

Categorical Database Generalization in GIS

Liu Yaolin

CENTRALE LANDBOUWCATALOGUS



0000 0911 4329

Promotors:

Prof. Dr. Ir. M. Molenaar,
Professor of Spatial Data Acquisition and Geoinformatics,
International Institute for Geo-Information Science and Earth Observation
Enschede, The Netherlands and Wageningen University,
The Netherlands

Prof. Dr. M.J. Kraak,
Professor of Cartography and Visualization and Geoinformatics,
International Institute for Geo-Information Science and Earth Observation
Enschede, The Netherlands and Utrecht University,
The Netherlands

**Examining
Committee:**

Prof. Dr. Ir. A Stein, Wageningen University
Prof. Dr. F.J. Ormeling , Utrecht University
Prof. Dr. H.F.L. Ottens, Utrecht University
Prof. Dr. Ir. P.J.M. van Oosterom, Delft University of Technology

NWO 8201, 3193

Liu Yaolin

Categorical Database Generalization in GIS

Thesis

To fulfill the requirements for the degree of doctor
on the authority of the rector magnificus
of Wageningen University

Prof. Dr.Ir. L. Speelman

to be publicly defended

on Wednesday 8th May 2002

at 2.00 hours in the auditorium of International Institute for Geo-Information
Science and Earth Observation, Enschede, The Netherlands

164 8868

Doctoral Thesis (2002)

Wageningen University
The Netherlands

© 2002 Liu Yaolin
ISBN 90-5808-648-8

Dissertation number 88
International Institute for Geo-information Science and Earth Observation

The research presented in this thesis was performed at
International Institute for Geo-information Science and Earth Observation
P.O. Box, 6
7500 AA Enschede,
The Netherlands

Propositions

1. Constraints in database generalization provide steering parameters that govern or guide the transformation process of the database.
- *This thesis*
2. Constraints on geo-spatial model define the new classification hierarchy and aggregation hierarchy associated with a target categorical database, constraints on objects specify the requirements of the geometric and thematic properties of the objects in the target database and constraints on relationships maintain the spatial and semantic relations between objects and between object types. Such three types of constraints play a key role in the process of land use database generalization.
- *This thesis*
3. Transformation units are basic for analysis, processing and decision-making in the context of spatial database transformation. Each type of transformation unit identifies regions of clustered spatial objects and triggers specific aggregation operations. Thus they limit the number of spatial objects to be processed
-*This thesis*
4. The integrated and extended version of FDS elaborated in this thesis, called IEFDS, is essential for organizing spatial data, for analyzing and querying spatial relations, for detecting spatial conflicts, for creating transformation units, and for aggregating operations in the context of spatial database generalization.
-*This thesis*
5. Geo-information infrastructure not only plays a very important role in achieving "e"-government administration, but also acts as a very important means of realizing honest government affairs.
6. Learning advanced science and technology, experience and culture from foreigners is a very important and indispensable approach to speeding up the social progress and economic development, especially for developing country.
7. Man is born not to solve the problems of the universe, but to find out where the problem begins, and then to restrain himself within the limits of the comprehensible.
-*Johann Goethe*
8. The Chinese proverbs: "take measures suited to local conditions", "do what is suited to the occasion" and "guide a matter along its course of development" are suited for database generalization likewise.

To Yanfang and Xingjian

Abstract

Liu Yaolin, 2002. *Categorical Database Generalization in GIS*. PhD Dissertation. Wageningen University, The Netherlands.

Categorical databases are widely used in GIS for different kinds of application, analysis, planning, evaluation and management. Database generalization that derives different resolution databases from a single database with more detail is one of the key research problems and a hot research point in the GIS and Cartography field. This dissertation presents a framework for categorical database generalization in GIS. It includes defining conceptual aspects of current categorical database generalization transformation and constraints for generalization transformation, elaboration on supporting data structure and transformation units, development of auxiliary analysis methods, and demonstration of some application examples.

Database generalization is considered as a transformation process. Three kinds of transformation are defined based on the characteristics of categorical database and categorical database generalization. They are geo-spatial model transformation, object transformation and relation transformation. Each transformation has a certain function and deals with some aspects of database. Geo-spatial transformation is mainly used to define the content framework of a new database and decide the theme of a new database. Object transformation and relation transformation deal with transformations of thematic and geometric aspects of objects and relationship between objects from an existing database to a new database.

Database generalization (transformation) requires a data structure that strongly supports data organization, spatial analysis and decision-making in a database. The design of a data structure should take two functions into account. One provides the basis for describing and organizing spatial objects and the relationships between them, and the other is for analyzing and supporting operations on spatial objects. This thesis introduces the IEFDS, an integrated and extended version of FDS, as a data model to support automated database generalization transformation. The addition to FDS is triangles. The triangles and their classification are proposed based on constituent properties of triangles in IEFDS which plays an important role in the extended adjacent and inclusion relations and extracting the skeleton line. Some examples of spatial query operations that make use of the extended adjacent relation and semantic triangles are also provided in this thesis.

In a categorical database, similarity between object types can be described by a similarity measure. The similarity is application-dependent. In a sense, the similarity will control and guide database transformation operations. The similarity evaluation model and similarity matrix are proposed for analyzing and representing similarity between objects and object types in this study which is based on Set-theory, classification and aggregation hierarchy.

The constraints such as transformation conditions play a key role in the process of database

generalization. Constraints can be used to identify conflicting areas, guide choices of operations and trigger operations as well as govern the database generalization. The processes of generalization should be performed by a series of operations under the control of constraints. Three types of constraints, data model, object and relationships based on an object-oriented database are proposed in landuse database generalization. These constraints can be specified interactively by users and varied to reflect different objectives or purposes. These types of constraints are application-dependent. This will make the database generalization process very flexible/adaptive, and the decision-making can be based on geographic meaning and not simply on the geometry of an object.

An important element proposed in this study is the transformation unit. It is an important process unit as many generalization problems need to be solved by considering a subset of related objects as a whole, rather than treating them individually. In a sense, the transformation unit is a basic analysis, processing, decision-making unit and a trigger to aggregation operation processes and it plays an important role in database transformation. The conflicted objects and its (their) related objects are organized into a transformation unit. A transformation unit that "brings together" a subset of objects can be created by conflicts in thematic and /or geometric aspects of objects or spatial relation among objects or integrating them. The main purpose of creating a transformation unit is for the preparation of an aggregation operation. It limits the area and number of a set of related objects in an aggregation operation. The different conflict types will create different types of transformation units. For this study, four types of transformation units are considered based on the constraints discussed. Each of which has a corresponding aggregation operation.

The auxiliary analysis methods (algorithms) are needed to actually perform spatial analysis and transformations. The most fundamental tasks are to identify where to generalize, how to generalize, and when to generalize. The thesis introduces a number of auxiliary analysis methods that have been developed to solve a number of important geometric and thematic problems in database transformation. These auxiliary analysis methods include semantic similarity matrix, computing a model of similarity, detection and creation of transformation units, area object aggregation analysis and the process based on transformation units, multi-neighborhood, object cluster and creation of catchments hierarchy etc.

Such examples of the application are included in the thesis as object cluster, land use aggregation and automated organization of hierarchical catchments. The application examples demonstrate the applicability and benefits of the IEFDS and similarity evaluation model. These supporting models play a key role in organizing thematic and geometric data, spatial analysis and spatial query in database generalization. It also proved that a lot of critical geometric and thematic problems in database generalization can be solved, or can be solved in a more efficient way, with the support of an adequate data model.

Abstract

Key words: Categorical database, categorical database generalization, Formal data structure, constraints, transformation unit, classification hierarchy, aggregation hierarchy, semantic similarity, data model, Delaunay triangulation network, semantic similarity evaluation model.

SAMENVATTING

Liu Yaolin, 2002, **Thematische database generalisatie in GIS**. Doctoraal proefschrift. Wageningen Universiteit, Nederland

Thematische databases, zoals een database voor grondgebruik, worden binnen GIS wijd verbreid gebruikt voor allerlei toepassingen, analyses, planning, evaluaties en management. Database generalisatie is momenteel één van de aandachtsvelden in GIS en kartografisch onderzoek. Via database generalisatie worden vanuit een enkele database meerdere lagere resolutie databases aangemaakt. Dit proefschrift biedt een kader voor database generalisatie in GIS. Het omvat de definitie van conceptuele aspecten van database generalisatie, randvoorwaarden voor generalisatie transformaties, de uitwerking van een gegevensstructuur ter ondersteuning van de transformatie-eenheden en aanvullende analyse methoden. Via enkele voorbeelden wordt de aanpak geïllustreerd.

Database generalisatie wordt beschouwd als een transformatieproces. Op basis van de karakteristieken van de thematische database en het generalisatieproces worden drie typen transformaties gedefinieerd. Het betreft de ruimtelijke modeltransformatie, de objecttransformatie en de relatietransformatie. Elke transformatie heeft een eigen functie binnen het totale generalisatieproces en heeft betrekking op een bepaald deel van de database. De ruimtelijk modeltransformatie wordt gebruikt om het inhoudelijk kader van de nieuwe database te definiëren en geeft ook het onderwerp van de database aan. De objecttransformatie en de relatietransformatie hebben betrekking op de geometrische en thematische aspecten van objecten en hun onderlinge relaties.

Database generalisatie vereist een ondersteunende gegevensstructuur ten behoeve van de interne organisatie, specifiek ruimtelijke analyse operaties om de juiste beslissingen tijdens het generalisatieproces te kunnen nemen. Bij het ontwerp van de gegevensstructuur is met twee factoren rekening gehouden. De eerste vormt de basis voor de beschrijving en organisatie van de objecten en hun onderlinge relaties en de tweede dient voor de analyse en ondersteunende operaties op de ruimtelijke objecten. Het proefschrift introduceert in dit kader de IUFDS, een geïntegreerde uitgebreide formele data structuur als gegevensmodel ter ondersteuning van geautomatiseerde database generalisatie transformaties. De uitbreiding van de FDS is de driehoek. De driehoeken en hun eigenschappen spelen een belangrijke rol in nabijheids- in omsluitingsrelaties en de extractie van hardlijnen. Enkele voorbeelden waarin bovengenoemde eigenschappen spelen, worden behandeld.

In een thematische database kan de similariteit tussen objecttypen worden omschreven door een similariteitsmaat. Similariteit is toepassingsonafhankelijk. In zekere zin is het de similariteit die de databases operaties controleert en stuurt. Een similariteitsevaluatiemodel en een similariteitsmatrix worden in deze studie voorgesteld ter analyse en representatie van similariteit tussen objecten en objecttypen. Beiden zijn gebaseerd op de set-theorie, op classificatie en aggregatie.

De randvoorwaarden bij de transformatie spelen een sleutelrol in het proces van de database generalisatie. De randvoorwaarden kunnen gebruikt worden om conflictgebieden te identificeren, en vervolgens kunnen ze helpen bij het kiezen van de juiste operatie en deze zelfs initiëren om de generalisatie sturen. Met andere woorden het generalisatieproces wordt uitgevoerd door een aantal operaties onder controle van de randvoorwaarden. Drie typen randvoorwaarden, gerelateerd aan het gegevensmodel, het object en de onderlinge relatie gebaseerd op een object-georiënteerde database, worden geïntroduceerd. Deze randvoorwaarden kunnen interactief door de gebruiker worden gevarieerd, afhankelijk van de verschillende doelstellingen van het generalisatieproces. De drie randvoorwaarden zijn toepassingsonafhankelijk. Dit maakt het generalisatieproces flexibel en aanpasbaar aan de omstandigheden, gebaseerd op de geografische betekenis van de objecten en niet alleen op de geometrie of attribuutwaarden van een object.

Een belangrijk onderdeel dat in dit werk wordt voorgesteld is de transformatie-eenheid. Deze heeft een belangrijke functie tijdens het oplossen van generalisatieproblemen, daar de te bewerken objecten vaak als een onderdeel van een groter geheel beschouwd dienen te worden en de objecten niet op individuele basis behandeld kunnen worden. De transformatie-eenheid is de basis voor de analyse, de verwerking en beslissingen, initieert aggregatieoperaties en is als zodanig van groot belang tijdens de database generalisatie. De objecten en mogelijk gerelateerde objecten waar zich tijdens het generalisatieproces conflicten kunnen voordoen worden georganiseerd in transformatie-eenheden. De conflicten die tot de vorming van de transformatie-eenheden leiden kunnen van verschillende aard zijn. Het kunnen de attributen, de geometrie of zelfs de onderlinge ruimtelijke relaties zijn die tot een conflict leiden. Hoofddoel van de aanmaak van transformatie-eenheden is de voorbereiding tot de aggregatieoperatie. Op deze manier worden het aantal objecten en het gebied tijdens de operatie beperkt. Afhankelijk van de aard van de conflicten zullen verschillende transformatie-eenheden worden gevormd. Tijdens deze studie worden vier transformatie-eenheden beschouwd, gebaseerd op de eerder genoemde randvoorwaarden. Elk resulteert in een eigen bijbehorende aggregatieoperatie.

De aanvullende analyse methoden (algoritmen) zijn nodig om de uiteindelijke specifieke analyse operaties en transformaties uit te voeren. De meest fundamentele taken hebben betrekking op problemen als het identificeren waar de generalisatie nodig is, hoe dit te doen en wanneer te generaliseren. De in het proefschrift geïntroduceerde aanvullende methoden, helpen bij het oplossen van een aantal belangrijk geometrische en thematische problemen tijdens de database transformatie. Ze omvatten onder andere de semantische similariteitsmatrix, een model om similariteit te berekenen, het opsporen en aanmaken van de transformatie-eenheden, vlakobjecten aggregatie analyse in relatie tot de transformatie-eenheden, de buurrelaties, object clustering etc.

Voorbeelden van toepassingen die in het proefschrift behandeld worden, zijn grondgebruik aggregatie en de automatische organisatie van hiërarchische stroomgebieden. De voorbeelden uit de toepassingen laten zien wat de mogelijkheden en voordelen zijn van de IUFDS en het evolutionaire similariteitsmodel. Deze ondersteunende modellen spelen een sleutelrol tijdens de

database generalisatie. Het toont bovendien aan dat een aantal cruciale geometrische en thematische problemen onderkend in database generalisatie opgelost kunnen worden of op een meer efficiënte wijze aangepakt kunnen worden wanneer ondersteund door een geschikt gegevensmodel.

Trefwoorden: thematische database, thematische database generalisatie, formele gegevensstructuur, randvoorwaarden, transformatie-eenheden, classificatiehiërarchie, aggregatie hiërarchie, semantische similariteit, gegevensmodel, delauney triangulatie netwerk, evolutionair semantische similariteitsmodel

Acknowledgements

The research presented in this thesis could not have been completed without the help and support of many people and organizations. I am very grateful to all of them.

I wish to express my sincere gratitude to my promotor Prof. Martien Molenaar for his scientific contribution and strong support in various aspects. The contribution of his rich knowledge of geo-information science particularly in the formal data schema (FDS) and the related theory proposed by him have played an important role in this research. During the entire course of study, he was very understanding and ready to provide constructive suggestions and comments to my research. We had quite a number of stimulating scientific discussions. I am very much impressed by his scientific knowledge and his meticulous scholarship. This thesis would not be in its current shape without his encouragement and continuous support. His comments and suggestions on the structure and contents of the thesis have considerably improved its quality and readability. His wife was also friendly and hospitable to my family and me. I am very grateful to my co-promotor Prof. M.J. Kraak for his valuable discussions and comments, for his inspiration and encouragement and for his consistent friendship.

I highly appreciate the generous financial support extended to me from the Netherlands Fellowship Program through ITC for the success of this research. I am very grateful to Dr. Elisabeth Kusters and Ms. Loes Colenbrander for their support and consistent friendship.

I heartily appreciate the discussions, meetings and support from ITC colleagues and staff: Mr. Ard Blenke was always helpful in providing computer hardware and software support. Dr. Abbas Farshad, Dr. B.G.H.Gorte, Dr. T.Bouloucos, Mr. K.A. Grabmaier, Mr. G.C.Huerneman, Dr. M.Sharif, Dr. K. Tempfli, Prof. Dr. W.Kainz, Dr. R.A. De By, Mr. R.L.G Lemmens, Yuxian Sun, Mr. H.P.J. Gieszen, Prof. R. Groot, Dr.D.B.P.Shrestha. Mr. T.M. Loran, Dr. J.Turkstra, Ms. Qinglian Tian, Mr. R.J.J. Dost, Mr. P.E. Schoonackers, Mr. R.Brinkman, Ms. M.S.Kreijns, Ms. G.M.J. Allessie, Ms. A.W.S.M.Geerdink-Schukking, Mr. A.Klijnstra, Ms. Lichun Wang, Prof. A.K.Skidmore, Prof. J.A Zinck, Ms. M. Verburg, Jan De Ruitter, Mr. J.P.G Bakk, Mr. W.H. Bakker, Mr. G.Reinink, Mr. V.Retsios. The secretariat of the Department of Geoinformatics, Saskia Tempelman and Ms. M.Smit, were always glad to lend me a hand when I needed help. I am indebted to Ms.Janice Collins for editing this thesis. I am thankful to Marga Koelen and C.M. Gerritsen for their support from the library. In connection with my accommodation in ITC international hotel, I acknowledge the support of Ms. Saskia.

I am thankful to many PhD colleagues and friends for their company, help and discussions on scientific and social issues: Dr. Ale Raza, Dr. Cheng Tao, Dr. Wang Donggen, Dr. Zheng Ding, Dr. Siyka Zlatanova, Dr. Saadi Mesgari, Dr. Liu Xuehua, Dr. Yang Hong, Tang Xingming, Wu lan, Wang Chunqing, Zhang Qingming, Xiao Yinhui, Zhou Yueqin, Huang Zhengdong, Cheng

Jianquan, Zu Sicai, Liu Guangzhen, Hu Debin, Yuan Guohuam , Wu Guofeng, Zhang Jianzhong, Patrick Ogao, J.L. Campos dos Santos, Masoud Kheirkhah, Ivan , Etien Luc Koua, J. Morales, A. Duker.

The Education Section of the Chinese Embassy in the Netherlands is acknowledged here for their support and encouragement.

I thank my employer, Wuhan University, formerly Wuhan Technical University of Surveying and Mapping (WTUSM) for allowing me to take study leave and pursue this study. My sincere gratitude goes to former president Prof. Dr. Li Deren of WTUSM and vice president Prof. Liu Jinlan of Wuhan University for their encouragement and support.

I am thankful to the administration team of the School of Resource and Environment of Wuhan University, Gan Fuxing, Yang Yuli, Du Qingyun, Hu Lihong, Hou Haobo, Qian Shahua and Wo Wenda, who did a lot of my work during my absence.

I am indeed grateful to Dr. Ai Tinghua, Dr. Peng Wanlin and Dr. Wang Zesheng for deep and fruitful discussions on the research and support for developing software. I am also grateful to Cheng Xiong, Jiao Li ming and Zhang Yumei for data acquisition.

I wish to express my deep gratitude to my parents, parents-in-law, sisters, sisters in law brothers and brothers-in-law for accepting my absence during my study. They have been a constant source of patience and encouragement. Here I particularly wish to dedicate my thesis to my mother in law and render my heartiest condolences upon her death during my study. There are no words that can express my deep love and respect for her. I shall always miss her counsel and care and remember her kindness forever.

Last but not least, I could not have completed this thesis without the support, love, understanding and patience of my wife Yanfang and son XingJian during the period of my study.

Contents

CHAPTER 1 Introduction.....	1
1.1 Needs for Database Generalization.....	1
1.2 Problems Associated with Database Generalization.....	2
1.3 Brief Review of Related Research.....	6
1.4 Objectives of the Research.....	7
1.5 Scope of the Research.....	8
1.6 Methodology.....	8
1.7 Structure of This Thesis.....	8
CHAPTER 2 Basics for Categorical Database Generalization.....	11
2.1 Introduction.....	11
2.2 Abstraction of Reality.....	11
2.3 Spatial Objects, Spatial Relations and Semantic Relations.....	13
2.3.1 Spatial Object.....	13
2.3.1.1 Definition of Objects.....	14
2.3.1.2 Definition of Object Types.....	15
2.3.2 Spatial Relations.....	16
2.3.2.1 Topological Relation.....	16
2.3.2.2 Spatial Metric Relation.....	17
2.3.2.3 Distance Relation.....	18
2.3.2.4 Direction Relation.....	18
2.3.2.5 Order Relation.....	18
2.3.3 Semantic Relations.....	19
2.3.3.1 Is-a Relation.....	19
2.3.3.2 Part-of Relation.....	19
2.3.3.3 Member-of Relation.....	19
2.4 Hierarchy.....	19
2.4.1 Two Important Types of Hierarchical Structure in Categorical Database Generalization.....	19
2.4.1.1 Classification Hierarchy.....	20
2.4.1.2 Aggregation Hierarchy.....	22
2.4.1.3 Object Associations.....	24
2.5 Categorical Data.....	24
2.6 Database Generalization.....	26
2.6.1 Differences between Database and Cartography Generalization.....	26
2.6.2 Database Generalization as A Database Transformation.....	27
2.6.2.1 Thematic Transformation.....	28
2.6.2.2 Geometric Transformation.....	29
2.6.2.3 Spatial Relation Transformation.....	29
2.6.3 Process of Database Generalization.....	33

CHAPTER 3 Categorical Database Generalization Transformation.....	35
3.1 Introduction.....	35
3.2 Background of Database Transformation.....	35
3.2.1 Relations among Geo-spatial Model, Taxonomic Hierarchy,.....	
Classification Hierarchy and Aggregation Hierarchy.....	35
3.2.2 Class, Object Type and Object.....	38
3.2.3 Intension and Extension of A Class.....	39
3.2.4 Formalizing Classification Hierarchy and Aggregation Hierarchy.....	39
3.2.5 Attribute Structure of A Class (Object Type) in Classification Hierarchy.....	44
3.2.6 Attribute Structure of A Class (Composite or Component Object Type) in.....	
Aggregation Hierarchy.....	44
3.2.7 Cardinal Attribute of A Class in Hierarchies.....	46
3.2.8 Sum Attributes of A Class In Hierarchies.....	46
3.3 Process of Database Transformation.....	46
3.4 Aspects of Database Transformation.....	47
3.4.1 Geo-spatial Model Transformation.....	47
3.4.1.1 Geo-spatial Model Transformation Based on A Classification Hierarchy....	48
3.4.1.1.1 Changing the Attribute Structure of Object Types from Detail.....	
to General.....	49
3.4.1.1.2 Changing the Domain of Attribute.....	54
3.4.1.1.3 Changing the Cardinality of An Object Type.....	56
3.4.1.1.4 Changing the Sum of Object Types.....	57
3.4.1.2 Transformation Based on An Aggregation Hierarchy.....	57
3.4.2 Object Transformation.....	60
3.4.2.1 Aspects of Object Transformation.....	60
3.4.2.2 Forms of Object Transformation.....	62
3.4.2.3 Types of Object Transformation.....	62
3.4.3 Relation Transformation Among Objects.....	63
3.4.3.1 Causes of Relation Changes.....	64
3.4.3.2 Forms of Changing Spatial Relations.....	64
3.5 Transformation Operations.....	66
3.6 A Framework of Transformation.....	67
3.7 Conclusions.....	67
CHAPTER 4 Constraints in Categorical Database Generalization.....	69
4.1 Introduction.....	69
4.2 Constraints for Database Transformation.....	69
4.2.1 Constraints.....	69
4.2.2 Functions of Constraints.....	69
4.3 Classification of Constraints.....	70
4.3.1 Classes of Constraints.....	70
4.3.2 Scope of Constraint Classes.....	71
4.3.2.1 Constraints on A Geo-Spatial Model.....	71
4.3.2.2 Constraints on Objects.....	73

Contents

4.3.2.3 Constraints on Relations among Objects.....	75
4.4 Transformation Unit Based on Constraints.....	76
4.4.1 Transformation Uni.....	77
4.4.2 Characteristics of Transformation Unit.....	77
4.4.3 Types of Transformation Unit.....	78
4.4.4 Examples of Transformation Units.....	80
4.5 Representation of Constraints and Operations.....	82
4.6 Summary.....	83
CHAPTER 5 Supporting Data Structure.....	85
5.1 Introduction.....	85
5.2 Formal Data Schema and Constrained Delaunay Triangulation.....	85
5.2.1 Formal Data Schema.....	85
5.2.2 Constrained Delaunay Triangulation (CDT).....	87
5.3 Integrated and Extended Formal Data Schema (IEFDS).....	88
5.3.1 Integrated FDS and CDT.....	88
5.3.2 Extended FDS.....	91
5.3.2.1 Extending FDS Based on Geometric CDT.....	91
5.3.2.2 Extending Adjacent and Inclusion Relations in FDS Based on Semantic CDT.....	97
5.3.2.3 Extending analysis Function in FDS.....	106
5.4 Query Based on IEFDS.....	107
5.5 Summary.....	114
CHAPTER 6 Auxiliary Analysis Methods.....	115
6.1 Introduction.....	115
6.2 Creating Transformation Units.....	115
6.2.1 Creating Transformation Unit Based on Thematic Constraints.....	116
6.2.2 Creating Transformation Unit based on Geometric Constraints.....	118
6.2.3 Creating Transformation Unit Based on Spatial Relation Constraints.....	119
6.2.4 Creating Transformation Unit Based on Geometric and Spatial Relation Constraint.....	121
6.3 Hierarchic Semantic Similarity.....	122
6.3.1 Hierarchic Semantic Similarity Matrix.....	122
6.3.2 Computing Model of Similarity.....	123
6.3.3 Aggregation Rules Based on Similarity.....	127
6.4 Area Object Aggregation Operations.....	128
6.4.1 Aggregation operation Based on TU1.....	128
6.4.2 Aggregation Operation Based On TU2.....	129
6.4.3 Aggregation Operation based On TU3.....	130
6.4.4 Aggregation Operation Based on TU4.....	133
6.5 Object Clustering.....	135
6.6 Multi-Order Neighborhood.....	137
6.7 Creating hierarchical Catchments of River Network.....	138

Contents

6.7.1 Straler's, Shreve's and Horton's Classification.....	138
6.7.2 Requirements of Creating Hierarchical Catchments of River Network.....	139
6.7.3 Process of Creating Hierarchical Catchments of River Network Based..... on CDT.....	140
CHAPTER 7 Application of Methods.....	147
7.1 Introduction.....	147
7.2 Object Clustering Based on CDT.....	147
7.2.1 Constructing Triangulation Network of Objects.....	148
7.2.2 Setting Distance Threshold Value between Objects	149
7.2.3 Object Aggregation.....	150
7.2.4 Analyzing Distance Threshold Value between Objects.....	151
7.3 Landuse Database Generalization.....	154
7.3.1 Thematic Transformation.....	156
7.3.2 Aggregating Objects Based on Thematic Transformation Unit.....	157
7.3.3 Detection of Small Geometric Objects.....	158
7.3.4 Building Transforming Units Based on Conflicted Objects.....	159
7.3.5 Object Aggregation.....	160
7.4 Automated Organization of Hierarchical Catchments of River Network.....	165
7.4.1 Ordering River Network.....	165
7.4.2 Constructing Catchments for Each Order River.....	166
7.4.3 Hierarchical Catchments.....	167
7.5 Discussions.....	168
CHAPTER 8 Conclusions and Future Work.....	171
8.1 Introduction.....	171
8.2 Summary.....	171
8.3 Conclusions.....	175
8.4 Future Work.....	176
References.....	179
Appendix I	195
Appendix II.....	197
Appendix III	199
Completed PhD studies at ITC.....	205
Curriculum Vitae.....	211

List of Figures

Figure 2.1	Data Model.....	12
Figure 2.2	Three schema architecture of database.....	12
Figure 2.3	Object structured data organization.....	14
Figure 2.4	Two geometric structures for spatial object.....	14
Figure 2.5	Structure for composite objects.....	15
Figure 2.6	Object type structure of objects.....	15
Figure 2.7	Object type and super object type structure of objects.....	15
Figure 2.8	Example of 8 topological relations between regions.....	17
Figure 2.9	Example of classification hierarchy.....	20
Figure 2.10	Example of aggregation hierarchy.....	22
Figure 2.11	Relation between classification and aggregation.....	24
Figure 2.12	Process of database generalization and map generalization.....	27
Figure 2.13	Example of class transformation.....	28
Figure 2.14	Example of aggregation.....	28
Figure 2.15	Example of conceptual neighborhood of topological relation.....	30
Figure 2.16	Conceptual neighborhood of direction for an object.....	32
Figure 3.1	Relations between a taxonomic system and a classification hierarchy.....	36
Figure 3.2	Land use classification hierarchy.....	37
Figure 3.3	Example of aggregation hierarchy.....	37
Figure 3.4	Diagram representing the relations between objects, classes and attributes.....	39
Figure 3.5	View of a subset of landuse database.....	43
Figure 3.6	Example of a hierarchy structure.....	43
Figure 3.7	Example of attribute structure of class transportation.....	44
Figure 3.8	Example of attribute structure of class farmland.....	45
Figure 3.9	The procedure of transforming state of a database.....	47
Figure 3.10	Example 1 of classification transformation.....	51
Figure 3.11	Example of classification hierarchy.....	51
Figure 3.12	Example 2 of classification transformation.....	53
Figure 3.13	Example of classification hierarchy change.....	54
Figure 3.14	Example of changing the range of attribute domain.....	55
Figure 3.15	Example 1 of changing the scale type of attribute domain.....	56
Figure 3.16	Example 2 of changing the scale type of attribute domain.....	56
Figure 3.17	Example of aggregation relations of types.....	59
Figure 3.18	Example of 1:1 mapping.....	63
Figure 3.19	Example of M:1 mapping.....	63
Figure 3.20	Example of M:N mapping.....	63
Figure 3.21	Example of a proximity relation change.....	65
Figure 3.22	Example of an inclusion relation change.....	65
Figure 3.23	Example of a connectivity relation change.....	65
Figure 3.24	Example of a visual connectivity relation change.....	66
Figure 3.25	A framework of database transformation.....	68

List of Figures

Figure 4.1	Examples of a simple transformation unit.....	79
Figure 4.2	Examples of a complex transformation unit.....	79
Figure 4.3	Case1 of transformation unit and its transformation.....	80
Figure 4.4	Case 2 of transformation unit and its transformation.....	81
Figure 4.5	Case 3 of transformation unit and its transformation.....	82
Figure 4.6	Case 4 of transformation unit and its transformation.....	82
Figure 5.1	A single-theme data model.....	86
Figure 5.2	Example of DT and CDT.....	87
Figure 5.3	Data model for database generalization.....	89
Figure 5.4	Logical structure of a geo-database.....	89
Figure 5.5	Part of a constrained Delaunay triangulation of a set of objects.....	90
Figure 5.6	Part of a constrained Delaunay triangulation of a set of objects after interpolation.....	90
Figure 5.7	Relations among triangles and objects.....	93
Figure 5.8	Structure of Constrained Delaunay Triangulation.....	98
Figure 5.9	Examples of T1.....	99
Figure 5.10	Examples of T2.....	100
Figure 5.11	Examples of T3.....	100
Figure 5.12	Adjacent relationship between nodes.....	102
Figure 5.13	Adjacent relationship between node and line.....	102
Figure 5.14	Adjacent relation between lines.....	103
Figure 5.15	Adjacent relation between areas.....	104
Figure 5.16	Adjacent relation between area and line.....	104
Figure 5.17	Adjacent relation between area and node.....	104
Figure 5.18	Inclusion relation between area and node.....	105
Figure 5.19	Inclusion relation between area and line.....	105
Figure 5.20	Inclusion relation between area and area.....	105
Figure 5.21	Example of skeleton linking methods in different types of triangles.....	107
Figure 5.22	Examples of skeleton lines.....	107
Figure 6.1	Example of thematic conflict of objects.....	117
Figure 6.2	Example of a hierarchical structure.....	122
Figure 6.3	Examples of aggregations based on TU2.....	130
Figure 6.4	Examples of aggregations based on TU3.....	132
Figure 6.5	Examples of aggregations Based on TU4.....	135
Figure 6.6	Example of object clustering.....	136
Figure 6.7	Example of multi-neighborhood.....	138
Figure 6.8	Classification of river system.....	138
Figure 6.9	Example of river system.....	145
Figure 6.10	Order of the river system.....	145
Figure 6.11	Example of catchments of order 4.....	145
Figure 6.12	Example of catchments of order 3.....	145
Figure 6.13	Example of catchments of order 2.....	145
Figure 6.14	Example of catchments of order 1.....	145
Figure 7.1	A set of lakes.....	148

List of Figures

Figure 7.2	Triangulation network of the objects.....	148
Figure 7.3	Objects and Triangulation network between the objects.....	149
Figure 7.4	Parameter setup dialog.....	149
Figure 7.5	Object cluster.....	150
Figure 7.6	Boundary of clustering objects.....	150
Figure 7.7	Result of aggregation.....	151
Figure 7.8	Result of object aggregation with the threshold value of 1 mm.....	151
Figure 7.9	Result of object aggregation with the threshold value of 3 mm.....	152
Figure 7.10	Result of object aggregation with the threshold value of 8 mm.....	152
Figure 7.11	Result of object aggregation with the threshold value of 10 mm.....	153
Figure 7.12	A subset of landuse database.....	154
Figure 7.13	Land use classification hierarchy before generalization.....	157
Figure 7.14	Land use classification hierarchy after generalization.....	157
Figure 7.15	Example of conflicted objects with dash line.....	157
Figure 7.16	Result of object aggregation based on semantic similarity.....	158
Figure 7.17	Examples of small objects with black boundary.....	158
Figure 7.18	Examples of transformation units.....	159
Figure 7.19	Example of a subset of landuse database generalization.....	160
Figure 7.20	Area of each class before and after generalization.....	162
Figure 7.21	Original landuse database	163
Figure 7.22	Result of landuse database generalization.....	164
Figure 7.23	Example of ordering river system with stream orders.....	166
Figure 7.24	Example of catchments of different orders of river links.....	167
Figure 7.25	Example of hierarchical catchments.....	168

List of Tables

Table 2.1	Topological distances between relations.....	31
Table 6.1	Example of semantic similarity matrix.....	123
Table 7.1	Relations between Threshold values and Transformation units.....	153
Table 7.2	Basic data of the land use database involved.....	155
Table 7.3	Number of area objects and area for each class after generalization.....	161
Table 7.4	Data of area change for each class after generalization.....	161

Chapter 1

Introduction

One of the most prevalent trends in database application is the multi-use of an existing database. GIS data are stored in a database at a certain resolution. But it is always required for GIS to provide different resolution and different detail information in real applications. Therefore, the problem of how to derive lower resolution geo-data from a higher resolution geo database has been the core of current research.

Automated generalization has been one of the most challenging issues in the digital map environment during the last three decades. With widespread and profound application of GIS, the GIS community requires the possibility of navigating dynamically from one resolution to another in order to derive or to update smaller scale maps or lower resolution database effectively from a higher resolution (more detail) for GIS applications and spatial data analysis or mapping. The emergence of the National Spatial Data Infrastructure (NSDI) in the past years (Goodchild, 1995) has given it a new importance.

After several decades of efforts, the achievement is still far from being satisfactory (Peng 1997). Generalization functionality is still severely lacking from today's GIS and digital cartographic systems. It is apparent that the automation of the generalization process requires a very flexible methodology that is able to make decisions based on geographic meaning (and not simply the geometry of an object). Geographical meaning requires both detailed spatial analytical techniques and constraint analysis in order to select and prioritize different generalization methods, and to select among resulting candidate solutions (Ruas 1998). The methods of generalization operations based on geographic meaning are still lacking now. Since there is no scale-changing function, a separate database at fixed levels of scales (or level of details) must be built for geographic data involving multiple scales. This approach results in redundancy in data collection, reducing the flexibility of data use, and therefore, in increasing expenditure of time, money and memory usage. Further, when a database of multiple but separate levels of scales is updated, inaccuracies and inconsistencies may easily be introduced.

1.1 Needs for Database Generalization

First of all there are some needs in terms of products derived from a geographical database:

- **Deriving New Database for Spatial Analysis, Decision-making and Application**

GIS data are stored in a database at a certain resolution. But it is always required for GIS to provide different resolution and different detail information for some applications. Different applications have different requirements to a corresponding database. Suppose that we have a detailed land use database from which the contents of the database for land management and land evaluation at different levels can be derived. In order to manage land at a certain

level or evaluate land use for a certain land use at a certain level, there is the need to generalize the detailed database. The database for land evaluation of course is different from the one for land management. Database generalization functions are crucial to the development and derivation of a database at multiple levels of resolution.

- **Visualization in GIS and Web.**

Database generalization improves higher quality visualization in GIS and Web, not only for aesthetic motivations, but also because the quality of visualization can dramatically influence the understanding of geographic data based on the result of database generalization (Burrough, 1998).

- **Database Generalization Preprocessing for Map Generalization**

Simplifying information in the database is the main purpose of database generalization. Geographic database generalization can be seen as a pre-stage for a map generalization and provides the basis for selection and representation of the contents of map. Map generalization concerns mainly visualization of geo-information.

1.2 Problems Associated with Database Generalization

Database generalization can be considered as the transformation of the content of a spatial database from high resolution to a lower resolution terrain representation (Molenaar 1996). The main objective of database transformation is to derive a new database with different (coarser) spatial/thematic/temporal resolutions from existing database(s) with more detail, for a particular application. To realize the objective, several problems in the transformation must be taken into account.

- **Geo-spatial Model and Its Transformation**

The real world is complex. It is not possible (and not necessary) for a spatial model to accommodate all the aspects of the reality. A geo-spatial model is an abstraction of the real world in the perspective of a particular field of interest. It specifies object types and relationships among the object types in the context of a database. Abstraction translates phenomena of the real world into instances of databases, by focusing only on relevant aspects of these phenomena. It plays an important role in database transformation. Hence, before a database can be constructed, one has to determine what aspects of reality are relevant to the application(s). In other words, the geo-spatial model must be specified. This includes specifying types of objects, the relationships among them, and how they should be represented. The defining mechanism of this model still needs further research.

Thus, a geographic database is defined by the generic spatial data model, together with the specification of particular database content. A database is the instance of conceptual data model. The emphasis is placed on representing features of a landscape, rather than the graphic display of the features. In this aspect, we can say that database generalization is the transformation from one data model of an existing database to another data model of a generalized database based on the application purpose and requirements. This aspect needs further research in database generalization. Before transforming an existing database to a new database, a new data model associated with the new database must be defined.

For a categorical database, the conceptual data model of a database has a close relationship with the classification and aggregation hierarchy and taxonomic system in application field. The classification and aggregation hierarchies play an important role in linking the definition of spatial objects at several scale levels (Molenaar 1998, Peng 1997, Peng and Tempfli 1996, Richardson 1993 and Smaalen 1996, Tang,A.,Adams,T. and Userly 1996. Parent,C.1998, Bo Su et al 1997) and the definition of spatial object types at several scale levels. These hierarchies play an essential role in defining the conceptual data model of the categorical database.

An adequate supporting conceptual data model is needed in database generalization. It decides the main contents of the database and helps to comprehend the semantic relations among the objects in a database, which is essential for spatial analysis and the implementation of generalization operation.

- **Object Transformation and Relations Transformation among objects**

Geo-spatial model transformations deal with the preservation of the logical context of objects and degree of detail (on the object type level). Transforming the objects and relations among the objects from the existing database to a new database is the concrete content of database transformation. The transformations of objects are involved with the geometric and thematic properties of the objects. The transformations of relations include spatial and semantic relations.

In a sense, objects and relations among objects are the concrete content of a database. An object is an instance of an object type. When an existing data model is transformed into a new data model, a set of object types which are included in an existing data model will be replaced by a set of object types of a new data model. When object types are changed, their instances will be changed as well. This replacement will result in the transformation of objects and relations in the existing database. Some objects in the existing database may disappear or are merged or form new objects in the new database. The change in objects will induce change in relations among the objects. For example, two spatial adjacent objects with different attributes in an existing database will be merged to form a homogeneous object, if attributes becomes the same. The adjacent relation between the objects will disappear after they have merged.

In the transformation of objects and relations among objects, spatial analysis and semantic analysis play a key role. In a sense, the result of semantic analysis decides the objects in a database which will be aggregated and the operator (s) will be triggered to process the aggregation of the objects. However relatively few research efforts have been devoted to such issues.

- **Transformation Conditions or Constraints**

Database transformations are controlled by a set of conditions (depending on application purpose and requirements), called constraints. These constraints govern, or guide the transformation process of a database. Some definitions of constraints have been given by Robinson et al (1985), Brassel and Weibel (1988), Weibel and Dutton, G.H.(1998)and Ruas and Plazanet (1996), Ruas (1998) and Liu and Molenaar (2001). Beard (1991) identifies types of constraints as graphic constraints, structure constraints, application constraints and procedural constraints from the map generalization process point of view. Weibel (1996) classifies the constraints into graphical, topological, structure, gestalt and process constraints based on constraints governing map generalization. And Ruas (1998) presents micro constraints, meso constraints and macro constraints according to the geographic analysis method. The constraints and their classification are still lacking in database generalization.

Before transformation, the constraints must be identified and classified. The constraints of the transformation are involved in the aspects of the conceptual data model, spatial and semantic properties of objects and relations among objects.

- **Transformation Operations**

To transform a database from a high resolution to a low resolution, some operations are needed. How many basic operations are needed or are rational in the database generalization? Some researchers have discussed some operations in the map generalization and database generalization from different points of view (Weibel 1992,1995, Langram 1991, Beard 1991, McMaster 1989, Molenaar, 1996, Peng 1996, Marc 1999, Mackness 1992,1993,1994 and Mackness & Purvess 1999,Colodoven et al 1999, Edward, G 1994, Stell et al 1999). Peng (1997) presents 11 operations according to the rules of contents of selection, changing spatial and thematic resolution for objects and object types. But it is obviously seen that there is some overlap between content-selection and changing spatial and thematic resolution, which have a cause-result relationship. Mackness (1994) gave 20 operations in which only spatial conflict is considered. Ruas and Langrange (1995) put forward 9 operations based on spatial constraints.

Most spatial analysis operations in GIS work at the level of the generic data model rather than the higher semantic level of a particular content specification. These generic operations can be combined using rules and parameters to compile automatically a new geographic database from existing information. For example, a simplified soils dataset

could be compiled from a more detailed database by applying some geo-processing operations. In this example, we have combined a thematic transformation (aggregated classes) with topological and geometric transformations (generalized border).

The classification of operations still lacks standardization. Operations to the object type level and operations to the object level are quite different. Object types are at higher level than objects in the database in the sense that the former is at the decision –making level and the latter at the operational level in the process of building a new database. The operations which will be introduced in the database generalization should reflect this characteristic.

- **Supporting Data Structure**

In a large database, efficiency in storage and access to multi-scale and multiple representation data as well as complex generalization operators need to be supported by powerful data models and data structures. Such existing data models as the Delaunay triangulation network (Delaunay 1934), Quad-tree (Samet, 1990), R-tree (Guttman, 1984), and Formal Data Structure (Molenaar 1989, 1991,1995) are applied to support automated generalization. Examples are available that applied some of these data structures and algorithms to support automated generalization (e.g., the BLG-tree, GAP tree and reactive data structure (Oosterom 1989,1995, Oosterom and Schenkelaars, 1996), Delaunay triangulation network (G.L. Bundy, C.B. Jones 1995; C.B. Jones 1996 et al., Peng 1997, Bouloucos, T. Kufoniyi and Molenaar 1990). However, research and development in this area is still at an early stage and requires much more effort.

Data structure should have two functions in database generalization: support data representation and data spatial analysis. The former has benefit from many years of development in the fields of automated cartography and GIS, in contrast, analytical data structure has had little or no attention paid to it.

- **Transformation Unit**

Generally, only single geographical objects are represented in databases: one road, one building, one lake, etc. However, for the characterization of the geographical space as well as for its generalization, operations are not only performed at the level of this simple object. Some operations are performed on groups of spatially organized objects, others are performed on parts of an object.

The classification and description of geographic objects is central to the generalization process. From a pragmatic point of view, we require meaningful ways of generalizing objects while retaining their distinguishing characteristics and their interdependencies with other objects. We know that it is necessary to give priority to certain qualities and characteristics that define the object being represented. Their description is a prerequisite to this abstraction process. The idea of a filter template in image processing will be

borrowed to define a transformation unit. In the image process, 3x3 and 5x5 will get different results if these two templates are used for the same image since they have different size and different constituent elements. Even the same size filter with different constituent elements applied to the same dataset will also result in a different processing result.

The geographic transformation unit is devised to manage groups of objects. The challenge is in deciding the most appropriate level at which objects are clustered. It is important that when considering the clustering of objects, we not only consider it at the geometric level but also at the semantic and topological level. A transformation unit can also be complex collections of other transformation units. It is important to stress that the composition of these units may vary- perhaps driven by the thematic intent, or the intended resolution transition and that one element or the unit might contribute/ be part of more than one of other units

Precisely how these transformation units might be formalized or prescribed is an important part of the research and is critical to the success of applying the agent paradigm in the map generalization process. It is therefore apparent that we need to define transformation units in terms of their overall tasks and the resolution dependent nature of their activities.

Based on previous considerations, database transformation should include conceptual data model transformations, object transformations and relationship transformations. These transformations are controlled or governed by a set of constraints and implemented by a set of operations.

1.3 Brief Review of Related Research

The review of related research concentrates on two aspects. One is from the research time and contents. The other is from the forms of generalization.

- **From the Time and Research Content Point of View**

Over the last thirty years the development of digital generalization has gone through several stages. From the 1960's to the early 1970's, it focused on algorithm development which emphasized individual object simplification in geometry aiming at linear features. Quite often with the goal of compacting and cleaning the data which was being digitized. The famous Douglas' algorithm has thus been introduced under the title "algorithm for reduction of the number of points required to represent a digitized line or its caricature" (Douglas and Peucker 1973).

From the middle 1970's to the middle 1980's, the research still kept a focus on geometry. As it is impossible to design any general algorithm for simplification, several studies have been conducted, with the goal of assessing the applicability of algorithms, either by

quantifying the effects of the algorithms (McMaster 1987; Muller J.C. 1987, 1988), or by characterizing the geometry of features (Buttenfield 1986, 1987, 1989).

From the middle 1980's to the middle 1990's, the research has set out to address several aspects which are relevant to a more comprehensive solution of generalization. Various conceptual models of the generalization process were developed (Brassel and Weibel, 1988; McMaster and Shea 1989) which may help to guide research on more concrete problems. The research on generalization operators and the developments of specific algorithms form the core of activities. The field has been dominated by graphic orientation. The approaches developed have usually been limited to single object, such as the methods for simplification and smoothing of linear features in an isolated fashion (McMaster 1987) with few notable exception (Nickerson 1986), conceptual framework of generalization (Brassel and Weibel 1988; Muller J.C, 1990, 1991 and 1992; Nickerson 1991; Shea 1991; Kilpeläinen 1992, Offerman 1993), and data modeling (Muller 1991; Nyerger 1991; Mark 1991). The use of expert systems in generalization (S.F. Keller, 1995, M.Heisser et al 1995, R.B. McMaster, Hardy, 2001) is limited by developing a set of rules that would be large enough to foresee all situations that could occur.

From the middle of 1990's to the present, research has been characterized by the following aspects: more complex operators such as displacement, merging and amalgamation, as well as the interrelationship between operators (Ruas 1997, Mackanness 1995); Some authors have discussed constraints in map generalization, such as Weibel and Dutton 1998, Beat and Weibel 1999, Beard 1991, Ruas 1998, 1999 etc. In the context of map generalization, a constraint can be defined as a design specification to which the solutions to a generalization problem should adhere (Weibel and Dutton 1998, Papadias et al 1997). Several authors also discussed the role of constraints in map generalization and have tried to classify the constraints. There is need for supporting data structure in generalization (G.L. Bundy, C.B. Jones and E. Furse, 1995; C.B, Jones et al 1995; P. van, Oosterom 1995 ,1998; Liu and Molenaar 1999,2001). Utilizing agent based on methodologies in order to provide solutions in autonomous map generalization is mentioned by some authors (Sylvie Lamy and Anne Ruas 1999, Celice Duchene et al 2001, Mathieu Barrault, 2001). The geographical entities have been designed as agents. The geographical agents are described by a set of characters that constrain the generalization operation, either because they should trigger the generalization (e.g. the size of a building, when too small), or because they could be damaged by the generalization (e.g. the global shape, the positional accuracy). The aim of the agent is to satisfy as much as possible all its constraints. Model-oriented generalization and database generalization research has been paid more and more attention to. Examples include Muller 1991; Rochardson 1993; Muler et al. 1995; Weibel, 1995; Peng and Molenaar, 1996; Van Smaalen, 1996; Peng and Tempfli, 1996; Peng, 1997. This is due to the rising awareness that many processes at the earth's surface can only be monitored and managed if they are understood in their geographical context. The monitoring and management of such processes requires the information at different scale levels (Molenaar 1996). Research in database generalization has largely focused on developing solutions for specific problems, in particular, the objective and scope, the requirement and problems.

The advent of the object-oriented paradigm therefore opens up new strategies for generalization (Buttenfield 1995, Tryfona, N. 1996). These apply particularly to single datasets used for multiple products, but also for maintaining a series of related but distinct datasets (Kilpeläinen 1995, and 1997, Harrie 1998, ESRI, 2000).

- **From the Form Point of View**

Forms of generalization be categorized into two types. One is map generalization which emphasizes resolution, symbol conflicts, visual quality, readability and aesthetics, and the other is database generalization or model generalization which focuses more on the content, completeness, and accuracy of the derived data, although both database generalization and cartographic generalization reduce data complexity.

1.4 Objectives of the Research

This study mainly focuses on categorical database generalization in GIS and intends to provide a method to perform meaningful generalization procedures. The main objectives of this study are directly related to the problems discussed above.

The main objectives of this study are:

- Identify the problems associated with database generalization considering objects in their spatial context;
- Develop a conceptual framework for categorical database in the context of GIS, based on a related object-oriented approach and classification and aggregation hierarchies;
- Select and enhance a supporting data model which is used to structure, classify, encode data and identify the relevant entities, their attributes as well as the relations between them for model generalization;
- Design a strategy for categorical database generalization;
- Design algorithms for categorical database generalization;
- Demonstrate the capability of the designed methods by means of the application of the proposed method taking a subset of the database as a case study;

1.5 Scope of the Research

This study will be limited to model generalization in the context of GIS, and it will focus specifically on categorical database in GIS. This is not only a need for clear definitions of individual object, composite object, and the relationships among the features, but also a need for geo-spatial models that express the relationships among the features and that act as driver

and constraint for automatic feature extract database. The main scope of the research is listed below:

- Review of database generalization aspects;
- Design of suitable object-oriented data model and generalization operations;
- Automatic organization of hierarchical catchments area from database;
- Automatic generalization of land use database;
- Similarity evaluation model;
- Constraint system for database generalization;
- Integration FDS with constrained Delaunay triangulation network.

1.6 Methodology

This research will follow the approach Formal Data Structure to organize spatial data and use constrained Delaunay triangulation network to analyze and measure spatial data; Define a new geo-spatial model associated with the target database to decide the contents of the new database based on the application requirements and the existing geo-spatial model; Build constraints and semantic similarity as control factors in database generalization which guide the generalization processes. The divide-and-conquer approach will also be used in implementation strategy of generalization according to transformation units which limit the range of objects to be processed in a process.

1.7 Structure of this thesis

This thesis consists of eight chapters which are divided into four parts. Part one includes the introduction, elaboration of the theoretical foundations and analysis of the status and prospects of the database generalization, contributed by chapter 1,2 and 3. The second part reports the design phase of data structure and algorithm for database generalization. This part comprises chapter 4, 5 and 6. The third part focuses on the implementation and testing phase indicated by chapter 7. It demonstrates how the design in the second part came into practice and explains the operations for the database. Finally chapter 8, referring to the conclusion, summarizes the most important achievements of the thesis.

Chapter 1 discusses the need for database generalization, defines the scope of the thesis , gives a brief review of previous work with respect to the defined scope, which leads to the identification of the remaining problems and the objectives of the research.

Chapter 2 reviews the important fundamental concepts in categorical database generalization for this research and also defines the terminology used in the thesis. The review follows the conceptual framework of database generalization and summarizes such relations as metric, order and topologic relations. The semantic relation and classification and aggregation hierarchy, relation change and roles in database generalization have also been discussed.

Chapter 3 presents the contents of categorical database generalization transformation, the formalization of classification hierarchy and aggregation hierarchy, the forms and contents of geo-spatial model transformation, object transformation and relation transformation, operations of transformation and the framework of database generalization.

Chapter 4 formulates the aspects of constraints in database generalization and systematically proposes the framework of constraints and operations in database generalization. Data mode constraints, object constraints and relation constraints are constructed for the system of constraints in database generalization. and the transformation unit concept is also introduced in this chapter.

Chapter 5 analyzes formal data structure and Delaunay triangulations network, an important and powerful data structure in computational geometry, to support developing algorithms for handling the following important geometric problems. In this chapter, an object-oriented and topological data model, the IEFDS, is introduced and later enhanced for handling spatial adjacent relationships and inclusion relationships among objects disconnected from each other. Examples of some of the most common spatial query operation in automated generalization are also put forward.

Chapter 6 introduces such algorithms as creation of transformation units based on thematic and geometric aspects of objects as well as spatial relation among objects, object aggregation based on different types of transformation units, automated organization of hierarchical catchments of river systems etc. The concept of multi-neighborhood based on constrained Delaunay triangulation is proposed. The computing model of semantic similarity among object types is also presented in this chapter.

Chapter 7 demonstrates how the supporting data model, operations and algorithms etc can be applied in the context of categorical database generalization such as land use generalization.

Chapter 8 summarizes the research work and concludes the major findings of the research and makes recommendations for future research.

Chapter 2

Basics for Categorical Database Generalization

2.1 Introduction

There are two types of generalizations that have different purposes. One is database generalization which is used as a tool to produce a derived database for spatial analysis and decision-making; and the other is cartographic generalization which is traditionally used as a tool to produce maps at smaller scales. Categorical database generalization is in the context of databases generalization. This chapter reviews concepts fundamental to database generalization and deals with basics for categorical database generalization by looking into relevant fields. The main aim is to lay the theoretical foundation for defining concepts, establishing framework, analysis and implementation of categorical database generalization.

2.2 Abstraction of Reality

The real world corresponds to a set of reality that is of interest. An abstraction is a simplified description of reality. A good abstraction is that information significant to the user is emphasized, and details that are immaterial or diversionary, at least for the time being, are suppressed. Modeling is one method of abstraction. It attempts to define the real phenomena through objects and their relationships and constraints. The real world can be described only in terms of models which delineate the concepts and procedures needed to translate real world observations into data that are meaningful in GIS. Modeling is the core of an information system. The most desirable way of dealing with real world (geographical) phenomena is to model them as they exist in reality. Therefore, the closer a data model represents real-world phenomena, the more comprehensive it is. In other words, all applications are looking for a data model that is able to provide a better and more authentic perception of the real-world. Traditionally, maps are produced as an underlying concept which is 'A two dimensional graphic image which shows the location of things in relation to the earth's surface at a given time '(Keates 1989).

Peuquet (1984) defined four levels of abstraction. In figure 2.1 the last three views of the data correspond to the major steps involved in database design and implementation (Peuquet, 1984). Pilouk (1996) elaborated on the process of abstraction proposed by Peuquet by introducing the construction phase. Molenaar (1995) proposed the involvement of various disciplines while modeling reality. The core of these approaches, i.e., in a conceptual, logical and physical level design, is based on ANSI/SPAPC architecture for the abstraction of reality.

- Conceptual data models provide easy to perceive high-level concepts. At this level, components of reality are defined as an object, attributes and relationships in a more abstract form. For example, a building would be represented by an area or a point feature.

- Logical data models bridge the gap between conceptual data models and physical data models. At this level, the design of a data structure for representing objects and relationships are dealt with. The most important implementation data models are relational, network, hierarchical and object-oriented (OO) data models
- Physical data models provide low-level concepts to describe how data is stored and accessed in the computer. At the physical level, data are stored on hardware.

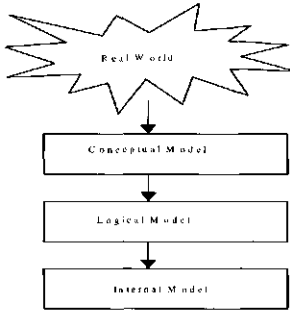


Figure 2.1 Data Model

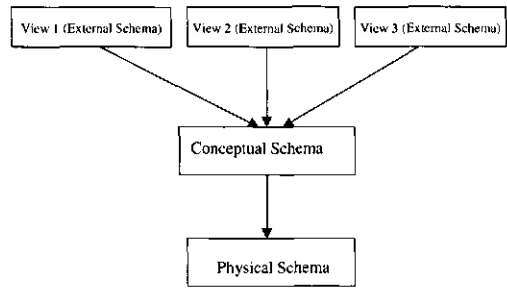


Figure 2.2 Three Schema architecture of a database

The conceptual data models, logical data models, and physical data models are relevant at different stages in database design. Figure 2.2 exhibits the main steps in database design.

There is no data modeling that can claim to be a 100% representation of reality, due to the fact there are several levels of modeling this reality. Information is lost because of abstracting from one level to another, or because of the concepts, definition and semantics used in different disciplines and in different societies for the same phenomenon or application. A database can be considered as a model of the reality and is the model in a digital form. It is assumed that one can build a database of a set of reality only if one knows how to describe it with words.

Data models provide concepts to describe the structure and contents of a database. This goal is similar to that of data types in programming languages that describe data within programs. The description of the structure and contents of a database is referred to as the database schema. The database schema is different from the data itself that populates the database. The data in a database at a particular point in time is referred to as database instance (or database state).

There are three levels used for describing the database contents of database schemas. They are the internal level, the conceptual level and the external level.

- At the conceptual level, the conceptual schema of a database system describes the logical structure of the entire set of data in a particular application environment.
- At the external level, the view schema describes how users or a group of users view those parts of the database they need for their tasks. Views can be used to restrict access to a database; they determine, read, write, insert, delete and provide protection for a database. The distinction between the conceptual schema and the external schema provides logical data independence.
- At the internal level, the physical schema (or Internal schema) describes how data are actually stored on disk. The physical level is invisible to application programs which access the database through the conceptual schema or external schema. It can be modified without changing the conceptual schema.

2.3 Spatial Objects, Spatial Relations and Semantic Relations

Spatial object, spatial relation and semantic relation are important concepts in GIS.

2.3.1 Spatial Objects

A spatial object is the representation of a real world object that contains both thematic and geometric information and is normally represented in a database by means of an “object identifier” with associated thematic and geometric data. Molenaar (1995) presents two principle structures for linking thematic and geometric data. The first structure is the field approach, which considers the earth’s surface as a spatial (-temporal) continuum. Several terrain aspects are represented in the form of attributes and the values of these attributes are considered to be position dependent. The representation of such a field in a geo-database requires that the continuum is described in the form of points or finite cells often in a regular grid or raster format. The attribute values are then evaluated for each point or cell. This structure has been represented in Figure 2.3.

The second structure of object oriented approach assumes that terrain features or objects can be defined as ones which each have a location or position and a shape and several non-geometric characteristics. These objects are represented in a database by means of an identifier to which the thematic data and the geometric data are linked, as in Figure 2.3.

The geometry of a spatial object can be described using a raster structure or vector structure (see Figure 2.4). The vector structure and the object-oriented structure approach are the ones adopted in this study.

Based on the complexity of spatial objects, they can be divided into two types of objects. One is the elementary object and the other is the composite object.

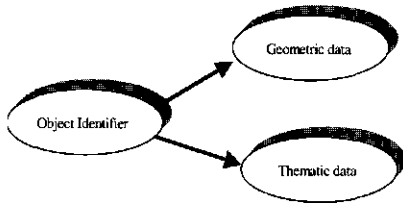


Figure 2.3 Object structured data organization (after Molenaar 1998)

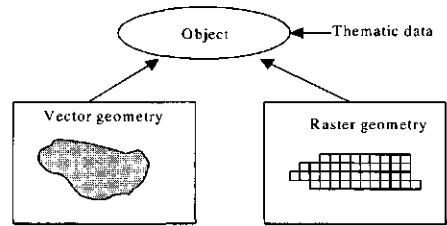


Figure 2.4 Two geometric structures for spatial objects (after Molenaar 1998)

2.3.1.1 Definition of Objects

Elementary Object

It is the most basic unit from the point of object-oriented view. The definition and the identification of elementary objects in a database depend mainly on four factors:

- Application discipline;
- User context;
- Aggregation level or scale or resolution;
- Classification level.

On each level, different elementary objects are relevant. Elementary objects at one level may be aggregates of elementary objects at another level (Molenaar, 1996, 1998).

Composite Object

A composite object is built from elementary objects that belong to different elementary object types (see Figure 2.5) (Molenaar, 1998, Husing, J. 1993) . This means that the elementary objects are the constituents of composite objects. Similar to elementary objects, composite objects at one level may be aggregates of composite objects at another level.

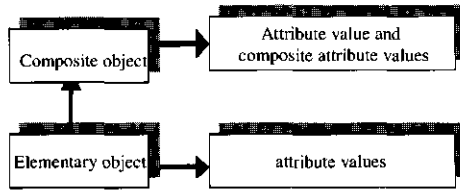


Figure 2.5 Structure for composite objects (after Molenaar 1998)

2.3.1.2 Definition of Object Types

Object Type

Object types are classes of spatial entities that have a common pattern of both state and behavior in a geo-spatial model within the framework of an application (see Figure 2.6). In reality, they may be a road, river, city, land use and so forth. For brevity, a definition for a particular entity class is called an entity type, also sometimes called a concept type (Sowa 1984). Figure 2.7 shows the object type and super object type structure of objects.

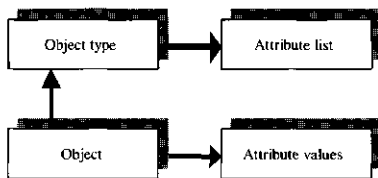


Figure 2.6 Object type structure of objects (after Molenaar 1998)

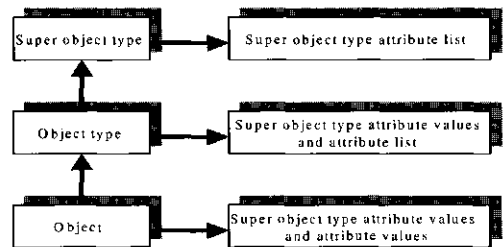


Figure 2.7 Object type and super-object type structure of objects (after Molenaar 1998)

Elementary Object Type

It is an abstraction that represents a class of similar elementary objects. The elementary object type in turn may be organized into super object types and so on. An elementary object is an instance of some elementary object type. Elementary object types together with the classification and aggregation hierarchies are important aspects in semantic data modeling and play a critical role in defining the concept of database generalization (Molenaar 1996).

Composite Object Type

It is also an abstraction that represents a group of similar composite objects. An instance of the composite-object-type is referred to as a composite-object. A composite-type can be the elementary-type of another (super) composite-type. For example, object type Farm is a combination of the types Yard and Field. In other words, Yard is part of Farm, and so is the Field.

2.3.2 Spatial Relations

The spatial relation refers to the relations between spatial objects (simple or complex). These relations can be described using a quantitative and qualitative approach. They can therefore, be measured through two major approaches based on geo-metric information and non-metric information. The variety of spatial relations can be grouped into three different categories:

- topological relations which are invariant under topological transformation of the reference objects (Egenhofer 1989, Egenhofer and Herring 1990, Egenhofer and Farnzosa, D.R. 1991, Egenhofer and Sharma, J. 1993, Egenhofer and Clementini, E. and Felice, P 1994);
- metric relations in terms of distance and directions;
- relations concerning the partial and total order of spatial objects (Kainz 1990).

2.3.2.1 Topological Relation

Topological relations are spatial relations that are preserved under such transformations as rotation, scaling and rubber sheeting. The model for binary topological relations is based on the usual concepts of point-set topology with open and closed sets (Alexandroff 1961). The definition of binary topological relations between two point sets, A and B, by the set intersections of A's interior (A°), boundary (∂A), and exterior (A^-) with the interior, boundary, and exterior of B, called the 9-intersection. The 9-intersection model is a comprehensive model for binary topological spatial relations and applies to objects of type, area, line, and point (Egenhofer and Herring 1990).

$$I(A,B) = \begin{pmatrix} \partial A \cap \partial B & \partial A \cap B^\circ & \partial A \cap B^- \\ A^\circ \cap \partial B & A^\circ \cap B^\circ & A^\circ \cap B^- \\ A^- \cap \partial B & A^- \cap B^\circ & A^- \cap B^- \end{pmatrix}$$

By considering the values empty (0) or non-empty (1), the model distinguishes 512 different topological relations between two point sets, some of which cannot be realized, depending on the dimensions of the objects and the dimensions of their embedding space. For a simple line

(1-dimensional, non-branching, without self-intersections) and region (2-dimensional, simply connected, no holes) embedded in R^2 , nineteen relations are referred to by their line-region (LR) number, which is the conversion of the first two rows in the intersection matrix from a binary number into a decimal number. The bottom row is ignored in the LR number, because it always produces three 1's for line relations in R^2 .

Figure 2.8 shows 8 topological relations and their corresponding 9-intersection matrix.

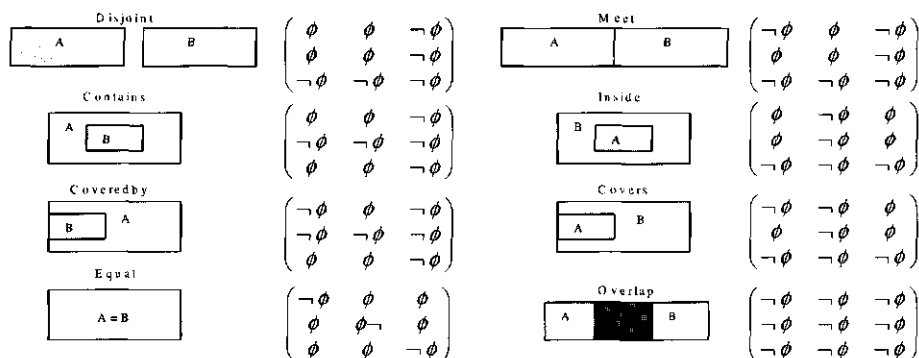


Figure 2.8 Example of 8 topological relations between regions

2.3.2.2 Spatial Metric Relation

Spatial metric relations are defined in metric space. Metric space can be defined as: let M be a non-empty set and $f: M \times M \rightarrow R$ a function, the metric on M is called a metric space (M, f) subject to the following conditions:

- $d(x, y) = 0 \Leftrightarrow x = y$; distance from x to y is zero;
- $d(x, y) \geq 0$; distance from x to y is greater or equal to zero;
- $d(x, y) \Leftrightarrow d(y, x)$; distance from x to y is equal to distance from y to x ;
- $d(x, y) + d(y, z) \geq d(x, z)$ distance from x to y plus distance from y to z is greater than or equal to the distance from x to z .

where R is the set of all real numbers. R^n is a Cartesian metric. $d(x, y)$ is called the distance function for M and M could be referred to as metric space (Moise, 1977). Basic relations regarding distance, direction and location could be addressed using metric space.

In Euclidean space, we can define the distance between two points x and y as:

$$D((x_1, x_2, \dots, x_n), (y_1, y_2, \dots, y_n)) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

2.3.2.3 Distance Relation

Distance relation reflects the distance between two objects. The distances are quantitative values determined through measurements or calculated from known coordinates of two objects in some reference system. Humans, however, frequently use approximations and qualitative notions such as near or far when reasoning about distance. Objects range from having no distance between them, zero, to being very close, to close, and then far based on increasing buffer distances. The actual definitions of these distances have been investigated elsewhere (Hong 1994; Hernandez et al, 1995, Donna, J. Peuquet, Q. and Zhan X. 1987, Frank A.U. 1996)).

Approximate distances are mapped onto quantitative distance using fuzzy sets (Dutta 1988, 1990) or mutually exclusive distance intervals or increasing ratio (Hong 1994). Reasoning with approximate distances, however, only provides meaningful results in conjunction with direction reasoning.

2.3.2.4 Direction Relation

Directions describe, qualitatively, the orientation between spatial objects (Frank A.U. 1991, 1992 and 1996). Most previous work defines directions using either object projections (Sharma 1996, Papadias D. and Egenhofen 1997) or centroids (Hernandez 1994). Each approach has its own advantages and shortcomings for a detailed discussion (Frank A.U. 1996). The set of cardinal direction relations can be expressed as following:

$$A = \{\text{NorthEast, North, North West, West, SouthWest, South, SouthEast, East}\}.$$

2.3.2.5 Order Relation

A relation on a set is an ordering relation if it is reflexive, anti-symmetric, and transitive. An example of a relation is set inclusion. A set S with a binary \leq is called a partially ordered set (or poset) (S, \leq) , if for every A, B and C in S .

- if A includes A , then A is contained in itself ($A \leq A$) (reflexivity);
- if A includes B , and B includes A , then A is equal to B ($A \leq B$ and $B \leq A \Rightarrow A=B$) (anti-symmetric);
- if B includes A and C includes B , then C includes A ($A \leq B$ and $B \leq C \Rightarrow A \leq C$) (transitive)

2.3.3 Semantic Relations

The hierarchy can be constructed by attribute structure, function, and order of objects etc in some applications. Semantic relations refer to the relations between object types (simple or complex). The variety of semantic relations can be grouped into three different categories: Is-a, Part-of and Member-of relation.

2.3.3.1 Is-a Relation

An IS-A relationship is used to establish super object type and sub object type relationships. That means one object type can be derived from another object type. The sub object type inherits the properties of the super object type. But the sub object type can define its own additional properties, which are not properties of the super object type.

2.3.3.2 Part-of Relation

A Part-of relationship is used to establish a higher order object type and lower order object type relationship. That means that a higher object type can be formed by lower object types that belong to a different classification hierarchy. The higher order object type may inherit the attribute from the lower object types.

2.3.3.3 Member-of Relation

A Member-of relationship is used to establish an object and associations of an object type relationship. That means that a given object can be part of several associations of the same object type. There are no need to be m:1 relation between a given object and an associations of a object type. They may be of m: n relations between them.

2.4 Hierarchy

Hierarchy is one of the major conceptual mechanisms to model the world. The idea is to deduce knowledge at the highest (coarsest) level of detail in order to reduce the amount of facts taken into consideration. Too much detail may not be helpful for analyzing the spatial distribution pattern of natural phenomenon or decision-making or meeting the requirements of application.

2.4.1 Two Important Types of Hierarchical Structure in Categorical Database Generalization

We looked in detail at two functions producing two different types of hierarchies: classification hierarchy and aggregation hierarchy. They play the key role in categorical database generalization. The classification hierarchy defines how classes relate to more generic super classes. The aggregation hierarchy is built by the rules which are involved with the thematic, geometric and spatial relation aspects of spatial objects .

2.4.1.1 Classification Hierarchy

Object types and super-types can be organized into a hierarchical structure called classification hierarchy (Smith and Smith, 1977; Thompson, 1989; Hughes, 1991; Molenaar, 1993). Classification hierarchy describes the relationship between object types and their generic super-object types. This hierarchical structure reflects a certain aspect of data abstraction. The terms sub object type and super-object type characterize generalization and refer to object types which are related by an is-a relation. The converse relation of super object type, the sub object type, describes a specialization of super object type. Classification hierarchy may have an arbitrary number of levels in which a sub object type has the role of a super object type for another, more specific object type.

It is important to note that super object type and sub object type are different abstraction levels for each object of their common extension. For instance, assuming Irrigated paddy field is a super-type of types Rice and Maize as shown in Figure 2.9. A model associated with a database that employs the type Irrigated paddy field as an elementary object type is usually less complex than another model that employs the types Rice and Maize as elementary object types. However, these two models have some inherent relationship due to the Is-a relationship between object types irrigated paddy field (Rice and Maize).

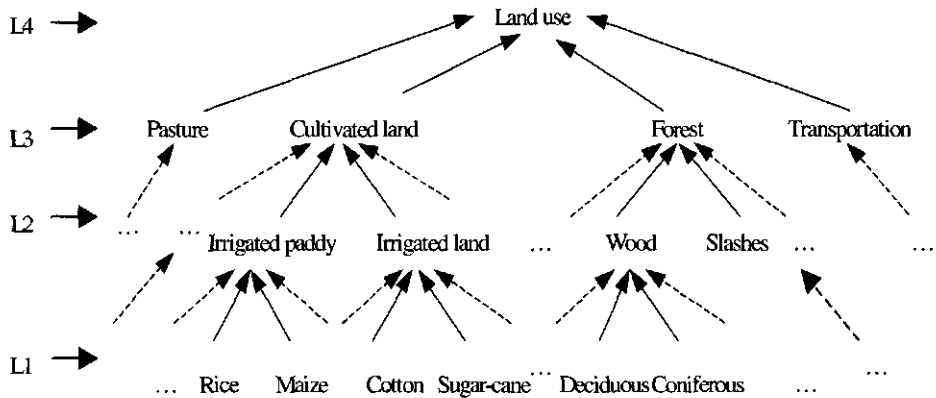


Figure 2.9 Example of classification hierarchy

Characteristics of Classification Hierarchy:

Classification hierarchy states explicitly which classes can be generalized to which generic classes. This can be visualized as a hierarchy of classes with the most generic classes at the top level. The classification hierarchy is not only very important in building a new database or reorganizing data in the existing database, but also in deriving aggregation hierarchy and constraints. It has the following characteristics which are helpful for defining the constraints, reasoning and decision-making process in database generalization:

- Classification hierarchy has a top-down character in the sense that each step down through the hierarchy gives an extension of the attribute structures for terrain objects;
- Each object type has its own attribute structure;
- The intension of object type can be expressed by a condition which specifies a subset of the set containing all the possible combinations of the values of the attribute;
- The extension of object types are collections of objects with the same attribute structure;
- A classification hierarchy as an abstraction type organizes levels of both object instances and object-type definitions;
- The object types inherit the attribute structure and the inheritance lines of attribute structure lead downward in the hierarchy;
- Object types at different levels in a classification hierarchy correspond to data of different complexity;
- Specifying an (elementary) object type implying, to a certain extent, determining the abstraction/complexity level of a geo-spatial model;
- Lower levels in the hierarchy correspond to lower abstraction levels resulting in more complex data, including both thematic and spatial aspects;
- Higher levels corresponding to higher abstraction levels and lead to less complex data;
- One object only belongs to an object type and a super object type;
- Object type and super object type are related by an Is-a relationship;
- The abstraction mechanism of classification is a prerequisite for all other abstraction mechanisms;
- Defining elementary object types and elementary objects.

Creation of Classification Hierarchy

A classification hierarchy may be a taxonomy system, such as soil classification, or land use classification, or may be derived from a sub taxonomy system according to application requirements, or may be defined according to some applications.

2.4.1.2 Aggregation Hierarchy

Another important structure is the aggregation hierarchy (Hughes, 1991; Molenaar, 1993). An aggregation model is composed of objects i.e., objects which consist of several other objects [Smith 1977]. This structure shows how composite (aggregated) objects can be built from elementary objects that belong to different classes and how these composite objects can be put together to build more complex objects and so on (Molenaar, 1998). In this article, a higher-order object type in the hierarchy is called composite-type, whereas an object type that is part of the composite-type is called component-type. Accordingly, an instance of the composite-type is referred to as a composite-object, and an instance of the component-type is regarded as a component-object or elementary object. For example, object type Farm is a combination of the object types Yard and Field. In other words, Yard is part of Farm, and so is the Field as shown in Figure 2.10. Farm is a composite object type and object type Yard and Field are elementary object types.

The fact that the simple objects can be aggregated into complex objects implies that also their attribute values may be aggregated. This means that complex objects inherit the attribute values from the objects which they are composed of. The structure also reflects some aspect of data abstraction.

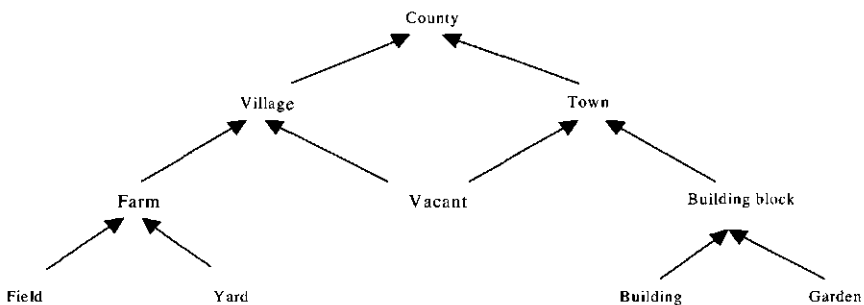


Figure 2.10 Example of aggregation hierarchy

Characteristics of Aggregation Hierarchy:

An aggregation hierarchy has a bottom-up character in the sense that the elementary objects from the lowest level are combined to compose increasingly complex objects as one ascends in the hierarchy. It has the following characteristics:

- Specifying the rules for building an aggregation hierarchy for a certain application;
- Expressing the relationship between a specific composite object and its constituent parts at different levels;
- Replacing the elementary object types in a model with their composite-type will result in transforming the model from a lower abstraction level to a higher abstraction level;
- Composite-types in the hierarchy correspond to higher abstraction levels and thus will result in less complex data;
- Component-types correspond to lower abstraction levels and hence will result in more complex data;
- The upward relationship of an aggregation hierarchy called “Part-of” links. The links relate a particular set of objects to a specific composite object and on to a specific, more complex object and so on;
- A composite-type can be the component-type of another (super) composite-type;
- In aggregation hierarchy, the inheritance lines lead upward. The compound objects inherit the attribute values from their constituent objects;
- The aggregation hierarchy has a bottom-up character: starting from the elementary objects, composite objects of increasing complexity are constructed in an upward direction.
- The composite objects inherit the attributes from their constituent parts (Molenaar and Richardson 1994).

An aggregation hierarchy has therefore a bottom-up character, in the sense that the elementary objects from the lowest level are combined to compose increasingly composite objects as one ascends in the hierarchy.

Creation of Aggregation Hierarchy

The definition of aggregation hierarchy is application-dependant and it must be established before the aggregation process. The different applications have different aggregation hierarchies, even though building these aggregation hierarchies are based on the same data set or thematic classification system. Classification hierarchies in combination with the topologic object relationship of the FDS (Molenaar, 1989) (see Chapter 5) support the definition of aggregation hierarchies of objects. An aggregation hierarchy is defined by the construction rules that describe how the objects on a given level are composed of objects at a lower level (Molenaar 1996).

- Rules specifying the classes of the elementary objects building an aggregated object of this type;

- Rules specifying the geometric and topologic relations among these objects.

For each level of the aggregation hierarchy, there should be rules for selecting the terrain objects that are aggregated to a particular composite object. In GIS these rules will be based partly on the topological relations between terrain objects (e.g., connective or adjacent) and partly on the thematic relation (e.g., the common class values).

In fact, for every type of composite object, a separate classification structure can be defined (see Figure 2.11).

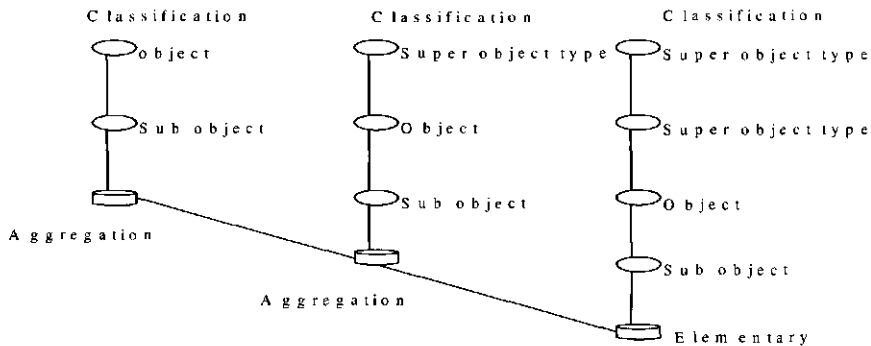


Figure 2.11 Relation between classification and aggregation (after Husing 1993)

2.4.1.3 Object Associations

Besides classification hierarchies and aggregation hierarchies, which are well defined, there is a third, less well defined type of relation between terrain objects: object associations. The association between objects is determined by Member-of relation. Unlike the hierarchies, which are characterized by many-to-one relationships (m:1), object associations represent many-to-many relationships (m : n). Associations are significant for composing geographic neighborhoods, i.e. a collection of objects.

2.5 Categorical Data

Chrisman (1982) defines a categorical coverage as “ an exhaustive partitioning of a two-dimensional space into arbitrarily shaped zones which are defined by membership in a particular category of a classification scheme”. Categories and zones should not be confused. The categories are conceptual entities conceived by the human mind on examination of the landscape. The zones become the physical, spatial manifestations of the concept” (Beard 1988, Shea et al, 1989, GrayS.V. and Egenhofer, M.J. 1993, Robin et al 1996, Frank,A.U. and Volta,G.S, 1997, Jaakkola 1998. Peter and Weibel 1999, Smith and Mark,2001).

Not only is a categorical database a collection of categorical data, it also contains relationships between categorical data elements, such as spatial and semantic relations. A typical example of a categorical data is land use database. Characteristics of Categorical Data can be summarized as following:

- Categorical data consists of two connected components:
 - (1) a spatial component and
 - (2) a thematic component. Categorical databases hold time fixed, control for theme, and measure the location, by searching for the largest areas with uniform properties;
- A categorical database is a set of spatial zones (geographic unit) constructed by a set of categories. The zones in a categorical database are an exhaustive subset of space and that not overlap (Beard 1988). Every zone (geographic unit) is evaluated according to predetermined classes resulting in regions comprised of homogenous value.

Let zones z_i cover the whole space Ω ;

$$\cup_i z_i = \Omega ;$$

$$z_i \cap z_j = \Phi .$$

- A categorical database has a theme that is based on the purpose of the examination. The theme of the categorical database is a domain of attribute values that describe the salient properties of a given space. Attribute values are often ordered hierarchically, typical for taxonomies.
- More than one new categorical hierarchy can be built from the same kind of initial categorical hierarchy in order to address different database purposes.
- Different levels of categorical database in details can be deduced from a single, more detailed categorical database. These databases will preserve data with a different amount of the original categorical detail.
- Similarities can be seen between categories and similar categories might be grouped into more general ones to reduce the categorical or spatial complexity to a level appropriate for a particular use. In an analysis, a user selects the most appropriate level of aggregation for the task at hand.
- Categorical database is a general case of multi-scale descriptions, where data at different levels of resolution are represented (Bertolotto et al 1995).

- Changing the spatial partition by aggregation of adjoining area does, in general, not lead to a changing categorical hierarchy. Changes to the partition of the attribute domain, i.e., changes in the categorical hierarchy, can be automatically propagated to the spatial partition, the reverse is not true: changes in the spatial domain do not propagate to the thematic domain.
- Each partition of the attribute domain results in a different set of zones. Therefore, the set of categories induces the zones. This reflects the basic concept of categorical data that one selects first the set of categories and then constructs the zones. In particular, any partition of the attribute leads to an induced partition of space.

For certain problems, the user can select a hierarchical level of detail and automatically get the corresponding categories and the categorical coverage.

2.6 Database Generalization

Database as models of (some portion of) a reality should have also the properties to express the real world at different levels in detail. Database Generalization can be defined as the transformation of the contents of a spatial database from higher resolution to a lower resolution terrain representation based on Molenaar (1996). The essential objectives of database generalization are:

- preserves the characteristics and integrity of geographic data while reducing the level of detail in its representation.;
- is application-dependant and driven;
- can help with both extracting appropriate information from source data and deriving new databases or data sets with less detail from the source database for analysis or applications at reduced scales.

2.6.1 Differences between Database Generalization and Map Generalization

A categorical database contains representations of categorical phenomena defined in terms of entities, whereas a cartographic database is a symbolized geographic database described in terms of graphic symbols. Database generalization and map generalization may take place in a GIS environment. For example, if we need to derive a reduced database to represent a given level of information, we can execute a selection and transformation to obtain a subset of data from the source database. But if we need to put the subset data onto a map, we have to satisfy certain map specifications, that is, the minimum spacing and minimum sizes of symbols, the balance of feature density, and so on, which directly affect the readability of the map. Some mapped features may have to be displaced, excluded, or simplified.

- Although both database generalization and map generalization reduce data complexity, the database generalization focuses more on the content, completeness, and accuracy of the derived data, while the latter deals with map space and resolution, symbol conflicts, visual quality, readability and aesthetics.
- Database generalization specifies what must be expressed to fulfill the purpose defined by the user.
- Map generalization specifies what can be expressed, taking graphic limitations into account.
- Map generalization is scale-dependent, while geographic information abstraction is not.
- Selections in database generalization are usually based on feature classifications and attributes, while map generalization is based on graphic constraints which are tied to scale.
- Database generalization transformation mainly concerns managing geographic meaning in a database, and map generalization mainly concerns structuring map presentations (Nyerges 1991)
- Geographic database generalization can be seen as a pre-stage for map generalization and to create datasets better suited to the requirements of spatial analyses (see Figure 2.12).

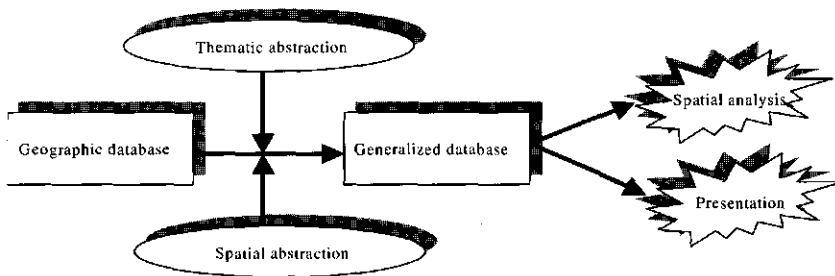


Figure 2.12 Process of database generalization and map generalization

2.6.2 Database Generalization as A Database Transformation

Simplifying information in the database is the main purpose of database generalization. This can be achieved by database transformation. Transformations can be performed with two components: one to handle attributes and thematic transformation; and the other to handle space

and spatial transformation (including geometric transformation and spatial relation transformation). Temporal transformations are not taken into account in this study.

2.6.2.1 Thematic Transformation

Changes in the thematic contents of an object are concerned with changes in one or more attribute value and changes in the attribute structure. Both kinds of change might cause the object to migrate to another object class.

- **Hierarchical Structure Transformation**

Two types of hierarchical changes could take place in database transformation. One is simplifying the original hierarchical structure through reducing the number of layers or number of object types (classification hierarchy) and the other is defining a new thematic hierarchical description of the objects (aggregation hierarchy).

Suppose that a database contains the information shown in Figure 2.13 (a). This is a detailed description of a terrain situation with agricultural fields, forest areas and natural grasslands. This description might be too detailed for structure analysis which should give information about the area covered by the different major types of land use and their spatial distribution (Molenaar 1996, Richidson 1993 and Rigaux and Scholl 1995). A less detailed spatial description can be obtained, if the original objects are aggregated to form larger spatial regions per major land use class. Figure 2.13 (b) shows that this less detailed description can be obtained in two steps.

It is certainly not always the case that object aggregation can be achieved within the framework of one class hierarchy. In many cases object aggregation will imply a completely different thematic description of the objects, so that other classes should be defined. Changes in the aggregation structure are illustrated in Figure 2.14 where farm yards and fields have been aggregated into farms and these in their turn into farm districts. The aggregation hierarchy has a bottom-up character in the sense that starting from the elementary objects composite objects of increasing complexity are constructed in an upward direction.

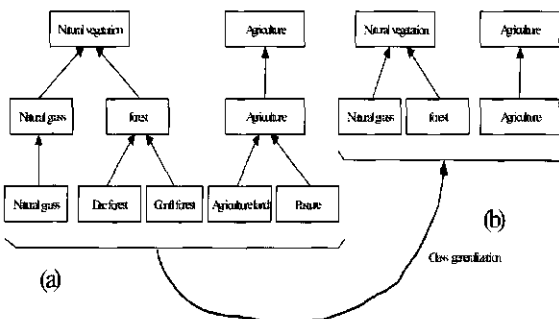


Figure 2.13 Example of class transformation

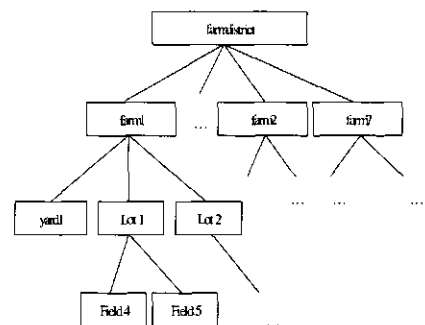


Figure 2.14 Example of aggregation

- **Attribute Structure and Value Transformation**

The changing hierarchical structure associated with a database can cause changes in the database. Changing the value range of an attribute or changing the value interval of an attribute can result in changes of the objects.

- **2.6.2.2 Geometric Transformation**

A change in the geometric structure of an object can mean a change in its position, its shape, or its size, including simplifying, merging, collapsing etc such as collapsing area to line or point etc. A change in position implies a corresponding change in the topology, distance and direction, which then has to be update in the database.

- **2.6.2.3 Spatial Relation Transformation**

When a database is transformed from one state to another state, the location and shape of objects in the database may be changed. These changes will cause changes in spatial relation (such as direction distance etc). Spatial relationships between geographic objects are time-dependent and can be changed due to space change in a database. When a database space is changed such as reduction, the two disjoint spatial objects may be viewed more closely in new database space than when they are in the original database space. This means that the spatial relationships of the objects are changed with respect to other spatial objects, while the change may not necessarily modify the topological relationship between these two overlapping or disjoint objects.

In order to describe changes between spatial relations and measure the similarity between spatial relation pairs, Egenhofer and Al-Taha (1992) introduced several concepts such as gradual change, topological distance and conceptual neighborhoods.

- Gradual change may be deformation to one of the two objects involved, such as changing the size of an object by expanding or reducing or corresponding to a scaling deformation of the object, that does not change the topology of the object.
- The topological distance is used as a measure to determine conceptual neighbor. It is the sum of the absolute values of the differences between corresponding entities of all 9-intersections. The topology distance between a relation and itself is 0, and it is between 1 and 9 for any other pair of topological relations. The shorter the topology distance between two relations, the smaller is their conceptual difference. This measure is used to identify its closest topological relationships for each of the eight region relationships.
- Two topological relations are conceptual neighborhoods if the transition from one relation to another is "smooth", so that no other relation is between the two relations when applying a gradual change. Conceptual neighborhood is used to identify those relations that are close

to each other and yield information about cognitive aspects of the relations. For each relation r_i , the relations $r_j \dots r_n$ with the shortest, non-zero topological distance are considered as r_i 's conceptual neighborhoods. Each relation is a conceptual neighbor of at least two, and at most four other relations.

Brun H.P and Egenhofer (1996) use the concept of conceptual neighborhood on the basis of gradual change to describe the similarity of two spatial relations.

• **Topological Relation Changes**

The model for binary topological relationships can be used as a framework to describe formally how to get its "closest" relationship from one topological relationship when deforming one of the two objects. The kinds of deformations considered are scaling (expansion and reduction), translation, and rotation. We are interested in gradual changes. Another assumption made is that only one kind of deformation occurs with the objects involved.

Initially two relations are slightly changed, or just one, only a bit more dramatically. The new scene is still similar, only less so, as the number and extent of the changes increases, the new scene becomes less and less similar. The change is gradual, from equivalent, to highly similar, to less and less similar.

The concept of gradual change has been used to model conceptual neighborhoods of topological relations (Egenhofer and AL-Taha 1992; Egenhofer and Mark 1995). Conceptual neighborhoods facilitate an ordering of topological relations, and support the determination of similar relations. Figure 2.15 shows the eight topological relations for simple area objects (Egenhofer and Franzosa 1991,1995).

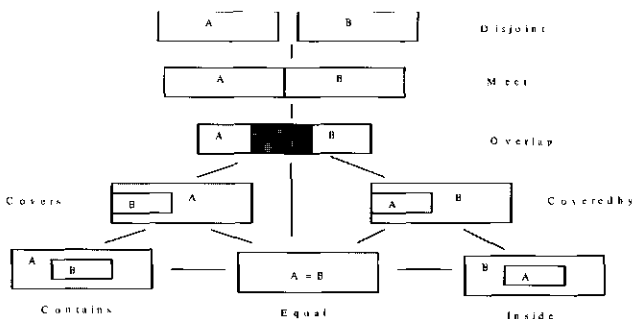


Figure 2.15 Example of conceptual neighborhood of topological relation

The figure illustrates these relations in the form of a conceptual neighborhood graph (Egenhofer and Al-Taha 1992, Hernandez 1994). Nodes in the graph denote relations that are linked

through an edge if they can be directly transformed to each other by continuous deformations (enlargement, reduction, movement). For instance, starting from relation disjoint and extending (or moving) one of the objects, we derive relation meet. With a similar extension we can get the transition from meet to overlap and so on. Disjoint and overlap are called 1st degree neighbors of meet. Depending on the allowed deformation and the relations of interest, several graphs may be obtained (e.g. Brun and Egenhofer 1996).

Topological similarity may have little meaning beyond simple, two-object scenes. Its real utility comes with a combination of equivalent concepts for distance and direction relations, and scenes with more than two objects. The measure of different topological relations can be used topological distance. Table 2.1 shows the topological distance between different relations. The results in the table are calculated based on the definition of topological distance.

Table 2.1 Topological distances between relations

Topological Distance	Disjoint	Meet	Contains	Inside	Covers	Coveredby	Equal
Disjoint	0	1	4	4	5	5	6
Meet		0	5	5	4	4	5
Contains			0	5	1	7	4
Inside				0	7	1	4
Covers					0	6	3
Coveredby						0	3
Equal							0

- **Distance Relation Changes**

For such distances, conceptual neighborhoods are derived by imposing an order relation < (less than) over the distance symbols, which corresponds to two objects gradually moving away from each other. Adjacent symbols are more similar than non adjacent symbols. For example, very close is more similar to close than to far, because zero < very close < close < far. Transitivity can be applied to this order of relations, supporting such statements as far is greater than very close. This type of reasoning supports the determination of the difference in spatial relations between two scenes, which can guide the process of determining the number and type of gradual changes required to transform one scene into another, which forms the basis of the similarity assessment presented here.

The measure of distance can be described by quality distance such as mentioned above or quantity distance, such as fuzzy distance (Papadias D et al 1999)

- **Direction Relation Changes**

It is assumed that each object can move in a continuous, smooth manner in any direction, but may not suddenly jump to a new location. (note, however, that all relations can be represented by fixing one object, and moving the other). Figure 2.16 illustrates how to produce, by gradual change, any direction relation given by any other direction relation. The lines in the figure show the links between conceptual neighbors of direction relations. The directions on the outside edge of the diagram are labeled with acronyms such as N for “north”, SSW for “South-Southwest”, etc. If two relations are not conceptual neighbors, then one cannot be derived from the other via gradual change without first producing one or more intermediate relation which is proportional to the similarity of relations.

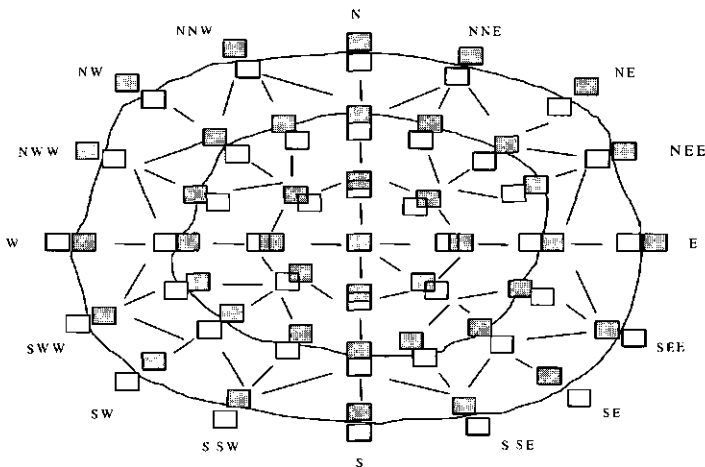


Figure 2.16 Conceptual neighborhood of directions for an object (after Bruns and Egenhofer 1996)

The structure of the figure shows the interplay between direction and topological relations. There are four rings, each representing a topological relation. The center ring represents the topological equal. The next ring shows all direction relations for the topological relation overlap. Further out are rings for meet and disjoint, in that order. Additional rings would represent associations between direction and distance relations, while preserving topology (disjoint) and the corresponding direction relations.

Cardinal directions could be quantitative values, such as azimuth or bearing, or qualitative symbols, such as north or north-east. The choice of measure depends on the application. Both azimuth and cardinal directions are used based on the specified query constraints.

These changes are often considered particularly interesting information, because they influence transformation decision-making and trigger transformation actions to be taken. When two relations have high similarity, then the objects which are involved may be merged in a database.

When the distance between two objects is too close and they have the same attribute after transformation, then two objects will merge to form a new object.

2.6.3 Process of Database Generalization

The process of database generalization can be considered as the process of abstraction from the source database to the target database with the intention of reducing detail in the contents of the database. This abstraction may include three aspects. They are data model abstraction, object abstraction and relation abstraction.

- Data model abstraction: simplifying a geo-spatial model, emphasizing significant object types and suppressing immaterial object types in the data model. Similar object types might be grouped into more general object types to reduce the complexity of the data model. For example, replacing elementary object types with composite object types or higher object types in a data model. The four types of abstraction are important and they are classification, association, generalization, and aggregation.
- Object abstraction: reducing object resolution (including thematic and geometric resolution) through aggregating objects violated by the constraints based on defined a geo-spatial model, characteristics of objects and relations among objects. For example, if two adjacent objects have high similarity in attribute, they can be merged or aggregated to form a new object in the sense that this is a kind of reducing spatial complexity to a level appropriate for particular use.
- Relation abstraction: similarity exists between spatial relation pairs. Eghenfer and Mark (1995) use conceptual neighborhoods to facilitate an ordering of topological relations and support the determination of similar relations such as meet is more similar to overlap than to contain. When the data model associated with a database is simplified and object resolution is reduced, the discernable degree of spatial relation in the database should also be reduced. The relation between two objects in a database can be recognized before transformation and the relation between the two objects may disappear after transformation in a sense that the relations between the objects are coarse or abstraction. For example two very close objects having a disjoint relation may be changed into a adjacent relation (meet) when the database is transformed.

Geo-spatial model abstraction is application-dependent. It decides the contents of a database and level in detail. In this sense, it is active abstraction. Object abstraction and relation abstraction are passive abstraction since they are controlled by Geo-spatial model abstraction.

Chapter 3

Categorical Database Generalization Transformation

3.1 Introduction

This chapter will mainly discuss the framework of database generalization transformation with emphasis on the role of the geo-spatial model, objects and relations among the objects. First, the aspects of database transformation are defined based on the concepts discussed in the previous chapter, such as data model, object, object types, constraints and operations, followed by the background of database transformation and formalization of hierarchy. Then the contents of geo-spatial model transformation, object transformation and relations transformation are elaborated in detail. Finally, transformation operations involved in the database transformation are introduced.

The database generalization can be considered as the transformation of the contents of a spatial database from a high resolution to a lower resolution terrain representation (Molenaar 1996). The main objective of database transformation is to derive a new database with different (coarser) spatial/thematic/temporal resolutions from existing more detailed database(s), for a particular application. To realize the objective, several aspects in the transformation must be taken into account. Based on the discussion in section 1.2, database transformation should include geo-spatial model transformations, object transformations and relationship transformations. These transformations are controlled or governed by a set of constraints (to be discussed in the next chapter) and implemented by a set of operations.

3.2 Background of Database Transformation

Before discussing database transformation problems, we have to be aware of some concepts and relations.

3.2.1 Relations among Geo-spatial Model, Taxonomic Hierarchy, Classification Hierarchy and Aggregation Hierarchy.

The contents of a categorical database are always closely related to a taxonomic system. i.e., soil database to a soil taxonomy system and land use database to a land use taxonomic system etc. The relations among them are described as following:

Taxonomic System and Classification Hierarchy

The taxonomic system is used in the real world to establish hierarchies of classes that permit us to understand, as fully as possible, the relationships among entities and between entities and

properties which are responsible for their character in the real world. A classification hierarchy is used in the context of a database. A classification hierarchy is expressed as an object type hierarchy that represents levels of object specificity. Furthermore, a classification hierarchy as an abstraction type organizes levels of both objects and object type definition and reflects the abstract level of objects in the database. For a categorical database which is always related to a taxonomic system in a certain application field, a classification hierarchy is derived from the taxonomic system. In this sense, we can say that the object types in the classification hierarchy correspond to the classes in the taxonomic system. The super object types and sub object types in the classification hierarchy correspond to the super classes and sub classes respectively. The objects of one object type correspond to the entities of that class. The relations between two hierarchies are shown in Figure 3.1. Figure 3.2 shows a taxonomic system for land use. This system can be easily transformed into a classification hierarchy in the database.

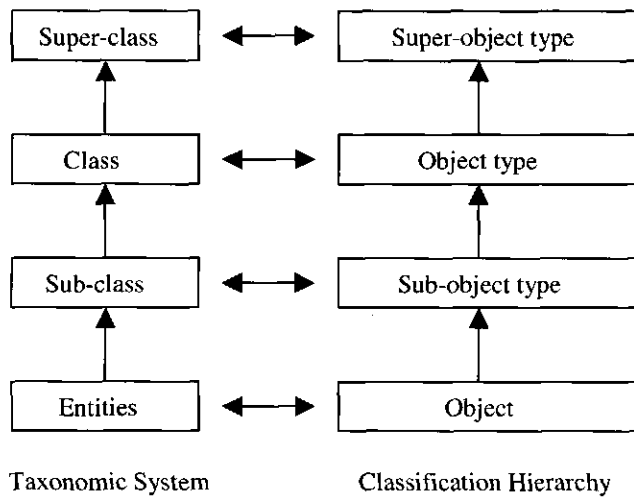


Figure 3.1 Relations between a taxonomic system and a classification hierarchy

Classification Hierarchy and Aggregation Hierarchy

An aggregation hierarchy is expressed as how a higher-order object is organized by lower-order object types that belong to a different classification hierarchy and how these higher order objects can be put together to build more complex objects and so on. A classification hierarchy in combination with the topologic object relations of formal data schema (to be discussed in chapter 5) supports the definition of aggregation hierarchies of objects. Classification hierarchy and aggregation hierarchy play an important role in linking the definition of the spatial objects at several detailed levels (Molenaar 1996, Peng 1997, Richardson 1993).

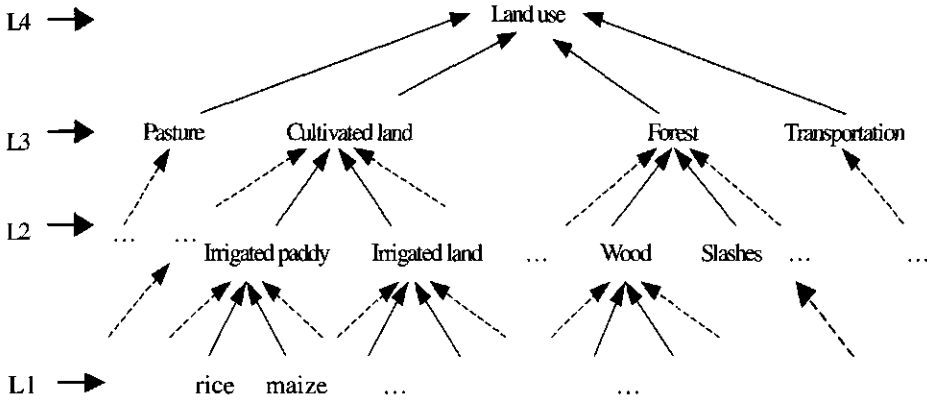


Figure 3.2 Land use classification hierarchy

Even though we can specify the relations between higher-order object types and lower order object types to build an aggregation hierarchy, the specifying relations are normally based on the classification hierarchies. In the function, the classification hierarchy will help us to find the objects we need, because it has sorted and categorized them in the categorical database. Once we have found them, the aggregation hierarchy tells us what to do to put them together meaningfully, for example, an aggregation of river and road object types into a transportation network develops a significantly different definition from the individual definitions of river classification and road classification (see Figure 3.3)

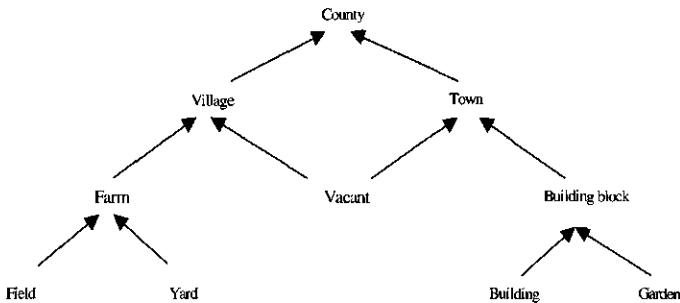


Figure 3.3 Example of aggregation hierarchy

Geo-spatial Model and Classification and Aggregation Hierarchy

Classification and aggregation hierarchy play a key role in defining geo-spatial models of categorical database since the object types in the geo-spatial model are meaningful within a certain classification and aggregation hierarchy. Before a categorical database can be built, the classification structure must be chosen (Molenaar 1998) and aggregation structure must be specified.

For the categorical database, object types, attribute structure of each object type and relationships among object types in the geo-spatial model are normally decided by the object types at the lowest level in the classification hierarchy or aggregation hierarchy. As shown in Figure 3.2, the object type rice, maize and so on at level 1 in the classification hierarchy are the object types of the geo-spatial model in the land use database.

It is important to realize that not only can an object type in a geo-spatial model be associated to a classification hierarchy, the attributes of an object type may be associated to a classification hierarchy. For example, a cadastral parcel may contain an attribute land use, which itself is associated to a land use classification hierarchy.

3.2.2 Class, Object Type and Object

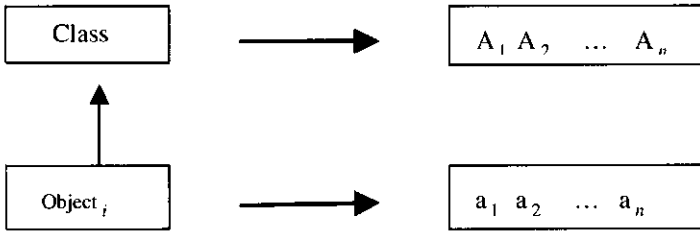
In this thesis, the class and object type have the same meaning. A class or object type determines a set of attributes to form its attribute structure. Each class c_j or object type c_j has its own attribute structure $List(c_j)$ as follows:

$$List(c_j) = \{A_1, \dots, A_i, \dots, A_n\}$$

A_i denotes one of the attributes of class c_j . Each attribute will have a name, a domain which will be specified by defining the range of the attribute values and scale type of the domain which indicates whether these values are from a nominal, an ordinal, an interval or ratio scale. An attribute A_i can be specified by a three tuple (Molenaar 1998):

$$A_i = \{NAME(A_i), SCALETYPE(A_i), DOMAIN(A_i)\}$$

For each object which belong to class c_j , the attribute structure defined by a class specifies the description structure of the object. A value is assigned to every attribute in the attribute structure of an object. For any object, its direct class is unique and lowest in a classification hierarchy since different classes have different attribute structures. These values must fall within the range of the attribute domain, which must be defined prior to the actual assignment of attribute values. The relationship between objects, object type and attributes can be seen in Figure 3.4.



E_ε **Figure 3.4** Diagram representing the relations between objects, classes and attributes
 T_I (after Molenaar 1998)
 attributes of the object) together with the list of attribute values.

3.2.3 Intension and Extension of A Class

The intension of a class c_j can be expressed as a condition for the value of a combination of attributes. The condition specifies a subset of the set containing all the possible combinations of the values of the attributes, denoted as $Int(c_j)$, while the set of all the objects that belong to c_j with the same attribute structure is commonly identified as the extension of the class c_j , denoted as $Ext(c_j)$.

3.2.4 Formalizing Classification Hierarchy and Aggregation Hierarchy

As discussed before, the object types in a geo-spatial model are normally determined by classification hierarchy and aggregation hierarchy. Changing classification hierarchy and aggregation hierarchy will result in changing the geo-spatial model associated with the database, and in turn changing the contents of the database. Changing the attribute structure and extension of classes at different levels in the classification hierarchy and aggregation hierarchy associated with a database will induce a new classification hierarchy and aggregation hierarchy and define a new data model of the database and rebuild the corresponding contents of the database.

Class hierarchy has been studied for many years in databases and knowledge bases, especially in relation to data abstraction and generalization (Smith and Smith 1977, Yee, Leung, Kwong, S.L. and He J.Z 1999, Molenaar 1996). However there is still a lack of formalization of classification hierarchy and aggregation hierarchy based on Set theory and properties of class. In the following part, the formalizations of classification hierarchy and aggregation hierarchy are discussed.

Let S be the set of objects $\{o_1, o_2, o_3, \dots, o_i\}$ in space, U be the set of classes $\{c_1, c_2, c_3, \dots, c_j\}$ for the database space. Before formalizing classification and aggregation hierarchy, the relations between classes must be identified. Seven types of relations among classes can be identified:

- Relations of equivalence among classes symbolized by \equiv ;
- Relations of inclusion among classes symbolized by \subseteq ;
- Relations of complete inclusion among classes symbolized by \subset ;
- Relations of composition among classes symbolized by \in ;
- Relations of disjunction among classes symbolized by \neq ;
- Relations of consistency among classes symbolized by ψ ;
- Relations of partial binary relation on U symbolized by $\leq c$.

Classification Hierarchy

Let $c_i, c_j \in U$ be two arbitrary spatial classes, there should be no objects that belong to the extensions of the two different classes of U . Based on the definition of classification hierarchy in the last chapter and relations among classes, we can formalize classification hierarchy with intension and extension of the classes.

- $c_i \psi c_j$, only if $\text{Ext}(c_i) \cap \text{Ext}(c_j) \neq \emptyset$;
(c_i is consistent with c_j)
- $c_i \equiv c_j$, only if $\text{Ext}(c_i) = \text{Ext}(c_j)$, denoted as $c_i = c_j$;
($\text{Ext}(c_i)$ of c_i is equal to $\text{Ext}(c_j)$ of c_j and c_i is identical to c_j).
- $c_i \subseteq c_j$, only if $\text{Ext}(c_i) \subseteq \text{Ext}(c_j)$, denoted as $c_i \leq c_j$.
($\text{Ext}(c_i)$ of c_i include $\text{Ext}(c_j)$ of c_j and c_i belongs to c_j) or
- $c_i \subset c_j$, if $\text{Ext}(c_i) \subset \text{Ext}(c_j)$, denoted as $c_i < c_j$. We also call c_i a sub-class of c_j and c_j a super-class of c_i . ($\text{Ext}(c_i)$ of c_i completely include $\text{Ext}(c_j)$ of c_j and c_i completely belongs to c_j).
- $c_i \neq c_j$, only if $\text{Ext}(c_i) \cap \text{Ext}(c_j) = \emptyset$.

(there is no common object between c_i and c_j , and c_i is different from c_j).

A class c_i is included by a class c_j if and only if the extension of c_i is subsumed by the extension of c_j . We call c_i 'IS-A' c_j .

Obviously, $\leq c$ is a partial binary relation on U , called the 'Belong to' relation, and $(U, \leq c)$ is a partially ordered set that we call a classification hierarchy.

The process of formalization of classification hierarchy depicts how the object types (classes) and super object types can be formed into a hierarchical structure. For creation of a new super object type in the classification hierarchy there will be:

- If any $A, B \in H$ and no $D \in H$ satisfying $\text{Ext}(D) = \text{Ext}(A) \cup \text{Ext}(B)$, then generate such a class D , and let $\text{Ext}(D) = \text{Ext}(A) \cup \text{Ext}(B)$, $\text{LIST}(D) = \text{LIST}(A) \cap \text{LIST}(B)$ and $D \in H$, noted as IS-A links;

The upward connections from objects to classes and classes to super classes are is-a links, which express that an object is an instantiation of a class and that a class is a special case of a more general super-class. At each level, the classes inherit the attribute structure of their super-classes at the next higher level and propagate it normally with an extension to the next lower level. At the lowest level in the hierarchy are elementary objects (Molenaar 1998).

Aggregation Hierarchy

Aggregation hierarchy expresses the relationship between a specific aggregated object and its constituent parts at different levels. This is different from classification hierarchies where classes at several generalization levels can be defined with their attribute structure and their intension and the objects can be assigned to these classes in a later stage of a mapping process.

Let S be the set of objects $\{o_1, o_2, o_3, \dots, o_i\}$ in space, U be the set of classes $\{c_1, c_2, c_3, c_i, \dots, c_j, c_l, c_m, c_n, c_l, \dots\}$ for the database space. $\text{List}(c)$ expresses attribute structure of class c and $\text{Vlist}(o)$ expresses attribute values of object o . $\text{Vlist}(o)$ expresses that object o has a list containing one value for every attribute of its class c . $\text{Geom}(o)$ expresses the geometric range of object o . We can define the formalization of aggregation hierarchy with intension and extension of the classes:

- $c_i \neq c_j$, only if $\text{Ext}(c_i) \cap \text{Ext}(c_j) = \emptyset$;
(there is no common object between c_i and c_j , and c_i is different from c_j);

- $c_i \leq d_k$ and $c_j \leq d_k$, if c_i and c_j are lower order class (object type) and d_k is higher order class (object type);
- $List(c_i) \subseteq List(d_k)$, if the attribute structure of class c_i is part of the attribute structure of class d_k ;
- $Vlist(o_i \in c_i) \subseteq Vlist(o_k \in d_k)$, if the attribute value of object o_i is part of the attribute value of object o_k ;
- $geom(o_i \in c_i) \subseteq geom(o_k \in d_k)$, if the range of object o_i is part of the range of object o_k ;
- $geom.(o_i) = \{T(o_i), E(o_i), N(o_i)\}$;
- Similar for o_j .

Object o_k is composed by object o_i and o_j if and only if the above conditions are met and denoted as :

$$o_k = Agge[o_i, o_j]$$

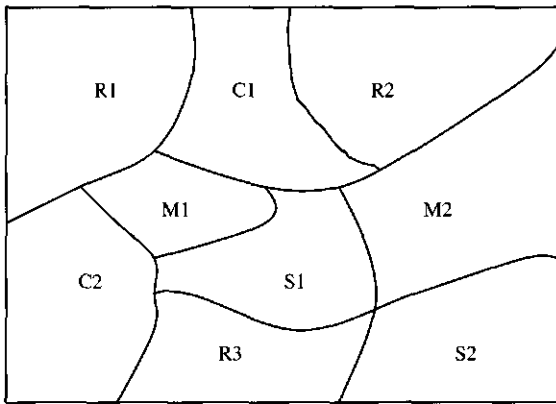
We call object o_i or o_j is 'PART-OF' object o_k .

The process of formalization of aggregation shows how a composite object (object type) can be built from elementary objects (object types) which are belong to different classification hierarchies and how these composite objects (object types) can be put together to build up more complex composite objects (object types) (Molenaar 1998) through defining the rules for selecting the objects that are to be aggregated to a particular composite object for each level of the aggregation hierarchy. These rules will be based partly on the topological relations between objects (e.g. connectivity or adjacent) and partly on the thematic relations (e.g. common class values or same attribute structure). For creation of a new composite object type in aggregation hierarchy, there will be:

- If $o_i, o_j \in c_i, o_k \in c_j$ and $o_i \in c_i$ and adjacent (o_k, o_i) and adjacent (o_k, o_j)
Then $o_l = \{o_i, o_j, o_k\}$ and $List(c_l) \supseteq \subseteq (List(c_i) \cup List(c_k) \text{ or } List(c_j) \cup List(c_k))$ and $o_l \in c$,

All classes can always be organized in a hierarchy in an order invariant form with appropriate addition or deletion of attributes and change of their attribute structure. For substantiation, we show a small land use classification hierarchy in the following example.

Figure 3.5 depicts a view of the subset of a landuse database with nine objects, and Figure 3.6 is the classification hierarchy for that subset. There, by our definition, $\text{Cultivated land} = \text{Ext}(\text{Cultivated land}) = \{R1, R2, R3, M1, M2, C1, C2, S1, S2\}$, $\text{Ext}(\text{Irrigated paddy}) = \{R1, R2, R3, M1, M2\}$, $\text{Ext}(\text{Irrigated land}) = \{C1, C2, S1, S2\}$, $\text{Ext}(\text{Rice}) = \{R1, R2, R3\}$, $\text{Ext}(\text{Maize}) = \{M1, M2\}$, $\text{Ext}(\text{Cotton}) = \{C1, C2\}$, $\text{Ext}(\text{Sugar cane}) = \{S1, S2\}$. It should be noted that on the class level the space in figure 3.5 is organized as a class hierarchy shown in figure 3.6. Corresponding to each class, there is a set of objects that constitutes the extension of the class.



Where:

- R1, R2, R3: Rice Objects;
- M1, M2 : Maize objects;
- C1, C2 : Cotton Objects;
- S1, S2 : Sugar cane objects

Figure 3.5 View of a subset of landuse database

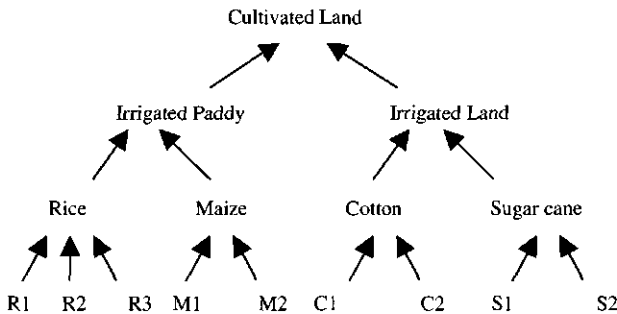


Figure 3.6 Example of a hierarchy structure

3.2.5 Attribute Structure of A Class (Object Type) in Classification Hierarchy

A class determines a set of attribute types which form the attribute structure of a class as discussed before. More and more attribute types are needed with the specialization of the classes in a classification hierarchy. The attribute list of a class must include all attribute types of its super class in the classification hierarchy, i.e. if $A, B \in U$ and $A \leq_c B$, then $LIST(A) = LIST(A) \cup LIST(B)$ or $LIST(B) = LIST(A) \cap LIST(B)$. This is apparent from the fact that for each class a more detailed specification of attributes is added compared to the less specific class. It was mentioned before that at each level the classes inherit the attribute structure of their super-class at the next higher level and propagate it normally with an extension to the next lower level.

Figure 3.7 shows the process of the attribute inheritance of attribute structure of the class at different levels. The inheritance line is top-down. 'Transportation' is a class having attributes P_1 (volume of goods transported) and P_2 (volume of passengers transported) and may have two sub-classes 'Road' and 'Railroad'. Except for P_1 and P_2 , its sub-classes may have other attributes, e.g. 'Road' may have attribute A_1 (width of a road) and A_2 (rank of road), and 'Railroad' may have attribute W_1 (speed of trains) and W_2 (rail type).

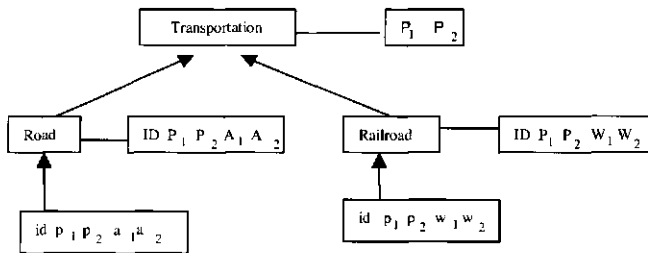


Figure 3.7 Example of attribute structure of class transformation

As discussed above, the classification hierarchy (U, \leq_c) has double semantic meanings. One is the extension enclosure, i.e. if $A, B \in U$ and $A \leq_c B$, then $Ext(A) \subseteq Ext(B)$, which expresses from bottom up the IS-A relationship between any two classes in the hierarchy. The other is attribute inheritance. The inheritance line is top down. i.e. if $A, B \in U$ and $A \leq_c B$, then $LIST(A) \supseteq LIST(B)$.

3.2.6 Attribute Structure of A Class (Composite or Component Object Type) in Aggregation Hierarchy

A composite object type is itself an individual object type, it has attribute structure of its own. For the attribute structure of classes in aggregation hierarchy, there are different ways to build it. Which way should be taken is application-dependent. The attribute structure of a composite object type may include all attribute structures of its component object types or part of its component object types or may create a new attribute structure, i.e. if A, B and $C \in U$, and $A \in B$ and $C \in B$, then $LIST(B) \supset LIST(A) \cup LIST(C)$. The elementary object types from the lowest level in aggregation hierarchy are combined to compose increasingly a complex composite object type as one ascends. If elementary objects are combined to form a composite object, their attribute values are often aggregated as well. Similar to the object in the classification hierarchy, the description structure of the object is determined by the attribute structure of a class. Values are assigned to the attributes per object. The value of the composite object is the function of corresponding values in its constituents. This is different from classification hierarchies where classes at several generalization levels can be assigned with their attribute structures and their intension, but where the objects can be assigned to these classes in a later stage of a mapping process (Molenaar 1998).

The attribute structure of a composite object type is simply the collection of attributes of all its component object types or maybe partly inherited from the attribute structure of its component object types, or partly from its components and some more new attributes may be added or specified which are completely different from the attributes of its constituents.

Figure 3.8 depicts one case of the inheritance process of the attribute value of a composite object type from its constituents in which the composite object type inherits all the attribute structure of its components. The inheritance line is upward.

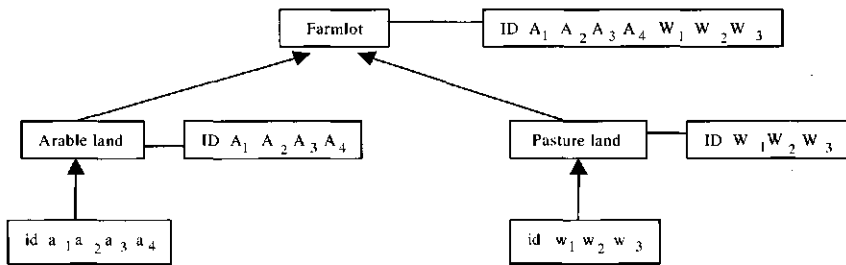


Figure 3.8 Example of attribute structure of class farmland

It is possible to define composite object types by means of their construction rules. These rules relate a particular set of object types to specific composite object types and on to a specific and more complex composite object type.

Similarly to classification, the aggregation hierarchy has double semantic meanings. One is the geometrical range of a composite object which should contain the geometrical ranges of its constituents the component objects, and the other is attribute value inheritance. The inheritance leads upward, which expresses from bottom up the PART-OF relationship between any two classes in the hierarchy. This is unlike classification hierarchies, in which the inheritance lines of attribute structures lead downward, such that the thematic description of terrain objects are more detailed the farther we go down the hierarchy.

3.2.7 Cardinal Attribute of A Class in Hierarchies

The cardinality of a set S is the number of elements that belong to that set. This number is denoted by $|s|$. Similarly the cardinality of a class is the number of objects that are members of that class, i.e., the number of objects that belong to the extension of the class. The cardinality of a class c_i is then $|\text{EXT}(c_i)|$. For the hierarchical relationship where C_i is a sub-class of C_j , the cardinality of C_j is equal to the sum of the cardinality of its subclasses.

3.2.8 Sum Attributes of A Class in Hierarchies

Some of the attributes in the attribute structure of a class may have values per object for which a sum can be computed for the class. Examples are the total of all arable farm lots, the total length of all rivers in an area. For the hierarchical relationship where C_i is a subclass of C_j , the sum of C_j is equal to the sum of its subclasses.

3.3 Process of Database Transformation

In fact, database generalization is a transformation from one existing state of a database at a certain detail level to a new state of less detail on the basis of the application and user's requirements. The state of database (SDB) can be specified by a four Tuple:

$$\text{SDB} = \{ M, O, R, C \}$$

Where:

SDB is the state of database at a certain detail level

M is the set of geo-spatial model, $M = \{m_i\}$;

O is the set of objects, $O = \{o_1, o_2, o_3, \dots, o_i\}$;

R is the set of relationships among the objects, $R = \{r \mid r \in O \times O\}$;

C is a set of conditions or constraints for transformation.

According to this idea, the transformation manipulates mainly the spatial data model associated with a database, the geometric and thematic descriptions of spatial objects and their relationships with a new set of constraints which are related to a certain application. The result of transformation is a database at less detail level than the existing one. Figure 3.9 depicts this process of the database transformation.

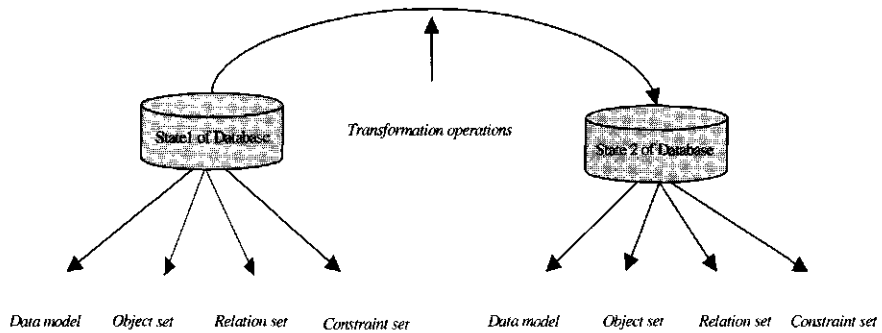


Figure 3.9 The procedure of transforming the state of a database

3.4 Aspects of Database Transformation

Based on the process of database transformation in section 3.3, database transformation is involved with three aspect transformations including:

- Geo-spatial model transformations;
- Object transformations and
- Relationship transformations.

The following part will give more detailed description for each transformation.

3.4.1 Geo-spatial Model Transformation

The geo-spatial model determines what object types and which instances of these object types should be contained in the generalized database. Changes in the thematic components of the model have a direct effect on the spatial partition. The reverse is not true: changes in the spatial domain do not propagate to the thematic domain. Main characteristics of change are given:

- Changing the geo-spatial model associated with a database will result in changing the structure and the content of the database. In the context of database generalization,

Classification hierarchy and aggregation are directly related to the content of the database.

- Changing classification hierarchy and aggregation hierarchy mean the changes in the level of definition of object types and objects when the database resolution (mapping scale) changes.
- The geo-spatial model determines the detailed level of the target database.

According to the relations between classification hierarchy and aggregation hierarchy, transformation can be divided into two types. One is geo-spatial model transformation based on classification hierarchy and the other is on aggregation hierarchy. In each transformation, five aspects of changes which will result in a database transformation are identified:

- Change of attribute structure of object types from the original hierarchy to the new hierarchy associated with a database;
- Change of domain of attribute of an object type;
- Change of type of attribute of an object type;
- Change of cardinality of an object type;
- Change of sum of an object type.

3.4.1.1 Geo-spatial Model Transformation Based on A Classification Hierarchy

Problems concerning geo-spatial model transformation based on classification hierarchy are related to the classification hierarchical structure as well as the properties of object types that a categorical database contains.

As discussed before, the geo-spatial model of an existing database is associated with a semantic classification hierarchy from which we can know how many classes exist, what relationship among classes are and what the elementary objects are in the existing database. Before implementing transformation, the object types which will be contained in the new data model associated with a new database must be built. This could be done by changing the attribute structure of object types in the classification hierarchy.

The characteristics of this hierarchy structure are summarized as following:

- All classes in a hierarchy are distinct because they all have their own unique attribute structure. An object not only inherits the attribute structure of its own class but also from the super class. i.e., attribute value list of the object contains the values of the attributes specified at a class level and a super class level;

- IS-A relation shows that when two or more classes have attributes in common, then a super class can be defined with an attribute structure LIST containing these common attributes as “super class-attributes” (Molenaar 1991 and 1993);
- Specifying an (elementary) object type implies, to a certain extent, determining the abstraction/complexity level of a database.

The classification hierarchy states explicitly which classes are generalized to which generic classes. This can be visualized as a hierarchy of classes with the most generic classes at the top level. Classification hierarchy transformation can be done based on IS-A relationships between object classes and IS-A relationships referring to class generalization result in the combination of several classes into a more general super class. It makes it possible to transform the more complex model to the less complex one.

The following section provides a detailed description of the changes related to the data model in the context of classification hierarchy associated with a database. These changes are considered to be a set, within the framework of database generalization defined in this thesis, and according to the given definitions of object types or class.

3.4.1.1.1 Changing the Attribute Structure of Object Types from Detail to General

It means that changing the attribute structure of an object type from detail to general in the classification hierarchy will result in a database transformation. It also induces the reduction of the number of object types in the existing data model. Changing the attribute structure of object types of an existing geo-spatial model to the ones at the next higher level in the same hierarchy would mean replacing the attribute structure of object types with the corresponding ones of super object types at the next higher level and also imply transforming the geo-spatial model of the database from a lower abstraction level to a higher abstraction level. Such a transformation will lead to a generalization process taking place, in order to convert instances of the sub-types to instances of the super-types. There are two types of changing attribute structure of object types in categorical database generalization:

Method 1: Thematic Generalization for All Classes-----Changing the Attribute Structure of All Object Types in the Existing Geo-spatial Model :

This will result in replacing all object types at the lowest level of the existing geo-spatial model with their corresponding super object types in the same existing data model and form a new data model associated with a categorical database. If all object types in the existing geo-spatial model have the same detail level before change, then all new object types in the new categorical database will have the same detail level after change. The new categorical database has a higher abstract level than the original categorical database. Establishing a new geo-spatial model needs two steps:

1) Selecting All Object Types Next Higher to the Object Type in A Classification Hierarchy

Let N_o to a set of object types in existing geo-spatial model which correspond to object types classification hierarchy ($Tree_o$) associated with the categorical database and E_o be a set of relations among object types N_o , and let N_n be all object types which correspond to the object types in a new classification hierarchy ($Tree_n$) associated with the target database and let E_n be a set of relations among object types N_n :

$$Tree_o = \{N_o, E_o\};$$

$$Tree_n = \{N_n, E_n\};$$

$$N_o = \{t_{o1}, t_{o2}, \dots, t_{om}\};$$

$$N_n = \{t_{n1}, t_{n2}, \dots, t_{nm}\};$$

If $N_n \subset N_o$, and $E_n \subset E_o$

then

$$Tree_n \subset Tree_o$$

If $N_n \subset N_o$, and if $\forall t_{oi} \in$ all object types at the lowest level in $Tree_o$ and $\forall t_{ni} \in$ all object types at the lowest level in $Tree_n$ then there does not exist an object type t_i in U , such that $t_{oi} < t_i$ and $t_i < t_{ni}$.

This means that all object types next higher to the lowest object type in the classification hierarchy associated with an existing database will be selected as a new geo-spatial model. Figure 3.10 shows this process. The part in the left of Figure 3.10 which is included by dash line will be deleted in a new geo-spatial model.

The different applications normally need different geo-spatial models in order to meet their requirements and purpose. For the land management case, the different levels of management such as local, regional and national need corresponding levels of database in detail for its efficient management.

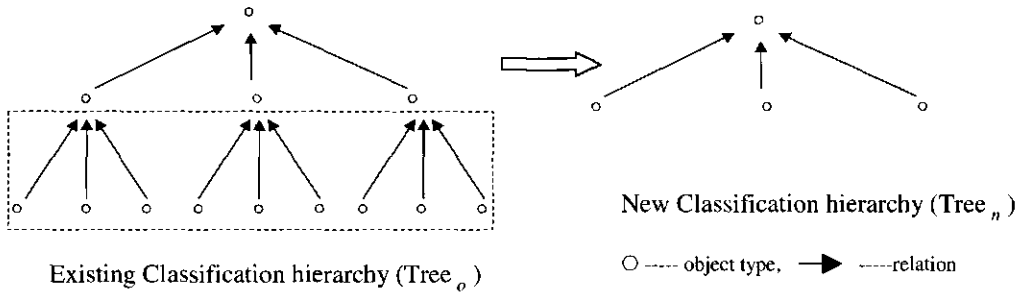


Figure 3.10 Example 1 of classification transformation

Figure 3.2 shows a land use classification hierarchy associated with a land use database and Figure 3.11 is its subset. Comparing these two figures, we found that the levels in figure 3.2 has one more layer than those in Figure 3.11. This means that the database with land use classification hierarchy in figure 3.11 is less detailed than the database with the land use classification hierarchy in Figure 3.2.

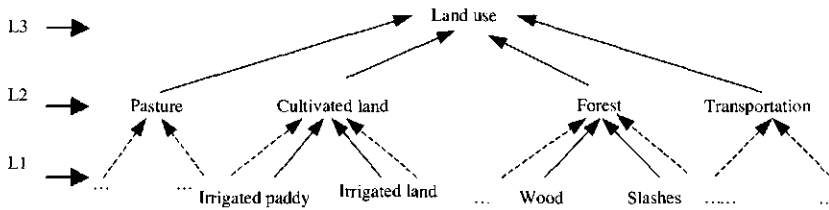


Figure 3.11 Example of Classification hierarchy

For example, assuming irrigated paddy field is a super-type of object types Rice and Maize as shown in Figure 3.2, a new data model that employs the object type Irrigated paddy field is usually less complex than another model that employs the object types Rice and Maize. However, these two models have some inherent relationship due to the IS-A relationship between object types Irrigated paddy field and Rice (and Maize).

2) Constructing Attribute Structure per Object Type in A New Geo-spatial Model :

Let A and B be object types of the existing model associated with a classification hierarchy. They have the different attribute structure LIST(A) and LIST(B) respectively and C to be the super-object type in the same hierarchy which will be the object type in new data model

with attribute structure LIST(C). The attribute structure LIST(C) of the super-object type C can get through the intersection of the attribute structure LIST(A) of the object type A and LIST(B) of the object type B. It can be expressed as:

$$\text{LIST}(A) = \{A_1, A_2, \dots, A_n\}$$

$$\text{LIST}(B) = \{B_1, B_2, \dots, B_n\}$$

$$\text{LIST}(C) = \{C_1, C_2, \dots, C_n\}$$

If $A, B, C \in U$ and $A \subseteq C$ and $B \subseteq C$,

$$\text{LIST}(C) = \text{LIST}(A) \cap \text{LIST}(B) = \{A_i \mid A_i \in \text{LIST}(A) \cap \in \text{LIST}(B)\}$$

and

$$\text{Ext}(C) = \text{Ext}(A) \cup \text{Ext}(B).$$

This means that the super-object type carries the common attributes of its sub object types, and also means that an object not only inherits the attributes of its sub object type, but also those from the super object types.

Method 2: Thematic Generalization of Selected Classes----Changing the Part of Attribute Structure of Object Types in the Existing Geo-spatial Model:

In some applications, some object types (classes) in a classification hierarchy associated with a categorical database will be emphasized, while the other object types will be suppressed. This will result in replacing part of the object types of an existing data model with their corresponding super object types and keeping the rest invariant in the same existing geo-spatial model and both parts will form a new geo-spatial model of the database. This means that converting part of object types of the existing geo-spatial model to their super object types which are application-relevant, while keeping the rest of object types invariant. So the new database may have different detail levels for different object types. This reflects that the objects and object types related to the application in the generalized database will be enhanced, while the objects and object types unrelated or less related to the application will be attenuated. The steps are similar to method 1.

1) Selecting Part of Super Object Types and Object Types from A Classification Hierarchy as a New Geo-spatial Model

Let N_o be a set of object types in an existing geo-spatial model which correspond to object types of a classification hierarchy (Tree_o) associated with the categorical database and E_o to be a set of relations among object types N_o , and let N_n be all object types which correspond to object types of a new classification hierarchy (Tree_n) associated with the

target database and E_n to be a set of relations among object types N_n :

$$Tree_o = \{N_o, E_o\};$$

$$Tree_n = \{N_n, E_n\};$$

$$N_o = \{t_{o1}, t_{o2}, \dots, t_{om}\};$$

$$N_n = \{t_{h1}, t_{h2}, \dots, t_{hn}\};$$

If $N_n \subseteq N_o$, and $E_n \subseteq E_o$

then

$$Tree_n \subseteq Tree_o$$

If $N_n \subseteq N_o$, and if $\exists t_{oi} \in$ all object types at the lowest level in $Tree_o$ and $\exists t_{hi} \in$ all object types at the lowest level in $Tree_n$, then there does not exist an object types t_i in U , such that $t_{oi} < t_i$ and $t_i < t_{hi}$.

This means that part of the lowest object types and object types at next higher to lowest object type in the classification hierarchy associated with the existing geo-spatial model will be selected as the ones in a new classification hierarchy. Figure 3.12 shows this process.

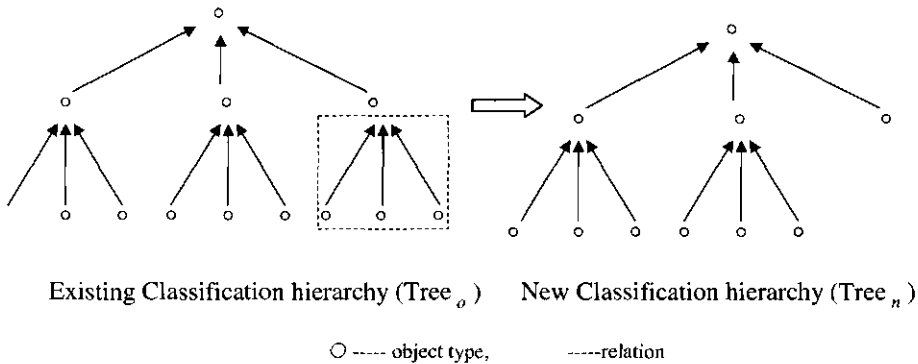


Figure 3.12 Example 2 of classification transformation

For a land evaluation case, the types of land use which need to be evaluated will be the most important elements in the database. Its instances will be the most important elementary objects. These objects do not need to be generalized in semantic except

geometric conflict of objects such as too small, too narrow etc. The other types of land use in the database will need more to be generalized in semantic and geometric properties of objects. Figure 3.13 which is derived from Figure 3.2 illustrates that the evaluation object type rice is not abstracted in semantic aspect, whereas the other classes are more or less generalized compared with Figure 3.2.

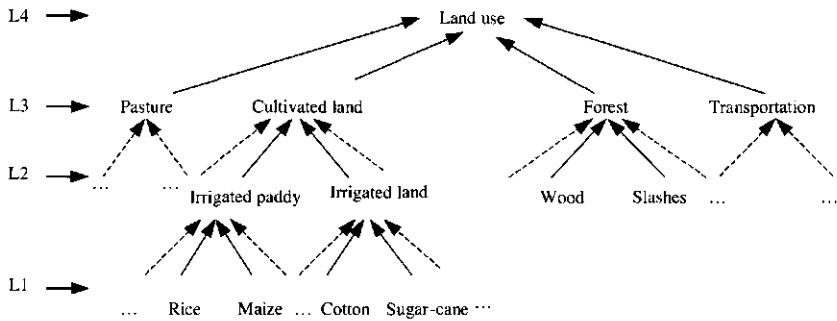


Figure 3.13 Example of classification hierarchy change

2) Constructing Attribute Structure Per Object Type in A New Geo-spatial Model:

The attribute structure of the lowest object types in the new model will be the same as ones in the existing model if they remain as the lowest object types in the new model, and if not, the method for constructing attribute structure of object types is the same as method 1.

3.4.1.1.2 Changing the Domain of Attribute

Each attribute has a name, a range of domain and the scale type of the domain as defined before. In addition to object types resulting in database transformation, changing the range, and scale type of domain of an attribute will also induce the transformation of a database and reduce the contents of the database.

- **Changing the Range of Attribute Domain**

Changes in the range of the domain of an attribute will propagate to the spatial domain and induce spatial repartition and result in a database transformation.

Supposing that we have a land evaluation database in which each evaluation unit has an evaluated value and an evaluated grade. Each grade has a range of the value within 100 as

following:

High suitability	(100 ~ 80)
Suitability	(79 ~ 60)
Marginal suitability	(59 ~ 30)
No suitability	(29 ~ 0)

If we change the above values of the domain of each attribute as following:

High suitability	(100 ~ 65)
Suitability	(64 ~ 45)
Marginal suitability	(44 ~ 20)
No suitability	(19 ~ 0)

then the contents of the database will be changed as well. This change of the domain of attribute will cause changes to the extension of each grade and the corresponding spatial distribution in the sense that the database transformation has taken place (see Figure 3.14).

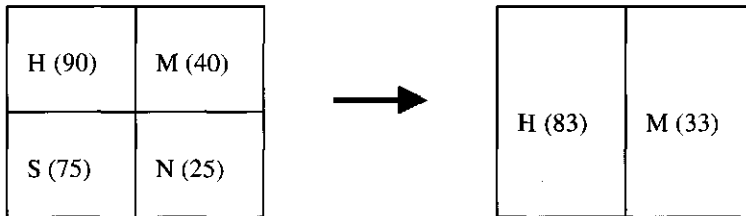


Figure 3.14 An example of changing the range of attribute domain

• **Changing the Scale Types of Attribute Domain**

Similar to changing the range of the domain of an attribute, changing scale type of the domain of an attribute will also result in database transformation. This transformation may or may not reduce the detail level of the original database.

Figure 3.15 shows the case of scale type of the domain from nominal to an interval. Changing the scale type of the domain results in changes in geometric and thematic properties of the objects in the database.

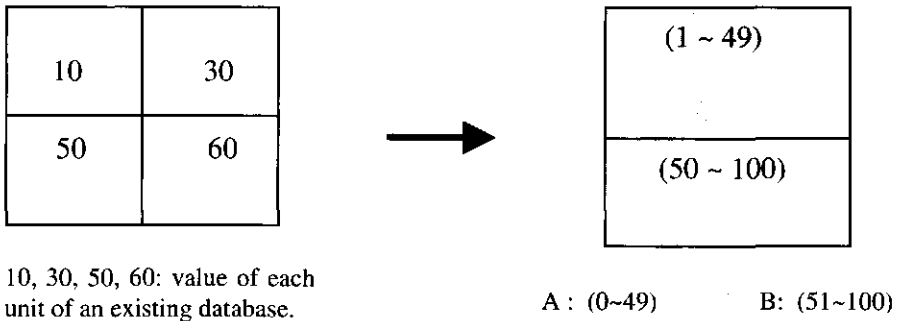


Figure 3.15 Example 1 of changing the scale type of attribute domain

Figure 3.16 shows an example of changing the scale type from nominal to ordinal. Changing the scale type of the domain of an attribute also induces changes in geometric and thematic properties of objects in the database.

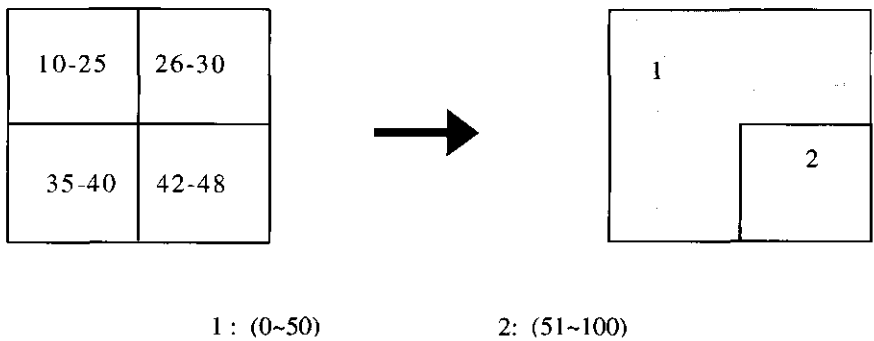


Figure 3.16 Example 2 of changing the scale type of attribute domain

3.4.1.1.3 Changing the Cardinality of An Object Type

The cardinality of an object type represents the number of objects which belong to an object type. Changing the cardinality of an object type will result in changing the number of the object type in the existing database. Reducing the number of objects of one object type in the database will induce database transformation to take place. In a sense, this is a statistic generalization. Before generalization, we should decide the number of objects for each object type to be kept in the new database.

How many objects which should appear in the generalized database are dependent on the given particular application. Richardson (1993) proposes a method to decide how many objects for each object type should be kept for a generalized database based on the Necessity Factor and the Reduction Factor.

3.4.1.1.4 Changing the sum of object types

When the objects of one object type are transformed into objects of its super object type at the next higher level in the database transformation process, the values of some attributes of each object of the object type should be summed and the sum assigned to the attribute of the corresponding super object type. For example, according to the hierarchical structure shown in Figure 3.2, super object type irrigated paddy has two object types rice and maize and there are 4 parcels (objects) which belong to object type maize and also there are another 3 parcels (objects) which belong to object type rice. The area of each object belonging to rice is 5 ha, 40 ha, 30 ha and 50 ha respectively. The area of each object belonging to maize is 40 ha, 70 ha and 45 ha respectively. There are two ways to get the area of super object irrigated paddy. One is summing each area of objects which belong to object type irrigated paddy before transformation. It will be 280 ha. The other is computing the area of super object irrigated paddy after database transformation. The two results may not be identical since the spatial distribution of objects and the operations on the objects must be taken into account. These will cause the area of the object to increase or reduce.

For example, there is a small object of object type A whose size is in conflict with the geometric constraint of the object and object type A has IS_A relationship with super object type B. Supposing that the small object is adjacent to the objects of object type which has super object type C. After merging the small object with its neighboring object, the area of the small object will be added to the area of super type C and not to super object type B. Then object type B will lose some area.

3.4.1.2 Transformation Based on An Aggregation Hierarchy

Similar to geo-spatial transformation based on a classification hierarchy, problems concerning transformation based on an aggregation hierarchy are related to the aggregation hierarchical structure as well as the properties of object types that a categorical database contains.

This structure also reflects some aspects of data abstraction and has the following characteristics:

- Composite object types at different levels in an aggregation hierarchy correspond to objects of different complexities.
- Specifying component object types implies, to a certain extent, the determination of the complexity level of a database.

- Specifying a composite object type leads to a spatial abstraction and to more complex thematic object description.
- A composite object type can be built up from elementary object types and some composite object types can be put together to build up more complex composite object types.
- The links PART-OF relate a particular set of objects to a specific composite object and on to a specific more complex composite object (composite object type) and so on.
- A composite-type can be the component object type of another (super) composite object type.
- The attribute structure of a composite object type inherits completely or partly the attribute structure of its component object type at the next lower level or maybe there are some new attributes which do not belong to the component object types to form attribute structure of the composite object type.

The new data model will consist of these composite object types. Therefore, the relationships PART_OF play a key role in establishing an aggregation hierarchy and constructing a new data model and transforming the contents of the existing database.

It is necessary to define composite types by means of their construction rules. For each level of an aggregation hierarchy, there should be some rules for selecting the objects that should be aggregated to a particular composite object or selecting elementary object types to form composite object type. These rules will be based partly on the topological relations between objects (e.g. connectivity or adjacency) and partly on the thematic relations among object types (e.g. attribute structure of object types).

The detailed processes of transformation based on an aggregation hierarchy is conducted through specifying rules which are involved in thematic, geometric and topologic aspects given below:

- **Specifying PART-OF Relation Between A Composite Object Type and Component Object Types (Between Classification Hierarchies)**

Thematic rules specify the object types that build certain aggregated object types; geometric-topologic rules specify which component objects form an aggregated object. In a sense, specifying or changing PART-OF relations between the composite object type (object) and component object types (objects) means reconstructing aggregation hierarchy and leads to building a new geo-spatial model at an aggregation level. This will result in a database transformation.

Figure 3.17 shows that component object type Yard and fields are aggregated into composite object type Farm and this composite object type with an other composite object type Vacant land to form a composite object type village and so on turn into county. A

component object filed can only be PART-OF one farm, which can only be PART-OF village, which is only PART-OF one county.

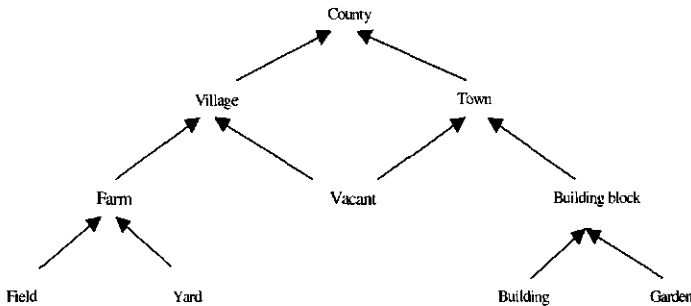


Figure 3.17 Example of aggregation relations of types

Note that when changing these specifying relationships between composite object type and component object types, the aggregation hierarchy will be changed as well. This will result in constructing a new geo-spatial model and the transformation of the database.

- **Establishing the Attribute Structure of Composite Object Type:**

After specifying the relations between a composite object type and component object types, the attribute structure of each composite object type in the aggregation hierarchy must be established. There are several methods to establish the intension of a composite object type, which depends on the application requirements. In other words, the establishment of the attribute structure of each composite object type is application-driven. There are several possible approaches to build attribute structure of a composite object type. These approaches are:

- Establishing the attribute structure of a composite object type through a union of the attribute structures of its component object types.

Note that a composite object type consists of all the attributes of its component object types. The attribute structure of the composite object type is a union of its component object types.

- Establishing the attribute structure of a composite object type through a union of part of the attribute structures of its component object types

Note that a composite object type consists of part of the attributes of its component

object types. The attribute structure of the composite object type is a union of part of its component object types.

- Establishing the attribute structure of a composite object type through a union of part of the attribute structures of its component object types and adding some new attributes

Note that a composite object type consists of part of the attributes of its component object types and some additional attribute to form the attribute structure of the composite object type.

- Establishing the attribute structure of a composite object type based on a new attribute structure

Based on PART-OF relation between composite object type and component object types, the attribute structure of a composite object type can be established without taking the attribute structures of its component object types into account. This depends on the application case. The procedure of establishing intension of a composite object is the same as previously discussed.

In fact, a database transformation process is conducted based on an aggregation hierarchy through the rules which form the aggregation hierarchy and control the spatial operations on objects in the database.

3.4.2 Object Transformation

In the previous part, the aspects of changing the geo-spatial model have been discussed in detail. The changes in a geo-spatial model associated with the database will cause changes in geometric characteristics and attribute values of objects. In the following, it will be described how the geometric characteristics and attribute values of the objects can be changed in the database transformation. Due to the changes in the geo-spatial model, the original spatial relations among objects in the database will be changed accordingly.

In contrast with changing a geo-spatial model, reorganization of the spatial partition by aggregation of adjoining objects does, in general, not lead to a change in the map legend. The geometric properties of objects which conflict with the constraints will result in object changes. This will be discussed in the next chapter.

3.4.2.1 Aspects of Object Transformation

Object transformation consists of creating new objects of new object types, object aggregation and assigning the attribute value for objects.

- **Creating A New Object of A New Object Type**

In section 3.4.1, establishing attribute structure of an object type (or composite object type) in a new geo-spatial model is discussed. The lowest object types in the new geo-spatial model are instanced based on thematic relations. This will result in creation of new objects.

Let Sc be super class

if $o_i \in Sc$ and $o_j \in Sc$ and $\text{adjacent}[o_i, o_j]=1$ then

create o_n with

$\text{part}[o_i, o_n]=1$ and $\text{part}[o_j, o_n]=1$

$\text{part}[o_i, o_n]$ and $\text{part}[o_j, o_n]$ express object o_i and o_j are a part of object o_n respectively.

- **Object Aggregation**

After creating new objects based on a new geo-spatial model, these new objects need to be processed according to their thematic and geometric properties. For example, two new adjacent objects may have the same attribute structure. Some rules of object aggregation are identified as follows:

- Aggregating a set of objects to form a new object based on geometric characteristics of objects;
- Aggregating a subset of adjacent (connected) objects of the same type or similar object types, or a subset of adjacent (connected) objects of the same type that have the same value(s) of a certain attribute to form a new object;
- Aggregating a subset of adjacent (unconnected) objects of the same type or similar object types, or a subset of adjacent (unconnected) objects of the same type that have the same value(s) of a certain attribute to form a new object;
- Aggregating a set of objects to form a new object based on the object function in which the related objects may not be within the framework of one classification hierarchy. For instance, aggregating farm yards and fields into farms, in which only the farm yards and fields that are adjoining and belong to the same farmer should be aggregated, and another example concerning the second case is to create an object university by aggregating those element-objects that are adjacent and belong to the same object university.
- The boundary of the composite object can be defined only through the geometric and thematic description of the component objects and spatial relationship among them,

which means that the boundary of the composite object can be delineated by simply aggregating the existing element-objects. For example, a building-block is defined as an aggregation of all the adjoining buildings and gardens.

- **Assigning the Attribute Values for the Objects.**

New objects need to be assigned some attribute values. The approaches for assigned attribute values are identified as following:

- Assigning attribute values of each object of composite object type in the new database, based on the corresponding attribute value of each object of the object type in the original database.
- Assigning attribute values for each composite object of a composite object type in a new database based on the sum or average of corresponding values of its constituents in the original database. For one application, object type Road may have attributes of number-of-lanes and traffic-volume, whereas for another application these may not appear. The value of the composite object may be the sum or average value of lanes of its constituents.

3.4.2.2 Forms of Object Transformation

Forms of object transformation could be:

- An elementary object of an elementary object type is transformed into an elementary object of an elementary object type.
- Elementary objects of an elementary object type are transformed into an object of a super object type.
- Component objects of a component object type are transformed into a composite object of a composite object type.
- Composite objects of a composite object type are transformed into a composite object of higher composite object type.

3.4.2.3 Types of Object Transformation

There are three types of object transformation, identified as follows:

- 1-1: An object in a source database is transformed into an object in a target database. The dimension of the object in the target database may be changed, such as area to line or line to point. This transformation mainly is controlled by semantic and geometric constraints. There is a special case, in which an object in a source database is deleted

and this object will not exist in a target database as shown in Figure 3.18.

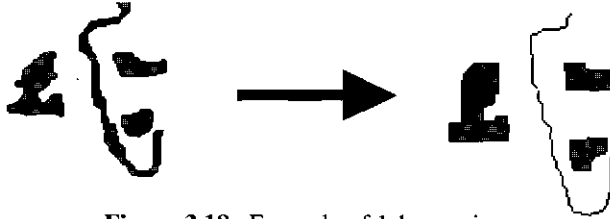


Figure 3.18 Example of 1:1 mapping

- M-1: Grouping a particular set of objects in a source database into a specific composite object in a target database based on semantic relations (PART-OF, IS-A), and adjacent (connected or unconnected) relations among the objects as shown in Figure 3.19.

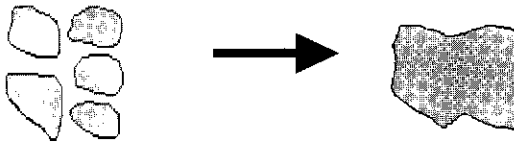


Figure 3.19 Example of M:1 mapping

- M-N: A set of objects which reflect (represent) some spatial pattern such as transportation network and drainage network in a source database are transformed into a higher level set of objects keeping the original spatial pattern structure intact in a target database as shown in Figure 3.20.

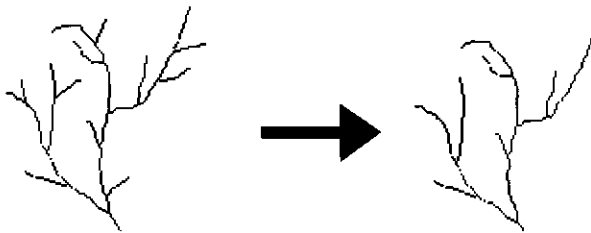


Figure 3.20 Example of M:N mapping

3.4.3 Relation Transformation Among Objects

Depending on the application domain, some spatial relations may be more important than others. As for database generalization in which the data are organized by Formal Data Structure (FDS) for a single valued vector map, we only use connectivity, adjacent, inclusion and proximity relations for database generalization. The object transformation from a source database to a target database will result in changing connectivity, inclusion and proximity relations among objects in a target database. The proximity relations or inclusion relations among objects at one detail level cannot be distinguished at a less detailed level. Several aspects which cause changes in relations have been identified as following:

3.4.3.1 Causes of Relation Changes

- Changing the geo-spatial model of a database resulting in spatial and relationship changes among objects and object types, such as two spatial adjacent objects with different attribute structures. When the geo-spatial model is changed and the two object types which belong to two different object types respectively become one general object type, the two objects will have the same attribute structure after transformation. This will result in changing the original spatial and semantic relations.
- Changing geometric characteristics of a spatial object results in spatial relationship transformation, for example, changing the geometric shape of objects, and merging or simplifying or deleting objects will result in inclusion and adjacent changes.
- Changing the spatial relation of any two objects will result in spatial relationship transformation among objects, for example, changing distance relation among objects will lead to a spatial relation change from visual adjacent to adjacent.
- Changing the thematic characteristics of spatial objects will result in spatial relationship transformation, for example, changing two adjacent objects with different attributes into two objects with the same attribute will result in losing the adjacent relation between two objects which will disappear.

3.4.3.2 Forms of Changing Spatial Relations

The forms of the spatial relation changes can be identified as follows:

- Proximity relation disappears when the distance between the neighboring objects is less than the distance threshold among the objects as shown in Figure 3.21;

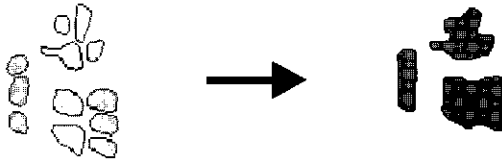


Figure 3.21 Example of a proximity relation change

- Inclusion relation between two objects disappears when the object which is included in another object violates the geometric constraints or when they share identical or similar attributes after transformation as shown in Figure 3.22;

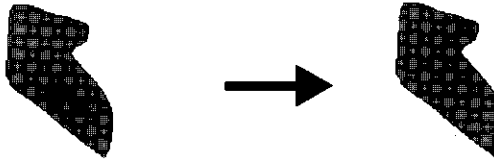


Figure 3.22 Example of an inclusion relation change

- Connectivity relation between two objects disappears when they share identical or similar attributes after transformation or when one of them violates geometric constraints as shown in Figure 3.23;

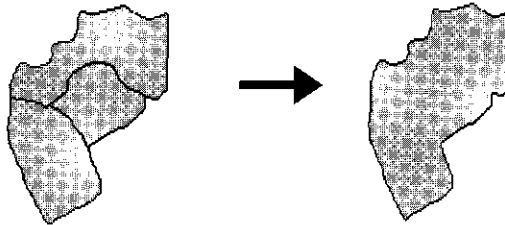


Figure 3.23 Example of a connectivity relation change

- Visual connectivity relation is changed into connectivity relation when the distance between two objects is too narrow and these two objects belong to different object types as shown in Figure 3.24.

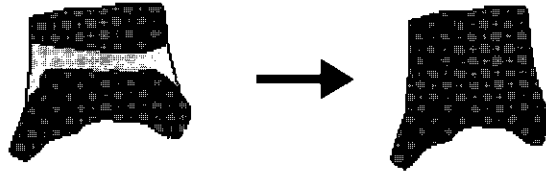


Figure 3.24 Example of a visual connectivity relation change

3.5 Transformation Operations

The basic operations related to database transformation can be categorized into two types. One is the operations on the geo-spatial model or object type associated with a database which will lead to changes in the abstraction levels; the other is the operations on objects in the database.

- **Operations on Geo-spatial Model**

Based on the concepts of geo-spatial model transformation discussed previously, operations on the geo-spatial model can be identified as follows:

- **Select:** Selecting a set of object types from an existing data model to form a new geo-spatial data model associated with a database or selecting attribute items from the attribute structure of an object type to form the attribute structure of a new object type.
- **Add:** Adding some attributes to the attribute structure of an object type in an existing data model to form an attribute structure of a new object type in a new geo-spatial model.

- **Operations on Objects**

Reducing the number of objects in a database is the main task in the database generalization (transformation) (Molenaar 1998, Weibel 1995). The operations on objects should reflect the characteristics of reducing the number of objects in the database. Three operations on objects will be involved in this thesis.

- **Select:** a selection operation that selects objects of object types associated with a new geo-spatial model from the existing database to form a target database based on the

requirements of application and constraints.

- **Aggregate:** an aggregation operation that reduces the number of objects by merging objects which are adjacent in spatial and have the same object type or similar in semantics to build a single new object. This operation includes the dissolution of boundaries of these objects to build a new object and assignment of a new value to the new object (at a higher aggregation level). For example, a cluster of small objects may be combined to form a larger object with some assigned attribute values.
- **Delete:** a deletion operation that deletes the objects which are unrequired within an object type in the new database.

3.6 A Framework of Transformation

To implement the database generalization, four stages can be taken into account. They are definition of a new geo-spatial model which is associated with a new database, data model transformation, object transformation and relation transformation. In a sense, data model transformation controls the object transformation and relation transformation. Relation transformation is also governed by object transformation. Figure 3.25 illustrates these stages.

3.7 Conclusions

This chapter presents the contents and the framework of database generalization. The database generalization as a transformation should include defining a new geo-spatial model, geo-spatial model transformation, object transformation and relation transformation based on related concepts of geo-data and GIS discussed in Chapter 2. The formalization of classification hierarchy and aggregation hierarchy should be defined first according to the intension and the extension of a class or object type.

The classification and aggregation hierarchies play an important role in linking the definitions of spatial objects at several scale levels and constructing a geo-spatial model. They are application-dependant. An adequately supporting geo-spatial model provides a description of object types and the relationships among them. IS-A relationship in the classification hierarchy and PART-OF relationships in the aggregation hierarchy relate a group of objects of lower object types to the objects of corresponding object types at the next higher level. Changing these two types of relations will change the data model and result in a database transformation. In an information abstraction process it is frequently necessary to build instances of the composite-type using the existing objects of the component-types.

Object and relation transformation deal mainly with the preservation of typical shapes (on the object level) or with the preservation of spatial patterns and alignments of objects, whereas geo-spatial model transformations deal with the preservation of the logical context of objects and degree of detail (on the object type level). Geo-spatial model transformations control object

transformations and relation transformations.

Before transforming an existing database to a new database, the geo-spatial model of the new database must be defined and the constraints which will influence the transformation must be identified and classified. They are application-dependant. Transforming the objects and relations among the objects from the existing database to a new database is the objective of database transformation. The transformations of objects are involved with the geometric and thematic properties of the objects. The transformations of relations include ones of spatial and semantic relations.

The operations in the database model transformation are divided into two types: (1) on the object types and (2) on the objects.

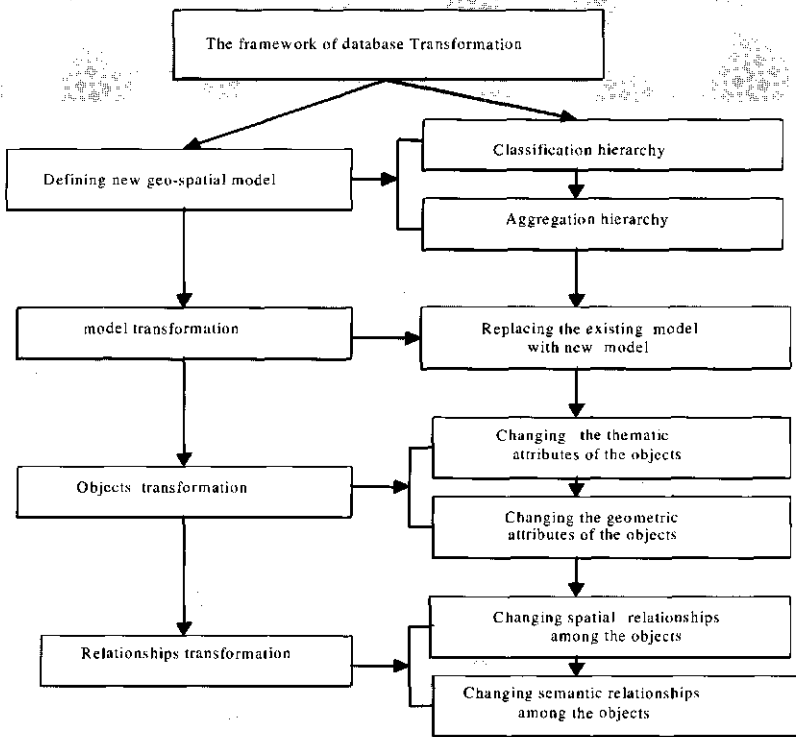


Figure 3.25 A framework of database transformation

Chapter 4

Constraints in Categorical Database Generalization

4.1 Introduction

As pointed out earlier, database transformation is based on some conditions and application-dependant rules. In this chapter, the classification of constraints is developed and the scope of these constraints is examined. The concept of transformation unit is given on the basis of the constraints and transformation requirements. The transformation unit (to be discussed in section 4.4 of this chapter) defines the processing unit which will consist of a set of objects based on their geometric, thematic and spatial relation characteristics in the process of database generalization transformation. Finally, the relationships between constraints and operations are discussed.

4.2 Constraints for Database Transformation

Some authors have discussed constraints in map generalization, such as Robinson et al. 1985, Brassel and Weibel 1988, Kemppainen 1992, Weibel and Dutton 1998, Beat and Weibel 1999, Beard 1991, Ruas 1998, 1999 etc. In the context of map generalization, a constraint can be defined as a design specification to which the solutions of a generalization problem should adhere (Weibel and Dutton 1998).

4.2.1 Constraints

In the context of database generalization, constraints can be defined as a set of specifications or conditions of a geo-spatial model, geometric and thematic characteristics of objects, and relationships among objects in a target (or generalized) database. This set of specifications governs or guides the process of database generalization transformation. They specify the nature of a database to be produced or to be generalized. Two types of constraints can be identified. One is the generalization constraints which incite generalization (e.g. size is too small) and the other is the maintenance constraints which incite the preservation of a characteristic. These constraints are essential to preserve the geographical meaning, to identify connections between constraints and methods and to build a new database or to generalize an existing database.

4.2.2 Functions of Constraints

During the process of generalization, constraints may be used:

- To provide steering parameters for how many object types and objects should appear in a database given a particular context and resolution, e.g., some threshold values such as

density, proximity, maximum and minimum etc. moreover, a priority of constraints derived from the examination of attribute values and the proposed generic generalization specifications.

- To detect and identify areas and objects which violate the constraints, for example, by matching objects with the object constraints to evaluate the quantity and severity of constraint violations;
- To create transformation units;
- To guide the choice of operators and generalization process according to constraint priorities;
- To control the process of generalization and the effect of an algorithm by detecting constraint violations on objects after each transformation.

When the process of generalization begins, we should be able to recognize and qualify constraint violations and characteristics associated with the information we have to generalize. Such characterization relies on the computation through appropriate methods of analysis.

For the categorical database transformation, we should specify: which object type (classes), what minimum size for the object of each object type (class) and which classes of component objects are to be used to build a composite object of this type etc in the new database before transformation. These are the examples of constraints.

4.3 Classification of Constraints

From a map generalization point of view, some authors (Beard 1991, Ruas 1998, and Beat,P and Weibel R. 1999) have proposed some classification of constraints such as classifying constraints into four classes: graphic, structural, application and procedural. The discussion on the classification of constraints concentrates more on categorical database transformation in this study. The classification of constraints should take the aspects of a geo-spatial model, spatial and semantic properties of objects and relations among objects into account. Constraints may further be distinguished by their scope.

4.3.1 Classes of Constraints

The main objective of constraints is specifying the necessary conditions that define a particular problem and organizing them consistently. Constraints can be characterized in various ways. In order to correspond with the types of database transformation discussed in the last chapter, we adopt the following categories:

- Geo-spatial model constraints
- Object constraints

- Relation constraints

- **Constraints on Geo-spatial Model:**

Constraints on a geo-spatial model define the new classification hierarchy and aggregation hierarchy associated with a target categorical database, decide how many object types should appear in a target database given a particular context and detail level, and determine the domain of each attribute, scale type of the domain of attribute and cardinal of object types. At the same time, the semantic relations among the object types should be clearly defined. Geo-spatial model constraints deal with the preservation of the logical context of objects and degree of detail.

- **Constraints on Objects:**

Constraints on objects specify the requirements of the geometric and thematic properties of the objects in the (generalized) target database. They provide some steering parameters and determine which objects can be retained after generalization in the database, such as minimum size properties which are mainly dictated by application requirements, not by graphic limits, and in which reasonable representation is emphasized, not legible representation.

- **Constraints on Relationships:**

Constraints on relationships ensure existing relationships of connectivity, adjacency and containment between objects and between object types are maintained. Maintaining relationships in the attribute domain is equally important. To generalize a set of objects, it is necessary to have a great deal of information on spatial and semantic relations of objects.

The object and relation constraints deal mainly with the preservation of typical shapes (on the object level) or with the preservation of patterns and alignments (relationships among objects) if multiple objects are involved.

4.3.2 Scope of Constraint Classes

Constraints may further be distinguished by their scope. The three types of constraints are discussed more in detail in the rest of this section. The objective is to understand the ontology of these constraints and how they can be used in the generalization process.

4.3.2.1 Constraints on A Geo-Spatial Model

In the categorical database, the geo-spatial model is a kind of multi-level structure (hierarchy). Both classification hierarchy and aggregation hierarchy reflect a certain aspect of data abstraction. They play an important role in linking the definition of spatial objects at several

scale levels (Molenaar 1996, Peng 1997, Peng and Tempfli 1997, Peng, 2000, Richardson 1993 and Smaalen 1996).

Based on the discussion about the relationships between a geo-spatial model, and a classification hierarchy and an aggregation hierarchy in Chapter 2, constraints on a geo-spatial model can be divided into constraints on a classification hierarchy and constraints on an aggregation hierarchy.

- **Classification Hierarchy Constraints**

Object types at different levels in a classification hierarchy correspond to data of different complexity. In this sense, specifying an (elementary) object type, to a certain extent, determine the abstraction/complexity level of a geo-spatial model. Changing the object types of an existing data model to the ones at a higher level in the same hierarchy would mean transforming the data model from a lower abstraction level to a higher abstraction level (Peng 1997) .The constraints on the classification hierarchy which control the processes and abstraction levels of generalization may include:

- Theme of a generalized database;
- Hierarchical structure associated with an existing database;
- Attribute structure of each object type;
- Intension of each object type;
- Domain of attribute;
- Cardinal of each object type;
- Lower levels in the hierarchy corresponding to lower abstraction levels result in more complex data, including both thematic and spatial aspects;
- Higher levels corresponds to higher abstraction levels lead to less complex data;
- Level in which an object type is located in its associated classification hierarchy corresponding to the degree of abstraction;
- Number of elementary object types;
- Number of attributes contained in an object type;
- One object only belonging to a class and a super class.

- **Aggregation Hierarchy Constraints**

A composite-type can be the component-type of another (super) composite-type. This implies that replacing the component-types in a model by their composite-type will result

in transforming the model from a lower abstraction level to a higher abstraction level (Peng 1997). Constraints for aggregation hierarchy may include:

- Theme of a generalized database;
- Hierarchical structure associated with an existing database;
- Composite-types in the hierarchy corresponding to higher abstraction levels will result in less complex data;
- Component-types corresponding to lower abstraction levels will result in more complex data;
- Level in which an object type is located in its associated aggregation hierarchy corresponds to the degree of abstraction;
- Level in which the associated domain of an attribute of an object type is located in its associated aggregation hierarchy corresponds to the degree of abstraction;
- Attribute structure of composite object types;
- Number of attributes contained in an object type;
- Number of composite-type;
- Specifications (rules) specifying the component types of the component objects for building a composite object of this type;
- Specifications (rules) specifying the geometric and topologic relationships among these objects;
- Part of relationship specifying a specific composite object and its constituent parts at different levels.
- Specifications specifying a specific object type and its constituent parts at different levels.

4.3.2.2 Constraints on Objects

The spatial object is an instance of an object type. Object types are classes of spatial entities in a geo-spatial model. In reality, they may be a road, city, river, land use unit and so on. Three types of constraints can be identified based on the characteristics of spatial objects. They are thematic constraints, geometric (spatial) constraints and temporal constraints. Temporal constraints and their related aspects are not discussed in this study.

- **Thematic Constraints**

Thematic constraints are specifications that indicate the thematic abstraction level of the objects in a database generalization. They will include:

- The same geo-phenomena should be described by the same thematic resolution throughout the entire database;
- No object has common boundaries with other objects having the same object type. If the case occurs, the separating boundary is dropped;
- Adjacent (connected) objects which belong to different object types may be aggregated if they belong to the same super object type;
- The area of a small eliminated object should be added to the area of the object which has highest similarity with the eliminated object among its neighboring objects, or be averaged to the area of each neighboring object if its neighboring objects have almost the same similarity with it;

These aspects, and the number of object types that a database contains, determine the thematic constraints of the database. Thematic constraints may be ranked by nominal, order, interval and ratio, but cannot be measured.

- **Geometric Constraints and Spatial Pattern**

The geometric (spatial) constraints of objects in a database mainly deal with aspects of the size, width and distribution structure of objects. It mainly meets the requirements of application. In other words, they are application-dependent. It comprises:

- Minimum object size (minimum size for area objects, or minimum length for line objects);
- Minimum detail of the objects that a database can contain
- Minimum space between objects, i.e., objects of a different object type may be aggregated if the distance between them is less than the minimum space or objects of the same object type may be amalgamated if the distance between them is less than the minimum space;
- Preserve the global distribution of objects;
- Preserve typical shapes and angularity of objects of each object type;
- Preserve the given size distribution of objects for each object type;
- The boundaries of the object of a super object type or composite object type must coincide with the boundaries of the geometric union of the boundaries of the constituent objects.

These two aspects of constraints of spatial objects apply partly to objects of an object type, partly to the entire database, and may take different values for different object types in the same database.

4.3.2.3 Constraints on Relations among Objects

Two types of relation constraints can be identified for spatial objects. They are spatial relation constraints and semantic relation constraints on objects. Spatial relation constraints are classified into topological relation constraints, direction relation constraints and distance relation constraints. Depending on the application domain, some spatial relations may be more important than others. We apply topological relationships (connectivity, adjacent, inclusion), distance relation and directional relation to a database generalization.

- **Topological Relations Constraints**

Topological relationships determine the neighbors of one object. They constrain the behaviors of objects in spatial aspects. Topological relations should be preserved after generalization in the database. Topological relations constraints may include:

- Topological constraints deal with basic topological relationships like connectivity, adjacency and containment, which should be maintained when data are generalized.
- An object must not move across the boundary of another object;
- Avoid introduction of illogical neighborhood relations (e.g., houses in a lake);
- Avoid separation of an object when deleting parts of it, i.e., maintaining connectivity;
- Avoid introduction of self-intersection of object outlines;

- **Directional Relation Constraints**

Cardinal directions describe, qualitatively, the orientation between objects (Frank 1991). The directional relationship between two objects is an important spatial property and can also be used as selection criterion for retrieving objects from a database. A set of spatial objects having certain alignments and patterns along a certain direction in a database reflect spatial distribution property of geographical entities. These alignments and patterns of the objects should be preserved after generalization. The direction relation constraints may include:

- Preserve typical alignments and patterns of objects within a group of objects in space;
- Preserve the relative location of one object in relation to other ones after generalization;
- Objects of the same object type may be amalgamated if they are distributed in a certain direction and the distance between them is less than the minimum space;
- Objects of different object types may be aggregated if they are distributed in a certain direction and the distance between them is less than the minimum space.

- **Distance Relation Constraints**

Distance relation can be described as the approximation among objects in the database.

The distance constraint specifies the distance between two objects that can be merged in database generalization. Distance relation constraints may contain:

- Avoid merging two adjacent objects of the same type when the distance between them is larger than the minimum space;

- **Semantic Relation Constraints**

Semantic relations are also of essential importance to reduce the number of objects in an object type. Semantic relation between two objects limits object behavior in the semantic aspect in the database.

These constraints depend on the database specification. Semantic constraints should be used as indicators. They may contain:

- Objects of a sub object type having IS-A relationship may be amalgamated to form an object of the higher object type;
- Objects of a component object type having PART-OF relationship may be aggregated to a composite object of the composite object type;
- Objects of a composite object type having PART-OF relationship may be aggregated to a composite object of the higher composite object type;
- A set of small objects having the same similarity in semantic aspects may be merged to a larger object;
- An object having a specific function should be maintained.

4.4 Transformation Unit Based on Constraints

The existing database must be transformed if the contents of the existing database do not meet the user's requirement. We call the object and object type which violate the constraints as conflicted object and conflicted object type respectively. For example an object whose area is too small violates a size constraint, two objects too close violate a distance relation constraint. We can say that the process of database transformation is the one of solving conflicted objects and object types.

The transformation of the conflicted object does not necessarily show a 1:1 relation between objects from both the existing database and the target database. In fact, the process of solving conflicted objects is one of analyzing and making a decision about the conflicted objects. Not only have the shape, size and thematic information of conflicted objects to be analyzed, but also adjacency (connected or unconnected) between objects. Alignment and global distribution also need to be checked. In order to analyze, describe and transform (generalize) conflicted objects, a notion of *transformation unit* is introduced which is used to identify, formulate and represent

the properties of a conflicted object and its neighboring objects as well as the relations between these objects.

4.4.1 Transformation Unit

A *transformation unit* is a group of objects or object types, in which there are adjacent relations among these objects or semantic relations among these object types and there is at least one object or object type violating the constraints. In building transformation units, the conflicted object is the seed around which a set of objects which have an adjacent relationship with it can be found to form a *transformation unit*. The conflicted object and the adjacent relation are essential to creating a transformation unit. The transformation unit plays an important role in the aggregation process in this thesis. For example, a *transformation unit* object can be a group of close land use objects or a district, a town or a street network within a town.

A *spatial transformation unit* can be expressed as a set of objects. Given a set of objects $O = \{o_1, o_2, \dots, o_n\}$, and a set of conflicted objects $O_s = \{o_{s1}, o_{s2}, \dots, o_{sm}\} \subseteq O$. *Transformation unit* can be formalized as follows:

Let TU be a set of transformation units;

$tu_i \in TU$;

$tu_i = \{ o_{si} \mid o_{si} \in O_s \wedge o_j \in O \wedge \text{adjacent}(o_{si}, o_j) \} \quad (1 \leq si \leq m, 1 \leq j \leq n)$.

Where:

tu_i : *i* transformation unit.

When the process of generalization begins, we should be able to recognize and qualify conflicted objects and construct *transformation units* based on them since they are the trigger of the transformation.

4.4.2 Characteristics of Transformation Unit

The characteristics of a transformation unit are given below:

- A *transformation unit* is a basic unit, which gives specific information on data organization in order to get a better understanding of geographic meaning; The characterization of a transformation unit is always driven by generalization purposes.
- A *transformation unit* is a trigger of the generalization transformation. It indicates where needs to be generalized;

- A *transformation unit* limits the area and the number of objects to be processed at one time;
- A *transformation unit* is a controller. According to the new database specifications, the constraints are utilized to limit the problem space. These constraints can be used to control and guide the process of database generalization transformation. As long as some constraints are violated, the process of generalization will be carried on;
- A *transformation unit* may be described by means of specific rules such as Constraint Delaunay Triangulation (CDT) used dynamically.
- A *transformation unit* may limit the range when analyzing certain relationships between geographical objects.
- A *transformation unit* simplifies the spatial analysis process. Generalization requires a deep spatial analysis as we have to remove a large set of information but we do not know a priori which information should be removed since a random shift and deletion of objects is intuitively not a good generalization method.
- A *transformation unit* allows us to greatly simplify reasoning in the generalization process and is helpful for choosing an aggregation operation.

4.4.3 Types of Transformation Unit

Peng (1997) groups all objects in the database into three types according to their geometric structure and spatial distribution such as linear-generalization-units, complex-generalization units and simplex-generalization-units. Ruas (1998) classifies the objects into Macro, meso and micro classes from the geographic analysis point of view. We categorize the objects in a database into semantic transformation unit, complex transformation unit and simple transformation unit based on the concepts of constraints for aggregation operation purposes in this study.

The object type transformation unit, simple transformation unit and complex transformation unit perform different roles during the generalization process.

- **Semantic Transformation Unit**

A semantic transformation unit consists of at least one conflicted object type and a set of object types which have the same super object type at the next higher level or the most similarity to the conflicted object type in the hierarchy associated with the database.

An object type transformation unit is necessary to control the quantity evolution and transformation of semantic and thematic aspects of a class. A set of sub object types which belong to an object type or are specified to a composite object type is a typical object type

transformation unit. An object type transformation unit gives information to its component objects to control quantity evolution.

- **Simple Transformation Unit**

A simple transformation unit consists of a conflicted object and its 1st neighboring objects with a direct adjacent relation (see Figure 4.1 (a)) or visual adjacent relation (see Figure 4.1 (b)) with the conflicted object.

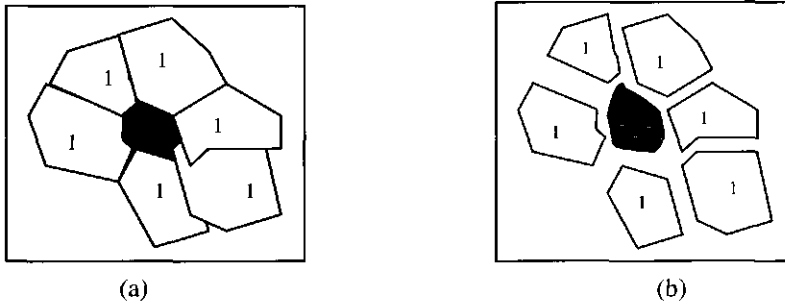


Figure 4.1 Example of a simple transformation

- **Complex Transformation Unit**

A complex transformation unit comprises a conflicted object and a set of objects with 1st and 2nd neighboring relations to the conflicted object. There are many types of complex transformation units. Figure 4.2 shows one example.

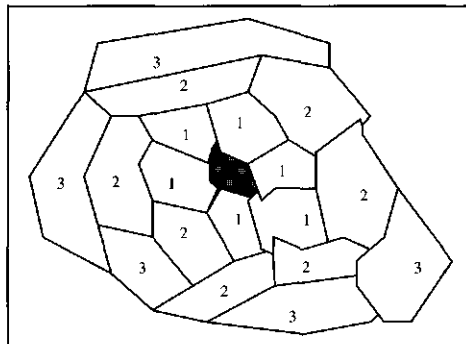


Figure 4.2 Example of a complex transformation unit

These three kinds of transformation units are basic concepts for constructing other types of transformation units and the other transformation units can be composed of them. The construction of transformation units in categorical database generalization will be discussed in Section 6.2.

4.4.4 Examples of Transformation Units

Basically, generalization is the process of a change of state as pointed out in the last chapter. At a specific time an object is active. If it is a conflicted object, it is the seed to construct various transformation units.

In the process of database generalization transformation, the constraints are analyzed and identified. All objects and object types selected for a generalized representation are tested against these constraints and any objects violating constraints are tagged as requiring seeds. Transformation units are constructed according to these conflicted objects and object types. Generalization operations or other actions are then used to correct these violations.

In order to illustrate these principles several transformation unit examples which are based on the conflicted object type and conflicted object are given below:

- Case 1: An object that is too small and the adjacent objects around it form a transformation unit. Figure 4.3 (a) depicts a transformation unit with a too small conflicted object c. Figure 4.3 (b), (c) and (d) show the possible results of transformation (this will be discussed in more detail in Chapter 6).

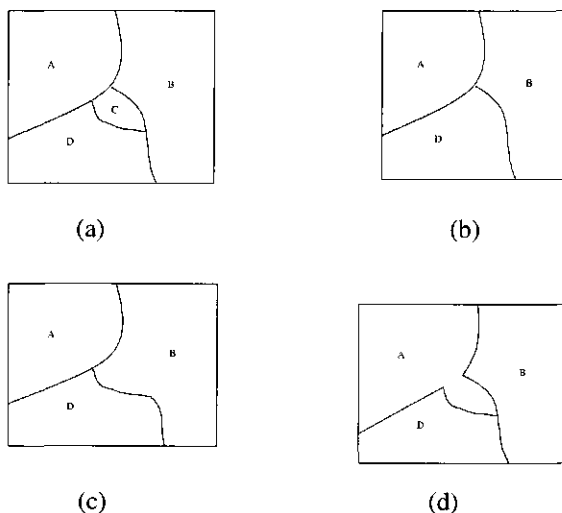


Figure 4.3 Case 1 of a transformation unit and its transformation

- Case 2: A set of small objects to form a transformation unit. Figure 4.4 (a) depicts a transformation unit with a set of small conflicted objects . Figure 4.4 (b), (c) and (d) show the possible results of transformation (this will be discussed in more detail in Chapter 6).

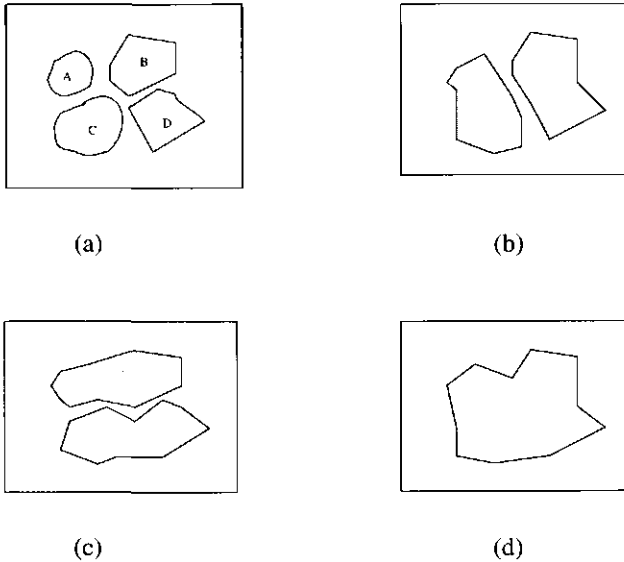
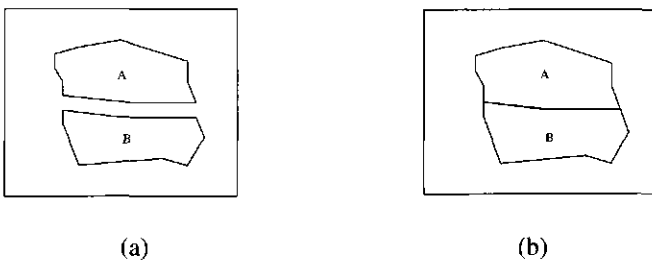


Figure 4.4 Case 2 of a transformation unit and its transformation

- Case 3 : If the distance between two objects is too short and less than the threshold, then they form a transformation unit. Figure 4.5 (a) depicts a transformation unit with a short distance between two objects, conflicted object A,B. Figure 4.5 (b), (c) and (d) show the possible results of transformation (this will be discussed in more detail in Chapter 6).



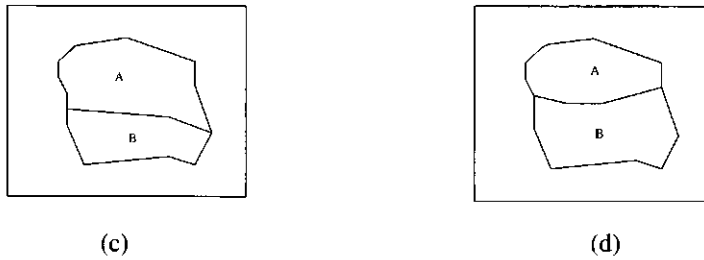


Figure 4.5 Case 3 of a transformation unit and its transformation

- Case 4 : A small object which is included in a larger object with its 1st and 2nd neighboring objects to form a transformation unit. Figure 4.6 (a) depicts a transformation unit with a small conflicted object A, which is included by another object. Figure 4.5 (b) and (c) show the possible results of transformation (this will be discussed in more detail in Chapter 6).

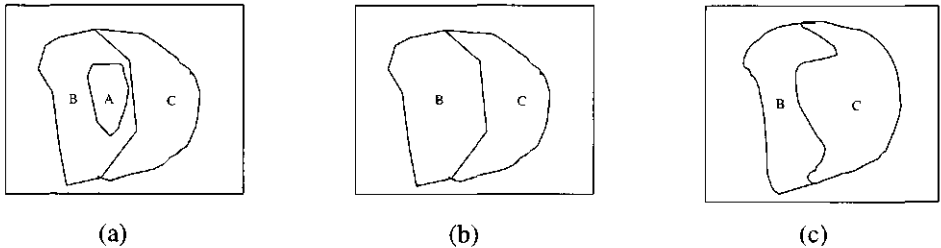


Figure 4.6 Case 4 of a transformation unit and its transformation

It is noted that all above examples of transformation are semantic-driven.

4.5 Representation of Constraints and Operations

The constraints and the operations play an important role in database generalization. Unlike each rule corresponding to an operation, the constraints correspond directly to the operations in database generalization. Constraints allow us to indicate where an action should be performed and how to construct transformation units. The operations perform the actions of generalization in support of data reduction in the database. If we want to use constraints as triggers to build transformation units, we need to represent them in the database. Some constraints apply partly to objects of an object type, partly to the entire database (object type), and they may take

different values for different object types in the same database. The constraints such as the attributes of the objects and the operations such as the methods of the objects can be encapsulated in classes according to Object-Oriented programming. They have the inheritance and polymorphism properties. The relationship of the constraints and the operations can be established and represented by the object-oriented method.

- Constraints related to an object can be represented by means of attributes at the object level (e.g. area too small, line too detailed) with either a flag or a quantitative value which describes the severity of the violation;
- Constraints related to a class of objects can be represented at the class level or by means of a specific attribute which is a constraint table that should be consulted during the process (Ruas 1998);
- Operations such as the methods of the object can be encapsulated in classes.

A constraint violation occurs when an object or a set of objects do not respect a constraint. For example an object whose area is too small violates a size constraint, two objects too close violate a distance constraint. An object violating the constraint will form a transformation unit as discussed in section 4.4.4. Some operations will need to operate on this transformation unit and solve the violation. Operations are chosen depending on application purpose and types of transformation units (to be discussed in chapter 6). Operations are applied to a database to correct, or preserve conditions specified by transformation units. In the context of this approach, the function of an operation must be clearly defined to anticipate or predict how it will interact with transformation units which are caused by the constraints.

4.6 Summary

Constraints describe explicitly either a set of the transformation specifications which must be respected in database transformation or some information in the existing database which must be maintained in the target database after transformation.

Constraints such as transformation conditions play a key role in the process of database generalization. Constraints can be used to identify conflict area, guide the choice of operations and trigger operations. They guide and govern database generalization. The processes of generalization should be performed by a series of operations under the control of constraints.

Constraints can also be classified into triggering constraints which will trigger some generalization operations and outcome constraints which will can stop generalization operations.

Three such types of constraints as geo-spatial model, object and relations based on an object-oriented database have been proposed in categorical database generalization. Constraints can be specified interactively by users and varied to reflect different objectives or purposes. All three

types of constraints are application-dependent. This will make the database generalization process very flexible/adaptive. And it ensures that decision-making is based on geographic meaning and not simply on the geometry of objects.

Chapter 5

Supporting Data Structure

5.1 Introduction

Database generalization is a complicated process of spatial analysis, decision-making and transformation of datasets. Whether or not a spatial object is changed not only depends on its geometric and thematic properties, but also the spatial relations and the semantic relations with its neighboring objects, and one can affect the other as long as they are neighbors direct.

Before a database transformation process is created, the new data model associated with a new (generalized) database needs to be specified. In a transformation process, the state of objects, spatial relations and thematic relations between objects and between objects and object types and between object types need to be examined, detected, identified and analyzed. Such a process will change the geometric and thematic properties of spatial objects whereas the spatial relations need to be maintained dynamically in order to meet transformation requirements. Dynamic maintenance of adjacent relations is a critical issue in the database transformation process. Database transformation requires a data structure which strongly supports data organization, spatial analysis and decision-making in a database. If there is no such data structure, a process involving a probably heavy computation and complicated algorithm is necessary in order to detect two adjacent objects and the dynamic maintenance of spatial and semantic relations is very difficult in order to complete the transformation process. The design of a data structure should take two functions into account. One provides the basis for describing and organizing spatial objects and the relationship among them and the other is for analyzing and supporting operations on spatial objects.

This chapter gives an outline of Formal Data Schema (FDS) and Constrained Delaunay Triangulation (CDT) first and then presents an Integrated and Extended Formal Data Schema(IEFDS). The FDS will be extended in formalization of spatial objects based on geometric properties of CDT and in extending adjacency and inclusion relations between different types of objects based on the roles of CDT, and in enhancing analysis function based on integration of FDS and CDT. The spatial query based on this data structure is also discussed.

5.2 Formal Data Schema and Constrained Delaunay Triangulation

This section will introduce briefly the concept of Formal Data Schema for single valued vector maps developed by Molenaar (1989, 1991) and constrained Delaunay triangulation (Delaunay 1934; G. Macedonio and M.T. Pareschi, 1991; Victor J.D. Tsai 1993).

5.2.1 Formal Data Schema

Formal Data Schema for single valued vector maps (SVVM) is an object-oriented topological (conceptual) data model, which combines aspects of object-oriented and topologic data model. point, line and area objects are represented with their geometric and thematic aspects. Their geometric representation contains information about topologic object relationships, whereas their thematic description is structured in object classes that may form generalization hierarchies. Such classification hierarchies in combination with the topologic object relationships of FDS support the definition of aggregation hierarchies of objects. These classification hierarchies and aggregation hierarchies play an important role in linking the definition of spatial objects at several scale levels (Molenaar 1996, Peng 1997, Peng and Tempfli 1996, Richardson 1993 and Smaalen 1996). The data structure consists of:

- Three object types, namely point object, line object, area object, classified according to the geometric description of the spatial object;
- Five geometric data types (geometric primitives), including coordinates, node, edge, triangle and face, the definition of which is based on planar-graph theory at node-edge level;
- A set of links between geometric data types (g-g links), and a set of links between geometric data types and object types (g-f links). It supports a number of elementary topological relationships, including area-area, line-line, point-point, area-line, area-point, and line-point relationships.
- The whole structure is shown in Figure 5.1 , in which, the term 'feature' is equivalent to 'spatial object', and the boundary of an 'area feature' is implicitly described by a list of arcs.

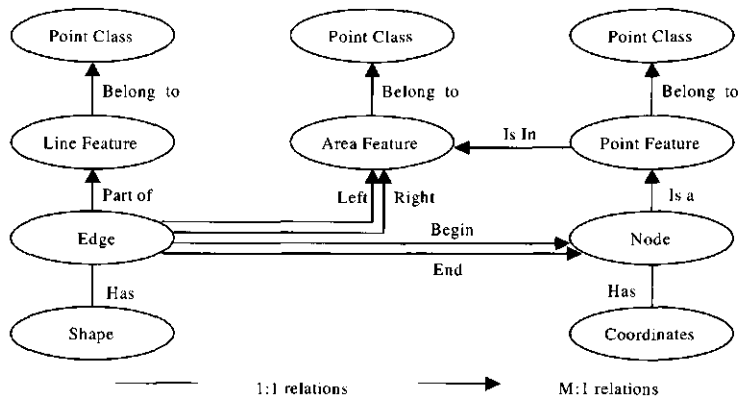


Figure 5.1 A single-theme data model (after Molenaar 1989)

Based on the characteristics of the categorical data and the database discussed in Section 2.5 and the properties of FDS, FDS will be a good conceptual data model to organize the categorical data for the purpose of database generalization.

The structure was extended to handle elevation by Pilouk and Tempfli (Pilouk and Tempfli, 1993) and further developed to handle 'multi-themes' by Kufoniya and Pilouk (Kufoniya and Pilouk, 1994). A tetrahedron-based data model was also used to handle 3D Modelling by Pilouk (Pilouk, 1996). A geometrically enhanced FDS was introduced by Peng (1997).

5.2.2 Constrained Delaunay Triangulation

A Delaunay triangulation is generally defined as a triangulation $W(N, E, T)$ of a set of points N with the empty circle property, that is, the circumcircle of any of its triangles $t \in T$ does not contain any point $n \in N$ (Preparata and Shamos, 1985). Here E is the set of all the triangle edges in the Delaunay triangulation. The Delaunay triangulation is unique and locally equiangular (Sibson, 1977, Solan, 1987), hence, it maximizes the minimum angle of its triangles compared to all other triangulations.

A constrained Delaunay triangulation $W(N, E, T, E_c)$ is an extension of the standard Delaunay triangulation which allows pre-described, non-intersection line segments (except at their endpoints) $E_c (\subset E)$ to be forced in as part of the triangulation. Note that triangles containing any of such pre-described edges may not necessarily be Delaunay triangles. Figure 5.2 shows examples of constrained and unconstrained Delaunay triangulation.

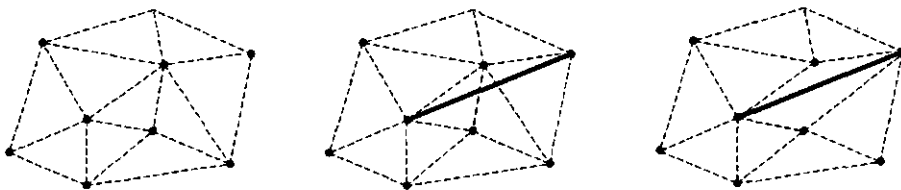


Figure 5.2 Example of DT and CDT (thick line =constraint)

An important property of the CDT is the adjacent relationship between two points connected by a Delaunay edge. Each Delaunay edge in a Delaunay triangle represents topology between two points. The triangulation aims at equilateral triangles or tries to approximate them as much as possible, so that the unexpected effect of long elongated edges can be minimized. Therefore the Delaunay triangulation is a very good candidate for spatial analysis amongst discrete data objects. Constrained Delaunay triangulation can be used for defining adjacency relation among connected or disconnected objects, conflict detection and displacements of spatial objects, and

finding the nearest neighboring object to a given object in generalization (Ware et al 1996, Chris B. Jones et al 1998, Wanning Peng 1997 etc). The constrained Delaunay triangulation can also be used to measure spatial relations such as measuring disjoint relation, distance relation and direction relation. Delaunay triangulation can be seen as the geometric primitives of the simplicial data structure (Christopher B. Jones, Byndy G.L and Ware M. 1995). The two areas sharing a common boundary are neighbors in a traditional data model. Similarly, two line segments are adjacent if they have a common node, but do not intersect. The spatial analysis of discrete non-connected objects has been approached using distance concepts in the traditional vector and raster models. As a result of this, the conventional data models are not able to hold a spatial adjacency properly for discrete objects (Gold, 1989). Distance concepts are often suggested to overcome this problem. Non-connected objects lying within a certain distance are regarded as neighbors. This distance could be measured through the support of constrained Delaunay triangulation.

For categorical database generalization, constrained Delaunay triangulation is very useful to analyze and measure local spatial relationship but not to organize the whole data set since a simple area object will consist of a lot of triangles that will lead to redundant data too much and also difficulty with semantic analysis among objects.

5.3 Integrated and Extended Formal Data Schema

Although FDS supports a number of elementary topological relationships, it does not support the spatial adjacency relationship among objects that are disconnected from each other and inclusion relationships (inclusion can be found if an object has only one neighbor). Topological relationships among “disconnected objects” are important to support spatial analysis (Peng 1997) and geometric operations that involve these kinds of objects and relationships among the objects. In the real world, the concept of “adjacent” may also include the adjacency relationship between those area objects that are geometrically disconnected from each other, as well as the adjacency relationship between line objects, between point objects, and moreover, the adjacency relationship between objects of different geometric description types.

Apparently, the FDS needs to be extended in the sense of adjacency (Peng 1997) and inclusion relationships, which are particularly important in automated database generalization and also needs more ability to support spatial measurement and operations. These can be achieved by dynamically integrating FDS and CDT. This integration of FDS and CDT will result in extending the adjacency relationships and inclusion between geometric data types. CDT may be generated dynamically and locally at a certain step of a generalization process.

The Delaunay point adjacency relationship is the basis on which adjacency relationships concerning other geometric data types and feature types are defined. This is because points are the most primitive geometric components of any spatial object.

5.3.1 Integrated FDS and CDT

We combine the advantages of FDS and CDT into a data model which is a dynamic integration of FDS and CDT in the database generalization transformation process. The data model is shown in Figure 5.3. Figure 5.4 shows the logical structure of a geo-database organized in a database based on a data model.

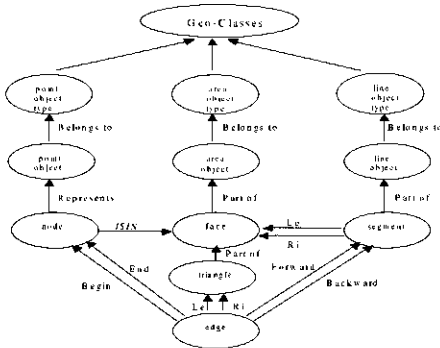


Figure 5.3 Data model for database generalization (modified Molenaar 1989)

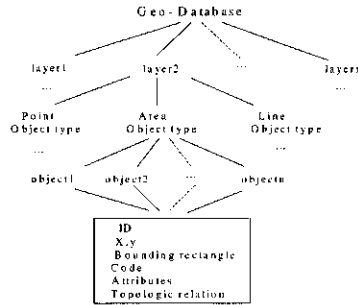


Figure 5.4 Logical structure of a geo-database

The key observation that both a constrained Delaunay triangulation network and object data can be represented as a planar graph forms the basis for the integration. The objects can be modeled as a triangulated irregular network, a planar graph with triangular faces, and the network can be modeled as a planar graph. Each network type will have its specific constraints that need to be described. We assume that the network shall be a valid planar graph, i.e. the edges are connected by nodes and have no self-intersecting curves. An important issue, which must be considered when integrating two structures, is the difference in resolution. A simple geometric measure is the average length of the FDS edges compared to the average length of the triangulation edges. If the average length of a FDS edge is shorter than one of the CDT edges, we say that the two structures are compatible. If the average length of FDS edge is longer than one of the CDT edges, FDS data will coarsen the CDT. A solution to the latter problem could be to interpolate points along the FDS edges to increase the resolution of the FDS data. Unlike the Delaunay triangulation of a set of points, the edges of a constrained Delaunay triangulation, in which constrained edges correspond to object boundary edges, do not always connect neighboring objects as shown in Figure 5.5. The figure shows that there is no direct edge connectivity between object o1 and its neighbor o2. In the real world, we consider object o1 and o2 to be neighbors. For a constrained Delaunay triangulation in which straight line constraining segments are similar in length to the separation distance between objects, the triangulation is such that the vertices of neighboring objects are usually connected by edges of the triangulation.

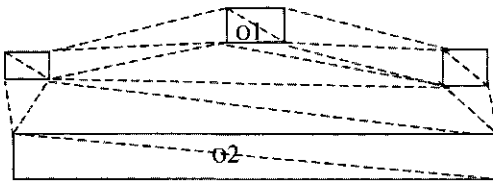


Figure 5.5 Part of a constrained Delaunay triangulation of a set of objects (solid lines as constrained edges, dashed lines as other edges)

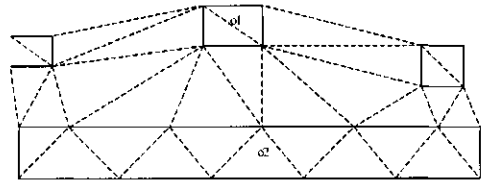


Figure 5.6 Part of a constrained Delaunay triangulation of a set of objects after interpolation (solid lines as constrained edges, dashed lines as other edges)

In order to reflect the neighboring relation between object o1 and o2, the constrained edges are interpolated and intermediate nodes are created along the edge as shown in Figure 5.6. The figure shows that there is direct edge connectivity between the two objects.

FDS data can be modeled as a planar graph with explicit topological and geometrical representation. The planar graph consists of the topological objects: *node*, *segment* and *face*. The *edge* is part of a *segment*. The *edge_i* is also part of a *segment* which meets triangulation requirements. It replaces *edge* when the FDS data is integrated with the CDT (see below).

The objects are modeled as a triangulation which consists of one or more triangles, where each *triangle* is composed of three *t-edges* and three *t-nodes* shared by other triangles. The integration method comprises two major steps:

The first step is to loop through all *edges* and convert them to *edge_i*s. For each *edge* the corresponding (line) segments of the curve are found. Each (line) segment is added to the TIN by inserting the start and end points as new nodes and then creating a constrained *t-edge* from the start to end node. The second step is to make sure the integrated model is consistent according to the network constraints. After integration, constraint checks and corrections, we have a FDS and CDT model that is the basis for creating a database generalization data structure. The process of integration of FDS and CDT is as follows:

Let *FdSedgeList* be FDS edge list with the data.

Let *TRledgeList* be empty list.

Let *LtsholdLength* be length threshold.

For each FDS edge $edge_{fds} \in FdSedgeList$, do the following:

- ```
{
 • check if length of $edge_{fds}$ is less than LtsholdLength, if yes, add $edge_{fds}$ to the list TRledgeList and move to the next edge in the FdSedgeList, otherwise do the following:
 {
```



- compute the coordinates of interpolating points;
  - form TRIedges based on interpolating result;
  - add TRIedges to the list *TRIEdgeList*.
- }
- until all edges in *FdSedgeList* have been checked.
- }

From the object point of view, based on this integration, we can divide the objects into three types. They are geometric objects, geographic objects and transformation units.

**Geographic objects** represent the initial geographic information with descriptive attributes.

**Geometric objects** (polygon, segments and nodes) describe the geometry of the terrain objects in order to allow shared geometry. They have topological relationships (e.g., description of inclusion and connectivity relationships (and not only connectivity)).

A **Transformation unit** is composed of a set of terrain objects. They are created in order to model specific analyzing, processing and reasoning necessary for generalization purposes.

### 5.3.2 Extended FDS

The FDS will be extended in three ways. One is extension based on geometric CDT, the second is extending adjacency and inclusion relations between different types of objects in FDS based on semantic CDT, the third is enhancing analysis function of FDS based on integration of FDS and CDT.

#### 5.3.2.1 Extending FDS Based on Geometric CDT

The following gives a list of notations to be used to define and describe the extended and integrated formal data structure. This section expresses relations defined by Molenaar (1990) for the case that all faces are triangles.

- Let  $N = \{ n_{ij} \}$  be a set of nodes,  $E = \{ e_{ij} \}$  be a set of edges within the framework of the FDS,  $E_c \subseteq E$  be a subset of constrained edge,  $E_{nc}$  be a subset of no-constrained edge  $\subseteq E$ ,  $T = \{ t_i \}$  be a set of triangles;
- Edge  $e_{ij}$  ( $i, j = 1, 2, 3$ ) of a triangle  $t_i$  has node  $n_{ij}$  as the begin node  $\rightarrow \text{Begin} [e_{ij}, n_{ij}] = 1$ , otherwise  $= 0$ , ( $i = 1, 2, \dots, n$ ) ( $j = 1, 2, 3$ ).
- Edge  $e_{ij}$  of a triangle  $t_i$  has node  $n_{ik}$  as the end node  $\rightarrow \text{End} [e_{ij}, n_{ik}] = 1$ ,

otherwise=0, ( $i=1,2,\dots,n$ ) ( $k=1,2,3$ ) ( $j \neq k$ ).

We will consider edges as edges of triangles. Each edge of a triangle has one triangle at its left side and one at its right side.

The following relationships can be defined between geometric elements of a planar graph based on the previous concepts of FDS and CDT:

A graph can be linked to a triangulation of the generic structure of Figure 5.7 (a). Edge  $\{n_{ij}, n_{ik}\}$  ( $j,k=1,2,3, j \neq k$ ) is one of three edges of a triangle. In fact we have the three edges of a triangle:

$$\begin{array}{ll} \text{Le}[\text{edge } \{n_{i1}, n_{i2}\}, t_i] = 1 & \longrightarrow & \text{Le}[e_{i1}, t_i] = 1 \\ \text{Ri}[\text{edge } \{n_{i1}, n_{i2}\}, t_{i1}] = 1 & & \text{Ri}[e_{i1}, t_{i1}] = 1 \quad \text{and} \\ \\ \text{Le}[\text{edge } \{n_{i2}, n_{i3}\}, t_i] = 1 & \longrightarrow & \text{Le}[e_{i2}, t_i] = 1 \\ \text{Ri}[\text{edge } \{n_{i2}, n_{i3}\}, t_{i2}] = 1 & & \text{Ri}[e_{i2}, t_{i2}] = 1 \quad \text{and} \\ \\ \text{Le}[\text{edge } \{n_{i3}, n_{i1}\}, t_i] = 1 & \longrightarrow & \text{Le}[e_{i3}, t_i] = 1 \\ \text{Ri}[\text{edge } \{n_{i3}, n_{i1}\}, t_{i3}] = 1 & & \text{Ri}[e_{i3}, t_{i3}] = 1 \end{array}$$

If an edge  $e_{ij}$  is part of the border of geometric object  $o_i$ , then  $t_i$  and  $t_{i1}$  or  $t_i$  and  $t_{i2}$  or  $t_i$  and  $t_{i3}$  must belong to different geometric objects respectively.

For any triangle  $t_i$  and its three adjacent triangles  $t_{ij}$  ( $j=1,2,3$ ), If  $t_i$  and  $t_{ij}$  ( $j=1,2,3$ ) belong to the same geometric object, then  $\text{Belong}[t_i, t_{ij}] = 0$ , otherwise  $\text{Belong}[t_i, t_{ij}] = 1$ .

Total number of edges of triangle  $t_i$  belonging to the geometric object is:

$$\text{Nedges}[t_i, o_i] = \sum_{j=1}^3 (\text{Belong}[t_i, t_{ij}])$$

If  $\text{Nedges}=1$ , there is one edge of the triangle  $t_i$  belonging to geometric object (see Figure 5.7 (c)).

If  $\text{Nedges}=2$ , there are two edges of the triangle  $t_i$  belonging to geometric object (see Figure 5.7 (b)).

If  $\text{Nedges}=3$ , there are three edges of the triangle  $t_i$  belonging to geometric object.

If  $Nedges=0$ , there is no edge of the triangle  $t_i$  belonging to geometric object (see Figure 5.7 (d)).

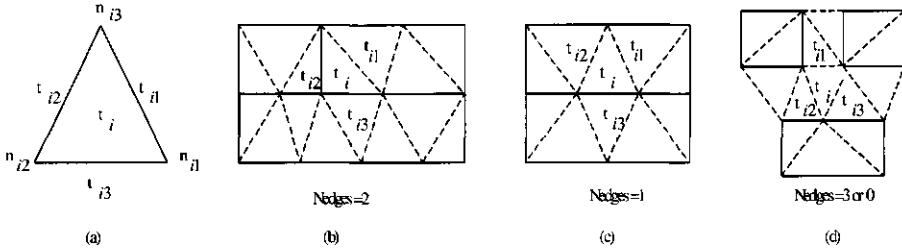


Figure 5.7 Relations among triangles and objects

• **Point Objects**

Any point object  $O_p$  is geometrically represented by a single node  $n_i \in N$ . it is expressed as:

If  $O_p$  is represented by  $n_i$  then  $Repr[n_i, O_p]=1$  otherwise  $Repr[n_i, O_p]=0$ .

• **Line Objects**

The geometry of a simple line object is represented by a set of edges of triangles. Each line object  $o_l$  is directed so that the relationship between edges of a triangle and the line object can be expressed as follows:

If an edge  $e_{ij}$  ( $j=1,2,3$ ) of a triangle has the same direction as the line object, there is a forward relationship

$$Forw[e_{i1}, o_l]=1, \text{ otherwise } Forw[e_{i1}, o_l]=0;$$

$$Forw[e_{i2}, o_l]=1, \text{ otherwise } Forw[e_{i2}, o_l]=0;$$

$$Forw[e_{i3}, o_l]=1, \text{ otherwise } Forw[e_{i3}, o_l]=0;$$

If an edge  $e_{ij}$  ( $j=1,2,3$ ) of a triangle has the opposite direction compared to the line object, there is a backward relationship

$$Back[e_{i1}, o_l]=1, \text{ otherwise } Back[e_{i1}, o_l]=0;$$

Back[e<sub>i2</sub>, o<sub>i</sub>]=1, otherwise Back[e<sub>i2</sub>, o<sub>i</sub>]=0;

Back[e<sub>i3</sub>, o<sub>i</sub>]=1, otherwise Back[e<sub>i3</sub>, o<sub>i</sub>]=0;

The fact that an edge e<sub>ij</sub> is part of the object can be established by the function:

$$\text{Part}_{11} [ t_i, o_i ] = \sum_{j=1}^3 \text{Max}(\text{Forw}[e_{ij}, o_i], \text{Back}[e_{ij}, o_i])$$

If Part<sub>11</sub> [ t<sub>i</sub>, o<sub>i</sub> ]=0, there is no edge of the triangle t<sub>i</sub> belonging to object o<sub>i</sub> .

If Part<sub>11</sub> [ t<sub>i</sub>, o<sub>i</sub> ]=1, there is one edge of the triangle t<sub>i</sub> belonging to object o<sub>i</sub> .

If Part<sub>11</sub> [ t<sub>i</sub>, o<sub>i</sub> ]=2, there are two edges of the triangle t<sub>i</sub> belonging to object o<sub>i</sub> .

If Part<sub>11</sub> [ t<sub>i</sub>, o<sub>i</sub> ]=3, there are three edges of the triangle t<sub>i</sub> belonging to object o<sub>i</sub> .

A line object will have a begin node n<sub>b</sub>=BEG(o<sub>i</sub>) and an end node n<sub>e</sub>=END(o<sub>i</sub>). These can be found through the edges of o<sub>i</sub>, and the direction of the object can then be specified by

$$\text{Dir}[o_i] = \{ n_b, n_e \}.$$

The total number of edges belonging to the line object is:

$$\text{Nledges}[o_i] = \sum_{i=1}^N (\text{Part}_{11} [ t_i, o_i ])$$

- **Area Objects**

The geometry of a simple area object is represented by one or more adjacent triangles, if a triangle t<sub>i</sub> is a part of an area object o<sub>i</sub> this will be represented by:

$$\text{Part}_{22} [ t_i, o_i ] = 1$$

Now it is possible to check whether edge e<sub>ij</sub> (j=1,2,3) of triangle t<sub>i</sub> is related through triangles t<sub>ij</sub> (j=1,2,3) to an area object o<sub>i</sub>. Therefore the following function should be evaluated:

$$Le[e_{i1}, o_i | t_i] = \text{MIN}(Le[e_{i1}, t_i], \text{Part}_{22}[t_i, o_i])$$

$$Le[e_{i2}, o_i | t_i] = \text{MIN}(Le[e_{i2}, t_i], \text{Part}_{22}[t_i, o_i])$$

$$Le[e_{i3}, o_i | t_i] = \text{MIN}(Le[e_{i3}, t_i], \text{Part}_{22}[t_i, o_i])$$

Each of three functions takes the value 1 if both functions at its right side have a value of 1, i.e., the edge should have the triangle at its left side, and the triangle should be part of the area object. In all other cases, the function at the right side of the equation takes the value of 0. Whether or not the edge is related to the object can then be found through

$$Le[e_{i1}, o_i] = \text{MAX}(Le[e_{i1}, o_i | t_i])$$

$$Le[e_{i2}, o_i] = \text{MAX}(Le[e_{i2}, o_i | t_i])$$

$$Le[e_{i3}, o_i] = \text{MAX}(Le[e_{i3}, o_i | t_i])$$

There is, at most, one triangle for which both

$$Le[e_{i1}, t_i] = 1 \text{ and } \text{Part}_{22}[t_i, o_i] = 1$$

$$Le[e_{i2}, t_i] = 1 \text{ and } \text{Part}_{22}[t_i, o_i] = 1$$

$$Le[e_{i3}, t_i] = 1 \text{ and } \text{Part}_{22}[t_i, o_i] = 1$$

If such a triangle exists, the function relating the edge to the object will have the value 1; in all other cases it will be 0; Hence, if edge  $e_{ij}$ , has area object  $o_i$  at its left side, then  $Le[e_{ij}, o_i] = 1$ . similarly, we can write

$$Ri[e_{i1}, o_i | t_{i1}] = \text{MIN}(Ri[e_{i1}, t_{i1}], \text{Part}_{22}[t_{i1}, o_i])$$

$$Ri[e_{i2}, o_i | t_{i2}] = \text{MIN}(Ri[e_{i2}, t_{i2}], \text{Part}_{22}[t_{i2}, o_i])$$

$$Ri[e_{i3}, o_i | t_{i3}] = \text{MIN}(Ri[e_{i3}, t_{i3}], \text{Part}_{22}[t_{i3}, o_i])$$

And

$$Ri[e_{i1}, o_i] = \text{MAX}(Ri[e_{i1}, o_i | t_{i1}])$$

$$Ri[e_{i2}, o_i] = \text{MAX}(Ri[e_{i2}, o_i | t_{i2}])$$

$$Ri[e_{i3}, o_i] = \text{MAX}(Ri[e_{i3}, o_i | t_{i3}])$$

If edge  $e_{ij}$ , has area object  $o_i$  at its right side, then  $Ri[e_{ij}, o_i]=1$ . otherwise the expression=0.

The transitions from indirect relationships between edges and area objects to direct relationships have been represented by many-- to -- one relationships and many-to-many relations. The relationships between triangles and area object are many - to - one, therefore, the derived relations between edges and triangles will be many-to-many. The combination of these two sets of functions gives, for edge  $e_{ij}$  ( $j=1,2,3$ )

$$B[e_{i1}, o_i] = Le[e_{i1}, o_i] + Ri[e_{i1}, o_i]$$

$$B[e_{i2}, o_i] = Le[e_{i2}, o_i] + Ri[e_{i2}, o_i]$$

$$B[e_{i3}, o_i] = Le[e_{i3}, o_i] + Ri[e_{i3}, o_i]$$

If an edge  $e_{ij}$  is part of the boundary of  $o_i$ , then only one of the functions  $Ri$  and  $Le$  is equal to 1, but not both; there fore, for such an edge we find  $B[e_{ij}, o_i] = 1$ . If  $e_{ij}$  has  $o_i$  at both its left and right sides, then  $B[e_{ij}, o_i]=2$ . In that case it runs through  $o_i$ . if  $B[e_{ij}, o_i]=0$ , there is no direct relationship between  $e_{ij}$  and  $o_i$ .

The boundary of  $o_i$  is:

$$\partial t_i = \{N_i, E_i\}, \text{ with } E_i = \{e_{ij} | B[e_{ij}, o_i] = 1\}.$$

- **Adjacent Area Object**

Based on the definition of adjacency relations using FDS (Molenaar, 1990), adjacent area object can be obtained according to CDT. When an edge  $e_{ij}$  ( $j=1,2,3$ ) of a triangle has an object  $o_i$  at its left side and not at its right side, and object  $o_j$  at its right side and not at its left side, these objects are adjacent at that edge  $e_{ij}$  ( $j=1,2,3$ ), i.e.,

$$\text{If } Le[e_{ij}, o_i] = 1 \text{ and } Ri[e_{ij} (j=1,2,3), o_i] = 0 \quad (j=1,2,3)$$

$$\text{And } Le[e_{ij}, o_j] = 0 \text{ and } Ri[e_{ij} (j=1,2,3), o_j] = 1 \quad (j=1,2,3)$$

$$\text{Then } \text{ADJACENT}[o_i, o_j | e_{ij}] = 1$$

This function is considered to be symmetrical, so that

$$\text{ADJACENT}[o_i, o_j | e_{ij}] = \text{ADJACENT}[o_j, o_i | e_{ij}]$$

If the objects do not overlap at all, i.e., if they have no common triangles and they are adjacent to at least one edge, then they are adjacent

$$\text{ADJACENT}[o_i, o_j] = 1$$

This function is also symmetrical, so that

$$\text{ADJACENT}[o_i, o_j] = \text{ADJACENT}[o_j, o_i].$$

- **Line And Area Objects**

Several important relationships between a line object  $o_l$  and an area object  $o_a$  can be found by checking for each edge  $e_{ij}$  of a triangle that is part of the line object, and the way in which it is related to the area object. That will be expressed by the functions:

$$\text{Le}[o_l, o_a | e_{ij}] = \text{MIN}(\text{Le}[e_{ij}, o_a], \text{Part}_{11}[e_{ij}, o_l])$$

$$\text{Ri}[o_l, o_a | e_{ij}] = \text{MIN}(\text{Ri}[e_{ij}, o_a], \text{Part}_{11}[e_{ij}, o_l])$$

For the relationship between a line object  $o_l$  and an area object  $o_a$ , we can write

$$\text{B}[o_l, o_a | e_{ij}] = \text{Le}[o_l, o_a | e_{ij}] + \text{Ri}[o_l, o_a | e_{ij}]$$

This function can also be expressed by

$$\text{B}[o_l, o_a | e_{ij}] = \text{B}[e_{ij}, o_a] \times \text{Part}_{11}[e_{ij}, o_l]$$

If this function has the value 2, then the line object runs through the area object at edge  $e_{ij}$ ; if the value =1, then it is at the border, and if it is 0, then the relationship between the two objects might be different at different edges.

### 5.3.2.2 Extending Adjacent and Inclusion Relations in FDS Based on Semantic CDT

- **Different roles of Triangles**

Assume that there are three objects O1, O2 and O3 as shown in Figure 5.8, In a constrained

Delaunay triangulation  $T$ , every boundary vertex of each object  $o_i$  corresponds to a triangle vertex and every boundary edge of each object  $o_i$  serves as a constraining edge. Each object  $o_i$  is defined by a unique object identifier and references to the triangles of  $T$  which lie within its boundary; each triangle  $t_i$  of  $T$  is described by a unique triangle identifier, references to each of its three constituent edges, plus a reference to the object within which it lies; each edge  $e_i$  is described by a unique edge identifier and references its start and end vertices; and each vertex  $v_i$  stores a unique vertex identifier plus  $x$  and  $y$  co-ordinate values. The supplementary topological information in the form of reference is added to the two triangles to which the edge (of triangle) belongs. If a triangle  $t_i$  lies within an object  $o_i$  then  $t_i$  is said to belong to  $o_i$  (Part<sub>22</sub> [  $t_i, o_i$  ] =1). Such information has been modeled in the following way as shown in Figure 5.8:

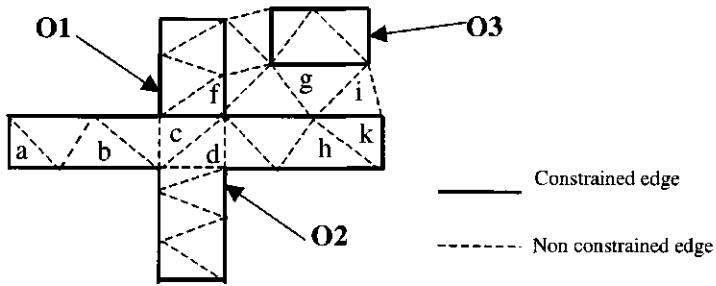


Figure 5.8 Structure of Constrained Delaunay Triangulation

Each triangle has the following properties for area objects:

- Its nodes must be on the boundary of objects.
- The three edges of a triangle of CDT are divided into two groups:
  - edges which are the part of the boundary of an object (named constrained edges);
  - edges which are not the part of the boundary of an object.

The triangles in CDT can be classified as four types through observation:

-



- Triangle having no constrained edge in its three edges, seen in triangle d in Figure 5.8, denoted as T1;
- Triangle having only one constrained edge in its three edges, seen in triangle b, g, i, h in Figure 5.8, denoted as T2;
- Triangle having two constrained edges in its three edges, seen in triangle a, f, k in Figure 5.8, denoted as T3;
- Triangle having three constrained edges in its three edges.

These types of triangles can be further subdivided as following:

- T1 can be subdivided into five groups. They are the three points of a triangle belonging to three different point objects, to three different line objects, to three different area objects, to one line object and two area objects, and to one area object and two line objects. (see Figure 5.9 (a), (b), (c), (d), (e)).

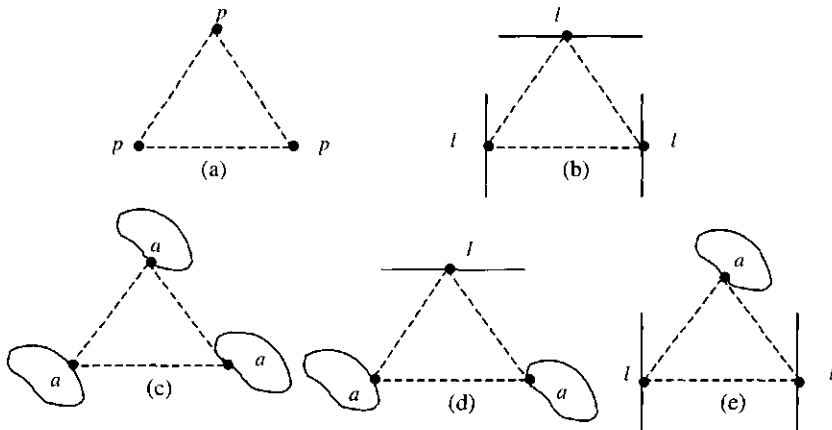


Figure 5.9 Examples of T1 (p=point, l=line, a=area)

- T2 can be subdivided into six groups according to the constituents of point, line, area objects in a triangle. (see Figure 5.10 (a), (b), (c), (d), (e) and (f))

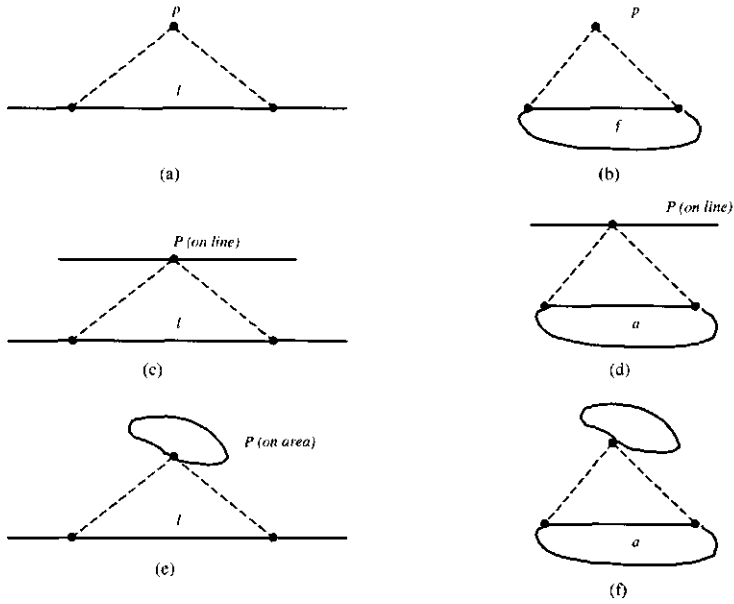


Figure 5.10 Examples of T2 (p=point, l=line and a=area)

- T3 can be subdivided into three groups. One is the triangles with two line constrained edges, and another is the triangles with two area object constrained edges and the third with one line object constrained edge (see Figure 5.11 (a), (b), (c), (d) and (e)).

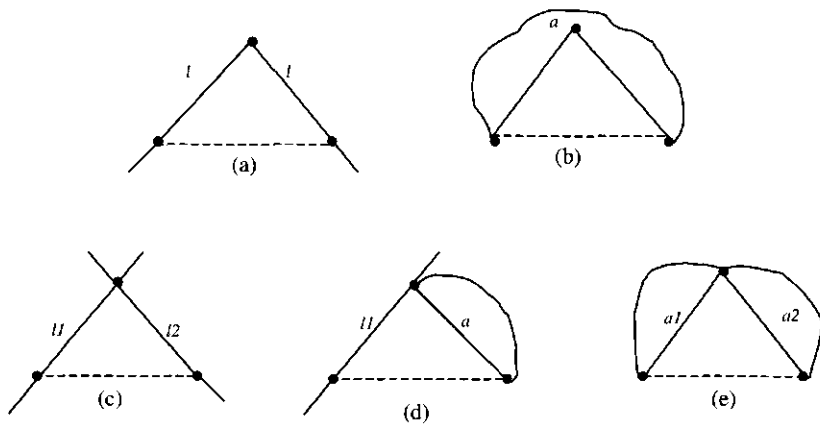


Figure 5.11 Examples of T3 (l=line and a=area)

• **Adjacent Relations Between Point, Line and Area Objects**

The adjacency relations among discrete objects play an important role in aggregation of objects in database transformation. The adjacent relationship can be grouped into four types in a triangle net:

- If two objects  $o_i$  and  $o_j$  are constrained edges (part of) of two triangles respectively and the two triangles share a common node, the two objects are adjacent, (see Figure 5.14 (b) and (d) ( point adjacent));
- If two objects,  $o_i$  and  $o_j$  share a connecting constrained edge of triangle, they are adjacent, seen in triangle f and c in Figure 5.8 (line adjacent);
- If two objects,  $o_i$  and  $o_j$  share a connecting triangle between them, they are adjacent, seen in object o3 and 01 in Figure5.8 (line adjacent); or
- If two objects  $o_i$  and  $o_j$  are constrained edges (part of) of two triangles respectively and the two triangles share common non-constrained edge, the two objects are adjacent, seen in triangle g and i in Figure 5.8 (area adjacent);
- If two objects  $o_i$  and  $o_j$  are constrained edges of a triangle, they are adjacent.

The triangle is used as a basic unit to analyze the geometric characteristics of an object and the adjacent and inclusion relations among the objects in this research.

The adjacent relationships among different types of discrete objects are described based on classes of triangles as following:

• **Adjacent Relationship between Points (Adjacent ( $p_i, p_j$ ))**

Let  $T=T1 \cup T2 \cup T3$

- Two node  $n_i \in N$  and  $n_j \in N$  are adjacent if they belong to a triangle  $t_i \in T$  (see Figure 5.12 );

$$\text{IF } (t_i \in T \wedge n_i \in t_i \wedge n_j \in t_i \wedge \text{Repr}[n_i, p_i] \wedge \text{Repr}[n_j, p_j]),$$

$$\text{then } e=\{ n_i, n_j \} \wedge e \in T$$

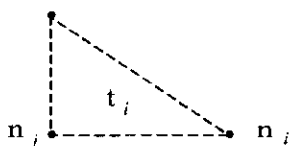
$$\text{and then } \text{Adjacent}(n_i, n_j)=\text{Adjacent}(n_j, n_i)=1.$$

• **Adjacent Relationship between Line and Point (Adjacent ( $p_i, l_i$ ))**

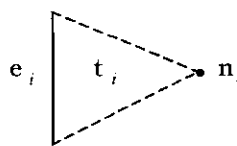
- A node  $n_i \in N$  and edge  $e_{ci} \in E$  are adjacent if they belong to a triangle  $t_i \in T2$  (see Figure 5.13);

IF ( $t_i \in T2 \wedge n_i \in t_i \wedge e_{ci} \in t_i \wedge n_i \notin e_{ci} \wedge \text{Repr}[n_i, p_i] \wedge \text{Part}_{11}[e_{ci}, l_i]$ )

and then **Adjacent ( $p_i, l_i$ ) = Adjacent ( $l_i, p_i$ ) = 1.**



**Figure 5.12** Adjacent relationship between nodes



**Figure 5.13** Adjacent relationship between node and line

• **Adjacent Relationship between Lines (Adjacent ( $l_i, l_j$ ))**

- Two objects  $l_i$  and  $l_j$  are adjacent if there exist two triangles  $t_i$  and  $t_j$  ( $t_i \in T2, t_j \in T2$ ) in which edge  $l_i$  and  $l_j$  are used as the constrained edge of two triangles respectively, and If the two triangles share a common non constrained edge (see triangle  $t_i$  and  $t_j$  in Figure 5.14 (a) and (c)) or if the two triangles share a common node (see triangle  $t_i$  and  $t_j$  in Figure 5.14 (b) and (d)) contrary to what Peng states (1997).

IF ( $(t_i \in T2 \wedge e_i \in t_i \wedge (\text{Part}_{11}[t_i, l_i] \neq 0) \wedge n_i \in t_i \wedge n_i \notin e_i \wedge t_j \in T2 \wedge e_j \in t_j \wedge (\text{Part}_{11}[t_j, l_j] \neq 0) \wedge n_j \in t_j \wedge n_j \notin e_j) \wedge (e_k \in t_i \wedge e_k \in t_j \vee n_k \in t_i \wedge n_k \in t_j)$ )

Then **Adjacent ( $l_i, l_j$ ) = Adjacent ( $l_j, l_i$ ) = 1.**

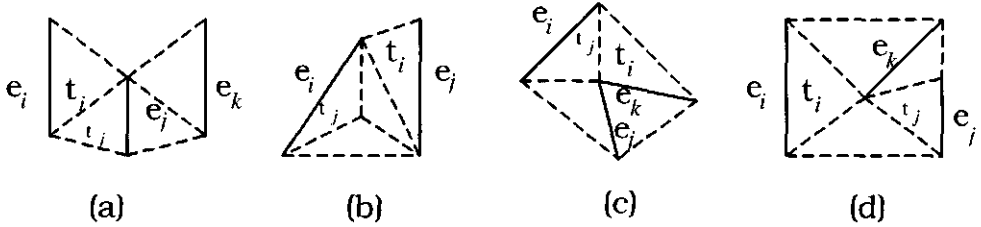


Figure 5.14 Adjacent relation between lines

• **Adjacent Relationship between Areas (Adjacent ( $f_i, f_j$ ))**

- Two areas  $f_i \in F$  and  $f_j \in F$  are adjacent if there exists at least one triangle  $t_i$  ( $t_i \in T1$ ) between  $f_i$  and  $f_j$  (see Figure 5.15).  $e_{ci}$  and  $e_{cj}$  are constrained edge.

$$\text{IF } (((t_i \in T2 \wedge t_i \in (\text{Part}_{22} [t_i, f_i]=0) \wedge t_i \in (\text{Part}_{22} [t_i, f_j]=0)) \wedge (e_{ci} \in t_i \wedge B(e_{ci}, f_i) \wedge n_i \in t_i \wedge n_i \notin e_{ci} \wedge n_i \in e_{cj} \wedge B[e_{cj}, f_j] \wedge e_{cj} \notin t_i)) \vee ((t_i \in T2 \wedge t_i \in (\text{Part}_{22} [t_i, f_i]=0) \wedge t_i \in (\text{Part}_{22} [t_i, f_j]=0)) \wedge (e_{cj} \in t_i \wedge (B(e_{cj}, f_j)=1) \wedge n_i \in t_i \wedge n_i \notin e_{cj} \wedge n_i \in e_{ci} \wedge (B[e_{ci}, f_i]=1) \wedge e_{ci} \notin t_i)))$$

$$\text{Adjacent}(f_i, f_j) = \text{Adjacent}(f_j, f_i) = 1.$$

• **Adjacent Relationship between Line and Area (Adjacent ( $l_i, f_j$ ))**

- A line  $l_i$  and an area  $f_j$  are adjacent if there exists at least one triangle  $t_i$  ( $t_i \in T2$ ) between  $l_i$  and  $f_j$  (see Figure 5.16).

$$\text{IF } ((t_i \in T2 \wedge (\text{Part}_{22} [t_i, f_j]=0) \wedge e_{ci} \in t_i \wedge (\text{Part}_{11} [t_i, l_i] \neq 0) \wedge n_i \in t_i \wedge n_i \notin e_{ci} \wedge n_i \in e_{cj} \wedge e_{cj} \notin t_i \wedge (B[e_{cj}, f_j]=1)) \vee (t_i \in T2 \wedge (\text{Part}_{22} [t_i, f_j]=0) \wedge e_{cj} \in t_i \wedge (B[e_{cj}, f_j]=1) \wedge n_i \in t_i \wedge n_i \notin e_{cj} \wedge n_i \in e_{ci} \wedge$$

$$e_{ci} \notin t_i \wedge (\text{Part}_{11}[t_i, l_i] \neq 0))$$

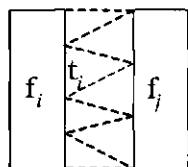
Then  $\text{Adjacent}(l_i, f_j) = \text{Adjacent}(f_j, l_i) = 1$

• **Adjacent Relationship between Point and Area (Adjacent  $(p_i, f_i)$ )**

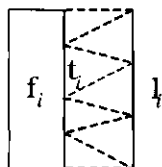
- A node  $n_i$  and a face  $f_i$  are adjacent if there exists at least one triangle  $t_i$  ( $t_i \in T2$ ) between  $n_i$  and  $f_i$  (see Figure 5.17).

$$\text{IF } (t_i \in T1 \wedge n_i \in t_i \wedge e_{ci} \in t_i \wedge n_i \notin e_{ci} \wedge \text{Repr}[n_i, p_i] \wedge (B[e_{cj}, f_j] = 1))$$

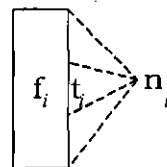
Then  $\text{Adjacent}(p_i, f_i) = \text{Adjacent}(f_i, p_i) = 1.$



**Figure 5.15** Adjacent relation between areas



**Figure 5.16** Adjacent relation between area and line



**Figure 5.17** Adjacent relation between area and node

• **Inclusion Relationships among Point, Line and Area Object**

• **Inclusion Relationship ---Area containing node (Inclusion  $(f_i, p_i)$ )**

- an area  $f_i$  contains a node  $n_i$  if there exists at least one triangle  $t_i$  ( $t_i \in T2$ ) between  $n_i$  and  $f_i$  and  $t_i$  is part of the area (see Figure 5.18).

$$\text{IF } (t_i \in T2 \wedge n_i \in t_i \wedge e_{ci} \in t_i \wedge n_i \notin e_{ci} \wedge \text{Repr}[n_i, p_i] \wedge (\text{Part}_{22}[t_i, f_j] = 1))$$

Then  $\text{Inclusion}(f_i, p_i) = 1.$

• **Inclusion Relationship ---Area Containing Line (Inclusion  $(f_i, l_i)$ )**

- a face  $f_i$  contains a line  $l_i$  if there exist at least one triangle  $t_i$  ( $t_i \in T2$ ) between  $l_i$  and  $f_i$  and  $t_i$  is part of the area (see Figure 5.19).

$$\text{IF } ((t_i \in T2 \wedge e_{ci} \in t_i \wedge (\text{Part}_{11}[t_i, l_i] \neq 0) \wedge n_i \in t_i \wedge n_i \notin e_{ci} \wedge n_i \in e_{cj} \wedge e_{cj} \notin t_i \wedge (\text{B}[e_{cj}, f_i]=1) \wedge (\text{Part}_{22}[t_i, f_i]=1)) \vee (t_i \in T2 \wedge e_{cj} \in t_i \wedge (\text{B}[e_{cj}, f_i]=1) \wedge n_i \in t_i \wedge n_i \notin e_{cj} \wedge n_i \in e_{ci} \wedge e_{ci} \notin t_i \wedge (\text{Part}_{11}[t_i, l_i] \neq 0))$$

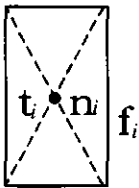
Then **Inclusion**  $(f_i, l_i)=1$ .

• **Inclusion Relationship ---Area Containing Area (Inclusion  $(f_i, f_j)$ )**

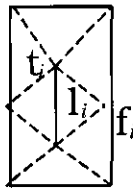
- An area  $f_i$  contains another area  $f_j$  if there exists at least one triangle  $t_i$  ( $t_i \in T2$ ) between  $f_i$  and  $f_j$  and  $t_i$  is part of the area  $f_i$  (see Figure 5.20).

$$\text{IF } ((t_i \in T2 \wedge e_{ci} \in t_i \wedge (\text{B}(e_{ci}, f_i)=1) \wedge (\text{Part}_{22}[t_i, f_i]=1) \wedge n_i \in t_i \wedge n_i \notin e_{ci} \wedge n_i \in e_{cj} \wedge e_{cj} \notin t_i \wedge (\text{B}[e_{cj}, f_j]=1)) \vee (t_j \in T2 \wedge e_{cj} \in t_j \wedge (\text{B}(e_{cj}, f_j)=1) \wedge (\text{Part}_{22}[t_j, f_j]=0) \wedge n_i \in t_j \wedge n_i \notin e_{ci} \wedge n_i \in e_{cj} \wedge e_{cj} \notin t_j \wedge (\text{B}[e_{cj}, f_i]=1)))$$

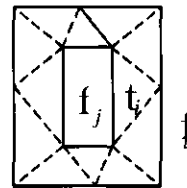
Then **Inclusion**  $(f_i, f_j)=1$ .



**Figure 5.18** Inclusion relation between area and node



**Figure 5.19** Inclusion relation between area and line



**Figure 5.20** Inclusion relation between area and area

### 5.2.3.3 Extending Analysis Function In FDS

- Nearness Degree

According to the concept of topological distance and neighborhood discussed in Section 2.6.2.3 of this thesis, the distance between two objects can be defined as the nearness degree. This distance can be measured by the triangles in constrained Delaunay triangulation. The shorter the distance between two objects is, the nearness degree between them will be stronger. Nearness degree between two objects is expressed by the minimum length of non-constrained edges of a set of triangles of T1 between two objects or average length of all non constrained edges of a set of triangles of T1 between two objects.

$$Ad = \text{minimum} \{ l_i \} \quad \text{where: } l_i \text{ is the length of non-constrained edges}$$

Or

$$Ad = \frac{\sum_i^n l_i}{n} \quad \text{where: } n \text{ is number of triangles which belong to T1.}$$

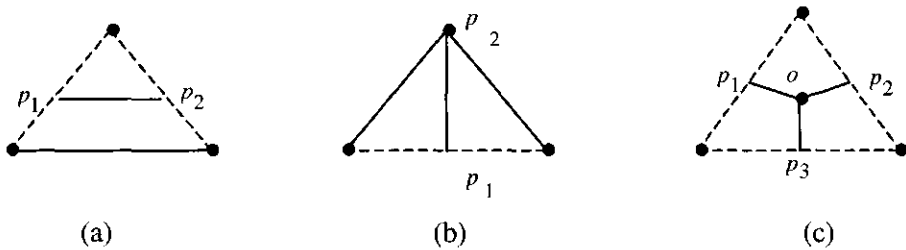
- Skeleton Line

The skeleton can be used for the 1-dimension analogue for an area object and is also useful during a merging operation in database transformation (Jones, C.B. 1995, Tinghua, Ai et al, 2000, 2001).

Figure 5.21 shows the process of extracting the skeleton.  $P_1$ ,  $P_2$  and  $P_3$  represent the midpoint of corresponding triangle edges respectively, and  $O$  is the barycenter of the triangle. There are three different types of linking methods. The process of skeleton generation is described by:

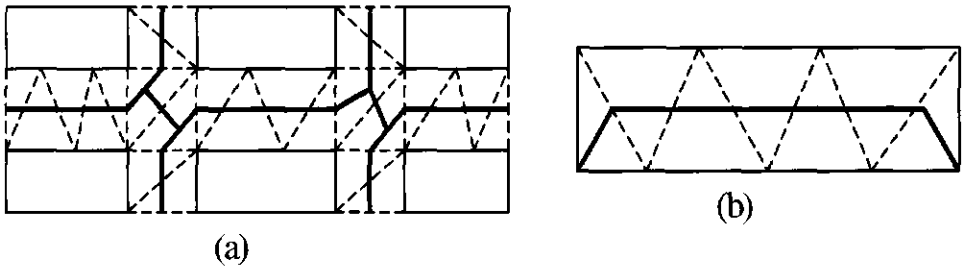
- If ( $t_i \in T2$ )      linking midpoints of two non-constrained edges of a triangle as shown in Figure 5.21(a).
- Elseif ( $t_i \in T3$ )    linking midpoint of non constrained edge of triangle with Opposition point of non constrained edge as shown in Figure 5. 21(b).
- Else ( $t_i \in T1$ )    linking the midpoint of each edge of a triangle with the barycenter of the triangle as shown in Figure 5.21 (c).





**Figure 5.21** Example of skeleton line and linking methods in three different types of triangles

A skeleton line can be extracted through connecting three different types of triangles described before as shown in Figure 5.22 (a), (b).



**Figure 5.22** Examples of skeleton line (dark line =Skeleton line)

#### 5.4 Query Based on IEFDS

The main purpose of querying a point object, a line object and an area object is to find all the adjacent geometric objects of the point object, the line object and the area object respectively, including neighboring point objects, neighboring line objects and neighboring area objects based on analyzing the types of a set of triangles described before.

Let pointobject, lineobject and areaobject be a point object, a line object and an area object respectively. Let pointobjectlist, lineobjectlist and areaobjectlist be a set of point objects, a set of line objects and a set of area objects respectively. Let tritanglelist be the list of triangles.

- **Point Object**

Find all the adjacent geometric objects of a point object including neighboring point objects, neighboring line objects and neighboring area objects based on analyzing the types of a set of

triangles and store them in `pointneighborpointobjectlist`, `pointneighborlineobjectlist` and `pointneighborareaobjectlist` for point objects, line objects and area objects respectively.

Let `pointneighborpointobjectlist`, `pointneighborlineobjectlist` and `pointneighborareaobjectlist` be empty respectively

Get a set of triangles through the `pointobject` and store them into `trianglelist`.

For each  $\text{triangle} \in \text{trianglelist}$ , do the following:

```

{
 tri=trianglelist.Get //get triangle from the list

 if tri==tri1 \wedge pointobject==one of three points of tri //relation between points
 {
 triPlist.Add(tri)
 for each tri \in triPlist, do the following
 {
 pointneighborobject=get_object(tri) //get a neighboring point object
 if pointneighborobject \notin pointneighborpointobjectlist \wedge pointobject
 pointneighbourpointobjectlist.Add (pointneighborobject)
 endif
 }
 } //find all adjacent points

 elseif tri==tri2 \wedge pointobject==one of three points of tri \wedge pointobject \notin constrained edge
 of tri \wedge constrained edge of tri \in lineobjectlist //relation between point and line
 {
 triLlist.Add(tri)
 for each tri \in triLlist, do the following
 {
 lineneighbourobject=get-object (tri) //get a neighboring line object
 if lineneighborobject \notin pointneighborlineobjectlist
 pointneighborlineobjectlist.Add(lineneighborobject)
 endif
 }
 }

 elseif tri==tri2 \wedge pointobject== the point of tri \wedge pointobject \notin constrained edge
 of tri \wedge constrained edge of tri \in Areaobjectlist //relations between point and area
 {
 triAlist.Add(tri)
 for each tri \in triAlist
 }
}

```

```

{
 areaneighborobject=get_object(tri) //get a neighboring area object
 if areaneighborobject ∉ pointneighborareaobjectlist
 pointneighborareaobjectlist.Add (areaneighborobject)
 endif
}
 if all tri having same attribute
 pointobject is inside areaobject
 else
 pointobject is outside areaobject
 endif
endif
}
}

```

- **Line Object**

Find all the adjacent geometric objects of a line object through analyzing the types of a set of triangles including neighboring point objects, neighboring line objects and neighboring area objects and store them in pointneighborpointobjectlist, pointneighborlineobjectlist and pointneighborareaobjectlist for point objects, line objects and area objects respectively.

Let pointneighborpointobjectlist, pointneighborlineobjectlist and pointneighborareaobjectlist be empty respectively

Get a set of triangles through the lineobject and store them into trianglelist.

For each triangle  $\in$  trianglelist, do the following:

```

{
 tri=trianglelist.Get //get triangle from the list

 if tri==tri2 ∧ lineobject==constrained edge of tri ∧ one point of tri ∉ constrained edge
 of tri ∧ one point of tri ∈ pointobjectlist
 //relation between line and point
 {
 triPlist.Add(tri)
 for each tri ∈ triPlist, do the following
 {
 pointneighborobject=get_object(tri) //get a neighboring point object
 if pointneighborobject ∉ lineneighborpointobjectlist
 lineneighborpointobjectlist.Add (pointneighborobject)
 endif
 }
 //find all adjacent points
 }
}

```

```

elseif tri==tri2 \wedge one of three points of tri \in lineobject \wedge the other two points of
 tri \in different line objects //relation between line and line
{
 triLlist.Add(tri)
 for each tri \in triLlist, do the following
 {
 lineighbourobject=get-object (tri) //get a neighboring line object
 if lineighborobject \notin lineighborlineobjectlist \wedge lineobject
 lineighborlineobjectlist.Add(lineighborobject)
 //find adjacent line objects
 endif
 }
 areaobject not intersecting with other line objects
}

elseif tri==tri2 \wedge (lineobject== constrained edge of tri \wedge one point of tri \notin constrained
 edge of tri \wedge one point of tri \in lineobjectlist or one point of tri \in lineobject \wedge
 lineobject \neq constrained edge of tri \wedge constrained edge of tri \in lineobjectlist)
 //relation between line and line
{
 triLlist.Add(tri)
 for each tri \in triLlist, do the following
 {
 if lineobject== constrained edge of tri
 lineighbourobject=get-object (tri) // get a neighboring line object
 else
 lineighbourobject=get-object (tri) // get a neighboring line object
 endif

 if lineighborobject \notin lineighborlineobjectlist
 lineighborlineobjectlist.Add(lineighborobject)
 endif
 }
 //find adjacent line objects
 lineobject not intersecting with other line objects
}

elseif tri==tri3 \wedge lineobject== one of two constrained edge of tri \wedge the other
 constrained edge of tri \neq lineobject \wedge constrained edge of tri \in lineobjectlist
 //relation between line and line
{
 triLlist.Add(tri)
 for each tri \in triLlist, do the following
 {
 lineighbourobject=get-object (tri) // get a neighboring line object
 }
}

```

```

 if lineneighborobject \notin (lineneighborlineobjectlist and lineobject)
 lineneighborlineobjectlist.Add(lineneighborobject) // get a neighboring line object
 endif
} //find adjacent line objects
lineobject intersecting with other line objects
}

elseif tri==tri2 \wedge (lineobject== constrained edge of tri \wedge one point of tri \notin constrained
edge of tri \wedge one point of tri \in areaobjectlist or one point of tri \in lineobject
 \wedge lineobject \neq constrained edge of tri \wedge constrained edge of tri \in areaobjectlist)
//relations between line and area
{
 triAlist.Add(tri)
 for each tri \in triAlist
 {
 if lineobject== constrained edge of tri2
 areaneighborobject=get_object(tri) // get a neighboring area object
 else
 areaneighborobject=get_object(tri) // get a neighboring area object
 endif
 if areaneighborobject \notin lineneighborareaobjectlist
 lineneighbourareaobjectlist.Add (areaneighborobject)
//find adjacent area objects
 endif
 }
 lineobject not intersecting with area objects
}

elseif tri==tri3 \wedge lineobject== one of two constrained edge of tri \wedge the other
constrained edge of tri \in areaobjectlistset
//relations between line and area
{
 triAlist.Add(tri)
 for each tri \in triAlist
 {
 areaneighborobject=get_object(tri) // get a neighboring area object
 if areaneighborobject \notin lineneighborareaobjectlist
 lineneighbourareaobjectlist.Add (areaneighborobject)
 endif // find adjacent area objects
 }
 lineobject intersecting with area objects
}
}

```

- **Area Object**

Find all the adjacent geometric objects of an area object through analyzing the types of a set of triangles including neighboring point objects, neighboring line objects and neighboring area objects and store them in `pointneighborpointobjectlist`, `pointneighborlineobjectlist` and `pointneighborareaobjectlist` for point objects, line objects and area objects respectively.

Let `pointneighborpointobjectlist`, `pointneighborlineobjectlist` and `pointneighborareaobjectlist` be empty respectively

Get a set of triangles through the `areaobject` and store them into `trianglelist`.

For each `triangle ∈ trianglelist`, do the following:

```
{
 tri=trianglelist.Get //get triangle from the list

 if tri==tri2 ∧ (areaobject==constrained edge of tri ∧ one point of tri ∉ constrained edge of
 tri ∧ one point of tri ∈ pointobjectlist)
 //relation between area and point
 {
 triPlist.Add(tri)
 for each tri ∈ triPlist, do the following
 {
 if areaobject==constrained edge of tri
 pointneighborobject=get_object(tri) // get a neighboring point object
 endif
 if pointneighborobject ∉ areaneighborpointobjectlist
 areaneighborpointobjectlist.Add (pointneighborobject)
 //find adjacent points
 endif
 }
 }

 elseif tri==tri2 ∧ areaobject== constrained edge of tri ∧ one point of
 tri ∈ lineobjectlist //relation between area and line
 {
 triLlist.Add(tri)
 for each tri ∈ triLlist, do the following
 {
 lineneighouobject=get-object (p) // get a neighboring line object
 if lineneighborobject∉ areaneighborlineobjectlist
 areaneighborlineobjectlist.Add(lineneighborobject)
 //find adjacent line objects
 endif
 }
 }
}
```

```

 areaobject not intersecting with other line objects
}

elseif tri==tri3 \wedge areaobject== one of two constrained edge of tri \wedge the other
 constrained edge of tri \in lineobjectlist
 //relation between area and line
{
 triLlist.Add(tri)
 for each tri \in triLlist, do the following
 {
 lineneighbourobject=get-object (tri) // get a neighboring line object
 if lineneighbourobject \notin areaneighborlineobjectlist
 areaneighborlineobjectlist.Add(lineneighbourobject)
 endif //find adjacent line objects
 }
 lineobject intersecting with other line objects
}

elseif tri==tri1 \wedge one point of tri \in areaobject \wedge the other two points of tri \notin areaobject the
 other two points of tri \in different area objects //relation between area and area
{
 triLlist.Add(tri)
 for each tri \in triLlist, do the following
 {
 areaneighbourobject=get-object (tri) // get a neighboring area object
 if areaneighbourobject \notin (areaneighborlineobjectlist \wedge areaobject)
 areaneighborlineobjectlist.Add(lineneighbourobject)
 endif //find adjacent area objects
 }
 areaobject disconnecting with other area objects
}

elseif tri==tri2 \wedge (areaobject== constrained edge of tri \wedge one point of tri \in areaobjectlist
or areaobject \neq constrained edge of tri \wedge one point of tri \in areaobject \wedge constrained edge of
 tri2 (a) \in areaobjectlist) //relations between area and area
{
 triAlist.Add(tri)
 for each tri \in triAlist
 {
 if areaobject== constrained edge of tri2
 areaneighbourobject=get_object(tri) // get a neighboring line object
 else
 areaneighbourobject=get_object(tri) // get a neighboring line object
 endif
 }
}

```

```

 if areanighborobject \notin areanighborarealist
 areanighbourarealist.Add (areanighborobject)
 endif // find adjacent area objects
}
areanobject adjacent or disconnected with area objects
}

elseif tri==tri3 \wedge areanobject== one of two constrained edge of tri \wedge the other constrained
 edge of tri \in arealist //relations between area and area
{
 triAlist.Add(tri)
 for each tri \in triAlist
 {
 areanighborobject=get_object(tri) // get a neighboring area object
 if areanighborobject \notin areanighborarealist
 areanighbourarealist.Add (areanighborobject)
 endif //find adjacent area objects
 }
 areanobject adjacent with area objects
}
}

```

## 5.5 Summary

This chapter introduces the IEFDS, an integrated and extended version of FDS, as a data model to support automated database generalization transformation and discusses the process of integration between FDS and CDT in more detail. The classification of triangle is proposed based on constituent properties of triangles in the constrained Delaunay triangulation network which plays an important role in the extended adjacent and inclusion relations and extracting the skeleton line. The concept of adjacent degree is also given in this study. This chapter also provides some examples of spatial query operations that make use of the extended adjacent relation and semantic triangles. These adjacent relationships and semantic triangles are of particular interest in automated database generalization.

Note that the CDT is introduced to define the adjacent relations and inclusion relations, but it is not necessarily part of the data model. It may be generated dynamically, and locally, at a certain step of a generalization process.



## Chapter 6

# Auxiliary Analysis Methods

### 6.1 Introduction

Previous chapters focus on such issues as basic concepts, database transformation, transformation constraints and the supporting data structure of transformation. The descriptions of spatial objects, object types, hierarchical structure, categorical data and spatial and semantic relations are given in Chapter 2 and the aspects and contents of categorical database transformation are illustrated in Chapter 3. The constraints in categorical database transformation are proposed in Chapter 4. The data structure IEFD which supports database transformation is discussed in Chapter 5. Categorical database transformation not only needs supporting data structure and defined transformation constraints, but also the algorithms to implement transformation. After defining a new geo-spatial model associated with a target database based on classification hierarchy and aggregation, whether one object in a database is transformed into another object not only depends on its own geometric and thematic properties, but also the spatial and semantic relations to its neighbor objects. Aspects of analyzing algorithms needed to realize object transformation in a categorical database are detection of violated objects, their neighbor objects and creation of different types of transformation units. Evaluating the similarity among objects in the transformation unit is an analyzing and decision-making process of database transformation, and aggregation operations are an implementation process of database transformation. This chapter mainly discusses the algorithms which are designed according to the requirements of categorical database transformation as mentioned before. These auxiliary analysis methods include semantic similarity matrix based on classification hierarchy, computing model of similarity based on classification hierarchy, detection and creation of transformation unit based on constraints, spatial relations and IEFDS, area object aggregation analysis and process based on transformation unit, multi-neighborhood, object clustering and creation of catchment hierarchy. Among them, semantic similarity matrix is used to express the similarity among object types in a classification hierarchy. The creation of a transformation unit identifies a spatial cluster of objects to be transformed in a database in the aggregation process. Computing model of similarity provides a method to calculate the similarity value among objects and object types within a hierarchy and helps to find out the reasonable aggregating objects for constrained object in a transformation unit. In a sense aggregation operations based on different types of transformation units will implement categorical database transformation. These auxiliary analysis methods have been developed to solve a number of important geometric and thematic problems in database transformation.

### 6.2 Creating Transformation Units

The transformation unit proposed in Chapter 4 in this study is an important process unit as many generalization problems need to be solved by considering a subset of related objects as a

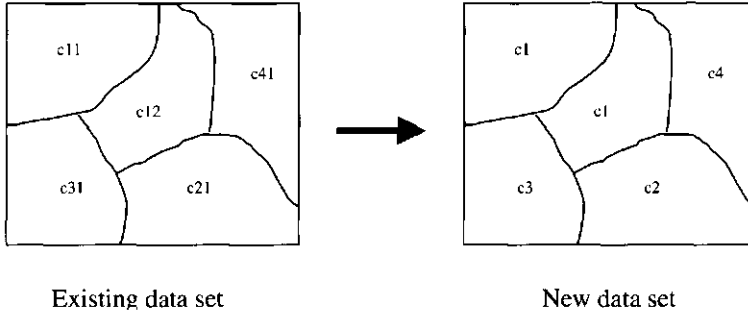
whole, rather than treating them individually. Few generalization problems can be solved by just looking at an individual object. In a sense, the transformation unit is a basic analysis, processing and decision-making unit in the aggregation operation process and plays an important role in database transformation. As discussed in Chapter 2 and Chapter 3, the transformations of classification hierarchy and aggregation hierarchy from an associated original database to a new database can be reached through simplifying the original hierarchical structure such as reducing the number of layers or number of object types or defining a new hierarchical structure (aggregation hierarchy) according to application requirements. The transformations will cause some conflicts in geometric and thematic aspects of objects and relations among objects. In order to solve these conflicts (violated constraints defined in Chapter 4) in a categorical database transformation, conflicted objects and its (their) related objects are organized into a transformation unit. A transformation unit that “brings together” a subset of objects can be created by conflicted object(s) and violated relations. The main purpose of creating transformation units is for the preparation of an aggregation operation. It restricts aggregation operations on a set of related objects at one time not on the whole database. This will make aggregation operation more efficient and effective. The different conflict types should have different types of transformation units. For categorical database transformation, four types of transformation units are identified based on constraints discussed in Chapter 4 and the characteristics of a categorical database in Chapter 2. Each type of transformation unit will store in a different list, each of which has a corresponding aggregation operation (see Section 6.4 for more detailed discussion).

The transformation units will trigger aggregation operations and can be created in different ways. On the one hand, a transformation unit can be built either bottom-up, i.e. by grouping objects (e.g. close buildings are grouped together to create a district), or top-down, i.e. by splitting a whole into parts (e.g. the districts are obtained by partitioning the town). On the other hand, the transformation unit can either be built a priori, in a stage of data enrichment prior to the generalization process (as in the case of the town and the districts), or be built dynamically during the generalization, when the need occurs (e.g. a transformation unit “group of area objects” can be created during the generalization of a small area object).

### 6.2.1 Creating Transformation Unit Based on Thematic Constraints

After classification hierarchy or aggregation hierarchy associated with an original database being changed into a new classification hierarchy or aggregation hierarchy with a target database, two or more objects of different object types in the original database may be changed into one object belonging to the same super object type or they have high similarity (see Section 6.3 for more detailed discussion) in the new database based on a new classification hierarchy. This means that there is a boundary between objects of same object type. This will result in the thematic conflict between objects. Figure 6.1 shows the thematic conflict object c1 after thematic transformation. Before an aggregation operation, thematic conflict of adjacent objects must be detected. Thematically conflicted objects with its neighbors will form a transformation unit of type TUI.

The main steps for creating transformation units based on thematic constraints include detecting thematically conflicted objects, analyzing these objects and their neighbors and creating a transformation unit.



**Figure 6.1** Example of thematic conflict of objects

The concrete procedure is as follows:

- Let *TheDatabase* be a database having a classification and aggregation structure;
- Let *ConflictObjectList* be an empty list;
- Let *TransformationUnitList* be an empty list;
- For each area object *CurrentObject* ∈ *TheDatabase*, do the following:
  - {
  - Check if *CurrentObject* has been included in any *ConflictObjectList* previously; this can be done by assigning a flag for each object. If the result is yes, or if the thematic properties are different from its adjacent objects, then move to the next area object in the same data set; otherwise do the following:
    - {
    - Use the query procedure described in Chapter 5 to get all the neighbors of the area objects *CurrentObject* and store them in a list *neighborslist*;
    - For each *neighbor* ∈ *neighborslist*, do the following:
      - {
      - If the thematic properties of the neighbor is different from the one of *CurrentObject*, or if it has been included in the *ConflictObjectList*, then move to the next neighbor in the list; otherwise move to the next step;
      - Stack *neighbor* into a stack *thestack*;

- ```

    }
    • Pop up an object theobject from thestack, store it in a list TempTransformationUnit within which objects have same thematic attributes or similarity but are adjacent to each other and the currentobjec=theobject, then repeat the above steps at this level until the stack is empty;

    • Store TempTransformationUnit into a list TransformationUnitList;
    }

    • Move to the next area object in the database and repeat all the steps at this level to detect other thematic conflicted objects and build transformation units.
}

```

6.2.2 Creating Transformation Unit Based on Geometric Constraints

A small area object (or a set of small area objects) with an area less than the area threshold will be aggregated with one or some of its neighborhood to form a new bigger object. The objects that violate geometric constraints and their spatial adjacent (connected) neighbors will form transformation units of type TU2 (see the examples in Section 4.4.4).

The main steps of creating a transformation unit based on geometric constraints includes detecting geometrically conflicted objects, analyzing the neighbor of each conflicted object and building a transformation unit based on analyzed results. The concrete procedure is given below:

- Let the *ConflictObjectList* and *TransformationUnitList* be an empty list;
- For each area object *CurrentObject*, do the following (detecting conflicted objects):


```

      {
      • Check if CurrentObject has been included in any detected object previously; this can be done by assigning a flag to each object. If the result is yes, or if the area of CurrentObject is larger than the threshold, then move to the next area object in the data set; otherwise do the following:
      {
      • Add CurrentObject to the list ConflictObjectList. Mark the object by setting its flag;

      • Use the query procedure described in Chapter 5 to get all the area object neighbors of the CurrentObject, and store them in a list NeighborsList where area objects are connected to each other;

      • For each neighbor  $\in$  NeighborsList, do the following:
      
```

- ```

{
 • If the area of neighbor is larger than the threshold, or if it has been included
 in ConflictObjectList , then move to the next neighbor in the list; otherwise
 move to the next step;

 • Push neighbor into a stack thestack;
}

• Pop up an object the object from thestack, and let CurrentObject = theobject, then
 repeat the above steps at this level until the stack is empty;
• The object contained in ConflictObjectList form conflicted objects within which
 objects are small but adjacent to each other;
}

• Move to the next area object in the data set and repeat all the steps at this level
 to detect other conflicted objects.
}

• For each conflicted object theConflictObject ∈ ConflictObjectList, do the following
 (building transformation unit):
{
 • Check if the ConflictObject has been included in TransformationUnitList, if the
 result is yes, then move to the next conflict object ConflictObjectList; otherwise do
 the following:
 {
 • Use the procedure described in Section 5.4 to get all the area object neighbors
 of the theConflictObject, and store the neighbors in neighborlist,

 • Store the ConflictObject and its neighbors in TransformationUnitList where
 area objects are connected to each other ;
 }

 • Repeat above steps until all conflicted objects in ConflictObjectList are checked
}

```

### 6.2.3 Creating Transformation Unit Based on Spatial Relation Constraints

If the distance between nearest neighbor objects or among a set of objects is less than the threshold, this set of objects will form a transformation unit of type TU3 (see the examples in Section 6.4.4.4 of this thesis). The objects in this type of transformation unit may belong to the same object type or may not.

The creation of this type of transformation unit based on spatial relations involves two main steps: first checking and analyzing the distance between two objects based on constrained Delanuy triangulation, then creating a transformation unit for an aggregation operation. The concrete process is as follows:

- Let the *ConflictObjectList* and *TransformationUnitList* be an empty list;
- For each area object *CurrentObject*, do the following:
  - {
  - Check if *CurrentObject* has been included in any *ConflictObjectList* and *TransformationUnitList* detected previously; If the result is yes, then move to the next area object in the data set; otherwise do the following:
    - {
    - Add *CurrentObject* to the list *ConflictObjectList*. Mark the object by setting its flag;
    - Use the query procedure described in Chapter 5 to get all the area object neighbors of the *CurrentObject*, and store them in a list *neighborslist*. Note that this problem is only applicable to the objects which are geometrically disconnected;
    - For each  $neighbor \in neighborslist$ , do the following:
      - {
      - If the *neighbor* and *CurrentObject* are not in conflict or the space between *neighbor* and *CurrentObject* is larger than the threshold, or if it has been included in *ConflictObjectList* and *TransformationUnitList*, then move to the next *neighbour* in the list; otherwise move to the next step;
      - Push *Neighbor* into a stack *thestack*;
      - }
    - Pop up an object *theobject* from *thestack*, store it in a list *TempTransformationUnitList* and let  $CurrentObject = theobject$ , then repeat the above steps at this level until the stack is empty;
    - Store *TempTransformationUnitList* in *TransformationUnitList* within which objects are in spatial conflict with each other;
    - }
  - Move to the next area object in the data set and repeat all the steps at this level to detect other conflicted objects.
  - }

### 6.2.4 Creating Transformation Unit Based on Geometric and Spatial Relation Constraints

If a small area object has an area less than the area threshold (maybe it locates inside another area object or maybe not) and the distance between it and its nearest neighbor object(s) is less than the distance threshold, then this set of objects that violates geometric constraints or spatial relation constraints will form a transformation unit of type TU4 (see the examples in Section 6.4.4.4). The objects in this type of transformation unit may be the same or may not.

The creation of this type of transformation unit follows the following steps: first checking the geometrically conflicted object, then analyzing the distance relation between it and its multi-neighborhood (see Section 6.5 of this chapter) based on the triangulation network, and finally creating a transformation unit based on geometric and spatial relations constraints. The concrete process is listed below:

- Let the *ConflictObjectList* and *TransformationUnitList* be an empty list;
- For each area object *CurrentObject*, do the following:
  - {
  - Check if *CurrentObject* has been included in any *ConflictObjectList* and *TransformationUnitList* detected previously; or area of *CurrentObject* is larger than the threshold, If the result is yes, then move to the next area object in the data set; otherwise do the following:
    - {
    - Add *CurrentObject* to the list *ConflictObjectList*. Mark the object by setting its flag;
    - Use the query procedure described in Chapter 5 to get all the area object neighbors of the *CurrentObject*, and store them in a list *neighborslist*; Note that the objects in the *neighborslist* consist of first order neighbors and second order neighbors of *CurrentObject*;
    - For each  $neighbor \in neighborslist$ , do the following:
      - {
      - If the area of *neighbor* is larger than the threshold, or if the distance between *CurrentObject* and  $neighbor \in neighborslist$  is not violated with the threshold or if it has been included in *ConflictObjectList* and *TransformationUnitList*, then move to the next neighbour in the list; otherwise move to the next step;
      - Push *Neighbor* into a stack *thestack*;
      - }
  - Pop up an object *theobject* from *thestack*, store *theobject* in a list

*TempTransformationUnitList* within which objects are small and conflict with each other and let *CurrentObject = theobject*, then repeat the above steps at this level until the stack is empty;

- Store *TempTransformationUnitList* in *TransformationUnitList*;
- }
  - Move to the next area object in the data set and repeat all the steps at this level to detect other conflict objects and build transformation units.
- }

### 6.3 Hierarchic Semantic Similarity

The similarity of objects and object types can be described by a similarity measure. The similarity is application-dependent. Classification hierarchy and aggregation hierarchy are ordered structures as discussed before. These hierarchies can reflect the similarity between object types both at the same level and at different levels. Whether two adjacent objects or a group of adjacent objects can be merged or aggregated depends on the attributes of the objects. If the attributes are the same or similar or the value of similarity is higher than the threshold, they can be merged or aggregated. Otherwise not. In a sense, the similarity will control and guide the database transformation operations.

#### 6.3.1 Hierarchic Semantic Similarity Matrix

For a hierarchical structure as shown in Figure 6.2, a semantic similarity matrix as shown in Table 6.1 can be defined based on the properties of the hierarchical structure. A, B and C in Figure 6.2 represent the different branches in the hierarchical structure. For later use, they are called sub tree. T in the same Figure is called the top of the structure and  $c_i$  are object or object type.

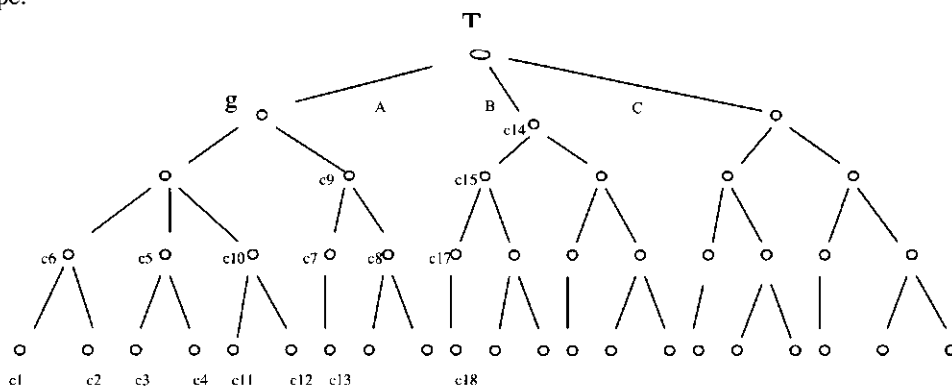


Figure 6.2 Example of a hierarchical structure



The semantic similarity matrix represents the similarity between object types.

**Table 6.1** Example of semantic similarity matrix

| SIMILARITY | Sub-type1 | Sub-type2 | ... | type1    | type2    | ... | Sup-type1 | Sup-type2 | ... |
|------------|-----------|-----------|-----|----------|----------|-----|-----------|-----------|-----|
| Sub-type1  | $s_{11}$  | $s_{12}$  | ... | $s_{14}$ | $s_{15}$ | ... | $s_{17}$  | $s_{18}$  | ... |
| Sub-type2  |           | $s_{22}$  | ... | $s_{24}$ | $s_{25}$ | ... | $s_{27}$  | $s_{28}$  | ... |
| ...        |           |           | ... | ...      | ...      | ... | ...       | ...       | ... |
| type1      |           |           |     | $s_{44}$ | $s_{45}$ | ... | $s_{47}$  | $s_{48}$  | ... |
| type2      |           |           |     |          | $s_{55}$ | ... | $s_{57}$  | $s_{58}$  | ... |
| ...        |           |           |     |          |          | ... | ...       | ...       | ... |
| Sup-type1  |           |           |     |          |          |     | $s_{77}$  | $s_{78}$  | ... |
| Sup-type2  |           |           |     |          |          |     |           | $s_{88}$  | ... |
| ...        |           |           |     |          |          |     |           |           | ... |

Where:

sub-type1, sub-type2 etc denote different elementary object types;

type1, type2 denote different object types;

sup-type1, sup-type etc denote (super) composite object type;

$s_{ij}$  denotes similarity value among object types.

The larger the value of an element in the matrix, the greater is the similarity between two object types that the element links. The matrix is a symmetric and reflexive one, and has the property that  $s_{ij}$  is equal to  $s_{ji}$  ( $s_{ij} = s_{ji}$ ) and  $s_{ii}$  is equal to  $s_{jj}$  ( $s_{ii} = s_{jj} = 1$ ) in the matrix.  $s_{ji}$  is a value between 0 and 1.

This matrix shows the similarity among different levels of object types. It will provide potential possibility to chose objects of different types to be merged or aggregated. The similarity matrix will be used as a look-up table for guiding or governing the aggregation process of spatial objects in semantics to a certain application.

The value of element  $s_{ij}$  in the matrix can be given by expert knowledge or by calculation (to be discussed in Section 6.3.2) based on aggregation hierarchy and classification hierarchy.

### 6.3.2 Computing Model of Similarity

Using Set theory, Tversky (1977) defines a similarity measure in terms of a matching process.

This measure produces a similarity value that is not only the result of the common, but also the result of the different characteristics between objects, which is in agreement with an information theory definition of similarity (Lin,D 1998, Rodriquez and Egenhofer 1999, Bishr, 1997, Chakroun et al,2000).

A similarity measure based on the normalization of Tversky's model and Set-theory function of intersection ( $A \cap B$ ) and difference ( $A-B$ ) is given in the following Equation (1), where  $c_i$  and  $c_j$  are different attribute structures; A and B correspond respectively to description set of  $c_i$  and  $c_j$  such as features;  $\|$  is the cardinality of a set; and  $\alpha$  is a function that defines the relation importance of the non-common characteristics.

$$S(c_i, c_j) = \frac{|A \cap B|}{|A \cap B| + \alpha(c_i, c_j) |A - B| + (1 - \alpha(c_i, c_j)) |B - A|} \quad (0 \leq \alpha \leq 1) \quad (1)$$

A natural approach to comparing the degree of generalization between object types is to determine the distances from these object types to the immediate super object types that subsumes them in a classification hierarchy as shown in Figure 6.1, that is, their least upper bound in a partially ordered set (Birkhooff,G, 1967). In a sense, the difference in the distances from these object types to the immediate super object types that subsumes them in a classification hierarchy reflects the difference in attribute structure (see Chapter 3 and Molenaar, 1998) between two object types.

A computational model that assesses similarity among objects and object types based on some definitions, concepts and hierarchical structure in Chapter 2 and 3 as well as Tversky's model is proposed. There are three distances. One is the distance (number of the link edges) from immediate super object type that subsumes  $c_i$  and  $c_j$  to the top of a hierarchy such as object type g to top of the tree T in Figure 6.1 which represents a common part of the attribute structure between two object types  $c_i$  and  $c_j$ . Another distance (number of the link edges) from immediate super object type that subsumes  $c_i$  and  $c_j$  to  $c_i$  such as object type g to  $c_1$  in Figure 6.1 which represents the different parts of an attribute structure between object type  $c_i$  and  $c_j$  ( $|c_i - c_j|$ ). And the third is the distance (number of the link edges) from immediate super object type that subsumes  $c_i$  and  $c_j$  to  $c_j$  such as object type g to  $c_5$  in Figure 6.1

which represents the different parts of attribute structures between object type  $c_i$  and  $c_j$  ( $|c_i - c_j|$ ), The proposed model is shown in Equation 2. It applied to two cases. One is for two given objects or object types belonging to the same sub tree such as sub tree A in Figure 6.1 and the other is for two given object types belonging to two different sub trees such as A and B as shown in Figure 6.1. For the first case, the model uses two types of distances to define the common and different properties between the given object types. One is the distance between given objects or object types and immediate super object types that subsumes them which reflects difference properties between two given object types and the other is the distance between immediate super object types that subsumes two given object types and the top of the hierarchical structure which reflects the common properties of two given object types. For the second case, the distance between immediate super object types that subsumes two given object types and the top of the hierarchical structure will be zero since the two given object types belong to different sub trees such as  $c_1$  and  $c_{15}$  in Figure 6.1. So this distance will be replaced by the correlation value between two sub trees in the Equation 2 (b).

$$s_{ij}(c_i, c_j) = \begin{cases} \frac{l}{l + \alpha(c_i, c_j) * d_{c_i} + (1 - \alpha(c_i, c_j)) * d_{c_j}} & (c_i \text{ and } c_j \in \text{same sub-tree}) \text{ (a)} \\ \frac{\beta}{\beta + \alpha(c_i, c_j) * d_{c_i} + (1 - \alpha(c_i, c_j)) * d_{c_j}} & (c_i \text{ and } c_j \notin \text{different sub-tree}) \text{ (b)} \end{cases} \quad (2)$$

Where:

- $l$ : the shortest distance (number of the link edge) from immediate super object type that subsumes  $c_i$  and  $c_j$  to the top of a hierarchy;
- $d_{c_i}$ : the shortest distance (number of the link edges) from immediate super object type that subsumes  $c_i$  and  $c_j$  to  $c_i$ ;
- $d_{c_j}$ : the shortest distance (number of the link edges) from immediate super object type that subsumes  $c_i$  and  $c_j$  to  $c_j$ ;
- $\alpha$ : a function of the distance (number of the link edge) between immediate super object type that subsumes  $c_i$  and  $c_j$  to the class  $c_i$  and  $c_j$ .
- $\beta$ : correlation degree among different sub-trees, such as similarity among agricultural land use, forest land use and building up land use, and its

value can be given by experts based on application requirement.

The  $\alpha(c_i, c_j)$  can be expressed as a function of the distance  $d_{ci}$  and  $d_{cj}$ . In order to get final values of  $\alpha$ , the function (Equation (3)) is defined as follows:

$$\alpha(c_i, c_j) = \begin{cases} \frac{d_{ci}}{d_{ci} + d_{cj}} & (d_{ci} \leq d_{cj}) \\ 1 - \frac{d_{ci}}{d_{ci} + d_{cj}} & (d_{ci} > d_{cj}) \end{cases} \quad (3)$$

where:

$d_{ci}$ : the shortest distance (number of the link edges) from immediate super object type that subsumes  $c_i$  and  $c_j$  to  $c_i$ ;

$d_{cj}$ : the shortest distance (number of the link edges) from immediate super object type that subsumes  $c_i$  and  $c_j$  to  $c_j$ .

This similarity function yields values between 0 and 1. The extreme value 1 represents the case that the two entity classes are completely the same, whereas the value 0 occurs when the two entity classes are completely different.

An example for computing an element of similarity matrix from a classification hierarchy is as follows: Taking Figure 6.1 as an example to calculate the similarity among object types and considering two cases (one is two object types at same sub tree (such as A in Figure 6.1) and the other is at different sub trees (such as A and B in Figure 6.1)), and supposing that the correlation value between A and B is 0.5, we have the following results based on the equations defined above:

Two objects or object types belong to A

$$S(c_1, c_2) = s_{12} = 3 / (3 + 0.5 * 1 + 0.5 * 1) = 0.75; \quad (\alpha = 0.5)$$

$$S(c_1, c_3) = s_{13} = 2 / (2 + 0.5 * 2 + 0.5 * 2) = 0.5; \quad (\alpha = 0.5)$$

$$S(c_1, c_5) = s_{15} = 2 / (2 + 0.66 * 2 + 0.34 * 1) = 0.546; \quad (\alpha = 0.66)$$

Two objects or object types belong to A and B respectively:

$$S(c_1, c_{14}) = s_{14} = 0.5 / (0.5 + 0.2 * 4 + 0.8 * 1) = 0.238; \quad (\alpha = 0.2)$$

$$S(c_1, c_{17}) = s_{17} = 0.5 / (0.5 + 0.43 * 4 + 0.57 * 1) = 0.127; \quad (\alpha = 0.43)$$

$$S(c_1, c_{18}) = s_{18} = 0.5 / (0.5 + 0.5 * 4 + 0.5 * 4) = 0.111; \quad (\alpha = 0.5)$$

### 6.3.3 Aggregation Rules Based on Similarity

Based on the similarity computation method, the aggregation rules need to be defined in categorical database transformation in order to search a suitable object to be aggregated in the transformation unit. For a given object violated geometric constraints or others, we must find one object or more in its transformation unit (neighbors) to aggregate with through the aggregation rules. Five rules are defined based on characteristics of categorical data and applied to categorical database transformation. These aggregation rules are feasible and accessible in categorical database transformation.

- **Rule 1:** for a given object, if some objects in their transformation unit belong to the same sub-tree as the given object does such as sub-tree A in Figure 6.1 and the other objects belong to another sub tree such as B in Figure 6.1, then the given object should be aggregated with the object belonging to the same sub tree as it in the transformation unit.
- **Rule 2:** for a given object, if all objects in their transformation unit belong to the same sub tree and there is an object with a maximum value of similarity between the given object and the other objects, then the given object will be aggregated with this object.
- **Rule 3:** for a given object, if all objects in their transformation unit belong to the same sub tree and there are two or more with an equivalent value of similarity between the given object and these objects, then the given object will be aggregated with the object which has the largest area or longest circumference among these objects in a transformation unit.
- **Rule 4:** for a given object, if all objects in their transformation unit belong to the different sub tree respectively and there is only one object with a maximum value of similarity between the given object and the other objects, then the given object will be aggregated with this object.
- **Rule 5:** for a given object, if all objects in their transformation unit belong to a different sub tree respectively and there are two or more objects with an equivalent value of similarity between the given object and the other objects, then the given object will be aggregated with the object which has the largest area or longest circumference among these objects in a transformation unit.

## 6.4 Area Object Aggregation Operations

After creation of different types of transformation units, the main task of database transformation is solving the conflicted objects in geometric, thematic and relational aspects in transformation units. As mentioned before, the transformation unit is the basis for an aggregation operation. It will facilitate the aggregation process in categorical database generalization. This is because the different types of transformation units will provide the information on conflicted objects, limit a set of objects to be processed and fix the object types and the similarity between objects to be evaluated. In categorical database generalization, the different types of aggregation operation (see following Section 6.4.1, 6.4.2, 6.4.3 and 6.4.4) will be triggered through visiting the transformation unit list which stores all data about transformation units.

The aggregation operation of spatial objects depends both on the geometric properties of spatial objects such as topological relations, and the semantic relations among the object types as discussed in the previous section. If a spatial object in a database will be merged with its neighboring objects, we have to decide which neighboring objects will be included in the transformation unit.

Based on the characteristics of four types of transformation units, four types of aggregation operation algorithms are designed (to be described later). The system will trigger corresponding aggregation operation according to the types of transformation unit.

### 6.4.1 Aggregation Operation Based on TU1

Let  $S$  be a similarity sub-matrix extracted from the similarity matrix as shown in Figure 6.2. The object types which objects belong to in the considered transformation unit are only included in this similarity sub-matrix.  $S$  will be extracted dynamically from the similarity matrix and used as look-up-table in the aggregation operation. Some of the objects in the transformation unit have the same attribute structure or similarity value to each other higher than the threshold based on the condition created for this type of transformation unit. After the aggregation operation, the common edge(s) between these objects will be deleted. The procedure for this type aggregation is as follows:

For each transformation unit  $CurrentTU$ , do the following:

- ```
{
• For each object  $theCurrentObject \in CurrentTU$ , do the following:
  {
    • Check if there exist objects having the same attribute structure or similarity value to  $S$  which is larger than the threshold, if the result is no, then move to the next transformation unit of this type, otherwise do the following:
      {
        • Find an object  $theobject \in CurrentTU$ , Get the edges of  $theCurrentObject$  and
```

theobject, and store them in a list *edgelist*;

- Delete the common edge in *edgelist* to form a new object *thenewobject* and update a topology relation among the objects and store *thenewobject* in *CurrentTU*;
- }
- Let *theCurrentObject*= *thenewobject*, repeat above steps at this level until there are no objects having the same attribute in *CurrentTU*;
- }
- Move to the next transformation unit in the data set and repeat all the steps at this level.
- }

6.4.2 Aggregation Operation Based On TU2

Let *S* be a semantic similarity sub-matrix that has the same definition as discussed in Section 6.4.1. The objects in this type of transformation unit are connected spatially to each other and one or some of them are too small and its or have areas that are less than the area threshold (see Figure 6.3 (a)). These objects may belong to the same sub-tree in a hierarchy or may not. The conflicted object(s) will be aggregated with a suitable object in a transformation unit through two ways based on the rules in Section 6.3.3. One is that the conflicted object will be aggregated with an object having maximum similarity value in *S* in the transformation unit and the other is with an object having a largest area or longest common edge with the conflicted object in the transformation unit. After the aggregation operation, the common edges of the objects are deleted and a new object is formed. The procedure for this type of aggregation operation is listed below:

For each transformation unit *CurrentTU* ∈ *TransformationUnitList*, do the following:

- {
- For each conflicted object *theCurrentObject* ∈ *CurrentTU*, do the following:
 - {
 - Check if objects exist that have higher similarity value from *S* than the threshold, if the result is no, then move to the next transformation unit; otherwise do the following:
 - {
 - Find all the objects having higher similarity value s_{ij} with *theCurrentObject* in *CurrentTU* and store them in a list *theTempobjectList*;
 - Check to find a maximum similarity value s_{ij} between *theCurrentObject* and *theobject* ∈ *theTempobjectList*;
 - If the result is Yes, do the following:
 - {
 - Get the edges of two objects *theCurrentObject* and *theobject*, and store them

in a list *edgelist*;

- Delete common edges in the *edgelist*. reorganize edges in the *edgelist* to form a new area object *thenewobject* and delete two old objects from the data set and update topology relation (see Figure 6.3 (a), (b), here, the similarity value between object A and B is higher than the one between A and C);
- }
 - Else
 - {
 - Find a *theobject* maximum area or longest common edge sharing with *TheCurrentObject* in the *TempobjectList*;
 - Get the edges of two objects *theCurrentObject* and *theobject*, and store them in a list *edgelist*;
 - Delete common edges in the *edgelist*. reorganize edges in the *edgelist* to form a new area object *thenewobject* and delete the two old objects from the data set and update thematic attribute and topology relation (see Figure 6.3 (c) here, the similarity value from S between object A and B is the same as the one between A and C, but area of object C is larger than B);
- }
 - Move to the next transformation unit and repeat all the steps at this level.

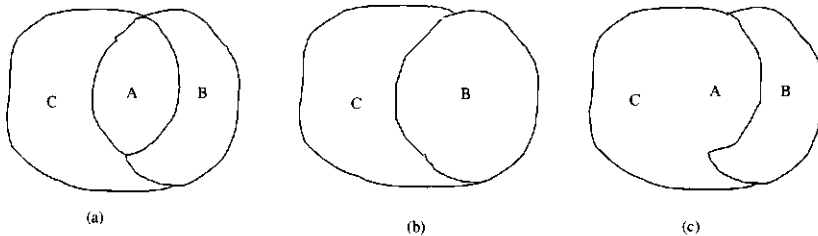


Figure 6.3 Examples of aggregations based on TU2

6.4.3 Aggregation Operation Based On TU3

Let *S* be a semantic similarity sub-matrix that has the same definition as discussed in Section 6.4.1. The objects in this type of transformation unit are near neighbors but not connected spatially to each other (see Figure 6.4 (a)) and the distance between two objects are too close and violate the distance threshold. These objects may belong to the same sub-tree in a hierarchy

or not. The objects with a distance between them less than the threshold will be aggregated to form a new object or aggregated to two other objects, in which the space between them will be parted based on the skeleton line (see Section 5.3.2.3 and Figure 6.4 (c)) and assigned to the two objects respectively. The decision on forming one object or two objects depends on the similarity value from S. The following part gives this type of aggregation operation procedure:

For each transformation unit $CurrentTU \in TransformationUnitList$, do the following:

- ```
{
```
- Check if there exist a pair of objects where the space between them is less than the threshold in  $CurrentTU$ ; if the result is no, then move to the next transformation unit; otherwise do the following:
 

```
{
```

    - Triangulate the objects in the transformation unit
    - If they have the same thematic attribute structure or the thematic similarity value between them is larger than the threshold from S. Do the following:
 

```
{
```

      - Get the edges of two objects, and store them in a list *TempEdgelist*.
      - Store the most outer edges c and d of a set of triangles between A and B in a list *TempEdgeList* (see Figure 6.4 (a));
      - Break down polygon A and B to form polygon  $A' (a_1, a_2)$  and  $B' (b_1, b_2)$  (keep clockwise way) based on the intersection points of c and d with A and B respectively and store them in *TempEdgeList* (see Figure 6.4 (a));
      - Delete edges  $a_2$  and  $b_2$  which are related to the triangles between A and B from *TempEdgeList*, and get edges  $A' (a_1)$  and  $B' (b_1)$  of polygon A and B, and store them in *TempEdgeList*.
      - Link outer edge and edge of polygon in *TempEdgeList* based on line adjacent relation between outer edge and edge of polygon, and form a new object *thenewobject*  $(a_1, c, b_1, d, a_1)$  (see Figure 6.4 (a) and (b));

```
}
```
    - Elseif they have different thematic properties or the thematic similarity value from S between them is less than the threshold, do the following:
 

```
{
```

      - Get the edges of two objects and store them in a list *TempEdgelist*.
      - Store the most outer edges c and d of a set of triangles between A and B in a list *TempEdgeList*.

```
}
```
- ```
}
```

- Extract skeleton line (see Section 5.3.2.3) z and store it in *TempEdgeList*.
 - Break down polygon A and B to form polygon A' (a_1, a_2) and B' (b_1, b_2) (keep clockwise way) based on the intersection points of c and d with A and B respectively and store them in *TempEdgeList*.
 - Break down outer edges c and d into $c(c_1, c_2)$ and $d(d_1, d_2)$ based on the intersection points of z with c and d , store c_1, c_2, d_1 and d_2 in *TempEdgeList*; and at the same time delete c and d from *TempEdgeList*.
 - Delete edges a_2 and b_2 which are related to the triangles between A and B from *TempEdgeList*, and get edges A' (a_1) and B' (b_1) of polygon A and B, and store them in *TempEdgeList*;
 - Link edges according to the order of edge of polygon, outer edge and skeleton line in *TempEdgeList* based on line adjacent relation between outer edges, edge of polygon and skeleton line and form two new objects *thenewobject* (see Figure 6.4 (c) and (d));
- }
- Store *thenewobject* in *CurrentTU* and delete the two old objects from the data set and update thematic attribute and topology relation;
- Repeat all steps at this level.
- }

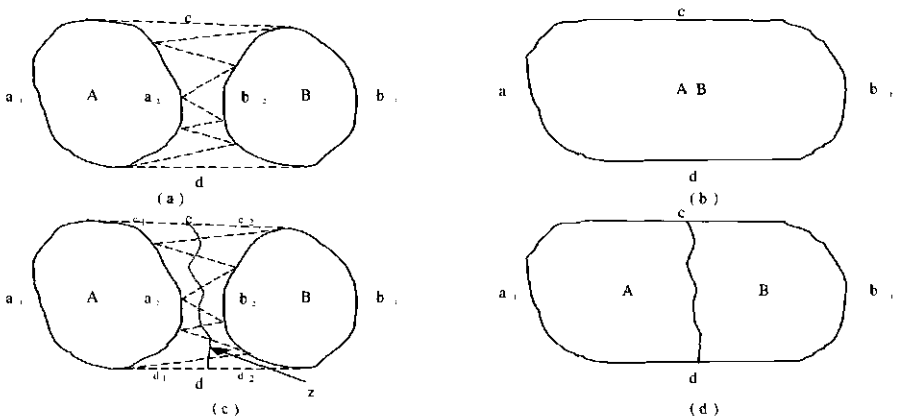


Figure 6.4 Examples of aggregation operations based on TU3

6.4.4 Aggregation Operation Based on TU4

Let S be a semantic similarity sub-matrix which has the same definition as in previous Section 6.4.1. The objects in the transformation unit are near neighbors (connected or disconnected spatially) with each other and there is at least a conflicted object violated geometric constraint (see Figure 6.5 (a)). These objects may belong to the same sub-tree in a hierarchy or may not. The conflicted object(s) may be aggregated with an object from its first order neighborhood or from its second order neighborhood (see Section 6.5). This depends on the similarity value among the objects, the distance between objects and the aggregation rules. The procedure for this type of aggregation operation is given below:

For each transformation unit $CurrentTU \in TransformationUnitList$, do the following:

- ```
{
```
- Check if conflicted objects and conflicted relations exist in  $CurrentTU$ , if the result is no, the move to the next transformation unit; otherwise do the following:
 

```
{
```

    - For each conflicted object  $theCurrentObject \in CurrentTU$ , do the following:
 

```
{
```

      - Triangulate the objects in the transformation unit;
      - Find the objects first order and second order neighbor of  $theCurrentObject$  through constrained triangulation (see Section 6.5), store them in two lists  $theNeighbor1$  and  $theNeighbor2$  respectively.
      - If  $theCurrentObject$  has the same thematic attribute structure as  $theobject \in theNeighbor1$  or the thematic similarity value  $s_{ij}$  from  $S$  between  $theCurrentObject$  and  $theobject \in theNeighbor1$  is larger than the value between  $theCurrentObject$  and  $theobject \in theNeighbor2$ ; do the following:
 

```
{
```

        - If object  $theCurrentObject$  is an inland in  $theobject$  ;
          - Delete object  $A$  and form  $thenewobject$  (see Figure 6.5 (a) and (b));
        - Else
 

```
{
```

          - Get the edges of two objects  $theCurrentObject$  and  $theobject$ , and store them in a list  $edgelist$ ;
          - Delete common edges in the  $edgelist$ . Reorganize edges in the  $edgelist$  to

form a new area object *thenewobject* and delete the two old objects from the data set and update topology relation;

}  
}

- Else if the thematic similarity value from *S* between *theCurrentObject* and *theobject*  $\in$  *theNeighbor2* is higher than the value between *theCurrentObject* and *theobject*  $\in$  *theNeighbor1* and the distance between the two objects is less than the threshold; do the following:

{

- Triangulate the objects in the transformation unit (see Figure 6.5 (c));
- Get the edges of two objects *theCurrentObject* and *theobject*, and store them in a list *TempEdgelist*.
- Store the most outer edges *c* and *d* of a set of triangles between *theCurrentObject* and *theobject* in a list *TempEdgeList*.
- Break down polygon *A* and *B* to form polygon *A'* ( $a_1, a_2$ ) and *B'* ( $b_1, b_2$ ) (clockwise) based on the intersection points of *c* and *d* with *theCurrentObject* and *theobject* respectively and store them in *TempEdgeList*.
- Delete edges  $a_2$  and  $b_2$  which are related to the triangles between *theCurrentObject* and *theobject* from *TempEdgeList*, and get edges *A'* ( $a_1$ ) and *B'* ( $b_1$ ) of polygon *theCurrentObject* and *theobject*, and store them in *TempEdgeList*.
- Link outer edge and edge of polygon in *TempEdgeList* based on line adjacent relation between outer edge and edge of polygon, and form a new object *thenewobject* ( $a_1, c, b_1, d, a_1$ ) (see Figure 6.5 (d));

}

- Store *thenewobject* in the CurrentTU, delete aggregated objects and update thematic attribute and topology relation;

}

- Repeat all above steps at this level.

}

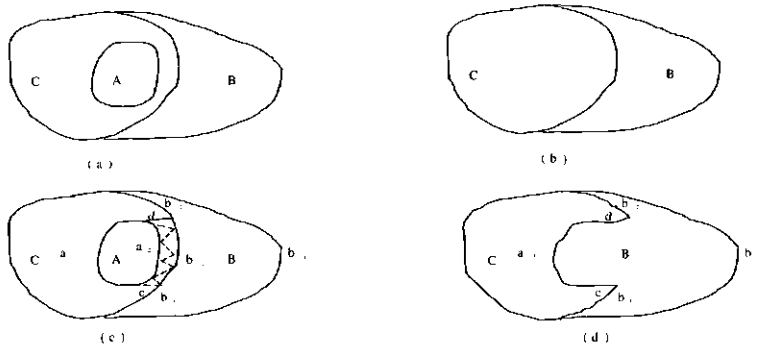


Figure 6.5 Examples of aggregation operations based on TU4

### 6.5 Object Clustering

Clustering a set of area objects in a database using constrained Delaunay triangulation is very efficient. Peng (1997) discusses this type of problem for a set of area objects. In his research, detection of regular linear groups of objects within a larger group is implemented by two parameters orientation and “width” of a triangle. This study concentrates on clustering a group of objects based on the average length of the edges of the triangles between two objects. The average length of three edges of a triangle between two objects is the basis for clustering objects in a set of objects in a database. If the length is less than the given threshold, then two objects could be classified as a group.

The particular example to be considered concerns clustering a set of area objects (see Figure 6.6 (a)). In Figure 6.6, the objects are rather similar sizes; the distances between neighboring area objects in the group are similar and are normally less than the distance to the nearest area object outside the group.

The algorithm which has been developed is based on this understanding and the Delaunay triangulation of area objects, and starts with triangulating a considered set of objects, tracing continuously the objects which have a distance less than the threshold between its adjacent (not connected) objects through checking the average edge length of the triangle between the two objects.

- Let *CurrentObjectList* be object list and store all considered objects ;
- Let *ClusterObjectList* be empty cluster list and store the result of clustering object;
- Triangulating a set of area objects  $\in$  *CurrentObjectList* which are disconnected;

- Delete the triangles within each area objects (see Figure 6.6 (b));
- For each object  $theCurrentObject \in CurrentObjectList$ , do the following:
  - {
    - Calculate the average length of three edges of triangles between  $theCurrentObject$  and the object of its neighbor; check if there exist object(s) of its neighbor having the distance (average edge length of a triangle) less than the threshold with  $theCurrentObject$ ; if no, move to the next object in  $CurrentObjectList$ ; otherwise do the following:
      - {
        - Push the object(s) of  $theCurrentObject$  's neighbor in a stack  $thestack$  and Store  $theCurrentObject$  and the object(s) in a temporary cluster list  $theTempClustesList$ ;
        - Pop up an object  $theobject$  from  $thestack$ , check if there exist object(s) of its neighbor having the distance less than the threshold with  $theobject$ ; if yes, push the object(s) in  $thestack$  and store the object(s) in  $theTempClustesList$ ; otherwise do the following:
          - {
            - Check if there exist the object(s) in  $thestack$ , if yes, then repeat the previous step (indicated by icon ■), otherwise do the following:
              - Store all objects in  $theTempClusterList$  to  $ClusterObjectList$ ;
- Move to the next area object in  $CurrentObjectList$  and repeat all the steps at this level until the  $CurrentObjectLis$  is empty.

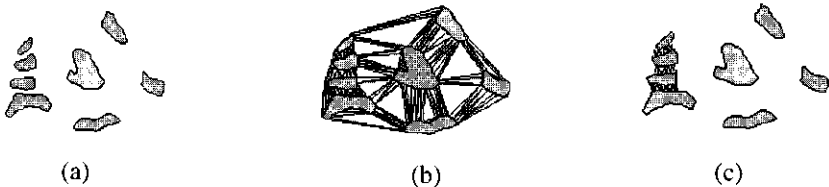


Figure 6.6 Example of object clustering.

Figure 6.6 (b) and (c) show the process and result of object clustering.

## 6.6 Multi-Order Neighborhood

Multi-order neighborhoods (K-order neighborhood) play an important role in a complicated aggregation operation. In a sense, it can be used to express the relation between two adjacent but not connected objects. Some authors describe the multi-order neighborhood using adjacent graph method (Molenaar 1998, Zhan, C. 2000 and Lang et al 2001). In this study, it will focus on using the constrained Delanauy triangulation network to describe and analyze a multi-order neighborhood. Suppose we have a set of area objects as shown in Figure 6.7 (a). For the object  $o$  in the figure, its first order neighborhood is defined based on a triangulation network as follows. Those area objects connected to a given area object  $o$  by a set of triangles which do not belong to  $o$  are object  $o$ 's first order neighborhood. The objects which are object  $o$ 's first order neighborhood are shown in Figure 6.7 (b). This set of area objects is called the 'first-order' of area object  $o$ . Then those area objects that are connected to the first-order neighbors by the triangles and the triangles are not first-order neighbors themselves, are called 'second-order neighbor' and can be obtained through the same method above as shown in Figure 6.7 (c). By continuing this process we can define k-order neighbor for any object.

Let *theneighborlist* be an empty list

For a given area object, do the following:

- ```
{
```
- Find all triangles *thetri* whose constrained edge is the edge of the given area object and not belonging to the given area object;
 - Store *thetri* in a list *thetrist*;
 - For each triangle $tri \in \textit{thetrist}$, do the following:


```
{
```

 - Identify the object which *tri* belongs to and check if the object has been included in *theneighborlist*, if the result is yes, do nothing; otherwise;
 - Store the object in *theneighborlist*;

```
}
```
 - For objects $\in \textit{theneighborlist}$,
 - Repeat above steps, finish other order neighbors search.

Figure 6.7 (a) is a set of objects. And Figure 6.7 (b) and (c) show the result of 1-order and 2-order neighbors of the center object respectively, in which 1-order and 2-order neighbors of the area object are searched by using above algorithms.

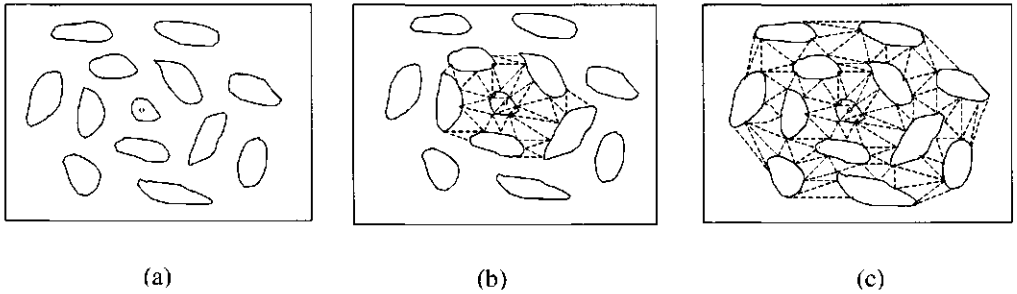


Figure 6.7 Example of multi-neighborhood

6.7 Creating Hierarchical Catchments of River Network

River catchments play an important role in river basin management and traditional map generalization. In generalization, the size of catchments will be one of the important factors for selecting the streams for a drainage network. Recharadson (1993) discusses the problem of river system generalization based on Strahler classification and Horton classification. The problems of formalization of catchments and catchment generalization based on FDS and Strahler classification are discussed by Molenaar and Martinez Casanovas (1996), Martinez Casanovas (1994) and Molenaar (1998). In this section, we mainly concentrate on discussing how to create hierarchical catchments of a river network based on Horton's river classification system and constrained Delaunay triangulation network. The river classifications are introduced briefly.

6.7.1 Straler's, Shreve's and Horton's Classification

Straler's (1957), Shreve's (1967) and Horton's (1945) classification are shown in Figure 6.8. Each river system has an intrinsic hierarchical structure that can be described by these three different stream ordering procedures. This hierarchical structure can be utilized as the basis for a feature elimination procedure.

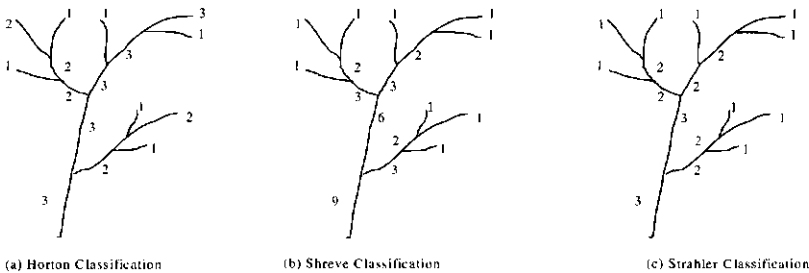


Figure 6.8 Classification of river systems

Stream order is a measure of the position of a stream in the hierarchy of tributaries. The criteria for stream order enumeration, though, may be geometric, topologic, or volumetric. Two topologic criteria, magnitude and order, are considered. The magnitude of a given link establishes its position in relation to its directly adjacent links. Thus it is independent of the overall size of the system, of what portion is being enumerated, or of changes that might be made in the system. Order values, on the other hand, describe the link's position in relation to the entire river system. Straler's and Shreve's classifications enumerate stream magnitude which create a hierarchic river network structure from the parts to create the whole.

Horton's system of ordering (Horton, 1945) has been shown to be the most useful for establishing a database structure that is amenable to an objective and simple generalization procedure that can be used in computerized data bases where the system may be required to generate different databases at quite different scales. In the system devised by Horton, unbranched fingertip tributaries are always designated as order 1, tributaries or streams of the 2nd order receive branches or tributaries of the 1st order, but these only; a 3rd order stream must receive one or more tributaries of the 2nd order but may also receive 1st order tributaries. A 4th order stream receives branches of the 3rd and usually also of lower orders, and so on. Using this system the order of the main channels is the highest. Horton assigned order 1 to the fingertip tributaries and the highest order to the main trunk. He wanted all fingertip tributaries to be of the same number of links. It is shown that in fact the Horton procedure can be used to answer both questions of "How many" and "which ones". It provides a framework for the establishment of an objective and geographically oriented generalization scheme (Grabrecht, J. Martz, L. 1997, Mazur, Z.R and Castner, H.W, 1990, Martinez Casanovas 1994, Rodriguez et al 1999).

Ordering is only the first step in the quantitative analysis in database generalization.

6.7.2 Requirements of Creating Hierarchical Catchments of River Network

Creating a hierarchical catchment area must meet several requirements as follows:

- River system must be complete;
- Catchment area of a order stream includes all catchments areas of its branches at the next lower order; in other words, each of catchment areas of its branches is a part of the catchment area.
- A drainage network consists of a set of drainage links connected by network nodes. Three types of nodes are encountered in a drainage network: the outlet node, upstream tips of the drainage network where drainage links originate (source nodes) and points at which two or more channel link join (junction nodes).

- Stream are considered to be basic units, called links, which are usually uniquely identified from their sources to their mouths and are fed by several tributaries.
- In order to create hierarchical catchments, Horton's classification needs to be adjusted. There will be only one highest order link in a river network according to Horton's classification, but there may be more than one link at each order of the other orders.

The strategy of creating a new catchment area is step by step from the highest order link to the lowest links. At first a catchment area of the highest order link will be created, and then catchment areas of all links for each order are constructed gradually in order downward till the lowest order.

The main procedures are discussed in the next part. The main steps include: 1) A river network is classified by Horton's classification system and gets the number of catchment levels through the number of Horton's river orders in a river network. In order to create hierarchical catchments effectively and efficiently, Horton's classification of a river network has to be changed based on the following rule: if the order of the river links is not the next lower order of its adjacent link, then it changes its order into the next lower order. This rule to all links in a river network is implemented except for the highest order link (see Figure 6.10); 2) triangulating through different order links respectively from the highest to the lowest order; 3) extracting skeleton lines between links or between link and boundary of catchments; 4) forming catchments of different order links.

6.7.3 Process of Creating Hierarchical Catchments of River Network Based on CDT

- Let *thenodeslist* be a node list and used to store the nodes;
- Let *thecatchmentslist* be an empty catchment boundary list and used to store boundaries of catchments;
- Let *thelinklist* be a link list and used to store river links;
- Let *maxorder*=number of order of river and *minorder*=1;
- Get highest order link from *thelinklist* and all nodes from *thenodeslist*, and form a convex, and this convex area may be considered as a catchment area of the highest order link .
- Let *theorder*=*maxorder*-1
- From *theorder* to *minorder*, do the following:
 - {
 - get all link objects at the *theorder* level and the link objects at the level directly above *theorder* from *thelinklist*, and store them in *templinklist*, and get all the nodes of links

whose order is lower than the order from *thenodeslist* and store them in *tempnodeslist*, and get all boundary of catchment area from *thecatchmentslist* and store them in *tempcatchmentlist*.

- construct constrained delaunay triangulation based on the data of *templinklist*, *tempnodeslist* and *tempcatchmentlist*.
- for each link object *linkobject* from *templinklist*, do the following:
 - {
 - for a considered link, follow the procedure described in Chapter 5 to get all its neighbor nodes, neighbor links object, and its neighbor boundary of catchment areas at the next higher or the higher order links;
 - get a subset of triangles which are relative to the considered link object and its neighbor objects (including source nodes, junction (outlet) nodes and boundary of catchment area) and store then in the list *temptrilist*;
 - find the triangles which have junction (outlet) node of the considered link object as their vertex and the considered link object as their constrained edges; push the triangles into the stack *starttristack*, The junction node will be the starting point to trace the boundary of catchments area and the triangles will be the starting triangles to trace catchment areas;
 - if *starttristack* is empty, stop tracing the boundary of catchment area of the considered link object and move to the next link object in *templinklist* if there are still link objects at the same level as *theorder* not to be processed; otherwise do the following:
 - {
 - pop up an object *theobject* from *starttristack*, let *trioject=theobject*;
 - add the junction node of the considered link object to the list *tempboundarypointlist*
 - for each triangle *trioject* ∈ *temptrilist*, do the following
 - {
 - if the triangle object *trioject* ∈ tri3
 - {
 - compute the middle point of non-constrained edge of *trioject*;
 - add the middle point to *tempboundarypointlist*;
 - }
 - }

- if the triangle object *triobject* \in tri2
 - {
 - if one of three vertexes of *triobject* is the source node of the considered link or the source node of the next lower /or next higher order link of considered link and the constrained edge is the boundary of catchments at the next higher order link, do the following:
 - {
 - add one vertex of the triangle on the constrained edge to the list *tempboundarypointlist*;
 - connect all points in *tempboundarypointlist* to form line object from junction node to last point through all middle points;
 - add the line object to the list *tempcatchmentlist* and empty *boundarypointlist*;
 - if the vertex of the triangle is the source node of the link whose order is lower than the considered link and the constrained edge of the triangle is the considered link, then check if the corresponding junction node of the source node is on the considered link object . If the result is yes, then do nothing to the *triobject*;otherwise do the following :
 - {
 - compute the middle points coordinates of unconstrained edges of the triangle;
 - add the middle points to *tempboundarypointlist*;
 - if the vertex of the triangle is on the adjacent link /or source node of the adjacent link at the same order of the considered link or on the adjacent link at the next higher order and the constrained edge of the triangle is the considered link, then do the following :
 - {
 - compute the middle points coordinates of unconstrained edges of the triangle;
 - add the middle points to *tempboundarypointlist*;
- if the triangle object *triobject* \in tri1
 - {
 - if one of three vertexes of *triobject* is outlet node of the lower order

neighboring link of the considered link and the other two vertexes of *trioject* are on the two neighboring links at the same order as the considered link respectively; or

- if one of three vertexes of *trioject* is source node of the considered link and the other two vertexes of *trioject* are on the two neighboring links at the same order as the considered link respectively, do the following:
 - {
 - compute the coordinates of the center point of *trioject*;
 - add the coordinates to the list *tempboundarypointlist*;
 - }
- if one of three vertexes of *trioject* is source node of the considered link object, and one of the other two vertexes of *trioject* is on a neighboring link at next higher or the same order and another is on the boundary of the catchments area at the higher order link of the considered link or;
- if one of three vertexes of *trioject* is source node of the considered link, and one of the other two vertexes of *trioject* is the source node of a neighboring link at next lower order and another is on the boundary of the catchments area at the higher order link of the considered link or;
- if two of three vertexes of *trioject* are on source node of two next lower order neighboring links of the considered link and another is on the boundary of the catchments area at the higher order link of the considered link or;
- if one of three vertexes of *trioject* is source node of considered link and the two other vertexes are on the different boundary of the catchments at the higher order link of the considered link respectively or;
- if one of three vertexes of *trioject* is source node of a neighboring lower order link of the considered link and the other two vertexes are on the neighboring links of the considered link and the considered link respectively; do the following:
 - {
 - add the vertex of *trioject* on the boundary of catchment to The list *tempboundarypointlist*;
 - }

- connect all points in *tempboundarypointslist* to form boundary object from junction node to last point through all middle points;
- add the line object to the list *tempcatchmentlist*;
- empty *tempboundarypointslist*;
}
- if one of the vertexes of *triobject* is on the considered link and the other two vertexes are source nodes of two neighboring lower order links of the considered link respectively or;
- if one of the vertexes of *triobject* is on the considered link and the other two vertexes are source and outlet node of neighboring lower order link of the considered link respectively or;
- if one of three vertexes of *triobject* is source node of a lower order neighboring link of the considered link and the other two vertexes of *triobject* are on two neighboring links of the considered link, then check if the corresponding junction node of the source node is on the considered link. If the result is yes, then do nothing to the *triobject*; otherwise do the following:
 - {
 - compute the coordinates of the center point of *triobject*;
 - add the coordinates to the list *tempboundarypointslist*.
}
- if three vertexes of *triobject* are source or outlet nodes of lower order links compared to the considered link, then do nothing.
}
- }
}
- add *tempcatchmentlist* to the list *catchmentlist*;
- empty *templinklist*, *tempnodeslist*, *tempboundarylist* and *tempcatchmentlist*
}
- let *theorder=order-1* and repeat the above steps;
}

Figure 6.9 gives test data of a river system and Figure 6.10 is its corresponding order classification system.



Figure 6.9 Example of a river system



Figure 6.10 Order of the river system

Figure 6.11 shows the catchments of order 4 of the river system. Figure 6.12 gives the catchments of order 3 of the river system. Figure 6.13 presents the catchments of order 2 of the river system and Figure 6.14 shows the catchments of order 1 of the river system .

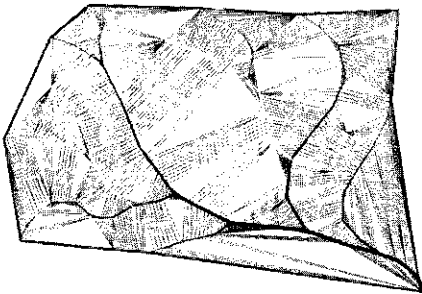


Figure 6.11 example of catchments of order 4

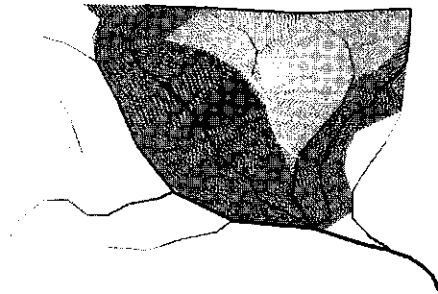


Figure 6.12 example of catchments of order 3

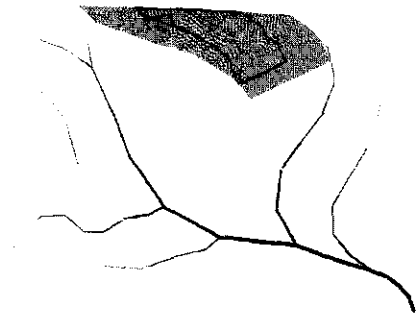
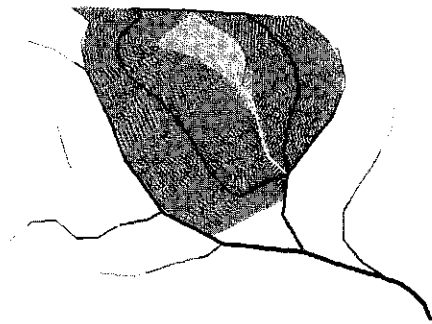


Figure 6.13 example of catchments of order 2 **Figure 6.14** example of catchments of order 1

Chapter 7

Application of Methods

7.1 Introduction

Chapter 5 explained the integration of the Formal Data Schema and Delanuy triangulation network. This has been called the IEFDS. Chapter 4 discussed the constraints for database generalization. These are pre-requirements for the definition of four different types of transformation units in Chapter 6. Semantic similarity indices were then introduced to decide which objects in such a unit should be combined into an aggregated object. The actual aggregation operation depends on the types of TU. In this chapter the examples will be elaborated to demonstrate the functionality of generalization processes based on these concepts. The examples of the application include object clustering, land use aggregation and automated organization of hierarchical catchments. The data used in the examples are both from a real data set and simulated data.

7.2 Object Clustering Based on CDT

The section intends to demonstrate the concept of a transformation unit based on geometric properties such as the distance between two objects. The distance between the objects is represented by the average length of three edges of the triangles between two objects in this study. This type of transformation unit can be reached by clustering spatial objects. The main requirement for clustering spatial objects is that clusters should reflect main spatial distribution properties of the spatial objects. There are two main aspects involved in the demonstration. One is to cluster the objects that are considered to be too close based on the average length of edges of triangles between two objects. These clustered objects form different transformation units. The average length of edges of triangles between objects as a constraint plays a key role in clustering objects. Therefore, the other is to analyze the influences of the average length on object clusters according to setting different threshold values. The objects among which the average length is less than the pre-threshold value in a transformation unit will be aggregated. A dataset of lakes at a scale of 1:10000 from ShenZheng Bureau of Urban Planning and Land Resource Management of ShenZheng city in P.R. China is applied to test the algorithms in Chapter 6. The data set consists of 70 different sizes of lake objects (see Figure 7.1). Some of them are too small or the distance between them is too close when the resolution of a database is changed from higher to lower. This data set maybe used to answer the questions listed below:

- How to analyze the spatial relation among the objects?
- How to identify a group of objects with a significantly closer distance relation than any other group?

- How to aggregate this group of objects?



Figure 7.1 A set of lakes

The main steps for clustering objects include:

- Constructing triangulation networks of the objects;
- Setting a distance threshold value between the objects;
- Object aggregation;
- Analyzing a distance threshold value between the objects

7.2.1 Constructing Triangulation Network of Objects

The first step for analyzing spatial properties of the objects is to construct constrained triangulation networks of the objects as shown in Figure 7.2.

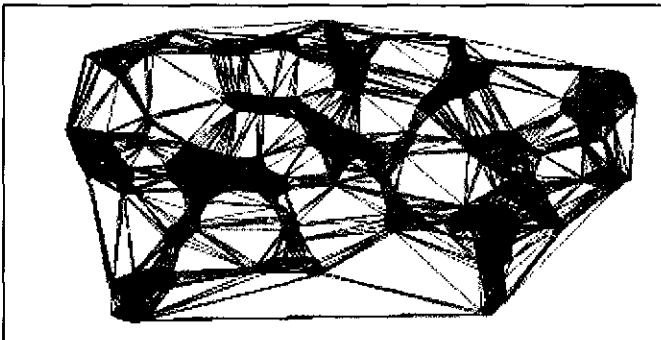


Figure 7.2 Triangulation network of the objects

The edges of the objects are constrained edges of triangulation. Then the triangles within the objects in Figure 7.2 are deleted. Figure 7.3 shows the result.

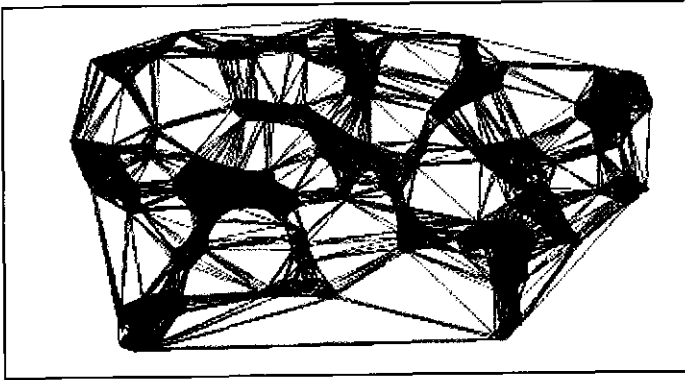


Figure 7.3 Objects and Triangulation network between the objects

7.2.2 Setting Distance Threshold Value between Objects

The threshold value between the objects is set based on experts or can also be set by statistical analysis such as histogram methods. Figure 7.4 gives a dialog to set the threshold value.

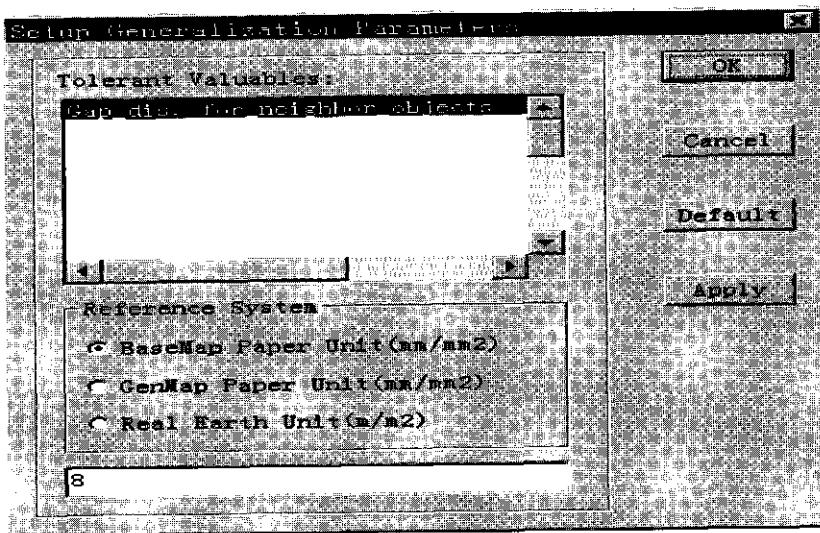


Figure 7.4 Parameter setup dialog

The different threshold values can be set through the dialog. After setting the threshold, the triangles between the objects are deleted if their average length of the three edges is larger than the threshold and only the triangles whose average length is less than the threshold are kept. The objects which are connected by a set of triangles are formed as several transformation units under the threshold value of 5mm as shown in Figure 7.5.



Figure 7.5 Object cluster

7.2.3 Object Aggregation

The groups of objects connected by a set of triangles will be transformation units (the procedure for creating this type of transformation unit is described in Section 6.2.2). The objects of these units will be aggregated to form the new objects.



Figure 7.6 Boundary of clustering objects

The boundary of the new object must be detected. The boundary of the new object consists of part of edges of objects and outer edges of triangles between the objects in a transformation unit in Figure 7.5 and 7.6. Connecting this set of the edges and outer edges of triangles will form the boundary of the new object as shown in Figure 7.6 (dark line of each group). Detecting this set of the edges of the objects and the outer triangle edges between the objects and building the boundaries is described in section 6.4.3. Figure 7.7 shows the result of the aggregation.

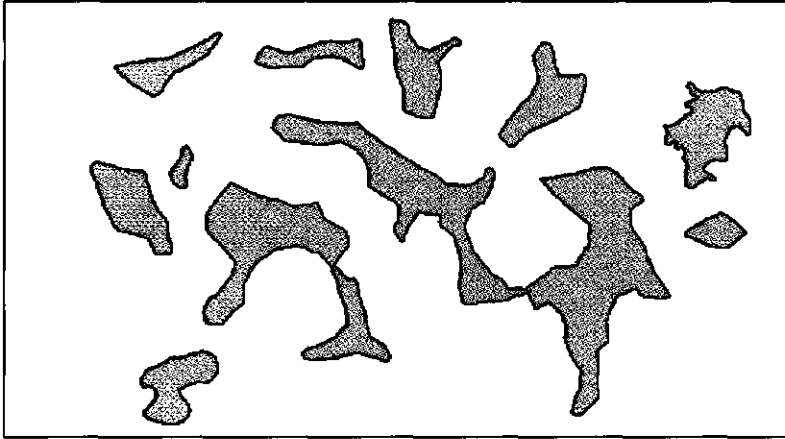


Figure 7.7 Result of aggregation

7.2.4 Analyzing Distance Threshold Value between Objects

There will be different numbers of transformation units when changing the distance threshold value between the objects. Figure 7.8, Figure 7.9, Figure 7.10 and Figure 7.11 express different object aggregation cases with the distance threshold values of 1, 3, 8 and 10 mm respectively.

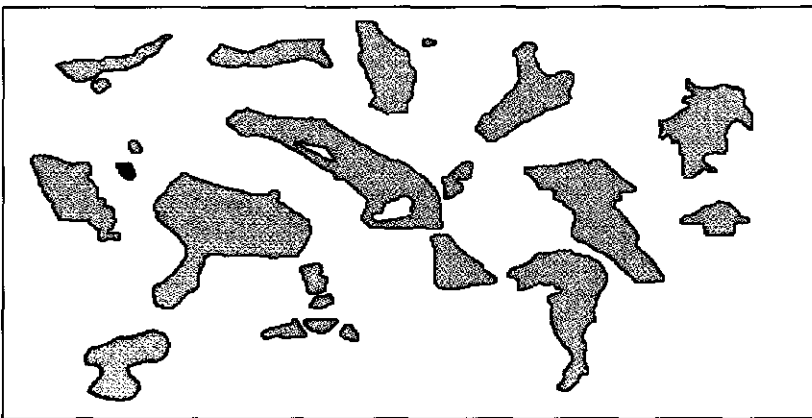


Figure 7.8 Result of object aggregation with the threshold value of 1 mm

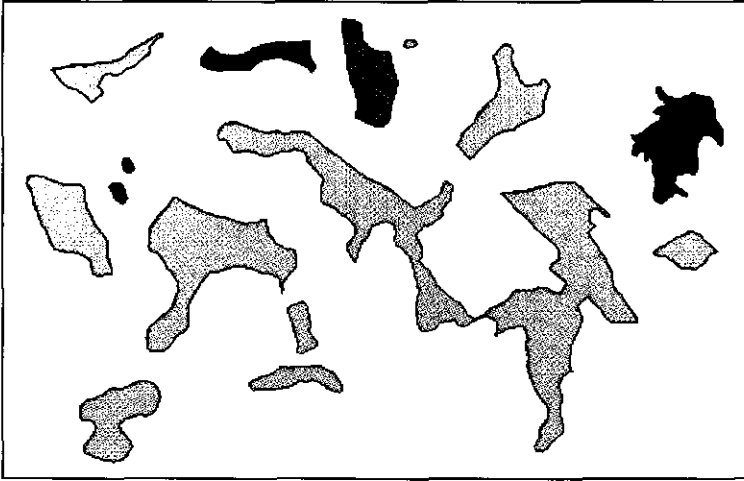


Figure 7.9 Result of object aggregation with the threshold value of 3 mm

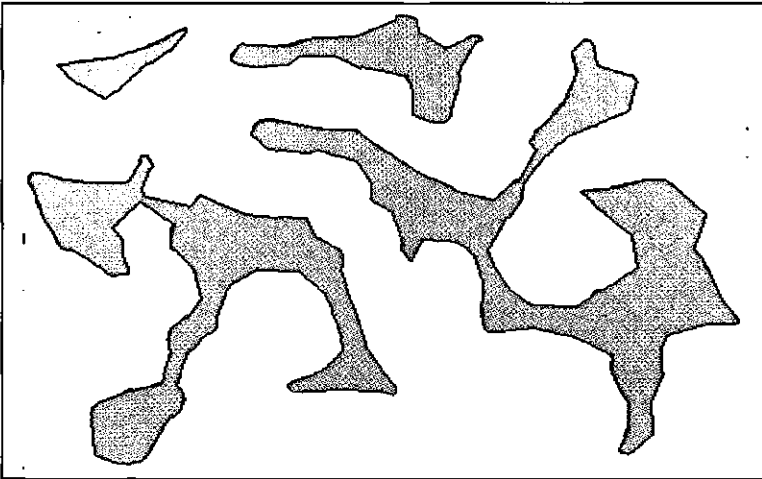


Figure 7.10 Result of object aggregation with the threshold value of 8 mm

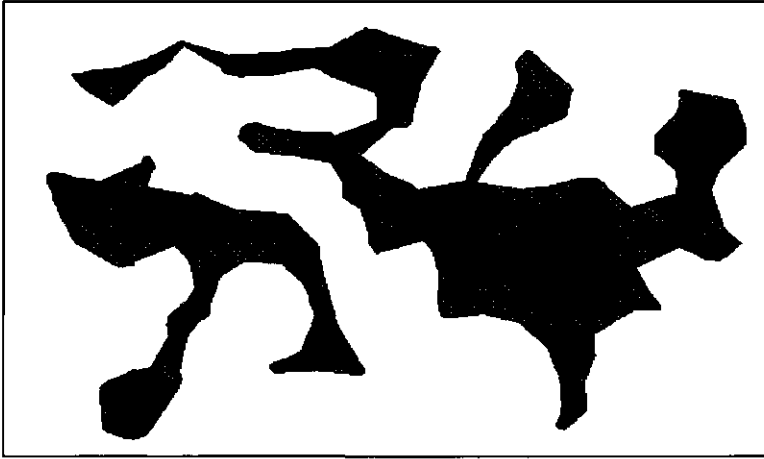


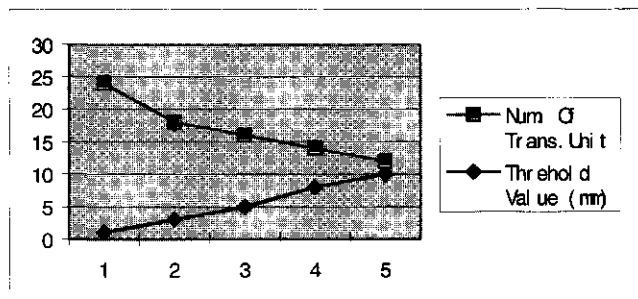
Figure 7.11 Result of object aggregation with the threshold value of 10 mm

It can be seen from Table 7.1 (a) and (b), the number of transformation units decrease as the distance threshold values increase. Changing a threshold value depends on the application requirement and experience as well as requiring quite a lot of knowledge of the data and the application field. From the algorithm point of view, selecting the average length of edges of the triangles between two objects as the distance between two objects is reasonable and feasible since it is easy to find a group of objects within a certain distance range and to build the boundary of a new object (aggregated object). In the results the shape of each aggregation object can reflect the main spatial distribution characteristics of the object clusters before aggregation.

Table 7.1 Relations between Threshold values and Transformation units

Threshold Value (mm)	Num. Of Trans.Unit
1	23
3	15
5	11
8	6
10	2

(a)



(b)

7.3 Land Use Database Generalization

Landuse database is a typical categorical database. Multiple representation of land use is very important in land evaluation, planning, monitoring and management at different levels. There are different requirements for land use in detail for different applications. This means that a database may be suitable for one application purpose but not for another application. For a landuse database, the requirements deriving a new database from this database may be different if it is applied to different purposes. For example, deriving a database from this database for land management at one level and for land evaluation at the same level will be different. For land evaluation, the classification hierarchy associated with a new database can be gained through emphasizing some object types that will be evaluated and suppressing other object types that are not involved in the evaluation from an original database. But for land management at one level, the classification hierarchy associated with a new database may be gained through reducing the number of levels of classification hierarchy associated with the original database. Generalization of this type of database not only considers how to organize thematic data and geometric data, but also how to analyze and process them in spatial and semantic aspects. The integrated and extended FDS with CDT (IEFDS) and semantic evaluation model can facilitate such requirements, because FDS has the ability to organize spatial data (including thematic and geometric), and CDT can be used for spatial analysis and semantic evaluation model for evaluating the similarity between objects or object types.

The dataset used for this example is part of a land use database from QiongHai city of Hainan province in P.R.China (see Figure 7.12).

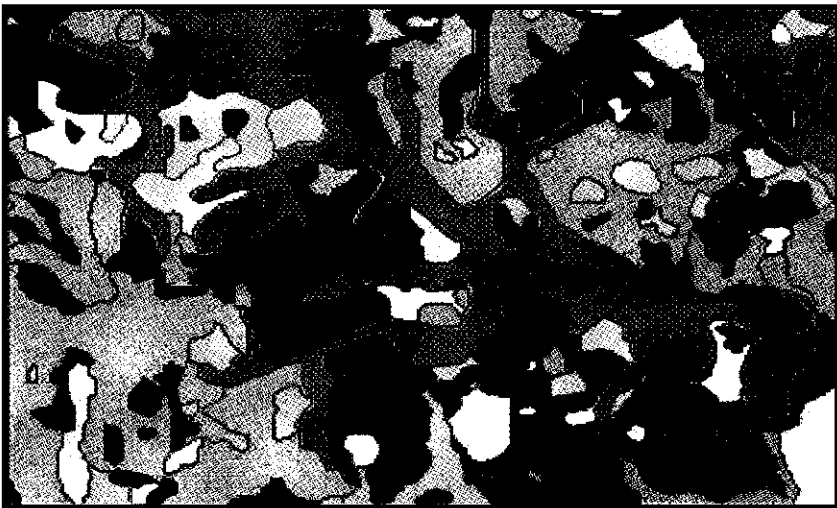


Figure 7.12 A subset of landuse database

The database was established based on a land use map at a scale of 1: 10000. Appendix I gives the codes for land use classification of China. Appendix II gives a semantic similarity matrix based on the involved land use classes and the model described in section 6.3. The land use data have a classification hierarchy at three levels. At the highest level there are three classes (agricultural land, construction land and unused land). At the second level these three classes are divided into seven classes (Cultivated land, Forest, ...), and at the lowest level there are 27 sub-classes (Irrigated paddy fields wood land,...). The database contains 395 area objects. The minimum area of an object in the database is 76 m² and the maximum area is 427946 m². Table 7.2 gives the basic information of the database.

Table 7.2 Basic data of the land use database involved

Sub-Class	No.of obj.	Area	Min. Area	Max.Area
C111	51	2943744	8351	1121978
C112	9	78189	1169	33941
C113	5	70413	1879	46541
C114	10	884473	4351	308591
C115	12	275525	594	74950
C121	32	656554	1327	75123
C122	17	511031	382	90909
C123	15	150211	1682	31657
C124	12	165501	1052	76255
C125	51	1496486	470	428739
C131	27	572369	639	116329
C132	23	721826	869	213893
C134	3	9197	607	6833
C135	3	90175	10914	60971
C151	1	1650	1650	1650
C153	1	48217	48217	48217
C154	10	106293	1236	32192
C156	1	8031	8031	8031
C158	2	140371	2397	137974
C211	22	334702	585	91453
C212	1	1805	1805	1805
C213	5	129498	2012	52919
C310	63	1767778	76	427946
C380	17	332169	464	84461

The data set of land use may be used to answer the questions listed below:

- How to define the new classification hierarchy and aggregation hierarchy;
- How to identify conflicted objects in thematic aspect after classification hierarchy transformation;
- How to identify conflicted objects in geometric and relational aspects;
- How to form the corresponding transformation unit based on conflicted types;
- How to aggregate them within a transformation unit.

The main process consists of:

- Simplifying the classification hierarchy associated with an original database based on the application requirements;
- Thematic transformation based on a new classification hierarchy;
- Aggregating objects based on transformation units that are created by thematically conflicted objects (see Section 6.2.1) as described in Section 6.4.1;
- Detecting conflicted objects in geometric aspects;
- Creating transformation units based on the conflicted objects and the procedures described in Section 6.2.2, 6.2.3, 6.2.4;
- Aggregating the objects using the procedure described in Section 6.4.2, 6.4.3, 6.4.4.

7.3.1 Thematic Transformation

As discussed in method 1 of section 3.4.1.1.1, the first step for categorical database generalization is to define the new classification and aggregation hierarchy. Figure 7.13 shows the classification hierarchy associated with the landuse database in Figure 7.12

Figure 7.14 illustrates the new classification hierarchy associated with a target land use database. In this example, the new classification is achieved through reducing the number of levels of the hierarchy. Based on the new reduced classification hierarchy defined in Figure 7.14, the attributes of the objects from an original land use database will be changed. This thematic transformation may cause thematic conflict of adjacent objects. This means that two adjacent objects belonging to different object types before transformation may belong to the same object type after transformation. Therefore, this will result in the problem that there is a boundary between the objects belonging to the same object type. This will violating the thematic constraints of the object defined in Chapter 4. Figure 7.15 shows t the thematically conflicted objects with a dash line.

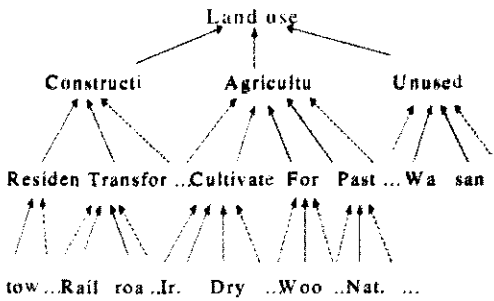


Figure 7.13 Land use classification hierarchy before generalization

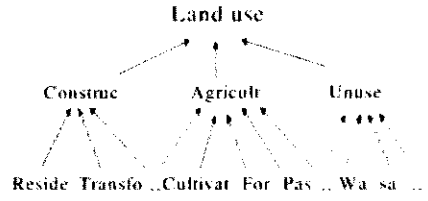


Figure 7.14 Land use classification hierarchy after generalization

7.3.2 Aggregating Objects Based on Thematic Transformation Unit

After establishing a new classification hierarchy, and implementing changed attributes of the objects in a database, a transformation unit must be formed based on thematic conflicts according to the procedure described in Section 6.2.1. The thematic conflict objects shown in Figure 7.15 as seeds with their neighbor objects will form transformation units. The similarity evaluation among objects within a transformation unit must be measured using the similarity evaluation model defined in Section 6.3.2 before object aggregation. The problems of the thematic conflict between spatial adjacent objects will be solved.

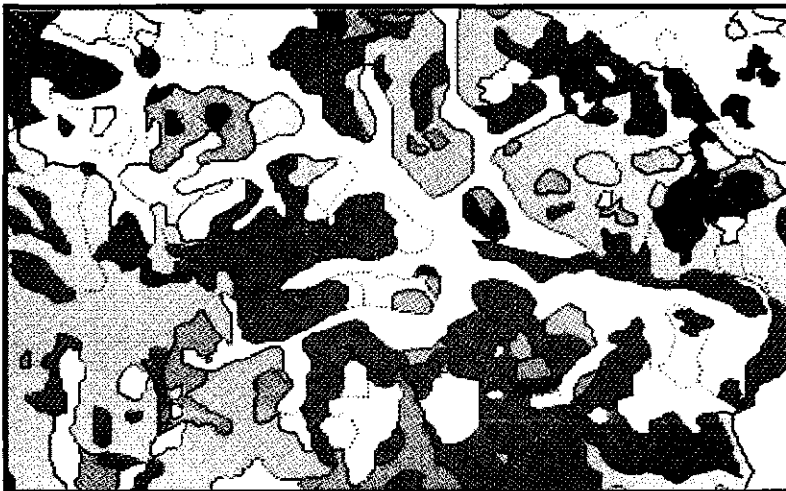


Figure 7.15 Example of conflicted objects with dash line

Figure 7.16 gives the result of object aggregation based on semantic similarity among objects. The thematically conflicted object will be aggregated with the object that has the same object type at the next higher level.

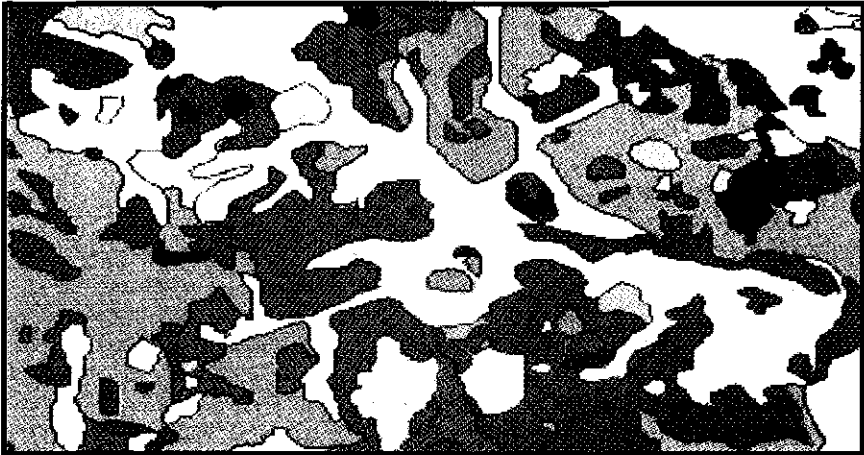


Figure 7.16 Result of object aggregation based on semantic similarity

7.3.3 Detection of Small Geometric Objects

A number of small objects or objects with an area less than the area threshold will be detected first (see Figure 7.17) in order to build the transformation units.

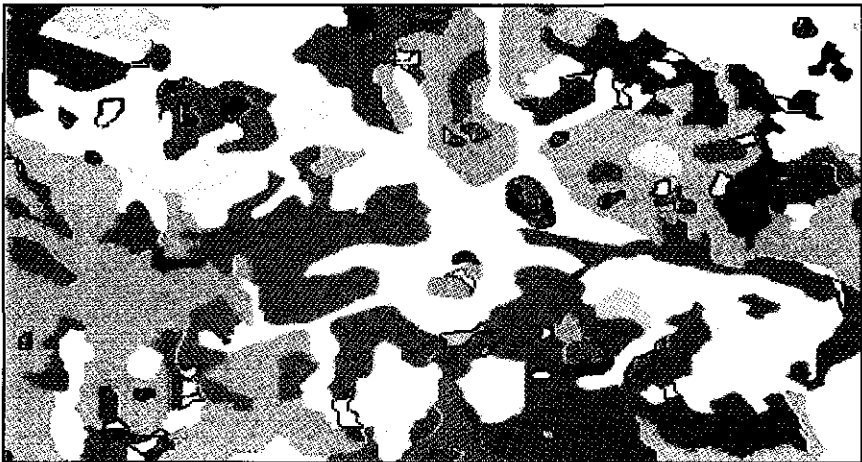


Figure 7.17 Examples of small objects with black boundary

The objects violating the constraints are called conflicted objects. These conflicted objects will be as seeds to form transformation units.

7.3.4 Building Transforming Units Based on Conflicted Objects

After the thematic conflicts among the adjacent (connected) objects have been deleted, the key points for the aggregation process of objects are to form transformation units based on the conflicted objects and to find one or more objects which have higher thematic similarity with the conflicted object in a transformation unit. The different types of transformation units can be created based on the types of the conflicted objects. For a conflicted object with its area less than the threshold area, it and its neighboring objects will form a transformation unit following the procedure described in Section 6.2.2. TU2 in Figure 7.18 represents this type of transformation unit, and only some of the conflicted objects of Figure 7.17 are presented. For two unconnected objects, if the distance between the two objects is less than the distance threshold value, then the two objects and their neighboring objects will form a transformation unit following the procedure described in Section 6.2.3. TU3 in Figure 7.18 represents this type of transformation unit. For a conflicted object with its area less than the threshold area, if the distance between it and the object(s) from its second order neighborhood is less than the distance threshold value, then it and the objects from its first order and second order neighborhood will form a transformation unit following the procedure described in section 6.2.4. TU4 in Figure 7.18 represents this type of transformation unit. The semantic similarity among the objects will be evaluated based on the model within a transformation unit as described in Section 6.3.2.

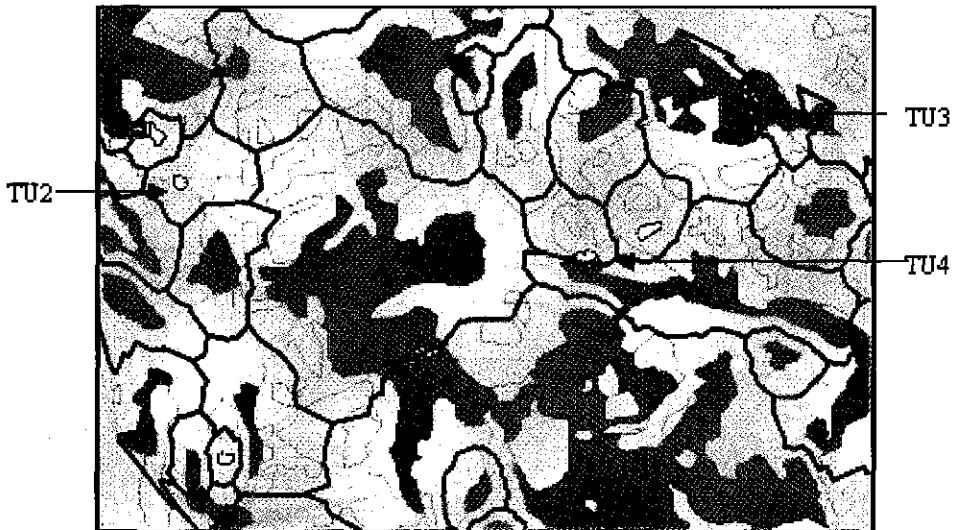


Figure 7.18 Examples of transformation units

7.3.5 Object Aggregation

After building transformation units mentioned above and setting the distance threshold between objects at 100 m and area threshold at $100 \times 100 \text{ m}^2$, the object(s) will be selected within a transformation unit based on having the highest semantic similarity with the conflicted object and aggregation rules in section 6.3.3 and be aggregated using the procedures described in section 6.4.2, 6.4.3 and 6.4.4. Figure 7.19 shows the result of part of landuse database generalization after generalization. It is very important that the ratio of area of each class is kept balanced at the super-class level before and after generalization. The characteristics of spatial distribution of each class are also needed to keep the same before and after generalization. Table 7.3 and Table 7.4 present some results about the generalized landuse database.



Figure 7.19 Example of a subset of landuse database generalization

Table 7.3 lists the number of area objects and area of each class after generalization. Table 7.4 (a) and (b) gives the data of area change of each class after generalization. The generalized land use database contains 239 area objects.

Table 7.3 Number of area objects and area for each class after generalization

Class	Num. of. obj.	Area (m ²)
C110	43	4250559
C120	62	2927244
C130	46	1410811
C150	25	296711
C210	23	432709
C300	40	2178174

Table 7.4 Data of area change for each class after generalization (m²)

Class	Original Area	Removed-Area	Received-Area	Gen-Area	Change-rate(%)
C110	4252334	96845	95133	4250559	0
C120	2979783	142970	90402	2927244	-1.8
C130	1393567	63719	80952	1410811	1.2
C150	304562	20802	12947	296711	-2.6
C210	466005	49287	15983	432709	-7.1
C300	2099947	98123	176329	2178174	3.7

In Table 7.4, the removed area denotes the area of one class changed into other classes and the received area denotes the area of one class gained from other classes. Gen-area denotes the area of each class after generalization. The area of each class after generalization can be calculated by the following equation:

$$G=A+B-C$$

Where: G: area for each class after generalization;

A: area for each class before generalization;

B: received area;

C: removed area.

Change rate of area for each class (%) is defined as the following equation:

$$R=(G-A)/A$$

As shown in Table 7.4, maximum change rate is -7.1% related to c210 since there are a lot of small area objects with sparse distribution that belong to the class. Normally small objects whose area is less than the area threshold will give their area to the objects of other classes after generalization. The area of each class also changes, but slightly (see Figure 7.20).

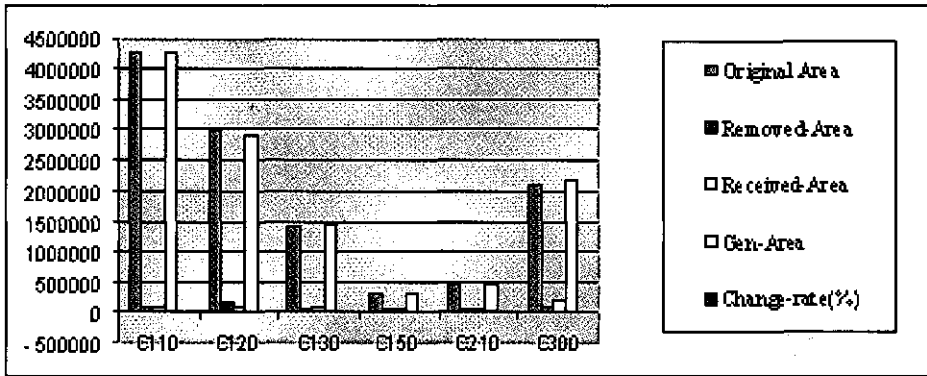


Figure 7.20 Area of each class before and after generalization

Object reduction index can be described as the following equation (after Bregt & Bulers)

$$\text{Object reduction index} = \frac{\text{Reduced number of area objects after generalization}}{\text{number of area objects before generalization}}$$

According to calculation of the equation, the object reduction index is 0.39 for the whole set.

Figure 7.21 shows the original land use database and Figure 7.22 shows the generalized land use database. The rectangle in Figure 7.21 and Figure 7.22 (blue dash line) represents the display range of Figure 7.12.

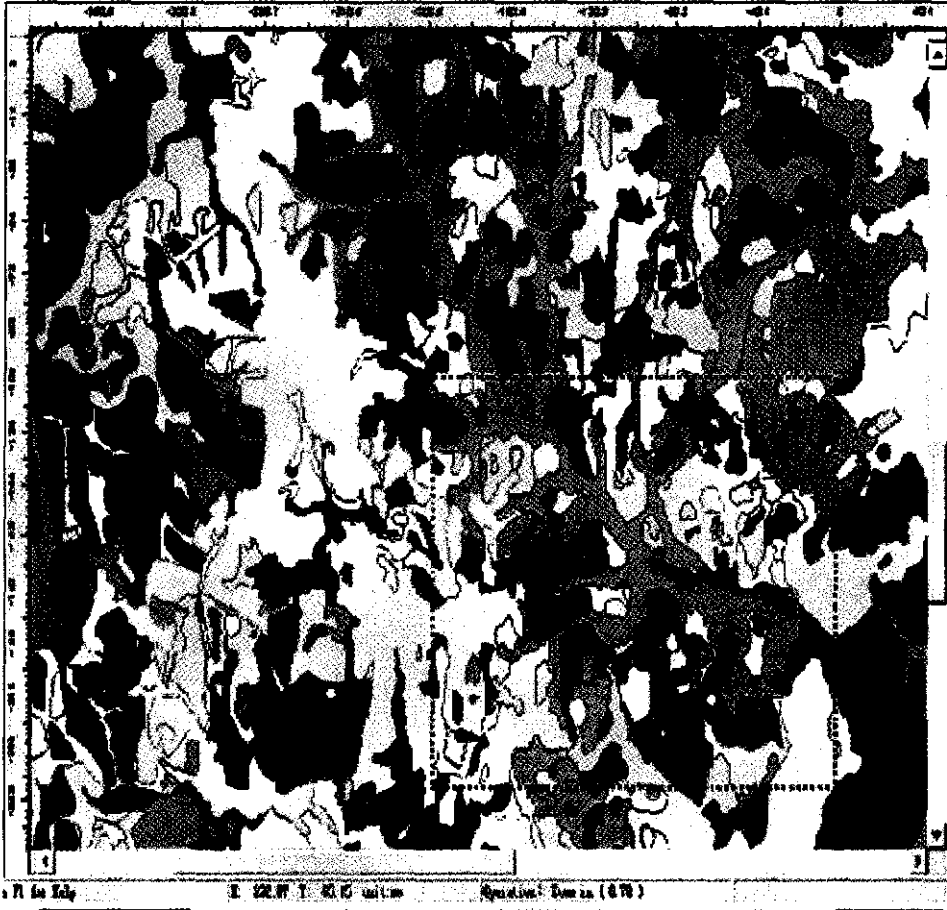
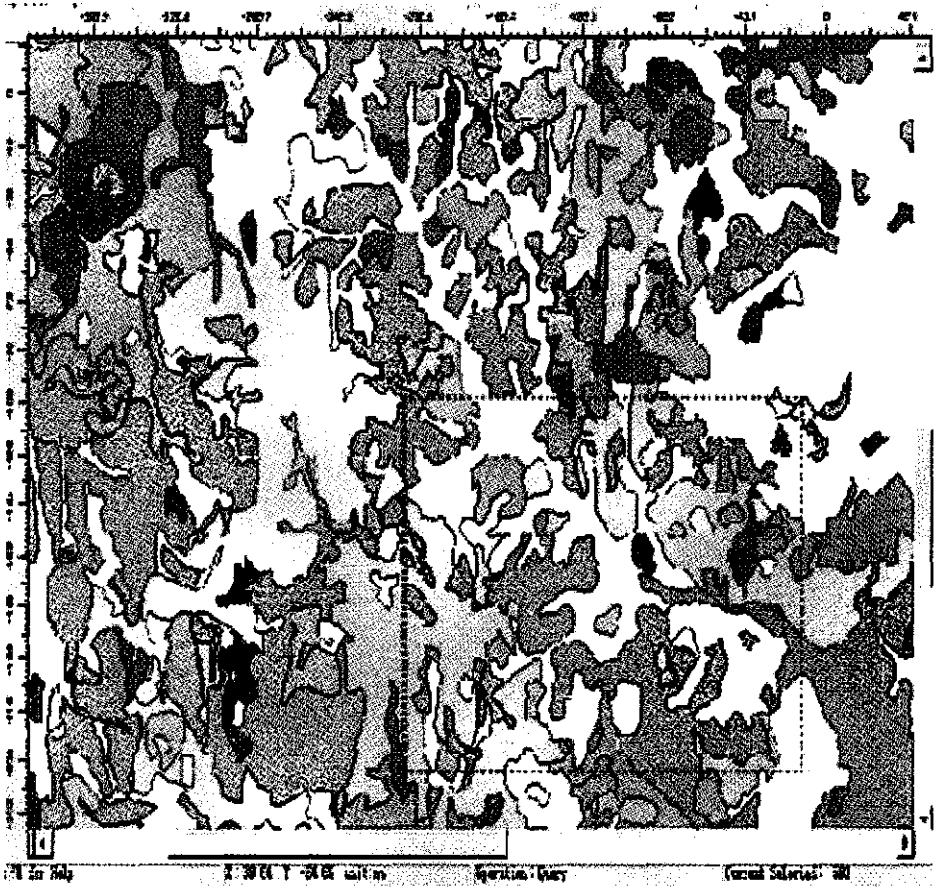


Figure 7.21 Original land use database



Legend







	Cultivated Land		Area of Cities and Town
	Forest		Water Area
	Garden Land		Unused Land

Figure 7.22 Result of landuse database generalization

7.4 Automated Organization of Hierarchical Catchments of River Network

Automated extraction of catchments of a river system play a very important role in generalization and catchment management as discussed in Section 6.6. The catchment area is a very important factor for selecting streams in the generalization of a river system and also an important management unit for hydrology. In this example, the hierarchical catchments of a river system are established mainly through analyzing properties of triangulation networks of different orders of river links of a river system, and the procedure is demonstrated as described in Section 6.6.3. The different types of triangles (see Section 5.3.2.2) and the extracted skeleton lines based on characteristics of the different types of triangles (see Section 5.3.2.3) play a key role in building hierarchical catchments. The experimental data used here is simulated data (see Figure 7.22). The purpose of this example is to test the algorithm described in Section 6.6.3 and illustrate the process of constructing hierarchical catchments of a river system. In the establishment of hierarchical catchments, the following questions have to be worked out:

- How to organize the data of a river system;
- How to build hierarchical catchments through constrained triangulation of a river system;
- How to identify the catchments for each river segment.

The main steps for automated establishment of hierarchical catchments are as follows:

- Organize the river data and classify the river systems for this study based on Horton's classification as described in Section 6.6.1 and section 6.6.2;
- Adjust Horton's classification results;
- Triangulating ordered river systems;
- Constructing catchments of each order link respectively using the procedure described in section 6.6.3.

7.4.1 Ordering River Network

The river systems must be classified before constructing hierarchical catchments. For this example, the river systems are classified based on Horton's classification system and the number of the orders of the whole river system can be gotten through the classification. The number of river orders will be equal to the number of catchments orders. This result of river classification is adjusted based on the following rule in order to extract catchment areas efficiently and effectively, that is: if the order of a link is not the next lower order of its immediate adjacent link except the highest order link, then it will change its order into the next lower order of its adjacent link. Figure 7.23 shows the adjusted result of a simulated river system with 4 levels after Horton's classification.

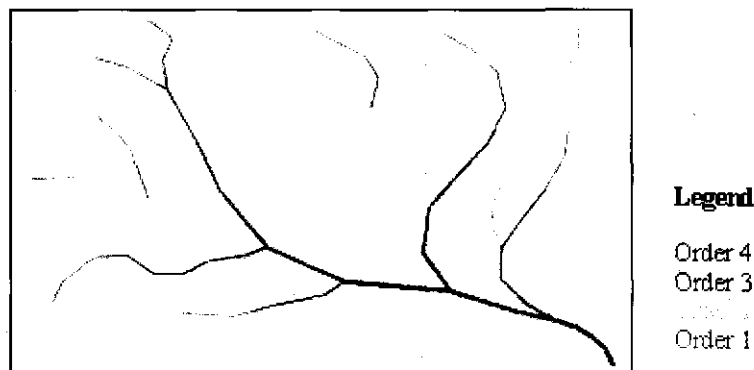
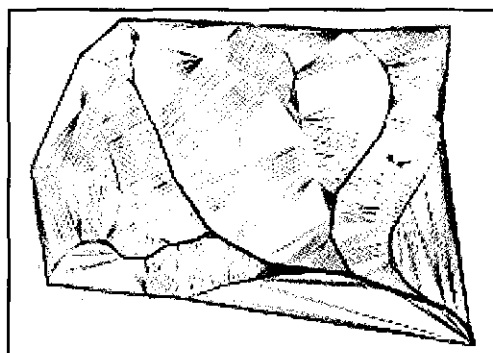


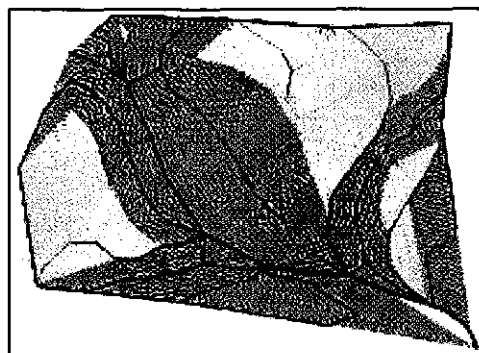
Figure 7.23 Example of ordering river system with stream orders

7.4.2 Constructing Catchments for Each Order River

Triangulating the river system using rivers (links) as constrained edge is shown in Figure 7.24 (a). The area of catchment of the main river (highest order (blue line)) will consist of the hull which is constructed by constrained triangulation network of the river system. The area of the catchments of other order links will be constructed by the following steps: extracting skeleton line as shown by black lines in Figure 7.24 (b), (c) and (d) based on the constructed triangulation network as described in Chapter 5; finding out two skeleton lines which start from the same outlet of the given river link; constructing the area of a given river link based on the two skeleton lines following the procedure described in Section 6.6.3. Figure 7.24 (b), (c) and (d) present the examples of catchments of order 3, 2 and 1 respectively.



(a)



(b)

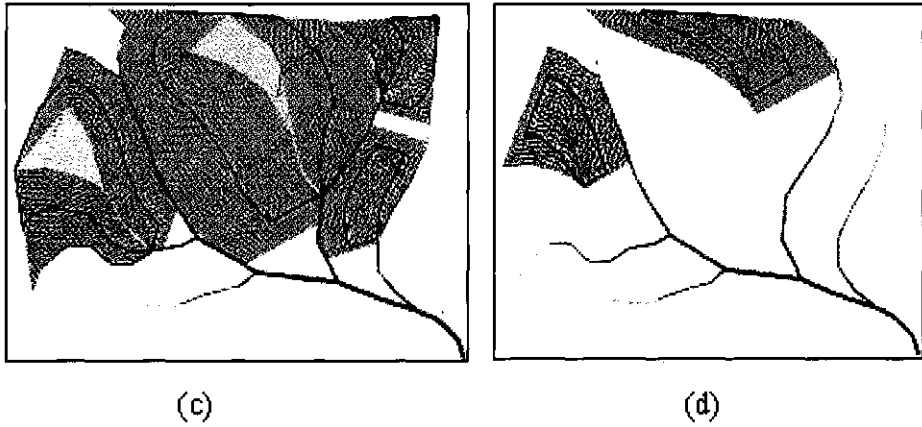
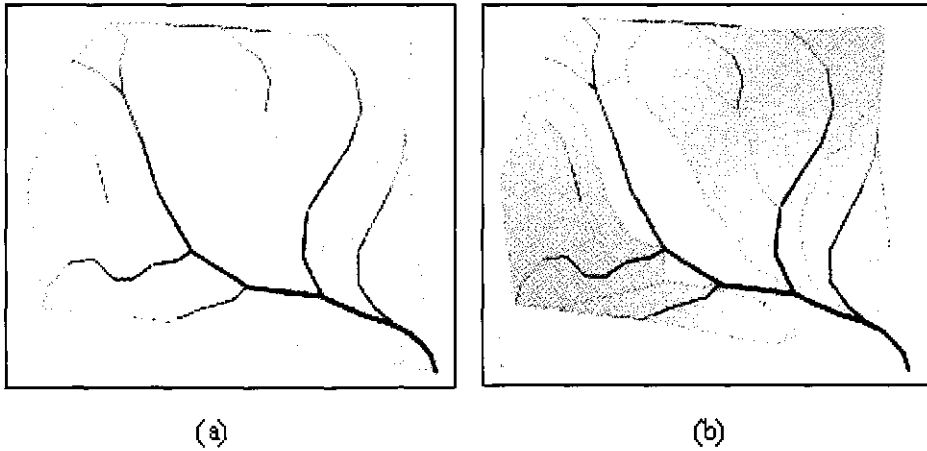


Figure 7.24 Example of catchments of different orders of river links

7.4.3 Hierarchical Catchments

The following figures reflect the characteristics of hierarchical catchments. Figure 7.25 (a) shows the catchment area of river order 4. Figure 7.25 (b) illustrates areas of catchments of river order 4 and 3. Figure 7.25 (c) gives areas of catchments of river order 4, 3 and 2. Figure 7.25 (d) expresses areas of catchments of all four orders.



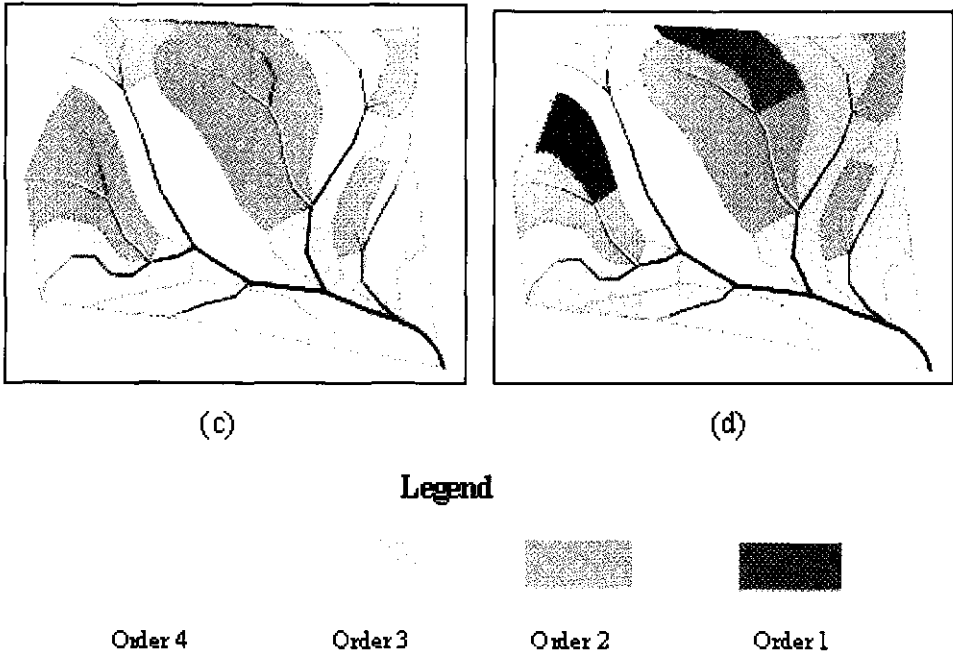


Figure 7.25 Example of hierarchical catchments

7.5 Discussion

The application examples in Section 7.2, 7.3 and 7.4 demonstrate the applicability and benefit of the integration and enhancement of FDS and CDT and similarity evaluation model. These supporting models play a key role in organizing thematic and geometric data, spatial analysis and spatial query in database generalization. They also illustrate the advantage and the power of IEFDS as a supporting data structure in spatial analysis, and it has been proved that by having the support of an adequate data model, a lot of critical geometric and thematic problems in database generalization can be solved, or can be solved in a more efficient way.

In categorical database generalization with a certain application purpose, to a certain extent, the similarity evaluation model decides how to select objects to be aggregated after defining a new geo-spatial model associated with a target database. This will guarantee the thematic quality of the target database.

The change rate value shown in Table 7.4 reflects that the method developed in this study for small sparsely distributed objects is not balanced since the area of the class which the objects belong to loses more than is received from the other classes. That the threshold values for all

classes are the same may not meet the requirements of different applications in which different threshold values for different classes may be needed. For this study, the similitude stream network data is used for the test, and for the real stream network data will be done in the future.

Chapter 8

Conclusions and Future Work

8.1 Introduction

This chapter summarizes the research carried out during the study. The major outcome of the research is briefly outlined in the summary. The chapter draws together the conclusions and recommends future work on the research.

8.2 Summary

Categorical databases as models of (some portion of) a real world have the properties to express the real world at different levels in detail and are used widely in spatial analysis, evaluation, planning, and management. This research concentrates on categorical database generalization. Several important aspects of categorical database generalization have been studied and discussed in the above seven chapters. They include:

- The main problems in categorical database generalization;
- Aspects of categorical database transformation including geo-spatial model transformations, object transformations and relationship transformations;
- Transformation Constraints which guide and control the process of categorical database generalization;
- The data model which supports the categorical database generalization;
- Transformation units which limit the region and number of spatial objects in a categorical database transformation;
- A semantic similarity model which is used to evaluate the similarity among object types in a categorical database and support selection of the most reasonable objects to be aggregated;
- The algorithms for handling thematic and geometric problems in categorical database generalization; and
- The implementation of the algorithms.

Categorical database generalization manipulates mainly geo-spatial models associated with a database, the geometric and thematic descriptions of spatial objects and their relationships

under the control of a new set of constraints which are related to a certain application. The result of transformation is a new database at a less detailed level than the existing one. Categorical database generalization as a transformation process of the contents of a database can derive a new categorical database from a higher resolution database to a lower resolution one. Three kinds of transformation are defined in Chapter 3. It includes geo-spatial model transformation, object transformation and relation transformation.

In the context of a categorical database, geo-spatial model transformation mainly deals with classification hierarchy and aggregation hierarchy associated with a database. Classification hierarchy and aggregation hierarchy are directly related to the content of the database. Geo-spatial model transformation includes a geo-spatial transformation based on classification hierarchy and a geo-spatial transformation based on aggregation hierarchy. The geo-spatial model transformation based on classification hierarchy can occur in many ways such as changing the attribute structure, changing the domain of attribute, changing the cardinality of the object type and changing the sum of the object type. The geo-spatial model transformation based on aggregation hierarchy can occur through specifying rules which are involved in thematic, geometric and topologic aspects such as specifying part of the relation between a composite object type and component object types and establishing the attribute structure of a composite object type. Geo-spatial model transformation mainly determines the theme of a database and what object types and which instances of the object types should be contained in the generalized database. It is application-dependant and determines the framework of contents of a database. Object transformation can be done through instancing new object types, aggregating objects and assigning attributes. Object transformation mainly analyzes and handles geometric and thematic properties of an object. Relation transformation mainly maintains and handles spatial relations among the objects such as deleting proximity relations, deleting inclusion relations, deleting connectivity relations and changing visual connectivity relation into connectivity relation etc.

Changing the geo-spatial model associated with a database will result in changing the structure and the content of the database. Object transformation is caused by geo-spatial model transformation. The changes in the geo-spatial model associated with a database will cause changes in object types, geometric characteristics and attribute values of objects, and result in changing the original spatial relations among objects in the database. Object transformation from a source database to a target database will also result in spatial relation transformation. Changes in the geo-spatial model can be automatically propagated to objects and spatial relations. In contrast to changing the geo-spatial model, object transformation does, in general, not lead to a change in the geo-spatial model. Before implementing object transformation and relation transformation, the geo-spatial model of the new database must be defined and transformed. Object transformation and relation transformation are implemented based on the defined geo-spatial model.

Not all transformations defined in Chapter 3 have been tested, some work will be done in the future.

It is important to note that categorical database generalization transformation is based on conditions that are called transformation constraints in this study. Categorical database transformations need constraints to control and guide the process of transformation. Three types of constraints have been proposed in database generalization. They are geo-spatial model constraints, object constraints and relation constraints. Geo-spatial model constraints, which define the new classification hierarchy and aggregation hierarchy with a target categorical database, can be divided into classification hierarchy constraints and aggregation hierarchy constraints. Object constraints include thematic constraints and geometric constraints. Relation constraints contain topological constraints, direction relation constraints, distance relation constraints and semantic relation constraints. Constraints can be specified interactively by users and varied to reflect different objectives or purposes. The constraints play an important role in categorical database transformation. They provide steering parameters for how many object types and objects should appear in a database. They are also used as parameters to detect and identify areas and objects that violate the conditions. These conflicted objects are used as the seeds to create transformation units in object transformation. In this research, focus on trigger constraints, outcome constraints requires a further study.

The transformation unit proposed in Chapter 4 in this study is an important process unit as many generalization problems need to be solved by considering spatial cluster related to objects as a whole, rather than treating them individually. Transformation units can be created based on constraints. Four types of transformation unit are identified in this study. They are transformation units based on thematic constraints, transformation units based on geometric constraints, transformation units based on spatial relation constraints and transformation units based on geometric and spatial relation constraints. Transformation units limit the region to be processed and limit the number of objects. They can trigger aggregation operations and control the process of transformation. They allow us to group objects according to their characteristics, and potential behaviors, in a categorical database generalization.

In the transformation operations, the basic operations related to categorical database transformation can be categorized into two types. One is operations on the geo-spatial model or object type associated with a database which will lead to changes at the abstraction levels; the other is the operations on objects in the database.

Database transformation is a complicated process of spatial analysis, decision-making and implementation on spatial objects. Data structure (or model) is essential for defining and operating generalization operators or procedures. Data structure in the database generalization can strongly support data organization, spatial analysis and decision-making in the process of transformation. So FDS is introduced particularly for this purpose. This data model combines aspects of object-oriented and topological data models. Its geometric representation contains information about topological object relationships, whereas its thematic description is structured on object types that may form generalization hierarchies. Such classification hierarchies in combination with the topological object relationships of FDS support the definitions of aggregation hierarchies of objects. By introducing the Delaunay triangulation network, we could formulate (and utilize) an extended set of adjacency relationships and inclusion relations

which are important for decision-making, and the implementation of generalization operations. In order to enhance the analysis function of the data model in database generalization, FDS has been integrated and extended in the sense of adjacency and inclusion relationships, which are particularly important in automated database generalization. This integrated and extended FDS is called IEFDS. In this model, triangles are divided into four types: triangles with no constrained edge at its three edges, triangles with only one constrained edge in its three edges, triangles with two constrained edges at its three edges and triangles with three constrained edges at its three edges. Each type of triangle can be further subdivided according to the constituents of different point, line and area objects in a triangle. They play an important role in organizing spatial data, detecting spatial conflicts (objects), identifying neighbors, forming transformation units, aggregating operations and implementing database generalization transformation such as extraction of skeleton lines. IEFDS in combination with classification hierarchy and aggregation hierarchy play an important role in linking the definition of objects at several levels, as well as spatial analysis and operations in database generalization. The process of the integration between FDS and CDT is discussed in Chapter 5. The concept of nearness degree among objects is proposed in Chapter 6. Some examples of spatial query operations that make use of IEFDS are presented but the aspect of consistency in the model was not discussed in the thesis.

•

After having the data model to support the description of spatial objects, and the topological relationships among them, we still need an algorithm to actually perform the analysis and transformation. So, Chapter 6 describes a number of algorithms which have been developed to handle the important thematic and geometric problems in database generalization. Such problems as:

- Creation of transformation unit based on different kinds of constraints;
- Aggregation operations based on different types of transformation units;
- Similarity evaluation model;
- Skeleton line;
- Object cluster;
- Multi-neighborhood;
- Creating hierarchical catchments of river network

Algorithms are based on IEFDS. These algorithms provide us with an efficient and useful means to transform the categorical database from higher resolution to lower resolution. Algorithms described in Chapter 6 were tested in Chapter 6 and Chapter 7.

Whether two objects can be aggregated or merged not only depends on spatial relations and geometric properties between them, but also on the semantic similarity between them. The degree of similarity of objects can be described by a similarity index and the degree of similarity of object types can be described by the similarity. The similarity among object types is application-dependent and can be represented by a semantic similarity matrix. The values of elements in the matrix reflect the degree of similarity. The larger the value, the more similar two object types are. The value can be given by expert knowledge or the calculation based on the aggregation hierarchy, classification hierarchy, requirements and purposes of database generalization. A computing model for similarity based on Set-theory, classification hierarchy and aggregation hierarchy is proposed in this study. Similarity between object types plays a key role in selecting objects to be aggregated in the aggregation process.

Having designed the integrated and extended formal data structure and Delanuy triangulation network, transformation unit and semantic evaluation model, the demonstration of some application has been implemented. The data used in the examples are from both a real data set and simulated data. The examples of the application include:

- Object clustering;
- Land use generalization; and
- Automated organization of hierarchical catchments of river system.

The results proved that the designed methods and algorithms are reasonable and feasible in the categorical database transformation.

8.3 Conclusions

On the basis of various issues addressed in this research, the following conclusions can be drawn:

- Database generalization as a transformation can be done through geo-spatial model transformation, object transformation and relation transformation. Geo-spatial model transformation determines the theme of a database and what object types should be contained in the database. In a sense, it determines the content of a database. Object transformation and relation transformation are essential components of database transformation.
- The process of deriving a new database is a transformation from one existing state of a database at a certain detailed level to a new state at a less detailed level according to the application and user's requirements. The state of the database can be specified by a four tuple. The tuple consists of geo-spatial models, a set of objects, a set of relations and a set of constraints.

- An adequate supporting geo-spatial model can provide a description of object types and the relationships among them. IEFDS, an integrated and extended version of FDS as a supporting data model plays a very important role in organizing spatial data, spatial relation analyzing and querying spatial relations, detecting spatial conflicts (objects), identifying neighbors, creating transformation units, aggregating operations and implementing database generalization transformation.
- The operations in categorical database transformation are divided into two types. One is on the object types in the database; the other is on the objects in the database.
- The transformation unit as a basic analysis, processing and decision-making unit plays an important role in database transformation. It limits the region of the objects and the number of objects to be processed. Each of four types of transformation unit can not only cluster related objects but also trigger different aggregation operations.
- Three types of constraints have been used in this study:
 - Constraints on geo-spatial model;
 - Constraints on objects; and
 - Constraints on relations.
- The classification of constraints fully reflects the characteristics of categorical database transformation. Constraints control and guide the creation of the transformation unit and transformation process.
- Before objects are aggregated, semantic similarity among objects must be evaluated in categorical database generalization. The similarity matrix and similarity evaluation model which has been proposed can be applied to reach the requirements.
- The results of tests proved that IEFDS as a supporting data model is effective and efficient, and the designed methods and algorithms are reasonable and feasible.

8.4 Future Work

This research does not address deeply all aspects related to categorical database generalization and all issues of database generalization. There are still some issues that need to be investigated, and some of the aspects dealt with in this research still need further study and development. The main areas for future work directly relevant to the research are as follows:

- Further investigation on completeness of types of transformation unit related to database generalization.

- Improving the semantic evaluation model for more expert knowledge use. The similarity computing model can provide the similarity value through computing distance between object types within a sub-tree. But there is still a lack of the similarity computing model between object types which belong to different sub-tree respectively.
- Implementing multi-resolution transformation since the requirements on thematic and geometric resolution of the different object types associated with a database are different for particular applications.
- Improving the algorithm for automated organization of hierarchical catchments of a river system. The designed algorithm only takes geometric factors into account in constructing the catchment area. The algorithm should be powerful if some thematic factors such as slope and aspects are added into the algorithm.
- Investigating further trigger and outcome constraints, which play a key role in triggering and stopping transformation operations.
- Investigating the mechanism of controlling parameters (constraints) which control the final results of database transformation.

References

- Alexandroff, p., 1961, *Elementary concepts of topology* (New York: Dover Publications, Inc).
- Beard M. K. 1988 *Multiple representations from a detail database: A scheme for automated generalization*. Ph.D Thesis, University of Wisconsin, Madison.
- Beard, K. and Mackaness W. 1991, *Generalization Operations and Supporting Structures*, AUTO-CARTO 10, pp.29-42.
- Beard, K., 1991, *Constraints on Rule Formation*, in *Map Generalization: Making Rules for Knowledge Representation* (eds Buttenfield, B. P. and McMaster, R. B.), Longman House Essex, United Kingdom, pp.121-135.
- Beat, P. and Weibel, R., 1999, *Integrating Vector and Raster-Based Techniques in Categorical Map Generalization*, *Third ICA Workshop on Progress in Automated Map Generalization*, Ottawa.
- Bishr, Y., 1997, *Semantic aspects of interoperable GIS*. ITC, Publication No. 56, Enschede: International Institute for Areospace Survey and Earth Sciences (ITC).
- Bo Su, LI,Z. , Lodwick, G.and Muller,J.C 1997 *Algebraic models for the aggregation of area features based upon morphological operators*. *International Journal of Information Science*, Vol.15,No3 pp. 233-246.
- Bouloucos, T., Kufoniyi, O. and Molenaar, M., 1990, *A relational data structure for single valued vector maps*, *International Archives of Photogrammetry and Remote Sensing*, 28, (3/2), 64-74.
- Brassel, K.E. and Weibel, R., 1988, *A view and conceptual framework of automated map generalization*, *International Journal of Geographic Information Systems*, 2,229-244.
- Bregt, A. and Bulens, J., 1996, *Application-oriented generalization of area objects*. In molenaar,M. (ed), *Methods for the Generalization of Geo-databases*, Delft: Netherlands Geodetic Commission, New Series, Nr.43,pp. 57-64.
- Bressel, K.E. and Weibel, R. 1988, *A review and conceptual framework of automated map generalization*. *International Journal of Geographic Information Systems*, Vol. 2, pp.229-244.
- Bruns,H.T. and Egenhofer M, (1996) *Similarity of spatial scenes* . In Kraak, M.J. and Molenaar,M. (eds.), *Advances in GIS Research II*, London:Taylor & Francis, pp.215-225.

References

Bundy,G.L and Jones, C.B. and Furse, E. 1995, Holistic generalization of large-scale cartographic data. in *Gis and Generalization* (eds. Muller, J.C., Lagrange, J.P. and Weibel, R.), pp.106-119.

Bundy,G.L., Jones, C.B. and Furse E. 1995, Holistic Generalization of large-scale cartography data, in *GIS and Generalization* (eds Muller J.C., Lagrange, J.P. and Weibel, R.), pp. 106-116.

Burrough, P.A. and McDonnell, R.A., 1998, *Principles of Geographic Information Systems*, Oxford University Press.

Buttenfield, B., 1989, Scale dependence and self similarity in cartographic line, *Cartographic* 26 (2). Pp.79-100.

Buttenfield, B.P. and McMaster, R.B. (eds.) 1991, *Map generalization: Making Rules for Knowledge Representation*. London, Longman., pp.217-227,.

Chakroun,H., Benie,G.B., Oneill,N.T., and Desilets,J. 2000, Spatial analysis weighting algorithm using Voronoi diagrams. *International Journal of Information Science*, Vol.14,No.4 pp.319-336.

Chrisman, S. 1982, *Methods of spatial analysis based on error in categorical maps*. Ph.D thesis, University of Bristol,England.

Clementini E. Di Felice P., Van Oosterom, 1993, A small set of formal Topological relationships suitable for end-user interaction, *Proceedings of the third international symposium on advances in spatial databases, SSD'93*, pp.227-295. Springer Verlag, Lecture Notes in Computer Science #692,

Colodoven A. Davis Jr and A.H.F. Laender 1999, Multiple representations in GIS: materialization through map generalization, geometric, and spatial analysis operations, *ACM GIS 99*, Kansas City , MO USA.

Delaunay, B. 1934, *Bulletin of the Academy of science of the USSR, Classe des Sciences Mathematiques et Naturelles*, 8, pp.793-800.

Douglas,D., and Peucker T. 1973, Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Canadian cartographer*, 10 (2), pp.112-122.

Dutta,S., 1988, Approximate spatial reasoning. In *Proceeding of first international conference on industrial and engineering applications of artificial intelligence and expert systems* ,pp.126-140.

Edwards, G., 1994, aggregation and disaggregation of fuszy polygons for spatial-temporal

References

modeling. In molenaar, M. and Hoop, S. (eds.), *Advanced Geographic Data Modelling*, Delft: Netherlands Geodetic Commission, New Series, Nr 40, pp. 141-154.

Egenhofer, J.M. 1989, A formal definition of binary relationships, In third International conference of foundation of data organization and algorithms (FODO), Paris, France, Edited by W. Latwin and H.J. Schek (New-York: Springer-Verlag), *Lecture Notes in computer Science*, vol. 367, pp.457-472.

Egenhofer, M.J. and Herring, J.R., 1990, A mathematical framework for the definition of topological relationships. In Brassel, K. and Kishimoto, H. (Eds.), *Proceedings of 4th International Symposium on Spatial Data Handling*, Zurich, pp.803-813.

Egenhofer M. and Franzosa, 1991, point-set topological spatial relations, *International Journal of Geographical Systems* 5(2), pp.161-174.

Egenhofer, J.M. and Franzosa, D.R. 1991, *Point-set topological spatial relations*, *IJGIS*, Vol.5. no.2 pp.161-174.

Egenhofer, M.J., and Al-Taha, 1992, Reasoning about gradual changes of topological relationships. IN: A. Frank, I. Campari, and U. Formentini (Eds.) *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*. *Lecture Notes in Computer Sciences* 639. Pp.196-219, Springer-Verlag, New York.

Egenhofer, J.M. and Sharma, J., 1993, Assessing the consistency of complete and incomplete topological information, *Geographic Systems*, Vol.1, pp.47-68.

Egenhofer, J.M. and Sharma, J., 1993, Topological relations between regions in R^2 and Z^2 , In the *Proceedings of 3rd International symposium (SSD'91)*. *Advances in Spatial databases*, Edited by D. Abel, Ooi, *Lecture Notes in Computer Science*, Vol.692, pp. 316-336.

Egenhofer M. and Franzosa, 1995, On the equivalence of topological relations. *International Journal of Geographical Information Systems* 9(2), pp.133-152.

Egenhofer, J.M. and Clementini, E. and Felice, P., 1994, Topological relations between regions with holes, *IJGIS*, Vol.8.no.2 pp.129-142.

Egenhofer M. and Mark, D. 1995, Modelling conceptual neighborhoods of topological region relations. *International Journal of Geographical Information Systems* 9(5), pp.555-565.

ESRI, 2000, *Map generalization in GIS: practical solutions with workstation arcinfo software*. An esri write paper.

Frank Andrew U.: 1991, *Qualitative Spatial Reasoning with Cardinal Directions*. ÖGAI 1991:

References

- Frank, A.U. 1992. 'Qualitative Spatial Reasoning about Distances and Directions in Geographic Space'. In *Journal of Visual Languages and Computing*, 3, pp: 343-371.
- Frank, A.U. 1996. Qualitative Spatial Reasoning: Cardinal Directions as an Example'. In *IJGIS*, 10 (3), pp: 269-290.
- Frank, A.U., Volta, G.S., & McGranaghan, M. 1997. 'Formalization of Families of Categorical Coverages'. In *IJGIS*, 11 (3), pp: 215-231.
- Garbrecht, J. and Martz, L. 1997, automated channel ordering and node indexing for raster channel networks. *Computers and Geosciences*. Vol.23 No.9, pp.961-966.
- Gold, C.M., 1989 Spatial adjacency- a general approach, *Proceeding AutoCarto 9*, pp. 298-308.
- Gold, C.M., 1992, the meaning of neighbor. In Frank, A.U. (eds). *Theories and methods of spatio-Temporal Reasoning in geographic space*, Lecture notes in computing science No639, Berlin: Springer-Verlag, pp. 220-235.
- Goodchild, C.F. 1995 Future directions for Geographic Information Science, *Proceedings of GeoInformatics'95. International Symposium on RS, GIS & GPS in Sustainable Development and Environmental and Monitoring*, Hong Kong, pp.1-9.
- Gray, S. Volta and Egenhofer, M.J. 1995, Interaction with GIS attribute data based on categorical coverages. *European conference on spatial information theory*, Marciana, Italy, A.Frank and I.Campari (eds.) *Lecture Notes in Computer Science*, Vol.716, Springer-Verlag, pp.215-233.
- Guttman, A. 1984 A dynamic index structure for spatial searching, *Proceedings of the SIGMOD conference*, Boston, pp.47-57.
- Hong, J.H. 1994 Qualitative distance and direction reasoning in geographic space. Ph.D. Thesis, Department of Surveying Engineering, University of Maine, Orono.
- Horton, R.E., 1945, Erosional development of streams and their drainage basins, hydrophical approach to quantitative morphology. *Bulletin of the Geological society of America* 56. 257-370.
- Hughes, J.G. 1991, *Objects-oriented Databases*, New York: Prentice Hall.
- Huisig, J. 1993, *Land Use Zones and Land use Patterns in the Atlantic Zone of Costa Rica*. Doctoral thesis, Wageningen: Wageningen University.
- Jaakkola, O., 1998, Multi-scale categorical database with automatic generalization transformation based on map algebra. *Cartography and Geographic Information Systems* 25(4):

References

195-207.

Jones C.B. and Luo, L.Q., 1994, Hierarchies and objects in a deductive spatial database. In Waugh, T.C. and Healy, R.C. (Eds.), *Advances in GIS Research*, London: Taylor & Francis, pp.588-603.

Jones, C.B., 1991, Database architecture for multi-scale GIS, In proceedings of database. In *Proceedings of Auto-Carto 10*, Bethesda: ACSM & ASPRS, pp.1-14

Jones, C.B., Bundy, G.L., and Ware, J.M. 1995, Map generalization with a triangulated data structure. *Cartographic and Geographic Information systems* 22(4) 317-331.

Jones, C. B. and Ware, M.J., 1998, Nearest neighbor search for linear and polygonal objects with constrained triangulations, 8th International Symposium on Spatial Data Handling, Canada pp. 13-19.

Kainz, W., 1990, Spatial relationships- topology versus order, In Brassel, K. and Kishimoto, H. (Eds.), *Proceedings of the 4th International symposium on spatial data handling*, Zurich: University of Zurich, pp. 814-819.

Kemppainen, H. H., 1992, Handling integrity constraints of complex objects in spatial databases. In *International Archives of Photogrammetry and Remote Sensing, Com.III XXIX, Part 3*, Washington, DC: ASPRS, pp.456-464.

Kilpelainen, T., 1992, Multiple representations and knowledge-based generalization of geographic data. In *International Archives of Photogrammetry and Remote Sensing, Com.III XXIX, Part 3*, Washington, DC: ASPRS, pp. 954-964.

Kilpelainen, T., 1995, Updating multiple representation geodata bases by incremental generalization. *Geo-Informationssysteme*, *8. 13-18.

Kilpelainen, T., and Sarjakoki, T., 1995, Incremental generalization for multiple representations of geographical objects. In Muller, J.C., Lagrange, J.P. and Weibel, R. (Eds.) *GIS and Generalization*, London: Taylor & Francis, pp. 209-218.

Kufoniyi, O and Pilouk, M., 1994, A vector data model integrating multitheme and relief geoinformation, proceedings of SDH'94, Vol.2, pp1061-1071.

Lang, D., Winter, S., & Frank, A.U. 2001. 'Neighborhood relations between fields with applications to cellular networks'. In *GeoInformatica*, 5 (2), pp: 127-144.

Langram, G. 1991, Generalization and parallel computation, In: Buttenfield, B.P. and

Laurini, R. and Thompson, D., 1993, *Fundamentals of Spatial Information Systems*, London:

References

Academic Press.

Lin, D. 1998, An Information theoretic definition of similarity (eds.) International conference on machine learning, ICML'98.

Liu Yaolin, Martien Molenaar, Menno-Jan Kraak, 2001, Spatial Object Aggregation Based on Data structure, Local Triangulation and Hierarchical analyzing method. Proceedings (CD-ROM) of 20th International Cartographic Conference 2001, Aug.6-10. Beijing, China.

Liu Yaolin, Martien Molenaar, Ai Tinghua 2001, Frameworks for Generalization Constraints and Operations Based on Object-Oriented Data Structure in Database Generalization. Proceeding (CD-ROM) of 20th International Cartographic Conference 2001, Aug.6-10.2001. Beijing, China

Liu Yaolin, Martien Molenaar, 2001 Generalization transformation based on constraints, spatial and relationships. ASPRS Annual Conference Proceedings (CD_ROM), Washington, DC. Society of America Bulletin, v.84 .ASPRS 2001 Annual Convention, St, Louis, Mo, April 23-27.

Macedonio G. and Pareschi, M.T., 1991, An algorithm for the triangulation of arbitrarily distributed points: applications to volume estimate and terrain fitting. Computers & Geosciences Vol.17.No.7,pp.859-874.

Mackaness, W. 1991, Map generalization, Integration and Evaluation of map generalization, In:

Mackaness, W. 1994 An algorithm for conflict identification and feature displacement in automated map generalization, Cartography and GIS 21(4).

Mackaness, W. and Beard, K. 1993, Use of Graph theory to support map generalization, Cartography and Geographic Information Systems, (20(4) 210-221,.

Mackaness, W. 1994, Knowledge of the synergy of generalization operation in automated map sign, the Canadian Conference on GIS Proceedings, Vol.1 Ottawa.

Mackaness, W.A. 1995, Analysis of urban road networks to support cartographic generalization, Cartography and Geographic Information Systems, Vol.22, No.4, pp.306-316.

Mackaness, W. and Purves, R. 1999, Issues and solution to displacement in map generalization, CD-Rom Proceedings of 19th International Cartographic conference, Ottawa.

Mark, D.M. 1991, Object modeling and phenomenon-based generalization, In Map generalization: Making rules for Knowledge Representation (eds Buttenfiled, B.P. and McMaster, R.B.), Longman House Essex, United Kingdom, pp.103-118.

References

- Martinez Casanovas, J.A., 1994, Hydrographic Information Abstraction for Erosion Modeling at a regional level. MSC. Thesis, Department of land surveying and remote sensing. Wageningen: Wageningen Agricultural University.
- Mazur, E. Rusak and Castner, H.W., 1990, Hortons's ordering scheme and the generalization of river networks. *The Cartographic Journal*. Vol. 27. pp. 104-112.
- McMaster R.B., 1987, Automated line generalization, *Cartographica*. 24(2), pp. 74-111.
- McMaster, R and Mark M. 1989, A conceptual framework for quantitative and qualitative raster-mode generalization. *Proceedings of America congress of surveying and mapping*.
- McMaster, R and Shea, S. 1989, Cartographic generalization in digital environment: A framework for implementation in a geographic information system. *Proceeding of America congress of surveying and mapping*.
- McMaster, R. 1991, Knowledge acquisition for Cartographic generalization, *Proceedings of ICA 91*, 1991.
- McMaster, R. B., 1991 Conceptual frameworks for geographical knowledge, in *Map Generalization: Making Rules for Knowledge Representation* (eds Buttenfield, B. P. and McMaster, R. B.), Longman House Essex, United Kingdom, pp. 21-39.
- McMaster, R.B. 1991, (eds.) *Map generalization: Making Rules for Knowledge Representation*. London, Longman., McMaster, R.B.), Longman House Essex, United Kingdom, pp. 1-20.
- McMaster, R., and Barnett, L., 1993, A Spatial-Object Level Organization of Transformations for Cartographic Generalization, *AUTO-CARTO 11*, Minnesota, pp. 386-393.
- Moise, E. 1977, *Geometric topology in dimension 2 and 3*, Springer-Verlag, New York.
- Molenaar, M., 1989, Single valued vector maps- a concept in GIS. *GeoInformationssysteme*, 2, 18-26.
- Molenaar, M., 1991 Terrain objects, data structures and query spaces, in *Geo-Informatik*, (ed Schilcher, M.), Siemens-Nixdorf Informationssysteme A.G., Munchen, 1991, pp. 53-70.
- Molenaar, M. 1993 Object hierarchies and uncertainty in GIS or why is standardisation so difficult, *GeoInformations-Systeme*, Vol. 6, No. 3, pp. 22-28.
- Molenaar, M., and Richardson, D.E. 1994, Object hierarchies for linking aggregation levels in GIS, *Proceedings of the Symposium of ISPRS Comm. IV*, Athens, Georgia USA, pp. 610-617.

References

- Molenaar, M. 1995 Spatial concepts as implemented in GIS. In Frank, A.U. (ed.) *Geographic Information Systems—Materials for a Post Graduate Course*, Vienna: Department of Geoinformation, Technical University Vienna, Volume 1, Chapter 1,91-154.
- Molenaar, M., 1995, An introduction into the theory of topologic and hierarchical object modeling in Geo-Information Systems, Wageningen Agricultural University, the Netherlands.
- Molenaar, M., 1995, Topological and hierarchical spatial object modeling for multiple scale representations in GIS, *Geotechnica*, Koln,'95,15-18.
- Molenaar, M.,1996, The Role of Topologic and Hierarchical Spatial Object Models in Database Generalization. In Molenaar,M.(Eds.) *Methods for the Generalization of Geo-databases*, Delft: Netherlands Geodetic Commission, New Series, Nr 43,pp.13-36.
- Molenaar, M. 1996, Multi-scale approaches for geo-data, *International Archives of Photogrammetry and Remote Sensing*, Vol. XXXI, Part B3, Vienna, Austria, pp. 542-554.
- Molenaar, M and Martinez Casasnovas, G.A.,1997, A formalism for the structural description of vector maps and its use for multi-scale representations: a hydrographic example. *Cartographica*, 33, 55-63.
- Molenaar, M., 1998,*An Introduction to the Theory of Spatial Object Modeling for GIS*, Taylor & Francis Ltd. London.
- Muller,J.C. 1987, Fractal and automated line generalization, *The Cartographic Journal*, Vol.24, pp.27-34
- Muller,J.C.1989, Theoretical considerations for automated map generalization. *ITC Journal*, 3/4, pp. 200-2-4.
- Muller,J.C.,1990, Generalization of spatial data database. In *Geographic Information Systems* (eds Maguire, D.J., Goodchild, M.F. and Rhind, D.W.) Vol.1, Longman Scientific & Technical Ltd, New York, pp. 137-475.
- Muller, J. C. 1991 Generalization of spatial data bases, in *Geographic Information Systems* (eds Maguire, D. J., Goodchild, M. F. and Rhind, D. W.), Vol. 1, Longman Scientific & Technical Ltd, New York, pp. 457-475.
- Muller, J.C., and Zeshen, W., 1992, Area-patch generalization: a competitive approach. *Cartographic Journal*, 29, 137-144.
- Muller, J.C., Weibel, R., Lagrange, J.P. and Salge, F. 1995, Generalization: state of the art and issues, in *GIS and Generalization* (eds Muller, J.C., Lagrange, J.P. and Weibel, R.), Taylor & Francis, pp. 3-17.

References

- Nickerson, B.G. and Freman, H. 1986, Development of a rule-based system for automatic map generalization, Proceedings of the second international symposium on spatial data handling, pp.537-556.
- Nickerson, B.G. 1991, Knowledge engineering for generalization. In Map generalization: Making rules for Knowledge Representation (eds Buttenfield, B.P. and McMaster, R.B.), Longman House Essex, United Kingdom, pp.40-55.
- Nyergers, T.L. 1991, Representing geographical meaning. In Map generalization: Making rules for Knowledge Representation (eds Buttenfield, B.P. and McMaster, R.B.), Longman House Essex, United Kingdom, pp. 59-85.
- Oosterom V.P. and Schenkelaars V. 1993 The design and implementation of a multi-scale GIS, Proceedings of EGIS'93, pp. 712-721.
- Oosterom, V.P., 1995, The GAP-tree, an approach to 'to-the-fly' map generalization of an area partitioning. In Muller, G.C, Lagrange, J.P., and Weibel, R. (ed.) GIS and Generalization methodology and practice. Taylor & Francis, pp.106-119.
- Oosterom V.P. and Schenkelaars V. 1996 Applying reactive data structure in an interactive multi-scale GIS, in *Methods for the generalization of geo-databases* (ed. M.Molenaar), The Netherlands Geodetic Commission, New Series, No.43, pp.37-56
- Oosterom, V.P. 1989 A reactive data structure for geographic information systems, Proceeding of Auto-Carto 9, pp. 665-674.
- Papadias, D and Theodoridis, Y., 1997, Spatial relations, minimum bounding rectangles, and spatial data structures. *International Journal of Information Science*, Vol.15, No.2 pp.111-138.
- Papadis, D. and Egenhofer, M. 1997, Algorithms for Hierarchical spatial reasoning. *GeoInformatica*, Vol.1(3), pp.251-273.
- Papadias, D., Karacapilidis, N. and Arkoumanis, D. 1999, Processing fuzzy spatial queries: a configuration similarity approach. *International Journal of Geographical Information Science*, 13 (2), 93-118
- Parent, C., 1998 Modeling spatial data in the MADS conceptual model. in proceedings of SDH'98, Vancouver, Canada.
- Peng, W., Molenaar, M. 1995 An object-oriented approach to automated generalization, Proceedings of GeoInformatics 95, International Symposium on RS, GIS & GPS in Sustainable Development and Environmental Monitoring, Hong Kong. pp. 295-304.

References

- Peng, W., Tempfli, K. 1996, An object-oriented design for automated database generalization, SDH 96, pp. 4B.15-4B.30.
- Peng, W., Tempfli, K. and Molenaar, M. 1996, Automated generalization in a GIS context, Proceedings of GeoInformatics'96, International Symposium on GIS/RS, Research, Development and Application, Florida, USA, pp.135-144.
- Peng, W. 1997, Automated Generalization in GIS, PhD Thesis, Wageningen Agricultural University (and ITC), The Netherlands.
- Peng, W. 2000, Database generalization: Concepts, problems and operation. International archive of photogrammetry and remote sensing Vol XXXXIII, part B4.
- Peter, B. and Weibel, R. 1999, Using vector and raster-based techniques in categorical map generalization. Third ICA workshop on progress in automated map generalization, Ottawa.
- Peuquet, D. 1984, A conceptual framework and comparison of spatial data models, Cartographica, Vol. 21, No. 4, pp.66-113.
- Peuquet, D.J and Zhan, C.X , 1986, an algorithm to determine the directional relationship between arbitrarily-shaped polygons in the plane. Pattern recognition Vol.20, No.1, pp.65-74
- Pilouk, M., and Tempfli, K. 1993, An integrated DTM-GIS data structure: a relational approach, Proceedings of AUTO-Carto 12, pp.59-68.
- Pilouk, M. 1996, Integrated modelling for 3D GIS, PhD Thesis, Wageningen Agricultural University, The Netherlands.
- Preparata, F., P. and Shamos, M.I. 1985, Computational geometry: an introduction, Springer, New York.
- Richardson, D. E. 1993, Automated spatial and thematic generalization using a context transformation model, PhD. Thesis, R&B Publications, Canada.
- Richardson, D.E., 1994, Generalization of spatial and thematic data using inheritance and classification and aggregation hierarchies. Spatial data handling conference 94'. Edinburgh, Scotland, sep. 5-9, Great Britain.
- Richardson, D.E., and Thomson, 1996. Integrating thematic, geographic and topological information in the generalization of road networks. In temporal, spatial, and semantic data integration for application in remote sensing and Geographic information systems. Richardson, D.E., ed. Cartographica Monograph, Fall.

References

Rigaux,P. and Scholl,M., 1995, Multi-scale partions: Application to spatial and statistical databases. In Egenhofer, M.J., and Herring,J.R. (Eds. Advances in spatial databases, Berlin:Springer-Verlag. Pp.170-183.

Robin Fuller and Nigel Brown, 1996, A CORINE map of Great Britain by automated means. Techniques for automatic generalization of the Land Cover Map of Great Britain. International Journal of Geographic Information Systems. Vol.10 (8), 937-953.

Robinson, V. and A. Frank, 1985, About different kinds of uncertainty in collections of spatial data. Proceedings, Auto Carto 7. Washington,D.C.

Rodríguez and M. Egenhofer, 1999, Putting similarity assessment into context: matching functions with the user's intended operations. Modeling and Using Context, CONTEXT'99, Trento, Italy. In: P. Bouquet, L. Serafini, Patrick Brézillon, Massimo Benerecetti, and Francesca Castellani (eds.), Lecture Notes in Computer Science, Vol. 1688, Spiringer-Verlag, pp. 310-323,.

Rodríguez, M. Egenhofer, and R. Rugg , 1999, Assessing semantic similarity among geospatial feature class definition. Interoperating Geographic Information Systems, Second International Conference, INTEROP'99, Zurich, Zwitzerland. In: . Vckovski, K. Brassel, and H.-J. Schek (eds.), Lecture Notes in Computer Science, Vol. 1580, Springer-Verlag, pp. 189-202,.

Rodriquez M.A., Max J. Egenhofer and Robert D. Rugg , 1999, Assessing semantic similarities among geospatial feature class definitions. In A. Vckovski, k. Brassel, and H.-J. Schek (eds.) Interoperating Geographic Information Systems, Interop '99, Zurich, Switzerland. Lecture Notes In Computer Science, Vol. 1580, pp189-202.

Ruas, A. and Lagrange, J.P. 1995, Data and knowledge modeling for generalization, in GIS and Generalization (eds Muller J.C., Lagrange, J.P. and Weibel, R.), pp.73-86.

Ruas, A. and Plzanet, C., 1996, Strategies For Automated Generalization, SDH 96, pp.6A1-15.
Ruas, A., OO-Constraint Modeling to Automate Urban Generalization process, SDH 98, pp.225-235.

Ruas A. and Mackaness. W.A. 1997, Strategies for automated generalization in ACIV# pp/1387-1394, Stockholm, Sweden.

Ruas, A, 1998, OO-constraint modeling to automate urban generalization process, Proceedings of the 8th International Symposium on Spatial Data Handling, Vancouver,BC 6b, pp.225-235

Samet, H. 1990, the design and analysis of spatial data structures, Reading, AddidonWesely, Massachusetts.

References

- Shea, K.S., and McMaster, R.B., 1989, Cartographic Generalization in a Digital Environment When and How to Generalize, AUTO-CARTO 9, Baltimore, Maryland, pp56-65.
- Shea, K.S. 1991, Design considerations for an artificially intelligent system, In Map generalization: Making rules for Knowledge Representation (eds Buttenfield, B.P. and
- Shreve, R.L. 1967, Infinite topologically random channel networks. *Journal of Geology* 75, 178-186.
- Sibson, R., 1977 Locally equiangular triangulations. *Comput.J.*, Vol.21, No.3, pp 243-245.
- Sloan, S.W., 1987 A fast algorithm for constructing Delaunay triangulations in the plane, *Advanced Engineering Software*, Vol.9, pp.34-55.
- Smaalen, J.W.N. 1996, A hierarchic rule model for geographic information abstraction. SDH'96, Delft, the Netherlands, pp. 4b.31.
- Smith, J.M. and Smith, D.C.P., 1977, Database abstractions: aggregation and generalization, *ACM Transactions on Database System*, 2, pp.105-133
- Smith, B and Mark, D., M. 2001, Geographic categories: an ontological investigation. *International Journal of Information Science*, Vol.15, No.7 pp.591-612.
- Stell, J.G and Worboys, M. 1999, generalization graphs using amalgamation and selection. In Guting, R.H., Papadias D., and Lochovsky, F.(eds.) *Advance in spatial databases lecture notes in computer science*. Springer.
- Strahler, A.N. 1957, Quantitative analysis of watershed geomorphology. *Transaction of the American geophysical union* 38(6). 913-920.
- Tang, A. Adams, T. and Usery, E.L. 1996, A spatial data model design for feature-based geographical information system. *International Journal of Information Science*, Vol.10, No.5 pp.643-659.
- Thompson, P.J. 1989, *Data with semantics: Data models and data management*, Van Nostrand Reinhold, New York.
- Tinghua, Ai, 2000, *Research on Data model and methods with the help of Urban Map Databases PhD Thesis*, Wuhan Technical University of Surveying and Mapping.
- Tinghua Ai, Renzhong, Liu Yaolin, 2000, A binary tree representation of curve hierarchical structure based on gestalt principles, *Proceedings of the ninth spatial handling symposium*. pp.2a 30-43.

References

Tinghua Ai, Liu Yaolin , 2001, Construction of Street Center-line Network in Urban GIS ---- A Method Based on Delaunay Triangulation, Proceedings of International conference of Urban Geo-informatics, Surveying and Mapping press house.

Tryfona,N. and Egenhofer,M.J.1996, Multi-resolution spatial databases: consistency among networks. Integrity in databases—Sixth international workshop on Foundations of models and languages for data and objects . S.Conrad,H.J Klein and K.D Schewe(eds.) pp. 119-132.

Tsai, J.D. 1993 Delaunay triangulation in TIN creation: an overview and a linear-time algorithm, I.J. Geographical information Systems, Vol.7,pp. 501-524.

Tversky,A. 1977, Features of similarity. Psychological review, , 84(40:PP.327-352).

van Smaalen, J. W. N. 1996, A hierarchic rule model for geographic information abstraction, SDH 96, pp. 4B.31-4B.41.

Ware, M.J., and Jones, Chris.B., 1996, A Spatial Model For Detecting and Resolving Conflict Caused By Scale Reduction. 7th International Symposium on Spatial Data Handling, Delft, The Netherlands, pp. 9A15-23.

Weibel, R 1992, Models and experiments for adaptive computer-assisted terrain generalization, Cartography and Geographic Information Systems, Vol.19,No.3, pp.133-152

Weibel, R. 1995, Three essential building blocks for automated generalization, in GIS and Generalization (eds Muller J.C., Lagrange, J.P. and Weibel, R.), pp. 56-69.

Weibel, R. 1995, Three essential building blocks for automated generalization. In: Muller, J-C., Lagrange,J-P. and Weibel, R. (eds.) GIS and Generalization: Methodology and Practice. London: Tayor & Francis,pp 56-69,.

Weibel. R. 1995, Special issues on automated map generalization, Cartography and Geographic Information Systems, 22(4), 1995.

Weibel, R., 1996, A Typology of Constraints to Line Simplification, SDH 96, pp.9A1-13.

Weibel, R. and Dutton, G.H. 1998, Constraints-Based Automated Map Generalization. Proceeding of the 8th International Symposium on Spatial Data Handling, Vancouver, BC,pp.214-224.

Worboys, M.F., 1995, GIS-A Computing Perspective, London: Taylor & Francis.

References

Yee Leung, Kwong S.,L and He,J.Z 1999, A generic concept-based object-oriented geographical information system. *International Journal of Geographical Information Science*, 13 (5),pp475-498.

Zeshen Wang, 2001 A Hybrid approach for automated area aggregation. CD of ICA conference, Beijing, Aug. 2001.

Zhang,C. and Murayanma,Y. 2000 testing local spatial autocorrelation using k-order neighbors. *International Journal of Information Science*, Vol.14,No.7 pp.681-692.

Publications During PhD Study

Liu Yaolin, Martien Molenaar, 1999, Multi-Scale representation for raster drainage network based on data model. Proceeding (CD_ROM) of 19th International Cartographic Conference, Ottawa, Canada

Liu Yaolin, 1999, Design and Development of Urban land evaluation information system Journal of land Science, Vol 10, No.2,pp102-106.(in Chinese)

Liu Yaolin, Liu Yanfang, 2000, Urban Environment Analysis . WuCE Publishing House.Wuhan, China, (in Chinese).

Liu Yaolin, Liu Yang, 2000, Urban land parcels based on Data structure and knowledge, Journal of Geo-science, Vol, 6, N0.3 209-215.(in Chinese),

Liu Yaolin, Fan yanping, 2000, The shortest path methods and application in urban land evaluation. Journal of Geomatics and information Science of Wuhan University, Vol.25 N0.5,pp. 475-482 (In Chinese)

Liu Yaolin, Martien Molenaar Menno-Jan Kraak, 2001, Spatial Object Aggregation Based on Data structure, Local Triangulation and Hierarchical analyzing method. Proceeding (CD-ROM) of 20th International Cartographic Conference 2001. Aug.6-10. Beijing, China.

Liu Yaolin, Martien Molenaar, Ai Tinghua, 2001 Frameworks for Generalization Constraints and Operations Based on Object-Oriented Data Structure in Database Generalization. Proceedings (CD-ROM) of 20th International Cartographic Conference 2001, Aug.6-10.2001. Beijing ,China

Liu Yaolin, Martien Molenaar, 2001, Generalization transformation based on constraints, spatial and relationships. ASPRS Annual Conference Proceedings (CD_ROM), Washington, DC. Society of America Bulletin, v.84 .ASPRS 2001 Annual Convention, St, Louis, Mo, April 23-27.

Liu Yaolin, Li Xinglin ,2001, Urban Land grading based market data approach, Journal of Geomatics and information Science of Wuhan University, Vol.26 N0.1,pp. 75-82 (In Chinese)

Li Deren, Liu Yaolin , 2001, Techniques on Land Information System, Geological Publishing House, Beijing, China.(in Chinese)

Tinghua Ai, Liu Yaolin , 2001, Construction of Street Center-line Network in Urban GIS --- A Method Based on Delaunay Triangulation, Proceedings of International conference of Urban Geo-informatics, Surveying and Mapping press house.

References

Tinghua Ai, Renzhong, Liu yaolin ,2000, A binary tree representation of curve hierarchical structure based on gestalt principles, Proceedings of the ninth spatial handling symposium.pp.2a 30-43.

Liu Yaolin, M.Molenaar, M.J. Kraak Categorical Database Generalization Aided by Data Structure, Proceedings of the GIS Research UK 10th Annual conference, GIS UK 2002, April 3-5, UK, pp.345-349.

Liu Yaolin, Martien Molenaar Menno-Jan Kraak, ,2002, Automated Organization of Hierarchical Catchments in River Network Based Constrained Delaunay Triangulation, Proceedings of XXII Conference, FIG2002, Washington,DC, April 19-26.

Appendix I

Land use classification (code)

1. Agricultural Land (100)

- 11 Cultivated land
 - 111 Irrigated paddy fields
 - 112 Rain fed paddy fields
 - 113 Irrigated land
 - 114 Dry land
 - 115 Vegetable plots
- 12 Garden Land
 - 121 Orchards
 - 122 Mulberry fields
 - 123 Tea fields
 - 124 Rubber plantation
 - 125 Other
- 13 Forest
 - 131 Wood land
 - 132 Shrubbery land
 - 133 Sparsely forest wood land
 - 134 Young forestation land
 - 135 Slashes
 - 136 Seeding nurseries
- 14 Pasture land
 - 141 Natural grass land
 - 142 Improved grassland
 - 143 Man-made grass land
- 15 Water area
 - 151 Rivers
 - 152 Lakes
 - 153 Reservoir
 - 154 Pond
 - 155 Reed land

Appendix I: Land use Classification

156	Beaches and flats
157	Irrigation canals and ditches
158	Hydraulic building
159	Glaciers and firus

2. Construction Land (200)

26	Residential quarters and industrial and mining land
261	Area of cities and towns
262	Residential quarters in rural areas
263	Isolated industrial and mining land
264	Salt pans
265	Special-used land
27	Land use for transportation
271	Railways
272	Roads
273	Rural roads
274	Civil airports
275	Harbors and wharfs

3 Unused lands (300)

381	Waste lands
382	Saline alkali land
383	Wetland
384	Sandy land
385	Bare land
386	Rock and shingle
387	Ridges
388	Others

Appendix II: Similarity Table

Appendix III Class Definition

```
typedef struct tagEDGETYPE
{ long   fromP;
  long   toP;
  long   fromObj;
  long   toObj;
  long   leftTri;
  long   rightTri;
} EDGETYPE;
```

```
typedef CArray<EDGETYPE,EDGETYPE> EDGETYPEARRAY;
```

```
typedef struct tagOBJECTTYPE
{ POINT *pt;
  long   ptNum;
  POINT *pb;
  long   pbNum;
  long   *Skeleton;
  long   SkeletonNum;
  POINT *Pd;
  POINT PO;
  double deltX,deltY;
  short Degree;
} OBJECTTYPE;
```

```
typedef CArray<OBJECTTYPE,OBJECTTYPE> OBJECTTYPEARRAY;
```

```
typedef struct tagNODETYPE
{ POINT P;
  long   SkeletonNum;
  long   Skeleton[3];
} NODETYPE;
```

```
typedef CArray<NODETYPE,NODETYPE> NODETYPEARRAY;
```

```
//===== Next definition is for landuse Generalizstion
```

```
typedef struct tag_LANDUSE_ARCTYPE
{ POINT *pt;
  int ptNum;
  int leftpoly;
  int rightpoly;
```

```
    int leftloop;
    int rightloop;
    int leftcategory;
    int rightcategory;
    int code;
    short valid;
} _LANDUSE_ARCTYPE;
typedef CArray<_LANDUSE_ARCTYPE,_LANDUSE_ARCTYPE>
_LANDUSE_ARCTYPEARRAY;

typedef struct tag _LANDUSE_POLYTYPE
{ int *stringNum;
  int loops;
  int **string;
  double area;
  int category;
  int toparcel;
} _LANDUSE_POLYTYPE;
typedef CArray<_LANDUSE_POLYTYPE,_LANDUSE_POLYTYPE>
_LANDUSE_POLYTYPEARRAY;

typedef struct tag _LANDUSE_PARCELTYPE
{ int category;
  int arcstringNum;
  int *arcstring;
  int polystringNum;
  int *polystring;
  double area;
} _LANDUSE_PARCELTYPE;
typedef CArray<_LANDUSE_PARCELTYPE,_LANDUSE_PARCELTYPE>
_LANDUSE_PARCELTYPEARRAY;

typedef struct tag _LANDUSE_PATCHTYPE
{ int category;
  int ptNum;
  POINT *pt;
  int linkpNum;
  POINT *linkp0;
  POINT *linkp1;
  int *parcelstring;
  int parcelstringNum;
} _LANDUSE_PATCHTYPE;
typedef CArray<_LANDUSE_PATCHTYPE,_LANDUSE_PATCHTYPE>
_LANDUSE_PATCHTYPEARRAY;
```

```
//=====END
```

```
class CTest_Liu : public CGeoGeneralize  
{
```

```
// Construction
```

```
public:
```

```
    CTest_Liu(CGeoEdit* pEdit);
```

```
// Attributes
```

```
public:
```

```
    CGeoMap*   m_pGeoMap;  
    CGeoEdit*  m_pEdit;  
    CAutoMapView* m_pView;  
    double m_arrowlength;  
    int m_SuperClass[100];  
    COLORREF m_LanduseColor[100];  
    LONGARRAY m_Group;  
    LONGARRAY *m_AllSupers;  
    _LANDUSE_ARCTYPEARRAY m_AllArcs;  
    _LANDUSE_POLYTYPEARRAY m_AllPolys;  
    _LANDUSE_PARCELTYPEARRAY m_AllParcels;  
    _LANDUSE_PATCHTYPEARRAY m_AllPatches;
```

```
// Operations
```

```
public:
```

```
    void DetectNeighborObjects(); // Grouping neighbor polygons, for Liu  
    void LandUseAnalysis(); // for Liu, Landuse Parcel neighbor analysis  
    bool _Landuse_InitData();  
    void _Landuse_DrawParcelClass();  
    void _Landuse_DefineColor();  
    void _Landuse_LoadSupewClass();  
    void _Landuse_CombineToSuperclass();  
    void _Landuse_ChangeSmallParcelCategory(LONGARRAY &smallparcel);  
    void _Landuse_MakeParcel();  
    void _Landuse_MakeGroupParcel();  
    void _Landuse_CreateTIN(int n, LONGARRAY &smallparcel);  
    void _Landuse_FreeVariables();  
    void _Landuse_GetLoopCoord(int PolyNo, int LoopNo, int &ptNum, POINT **pt);  
    void _Landuse_GetPolyCoord(int PolyNo, int &Num, int **ptNum, POINT **pt);
```

Appendix III: Class Definition

```
void _Landuse_DrawPolys();
void _Landuse_DrawOnePoly( int PolyNo,int width, COLORREF pencolor,COLORREF
brcolor);
void _Landuse_DrawArcs(int valid);
void _Landuse_DrawInvalidArcs();
void _Landuse_DrawOneArc(int ArcNo,int width, COLORREF pencolor);
void _Landuse_DrawParcels();
void _Landuse_DrawOneParcel( int ParcelNo,int width, COLORREF pencolor,COLORREF
brcolor,int aec,int poly);
void _Landuse_DrawOneGroup(int Nogroup,int arc,int group);
void _Landuse_DrawOnePatch(int npatch);

void BuildingClusterAnalysis(); // VORONOI diagram construction and Generalization
void DistanceRelationAnalysis(); // Iso-Distance_relation Demo.
void ContourStructure(); // Perform Tang's algorithm

bool ConstructTIN(int objnum, int *ptNum, POINT **pt, int Interval);
int FindEntryTri(int *Ntri);

void ReleaseVariables(NODETYPEARRAY &AllNode,SKELETONTYPEARRAY
&AllSkeleton, OBJECTTYPEARRAY &AllObjects, INTARRAY &ConflictSkeleton,
INTARRAY &ConflictObject, int Nogroup, LONGARRAY *group);

bool GenerateGrowthPolygon(SKELETONTYPEARRAY &AllSkeleton, int objnum, int
*ptNum, POINT** pt, OBJECTTYPEARRAY &AllObjects);

void SortSkeletonOnWidth(SKELETONTYPEARRAY &AllSkeleton);

void GetNodeRelate(SKELETONTYPEARRAY &AllSkeleton, NODETYPEARRAY
&AllNode);

void GetSkeletonDirection(SKELETONTYPEARRAY &AllSkeleton, NODETYPEARRAY
&AllNode);

void GetNeighborObjects(SKELETONTYPEARRAY &AllSkeleton,
OBJECTTYPEARRAY &AllObjects, int no,INTARRAY &NeighborObjects);

void GetNeighborDegree( SKELETONTYPEARRAY &AllSkeleton, OBJECTTYPEARRAY
&AllObjects, int centerno,int h);

void GenerateObjectMoveDirection(SKELETONTYPEARRAY
&AllSkeleton,OBJECTTYPEARRAY &AllObjects); POINT
GetIntegrateMovementDirection(OBJECTTYPE o0, short *IsConflict);
```



```
int DrawSkeletonGrowthPolygon(SKELETONTYPEARRAY
&AllSkeleton,OBJECTTYPEARRAY &AllObjects);

void GetConflict(double distance,SKELETONTYPEARRAY &AllSkeleton, INTARRAY
&ConflictSkeleton, INTARRAY &ConflictObject);

void SelectConflictSkeleton(SKELETONTYPEARRAY
&AllSkeleton,OBJECTTYPEARRAY &AllObjects,
INTARRAY &ConflictSkeleton, INTARRAY &ConflictObject);

void ClassifyConflictObject(SKELETONTYPEARRAY
&AllSkeleton,OBJECTTYPEARRAY &AllObjects, INTARRAY &ConflictSkeleton,
INTARRAY &ConflictObject, int &uj, LONGARRAY **group);

void DrawConflictSkeletonObject(SKELETONTYPEARRAY
&AllSkeleton,OBJECTTYPEARRAY &AllObjects, INTARRAY &ConflictSkeleton,
INTARRAY &ConflictObject);

void DrawObjectMoveDirection(SKELETONTYPEARRAY
&AllSkeleton,OBJECTTYPEARRAY &AllObjects, INTARRAY &ConflictSkeleton,
INTARRAY &ConflictObject,int eachdirection,int wholedirection);
void DisplaceConflictObjects(SKELETONTYPEARRAY
&AllSkeleton,OBJECTTYPEARRAY &AllObjects, INTARRAY &ConflictSkeleton,
INTARRAY &ConflictObject);

void DrawDistributionDensity(OBJECTTYPEARRAY &AllObjects);

void DrawDistanceRelation(SKELETONTYPEARRAY &AllSkeleton,
OBJECTTYPEARRAY &AllObjects, int BackObject, int centerno);

void DrawPolygonVoronoi(SKELETONTYPEARRAY &AllSkeleton,
OBJECTTYPEARRAY &AllObjects, int BackObject, int centerno);
bool CombineObjects( int Nogroup, LONGARRAY *group, int dJoinDist, int dRasterWidth,
OBJECTTYPEARRAY &AllObjects, int &newobjnum, int **ppNum, POINT ***pp);

void ConnectGroupObjects(SKELETONTYPEARRAY &AllSkeleton, OBJECTTYPEARRAY
&AllObjects,INTARRAY &ConflictSkeleton);

void DrawPoint(POINT retpt,COLORREF pencolor, int width);

void DrawPolyline(POINT* retpt, int retptNum, COLORREF pencolor, int width);

void DrawPolygon(POINT* retpt, int retptNum,COLORREF pencolor,COLORREF brcolor, int
width);
```

Appendix III: Class Definition

```
void DrawPolygons(POINT **pi,int *piNum,int Nums,int *which, COLORREF
pencolor,COLORREF brcolor, int width);

void DrawPolyPolygon(POINT* retpt, int *retptNum, int count,
COLORREF pencolor,COLORREF brcolor, int width);

void DrawRefreshStudyArea();

int TwoLineIntersect(POINT *pt,int ptNum,POINT *pi,int piNum, POINT &IntersectP);

void ConnectLines(double MatchDis,POINT** pp, int* ppNum, int objnum,
POINT** retpt,int** retptNum,int& retNum );

double AngleBetweenP01_P02(POINT P1,POINT P0,POINT P2);

public:
    virtual ~CTest_Liu();

protected:

};
```

Completed PhD Studies at ITC

1. **Akinyede**, 1990, Highway cost modelling and route selection using a geotechnical information system
2. **Pan He Ping**, 1990, 90-9003757-8, Spatial structure theory in machine vision and applications to structural and textural analysis of remotely sensed images
3. **Bocco Verdinelli, G.**, 1990, Gully erosion analysis using remote sensing and geographic information systems: a case study in Central Mexico
4. **Sharif, M**, 1991, Composite sampling optimization for DTM in the context of GIS
5. **Drummond, J.**, 1991, Determining and processing quality parameters in geographic information systems
6. **Groten, S.**, 1991, Satellite monitoring of agro-ecosystems in the Sahel
7. **Sharifi, A.**, 1991, 90-6164-074-1, Development of an appropriate resource information system to support agricultural management at farm enterprise level
8. **Zee, D. van der**, 1991, 90-6164-075-X, Recreation studied from above: Air photo interpretation as input into land evaluation for recreation
9. **Mannaerts, C.**, 1991, 90-6164-085-7, Assessment of the transferability of laboratory rainfall-runoff and rainfall - soil loss relationships to field and catchment scales: a study in the Cape Verde Islands
10. **Ze Shen Wang**, 1991: 90-393-0333-9, An expert system for cartographic symbol design
11. **Zhou Yunxian**, 1991, 90-6164-081-4, Application of Radon transforms to the processing of airborne geophysical data
12. **Zuviria, M. de**, 1992, 90-6164-077-6, Mapping agro-topoclimates by integrating topographic, meteorological and land ecological data in a geographic information system: a case study of the Lom Sak area, North Central Thailand
13. **Westen, C. van**, 1993, 90-6164-078-4, Application of Geographic Information Systems to landslide hazard zonation
14. **Shi Wenzhong**, 1994, 90-6164-099-7, Modelling positional and thematic uncertainties in integration of remote sensing and geographic information systems
15. **Javelosa, R.**, 1994, 90-6164-086-5, Active Quaternary environments in the Philippine mobile belt
16. **Lo King-Chang**, 1994, 90-9006526-1, High Quality Automatic DEM, Digital Elevation Model Generation from Multiple Imagery
17. **Wokabi, S.**, 1994, 90-6164-102-0, Quantified land evaluation for maize yield gap analysis at three sites on the eastern slope of Mt. Kenya
18. **Rodriguez, O.**, 1995, Land Use conflicts and planning strategies in urban fringes: a case study of Western Caracas, Venezuela
19. **Meer, F. van der**, 1995, 90-5485-385-9, Imaging spectrometry & the Ronda peridotites
20. **Kufoniyi, O.**, 1995, 90-6164-105-5, Spatial coincidence: automated database updating and data consistency in vector GIS
21. **Zambezi, P.**, 1995, Geochemistry of the Nkombwa Hill carbonatite complex of Isoka District, north-east Zambia, with special emphasis on economic minerals

22. **Woldai, T.**, 1995, The application of remote sensing to the study of the geology and structure of the Carboniferous in the Calañas area, pyrite belt, SW Spain
23. **Verweij, P.**, 1995, 90-6164-109-8, Spatial and temporal modelling of vegetation patterns: burning and grazing in the Paramo of Los Nevados National Park, Colombia
24. **Pohl, C.**, 1996, 90-6164-121-7, Geometric Aspects of Multisensor Image Fusion for Topographic Map Updating in the Humid Tropics
25. **Jiang Bin**, 1996, 90-6266-128-9, Fuzzy overlay analysis and visualization in GIS
26. **Metternicht, G.**, 1996, 90-6164-118-7, Detecting and monitoring land degradation features and processes in the Cochabamba Valleys, Bolivia. A synergistic approach
27. **Hoanh Chu Thai**, 1996, 90-6164-120-9, Development of a Computerized Aid to Integrated Land Use Planning (CAILUP) at regional level in irrigated areas: a case study for the Quan Lo Phung Hiep region in the Mekong Delta, Vietnam
28. **Roshannejad, A.**, 1996, 90-9009284-6, The management of spatio-temporal data in a national geographic information system
29. **Terlien, M.**, 1996, 90-6164-115-2, Modelling Spatial and Temporal Variations in Rainfall-Triggered Landslides: the integration of hydrologic models, slope stability models and GIS for the hazard zonation of rainfall-triggered landslides with examples from Manizales, Colombia
30. **Mahavir, J.**, 1996, 90-6164-117-9, Modelling settlement patterns for metropolitan regions: inputs from remote sensing
31. **Al-Amir, S.**, 1996, 90-6164-116-0, Modern spatial planning practice as supported by the multi-applicable tools of remote sensing and GIS: the Syrian case
32. **Pilouk, M.**, 1996, 90-6164-122-5, Integrated modelling for 3D GIS
33. **Duan Zengshan**, 1996, 90-6164-123-3, Optimization modelling of a river-aquifer system with technical interventions: a case study for the Huangshui river and the coastal aquifer, Shandong, China
34. **Man, W.H. de**, 1996, 90-9009-775-9, Surveys: informatie als norm: een verkenning van de institutionalisering van dorp - surveys in Thailand en op de Filipijnen
35. **Vekerdy, Z.**, 1996, 90-6164-119-5, GIS-based hydrological modelling of alluvial regions: using the example of the Kisaföld, Hungary
36. **Pereira, Luisa**, 1996, 90-407-1385-5, A Robust and Adaptive Matching Procedure for Automatic Modelling of Terrain Relief
37. **Fandino Lozano, M.**, 1996, 90-6164-129-2, A Framework of Ecological Evaluation oriented at the Establishment and Management of Protected Areas: a case study of the Santuario de Iguaque, Colombia
38. **Toxopeus, B.**, 1996, 90-6164-126-8, ISM: an Interactive Spatial and temporal Modelling system as a tool in ecosystem management: with two case studies: Cibodas biosphere reserve, West Java Indonesia: Amboseli biosphere reserve, Kajiado district, Central Southern Kenya
39. **Wang Yiman**, 1997, 90-6164-131-4, Satellite SAR imagery for topographic mapping of tidal flat areas in the Dutch Wadden Sea
40. **Asun Saldana-Lopez**, 1997, 90-6164-133-0, Complexity of soils and Soilscape patterns on the southern slopes of the Ayllon Range, central Spain: a GIS assisted modelling approach

41. **Ceccarelli, T.**, 1997, 90-6164-135-7, Towards a planning support system for communal areas in the Zambezi valley, Zimbabwe; a multi-criteria evaluation linking farm household analysis, land evaluation and geographic information systems
42. **Peng Wanning**, 1997, 90-6164-134-9, Automated generalization in GIS
43. **Lawas, C.**, 1997, 90-6164-137-3, The Resource Users' Knowledge, the neglected input in Land resource management: the case of the Kankanaey farmers in Benguet, Philippines
44. **Bijker, W.**, 1997, 90-6164-139-X, Radar for rain forest: A monitoring system for land cover Change in the Colombian Amazon
45. **Farshad, A.**, 1997, 90-6164-142-X, Analysis of integrated land and water management practices within different agricultural systems under semi-arid conditions of Iran and evaluation of their sustainability
46. **Orlic, B.**, 1997, 90-6164-140-3, Predicting subsurface conditions for geotechnical modelling
47. **Bishr, Y.**, 1997, 90-6164-141-1, Semantic Aspects of Interoperable GIS
48. **Zhang Xiangmin**, 1998, 90-6164-144-6, Coal fires in Northwest China: detection, monitoring and prediction using remote sensing data
49. **Gens, R.**, 1998, 90-6164-155-1, Quality assessment of SAR interferometric data
50. **Turkstra, J.**, 1998, 90-6164-147-0, Urban development and geographical information: spatial and temporal patterns of urban development and land values using integrated geo-data, Villaviciencia, Colombia
51. **Cassells, C.**, 1998, Thermal modelling of underground coal fires in northern China
52. **Naseri, M.**, 1998, 90-6164-195-0, Characterization of Salt-affected Soils for Modelling Sustainable Land Management in Semi-arid Environment: a case study in the Gorgan Region, Northeast, Iran
53. **Gorte B.G.H.**, 1998, 90-6164-157-8, Probabilistic Segmentation of Remotely Sensed Images
54. **Tenalem Ayenew**, 1998, 90-6164-158-6, The hydrological system of the lake district basin, central main Ethiopian rift
55. **Wang Donggen**, 1998, 90-6864-551-7, Conjoint approaches to developing activity-based models
56. **Bastidas de Calderon, M.**, 1998, 90-6164-193-4, Environmental fragility and vulnerability of Amazonian landscapes and ecosystems in the middle Orinoco river basin, Venezuela
57. **Moameni, A.**, 1999, Soil quality changes under long-term wheat cultivation in the Marvdasht plain, South-Central Iran
58. **Groenigen, J.W. van**, 1999, 90-6164-156-X, Constrained optimisation of spatial sampling: a geostatistical approach
59. **Cheng Tao**, 1999, 90-6164-164-0, A process-oriented data model for fuzzy spatial objects
60. **Wolski, Piotr**, 1999, 90-6164-165-9, Application of reservoir modelling to hydrotopes identified by remote sensing
61. **Acharya, B.**, 1999, 90-6164-168-3, Forest biodiversity assessment: A spatial analysis of tree species diversity in Nepal
62. **Akbar Abkar, Ali**, 1999, 90-6164-169-1, Likelihood-based segmentation and classification of remotely sensed images

63. **Yanuariadi, T.**, 1999, 90-5808-082-X, Sustainable Land Allocation: GIS-based decision support for industrial forest plantation development in Indonesia
64. **Abu Bakr, Mohamed**, 1999, 90-6164-170-5, An Integrated Agro-Economic and Agro-Ecological Framework for Land Use Planning and Policy Analysis
65. **Eleveld, M.**, 1999, 90-6461-166-7, Exploring coastal morphodynamics of Ameland (The Netherlands) with remote sensing monitoring techniques and dynamic modelling in GIS
66. **Yang Hong**, 1999, 90-6164-172-1, Imaging Spectrometry for Hydrocarbon Microseepage
67. **Mainam, Félix**, 1999, 90-6164-179-9, Modelling soil erodibility in the semiarid zone of Cameroon
68. **Bakr, Mahmoud**, 2000, 90-6164-176-4, A Stochastic Inverse-Management Approach to Groundwater Quality
69. **Zlatanova, Z.**, 2000, 90-6164-178-0, 3D GIS for Urban Development
70. **Ottichilo, Wilber K.**, 2000, 90-5808-197-4, Wildlife Dynamics: An Analysis of Change in the Masai Mara Ecosystem
71. **Kaymakci, Nuri**, 2000, 90-6164-181-0, Tectono-stratigraphical Evolution of the Cankori Basin (Central Anatolia, Turkey)
72. **Gonzalez, Rhodora**, 2000, 90-5808-246-6, Platforms and Terraces: Bridging participation and GIS in joint-learning for watershed management with the Ifugaos of the Philippines
73. **Schetselaar, Ernst**, 2000, 90-6164-180-2, Integrated analyses of granite-gneiss terrain from field and multisource remotely sensed data. A case study from the Canadian Shield
74. **Mesgari, Saadi**, 2000, 90-3651-511-4, Topological Cell-Tuple Structure for Three-Dimensional Spatial Data
75. **Bie, Cees A.J.M. de**, 2000, 90-5808-253-9, Comparative Performance Analysis of Agro-Ecosystems
76. **Khaemba, Wilson M.**, 2000, 90-5808-280-6, Spatial Statistics for Natural Resource Management
77. **Shrestha, Dhruba**, 2000, 90-6164-189-6, Aspects of erosion and sedimentation in the Nepalese Himalaya: highland-lowland relations
78. **Asadi Haroni, Hooshang**, 2000, 90-6164-185-3, The Zarshuran Gold Deposit Model Applied in a Mineral Exploration GIS in Iran
79. **Raza, Ale**, 2001, 90-3651-540-8, Object-oriented Temporal GIS for Urban Applications
80. **Farah, Hussein**, 2001, 90-5808-331-4, Estimation of regional evaporation under different weather conditions from satellite and meteorological data. A case study in the Naivasha Basin, Kenya
81. **Zheng, Ding**, 2001, 90-6164-190-X, A Neural - Fuzzy Approach to Linguistic Knowledge Acquisition and Assessment in Spatial Decision Making
82. **Sahu, B.K.**, 2001, Aeromagnetics of continental areas flanking the Indian Ocean; with implications for geological correlation and Gondwana reassembly
83. **Alfestawi, Y.**, 2001, 90-6164-198-5, The structural, paleogeographical and hydrocarbon systems analysis of the Ghadamis and Murzuq Basins, West Libya, with emphasis on their relation to the intervening Al Qarqaf Arch
84. **Liu, Xuehua**, 2001, 90-5808-496-5, Mapping and Modelling the Habitat of Giant Pandas in Foping Nature Reserve, China

85. **Oindo, Boniface Oluoch**, 2001, 90-5808-495-7, Spatial Patterns of Species Diversity in Kenya
86. **Carranza, Emmanuel John**, 2002, 90-6164-203-5, Geologically-constrained Mineral Potential Mapping
87. **Rugege, Denis**, 2002, 90-5808-584-8, Regional Analysis of Maize-Based Land Use Systems for Early Warning Applications

Curriculum Vitae

Liu Yaolin was born in 1960 in Huanggang city, Hubei Province, China. He received his Bachelor of Science in Geo-mechanics from China University of Geo-science in 1982. From 1982 to now, he has been working in the Department of Cartography of the former Wuhan Technical University of Surveying and Mapping which was merged into Wuhan University in Aug., 2000. In the meantime, he received a Master of Science in Remote Sensing Application in automated classification methods from China University of Geo-science in 1988 and a Post Graduate Diploma in Soil Surveying with specialization in remote sensing from International Institute for Aerospace Survey and Earth Science in 1989.

He became Vice Dean of School of Cartography and Land Information at former Wuhan Technical University of Surveying and Mapping in 1994 and Dean in 1997. Since 2000, he has been Dean of School of Resource and Environment at Wuhan University.

Liu Yaolin has been involved in various scientific research projects covering thematic mapping, database generalization, land evaluation, planning, land information system, remote sensing applications and urban environment analysis.

He started his PhD research in March of 1998 and will complete the research in April of 2002. He is the author and co-author of a number of scientific articles and books.