects. This is a reference feature model. It requires a query to focus the feature of study. This approach requires concept hierarchies to progressively prune large search spaces. It first mines frequent patterns at a higher concept level and deepens to lower concept levels for those mined frequent patterns. One example of spatial association rule discovered by this approach is given below.

$$\text{is_a}(x, \text{house}) \wedge \text{near_by}(x, \text{park}) \Rightarrow \text{is_expensive}(x)(95\%). \quad (8.3)$$

The rule suggests that 95% of houses attached to parks are expensive. This rule uses several predicates that contain either spatial-spatial relationship (*near_by*) or spatial-aspatial relationship (*is_a* and *is_expensive*). Here, predicates are predefined before mining association rules and only rules that are somehow related to the predicates are extracted. Thus, this approach is seen as a model-driven approach rather than a data-driven. Defining predicates requires domain knowledge and necessitates

concept hierarchies that are difficult to define. For instance, it is hard to classify a house (\$200,000(USD)) as expensive or not. The decision purely depends on domain knowledge or personal decision. Thus, concept hierarchies are task-specific and thus applicability is limited to the case at hand.

8.2.2 Algorithms for mining multivariate positive associations

In this section, we propose two approaches for mining spatial association rules using cluster analysis that automatically reveals positive correlations among layers. Since these approaches do not require domain-specific concept hierarchies but rather build cluster hierarchies from data, these are applicable to various tasks and suitable for data-rich environments.

Figure 8-8 explains a vertical-view approach and a horizontal-view approach with five geographical layers. If we pinpoint a location within R , the location will have five associated attributes corresponding to the five layers. Figure 8-8(b) shows these 5 attributes vertically (referred as an attribute cube). Values of attributes become true (1) if the location lies within regions (clusters) of corresponding layers, false (0) otherwise. The vertical-view approach tries to discover interesting associations from the whole set of attribute cubes. For instance, a spatial association rule “layer(1) $\wedge$ layer(2) $\Rightarrow$ layer(4) (70%)” is derived if 70% of attribute cubes satisfying attribute values of layer 1 and layer 2, also have the value true in layer 4.

The horizontal-view approach overlays all the layers into a target layer, and then attempts to find associations from the target layer using intersection (overlapping) areas. Figure 8-8(c) overlays the first, second and third layers depicted in Figure 8-8(a). The first and second layers intersect while the third layer does not intersect with the other two. Thus, the association between the first layer and the second layer is higher than that of the first and the third and that of the second and the third. In

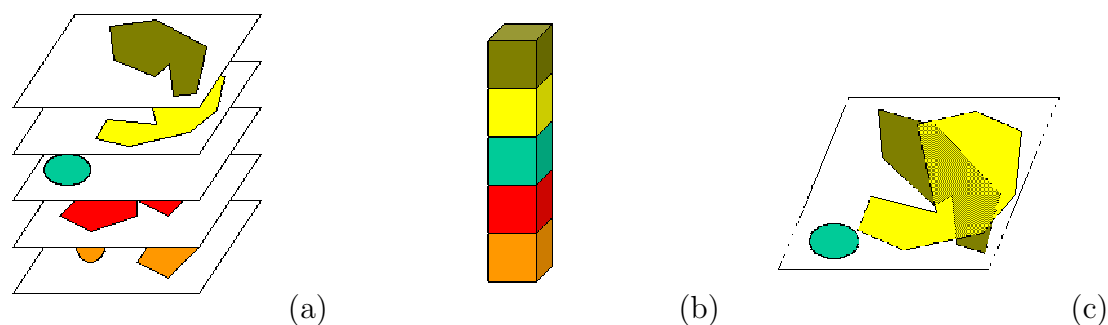


Figure 8-8: Mining multivariate associations: a vertical-view approach *vs* a horizontal-view approach (the number of layers = 5) (a) 5 layers; (b) A vertical-view approach; (c) A horizontal-view approach.

the two approaches, we only consider spatial clusters of point data as candidates for mining associations. We believe that there are some reasons (possibly attractors) for spatial concentrations. That is, particular environments (pattern generators) attract phenomena and thus result in concentrations. Mining multivariate positive associations using cluster polygonization is to find possible attractors or some positive contributors to spatial clusters. Noise points are the points that are not affected by the attractors. Thus, they are ignored for mining cluster associations.

Now, we discuss details of the proposed approaches.

Vertical-view approach

The vertical-view approach is similar to the popular raster model in the sense that they model space with regular cells. The algorithmic procedure of the vertical-view approach is as follows.

1. Find spatial clusters for point data layers.
2. Segment all the layers with a finite number of rectangles.
3. Construct a $M \times N$ Boolean table (with the binary $\{0,1\}$ values).

4. Apply mining association rules algorithms to the table.

We first compute spatial clusters of point data by cluster analysis. After that, we segment R and construct attribute cubes. With attribute cubes, we build a Boolean $M \times N$ table (with the binary domain $\{0,1\}$), where M denotes the number of attribute cubes and N denotes the number of layers. In the table, $[M_i, N_j] = 1$ if an attribute cube M_i satisfies clustered event in a layer N_j . Finally, we mine associations that correlate the presence of a set of layers with another set of layers. Since the Boolean table of layer-based GIS is exactly same as that of transactional databases except layers replace items and locations replace transactions, it is now straightforward to discover associations among layers using traditional mining association rules algorithms [2, 3, 60, 74, 117, 126, 138]. We illustrate this with an example.

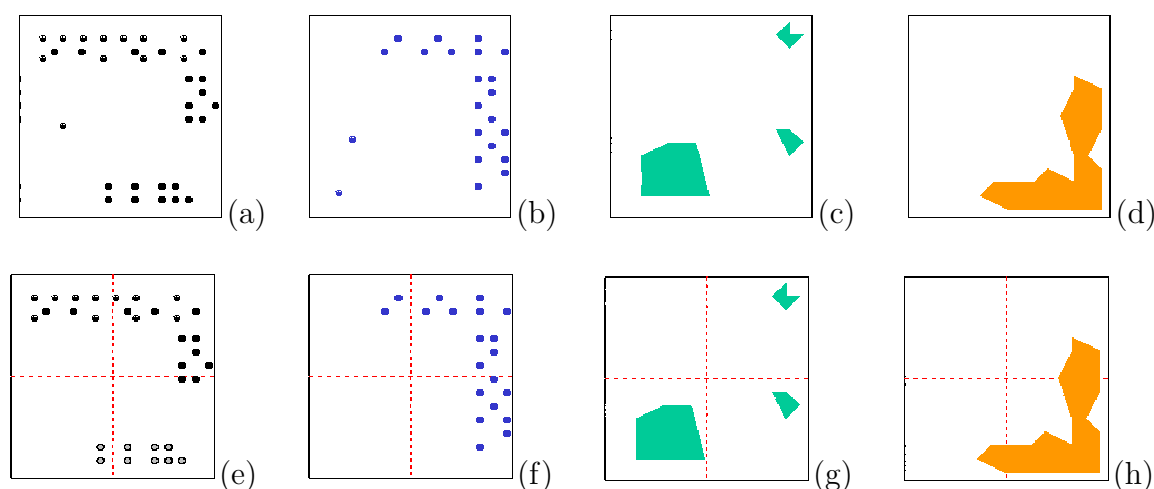


Figure 8-9: An example of the vertical-view approach with 4 cells for mining multivariate association rules (the number of layers = 4): (a) Railway stations (point data); (b) Crime incidents (point); (c) Parks (area); (d) Urban areas (area); (e) Clustered railway stations with 4 cells; (f) Clustered crime incidents with 4 cells; (g) Parks with 4 cells; (h) Urban areas with 4 cells.

Figure 8-9 and Table 8.1 illustrate the vertical-view approach with 4 geographical layers. Let us assume that Figure 8-9(a)-(d) show railway stations (point), crime incidents (point), parks (area) and urban areas (area), respectively. The first process

is to find homogeneous groups of spatial concentrations of point data layers (using Short-Long clustering). Two clusters of railway stations and one cluster of crime incidents are shown in Figure 8-9(e) and Figure 8-9(f), respectively. Noise points are ignored. Then, we frame R with collectively exhaustive and mutually exclusive cells. A 2×2 grid is used in this case. After that, we compute a 4×4 Boolean table before we apply a mining association rules algorithm. Table 8.1 presents the Boolean table.

Table 8.1: A 4×4 Boolean table.

	<i>layer(1)</i>	<i>layer(2)</i>	<i>layer(3)</i>	<i>layer(4)</i>
<i>location(1)</i>	1	1	0	0
<i>location(2)</i>	1	1	1	1
<i>location(3)</i>	1	0	1	1
<i>location(4)</i>	1	1	1	1

In the table, $\text{layer}(j)$ ($0 \leq j \leq N$) corresponds to the j -th column and denotes j -th geographical layer. Similarly, $\text{location}(i)$ ($1 \leq i \leq M$) corresponds to the i -th row and denotes i -th cell (attribute cube) in R (numbered in Morton order, Z). The value of the table $t[\text{location}(i), \text{layer}(j)]$ is 1 if clustered events in the j -th layer occur in numbered cell $\text{location}(i)$, and $t[\text{location}(i), \text{layer}(j)]$ is 0 otherwise. For instance, $t[\text{location}(1), \text{layer}(1)] = 1$ because members of clusters of railway stations lie within the top-left cell as depicted in Figure 8-9(e). We mine multivariate associations from this Boolean table. One of association rules is as follows:

$$\text{layer}(1) \wedge \text{layer}(2) \Rightarrow \text{layer}(4)(66.7\%). \quad (8.4)$$

With 50% support ($\text{location}(2)$ and $\text{location}(4)$ out of 4 attribute cubes), 66.7% of locations, that are nearby railway stations ($\text{layer}(1)$) and have crime incidents ($\text{layer}(2)$), fall within urban areas ($\text{layer}(4)$). In this approach, the granularity of cells plays a critical role.

Figure 8-10 and Table 8.2 illustrate the vertical-view approach with a 4×4 grid for the same dataset shown in Figure 8-9. Now, the bottom-right cell (attribute cube

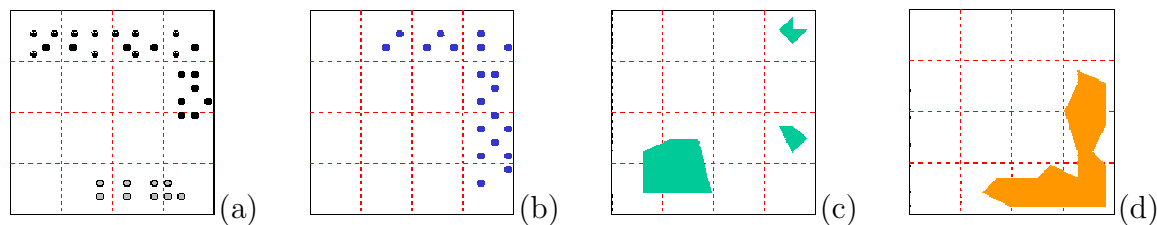


Figure 8-10: An example of the vertical-view approach with 16 cells for mining multivariate association rules (datasets are same as those in Figure 8-9): (a) Clustered railway stations with 8 cells; (b) Clustered crime incidents with 8 cells; (c) Parks with 8 cells; (d) Urban areas with 8 cells.

Table 8.2: A 16×4 relational table.

	<i>layer</i> (1)	<i>layer</i> (2)	<i>layer</i> (3)	<i>layer</i> (4)
<i>location</i> (11)	1	0	0	0
<i>location</i> (12)	1	1	0	0
<i>location</i> (13)	0	0	0	0
<i>location</i> (14)	0	0	0	0
<i>location</i> (21)	1	1	0	0
<i>location</i> (22)	1	1	1	0
<i>location</i> (23)	0	0	0	0
<i>location</i> (24)	1	1	0	1
<i>location</i> (31)	0	0	1	0
<i>location</i> (32)	0	0	1	0
<i>location</i> (33)	0	0	1	0
<i>location</i> (34)	1	0	1	1
<i>location</i> (41)	0	0	0	0
<i>location</i> (42)	1	1	1	1
<i>location</i> (43)	1	0	0	1
<i>location</i> (44)	1	1	0	1

location(44)) in Figure 8-10 has values (1, 1, 0, 1) since the cell contains (clustered) events of layers layer(1), layer(2), and layer(4), but layer(3). With the 50% support constraint, now we are not able to find the rule “layer(1) $\wedge$ layer(2) $\Rightarrow$ layer(4)”, since its support is only 18.8% (3/16). However, the confidence is now 50%, since three locations (location(24), location(42) and location(44)) out of six satisfying layer(1) and layer(2), also satisfy layer(4). One of advantages of the vertical-view approach is that it is easy to apply transactional mining association rules algorithms, since the vertical-view approach uses Boolean tables. However, it has a main drawback. That is, rules discovered by the vertical-view approach are heavily dependent on the granularity that is difficult to determine. Similar difficulties are found when modeling point data using raster-like modeling as discussed in Section 3.1.

Horizontal-view approach

Since the vertical-view approach needs a parameter to determine the granularity, it is regarded as an argument-dependent approach. Argument-tuning resulting in totally different outputs is prohibitive in data-rich environments. Thus, GDM favors argument-less or argument-free approaches in order to reduce preprocessing time and to minimize inconsistency of results. Here, we propose an automatic approach for mining multivariate associations that minimizes the need for user-supplied parameters. This approach is based on intersection of areas of clusters similar to cluster reasoning discussed in Section 8.1.5.

Since only spatial clusters are considered for mining associations, we need to redefine support and confidence. The following definitions are made to explain the working principle of the horizontal-view approach.

Definition 8.2.2.1 *Let X and Y be sets of layers representing the antecedent and the consequent, respectively. The Support with Clusters (SwC) is the ratio of the area that satisfy both X and Y to the area of study region R . That is,*

$$SwC = area(X \cap Y) / area(R). \quad (8.5)$$

If X_i or Y_j (i -th layer of the antecedent or j -th layer of the consequent) is a point data layer, then $area(X_i)$ or $area(Y_j)$ becomes the sum of areas of polygonized clusters (polygons of clusters). Note that, $area(X)$ is the intersection of all $area(X_i)$ for $X_i \in X$ while $area(Y)$ is the intersection of all $area(Y_i)$ for $Y_i \in Y$. Then, $area(X \cap Y)$ is a vertical intersection of $area(X)$ and $area(Y)$.

Definition 8.2.2.2 *The Confidence with Clusters (CwC) for a rule $X \Rightarrow Y$ is the conditional probability of areas of the consequent given areas of the antecedent. That is,*

$$CwC = area(X \cap Y) / area(X). \quad (8.6)$$

Definition 8.2.2.3 *A Clustered Spatial Association Rule (CSAR) is an expression in the form of*

$$X \Rightarrow Y(CwC\%), \text{ for } X \cap Y = \emptyset. \quad (8.7)$$

The interpretation is as follows: $CwC\%$ of areas of X intersect with areas of Y .

The algorithmic procedure of the horizontal-view approach is as follows.

1. Find spatial clusters for point data layers.
2. Extract boundaries of clusters for point data layers.
3. Compute areas of the detected clusters for point data layers and areas of area data layers.
4. Overlay all layers under study.

5. Apply mining association rules algorithms with *CwC* and *SwC* to detect CSARs.

The horizontal-view approach first detects spatial clusters with Short-Long clustering and polygonizes those clusters with the cluster-to-area transformation discussed in Section 8.1. Since Short-Long clustering is extensible to the presence of obstacles in Minkowski metrics, the horizontal-view approach allows users to explore multivariate associations in various spatial settings. In addition, the cluster hierarchy revealed by multi-level Short-Long clustering allows users to efficiently mine hierarchical associations. Here, only strong association rules at a certain level are further investigated at the next level. We illustrate the working principle of the horizontal-view approach with real datasets in the next section.

8.2.3 Performance evaluation

In this section, we examine complex real crime data from the south east Queensland region of Australia, where the capital city Brisbane is located, with the horizontal-view approach. Similar to most urban areas, understanding of crime activity in this region is important for regional planners and criminologists as well as policing agencies. We consider 217 suburbs around Brisbane as the study region for this case study. The study region continues to experience significant and sustained population growth [105]. However, raw crime data ¹ in this region are too complex and extremely huge, thus it seems to be a difficult task even for domain experts to detect valuable patterns of crime incidents. Crime statistics provided by Queensland Police Service have three main categories: “offences against the person”, “offences against property” and “other offences”. The first category “offences against the person” consists of subcategories: homicide, assaults, sexual offences, robbery, extortion, kidnapping and other offences against the person. The second category “offences against property” is composed of breaking and entering, arson, other property damage, motor vehicle theft, stealing, fraud and other offences against property. The third category “other

¹We use crime data recorded by Queensland Police Service in the year of 1997 in the study region.

offences” includes drug offences, prostitution, liquor, gaming offences, trespassing and vagrancy, good order offences, traffic and related offences and miscellaneous offences. In addition, the subcategories could have several subsubcategories. For instance, homicide consists of attempted murder, conspiracy to murder, driving causing death and manslaughter.

The complex structure of crime data is not the only concern for crime activity analysis. We must consider feature data (parks, railway stations, schools etc.) along with crime data to detect the relationship of crime activity to salient features. GIS and crime mapping software have been dominant tools for exploring crime activity [106]. However, these tools are not quantitatively exploratory, but rather visually assistant with choropleth mapping, buffering, overlaying, intersecting and containment operations. Although these naive operations may provide valuable insights into the complex raw crime data, these necessitate heavy human involvement. Thus, they are unsuitable for data-rich environments.

For illustration purpose, we experiment the horizontal-view approach with the three main types of crime and three feature data in this case study. Figure 8-11 depicts three main categories of crime occurred in the year of 1997 and three feature data in *R*. Although visual displays provide general patterns, too much information causes confusion. Figure 8-11(a) depicts 9,618 incidents of “offences against the person”. And, Figure 8-11(b) displays 113,618 incidents of “offences against property” while Figure 8-11(c) shows 2,124 incidents of “other offences” in the region. Feature data, reserves (249), parks including caravan parks (462) and schools (306), are shown from Figure 8-11(d) to Figure 8-11(f).

Clustering summarizes complex distributions of raw crime data and polygonization of clusters provides regions of concentrations as hot spots. Figure 8-12 illustrates the horizontal-view mining approach with cluster regions of real data depicted in Figure 8-11.

Figures from Figure 8-12(a) to Figure 8-12(c) display overlays between areas of clus-

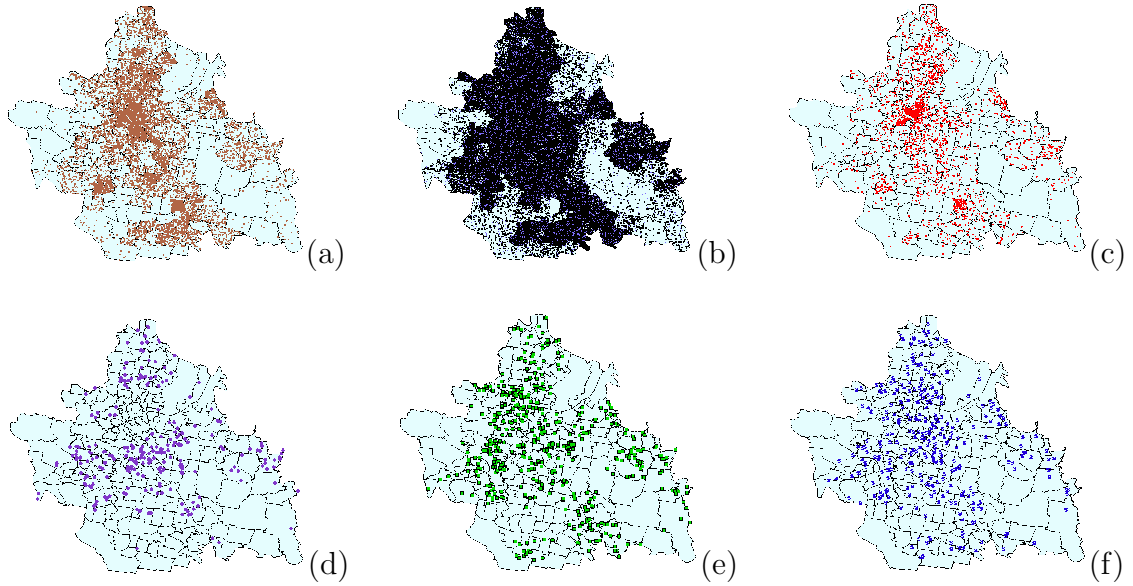


Figure 8-11: Real datasets recorded in the year of 1997 at 217 suburbs around Brisbane for mining multivariate associations with the horizontal-view approach: (a) Total offences against person (9618); (b) Total offences against property (113618); (c) Other offences (2124); (d) Reserves (249); (e) Parks including caravan parks (462); (f) Schools (306).

ters of “offences against the person” and areas of clusters of feature data. Similarly, overlays between areas of clusters of “offences against property” and areas of clusters of feature data are described from Figure 8-12(d) to Figure 8-12(f) and overlays between areas of clusters of “other offences” and areas of clusters of feature data are depicted from Figure 8-12(g) to Figure 8-12(i). Visual inspection interprets parks and schools are more associated with crime than reserves. This is quantitatively described in Table 8.3. Confidence with Clusters (CwC) of a CSAR “Offences against the person $\Rightarrow$ Reserves (44.93%)” is much lower than CwC of CSARs “Offences against the person $\Rightarrow$ Parks (85.29%)” and “Offences against the person $\Rightarrow$ Schools (77.5%)”. These rules imply that more than 70% of locations, where “offences against the person” occur, are around parks and schools.

The number of possible CSARs will increase exponentially as the number of layers

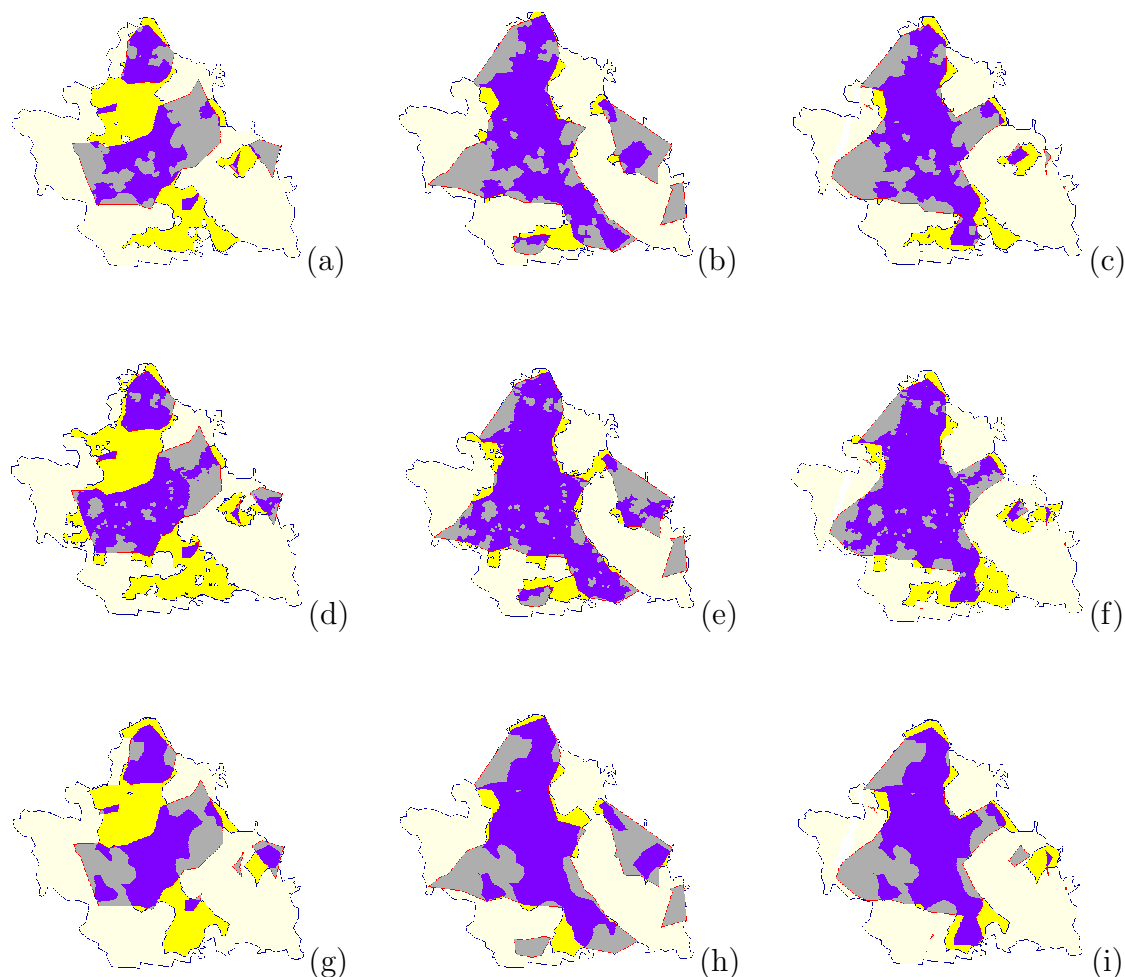


Figure 8-12: Illustration of the horizontal-view approach with the real datasets shown earlier in Figure 8-11: (a) Overlay of *areas* (offences against the person) and *area* (reserves); (b) Overlay of *areas* (offences against the person) and *area* (parks); (c) Overlay of *areas* (offences against the person) and *area* (schools); (d) Overlay of *areas* (offences against property) and *area* (reserves); (e) Overlay of *areas* (offences against property) and *area* (parks); (f) Overlay of *areas* (offences against property) and *area* (schools); (g) Overlay of *areas* (other offences) and *area* (reserves); (h) Overlay of *areas* (other offences) and *area* (parks); (i) Overlay of *areas* (other offences) and *area* (schools).

Table 8.3: CSARs discovered by the horizontal-view approach for the real data shown earlier in Figure 8-11.

	<i>SwC</i> (%)	<i>CwC</i> (%)
Offences against the person $\Rightarrow$ Reserves	15.40	44.93
Reserves $\Rightarrow$ Offences against the person	15.40	50.99
Offences against the person $\Rightarrow$ Parks	29.23	85.29
Parks $\Rightarrow$ Offences against the person	29.23	57.33
Offences against the person $\Rightarrow$ Schools	26.56	77.50
Schools $\Rightarrow$ Offences against the person	26.56	59.85
Offences against the property $\Rightarrow$ Reserves	20.83	47.44
Reserves $\Rightarrow$ Offences against the property	20.83	68.99
Offences against the property $\Rightarrow$ Parks	36.25	82.56
Parks $\Rightarrow$ Offences against the property	36.25	71.10
Offences against the property $\Rightarrow$ Schools	33.42	76.11
Schools $\Rightarrow$ “Offences against the property”	33.42	75.31
Other offences $\Rightarrow$ Reserves	17.81	50.47
Reserves $\Rightarrow$ Other offences	17.81	58.97
Other offences $\Rightarrow$ Parks	29.90	84.74
Parks $\Rightarrow$ Other offences	29.90	58.64
Other offences $\Rightarrow$ Schools	28.35	80.36
Schools $\Rightarrow$ Other offences	28.35	63.89

under study (in this case, crime types and features) grows. Thus, it is almost impossible for analysts to find interesting associations manually. The horizontal-view approach automatically generates strong CSARs with user-specified minimum *CwC* and minimum *SwC*. For instance, with 30% minimum *SwC* and 75% minimum *CwC*, three strong CSARs, when the size of the antecedent and the size of consequent are 1, are discovered. These are as follows:

$$\text{Offences against property} \Rightarrow \text{Parks} \text{ (36.25\% } SwC, 82.56\% CwC), \quad (8.8)$$

$$\text{Offences against property} \Rightarrow \text{Schools} \text{ (33.42\% } SwC, 76.11\% CwC), \quad (8.9)$$

$$\text{Schools} \Rightarrow \text{Offences against property} \text{ (33.42\% } SwC, 75.31\% CwC). \quad (8.10)$$

These rules imply that most “offences against property” (more than 75%) are taking

place around parks and schools. Further, with more than 75% *CwC*, locations of schools imply occurrences of “offences against property”. Many hypotheses can now be derived for input into CDA. For example, this very simple illustration already suggests that possibly residents living around schools shall consider additional actions for reducing crime against their property.

8.2.4 Discussion

For the analysis of crime data, KDD techniques have been applied by the Federal Bureau of Investigation (FBI) as a part of the investigation of the Oklahoma City bombing the Unabomber case, and many lower-profile crimes [15]. KDD has also been used by Treasury Department of the US to hunt for suspicious patterns in international funds transfer records; patterns that may indicate money laundering or fraud [15]. Australian examples are the Health Insurance Commission of Australia using KDD to investigate fraud within the Medicare system [65] and National Roads and Motorists’ Association (NRMA) Insurance Ltd [145]. However, in the above examples, the techniques have been limited to the attribute-oriented side of the data. The where and when are crucial for patterns of crime. Recent progress towards developing GIS for crime analysis emphasizes the relevance that geo-reference has to understand patterns in crime [67]. In a limited exploratory approach [112], systems have been developed to geographically visualize patterns of vehicle crime, domestic burglaries, drug-related crime and public disorder in inner city areas. For example, hypotheses are typically human-generated and reinforced with the display of the most contrasting socio-demographic characteristics of areas with high levels of criminal activity. This type of approach may lead to hypotheses that relate the location patterns of crime to the negative socio-demographic characteristics of areas associated with these patterns [53]. The application of GIS technology to crime analysis by mapping and display may use statistical methods to produce maps delineating risk surfaces for crime data [91]. These are confirmatory analyses that, for example, have contributed to the consideration of geographic distribution of criminal behavior as

a primary factor for planning residential neighborhoods [53]. However, today vast collections of spatio-temporal data are gathered without any previous hypothesis, and less as parts of a structured experiment. Moreover, it is impossible that trained researchers examine all possible interesting patterns in such huge amounts of data. We require an intelligent assistant to process the data and to autonomously (or at least with very little guidance) analyze data. We have presented methods and algorithms that increase even more the capacity of GIS to become intelligent pattern spotters beyond just as repositories to store, manipulate and retrieve data. Note that, this direction complements traditional statistical analyses on geo-referenced data. The intent is to design and deploy autonomous partners in suggesting hypotheses in KDD process. We have developed algorithms that enable this exploratory capability within the context of spatial data. These approaches are not application-dependent, since these algorithms do not require task-specific concept hierarchies. They are applicable to environmental databases, marine databases, underground databases and astronomy databases.

8.3 Remarks

Clustering methods are becoming more sophisticated and effective. Post-clustering processes that seek to identify possible lures (positive associations) are now in demand. Detection of cluster boundaries is a natural choice for reasoning about clusters such as matching boundaries with various feature data [80, 125], cluster polygonization and mining associations [39]. Traditional statistical analysis including χ^2 test, *nearest neighbor distances*, the *K-function* and Moran's *I* and Geary's *c* statistics focuses on the spatial correlation structure (spatial dependence or autocorrelation). This correlation analysis is confined to symmetric relationships. Thus, it is difficult to find non-symmetric causal factors for spatial clusters with this symmetric approach. Association analysis is a solid candidate for mining causal factors since it reveals non-symmetric causal relationships.

In this chapter, we introduced a new mining associations method using cluster analysis in order to detect positive associations among multivariate datasets. It is a data-oriented spatial-dominant generalization method. We first introduce a fast and robust cluster polygonization algorithm that is well-suited for massive spatial databases. This approach minimizes user-oriented bias and maximizes user-friendliness by automatically deriving cluster boundaries from data. And then, we develop two general purpose mining multivariate positive associations using the cluster polygonization. Instead of using task-specific predicates, these approaches use cluster structures to effectively and efficiently mine multivariate positive associations hidden in large spatial databases. These proposed algorithms seek to find causal factors for spatial aggregations and suggests manageable number of highly plausible hypotheses for CDA.

Chapter 9

Conclusions and Future Work

This chapter summarizes the content of this thesis, discusses implementation issues and suggests future work arising from its findings.

9.1 Summary

Since the specialist meeting of the research initiative on “Very Large Spatial Databases” at Santa Barbara in 1989 [132] organized by the National Center for Geographic Information and Analysis (NCGIA), management of large spatial databases (modeling and structuring, storage and retrieval, parallelism and integration of spatial data) has gained great attention. Advances in data mining and KDD technologies have been providing an impetus to the movement of mining very large spatial databases to find hidden spatial patterns. The recent specialist meeting on “Discovering Geographic Knowledge in Data-Rich Environments” at Washington [102] in 1999 by the NCGIA, emphasizes the need for intelligent pattern spotters in GIS.

In this thesis, we proposed a spatial-dominant generalization approach that mines multivariate positive associations using clustering analysis. First, we proposed a gen-

eral framework of multi-purpose exploratory spatial clustering embedded within the Template-Method Pattern. Although the template provides a variety of functionality for spatial clustering in various settings, its object-oriented structure allows users to plug in application-dependent functionality as desired. We designed and implemented Short-Long clustering based on the framework. It incorporates the peculiar characteristics of spatial data that make space special. Thus, it overcomes the drawbacks of traditional clustering approaches that fail to identify some geographically interesting concentrations. The Short-Long clustering method is data-driven rather than model-driven. It derives values for parameters from data and thus maximizes user-friendliness and minimizes user-oriented bias. Its effectiveness is not limited to shapes of clusters. It considers both local interactions and global interactions, so it is able to detect global hot spots and localized excesses. It is capable of finding clusters of arbitrary shapes, clusters of heterogeneous densities, clusters of different sizes, closely located high-density clusters, clusters connected by multiple links, sparse clusters near to high-density clusters and clusters containing clusters. It is robust to noise and insensitive to the presentation order of data points.

Sheer volume of spatial data stored in GIS is not the only concern. The heterogeneity of multi-layered datasets is another [132]. This multi-purpose exploratory clustering allows users to explore heterogeneous types of data stored in different layers. We illustrated the extensibility of Short-Long clustering to other proximity graphs. This allows “what-if” analysis with the Short-Long clustering method using various proximity graphs. The capability of Short-Long clustering to use other proximity graphs, especially $k\text{-}UNN(P)$, allows it to find quality volumetric spatial concentrations in three-dimensions within $O(n \log n)$ time. Since this variant of $k\text{-}NN(P)$ is linear in size and subquadratic in time for any dimension, Short-Long clustering could potentially be used for high-dimensional data. This enables the incorporation of aspatial dimensions and time dimension. Also in some spatial settings, interactions between map objects are obstructed by obstacles, and non-Euclidean metrics approximate better real-world situations. We provided solutions to the problem of clustering in the presence of obstacles and proposed at least four approaches to handle obstacles with

the Short-Long clustering method. Each approach offers valid modeling of geometry and different computational trade-offs. The availability of the four approaches allows users to prefer one above the other, depending on the exploratory task at hand. The presence of obstacles is not the only factor invalidating the straight path between map objects. In some spatial settings, the Manhattan distance and the Dominance distance approximate better real-world situations. This is true especially only where axis-parallel paths are available. We further investigated the Short-Long clustering method in the Manhattan metric and the Dominance metric, also in the absence and presence of obstacles. This investigation extended the four obstacle handling algorithms to these non-Euclidean metrics. This versatility of the Short-Long clustering method results in a multi-purpose clustering tool that identifies spatial aggregations in various spatial settings regardless of the heterogeneity of multi-layered datasets.

Often, single-level clustering is not sufficient for revealing cluster hierarchies hidden in complex real datasets. We discussed the problem of traditional hierarchical clustering and the difficulty of visualizing cluster hierarchies. We proposed multi-level Short-Long clustering to suggest possible hierarchical spatial concentrations with its associated visualization. Multi-level Short-Long clustering is top-down and divisive. It recursively split clusters into subclusters unless they exhibit point-centric statistics similar to CSR. The termination condition is derived from the analysis of intra-cluster edges in $DD(P)$. If intra-cluster edges of a cluster exhibit no spatial concentrations, then multi-level clustering stops splitting. Otherwise, it keeps splitting until the termination condition is met. Thus, multi-level Short-Long clustering does not require a termination condition that is hard to determine. Note that, termination conditions are user-supplied in typical hierarchical clustering methods. Multi-level Short-Long clustering automatically reveals a termination condition and a cluster hierarchy for P . Since the depth of cluster hierarchies of multi-level Short-Long clustering is bounded by $O(\log n)$, it requires only $O(n \log n)$ time. This is a significant advantage over traditional hierarchical clustering that typically requires $O(n^2)$ time. The commonly used dendrograms for small datasets become too large and complex for large datasets. They become unreadable and impractical. We introduced a new hierarchical visual-

ization that is associated with cluster weights. Pendrograms are not only more readable, but more informative. They allow the user to compare and contrast intra-level clusters and inter-level clusters. Each node representing a cluster in a Pendrogram contains the size and distribution of the cluster. Thus, a Pendrogram summarizes the distribution of P . This visualization provides qualitative and quantitative overview of complex cluster hierarchies and assists focused analysis in a more readable form. These are very important aspects in data visualization [57].

Interpretability and usability of clustering results are of fundamental importance [61] (refer to the definition of KDD in Chapter 1). We proposed an automatic pattern spotter that finds high level description of clusters, and an effective and efficient cluster polygonization process towards mining causal associations. The cluster polygonization process analyzes the distribution of intra-cluster edges and the distribution of inter-cluster edges in Delaunay diagrams. It approximates shapes of clusters and suggests polygons of clusters. Then, polygons of clusters are used for intersection/overlay to measure associations. This approach overcomes the problem of traditional mining associations using partial information. We illustrated that approaches based on medoids and partial boundaries might miss interesting relationships and are unsuitable for reasoning about clusters. Using cluster polygonization algorithms, we can mine causal positive associations without domain-specific concept hierarchies. We can automatically derive asymmetric causal relationships from the cluster structure of a dataset.

9.2 Implementation issues

We have developed an application to demonstrate and visualize Short-Long clustering and the mining of associations using cluster polygonization. It is a stand-alone application designed with an object-oriented approach. However, it can be easily plugged into GIS using scripting languages. For example, using the language AV-

ENUE for ArcView ¹. In Section 3.3, we presented the framework for multi-purpose boundary-based spatial clustering. This is the design framework of our application and it was implemented in the C++ programming language using LEDA (Library of Efficient Data types and Algorithms) version 4.2 [101], CGAL (Computational Geometry Algorithms Library) version 2.1 ², and the library **delvor** [73]. The graphical user interface of the application is shown in Figure 9-1.

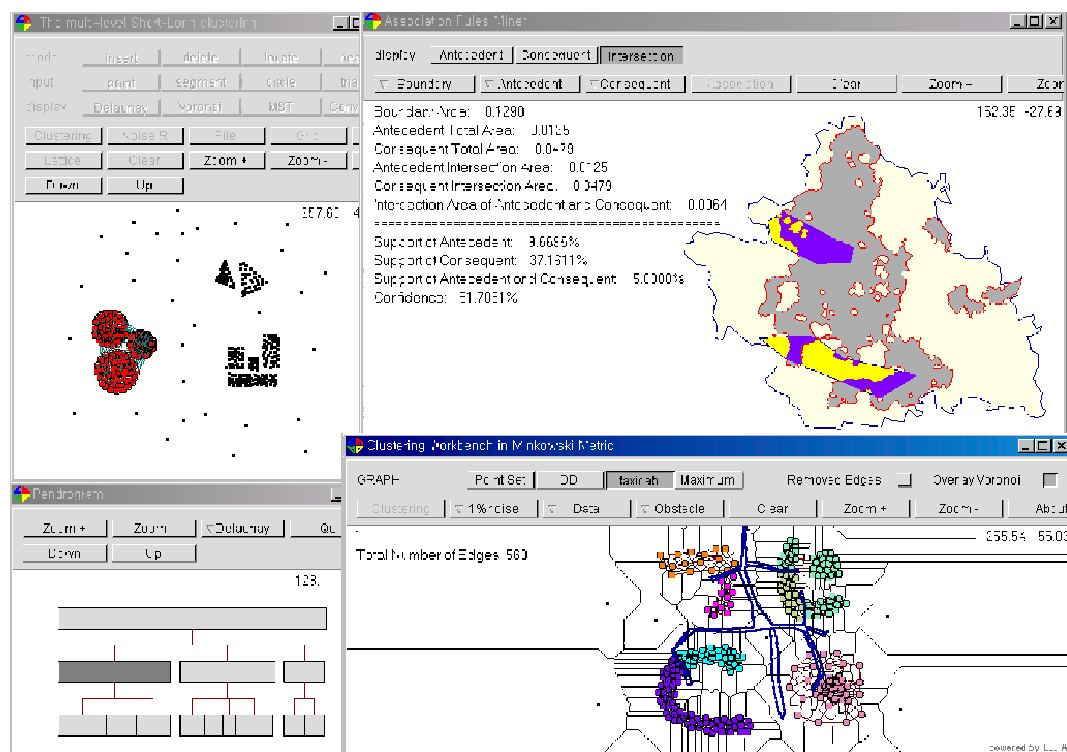


Figure 9-1: Graphical user interface of Short-Long clustering towards mining associations.

9.3 Immediate avenues for further research

The advances presented in this thesis can be continued in the following directions.

¹ArcView is a trademark of ESRI. For documentation see <http://www.esri.com/>.

²For documentation and code see <http://www.algorithmic-solutions.com/>.

- Geoinformation is dynamic in many domains. The datasets may experience deletions, insertions and updates. This would require an incremental or dynamic clustering approach. A little reflection shows that although the Short-Long clustering methodology uses some global point-centric statistics, these can be updated efficiently.

For example, the 2D Delaunay diagram can be updated for the insertion or deletion of a point within $O(\log n)$ time. The insertion or deletion of a point affects only $O(1)$ edges (and this is in a very localized region of the update). Therefore, point centric-statistics like $Local_St_Dev(p_i)$ will be updated for $O(1)$ points p_i . Since the degree of a point is $O(1)$, each update would require $O(1)$ time. Global point-centric statistics like $Mean_St_Dev(P)$ can be updated within $O(1)$ time by storing the number of points n and $\sum_{i=1}^n Local_St_Dev(p_i)$ (which can be updated within $O(1)$ time). Finally, the classification of edges performed in the three phases of the Short-Long clustering method can be balanced and amortized. That is, not all edges will be re-classified, but only those in the vicinity of the update. After $O(n)$ updates, the entire classification can be repeated.

It remains to implement and thoroughly test these ideas for the different proximity graphs where the Short-Long clustering method applies.

- We have presented association rule mining [3] as an exploratory technique to analyze large volumes of data towards the discovery of hypotheses and possible causal relationships. The development and refinement of algorithms for finding association rules remains a core activity of data mining research, while issues like learning classification rules has been shared with research in machine learning, and estimation has been shared with research in statistics. Despite the large amount of research on association rules, mining single-dimensional Boolean association rules remains the central framework of the problem. Association rule mining has been extended in natural directions.

A generalization closer to the context of spatio-temporal mining is the consider-

ation of values in a transaction that are not Boolean, but quantitative [134]. In this context, a transaction may involve numerical values. However, the line of attack for this approach essentially consists of discretizing the range of numerical values of the corresponding variable. Once this is done, the problem reverts back to Boolean association rules, since the predicates are Boolean predicates of the form “Transaction T contains variable x_j in range r_j ”.

Other approaches aim at reflecting weights of items in transactional databases [21, 120, 143]. These generalizations are insufficient for the type of spatio-temporal applications we have in mind. The problem is that the set of transactions labeled by their identifiers is discrete. No matter how large, it is a finite set. We need to extend association rule mining to where the set of transactions is a continuum in space and/or time. That is, the challenge remains to incorporate cluster densities in spatial association mining that incorporates the notion of population-at-risk.

The exploratory analysis of spatial data with association rules is essentially performed under four models, namely, the *reference feature model* [83], the *window centric model* [130, 137], the *event centric model* [130] and the *spatio-dominant generalization model* [39, 44, 94, 108]. We believe that the approach presented here can be extended to handle contexts involving geographical data considering the density of each region for each layer.

- Naturally, the use of our approach still faces the challenge of its extension to various types of data including binary data, categorical (nominal) data, ordinal data, continuous data, text data (meta-data), areal data and multimedia data. This avenue of research needs to establish first a suitable model of proximity information. Once a discrete proximity graph is found, the remaining parts of our approach should apply with little modification.
- The implicit model of the Short-Long clustering method is a partition. Although the notion that a point belongs to only one cluster is relaxed by our introduction of the multi-level Short-Long clustering approach, the clusters themselves

constitute a partition of the data at each level of the multi-level clustering hierarchy. However, there are many natural situations in which the clusters are not to form a partition, and data points can belong to two or more clusters. This can be modeled by probabilistic approaches of fuzzy membership.

We believe that an immediate direction for further research is the extension of our method to less crisp classification of edges. Such a classification could then result in probabilistic clusters or fuzzy clustering which could represent situations where clusters overlap.

- Finally, it is a matter of practical reality that very large datasets cannot be managed in main memory. Data structures like R-trees or Quad-trees are common in commercial GIS to manage datasets, in particular point datasets. Therefore, data structures like Delaunay diagrams presented here cannot be expected to fit in main memory for the very large datasets that some data mining applications have in mind. The proximity graphs used by Short-Long clustering are embedded in the space of the point data. For example, the 2D Delaunay diagram is a planar embedding made mostly of triangles. It remains to implement the indexing of this planar embedding with a data structure like R-trees so that the operation on the Delaunay diagram required by Short-Long clustering can be accessed through this spatial index. A simple alternative in this case would be to place each Delaunay diagram face in a bounding box and insert the boxes into the R-tree, with some pointers as well to create the links of the planar map. This approach requires delicate implementation but it is certainly promising.

Bibliography

- [1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In L. M. Haas and A. Tiwary, editors, *Proceedings of the ACM SIGMOD'98 International Conference on Management of Data*, pages 94–105, Seattle, Washington, 1998. ACM Press.
- [2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining Association Rules between Sets of Items in Large Databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the ACM SIGMOD'93 International Conference on Management of Data*, pages 207–216, Washington, D.C., 1993. ACM Press.
- [3] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, Santiago de Chile, Chile, 1994. Morgan Kaufmann.
- [4] N. Ahuja. Dot Pattern Processing Using Voronoi Neighborhoods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4(3):336–343, 1982.
- [5] M. S. Aldenderfer and R. K. Blashfield. *Cluster Analysis*. Sage Publications, Beverly Hills, 1984.
- [6] M. Ankerst, M. M. Breunig, H. P. Kriegel, and J. Sander. OPTICS: Ordering Points To Identify the Clustering Structure. In A. Delis, C. Faloutsos, and

- S. Ghandeharizadeh, editors, *Proceedings of the ACM SIGMOD'98 International Conference on Management of Data*, pages 49–60, Philadelphia, Pennsylvania, 1999. ACM Press.
- [7] L. Anselin. What is Special About Spatial Data? Alternative Perspectives on Spatial Data Analysis. Technical Report 89-4, National Center for Geographic Information and Analysis, University of California, 1989. [Available at http://www.ncgia.ucsb.edu/Publications/Tech_Reports/89/89-4.DOC].
- [8] L. Anselin. Spatial Data Analysis with GIS: An Introduction to Application in the Social Sciences. Technical Report 92-10, National Center for Geographic Information and Analysis, University of California, 1992. [Available at http://www.ncgia.ucsb.edu/Publications/Tech_Reports/92/92-10.PDF].
- [9] L. Anselin. Interactive Techniques and Exploratory Spatial Data Analysis. In P. A. Longley, M. F. Goodchild, D. J. Maguire, and D. W. Rhind, editors, *Geographical Information Systems: Principles and Technical Issues*, volume 1, pages 253–266. John Wiley & Sons, New York, second edition, 1999.
- [10] T. C. Bailey and A. C. Gatrell. *Interactive Spatial Analysis*. Longman Scientific & Technical, Harlow, UK, 1995.
- [11] J. D. Banfield and A. E. Raftery. Model-based Gaussian and Non-Gaussian Clustering. *Biometrics*, 49:803–821, 1993.
- [12] N. Beckmann, H. P. Kriegel, R. Schneider, and B. Seeger. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In H. Garcia-Molina and H. V. Jagadish, editors, *Proceedings of the ACM SIGMOD'90 International Conference on Management of Data*, pages 322–331, Atlantic City, NJ, 1990. ACM Press.
- [13] M. W. Bern, D. Eppstein, and J. R. Gilbert. Provably Good Mesh Generation. *Journal of Computer and System Sciences*, 48(3):384–409, 1994.

- [14] T. Bernhardsen. *Geographic Information System*. John Wiley & Sons, New York, second edition, 1999.
- [15] M. J. A. Berry and G. Linoff. *Data Mining Techniques — for Marketing, Sales and Customer Support*. John Wiley & Sons, New York, 1997.
- [16] J. E. Besag and J. Newell. The Detection of Clusters in Rare Diseases. *Journal of the Royal Statistical Society A*, 154:143–155, 1991.
- [17] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.
- [18] B. N. Boots. Using Angular Properties of Delaunay Triangles to Evaluate Point Patterns. *Geographical Analysis*, 18(3):250–260, 1986.
- [19] P. S. Bradley, U. M. Fayyad, and C. Reina. Scaling Clustering Algorithms to Large Databases. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 9–15, New York City, New York, 1998. AAAI Press.
- [20] P. A. Burrough. *Principles of Geographical Information Systems for Land Resources Assessment*. Oxford University Press, New York, 1986.
- [21] C. H. Cai, A. W. C. Fu, C. H. Cheng, and W. W. Kwong. Mining Association Rules with Weighted Items. In *Proceedings of the 1998 International Database Engineering and Applications Symposium*, pages 68–77, Wales, U.K, 1998. IEEE Computer Society.
- [22] G. Celeux and G. Govaert. Gaussian Parsimonious Clustering Models. *Pattern Recognition*, 28(5):781–793, 1995.
- [23] V. Cherkassky and F. Muller. *Learning from Data - Concept, Theory and Methods*. John Wiley & Sons, New York, 1998.
- [24] L. P. Chew. Constrained Delaunay triangulations. *Algorithmica*, 4:97–108, 1989.

- [25] J. Dangermond. A Classification of Software Components Commonly Used in Geographic Information Systems. In D. J. Peuquet and D. F. Marble, editors, *In Introductory Readings in Geographic Information Systems*, pages 30–51. Taylor & Francis, London, 1990.
- [26] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.
- [27] B. D. Dent. *Cartography: Thematic Map Design*. WCB McGraw Hill, Boston, fifth edition, 1999.
- [28] P. J. Diggle. *Statistical Analysis of Spatial Point Patterns*. Academic Press, London, 1983.
- [29] P. Eades, Lin. X., and R. Tamassia. An Algorithm for Drawing a Hierarchical Graph. *International Journal of Computational Geometry and Applications*, 6(2):145–155, 1996.
- [30] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the Shape of a Set of Points in the Plane. *IEEE Transactions on Information Theory*, 29(4):551–559, 1983.
- [31] C. Eldershaw and M. Hegland. Cluster Analysis using Triangulation. In B. J. Noye, M. D. Teubner, and A. W. Gill, editors, *Computational Techniques and Applications: CTAC97*, pages 201–208. World Scientific, Singapore, 1997.
- [32] S. Eptner and M. S. Krishnamoorthy. A Multiple-Resolution Method for Edge-Centric Data Clustering. In *Proceedings of the 1999 ACM CIKM International Conference on Information and Knowledge Management*, pages 491–498, Kansas City, Missouri, 1999. ACM Press.
- [33] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In E. Simoudis,

- J. Han, and U. M. Fayyad, editors, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, Portland, Oregon, 1996. AAAI Press.
- [34] V. Estivill-Castro. Hybrid Genetic Algorithms Are Better for Spatial Clustering. In R. Mizoguchi and J. Slaney, editors, *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, Lecture Notes in Artificial Intelligence 1886, pages 424–434, Melbourne, Australia, 2000. Springer.
- [35] V. Estivill-Castro and M. E. Houle. Robust Clustering of Large Geo-referenced Data Sets. In N. Zhong and L. Zhou, editors, *Proceedings of the 3rd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Lecture Notes in Artificial Intelligence 1574, pages 327–337, Beijing, China, 1999. Springer.
- [36] V. Estivill-Castro and I. Lee. AMOEBA: Hierarchical Clustering Based on Spatial Proximity Using Delaunay Diagram. In P. Forer, A. G. O. Yeh, and J. He, editors, *Proceedings of the 9th International Symposium on Spatial Data Handling*, pages 7a.26–7a.41, Beijing, China, 2000.
- [37] V. Estivill-Castro and I. Lee. AUTOCLUST+: Automatic Clustering of Point-Data Sets in the Presence of Obstacles. In J. F. Roddick and K. Hornsby, editors, *Proceedings of the International Workshop on Temporal, Spatial and Spatio-Temporal Data Mining*, Lecture Notes in Artificial Intelligence 2007, pages 133–146, Lyon, France, 2000. Springer.
- [38] V. Estivill-Castro and I. Lee. AUTOCLUST: Automatic Clustering via Boundary Extraction for Mining Massive Point-Data Sets. In *Proceedings of the 5th International Conference on Geocomputation*, Kent, UK, 2000. GeoComputation CD-ROM: GC049.
- [39] V. Estivill-Castro and I. Lee. Data Mining Techniques for Autonomous Exploration of Large Volumes of Geo-referenced Crime Data. In D. V. Pullar, editor, *Proceedings of the 6th International Conference on Geocomputation*, Brisbane, Australia, 2001. GeoComputation CD-ROM.

- [40] V. Estivill-Castro and I. Lee. Fast Spatial Clustering with Different Metrics and in the Presence of Obstacles. In W. G. Aref, editor, *Proceedings of the 9th International Symposium on Advances in Geographic Information Systems*, pages 142–147, Atlanta, GA, 2001. ACM Press.
- [41] V. Estivill-Castro and I. Lee. Argument Free Clustering via Boundary Extraction for Massive Point-data Sets. *Computers, Environments and Urban Systems*, 26(4):315–334, 2002.
- [42] V. Estivill-Castro and I. Lee. Multi-Level Clustering and its Visualization for Exploratory Spatial Analysis. *Geoinformatica*, 6(2):123–152, 2002.
- [43] V. Estivill-Castro, I. Lee, and A. T. Murray. Criteria on Proximity Graphs for Boundary Extraction and Spatial Clustering. In D. Cheung, Q. Li, and G. Williams, editors, *Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Lecture Notes in Artificial Intelligence 2035, pages 348–357, Hong Kong, China, 2001. Springer.
- [44] V. Estivill-Castro and A. T. Murray. Discovering Associations in Spatial Data - An Efficient Medoid Based Approach. In X. Wu, K. Ramamohanarao, and K. B. Korb, editors, *Proceedings of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Lecture Notes in Artificial Intelligence 1394, pages 110–121, Melbourne, Australia, 1998. Springer.
- [45] V. Estivill-Castro and J. Yang. Non-crisp Clustering by Fast, Convergent, and Robust Algorithms. In L. D. Raedt and A. Siebes, editors, *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, Lecture Notes in Computer Science 2168, pages 103–114, Freiburg, Germany, 2001. Springer.
- [46] D. Fasulo. An Analysis of Recent Work on Clustering Algorithms. Technical Report 01-03-02, Department of Computer Science & Engineering, University of Washington, 1999. [Available at <ftp://ftp.cs.washington.edu/tr/2001/03/UW-CSE-01-03-02.PS.Z>].

- [47] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery: An Overview. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery in Databases*, pages 1–34. AAAI Press, Menlo Park, 1996.
- [48] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge Discovery and Data Mining: Towards a Unifying Framework. In E. Simoudis, J. Han, and U. M. Fayyad, editors, *Proceedings of the 2th International Conference on Knowledge Discovery and Data Mining*, pages 82–88, Portland, Oregon, 1996. AAAI Press.
- [49] C. Fraley. Algorithms for Model-Based Gaussian Hierarchical Clustering. *SIAM Journal on Scientific Computing*, 20(1):270–281, 1998.
- [50] C. Fraley and A. E. Raftery. How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis. *The Computer Journal*, 41(8):578–588, 1998.
- [51] K. R. Gabriel and R. R. Sokal. A New Statistical Approach to Geographic Variation Analysis. *Systematic Zoology*, 18:259–278, 1969.
- [52] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns*. Addison-Wesley, Reading, MA, 1995.
- [53] S. Gandhi and J. W. Brubbs. Crime Analysis Using GIS at the City of Orlando, Florida: Implications for Housing Policies. In *Proceedings of the Urban and Regional Information Systems Association Annual Conference*, pages 122–132, Washington, D.C., 1996. Urban and Regional Information System Association.
- [54] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman and Company, New York, 1979.
- [55] C. M. Gold. Problems with Handling Spatial Data - The Voronoi Approach. *Canadian Institute of Surveying and Mapping Journal*, 45(1):65–80, 1991.

- [56] P. Gould. Letting the Data Speak for Themselves. *Annals of the Association of American Geographers*, 71:166–176, 1981.
- [57] G. G. Grinstein and M. O. Ward. Introduction to Data Visualization. In U. M. Fayyad, G. G. Grinstein, and A. Wierse, editors, *Information Visualization in Data Mining and Knowledge Discovery*, pages 21–45. Morgan Kaufmann Publishers, San Francisco, USA, 2002.
- [58] S. Guha, R. Rastogi, and K. Shim. CURE: An Efficient Clustering Algorithm for Large Databases. In L. M. Haas and A. Tiwary, editors, *Proceedings of the ACM SIGMOD’98 International Conference on Management of Data*, pages 73–84, Seattle, Washington, 1998. ACM Press.
- [59] S. Guha, R. Rastogi, and K. Shim. ROCK: A Robust Clustering Algorithm for Categorical Attributes. In *Proceedings of the 15th International Conference on Data Engineering*, pages 512–521, Sydney, Australia, 1999. IEEE Computer Society.
- [60] J. Han and Y. Fu. Discovery of Multiple-Level Association Rules from Large Databases. In U. Dayal, P. M. D. Gray, and S. Nishio, editors, *Proceedings of the 21st International Conference on Very Large Data Bases*, pages 420–431, Zurich, Switzerland, 1995. Morgan Kaufmann.
- [61] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, San Francisco, C.A., 2000.
- [62] J. Han, M. Kamber, and K. H. Tung. Spatial Clustering Methods in Data Mining. In H. J. Miller and J. Han, editors, *Geographic Data Mining and Knowledge Discovery*, pages 188–217. Cambridge University Press, Cambridge, UK, 2001.
- [63] F. Harary. *Graph Theory*. Addison-Wesley, Reading, MA, 1969.
- [64] J. A. Hartigan. *Clustering Algorithms*. John Wiley & Sons, 1975.

- [65] H. He, W. Graco, and X. Yao. Application of Genetic Algorithm and k-Nearest Neighbour Method in Medical Fraud Detection. In B. McKay, X. Yao, C. S. Newton, J. H. Kim, and T. Furuhashi, editors, *Proceedings of the 2nd Asia-Pacific Conference on Simulated Evolution and Learning*, Lecture Notes in Computer Science 1585, pages 74–81, Canberra, Australia, 1998. Springer.
- [66] A. Hinneburg and D. A. Keim. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 58–65, New York City, New York, 1998. AAAI Press.
- [67] A. Hirschfield, P. Brown, and P. Todd. GIS and the Analysis of Spatially-Referenced Crime Data: Experiences in Merseyside. U. K. *Journal of Geographical Information Systems*, 9(2):191–210, 1995.
- [68] B. Hoch. Personalizing e-Commerce Data Mining Bolsters CRM Objectives. *Business Geographics*, March/April 2001.
- [69] F. Höppner, F. Klawonn, R. Kruse, and T. Runkler. *Fuzzy Cluster Analysis*. John Wiley & Sons, 1999.
- [70] Z. Huang. A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. In *Proceedings of the Second Workshop on Research Issues on Data Mining and Knowledge Discovery*, Tucson, Arizona, 1997.
- [71] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988.
- [72] A. K. Jain, M. N. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):264–323, 1999.
- [73] M. Jünger, V. Kaibel, and S. Thienel. Computing Delaunay Triangulations in Manhattan and Maximum Metric. Technical Report 94.174, Institut für Informatik, Universität zu Köln, 1994. [Available at <http://math.tu-berlin.de/~kaibel/papers/mandel.ps.gz>].

- [74] M. Kamber, J. Han, and J. Chiang. Metarule-Guided Mining of Multi-Dimensional Association Rules Using Data Cubes. In D. Heckerman, H. Mannila, and D. Pregibon, editors, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 207–210, Newport Beach, California, 1997. AAAI Press.
- [75] I. Kang, T. Kim, and K. Li. A Spatial Data Mining Method by Delaunay Triangulation. In *Proceedings of the 5th International Workshop on Advances in Geographic Information Systems*, pages 35–39, Las Vegas, Nevada, 1997. ACM Press.
- [76] G. Karypis, E. Han, and V. Kumar. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. *IEEE Computer: Special Issue on Data Analysis and Mining*, 32(8):68–75, 1999.
- [77] L. Kaufman and P. J. Rousseuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York, 1990.
- [78] J. M. Keil and C. A. Gutwin. The Delauney Triangulation Closely Approximates the Complete Euclidean Graph. In F. K. H. A. Denhe, J.-R. Sack, and N. Santoro, editors, *Proceedings of the First Workshop on Algorithms and Data Structures*, Lecture Notes in Computer Science 382, pages 47–56, Ottawa, Canada, 1989. Springer.
- [79] V. Klee. On the Complexity of d -dimensional Voronoi Diagrams. *Archiv de Mathematik*, 34:75–80, 1980.
- [80] E. M. Knorr, R. T. Ng, and D. L. Shilvock. Finding Boundary Shape Matching Relationships in Spatial Data. In M. Scholl and A. Voisard, editors, *Proceedings of the 5th International Symposium on Spatial Databases*, Lecture Notes in Computer Science 1262, pages 29–46, Berlin, Germany, 1997. Springer.
- [81] K. Koperski. *A Progressive Refinement Approach to Spatial Data Mining*. PhD thesis, Computing Science, Simon Fraser University, B.C., Canada, 1999.

- [82] K. Koperski, J. Adhikary, and J. Han. Spatial Data Mining: Progress and Challenges. In *Proceedings of the First Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 1–10, Montreal, Canada, 1996.
- [83] K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Information Databases. In M. J. Egenhofer and J. R. Herring, editors, *Proceedings of the 4th International Symposium on Large Spatial Databases*, Lecture Notes in Computer Science 951, pages 47–66, Portland, Maine, 1995. Springer.
- [84] E. F. Krause. *Taxicab Geometry*. Addison-Wesley, California, 1975.
- [85] D. Krznaric. *Progress in Hierarchical Clustering & Minimum Weight Triangulation*. PhD thesis, Computer Science, Lund University, Lund, Sweden, 1998.
- [86] C. Larman. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design*. Prentice Hall, New York, 1998.
- [87] D. T. Lee. Two-Dimensional Voronoi Diagrams in the l_p -Metric. *Journal of the Association for Computing Machinery*, 27(4):604–618, 1980.
- [88] I. Lee. MESCAT: Multi-purpose Exploratory Spatial Clustering Analysis Toolkit. In J. Fong and M. K. Ng, editors, *Proceedings of the International Workshop on Mining Spatial and Temporal Data*, pages 112–121, Hong Kong, China, 2001. City University of Hong Kong.
- [89] I. Lee and V. Estivill-Castro. Effective and Efficient Boundary-based Clustering for Three-dimensional Geoinformation Studies. In H. Lu and S. Spaccapietra, editors, *Proceedings of the 3rd International Symposium on Cooperative Database Systems for Advanced Applications*, pages 87–96, Beijing, China, 2001. IEEE Computer Society.
- [90] I. Lee and V. Estivill-Castro. Polygonization of Point Clusters through Cluster Boundary Extraction for Geographical Data Mining. In D. Richardson and P. v. Oostrom, editors, *Proceedings of the 10th International Symposium on Spatial Data Handling*, pages 27–40, Ottawa, Canada, 2002. Springer.

- [91] J. Lee. Point Pattern Analysis with Density Estimation for Frequency Data. In *Proceedings of the GIS/LIS-95 Annual Conference and Exposition*, pages 598–607. American Society of Photogrammetry and Remote Sensing, 1995.
- [92] G. Liotta. Low Degree Algorithms for Computing and Checking Gabriel Graphs. Technical Report 96-28, Department of Computer Science, Brown University, 1996. [Available at <ftp://ftp.cs.brown.edu/pub/techreports/96/cs96-28.ps.Z>].
- [93] P. A. Longley, M. F. Goodchild, D. J. Maguire, and D. W. Rhind. Introduction. In P. A. Longley, M. F. Goodchild, D. J. Maguire, and D. W. Rhind, editors, *Geographical Information Systems: Principles and Technical Issues*, volume 1, pages 1–20. John Wiley & Sons, New York, second edition, 1999.
- [94] W. Lu, J. Han, and B. C. Ooi. Discovery of General Knowledge in Large Spatial Databases. In *Proceedings of 1993 Far East Workshop on Geographic Information Systems*, pages 275–289, Singapore, 1993.
- [95] J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the 5th Berkeley Symposium on Maths and Statistics Problems*, volume 1, pages 281–297, 1967.
- [96] R. J. Marshall. A Review of Methods for the Statistical Analysis of Spatial Patterns of Disease. *Journal of the Royal Statistical Society A*, 154(3):421–441, 1991.
- [97] C. J. Matheus, P. K. Chan, and G. Piatetsky-Shapiro. Systems for Knowledge Discovery in Databases. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):903–913, 1993.
- [98] D. W. Matula and R. R. Sokal. Properties of Gabriel Graphs Relevant to Geographic Variation Research and the Clustering of Points in the Plane. *Geographical Analysis*, 12(3):205–222, 1980.
- [99] I. L. McHarg. *Design with Nature*. Natural History Press, New York, 1969.

- [100] L. McKee. Geography Connects Cyberspace with the Real World. *GEO World*, February 2001.
- [101] K. Mehlhorn and S. Näher. *LEDA A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, Cambridge, UK, 1999.
- [102] H. J. Miller and J. Han. Discovering Geographic Knowledge in Data Rich Environments: Report of a Specialist Meeting. Varenus Reports, National Center for Geographic Information and Analysis, University of California, 1999. [Available at http://www.ncgia.ucsb.edu/Publications/Varenus_Reports/KDD-final-report.PDF].
- [103] H. J. Miller and J. Han. *Geographic Data Mining and Knowledge Discovery: An Overview*. Cambridge University Press, Cambridge, UK, 2001.
- [104] A. T. Murray and V. Estivill-Castro. Cluster Discovery Techniques for Exploratory Spatial Data Analysis. *International Journal of Geographical Information Science*, 12(5):431–443, 1998.
- [105] A. T. Murray, I. McGuffog, J. S. Western, and P. Mullins. Exploratory Spatial Data Analysis Techniques for Examining Urban Crime. *British Journal of Criminology*, 41:309–329, 2001.
- [106] A. T. Murray and T. Shyy. Integrating Attribute and Space Characteristics in Choropleth Display and Spatial Data Mining. *International Journal of Geographical Information Science*, 14:649–667, 2000.
- [107] F. Murtagh. Comments on Parallel Algorithms for Hierarchical Clustering and Cluster Validity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):1056–1057, 1992.
- [108] R. T. Ng and J. Han. Efficient and Effective Clustering Method for Spatial Data Mining. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 144–155, Santiago de Chile, Chile, 1994. Morgan Kaufmann.

- [109] A. Okabe, B. N. Boots, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, West Sussex, 1992.
- [110] A. Okabe, B. N. Boots, K. Sugihara, and S. N. Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, West Sussex, second edition, 2000.
- [111] S. Openshaw. A Mark 1 Geographical Analysis Machine for the Automated Analysis of Point Data Sets. *International Journal of Geographical Information Science*, 1(4):335–358, 1987.
- [112] S. Openshaw. Two exploratory space-time-attribute pattern analysers relevant to GIS. In A. S. Fotheringham and P. A. Rogerson, editors, *Spatial Analysis and GIS*, pages 83–104. Taylor & Francis, London, 1994.
- [113] S. Openshaw. Geographical Data Mining: Key Design Issues. In *Proceedings of the 4th International Conference on Geocomputation*, 1999.
- [114] S. Openshaw and P. J. Taylor. A Million or so Correlation Coefficients: Three Experiments on the Modifiable Areal Unit Problem. In N. Wrigley, editor, *Statistical Methods in the Spatial Sciences*, pages 127–144. Routledge & Kegan Paul, London, 1979.
- [115] J. O’Rourke. *Computational Geometry in C*. Cambridge University Press, New York, 1993.
- [116] J. F. Paper and D. J. Maguire. Design Models and Functionality in GIS. *Computers and Geosciences*, 18(4):387–394, 1992.
- [117] J. S. Park, M. S. Chen, and P. S. Yu. An Effective Hash Based Algorithm for Mining Association Rules. In M. J. Carey and D. A. Schneider, editors, *Proceedings of the ACM SIGMOD’95 International Conference on Management of Data*, pages 175–186, San Jose, California, 1995. ACM Press.

- [118] C. Posse. Hierarchical Model-based Clustering for Large Datasets. Technical Report 363, Department of Statistics, University of Washington, 1999. [Available at <http://www.stat.washington.edu/www/research/reports/1999/tr363.ps>].
- [119] D. Pullar and M. J. Egenhofer. Towards Formal Definitions of Spatial Relationships among Spatial Objects. In *Proceedings of the 3rd International Symposium on Spatial Data Handling*, pages 225–242, Sydney, Australia, 1988. International Geographical Union.
- [120] G. D. Ramkumar and A. N. Swami. Clustering Data Without Distance Functions. *Data Engineering Bulletin*, 21(1):9–14, 1998.
- [121] J. R. Robertson. Airborne Imagery Fuels GIS Growth. *GEO World*, March 2001.
- [122] J. F. Roddick, K. Hornsby, and M. Spiliopoulou. An Updated Bibliography of Temporal, Spatial, and Spatio-temporal Data Mining Research. In J. F. Roddick and K. Hornsby, editors, *Proceedings of the International Workshop on Temporal, Spatial and Spatio-Temporal Data Mining*, Lecture Notes in Artificial Intelligence 2007, pages 147–164, Lyon, France, 2000. Springer.
- [123] J. F. Roddick and B. G. Lees. Paradigms for Spatial Data Warehousing for Geographic Knowledge Discovery. In H. J. Miller and J. Han, editors, *Geographic Data Mining and Knowledge Discovery: An Overview*. Taylor and Francis, 2001.
- [124] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley & Sons, New York, 1987.
- [125] J. Sander. *Generalized Density-Based Clustering for Spatial Data Mining*. PhD thesis, Computer Science, University of Munich, München, Germany, 1998.
- [126] A. Savasere, E. Omiecinski, and S. Navathe. An Efficient Algorithm for Mining Association Rules in Large Databases. In U. Dayal, P. M. D. Gray, and S. Nishio, editors, *Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 432–444, Zurich, Switzerland, 1995. Morgan Kaufmann.

- [127] E. Schikuta and M. Erhart. The BANG-Clustering System: Grid-Based Data Analysis. In X. Liu, P. R. Cohen, and M. R. Berthold, editors, *Proceedings of the 2nd International Symposium on Intelligent Data Analysis*, Lecture Notes in Computer Science 1280, pages 513–524, London, UK, 1997. Springer.
- [128] T. Schreiber. A Voronoi Diagram Based Adaptive K-Means-Type Clustering Algorithm for Multidimensional Weighted Data. In H. Bieri and H. Noltemeier, editors, *Proceedings of the 7th International Symposium on the Advances in Spatial and Temporal Databases*, Lecture Notes in Computer Science 553, pages 265–275. Springer, 1991.
- [129] G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases. In A. Gupta, O. Shmueli, and J. Widom, editors, *Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 428–439, New York City, New York, 1998. Morgan Kaufmann.
- [130] S. Shekhar and Y. Huang. Discovering Spatial Co-location Patterns: A Summary of Results. In C. S. Jensen, M. Schneider, and V. J. Seeger, B. Tsotras, editors, *Proceedings of the 7th International Symposium on the Advances in Spatial and Temporal Databases*, Lecture Notes in Computer Science 2121, pages 236–256, Redondo Beach, CA, 2001. Springer.
- [131] R. Sibson. SLINK: An Optimally Efficient Algorithm for the Single-link Cluster Method. *The Computer Journal*, 16(1):30–34, 1973.
- [132] T. R. Smith and A. Frank. Report on Workshop on Very Large Spatial Databases. Technical Report 89-13, National Center for Geographic Information and Analysis, University of California, 1989. [Available at http://www.ncgia.ucsb.edu/Publications/Tech_Reports/89/89-13.DOC].
- [133] E. Son, I. Kang, T. Kim, and K. Li. A Spatial Data Mining Method by Clustering Analysis. In R. Laurini, K. Makki, and N. Pissinou, editors, *Proceedings*

- of the 6th International Symposium on Advances in Geographic Information Systems*, pages 157–158, Washington, DC, 1998. ACM Press.
- [134] R. Srikant and R. Agrawal. Mining Quantitative Association Rules in Large Relational Tables. In H. V. Jagadish and I. S. Mumick, editors, *Proceedings of the ACM SIGMOD'96 International Conference on Management of Data*, pages 1–12, Montreal, Canada, 1996. ACM Press.
- [135] W. R. Tobler. A Computer Movie Simulating Urban Growth in the Detroit Region. *Economic Geography*, 46(2):234–240, 1970.
- [136] G. T. Toussaint. The Relative Neighbourhood Graph of a Finite Planar Set. *Pattern Recognition*, 12:261–268, 1980.
- [137] I. Tsoukatos and D. Gunopoulos. Efficient Mining of Spatiotemporal Patterns. In C. S. Jensen, M. Schneider, and V. J. Seeger, B. Tsotras, editors, *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases*, Lecture Notes in Computer Science 2121, pages 425–442, Redondo Beach, CA, 2001. Springer.
- [138] S. Tsur, J. D. Ullman, S. Abiteboul, C. Clifton, R. Motwani, S. Nestorov, and A. Rosenthal. Query Flocks: A Generalization of Association-Rule Mining. In L. M. Haas and A. Tiwary, editors, *Proceedings of the ACM SIGMOD'98 International Conference on Management of Data*, pages 1–12, Seattle, Washington, 1998. ACM Press.
- [139] A. K. H. Tung, J. Hou, and J. Han. COE: Clustering with Obstacles Entities, A Preliminary Study. In H. Terano, T. abd Liu and A. L. P. Chen, editors, *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Lecture Notes in Artificial Intelligence 1805, pages 165–168, Kyoto, Japan, 2000. Springer.
- [140] A. K. H. Tung, J. Hou, and J. Han. Spatial Clustering in the Presence of Obstacles. In *Proceedings of the 17th International Conference on Data Engineering*, pages 359–367, Heidelberg, Germany, 2001. IEEE Computer Society.

- [141] W. Wang, J. Yang, and R. R. Muntz. STING: A Statistical Information Grid Approach to Spatial Data Mining. In M. Jarke, M. J. Carey, K. R. Dittrich, F. H. Lochovsky, P. Loucopoulos, and M. A. Jeusfeld, editors, *Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 186–195, Athens, Greece, 1997. Morgan Kaufmann.
- [142] W. Wang, J. Yang, and R. R. Muntz. STING+: An Approach to Active Spatial Data Mining. In *Proceedings of the 15th International Conference on Data Engineering*, pages 116–125, IEEE Computer Society Press, 1999. Sydney, Australia.
- [143] W. Wang, J. Yang, and P. S. Yu. Efficient mining of weighted association rules (WAR). In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 270–274, Boston, MA, 2000. AAAI Press.
- [144] R. Webster and M. A. Oliver. *Statistical Methods in Soil and Land Resource Survey*. Oxford University Press, New York, 1990.
- [145] G. J. Williams. Evolutionary Hot Spots Data Mining - An Architecture for Exploring for Interesting Discoveries. In N. Zhong and L. Zhou, editors, *Proceedings of the 3rd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Lecture Notes in Artificial Intelligence 1574, pages 184–193, Beijing, China, 1999. Springer.
- [146] J.D. Wilson. Mergers and Acquisitions Remake the Competitive Landscape. *GEO World*, November 2001.
- [147] I. H. Witten and E. Frank. *Data Mining*. Morgan Kaufmann Publishers, San Mateo, CA, 2000.
- [148] M. F. Worboys. *GIS: A Computing Perspective*. Taylor & Francis, London, 1995.
- [149] C. T. Zahn. Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters. *IEEE Transactions on Computers*, 20(1):68–86, 1971.

-
- [150] B. Zhang, M. Hsu, and U. Dayal. K-Harmonic Means - A Spatial Clustering Algorithm with Boosting. In J. F. Roddick and K. Hornsby, editors, *Proceedings of the International Workshop on Temporal, Spatial and Spatio-Temporal Data Mining*, Lecture Notes in Artificial Intelligence 2007, pages 31–45, Lyon, France, 2000. Springer.
- [151] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In H. V. Jagadish and I. S. Mumick, editors, *Proceedings of the ACM SIGMOD'96 International Conference on Management of Data*, pages 103–114, Montreal, Canada, 1996. ACM Press.