UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

VANIA BOGORNY

Enhancing Spatial Association Rule Mining in Geographic Databases

Thesis presented in partial fulfillment of the requirements for the degree of Doctor of Computer Science

Prof. Dr. Luis Otavio Alvares
Advisor

Prof. Dr. Paulo Martins Engel
Co-advisor

Porto Alegre, October 2006

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Bogorny, Vania

Enhancing Spatial Association Rule Mining in Geographic Databases / Vania Bogorny – Porto Alegre: Programa de Pós-Graduação em Computação, 2006.

120 f.:il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2006. Orientador: Luis Otavio Alvares; Co-orientador: Paulo Martins Engel.

1.Spatial Association Rules. 2.Geographic Databases 3.Frequent Geographic Pattern Mining. I.Luis Otavio Alvares. II. Paulo Martins Engel. III. Título.

ACKNOWLEDGMENTS

Agradeço a Deus por te me concedido a oportunidade de estar nesta vida, por ter me dado este corpo perfeito e saudável para que pudesse desenvolver-me e de alguma forma contribuir para a humanidade.

Aos meus orientadores Luis Otavio Álvares e Paulo Martins Engel, pela confiança, pelo incentivo, paciência e principalmente por acreditar neste trabalho que rendeu tantos frutos e ainda produzirá muitos resultados num futuro próximo.

Em especial ao CNPq pela bolsa de doutorado no país, que com o auxílio da taxa de bancada permitiu-me participar de duas conferências internacionais que abriram as portas para o pós-doutorado na Universidade de Hasselt. Também gostaria de agradecer à CAPES pela bolsa de doutorado sanduíche, realizado na Universidade de Minnesota.

A todos os revisores anônimos dos diversos artigos submetidos que com as suas críticas e sugestões contribuíram para o aperfeiçoamento deste trabalho.

Agradeço à Universidade Federal do Rio Grande do Sul pela oportunidade concedida para a realização deste trabalho. Agradeço também ao governo brasileiro, pelas universidades federais que permitem a qualificação de pessoal sem nenhum custo ao estudante.

A todos os funcionários do Instituto de Informática, pelo bom atendimento e pela colaboração com informações e serviços prestados.

Agradeço em especial à minha família por todo o apoio concedido.

Agradeço ao amigo Formiga, que embora o nome lembre um ser pequeno, foi um grande incentivo para que eu iniciasse o doutorado.

Aos amigos e colegas que contribuíram de forma direta para a validação desta tese, em especial ao Sandro Camargo, Andrey Tietbohl, e João Valiati. Ao João, meu muito obrigada porque mesmo finalizando sua tese, dedicou-se para concluir a implementação dos algoritmos que permitiram a realização dos experimentos finais.

A todos os demais colegas e amigos com os quais convivi no Instituto de Informática durante estes quatro anos. Em especial à Daniela Leal Musa pela grande amizade, sugestões e conselhos ao longo destes anos.

TABLE OF CONTENTS

LIST OF ABREVIATIONS

KDD	Knowledge Discovery in Databases
KDGD	Knowledge Discovery in Geographic Databases
SDBMS	Spatial Database Management System
DM	Data Mining
SQL	Structure Query Language
CBM	Calculus Based Method
DEM	Dimension Extended Method
GIS	Geographic Information Systems
GDB	Geographic Databases
OGC	Open GIS Consortium
GKB	Geographic Knowledge Base
MBR	Minimum Boundary Rectangle
SAR	Spatial Association Rule Mining
FGP	Frequent Geographic Patterns
MFGP	Maximal Frequent Geographic Patterns

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The association rule mining technique emerged with the objective to find novel, useful, and previously unknown associations from transactional databases, and a large amount of association rule mining algorithms have been proposed in the last decade. Their main drawback, which is a well known problem, is the generation of large amounts of frequent patterns and association rules. In geographic databases the problem of mining spatial association rules increases significantly. Besides the large amount of generated patterns and rules, many patterns are *well known* geographic domain associations, normally explicitly represented in geographic database schemas. The majority of existing algorithms do not warrant the elimination of all well known geographic dependences. The result is that the same associations represented in geographic database schemas are extracted by spatial association rule mining algorithms and presented to the user. The problem of mining spatial association rules from geographic databases requires at least three main steps: compute spatial relationships, generate frequent patterns, and extract association rules. The first step is the most effort demanding and time consuming task in the rule mining process, but has received little attention in the literature. The second and third steps have been considered the main problem in transactional association rule mining and have been addressed as two different problems: *frequent pattern mining* and *association rule mining*. Well known geographic dependences which generate well known patterns may appear in the three main steps of the spatial association rule mining process. Aiming to eliminate well known dependences and generate more interesting patterns, this thesis presents a framework with three main methods for mining frequent geographic patterns using *knowledge constraints*. Semantic knowledge is used to avoid the generation of patterns that are previously known as non-interesting. The first method reduces the input problem, and all well known dependences that can be eliminated without loosing information are removed in data preprocessing. The second method eliminates combinations of pairs of geographic objects with dependences, during the frequent set generation. A third method presents a new approach to generate non-redundant frequent sets, the maximal generalized frequent sets without dependences. This method reduces the number of frequent patterns very significantly, and by consequence, the number of association rules.

Keywords: spatial data mining, knowledge discovery in geographic databases, geo-ontologies, geographic database schemas, geographic domain knowledge, spatial association rules, geographic data preprocessing, frequent geographic pattern mining

Melhorando a Mineração de Regras de Associação Espacial em Bancos de Dados Geográficos

RESUMO

A técnica de mineração de regras de associação surgiu com o objetivo de encontrar conhecimento novo, útil e previamente desconhecido em bancos de dados transacionais, e uma grande quantidade de algoritmos de mineração de regras de associação tem sido proposta na última década. O maior e mais bem conhecido problema destes algoritmos é a geração de grandes quantidades de conjuntos freqüentes e regras de associação. Em bancos de dados geográficos o problema de mineração de regras de associação espacial aumenta significativamente. Além da grande quantidade de regras e padrões gerados a maioria são associações do domínio geográfico, e são bem conhecidas, normalmente explicitamente representadas no esquema do banco de dados. A maioria dos algoritmos de mineração de regras de associação não garantem a eliminação de dependências geográficas conhecidas a priori. O resultado é que as mesmas associações representadas nos esquemas do banco de dados são extraídas pelos algoritmos de mineração de regras de associação e apresentadas ao usuário. O problema de mineração de regras de associação espacial pode ser dividido em três etapas principais: extração dos relacionamentos espaciais, geração dos conjuntos freqüentes e geração das regras de associação. A primeira etapa é a mais custosa tanto em tempo de processamento quanto pelo esforço requerido do usuário. A segunda e terceira etapas têm sido consideradas o maior problema na mineração de regras de associação em bancos de dados transacionais e tem sido abordadas como dois problemas diferentes: “frequent pattern mining” e “association rule mining”. Dependências geográficas bem conhecidas aparecem nas três etapas do processo. Tendo como objetivo a eliminação dessas dependências na mineração de regras de associação espacial essa tese apresenta um *framework* com três novos métodos para mineração de regras de associação utilizando restrições semânticas como conhecimento a priori. O primeiro método reduz os dados de entrada do algoritmo, e dependências geográficas são eliminadas parcialmente sem que haja perda de informação. O segundo método elimina combinações de pares de objetos geográficos com dependências durante a geração dos conjuntos freqüentes. O terceiro método é uma nova abordagem para gerar conjuntos freqüentes não redundantes e sem dependências, gerando conjuntos freqüentes máximos. Esse método reduz consideravelmente o número final de conjuntos freqüentes, e como conseqüência, reduz o número de regras de associação espacial.

Palavras-chave: mineração de dados espaciais, descoberta de conhecimento em bancos de dados geográficos, ontologias geográficas, esquemas de bancos de dados geográficos, conhecimento do domínio geográfico, regras de associação espacial, pré-processamento de dados geográficos, mineração de conjuntos freqüentes.

1 INTRODUCTION AND MOTIVATION

Large amounts of geographic data have been used more and more in many areas in different application domains such as urban planning, transportation, telecommunication, marketing, etc. These data are stored under Geographic Database Management Systems (GDBMS), and manipulated by Geographic Information Systems (GIS). The latter is the technology which provides a set of operations and functions for geographic data analysis. However, within the large amount of data stored in geographic databases there is implicit, non-trivial, and previously unknown knowledge that cannot be discovered by GIS. Specific techniques are necessary to find this kind of knowledge, which is the objective of Knowledge Discovery in Databases (KDD).

KDD is an interactive process which according to (FAYYAD, 1996) consists of five main steps: *selection, preprocessing, transformation, data mining* and *evaluation/interpretation. Selection, preprocessing* and *transformation* are data preparation steps in which data are rearranged to the format required by data mining algorithms. It is stated that between 60 and 80 percent of the time and effort in the whole KDD process are required for data preparation in non-spatial databases (ADRIANNS, 1996). For knowledge discovery in geographic databases the problem of data preprocessing increases significantly because of the complexity of geographic data that must be considered.

Data Mining (DM) is the step of applying discovery algorithms that produce an enumeration of patterns over the data. Most of these algorithms operate with a restrictive *single table* input format. This limitation causes a *gap* between geographic databases and data mining algorithms. Some of the existing data mining techniques are association rules, clustering, classification, trend detection, etc.

Interpretation is the step where the patterns discovered by data mining algorithms are visualized and analyzed. The discovered patterns are directly related to the type and the quality of the dataset submitted to DM algorithms as well as the characteristics of the data considered for data mining. This step may revert to *selection, preprocessing, transformation,* or *data mining* to repeat the process when no interesting patterns were discovered or when other aspects should be considered.

Different methods for general knowledge discovery have been proposed in the literature for mining spatial/geographic databases. Some specify data mining query languages such as GMQL (Geo-Miner Query Language) (HAN, 1997), LARECOS (BIGOLIN, 1998), and SDMOQL (Spatial Data mining Object Query Language) (MALERBA, 2002). Most of these languages are neither implemented in available data

mining systems nor in database management systems. Most GDBMS follow the Structured Query Language (SQL), which became the standard language to manipulate databases, and do not implement data mining languages or operations to automate or semi-automate spatial data mining.

Other approaches define new operations such as *get_nGraph*, *get_neighborhood* and *create_nPaths* to compute geographic neighbors (ESTER, 2000). A few approaches create software prototypes, such as GeoMiner (HAN, 1997), ARES (APPICE, 2005), and INGENS (MALERBA, 2003).

In this thesis we address the spatial association rule mining technique, which has been largely used for knowledge discovery in both transactional databases and geographic databases. An association rule is a general form of dependence where an element has some dependence with another element. More specifically, an association rule consists of an implication of the form $X \rightarrow Y$, where X and Y are sets of elements co-occurring in a given tuple in a dataset (AGRAWAL, 1994).

In spatial association rules (SAR) at least one element in X or Y is a spatial predicate (KOPERSKI, 1995). Spatial predicates represent materialized spatial relationships between geographic entities, such as *close*, *far*, *contains*, *within*, *touches*, etc. For example, $is\_a(slum) \wedge contains(water\_network) = NO \rightarrow hepatitisIncidence = high$. Because of spatial relationships real world entities can affect the behavior of other features in the neighborhood. This makes spatial relationships be the main characteristic of geographic data to be considered in spatial data mining (LU, 1993)(ESTER, 2000) (KOPERSKI, 1995) and the main characteristic which differs geographic/spatial data mining from transactional data mining.

The problem of mining SAR from geographic databases requires at least three main steps: compute spatial relationships, generate frequent sets (also called frequent patterns), and extract association rules.

Although the first step is the most effort and time consuming task in the process of mining SAR (SHEKHAR, 2003), it has received little attention in the literature. Not only the most effort is required from the data mining user, but the computational cost for mining SAR also relies on this step. The second and third steps have been considered the main problems in transactional association rule mining (HAN, 2000), and have been addressed in the literature as two different problems: *frequent pattern mining* and *association rule mining*.

Association rule mining algorithms in general generate a large amount of patterns and rules, which the user has to analyze in order to find novel and useful knowledge. In SAR mining this problem increases significantly. Besides the large amount of generated patterns and rules, most are well known geographic domain associations (MALERBA, 2001)(MENNIS, 2005). For example, $contains(island) \rightarrow contains(water)$ or $intersects(gasStation) \rightarrow intersects(street)$ are examples of spatial associations that represent well known dependences, since islands must be related to at least *one* water body and gas stations must intersect at least *one* street.

Users of some domains may not be interested in geographic domain associations. A doctor, for example, may not be interested in rules such as $is\_a(island) \rightarrow within(water)$, but in rules such as $is\_a(island) \wedge humidity=high \rightarrow disease=malaria$. In spatial association rule mining a large amount of rules are well known geographic domain associations which do not hold considerable information. The mixed

presentation of thousands of interesting and uninteresting rules can discourage users from interpreting them in order to find patterns of either novel or unexpected knowledge.

Patterns in the discovery process should be considered interesting when they represent *unknown* strong regularities, rare exceptions, or when they help to distinguish different groups of data. (POHLE, 2003). In geographic databases, however, there is a large number of patterns intrinsic to the data, which represent strong regularities, but do not add novel and useful knowledge to the discovery.

Geographic domain dependences are well known by specialists in geography or geographic database designers. They are mandatory spatial associations normally explicitly represented in geographic database schemas (BOGORNY, 2006b) and geo-ontologies (BOGORNY, 2005b). However, existing approaches for mining SAR do not make use of geographic domain knowledge to reduce the amount of non-novel patterns. Only little attention has been devoted to the questions of reducing non-novel associations extracted from geographic databases, and no prior knowledge has been used for this purpose. Existing approaches such as (KOPERSKI, 1995) (APPICE, 2003, 2005) (LISI, 2004) (CLEMENTINI, 2000) (MENNIS, 2005) use generalization/specialization concept hierarchies of geographic and non-geographic data to specify data at different granularity levels, but not to reduce the number of well known geographic domain patterns.

The association rule mining technique has emerged with the objective to find novel, useful, and interesting associations, *hidden* among data (AGRAWAL, 1993), but not *explicit* and *well known* associations. Existing association rule mining algorithms propose different thresholds and syntactic constraints for reducing the number of patterns and rules, but only the data by themselves have been considered, while the database schema, which is a rich knowledge repository, has not been used as prior knowledge to eliminate well known patterns.

In transactional association rule mining the schema might not be useful, since items and transactions can be stored in a single relation. In geographic databases, however, every different geographic object type is normally stored in a different relation, since most geographic databases follow the relational approach (SHEKHAR, 2003) (RIGAUX, 2002).

From the database design point of view, the objective of data modeling is to bring together all relevant object types of the application, their associations/relationships, and their constraints (SHEKHAR, 2003)(ELMASRI, 2003). Many geographic object types have mandatory associations, represented in the schema by *one-one* and *one-many* cardinality constraints, which the database designer has the responsibility to warrant when the schema is conceived. The representation is usually in the third normal form, intending to reduce anomalies and warrant integrity constraints (ELMASRI, 2003).

In contrast to database schema modeling, where associations between data are *explicitly* represented, association rule mining algorithms should find *implicit* and novel associations. While the former represents data into the third normal form, the latter usually denormalize data in one single table or one single file. This transformation brings the associations explicitly represented in the database schema to the dataset to be mined and, as a consequence, many well known associations, specified in the schema, are extracted by association rule mining algorithms and presented to the user.

Aiming to reduce the number of well known patterns in spatial association rule mining this thesis presents a deep study on the whole process of mining spatial association rules from geographic databases.

1.1 Objectives and Methodology

This research has the objective to investigate real problems in spatial association rule mining and propose solutions. The purpose is to use prior semantic knowledge to eliminate patterns and rules that are previously known as non-interesting.

The methodology adopted by this research includes the following main steps:

1) Develop a data preprocessing software prototype to generate different datasets from geographic databases to apply association rule mining algorithms.
2) Perform experiments with real datasets in order to identify the different types of well known patterns that may appear in SAR mining. This step includes experiments with different spatial relationships and geographic data at different granularity levels.
3) Perform case studies with real geographic database schemas in order to evaluate the amount of well known patterns explicitly represented in these knowledge repositories.
4) Investigate how semantic knowledge can be used to improve geographic data preprocessing and to reduce well known patterns in SAR.
5) Propose methods to eliminate well known patterns in SAR mining.
6) Perform experiments with real geographic databases in order to validate the proposed solutions.

1.2 Scope and Outline

The scope of this thesis is limited to the use of prior knowledge to reduce *well known* patterns in geographic data preprocessing and spatial association rule mining. It is important to emphasize that the focus of this thesis is on the elimination of well known geographic domain associations, and not on reducing redundant association rules which has been the objective of many works in transactional data mining.

The remaining of this thesis is organized as follows: Chapter 2 presents some background concepts about geographic data, spatial relationships, spatial integrity constraints, as well as geographic dependences. Indeed, two case studies with real geographic database schemas are presented in order to evaluate the large amount of well known geographic dependences that are explicitly represented in GDB schemas.

Chapter 3 introduces the concepts of frequent patterns and spatial association rules, the problems generated by geographic dependences in both data preprocessing and spatial association rule mining, and what has been done so far to reduce this problem.

Chapter 4 presents a framework to improve geographic data preprocessing and spatial association rule mining using prior knowledge. Three methods are presented to eliminate well known dependences in SAR mining.

Chapter 5 presents extensive experiments performed with real geographic databases to show the significant reduction of the number of frequent patterns and spatial association rules using prior knowledge.

Chapter 6 presents an overview about the implementation of the data preprocessing module of the proposed framework, and Chapter 7 concludes the thesis and suggests directions of future research.

2 GEOGRAPHIC DATABASES: BASIC CONCEPTS

Geographic databases (GDB) store real world entities/objects, also called spatial features, located in a specific region (OPEN GIS CONSORTIUM, 1999a). Spatial features (e.g. Canada, France) belong to a feature type (e.g. country), and have both non-spatial attributes (e.g. name, population) and spatial attributes (geographic coordinates *x*,*y*).

Spatial feature types in geographic databases are usually stored in different relations, because most geographic databases follow the relational or object-relational approach (SHEKHAR, 2003) (RIGAUX, 2002). Figure 2.1 shows an example of geographic data stored in relational databases, where the feature types street, water resource, and gas station are different relations (database tables) with spatial (shape) and non-spatial attributes.

(a) Street

Gid	Name	Shape
1	Itaqui	Multiline [$(x_1,y_1),(x_2,y_2),..$]
2	Ijui	Multiline [$(x_1,y_1),(x_2,y_2),..$]

(b) WaterResource

Gid	Name	Shape
1	Jacui	Multiline [$(x_1,y_1),(x_2,y_2),..$]
2	Guaiba	Multiline [$(x_1,y_1),(x_2,y_2),..$]
3	Uruguai	Multiline [$(x_1,y_1),(x_2,y_2),..$]

(c) GasStation

Gid	Name	DieselVol	GasVol	Shape
1	BR	20000	85000	Point[(x_1,y_1)]
2	ESSO	30000	95000	Point[(x_1,y_1)]
3	ELF	25000	120000	Point[(x_1,y_1)]

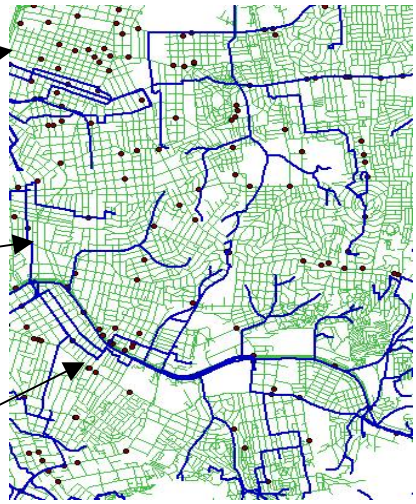


Figure 2.1: Examples of geographic data storage in relational databases

The spatial attributes of geographic object types, represented by *shape* in Figure 2.1, have intrinsic spatial relationships (e.g. close, far, contains, intersects). Because of these relationships real world entities can affect the behavior of other features in the neighborhood. This makes spatial relationships be the main characteristic of geographic data to be considered for data mining and knowledge discovery (ESTER, 2000) and the main characteristic which differs spatial data mining from classical data mining.

The process of extracting spatial relationships brings together many interesting and uninteresting spatial associations. Figure 2.2 shows an example of *implicit* and *non-standard* spatial relationships among gas stations, industrial residues repositories, and water resources. There is no explicit pattern among these data. Considering, for example, that water analyses showed high chemical pollution, the extraction of spatial relationships among water resources, gas stations, and industrial residues repositories will be interesting for knowledge discovery.

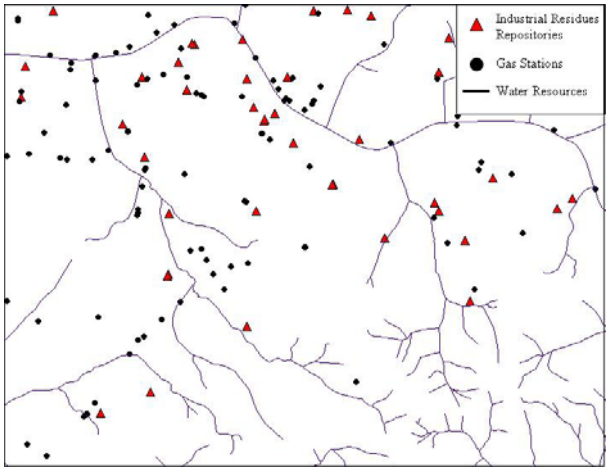


Figure 2.2: Examples of implicit spatial relationships

Figure 2.3 shows two examples of spatial relationships that represent well known geographic dependences. In Figure 2.3 (left), viaducts intersect streets and bridges intersect both water resources and streets, since both bridges and viaducts have the semantics of connecting streets. In Figure 2.3 (right), there is a *well known* geographic dependence, where every gas station intersects at least one street.

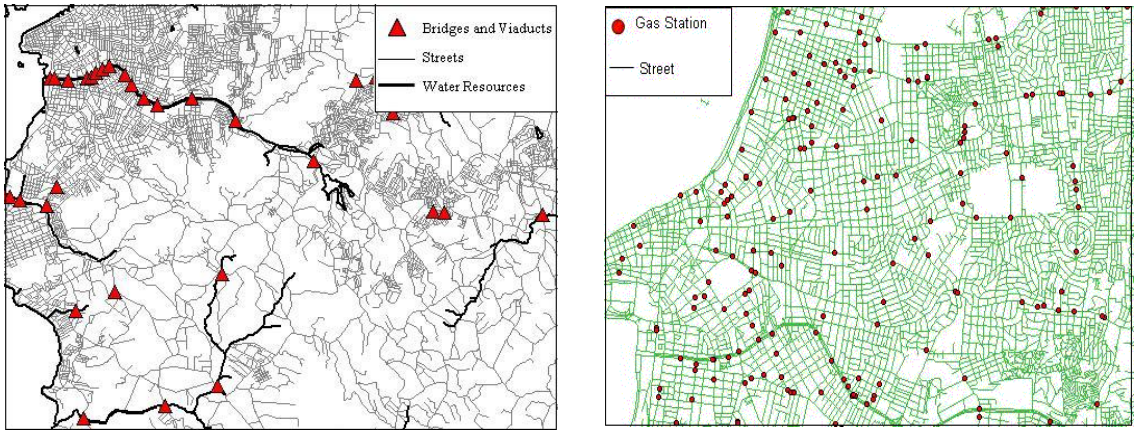


Figure 2.3: Examples of spatial relationships that produce well known geographic patterns

The main difference between the examples shown in Figure 2.2 and Figure 2.3 is that in the former spatial relationships may hold or not, and may conduce to more interesting patterns. In the latter, under rare exceptions or some geographic location inconsistency, the spatial relationships hold for practical purposes in a 100% of the cases, and will produce well known geographic domain patterns in the discovery process. If considered in association rule mining, well known spatial relationships will generate rules such as *is_a*(*Viaduct*) → *intersects*(*Street*) or *is_a*(*GasStation*) → *intersects*(*Street*).

Well known geographic dependences are mandatory spatial relationships that represent spatial integrity constraints (SERVIGNE, 2000) which must hold in order to warrant the consistency of the data. They are normally explicitly represented in geographic database schemas, as will be shown later with two case studies.

The following sections present an overview of different types of spatial relationships, whose relationships represent well known dependences and how they are represented in geographic database schemas.

2.1 Spatial Relationships and Spatial Integrity Constraints

There are basically 3 types of spatial relationships to consider: *distance, direction,* and *topological* (GUTING, 1994). Distance relationships are based on the Euclidean distance between two spatial features, as shown in Figure 2.4 (a). Let *dist* be a distance function, *operator* be an arithmetic predicate from the set {<, >, >=, <=, =}, let *d* be a real number and let *A* and *B* be spatial features, the distance relationship between *A* and *B* is expressed as *dist* (A, B) *operator d*.

Direction/Order relationships deal with the order as spatial features are located in space in relation to each other or in relation to a reference object, as shown in Figure 2.4 (b).

Topological relationships characterize the type of intersection between two spatial features, and remain invariant under topological transformations such as rotating and scaling. There are many approaches in the literature to formally define a set of topological relationships among points, lines, and polygons. Most of them are based on the intersections model (EGENHOFER 1991, 1995), being co-related sharing interior and boundary (4-intersection model), or interior, boundary, and exterior (9-intersection model). These intersections may be combined by the logical operators *and* ($\wedge$) and *or* ($\vee$). Hadzilacos (1992) extended Egenhofer's approach for the combinations of points, lines, and polygons. The intersection models provide 8 binary topological relations: *crosses, contains, within, covers, coveredBy, equals, disjoint,* and *overlaps*.

The topological relationships can also be defined according to the calculus based method (CBM) and the dimension extended method (DEM), both proposed by Clementini (1993). These methods define 6 topological relationships: *contains, within, equals, crosses, overlaps,* and *disjoint*. Figure 2.4(c) shows some examples of topological relationships.

At a high concept level topological relationships are either *disjoint* or *connected* (intersect). When objects are connected, then they can have only one topological relationship of type *crosses, contains, within, covers, coveredBy, equals,* and *overlaps*.

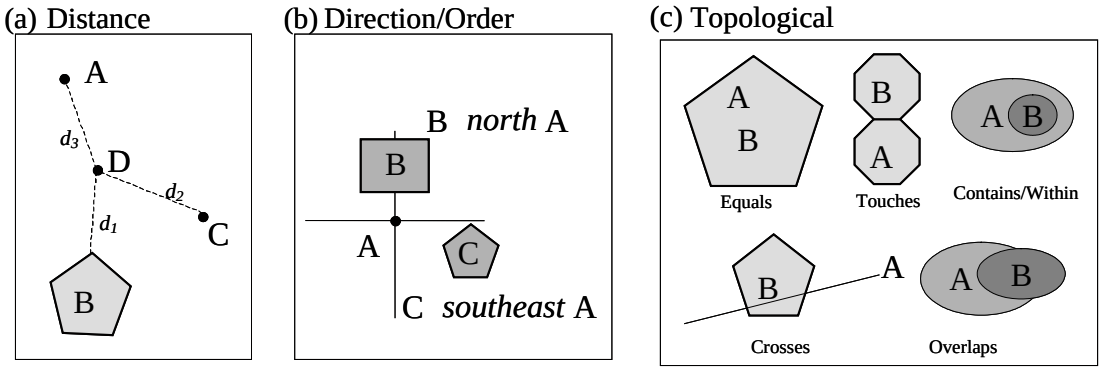


Figure 2.4: Examples of spatial relationships

Spatial relationships between two spatial features can be *possible*, *mandatory* or *prohibited*. *Possible* relationships can either exist or not (e.g. "roads cross rivers", "counties contain factories") in the database. *Mandatory* and *prohibited* spatial relationships represent spatial integrity constraints (SERVIGNE, 2000) which must hold if the database is consistent.

Spatial integrity constraints encompass the peculiarities of geographic data and their spatial relationships (COCKCROFT, 1997). Their purpose is to warrant as well as maintain both the quality and the consistency of spatial features in geographic databases. Spatial integrity constraints between two spatial feature types *A* and *B* can be specified by relationships with cardinality constraints (SERVIGNE, 2000) (BOGORNY, 2001) (SHEKHAR, 2003 pp.37). For instance, a mandatory relationship between gas station and street can be represented by the cardinality *one-one* or *one-many*, such that every gas station must be related to at least one street. Mandatory constraints represent well known *geographic dependences* which generate well known patterns when considered in SAR mining.

2.2 Geographic Dependences

In geographic space, "*everything is related to everything else, but nearby things are more related than distant things"* (TOBLES apud SHEKHAR, 2003, p. 186). However, some things are *always* related to others. When this happens, then we can say that there is a *geographic dependence*.

Definition 1 (geographic dependence) is a mandatory spatial relationship between two geographic feature types *A* and *B* where every instance of *A* must be spatially related to at least *one* instance of *B.*

Geographic dependences are *well known* because they are explicitly represented by database designers in geographic database schemas in order to warrant the spatial integrity of geographic data. Geographic database schemas are normally extended relational or object-oriented schemas (SHEKHAR, 2003). There is an emerging trend toward extending both Entity Relationship (ER) and Object-Oriented (OO) diagrams with pictograms to provide special treatment to spatial data types (SHEKHAR, 2003, pp.205). (PARENT, 1998) (BORGES, 2001) (ROCHA, 2001) (LISBOA, 2000) are approaches which extend ER and OO diagrams for geographic applications. In both ER and OO approaches, mandatory relationships among entities are represented through associations with cardinality constraints (SHEKHAR, 2003). In geographic database schemas, geographic dependences may be defined as a spatial relationship (e.g. touches,

contains) or as a single association or aggregation with cardinalities *one-one* or *one_many*.

Figure 2.5 shows an example of part of a conceptual geographic database schema, represented in a UML class diagram (BOOCH, 1998), and part of its respective logical schema for relational and OO databases. The schema in Figure 2.5 represents some of the spatial feature types shown in Figure 2.2 and Figure 2.3. Notice that there are many mandatory associations (e.g gas station and street, street and county, water resource and county, and island and water resource) while possible relationships which do not represent well known dependences and may be interesting for KDD are not represented in the schema (e.g. gas stations and water resources).

In the logical level, mandatory relationships expressed by cardinalities *one-many* and *one-one* normally result in foreign-keys in relational geographic databases, and in pointers to classes, in object-oriented geographic databases (ELMARSI, 2003).

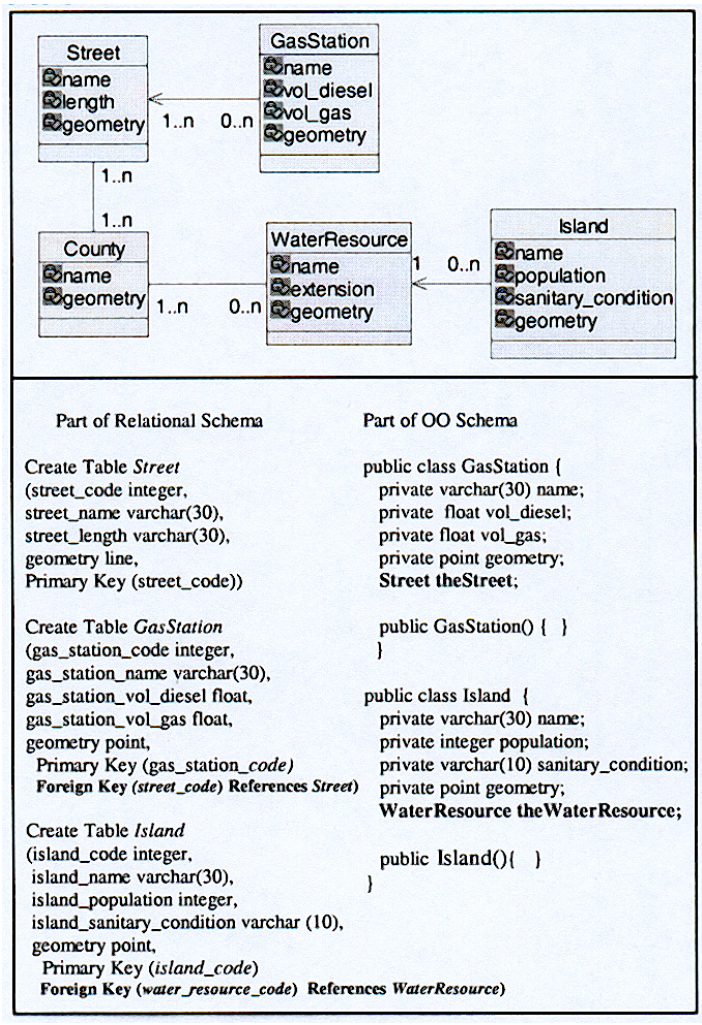


Figure 2.5: Part of a conceptual and logical geographic database schema

In (BOGORNY, 2006b) we presented an algorithm to extract geographic dependences from database schemas.

In order to evaluate the amount of well known geographic dependences explicitly represented in real geographic database schemas, two case studies with real schemas were performed: the Brazilian Army data model, which has been the basis to construct

geographic maps for the whole country, and the data warehouse developed in the project *iPara* for the Para state, in Brazil.

2.3 Geographic Database Schemas: a Case Study

The geographic database schema developed by the Brazilian Army contains most geographic elements that are part of the Brazilian terrain model, which under a few variations, is similar to any terrain model represented in geographic databases. Because of the large number of objects and relationships to be represented, geographic data conceptual schemas are usually designed in different packages/superschemas.

The geographic database schema developed by the Brazilian Army follows the conceptual framework *GeoFrame* proposed in (LISBOA, 2000), and has 8 main packages: edification, infra-structure, hydrography, vegetation, administrative regions, referential, relief, and toponymy. The package infra-structure, for example, is divided in six sub-schemes, including information about transportation, energy, economy, communication, etc. The hydrography package, for instance, represents objects such as streams, oceans, and lakes.

Information of different packages may be extracted for data mining, and the number of *one-one* and *one-many* relationships varies from one package to another. For example, the hydrography package, which is shown in Figure 2.6, has a total of 24 geographic objects types (16 from its own package and 8 from others) which share a total of 15 *one-many* and *one-one* relationships when super classes are concrete, and more that 20 if super classes are abstract.

The infra-structure level, for example, has 73 geographic objects in its own package and has relationships with 88 objects from other packages. Among the 88 relationships, 70 are mandatory *one-one* dependences.

The transportation level, shown in Figure 2.7, has a total of 20 objects and 18 associations/aggregations, among which, 10 are *one-one* or *one-many*. In this schema we have two classical well known associations between gas stations and roads as well as roads and crossways.

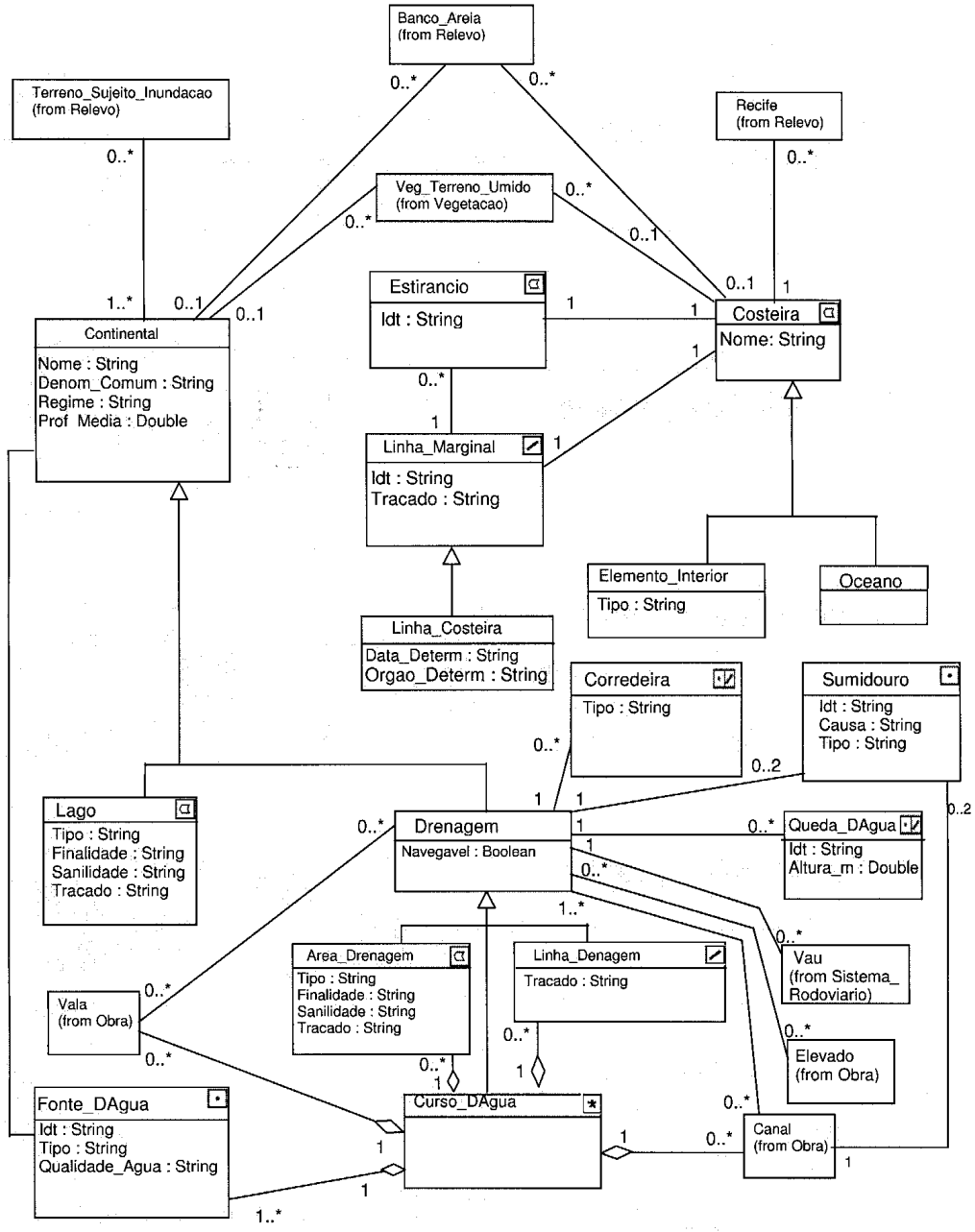


Figure 2.6: Hydrographic conceptual object-oriented schema of the Brazilian Geographic Territory (MCOO of EBG - Brazilian Army – STI – DSG - 1°DL)

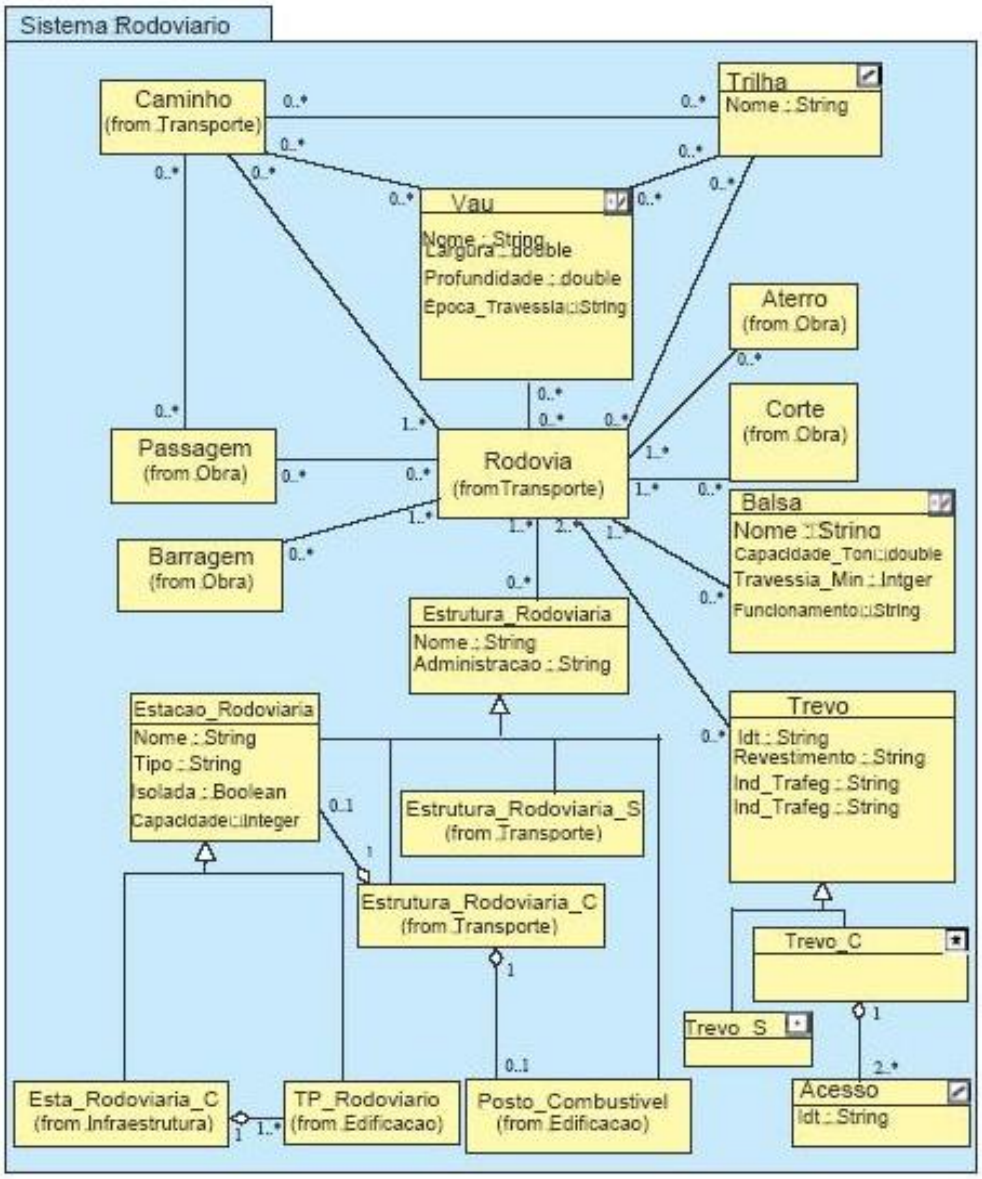


Figure 2.7: Transportation conceptual object-oriented schema of the Brazilian Geographic Territory (MCOO of EBG - Brazilian Army – STI – DSG - 1°DL)

The conceptual schema of the project *iPara* is a geographic data warehouse developed in cooperation with geographic database research group SIGMODA of II/UFRGS, COHAB-PA, and SEIR-PA. It integrates general geographic data of the state of Para (SIGIEP) and the urban geographic database (SIME). The *iPara* conceptual model is designed according to the *GeoFrame-T* framework proposed by (ROCHA, 2001).

The complete conceptual schema of *iPara* has more than 20 different schemas including Hidrography, Infra-Structure, and Transportation.

The Transportation package, for example, shown in Figure 2.8, has 27 objects with 16 *one-one* and *one-many* associations.

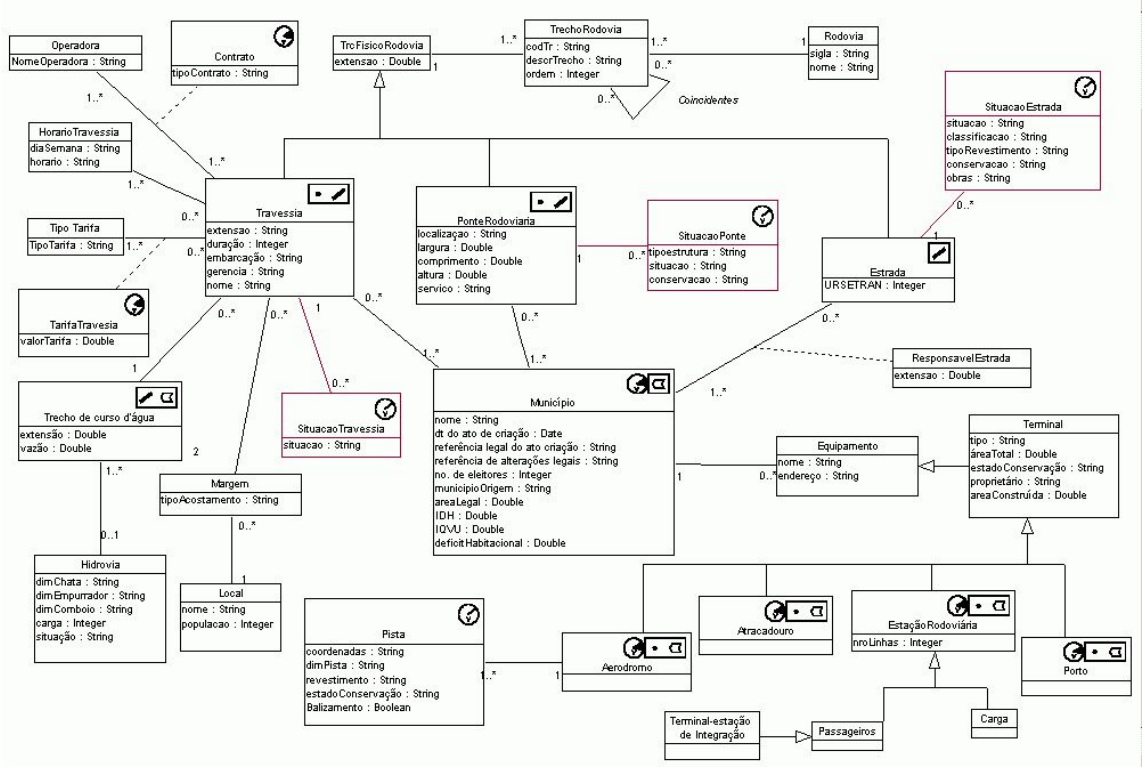


Figure 2.8: Transportation schema of the *iPara* project

In the Water Distribution schema, shown in Figure 2.9, among 7 objects, all 5 relationships are *one-many* or *one-one* associations.

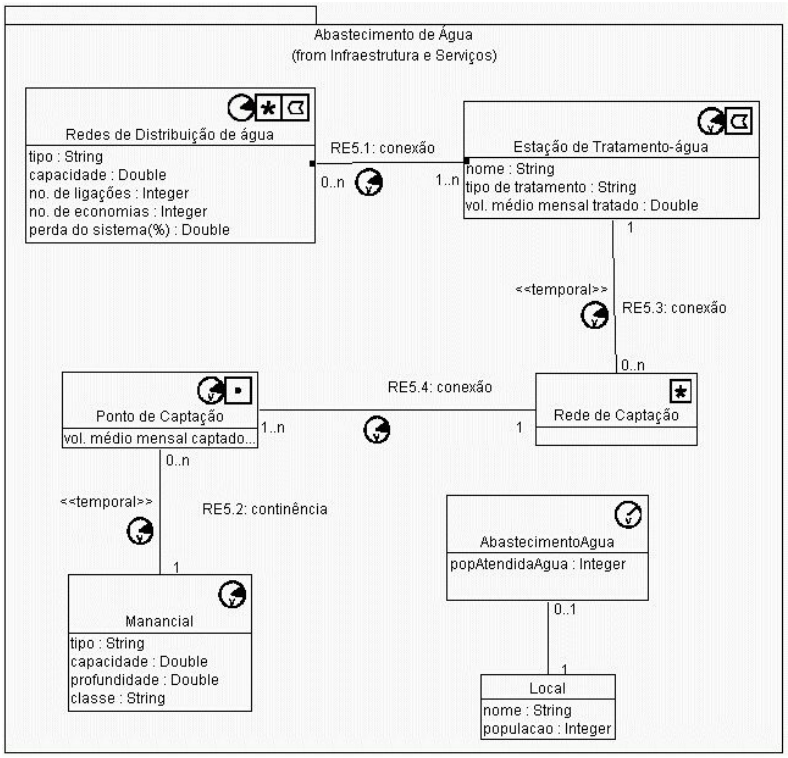


Figure 2.9: Water distribution conceptual schema of the *iPara* project

In the schema shown in Figure 2.10, for example, the 15 relationships are all *one-one* or *one-many* associations. For all theses pairs of geographic objects, if they together

pass some basic constraints such as minimum support in SAR mining, a large number of well known rules will be generated having these dependences.

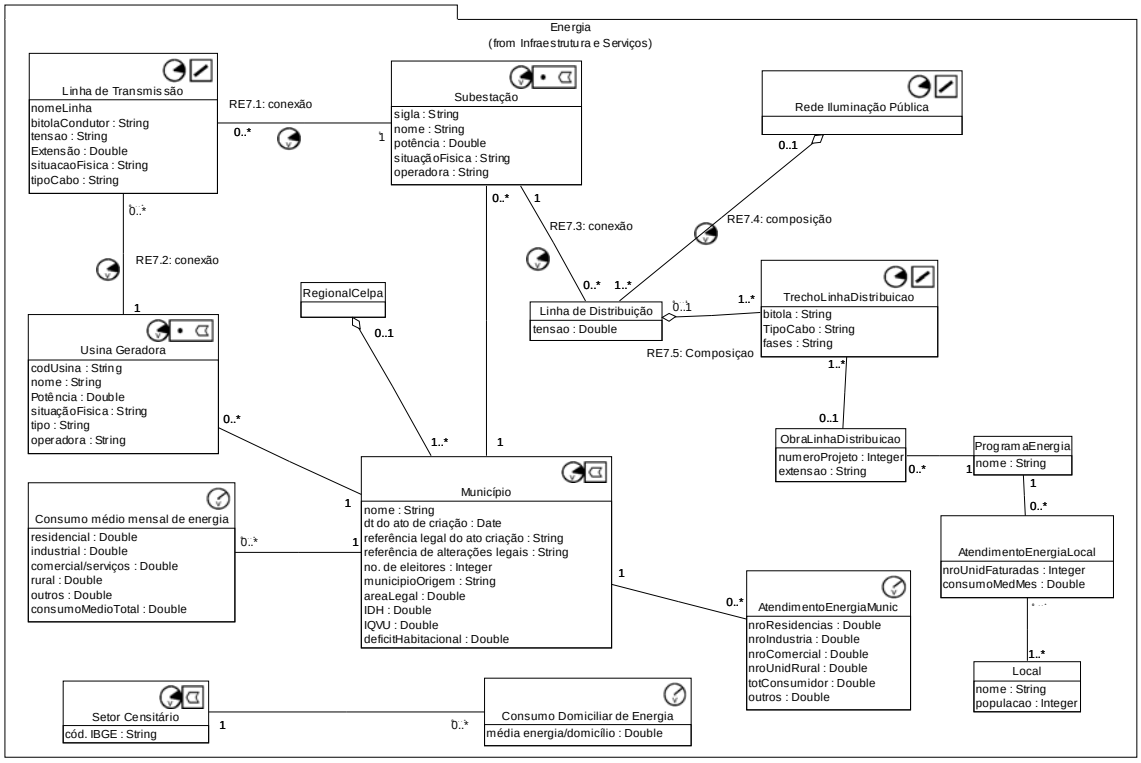


Figure 2.10: Energy conceptual schema of the *iPara* project

The case study with two real GDB schemas showed that a large number of mandatory well known geographic dependences is explicitly represented in geographic conceptual data modeling.

An interesting study which shows that mandatory *one-one* and *one-many* associations produce well known patterns can be found in (SILVA, 2003), that uses the association rule mining technique for mining geographic database schemas. The study had the objective of finding strong spatial patterns in order to infer candidates of design patterns for geographic conceptual data modeling. The main objective was to create a pattern catalogue to help the database designer in GDB conceptual data modelling. Many discovered patterns in such study refer to associations with cardinalities *one-one* or *one-many*, which are exactly those we argue that produce well known patterns when the objective is to find novel and useful knowledge.

Not only geographic database schemas are rich knowledge repositories that can be used in SAR mining to reduce well known patterns. Geographic ontologies store a large amount of semantic knowledge that can be used to improve geographic data preprocessing (BOGORNY, 2005b) and reduce well known patterns in spatial association rule mining (BOGORNY, 2006c), as will be briefly introduced in the following section.

2.4 Geo-Ontologies and Spatial Integrity Constraints

Ontology is an explicit specification of a conceptualization (GRUBER, 1993). More specifically ontology is a logic theory corresponding to the intentional meaning of a formal vocabulary, that is, an ontological commitment with a specific conceptualization of the world (GUARINO, 1998). It is an agreement about the concepts meaning and structure for a specific domain. Each concept definition must be unique, clear, complete and non-ambiguous. The structure represents the properties of the concept, including a description, attributes, and relationships with other concepts.

Ontologies have been used recently in many and different fields in Computer Science, such as Artificial Intelligence, Databases, Conceptual Modelling, Semantic Web, etc. Because of this reason, a relevant number of ontologies have already been proposed, and a number of models, languages and tools were developed. Chaves (2005a) besides defining a geo-ontology for administrative data for the country of Portugal, defines a meta-model, named GKB (Geographic Knowledge Base), which is a starting point to define an ontology for geographic data.

In geo-ontologies, spatial integrity constraints are represented by properties of geographic data. They are specified as restriction properties, i.e., are defined as a spatial or non-spatial relationship with the corresponding minimum and maximum cardinalities. For instance, a concept *island*, which is a piece of land surrounded by water, must have a mandatory *one-one* relationship with the concept *water*.

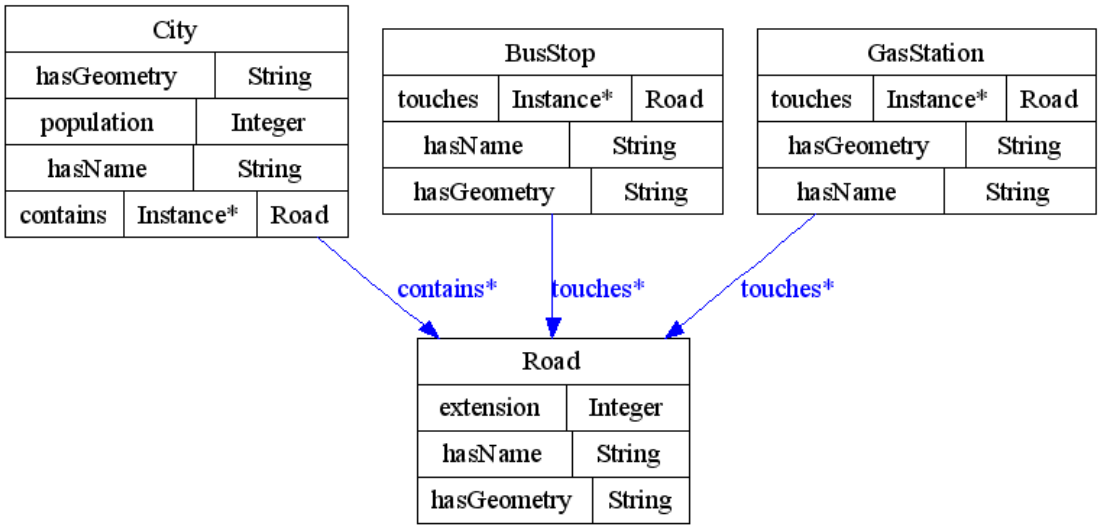
Figure 2.11 shows a small example of a geographic ontology with the specification of different topological relationships in order to illustrate how mandatory semantic constraints are represented.

In the example in Figure 2.11, bus stop and gas station have a mandatory constraint with road because every gas station and every bus stop must topologically *touch* one or more instances of road. Roads, however, do not necessarily have gas stations or bus stops. The uni-directional association represents the mandatory relationships that cities, bus stops and gas stations have with Road. Notice in the OWL representation that minimum cardinality 1 is explicitly represented and can be easily retrieved.

To evaluate the amount of well known dependences in real geo-ontologies we analysed the first geo-ontology of Portugal, named geo-net-pt01 (CHAVES, 2005b). Although not all elements of the geographic domain have been defined in geo-net-pt01, there are many *one-one* and *one-many* dependences.

The repository of the geo-ontology stores 3 levels of information: geo-administrative, geo-physical, and network. The geo-administrative level stores administrative information about territorial division, and includes geographic feature types such as municipalities, streets, etc. The network level stores non-spatial data and relationships about the geo-administrative layer (e.g population of a district). The geo-physical level stores feature types including continents, oceans, lakes, bays, water bodies, etc.

In geo-net-pt01, among 58 different spatial feature types, we found 55 *one-one* relationships.




```
<owl:Class rdf:ID="Bus Stop">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:valuesFrom rdf:resource="#Road"/>
    <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int" >1</owl:minCardinality>
    <owl:onProperty>
     <owl:ObjectProperty rdf:ID="touches"/>
    </owl:onProperty>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
 </owl:Class>
 <owl:Class rdf:ID="GasStation">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int" >1</owl:minCardinality>
    <owl:onProperty>
     <owl:ObjectProperty rdf:about="#touches"/>
    </owl:onProperty>
    <owl:valuesFrom rdf:resource="#Road"/>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
 </owl:Class>
```

Figure 2.11: Geo-Ontology representation and OWL code

In this chapter we introduced geographic databases, spatial relationships, and well known geographic dependences which are represented by cardinality constraints one-one and one-many in geographic database schemas and geo-ontologies. In the following chapter we introduce the concept of non-spatial and spatial association rules as well as studies that show different problems identified in the process of mining spatial association rules.

3 THE PROLEM OF MINING ASSOCIATION RULES IN GEOGRAPHIC DATABASES

Association rules have been largely used to extract patterns from spatial and non-spatial databases. The main difference between non-spatial and spatial association rules are the spatial aspects (spatial relationships) of geographic data that must be considered in spatial association rule mining. Among the spatial relationships there are many well known associations which generate a large number of well known patterns and association rules. In this chapter we introduce spatial and non-spatial association rules and explain the main problems that well known geographic associations generate in the association rule mining technique and what has been done so far to overcome these problems.

3.1 Association Rules

Association rules consist of an implication of the form $X \rightarrow Y$, where X and Y are sets of items co-occurring in a given tuple (AGRAWAL, 1993). The formal problem statement for defining association rules can be specified as follows: let $F = \{f_1, f_2,\ldots,f_k, \ldots, f_n\}$ be a set of items, and Ψ(dataset) be a set of rows (transactions) W, where each W is a set of items (tuple) such that $W \subseteq F$. Each W is represented as a binary vector, with an element $w[k] = 1$, if W contains the attribute f_k, and $w[k] = 0$, otherwise. There is exactly one row in the dataset to be mined for each transaction. Considering X as a subset of F, W contains X if, for all f_k in X, $w[k] = 1$. Similarly, being Y a subset of F, W contains Y if, for all f_k in Y, $w[k] = 1$.

An association rule consists of an implication of the form $X \rightarrow Y$, where $X \subset F$, $Y \subset F$ and $X \cap Y = \varnothing$. The support s of an itemset X is the percentage of rows in which the itemset X occurs as a subset. The support of the rule $X \rightarrow Y$ is given as $s(X \cup Y)$.

The rule $X \rightarrow Y$ is satisfied in Ψ with confidence factor $0 \leq c \leq 1$, if at least c% of the instances in Ψ that satisfy X also satisfy Y. The notation $X \rightarrow Y\ (c)$ specifies that the rule $X \rightarrow Y$ has confidence factor of c. More precisely, the confidence factor is given as $s(X \cup Y)/s(X)$.

The problem of mining association rules can be decomposed in two steps (AGRAWAL, 1993):

- *Find all large/frequent itemsets*: an itemset is frequent if its support is at least equal to a certain threshold, called *minsup*.

- *Generate high confidence rules*: a rule is strong if its support is at least equal to minimum support and the confidence is higher or equal to a certain threshold, called *minconf.*

Assertion 1. (AGRAWAL, 1993) If a predicate set Z is large, then every subset of Z will also be large. If the set Z is not large, then every set that contains Z is not large too. All rules derived from Z satisfy the support constraint if Z satisfies the support constraints.

Association rule mining algorithms under rare exceptions (HAN, 2000, 2004) generate *candidate sets* and then compute their frequency, in order to generate *frequent* sets, as in Apriori (AGRAWAL, 1994). The candidate generation is performed with multiple passes over the dataset. In the first pass, the support of the individual elements is computed to determine large-itemsets, called *frequent k-itemsets*. In the subsequent passes, given k as the number of the current pass, the large sets L_{k-1} in the previous pass (k -1) are grouped into sets C_k with k elements, which are called *candidate sets*. The support of each candidate set is computed, and if it is equal or higher than the minimum support, then this set is considered *frequent*. This process continues until the large set in the pass results in an empty set. Association rules are extracted from the resultant *frequent sets*, i.e., candidate sets that reached minimum support.

Figure 3.1 (a) shows an example of a dataset with six transactions and five items (A,C,D,T,W). In Figure 3.1 (b) are the k frequent sets with minimum support 50%, i.e., which appear in at least 50% of the transactions.

a) dataset

Tid	itemset
1	A, C, D,T, W
2	C, D, W
3	A, D, T, W
4	A, C, D, W
5	A, C, D, T, W
6	C, D, T

b) frequent itemsets with minimum support 50%

Set *k*	Frequent sets
k=1	{A}, {C}, {D}, {T}, {W}
k=2	{A,C}, {A,D}, {A,T}, {A,W}, {C,D}, {C,T}, {C,W}, {D,T}, {D,W}, {T,W}
k=3	{A,C,D}, {A,C,W}, {A,D,T}, {A,D,W}, {A,T,W}, {C,D,T}, {C,D,W}, {D,T,W}
k=4	{A,C,D,W}, {A,D,T,W}

Figure 3.1: Dataset with 6 tuples and frequent sets with minimum support 50%

A huge number of algorithms for non-spatial data has been proposed to reduce the computational time, the number of candidates or frequent sets (also called patterns), and the number of association rules. Algorithms that aim to reduce both the number of frequent sets and association rules, in general, can be classified in two types: Apriori-like approaches, which generate *frequent sets* and define different measures to reduce the number of rules; and approaches that generate *closed frequent sets*, which reduce the number of frequent sets and redundant rules.

Although there are still many works that propose different methods to either generate or prune association rules, the rule generation is considered a trivial step (BURDICK, 2001). Research has focused on the frequent set generation, which is the most time consuming step (UNO, 2004) (BURDICK, 2001) (HAN, 2000) in the association rule mining process. In 2003 and 2004, for example, the FIMI (Frequent Itemset Mining Implementations) Workshop held in conjunction with the IEEE ICDM (International Conference on Data Mining) focused on different methods for mining and

implementating *frequent itemsets.* Some of the presented works include (UNO, 2003, 2004)(ORLANDO,2003) (GOETHALS, 2003).

Apriori-like approaches discuss objective and subjective interestingness of the rules. However, according to (BAYARDO, 1999a) it is difficult to come up with a single metric that quantifies the "interestingness" or "goodness" of an association rule. As a result, several metrics have been proposed and used, such as the already mentioned support and confidence, entropy gain (MORIMOTO,1998), gini (FUKUDA, 1996), improvement (BAYARDO, 1999b), conviction (BRIN, 1997), etc. In most approaches, non-interesting rules are normally eliminated during the rule generation, i.e., a posteriori, when frequent sets have already being generated.

In (BAYARDO, 1999a, 1999b) (TAN, 2002) (SRIKANT, 1997) (FREITAS, 1998) (MELANDA, 2004) different thresholds and item constraints are applied and only rules that satisfy the constraints are generated and presented to the user. (PADMANABHAN, 1998) and (SILBERCHATZ, 1996) propose methods that consider a set of expectations or beliefs about a domain, and patterns are considered interesting when they are unexpected and contradict believes. These approaches require the definition of complex belief information, such as conditional probabilities, which according to (LIU, 2000) are difficult to obtain in practice. Srikant (1995), for example, used concept hierarchies to eliminate candidate sets that contain parent (e.g. cloth) and child (e.g. jacket, dress) of a hierarchy in the same set. In practice it is not common to mine the same data at different granularity levels in one single mining process. This method reduces rules that should be avoided in data preprocessing such as *jacket=yes* → *clothes=yes*. Such data should not be considered together in the same mining process.

Approaches that generate closed frequent sets and its variations such as (PEI, 2000) (PASQUIER, 1999a) (BASTIDE, 2000) (ZAKI 2000,2002) (BONCHI, 2003a,2003b,2004) (HAN, 2000), compute frequent sets followed by the elimination of those which are not closed. To understand the concept of closed frequent sets, let us consider the dataset in Figure 3.2 (a), the frequent itemsets and the transactions where they occur (Figure 3.2 c) and their closed frequent sets shown in Figure 3.2 (b). The set {A,D,W}, for example, is a *frequent itemset* because it reaches minimum support (50%). It is also a *closed frequent itemset* because in the set of transactions (1345) where it occurs in the dataset, no set larger than {A,D,W} (with more than 3 elements) in the same transactions reaches minimum support. The frequent set {A,D,T}, for example, appears in the transactions 135, but in the same transactions, a larger set {A,D,T,W} can be generated. In this case the tidset(A,D,T) = 135 and the tidset(A,D,T,W) = 135 and $\{A,D,T\} \subset \{A,D,T,W\}$, so the frequent itemset {A,D,T} is not closed.

Definition 2 (Closed Frequent Set) (PASQUIER, 1999a)**:** a frequent itemset L is a closed frequent itemset if $\Omega(L)=L$.

The closure operator Ω associates with a frequent itemset L the maximal set of items common to all transactions (tidset) containing L. The closure operator allows the definition of all frequent closed itemsets which constitute the minimal non-redundant frequent sets. Non-closed frequent itemsets have same support as its respective closed frequent itemset, so the maximal frequent itemsets are maximal closed frequent sets. This property warrants that no information is lost and that rules generated from non-closed frequent itemsets are redundant to rules generated from their respective closed frequent sets.

a) dataset

Tid	itemset
1	A, C, D,T, W
2	C, D, W
3	A, D, T, W
4	A, C, D, W
5	A, C, D, T, W
6	C, D, T

b) *closed* frequent itemset

Set *k*	Closed frequent sets
k=1	{D}
k =2	{C,D}, {D,T}, {D,W}
k =3	{A,D,W}, {C,D,T}, {C,D,W}
k =4	{A,C,D,W}, {A,D,T,W}

c) frequent itemsets in the same tidset

TidSet	Frequent itemsets *L*
123456	**{D}**
12456	{C}, **{C,D}**
12345	{W}, **{D,W}**
1245	{C,W}, **{C,D,W}**
1345	{A}, {A,D}, {A,W}, {**A,D,W}**
1356	{T}, {**D,T}**
145	{A,C}, {A,C,W}, {A,C,D}, {**A,C,D,W}**
135	{T,W}, {A,T}, {A,D,T}, {A,T,W}, {D,T,W}, {**A,D,T,W}**
156	{C,T},{**C,D,T}**

Figure 3.2: Dataset with 6 tuples and *closed* frequent sets with minimum support 50%

In Figure 3.2 (c), the maximal frequent itemset for the transactions 12345 is {D,W}. For the transactions 1245, the set {C,W}⊂{C,D,W}, so {C,D,W} is the maximal. For the transactions 145, notice that {A,C}⊂{A,C,D,W}, {A,C,W}⊂{A,C,D,W}, {A,C,D}⊂{A,C,D,W}, so {A,C,D,W} is maximal. For the transactions 135, the most general frequent itemset is {A,D,T,W}, and all other sets in these transactions will generate redundant rules. According to (PASQUIER, 1999a), all frequent sets *L* that occur in the same transactions generate rules with same support and same confidence. As the maximal *L* for each set of transactions contains the maximal number of elements, all other sets generate redundant rules.

Definition 3 (Minimal non-redundant rule) (BASTIDE, 2000)**:** an association rule *r:* $l_1 \rightarrow l_2$ is a minimal non-redundant association rule if there is no rule *r'*: $l'_1 \rightarrow l'_2$ with support (*r*)=support(*r'*), confidence (*r*)=confidence(*r'*), $l'_1 \subseteq l_1$, and $l'_2 \subseteq l_2$.

Considering definition 3, a rule generated from a frequent itemset {A,W}, such as A→W, is redundant in relation to a rule A→DW, generated from the closed frequent itemset {A,D,W}. The closed frequent set generation does not warrant the elimination of all redundant rules, although they eliminate redundant frequent sets. There are many papers that propose to reduce redundant rules extracted from closed frequent sets (ZAKI, 2000) (LIU, 1999) (BASTIDE, 1999a). In (ZAKI, 2000) (LIU, 1999) (PASQUIER, 1999b) different methods are presented to remove either redundant or insignificant rules generated from closed frequent sets. However, redundant rules are defined based on different concepts. In the literature there are more than 5 definitions

for non-redundant rules, and according to these definitions, different pruning methods are proposed.

For example, in (ZAKI, 2000), from a closed frequent itemset {A,B,C,D} the most general non-redundant rule is the shortest rule, i.e., with the lowest number of elements. According to (ZAKI, 2000), considering that association rules that can be generated from a closed frequent set have same support and same confidence, only the shortest rule is non-redundant. So a rule such as A→B could be the most general non-redundant rule extracted from a frequent set {A,B,C,D}. According to (PASQUIER, 1999b), which introduced the closed frequent pattern mining method, minimal non-redundant rules are all those that have same support, same confidence, and smaller antecedent and larger consequent. For example A→BCD.

In the following section we introduce SAR, the different types of non-interesting SAR generated because of geographic dependences, and what has been done so far to reduce these problems.

3.2 Spatial Association Rules

Spatial association rules consist of an implication of the form $X \rightarrow Y$, where X and Y are sets of predicates, and at least one element in X or Y is a spatial predicate (KOPERSKI, 1995).

While in *transactional* association rule mining every row in the dataset is usually a transaction and columns are items, in *spatial* association rule mining every row is an instance (e.g. Porto Alegre) of a reference object type (e.g. city), called *target feature type*, and columns are predicates. Every predicate is related to a non-spatial attribute (e.g. population) of the target feature type or a spatial predicate. Spatial predicate is a *relevant feature type* that is spatially related to specific instances of the target feature type (e.g. contains_factory). In SAR mining the set $F = \{f_1, f_2, \ldots, f_k, \ldots, f_n\}$ is a set of non-spatial attributes and spatial predicates, and Ψ (dataset) is a set of instances of a reference feature type, where each instance is a row W such that $W \subseteq F$. There is exactly one tuple in the dataset Ψ for each instance of the reference feature type.

While the problem of mining non-spatial association rules is performed in two steps, the problem of mining *spatial* association rules is decomposed in at least three main steps, where the first one is usually performed as a data preprocessing method because of the high computational cost:

a) *Extract spatial predicates*: spatial predicate is a spatial relationship (e.g. distance, order, topological) between the reference feature type and a set of relevant feature types;

b) *Find all frequent patterns/predicates/sets*: a set of predicates is a frequent pattern if its support is at least equal to a certain threshold, called *minsup*;

c) *Generate strong rules*: a rule is strong if it reaches minimum support and the confidence is at least equal to a certain threshold, called *minconf.*

It is well known that spatial joins to extract spatial predicates are the processing bottleneck in spatial data mining, but only little attention has been devoted to this problem. In (LU, 1993; KOPERSKI, 1995) a top-down progressive refinement method

is proposed and spatial approximations are calculated in a first step, and in a second step, more precise spatial relationships are computed to the outcome of the first step. The method has been implemented in the module Geo-Associator of the GeoMiner system (HAN, 1997), which is no longer available. Ester (2000) proposed new operations such as graphs and paths to compute spatial neighborhoods. However, these operations are not implemented by most GIS, and to compute all relationships between all objects in the database in order to obtain the graphs and paths is computationally expensive for real databases. Appice (2003) proposed an upgrade of Geo-Associator to first-order logic, and all spatial relationships are extracted. This process is computationally expensive and many spatial relationships might be unnecessarily computed (KLOSGEN, 2002). In (BOGORNY, 2005b, 2007) we proposed to use geo-ontologies as prior knowledge to compute only topological relationships semantically consistent, and only among a target feature type and relevant feature types specified by the user. While the above approaches consider different spatial relationships and any geometric object type, a few approaches such as (SHEKHAR, 2001; HUANG, 2004; YOO, 2005) compute only distance relationships for point object types.

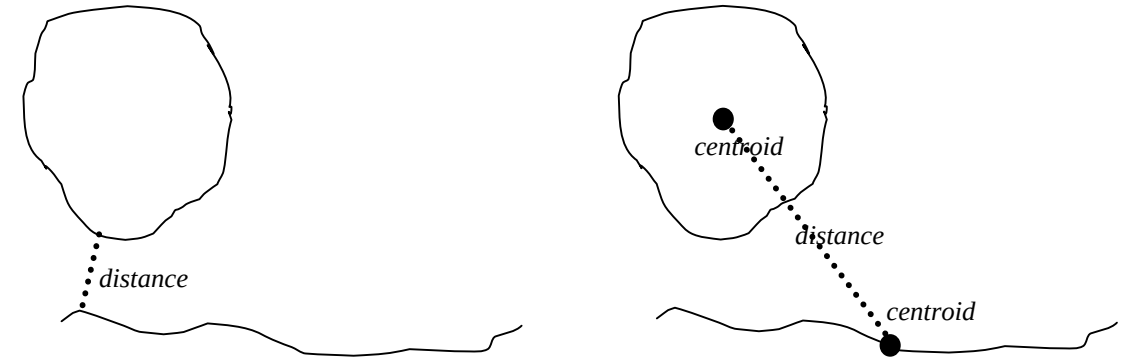
Spatial relationships are computed with spatial joins between all instances *t* (e.g. Porto Alegre) of a target feature type *T* (e.g. city) and all instances *o* (e.g. rio de la Plata) of every relevant feature type *O* (e.g. river) in a set of relevant feature types *S* (e.g. river, port, street, factory) that have any spatial relationship (e.g. touches, contains, close, far) with *T*. Being *T* a set of instances $T=\{t_1, t_2,\ldots,t_n\}$, $S = \{ O_1, O_i,\ldots, O_m\}$, and $O_i = \{ o_1, o_2,\ldots, o_q\}$, the extraction of spatial predicates implies the comparison of every instance of *T* with every instance of *O*, for all *O* in *S*.

Existing spatial association rule mining algorithms are in general Apriori-like approaches, i.e., generate candidates and frequent sets, and then extract association or co-location rules. In SAR mining the candidate generation is not a problem as it is in transactional databases. According to (SHEKHAR, 2003 pp.205) the number of predicates is much smaller than the number of items in transactional databases. Therefore, the computational cost relies on the spatial predicate extraction (step a), and depends on the number of instances of the target feature type and the relevant feature types, as well as their respective geometric representation.

The number of spatial association rule mining algorithms is much smaller than transactional rule mining algorithms, and can be classified in two main types. The first is based on *quantitative reasoning*, which mainly computes distance relationships during the frequent set generation. These approaches (SHEKHAR, 2001) (HUANG 2004) (YOO, 2005) deal with geographic data (coordinates *x,y*) directly. Although they have the advantage of not requiring the definition of a reference object, they have some general drawbacks: usually deal only with points, consider only quantitative relationships, and do not consider non-spatial attributes of geographic data, which may be of fundamental importance for knowledge discovery. For spatial objects/features represented by lines or polygons, their centroid[1] is extracted. Indeed, geographic coordinates are transformed into integer values, which reduce precision still further. This process loses significant information, and generates non-real patterns (e.g. the

[1] the center of mass of an object of uniform density

Mississippi River *intersects* many states considering its real geometry, but is *far from* the same states if only the centroid is extracted). Figure 3.3 shows an example of how the distance relationship can vary between two spatial features *A* and *B* when considering their original geometry (Figure 3.3a) and their centroid (Figure 3.3b).



(a) Distance between polygon and line

(b) Distance between the centroid of polygon and the centroid of line

Figure 3.3: Distance relationship for real geometry (left) and for the centroid (right)

The second category (KOPERKI, 1995) (CLEMENTINI, 2000) (APPICE, 2003,2005) (MALERBA, 2001) (LISI, 2004) (MENNIS, 2005) (BOGORNY, 2006a, 2006b, 2006c, 2006e, 2007) is based on *qualitative reasoning*, which usually considers distance and topological relationships between a reference geographic object type and a set of relevant feature types represented by any geometric primitive (e.g. points, lines, and polygons). Relationships are normally extracted in a first step, in data *preprocessing* tasks, while frequent sets are generated in another step.

In both qualitative and quantitative reasoning approaches prior knowledge has rarely been used to eliminate irrelevant geographic domain patterns and to produce more interesting rules. (KOPERSKI, 1995) presented an approach which exploits taxonomies of both spatial feature types and spatial relationships only for mining spatial association rules at different granularity levels. Only minimum support is used to prune frequent sets and spatial association rules. A similar method has still been used by Mennis (2005). Clementini (2000) extended this method for mining multi-level spatial association rules from geographic objects with broad boundaries.

In (MALERBA, 2001) (APPICE, 2003) (LISI, 2004) both frequent sets and rules are pruned a posteriori. The user can define a pattern constraint and specify how many times a predicate should appear in the frequent sets or in association rules. For example, a pattern constraint such as *pattern_constraint [[intersects(x,road),5]]* removes from the frequent sets the specified predicate when it appears in less than 5 sets. This step is performed after all frequent sets have already been generated. A rule constraint such as *body_constraint [[intersects(x,road),contains(x,hospital)],10]* only shows rules with the specified predicates if they appear in at least 10 association rules, otherwise they are removed. In (APPICE, 2005) this method has been extended with the possibility to specify one more cardinality for a constraint. This method can be used to remove well known rules. For example, a constraint such as *pattern_constraint[[intersects(gas station),(intersects(road)],0,0))* would remove from the set of frequent sets all combinations having this pair of predicates. The minimum and maximum cardinality 0 defines that the pair of predicates should not appear in the resultant set of rules.

The pruning method proposed in (MALERBA, 2001) (APPICE, 2003, 2005) (LISI, 2004) has some general disadvantages that make the method hard to be used with real databases. First, in data preprocessing all spatial relationships must be computed from geographic databases and transformed to first-order logic. In large geographic databases the extraction of all relationships is non-trivial, and many relationships can be unnecessarily computed (KLOSGEN, 2002). Second, the pruning step is very hard for the data mining user, since for every different relationship or geographic element, a different pattern constraint must be specified to remove non-interesting rules. Moreover, as concluded by the authors about their proposed method a lot of knowledge is required from the data mining user. Third, it is hard for the data mining user to a priori know all possible frequent sets and rules that might have a non-interesting pattern or rule. At lower granularity levels, which will be explained latter in this chapter (Section 3.3.3), for example, such difficulty increases since a different constraint must be specified for every different relationship and feature at a different concept level. For example, to eliminate a dependence between *gas station* and *road*, some of the constraints that the user has to specify include:

pattern_constraint([contains(X,GasStation),crossed_by(X,Road)],0,0),
pattern_constraint([contains(X,GasStation),contains(X,Road)],0,0),
pattern_constraint([contains(X,GasStation),touches(X,Road)],0,0)),
pattern_constraint([contains(X,Large_GasStation),crossed_by(X,StateHighWay)],0,0),
pattern_constraint([contains(X,Large_GasStation),contains(X,NationalHighWay)],0,0),
pattern_constraint([contains(X,Large_GasStation),contains(X,NationalHighWayBR-116)],0,0)

In this method, patterns and rules that could be pruned earlier are cut off in post-processing steps. In the remaining of this chapter we explain the problems of mining spatial association rules with geographic dependences.

3.3 The General Problem of Mining Spatial Association Rules with Geographic Dependences

As have already been mentioned, at least three steps are required to extract patterns from GDB: the computation of spatial neighborhood relationships, the generation of frequent sets, and the generation of association rules. In the first step, the target feature type is compared with all relevant feature types to extract spatial predicates. In the second step, predicates are compared with each other to generate frequent predicate sets, from which rules are generated in a third step. Well known geographic dependences appear in all steps, and in different ways, producing different amounts of well known patterns. In the following sections we show how geographic dependences appear in these steps.

3.3.1 Geographic Dependences between the Target Feature Type and Relevant Feature Types

In data preprocessing, time and effort are required from the data mining user to extract spatial relationships and transform geographic data into a single table or a single file, which is the input format required by most SAR mining algorithms. Even in multi-relational data mining (KLOSGEN, 2002) (APPICE, 2003, 2005) (MALERBA, 2001) (LISI, 2004), where geographic data are converted to first order logic, the process of extracting spatial relationships is required.

Geographic dependences cause two main problems in data preprocessing: generate a large amount of non-interesting association rules and require unnecessary spatial joins. In Sections 3.3.1.1 and 3.3.1.2 these problems are explained in more detail.

3.3.1.1 Geographic Dependences and Non-Interesting Rules

Table 3.1 shows an example of a spatial dataset at a general granularity level, which will be used all over the following sections. Every row is a city and predicates refer to different geographic object types (port, water body, hospital, street, and factory) spatially related to city. Let us consider two geographic dependences: city and street, and port and water body, where the former is between the target feature type and a relevant feature type and the latter is among two relevant feature types.

Table 3.1: Example of a preprocessed dataset for mining frequent sets and SAR

Tuple (city)	Spatial Predicates
1	contains(Port), contains(Hospital), contains(Street), contains(Factory), crosses(Water Body)
2	contains(Hospital), contains(Street), crosses(Water Body)
3	contains(Port), contains(Street), contains(Factory), crosses(Water Body)
4	contains(Port), contains(Hospital), contains(Street), crosses(Water Body)
5	contains(Port), contains(Hospital), contains(Street), contains(Factory), crosses(Water Body)
6	contains(Hospital), contains(Street), contains(Factory)

In the dataset shown in Table 3.1, the dependence between the target feature type city and the relevant feature type street is expressed by the predicate *contains(Street)* which has a 100% support. Predicates with 100% support appear in half of the total number of frequent sets, as shown in Table 3.2 where *minsup* 50% was considered.

Table 3.2: Frequent sets with support 50%

Size *k*	Frequent sets with support 50%
1	{contains(Port)}, {contains(Hospital)}, **{contains(Street)},** {contains(Factory)}, {crosses(WaterBody)}
2	{contains(Port),contains(Hospital)}, {contains(Port),**contains(Street)**}, {contains(Port),contains(Factory)}, {contains(Port),crosses(WaterBody)}, {contains(Hospital), **contains(Street)**}, {contains(Hospital),contains(Factory)}, {contains(Hospital),crosses(WaterBody)}, {**contains(Street)**,contains(Factory)}, {**contains(Street)**,crosses(WaterBody)}, {contains(Factory),crosses(WaterBody)},
3	{contains(Port),contains(Hospital), **contains(Street)**}, {contains(Port),contains(Hospital),crosses(WaterBody)}, {contains(Port), **contains(Street)**,crosses(WaterBody)}, {contains(Port),contains(Factory),crosses(WaterBody)}, {contains(Port), **contains(Street)**,contains(Factory)}, {contains(Hospital), **contains(Street)**,contains(Factory)} {contains(Hospital), **contains(Street)**,crosses(WaterBody)}, {**contains(Street)**,contains(Factory),crosses(WaterBody)}
4	{contains(Port),contains(Hospital), **contains(Street)**,crosses(WaterBody)} {contains(Port), **contains(Street)**,contains(Factory),crosses(WaterBody)}

In this example, for *minsup* 50%, among the 25 frequent sets shown in Table 3.2, 13 have the dependence. From these frequent sets, a large number of non-interesting rules is generated. For example, a rule such as *contains(Factory)* → *contains(Street)* expresses that cities that contain factories do also contain streets. Although such a rule seems to be interesting, it can be considered obvious due the simple fact that all cities contain streets, having they factories, or not. The same type of rule will be generated for all other predicates which together with the predicate *contains(Street)* reach minimum support and pass the constraint *minconf*.

In order to evaluate the number of both frequent sets and association rules generated with a geographic dependence between the target feature type and one single relevant feature type, Table 3.3 summarizes the frequent set and rule generation from the dataset in Table 3.1 considering different *minsup* and *minconf* 70%. Considering low minimum support (20%), 31 frequent sets and 180 rules were generated, among which 16 frequent sets and 130 rules had the dependence *contains(Street)*. Notice that increasing minimum support to 50% does not warrant the elimination of the geographic dependence. Although the number of frequent sets is reduced to 25 and rules to 96, 13 frequent sets and 72 rules still had the dependence.

Table 3.3: Frequent sets and rules with dependences

MinSup %	Total FrequentSets/ Rules	Rules with Dependence / Rules without Dependence	FrequentSets with dependence / FrequentSets without dependence
20	31 / 180	130/ 50	16/15
50	25 / 96	72 / 24	13/12

Appendix A shows part of the association rules generated from a real dataset where the target feature type is vegetation (e.g. cropland, grassland, rice field) and the relevant feature types include river, bridge, tunnel, road, build up areas, trees, etc. Almost the whole region considered is surrounded by trees, which is a separate feature type, and is a predicate that has 100% support in the dataset. By consequence, as can be observed in the highlighted predicates in Appendix A, almost all generated rules contain this predicate.

Now let us consider the *closed frequent sets* extracted from Table 3.2. The number of sets is significantly reduced, as can be observed in Table 3.4. However, notice that the dependence is not eliminated by simply reducing the number of frequent sets. The geographic dependence with a 100% support appears in all closed frequent sets.

Table 3.4: Closed frequent sets

Size *k*	Closed frequent sets with support 50%
1	{**contains(Street)**}
2	{contains(Hospital), **contains(Street)**}, {**contains(Street)**,contains(Factory)}, {**contains(Street)**,crosses(WaterBody)},
3	{contains(Port), **contains(Street)**,crosses(WaterBody)}, {contains(Hospital), **contains(Street)**,contains(Factory)} {contains(Hospital), **contains(Street)**,crosses(WaterBody)},
4	{contains(Port),contains(Hospital), **contains(Street)**,crosses(WaterBody)} {contains(Port), **contains(Street)**,contains(Factory),crosses(WaterBody)}

Considering the experiments shown in Table 3.3 and the closed frequent sets in Table 3.4 we can say that neither minimum support nor the generation of closed frequent sets warrant the elimination of well known geographic dependences between the target feature type and the relevant feature types.

3.3.1.2 Geographic Dependences and Spatial Joins

Geographic dependences between the target feature type and the relevant feature types besides generating a large number of well known patterns and association rules, require unnecessary spatial joins. Considering the semantics of the target feature type and the relevant feature types during the spatial predicate computation, the number of spatial joins can be reduced, as shown with a few examples in Table 3.5 considering the topological relationships standardized by the OGC (OPEN GIS CONSORTIUM, 2001).

Table 3.5: Possible and mandatory topological relationships considering semantics of spatial feature types

Topological Relation / Semantic Combinations	Disjoint	Overlaps	Touches	Contains	Within	Crosses	Equals
Gas Station and Road			✓				
Bridge and Water Body						✓	
City Hall and City					✓		
Water Body and Road	✓		✓			✓	
Treated Water Net and City			✓		✓	✓	

Without considering semantics, *all* topological relationships between a target feature type and a relevant feature type will necessarily be tested in order to verify which one holds. By considering semantics, only a few relationships would be tested, as shown in Table 3.5. For example, between the feature types gas station and road, only the relationship *touches* is semantically possible. A city hall, for example, must be *within* a city, while a water body can have a relationship *disjoint, touches,* or *crosses* with road.

Although the topological relationships shown in Table 3.5 are semantically possible, not all of them are interesting for knowledge discovery. So, if besides considering the semantics of spatial features we also consider spatial integrity constraints, i.e., if they have a mandatory relationship, it is possible to reduce still further the number of topological relationships. Moreover, it is possible to define which topological relationships should be computed for knowledge discovery.

Remembering that *mandatory* relationships produce well known patterns and that only *possible* relationships are interesting for knowledge discovery, Table 3.6 shows the possible topological relationships to be computed for knowledge discovery for the same pairs shown in Table 3.5. The pairs gas station and road, bridge and water body, city hall and city, as well as treated water net and city have *mandatory* one-one or one-many constraints and no relationship is necessary for KDD. Only topological relationships between water body and road would be computed for these examples.

In this section we showed the high number of patterns and rules that can be generated by well known geographic dependences between the target feature type and a relevant feature type. Such dependences can be eliminated by pruning the input space,

as we propose in (BOGORNY, 2006a, 2007). Considering semantics, in (BOGORNY, 2007) we presented a geographic data preprocessing algorithm that uses geo-ontologies as prior knowledge to extract spatial predicates for mining spatial association rules (details will be presented in Chapter 4).

Table 3.6: Possible topological relationships for knowledge discovery

Topological Relation / Semantic Combinations	Disjoint	Overlaps	Touches	Contains	Within	Crosses	Equals
Gas Station and Road							
Bridge and Water Body							
City Hall and City							
Water Body and Road	✓		✓			✓	
Treated Water Net and City							

Geographic dependences, however, may also exist among relevant features. In the dataset shown in Table 3.1, there is another dependence, but among two relevant feature types (*port* and *water body*), where all cities which have *ports* do also have *water bodies,* because every *port* must be related to at least one *water body*. In this case, however, we cannot prune the input space because either *water body* or *port* may have an interesting association with any other relevant feature type (hospital, factory). In the following section we describe the problem of geographic dependences among relevant feature types.

3.3.2 Geographic Dependences among Relevant Feature Types

A dependence between the target feature type and any relevant feature type appears in the dataset as one single predicate. A dependence among relevant feature types appears when pairs of predicates generate a frequent set, as shown in Table 3.7 in bold style ({*contains*(*Port*),*crosses*(*WaterBody*)}), considering the same dataset presented in Table 3.1.

As can be observed in Table 3.7, geographic dependences between two relevant feature types appear the first time in frequent sets with 2 elements, where k=2. Notice that since the dependence has minimum support, i.e., is a frequent predicate set, this dependence is replicated to many frequent sets of size k>2. It appears with all other predicates which together with the pair that has a dependence reach minimum support. Considering this example and 50% minimum support, one single geographic dependence between two relevant feature types participates in 6 frequent sets (30% of the total number of frequent sets with size k≥2). Notice that the number of rules having a geographic dependence will be much larger than the number of frequent sets, mainly when the largest frequent set (with 4 elements) contains the dependence.

Pruning the frequent sets by generating closed frequent sets does not eliminate geographic dependences between two relevant feature types, since the pair ({*contains*(*Port*),*crosses*(*WaterBody*)}) participates in three closed frequent sets. The sets 18, 24, and 25 in Table 3.7 are closed frequent sets that contain the geographic dependence.

Table 3.7: Frequent sets with minimum support 50%

Set k	Frequent sets with support 50%	set
1	{contains(Port)},	1
	{contains(Hospital)},	2
	{contains(Street)},	3
	{contains(Factory)},	4
	{crosses(WaterBody)},	5
2	{contains(Port),contains(Hospital)},	6
	{contains(Port),contains(Street)},	7
	{contains(Port),contains(Factory)},	8
	{contains(Port),crosses(WaterBody)},	9
	{contains(Hospital),contains(Street)},	10
	{contains(Hospital),contains(Factory)},	11
	{contains(Hospital),crosses(WaterBody)},	12
	{contains(Street),contains(Factory)},	13
	{contains(Street),crosses(WaterBody)},	14
	{contains(Factory),crosses(WaterBody)},	15
3	{contains(Port),contains(Hospital),contains(Street)},	16
	{contains(Port),contains(Hospital),crosses(WaterBody)},	17
	{contains(Port),contains(Street),crosses(WaterBody)},	18
	{contains(Port),contains(Factory),crosses(WaterBody)},	19
	{contains(Port),contains(Street),contains(Factory)},	20
	{contains(Hospital),contains(Street),contains(Factory)}	21
	{contains(Hospital),contains(Street),crosses(WaterBody)},	22
	{contains(Street),contains(Factory),crosses(WaterBody)}	23
4	***{contains(Port),contains(Hospital),contains(Street),crosses(WaterBody)}***	**24**
	{contains(Port),contains(Street),contains(Factory),crosses(WaterBody)}	**25**

The same dependence replication process that occurs in the frequent set generation happens during the rule generation, as can be observed in Table 3.8. A few examples of association rules generated with Apriori over frequent predicate sets of size 2, 3, and 4 (the frequent sets 9, 17, and 24 of Table 3.7) are shown in Table 3.8. Rules 1 and 2 are generated from the set with 2 elements, and represent a single geographic dependence and its inverse. Rules 3, 4, 5, and 6 reproduce rules 1 and 2 with an additional element in the antecedent or the consequent. The same happens with frequent sets that contain 4 elements. Rules 7, 8, and 9 are the same rules 1 and 2 with two additional elements that combined with the dependence passed the *minconf* constraint (70%).

Table 3.8: Examples of association rules with frequent sets of size 2, 3, and 4 with a geographic dependence

Set	Rule	Possible Rules
k=2	1	***contains(Port) → crosses(Water Body)***
k=2	2	***crosses(Water Body) → contains(Port)***
k=3	3	contains(Hospital) ^ ***contains(Port) → crosses(Water Body)***
k=3	4	contains(Hospital) ^ ***crosses(Water Body) → contains(Port)***
k=3	5	contains(Hospital) → ***contains(Port) ^ crosses(Water Body)***
k=3	6	***contains(Port) ^ crosses(Water Body)*** → contains(Hospital)
k=4	7	contains (Street) ^ ***contains(Port) → crosses(Water Body)*** ^ contains (Hospital)
k=4	8	contains (Street) → ***contains(Port) ^ crosses(Water Body)*** ^ intersects (Hospital)
k=4	9	contains (Street) ^ intersects (Hospital) → ***contains(Port) ^ crosses (Water Body)***

Appendix B shows some of the resultant rules of an experiment with a real dataset having one pair with a dependence between bridge and river. In this experiment, notice that independently of the topological relationship, all rules that contain the relevant feature type bridge, also contain river. As can be observed, a large number of rules containing the dependence is generated. In this same experiment we observe rules having at least two predicates with the same feature type and different topological relationships. These problems will be discussed latter in the remaining of this chapter.

In this section we discussed the problem of geographic dependences considering data at a general granularity level. However, association rules can be extracted from data at different granularity levels. In the next section we explain the process of mining association rules at different concept levels and the dissemination of geographic dependences at different levels.

3.3.3 Geographic Dependences among Relevant Feature Types at Different Granularity Levels

According to the objective of the discovery associations rules can be extracted from data at a more specialized granularity level or at a higher concept level (HAN, 1995a). For example, having some regions in a metropolitan area high air pollution incidence, it might be interesting to consider spatial predicates of factories in a more generalized level such as *intersects(Factory)*. In some specific cases, it might be interesting to consider spatial predicates of the different types of factories such as *intersects(ChemicalFactory), intersects(MetalurgicalFactory)*. In very specific cases, it might be interesting to consider the instances of factories, such as *intersects(ChemicalFactoryX), intersects(MetallurgicalFactoryY).*

An association rule in a general concept level could be, for example, *intersects(Road)* → *intersects(Factory).* Lower level rules could be, for example, *intersects(Road)* → *intersects(ChemicalFactory)* or *intersects(Highway)* → *intersects(ChemicalFactoryX).*

The method of mining data at different granularity levels has been presented by (LU, 1993), which uses concept hierarchies for generalization of spatial and non-spatial data to the discovery of general knowledge. Han (1995b) presented an algorithm for extracting association rules at multiple granularity levels from classical databases, and Koperski (1995) extended the method for mining multiple-level SAR from geographic databases. Concept hierarchies for mining spatial association rules at different granularity levels have been largely used in many approaches such as (CLEMENTINI, 2000), (MALERBA, 2001), (APPICE, 2005) (MENNIS, 2005).

Concept hierarchies are used to facilitate the extraction of knowledge at different granularity levels (HAN, 1995a, 1995b). An example of a concept hierarchy of water resource is defined in Figure 3.4.

Concept hierarchies should be used to represent data at different granularity levels in refinement mining processes, i.e., to extract more general or more specific knowledge from data in different mining processes, as proposed by (LU, 1993) (KOPERSKI, 1995) (CLEMENTINI, 2000) and (MENNIS, 2005). The inclusion of the same data at different granularity levels (e.g. water, river, jacui_river, lake) in the same mining process as in (SRIKANT, 1995)(HAN, 1995b) will generate redundant and non-novel rules that simply associate one level of the hierarchy (e.g. water) with another (e.g. river) and with many attributes if they together reach minimum support. Rules such as

contains(lake) → *contains(water)* will be generated, and different methods (SRIKANT, 1995) (HAN, 1995b) were proposed to eliminate rules that are only generated because of the mixture of the same data at different granularities. In our point of view, different granularities of the same data should be used in refinement mining steps, while different data (e.g. water body, road) can be mined at different granularities in the same mining process.

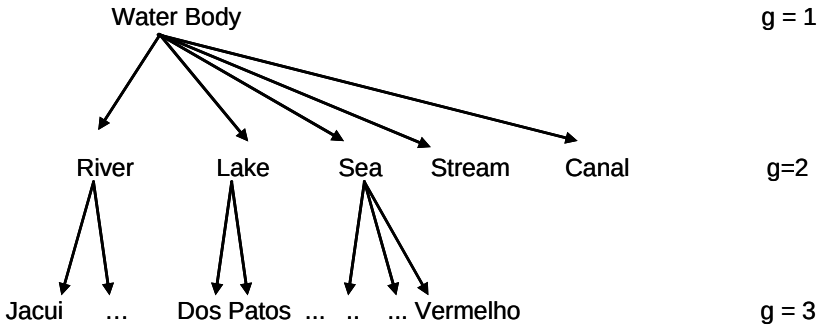


Figure 3.4: A concept hierarchy of water body

Concept hierarchies can be provided by knowledge engineers, domain experts or data mining users, or automatically generated from databases. In some cases concept hierarchies can be encoded in a database schema (CLEMENTINI, 2000) and can be dynamically generated for knowledge discovery (HAN, 1994).

Geographic dependences exist among geographic data independently of granularity level. Considering an *is_a* hierarchy and a geographic dependence at any level, this dependence is inherited by all sublevels, as in object oriented databases, for instance. For a concept hierarchy of *road*, for example, a dependence between *gas station* and *road* is inherited by all specializations of road. For example, an association rule at level 2, such as *intersects(GasStation)* → *intersects (National_Highway)* encodes the dependence between *gas station* and *road* defined at a granularity level 1. The same happens in a more specific level, in rules such as *intersects(GasStation)* → *intersects(National_Highway_BR-101).*

Let us consider another example, where *islands* must exist within a water body, which can be of type *river*, *lake*, *stream*, or *sea*. Considering the hierarchy shown in Figure 3.4, a dependence between island and water body should be specified at the highest granularity level, since an island must be related to a water body that can be of any type. Although the dependence belongs to a higher granularity level, it is applicable (inherited) to all lower levels. Mining data at a higher level, the dependence will generate rules such as *is_a(Island)* → *within(WaterBody)*. At a second level, rules such as *is_a(Island)*→ *within(River)* and/or *is_a(Island)*→ *within(Lake)* will be generated because of the dependence between island and water body.

At more specialized granularity levels the support of predicate sets is lower, and the value of minimum support must be lower as well, in order to avoid the elimination of interesting associations. Let us consider the dependence between *port* and *water body*, being water body at a second granularity level (e.g. river, stream, canal), as shown in Figure 3.5 (a). Considering minimum support 30%, 34 frequent sets will be generated (29 with size $k \geq 2$), as shown in Figure 3.5 (b). Among the 29 sets of size $k \geq 2$ from which it is possible to generate association rules, 10 contain the geographic dependence. This corresponds to 35% of the frequent sets of size $k \geq 2$.

(a) dataset

Tuple (city)	Spatial Predicates
1	contains(Port), contains(Hospital), contains(Street), contains(Factory), crosses(River)
2	contains(Hospital), contains(Street), crosses(Stream)
3	contains(Port), contains(Street), contains(Factory), crosses(Canal)
4	contains(Port), contains(Hospital), contains(Street), crossess(Canal)
5	contains(Port), contains(Hospital), contains(Street), contains(Factory), crosses(River)
6	contains(Hospital), contains(Street), contains(Factory)

(b) frequent predicate sets

k	Frequent sets with support 30%
1	{contains(Port)}, {contains(Hospital)},{contains(Street)},{contains(Factory)}, {crosses(River)}, {crosses(Canal)}
2	{contains(Port),contains(Hospital)}, {contains(Port),contains(Street)}, {contains(Port),contains(Factory)}, ***{contains(Port),crosses(River)},*** ***{contains(Port),crosses(Canal)},*** {contains(Hospital),contains(Street)}, {contains(Hospital),contains(Factory)}, {contains(Hospital),crosses(River)}, {contains(Street),contains(Factory)}, {contains(Street),crosses(River)}, {contains(Street),crosses(Canal)}, {contains(Factory),crosses(River)},
3	{contains(Port),contains(Hospital),contains(Street)}, {contains(Port),contains(Hospital),contains(Factory)}, ***{contains(Port),contains(Hospital),crosses(River)},*** {contains(Port),contains(Street),contains(Factory)}, ***{contains(Port),contains(Street),crosses(River)},*** ***{contains(Port),contains(Factory),crosses(River)},*** {***contains(Port),contains(Street),crosses(Canal)},*** {contains(Hospital),contains(Street),contains(Factory)} {contains(Hospital),contains(Street),crosses(River)}, {contains(Street),contains(Factory),crosses(River)}
4	{contains(Port),contains(Hospital),contains(Street),contains(Factory)}, ***{contains(Port),contains(Hospital),contains(Street),crosses(River)}*** ***{contains(Port),contains(Hospital),contains(Factory),crosses(River)},*** ***{contains(Port),contains(Street),contains(Factory),crosses(River)}*** {contains(Hospital),contains(Street),contains(Factory),crosses(River)}
5	***{contains(Port),contains(Hospital),contains(Street),contains(Factory),crosses(River)***}

Figure 3.5: (a) Dataset having water body at granularity level 2 and (b) frequent predicate sets with support 30%

In Figure 3.6 (a) water is at a third granularity level. Notice that at this level the support of predicates decreases still further. Even with high minimum support (30%) there are still dependences among the frequent sets, as shown in Figure 3.6 (b) in bold style. For minimum support 30%, 25 frequent sets of size $k \geq 2$ are generated, and 8 still have the dependence.

Although the dependence replication process varies according to the granularity level and minimum support, the examples show that minimum support does not warrant the elimination of well known geographic dependences in spatial association rule mining.

Appendix C shows some examples of association rules generated from a real dataset where the relevant feature type bus stop has a dependence with road, but the latter is represented at a second granularity level as street (rua) and avenue (avenida) in the dataset. As can be observed, all association rules having the predicate bus stop (parada de ônibus) do always have a predicate with either street (pink highlighted) or avenue(yellow highlighted), and sometimes with both. This occurs because bus stops must be related to one road, which may be of type street or avenue. The especialization of road in many feature types (street and avenue) increases the number of attributes in the dataset, and by consequence, the number of pairs having dependences (e.g. {*contains(BusStop), contains(Street)}, {contains(BusStop), contains(Avenue)*}).

(a) dataset

Tuple (city)	Spatial Predicates
1	contains(Port), contains(Hospital), contains(Street), contains(Factory), crosses(RiverX)
2	contains(Hospital), contains(Street), crosses(StreamX)
3	contains(Port), contains(Street), contains(Factory), crosses(CanalX)
4	contains(Port), contains(Hospital), contains(Street), crosses(CanalY)
5	contains(Port), contains(Hospital), contains(Street), contains(Factory), crosses(RiverX)
6	contains(Hospital), contains(Street), contains(Factory)

(b) frequent predicate sets

k	Frequent sets with support 30%
1	{contains(Port)}, {contains(Hospital)}, {contains(Street)}, {contains(Factory)}, {crosses(RiverX)}
2	{contains(Port),contains(Hospital)}, {contains(Port),contains(Street)}, {contains(Port),contains(Factory)}, ***{contains(Port),crosses(RiverX)},*** {contains(Hospital),contains(Street)}, {contains(Hospital),contains(Factory)}, {contains(Hospital),crosses(RiverX)}, {contains(Street),contains(Factory)}, {contains(Street),crosses(RiverX)}, {contains(Factory),crosses(RiverX)},
3	{contains(Port),contains(Hospital),contains(Street)}, {contains(Port),contains(Hospital),contains(Factory)}, ***{contains(Port),contains(Hospital),crosses(RiverX)},*** {contains(Port),contains(Street),contains(Factory)}, ***{contains(Port),contains(Street),crosses(RiverX)},*** ***{contains(Port),contains(Factory),crosses(RiverX)},*** {contains(Hospital),contains(Street),contains(Factory)} {contains(Hospital),contains(Street),crosses(RiverX)}, {contains(Street),contains(Factory),crosses(RiverX)}
4	{contains(Port),contains(Hospital),contains(Street),contains(Factory)}, ***{contains(Port),contains(Hospital),contains(Street),crosses(RiverX)}*** ***{contains(Port),contains(Hospital),contains(Factory),crosses(RiverX)},*** ***{contains(Port),contains(Street),contains(Factory),crosses(RiverX)}*** {contains(Hospital),contains(Street),contains(Factory),crosses(RiverX)}
5	***{contains(Port),contains(Hospital),contains(Street),contains(Factory),crosses(RiverX)}***

Figure 3.6: Dataset with water body at granularity level 3 and frequent predicate sets with support 30%

So far we have analyzed the large number of non-novel frequent patterns and spatial association rules generated because of *geographic dependences* among relevant feature

types. In frequent geographic pattern mining not only geographic dependences generate non-interesting patterns and rules. When mining data at higher granularity levels some spatial relationships may generate non-interesting patterns as well. At lower granularity levels, however, some spatial relationships may not be captured by the rule mining algorithm. These problems will be addressed in the following sections.

3.3.3.1 Non-Interesting Patterns Generated at Higher Granularity Levels

A large number of non-interesting frequent sets and spatial association rules is generated when mining data at higher granularity levels, i.e., when only the feature types are considered without their respective instances.

The first problem is the generation of sets of spatial predicates that have the same *feature type* with different topological relationships (e.g. *touches(WaterBody), contains(WaterBody)*). This normally occurs when considering distance or topological relationships, since the target feature type may have different spatial relationships with different instances of a relevant feature type. Examples of such rules are shown in Appendix B, highlighted in pink color, where road appears in both antecedent and consequent of the rules having a different topological relationship. When considering topological relationships a large amount of frequent sets is generated containing pairs of predicates with different topological relationships and the same feature type.

Going further into details of this problem let us consider a real example, observing the geographic map shown in Figure 3.7. Small polygons are slums, large polygons are districts, and black lines are water bodies of the city of Porto Alegre. Let us suppose that district is the target feature type and slums and water bodies are the relevant feature types.

In Figure 3.7 we can observe the different topological relationships that districts may have with both slums and water bodies. The district *Nonoai*, for example, "*contains"* slums (e.g. 159, 163, 187) "*touches"* slums (e.g. 180), and "*overlaps"* slums (e.g. 174). The district *Teresopolis*, for example, "*contains"* water bodies (e.g. 93, 338) and has a *crosses* relationship with water bodies (e.g. 339). Different topological relationships may generate redundant and non-interesting rules for the same geographic feature at a general granularity level, i.e., when the specific instance is not considered. For example, *contains(Slum) → touches(Slum), contains(Slum) → overlaps(Slum),* or *touches(Slum)→overlaps(Slum).* A rule such as *contains(Slum) → touches(Slum)* expresses that districts that *contain* slums do also *touch* slums or vice-versa. This kind of rule does not contribute to the discovery of novel and useful knowledge. In fact, this kind of rule does not represent a cause and an effect. An interesting association rule with these two predicates could be, for example, a rule such as *criminalityRate=high → contains(Slum)* or *criminalityRate=high → touches(Slum),* with an additional predicate. Pairs of predicates with the same feature type and different spatial relationships produce a large number of non-interesting rules.

Another problem that occurs in frequent geographic pattern mining is the generation of sets of spatial predicates that contain different feature types (e.g. Street, Avenue) that have the same parent (e.g. Road) in a concept hierarchy. This occurs when mining data at more specialized granularity levels. The predicates having "brothers" of a concept hierarchy may have either same spatial relationship (e.g ***contains(****Street),* ***contains(****Avenue)*) or not (e.g ***contains****(Street),* ***touches(****Avenue)*). A large amount of frequent sets with different combinations of "brothers" in a concept hierarchy are

generated when mining data at lower granularity levels. Some examples of association rules having this problem are shown in Appendix C, and can also be observed in Figure 3.7.

Let us suppose that the relevant feature type water body, shown in Figure 3.7, can be specialized at a lower granularity level and that the water bodies 93 and 338 are *streams* and that 339 and 94 are *rivers.* Without considering the respective instances of streams and rivers in the mining process, the combination of "brothers" in the concept hierarchy (Stream and River) will generate rules such as *contains(Stream)→ contains(River)* or *contains(Stream)→ crosses(River).* We argue that frequent sets and association rules that contain "brothers", i.e., different feature types that have the same parent in a concept hierarchy do not contribute to the discovery of novel and useful knowledge. In contradiction to a pattern expressed by the rule *contains(Stream) → contains(River),* districts do not contain rivers because they contain streams or vice-versa. They do have water bodies that are of type stream and river. Such rule has no *cause→effect* and is meaningless. Interesting association rules with these predicates could be, for example, *pollution=high→ contains(Stream)* or *pollution=high→contains(River),* but having another meaningful predicate.

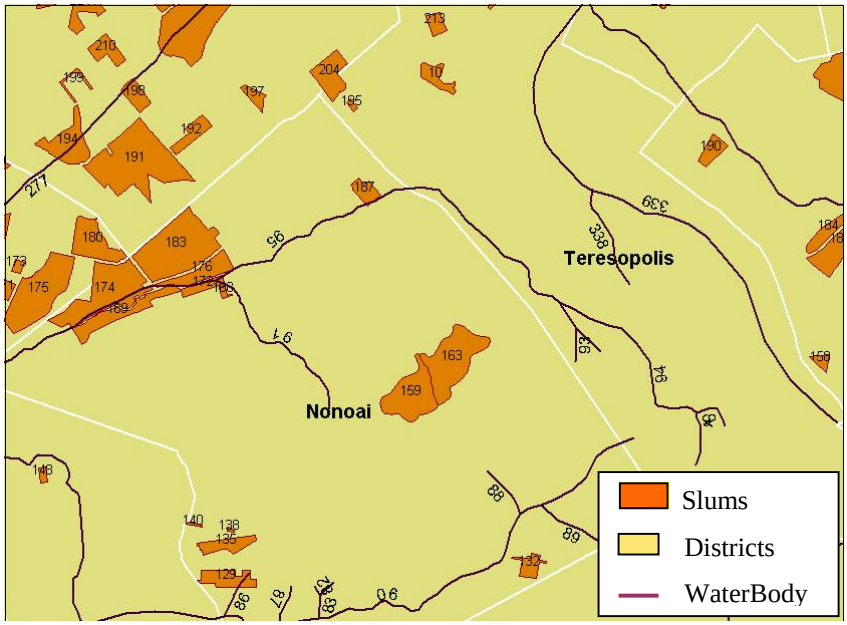


Figure 3.7: Part of a geographic map of the Porto Alegre city representing districts, slums, and water bodies

Figure 3.8 (a) shows an example of a dataset where different topological relationships are considered for the feature types slum and water body. The latter is represented at lower granularity level (river and stream). Considering minimum support 50%, 21 frequent sets are generated from this dataset, as shown in Figure 3.8 (b). Among the 21 frequent sets, 7 sets contain pairs of predicates with more than one relationship for the same feature type (slum). Indeed, among the 21 frequent sets, 4 contain the pairs of geographic feature types (stream and river) with the same parent in a concept hierarchy.

The generation of frequent sets with the same spatial feature type varies according to minimum support. However, the combination of pairs of the same feature type (slum) with different relationships (contains, touches, and overlaps) appears the first time

during the generation of frequent sets with 2 elements. The same is valid for an object (water body) represented at different granularities (river and stream), having different spatial relationships (crosses and contains), or not. The replication process is similar to the dependence replication explained in the previous sections.

(a) dataset

Tuple (district)	Spatial Predicates
Nonoai	contains(Slum), touches(slum), overlaps (slum), crosses(river), contains(stream)
Teresopolis	contains(Slum), , overlaps (slum), crosses (river), contains (stream)
Vila Nova	contains(Slum), touches(slum), overlaps (slum), crosses (river), contains (stream)
Cristal	contains(Slum), touches(slum), overlaps (slum), touches (river), crosses (stream)
Camaqua	contains(Slum),
Bela Vista	, crosses (stream)

(b) Frequent predicate sets

k	Frequent sets with support 50%
1	{contains(Slum)}, {touches(Slum)}, {overlaps(Slum)}, {crosses(River)}, {contains(stream)}
2	***{contains(Slum),touches(Slum)},*** ***{contains(Slum),overlaps(Slum)},*** {contains(Slum),crosses(River)}, {contains(Slum),contains(Stream)}**,** ***{touches(Slum),overlaps(Slum)},*** {touches(Slum),crosses(River)}, {touches(Slum),contains(Stream)}, {overlaps(Slum),crosses(River)}, {overlaps(Slum),contains(Stream)}, ***{crosses(River),contains(Stream)},***
3	***{contains(Slum),touches(Slum),overlaps(Slum)},*** ***{contains(Slum),overlaps(Slum), crosses(River)},*** ***{contains(Slum),overlaps(Slum), contains(Stream)},*** ***{contains(Slum),crosses(River), contains(Stream)},*** ***{overlaps(Slum),crosses(River), contains(Stream)},***
4	***{contains(Slum),overlaps(Slum),crosses(River), contains(Stream)}***

Figure 3.8: Dataset and frequent predicate sets with 50% minimum support

Association rules having spatial feature types with the same parent in a concept hierarchy are generated in any pattern mining process where the instances of the relevant feature types are not considered. As data can be represented at different granularity levels, being 1 the more general level (e.g. water body) and *n* the more specialized (e.g. river_Jacui), we can say that this problem may occur at all granularity levels $g < n$. Some more examples of these problems can be observed in the result of two different experiments, with two different databases, shown in Appendix B (pink highlighted) and Appendix C (blue highlighted).

In this section we addressed problems generated when mining data at granularities $g < n$, where many non-interesting patterns are generated. In the following section, we address a problem when mining data at granularity *n*.

3.3.3.2 Missing Patterns at Lower Granularity Levels

While in the previous section the problem was the generation of non-interesting rules extracted from data at higher granularity levels, in this section we address a problem that occurs when mining data at the lowest granularity level, i.e., when the instances of the feature types are taken in account.

This problem will be analyzed considering the same dataset shown in Figure 3.8(a), but at the lowest granularity level (*n*), which we also call the *feature instance* granularity level. At this level, besides the feature type, all instances participate in the mining process. Depending on minimum support, frequent pattern mining algorithms may not catch some patterns when different spatial relationships are considered for the same feature instance.

Table 3.9 represents the dataset shown in Figure 3.8(a) considering the instances of relevant feature types slum and water body. Notice that the instances of relevant feature types are not duplicated in the same row, i.e., every district (row) has only one topological relationship with every instance of the relevant feature types slum and water body. Observe in Table 3.9 that considering all different topological relationships (e.g. *contains, within, crosses, overlaps)* in some cases the pattern mining algorithm will not match predicates that should be grouped in a pair to generate a frequent set. Let us observe slum 180 in the map shown in Figure 3.7. It touches the district *Nonoai* because both slum and district only intersect boundaries. The *slum 180*, however is also *within* or *covered by* the district *Cristal*. In this case, the predicates *touches(Slum_180)* and *contains(Slum_180)* in Table 3.9 will be considered as different predicates. In these cases it might be interesting to consider only the general topological relationship (intersects and non-intersects) such that relationships with the same instance can be considered as one single predicate, as for example *intersects(Slum_180)*. Otherwise, at the feature instance granularity level, predicates such as *touches(Slum_180)* and *contains(Slum_180)* may be eliminated by minimum support.

Table 3.9: Dataset at the lowest granularity level – feature instance

Tuple (district)	Spatial Predicates
Nonoai	contains(Slum_169), contains(Slum_183), contains(Slum_176), contains(Slum_172), contains(Slum_188), contains(Slum_187), contains(Slum_159) contains(Slum_163) contains(Slum_32), contains(Slum_129) contains(Slum_135) contains(Slum_138) contains(Slum_140), ***touches(slum_180),*** **overlaps (slum_174),** crosses(river_95), contains(stream_91), contains(stream_86) , contains(stream_87) , contains(stream_82) , contains(stream_83) , **crosses(stream_90)** , crosses(stream_89) , **crosses(stream_95)**
Teresopolis	overlaps(Slum_204), contains(Slum_195), contains(Slum_10), contains(Slum_213), crosses(river_339), contains(Stream_338), contains(Stream_93) contains(Stream_94) contains(Stream_98), **crosses(stream_90)**
Vila Nova	…, crosses (stream_446), …
Cristal	***contains(Slum_180),*** **overlaps (slum_174),** touches(river_Guaiba), crosses(stream_277), **crosses(stream_95),** crosses(stream_446)
Camaqua	contains(Slum_124), ...
Bela Vista	, …

For example, suppose that the dataset in Figure 3.9 has a non-spatial attribute criminality rate, which is high for districts Nonoai and Cristal. If this rate is high because both districts intersect slum 180, such pattern will only be extracted if a general topological relationship is considered. Otherwise, no pattern will be generated.

In this chapter we presented in detail many different problems in frequent geographic pattern mining. These problems have not been addressed in previous works as far as we know. Among the different problems, the main question is the large amount of frequent sets and association rules generated by geographic dependences that are previously known as non-interesting. To overcome these problems, in the following chapter we propose a general framework to enhance the process of mining SAR in geographic databases. Therefore, the main contribution is the use of prior knowledge.

4 A GENERAL FRAMEWORK FOR MINING SPATIAL ASSOCIATION RULES WITHOUT WELL KNOWN DEPENDENCES

In this chapter we present a unified framework for mining spatial association rules from geographic databases. We address the problems introduced in the previous chapter presenting a first contribution for geographic data preprocessing, which in this context refers to all data preparation steps covering selection, spatial join processing, and transformation of geographic databases for SAR mining.

For SAR mining we present two algorithms, Apriori-KC and Max-FGP, specifically developed for mining SAR from geographic databases. These algorithms were developed with the objective of eliminating well known dependences and redundant frequent sets.

Figure 4.1 shows the proposed framework, named GeoARM (Geographic Association Rule Miner) that supports the complete discovery process for mining SAR from geographic databases. To better understand the process, the framework can be analyzed in three general levels: data repository, data preprocessing, and frequent pattern mining.

At the bottom are the geographic databases, stored in GDBMS constructed under OGC specifications (OPEN GIS CONSORTIUM, 1999b). There is also a knowledge base, which stores all information that may be used as prior knowledge to improve geographic data preprocessing and SAR mining. This repository stores the set of pairs of geographic objects with dependences, geographic database schemas, geo-ontologies, and concept hierarchies.

In the center of Figure 4.1 is our first contribution (BOGORNY, 2005a, 2005b, 2006a). It is the spatial data preprocessing level which covers the *gap* between data mining tools and geographic databases. At this level data repositories are accessed through JDBC/ODBC connections and data are retrieved, preprocessed, and transformed into the single table format. Dependences between the target feature type and relevant feature types are also removed in this step. This step prunes the input space and reduces the number of spatial joins, as explained in Section 5.1. According to (BONCHI, 2003) pruning the input space is still the most efficient way to reduce frequent sets.

On the top are the algorithms for mining spatial association rules. At this level, we present contributions for the frequent set generation step in the process of mining SAR.

We show two methods to eliminate all well known geographic dependences among relevant features types, i.e., the dependences which cannot be removed from the input dataset. The first method is an Apriori-like approach (BOGORNY, 2006b, 2006c) that eliminates the exact geographic dependences specified in a set of dependences. This step is explained in more detail in section 4.2.

The second method generates *maximal* frequent sets (BOGORNY, 2006e). This method applies the idea of closed frequent sets, but adapted to consider the elimination of geographic dependences. This step is detailed in Section 4.3.

Although we present different methods to generate frequent sets for mining spatial association rules, the way as dependences are eliminated can be applied to any frequent pattern or spatial association rule mining algorithm. The main strength of our framework is its simplicity, and very little background knowledge is required from the data mining user.

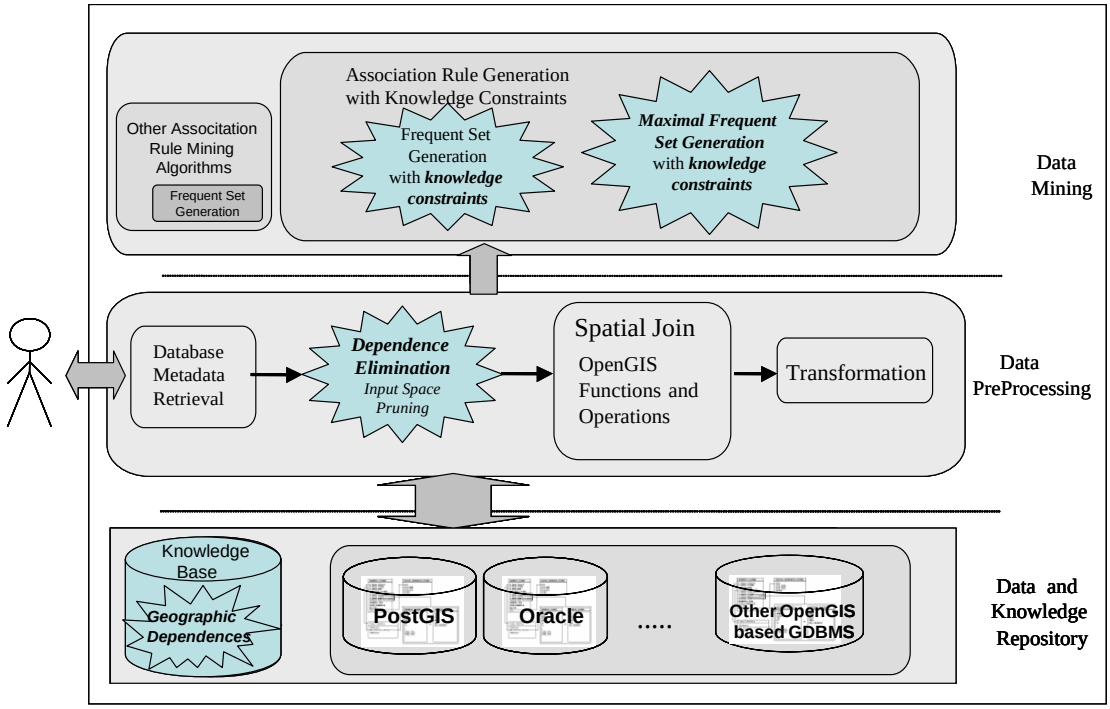


Figure 4.1: GeoARM: a Unified Framework for mining SAR from geographic databases

4.1 Data PreProcessing

The data preprocessing steps using prior knowledge can be performed in different ways, according to the knowledge represented in the knowledge base on the bottom of the framework presented in Figure 4.1. In this thesis we consider that in the knowledge repository there is a set of pairs of geographic feature types with dependences, that we call hereafter *knowledge constraints*. The set ϕ of knowledge constraints can be generated in different ways. When the knowledge base contains database schemas then the set ϕ can be generated with processes of reverse engineering (CHIFOSKY, 1990). Different approaches to extract dependences from relational databases with reverse engineering are available in the literature. For knowledge discovery in non-geographic databases reverse engineering has been used to understand the data model

(MCKEARNEY, 1996) in legacy systems, or to automatically extract SQL queries (SHOVAL, 1993), but not as prior knowledge to reduce well known patterns.

In (BOGORNY, 2006b) we present an algorithm to extract geographic dependences from database schemas, which is shown in Figure 4.2. If the database is relational, then the algorithm searches for all foreign keys. For each foreign key, the name of the table which it references is retrieved, as well as the name of the table where the foreign key is specified. The name of both relations is stored in a set of knowledge constraints ϕ. If the database is object-oriented, then the same steps are performed, but searching for classes with attributes which refer to other classes.

```
Given: a relational database schema
Find all foreign_keys
    For each foreign_key
        Insert into ϕ the name of the table
        which the foreign_key references and the name of
        the table to which the foreign_key belongs;
return ϕ;

Given: an OO database schema
Find all classes
    For each class in the database schema
        If there are references to other classes
                Insert the class name and the
                referenced class into ϕ
return ϕ;
```

Figure 4.2: Pseudo-code of the algorithm to extract geographic dependences from geographic database schemas

When the knowledge base contains an ontology, then the set of pairs with dependences ϕ can be generated by an algorithm that simply extracts the properties of the spatial feature types in the ontology. Otherwise, data preprocessing can be performed considering the whole ontology, as in the example shown in Appendix AV (BOGORNY, 2007).

The set of knowledge constraints ϕ can also be provided by the user. In this case a larger set of dependences can be specified; not only associations explicitly represented in either database schemas or ontologies, but other application domain dependences which generate well known patterns. In (BOGORNY, 2006d) we developed a graphical GUI to automate geographic data preprocessing where the user can specify pairs with dependences. Details will be presented in Chapter 6.

In Sections 4.1.1 and 4.1.2 we describe the data preprocessing tasks considering that ϕ is the set the geographic feature types with well known dependences. The focus is on using this set to improve geographic data preprocessing.

4.1.1 Data Preprocessing Tasks: The Input Space Pruning Method

There are four main steps to perform the tasks of extracting spatial predicates for mining SAR: *database metadata retrieval, dependence elimination (input space pruning), spatial join,* and *transformation,* as shown in Figure 4.1.

The *database metadata retrieval* step connects to the specified database and retrieves relevant information through the Open GIS database schema metadata,

including all database relations (feature types). The user chooses among the database relations the target feature type *T*, the target feature non-spatial attributes *A* and the set of relevant feature types *S* that may have some influence on *T*, the spatial relationships *R*, and the granularity level *g* for every different relevant feature type *O* in *S*, when a concept hierarchy H_O is given for *O*.

Figure 4.3 presents an overview of the data preprocessing algorithm, where *D* is the geographic database, ϕ is the set of pairs of geographic feature types with dependences, *T* is the target feature type, *A* is the set of non-spatial attributes of *T*, *S* is the set of relevant feature types *O*, and *R* is the spatial relationship (e.g. topology, distance). The objective is to find the dataset Ψ, without dependences between *T* and *S*, in the appropriate input format for mining frequent sets and spatial association rules.

```
Given: D, //geographic database
       ϕ, //set of dependences
       T, //target feature type
       A, //target feature type non-spatial attributes
       S, //set of relevant feature types
       R, //spatial relationships
       G, //set of granularity levels
       H; //set of concept hierarchies
Find: a dataset Ψ without geographic dependences between T and S;

Method:
     Ψ = select A from T;
     Dependence_Elimination
     begin
        For (i=1; i=#O in S, i++) do
        begin
           If (T has a dependence with Oi in ϕ)
               Remove Oi from S; //input pruning
           Else
               Ψ = Ψ ∪ spatial_join (R,(T),(Oi));
        end;
     end;
     Transformation (Ψ,G,H);
```

Figure 4.3: Pseudo-code of data preprocessing function to compute spatial predicates named *spatial_predicate_extraction*

The *Dependence Elimination* step verifies all associations between the target feature type and all relevant feature types. It searches the set of knowledge constraints ϕ and if *T* has a dependence with any $O \subseteq S$, then *O* is eliminated from *S*. For each relevant feature type removed from *S*, no spatial join is required to extract spatial relationships. By consequence, neither frequent sets nor spatial association rules will be generated with this relevant feature type.

The *Spatial Join* computes and materializes the relationships *R* between *T* and *O*, for all relevant feature types $O \subseteq S$. It extracts from a given geographic database *D* the spatial relationships *R* between the instances of the reference object type *T* and all instances of the relevant feature types $O \subseteq S$ in the granularity level $g_O \subseteq G$, according to a concept hierarchy $h_O \subseteq H$ and for every *O* in *S*. Different relevant feature types may be represented at different granularity levels.

If no concept hierarchies are provided, two different granularity levels can be automatically generated: *feature instance* and *feature type*. The feature type granularity level considers the database relations exactly as they are defined in the geographic database. For example, if there is a relation named "WaterBody", then predicates can be represented at this level, such as *contains(WaterBody)*, *touches(WaterBody)*. If there is a database relation for different types of water bodies, such as "River", "Lake", "Stream", than predicates are represented at lower levels (e.g. *contains(River)*, *contains(Lake)*), as the relations are represented in the database.

At the feature instance granularity level the same criterion is applicable, but with the difference that all instances of every relevant feature type are taken in account. For example, *contains(WaterBody_**Jacui**)*, *touches(WaterBody_**Guaiba**)*, or *contains(River_**Jacui**)*, *touches(Lake_**Guaiba**)*.

A granularity g is an integer number with size 1..n where 1 is the most general level and n is the more specialized. If no concept hierarchy is provided for O, g=1 is the feature type granularity level and g=2 is the feature instance granularity level that can be automatically generated for any database.

Four types of spatial relationships can be materialized by the spatial join step:

a) *Topological*: computes the detailed topological relationships (e.g. touches, contains);
b) *Intersection*: extracts more general topological relationships, intersects and non-intersects;
c) *Order*: extracts orientation spatial relationships;
d) *Distance*: computes neighborhood relationships given some distance parameters. Because close objects are more co-related than far objects, *close* is considered dominant over *far* at granularity levels $g<n$ where n is the lowest level. For example, if an instance of T is close to some instances of O and far from others, then *close* is materialized.

Spatial joins to extract spatial predicates are performed on-the-fly with operations provided by the GDBMS, and only over the relevant feature types defined by the user. Data preprocessing in the proposed framework follows the Open GIS Consortium specifications (OGC, 1999b), what makes *GeoARM* interoperable with all GDBMS constructed under OGC specifications (e.g. Oracle, PostGIS, MySQL, etc).

The *Transformation* step transposes as well as discretisizes the dataset Ψ into the single table representation understandable by association rule mining algorithms.

4.1.2 Understanding the Input Space Pruning Method

To better understand the input space pruning method and the effect that the elimination of one single geographic dependence between the target feature type and a relevant feature type has in the frequent set generation, let us consider the example shown in Figure 4.4.

Figure 4.4(a) shows the dataset with six tuples and five predicates shown in Table 3.1. For simplicity we use literals for the predicates. Every row in the dataset is a city and the predicate sets are relevant feature types (port, hospital, street, factory, water body) with spatial relationships with the target feature type city. The predicates are described in Figure 4.4 (c). In Figure 4.4(b) are the k frequent sets with minimum

support 50%, i.e., which appear in at least 50% of the tuples in the dataset in Figure 4.4(a).

a) dataset

Tid (city)	Predicate Set
1	A, C, D,T, W
2	C, D, W
3	A, D, T, W
4	A, C, D, W
5	A, C, D, T, W
6	C, D, T

b) frequent predicate sets with minsup 50%

Set *k*	Frequent sets
k=1	{A}, {C}, {D}, {T}, {W}
k=2	{A,C}, {A,D}, {A,T}, {A,W}, {C,D}, {C,T}, {C,W}, {D,T}, {D,W}, {T,W}
k=3	{A,C,D}, {A,C,W}, {A,D,T}, {A,D,W}, {A,T,W}, {C,D,T}, {C,D,W}, {D,T,W}
k=4	{A,C,D,W}, {A,D,T,W}

c) predicates

A = contains(Port), C = contains(Hospital), W = crosses(WaterBody),
T = contains(Factory), D = contains(Street),

Figure 4.4: Dataset with 6 tuples and frequent predicate sets with minimum support 50%

Notice in Figure 4.4(a) that the predicate *D* has a 100% support and represents a dependence between city and the relevant feature type street. Applying our method proposed in this section, the predicate *D* is removed from the dataset. By consequence, no frequent set containing *D* will be generated.

Figure 4.5 (left) shows the meet-semilattice of the frequent itemsets that would be generated by any association rule mining algorithm which does not eliminate geographic dependences. This algorithms would generate 4-level frequent sets (e.g.{A,C,D,W}). Using our input space pruning method, i.e., removing *D* from the dataset, avoids the generation of 13 frequent sets, scratched out in Figure 4.5 (left). Instead of 25 frequent sets, only 12 would be generated using our method, as shown in Figure 4.5 (right). Indeed, the input space pruning method eliminates the largest frequent sets which are generated because of the high support of the dependence.

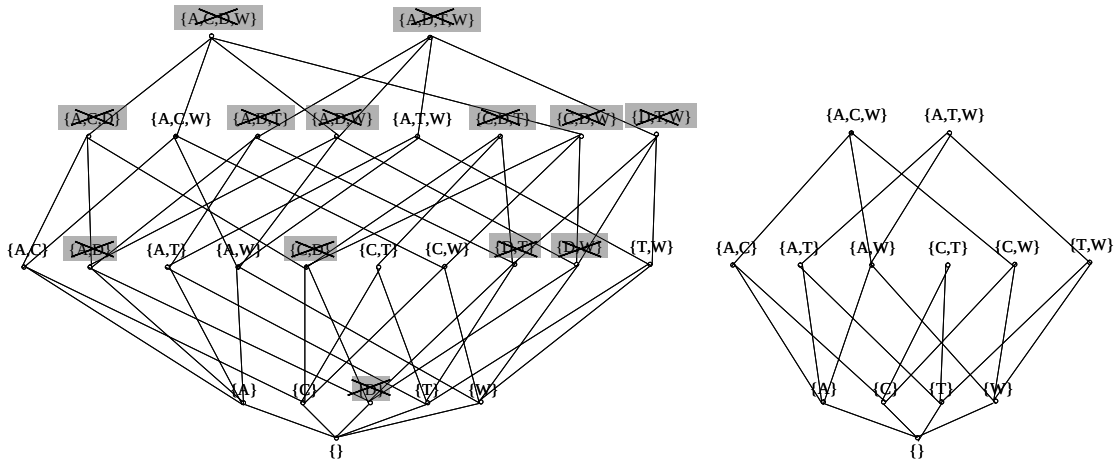


Figure 4.5: (left) Meet-semilattice of frequent predicate sets with dependence {D} and (right) meet-semilattice of frequent predicate sets without dependence {D}

Our method reduces one level in the frequent sets for each dependence that is eliminated. This avoids the generation of the largest sets, which normally generate the largest number of rules.

4.2 Frequent Set Generation with Knowledge Constraints

The most efficient way to eliminate well known geographic dependences in SAR mining that cannot be eliminated in data preprocessing is to eliminate candidate sets in which the dependences appear the first time. For this purpose we propose the Apriori-KC algorithm to generate frequent geographic patterns with knowledge constraints (BOGORNY, 2006b, 2006c). Knowledge constraints are the pairs of relevant feature types with semantic dependences that are a priori known as non-interesting. Considering this constraints as prior knowledge, we propose Apriori-KC to generate frequent patterns for geographic data, while Apriori is more appropriate for generating frequent patterns for non-spatial data.

4.2.1 Apriori-KC

Given a knowledge base that contains a set of pairs of geographic objects with dependences ϕ called *knowledge constraints*, and a set of concept hierarchies *H*, a dataset Ψ generated by the function *spatial_predicate_extraction* presented in the previous section, and a *minsup* threshold, multiple passes are performed over the dataset Ψ to generate frequent predicate sets, as shown in Figure 4.6.

```
Given: ϕ, // set of knowledge constraints
       Ψ, // dataset generated with spatial_predicate_extraction
       minsup, // minimum support
       H; //concept hierarchy
L1 = {large 1-predicate sets};
For ( k = 2; Lk-1 != ∅; k++ ) do  begin
    Ck = apriori_gen(Lk-1); // Generates  new  candidates
    If (k=2)
                // remove pairs with dependences
       (step 1) Delete from C2 all pairs with a dependence in ϕ ;
                 // remove pairs with hierarchical dependences
       (step 2) Delete from C2 all pairs with a hierarchical dependence H in ϕ ;
                 // remove pairs with same feature types and different
                 // topological relationships
       (step 3) Delete from C2 all pairs with the same feature type ;
                 // remove pairs with different feature types that have
                 // the same parent in H
       (step 4) Delete from C2 all pairs with the same parent in H

    Forall  rows w ∈ Ψ  do begin
        Cw = subset(Ck, w); // Candidates contained in w
        Forall candidates c ∈ Cw do
            c.count++;
    End;
    Lk = {c ∈ Ck | c.count ≥ minsup};
End;
Answer = ∪kLk
```

Figure 4.6: Apriori-KC to generate frequent geographic patterns without well known dependences

In the first pass, the support of the individual elements is computed to determine large-predicate sets. In the subsequent passes, given *k* as the number of the current pass, the large sets L_{k-1} in the previous pass (*k* -1) are grouped into sets C_k with *k* elements, which are called *candidate sets*. This is performed by the *apriori_gen* function, described in (AGRAWAL, 1994).

The support of each candidate set is computed, and if it is equal or higher than the minimum support, then this set is considered *frequent*. This process continues until the large set in the pass results in an empty set.

To eliminate well known geographic dependences we added two more steps (steps 1 and 2 in Figure 4.6) to Apriori, which are performed only once, when *k*=2, such that all pairs of elements defined in the set of knowledge constraints ϕ are removed from C_2. All dependences are removed from candidate sets with two elements, when they appear the first time, before computing their frequency. According to Assertion1, this step *warrants* that pairs of geographic objects with a dependence in ϕ will neither appear together in the frequent sets nor in the spatial association rules. This makes our approach effective and independent of any threshold such as minimum support, minimum confidence, lift, etc.

It is important to emphasize that no information is lost with our method to eliminate pairs of predicates. Only well known patterns and pairs that generate non-interesting association rules will be eliminated. For instance, suppose that {A,B} is a frequent set having a dependence. This pair is eliminated with the purpose to avoid the generation of larger frequent sets that contain the dependence, such as {A,B,C}, for example. If the set {A,B,C} has minimum support, then the pairs {A,B}, {A,C}, and {B,C} reached minimum support too. As we eliminate only pairs with dependences, {A,C} and {B,C} which combine the predicate *C* with both *A* and *B* separately, are still generated, and no information is lost. The pruning method only eliminates patterns that are well known, and does not sacrifice the result quality.

Going into more detail of the proposed algorithm for mining frequent geographic patterns, the step 1 in the algorithm eliminates the candidates that contain geographic objects with a dependence in ϕ. For example {*contains(Island), contains(Water)*}. This will avoid the generation of rules such as *contains(Island)*→*contains(Water).* Rules such as *pollution=high* → *contains(Island)* and *pollution=high*→*contains(Water)* are NOT eliminated by the proposed method.

The second step 2 removes hierarchical dependences, i.e, dependences that are inherited when data are represented in the dataset at a granularity lower than that in which the dependence is specified. For example, suppose that a dependence between island and water is defined in ϕ, but water is represented in the dataset Ψ in a more specialized level such as river, lake, and stream. In this case, all pairs in C_2 that combine island with any type of water are eliminated. This steps avoids the generation of rules such as *contains(Island)*→*contains(Lake)* or *contains(Island)*→*contains(River).*

One can argue that rules extracted from data at lower granularity levels having a geographic dependence might be interesting, as for example, *contains(Island)* → *contains(River) and pollution=high,* and should not be eliminated. Although this rule seems to be interesting, high pollution is simply an attribute which does not aggregate any information to the geographic dependence. Moreover, if high pollution is in fact related to either island or river, this information will still be represented by the pairs

{*contains(Island), pollution=high*} and {*contains(River), pollution=high*}, which are not eliminated by our method.

Besides removing well known geographic dependences, Apriori-KC also eliminates combinations of pairs of predicates that generate non-interesting rules having the same geographic feature type with either same or different spatial relationship. These pairs are eliminated by step 3 in the algorithm shown in Figure 4.6.

In step 3 pairs with the same feature type and different topological relationships (e.g. *contains(Water), touches(Water)*) are eliminated. This elimination occurs even if no concept hierarchy is specified, and avoids the generation of rules such as *contains(Water)* → *touches(Water).*

Step 4 removes pairs of spatial predicates that contain different feature types that have the same parent in concept hierarchy (e.g. lake, river, stream). This step eliminates pairs of predicates that contain "brothers" in a concept hierarchy, and they may have either same spatial relationship (e.g. *touches(River), touches(Lake)*) or not (e.g. *touches(River), contains(Lake)*). This step avoids the generation of a large number of non-interesting rules such as *contains(River)* → *contains(Lake).* As we have explained in Chapter 3 a rule such as *contains(River)* → *contains(Lake)* is not a rule *cause*→*effect.* Suppose that the target feature type is city that has a contains relationship with both river and lake. It makes no sense to say that cities contain lake because they contain river.

Our solution proposed in this section warrants the elimination of all well known geographic dependences among relevant features and pairs that generate non-interesting rules. This solution can be implemented in any Apriori-like algorithm that prunes patterns and rules with different types of constraints.

4.2.2 Understanding the Apriori-KC Pruning Method

To evaluate the dependence elimination process let us consider the dataset shown in Figure 4.4(a) and the frequent sets with support 50% in Figure 4.4 (b). Now let us consider the dependence between *A* and *W*, where all rows in the dataset where *A* occurs, *W* occurs as well. By eliminating the pair with the dependence {A,W} in the second pass, when it occurs the first time, the sets {A,W}, {A,C,W}, {A,D,W}, {A,T,W}, {A,C,D,W}, and {A,D,T,W} will not be generated, as shows the meet-semilattice in Figure 4.7 (left). The elimination of the set with 2 predicates avoids the generation of larger sets containing the pairs with dependences.

In this example, instead of 25 frequent sets with dependences, only 19 frequent sets and without well known dependences will be generated (Figure 4.7 right), and no information is lost. Only frequent sets that contain both A and W together are not generated, while sets that contain either A or W are still created. For example, the set {A,C,D,W} is not generated, but all other sets containing either *A* or *W* remain among the resultant frequent sets ({A,C}, {A,D}, {A,T}, {C,W}, {D,W},{T,W}, {A,C,D},{A,D,T}, {C,D,W}, and {D,T,W}).

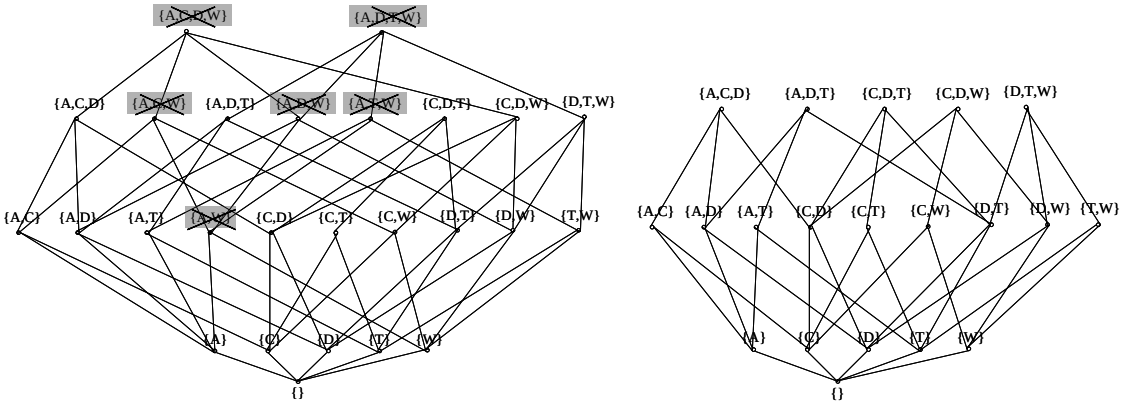


Figure 4.7: (left) Meet-semilattice of frequent itemsets with the dependence {A,W} and (right) meet-semilattice without dependence {A,W}

The elimination process explained in this example with the pair {A,W} is applicable to any of the 4 elimination steps presented in the algorithm. The elimination of dependences between relevant feature types is not as effective as the elimination of dependences between the target feature type and one relevant feature type as shown in the previous section. However, applying both pruning methods (input space and frequent sets with size 2), as proposed in (BOGORNY, 2006c) one dependence between the target feature type and a relevant feature type {D} and among two relevant feature types {A,W} can reduce the frequent sets much further, as shows the meet-semilattice in Figure 4.8.

While the number of frequent sets generated by Apriori in this example would be 25, and containing geographic dependences (Figure 4.8 left), our method would generate only 9, and without dependences, as shown in Figure 4.8 (right).

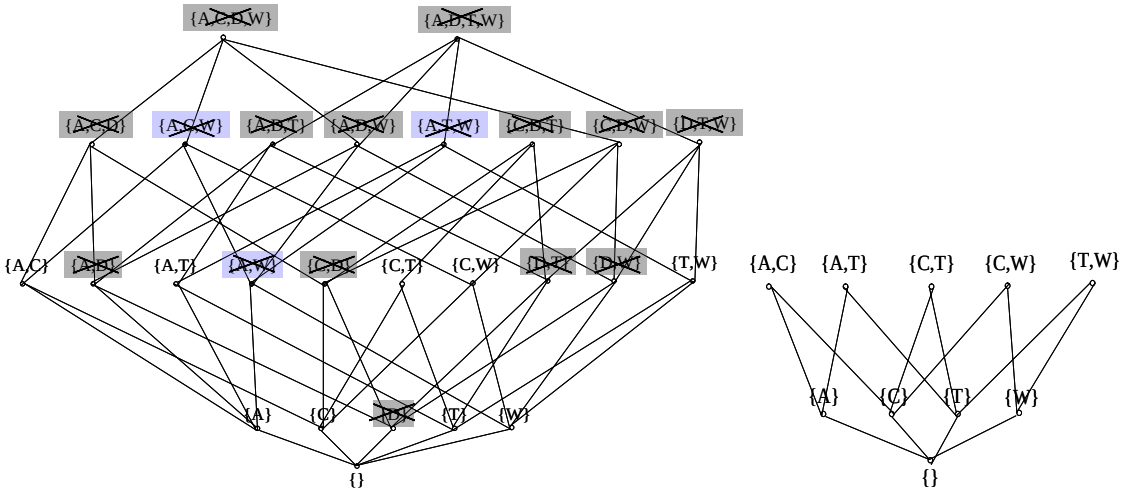


Figure 4.8: (left) Meet-semilattice of frequent sets with dependences {D} and {A,W}, and (right) meet-semilattice without dependences {D} and {A,W}

4.3 Maximal Non-Redundant Frequent Sets with Knowledge Constraints

Although closed frequent sets as far as we know have not been used in SAR mining, it is a very useful technique to reduce the number of both frequent sets and association rules.

Considering that the closed frequent set approach generates much less frequent sets because it eliminates frequent sets which generate redundant association rules, one can argue that geographic dependences should be eliminated from closed frequent sets, and not from frequent sets. In this section we evaluate the dependence elimination considering the closed frequent set approach and propose a new algorithm named Max-FGP (Maximal Frequent Geographic Patterns) (BOGORNY, 2006e) that will be explained along with this section.

4.3.1 Geographic Dependences and Closed Frequent Sets

To evaluate geographic dependences in closed frequent sets, let us consider the closed frequent itemsets shown in Figure 4.9, represented in bold style and organized according to the respective set of transactions in which they generate a closed frequent set. To follow the terminology commonly used in the frequent pattern mining literature, from this point we may also refer to a row or a tuple in the dataset as a "transaction" (tid) and a set of rows as a "set of transactions" (tidset). Notice in Figure 4.9(b) that the closed frequent itemset is the maximal non-redundant frequent itemset for every different tidset ({D}, {C,D}, {D,W}, {C,D,W}, {A,D,W}, {D,T}, {A,C,D,W}, {A,D,T,W}, {C,D,T}).

(a) dataset

Tid	itemset
1	A, C, D,T, W
2	C, D, W
3	A, D, T, W
4	A, C, D, W
5	A, C, D, T, W
6	C, D, T

(b) Frequent predicate set and tidset with minimum support 50%

TidSet	Frequent sets *L*
123456	**{D}**
12456	{C}, **{C,D}**
12345	{W}, **{D,W}**
1245	{C,W}, **{C,D,W}**
1345	{A}, {A,D}, {A,W}, {**A,D,W}**
1356	{T}, {**D,T}**
145	{A,C}, {A,C,W}, {A,C,D}, {**A,C,D,W}**
135	{A,T}, {T,W}, {A,D,T}, {A,T,W}, {D,T,W}, {**A,D,T,W}**
156	{C,T},{**C,D,T}**

c) predicates

A = contains(Port), C = contains(Hospital), W = crosses(WaterBody),
T = contains(Factory), D = contains(Street),

Figure 4.9: Frequent sets and closed frequent sets

Considering the pair {A,W} with a dependence, observe in Figure 4.9 (b) that the closed frequent set approach does not warrant the elimination of well known geographic dependences, since among the 9 closed frequent sets, 3 have the dependence {A,W} ({A,D,W}, {A,C,D,W},{A,D,T,W}).

If we eliminate geographic dependences from the closed frequent sets ({A,D,W}$_{(1345)}$, {A,C,D,W}$_{(145)}$, and {A,D,T,W}$_{(135)}$), the information in the non-closed frequent sets occurring in the same transactions of the closed frequent sets is lost. The elimination of {A,D,W} eliminates information of the sets {A,D} and {A,W} in transactions 1345. The elimination of {A,C,D,W} loses the information of {A,C}, {A,C,W}, and {A,C,D} in the tidset 145, and the elimination of {A,D,T,W} loses the information of the sets {A,T}, {T,W},{A,D,T}, {A,T,W}, and {D,T,W} in the tidset 135.

Figure 4.10 (left) shows the meet-semilattice of closed frequent sets containing the dependence {A,W}. By eliminating from the closed frequent sets all sets with the geographic dependence {A,D,W},{A,C,D,W}, {A,D,T,W}, as shown in Figure 4.10 (right), the information lost cannot be retrieved anymore.

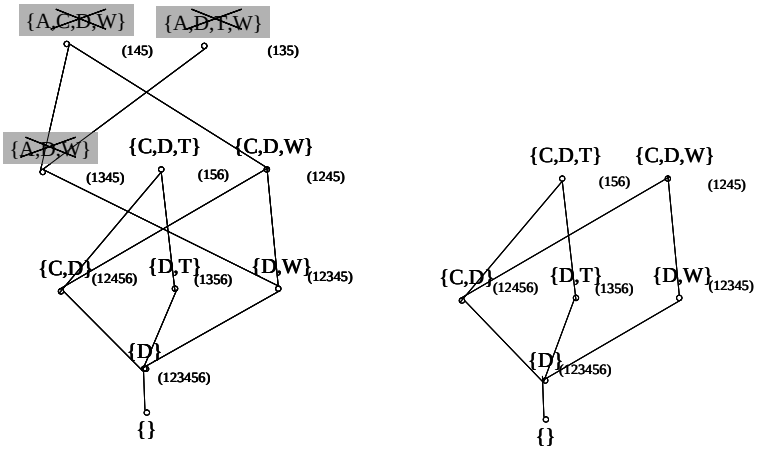


Figure 4.10: (left) Closed frequent sets with geographic dependences and (right) closed frequent sets without well known geographic dependences

If we eliminate geographic dependences from the frequent sets and than apply the closet frequent set approach, the resultant frequent sets are not closed in relation to the dataset, although they are in relation to the frequent sets. For example, in Figure 4.9 (b) there are 6 frequent sets obtained from the tidset 135 ({A,T}, {T,W}, {A,D,T}, {A,T,W}, {D,T,W}, {A,D,T,W}). Two frequent sets {A,T,W} and {A,D,T,W} contain the dependence {A,W}, and {A,D,T,W} is the closed frequent set. By removing the sets with dependences {A,T,W} and {A,D,T,W} from the frequent sets, the remaining sets {A,T}, {T,W}, {A,D,T}, {D,T,W} are not closed in relation to the dataset because a larger set {A,D,T,W} can be generated from the dataset. Indeed, {A,T} and {T,W} are redundant in relation to {A,D,T} and {D,T,W} respectively.

To overcome this problem by removing redundant frequent sets without well known dependences we propose to generate *maximal non-redundant frequent sets* with the algorithms Max-FGP, presented in the following section.

4.3.2 Max-FGP

Max-FGP eliminates well known geographic dependences as in Apriori-KC, and eliminates all redundant frequent sets that occur in the same transactions, similar to the closed frequent set approach. Considering our example shown in Figure 4.9(b) we propose to first eliminate the dependences, and then for all frequent sets occurring in the same transactions, only the maximal non-redundant frequent sets remain, the others are eliminated.

To illustrate this process, Figure 4.11 (left) shows the meet-semilattice of the frequent sets without well known geographic dependences {A,W} generated by Apriori-KC. The elimination of the redundant frequent sets from these sets results in the meet-semilattice shown in Figure 4.11 (right). This method reduces the 19 frequent sets without dependences to 11 maximal non-redundant frequent sets without dependences and without losing information.

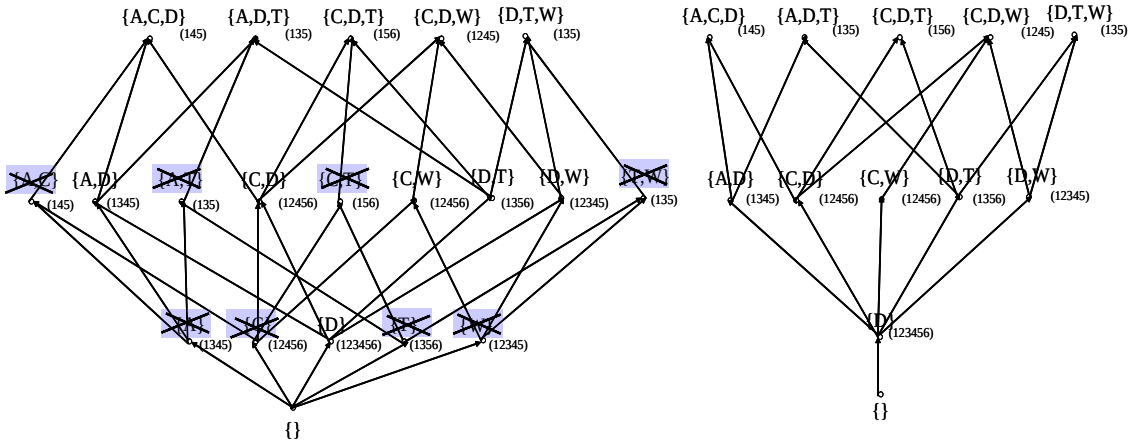


Figure 4.11: (left) Frequent sets without geographic dependences and (right) maximal non-redundant frequent sets without well known dependences

Definition 4 (maximal frequent geographic patterns without well known dependences): a frequent geographic pattern L is maximal when it has no well known geographic dependence in $\varnothing$ such that $L\text{-}\varnothing=L$ and $M(L)=L$.

The Maximal operator M associates with a frequent predicate set L the maximal set of predicates common to all transactions containing L without well known geographic dependences, i.e., L is maximal if there is no frequent predicate set L' in the same transactions of L such that $L \subset L'$.

Considering the frequent sets generated from the tidset 135 without well known dependences ({A,T}, {T,W}, {A,D,T}, and {D,T,W}) shown in the meet-semilattice in Figure 4.11 (left), notice that {T,W}⊂{D,T,W} and {A,T}⊂{A,D,T}, so neither {T,W} nor {A,T} are maximal. However, {A,D,T}⊄{D,T,W}, so both {A,D,T} and {D,T,W} are maximal. Considering the tidset 135, while only one frequent set is closed ({A,D,T,W}), but having a geographic dependence, two frequent sets ({A,D,T} and {D,T,W}) in these set of transactions (135) are *maximal*, but *without* well known geographic dependences. In this example we can observe that it is possible to have more maximal frequent sets than closed frequent sets, since the dependence elimination avoids the generation of the largest frequent sets, which in general, are closed.

Figure 4.12 shows an overview of the Max-FGP algorithm that extracts maximal frequent geographic patterns without well known dependences. Given a set L of frequent sets generated with Apriori-KC presented in Section 4.2 and a geographic dataset Ψ generated with our preprocessing method presented in Section 4.1, Max-FGP starts the generalization similarly to the closed frequent set approach. All frequent sets M, with size k are compared to the sets with size k+1. When a set $M_k \subset M_{k+1}$ and the set of transactions (tidset) in which M_k appears is the same as the transactions where $M_{k+1,}$ appears, than we can say that M_k is redundant, while M_{k+1} is more general. When this

occurs, M_k is removed from M. This process continues until all frequent sets in M have been tested.

```
Given: L_k; // frequent sets generated with Apriori-KC
       Ψ; // dataset generated with spatial_predicate_extraction

Find: Maximal M

   // find maximal generalized predicate sets

M = L;
For ( k = 2; M_k != ∅; k++ ) do  begin
     For ( j = k+1;  M_j!=0;  j++ ) do  begin
           If  (tidSet (M_k) = tidSet (M_j))
                    If (M_k ⊂ M_j)   // M_j is more general than M_k
                                          Delete M_k from M;
    End;
End;
Answer = M;
```

Figure 4.12 : Pseudo-code of the algorithm Max-FGP

In this chapter we presented three main methods to improve the process of mining spatial association rules from geographic databases: intelligent geographic data preprocessing; pruning geographic dependences and predicates with the same feature types or relationships in the frequent set generation; and the computation of maximal non-redundant frequent sets without well known dependences.

In the next chapter we evaluate the proposed methods with experiments performed over real geographic databases.

5 EXPERIMENTS AND EVALUATION

In this chapter we present experiments with different real geographic databases in order to evaluate and validate the three pruning methods for frequent geographic pattern mining proposed in this thesis: data preprocessing (or input space pruning), frequent set pruning, and maximal frequent set generation. The first database was provided by Procempa, which stores many different spatial feature types of the city of Porto Alegre. This database contains information about infra-structure, transportation, hydrography, etc. The second database is from a region in Northern Brazil, and contains information about vegetation, transportation, and hydrography.

Both geographic databases were preprocessed with different spatial feature types, different spatial relationships, and different granularity levels, in order to generate the following datasets, which were mined to extract frequent geographic patterns using the new methods proposed in this thesis:

a) *dataset* 1: extracted from database 1, this is a dataset with 18 spatial predicates including the relevant feature types trees, treated water network, slums, cellular antennas, water resources, water collection points, illumination points, hydrants, hydrographic sub-basin, gas stations, streets, schools, hospitals, health centers, industrial residues repositories, water treatment stations, and artesian wells. This dataset has *two* dependences between the target feature type (district – 109 multi-polygons) and the relevant feature types (*illumination points* and *treated water network*), and *three* dependences among the relevant feature types (*water resource* and *hydrographic sub-basin, water resource* and *water collection points, gas stations* and *streets*). This dataset was preprocessed considering the spatial relationship *intersects.*

b) *dataset* 2: extracted from database 1, this is a dataset with 17 spatial predicates including the relevant feature types trees, treated water network, slums, cellular antennas, streams, water collection points, illumination points, hydrants, bus stops, streets, hydrographic basin, sewer network, etc. This dataset has *one* dependence between the target feature type (district – 109 multi-polygons) and the relevant feature types (*illumination points*) and *two* dependences among the relevant feature types (*stream* and *hydrographic sub-basin, bus stop* and *street*). This dataset was preprocessed considering the spatial relationship *intersects.*

c) *dataset* 3: extracted from database 1, this dataset has 15 spatial predicates including feature types streams, slums, hospitals, gas stations, streets, etc. This

dataset has *one* dependence between the target feature type (census sectors – 2.159 multi-polygons) and the relevant feature type *illumination points* and *one* dependence among the relevant feature types (*gas station* and *streets*). This dataset was also preprocessed considering the spatial relationship *intersects.*

d) *dataset* 4: extracted from database 1, this is exactly the same dataset 2, but the feature type street is considered at granularity level 2 (*street*, *avenue*, and *square*). In this dataset there is *one* dependence between *stream* and *hydrographic sub-basin* and 3 hierarchical dependences: *bus stop* and *street*, *bus stop* and *avenue* and *bus stop* and *square.* This dataset was also preprocessed considering the spatial relationship *intersects.*

e) *dataset* 5: extracted from database 1, this dataset was preprocessed considering *topological* relationships between the target feature type (districts – 109 multi-polygons) and 14 relevant feature types, generating a total of 70 predicates. This dataset has a large amount of dependences among relevant feature types. The dependence between *streams* and *hydrographic sub-basins* with different topological relationships generated 4 pairs of dependences {*contains(Stream), overlaps(SubBasin)}, {contains(Stream), within(SubBasin)}, {crosses(Stream), overlaps(SubBasin)}, {crosses(Stream), within(SubBasin)}*. Besides the 4 pairs with dependences, 9 pairs of predicates with the same feature type and different topological relationships were also in this dataset, resulting in a total of 13 pairs to be eliminated (e.g. *{overlaps(subBasin), within(subBasin)}, {crosses(Stream), contains(Stream)}, {contains(Slum), overlaps(Slum)}*).

f) *dataset* 6: extracted from database 2, this dataset has 3 non-spatial attributes and *topological* relationships were computed between the target feature type vegetation (300 multi-polygons) and 7 relevant feature types, resulting in a total set of 38 predicates. Two dependences among relevant feature types were in this dataset: *bridge* and *river,* and *tunnel* and *road.* For different topological relationships these two dependences were replicated in six pairs: {*touches(Tunnel), crosses(Road)*}, {*touches(Tunnel), contains(Road)*}, {*contains(Tunnel), crosses(Road)*}, {*contains(Tunnel), contains(Road)*}, {*contains(Bridge), crosses(River)*}, {*contains(Bridge), contains(River)*}. Besides the pairs with dependences, 7 pairs have same feature types {*contains(DisturbedSoil), overlaps(DisturbedSoil)}, {contains(River), crosses(River)}, {overlaps(GroundSurface), touches(GroundSurface)}, {contains(Tunnel), touches(Tunnel)}, {contains(Road), crosses(Road)}, {contains(GrassField), touches(GrassField)},* and *{contains(RiceField), touches(RiceField)}*.

g) *dataset* 7: extracted from database 1, with 20 spatial predicates, this dataset is similar to dataset 1, but considering only dependences between the target feature type (district – 109 multi-polygons) and the relevant feature types *illumination points, hydrants,* and *blocks.* This dataset was preprocessed considering topological relationships, while in dataset 1 only intersects was considered.

h) *dataset* 8*:* extracted from database 1, this dataset has 12 predicates and 105 rows, where the relationship *distance* was considered between the target feature type (health center) and 6 relevant feature types. One dependence between the relevant feature types gas stations and streets was in this dataset. Considering

close (>600 meters and <1000 meters,) and very close (<599 meters) relationships, this dependence is transformed in 4 pairs to be eliminated {*close(GasStation), close(Street)*}, {*close(GasStation), veryClose(Street)*}, {*very_close(GasStation), close(Street)*}, and {*very_close(GasStation), very_close(Street)*}.

i) *dataset* 9: this dataset is similar to dataset 8. *Distance* relationships were considered and exactly the same dependences were considered. The difference is that 18 predicates were generated with 9 relevant feature types. This dataset will be used to show that although the number of predicates increases, the dependence replication remains constant in the frequent set generation.

j) *dataset* 10*:* extracted from database 1, this dataset has 411 predicates represented at the feature instance granularity level. Three predicates are non-spatial attributes – criminality rate, danger, and treated water network. The remaining predicates are the instances of 13 different relevant feature types including sewer network, police offices, police covering areas, hospitals, schools, health centers, artesian wells, streams, illumination points, hydrants, etc. This dataset has 513 rows, each of which is a slum, represented in the geographic database by 513 multi-polygons.

k) *dataset* 11*:* extracted from database 2, this dataset has 50 predicates and 300 rows generated at the feature type granularity level, considering *topological* relationships. Relevant feature types include different types of vegetation such as disturbed soil, grassfield, ricefield, cropland, land subject to inundation, and build up area. These predicates have the same parent in a concept hierarchy “soil type”, and generated predicates with different topological relationships.

l) *dataset* 12: extracted from database 2, this dataset is similar to dataset 11. Generated at the feature type granularity level, the relationship *intersects* was considered, and 10 predicates were generated from relationships between the target feature type and 10 relevant feature types.

In the experiments described in this chapter, data preprocessing was performed with Weka_GDPM (BOGORNY, 2006d) which is described in Chapter 6. The data mining steps were performed with a prototype implemented in Matlab. Experiments were performed in a Pentium M, 1.5 GHz, with 752MB or ram memory, and Windows-XP operational System.

An overview of the experiments performed with the different datasets is shown in Figure 5.1. The boxes are different datasets and the rectangles summarize the experiments, which will be described in the following sections.

In Section 5.1 we evaluate the data preprocessing method to extract spatial predicates (spatial join), as well as the frequent set and spatial association rule reduction using Apriori and the closed frequent set approach when removing geographic dependences between the target feature type and the relevant feature types. In this section we present different experiments, using datasets 2, 3, and 7. We evaluate the data preprocessing method using Apriori and the closed frequent set approach, because the input space pruning method is independent of rule mining algorithm. Section 5.1 describes the experiment “A” shown in Figure 5.1.

In Section 5.2 both algorithms Apriori-KC and Max-FGP are evaluated with both input space and frequent set pruning, using datasets 1, 2, 3, 8, 9, and 10. This section describes the experiment "B" illustrated in Figure 5.1.

In Section 5.3 we evaluate both algorithms Apriori-KC and Max-FGP for the hierarchical dependence elimination when mining geographic data at different granularity levels. In this evaluation datasets 2 and 4 are used, as can be observed in Figure 5.1 "C".

In Section 5.4 the dependence elimination is evaluated for frequent sets generated with same feature types having different topological relationships (e.g. {*contains(River), crosses(River)*}). In this experiment we also evaluate the dependence elimination when one pair of dependences is replicated to many pairs because of different topological relationships (e.g. {*contains(Bridge), contains(River)*}, {*contains(Bridge), touches(River)*}). In this experiment datasets 5 and 6 were used (see experiment "D" in Figure 5.1).

In Section 5.5 we evaluate step 4 of Apriori-KC, i.e., the elimination of pairs of predicates that contain geographic objects with the same parent in a concept hierarchy. This experiment uses the datasets 11 and 12, as shown in Figure 5.1 "E".

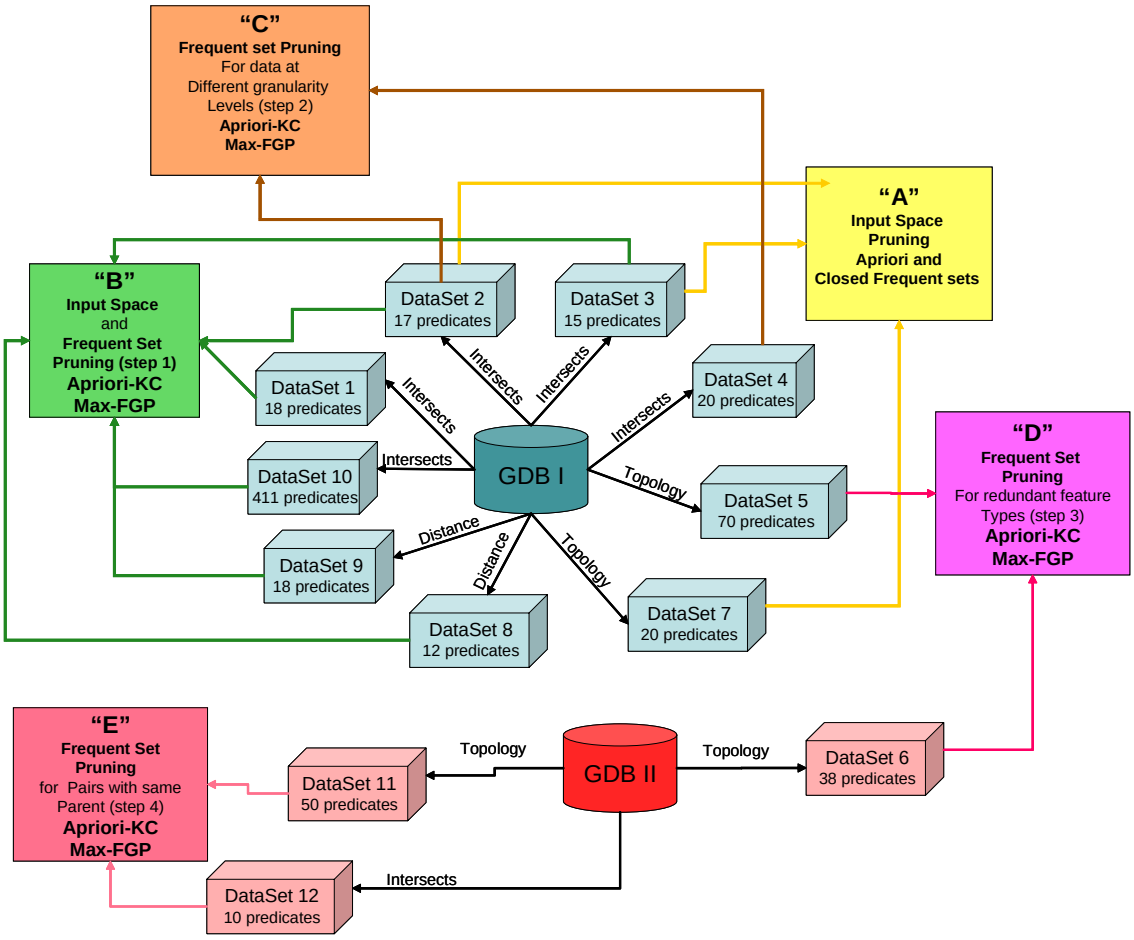


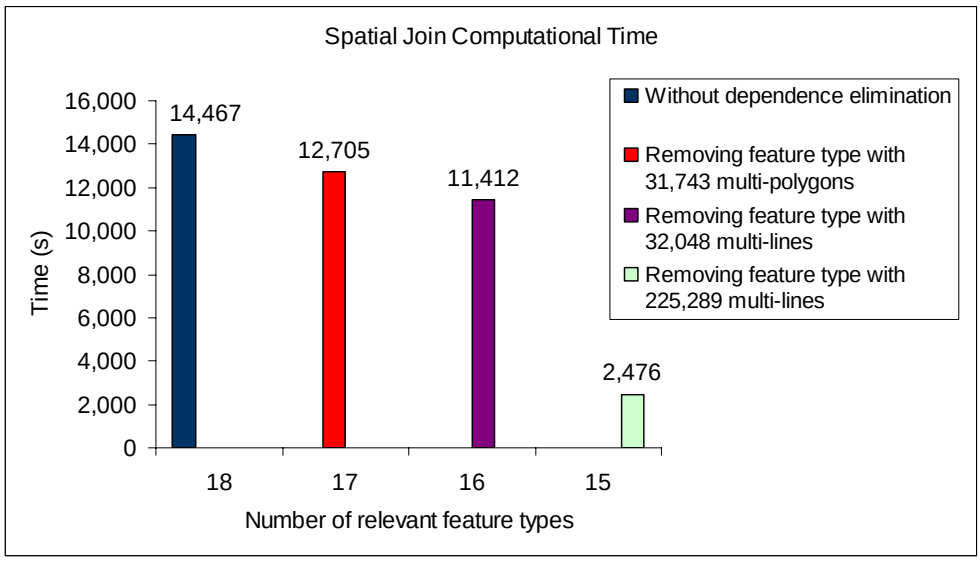
Figure 5.1: Experimental Scheme

In the remaining of this chapter we may refer to steps of dependence elimination between the target feature type and relevant feature types as "input space pruning". We

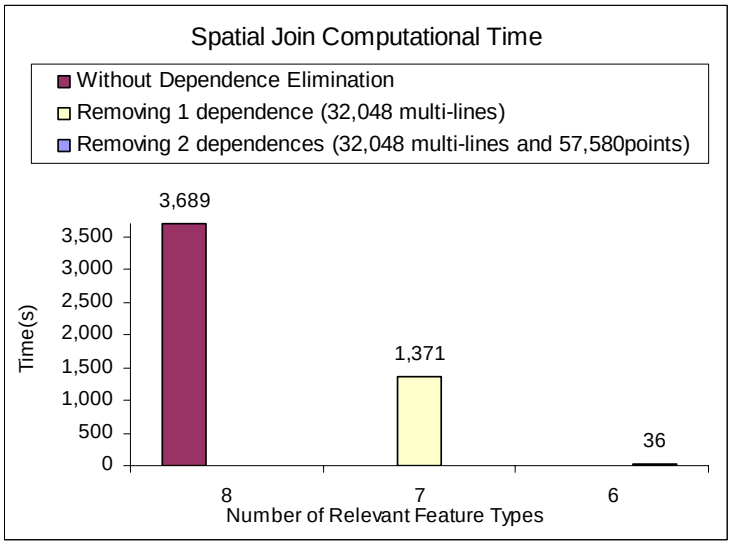
may also refer to dependence elimination among relevant feature types (steps 1 and 2 of Apriori-KC) and steps 3 and 4 in Apriori-KC as “frequent set pruning”.

5.1 Data Preprocessing - Pruning Input Space

The input space pruning method proposed to eliminate geographic dependences between the target feature type and any relevant feature type is applicable to any association rule mining algorithm, since the dependence elimination occurs in data preprocessing, before the frequent set computation. We first evaluate the spatial join computational time, with dataset 7, as shown in Figure 5.2.



(a)



(b)

Figure 5.2: Spatial join computational time

The spatial join computational time is totally data dependent. As can be observed in Figure 5.2 (a), where we preprocessed a geographic database considering 18 relevant feature types, if one relevant feature type with 31,743 multi-polygons is eliminated, spatial join computational time reduces in 13%. If one more relevant feature type with

32,048 multi-lines is eliminated, spatial join computational time reduces around 23%. When one more relevant feature type with 225,289 multi-lines is eliminated, computational time reduces 83%. This experiment shows that according to the geometry type and the number of instances of the relevant feature type that is eliminated, the spatial join computational time can be dramatically reduced.

Figure 5.2(b) shows another experiment, where a dataset with only 8 relevant feature types was generated. The elimination of different relevant feature types that have geographic dependences with the target feature type can reduce the computational time very significantly. In this experiment, time reduces in 99% when the eliminated feature types have the highest number of instances.

To evaluate the frequent set reduction by pruning the input space, Figure 5.3 (left) shows an experiment performed over dataset 7 using the Apriori algorithm. This dataset was mined considering different values of *minsup* (10%, 15%, and 20%). For each different *minsup*, the dataset was mined three times: (a) without removing dependences, (b) removing one dependence (column), and (c) removing two dependences (columns). As can be observed in Figure 5.3(left) the input space pruning reduces frequent patterns for all different values of minimum support. The elimination of one dependence pruned the frequent sets around 50%, and the elimination of two dependences reduced the total number in around 75% for any value of minimum support.

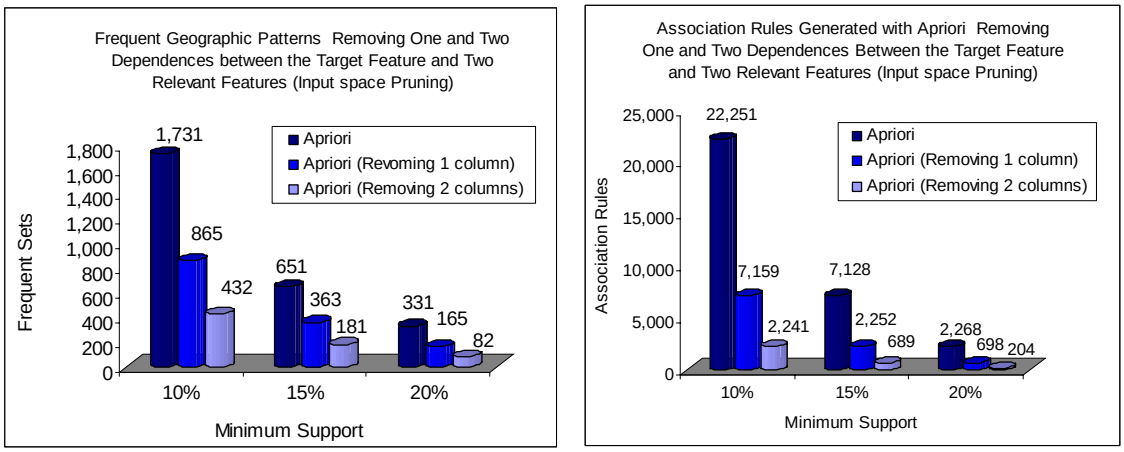


Figure 5.3: (left) Frequent sets and (right) association rules generated with Apriori after pruning input space using dataset 7

The frequent set reduction warrants the association rule reduction, since associations are generated from frequent sets. Using Apriori to generate frequent sets and association rules and considering *minconf* 70%, Figure 5.3(right) shows that the rule reduction is still more significant. By removing one single dependence in data preprocessing the rule reduction reaches around 70%. The elimination of two dependences reduced the number of rules in 90% for any value of *minsup*.

A second experiment was performed, now over dataset 2, and considering lower minimum support (5%, 10%, and 15%). On this experiment we evaluated the number of frequent sets generated by Apriori, the number of closed frequent sets, as well as the computational time. Figure 5.4 (left) shows that the frequent sets generated by Apriori are reduced in more than 50% for any value of minimum support when one single dependence (column) is eliminated in data preprocessing. For the closed frequent sets the reduction is not as significant as it is for Apriori. Only one closed frequent set is

eliminated for any value of *minsup*. However, the computational time to generate the closed frequent sets after input space pruning is reduced in more than 60%, as shown in Figure 5.4 (right). For Apriori the computational time is reduced in 50% for any value of *minsup*, by removing one single dependence.

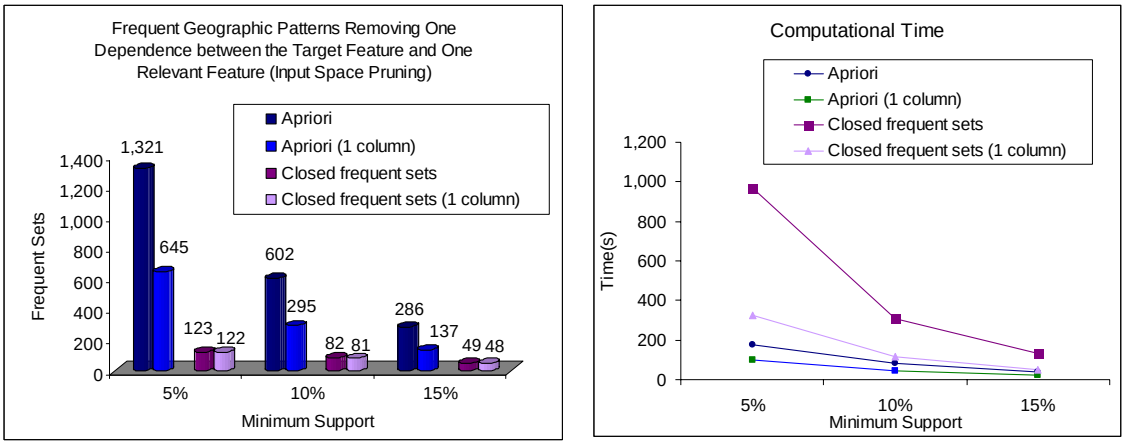


Figure 5.4: (left) Number of frequent sets and closed frequent sets and (right) computational time to generate frequent sets and closed frequent sets with input space pruning using dataset 2

A third experiment was performed, now using dataset 3. The result is shown in Figure 5.5. In dataset 3 although the number of rows is much higher (target feature type has 2,159 multi-polygons) than in dataset 2 (target feature type has 109 polygons), spatial predicates have lower support. By consequence, the number of frequent sets generated by Apriori is very low. Considering *minsup* 5%, 10%, and 15% the number of frequent sets generated by Apriori is reduced in more than 50% for any value of minimum support applying the input space pruning method. The closed frequent sets are reduced in one set, which is the largest closed frequent set. Similarly to the experiment performed over dataset 2, the computational time for both algorithms is reduced in around 50% with the elimination of one dependence in data preprocessing.

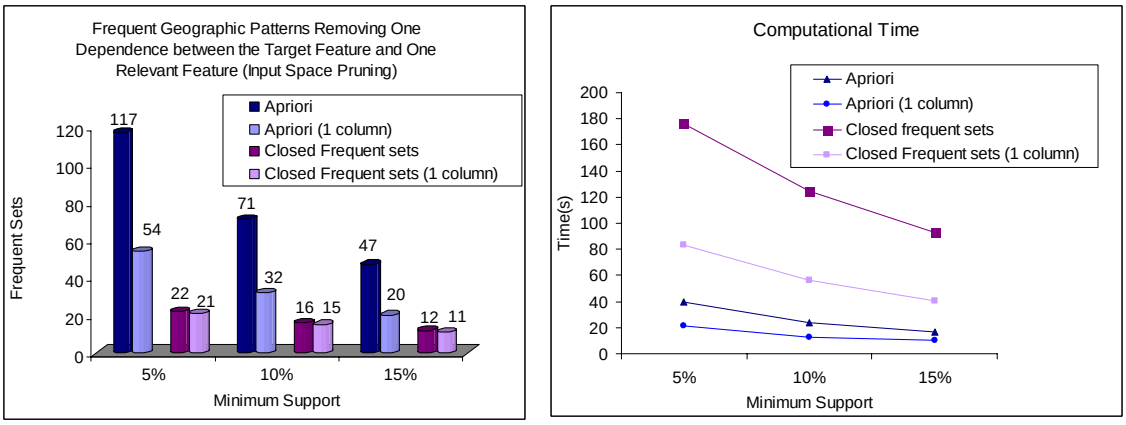


Figure 5.5: (left) Number of frequent sets and closed frequent sets and (right) computational time to generate frequent sets and closed frequent sets with input space pruning using dataset 3

We can conclude with the different experiments, using different datasets, and different values of *minsup*, that the percentage reduction in both time and frequent sets is very significant when our input space pruning method is used to eliminate geographic

dependences between the target feature type and relevant feature types. In all experiments the elimination of one dependence reduces frequent sets in around 50%.

Without considering our input space pruning method we can say that Apriori and the closed frequent set approach deal with a number of frequent sets *Q*, where *n* is the number of predicates in the dataset, such that

$$Q = \sum_{i=2}^{n} C_n^i$$

Using our input space pruning method the number of frequent sets is reduced to *Q'*, where *d* is the number of dependences eliminated in data preprocessing between the target feature type and the relevant feature types, such that

$$Q' = \sum_{i=2}^{n-d} C_{n-d}^i$$

5.2 Evaluating Apriori-KC and Max-FGP for single Dependence Elimination

In this section we evaluate Apriori-KC and Max-FGP for many different datasets. Indeed, only the single dependence elimination (step 1 of Apriori-KC) is evaluated.

5.2.1 Experiment with Dataset 3

Figure 5.6 shows the result of an experiment performed over dataset 3 where only the dependence among relevant feature types (1 pair) was eliminated during the frequent set generation, without input space pruning. As can be observed in Figure 5.6(left), the elimination of one single dependence with Apriori-KC reduces the number of frequent sets for all different values of minimum support. This reduction reaches between 28% and 30% of the total number of frequent sets generated for any value of *minsup*. Max-FGP reduces this number much further, in more than 70% for any value of *minsup*.

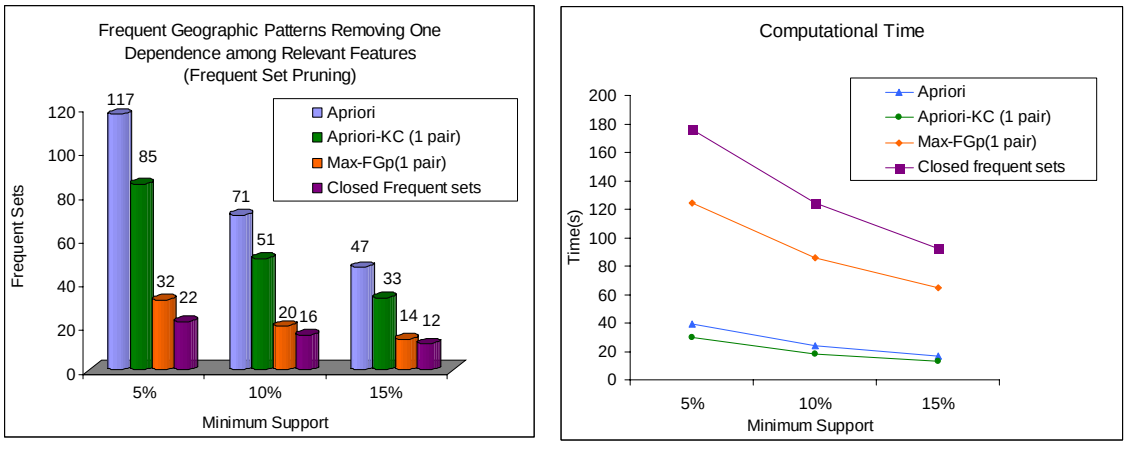


Figure 5.6: (left) Pruning frequent sets and (left) computational time

The closed frequent sets generate the lowest number of frequent sets. However, among the 22, 16, and 12 closed frequent sets generated for *minsup* 5%, 10%, and 15% respectively, 17, 11, and 7 contain the pair with the dependence. These numbers represent 77%, 68%, and 58% of the total number of closed frequent sets for the

respective *minsup* 5%, 10%, and 15%. At lower *minsup* most closed frequent sets contain the well known geographic dependence. Geographic dependences can be reduced in frequent sets by increasing *minsup*, but are far from be totally eliminated by the *minsup* threshold.

The computational time to extract frequent patterns and maximal patterns with Apriori-KC and Max-FGP is also reduced, as shown in Figure 5.6(right). The computational cost to generate closed frequent sets and maximal frequent sets, of course is higher than to simply generate frequent sets as in Apriori or Apriori-KC. The additional verification to generate maximal or closed frequent sets requires extra scans over the dataset, as well as the comparison of all sets of size k with sets of size k+1. This experiment shows that the closed frequent set approach, apart from not eliminating well known geographic dependences, requires more computational time than our two methods (Apriori-KC and Max-FGP), which both eliminate geographic dependences.

Figure 5.7(left) shows a similar experiment, but in this case well known dependences were eliminated in both input space (between the target feature and relevant features) and during the frequent set generation (among relevant features). The total number of frequent sets generated by Apriori-KC is reduced in around 70% by removing one column in data preprocessing and one pair with dependences during the frequent set generation, independently of minimum support. Max-FGP reduces this number in 75% for any value of *minsup*.

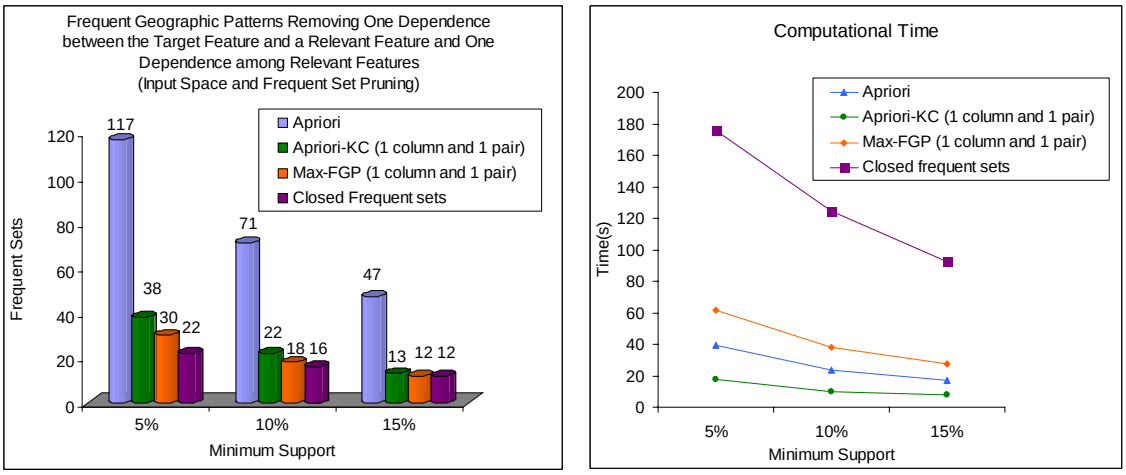


Figure 5.7: (left) Pruning both input space and frequent sets (right) computational time

Figure 5.7 (right) shows that besides generating less frequent sets and maximal frequent sets, time reduces significantly when removing any kind of dependences. However, pruning both input space and frequent sets makes our methods still more efficient.

These experiments showed that in the geographic domain most frequent sets contain well known geographic dependences. Our input space pruning method and Apriori-KC completely eliminate such dependences, while Max-FGP eliminates all redundant frequent sets without well known dependences.

5.2.2 Experiments with Dataset 2

Figure 5.8 shows the experiment performed over dataset 2, that has two dependences among relevant feature types. Figure 5.8 (left) shows Apriori and closed frequent sets *without* removing any geographic dependences, in order to evaluate the amount of

frequent sets that contain dependence. Considering *minsup* 5%, from a total of 1.321 frequent sets generated by Apriori, 454 contain dependences. The closed frequent set technique reduced this number in 90%, to 123 closed frequent sets. However, among these 123 closed frequent sets, 66 contain geographic dependences. Considering the different values of minimum support we observe that the number of both frequent sets and closed frequent sets reduces significantly, but geographic dependences are not eliminated. While among the frequent sets generated by Apriori around 30% contain dependences, among the closed frequent sets around 50% contain geographic dependences, for different values of minimum support. This experiment shows again that the closed frequent set approach reduces the frequent sets but does not eliminate well known geographic dependences.

As geographic dependences cannot be removed from closed frequent sets without losing information, Figure 5.8 (right) shows the respective frequent sets generated by Apriori-KC and Max-FGP over the same dataset. Apriori-KC reduced around 30% the number of frequent sets generated by Apriori, removing all dependences. Max-FGP pruned the number of frequent sets without geographic dependences much further. This reduction reaches around 80% for any value of minimum support.

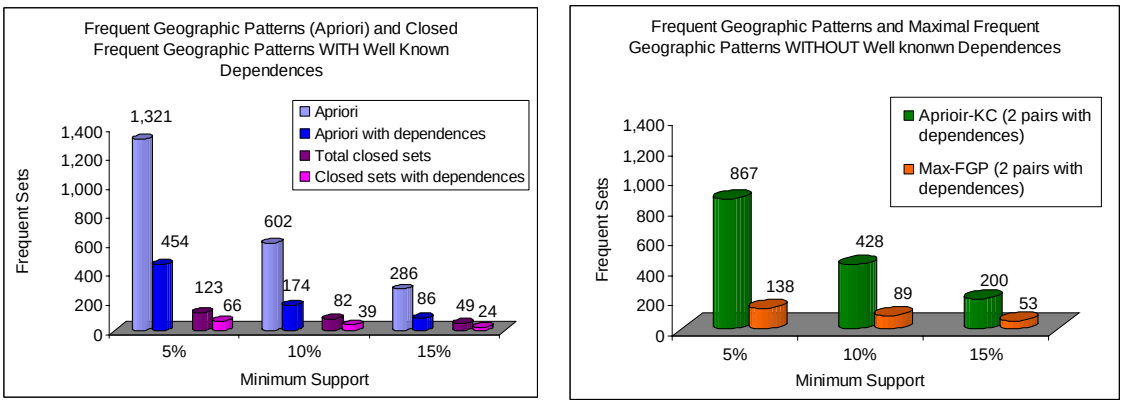


Figure 5.8: (left) Frequent sets (Apriori) and closed frequent sets WITH geographic dependences and (right) frequent sets (Apriori-KC) and maximal frequent sets (Max-FGP) WITHOUT dependences

Besides the significant reduction in the number of frequent sets, Apriori-KC and Max-FGP are more efficient. The elimination of geographic dependences reduces the frequent sets with size *k*, which by consequence reduces the number of frequent sets with size *k*+1. Figure 5.9 shows the computational time for the extraction of frequent sets with Apriori and the closed frequent set approach (which both do not eliminate geographic dependences), and Apriori-KC and Max-FGP (which do eliminate geographic dependences).

In general words, while Apriori generates frequent sets Q with n predicates, such that

$$Q = \sum_{i=2}^{n} C_n^i$$

Apriori-KC generates frequent sets

$$Q'(n,1) = \sum_{i=2}^{n-1} \left(C_n^i - C_{n-2}^{n-i} \right)$$

when one pair with a dependence is eliminated, and

$$Q''(n,2) = \sum_{i=2}^{n-2} \left(C_n^i - 2.C_{n-2}^{n-i} \right)$$

when 2 dependences are eliminated, and so on.

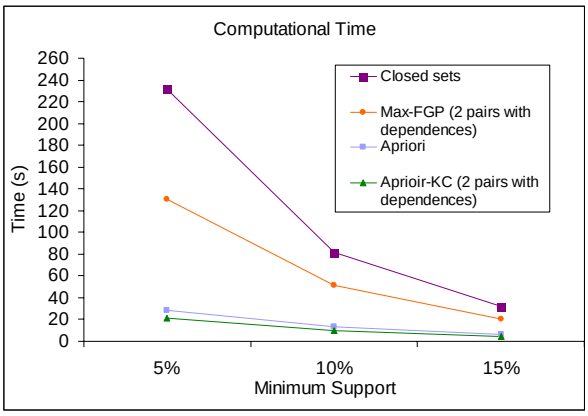


Figure 5.9: Computational time to generate frequent sets (Apriori) and closed frequent sets without removing dependences, and frequent sets (Apriori-KC) and maximal frequent sets (Max-FGP) removing 2 pairs of objects with dependences

By reducing the number of frequent sets, the number of association rules is automatically reduced, since rules are extracted from frequent sets. The general percentage reduction of rules for different numbers of attributes, produced when zero (as reference), one, and two pairs of dependences are eliminated is shown in Figure 5.10. This graphic considers all possible rules (*minconf=0*). The elimination of one pair generates only 55% of the total number of rules, i.e., eliminates well known rules in 45%. When two pairs are eliminated, only 30% of the rules are created, and the reduction increases to 70%. Notice that even if the number of elements increases, these values represent saturation points for these curves. So we can conclude that the higher the number of well known dependences, the more significant is the rule reduction.

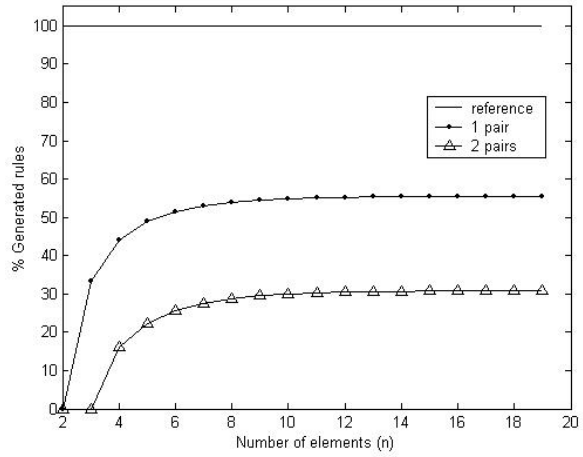


Figure 5.10: Percentage reduction of association rules considering zero (reference), one, and two pairs of dependences with an increasing number of elements (predicates)

5.2.3 Experiment with Dataset 1

Besides generating frequent sets without well known geographic dependences, our methods tend to reduce computational time when the number of dependences increases, since less frequent sets will be generated. This can be clearly visualized in the experiment shown in Figure 5.11, performed over dataset 1, with minimum support 10%, 20%, and 30% , where one, two, and three pairs of dependences were eliminated from the dataset. Figure 5.11 (left) shows three graphics removing 1 pair, two pairs, and three pairs of dependences respectively. Figure 5.11(right) shows the computational time.

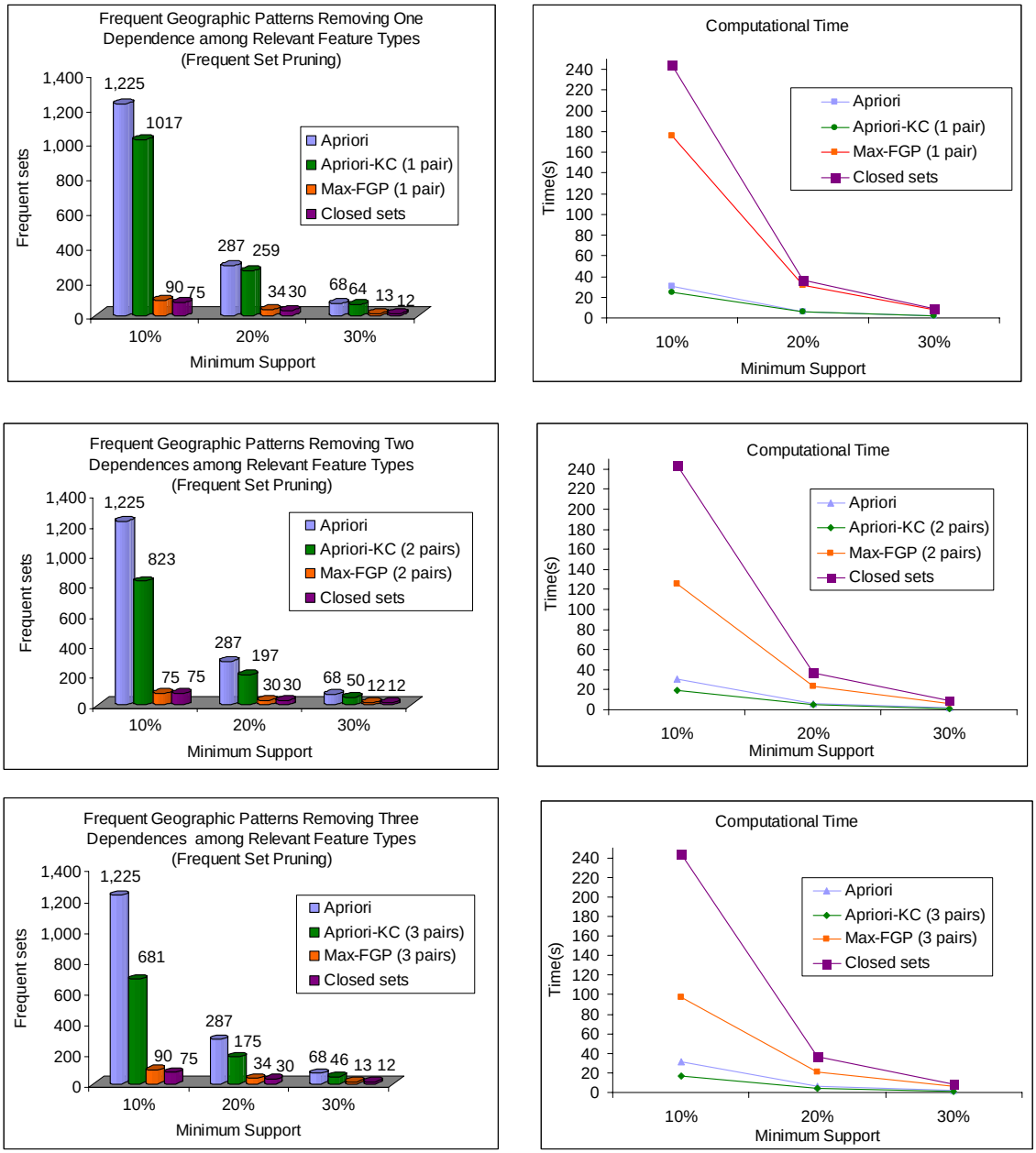


Figure 5.11: (left) Frequent geographic patterns when one, two, and three pairs of dependences are eliminated and (right) computational time

In Figure 5.11(left) we observe that the higher the number of dependences to be eliminated the lower is the number of frequent sets generated by Apriori-KC in relation

to Apriori. For minimum support 5%, for instance, Apriori-KC reduces the number of frequent sets generated by Apriori in around 20% when one dependence is removed. This reduction increases to around 30% when two pairs are eliminated and 45% when 3 pairs are removed. Max-FGP, however, reduces the final number of frequent sets much further. In relation to Apriori-KC, Max-FGP reduces the number of frequent sets in around 80%, and in relation to Apriori the reduction reaches between 80% and 95% for different values of minimum support.

In this experiment the elimination of two pairs with dependences generated the same number of closed frequent sets and maximal frequent sets. However, while in the maximal frequent sets there are no dependences, more than 30% of the closed frequent sets contain dependences. In this experiment, among the 75, 30, and 12 closed frequent sets generated for different values of *minsup*, 40, 15, and 4 contain respectively at least one of the three pairs of dependences, that are *not* in the maximal frequent sets. So although the number of closed frequent sets is the same as the number of maximal frequent sets, more than 30% of the closed sets contain dependences.

Figure 5.11(right) shows that both Apriori-KC and Max-FGP are more efficient than Apriori and closed sets respectively, for any value of minimum support. Moreover, the gain is higher for lower *minsup*, i.e., when the number of frequent sets is larger. This can be clearly observed for *minsup* 10%. Indeed, our methods get more efficient when the number of dependences increases.

Figure 5.12 shows an analysis of the frequent sets generated by the different algorithms, for the experiment shown in Figure 5.11, for minimum support 10%. In Figure 5.12 (left) notice that the higher the number of pairs to be eliminated, the lower is the number of predicates in the frequent sets. By removing 1 pair, Apriori-KC does not generate frequent sets with 9 predicates. When 3 pairs are eliminated, the largest frequent set contains 7 predicates.

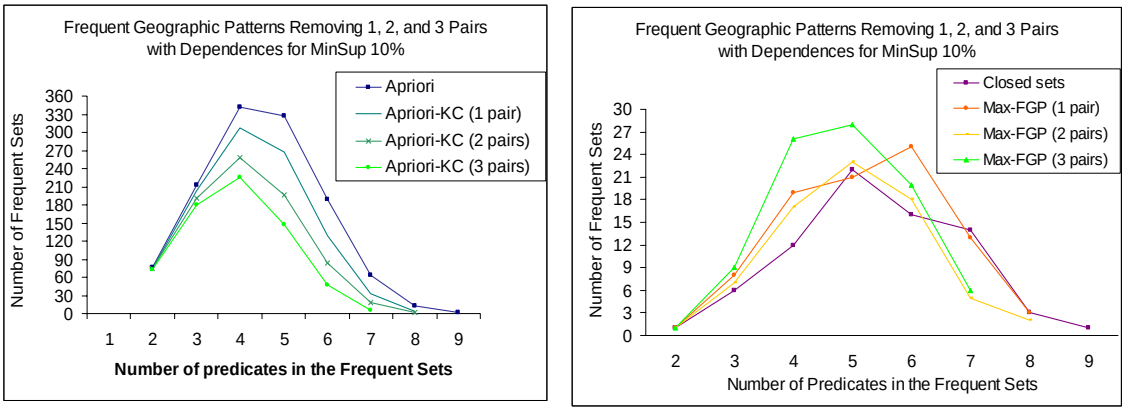


Figure 5.12: (left) Number of elements in frequent sets and (right) number of elements in maximal and closed frequent sets

A similar elimination occurs with Max-FGP, shown in Figure 5.12 (right). The closed frequent set approach generates sets with more elements (9 in the example), while Max-FGP generates frequent sets with less elements (8 when 1 and 2 pairs are eliminated and 7 when 3 pairs are eliminated).

The generation of frequent sets containing less elements has the advantage of generating less association rules, since the smaller the number of elements in a frequent set, the smaller is the number of rules.

5.2.4 Experiment with Datasets 8 and 9

In this section we present the result of two experiments performed over datasets 8 and 9, considering *minsup* 5%, 10%, and 15%. In both datasets the distance relationship was considered between the target feature type and the relevant feature types. The difference between the two datasets is the number of spatial predicates. Both datasets contain one geographic dependence among relevant feature types, which is replicated in four pairs because of the different possible relationships, as described at the beginning of this chapter.

The first experiment, performed over dataset 8, considering 12 predicates, is shown in Figure 5.13. As can be observed in Figure 5.13(left), the number of frequent sets generated by Apriori-KC is reduced in around 30% in relation to Apriori for any value of *minsup*. This reduction is more significant for Max-FGP, which eliminates redundant frequent sets and reduces the frequent sets in relation to Apriori in an average of 40%. In this experiment, apart from not eliminating geographic dependences, the closed frequent set approach generates more sets than Max-FGP. Figure 5.13(right) shows that both Apriori-KC and Max-FGP are more efficient than Apriori and closed frequent set approach, respectively, as in all previous experiments.

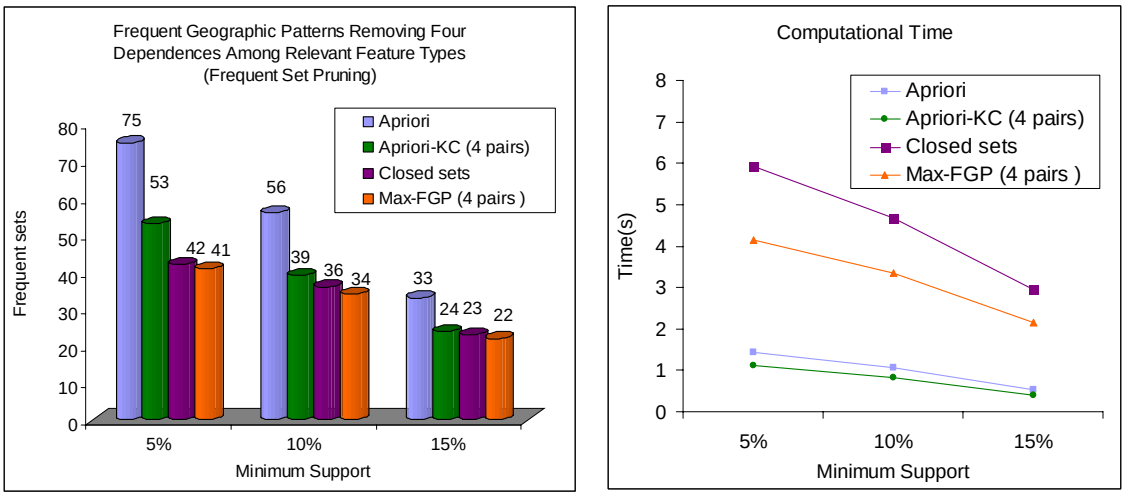


Figure 5.13: (left) Number of frequent sets and (right) computational time for mining frequent geographic patterns from dataset 8

Now let us compare the result of the experiment performed over dataset 9, that has more predicates than dataset 8, having the same geographic dependence and considering the same values of *minsup*. As can be observed in Figure 5.14(left), the number of frequent sets, closed frequent sets, and maximal frequent sets increased for different values of *minsup* in relation to the experiment in Figure 5.13(left). However, the *percentage reduction* of number of frequent sets after dependence elimination remains similar in both experiments (Figure 13 and Figure 14). For example, in Figure 5.13(left), for *minsup* 5%, the elimination of 4 pairs with dependences with Apriori-KC reduced the number of frequent sets generated by Apriori from 75 to 53, corresponding to a reduction of 29%. In Figure 5.14(left), for *minsup* 5%, Apriori-KC reduced the frequent sets generated by Apriori in 27%. For *minsup* 10%, the frequent sets in both experiments shown in Figure 5.13(left) and 5.14(left) is reduced in the same proportion. For *minsup* 15%, in both experiments the dependence elimination reduced around 28% of the total number of frequent sets. This experiments show that geographic dependences generate a certain percentage of frequent sets, independently of the number

of predicates in the dataset. This analysis is similar to that shown in Figure 5.10 for association rules.

While the frequent set reduction of Apriori-KC in relation to Apriori remains similar in the experiments performed over both datasets 8 and 9, Max-FGP is further more effective than both Apriori and Apriori-KC in dataset 9, as shown in Figure 5.14(left). While in Figure 5.13(left) Max-FGP reduced the number of frequent sets in relation to Apriori in around 45%, in Figure 5.14(left) this reduction reaches 80%. So we can conclude from this experiment that dataset 9 (which contains the same dependences as dataset 8 but with more attributes) generates significantly more redundant frequent sets than dataset 8. Therefore, Max-FGP if very effective.

Figure 5.14(right) shows that even increasing the number of attributes and removing the same dependences as in the previous experiment, Apriori-KC and Max-FGP are more efficient than Apriori and the closed frequent set approach, respectively.

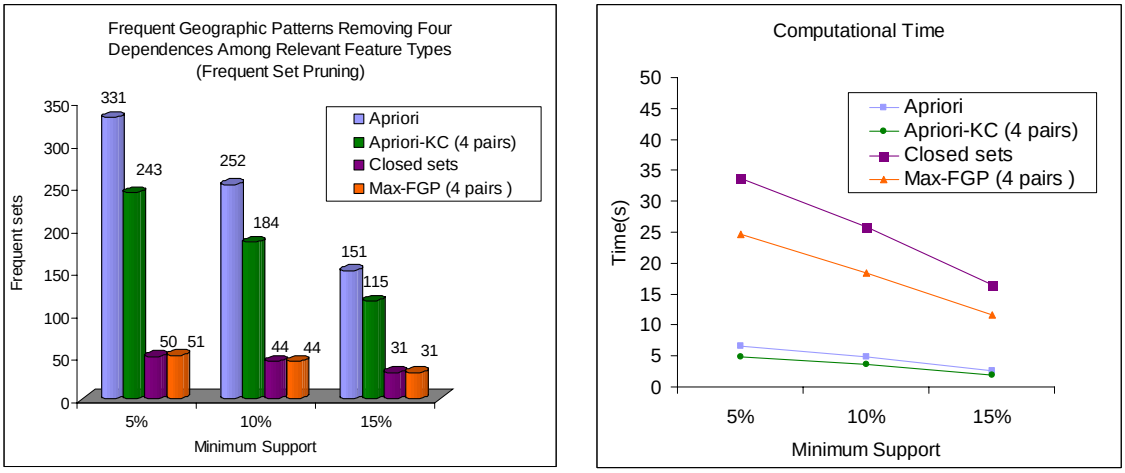


Figure 5.14: (left) Number of frequent sets and (right) computational time for mining frequent geographic patterns from dataset 9

5.2.5 Experiment with Dataset 10

This experiment was performed over dataset 10, which has 411 attributes at the feature instance granularity level. As we have already mentioned in chapter 3, predicates at the feature instance granularity have very low support, so this experiment was performed with *minsup* 3%, 4%, and 5%.

At the feature instance granularity level, geographic dependences also have lower support, because they represent the exact instances of two feature types that have a well known dependence. The support of pairs of predicates with feature instances such as {*intersects(*Stream_*ArroioPassoDasPedras), intersects(*Basin_*VarzeaDoGravatai)*} is much lower than the support of the pair of predicates with feature types {*intersects(Stream), intersects(Basin)*}, which is a generalization of all instances.

Even having lower support, geographic dependences appear in the frequent sets, as shown in Figure 5.15 (left). For *minsup* 3%, around 20% of the frequent sets contain dependences. Max-FGP reduced the number of frequent sets in 40%. For *minsup* 4% Apriori-KC reduced in 20% the number of frequent sets and Max-FPG around 35% in relation to Apriori. For *minsup* 5%, however, the reduction is not as significant, because less predicates with dependences reach minimum support.

Among the 411 predicates, 149 are streams that intersect at least one of the 513 slums. These 149 streams have a dependence with one of the 28 hydrographic basins. In this case there are at least 149 pairs of dependences, combining every stream with its respective basin. In this experiment, among the 149 possible pairs of dependences, only 2 reached *minsup* 5%, 10 reached *minsup* 4%, and 15 reached *minsup* 3%. Because of this low support the frequent set reduction is less significant than in previous experiments when mining data at higher granularity levels.

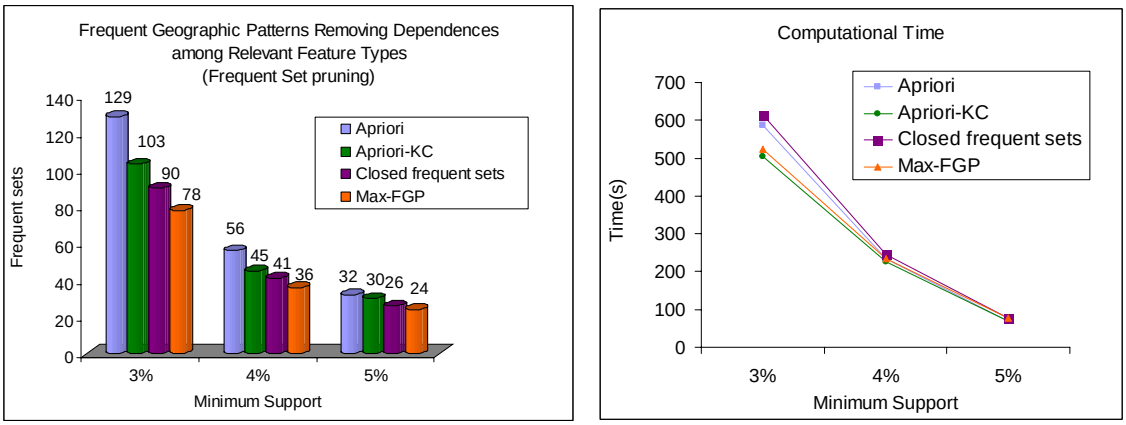


Figure 5.15: (left) Number of frequent sets and (right) computational time for mining frequent geographic patterns from dataset 10

In this experiment for all values of minimum support, Max-FGP generated less maximal frequent sets than the closed frequent set approach. Among the 90, 41, and 26 closed frequent sets, 23, 10, and 4 contain dependences, respectively, corresponding to an overall average of 20%.

In Figure 5.15(right) we observe that because of the large number of predicates (411) and the elimination of a low number of dependences, the computational time increases significantly in relation to the previous experiments with less predicates. However, both Apriori-KC and Max-FGP are respectively still more efficient than either Apriori or the closed frequent set approach. Indeed, in this experiment Max-FGP was more efficient than Apriori, what has not occurred in previous experiments when mining data at the feature type granularity level.

5.3 Evaluating Apriori-KC and Max-FGP for Hierarchical Dependence Elimination

In this section we evaluate Apriori-KC and Max-FGP to compare the number of frequent sets generated for datasets 2 and 4, both at the feature type granularity level, but the former is at a high granularity level and later has one attribute at a lower granularity level. Dataset 2 has the feature type street at granularity level 1, and dataset 4 has this feature type at granularity level 2 (street, avenue, and square). In this experiment we evaluate the dependence elimination steps 1 and 2 of Apriori-KC, presented in chapter 4.

The result of the experiment performed with Apriori and the closed frequent sets over both datasets is shown in Figure 5.16 (left), where the number of frequent sets generated by Apriori for dataset 2 (at granularity 1) is 1,321. This number was increased to 2,719 (granularity 2) with dataset 4. Apriori generated around 50% more frequent

sets considering one single feature type at granularity 2, for any value of minimum support. The closed frequent sets have not increased as much at granularity 2.

Figure 5.16 (right) shows the frequent sets and maximal frequent sets generated by Apriori-KC and Max-FGP for datasets 2 and 4 at different granularities. At granularity level 1, where only 2 pairs of dependences were eliminated, Apriori-KC reduced the number of frequent sets generated by Apriori in around 30% for any value of minimum support. However, when mining frequent sets from dataset 4 with one attribute at granularity 2, the reduction reaches more than 40%. This occurs because instead of 2 pairs with dependences, 4 pairs were eliminated (1 single dependence and 3 hierarchical dependences).

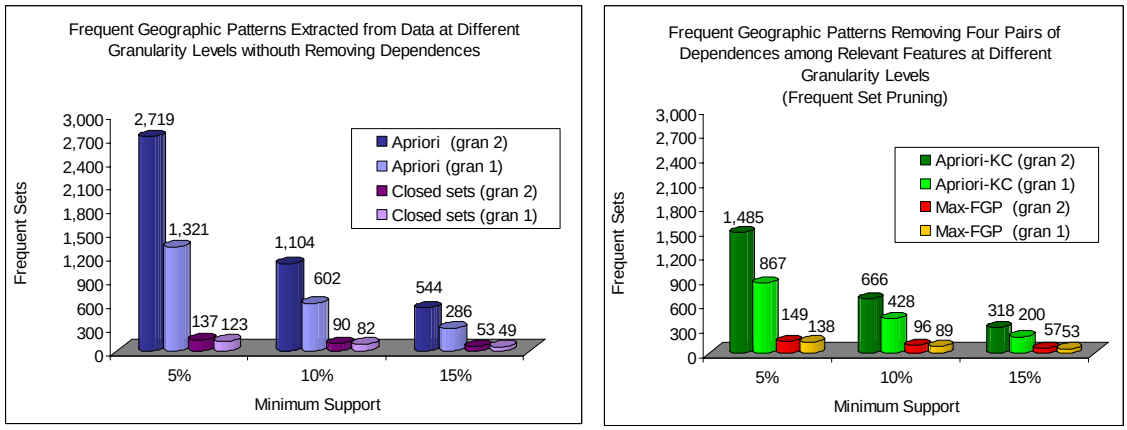


Figure 5.16: Frequent sets generated from datasets 2 and 4 at different granularity levels: (left) frequent sets without removing dependences and (right) frequent sets without dependences

In Figure 5.16 (right) we can observe the significant reduction of frequent sets when redundant sets are eliminated by Max-FGP. For minimum support 5% and granularity 2, for instance, Max-FGP reduced the number of frequent sets in relation to Apriori-KC in 90%, and in 95% in relation to Apriori (in Figure 5.16 left). At lower granularity levels (gran 2) the number of frequent sets having dependences increases, as well as the number of redundant frequent sets. Therefore we can see that Max-FGP is still more effective when mining data at lower granularity levels.

In Figure 5.17 (left) we can better observe the different number of frequent sets generated by Apriori and Apriori-KC for data at different granularities. In Figure 5.17 (right) we can observe that Aprioi-KC is more efficient then Apriori for data at any granularity. However, at granularity 2 the gain of time is more significant in relation to Apriori (reduces from around 115s to 55s). This occurs because at granularity 2 there are more predicates, and by consequence more dependences are eliminated.

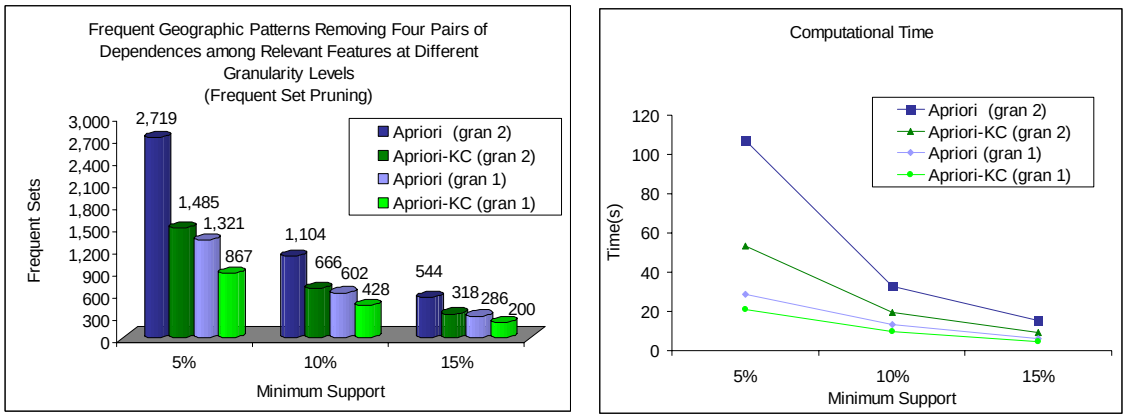


Figure 5.17: (left) Frequent sets removing single geographic dependences and hierarchical dependences from data at different granularity levels and (right) computational time

As can be observed in Figure 5.17 (left), for *minsup* 5% and granularity 2, Apriori-KC eliminates 1,234 frequent sets (45%), while at granularity 1 for same *minsup* only 454 frequent sets are eliminated (35%). At lower granularity levels Apriori-KC is more effective and efficient for dependence elimination.

Figure 5.18 shows the computational time to generate frequent sets with Apriori and Apriori-KC, closed frequent sets, and maximal frequent sets for data at different granularities. Observe that Apriori-KC and Max-FGP are more efficient than Apriori and the closed frequent set approach, respectively. The closed frequent sets approach apart from not removing dependences, takes 60% more time to compute closed frequent sets than Max-FGP takes to generate maximal frequent sets. Indeed, for lower *minsup* the time gained by our methods is more significant.

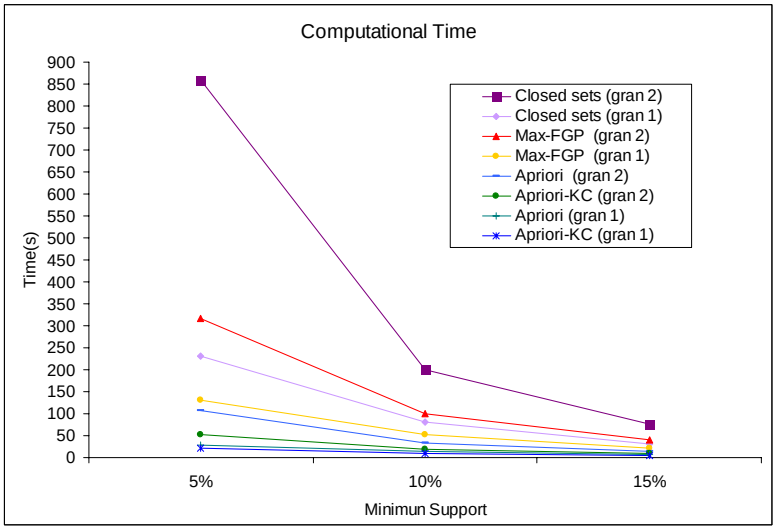


Figure 5.18: Computational time to generate frequent geographic patterns from data at different granularity levels

5.4 Evaluating Apriori-KC and Max-FGP for Predicates with Same Feature Types and Different Relationships

In this section two experiments will be presented. The first was performed over dataset 5 and the second over dataset 6. In both experiments we evaluate the step 3 implemented in Apriori-KC. We will also evaluate the number of frequent sets generated by Max-FGP when predicates with same feature types and different spatial relationships are eliminated by Apriori-KC.

5.4.1 Experiment with Dataset 5

Topological relationships generate many frequent sets with the same feature type and different relationships (e.g. *contains(River), touches(River)*). As a consequence, when such a feature type has a dependence with any other spatial feature type, the number of well known dependences increases. For instance, a dependence between gas station and streets can generate frequent sets such as {*contains(GasStation), touches(Street)*}, {*contains(GasStation), crosses(Street)*}, {*contains(GasStation), contains(Street)*} when different topological relationships are considered.

Figure 5.19 shows the result of an experiment performed over dataset 5, where one pair of dependences produced 4 combinations with different topological relationships. Besides the 4 pairs with a dependence, 9 pairs had same feature types with different relationships. So a total of 13 pairs with either dependences or same feature types was eliminated.

As can be observed in Figure 5.19 (left), the elimination of 13 combinations by Apriori-KC reduced the number of frequent sets generated by Apriori in around 60% for different values of minimum support. Max-FGP reduced this numbers much further, and more than the closed frequent sets. Among the closed frequent sets, more than 60% have at least one dependence, independently of minimum support. For minimum support 5%, for example, among the 159 closed frequent sets, 103 contain either the dependences or predicates with same feature type. For minimum support 10% and 15%, the respective 118 and 101 closed frequent sets contain 72 and 61 dependences.

The computational time to eliminate dependences and frequent sets with same feature types is still reduced with our methods, as shown in Figure 5.19 (right).

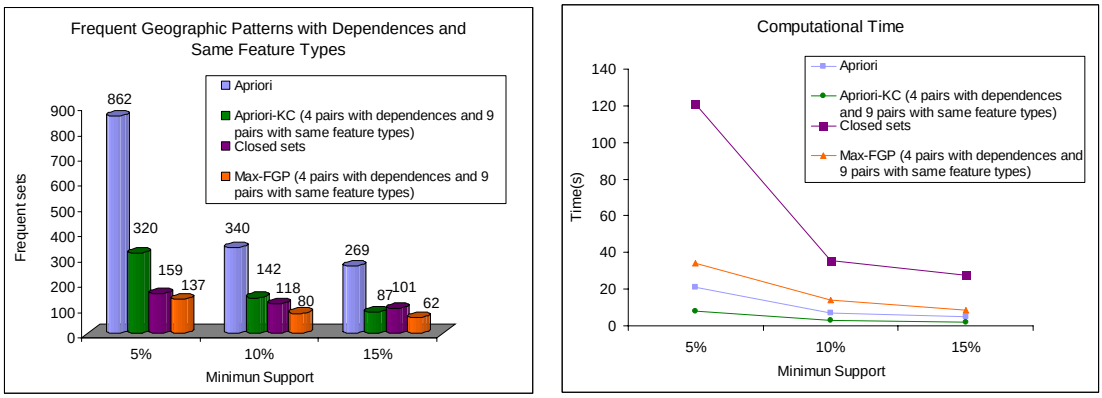


Figure 5.19: (left) Frequent sets and (right) computational time to eliminate pairs with dependences and pairs with same feature types and different topological relationships

Figure 5.20 shows this experiment where the four pairs with a dependence and all 13 pairs with both dependences and same feature type were eliminated separately. As can

be observed, not only dependences generate a large number of frequent sets. For minimum support 5% and 10% there are more frequent sets with same feature type than with dependences.

In order to evaluate the replication of geographic dependences and combinations of same feature types when mining frequent geographic patterns considering topological relationships, a second experiment was performed, with dataset 6, extracted from database 2. This experiment is presented in the following section.

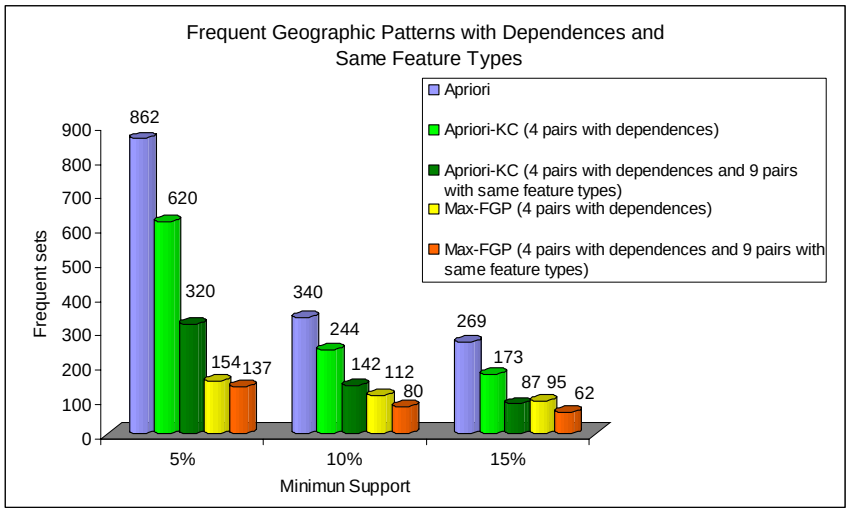


Figure 5.20: Frequent sets with dependences and pairs with same feature types and different topological relationships mined from dataset 5

5.4.2 Experiment with Dataset 6

This experiment was performed over dataset 6 and its result is shown in Figure 5.21. For any value of minimum support, Apriori-KC generated less frequent sets than the closed frequent set approach. In this experiment, the closed frequent sets contain less dependences than in the previous experiment performed over dataset 5, since the total number of frequent sets is much smaller. The respective number of closed frequent sets 34, 17, and 9 generated for *minsup* 5%, 10%, and 15%, contain respectively 13, 8, and 2 dependences. These values still represent 38 % for lower minimum support 5%, 47% for minimum support 10% and 22% for minimum support 15%.

It is important to consider that for any geographic database preprocessed considering topological relationships, the frequent set reduction with Apriori-KC and Max-FGP is data dependent. For example, a pair of predicates with the same feature type such as {*contains(Street),crosses(Street)*} in a *city* will have much higher support than a pair of predicates such as {*contains(River), crosses(River)*}, since normally cities have much more streets than rivers. The same holds for dependences. A dependence between bridge and river has much lower support than a dependence between gas stations and streets, in most real databases.

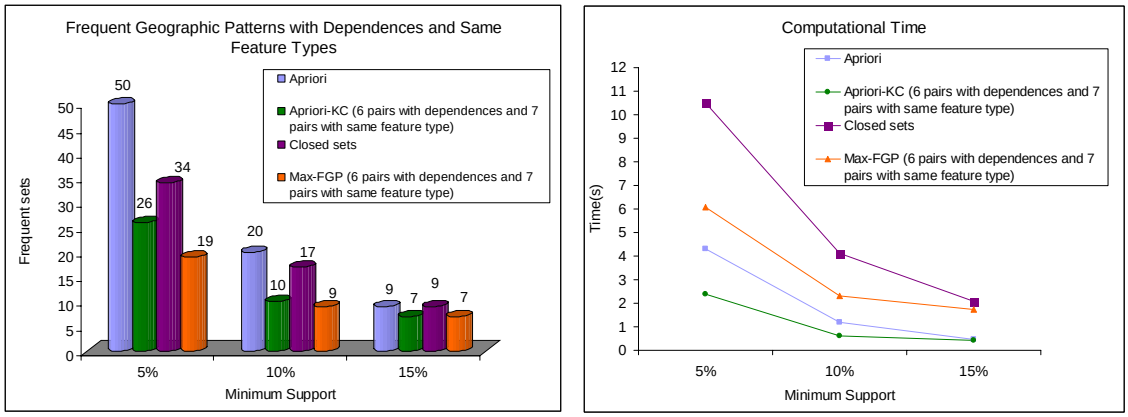


Figure 5.21: (left) Frequent sets and (right) computational time to eliminate pairs with dependences and pairs with same feature types and different topological relationships from dataset 6

5.5 Evaluating Apriori-KC and Max-FGP for Spatial Feature Types with same Parent

In the previous sections we evaluated steps 1 to 3 implemented in Apriori-KC. In this section we describe two experiments performed over dataset 11 and 12, extracted from database 2. In this experiment we evaluate step 4, implemented in Aprori-KC, which eliminates pairs of predicates that contain feature types that are "brothers" in a concept hierarchy (e.g. *contains(River), contains(Lake)*). In these experiments we considered minimum support 3% and 5%, otherwise the number of generated frequent sets would be very low.

Figure 5.22 (left) shows the number of frequent sets and Figure 5.22 (right) shows the computational time. Apriori-KC reduces the number of frequent sets in around 35% in relation to Apriori. In this experiment Max-FGP generates less maximal frequent sets than closed frequent sets. This reduction reaches around 30% for different values of minimum support.

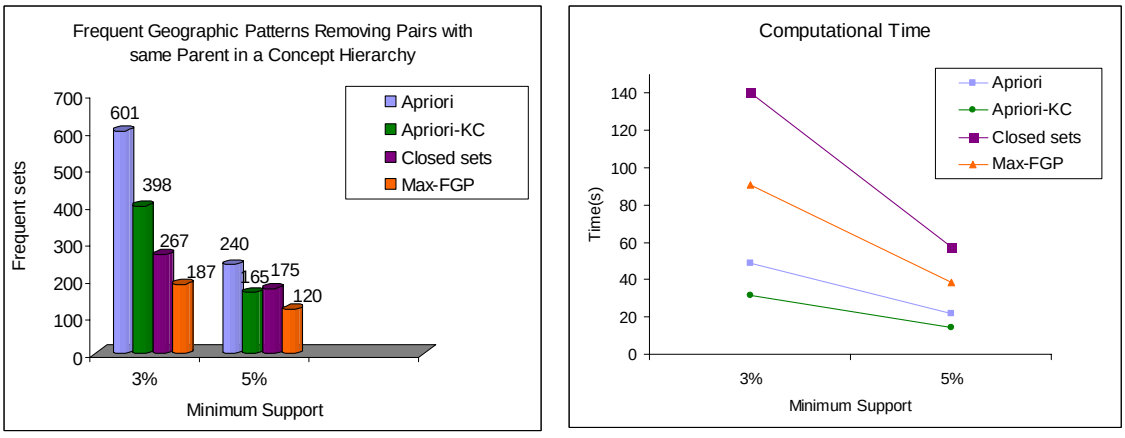


Figure 5.22: (left) Frequent sets and (right) computational time to eliminate pairs of predicates with feature types that have same parent in concept hierarchies mining dataset 11

In Figure 22 (right) we observe that similarly to the previous experiments, both Apriori-KC and Max-FGP remain more efficient then Apriori and Closed frequent set approach, respectively.

Figure 23 shows the experiment performed over dataset 12, and the percentage reduction of both frequent sets and computational. In this experiment, both Max-FGP generated 40% less frequent sets than the closed frequent set approach. Similarly to previous experiments both Apriori-KC and Max-FGP are more efficient than Apriori and the closed frequent set approach, respectively.

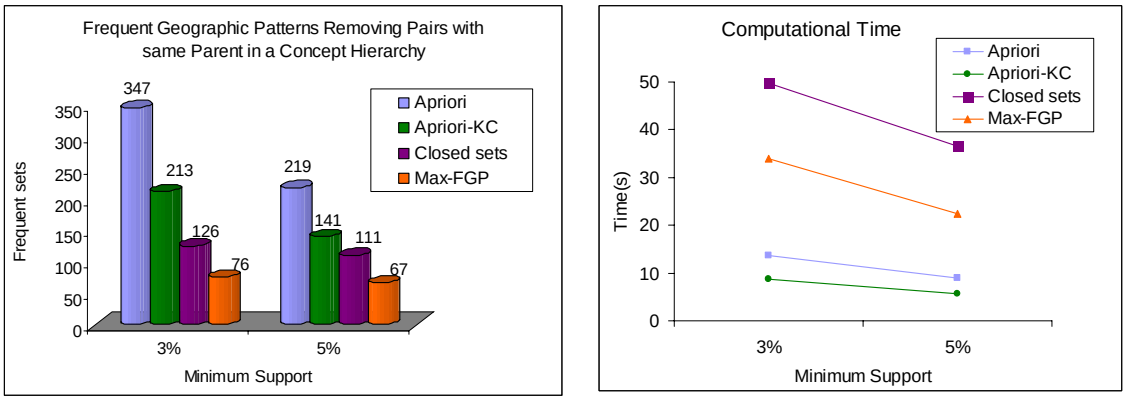


Figure 5.23: (left) Frequent sets and (right) computational time to eliminate pairs of predicates with feature types that have same parent in concept hierarchies mining dataset 12

The experimental results with real geographic databases presented in this chapter confirm that the different methods proposed in this thesis for mining frequent geographic patterns are more efficient and effective than Apriori and algorithms that generate closed frequent sets. We evaluated these methods considering many different datasets, with different numbers of predicates, different values of minimum support, and removing different types of dependences.

All different experiments showed the significance of the proposed methods for removing well known dependences and redundant frequent sets. Considering all these experiments we recommend to use Max-FGP for mining frequent geographic patterns from geographic databases.

In the next chapter we present an overview of the data preprocessing method which has been implemented in Weka in order to automate geographic data preprocessing and the elimination of geographic dependences between the target feature type and relevant feature types.

6 WEKA-GDPM: GEOGRAPHIC DATA PREPROCESSING PROTOTYPE

In this chapter we present Weka-GDPM (BOGORNY, 2006d) a geographic data preprocessing module (GDPM), which integrates the classical data mining toolkit Weka (WITTEN 2005) to GDMS developed under OGC specifications (OPEN GIS CONSORTIUM, 1999b). Weka is a free and open source classical data mining toolkit which provides friendly graphical user interfaces to perform the whole KDD process. Weka implements a variety of data mining algorithms, and has been largely used for mining non-spatial databases. The main objective of Weka-GDPM is to automate geographic data preprocessing in order to reduce time and effort from the data mining user.

The prototype presented in this chapter covers the preprocessing method proposed in this thesis. The implementation of Apriori-KC and Max-FGP into Weka is ongoing work being developed by an undergraduate student.

Weka is developed in Java and has a data preprocessing module named weka.Explorer to preprocess non-spatial databases. At this interface, which is shown in Figure 6.1, the user can choose to open a web site, an *arff* file, which is the input text file format required by Weka, or a database. The button *OpenDB* calls the window shown in Figure 6.2 (left) in order to provide the information necessary to connect to a database.

To support GDPM the button *GeographicData* was added to the database connection interface, as can be observed in Figure 6.2 (left). With the information provided to this interface Weka connects to any OGC database through JDBC, and calls the GDPM interface already connected to the database provided by the *url*. Figure 6.2 (right) shows an overview of GDPM interface.

In order to understand how GDPM works, let us first understand how geographic data are manipulated. The GDBMS or GIS implement specific operations and functions to manipulate and visualize spatial data. The OGC is an organization dedicated to develop standards for spatial operations and spatial data integration, aiming to provide interoperability for GIS. Among many specifications established by the OGC, there are two of fundamental importance and which are used in Weka-GDPM: operations to compute *spatial relationships* and the *database schema* metadata.

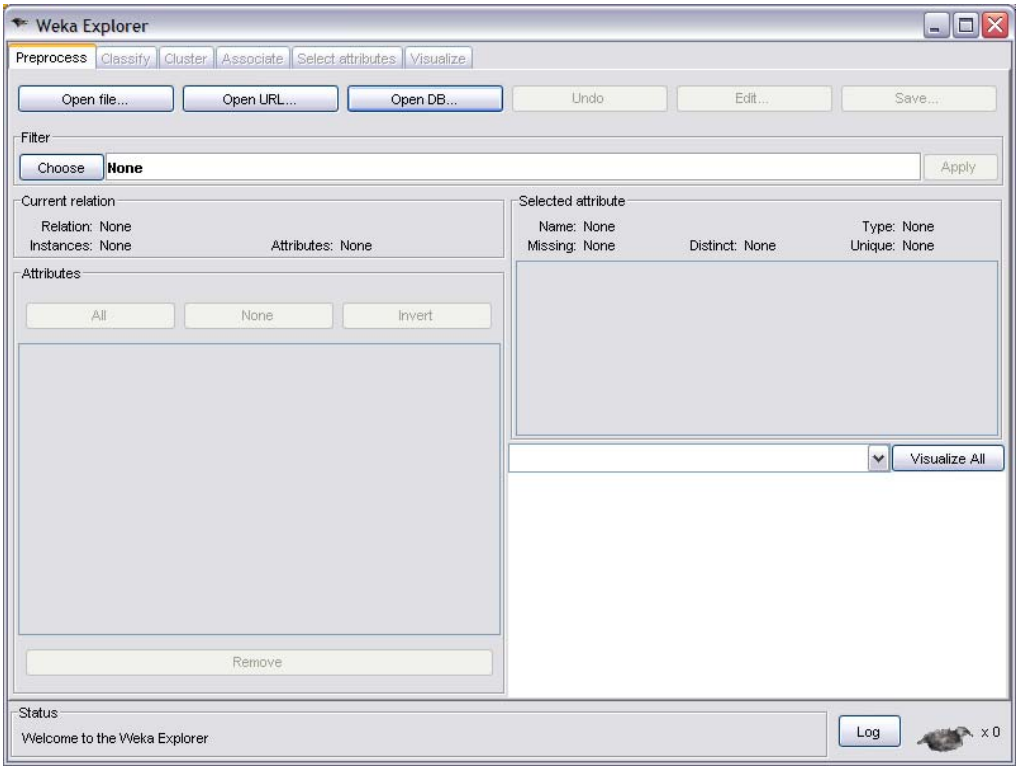


Figure 6.1: Weka Explorer GUI

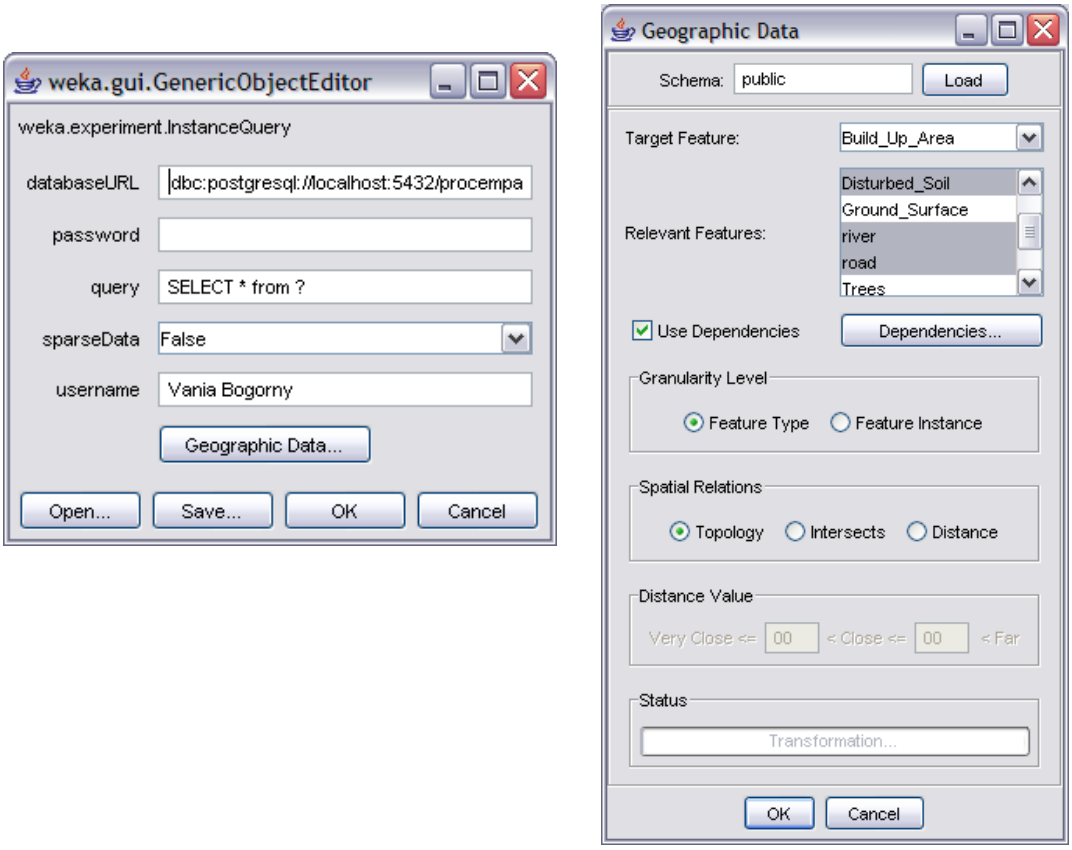


Figure 6.2: (left) Interface to access geographic databases and (right) Geographic data preprocessing interface - GDPM

The OGC defines a standard set of operations to compute spatial relationships for SQL which are implemented by most GDBMS. The database schema metadata are stored in a database relation named *geometry_columns*, which is generated during the creation of the database and is automatically instantiated when geographic data are loaded the first time. This relation stores all database characteristics, as shown in an example in Figure 6.3(d). From this relation it is possible to obtain the database schema name, all geographic relation names, the name of the geometry column, and its type (e.g. point, line). This relation consists of a row for each spatial feature type in the geographic database, i.e., each relation that has geometric attributes. The metadata stored in *geometry_columns* are used by GDPM to automatically preprocess GDB.

Street

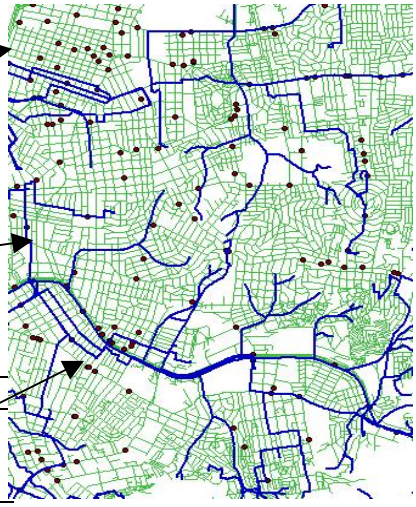
Gid	Name	Shape
1	Lajeado	Multiline [$(x_1,y_1),(x_2,y_2),..$]
2	Itaqui	Multiline [$(x_1,y_1),(x_2,y_2),..$]

(b) WaterResource

Gid	Name	Shape
1	Jacui	Multiline [$(x_1,y_1),(x_2,y_2),..$]
2	Guaiba	Multiline [$(x_1,y_1),(x_2,y_2),..$]
3	Uruguai	Multiline [$(x_1,y_1),(x_2,y_2),..$]

(c) GasStation

Gid	Name	VolDiesel	VolGas	Shape
1	BR	20000	85000	Point[(x_1,y_1)]
2	IPF	30000	95000	Point[(x_1,y_1)]
3	Elf	25000	120000	Point[(x_1,y_1)]



(d) GEOMETRY_COLUMNS

F_table_schema	**F_table_name**	**F_geometry_column**	**Type**	**SRID**
Public	Street	Shape	Multiline	-1
Public	WaterResource	Shape	Multiline	-1
Public	GasStation	Shape	Point	-1

Figure 6.3: Geographic data storage structure in OGC based GIS

As Weka cannot handle geometric attributes, geographic data need to be selected from the database and their spatial relationships computed with operations provided by the GDMS, and transformed into the single table input format. Figure 6.4. shows an example of a spatial dataset preprocessed and transformed into an *arff* file to be mined with Weka.

```
@relation 'geographic_data'

@attribute f_code_des
{null,Cropland,Grassland,Land_Subject_to_Inundation,Rice_Field,Scrub_Brush}
@attribute overlaps_Road {yes}
@attribute crosses_River {yes}
@attribute contains_River {yes}
@attribute crosses_Tunel {yes}
@attribute overlaps_River {yes}
@attribute crosses_Road {yes}
@attribute within_Bridge {yes}
@attribute within_River {yes}
@attribute touches_River {yes}
@attribute contains_Bridge {yes}
@attribute touches_Tunel {yes}
@attribute contains_Road {yes}
@attribute contains_Tunel {yes}
@attribute overlaps_Bridge {yes}
@attribute touches_Road {yes}
@attribute overlaps_Tunel {yes}
@attribute crosses_Bridge {yes}
@attribute within_Tunel {yes}
@attribute within_Road {yes}
@attribute touches_Bridge {yes}

@data

Land_Subject_to_Inundation,yes,?,yes,?,yes,?,?,?,?,?,?,yes,yes,?,?,yes,?,?,?,?
Land_Subject_to_Inundation,yes,?,yes,?,yes,?,?,?,?,yes,?,yes,yes,?,?,yes,?,?,?,?
Land_Subject_to_Inundation,yes,?,yes,?,yes,?,?,?,?,yes,?,yes,yes,?,?,yes,?,?,?,?
Land_Subject_to_Inundation,yes,?,yes,?,yes,?,?,?,?,yes,?,yes,yes,?,?,yes,?,?,?,?
Land_Subject_to_Inundation,yes,?,yes,?,yes,?,?,?,?,yes,?,yes,yes,?,?,yes,?,?,?,?
Land_Subject_to_Inundation,yes,?,yes,?,yes,?,?,?,?,?,?,?,yes,?,?,yes,?,?,?,?
Cropland,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,yes,?,?,?,?
Cropland,yes,?,?,?,?,yes,?,?,?,?,?,yes,?,?,?,?,?,yes,?,?,?,?
Cropland,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?
Cropland,yes,?,?,?,yes,?,?,?,?,?,?,?,?,?,yes,?,?,?,?
Rice_Field,yes,?,yes,?,yes,?,?,?,?,yes,?,?,?,?,?,yes,?,?,yes,?
```

Figure 6.4: Weka input format (*arff* file)

To generate a dataset as shown in Figure 6.4 (*arff* file), the user provides the geographic database schema into the GDPM interface shown in Figure 6.2 (right). The load button brings to the *Target Feature* and *Relevant Features* interface all geographic database relations (from the relation geometry_columns) that belong to the informed schema. This allows the user to choose only the spatial feature types of interest, and not the whole geographic database.

GDPM automatically generates data at two granularity levels for distance, topological, and high level topological relationships (intersects), without any concept hierarchy, as have been explained in Chapter 4.

The user can choose the type of spatial relationships and the granularity level in which data should be generated. For distance relationships one or two distance parameters must be provided.

Weka-GDPM supports the definition of geographic dependences, as can be observed in Figure 6.2 (right). The check box *Use Dependences* enables the button to specify the pairs of geographic feature types with dependences. When this button is clicked the interface shown in Figure 6.5 is called. In this interface appear two combos that contain all database spatial feature types with an instance in the relation *geometry_columns.* At these combos the user can choose the pairs that contain dependences, such as tunnel and road, as shown in the example in Figure 6.5. After selecting the pairs and pressing the *Add* button, a new dependence is created with the respective selected pair. The pair will

than appear in the list of dependences in the same window. To remove any dependence the user must select the pair and click on the button *Remove.*

After dependences have been defined, the button *OK* must be pressed in order to store the dependences into a text file and a database relation named *knowledgeConstrains.* The database relation is used for the dependence elimination in data preprocessing and the text file will be used for dependence elimination during the frequent set generation, which is performed in another step.

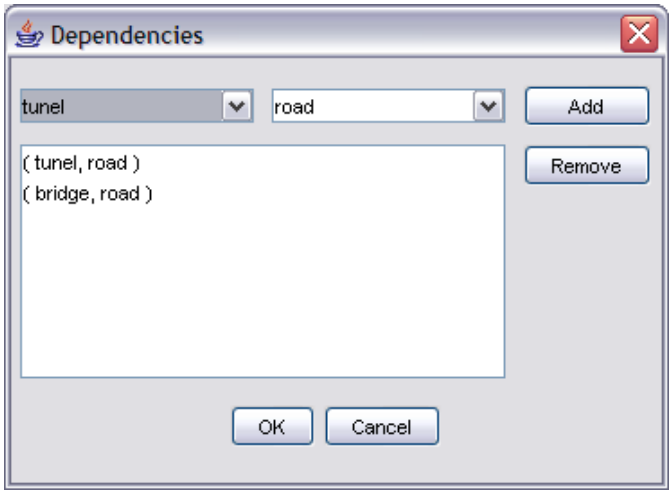


Figure 6.5: Geographic dependence definition interface

Since dependences have been specified and all parameters have been provided to GDPM (in Figure 6.2 right) interface, the first step performed by GDPM is the dependence elimination. Before performing the spatial join step between the target feature type and each relevant feature type, GDPM searches for a dependence in the relation *knowledgeConstraints.* When a dependence is found, the relevant feature type is removed from the set and the next relevant feature type is tested. After all dependences have been removed the spatial join and transformation modules start.

6.1 The Spatial Join Process

In this section we describe the steps to compute the spatial joins in Weka-GDPM. Up to now it supports topological, distance, and general topological relationships (intersects and non-intersects).

6.1.1 Topological Relationships

Topological relationships are mutually exclusive such that only one topological relationship holds between two spatial feature *instances* (e.g. Porto Alegre city and Canoas city). At the feature instance granularity level every instance of the target feature type may have only one topological relationship with an instance of a relevant feature type. Table 6.1 (left) illustrates an example of the spatial join computation where city 1 has the relationship *contains* with River_1, *crosses* with River_2, and *contains* with Slum_1. At the feature instance granularity level, when transforming the spatial join output (Table 6.1 left) into the Weka input format (Table 6.1 right), the relevant feature type name with the respective instance is transformed in an attribute name. The value of this attribute will be the respective topological relationship. For the relevant feature instances that have no relationship with an instance of the target feature (e.g.

River_3 and city_1, River_1 and city_3), the attribute value is filled with "?", which is the symbol used by Weka to represent the absence of an attribute.

Table 6.1: *Feature Instance* Granularity Level for topological relationships

TargetF_id (city)	RelevantF Instance	Relationship
1	River_1	Contains
1	River_2	Crosses
2	River_3	Contains
2	River_4	Crosses
3	River_2	Crosses
1	Slum_1	Contains
2	Slum_2	Contains

TargetF_id (city)	River_1	River_2	River_3	River_4	Slum_1	…
1	Contains	Crosses	?	?	Contains	
2	?	?	Contains	Crosses	Contains	
3	?	Crosses	?	?	?	

At the feature type granularity level, as shown in Table 6.2 (left), the relevant feature instance is not stored in the spatial join output. For example, city 1 has two topological relationships with the relevant feature type River, *contains* and *crosses* (with different rivers). At this granularity level, to preserve the type (semantics) of the topological relationship, we need to create a different attribute name (e.g. *contains_river*, *crosses_river*) for every relevant feature type with a different topological relationship with the target feature. This is necessary because Weka cannot handle duplicated attribute names (river) with two different values (contains and crosses). Indeed, it is difficult to specify dominance between topological relationships. As this problem has not been addressed in the literature so far, we propose to preserve the relationship type by concatenating it to the feature type name, while the attribute value receives the string "yes" when the relationship holds and " ?" if there is no topological relationship, as shown in Table 6.2(right).

Table 6.2: *Feature Type* Granularity Level for topological relationships

TargetF_id	RelevantF Type	Relationship
1	River	Contains
1	River	Crosses
2	River	Contains
2	River	Crosses
3	River	Crosses
1	Slum	Contains
2	Slum	Contains

TargetF_id (city)	Contains_River	Crosses_River	Contains_Slum	…
1	Yes	Yes	Yes	
2	Yes	Yes	Yes	
3	?	Yes	?	

Considering that at lower granularity levels predicates with topological relationships have very low support, as explained in Chapter 3, this prototype implements high level topological relationships: *intersects* and *non-intersects*.

Table 6.3 (left) shows two datasets at with the feature instance (top) and feature type granularity level (bottom) for the intersects relationship, and Table 6.3 (right) shows the respective transformation for the Weka input format.

Table 6.3: (left) Feature instance and feature type granularity for high level topological relationships (intersects and non-intersects) and (right) Weka input format

TargetF_id (city)	RelevantF Instance	Relationship
1	River_1	Intersects
1	River_2	Intersects
2	River_3	Intersects
2	River_4	Intersects
3	River_2	Intersects
1	Slum_1	Intersects
2	Slum_2	Intersects

TargetF_id (city)	River_1	River_2	River_3	River_4	Slum_1	…
1	intersects	intersects	?	?	intersects	…
2	?	?	intersects	intersects	intersects	…
3	?	intersects	?	?	?	…

TargetF_id (city)	RelevantF Type	Relationship
1	River	Intersects
2	River	Intersects
3	River	Intersects
1	Slum	Intersects
2	Slum	Intersects

TargetF_id (city)	Intersects_River	Intersects_Slum
1	Yes	Yes
2	Yes	Yes
3	yes	?

6.1.2 Distance Relationships

Distance relationships are computed according to the distance parameters provided by the user. If only one distance parameter is provided, neighborhoods are considered *very close* if their distance from the target feature is less or equal to *dist1*. When two distance measures are informed, than neighborhoods are considered *very close* if their distance from the target feature is less or equal to *dist1*, and *close* if their distance is between *dist1* and *dist2*, as shown in the example in Table 6.4 (left) for the feature instance and feature type granularity level.

Table 6.4: (left) Feature instance and feature type granularity for high level for distance relationships and (right) Weka input format

TargetF_id (city)	RelevantF Instance	Relationship
1	River_1	VeryClose
1	River_2	Close
2	River_3	Close
2	River_4	Close
3	River_2	VeryClose
1	Slum_1	Close
2	Slum_2	VeryClose

TargetF_id	River_1	River_2	River_3	River_4	…
1	VeryClose	Close	?	?	…
2	?	?	Close	Close	…
3	?	VeryClose	?	?	…

TargetF_id (city)	RelevantF Type	Relationship
1	River	VeryClose
1	River	Close
2	River	Close
3	River	VeryClose
1	Slum	Close
2	Slum	VeryClose

TargetF_id	VeryClose_River	Close_River	Close_Slum	…
1	Yes	Yes	Yes	…
2	?	Yes	?	…
3	Yes	?	?	…

The relationship *far* is not considered because experiments showed the generation of an enormous amount of non-interesting patterns. For example, the city 1 is very close to river 1 and close to river 2, but far from all other rivers. For distance relationships we

can say that *close* is dominant over *far*, because all spatial objects that are *not close*, will be far. Just in case someone would like to consider things that are *far*, the value of the distance metric *dist2* can be increased in order to cover the relationship far.

The transformation process to convert the spatial join output (Table 6.4 left) to the Weka input format (Table 6.4 right) is the same as for topological relationships.

In the previous sections we described the details to be considered for preprocessing geographic databases. In the following sections we describe implementation details.

6.2 Spatial Join Implementation

The spatial join step is performed among the target feature type and all selected relevant feature types that have no dependence with the target feature. The selected spatial relationships are computed in the geographic database, with the spatial operations implemented by the GDBMS, which follows the OGC specifications.

The spatial join output is stored in a database temporary relation called *<target_feature_type_name>_temp*. This relation contains the attributes *<gid_target_feature_type>*, *<relevantF_name>*, and *<relationship>*, as described in the previous sections.

If there are no spatial relationships between the given target feature type and relevant feature types, the message shown in Figure 6.6 is presented to the user. Otherwise, spatial relationships are computed and the transformation method is called, as described in the following section.

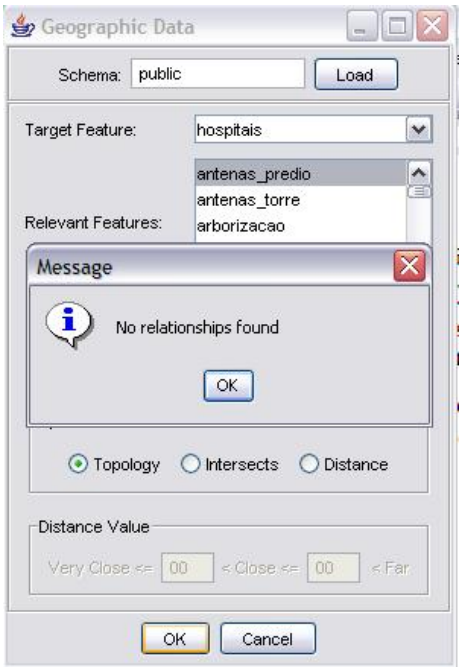


Figure 6.6: Information message when no relationships are found

6.3 Transformation Implementation

The transformation module is performed in memory and generates an *arff* file named *geographic_data.arff*. Since the *arff* file may contain a large number of attributes, mainly when mining data at the feature instance granularity level, the transformation

result cannot be stored in a database relation. Most GDMS allow relations a limited number of columns.

Performing the transformation step in memory makes the process much faster. Going into detail about the implementation, the transformation step implies in obtaining the non-spatial attributes of the target feature type (obtained from the database relation target_feature_type), and the spatial predicates, which are all different predicates generated by the spatial join step.

The transformation step reads the database relation <*target_feature_type_name*>*_temp* in order to get the spatial attributes (spatial predicates) that will be part of the header of the *arff* file (see *@attribute* in Figure 6.4). At the feature instance granularity level, every different value of the column *RelationF* in table <*target_feature_type_name*>*_temp* will result in a new attribute in the *arff* file.

The possible values of these attributes are stored in the column *relationships* in the relation <*target_feature_type_name*>*_temp*. For topological relationships, possible values of the *relevantF* column are: "CONTAINS", "TOUCHES", "WITHIN", "OVERLAPS", "CROSSES", and “EQUALS”. For the high level topological relationships the possible value is "INTERSECTS". For distance relationships the possible attribute values are "VERY_CLOSE" and "CLOSE".

At the feature type granularity level, where relevant feature types may have different topological relationships with the target feature type, the attribute names in the *arff* file will be the different values of the column *relevantF* concatenated with the value of the column *relationship* stored in the temporary table <*target_feature_type_name*>*_temp*. For the feature type granularity level, the values of the attributes in the matrix will be “YES” when the relationship holds in the respective value of column *relationship* in the table <*target_feature_type_name*>*_temp*, and “?” otherwise.

After discovering the attribute names that will be the header of the *arff* file, a matrix is created. There will be exactly one row in the matrix for every different instance of the target feature type, i.e., for every different *gid* (geographic identifier). The columns of the matrix will be the attribute names discovered in the previous step. The matrix is initialized with "?" strings, to fill the attribute values if there is no relationship between an instance of the target feature type and a relevant feature (instance or type).

To fill the matrix with the respective relationships, the temporary relation is scanned again. Then the matrix is updated with all records that have a relationship (different from "?"). Having the matrix matched the temporary table, finally the *arff* file will be created.

To create the *arff* file GDPM starts searching for the non-spatial attributes of the target feature type in the database, saving their names into a vector, named *columns*. The header of the file is then created writing each column in *columns* as an attribute name. The type to be created in the *arff* file depends on the respective attribute type in the database. For the type *string* we need to select all distinct values that the attribute may have in order to create the header of a string attribute with all its possible values. For this purpose a SQL query is performed over the string attributes of the target feature type. After that, GDPM enumerates these values in the *arff* file, additionally including the value *null* to catch the null values in the database.

Since non-spatial attributes have already been created, GDPM writes the spatial predicates (which are columns in the matrix generated above) as attribute names in the

arff file. In the sequence, the different values that each attribute may have are written in an enumeration of values.

To fill the body of the file (see *@data* in Figure 6.4), GDPM starts executing a SQL query that returns all instances of the target feature type with its respective non-spatial attribute values. For every instance, the respective non-spatial attribute values returned by the query as well as the spatial predicates (in the matrix) are added to the file.

In order to avoid errors in the *arff* file all characters that are not among the sets [A-Z,a-z,0-9] in the database, are replaced in both attributes and values to the character '_'. Since the *arff* file has been created, a message is presented to the user, as shown in Figure 6.7.

Figure 6.7: Finishing message when the *arff* file is successfully created

7 CONCLUSIONS AND FUTURE WORKS

In this thesis we addressed the problem of mining non-obvious spatial association rules from geographic databases. We showed that a large amount of patterns extracted from geographic databases are well known a priori, and do only hind the discovery process.

We showed that geographic database schemas and geo-ontologies are rich knowledge repositories that have not been used in spatial association rule mining so far. The case studies with different real geographic database schemas showed that a very large number of associations in geographic databases is one-one and one-many, and produce a large number of well known patterns if considered in frequent geographic pattern mining. Existing data mining algorithms do only consider the data by themselves while the schema has not been considered. By consequence, the same mandatory associations represented in database schemas are extracted as *novel patterns* by rule mining algorithms.

While dozens of algorithms for either transactional or geographic databases propose *syntactic constrains* to prune the number of frequent sets and association rules, we proposed to use *semantic constraints*. Semantic constraints are explicitly represented in geographic database schemas and geo-ontologies by relationships with cardinalities one-one and one-many.

At least three main steps are required for mining SAR from geographic databases:

a) extract spatial predicates;

b) generate frequent sets or frequent geographic patterns, and;

c) extract spatial association rules.

In general words, this thesis presented a novel contribution to improve the three main steps for mining SAR. We presented two main contributions to the first step (a). First, we automated the geographic data preprocessing tasks developing for the Weka data mining toolkit a geographic data preprocessing module (GDPM). We reduced a problem that according to (ADRIAANS, 1994) for transactional databases consumes between 60 and 80% of the effort expended in the whole KDD process. For geographic databases, this time increases because of the complexity of geographic data.

The second contribution is the elimination of all well known geographic dependences that generate well known association rules. Our solution of pruning the input space by removing well known geographic dependences between the target

feature type and the relevant feature types has some general advantages: no spatial relationships will be computed for the relevant feature types eliminated in data preprocessing; the number of frequent sets and association rules generated by any rule mining algorithm will be significantly reduced; the computational time to generate frequent sets and rules decreases; and no well known patterns between the target feature type and the eliminated relevant feature types will be generated.

For the second step (the frequent set generation) we also presented two main contributions. We proposed two efficient methods that automatically improved the third (the association rule generation): Apriori-KC and Max-FGP. With Apriori-KC we eliminate from the candidate sets all pairs of geographic objects that have well known dependences. This method reduces the number of frequent sets independently of minimum support and warrants that dependences among relevant feature types are completely eliminated. The main advantage of this dependence elimination method is its simplicity, since in one single step all dependences among the relevant feature types are eliminated.

Not only exact dependences are eliminated with Apriori-KC, but hierarchical dependences are eliminated as well. This avoids the generation of rules such as *contains(Island)→crosses(River)* or *contains(Island)→contains(Lake)*.

Apriori-KC also eliminates candidates with two predicates that contain the same feature type and different topological relationships. This avoids the generation of meaningless rules such as *contains(Water)→touches(Water)*, which do not add any novel knowledge to the discovery. Pairs of predicates hat have the same parent in a concept hierarchy are also eliminated. This avoids the generation of rules such as *contains(Lake)→contains(Water)*.

The third step was addressed *indirectly*. The elimination of dependences in data preprocessing and frequent set generation automatically reduces the number of association rules, which can be generated by any rule generation algorithm.

With Max-FGP the number of frequent sets generated by Apriori-KC is reduced much further. From the frequent sets without well known dependences generated by Apriori-KC we generate only the maximal non-redundant frequent sets, similarly to the closed frequent set approach proposed for transactional databases. The generation of maximal frequent geographic patterns is a novel technique that as far as we know has not been applied to geographic data. We showed that the closed frequent set approach does not eliminate well known geographic dependences and predicate sets with same feature types. Moreover, if applied over frequent sets after dependences elimination the result quality is sacrificed and a relevant amount of information is lost.

Our methods proposed in this thesis warrant the elimination of all well known geographic dependences in frequent geographic pattern mining. Indeed, they can be applied to any Apriori-like algorithm that prunes frequent patterns and association rules with different constraints (e.g. lift, improvement), since the dependences are eliminated early.

Experiments showed that independent of the number of elements (predicates), one dependence is enough to prune almost half of the total number of frequent sets, and the higher the number of the dependences, the larger is the reduction.

The main contribution is for the data mining user, that will perform geographic data preprocessing very easily, as well as analyze much less obvious patterns. The method is

effective independently of other thresholds, and it warrants that spatial associations that are previously known as non-interesting will not appear among the discovered patterns.

We showed that existing approaches for mining spatial association rules are based on the paradigm of minimum support, minimum confidence and their variations, that they generate a large number of rules and do not warrant the elimination of all well known geographic dependences.

Approaches for mining *spatial* association rules or *co-location* rules have some general drawbacks. The quantitative reasoning approach which extracts frequent patterns and co-location rules from spatial data directly considers only distance relationships and is limited to geographic objects represented by points. For mining real geographic databases this method may generate unrealistic patterns when the centroid is considered. Indeed, the problem of generating well known patterns is not addressed in this approach.

Among qualitative reasoning approaches which preprocess geographic databases in a first step and generate frequent sets and extract association rules in a second step, only a few address the problem of generating well known rules. Those which remove well known patterns do it a posteriori, while our methods remove well known dependences a priori. Indeed these approaches compute all spatial relationships and transform data to first order logic and extract patterns from the whole database. This process is computationally expensive and non-trivial for real problems.

Existing approaches have not addressed the problem of generating meaningless patterns and rules created when either different topological relationships (e.g. *contains(Water)* → *touches(Water)*) are considered or geographic data have same parent in a concept hierarchy (e.g. *contains(Water)*→*crosses(Canal)*). Our experiments showed that the number of patterns generated by this kind of combinations in real geographic databases is large. In this thesis we addressed this problem and solved it very efficiently.

We also showed that the use of different topological relationships when mining geographic data at feature instance granularity level, may not generate patterns, and information might be lost. This problem has not been addressed in the spatial data mining literature.

In summary, we can conclude that when mining spatial association rules with data at high granularity levels, spatial relationships may be considered at lower granularities (e.g. touches, contains). However, when mining data at lower granularities, mainly when the instances of the relevant features are considered, spatial relationships must be extracted at higher granularities (e.g. intersects and non-intersects).

Future Trends

Data mining techniques to extract knowledge from large spatial and non-spatial databases have mainly considered syntactic constraints and the data by itself, without considering semantics. The result is that the same geographic dependences that are well known by GDB designers and are explicitly represented in GDB schemas and geo-ontologies to warrant the consistency of the data, are extracted by data mining algorithms, which should discover only novel and useful patterns. When dealing with geographic data, which are semantically interdependent because of their nature, the

meaning of data needs to be considered, at least to avoid the extraction of well known patterns.

There is an emerging necessity to consider semantic geographic domain knowledge in spatial data mining. The large amount of knowledge explicitly represented in geographic database schemas and spatio-temporal ontologies needs to be incorporated into data mining techniques, since they provide a valuable source of domain knowledge. How to use this knowledge in data mining systems and for which purposes are still open problems. In this thesis we presented an efficient solution, addressing a small piece of these problems. We used prior knowledge in spatial association rule mining to reduce well known patterns, but the use of domain knowledge in different data mining techniques such as clustering, classification, and outlier detection are still open problems. In clustering, for example, the use of semantics could either avoid the separation of geographic objects that have mandatory constraints or organize them into the same cluster without the necessity of computing their relationship. The use of prior knowledge to evaluate the interestingness of patterns extracted with the different techniques still needs to be addressed.

Future ongoing work is the implementation of Apriori-KC and Max-FGP into Weka, as well as the submission of Weka-GDPM to the Weka developers group from the Waikato University in New Zeland.

Future work also includes the investigation of how semantic knowledge can be used in spatio-temporal data mining, more specifically in trajectory databases generated from data collected by mobile devices. How semantic knowledge can be used to improve the process of knowledge discovery in dynamic data which change in both time and space will be addressed in the near future.

REFERENCES

ADRIAANS, P.; ZANTINGE, D. **Data mining**. Harlow, England: Addison Wesley Longman, 1997.

AGRAWAL, R.; IMIELINSKI, T.; SWAMI, A. Mining association rules between sets of items in large databases. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 1993, Washington, D.C. **Proceedings…** New York: ACM Press, 1993. p. 207-216.

AGRAWAL, R.; SRIKANT, R. Fast Algorithms for Mining Association Rules in Large Databases. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES, VLDB, 20., 1994, San Francisco. **Proceedings…** California: Morgan Kaufmann, 1994. p.487 – 499.

APPICE, A. et al. Discovery of Spatial Association Rules in Geo-Referenced Census Data: A Relational Mining Approach. **Intelligent Data Analysis,** [S.l.], v.7, n.6, p. 542-566, 2003.

APPICE, A. et al. Mining and Filtering Multi-level Spatial Association Rules with ARES. In: INTERNATIONAL SYMPOSIUM ON METHODOLOGIES FOR INTELLIGENT SYSTEMS, ISMIS, 15., 2005, New York. **Proceedings…** [S.l.]: Springer, 2005. p.342-353 (Lecture Notes in Computer Science, 3488).

BASTIDE, Y. et al. Mining minimal non-redundant association rules using frequent closed itemsets. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL LOGIC, CL, 1., 2000, London. **Proceedings…** [S.l.]: Springer. 2000. p.972-986. (Lecture Notes in Computer Science, 1861).

BAYARDO JR, R.J.; AGRAWAL, R. Mining the Most Interesting Rules. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, ACM SIGKDD, 5., 1999, San Diego. **Proceedings…**[S.l.]:ACM Press, 1999a. p. 145-154.

BAYARDO JR, J.R.; AGRAWAL, R.; GUNOPULOS, D. Constraint-Based Rule Mining in Large, Dense Databases. In: IEEE ICDE INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 5., 1999, Sydney, Australia. **Proceedings…**[S.l.]: IEEE Computer Society Press, 1999b. p. 188-197.

BIGOLIN, N. M.; MARSALA, C. Fuzzy Spatial OQL for Fuzzy Knowledge Discovery in Databases. In: EUROPEAN CONFERENCE ON PRINCIPLES AND PRACTICE OF KNOWLEDGE DISCOVERY IN DATABASES, PKDD, 3., 1998, Nantes, France. **Proceedings…** [S.l.]: Springer-Verlag, 1998. p.246-254.

BOGORNY, V.; IOCHPE, C. Extending the OpenGIS Model to Support Topological Integrity Constraints. In: BRAZILIAN SYMPOSIUM ON DATABASES, SBBD, 16., 2001, Rio de Janeiro. **Proceedings...** Rio de Janeiro:COPPE/UFRJ, 2001. p. 25-39.

BOGORNY, V.; ENGEL, P. M.; ALVARES, L.O.: A Reuse-Based Spatial Data Preparation Framework for Data Mining. In INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, SEKE, 17., 2005, Taipei, Taiwan. **Proceedings…** [S.l.]: Knowledge Systems Institute, 2005a. p.649-652.

BOGORNY, V.; ENGEL, P. M.; ALVARES, L.O. Towards the Reduction of Spatial Join for Knowledge Discovery in Geographic Databases using Geo-Ontologies and Spatial Integrity Constraints. In: WORKSHOP ON KNOWLEDGE DISCOVERY AND ONTOLOGIES OF THE ECML/PKDD, KDO, 2., 2005, Porto. **Proceedings…** [S.l.:s.n.], 2005b. p.51-58.

BOGORNY, V.; ENGEL, P. M.; ALVARES, L.O. GeoARM – an interoperable framework to improve geographic data preprocessing and spatial association rule mining. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, SEKE, 18., 2006, San Francisco. **Proceedings…** [S.l.]: Knowledge Systems Institute, 2006a. p. 70-84.

BOGORNY, V.; CAMARGO, S.; ENGEL, P. M.; ALVARES, L.O. Towards elimination of well known geographic domain patterns in spatial association rule mining. In: IEEE INTERNATIONAL CONFERENCE ON INTELLIGENT SYSTEMS, IEEE-IS, 3., 2006, London. **Proceedings…** [S.l.]: IEEE Computer Society, 2006a. p. 532-537.

BOGORNY, V.; CAMARGO, S.; ENGEL, P.; ALVARES, L. O. Mining Frequent Geographic Patterns with Knowledge Constraints. In: ACM INTERNATIONAL SYMPOSIUM ON ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS, ACM-GIS, 14., 2006, Arlington. **Proceedings…** 2006c. Accepted for publication.

BOGORNY, V.; PALMA, A; ENGEL. P. ; ALVARES, L.O. Weka-GDPM: Integrating Classical Data Mining Toolkit to Geographic Information Systems. In: WORKSHOP ON DATA MINING ALGORITHMS AND APPLICATIONS OF THE SBBD, WAAMD, 1., 2006, Florianopolis, Brazil. **Proceedings…** 2006d. Accepted for publication.

BOGORNY, V.; VALIATI, J.; CAMARGO, S.; ENGEL, P.; ALVARES, L. O.: Mining Maximal Generalized Frequent Geographic Patterns with Knowledge Constraints. In: IEEE INTERNATIONAL CONFERENCE ON DATA MINING, ICDM, 6., 2006, Hong-Kong. **Proceedings…** 2006e. Accepted for publication.

BOGORNY, V.; ENGEL, P. M.; ALVARES, L.O. Enhancing the Process of Knowledge Discovery in Geographic Databases using Geo-Ontologies. In: NIGRO, H.

O.; CISARO, S.G.; XODO, D. (Ed.). **Data Mining with Ontologies**: Implementations, Findings, and Frameworks. Accepted by Idea Group for publication in 2007.

BONCHI, F. et al. ExAMiner: Optimized level-wise frequent pattern mining with monotone constraints. In: IEEE INTERNATIONAL CONFERENCE ON DATA MINING, ICDM, 3., 2003, Melbourne. **Proceedings…** [S.l.]: IEEE Computer Society, 2003a. p.11-18.

BONCHI, F. et al. ExAnte: Anticipated Data Reduction in Constrained Pattern Mining. In: EUROPEAN CONFERENCE ON PRINCIPLES AND PRACTICE OF KNOWLEDGE DISCOVERY IN DATABASES, PKDD, 7., 2003, Cavtat-Dubrovnik. **Proceedings…** [S.l.]: Springer, 2003b. p. 59-70.

BONCHI, F.; LUCCHESE, C. On Closed Constrained Frequent Pattern Mining. In: IEEE INTERNATIONAL CONFERENCE ON DATA MINING, ICDM, 4., 2004, Brighton. **Proceedings…** [S.l.]: IEEE Computer Society, 2004. p. 35-42,

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **The unified modeling language:** user guide. Reading: Addison-Wesley, 1998.

BORGES, K.A.V.; LAENDER, A. H. F.; DAVIS JR, C. OMT-G: an object-oriented data model for geographic applications. **GeoInformatica,** [S.l.], v.5, n.3, p.221-260, 2001.

BOULICAUT, J.F.; JEUDY, B. Mining free itemsets under constraints. In: INTERNATIONAL DATABASE ENGINEERING & APPLICATIONS SYMPOSIUM, IDEAS, 2001, Washington DC. **Proceedings…** [S.l.]: IEEE Computer Society, 2001. p. 322-329.

BRIN, S.; MOTWANI, R.; ULLMAN, J.; TSUR, S. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 1997, Tuscon. **Proceedings**... [S.l.]: ACM Press, 1997. p. 255-264.

BURDICK, D.; CALIMLIM, M.; GEHRKE, J. MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases. In: IEEE INTERNATIONAL CONFERENCE DATA ENGINEERING, ICDE, 17., 2001, Heidelberg. **Proceedings…** [S.l.]: IEEE Computer Society, 2001. p. 443-452.

CHAVES, M. S.; SILVA, M. J.; MARTINS, B. A Geographic Knowledge Base for SemanticWeb Applications. In: BRAZILIAN SYMPOSIUM ON DATABASES, SBBD, 20., 2005, Uberlândia. **Proceedings...** [S.l.]: UFU, 2005a. p.40-54.

CHAVES, M. S.; SILVA, M. J.; MARTINS, B. **GKB - Geographic Knowledge Base**. Lisboa: DI/FCUL, 2005. (TR05-12).

CHIFOSKY, E.J.; CROSS, J.H. Reverse Engineering and Design Recovery: a taxonomy. **IEEE Software**, [S.l.], v.7, p.13-17, Jan. 1990.

CLEMENTINI, E.; DI FELICE, P.; VAN OSTERN, P. A small set of formal topological relationships for end-user interaction. In: ABEL, D; OOI, B.C. (Ed.). **Advances in Spatial Databases**. [S.l.]: Springer-Verlag, 1993. p. 277-295. (Lecture Notes in Computer Science, v. 692).

CLEMENTINI, E.; DI FELICE, P.; KOPERSKI, K. Mining multiple-level spatial association rules for objects with a broad boundary. **Data & Knowledge Engineering**, [S.l.], v.34, n.3, p.251-270, Sept. 2000.

COCKCROFT, S. A Taxonomy of Spatial Data Integrity Constraints. **Geoinformatica**, [S.l.], v.1, n.4, p.327-343, Dec. 1997.

EGENHOFER, M.J.; FRANZOSA, F. Point Set Topological Spatial Relations. **International Journal Of Geographical Information Systems**, [S.l.], v.5, n.2, p.161-174, 1991.

EGENHOFER, M.; FRANZOSA, R. On the equivalence of topological relations. **International Journal of Geographical Information Systems**, [S.l.], v.9, n.2, p.133-152, 1995.

ELMASRI, R.; NAVATHE, S. **Fundamentals of database systems.** 4 th ed. [S.l.]:Addison Wesley, 2003.

ESTER, M. et al. Spatial Data Mining: Database Primitives, Algorithms and Efficient DBMS Support. **Journal of Data Mining and Knowledge Discovery**, [S.l.], v.4, n.2-3, p.193-216, July 2000.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. **AI Magazine**, [S.l.], v.17, n.3. p.37-54, 1996.

FUKUDA, T. et al. Mining optimized association rules for numeric attributes. In: ACM SIGMOD SYMPOSIUM ON PRINCIPLES OF DATABASE SYSTEMS, PODS, 15., 1996, Montreal. **Proceedings…** [S.l.]: ACM Press, 1996. p.182-191.

GOETHALS, B.; E ZAKI, M. J. Advances in frequent itemset mining implementations: In: IEEE ICDM WORKSHOP ON FREQUENT ITEMSET MINING IMPLEMENTATIONS, FIMI, 1., 2003, Melbourne. **Proceedings…** [S.l.]: CEUR-WS.org, 2003.

GRUBER, T. R. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. Formal Ontology in Conceptual Analysis and Knowledge Representation. **International Journal of Human-Computer Studies**, [S.l.], v.43, p.907-928, 1993.

GUARINO, N. Formal Ontology and Information Systems. In: INTERNATIONAL CONFERENCE ON FORMAL ONTOLOGY IN INFORMATION SYSTEMS, FOIS, 1., 1998, Trento. **Proceedings…** [S.l.]: IOS Press, 1998. p.3-15.

GUTING, R. H. An Introduction to Spatial Database Systems. **The International Journal on Very Large Data Bases**, [S.l.], v.3, n.4, p. 357 – 399, Oct. 1994.

HADZILACOS, T.; TRYFONA, N. A Model for Expressing Topological Integrity Constraints in Geographic Databases. In: INTERNATIONAL CONFERENCE GIS - FROM SPACE TO TERRITORY: THEORIES AND METHODS OF SPATIO-TEMPORAL REASONING IN GEOGRAPHIC SPACE, GIS, 1992, Pisa. **Proceedings…** London: Springer, 1992. p.252-268.

HAN, J.; FU, Y. Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases. In: AAAI WORKSHOP ON KNOWLEDGE

DISCOVERY IN DATABASES, KDD, 1994, Seattle. **KDD Workshop**. [S.l.:s.n.], 1994. p. 157-168.

HAN, J. Mining Knowledge at Multiple Concept Level. In: INTERNATIONAL CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, CIKM, 4., 1995, Baltimore. **Proceedings…** [S.l.]: ACM, 1995a. p. 19-24.

HAN, J.; FU, Y. Discovery of Multiple-Level Association Rules from Large Databases. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, VLDB, 21., 1995, Zürich. **Proceedings…** [S.l.]: Morgan Kaufmann, 1995b. p. 420-431.

HAN, J.; KOPERSKI, K.; STEFANVIC, N. GeoMiner: a system prototype for spatial data mining. In: ACM-SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 1997, Tucson. **Proceedings…** [S.l.]: ACM Press, 1997. p.553-556.

HAN, J.; PEI, J.; YIN, Y. Mining Frequent Patterns without Candidate Generation. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 2000, Dallas. **Proceedings…**[S.l.]: ACM Press, 2000. p. 1-12.

HAN J. et al. Mining frequent patterns without candidate generation: a frequent-pattern tree approach. **Data Mining and Knowledge Discovery,** [S.l.], v.8, n.1, p. 53-87, Nov. 2004.

HUANG, Y.; SHEKHAR, S.; XIONG, H. Discovering Co-location Patterns from Spatial Datasets: A General Approach. **IEEE Transactions on Knowledge and Data Engineering**, [S.l.], v.16, n.12, Dec. 2004.

KLOSGEN, W.; MAY, M. Spatial Subgroup Mining Integrated in Object-Relational Spatial Database. In: EUROPEAN CONFERENCE ON PRINCIPLES OF DATA MINING AND KNOWLEDGE DISCOVERY, PKDD, 6., 2002, Helsinki. **Proceedings…** [S.l.]: Springer, 2002. p.19-23.

KOPERSKI, K.; HAN, J. Discovery of Spatial Association Rules In Geographic Information Databases. In: INTERNATIONAL SYMPOSIUM ON LARGE GEOGRAPHICAL DATABASES, SSD, 4., 1995, Portland. **Proceedings…** [S.l.]: Springer, 1995. p.47-66.

LISI, F.A.; MALERBA D. Inducing Multi-Level Association Rules from Multiple Relation. **Machine Learning Journal**, [S.l.], v.55, n.2, p.175-210, 2004.

LISBOA FILHO, J.; IOCHPE, C. Specifying analysis patterns for geographic databases on the basis of a conceptual framework. In: ACM INTERNATIONAL SYMPOSIUM ON ADVANCES IN GEOGRAPHIC INFOFRMAION SYSTEMS, ACM-GIS, 7., 1999, Kansas. **Proceedings…** New York : ACM Press, 1999. p. 7-13.

LIU, B.; WYNNE, H.; YIMING, M. Pruning and summarizing the discovered associations. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 5., 1999, San Diego, California. **Proceedings…** [S.l.]: ACM, 1999. p.125–134.

LIU, B. et al. Analyzing the subjective interestingness of association rules. **IEEE Intelligent Systems**, Los Alamitos, CA, v.15, n.5, p.47-55, Sept./Oct. 2000.

LU, W.; HAN, J.; OOI, B. C. Discovery of general knowledge in large spatial databases. In: FAR EAST WORKSHOP ON GEOGRAPHIC INFORMATION SYSTEMS, 1993, Singapura. **Proceedings…** [S.l.:s.n.], 1993. p.275-289.

MALERBA, D.; LISI, F. A. Discovering Associations between Spatial Objects: An ILP Application. In: INTERNATIONAL WORKSHOP ON INDUCTIVE LOGIC PROGRAMMING, 2001. **Proceedings…** Berlin: Springer, 2001. p.156-163.

MALERBA, D.; APPICE, A.; VACCA N.: SDMOQL: an OQL-based data mining query language for map interpretation tasks. In: WORKSHOP ON DATABASE TECHNOLOGIES FOR DATA MINING, DTDM, Prague, 2002. **Proceedings…** [S.l.]: Springer. 2002.

MALERBA, D. et al. Empowering a GIS with inductive learning capabilities: the case of INGENS. **Journal of Computers, Environment, and Urban Systems**, [S.l.], v.27, p. 265-281, 2003.

MCKEARNEY, S.; ROBERTS, H. Reverse engineering databases for knowledge discovery. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 2., 1996, Portland. **Proceedings…** [S.l.]: AAAI Press, 1996. p.375-378.

MELANDA, E. O.; REZENDE, S. O. Combining Quality Measures to Identify Interesting Association Rules. In: IBERO-AMERICAN CONFERENCE ON ARTIFICIAL INTELLIGENCE, IBERAMIA, 9., 2004, Puebla. **Proceedings…** [S.l.]: Springer, 2004. p. 441-453.

MENNIS, J.; LIU, J.W. Mining Association Rules in Spatio-Temporal Data: An Analysis of Urban Socioeconomic and Land Cover Change. **Transactions in GIS,** [S.l.], v.9, n.1, p. 5-17, Jan. 2005.

MORIMOTO, Y. et al. Algorithms for Mining Association Rules for Binary Segmentations of Huge Categorical Databases. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES, VLDB, 24., 1998, New York. **Proceedings…** [S.l.]: Morgan Kaufmann, 1998. p. 380-391.

OPEN GIS CONSORTIUM. **Topic 5, the OpenGIS abstract specification–OpenGIS features–Version 4.** 1999a. Available at <http://www.OpenGIS.org/techno/specs.htm>. Visited on Aug. 2005.

OPEN GIS CONSORTIUM. **OpenGIS simple features specification for SQL.** 1999b. Available at <http://www.opengeospatial.org/docs/99-054.pdf>. Visited on Aug. 2005.

OPEN GIS CONSORTIUM. **Feature Geometry.** 2001. Available at <http://www.opengeospatial.org/specs>. Visited on Aug. 2005.

ORLANDO, S. et al. A multi–strategy algorithm for discovering frequent sets in large databases. In: IEEE ICDM WORKSHOP ON FREQUENT ITEMSET MINING IMPLEMENTATIONS, FIMI, 2003, Melbourne. **Proceedings…** [S.l.:s.n.], 2003.

PARENT, C. et al. Modeling spatial data in the MADS conceptual model. In: INTERNATIONAL SYMPOSIUM ON SPATIAL DATA HANDLING, SDH, 8., 1998, Vancouver. **Proceedings…**[S.l.:s.n.], 1998. p. 138-150.

PASQUIER, N. et al. Discovering frequent closed itemsets for association rules. In: INTERNATIONAL CONFERENCE ON DATABASE THEORY, ICDT, 7., 1999, Jerusalem. **Proceedings…** [S.l.]: Springer, 1999a. p.398-416.

PASQUIER, N. et al. Efficient Mining of Association Rules using Closed Itemset Lattices. **Information Systems**, [S.l.], v.24, n.1, p.25-46, Mar. 1999b.

PADMANABHAN, B.; TUZHILIN, A. A belief-driven method for discovering unexpected patterns. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 4., 1998. **Proceedings…** New York: ACM Press, 1998. p.94-100.

PEI, J.; HAN,J.; MAO, R. CLOSET: an Efficient Algorithm For Mining Frequent Closed Itemsets. In: ACM SIGMOD WORKSHOP ON RESEARCH ISSUES IN DATA MINING AND KNOWLEDGE DISCOVERY, 2000, Dallas. **Proceedings…** [S.l.:s.n.], 2000. p.972-986.

POHLE, C. Integrating and Updating Domain Knowledge with Data Mining. In: PhD WORKSHOP HELD IN CONJUNCTION WITH THE INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES, VLDB, 2003. **Proceedings…** Berlin: CEUR-ws.org, 2003.

RIGAUX, P.; SCHOLL, M.; VOISARD, A. **Spatial Databases:** With Application to GIS. San Francisco: Morgan Kaufmann, 2002.

ROCHA, L.V.; EDELWEISS, N.; IOCHPE, C. GeoFrame-T: A Temporal Conceptual Framework for Data Modeling. In: ACM INTERNATIONAL SYMPOSIUM ON ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS, ACM-GIS, 11., Atlanta. **Proceedings…**[S.l.]: ACM, 2001. p.124-129.

SERVIGNE, S. et al. A Methodology For Spatial Consistency Improvement of Geographic Databases. **Geoinformatica**, [S.l.], v.4, n.1, p.7-34, 2000.

SHEKHAR, S.; LU, C.-T.; ZHANG, P. Detecting graph-based spatial outliers: algorithms and applications (a summary of results). In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 7., 2001, San Francisco. **Proceedings…** [S.l.]: ACM, 2001. p.371-376.

SHEKHAR, S., CHAWLA, S. **Spatial databases**: a tour**.** Upper Saddle River, NJ: Prentice Hall, 2003.

SHOVAL, P.; SHREIBER, N. Database reverse engineering: from the relational to the binary relationship model. **Data and Knowledge Engineering**, [S.l.], v.10, p.293-315, 1993.

SILBERSCHATZ, A.; TUZHILIN, A. What makes patterns interesting in knowledge discovery systems. **IEEE Transactions on Knowledge and Data Engineering,** [S.l.], v.8, n.6, p.970 – 974, Dec. 1996.

SILVA, C.M.S. **Utilizando o processo de descoberta de conhecimento em banco de dados para identificar candidatos a padrão de análise para bancos de dados geográficos**. 2003. 143 f. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

SRIKANT, R.; AGRAWAL, R. Mining Generalized Association Rules. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES, VLDB, 21., 1995, Zurich. **Proceeding…**[S.l.]: Morgan Kaufmann, 1995. p.407-419.

SRIKANT, R.; VU, Q.; AGRAWAL, R. Mining association rules with item constraints. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 2., 1997, Newport Beach. **Proceedings…** [S.l.]: AAAI Press, 1997. p.67-73.

TAN, P.-N.; KUMAR, V.; SRIVASTAVA, J. Selecting the right interestingness measure for association patterns. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 8., 2002, Edmonton. **Proceedings…** [S.l.]: ACM, 2002. p. 32-41.

ZAKI. M. Generating Non-redundant Association Rules. In: ACM SIGKDD INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 6., 2000, Boston. **Proceedings…** [S.l.]: ACM, 2000. p.34-43.

ZAKI., M.; HSIAO, C.. In: INTERNATIONAL CONFERENCE ON DATA MINING, SIAM, 2., 2002, Arlington. **Proceedings…** [S.l.]:SIAM, 2002.

UNO, T.et al. An efficient algorithm for numerating frequent closed item sets. In: IEEE ICDM WORKSHOP ON FREQUENT ITEMSET MINING IMPLEMENTATIONS, FIMI, 2003, Melbourne. **Proceedings…** [S.l.:s.n.], 2003.

UNO, T.; KIYOMI, M.; ARIMURA, H. Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets. In: IEEE ICDM WORKSHOP ON FREQUENT ITEMSET MINING IMPLEMENTATIONS, FIMI, 2004, Brighton. **Proceedings…** [S.l.:s.n.], 2004.

YOO, J.S.; SHEKHAR, S; CELIK, M. A Join-less Approach for Co-location Pattern Mining: A Summary of Results. In: IEEE INTERNATIONAL CONFERENCE ON DATA MINING, ICDM, 5., 2005, Houston. **Proceedings…** [S.l.]: IEEE Computer Society, 2005. p.813-816.

WITTEN, I.; FRANK, E. **Data Mining:** Practical Machine Learning Tools And Techniques with Java Implementations. San Francisco, CA: Morgan Kaufmann, 2005.

APPENDIX A EVALUATING DEPENDENCES BETWEEN THE TARGET FEATURE TYPE AND RELEVANT FEATURE TYPES

=== Run information ===

Instances: 300

Attributes: 35

Crosses_build_up_area, overlaps_tunel, within_bridge, contains_River, bridge_overlaps, touches_tunel, Contains_disturbed_soil,Touches_bridge, Overlaps_Trees, Crosses_disturbed_soil, within_river, crosses_bridge, Overlaps_build_up_area, touches_river, contains_Tunel, Within_build_up_area, touches_road, overlaps_Road, within_tunel, contains_Bridge, Touches_build_up_area, within_road, Contains_trees, within_Trees,Within_disturbed_soil, crosses_Tunel,Crosses_bridge,Contains_build_up_area, overlaps_River,Touches_disturbed_soil, contains_Road, crosses_River, Overlaps_disturbed_soil, touches_bridge, crosses_Road

Apriori

=======

Minimum support: 0.15

Minimum metric <confidence>: 0.9

Number of cycles performed: 17

Generated sets of large itemsets:

Size of set of large itemsets L(1): 9

Size of set of large itemsets L(2): 23

Size of set of large itemsets L(3): 26

Size of set of large itemsets L(4): 14

Size of set of large itemsets L(5): 3

Best rules found:

1. overlaps_River=yes 216 ==> within_Trees=yes 216 conf:(1)
2. overlaps_Road=yes 190 ==> within_Trees=yes 190 conf:(1)
3. overlaps_Trees=yes 180 ==> within_Trees=yes 180 conf:(1)
4. overlaps_Road=yes overlaps_River=yes 149 ==> within_Trees=yes 149 conf:(1)
5. overlaps_tunel=yes 135 ==> within_Trees=yes 135 conf:(1)
6. overlaps_Trees=yes overlaps_Road=yes 133 ==> within_Trees=yes 133 conf:(1)
7. overlaps_Trees=yes overlaps_River=yes 128 ==> within_Trees=yes 128 conf:(1)
8. overlaps_tunel=yes overlaps_River=yes 107 ==> within_Trees=yes 107 conf:(1)
9. overlaps_Trees=yes overlaps_Road=yes overlaps_River=yes 104 ==> within_Trees=yes 104 conf:(1)
10. overlaps_tunel=yes overlaps_Road=yes 93 ==> within_Trees=yes 93 conf:(1)
11. overlaps_tunel=yes Overlaps_Trees=yes 86 ==> within_Trees=yes 86 conf:(1)
12. overlaps_tunel=yes overlaps_Road=yes overlaps_River=yes 77 ==> within_Trees=yes 77 conf:(1)
13. overlaps_tunel=yes Overlaps_Trees=yes overlaps_Road=yes 70 ==> within_Trees=yes 70 conf:(1)
14. contains_River=yes 69 ==> within_Trees=yes 69 conf:(1)
15. contains_River=yes overlaps_River=yes 67 ==> within_Trees=yes 67 conf:(1)

16. overlaps_tunel=yes Overlaps_Trees=yes overlaps_River=yes 66 ==> within_Trees=yes 66 conf:(1)
17. within_road=yes 62 ==> within_Trees=yes 62 conf:(1)
18. contains_Bridge=yes 62 ==> within_Trees=yes 62 conf:(1)
19. contains_River=yes overlaps_Road=yes 59 ==> within_Trees=yes 59 conf:(1)
20. contains_Bridge=yes overlaps_River=yes 58 ==> within_Trees=yes 58 conf:(1)
21. contains_River=yes overlaps_Road=yes overlaps_River=yes 57 ==> within_Trees=yes 57 conf:(1)
22. overlaps_tunel=yes Overlaps_Trees=yes overlaps_Road=yes overlaps_River=yes 56 ==> within_Trees=yes 56 conf:(1)
23. overlaps_Road=yes contains_Bridge=yes 56 ==> within_Trees=yes 56 conf:(1)
24. contains_Road=yes 54 ==> within_Trees=yes 54 conf:(1)
25. overlaps_Road=yes contains_Road=yes 53 ==> within_Trees=yes 53 conf:(1)
26. overlaps_Trees=yes contains_Bridge=yes 53 ==> within_Trees=yes 53 conf:(1)
27. overlaps_Road=yes contains_Bridge=yes overlaps_River=yes 52 ==> within_Trees=yes 52 conf:(1)
28. contains_River=yes Overlaps_Trees=yes 52 ==> within_Trees=yes 52 conf:(1)
29. overlaps_Trees=yes overlaps_Road=yes contains_Bridge=yes 51 ==> within_Trees=yes 51 conf:(1)
30. contains_River=yes Overlaps_Trees=yes overlaps_River=yes 51 ==> within_Trees=yes 51 conf:(1)
31. overlaps_Trees=yes contains_Bridge=yes overlaps_River=yes 50 ==> within_Trees=yes 50 conf:(1)
32. overlaps_Trees=yes overlaps_Road=yes contains_Bridge=yes overlaps_River=yes 48 ==> within_Trees=yes 48 conf:(1)
33. contains_River=yes Overlaps_Trees=yes overlaps_Road=yes 47 ==> within_Trees=yes 47 conf:(1)
34. overlaps_River=yes contains_Road=yes 47 ==> within_Trees=yes 47 conf:(1)
35. contains_River=yes Overlaps_Trees=yes overlaps_Road=yes overlaps_River=yes 46 ==> within_Trees=yes 46 conf:(1)
36. overlaps_Road=yes overlaps_River=yes contains_Road=yes 46 ==> within_Trees=yes 46 conf:(1)
37. overlaps_Trees=yes contains_Road=yes 45 ==> within_Trees=yes 45 conf:(1)
38. contains_Road=yes 54 ==> overlaps_Road=yes within_Trees=yes 53 conf:(0.98)
39. within_Trees=yes contains_Road=yes 54 ==> overlaps_Road=yes 53 conf:(0.98)
40. contains_Road=yes 54 ==> overlaps_Road=yes 53 conf:(0.98)
41. contains_River=yes Overlaps_Trees=yes 52 ==> within_Trees=yes overlaps_River=yes 51 conf:(0.98)
42. contains_River=yes Overlaps_Trees=yes within_Trees=yes 52 ==> overlaps_River=yes 51 conf:(0.98)
43. contains_River=yes Overlaps_Trees=yes 52 ==> overlaps_River=yes 51 conf:(0.98)
44. contains_River=yes Overlaps_Trees=yes overlaps_Road=yes 47 ==> within_Trees=yes overlaps_River=yes 46 conf:(0.98)
45. contains_River=yes Overlaps_Trees=yes overlaps_Road=yes within_Trees=yes 47 ==> overlaps_River=yes 46 conf:(0.98)
46. overlaps_River=yes contains_Road=yes 47 ==> overlaps_Road=yes within_Trees=yes 46 conf:(0.98)
47. within_Trees=yes overlaps_River=yes contains_Road=yes 47 ==> overlaps_Road=yes 46 conf:(0.98)
48. contains_River=yes Overlaps_Trees=yes overlaps_Road=yes 47 ==> overlaps_River=yes 46 conf:(0.98)
49. overlaps_River=yes contains_Road=yes 47 ==> overlaps_Road=yes 46 conf:(0.98)
50. contains_River=yes 69 ==> within_Trees=yes overlaps_River=yes 67 conf:(0.97)
51. contains_River=yes within_Trees=yes 69 ==> overlaps_River=yes 67 conf:(0.97)
52. contains_River=yes 69 ==> overlaps_River=yes 67 conf:(0.97)
53. contains_River=yes overlaps_Road=yes 59 ==> within_Trees=yes overlaps_River=yes 57 conf:(0.97)
54. contains_River=yes overlaps_Road=yes within_Trees=yes 59 ==> overlaps_River=yes 57 conf:(0.97)

APPENDIX B EVALUATING DEPENDENCES AMONG RELEVANT FEATURE TYPES

```
Instances:    300
Attributes:   30
        Touches_build_up_area,overlaps_Road, Contains_disturbed_soil
         Crosses_disturbed_soil, Overlaps_disturbed_soil,
         crosses_River,contains_River,crosses_Tunel, overlaps_River
         Overlaps_build_up_area, crosses_Road, Touches_disturbed_soil,
         within_bridge, within_river, touches_river, contains_Bridge,
         touches_tunel, contains_Road, contains_Tunel,bridge_overlaps,
         touches_road,overlaps_tunel,crosses_bridge, within_tunel,
         Crosses_build_up_area, Within_disturbed_soil,Within_build_up_area,
         within_road, Contains_build_up_area, touches_bridge,
```

Apriori
Minimum support: 0.08
Minimum metric <confidence>: 0.7
Number of cycles performed: 92

Generated sets of large itemsets:
Size of set of large itemsets L(1): 10
Size of set of large itemsets L(2): 25
Size of set of large itemsets L(3): 31
Size of set of large itemsets L(4): 21
Size of set of large itemsets L(5): 3
Best rules found:

1. contains_River=yes contains_Bridge=yes 38 ==> overlaps_River=yes 38 conf:(1)
2. overlaps_Road=yes contains_River=yes contains_Bridge=yes 34 ==> overlaps_River=yes 34 conf:(1)
3. contains_Road=yes overlaps_tunel=yes 31 ==> overlaps_Road=yes 31 conf:(1)
4. contains_River=yes contains_Bridge=yes overlaps_tunel=yes 30 ==> overlaps_River=yes 30 conf:(1)
5. overlaps_River=yes contains_Road=yes overlaps_tunel=yes 28 ==> overlaps_Road=yes 28 conf:(1)
6. overlaps_Road=yes contains_River=yes contains_Bridge=yes overlaps_tunel=yes 27 ==> overlaps_River=yes 27 conf:(1)
7. contains_Bridge=yes contains_Tunel=yes 27 ==> overlaps_River=yes 27 conf:(1)
8. contains_River=yes contains_Bridge=yes contains_Road=yes 26 ==> overlaps_River=yes 26 conf:(1)
9. overlaps_Road=yes contains_River=yes contains_Bridge=yes contains_Road=yes 25 ==> overlaps_River=yes 25 conf:(1)
10. contains_River=yes contains_Bridge=yes contains_Tunel=yes 25 ==> overlaps_River=yes 25 conf:(1)
11. contains_Bridge=yes contains_Road=yes overlaps_tunel=yes 25 ==> overlaps_Road=yes 25 conf:(1)
12. overlaps_Road=yes contains_Bridge=yes contains_Tunel=yes 25 ==> overlaps_River=yes 25 conf:(1)

13. overlaps_River=yes contains_Bridge=yes contains_Road=yes overlaps_tunel=yes 24 ==> overlaps_Road=yes 24 conf:(1)
14. overlaps_disturbed_soil=yes 24 ==> Overlaps_build_up_area=yes 24 conf:(1)
15. overlaps_build_up_area=yes 24 ==> Overlaps_disturbed_soil=yes 24 conf:(1)
16. contains_Road=yes 54 ==> overlaps_Road=yes 53 conf:(0.98)
17. overlaps_River=yes contains_Road=yes 47 ==> overlaps_Road=yes 46 conf:(0.98)
18. contains_River=yes overlaps_tunel=yes 44 ==> overlaps_River=yes 43 conf:(0.98)
19. overlaps_Road=yes contains_River=yes overlaps_tunel=yes 37 ==> overlaps_River=yes 36 conf:(0.97)
20. contains_River=yes contains_Tunel=yes 35 ==> overlaps_River=yes 34 conf:(0.97)
21. contains_River=yes 69 ==> overlaps_River=yes 67 conf:(0.97)
22. contains_Bridge=yes contains_Road=yes 33 ==> overlaps_Road=yes 32 conf:(0.97)
23. contains_River=yes contains_Road=yes 32 ==> overlaps_River=yes 31 conf:(0.97)
24. contains_River=yes contains_Road=yes 32 ==> overlaps_Road=yes 31 conf:(0.97)
25. overlaps_River=yes contains_Bridge=yes contains_Road=yes 31 ==> overlaps_Road=yes 30 conf:(0.97)
26. overlaps_Road=yes contains_River=yes contains_Road=yes 31 ==> overlaps_River=yes 30 conf:(0.97)
27. contains_River=yes overlaps_River=yes contains_Road=yes 31 ==> overlaps_Road=yes 30 conf:(0.97)
28. overlaps_Road=yes contains_River=yes contains_Tunel=yes 30 ==> overlaps_River=yes 29 conf:(0.97)
29. overlaps_Road=yes contains_River=yes 59 ==> overlaps_River=yes 57 conf:(0.97)
30. contains_River=yes contains_Bridge=yes contains_Road=yes 26 ==> overlaps_Road=yes 25 conf:(0.96)
31. contains_Road=yes contains_Tunel=yes 26 ==> overlaps_River=yes 25 conf:(0.96)
32. contains_Road=yes contains_Tunel=yes 26 ==> contains_River=yes 25 conf:(0.96)
33. contains_Road=yes contains_Tunel=yes 26 ==> overlaps_Road=yes 25 conf:(0.96)
37. contains_Bridge=yes contains_Road=yes overlaps_tunel=yes 25 ==> overlaps_Road=yes overlaps_River=yes 24 conf:(0.96)
38. overlaps_Road=yes contains_Bridge=yes contains_Road=yes overlaps_tunel=yes 25 ==> overlaps_River=yes 24 conf:(0.96)

APPENDIX C HIERARCHICAL DEPENDENCES

Instances: 109
Attributes: 17
Dengue, contains_arborizacao, croses_redes_adutoras, contains_hospitais, contains_eta_pol, contains_vilas, contains_antenas_predio, contains_pontos_de_coleta, contains_ponto_captacao, crosses_recurso_hidrico, contains_eta_pt, contains_posto_saude, contains_ramal_e, contains_ParadaOnibus contains_Rua, crosses_rua, contains_avenida, crosses_Avenida, contains_ponte
Apriori
Minimum support: 0.2
Minimum metric <confidence>: 0.9
Number of cycles performed: 16
Generated sets of large itemsets:
Size of set of large itemsets L(1): 13
Size of set of large itemsets L(2): 46
Size of set of large itemsets L(3): 47
Size of set of large itemsets L(4): 19
Size of set of large itemsets L(5): 3
Best rules found:

1. contains_ParadaOnibus=Yes 89 ==> crosses_Avenida=Yes 89 conf:(1)
2. crosses_Avenida=Yes 89 ==> contains_ParadaOnibus=Yes 89 conf:(1)
3. dengue=NAO contains_ParadaOnibus=Yes 75 ==> crosses_Avenida=Yes 75 conf:(1)
4. dengue=NAO crosses_Avenida=Yes 75 ==> contains_ParadaOnibus=Yes 75 conf:(1)
5. contains_Rua=Yes 61 ==> contains_ParadaOnibus=Yes crosses_Avenida=Yes 61 conf:(1)
6. contains_ParadaOnibus=Yes contains_Rua=Yes 61 ==> crosses_Avenida=Yes 61 conf:(1)
7. contains_Rua=Yes crosses_Avenida=Yes 61 ==> contains_ParadaOnibus=Yes 61 conf:(1)
8. contains_Rua=Yes 61 ==> crosses_Avenida=Yes 61 conf:(1)
9. contains_Rua=Yes 61 ==> contains_ParadaOnibus=Yes 61 conf:(1)
10. dengue=NAO contains_Rua=Yes 54 ==> contains_ParadaOnibus=Yes crosses_Avenida=Yes 54 conf:(1)
11. dengue=NAO contains_ParadaOnibus=Yes contains_Rua=Yes 54 ==> crosses_Avenida=Yes 54 conf:(1)
12. dengue=NAO contains_Rua=Yes crosses_Avenida=Yes 54 ==> contains_ParadaOnibus=Yes 54 conf:(1)
13. dengue=NAO contains_Rua=Yes 54 ==> crosses_Avenida=Yes 54 conf:(1)
14. dengue=NAO contains_Rua=Yes 54 ==> contains_ParadaOnibus=Yes 54 conf:(1)
15. contains_ramal_e=Yes contains_ParadaOnibus=Yes 37 ==> crosses_Avenida=Yes 37 conf:(1)
16. contains_ramal_e=Yes crosses_Avenida=Yes 37 ==> contains_ParadaOnibus=Yes 37 conf:(1)
17. contains_ramal_e=Yes contains_Rua=Yes 35 ==> contains_ParadaOnibus=Yes crosses_Avenida=Yes 35 conf:(1)
18. contains_ramal_e=Yes contains_ParadaOnibus=Yes contains_Rua=Yes 35 ==> crosses_Avenida=Yes 35 conf:(1)
19. contains_ramal_e=Yes contains_Rua=Yes crosses_Avenida=Yes 35 ==> contains_ParadaOnibus=Yes 35 conf:(1)
20. contains_ramal_e=Yes contains_Rua=Yes 35 ==> crosses_Avenida=Yes 35 conf:(1)
21. contains_ramal_e=Yes contains_Rua=Yes 35 ==> contains_ParadaOnibus=Yes 35 conf:(1)

22. crosses_recurso_hidrico=Yes 32 ==> contains_ponte=Yes 32 conf:(1)
23. contains_ParadaOnibus=Yes contains_ponte=Yes 31 ==> crosses_Avenida=Yes 31 conf:(1)
24. crosses_Avenida=Yes contains_ponte=Yes 31 ==> contains_ParadaOnibus=Yes 31 conf:(1)
25. contains_pontos_de_coleta=Yes contains_ParadaOnibus=Yes 31 ==> crosses_Avenida=Yes 31 conf:(1)
26. contains_pontos_de_coleta=Yes crosses_Avenida=Yes 31 ==> contains_ParadaOnibus=Yes 31 conf:(1)
27. dengue=NAO contains_ramal_e=Yes contains_ParadaOnibus=Yes 30 ==> crosses_Avenida=Yes 30 conf:(1)
28. dengue=NAO contains_ramal_e=Yes crosses_Avenida=Yes 30 ==> contains_ParadaOnibus=Yes 30 conf:(1)
29. croses_redes_adutoras=Yes contains_ParadaOnibus=Yes 30 ==> crosses_Avenida=Yes 30 conf:(1)
30. croses_redes_adutoras=Yes crosses_Avenida=Yes 30 ==> contains_ParadaOnibus=Yes 30 conf:(1)
31. dengue=NAO croses_redes_adutoras=Yes contains_ParadaOnibus=Yes 29 ==> crosses_Avenida=Yes 29 conf:(1)

APPENDIX D GEOGRAPHIC DATA PREPROCESSING USING GEO-ONTOLOGIES

An overview of the algorithm that implements geographic data preprocessing tasks using geographic ontologies is presented in Figure A. The *Dependence Elimination* step searches the ontology ❄ and verifies the properties of *T*. If *T* has a mandatory dependence *M* with any *O* in *S*, then *O* is eliminated from the set *S* of relevant feature types. Notice that for each relevant feature type removed from the set *S*, no spatial join is required to extract spatial relationships. By consequence, no spatial association rule will be generated with this relevant feature type. If a prohibited relationship *P* is defined between *T* and *O* in the ontology ❄, then the set of possible relationships to compute for data mining is given by $D_{(T,O)} = R - P_{(T,O)}$, where *R* is the set of all topological relationships *R = {touches, contains, within, crosses, overlaps, equals, disjoint}*. If there is no property of *T* in ϕ that relates *T* and *O*, then all relationships are computed.

The *Spatial Join* step computes the spatial relationships *D* between *T* and all remaining *O* in *S*. Spatial joins *D* to extract spatial predicates are performed on-the-fly with operations provided by the GIS. Before computing spatial joins, MBR (Minimum Boundary Rectangle) is performed for accelerating the extraction of spatial relationships.

```
Given:
      GDB, // geographic database
       ❄, // geographic ontology
       T, // target feature type
       S, // set of relevant feature types O
       R; // set of all topological relationships
Variables:
       D; // relationships to compute for Data mining
Find: a dataset Ψ without geographic dependences between T and S;

Method:
 Dependence_Elimination
 Begin
      Ψ = T – geometry column;
      For (i=1; i=#O in S, i++) do
      Begin
        Find T in ❄;
            If (T has a one-one or one-many property with O_i in ❄)
                Remove O_i from S;   // dependence elimination
            Else
                If (T has prohibited properties P with O_i in ❄)
                        D = R – P; // possible relationships to compute
                Else
                        D = R // all topological relationships
            Ψ = Ψ + Spatial_Join (D,T,O_i);//computes spatial relationships D between T and O
       End;
  End;
  Transformation (Ψ) // transforms the resultant dataset into the data mining algorithm
                        // format preserving the non-spatial attributes of T;
```

Figure A: Pseudo-code of the data preprocessing algorithm

The *Transformation* step transposes as well as discretizes the *Spatial Join* module output (Ψ) into the single table format understandable by association rule mining algorithms.

APPENDIX E CONTRIBUIÇÕES DA TESE

A mineração de dados tem como objetivo principal descobrir conhecimento implícito, não trivial, novo e não conhecido a priori. Uma das técnicas utilizadas na mineração é a obtenção de regras de associação. Esta técnica tem sido extensivamente utilizada em bancos de dados transacionais e dezenas de algoritmos foram propostos na última década. Já em bancos de dados geográficos pouquíssimos trabalhos têm sido desenvolvidos para descoberta de conhecimento utilizando a técnica de regras de associação.

O maior e bem conhecido problema da técnica de mineração de regras de associação é o grande número de regras geradas, que precisam ser analisadas uma a uma pelo usuário, na busca por algum conhecimento novo e interessante. Na mineração de regras de associação em bancos de dados geográficos este problema é ainda mais grave devido ao grande número de padrões naturais e bem conhecidos que existem entre os dados geográficos. Enquanto em bancos de dados transacionais os itens são normalmente independentes, em bancos de dados geográficos os objetos estão naturalmente co-rrelacionados e muitos são dependentes uns dos outros. Por exemplo, uma ilha está sempre relacionada a um recurso hídrico, um posto de gasolina está sempre relacionado a uma via, um trevo está sempre relacionado a duas ou muitas estradas, e assim por diante. Quando considerados na mineração de regras de associação espacial, objetos geográficos interdependentes são agrupados devido às suas dependências naturais e geram uma grande quantidade de regras óbvias como por exemplo *contém_ilha*→*contém_rio.*

Por um lado, embora existam muitos padrões bem conhecidos/óbvios em bancos de dados geográficos, é difícil definir medidas objetivas que eliminem estes padrões, ou que sejam capazes de definir se um padrão é interessante ou não. Por outro lado, conhecendo os padrões que não são interessantes é possível utilizá-los como conhecimento a priori para evitar que os algoritmos de mineração de regras de associação extraiam estes mesmos padrões.

As dependências geográficas são bem conhecidas pelo projetista de bancos de dados geográficos e normalmente são explicitamente definidas nos esquemas de bancos dados geográficos. Essas dependências também são representadas em ontologias geográficas, uma vez que elas fazem parte do conceito dos dados geográficos e representam restrições de integridade espacial que precisam ser garantidas a fim de manter a qualidade dos dados. As dependências geográficas também podem ser vistas como restrições semânticas. Por exemplo, considerando que um objeto geográfico é um ponto

e um outro objeto é um polígono, não podemos afirmar que estes objetos tem uma dependência. Entretanto, quando sabemos que o ponto é uma ilha e que o polígono é um lago, então podemos identificar essa dependência geográfica.

Considerando a grande quantidade de padrões bem conhecidos que os atuais algoritmos de mineração de regras de associação espacial geram, nesta tese se propõe o uso de conhecimento a priori a fim de reutilizar o conhecimento do projetista do banco de dados geográfico representado no esquema de dados ou o existente em ontologias do domínio geográfico, para evitar que esses padrões sejam extraídos e mostrados ao usuário de mineração de dados.

O Capítulo 1 da tese apresenta em maiores detalhes a motivação que levou à realização deste trabalho. O Capítulo 2 apresenta os conceitos básicos sobre bancos de dados geográficos bem como um estudo de caso sobre esquemas de bancos de dados geográficos reais. O Capítulo 3 apresenta o estado da arte em mineração de regras de associação em dados transacionais e dados geográficos. Também é apresentada no Capítulo 3 uma análise detalhada do número de conjuntos freqüentes e regras de associação gerados pelos padrões bem conhecidos, existentes nos dados geográficos.

Esta tese é um dos primeiros trabalhos que utiliza conhecimento a priori para reduzir o número de padrões bem conhecidos (óbvios) na mineração de regras de associação em bancos de dados geográficos. Diferentes soluções para reduzir o problema são apresentadas. Estas soluções estão descritas no Capítulo 4 e os experimentos para mostrar a eficiência e eficácia das soluções propostas são apresentados no Capítulo 5. Os experimentos foram realizados com bancos de dados reais. O capítulo 6 descreve um protótipo desenvolvido para facilitar o pré-processamento de dados geográficos para a mineração. Finalmente, o Capítulo 7 apresenta as conclusões da tese e os trabalhos futuros.

As principais contribuições desta tese incluem:

1. Automatização do pré-processamento de dados geográficos, integrando a ferramenta Weka com bancos de dados geográficos seguindo as normas estabelecidas pelo padrão OpenGIS visando a interoperabilidade com diferentes bancos de dados. Este trabalho foi publicado em (BOGORNY, 2005a, 2006d);

2. Uso de ontologias geográficas para reduzir o número de junções espaciais no pré-processamento de dados geográficos. O resultado deste trabalho foi publicado em (BOGORNY, 2005b);

3. Uso de ontologias para redução de regras de associação espacial que contenham padrões óbvios. Neste método ontologias geográficas são utilizadas tanto no pré-processamento para reduzir o número de junções espaciais bem como as dependências possíveis de serem eliminadas no pré-processamento. Também é proposta uma alteração no algoritmo Apriori que elimina pares de objetos geográficos com dependências utilizando a ontologia como base de conhecimento. Este trabalho será publicado em (BOGORNY, 2007) ;

4. Uso de conhecimento extraído esquemas ou definido pelo usuário, a fim de eliminar parcialmente as dependências geográficas no processo de mineração de regras de associação espacial. Este método de eliminação de dependências é aplicado no pré-processamento de dados geográficos e aumenta a eficiência

e eficácia de qualquer algoritmo de mineração de regras de associação espacial, uma vez que a eliminação das dependências é realizada numa etapa anterior a mineração. Esta contribuição foi publicada em (BOGORNY, 2006a)

5. Uso de conhecimento a priori, extraído de esquemas ou definido pelo usuário, para eliminar parcialmente as dependências geográficas bem conhecidas no processo de mineração de regras de associação espacial. Este método é aplicado durante a geração dos conjuntos freqüentes, a partir dos quais posteriormente são geradas as regras, para eliminar dependências que não podem ser removidas no pré-processamento. Para isso foi alterado o algoritmo Apriori. O resultado deste trabalho foi publicado em (BOGORNY, 2006b)
6. Uso de conhecimento a priori, extraído de esquemas ou definido pelo usuário, para excluir completamente as dependências geográficas na mineração de regras de associação espacial. Este método reúne as duas abordagens anteriores, ou seja, elimina todas as dependências passíveis de serem removidas no pré-processamento sem que haja perda de informação, e as demais são eliminadas durante a geração dos conjuntos freqüentes. Este trabalho foi publicado em (BOGORNY, 2006c).
7. Eliminação de dependências hierárquicas quando os dados são minerados em níveis de granularidade mais gerais ou mais específicos. A mineração de dados em níveis de granularidade menores gera mais atributos e como conseqüência mais regras são geradas contendo as dependências bem conhecidas. Os experimentos mostram que nesses casos o método proposto nesta tese reduz significativamente o número total de padrões. Esta contribuição ainda não foi publicada.
8. Esta tese também apresenta uma contribuição para a redução não somente de padrões bem conhecidos gerados pelos relacionamentos espaciais entre diferentes objetos geográficos, mas também para a redução das regras geradas pelos mesmos objetos com diferentes tipos de relacionamento espacial (topológicos). Por exemplo, *toca_rio*→*contem_rio*. Esta contribuição também ainda não foi publicada.
9. Nas contribuições apresentadas acima as dependências são eliminadas no pré-processamento ou durante a geração dos conjuntos freqüentes (dentro do algoritmo de mineração). Essa eliminação além de eliminar padrões bem conhecidos, reduz significativamente o número total de conjuntos freqüentes. Na literatura de mineração de regras de associação em bancos de dados transacionais existem dois tipos de algoritmos, os que geram conjuntos freqüentes e os que geram conjuntos freqüentes fechados (*closed frequent sets*). Os conjuntos freqüentes fechados são gerados numa segunda etapa, após a geração dos conjuntos freqüentes. Conjuntos freqüentes fechados para mineração de regras de associação espacial em bancos de dados geográficos foram usados pela primeira vez nesta tese. E este método reduz significativamente o número final de conjuntos e regras. Este trabalho será publicado em (BOGORNY, 2006e).