A GRAPH-BASED APPROACH FOR SEMANTIC DATA MINING

by

HAISHAN LIU

A DISSERTATION

Presented to the Department of Computer and Information Science
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

September 2012

DISSERTATION APPROVAL PAGE

Student: Haishan Liu

Title: A Graph-based Approach for Semantic Data Mining

This dissertation has been accepted and approved in partial fulfillment of the requirements for the Doctor of Philosophy degree in the Department of Computer and Information Science by:

Dr. Dejing Dou	Chair
Dr. Arthur Farley	Member
Dr. Allen Malony	Member
Dr. Daniel Lowd	Member
Dr. Don Tucker	Outside Member

and

Kimberly Espy	Vice President for Research & Innovation/ Dean of the Graduate School

Original approval signatures are on file with the University of Oregon Graduate School.

Degree awarded September 2012

DISSERTATION ABSTRACT

Haishan Liu

Doctor of Philosophy

Department of Computer and Information Science

September 2012

Title: A Graph-based Approach for Semantic Data Mining

Data mining is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. It is widely acknowledged that the role of domain knowledge in the discovery process is essential. However, the synergy between domain knowledge and data mining is still at a rudimentary level. This motivates me to develop a framework for explicit incorporation of domain knowledge in a data mining system so that insights can be drawn from both data and domain knowledge. I call such technology "semantic data mining."

Recent research in knowledge representation has led to mature standards such as the Web Ontology Language (OWL) by the W3C's Semantic Web initiative. Semantic Web ontologies have become a key technology for knowledge representation and processing. The OWL ontology language is built on the W3C's Resource Description Framework (RDF) that provides a simple model to describe information resources as a graph. On the other hand, there has been a surge of interest in tackling data mining problems where objects of interest can be best described as a graph of interrelated nodes. I notice that the interface between domain knowledge and data mining can be achieved by using graph representations. Therefore I explore a graph-based approach for modeling both knowledge and data and for analyzing the combined information source from which insight can be drawn systematically.

In summary, I make three main contributions in this dissertation to achieve semantic data mining. First, I develop an information integration solution based on metaheuristic optimization when data mining task require accessing heterogeneous data sources. Second, I describe how a graph interface for both domain knowledge and data can be structured by employing the RDF model and its graph representations. Finally, I describe several graph theoretic analysis approaches for mining the combined information source. I showcase the utility of the proposed methods on finding semantically associated itemsets, a particular case of the frequent pattern mining. I believe these contributions in semantic data mining can provide a novel and useful way to incorporate domain knowledge.

This dissertation includes published and unpublished coauthored material.

CURRICULUM VITAE

NAME OF AUTHOR: Haishan Liu

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene, OR, USA
Shanghai Jiao Tong University, Shanghai, China

DEGREES AWARDED:

Doctor of Philosophy in Computer Science, 2012, University of Oregon
Master of Science in Computer Science, 2008, University of Oregon
Bachelor of Information Security, 2006, Shanghai Jiao Tong University.

AREAS OF SPECIAL INTEREST:

Data mining, machine learning, the Semantic Web

GRANTS, AWARDS AND HONORS:

Graduate Teaching & Research Fellowship, Computer and Information Science, 2006 to present

NSF Travel Grant, International Semantic Web Conference, 2010

NSF Travel Grant, International Conference on Data Mining, 2011

PUBLICATIONS:

Haishan Liu, Dejing Dou and Hao Wang, Breaking the Deadlock: Simultaneously Discovering Attribute Matching and Cluster Matching with Multi-Objective Metaheuristics, Journal On Data Semantics, Volume 1, Number 2 (2012), Pages 133-145

Haishan Liu, Gwen Frishkoff, Robert Frank, Dejing Dou, Sharing and integration of cognitive neuroscience data: Metric and pattern matching across heterogeneous ERP datasets, Neurocomputing, Volume 92, 1 September 2012, Pages 156-169, ISSN 0925-2312.

Haishan Liu, Paea LePendu, Ruoming Jin, and Dejing Dou. A Hypergraph-based Method for Discovering Semantically Associated Itemsets. In Proceedings of the 11th IEEE International Conference on Data Mining (ICDM), Pages 398-406, 2011.

Haishan Liu and Dejing Dou. Breaking the Deadlock: Simultaneously Discovering Attribute Matching and Cluster Matching with Multi-Objective Simulated Annealing. In Proceedings of the International Conference on Ontologies, Databases and Application of Semantics (ODBASE), pages 698–715, 2011.

Dejing Dou, Han Qin, and Haishan Liu. Semantic Translation for Rule-based Knowledge in Data Mining. In Proceedings of the 22nd International Conference on Database and Expert Systems Applications (DEXA), part II, pages 74–89, 2011.

Haishan Liu. Towards Semantic Data Mining. In: Doctoral Consortium of the 9th International Semantic Web Conference (ISWC), November 2010.

Christopher Townsend, Jingshan Huang, Dejing Dou, Shivraj Dalvi, Patrick J. Hayes, Lei He, Wen-chang Lin, Haishan Liu, Robert Rudnick and Hardik Shah. OMIT: Domain Ontology and Knowledge Acquisition in MicroRNA Target Prediction. On-The-Move Conferences 2010 (ODBASE): 1160–1167

Haishan Liu, Gwen Frishkoff, Robert Frank, and Dejing Dou. Ontology-based Mining of Brainwaves: A Sequence Similarity Technique for Mapping Alternative Descriptions of Patterns in Event Related Potentials (ERP) Data. In Proceedings of the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pages 43–54, 2010.

Gwen Frishkoff, Paea LePendu, Robert Frank, Haishan Liu, and Dejing Dou. Development of Neural Electromagnetic Ontologies (NEMO): Ontology-based Tools for Representation and Integration of Event-related Brain Potentials. In Proceedings of the International Conference on Biomedical Ontology (ICBO), pages 31–34, 2009.

Haishan Liu and Dejing Dou. An Exploration of Understanding Heterogeneity through Data Mining. In Proceedings of KDD'08 Workshop on Mining Multiple Information Sources, pages 18–25, 2008.

Jongwan Kim, Dejing Dou, Haishan Liu, and Donghwi Kwak. Constructing a user preference ontology for anti-spam mail systems. In Proceedings of the 20th Canadian Conference on Artificial Intelligence (Canadian AI), pages 272–283, 2007.

ACKNOWLEDGEMENTS

Toward my many mentors I feel the deepest gratitude and respect. In particular, I acknowledge the tireless support of my research advisor, Dr. Dejing Dou, who provided guidance to me through all the steps that led to this dissertation. I also sincerely thank my committee members and the Computer and Information Science (CIS) faculty, staff for their time, encouragement, and instruction. I have fond remembrances of my assistantships on the Neural ElectroMagnetic Ontologies (NEMO) projects and I thank those colleagues for wonderful experiences.

I humbly acknowledge the funding support of the CIS department, Dr. Dejing Dou, and NEMO during my studies.

I wish, finally, to thank my parents and family for giving me the opportunity to fulfil my dreams and the support to achieve my ambitions. Without them this work would not have been concluded.

This work is dedicated to my wife Xiaofei Zhang for her support and endurance in all those moonlight nights we are apart.

TABLE OF CONTENTS

Chapter Page

I. INTRODUCTION 1

II. BACKGROUND 10

2.1. Domain Knowledge in Data Mining 10

2.2. Graph-based Approach for Knowledge Representation 25

2.3. Graphs in Data Mining 29

2.4. Integration of Heterogeneous Information Sources 34

III. GRAPH REPRESENTATION 40

3.1. Graph Representation for Domain Knowledge 40

3.2. Graph Representation for Relational Structures 45

3.3. Combining Data Graphs and Ontology Graphs 50

IV. INTEGRATION OF HETEROGENEOUS INFORMATION SOURCES 55

4.1. Overview 55

4.2. Method 59

4.3. Case Studies 67

4.4. Discussion 79

Chapter Page

V. GRAPH-BASED MINING FOR SEMANTICALLY ASSOCIATED ITEMSETS 83

5.1. Overview 84

5.2. Method 86

5.3. Case Studies 95

5.4. Discussion 104

VI. MINING SEMANTICALLY ASSOCIATED ITEMSETS WITH ONTOLOGIES 112

6.1. Method 113

6.2. Case Studies 118

VII. CONCLUSION 129

7.1. Future Work 130

7.2. Concluding Remarks 135

REFERENCES CITED 137

LIST OF FIGURES

Figure Page

1.1. The proposed workflow for semantic data mining 5

2.1. An example of simple graph vs. hypergraph 30

2.2. A simple graph of friendship relationship 31

2.3. A comparison between the Euclidean and the commute time distance. 32

2.4. An example of cluster density profiles 39

3.1. A comparison between a simple graph and a hypergraph 42

3.2. An example incidence matrix of a hypergraph 43

3.3. A comparison between the directed labeled graph and the RDF bipartite graph 45

3.4. The RDF bipartite graph for a nominal-valued table 47

3.5. An example relational table and a domain ontology 52

3.6. The RDF bipartite graph representation 53

3.7. Transforming the RDF bipartite graph to suit mining need 54

3.8. An example RDF bipartite graph that represents various semantic relationships 54

4.1. The distribution of synthetic datasets 69

4.2. An example Pareto front obtained from matching two synthetic datasets 70

4.3. A comparison between methods on the neuroscience dataset 75

5.1. An example of the hypergraph coarsening process. 86

5.2. The degree distributions of experiment datasets. 105

5.3. An example star graph 107

Figure Page

5.4. An example connection graph between people and movies 109

5.5. The 3-D plot of node vectors in the transformed space 110

6.1. An example RDF bipartite graph and a detailed anatomy of its biadjacency matrix 115

6.2. The may_treat subgraph 127

LIST OF TABLES

Table Page

2.1. Summary of systems that employ domain knowledge. 18

2.2. Summary of ontology-based data mining systems. 24

3.1. An example relational table with nominal features and the corresponding RDF 47

3.2. An example relational table with binary features. 48

3.3. An example expanded relational table with binary features. 49

3.4. Nominal value expansion for a relational table and the resulting RDF triples 50

4.1. Example distance matrices 58

4.2. Running time of the annealing process on the synthetic dataset 71

4.3. The performance of MOSA on the neuroscience dataset 76

4.4. The performance of MOGA on the neuroscience dataset 77

4.5. Statistical distribution of attributes in the Wine Quality dataset 78

4.6. The performance of MOSA and MOGA on the Wine Quality dataset 79

5.1. Top semantically associated itemsets on the synthetic dataset 97

5.2. Top s_{CT} results on the shopping cart dataset 98

5.3. Top s_{L+} results on the shopping cart dataset 99

5.4. Top itemsets ranked by the frequency of occurrences on the shopping cart data 99

5.5. A comparison between s_{CT} and s_{L+} 101

5.6. Top s_{CT} results on the kidney failure cohort 102

5.7. Top s_{L+} results on the kidney failure cohort 102

Table Page

5.8. Top s_{L+} results on the whole electronic health dataset 104

6.1. Overview of the test cases 119

6.2. Top results on the Foodmart dataset 122

6.3. Top results on the Foodmart dataset excluding concepts only in the ontology 123

6.4. Top results on the electronic health dataset 125

6.5. Rankings of three semantic associations in health data under different settings 126

6.6. Rankings of associations on the noisy may_treat graph 127

CHAPTER I

INTRODUCTION

Data mining, also referred to as knowledge discovery in databases (KDD), is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data [1]. But the measure of what is meant by "useful" to the user is dependent on the user as well as the domain within which the data mining system is being used. Therefore, the role of domain knowledge in the discovery process is essential. Fayyad *et al.* [2] contended that the use of domain knowledge is important in all stages of the data mining process including, for example, data transformation, feature reduction, algorithm selection, post-processing, model interpretation and so forth.

The first step towards using domain knowledge is to acquire the knowledge from experts and thus model and codify the knowledge in the computer. Russell and Norvig [3] emphasized that a data mining system must have some method for obtaining the background knowledge and can no longer make naive speculations, and should use its background knowledge to learn more and more effectively. This process of modeling knowledge in computer systems to facilitate problem solving is studied in the field of knowledge representation/engineering. Research in this field has resulted in many sophisticated technologies such as expert systems. However, at present, knowledge representation and data mining remain largely separate disciplines. Although it is widely stated that exploring the fusion of the two fields is worthwhile in many applications where substantial human expertise exists alongside data resources, as many researchers have pointed out, work along this line is still in its infancy [4–8].

This problem motivates us to develop a framework for explicit incorporation of domain knowledge in a data mining system so that insights can be drawn from both knowledge and data in a systematic and holistic way. We call such technology "semantic data mining." This dissertation contributes a first step towards realizing this goal by providing a graph-based formalism and analysis methods thereof, to systematically incorporate a specific kind of ontological domain knowledge that can be directly encoded in the W3C's Resource Description Framework (RDF) triple syntax. We showcase the utility of such method by providing theoretical, methodological and empirical insights into solving some certain non-trivial data mining tasks, such as the semantic association mining as detailed later in the chapter.

Recent research in knowledge representation, particularly in the area of W3C's Semantic Web [9] that seeks to embed semantic content in web pages, has led to mature standards such as the Web Ontology Language (OWL [10]) for authoring ontologies. An ontology is an explicit specification of a conceptualization [11]. Today, Semantic Web ontologies have become a key technology for intelligent knowledge processing, providing a framework for sharing conceptual models about a domain [12]. Ontologies explicate domain knowledge hence providing a way to separate knowledge from implementations [13]. Much effort has been devoted to developing tools for coding and managing OWL ontologies [14, 15]. Ontologies are used in various contexts, particularly those dealing with information that encompasses a limited and defined domain, and where sharing data is a common necessity, such as scientific research. Prominent examples of such efforts include the Gene Ontology (GO [16]), Unified Medical Language System (UMLS [17]), and more than 300 ontologies in the NCBO

(National Center for Biomedical Ontology) such as the Neural ElectroMagnetic Ontologies (NEMO [18, 19]).

The OWL is built on the W3C's Resource Description Framework (RDF) that provides a simple model to describe information resources as a graph. The core of the framework is the RDF statements consisting of triples including a subject (the resource being described), a predicate (the property) and an object (the property value). This simple model of assertions leads to a network of information resources, interrelated by properties which establish relations, or links, between resources and property values. The term RDF Graph is defined as a set of RDF triples (which can be illustrated by a node-arc-node link). Therefore, any collection of RDF data is an RDF Graph [20].

At the same time, there has been a surge of interest in tackling the problem of mining semantically rich datasets, where objects are linked in a number of ways. In fact, many datasets of interest today are best described as a linked collection, or a graph, of interrelated objects [21]. These graphs may represent homogeneous networks, in which there is a single-object type and link type (such as web pages connected by hyperlinks), or richer, heterogeneous networks, in which there may be multiple object and link types (such as DBpedia, a data source containing structured information from Wikipedia). Many traditional information resources and formats can be viewed as graphs or linked collections as well. Such links and their characteristics are explored, often implicitly, by well-established data analysis and mining techniques, whose formalism of the problem are however typically not based on graphs. For example, consider a simple transaction table and the problem of frequent itemset generation. An itemset is deemed frequent if its *support*, i.e., the percentage of transactions which contain that itemset, is above a threshold.

If we characterize the "co-occurrence" relationship (items appearing together in a tuple) as a link between items, then the transaction table can be viewed as a graph consisting of a set of items connected by such links. Furthermore, in this sense, the problem of frequent pattern mining can be reformulated as to identify sets of nodes in the graph that are heavily connected by the co-occurrence links.

We believe the interface between domain knowledge and data mining can be achieved, to some extent, by using graph representations in which distinct sorts of knowledge that have been traditionally differently represented can be structured in a unified manner. For example, previously, one important aspect of the distinction between domain knowledge and data is the different representations for *ontological* and *factual* knowledge. Ontological knowledge is related to general categories, also called concepts or classes (such as those defined in OWL ontologies). Factual knowledge makes assertions about a specific situation (*e.g.*, this specific entity belongs to a certain category, and has a certain relationship with another entity, such as those defined in knowledge bases or relational databases). However, this distinction can be obscured by the simple semantics of the RDF model given the fact that RDF allows a combined specification of both schema and data structured under this schema. Since RDF's abstract triple syntax has a graph nature, and graphs are one of the most fundamental objects studied in mathematics that have a strong relation with logical languages and data structures, it is promising to develop graph-based approaches that provide a common ground to interface with both domain knowledge and data mining. Therefore, in this dissertation, we explore a particular graph-based method for modeling both knowledge and data, and for analyzing the combined information source from which insight can be derived systematically. We expect this novel paradigm to contribute to the development of new principles towards semantic data mining.

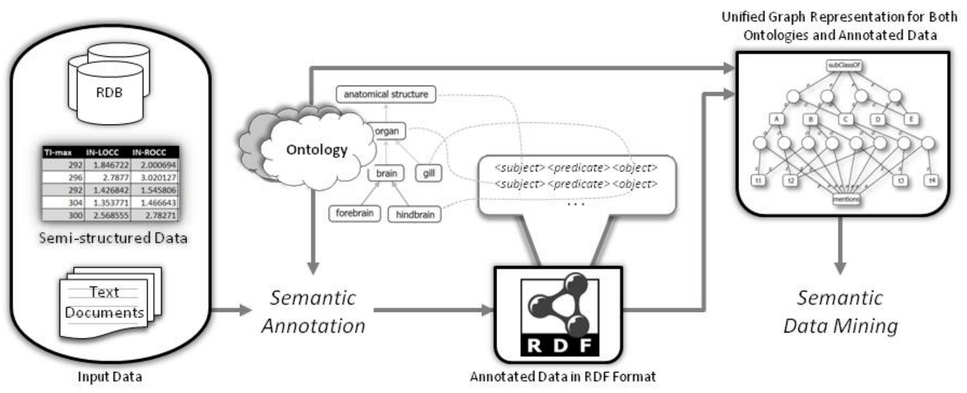


FIGURE 1.1. The proposed workflow for semantic data mining.

Figure 1.1 illustrates the proposed workflow for semantic data mining. Starting from a source of input data in some certain format, the first step is to identify suitable ontologies that encode concepts and relationships needed to describe the domain. Then semantic annotation is performed to link the basic element of data with formal semantic descriptions in ontologies [22]. Next, a growing number of mining tasks require an integration step to be performed in order to access and derive insights from heterogeneous information sources. Finally, the integrated data can be stored in the RDF format and be represented, together with the ontologies, by an expressive and flexible graph model for subsequent analyses.

Since graph-based semantic data mining is a new field, many interesting research directions related to it are yet to be explored. This dissertation studies three such research directions. Brief descriptions of these directions are presented below.

1. **A graph representation for both domain knowledge and data mining:** The interface between knowledge representation and data mining is achieved by employing the RDF model and by the fact that RDF allows a combined specification of both schema and data structured under this schema. RDF's ability to represent disparate, abstract concepts has led to its increasing use in knowledge representation. The RDF core vocabulary and the RDF Schema provide the most basic predefined concepts to express express schematic information. "Richer" schema and ontology defining languages (e.g., OWL) that are built upon RDF continue to evolve.

   On the other hand, in practice, vast amounts of data often persist in relational databases (RDB). Mapping from RDB to RDF has gained increasing attention and led to the implementation of generic mapping tools as well as domain–specific applications. The W3C launched the RDB2RDF incubator group to explore issues involved in mapping RDB to RDF [23]. An outstanding advantage of expressing data in RDF is the explicit modeling of relationships between entities that are either implicit or non–existent in RDB. In this way, one can achieve the incorporation of "domain semantics," an important aspect to fully leverage the expressivity of RDF models that enables data mining systems to explore beyond pure data. Furthermore, the role of RDF as an integration platform for data from multiple sources is another main motivation for mapping RDB to RDF.

   RDF's abstract triple syntax has a graph nature. Graphs are mathematical objects that enjoy wide-spread usage for many tasks, which include the visualization and analysis of data for humans, mathematical reasoning, and the implementation as a data structure for developing data mining algorithms. Besides the common graph-theoretic model of RDF as labeled, directed multi-graphs, Hayes has established that RDF can be also represented as hypergraphs (bipartite graphs) [20]. This result constitutes an important aspect of the theoretical basis of this dissertation and is discussed in Chapter II. Our novel contribution is a set of methods to represent data in relational structures using graphs in ways that are consistent with the RDF hypergraph/bipartite representation. The unified graph representation for both the data and domain knowledge encoded in ontologies is the basis for developing meaningful semantic data mining algorithms.

2. **An advanced method to enable data integration and meta-analysis at the same time:** The presence of heterogeneity among schemas and ontologies supporting vast amounts of information demands advanced solution for semantic integration of disparate data sources to facilitate interoperability and reuse of the information. Another challenging task given multiple data sources is to carry out meaningful meta-analysis that combines results of several studies on different datasets to address a set of related research hypotheses.

   We identify two prominent problems in enabling data integration and meta-analysis, namely, attribute matching and cluster matching. It can be shown that these two problems are interlocked with each other and cannot be solved separately. Therefore we develop a solution that casts them to combinatorial optimization problems with distinct yet interrelated objective functions. The core idea is a novel approach using multi-objective heuristics to discover attribute matching and cluster matching simultaneously. Details of the methods are presented in Chapter IV.

3. **A graph theoretic analysis approach for mining the combined information source of both data and knowledge:**

   The particular mining problem we aim to solve in this dissertation is motivated by a simple scenario illustrated by Swanson [24] years ago while studying Raynauld's syndrome. He noticed that the literature discussed Raynauld's syndrome (Z), a peripheral circulatory disease, together with certain changes of blood in human body (Y); and, separately, the consumption of dietary fish oil (X) was also linked in the literature to similar blood changes. But fish oil and Raynauld's syndrome were never linked directly in any previous studies. Swanson reasoned (correctly) that fish oil could potentially be used to treat Raynauld's syndrome, i.e., $X \rightsquigarrow Y \rightsquigarrow Z$. We call such indirectly associated items, (X, Z), *semantically associated itemsets*.

   Our approach is based on the RDF hypergraph/bipartite representation to capture both ontologies and data. We can weight each hyperedge so that certain links can carry appropriate strength. Then, drawing inspiration from a rich body of literature on graph mining and graph spectral analysis, we explore some highly efficient and scalable similarity measures over the bipartite graph to generate frequent itemsets, including associations that may not necessarily be co-frequent. Details of these approaches are presented in Chapter V and VI.

The remainder of this dissertation is organized as follows. In Chapter II, we discuss the background areas related to the original research work presented. The main contributions of this dissertation are presented in Chapters III, IV, V and VI. Chapter III discusses the theory of hypergraph-based representation of both knowledge and data. Chapter IV introduces the method for integration of heterogeneous information sources. Chapters V is dedicated to the hypergraph-based analysis method based solely on data without ontologies, while Chapter VI describes ways to incorporate ontologies. Both chapters focus on solving a special kind of mining task called semantically associated itemset discovery. Finally, in Chapter VII, we discuss future directions for the research work and provide some concluding remarks.

This dissertation includes published and unpublished coauthored materials. I acknowledge the contribution of Dr. Dejing Dou, my advisor, who participated in the design and development of the principles of semantic data mining described in Chapter III, IV, V and VI. I am also thankful to coauthors Dr. Gwen Frishkoff and Robert Frank who contributed to the study on the neuroscience dataset in Chapter IV, Dr. Paea LePendu and Dr. Nigam Shah who contributed to the study on the electronic health dataset in Chapter V and VI, and Dr. Ruoming Jing who contributed to the design of graph-based mining algorithms in Chapter V and VI.

CHAPTER II

BACKGROUND

This chapter covers the background areas and related work necessary to understand the contributions of this dissertation. It discusses the current state of the art efforts to incorporate domain knowledge in data mining. In addition, it describes the use of graphs in data mining with a focus on graph-based similarities. Researches in metaheuristics optimization, schema matching, cluster comparison and so forth are also briefly discussed.

2.1. Domain Knowledge in Data Mining

Domain knowledge relates to information about a specific domain or data that is collected from previous systems or documentation, or elicited from domain experts. In the rest of the section, we highlight a body of studies that aims at exploring ways to employ domain knowledge in data mining. The results from these studies strongly attest to the positive influence of domain knowledge. Domain knowledge can affect the discovery process within a data mining system in at least two ways. First, it can make patterns more visible by generalizing the attribute values, and second, it can constrain the search space as well as the rule space.

In order to effectively summarize and compare different previously proposed systems, we propose a reference framework to classify different kinds of domain knowledge at a very high abstraction level as detailed in the following.

- Knowledge about the domain: This category contains information related to a specific domain of discourse, usually obtained from either domain experts or previous data mining processes. Examples of such knowledge include concept hierarchy, integrity constraints, *etc.*

– Knowledge about the data: This category contains information about a dataset, including how it is generated, transformed and evolved. This knowledge is obtained from data producers (people who carry out experiments or collect data) or database managers. For example, in a database of spatial information, one of the images may have been recorded with a very skew angle on the object. When processing the database the discovery process must take this information into account.

– Knowledge about the data mining process: This category contains information pertaining to specific data mining tasks, including goals, parameters and variables related to the experiment. For example, attributes of interest within data, and the measure of interestingness for discovered patterns.

The summarized work can be divided into two groups. The first group does not explicitly leverage any knowledge representation approaches to model domain knowledge. The second group explores mainly ontological knowledge (concept hierarchy) and uses formal ontology languages to encode such knowledge. The kind of domain knowledge involved in the first group is broader which covers all categories discussed in the above reference classification scheme. However, it is achieved at the cost of less formality which often results in ad-hoc expression of domain knowledge that has a very application-specific form, scop and granularity.

In one of the earliest studies on the subject, Pazzani and Kibler [25] developed a general purpose relational learning algorithm called FOCL, which combines explanation-based and inductive learning. In a later study, they conducted an experiment comparing FOCL with a domain theory to FOCL without a domain theory. A partial knowledge base of an expert system was used as the domain theory. They found incorporating domain theory significantly reduced misclassification costs when larger training sets were used.

In another study, Ambrosino and Buchanan [26] examined the effects of adding domain knowledge to a rule induction program for predicting the risk of mortality in patients with community–acquired pneumonia. They developed a graphical data exploration tool for domain experts to encode domain knowledge and interact with the data mining process. The domain experts participated in two stages of mining. They were first asked to modify the existing set of attributes according to their domain knowledge, and then they were prompted with mining results and were able to modify the mined models (rules) directly. The experiment contained an experimental where the domain knowledge was incorporated as mentioned above, and a control group without domain knowledge. The experimental group performed significantly better (lower percent mean error) than the control group.

Sinha and Zhao [27] conducted an extensive comparative study on the performance of seven data mining classification methods—naive Bayes, logistic regression, decision tree, decision table, neural network, k-nearest neighbor, and support vector machine—with and without incorporating domain knowledge. The application they focused on was in the domain of indirect bank lending. An expert system capturing a lending expert's knowledge of rating a borrower's credit is used in combination with data mining to study if the incorporation of domain knowledge improves classification performance. In their study, the domain knowledge used was partial, meaning that it could only lead to intermediate results but was not sufficient to make the final prediction. They cascaded the expert system with a data mining classifier. The experiment adopted an experimental vs. control paradigm, similar to Ambrosino et al.'s early experiment in 1999. The prediction proposed by the expert system was added to other inputs. Classifiers built using input data enhanced by the expert system's output formed the experimental group. For the control group, classifiers were built using the original set of input attributes (bypassing the expert system). Their results showed that incorporation of domain knowledge significantly improves classification results with respect to both misclassification cost and AUC (Area Under Curve). Hence they concluded by calling for more attention in combining domain knowledge and data mining. They articulated that, in knowledge engineering, the focus is on the knowledge of a human expert in a specific problem area. On the other hand, the focus of data mining is on the data available in an organization. Expert systems and data mining methods could play complementary roles in situations where both knowledge and data are available.

Hirsh and Noordewier [4] argued that if learning is to be successful, the training data must be encoded in a form that lets the learner recognize underlying regularities. The application domain they focused on was the problem of DNA sequence classification. They proposed to use background knowledge of molecular biology to re-express data in terms of higher-level features that molecular biologists use when discussing nucleic-acid sequences. The high level features were Boolean valued, representing the presence or absence of the feature in a given DNA sequence. Using C4.5 decision trees and backprop neural networks, they conducted experiments with and without the higher-level features. For both learning methods, the use of higher-level features resulted in significantly lower error rates.

Pohle [28] contended that data mining techniques are good at generating useful statistics and finding patterns in large volumes of data, but "they are not very smart in interpreting these results, which is crucial for turning them into interesting, understandable and actionable knowledge." The author viewed the lack of sophisticated tool to support incorporating human domain knowledge into the mining process as being the main factor responsible for the limitation. They also pointed out that ontologies were valuable technologies to incorporate domain knowledge and thus they propose to exploit ontologies when integrating knowledge mined from knowledge discovery process to an existing knowledge base.

Kopanas *et al.* [7] conducted large scale data mining experiment exploring the role of domain knowledge in different phases of a large-scale data mining project, using a case study of customer insolvency in the telecommunication industry. They argued against the claim that data mining approaches eventually will automate the process and lead to discovery of knowledge from data with little or no support of domain experts and domain knowledge. For each stage in data mining they identified types of domain knowledge involved to either improve the performance or, in some case, make data mining process possible at all. They found that though domain knowledge plays a critical role mainly in the initial and final phases of the project, it influences the other phases to some degree as well. For example, in the problem definition stage, domain knowledge involves business and domain requirements and other implicit, tacit knowledge. In the data preparation stage, the useful domain knowledge involves semantics of corporate database. In the data preprocessing and transformation stage, domain knowledge includes tacit and implicit knowledge for inferences. In feature and algorithm selection stage, main type of knowledge involves how to interpret selected features. In mining stage, domain knowledge focuses on inspection of discovered knowledge. In the evaluation stage, domain knowledge defines performance criteria related to business objectives. In the fielding knowledge base stage (incorporating mined knowledge with an existing knowledge base), domain knowledge provides supplementary information for implementing the fusion.

In another study, Weiss *et al.* [5] combined an expert system with a data mining method for generating better sales leads. They developed an expert system that interviews executives of small and medium-sized companies and, based on their responses, recommends promising sales leads. The question-answer pairs and the recommended solutions were stored as examples to be mined by the method of rule induction. The study demonstrated how a knowledge base can be used to guide a machine learning program. The techniques developed in the study would be useful for consultation systems whose questions have different costs of acquisition.

Daniels *et al.* [6] demonstrated that data mining systems can be successfully combined with explicit domain knowledge. They pointed out that in theory there are two extreme situations that may occur with respect to the availability of domain knowledge. The first is when no prior knowledge that is available. The second is when all relationships are known with certainty, up to a limited number of parameters. They then claimed that their study was positioned somewhere between these extremes. The authors focused on a special type of a priori knowledge, monotonicity, i.e., the sign of relationship between the dependent and independent variables, for economic decision problems. Prior knowledge was implemented as monotonicity constraints in the decision tree and neural network classifiers. Addition of the knowledge resulted in smaller decision trees, and smaller variations of error on the training and test sets for neural networks. The authors also claimed that the framework developed might serve as a tool to implement normative requirements. However, since monotonicity constraints were incorporated in the decision tree and neural networks by designing specific algorithms, it is not obvious how to generalize the algorithm design process to include other normative domain knowledge.

Yoon *et al.* [29] studied semantic query optimization, a field that endeavors to optimize data mining queries by taking advantage of domain knowledge. The authors demonstrated that significant cost reduction can be achieved by reformulating a query into a less expensive yet equivalent query that produces the same answer as the original one. They identified that in most cases, exhaustive analysis of data is infeasible. It is often necessary to perform a relatively constrained search on a specific subset of data for desired knowledge. The domain knowledge they utilized was classified into three categories, interfiled, category, and correlation, all of which can be represented in rule forms. When a data mining query is received, they first identify domain knowledge that is relevant to the query, and transform it accordingly. On the other hand, to select relevant domain knowledge without an exhaustive search of all domain knowledge, they developed a method that built tables for domain knowledge indexed by attributes.

Vikram and Nagpal [30] developed an iterative association rule mining algorithm to integrate user's domain knowledge with association rule mining. The knowledge they request from the users is attributes of interest. According to users' specification, database is scrutinized to produce a working subset that only contains the attributes of interest while the rest are excluded. With this dataset, the Apriori procedure searches for frequent large itemsets. The advantage is apparent since irrelevant records are filtered out, the result is more meaningful and the running time is also reduced.

We summarize the above surveyed research systems in Table 2.1. Each system is characterized by 1) its domain of application, 2) type of domain knowledge employed, 3) usage of domain knowledge, and 4) data mining techniques that are adapted to incorporate the domain knowledge.

Next, we describe another line of research on using domain knowledge encoded in ontologies.

Staab and Hotho [31] presented an ontology-based text clustering approach. They developed a preprocessing method, called COSA, one of the earliest to utilize the idea of mapping terms in the text to concepts in the ontology. The authors pointed out that the size of the high-dimensional term vector representation of the text document is the principal problem faced by previous algorithms. By mapping terms to concepts, it essentially aggregates terms and reduces the dimensionality.

The mapping of terms to concepts can be also seen as a process of semantic annotation. It was realized in COSA by using some shallow and efficient natural language processing tools. After the mapping process, COSA further reduced the dimensionality by aggregating concepts using the concept heterarchy defined in the "core ontology" used in their framework. The concept heterarchy should be thought of as equivalent to the subsumption hierarchy (taxonomy) in OWL. The idea was navigating the hierarchy top-down splitting the concepts with most support (number of mapping terms) into their sub-concepts and abandoning the concepts with least support. COSA pioneers in incorporating ontology in text clustering and displays some generality over the confines of any specific domain.

System	**Problem domain**	**Type of domain knowledge**	**Usage of domain knowledge**	**Data mining method**
Daniels *et al.* [6]	Business Intelligence	Monotonicity constraints	modify mining algorithms to embody the knowledge directly	Decision tree and neural network
Ambrosino *et al.* [26]	Medical decision	Attribute-relation, interpretation of result	Experts interact directly with mining in both pre– and post– processing stages	Decision tree
Pazzani *et al.* [25]	Predicate learning	Taxonomy, attribute-relation rules, attribute correlations	Preprocessing data	FOCL
Sinha *et al.* [27]	Business Intelligence	Expert rules	Rule's prediction cascaded as an input to classifier	Seven typical classification algorithms
Yoon *et al.* [29]	Query optimization	Taxonomy, attribute relation rules and correlation	Transform data mining queries	Not specified
Hirsh *et al.* [4]	DNA sequence classification	Attribute relation rules	Forming new set of attributes	C4.5 and neural network
Vikram *et al.* [30]	Association rule mining	Attribute of interest	Preprocessing data	Association rules
Weiss *et al.* [5]	Consultation	Question-answer pairs derived from interviewing experts	Question-answer pairs serve as part of the input to a mining system	No restriction
Kopanas *et al.* [7]	Business intelligence	Comprehensive information pertaining to a domain	For each stage of mining, discussing the use of certain type of domain knowledge in general	No restriction

TABLE 2.1. Summary of systems that employ domain knowledge.

Wen *et al.* [32] devised a framework that solved the genomic information retrieval problem by using ontology-based text clustering. The core idea was an extension to COSA. Documents containing genomic information were first annotated based on UMLS so that the terms are mapped to concepts. Then the authors pointed out that even the dimension of clustering space is dramatically reduced, there still exists the problem that a document is often full of class-independent "general" words and short of class-specific "core" words, which leads to the difficulty of document clustering because class-independent words are considered as noise in clustering. To solve this problem, the authors proposed a technique for concept frequency re-weighing which took into consideration the concept subsumption hierarchy defined in the domain ontology. Finally, from the re-weighed concept vector representation, a cluster language model could be generated for information retrieval.

Fang *et al.* [33] proposed an ontology-based web documents classification and ranking method. The contribution of this work was the introduction of a way to automatically augment or tailor the existing ontology to fit the specific purpose, while in previous work one had to either manually create an ontology from scratch or adopt some well established domain ontology. Their technique was to enrich a certain ontology using terms observed in the text document. This was done with the help of WordNet [34]. Specifically, for example, if the sense of a term appears to be a synonym of the sense of a concept according to WordNet, the term is added to the ontology as a sibling of the concept. The enriched ontology is then treated as a representation of the category to which some text document is classified. The proposed classification was done by simply comparing the similarity between ontologies and the term vector representing the text document. This implied that first, multiple ontologies should be provided for choice, and second, for each category of the corpus there should be one corresponding ontology. These assumptions appeared cumbersome though the authors pointed to the Open Directory Project as a source for initial ontologies in their experiment. Moreover, this process did not fit into traditional classification as there was no training phase. It was more similar to clustering with known number of clusters.

Cheng *et al.* [35] studied document clustering problem as a means to efficient knowledge management. They utilized ontologies to overcome the ambiguity problem frequently seen in natural language since "an ontology includes a selection of specific sets of vocabulary for domain knowledge model construction, and the context of each vocabulary is represented and constrained by the ontology." They developed a system called Ontology-based Semantic Classification (OSC) Framework that consisted of two main components: Content-based Free Text Interpreter (CFTI) and Context-based Categorization Agent (CCA). CFTI leveraged on the link grammar capability for syntactical analysis of a sentence. At the same time, the lexical meaning analysis of a sentence was supported through the integration with ontological models such as the WordNet. The context models produced from CFTI correlated the content of a particular document with the context of the user. The role of the CCA was to further enhance the usability of these context models by classifying them according to the user interest. The OSC framework seemed appealing but the authors did not provide any implementation details nor experiment results. It was more of a research proposal and it would be interesting to see the performance of the system when the authors make the proposal a reality.

Taghva *et al.* [36] reported on the construction of an ontology that applied rules for identification of features to be used for an email classification system, called "Ecdysis." The ontology was designed for the purpose of encoding expert rules deciding the email category. Therefore it contained only those concepts that were aspects of such rules. CLIPS was used to implement rules and the inference with rules was based on a "match-and-fire" mechanism: One or more features of an email instance would be matched with instances of classes from the ontology. If there was a successful match, then the rule would fire, causing the email to have some certain feature. This feature became one of many that could be used for training and classification with a Bayesian classifier. The authors claimed that preliminary tests showed that these additional features enhanced the accuracy of the classification system dramatically.

Tenenboim *et al.* [37] proposed an automatic method for classification of news using hierarchical news ontology. The system they developed was called "ePaper." It was designed to aggregate news items from various news providers and deliver to each subscribed user a personalized electronic newspaper, utilizing content-based and collaborative filtering methods. The ePaper could also provide users "standard" (*i.e.*, non-personalized) editions of selected newspapers, as well as browsing capabilities in the repository of news items. The classification task performed in the ePaper system aimed at classifying each incoming news document to one or several concepts in the news ontology. In this sense, only the target classes in the classification process were annotated by ontological terms. Since the users' profiles were also defined using the same set of ontological terms, a content-based filter was able to compare the similarity between a user's profile and classified categories of news. Based on results of the classifier and content-based filter, the personalization engine of the system was able to provide a personalized paper.

Lula *et al.* [38] proposed an ontology-based cluster analysis framework. They discussed various aspects of similarity measure between objects and sets objects in an ontology-based environment. They devised an ontology-based aggregation function to calculate similarity between two objects which takes into account taxonomy similarity, relationship similarity and attribute similarity. For example, path distance, Jaccard coefficient and measures based on information theory can be used to calculate the taxonomy similarity. Relationship similarity can be determined by calculating similarity of objects that participate in the relationship. Attribute similarity can be determined by comparing values of the attributes. The authors claimed that the framework with ontology-based similarity measure opened the possibility for various clustering application. But apparently much work still remained. It was unclear how the aggregation function was defined though each of its components could be solved separately. A proper aggregation was highly possible to be application-specific, which might suggest the need of a learning framework to derive such function.

Li *et al.* [39] developed a new decentralized P2P architecture-ontology-based community overlays. The system exploited the semantic property of the content in the network to cluster nodes sharing similar interest together to improve the query and searching performance. To do that, they proposed a query routing approach that organized nodes into community overlays according to different categories defined in the nodes' content ontology. A community overlay was composed of nodes with similar interest. Queries were only forwarded to semantically related overlays, thus alleviating the traffic load. According to taxonomic information in the ontology, peers (nodes) could be clustered into ontological terms. This study introduced a new data mining application besides text document clustering. But their principle remained the same as other related work: ontology is used as an abstraction to data. By incorporating ontologies, some performance metrics of the data mining task can be improved.

Adryan *et al.* [40] developed a system called GO-Cluster which used the tree structure of the Gene Ontology database as a framework for numerical clustering, and thus allowing a simple visualization of gene expression data at various levels of the ontology tree. Shen *et al.* [41] proposed a new method of association rules retrieval that was based on ontology and Semantic Web. They argued that ontology-based association rules retrieval method can better deal with the problems of rule semantics sharing, rule semantics consistency and intelligibility.

In Table 2.2, we summarize the surveyed data mining systems that make use of ontologies. The table indicates how the solution space is covered by different systems. It shows a large fraction of systems are in the domain of text mining. Most of them make use of taxonomic information provided by ontologies. Only two systems consider incorporating rules. Most systems adopt readily available domain ontologies, while Fang et al's approach can create ontologies on the fly. We also notice that all systems are limited in that they only deal with unstructured input. The importance of automated semantic annotation is generally overlooked in most work.

System	**Ontology construction**	**Annotation method**	**Type of sources**	**Data mining method**
Staab *et al.* (COSA) [31]	Manual creation	Shallow NLP method	Text	Clustering based on "bag-of-concept" representation plus concept aggregation
Wen *et al.* [32]	Off-the-shelf (UMLS)	Manual	Text	Clustering based on "bag-of-concept" representation plus concept frequency reweighing
Fang *et al.* [33]	Manual creation of "core" ontology and update on the fly	Manual	Text	Clustering based on "bag-of-concept" representation plus feed back to enrich ontology
Cheng *et al.* (OSC) [35]	Off-the-shelf (WordNet)	Rule-based NLP	Text	Not specified
Taghva *et al.* (Ecdysis) [36]	Manually creation, incorporated with a rule inference system	Manual	Email / text	Classification with additional features derived from rules
Tenenboim *et al.* [37]	Manual creation	Manual	News archive /text	Not specified
Lula *et al.* [38]	Not specified	Manual	Text	Hierarchical agglomerative clustering
Li *et al.* [39]	Off-the-shelf (Open Directory Project)	Manual	P2P user / resource profile data	Not specified
Adryan *et al.* [40]	Off-the-shelf (Gene Ontology)	Manual	Gene expressions	Hierarchical clustering with instance regrouping based on GO annotation

TABLE 2.2. Summary of ontology-based data mining systems.

2.2. Graph-based Approach for Knowledge Representation

Graph-based approaches for representing knowledge have long been used in philosophy, psychology, and linguistics. The computer counterpart to this means is the so-called *semantic network* that represents knowledge in patterns of interconnected nodes and arcs which were first developed for artificial intelligence and machine translation.

The semantic network, and graph-based approaches for knowledge representation in general, are motivated by the desirable qualities of graph for both modeling and computation. From a modeling viewpoint, basic graphs are easily understandable by users, and it is always possible to split up a large graph into smaller ones while keeping its semantics. From the computational viewpoint, the graph is one of the most studied objects in mathematics. Considering graphs instead of logical formulas provides another view of knowledge constructs (*e.g.*, some notions like path, cycle, or connected components are natural on graphs) and provides insights to algorithmic ideas [42]. In light of these motivations, what is common to all semantic networks is a declarative graphic representation that can be used either to represent knowledge or to support automated systems for reasoning about knowledge.

According to Sowa [43], the following are six of the most common kinds of semantic networks.

1. Definitional networks focus on the is-a or subtype relation among concepts. The resulting network, also called a generalization or subsumption hierarchy, supports the rule of inheritance to propagate properties from a supertype to all of its subtypes. The information in these networks is often assumed to be necessarily true.

2. Assertional networks are designed to assert propositions. Unlike definitional networks, the information in an assertional network is assumed to be contingently true, unless it is explicitly marked with a modal operator. Some assertional networks have been proposed as models of the conceptual structures underlying natural language semantics.

3. Implicational networks use implication as the primary relation for connecting nodes. They may be used to represent patterns of beliefs, causality, or inferences.

4. Executable networks include some mechanism, such as marker passing or attached procedures, which can perform inferences, pass messages, or search for patterns and associations.

5. Learning networks build or extend their representations by acquiring knowledge from examples. The new knowledge may change the old network by adding and deleting nodes and arcs or by modifying numerical values, called weights, associated with the nodes and arcs.

6. Hybrid networks combine two or more of the previous techniques, either in a single network or in separate, but closely interacting networks.

Knowledge such as subsumption hierarchy is best captured by definitional networks. Distance (similarity) measures can usually be reasonably defined on such network, which is essential in many data mining tasks. It is possible to extend data mining algorithms that depend on analyzing distances between entities in factual knowledge to work with distances between those in ontological knowledge.

In addition, one of the most prominent knowledge representation formalism families among current systems, description logics, formerly called terminological logics or concept languages, have been a successful attempt to combine well-defined logical semantics with efficient reasoning [43]. They are derived from an approach proposed by Woods [44] and implemented by Brachman [45] in a system called Knowledge Language One (KL-ONE). Recent description logics are DAML+OIL [46] and its successor OWL [10], which are intended for representing knowledge in the Semantic Web [9]—a giant semantic network that spans the entire Internet.

2.2.1. Graph Representation of RDF

According to the W3C specification for the RDF semantics [47], an RDF graph, or simply a graph, is defined as a set of RDF triples. A subgraph of an RDF graph is a subset of the triples in the graph. A triple is identified with the singleton set containing it, so that each triple in a graph is considered to be a subgraph. A proper subgraph is a proper subset of the triples in the graph. A ground RDF graph is one with no blank nodes. RDF triples can be visualized as a *directed labeled graph* (see details in Chapter III). The directed labeled graph model for RDF is straightforward and convenient in most cases. But inconsistency arises when using triples to make assertions on predicates. The directed labeled graph model of RDF makes the artificial distinction between resources and properties. The results of the understanding of RDF bounded by this model becomes especially evident in the limitations of current RDF query languages as studied in [48].

A hypergraph [49] is a generalization of a traditional graph where edges, called hyperedges, can connect more than two vertices. If each edge in a hypergraph covers the same number of nodes, it is called r-uniform hypergraph, r being the number of nodes on each edge.

Hayes has proposed to use hypergraphs to represent RDF [20]. In his proposal, any RDF graph can be represented by a simple ordered 3-uniform hypergraph, in which an RDF triple corresponds to a hypergraph edge, the nodes being the subject, predicate and object in this order. In this way, both meta-data and data level statements can be integrated in a consistent model. This result constitutes an important aspect of the theoretical basis of our proposed graph representation for the combined information source of both data and knowledge.

Definition 2.1 (Hypergraph). Formally, a hypergraph $G = (V, E)$, is a pair in which V is the vertex set and E is the hyperedge set where each $e \in E$ is a subset of V. A weighted hypergraph is a hypergraph that has a positive number $w(e)$, called the weight of a hyperedge e, associated with each hyperedge. We denote a weighted hypergraph by $G = (V, E, w)$. The degree of a vertex $v \in V$, $d(v)$, is defined as $d(v) = \sum_{e \in adj(v)} w(e)$, where $adj(v)$ denotes the set of edges that are adjacent to v. The degree of a hyperedge e, denoted as $\delta(e)$, is the number of vertices in e, *i.e.*, $\delta(e) = |e|$. A hyperedge e is said to be incident with a vertex v when $v \in e$. The hypergraph incidence matrix $\mathbf{H} \in \mathbb{R}^{|V| \times |E|}$ is defined as

$$h(v, e) = \begin{cases} 1, & v \in e \\ 0, & otherwise \end{cases}$$

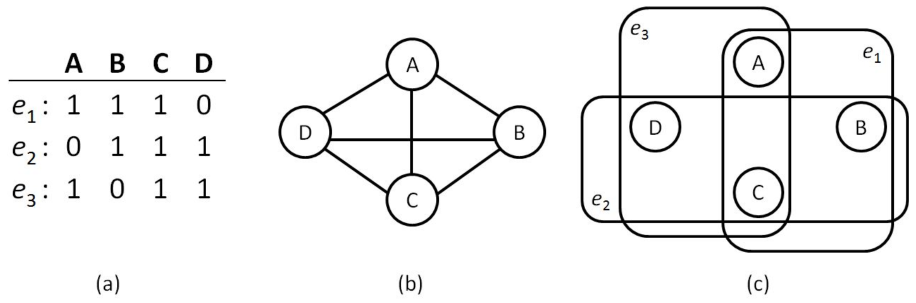
Throughout the rest of the dissertation, the diagonal matrix forms for $\delta(e)$, $w(e)$, $d(v)$ are denoted as $\mathbf{D}_e$, $\mathbf{W} \in \mathbb{R}^{|E|}$, and $\mathbf{D}_v \in \mathbb{Z}^{|V|}$, respectively.

2.3. Graphs in Data Mining

2.3.1. Graph Representation of Relational Structure

An object set endowed with pairwise relationships can be naturally illustrated as a graph in which vertices represent objects, and any two vertices that have some kind of relationship are joined together by an edge. In the case of frequent itemset mining, a set of objects with the co-occurrence relationship can be represented as directed or undirected graphs.

For illustrating this point of view, let us consider a relational table depicted in Figure 2.1(a). One can construct an undirected graph where the set of vertices is the set of relational attributes (column items) and an edge joins two vertices if the they co-occur in a tuple (as illustrated in Figure 2.1(b)). This graph is called the *Gaifman graph* [50] of a relational structure. The undirected graph can be further enriched by assigning to each edge a weight equal to the support of the 2-itemset consisting of vertices incident to the edge. Cliques (complete subgraphs) in the Gaifman graph, or *Gaifman cliques* for short, are of particular interest because every tuple (ground atom) in data corresponds to a Gaifman clique. However, ambiguity arises as not all Gaifman cliques have matching tuple in the data. There exists cases where cliques are incidental in the sense that several relational ground atoms play together to induce a clique configuration in the Gaifman graph, but no ground atom covers the entire clique (e.g., the clique of $\{A, B, C, D\}$ in Figure 2.1(b) does not correspond to any tuple in the relational table). Further more, given the Gaifman graph, we lose the information of how nodes are related. For example, if A, B and C are products purchased by a particular customer as indicated by a record in the transactional table, this information is no longer available in the graph.



	A	B	C	D
e_1:	1	1	1	0
e_2:	0	1	1	1
e_3:	1	0	1	1

FIGURE 2.1. An example of simple graph vs. hypergraph for representing a relational table: (a) the transaction table; (b) the Gaifman graph representation of the table; (c) The hypergraph representation of the table

A natural way to remedy the ambiguity is to represent the relational data as a hypergraph (see Section 2.2.1 for the definition). An edge in the hypergraph, or hyperedge, can connect more than two vertices. In other words, every hyperedge is an arbitrary nonempty subset of vertices. It is obvious that a simple graph is a special kind of hypergraph. In Chapter III, we propose to employ hypergraphs to model relational structure. In Chapter V and VI we describe ways to find semantically associated itemsets using hypergraphs. For example, we can construct a hyperedge for each tuple in the relational table. The relational attributes constitute the universe of vertices in the hypergraph. In this representation, each hyperedge has an exact one-to-one correspondent tuple (see Figure 2.1(c), for example).

2.3.2. Graph-based similarity

Data mining algorithms rely on the notion of similarity between data points to make meaningful inferences. When data is in $\mathbb{R}^d$, the standard similarity measure is the Euclidean distance. When data has an explicit link structure, shortest path distance is commonly used. However, neither of these measures incorporates the intuition that two data points are similar to each other if they are connected by a high density region. This latter concept of similarity measure has been shown in experiments to lead to significant improvement in a number of learning tasks, see, for example, [51–53].

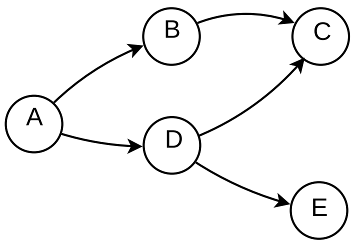


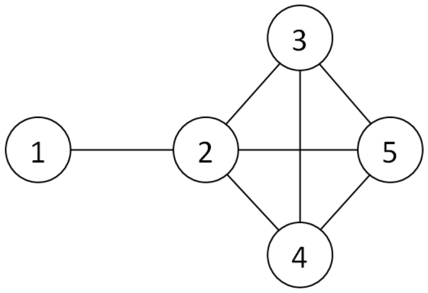
FIGURE 2.2. A simple graph of friendship relationship.

Take the simple graph in Figure 2.2, for example, suppose given a task of friend recommendation based on the information in this graph, the interesting question is whether C or E is a better choice of recommendation to A. To answer this question, it is natural to compare the similarity measures $s(A, C)$ and $s(A, E)$. In a rough sense, on can identify in the graph that there are two paths between A and C, while only one between A and E. It's intuitive to conclude that A and C are more similar, or closer, than A and E. This gives us a hint that meaningful similarity measures on graphs should satisfy the following two desired properties:

1. The more paths connecting two nodes, the closer they are.

2. The shorter the paths, the closer they are.

In other words, the more "short" connections between two given nodes, the more similar those nodes are. To this end, in Chapter V and VI, we propose to employ several quantities that satisfy these properties based on the concept of random walk. In the following example, we quantitatively show the property of random walk commute time distance, which characterizes the expected number of steps to take a round trip between a starting node and a target node.



	Euclidian Distance					Commute Distance				
Index	1	2	3	4	5	1	2	3	4	5
1	0	1	1.85	1.85	2.41	0	12.83	19.79	19.79	20.34
2	1	0	1	1	1.41	12.83	0	6.96	6.96	7.51
3	1.85	1	0	1.41	1	19.79	6.96	0	7.51	6.96
4	1.85	1	1.41	0	1	19.79	6.96	7.51	0	6.96
5	2.41	1.41	1	1	0	20.34	7.51	6.96	6.96	0

FIGURE 2.3. A comparison between the Euclidean and the commute time distance.

Figure 2.3 shows a graph of five nodes with a specific edge configuration (the so-called "lollipop graph"). The Euclidean distances between each pair of nodes are shown in the left-hand side of the corresponding table above and the respective commute time distances are shown on the right-hand side. It can be seen that node 1 and node 3 are equally close to node 2 in terms of their Euclidean distances. However, node 2 and 3 are considered much closer under commute time distance because they are within a much more densely connected subgraph. This shows that, unlike Euclidean distance and shortest path distance, commute time distance between two nodes captures both the length of paths between them and their local neighborhood densities. We also explore other random walk-based measures including the pseudoinverse of the Laplacian matrix and the stationary probability that are closely related to commute time distance. In the following, we describe random walk on simple graphs, and the extension of random walk to hypergraphs is presented in Chapter V.

2.3.2.1. Random Walks on simple graphs:

Given a graph and a starting point we select a neighbor of it at random and move to this neighbor then we select a neighbor of this point at random and move to it etc. The random sequence of points selected this way is a random walk on the graph. In other words, a random walker can jump from vertex to vertex and each vertex therefore represents a state of the Markov chain. The average first-passage time $m(k|i)$ [54] is the average number of steps needed by a random walker for reaching state k for the first time, when starting from state i. The symmetrized quantity $n(i, j) = m(j|i) + m(i|j)$ called the average commute time [54], provides a distance measure between any pair of states. The fact that this quantity is indeed a distance on a graph was proved independently by Klein and Randic [55] and Gobel and Jagers [56].

The Laplacian matrix $\mathbf{L}$ of a graph is widely used for finding many properties of the graphs in spectral graph theory. Given node degree matrix $\mathbf{D}$ and graph adjacency matrix $\mathbf{A}$, the Laplacian matrix of the graph is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. The normalized Laplacian is given by $\mathbf{L}_N = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, where $\mathbf{I}$ is the identity matrix. The average commute time $n(i, j)$ can be computed in closed form from the Moore-Penrose pseudoinverse of $\mathbf{L}$ [57], denoted by $\mathbf{L}^{+}$.

Various quantities derived from random walk on graph has been used in a number of applications. Fouss et al. [58] compared twelve scoring algorithms based on graph representation of the database to perform collaborative movie recommendation. Pan et al. [59] developed a similarity measure based on random walk steady state probability to discover correlation between multimedia objects containing data of various modalities. Yen et al. [60] introduced a new k-means clustering algorithm utilizing the random walk average commute time distance. Zhou et al. [61] presented a unified framework based on neighborhood random walk to integrate structural and attribute similarities for graph clustering.

2.4. Integration of Heterogeneous Information Sources

This section describes various background areas related to the contributions of the matching work based on metaheuristics optimization in Chapter IV that focuses on resolving heterogeneities in schema/ontologies as well as enabling cross dataset meta-analysis.

2.4.1. The Multiobjective Optimization Problem and Pareto-Optimality

Multi-objective optimization problem (also called multi-criteria, multi-performance or vector optimization) can be defined mathematically as to find the vector $X = [x_1, x_2, \ldots, x_k]^T$ which satisfies the following m inequality constraints and l equality constraints:

$$g_i(X) \geq 0, i = 1, 2, \ldots, m$$

$$h_i(X) = 0, i = 1, 2, \ldots, l$$

and optimize the objective function vector

$$F(X) = [f_1(X), f_2(X), \ldots, f_N(X)]^T$$

where $X = [x_1, x_2, \ldots, x_k]^T$ is called the decision variable vector.

Real-life problems require simultaneous optimization of several incommensurable and often conflicting objectives. Usually, there is no single optimal solution, but there is a set of alternative solutions. These solutions are optimal in the sense that no other solutions in the search space are superior to each other when all the objectives are considered [62]. They are known as Pareto-optimal solutions. To define the concept of Pareto optimality, we take the example of a minimization problem with two decision vectors $a, b \in X$. Vector a is said to dominate b if

$$\begin{aligned} &\forall i = \{1, 2, \ldots, N\} \ : \ f_i(a) \leq f_i(b) \\ &and \\ &\exists j = \{1, 2, \ldots, N\} \ : \ f_j(a) < f_j(b). \end{aligned}$$

When the objectives associated with any pair of non-dominated solutions are compared, it is found that each solution is superior with respect to at least one objective. The set of non-dominated solutions to a multi-objective optimization problem is known as the Pareto-optimal set (Pareto front) [63].

2.4.1.1. Metaheuristics on Solving Multi-Objective Optimization Problems

Metaheuristics are used for combinatorial optimization in which an optimal solution is sought over a large, discrete search-space. Popular metaheuristics for combinatorial problems include simulated annealing by Kirkpatrick et al. [64], and genetic algorithms by Holland et al.[65]. Extensive previous research has been devoted to extend these methods to multi-objective optimization problems as discussed in the following, which yield sets of mutually non-dominating solutions that are an approximation to the true Pareto front.

Simulated Annealing in Multi-Objective Optimization: Simulated annealing is based on an analogy of thermodynamics with the way metals cool and anneal. It has been proved to be a compact and robust technique. Simulated annealing was started as a method or tool for solving single objective combinatorial problems, these days it has been applied to solve single as well as multiple objective optimization problems in various fields. A comprehensive survey can be found in [62].

Evolutionary Multi-Objective Optimization: Evolutionary multi-objective optimization covers the use of many types of heuristic optimizers inspired by the natural process of evolution. As in nature, a population of individuals (solutions to the problem) exist and, through a process of change and competition between these individuals, the quality of the population is advanced. Deb [66] provides an introduction of evolutionary algorithms (e.g., genetic algorithm) for multi-objective as the state of the art.

2.4.2. The Schema Matching Problem

Our study of matching alternative attribute sets is closely related to the schema matching problem in data integration. According to the type of instance value, various instance-based approaches have been developed in previous research. For example, for textual attributes, a linguistic characterization based on information retrieval techniques can be applied [67]; for nominal attributes, evaluation of the degree of overlap of instance values is a preferred approach. Larson et al. [68] and Sheth et al. [69] discussed how relationships and entity sets could be integrated primarily based on their domain relationships. Similarity of partially overlapped instance set can be also calculated based on measures such as Hamming distance and Jaccard coefficient; for numeric attributes, most methods use aggregated statistics to characterize the attributes, e.g., 'SSN' and 'PhoneNo' can be distinguished based on their respective patterns [67]. Hybrid systems that combine several approaches to determine matching often achieve better performance. For example, SemInt [70] is a comprehensive matching prototype exploiting up to 15 constraint-based and 5 content-based matching criteria. The LSD (Learning Source Descriptions) [71] system uses several instance-level matchers (learners) that are trained during a preprocessing step. The iMAP [72] system uses multiple basic matchers, called searches, e.g., text, numeric, category, unit conversion, each of which addresses a particular subset of the match space.

2.4.3. The Cluster Matching Problem

In framing our solution to the schema matching problem, in Chapter IV, we also aim at addressing another challenging task, namely, the problem of finding correspondences among distinct patterns that are observed in different experiments. This is to enable meta-analysis across mining results derived from different sites.

This work is motivated by the problem in our collaborative cross-lab neuroscience ERP (Event Related Potential) pattern analysis [19, 73]. Due to the data-driven strategy we adopt to extract ERP patterns from data, it is natural to formulate the pattern matching problem as the cluster comparison problem. To represent clusterings in a way that meaningful similarity measure can be defined, we choose a clustering representation called density profiles proposed by Bae et al. [74] and a clustering similarity index known as ADCO (Attribute Distribution Clustering Orthogonality). The definition of density profile and the ADCO method are briefly described in the following.

Density Profile: To represent clusters using density profiles, the attribute's range in each cluster is first discretized into a number of bins, and the similarity between two clusters corresponds to the number of points of each cluster falling within these bins. The formal definition for this number of points is the *density* of an attribute-bin region for cluster c_k in clustering C, denoted as $dens_C(k, i, j)$. It refers to the number of points in the region (i, j)—the j-th bin of the i-th attribute—that belongs to the cluster c_k. For example, for clustering C in Fig. 2.4, $dens_C(1, 1, 1) = 8$, because there are 8 data points in region $(1, 1)$—the first bin of the first attribute x—that belongs to the first cluster c_1.

The density profile vector V_C for a clustering C is formally defined as an ordered tuple:

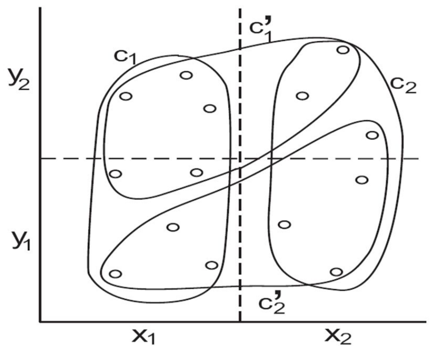


FIGURE 2.4. An example of cluster density profiles: Two clusterings C=$\{c_1, c_2\}$ and C'=$\{c'_1, c'_2\}$. Two attributes X (attribute 1) and Y (attribute 2) are discretized into 2 bins each. See [74] for details.

$$\begin{aligned} V_C = \Big[& dens_C(1,1,1),\ \ldots,\ dens_C(1,1,Q), \\ & dens_C(1,2,1),\ \ldots,\ dens_C(1,M,Q), \\ & dens_C(2,1,1),\ \ldots,\ dens_C(N,M,Q) \Big], \end{aligned} \qquad \text{(Equation 2.1.)}$$

where Q is the number of bins in each of the M attributes, and N is the number of clusters in C.

The ADCO measure: After the density profile vectors of two clusterings C and C' are obtained, the degree of similarity between C and C' can be determined by calculating the dot product of the density profile vectors: $sim(C, C') = V_C \cdot V_{C'}$.

The $ADCO(C, C')$ measure is defined as $sim(C, C')$ normalized by the maximum achievable similarity when using either of the two clusterings:

$$ADCO(C, C') = \frac{sim(C, C')}{NF(C, C')}, \qquad \text{(Equation 2.2.)}$$

where $NF(C, C') = max\big[sim(C, C),\ sim(C', C')\big]$.

CHAPTER III

GRAPH REPRESENTATION

The synergy between domain knowledge and data mining can be achieved by employing the RDF model given the fact that RDF allows a combined specification of both schema information and data structured under the schema. In light of Hayes et al's proposal to represent RDF as hypergraphs [20], we develop a set of rules to represent data in transactional tables as hypergraphs or bipartite graphs with minimal loss of semantics. We then propose a novel way to combine the graph representations of data and domain knowledge encoded in ontologies as a unified information source from which valuable insights can be drawn upon.

3.1. Graph Representation for Domain Knowledge

As we have mentioned in Chapter II, graph-based approaches for representing knowledge have long been used in philosophy, psychology, and linguistics. The computer counterpart to this means is the so-called *semantic network* that represents knowledge in patterns of interconnected nodes and arcs which were first developed for artificial intelligence and machine translation [43]. Knowledge such as subsumption hierarchy can be best captured by the semantic network. Distance (similarity) measures can usually be reasonably defined on the network, which is essential in many data mining tasks. In addition, one of the most prominent formalism families among current systems, description logics, have been proven to be successful. Its latest development, OWL, is intended for representing knowledge in the semantic web [9]—a giant semantic network that spans the entire Internet. OWL ontologies can be used along with information written in RDF, and OWL ontologies themselves are primarily exchanged as RDF documents. RDF's abstract triple syntax has a graph nature.

We focus on describing various definitions of graph representation models for RDF in this section. The term "RDF graph" is formally defined as follows according to the W3C specification for RDF semantics [47]:

Definition 3.1 (RDF graph). An RDF graph is defined as a set of RDF triples. A subgraph of an RDF graph is a subset of the triples in the graph. A triple is identified with the singleton set containing it, so that each triple in a graph is considered to be a subgraph. A proper subgraph is a proper subset of the triples in the graph. A ground RDF graph is one with no blank nodes.

RDF triples can be visualized as a *directed labeled graph* as follows:

$$\boxed{\text{subject}} \xrightarrow{\textit{predicate}} \boxed{\text{object}},$$

where subjects and objects are represented as nodes, and predicates as edges. The directed labeled graph model for RDF is straightforward and convenient in most cases. But inconsistency arises when using triples to make assertions on predicates. The directed labeled graph model of RDF makes the artificial distinction between resources and properties, which may cause inconsistency in the graph representation. The following example illustrates this point of view.

Example 3.2 (Inconsistent representation of the RDF directed labeled graph model). In this example, a set of RDF statements is asserted to describe relationships among a group of people. The information expressed includes two different levels, *i.e.*, the meta (ontological) data level and factual data level. The factual data level consists of following statements: $\langle a\ collaborate\ b\rangle$, $\langle b\ coauthor\ c\rangle$, $\langle a\ influence\ d\rangle$ and $\langle d\ friendOf\ e\rangle$. The meta-data level contains

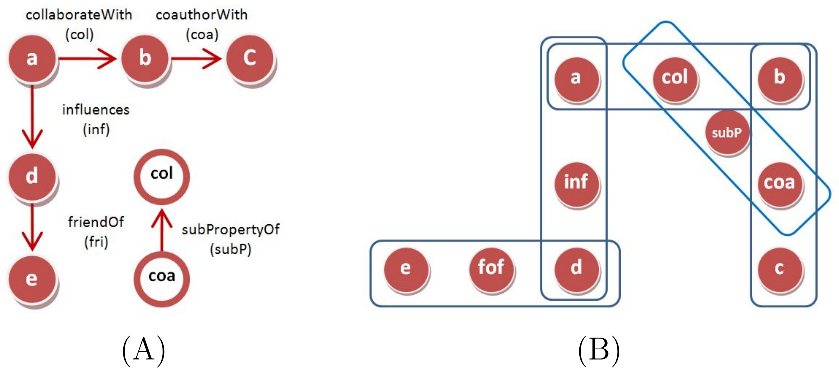


FIGURE 3.1. A comparison between a simple graph and a hypergraph. The figure shows an example of nodes connected by different links, represented by A) a simple graph, and B) a hypergraph.

one single statement asserting that *coauthor* is a sub-property of *collaboration*: $\langle$*coauthor subProperty collaboration*$\rangle$. In this case, the representation of *collaboration* and *coauthor* is inconsistent — they are represented as nodes at the factual data level and edges at the meta-data level (see Figure 3.1(A)).

To overcome the inconsistency, Hayes et al. [20] proposed to model RDF as a *hypergraph*. A hypergraph [49] is a generalization of a traditional graph where edges, called hyperedges, can connect more than two vertices. If each edge in a hypergraph covers the same number of nodes, it is called r-uniform hypergraph, r being the number of nodes on each edge. Any RDF graph can be represented by a simple ordered 3-uniform hypergraph, in which an RDF triple corresponds to a hyperedge, with incident nodes being the subject, predicate and object from the triple. In this way, both meta-data and data level statements can be integrated in a consistent model. In Fig. 3.1(B), the information in Example 3.2 is represented by a hypergraph and the inconsistency in the directed labeled graph representation is eliminated.

The formal definition of a hypergraph is given in Definition 2.1, Chapter II. Furthermore, a hypergraph $G = (V, E)$ can be transformed to a *bipartite graph* BG as follows:

Definition 3.3 (**Transformation from an RDF hypergraph (H) to an RDF bipartite graph (BG)**)**.** Let the node set V and edge set E from H be the two partitions the BG. The node pair (v_1, e_1) is connected by an edge if and only if vertex v_1 is contained in edge the e_1 in H. Conversely, any bipartite graph with fixed parts and no unconnected nodes in the second part has a corresponding hypergraph. This bipartite graph can be represented by an incidence matrix. Such matrix can be also viewed as a node adjacency (bi-adjacency) matrix of the bipartite graph.

$$\begin{array}{c} \\ E_1 \\ E_2 \\ E_3 \\ E_4 \\ E_5 \end{array} \begin{array}{c} \begin{array}{cccccccccc} \text{a} & \text{b} & \text{c} & \text{d} & \text{e} & \text{coa} & \text{col} & \text{inf} & \text{fof} & \text{subP} \end{array} \\ \left(\begin{array}{cccccccccc} 1 & 1 & & & & & 1 & & & \\ & 1 & 1 & & & 1 & & & & \\ 1 & & & 1 & & & & 1 & & \\ & & & 1 & 1 & & & & 1 & \\ & & & & & 1 & 1 & & & 1 \end{array}\right) \end{array}$$

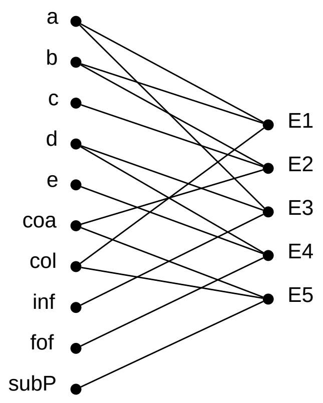


FIGURE 3.2. An example Incidence matrix representing the hypergraph of figure 3.1(B) and the corresponding incidence graph.

RDF bipartite graphs have many desirable properties for developing intuitive mining algorithms because they turn hypergraphs into a simple form so that many algorithms designed on simple graphs can be readily applied. Therefore, we propose to use bipartite graphs to represent domain knowledge and data expressed in RDF.

Example 3.4 (Hypergraph incidence matrix and corresponding bipartite graph). Figure 3.2 (A) shows the incidence matrix according to the hypergraph in Figure 3.1 from Example 3.2. And Figure 3.2 (B) shows the corresponding bipartite graph. Hypergraph incidence matrices represent membership of a node in an edge with a "1" in the corresponding entry.

Example 3.4 illustrates the general method that can be applied to all hypergraphs to transform to their bipartite graph form. In the case of a hypergraph representing an RDF graph, since nodes in an RDF statement are ordered (subject followed by predicate then object), this ordering must be preserved in the incidence matrix. A *labeled bipartite graph* can be derived to further capture the ordering and roles of nodes.

Definition 3.5 (RDF labeled bipartite graph). In the hypergraph incidence matrix, instead of using "1/0" according to the occurrence of a node in a hyperedge, we choose to label them by S, P or O to represent the role (subject, predicate, or object) of the node from the underlying RDF statement. Hence, when deriving the bipartite graph of a hypergraph incidence matrix, an edge is added for every S, P, O entry of the matrix, and this edge will be labeled with a corresponding character (S, P, or O). Thus, the only difference between the graph derived from the incidence matrix of any hypergraph and an RDF hypergraph is the fact that each edge has one out of three labels [20].

In the rest of the dissertation, when we use the term RDF bipartite graph, we mean RDF labeled bipartite graph for short.

Example 3.6 (RDF labeled bipartite graph). Figure 3.3 illustrates an example of RDF hypergraph represented as labeled bipartite graph. The left side shows a portion of an ontology in biomedical domain on zebrafish anatomy [75] visualized as a directed labeled graph. Two different relationships are depicted in the figure, namely, “subClassOf” and “part_of.” The corresponding labeled bipartite graph representation is shown on the right side. Circle nodes are *statement nodes* representing RDF statements. Each statement node is connected to three *value nodes* representing three components of a statement (subject, predicate, and object). Edge labels S, P, and O indicate the role of the value nodes in the statement.

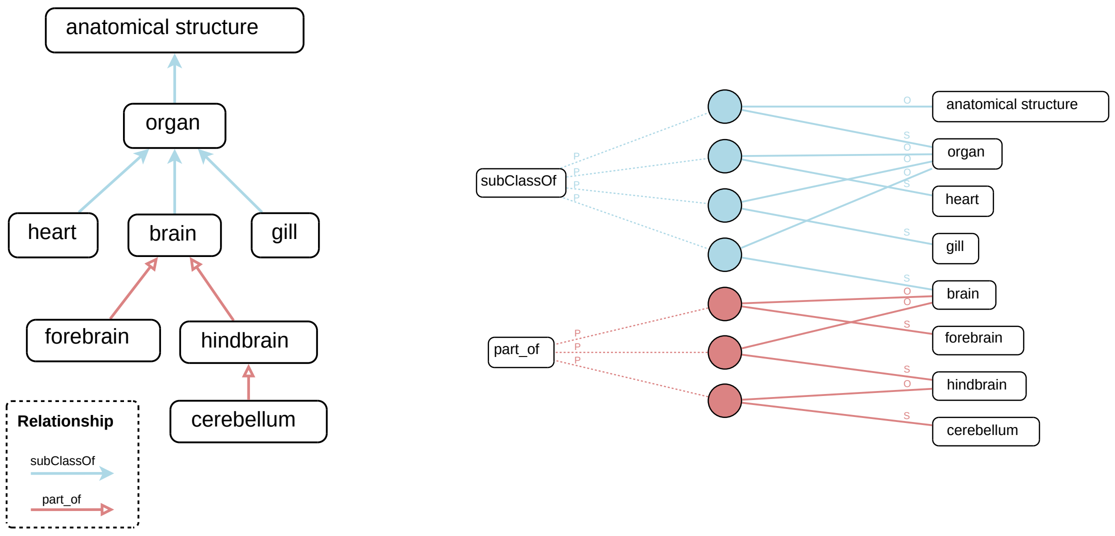


FIGURE 3.3. A comparison between the directed labeled graph and the RDF bipartite graph: (A) A portion of a zebrafish anatomy ontology represented as a directed labeled graph, and (B), an RDF bipartite graph

3.2. Graph Representation for Relational Structures

Various graph representations for relational structures have been proposed in the literature to tackle different data mining tasks. For example, in the case of frequent itemset mining, a set of objects with co-occurrence relationships can be represented as directed or undirected graphs (see the example in Figure 2.1, Chapter II).

Another way to represent a relational structure is to first transform it to RDF, and given the graph nature of RDF, the relational structure can then be represented as a graph. Mapping from an RDB (relational database) to RDF has gained increasing attention and led to the implementation of generic mapping tools as well as domain–specific applications. The W3C launched the RDB2RDF incubator group to explore issues involved in mapping RDB to RDF. A work-in-progress survey paper has been published documenting approaches in this field [23].

A straightforward method for mapping an RDB to RDF is discussed by Berners-Lee [76] as defined in the following.

Definition 3.7 (**Context–independent mapping from an RDB to RDF**). Without linking to any explicit definition of domain semantics (such as those defined in domain ontologies), an RDB can be transformed to RDF following the steps below:

1. An RDB row is an RDF subject node.

2. A column of an RDB table is a predicate node.

3. A cell value of an RDB table is an object node.

Many systems leverage these mappings to automatically generate mappings from an RDB to RDF. Even though these automatically generated mappings often do not capture complex domain semantics that are required by many applications, these mappings can be used as a starting point to create more customized, domain–specific mappings.

Example 3.8 (RDF bipartite graph for a nominal-valued RDB). Table 3.1 (A) shows a relational table with nominal features. The table has m rows, annotated by labels $r_1 \dots r_m$, and n columns named $f_1 \dots f_n$. Applying the steps in Definition 3.7 for mapping an RDB to RDF mapping, the corresponding RDF statements are listed in Table 3.1 (B). From these statements, an RDF bipartite graph is derived, (see Figure 3.4), as the graph representation for the underlying relational table in Table 3.1 (A).

	f_1	$\cdots$	f_n
r_1 :	v_{11}	$\cdots$	v_{1n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$
r_m :	v_{m1}	$\cdots$	v_{mn}

(A)

s	p	o
<r_1>	<f_1>	<v_{11}>
<r_1>	<f_n>	<v_{1n}>
<r_m>	<f_1>	<v_{m1}>
<r_m>	<f_n>	<v_{mn}>

(B)

TABLE 3.1. An example of a relational table with nominal features (A) and its corresponding RDF triple form (B).

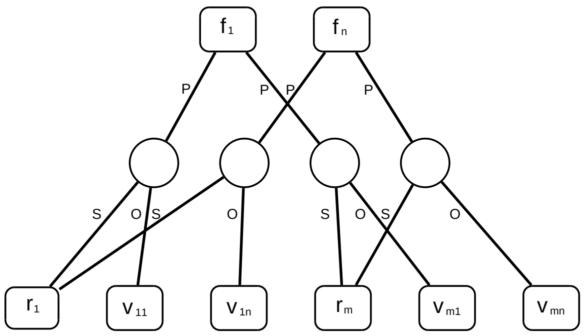


FIGURE 3.4. The RDF bipartite graph for a nominal-valued table based on RDF triples in Table 3.1 (B).

For relational tables with binary (Boolean) features, the RDF representation can be more compact. In some applications, only cells with positive (“1”) values are of interest. In such case, an auxiliary predicate can be introduced to link a row with positive cell values in that row. Example 3.9 illustrates this point.

Example 3.9 (RDF bipartite graph from positive values of a binary-valued RDB). Table 3.2 (A) shows an m-by-n relational table with binary features. We use an auxiliary predicate `<mentions>` to denote a positive occurrence of a feature in one row. For example, the statement `<`r_1`>` `<mentions>` `<`f_n`>` corresponds to the value "1" in the n-th feature in the first row. Consequently, the whole Table 3.2 (A) maps to only two RDF statements in Table 3.2 (B).

	f_1	$\cdots$	f_n
r_1 :	0	$\cdots$	1
$\vdots$	$\vdots$	$\ddots$	$\vdots$
r_m :	1	$\cdots$	0

(A)

s	p	o
`<`r_1`>`	`<mentions>`	`<`f_n`>`
`<`r_m`>`	`<mentions>`	`<`f_1`>`

(B)

TABLE 3.2. An example relational table with binary features.

Using the auxiliary predicate (`<mentions>`) greatly simplifies the resulting RDF graph by reducing the number of distinct predicates from n, according to the process in Definition 3.7, to only 1. This has profound implications for developing efficient analysis and mining methods based on the RDF bipartite graph.

However, the auxiliary predicate is feasible only when linking a row node with its positive value nodes in a binary-valued scenario. If negative cell values are also of interest and need to be included, the trick shown in the following example can be performed so that we can still use a single auxiliary predicate.

Example 3.10 (RDF bipartite graph from both positive and negative values of a binary-valued RDB). Table 3.3 (A) is derived from Table 3.2 (A) by adding a reverse column for each of its original columns: For each f_i, $i \in [1, n]$, a reverse f'_i is created so that feature values $v_{ki} = \neg v_{ki'}$, $\forall k \in [1, m]$. In this way, we can use the auxiliary predicate `<mentions>` to link to negative values by using the reverse column, because, for example, `<`r_1`>` `<mentions>` `<`f_1'`>` is equivalent to `<`r_1`>` `<mentions>` $\neg$`<`f_1`>` . Table 3.3 (B) shows the RDF statements based on Table 3.3 (A) which essentially captures information of both positive and negative values from Table 3.2 (A).

	f_1	f_1'	$\cdots$	f_n	f_n'
r_1 :	0	1	$\cdots$	1	0
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
r_m :	1	0	$\cdots$	0	1

(A)

s	p	o
`<`r_1`>`	`<mentions>`	`<`f_1'`>`
`<`r_1`>`	`<mentions>`	`<`f_n`>`
`<`r_m`>`	`<mentions>`	`<`f_1`>`
`<`r_m`>`	`<mentions>`	`<`f_n'`>`

(B)

TABLE 3.3. An example expanded relational table with binary features.

The process of adding reverse columns to binary-valued RDB tables described in Example 3.10 can be extended to nominal-valued tables as well. By doing this we can achieve the desirable property of having only one predicate in the resulting RDF representation. The process is called RDB nominal value expansion as defined below.

Definition 3.11 (RDB nominal value expansion). In a nominal valued RDB table, for each feature f_i taking values on the set $V_i = \{v_{i1}, v_{i2}, \ldots\}$, we denote $|V_i|$ as the number of distinct values of f_i. The RDB nominal value expansion is the process to transform each nominal feature f_i to $|V_i|$ number of binary features $(f_{i1}, f_{i2}, \ldots, f_{i|V_i|})$. The value of k-th row in $f_{ij}, (j \in [1, |V_i|)$, is "1", if and only if f_i takes the value v_{ij} in the k-th row.

Example 3.12 (RDB nominal value expansion). Table 3.4 (A) shows a nominal-valued RDB table with meaningful column names and values. We use the notation, Outlook={sunny, overcast, rainy}, to denote the set of distinct values the feature "Outlook" can take on. Similarly, we have Temperature={hot, mild, cool}, and Humidity={high, low}. Table 3.4 (B) shows the resulting table after nominal value expansion based on Definition 3.11. RDF statements are then derived using one single auxiliary predicate `<mentions>`, as partly shown in Table 3.4 (C).

	O	T	H
r_1 :	sunny	hot	high
r_2:	rainy	cool	low
$\vdots$	$\vdots$		
r_m :	overcast	mild	low

(A)

	O_s	O_o	O_r	T_h	T_m	T_c	H_h	H_l
r_1 :	1	0	0	1	0	0	1	0
r_2 :	0	0	1	0	0	1	0	1
$\vdots$	$\vdots$							
r_m :	0	1	0	0	1	0	0	1

(B)

s	p	o
<r_1>	`<mentions>`	`<O_s>`
<r_1>	`<mentions>`	`<T_h>`
<r_1>	`<mentions>`	`<H_h>`
<r_2>	`<mentions>`	`<O_r>`
	...	

(C)

TABLE 3.4. Nominal value expansion for a relational table and the resulting RDF triples. (A) shows the original table where O stands for "outlook," T for "temperature" and H for "humidity". (B) shows the expanded table. (C) shows the corresponding RDF triples derived from (B).

3.3. Combining Data Graphs and Ontology Graphs

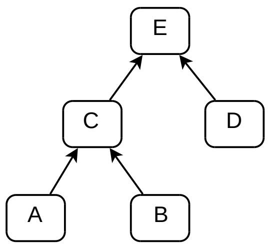
In order to facilitate the synergy between data and domain knowledge in a mining framework, information from both sources needs to be first combined. This is achieved by the process called *semantic annotation*. Semantic annotation aims at assigning formal semantic descriptions to the basic element of data, and it is crucial in realizing semantic data mining by bridging formal semantics in domain knowledge with data. A number of previous research efforts have been devoted to this direction, resulting in various methodologies and systems, such as the NCBO (National Center of Biomedical Ontology) annotator [77] that generates the electronic health dataset for our experiments described in Chapter V and VI. Also, readers are referred to Reeve and Han [78] for a general survey on semantic annotation.

In the following, we assume data is annotated, meaning that links from entities in data to formal semantic descriptions (such as those in ontologies) are already established. A unified graph incorporating information from both data and ontologies can be created. Data mining algorithms dealing with such unified graph representation can enjoy the benefit of a seamless integration of domain knowledge. The following example shows the combination of an ontology graph and a data graph.

Example 3.13 (**Combining an ontology graph and a data graph**).
Figure 3.5 (A) shows a simple ontology in a certain domain with only subsumption relationships defined for five concepts (A–B). Figure 3.5 (B) shows a binary-valued RDB table in the same domain with the set of concepts (A–B) being features. We use the same concept labels in the ontology and the RDB table because we assume the mapping between the ontology nodes and the table features are pre-assigned manually or established by automatic annotation. Figure 3.6 (B) shows the RDF statements derived from both the ontology and the RDB table. Figure 3.6 (A) demonstrates the combined RDF bipartite graph.

Nodes in Figure 3.6 (A) can be rearranged to a particular form as shown Figure 3.7. This graph demonstrates a tripartite structure where row nodes (r_1–r_5) fall on one partition, column nodes (A–E) on another, and statement nodes in between. A plethora of graph mining techniques can be leveraged to analyze the path configuration in this graph to answer interesting questions such as the grouping of rows and columns (*e.g.*, to solve the the task of clustering and association mining respectively). Predicate nodes can serve as hints to introduce different weights to paths in order to distinguish different semantic types and capture their relative strengths.

Edge and statement nodes in Figure 3.7 are depicted in two colors to signify their sources of origin. Red edges and nodes denote information from data (RDB table), and blue ones from the ontology. We notice from the graph that the contribution of the ontology can be viewed as to introduce extra paths of different semantic types (The data is structured under the “mentions” relationship and the ontology is structured under the subsumption relationship). In this way, a data mining algorithm that is able to deal with the data graph can be naturally extended without major modifications to handle domain knowledge coded in the ontology.



(A)

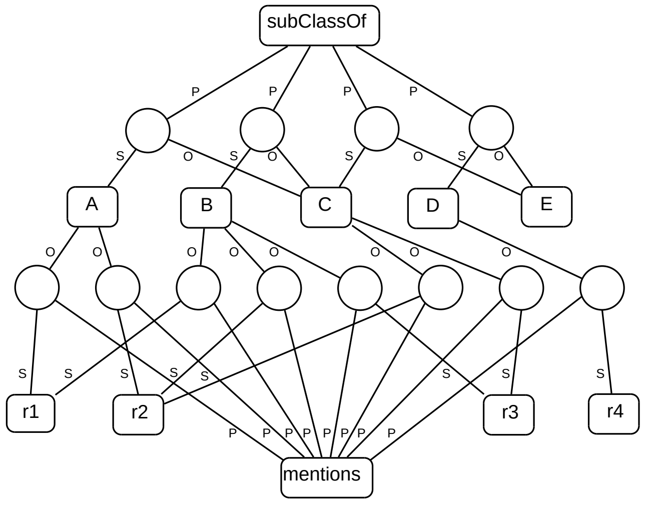
	A	B	C	D	E
r_1 :	1	1	0	0	0
r_2 :	1	1	1	0	0
r_3 :	0	1	1	0	0
r_3 :	0	0	0	1	0

(B)

FIGURE 3.5. An example relational table and a domain ontology. The binary-valued relational about five concepts (“A”–“E”) is shown in table (B), and the ontological relationship among these concepts is shown as a directed graph in (A).

3.3.1. Representing Different Kinds of Ontological Semantics

In order to leverage the increasingly larger and richer collection of domain ontologies, especially in scientific fields such as the biomedical domain, we propose to use weights to distinguish paths in the RDF bipartite graph representing different semantic types or relationships (properties) from the ontology, such as



s	p	o
`<A>`	`<subClassOf>`	`<C>`
`<B>`	`<subClassOf>`	`<C>`
`<C>`	`<subClassOf>`	`<E>`
`<D>`	`<subClassOf>`	`<E>`
`<r1>`	`<mentions>`	`<A>`
`<r1>`	`<mentions>`	`<B>`
`<r2>`	`<mentions>`	`<A>`
`<r2>`	`<mentions>`	`<B>`
`<r2>`	`<mentions>`	`<C>`
`<r3>`	`<mentions>`	`<B>`
`<r3>`	`<mentions>`	`<C>`
`<r4>`	`<mentions>`	`<D>`

(A) (B)

FIGURE 3.6. The RDF bipartite graph representation (A) given triples shown in (B) based on the information described in Figure 3.5.

class subsumption, part_of, and other general or domain–specific properties. The weights also characterize the relative importance of the paths.

Example 3.14 (**Assigning weights to different relationship**)**.** Figure 3.8 shows an example of an RDF bipartite graph representing information about a group of people (A–E) where multiple relationships can be identified among them. For example, A, B, C and D are linked by the coauthorship relationship, while D and E are linked by the more general collaboration relationship (in fact, coauthorship is defined as a sub-property of collaboration in the ontology). A, B and C are professors, D and E are PhD students, and both professors and PhD students are researchers. In this complex network of relationships, we can distinguish their roles and importance by assigning application–specific weights to the related paths (*e.g.*, different colorings in the graph denotes different weights).

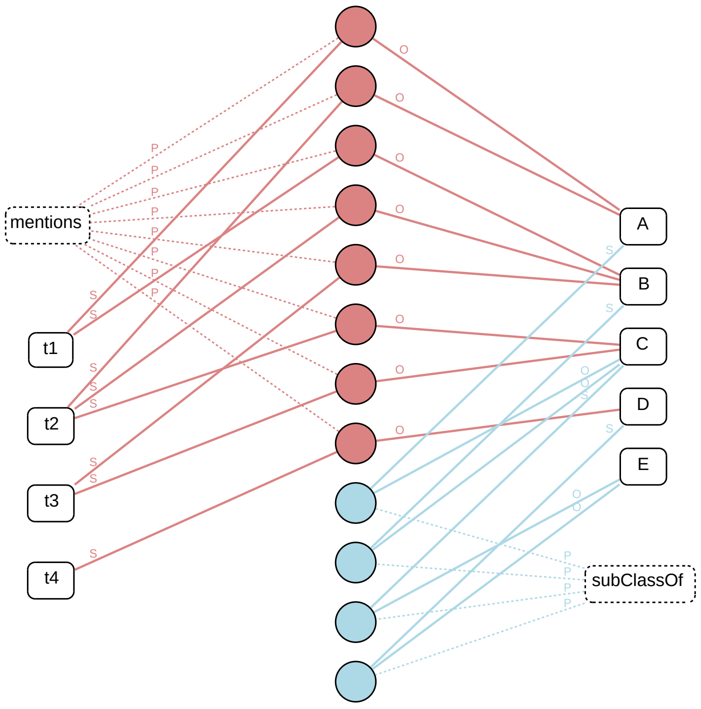


FIGURE 3.7. Transforming the RDF bipartite graph to suit mining need: This figure shows that, grouping the nodes according to whether they are row elements or column elements in Figure 3.5 (B), the bipartite graph shown in Figure 3.6 (A) can be further transformed to a tripartite graph.

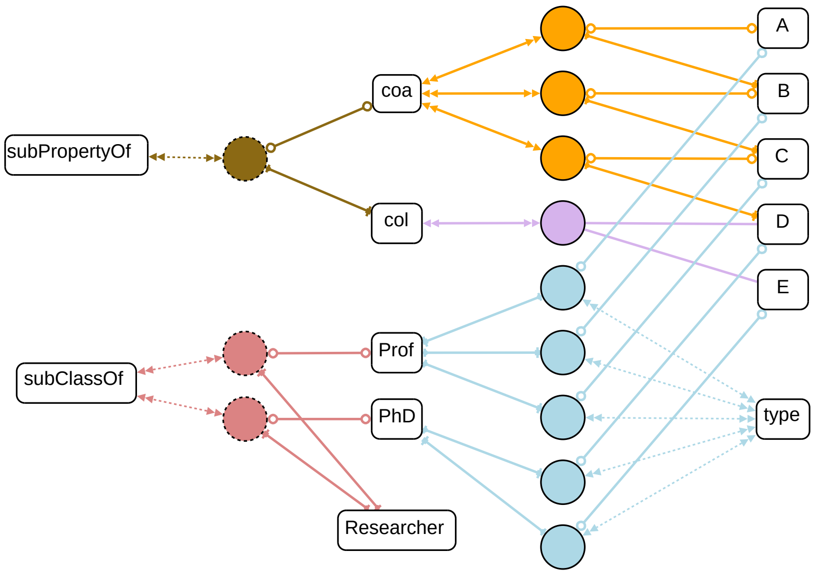


FIGURE 3.8. An example RDF bipartite graph that represents various semantic relationships.

CHAPTER IV

INTEGRATION OF HETEROGENEOUS INFORMATION SOURCES

I have described the method to model data and domain knowledge encoded in ontologies in a unified graph representation. In practice, it is common that raw data reside in disparate sources and alternative ontologies or schemas are present in a domain. When a data mining system is required to access multiple sources of information, how to resolve heterogeneities is a challenging task. This chapter makes a contribution in this direction.

This chapter consists of work published in volume 1 of the "Journal on Data Semantics" in 2012 [79] and that in volume 92 of the journal "Neurocomputing" in 2012 [80]. Dr. Dejing Dou initially identified work. Dr. Gwen Frishkoff and Robert Frank contributed the heterogeneous neuroscience dataset and provided valuable insights on the experimental results. Hao Wang performed the evolutionary multi-objective optimization.

4.1. Overview

The presence of heterogeneity among schemas supporting vast amounts of information demands an advanced solution for semantic integration of disparate data sources to facilitate interoperability and reuse of the information. The challenge is especially pronounced in many scientific domains where a massive amount of data is produced independently and thus each has its own data vocabulary. While manual integration is time-consuming and requires expensive specialized human capital, the development of automatic approaches becomes essential to aid inter-institute collaborations.

In our attempt to tackle the problem, we focus on developing a method to solve a specific kind of integration problem involving matching alternative ontologies or schemas. We recognize several key constraints that make our problem challenging and can cause conventional methods to be ineffective. They are, namely, 1) little-to-no string-based or linguistic similarity between vocabularies, and 2) numeric typed data instances. The discovery of matching between numeric-typed attributes used in different datasets is a common task in integrating scientific datasets that have been collected and analyzed in different research labs. We call such task the *attribute matching* problem.

Another challenging task given multiple data sources is to carry out meaningful meta-analysis that combines results of several studies on different datasets to address a set of related research hypotheses. Finding correspondences among distinct patterns that are observed in different scientific datasets is an example of meta-analysis. Supposing the patterns are derived by clustering analysis, this problem can be addressed by the application of cluster comparison (or cluster matching) techniques. Clustering is an unsupervised data mining task widely used to discover patterns and relationships in a variety of fields. The clustering result provides a pattern characterization from a data-driven perspective. If similar results are obtained across multiple datasets, this leads in turn to a revision and refinement of existing domain knowledge, which is a central goal of meta-analysis. However, there are noticeably few cluster comparison methods that are able to compare two clusterings derived from different datasets. The difficulty for the comparison is further exacerbated by the fact that the datasets may be described by attributes from heterogeneous schemas or ontologies. Even those methods that are able to measure clustering similarity across different datasets (e.g., the ADCO [74] method) have to assume homogeneous meta-data (e.g., the same schemas).

Given this situation, in order to carry out cluster comparison for meta-analysis, researchers often need to perform ontology or schema matching first in order to mitigate the gap for meta-data. In the work reported in [73], we examine a practical attribute matching problem on neuroscience data where schema elements from one dataset share no lexical similarity with those from the other. Moreover, structural similarity is also limited. One can only resort to instance-based (extensional) methods. However, since all attributes are numerical, information clues available to an instance-level matcher are very restricted. Traditional instance-based matchers typically make use of constraint-based characterization, such as numerical value ranges and averages to determine correspondences. However, this is often too rough in the case of an all-numerical dataset. Two attributes may have similar ranges and averages but totally different internal value distributions (an example is shown in Section 4.3.1). Given this, we propose to represent the attribute value distribution at a finer granularity by partitioning the values into groups. To do this, clustering is performed, and the resulting clusters are then aligned across two datasets (assuming that the same pattern exists in both datasets). In this way, each attribute can be characterized by, instead of a single value, a vector of per-cluster statistical quantities (which we also call the *segmented statistical characterization*). A distance function can then be applied based on this representation. Table 4.1(A) shows an example distance table on the cross join of two sets of attributes. To discover attribute matching from this table

$$\begin{array}{c|ccc} & a'_1 & \cdots & a'_m \\ \hline a_1 & d_{11'} & \cdots & d_{1m'} \\ \vdots & & \ddots & \\ a_m & d_{m1'} & & d_{mm'} \end{array} \qquad \begin{array}{c|ccc} & c'_1 & \cdots & c'_n \\ \hline c_1 & d_{11'} & \cdots & d_{1n'} \\ \vdots & & \ddots & \\ c_n & d_{n1'} & & d_{nn'} \end{array}$$

(A) (B)

TABLE 4.1. Example distance matrices between (A) two sets of attributes and (B) two sets of clusters, respectively.

can be reduced to solving a minimum assignment problem (assuming matching is bijective), which is a classical combinatory optimization problem that has a polynomial solution using the Hungarian Method [81].

Unfortunately, however, the above solution requires the alignment of clusters across datasets, which is a difficult problem in its own right. If fully automated, as mentioned above, methods such as ADCO adopt a so called *density profile* [74] representation of clusters that requires homogeneous meta-data or a priori knowledge about the attribute matching in heterogeneous scenarios. Then the cluster matching can be carried out in a similar manner to the attribute matching by being solved as an assignment problem (see Table 4.1(B), for example). This leads to a circular causality, or a deadlock, between the attribute matching (under the segmented statistical characterization) and cluster matching (under the density profile representation) — none of them can be solved automatically without the other one being solved first.

To address this difficulty, viewing the two matching problems as combinatorial optimization problems with distinct yet interrelated objective functions, we propose a novel approach using a multi-objective heuristics to discover attribute matching and cluster matching simultaneously. The objectives in the optimization are to minimize distances of attribute matching and cluster matching respectively. We explore the widely used simulated annealing algorithm as the metaheuristics algorithm and briefly compare its performance with the evolutionary multi-objective algorithm in experiments.

4.2. Method

Problem Definition: We tackle two matching tasks in this work, namely, the attribute matching and cluster matching problems. The solution is to cast the dual matching problems to a multi-objective optimization problem so that the matchings can be solved simultaneously. The two objective functions to be optimized are defined as the total distance of corresponding elements in attribute and cluster matching respectively. To this end, we explore methods to represent attributes and clusters so that distance measure can be reasonably defined. We assume that the optimal matching lies at the Pareto front in this multi-objective problem.

We use metaheuristics search algorithm to solve this multi-objective optimization problem. In the following we describe an adaption of the widely used simulated annealing algorithm to multi-objective optimization in order to solve the matching problems. Later in the Experiment Section, we briefly describe an evolutionary multi-objective algorithm and compare their performance.

4.2.1. The Multi-Objective Simulated Annealing Framework

Simulated annealing (SA) is a generic probabilistic metaheuristic for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. We briefly describe SA in Section 2.4.1.1, Chapter II. To solve the dual matching problems, we adopt an adaptation of SA for multi-objective optimization. The resulting algorithm is the so-called multi-objective simulated annealing (MOSA [82]), in which the acceptance criterion in the annealing process is established based on the idea of Pareto-domination based fitness. Specifically, fitness of a solution is defined as one plus the number of dominating solutions in Pareto-optimal set. The larger the value of fitness, the worse is the solution. Initially, the fitness difference between the current and the generated solution is small and the temperature is high so almost any move is accepted. This gives a way for the search to explore as much of the solution space as possible. As the number of iterations increases, temperature decreases and the fitness difference between the current and generated solutions may increase. Both of them make the acceptance more selective and can result in a well-diversified set of Pareto-optimal solutions. Details of the multi-objective simulated annealing algorithm are outlined in Algorithm 1.

Formally, the processes involved in the proposed multi-objective simulated annealing framework can be defined as follows.

$$\begin{aligned}
X &= [x_a, x_c] \\
F &= [f_a, f_c] \\
P_a([x_a^{(n-1)}, x_c^{(n-1)}]) &= [x_a^{(n)}, x_c^{(n-1)}] \\
P_c([x_a^{(n-1)}, x_c^{(n-1)}]) &= [x_a^{(n-1)}, x_c^{(n)}] \\
G_{c|a}([x_a^{(n)}, x_c^{(n-1)}]) &= [x_a^{(n)}, x_c^{(n)}] \\
G_{a|c}([x_a^{(n-1)}, x_c^{(n)}]) &= [x_a^{(n)}, x_c^{(n)}] \\
G \circ P([x_a^{(n-1)}, x_c^{(n-1)}]) &= [x_a^{(n)}, x_c^{(n)}]
\end{aligned}$$

Algorithm 1 Multi-Objective Simulated Annealing

```
Input: Empty Pareto-optimal set of solutions $\mathbf{\Sigma}$
Input: Empty current decision vector $\mathbf{X} = [x_a, x_c]$
Input: Initial temperature $T$
  $count = 0$
  while $T > threshold$ do
    $initialize(\mathbf{X})$
    If $\mathbf{X}$ is pareto-optimal, put $\mathbf{X}$ in $\mathbf{\Sigma}$
    $\mathbf{X}' = generate\_solution(\mathbf{X})$
    $S_{\mathbf{X}'} = evaluate\_solution(\mathbf{X}')$
    $\Delta S = S_{\mathbf{X}'} - S_X$
    if $r = rand(0, 1) < exp(\frac{-\Delta S}{T})$ then
      $\mathbf{X} = \mathbf{X}'$
      $S_X = S_{X'}$
    end if
    count = count + 1
    //Periodically restart
    if $count == restart\_limit$ then
      $\mathbf{X} = select\_random\_from\_Pareto(\mathbf{\Sigma})$
      continue
    end if
    $reduce\_temperature(T)$
  end while
```

X is the decision vector that contains two variables for attribute matching, x_a, and cluster matching, x_c, respectively (details in Section 4.2.2). F is the objective function vector that contains two criterion functions (f_a and f_c) to evaluate attribute matching and cluster matching decisions (details in Section 4.2.4). P is the random perturbation function that takes a decision vector in the $(n-1)$th iteration and partially advances it to the nth iteration (we use P_a or P_c to distinguish between the random selections). The partial candidate decision generation function G takes the output of P and fully generate a decision vector for the nth iteration (by advancing the left-out variable in P to its nth iteration). Thus, the compound function $G \circ P$ fulfils the task of generating an nth-iteration candidate decision vector given the $(n-1)$th one (details in Section 4.2.5.2).

4.2.2. Decision Variable

The domains of the decision variables in the matching problems take values on a permutation space. In other word, by formalizing the problem of finding correspondent elements of two sets S and S' of cardinality n as an optimization problem, the solution is completely specified by determining an optimal permutation of $1, \ldots, n$. For instance, for two sets of three elements, their indexes range over $\{0, 1, 2\}$. Applying a permutation $\pi = \{2, 0, 1\} \in S_3$ on S' can be viewed as creating a mapping (bijection) from elements on the new positions of S' to elements on the corresponding positions in S. In this example, the permutation π on S' specifies the following correspondences: $S_0 \leftrightarrow S'_2$, $S_1 \leftrightarrow S'_0$, and $S_2 \leftrightarrow S'_1$.

Formally, let P_n ($n \in \mathbb{N}$) be the symmetric group of all permutations of the set $\{1, 2, \ldots, n\}$. Given two sets S and S' with the same cardinality of n, performing identity permutation on one set and an arbitrary permutation $\pi \in S_n$

on the other specifies a matching (or mathematically speaking, mapping) between the two sets. In the multi-objective optimization formalism for solving the attribute matching and cluster matching problems, the decision vector has two variables: $X = [x_a, x_c]$. If we have M attributes and N clusters to match respectively, then $x_a \in P_M$ and $x_c \in P_N$.

4.2.3. Data Representation

The central objects of interest in our study, namely, the numeric-typed attributes and clusters, need to be represented in ways that meaningful quantities can be defined to measure the "goodness" of a matching decision. To this end, we propose to use the *segmented statistical characterization* to represent attributes, and the *density profiles* to represent clusters. Details of these representations are described below.

4.2.3.1. Representation of Attributes:

Numeric-typed attributes can be represented by the segmented statistical characterization, in which data instances are first partitioned into groups (e.g., through unsupervised clustering) and then characterized by a vector of indicators, each denoting a statistical characterization of the corresponding group. For example, if values of an attribute A are clustered into n groups, then it can be represented by a vector of segmented statistical characterization as follows:

$$V_A = \Big[\mu_1, \mu_2, \ldots, \mu_n\Big],$$

where we choose the mean value μ_i for cluster i as the statistical indicator in our implementation.

4.2.3.2. Representation of Clusters:

Clusters can be represented by density profiles [74] as described in Section 4, Chapter II. The attribute's range in each cluster is discretized into a number of bins, and the similarity between two clusters corresponds to the number of points of each cluster falling within these bins. Given this, density profile vector V_C for a clustering C is formally defined as an ordered tuple by Equation 2.1 where $dens_C(k, i, j)$ refers to the number of points in the region (i, j)—the j-th bin of the i-th attribute—that belongs to the cluster c_k of clustering C.

4.2.4. Objective Functions

The objective functions in the attribute matching and cluster matching problems are criteria to evaluate the "goodness" of matchings. We use the sum of pair-wise distances between matched elements (see Table 4.1 for example) as the objective function. Given this, to determine the form of objective functions amounts to defining proper pair-wise distance measures for the attribute and cluster matching problems respectively, as detailed in the following.

4.2.4.1. Distance function between two attributes

The pairwise distance $\mathcal{L}$ between two attributes is defined as the Euclidean distance between their segmented statistical characterization vectors, and f_a calculates the sum of pair-wise distances under the attribute matching specified by x_a:

$$f_a(x_a) = \sum_{k=1}^{M} \mathcal{L}\Bigg((V_a)^k,\ (V_a')^{x_a(k)} \Bigg) = \sum_{k=1}^{M} \sqrt{ \sum_{i=1}^{N} \left(\mu_i^k - (\mu')_i^{x_a(k)} \right)^2 } \ , \qquad \text{(Equation 4.1.)}$$

where $x_a \in P_M$.

4.2.4.2. Distance function between two clusters

The ADCO similarity described in Equation 2.2 of Section 2.4.3, Chapter II, can be transformed to a distance defined as follows [74]:

$$D_{ADCO}(C, C') = \begin{cases} 2 - ADCO(C, C') & if\ C \neq C' \\ 0 & otherwise \end{cases} \qquad \text{(Equation 4.2.)}$$

We use D_{ADCO} as the pair-wise distance between two clusters under the density profile representation, and f_c calculates the sum of pair-wise distances under the cluster matching specified by x_c

$$f_c(x_c) = \sum_{k=1}^{N} D_{ADCO}\Bigg((V_c)^k,\ (V_c')^{x_c(k)} \Bigg) = \sum_{k=1}^{N} \Bigg(2 - \sum_{i=1}^{M} \sum_{j=1}^{Q} \Big(dens(k,i,j) \times dens(x_c(k),i,j) \Big) \Big/ \max\Bigg[\sum_{i=1}^{M} \sum_{j=1}^{Q} dens(k,i,j)^2, \sum_{i=1}^{M} \sum_{j=1}^{Q} dens(x_c(k),i,j)^2 \Bigg] \Bigg) , \qquad \text{(Equation 4.3.)}$$

where $x_c \in P_N$.

4.2.5. Generation of New Solution

In each iteration of the simulated annealing process, we randomly generate a candidate decision in the neighborhood of the last-iteration decision by applying two consecutive processes, namely, the random perturbation and the partial candidate decision generation, as described below.

4.2.5.1. Random Perturbation:

In each iteration, we select at random one variable (either x_a or x_c) in the decision vector and perturb it by randomly swapping two positions in the selected variable. This advances that variable from the $(n-1)$th iteration to the nth iteration. Then the following partial candidate generation process is carried out to bring the other variable also to the nth iteration.

4.2.5.2. Partial candidate decision generation

Given $x_c^{(n)}$, derive $x_a^{(n)}$:

$$\begin{aligned} x_a^n &= \arg\min_{\pi} f_a(\pi, x_c^{(n)}) \\ &= \arg\min_{\pi} \sum_{k=1}^{M} \mathcal{L}\Big((V_a)^k,\ (V_a')^{\pi(k)} \Big) \\ &= \arg\min_{\pi} \sum_{k=1}^{M} \sqrt{\sum_{i=1}^{N} \Big(\mu_i^k - (\mu')^{\pi(k)}_{x_c^{(n)}(i)} \Big)^2} \end{aligned} \qquad \text{(Equation 4.4.)}$$

Given $x_a^{(n)}$, derive $x_c^{(n)}$:

$$\begin{aligned}
x_c^n &= \arg\min_{\pi} f_c(\pi, x_a^{(n)}) \\
&= \arg\min_{\pi} \sum_{k=1}^{N} D_{ADCO}\Big((V_c)^k,\ (V_c')^{\pi(k)} \Big) \\
&= \arg\max_{\pi} \sum_{k=1}^{N} \left(\sum_{i=1}^{M} \sum_{j=1}^{Q} \Big({\scriptstyle dens(k,i,j) \times dens(\pi(k), x_a^{(n)}(i), j)} \Big) \Big/ \right. \\
&\left. \max \left[\sum_{i=1}^{M} \sum_{j=1}^{Q} {\scriptstyle dens(k,i,j)^2}, \sum_{i=1}^{M} \sum_{j=1}^{Q} {\scriptstyle dens(\pi(k), x_a^{(n)}(i), j)^2} \right] \right) \qquad \text{(Equation 4.5.)}
\end{aligned}$$

To calculate π that satisfies Equation 4.4 and Equation 4.5, rather than iterating through all possible permutations, we can consider the equation as a minimum-cost assignment problem. Table 4.1(A), for example, illustrates a distance table between two attribute sets A and A'. Matching of the two sets can be considered as an assignment problem where the goal is to find an assignment of elements in $\{A_i\}$ to those in $\{A_i'\}$ that yields the minimum total distance without assigning each A_i more than once. This problem can be efficiently solved by the Hungarian Method in polynomial time of $O(K_{min}^3)$ [81]. It is worth noticing that by formulating the problem as the assignment problem, we assume the matching between two sets to be a one-to-one function.

4.3. Case Studies

Because we are interested in understanding the property of the Pareto front obtained by our method, we conducted a series of experiments to highlight tradeoffs of the objectives functions. First, to illustrate the proposed method is indeed capable of determining matchings between numeric-typed attributes and clusters, we synthesized a dataset simulating some extreme conditions under which previous methods are ineffective. Also, from the results obtained on the synthetic dataset, we empirically study tradeoffs between the two objective functions. Then, to evaluate the scalability of the method, we carry out a series of tests on a set of data with varied sizes. Finally, encouraged by these results, we applied our methods to actual neuroscience ERP (event-related potentials) data to highlight the applicability of our method to the neuroscience domain.

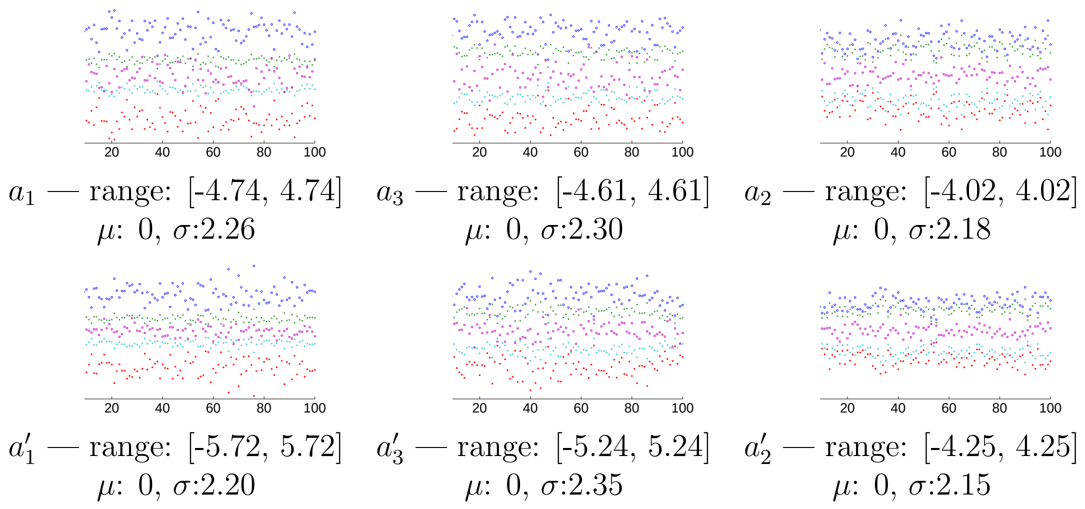
4.3.1. Synthetic Dataset

4.3.1.1. Data Generation:

In the synthetic dataset, tables are generated in such a way that each attribute consists several Gaussians with distinct means and standard deviations, and for one attribute in the source table, there exists exactly one attribute in the target table whose Gaussians possess the same configuration (hence they match each other). However if the attribute is viewed as a single distribution, as is typical in previous methods, its mean and standard deviation would be indistinguishable from those of other attributes in the same table. For example, Figure 4.1 illustrates the value distributions of three attributes (a_1, a_2, and a_3) from one dataset and their corresponding counterparts (a'_1, a'_2, and a'_3) from another.

4.3.1.2. Results:

Figure 4.2 illustrates the Pareto front obtained from matching two synthetic datasets, each having 20 attributes and 5 clusters. Most notably, the gold standard results for both attribute matching and cluster matching are obtained from the left-most point on the Pareto front. In other words, given the decision variables (X) corresponding to that point, we obtained 100% correct matching results. We



a_1 — range: [-4.74, 4.74] μ: 0, σ:2.26 a_3 — range: [-4.61, 4.61] μ: 0, σ:2.30 a_2 — range: [-4.02, 4.02] μ: 0, σ:2.18

a'_1 — range: [-5.72, 5.72] μ: 0, σ:2.20 a'_3 — range: [-5.24, 5.24] μ: 0, σ:2.35 a'_2 — range: [-4.25, 4.25] μ: 0, σ:2.15

FIGURE 4.1. The distribution of synthetic datasets is shown in the scatter plots of data instances from three sample attributes in one dataset (upper frame) and those of their corresponding attributes from another (lower frame) are illustrated.

further observed that in our subsequent tests on other synthetic datasets with varied number of attributes and clusters, the derived Pareto fronts all contain the gold standard result, and the point corresponding to the gold standard can always be found towards the minimum end of f_a. Given this, we propose the following method to reduce the Pareto-optimal set to a single point corresponding to the most favored choice (X^*) in the decision space. The idea is to find the decision with the minimum weighted sum of objective values in the obtained Pareto-optimal set, i.e., $X^* = \arg\min_X \left[\alpha f_a(X) + \beta f_c(X)\right]$, where α and β are weights. We first conducted preliminary experiments to determine the best values for α and β (0.8 and 0.2 respectively) and used them in all subsequent experiments. This method works markedly well on the synthetic datasets. For all the tests described in Table 4.2, 100% correct results for both attribute and cluster matchings are obtained (hence we omit the precision in the table).

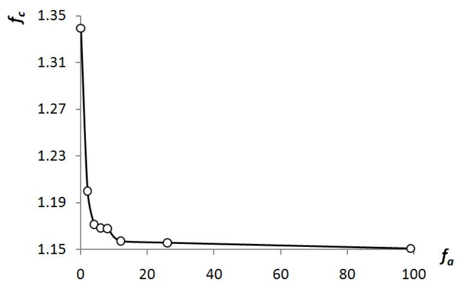


FIGURE 4.2. An example Pareto front obtained from matching two synthetic datasets with 20 attributes and 5 clusters.

Notice that it is common in multi-objective optimization problems that a non-dominated set may be too large for decision makers to reasonably consider. However, it is shown in Figure 4.2 (as well as results from other experiments described in the following) that this is not the case using our method on datasets of representative sizes in attribute and clustering matching problems. The number of resulting Pareto optimal solutions is small enough to be presented to decision makers without the need of any means of reducing or organizing the non-dominated set. The reason why we use a straightforward weighted sum method to compute the most significant solution from Pareto front is because it empirically works well on our test cases. This step is not obliged because a decision maker can go over solutions in Pareto front and decide which one is the best.

4.3.1.3. Running Time

We systematically altered the number of attributes and clusters present in the data and conducted a series of tests to show the scalability of the proposed method. The running time under different configurations is reported in Table 4.2. The time is calculated by averaging over 5 runs of each test (on a 2.53GHz dual-core

CPU with 4 gigabytes memory), each run having 1000 iterations in the simulated annealing process.

# attributes	# clusters	time (sec)
5	20	0.28
20	20	1.81
20	40	7.04
20	60	17.80
40	20	4.66
40	40	11.74
40	60	25.93
60	20	10.95
60	40	20.70
60	60	37.35
100	100	172.23

TABLE 4.2. Running time of the annealing process on synthetic datasets with varied configurations of attribute and cluster sizes. The time is obtained by averaging over results of 5 runs of each test.

The main computationally expensive part of the annealing process is the generation of new candidate solution phase (function G) in which an assignment problem is solved using the Hungarian method. The complexity of the Hungarian method is cubic and is already the most efficient algorithm for solving the assignment problem (e.g., a brute force algorithm has a factorial complexity). In scenarios where the size of the problem is huge (both the number of attributes and the number of clusters are large), our method can become computationally costly. For example, the ARCENE dataset [83] from the UCI machine learning repository contains mass-spectrometric output with 10,000 continuous input variables. ARCENE's task is to distinguish cancer versus normal patterns and the dataset is typically used as a benchmark for classification and feature selection algorithms. To match sets of attributes at this scale will definitely require more advanced adaptation of our metaheuristics search algorithm, such as approximation or partitioning of the search space to enable parallelism. On the other hand, as we have shown in the synthetic test case and will elaborate upon in latter studies, our method boasts significant accuracy and the unique ability to distinguish attributes with similar statistics. For the ARCENE dataset, we create an artificial matching problem by first randomly selecting a subset of data with 150 attributes as the source, and then make a target dataset by injecting a small amount of noise to the source. We then run the simulated annealing algorithm to find both attribute and cluster matchings and achieved 132/150 accuracy for attribute matching and 4/5 accuracy for cluster matching. A baseline method that simply utilizes one single statistics for each attribute scores 95/150 accuracy. This shows that our method is able to provide a practical trade-off between accuracy and scalability.

4.3.2. Neuroscience Dataset

4.3.2.1. Data Acquisition

To address the problems of attribute and cluster matching in a real-world neuroscience application, we used a set of realistic simulated ERP (event-related potentials) datasets, which were designed to support evaluation of ERP analysis methods [18]. The datasets were specifically designed to simulate heterogeneous data from different groups of subjects under different conditions (via distinct simulated brain activities), as well as distinct measurement methods (spatial and temporal metrics) and distinct patterns (reflecting two different pattern decomposition techniques). Real ERP data arise from superposition of latent scalp-surface electrophysiological patterns, each reflecting the activity of a distinct cortical network that cannot be reconstructed from the scalp-measured data with any certainty. Thus, real ERP data are not appropriate for evaluation of ERP pattern mapping. By contrast, simulated ERP data are derived from known source patterns and therefore provide the necessary gold standard for evaluation of our proposed methods.

The raw data for this study consist of 80 simulated event-related potentials (ERPs), in which each ERP comprises simulated measurement data for a particular subject ($n = 40$). The 40 simulated subjects are randomly divided into two 20-subject groups, SG1 and SG2, each containing 40 ERPs (20 subjects in 2 experimental conditions). Each ERP consists of a superposition of 5 latent varying spatiotemporal patterns. These patterns were extracted from the two datasets, SG1 and SG2, using two techniques: temporal Principal Components Analysis (tPCA) and spatial Independent Components Analysis (sICA), two data decomposition techniques widely used in ERP research [84]. To quantify the spatiotemporal characteristics of the extracted patterns, two alternative metric sets, m1 and m2, were applied to the two tPCA and the two sICA derived datasets. For a complete explanation of these alternative metrics, please see Appendix in [18].

In summary, the simulated ERP data generation process yielded eight test datasets in total, reflecting a 2 (attribute sets) $\times$ 2 (subject groups) $\times$ 2 (decomposition methods) factorial design. Therefore, for each attribute set there are 4 datasets generated from different combinations of subject groups and decomposition methods, resulting $4 \times 4 = 16$ cases for the studies of attribute matching and cluster matching. The reason to include such variabilities was to test the robustness of our matching method to different sources of heterogeneities across the different datasets. Within all test datasets, 5 major ERP spatiotemporal patterns are present. They are P100, N100, N3, MFN, and P300. These patterns can be identified in the datasets by clustering analysis. Pretending that the latent patterns underlying discovered clusters are unknown, we hope to match clusters across datasets to recover the fact that the same patterns are present in all datasets.

4.3.2.2. Results

We applied the weighted sum method as the post-processing step after obtaining the Pareto-optimal solutions to determine the most favored choice using the parameters (α and β) discovered in the preliminary experiments on synthetic datasets (cf. Section 4.3.1). The accuracy of attribute matching and cluster matching along with the number of points in the Pareto front are listed in Table 4.3 (all these results are obtained by taking average from 5 runs for each test case).

It can be observed from the results in Table 4.3 that more different factors involved in the acquisition of the two datasets for matching can negatively affect the matching performance. For example, in test case 1, the two datasets are drawn from the same subject group (SG1) and preprocessed using the same decomposition method (sICA); whereas in test case 4, the subject groups and decomposition methods are all different, resulting in greater variability and hence the performance is less satisfactory.

It is worth noticing that our method greatly outperforms a baseline method called WS (see Figure 4.3) that determines attribute matching based on data distribution at the whole attribute level, which is typical in previous systems such as SemInt [70]. In this figure we also demonstrate the accuracy of the segmented statistics characterization with expert-labeled patterns, meaning that the data is partitioned and aligned in the most accurate way, which marks the best achievable attribute matching performance. But it is not feasible because

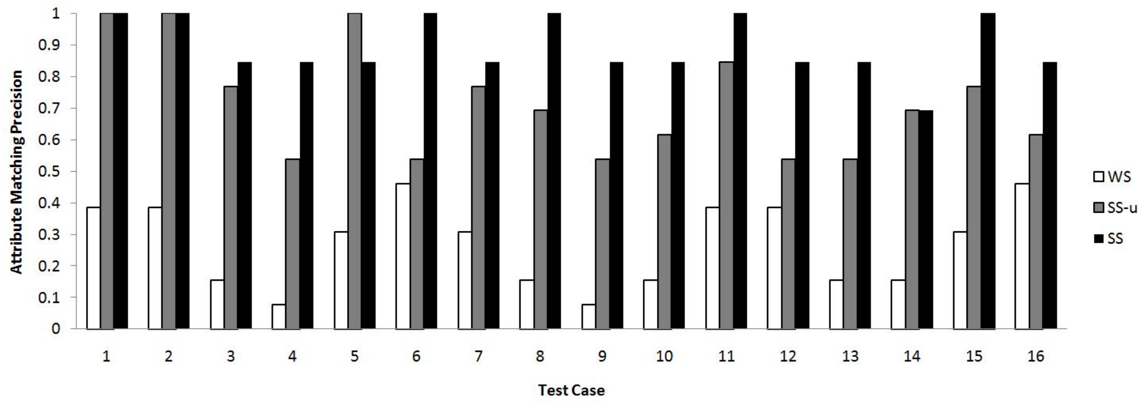


FIGURE 4.3. A comparison between methods on the neuroscience dataset over the 16 test cases is shown. The three methods being compared are matching based on whole-attribute statistics (WS), segmented attribute statistics without knowing a priori cluster matching (SS-u), and segmented attribute statistics with expert-aligned clusterings (SS).

manually recognizing patterns (partitioning data) and aligning them across datasets requires a priori knowledge of attributes in the datasets which is exactly what the problem of attribute matching tries to discover (the circular causality problem). On the other hand, our method does not require human involvement (except the specification of the number of clusters (patterns) present in the data in order to run the clustering analysis) in determining both the attribute matching and cluster matching and is able to achieve close-to-optimal results.

4.3.3. Comparison with Multi-Objective Genetic Algorithm

The concept of genetic algorithms (GA) was developed by Holland and his colleagues [65]. GA is first inspired by the evolutionary process in which weak and unfit species within their environment are faced with extinction and stronger ones have greater opportunities to pass their genes to next generation. Comparing to simulated annealing, GA often offers a different perspective in the field of numerical

Test case	Source params	Target params	P_a	P_c	$\|\Sigma\|$
1	$\langle$ SG1, sICA, m1 $\rangle$	$\langle$ SG1, sICA, m2 $\rangle$	13/13	5/5	5
2	$\langle$ SG1, sICA, m1 $\rangle$	$\langle$ SG2, sICA, m2 $\rangle$	13/13	5/5	6
3	$\langle$ SG1, sICA, m1 $\rangle$	$\langle$ SG1, tPCA, m2 $\rangle$	10/13	5/5	6
4	$\langle$ SG1, sICA, m1 $\rangle$	$\langle$ SG2, tPCA, m2 $\rangle$	7/13	3/5	8
5	$\langle$ SG2, sICA, m1 $\rangle$	$\langle$ SG1, sICA, m2 $\rangle$	11/13	3/5	7
6	$\langle$ SG2, sICA, m1 $\rangle$	$\langle$ SG2, sICA, m2 $\rangle$	13/13	5/5	7
7	$\langle$ SG2, sICA, m1 $\rangle$	$\langle$ SG1, tPCA, m2 $\rangle$	10/13	5/5	6
8	$\langle$ SG2, sICA, m1 $\rangle$	$\langle$ SG2, tPCA, m2 $\rangle$	9/13	2/5	8
9	$\langle$ SG1, tPCA, m1 $\rangle$	$\langle$ SG1, sICA, m2 $\rangle$	7/13	5/5	4
10	$\langle$ SG1, tPCA, m1 $\rangle$	$\langle$ SG2, sICA, m2 $\rangle$	8/13	5/5	6
11	$\langle$ SG1, tPCA, m1 $\rangle$	$\langle$ SG1, tPCA, m2 $\rangle$	11/13	5/5	6
12	$\langle$ SG1, tPCA, m1 $\rangle$	$\langle$ SG2, tPCA, m2 $\rangle$	7/13	3/5	5
13	$\langle$ SG2, tPCA, m1 $\rangle$	$\langle$ SG1, sICA, m2 $\rangle$	7/13	3/5	5
14	$\langle$ SG2, tPCA, m1 $\rangle$	$\langle$ SG2, sICA, m2 $\rangle$	9/13	5/5	6
15	$\langle$ SG2, tPCA, m1 $\rangle$	$\langle$ SG1, tPCA, m2 $\rangle$	10/13	3/5	8
16	$\langle$ SG2, tPCA, m1 $\rangle$	$\langle$ SG2, tPCA, m2 $\rangle$	8/13	3/5	8

TABLE 4.3. The performance of MOSA on the neuroscience dataset over the 16 test cases. The source and target parameter configuration of the data acquisition process of each test case are shown. P_a and P_c denote the accuracy of attribute matching and cluster matching respectively. Σ is the number of points in the obtained Pareto-front. The quantities listed in the table are obtained by averaging over 5 runs of each test.

optimization. Starting from a number of random generated population and then performing cross over and evolve, GA has the ability to search in parallel around different and often fully scattered instances in the solution space, in contrast to the "single thread" search in simulated annealing. We also implemented the Multi-Objective Genetic Algorithm (MOGA) developed by Fonseca *et al.* [85] as a metaheuristic to solve the dual matching problem.

To compare the performance of GA and SA, we first carry out an experiment on the same set of neuroscience data, as shown in Table 4.4. The iteration parameters of both algorithms are tuned so that the convergence time are about the same. The performance are then compared under such setting. We manually examine the Pareto front derived in each test case and find the solution that is the closest to the gold standard and the accuracies are reported in Table 4.4 (each number is averaged over 5 independent runs).

Test Case	P_a (%)	P_c (%)	Σ
1	100	100	9
2	98.2	96.6	10
3	53.4	98.0	9
4	53.3	98.0	11
5	100	98.2	5
6	71.2	96.0	6
7	59.4	94.4	6
8	59.7	98.8	6
9	25.2	100.0	6
10	38.5	100.0	5
11	77.7	99.2	7
12	69.2	100.0	9
13	38.7	100.0	9
14	40.3	98.8	11
15	45.0	96.0	8
16	84.6	98.8	16

TABLE 4.4. The performance of MOGA on the neuroscience dataset over the 16 test cases. The source and target parameter configuration of each test case is the same as in Table 4.3.

The number of population kept in each generation is an important parameter regarding the complexity and performance in MOGA. Intuitively, the more instances we keep, the broader the search space we can explore in each generation. Table 4.4 shows the result with the number of population set to 4. We have also tested other settings and found out that the accuracy in most cases increase with the number of population but in rare cases the performance deteriorates. The overall performance of MOGA is comparable to that of MOSA but appears to be less robust. It is worth noticing that the metaheuristics (MOSA and MOGA) we employed in the experiments are simple algorithms. More modern and sophisticated methods that explore various fitness assignment procedure, elitism, or diversification approaches will be very likely to improve the performance.

		fixed acidity	volatile acidity	citric acid	residual sugar	chlorides
mean	data1	6.86	0.28	0.34	6.35	0.05
	data2	6.85	0.28	0.33	6.43	0.05
stdev	data1	0.84	0.1	0.12	4.98	0.02
	data2	0.86	0.1	0.12	5.16	0.02

		total sulfur dioxide	density	pH	sulphates	alcohol	quality	free sulfur dioxide
mean	data1	138.98	0.99	3.19	0.49	10.53	5.88	35.58
	data2	137.68	0.99	3.19	0.49	10.49	5.88	35.02
stdev	data1	41.86	0.02	0.16	0.11	1.25	0.89	16.4
	data2	43.18	0	0.15	0.12	1.22	0.89	17.61

TABLE 4.5. Statistical distribution of attributes in the Wine Quality dataset.

4.3.3.1. Wine Quality Dataset

To further evaluate our method, we carried out another experiment on a real-world wine quality dataset [86] that is available through the UCI machine learning repository[1]. This dataset has 12 attributes and 4898 records. We apply uniform sampling to split it into two equal-sized subsets. The attributes are anonymized and randomly reordered in each subset to generate artificial heterogeneity.

We apply the proposed method with MOSA and MOGA as metaheuristics respectively. The test is focused on attribute matching because the gold standard is known while the gold standard of cluster matching is unknown. Table 4.5 summarizes the statistics for each attributes in the dataset. For both MOSA and MOGA derived Pareto optimal solutions, we manually select the one that is the closest to the gold-standard matching (e.g., the solution with 10 out 12 attributes matched correctly). Each metaheuristic is invoked 5 times and the matching

[1]`http://archive.ics.uci.edu/ml/datasets/Wine+Quality`

accuracy is averaged over these runs. The performance for attribute matching is shown in Table 4.6. The result demonstrates a markedly high accuracy for both MOSA and MOGA. We notice that in most runs the Pareto fronts derived from MOSA and MOGA contain the gold standard matching (hence the high accuracy). It suggests a strategy to reduce the Pareto front in the matching problem by running MOSA or MOGA repeatedly after some times and only those "stable" points that appear more than certain proportion of the times are considered to be presented to decision makers.

	MOSA	MOGA
accuracy (%)	95.5	92.3
running time	517	3356

TABLE 4.6. The performance of MOSA and MOGA on the Wine Quality dataset.

4.4. Discussion

4.4.1. Choices of the Data Representation Methods

Our choices of the methods to represent attributes and clusters are constrained by the specific challenges that are present in the matching problems.

Due to the nature of many scientific datasets, especially such as those in the neuroscience case study, our work on attribute matching is faced with following unique challenges. First, the data under study is semi-structured, thus invalidating those matching methods that presume a complete, known-in-advance schematic structure. In addition, totally different labels (usually acronyms or pseudowords) are widely adopted for the same or similar metrics, rendering lexical similarity-based methods unsuitable.

Moreover, an important limitation of previous instance-based matching methods is their inability to handle numerical instances. Only a handful number of existing methods have shown good performance on matching numeric attributes. iMAP [72] and SemInt [70] are two such methods; each of them has assumptions that make them unsuitable for our task in the neuroscience case study. The iMAP method requires the existence of joint paths between two tables through which data instances can be cross-referenced; however, two datasets can be drawn from different cohorts and therefore cannot be cross-referenced, because there are no overlapping instances. The SemInt method calculates statistics, such as maximum, minimum, mean, variance, *etc.*, of data content to characterize numeric attributes. The statistics are extracted by running aggregation queries against the whole set of attribute values. However, it is possible that two different attributes could have similar mean values, such as shown in the synthetic data case study; thus, SemInt statistics may be too coarse-grained to represent distinct ERP attributes.

Therefore, we choose to represent attributes using the segmented statistical characterization method to examine the grouping structure of attribute values, thus supporting fine-grained comparisons between attributes. As a result, we are able to calculate the straightforward Euclidean distance between attributes and to accurately capture the dissimilarity between them.

On the other hand, we have formulated the pattern matching problem motivated in the cross-lab collaborative ERP analysis as the cluster comparison problem. The cluster comparison problem is closely related to the cluster validity problem, such as the technique of external, or relative, indexing, which is used to compare different clustering results. Most previous methods based the comparison on evaluation of cluster membership ([87–89]). However, these methods are inappropriate for comparison of clustering results based on different datasets. Our motivation, in particular, is to find correspondences among ERP patterns from distinct datasets with non-overlapping observations (different study participants) in the neuroscience case study. For this reason, we examine methods that does not assume overlap in cluster membership across datasets.

We therefore choose to represent clusters as density profiles and use the ADCO clustering similarity index [74]) that is based on the density profile representation. The density profile representation does not assume common cluster membership and the ADCO measure can determine the similarity between two clusterings based on the distribution of data points along each attribute.

4.4.2. Single Objective vs. Multi-objective Approaches

In the work reported in [73, 80] we assume the cluster matching is known prior to the attribute matching. Then the attribute matching alone is simply a single objective problem. However, as we pointed out in the Introduction section, this is a gross simplification because attribute matching and cluster matching are intertwined and usually none can be known without the knowledge of the other. Therefore in this work, we focus on tackling this deadlock.

We argue that the single objective approach is not applicable given the way we represent attributes and clusters. Specifically, we represent an attribute as an ordered tuple, $< v_1, v_2, \ldots, v_3 >$, where v_i is some statistics of the attribute in a cluster c_i of one dataset. Two attributes from different datasets can be compared only when we are able to arrange the tuples so that matching positions correspond to the same cluster. This assumes a certain kind of cluster matching. Vice versa, it is also true for cluster matching in that we need some input on attribute matching.

Essentially the problem at hand is to search in two permutation spaces, one for each matching problem, which naturally leads to our multi-objective approach. If one was to adopt a single objective approach, the two spaces would have to be concatenated and variables aggregated by some functions (e.g., weighted sum). We argue it might be flawed because there is no way to justify the ad hoc choice of such functions. On the contrary, the multi-objective approach based on Pareto optimality circumvents the choice of aggregation, but focuses on obtaining a non-dominating set of solutions (the Pareto set). We demonstrate in our case studies one simple way to utilize the Pareto set by combining both objectives based on weights that are determined through the pilot experiments. Note that applying weights before and after the optimization is fundamentally different. The former carries more systematic risk of missing true optimum due to the arbitrary choice of weights, while the latter is just one way to post-process the Pareto set that is very likely to contain the optimum. In practice, the Pareto set itself can be well treated as the final product of the matching analysis. Note that we show the sizes of Pareto sets in Table 4.3 for the neuroscience test case, which are all reasonably small for examination to hand-pick best solutions.

CHAPTER V

GRAPH-BASED MINING FOR SEMANTICALLY ASSOCIATED ITEMSETS

I have described the unified representation for both data and ontologies based on RDF hypergraphs or bipartite graphs in Chapter III, as well as methods to resolve heterogeneities from disparate sources in Chapter IV. The main research challenge remaining is to develop appropriate analysis methods based on the unified graph representation to solve data mining problems. In this and the next chapters, several such methods are described and their capabilities, limitations and possible directions for improvements are studied.

This chapter focuses on a particular mining task that aims at finding semantically associated itemsets to showcase the utility of the methods. More specifically, if a mining task does not require the use of domain knowledge from ontologies, the RDF hypergraph can be coarsened to a compact form for better scalability. The emphasis of this chapter is to present details of the coarsened RDF hypergraph and similarity measures designed based on it to discover semantically associated itemsets without the incorporation of ontologies. I will cover the usage of the RDF bipartite graph in cases where ontologies are needed in the next chapter.

This chapter consists of work published in “Proceedings of the 11th IEEE International Conference on Data Mining” in 2011 [90]. Dr. Dejing Dou and Dr. Ruoming Jin provided valuable insights on the design of the hypergraph-based similarity measures. Dr. Paea LePendu and Dr. Nigam Shah contributed the electronic health dataset and helped evaluate the experimental results.

5.1. Overview

5.1.1. Semantically Associated Itemsets

The problem we aim to solve is to find semantically associated itemsets, a particular kind of frequent itemset mining task. In the traditional sense, an itemset is called frequent if its support (number of times the itemset occur in the dataset) is no less than a given threshold. The original goal for finding associations came from the need to analyze supermarket customer behavior in terms of products that are often purchased together. However, we notice that the measure of support essentially restrains pattern discovery to account for only directly associated items (*e.g.*, products purchased together in one transaction) while ignoring possible indirect ones. A prominent example of meaningful indirect associations was given by Swanson's landmark paper published in 1987 [24] that described the relationship between fish oil and Raynauld's syndrome through their mutual connections with some certain changes in blood.

Such indirect associations can be best captured by graphs. In general, an object set endowed with pairwise relationships can be conceptually viewed as a graph in which vertices represent objects, and any two vertices that have some kind of relationship are joined together by an edge. In this sense the traditional measure of support evaluates the significance of an itemset by the number of direct edges (of one-hop length) between item nodes. Extending this notion to allow paths with arbitrary lengths to be taken into account, we are able to evaluate the significance of an itemset in terms of the indirect connections among its nodes. From here on, we call the itemset associated by the indirect connection via multi-hop paths the *semantically associated itemset*, or simply the *semantic association*. In this chapter, we focus on describing graph-based algorithms to find semantically associated itemsets.

The usage of the term semantic association conforms with the definition proposed by Sheth *et al.* [91] for connections between entities in an RDF graph. Specifically, they defined the semantic association based on if there exists a sequence of interconnected links between two given entities. In our study of semantically associated itemsets in transaction data, the link between entities can be as simple as the "co-occurrence" relationship if more complicated relationships in ontologies are not concerned. Under Sheth *et al.*'s definition, the semantic association between transaction items i_0 and i_n can be established by identifying a link of the form $i_0, P_c, i_1, P_c, \ldots, i_{n-1}, P_c, i_n$, in which P_c denotes the property, or relationship, that connects two items (*e.g.*, co-occurrence). Given this, the problem of finding meaningful semantic association becomes how to define a proper graph representation and effective analysis methods that can be carried out to evaluate the strength of semantic associations.

To develop solutions for semantically associated itemsets, we first rule out simple graphs as the candidate representation of data due to the ambiguity and information loss, as is illustrated in Section 2.3.1, Chapter II. The RDF hypergraph or bipartite graph comes to remedy as it preserves the semantics in the original table and contains no ambiguity. It is also able to represent ontologies in the same way so that analysis approaches on the RDF hypergraph can utilize information from both data and domain knowledge. However, if a mining task does not require the use of domain knowledge from ontologies, the RDF hypergraph can be coarsened to a more compact form to achieve better scalability. In the rest of this chapter, we describe in detail the methods for discovering semantically associated itemsets on the coarsened RDF hypergraph without the incorporation of ontologies.

5.2. Method

In this section, we present our method for discovering semantically associated itemsets based on hypergraphs when ontologies are not present in the mining task. We first introduce an alternative hypergraph representation that is more compact to model the data. The process to generate such hypergraph is called *RDF hypergraph coarsening* as described in Definition 5.1. Then, two similarity measures based on the coarsened hypergraphs are described to discover semantically associated 2-itemsets. Finally, methods to generate k-itemsets are presented.

5.2.1. RDF Hypergraph Coarsening

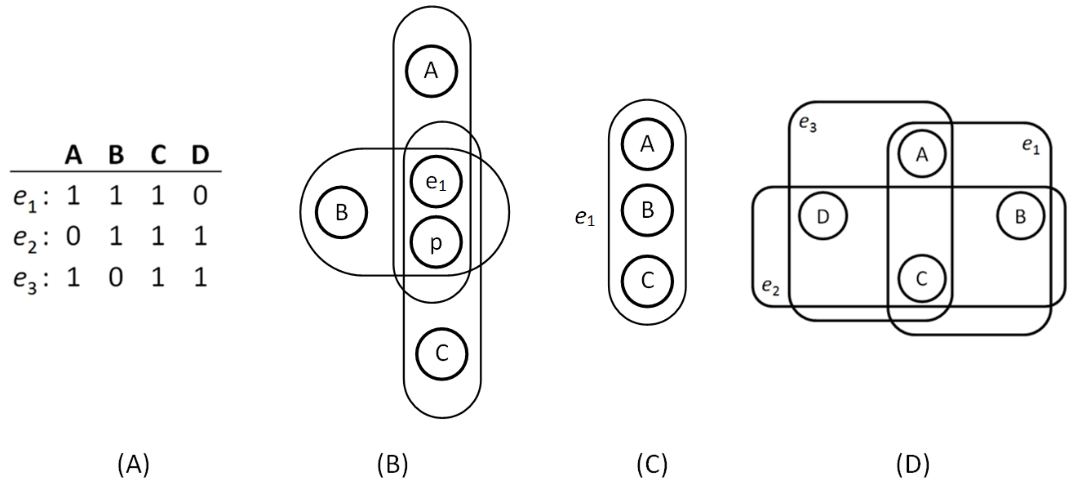


FIGURE 5.1. An example of the hypergraph coarsening process.

Definition 5.1 (RDF hypergraph coarsening). RDF hypergraph coarsening is the process of generating a compact form given an input RDF hypergraph for a relational table by merging vertices into larger groups and removing less significant vertices. The choice of vertices is pertinent to specific mining tasks. Notice that RDF hypergraph is 3-uniform and in the case of RDF hypergraph for relation tables, each hyperedge has three nodes corresponding to the RDF statement of the form `<row>, <p>, <column>`. The `<p>` node is an auxiliary predicate denoting the context-independent semantic relationship between the row and column nodes (such as the general `<mentions>` relationship), and since it is incident to all RDF hyperedges it is first removed in the coarsening process as it bears the least amount of information. Next, if the mining task focuses on discovering patterns among column nodes (such as in frequent pattern mining), we can place column nodes that coincide with the same row nodes into a new hyperedge and subsequently remove the row node. The result is a column-oriented coarsened RDF hypergraph. The vice versa can be carried out for mining tasks that focus on row nodes (such as clustering).

Example 5.2 (**Generation of a column-oriented coarsened hypergraph for a relational table**)**.** Figure 5.1 (A) shows a sample relational table. Using the method described in Example 3.9, Chapter III, we can represent this binary-valued table to an RDF hypergraph as is shown in Figure 5.1 (B). We can see there are three hyperedges for the first row in the table corresponding to three RDF statements, *i.e.*, <e_1, `p, A`>, <e_1, `p, B`>, and <e_1, `p, C`>. Figure 5.1 (C) illustrates the coarsened hypergraph according to Definition 5.1. Supposing we are interested in discovering relationships between column nodes `A, B` and `C` in a frequent pattern mining task, we can remove the nodes e_1 and `p` that are commonly incident to all the three hyperedges, and then place nodes `A, B` and `C` on a single hyperedge. Figure 5.1 (D) shows the coarsened hypergraph for all rows from the relational table in Figure 5.1 (A).

Given this method, we can construct a coarsened hypergraph for any relational table in mining tasks where ontologies are not required. The relational attributes constitute the universe of vertices in the hypergraph. Based on the coarsened hypergraph, our approach for mining semantically associated itemsets starts by first generating 2-itemsets as detailed below.

5.2.1.1. Methods for Generating 2-itemsets

A 2-itemset $\langle i, j \rangle$ is considered semantically associated if the hypergraph-based similarity measure $s(i, j)$ exceeds some threshold. In the following, we describe two similarity measures s_{CT} and s_{L+} based on, respectively, the average commute time distance on hypergraphs and the inner-product-based representation of the pseudoinverse of hypergraph Laplacian. Given discovered semantically associated 2-itemsets, we propose a hypergraph expansion method along with two search strategies, namely, the clique and connected component search, in the resulting graph for finding semantically associated k-itemsets ($k > 2$).

We first introduce the concept of random walk on hypergraphs as an extension to random walk on simple graphs. Several key quantities are defined, especially the Laplacian for hypergraphs, based on which the similarity measures s_{CT} and s_{L+} can be calculated.

Random Walk on Hypergraphs

We can associate each hypergraph with a natural random walk which has the transition rule as described in [92]. Given the current position $u \in V$; first choose a hyperedge e over all hyperedges incident with u with the probability proportional to $w(e)$ (the edge weight); and then choose a vertex $v \in e$ uniformly at random. Obviously, it generalizes the natural random walk defined on simple graphs. Let $\mathbf{P}$ denote the transition probability matrix of this hypergraph random walk. Then each entry of $\mathbf{P}$ is

$$p(u,v) = \sum_{e \in E} w(e) \frac{h(u,e)}{d(u)} \frac{h(v,e)}{\delta(e)} .$$

In matrix notation, $\mathbf{P} = \mathbf{D}_v^{-1}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T$. Zhou et al. [92] defined the following normalized hypergraph Laplacian $\mathcal{L}$ based on the random walk model:

$$\mathcal{L} = \mathbf{I} - \boldsymbol{\Theta}, \text{ where } \boldsymbol{\Theta} = \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}}. \qquad \text{(Equation 5.1.)}$$

Average Commute Time Similarity s_{CT}

To compute commute-time distance between vertices in a hypergraph, we need to first define the combinatory hypergraph Laplacian $\mathbf{L}$. It follows from Zhou et al's definition of normalized hypergraph Laplacian in Equation 5.1:

$$\mathbf{L} = \mathbf{D}^{1/2}\mathcal{L}\mathbf{D}^{1/2} = \mathbf{D}_v - \mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T \qquad \text{(Equation 5.2.)}$$

The average commute time $n(i,j)$ on simple graph can be computed in closed form from the Moore-Penrose pseudoinverse of $\mathbf{L}$ [57], denoted by $\mathbf{L}^+$ with elements $l_{ij}^+ = [\mathbf{L}^+]_{ij}$. It can be shown that $n(i,j)$ on hypergraph can be calculated in the same manner. The pseudoinverse $\mathbf{L}^+$ is given by the following equation:

$$\mathbf{L}^+ = (\mathbf{L} - \mathbf{e}\mathbf{e}^T/n)^{-1} + \mathbf{e}\mathbf{e}^T/n, \qquad \text{(Equation 5.3.)}$$

where $\mathbf{e}$ is a column vector made of 1s (i.e., $\mathbf{e} = [1, 1, \ldots, 1]^T$). The formula for the computation of $n(i, j)$ takes the form of the following equation:

$$n(i, j) = V_G(l^+_{ii} + l^+_{jj} - 2l^+_{ij}), \qquad \text{(Equation 5.4.)}$$

where $V_G = tr(\mathbf{D}_v)$ is the volume of the hypergraph. If we define $\mathbf{e}_i$ as the ith column of $\mathbf{I}$ (i.e., $\mathbf{e}_i = [\underset{1}{0}, \ldots, \underset{i-1}{0}, \underset{i}{1}, \underset{i+1}{0}, \ldots, \underset{n}{0}]^T$), Equation 5.4 can be transformed to:

$$n(i, j) = V_G(\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{L}^+ (\mathbf{e}_i - \mathbf{e}_j), \qquad \text{(Equation 5.5.)}$$

Since $n(i, j)$ can be proven to be a distance, it is straightforward to convert it to a similarity measure $s_{CT}(i, j)$ by, for example, calculating the reciprocal $1/n(i, j)$.

Pseudoinverse-based Inner-Product Similarity s_{L+}

Equation 5.5 can be mapped into a new Euclidean space that preserves the commute time distance:

$$\begin{aligned} n(i, j) &= V_G(\mathbf{e}_i - \mathbf{e}_j)^T \mathbf{L}^+ (\mathbf{e}_i - \mathbf{e}_j) \\ &= V_G(\mathbf{x}'_i - \mathbf{x}'_j)^T (\mathbf{x}'_i - \mathbf{x}'_j) \\ &= V_G \|\mathbf{x}'_i - \mathbf{x}'_j\|^2, \qquad \text{(Equation 5.6.)} \end{aligned}$$

where $\mathbf{x}'_i = \mathbf{\Lambda}^{1/2} \mathbf{U}^T \mathbf{e}_i$, $\mathbf{U}$ is an orthonormal matrix made of eigenvectors of $\mathbf{L}^+$ (ordered in decreasing order of corresponding eigenvalue λ_k) and $\mathbf{\Lambda} = \mathbf{Diag}(\lambda_k)$. In this way, the transformed node vectors $\mathbf{x}'_i$ are exactly separated in the new n-dimensional Euclidean space. From this definition, it follows that $\mathbf{L}^+$ is the matrix containing inner products of the transformed vectors $\mathbf{x}'_i$ as shown below:

$$\begin{aligned}\mathbf{x}_i'^T \mathbf{x}'_j &= (\mathbf{\Lambda}_i^{1/2}\mathbf{x}_i)^T \mathbf{\Lambda}_j^{1/2}\mathbf{x}_j = \mathbf{x}_i^T \mathbf{\Lambda} \mathbf{x}_j \\ &= \mathbf{e}_i^T \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \mathbf{e}_j = \mathbf{e}_i^T \mathbf{L}^+ \mathbf{e}_j = l_{ij}^+. \end{aligned} \qquad \text{(Equation 5.7.)}$$

Therefore, $\mathbf{L}^+$ can be considered as a similarity matrix for the nodes—that is

$$s_{L^+}(i,j) = l_{ij}^+ \,. \qquad \text{(Equation 5.8.)}$$

The inner-product-based similarity measures are well-studied for the vector-space model of information retrieval. It has been shown that when computing proximities between documents, inner-product-based measures outperform Euclidean distances [93].

5.2.1.2. Effective Computation

In high dimensional data sets, the computation of the hypergraph Laplacian and the pseudoinverse becomes intractable. We discuss two approaches to mitigate this scalability problem.

To compute hypergraph Laplacian $\mathbf{L}$ in Equation 5.2 requires multiplication of hypergraph incidence matrices $\mathbf{H}$ and its transpose $\mathbf{H}^T$. Since $\mathbf{H}$ grows in proportion to the size of underlying transaction data (each node corresponds to a column and each hyperedge corresponds to a row), it eventually becomes unable to fit in memory when the size exceeds a certain amount. In this case the computation can still be carried out using a block partitioned matrix product by performing operations only on the submatrices of tractable sizes. Owing to the fact that, in most cases, $|V|$ is much smaller than $|E|$, $\mathbf{H}$ can then be partitioned into s

vertical stripes and the square matrix $\mathbf{D}_e$ into s diagonal blocks. The multiplication in Equation 5.2 can be calculated by $\mathbf{H}\mathbf{D}_e^{-1}\mathbf{H}^T = \sum_{\gamma=1}^{s} \mathbf{H}_\gamma \mathbf{D}_{e\gamma}^{-1} \mathbf{H}_\gamma^T$. Notice that $\mathbf{H}$ is sparse in many applications which can be exploited to gain high performance.

As the number of nodes grows, to compute pseudoinverse in closed form using Equation 5.3 also becomes intractable. A procedure based on Cholesky factorization to compute $\mathbf{L}^+$ for large sparse matrices [94] allows to compute $\mathbf{L}^+$ in a column-by-column manner. In particular, the procedure involves the following steps for computing the ith column of $\mathbf{L}^+$:

1. Compute the projection $\mathbf{y}_i$ of base vector $\mathbf{e}_i$ on the column space of $\mathbf{L}$.

2. Find a solution l_i^{*+} of the linear system $\mathbf{L}\mathbf{l} = \mathbf{y}_i$.

3. Project l_i^{*+} on the row space of $\mathbf{L}$ to get l_i^+.

Since $\mathbf{L}$ is symmetric, its row space is the same as its column space. The projection in step 1 and 2 can be represented by the matrix $(\mathbf{I} - \mathbf{e}\mathbf{e}^T/n)$. The equation in step 2 can be solved by first solving a reduced linear system: $\hat{\mathbf{L}}\hat{\mathbf{l}} = \hat{\mathbf{y}}_i$, where $\hat{\mathbf{L}}$, $\hat{\mathbf{l}}$, and $\hat{\mathbf{y}}$ are obtained respectively by removing the last row from $\mathbf{l}$, $\mathbf{y}$, and last row and column from $\mathbf{L}$. We observe that $\hat{\mathbf{L}}$ is full rank and positive definite and hence is able to be decomposed using the Cholesky factorization, $\hat{\mathbf{L}} = \mathbf{R}\mathbf{R}^T$. Since $\mathbf{R}$ is lower-triangular, one solution of $\hat{\mathbf{L}}\hat{\mathbf{l}} = \mathbf{R}\mathbf{R}^T\hat{\mathbf{l}} = \hat{\mathbf{y}}_i$ can be efficiently obtained by two back-substitutions. After solving the reduced linear system, the solution to the original equation in step 2 is therefore $(\mathbf{l}_i^{*+}) = [\hat{\mathbf{l}}_i^{*+}, 0]^T$. With the help of this technique, we are able to analyze datasets of a million rows and 10 thousand columns.

5.2.1.3. Methods for Generating k-itemset ($k > 2$)

Now, we consider finding semantically associated k-itemset ($k > 2$) from given 2-itemsets. As is common in hypergraph theory, we can associate an induced graph $G(H)$ with every hypergraph H by expanding every hyperedge e in H to a clique in $G(H)$. Edges in the induced graph $G(H)$ can be called *subedges* to avoid unnecessary confusion. We can further construct a pruned graph $G'(H)$ from $G(H)$ by applying the following inclusion rule on each subedge: the similarity between the incident nodes of a subedge has to be greater than a user-specified threshold θ. More formally, given a hypergraph $H = (V, E)$, the pruned subgraph is defined as $G'(H) = \{V, E'\}$, where

$$\begin{aligned} E' = \{(u, v) \in V^2 \ : \ & u \neq v \text{ and} \\ & u, v \in e \text{ for some } e \in E \text{ and} \\ & s(u, v) > \theta\}. \end{aligned}$$

We only use the pruned induced graph to model the local neighborhood relationship between data points, which is essentially a *similarity graph* for data points under the hypergraph similarity measures (*i.e.*, s_{CT} or s_{L+}). It is worth noticing that each hypergraph has a unique induced graph but the same is not true the other way around. The induced graph alone is not an ideal representation of the data (see the discussion on the ambiguity of Gaifman graphs in Section 2.3.1, Chapter II). Therefore we develop ways to use hypergraphs and hypergraph-based measures to characterize the similarity between data points.

Given $G'(H)$, finding semantically associated k-itemset ($k > 2$) can be solved in two ways: finding cliques or connected components in $G'(H)$.

5.2.1.4. Cliques of $G'(H)$

Finding cliques in $G'(H)$ corresponds to searching and testing in the powerset of V. Given the fact that every subset of a clique is also a clique, this downward-closure property can make clique discovery algorithm efficient in a way similar to the Apriori algorithm for finding frequent itemsets — with a "bottom up" manner, the candidate generation step extends valid $k-1$ length itemsets one item at a time, and groups of candidates are tested against $G'(H)$ to determine if they form cliques. The algorithm terminates when no further successful extensions are found.

5.2.1.5. Connected Components of $G'(H)$

Complete subgraph (*i.e.*, clique) is a very strong requirement that can limit the approach to restricted cases of semantically associated itemsets. One way to relax this requirement is to find connected components of $G'(H)$, which can be viewed as a closure under semantic association. The number of connected components equals the multiplicity of the eigenvalue 0 of the Laplacian matrix of $G'(H)$. Although the set of connected components is not downward closed, there is efficient way to find all connected components of a graph in linear time using either breadth-first search or depth-first search. In either case, a search that begins at some particular vertex will find the entire connected component containing the vertex. When the search returns, loop through other vertices and start a new search whenever the loop reaches a vertex that has not already been included in a previously found connected component.

5.2.1.6. Ranking of Itemsets

Once the semantically associated 2-itemsets and k-itemsets are generated, they can be ranked by a quantity indicating the strength of association among items in the set. We compute this quantity by averaging the total pairwise similarities over the number of subedges of the itemset's corresponding clique or connected component in $G'(H)$.

5.3. Case Studies

Because we are interested in understanding the differences between the s_{CT} and s_{L+} similarity measures for generating semantically associated itemsets, we conducted a series of experiments to highlight their tradeoffs. First, to illustrate the power of hypergraphs in finding associations via linking items, we synthesized a dataset for the *fish oil* example. Next, to illustrate the tradeoffs between the two methods, we evaluated both methods against a commonly used *shopping cart* dataset. Finally, encouraged by these results, we applied these methods to actual *electronic health records* to highlight their scalability and applicability to the medical domain.

5.3.1. Fish Oil

5.3.1.1. Dataset

As mentioned in Section 5.1.1, *fish oil* and *Raynaud's syndrome* have been shown by Swanson [24] to be linked together indirectly via various *blood changes*. He found these associations from examining biomedical texts. As a proof of concept, we replicated this situation by synthesizing a table of 50 rows, which is about the same scale as in Swanson's experiment. Each row represents a set of terms generated to represent biomedical text. Each set of terms was specifically generated so that *fish oil* and *Raynaud's syndrome* never appear together. The column headers include *fish oil, blood changes*, *Raynaud's syndrome*. Six other random variables acted as noise. We then applied the s_{CT}, s_{L+} to the dataset. Specifically, we set a threshold for first generating top-15 2-itemsets using either similarity measure. Based on the generated 2-itemsets we used clique search to generate $(k > 2)$-itemsets.

5.3.1.2. Results

The hypergraph approach finds significant links between *fish oil* and *Raynaud's syndrome*, as demonstrated particularly well by the s_{CT} method as shown in Table 5.1. Even the triplet was discovered by the clique search technique. Most notably, because their co-occurrence is zero, the association would never be discovered by traditional frequent itemset techniques such as the Apriori algorithm [95].

The s_{L+} method also picks-up the association, but it was fairly weak: the association is ranked 23rd among all 2-itemsets (column 3 in Table 5.1 lists the ranking of the s_{CT} results given by the s_{L+}). However, as our next evaluations suggest, s_{L+} demonstrates other favorable qualities.

5.3.2. Shopping Cart

5.3.2.1. Dataset

To better understand how the s_{CT} method compares against the s_{L+} method, we tested them on a business shopping cart dataset. This dataset contains purchase

$\mathbf{s_{CT}}$	$\mathbf{s_{L+}}$ rank	**Freq**	**Itemset**
0.83	2	25	⟨ *blood_change, fish_oil* ⟩
0.83	1	25	⟨ *blood_change, Raynaud_synd* ⟩
0.79	–	**0**	⟨ ***blood_change, fish_oil, Raynaud_synd*** ⟩
0.76	–	10	⟨ *blood_change, fish_oil, f* ⟩
0.76	7	16	⟨ *blood_change, f* ⟩
0.76	6	16	⟨ *blood_change, d* ⟩
0.76	3	16	⟨ *blood_change, b* ⟩
0.75	9	15	⟨ *blood_change, a* ⟩
0.75	4	15	⟨ *blood_change, e* ⟩
0.73	10	14	⟨ *blood_change, c* ⟩
0.72	**23**	**0**	⟨ ***fish_oil, Raynaud_synd*** ⟩
0.70	10	10	⟨ *fish_oil, f* ⟩
0.70	–	10	⟨ *fish_oil, d* ⟩
0.70	9	9	⟨ *fish_oil, b* ⟩
0.68	20	6	⟨ *Raynaud_synd, f* ⟩

TABLE 5.1. Top semantically associated itemsets on the synthetic dataset generated by s_{CT}.

information on 100 grocery items (represented by Boolean column headers) for 2,127 shopping orders (corresponding to tuples). We applied s_{L+} and s_{CT} and set a threshold to include top-100 2-itemsets, based on which we subsequently used clique search to generate ($k > 2$) itemsets. The top-10 2-itemset results and ($k > 2$)-itemsets corresponding to maximum cliques generated by s_{CT} and s_+ are reported in Table 5.2 and 5.3 respectively.

5.3.2.2. Results

Unlike the experiment on the fish oil dataset, we do not have specific hypothesis to validate in this test. After examining the results from both measures, we can only conclude they make intuitive sense. However, we observe that the difference between the s_{CT} and s_{L+} becomes more significant in this experiment. The s_{CT} tends to include itemsets with high support and the effect of indirect links

	$\mathbf{s_{CT}}$	**Freq**	**Itemset**
2-itemsets	0.74	39	$\langle$ *Cheese, Soup* $\rangle$
	0.73	32	$\langle$ *Cheese, Dried Fruit* $\rangle$
	0.72	36	$\langle$ *Dried, Fruit Soup* $\rangle$
	0.72	38	$\langle$ *Cookies, Soup* $\rangle$
	0.71	24	$\langle$ *Cheese, Cookies* $\rangle$
	0.70	30	$\langle$ *Cookies, Dried Fruit* $\rangle$
	0.68	31	$\langle$ *Cheese, Preserves* $\rangle$
	0.67	24	$\langle$ *Cheese, Wine* $\rangle$
	0.67	21	$\langle$ *Preserves, Soup* $\rangle$
	0.67	28	$\langle$ *Soup, Wine* $\rangle$
(k>2)-itemsets	0.64	0	$\langle$ *Canned Vegetables, Cheese, Cookies, Dried Fruit, Frozen Vegetables, Nuts, Preserves, Soup, Wine* $\rangle$

TABLE 5.2. Top s_{CT} results on the shopping cart dataset.

is less pronounced. On the other hand, s_{L+} promotes items with support values towards the lower end. We also observe one drawback of the s_{CT} that the result is centered around items with large frequencies (*i.e.*, many direct links to other nodes) and hence in a sense limiting the information (most itemsets are about *cheese*, *soup* and *cookies*). By contrast, s_{L+} produces more diversified itemsets. This point can be further illustrated in Table 5.4 where we list the top 2-itemsets ranked by the frequency of occurrences. The list is similar to the s_{CT}-based result but vastly different from the s_{L+}-based result.

Finally we tested our methods on the dataset of electronic health records of real patients. This dataset is different from the above two datasets not only in scale but also in practical importance as described in the following.

	$\mathbf{s_{L+}}$	**Freq**	**Itemset**
2-itemsets	10.17	3	$\langle$ *Sardines, Conditioner* $\rangle$
	8.17	6	$\langle$ *Toothbrushes, Nasal Sprays* $\rangle$
	6.70	6	$\langle$ *Yogurt, Anchovies* $\rangle$
	6.25	5	$\langle$ *Sports Magazines, Cottage Cheese* $\rangle$
	5.82	5	$\langle$ *Tofu, Sour Cream* $\rangle$
	5.79	3	$\langle$ *Toothbrushes, Acetominifen* $\rangle$
	4.77	4	$\langle$ *Sauces, Nasal Sprays* $\rangle$
	4.46	3	$\langle$ *Sports Magazines, Gum* $\rangle$
	4.43	4	$\langle$ *Sunglasses, Paper Dishes* $\rangle$
	4.05	5	$\langle$ *Tofu, Canned Fruit* $\rangle$
(k>2)-itemsets	4.51	2	$\langle$ *Canned Fruit, Sour Cream, Tofu* $\rangle$
	2.01	1	$\langle$ *Batteries, Cereal, Cooking Oil* $\rangle$
	1.75	5	$\langle$ *Canned Vegetables, Nuts, Waffles* $\rangle$

TABLE 5.3. Top s_{L+} results on the shopping cart dataset.

Itemset	**Freq**
$\langle$ *Cheese , Soup* $\rangle$	39
$\langle$ *Cookies , Soup* $\rangle$	38
$\langle$ *Dried Fruit , Soup* $\rangle$	36
$\langle$ *Cheese , Dried Fruit* $\rangle$	32
$\langle$ *Cheese , Preserves* $\rangle$	31
$\langle$ *Cookies , Dried Fruit* $\rangle$	30
$\langle$ *Cereal , Soup* $\rangle$	29
$\langle$ *Cookies , Preserves* $\rangle$	29
$\langle$ *Frozen Vegetables , Soup* $\rangle$	29
$\langle$ *Nuts , Preserves* $\rangle$	29

TABLE 5.4. Top itemsets ranked by the frequency of occurrences on the shopping cart dataset.

5.3.3. Electronic Health Records

5.3.3.1. Dataset

In our final evaluation, we analyzed the electronic health records of real patients. Applying methods like the ones we have described to this kind of data is particularly relevant because of recent legislation aimed at increasing the meaningful use of electronic health records. Discovering meaningful semantically associated itemsets among the set of drugs and diseases identified in the patient's clinical note is a critical step toward identifying combinations of drug classes and co-morbidities, or risk-factors and co-morbidities that are common in patients with a certain outcome (for example, those suffering from myocardial infarction), toward building predictive risk models, as well as toward providing probable hypotheses about the possible causes of that outcome.

We obtained the set of drugs and diseases for each patient's clinical note by using a new tool, the *Annotator Workflow*, developed at the National Center for Biomedical Ontology (NCBO). The patient notes are from Stanford Hospital's Clinical Data Warehouse (STRIDE). These records archive over 17-years worth of patient data comprising of 1.6 million patients, 15 million encounters, 25 million coded ICD9 diagnoses, and a combination of pathology, radiology, and transcription reports totaling over 9 million clinical notes (i.e., unstructured text).

From this set of 1.6 million patients, we extracted a cohort of patients that suffered from kidney failure. Out of those records, we applied our algorithms to all previous records in the patient's timeline, looking at just the set of drugs. Therefore, at a very simplistic level, the experiment result shows that semantically

	Support	
	Shopping cart	Electronic health
$\mathbf{s_{CT}}$	0.58	0.82
$\mathbf{s_{L+}}$	0.32	0.06

TABLE 5.5. A comparison between s_{CT} and s_{L+} based on the Kendall-τ score between rankings of itemsets generated by s_{CT}, s_{L+} and support in the two experiments.

associated itemsets in this context could possibly represent sets of drugs that could lead toward kidney failure when used in combination.

5.3.3.2. Results

The cohort dataset described above contains 467,791 rows (corresponding to patients' clinical notes) and 10,167 columns (corresponding to annotated terms appeared in the notes). With the help of the techniques described in Section 5.2.1.2, we are able to compute L^+ in a tractable amount of time (Equation 5.2 and Equation 5.3 are calculated within 4 hours on a Quad-Core AMD Opteron(tm) Processor with 8 gigabyte memory), based on which we can efficiently derive the s_{L+} itemsets. However, the calculation of s_{CT} on this scale is intractable because an exact computation of all pair-wise s_{CT} requires filling in a $|V| \times |V|$ similarity table. In order to ameliorate the computational cost, we exploit domain knowledge to identify 582 terms of particular interest and then apply both s_{CT} and s_{L+} on the reduced dataset. The results are shown in Table 5.6 and 5.7 respectively, where we list top-10 2-itemsets and all (k >2)-itemsets corresponding to the maximum clique.

It is clear that, continuing the trend shown in the FoodMart analysis, the s_{CT} result becomes increasingly concordant with the support-based method.

	$\mathbf{s_{CT}}$	**Freq**	**Itemset**
2-itemsets	0.80	39204	⟨ *Calcium Chloride, Amiloride* ⟩
	0.77	29325	⟨ *Calcium Chloride, Aspirin* ⟩
	0.76	28644	⟨ *Calcium Chloride, Probenecid* ⟩
	0.73	24805	⟨ *Calcium Chloride, Furosemide* ⟩
	0.72	34271	⟨ *Calcium Chloride, Calcium* ⟩
	0.71	21481	⟨ *Calcium Chloride, Disulfiram* ⟩
	0.70	16814	⟨ *Calcium Chloride, Amphetamine* ⟩
	0.66	19850	⟨ *Calcium Chloride, Prednisone* ⟩
	0.65	12231	⟨ *Aspirin, Amiloride* ⟩
	0.65	12106	⟨ *Probenecid, Amiloride* ⟩
(k>2)-itemsets	0.56	0	⟨ *Calcium Chloride, Disul-firam, Amphetamine, Aceta-minophen, Calcium, Aspirin, Probenecid, Amiloride, Prednisone, Furosemide* ⟩

TABLE 5.6. Top s_{CT} results on the kidney failure cohort of the electronic health dataset.

	$\mathbf{s_{L+}}$	**Freq**	**Itemset**
2-itemsets	0.820	354	⟨ *sevoflurane, remifentanil* ⟩
	0.691	978	⟨ *frovatriptan, almotriptan* ⟩
	0.633	693	⟨ *Etomidate, Rocuronium* ⟩
	0.496	234	⟨ *Atazanavir, Pyrimethamine* ⟩
	0.420	3004	⟨ *ciclesonide, Fluorometholone* ⟩
	0.377	231	⟨ *naratriptan, Mefenamic Acid* ⟩
	0.373	1792	⟨ *ciclesonide, Vincristine* ⟩
	0.332	92	⟨ *Rocuronium, sevoflurane* ⟩
	0.325	1368	⟨ *tazarotene, halobetasol propionate* ⟩
	0.322	506	⟨ *Buprenorphine, alosetron* ⟩
(k>2)-itemsets	0.131	701	⟨ *Ketorolac, Flurbiprofen, Ketorolac, Etodolac, Sulindac, Piroxicam, Ketoprofen* ⟩

TABLE 5.7. Top s_{L+} results on the kidney failure cohort of the electronic health dataset.

For illustrating this point of view, we calculate the Kendall-τ score between the ranking of itemsets generated by s_{CT}, s_{L+}, and support as shown in Table 5.5. We observe from the table that as s_{CT} converges to support, s_{L+} becomes even more distinct from it. The result is that the itemsets discovered by s_{CT} contain mostly general terms that are repeatedly found in the patients' notes. The association is reasonable but hardly interesting. On the contrary, the s_{L+} result is not affected by the dimension of data or the presence of items with massive support. It identifies itemsets of relatively low support but more closely bonded by indirect links.

To demonstrate the scalability of the method based on the s_{L+}, we also conducted the same analysis on the data of the whole cohort after 2010. The data consisted of 1 million rows and 10 thousand columns. We were able to produce the s_{L+} based 2-itemsets in 6 hours. The top results are shown in Table 5.8.

The discovered s_{L+} itemsets provide much valuable insights on the possible interrelationships between drugs. Some of them have been studied in the literature. For example, *sevoflurane/remifentanil* can be used for anaesthesia; *frovatriptan* and *almotriptan* are both oral treatment of migraine headache; *Etomidate* and *Rocuronium* can be used for rapid sequence intubation; etc. This area of research is still very new and there are no good gold standards to compare our results against. However, for single-item drugs that lead to kidney failure, SIDER[1] database lists drugs and their side-effects. Most notably, multi-itemsets are difficult to identify, but our methods have found not only *Ketoprofen* but it has also group other drugs like it (see the ($k > 2$)-itemset shown in Table 5.7, all of the items are anti-inflammatories). Our results are a matter of on-going evaluation with medical experts.

[1]`http://sideeffects.embl.de/se/C0035078/all`

$\mathbf{s_{L+}}$	**Itemset**
0.0301	⟨ *White faced hornet venom, Yellow hornet venom* ⟩
0.0195	⟨ *Trichloroacetic Acid, Trichloroacetate* ⟩
0.0108	⟨ *Cloxacillin Sodium, benzathine cloxacillin* ⟩
0.0101	⟨ *Methacycline, Methacycline hydrochloride* ⟩
0.01	⟨ *Entamoebiasis, Hepatic, Liver Abscess, Amebic* ⟩
0.0086	⟨ *butenafine, Butenafine hydrochloride* ⟩
0.0085	⟨ *Acetone, Cantharidin* ⟩
0.0085	⟨ *ethyl cellulose, Cantharidin* ⟩
0.0085	⟨ *ethyl cellulose, Acetone* ⟩
0.0085	⟨ *Poloxamer 407, Eucalyptol* ⟩

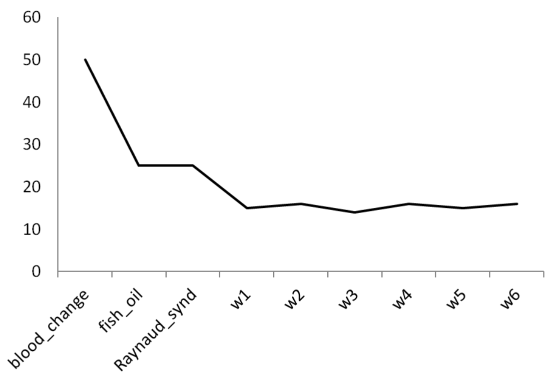
TABLE 5.8. Top s_{L+} results on the whole electronic health dataset after 2010. The dataset contains 1 million rows and 10k columns.

5.4. Discussion

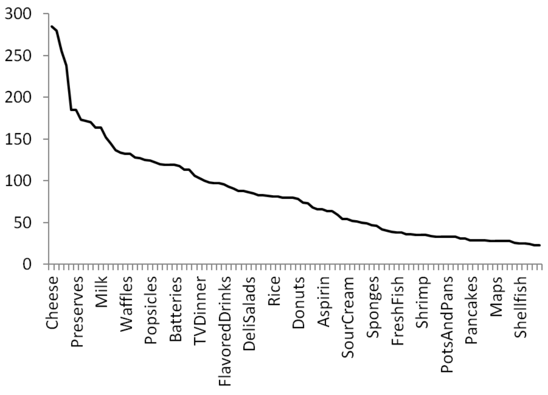
We have observed in the experiments that with the increase of the data size, the commute time based similarity s_{CT} converges to support, while the inner product similarity s_{L+} remains distinct. In this section, we study the cause of this phenomenon.

The core attribute that affects the behavior of s_{CT} and s_{L+} is the node degree distribution of the graph. For graphs that are relatively uniform (the out degree distribution of the graph does not follow a skewed distribution), s_{CT} and s_{L+} appear equally useful. However, for realistic data where the degree distribution follows a Zipf or power-law relationship, the commute time distance displays a bias toward high degree nodes. It is well known that real-world large graphs follow a power law, hence the degradation of s_{CT} in such cases.

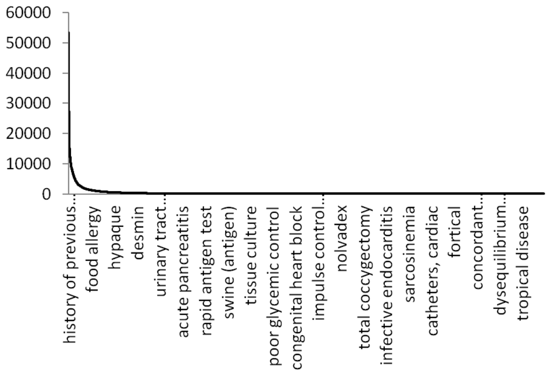
Figure 5.2 demonstrates the degree distributions of the three experiments datasets which gradually evolve into a power-law distribution. The electronic health dataset even exhibits a Zipf-like distribution as illustrated by the near linear pattern on the log-log plot.



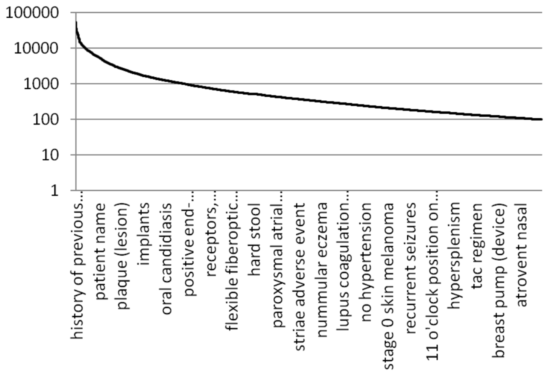
(a) Fish oil



(b) Shopping cart



(c) Electronic health



(d) Electronic health (log-log scale)

FIGURE 5.2. The degree distributions of experiment datasets.

Below we further explore mathematically the reason why s_{CT} and s_{L+} behave in the respective ways. We have shown in Equation 5.6 that the random walk commute time distance (which is inversely proportional to s_{CT}) can be calculated using the following formula:

$$n(i,j) = V_G||\mathbf{x}'_i - \mathbf{x}'_j||^2$$

The transformed node vector x'_i is derived by first projecting the unit node vector to the new space spanned by the eigenvectors of $\mathbf{L}$: $\mathbf{x}_i = \mathbf{U}^T\mathbf{e}_i$, which is then further scaled to $\mathbf{x}'_i = \mathbf{\Lambda}^{1/2}\mathbf{x}_i$, where $\mathbf{U}$ is an orthonormal matrix made of the eigenvectors of $\mathbf{L}^+$ ordered in decreasing order of corresponding eigenvalue λ_k, and $\mathbf{\Lambda} = diag(\lambda_k)$. On the other hand, quite remarkably, as is shown in Equation 5.8, elements of the pseudoinverse of the Laplacian matrix are the inner products of the transformed node vectors (which are defined as s_{L+}):

$$l_{ij}^+ = \mathbf{x}_i'^T\mathbf{x}'_j.$$

This means that we can construct an embedding which maps the vertices v_i of the graph on points $\mathbf{x}'_i \in \mathcal{R}^n$ such that the commute distances on the graph coincide with the Euclidean distances between the points $\mathbf{x}'_i$, and the inner-product similarities between the points of $\mathbf{x}'_i$ correspond to elements of the pseudoinverse of the graph Laplacian.

In large graphs following a power law distribution, there are abundant subgraphs with a star structure, where a high degree node is in the middle connecting to a large number of leaves. It is therefore particularly pertinent to study the spectral properties of star graphs for a comprehensive understanding of s_{CT} and s_{L+}.

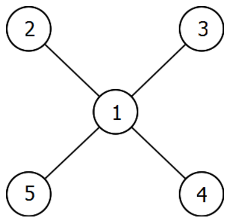
Lemma 5.1. For a star graph S_n of order n, its Laplacian and pseudoinverse of the Laplacian have the following properties:

1. The eigenvalues of the Laplacian $\mathbf{L}(S_n)$ are 0, 1 (with multiplicity $n-2$), and n.

2. The eigenvalues of the Laplacian $\mathbf{L}^{+}(S_n)$ are 0, 1 (with multiplicity $n-2$), and $1/n$.

3. The linearly independent $n-2$ eigenvectors of the eigenvalue 1 are such that the eigencomponent corresponding to the central vertex is 0.

For the proof of this Lemma, readers are referred to [96] for details.



$$\mathbf{U} = \begin{array}{c} \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \end{array} \begin{array}{ccccc} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 & \mathbf{v}_4 & \mathbf{v}_5 \\ \hline -0.89 & 0.45 & 0.00 & 0.00 & 0.00 \\ 0.22 & 0.45 & 0.60 & 0.71 & 0.71 \\ 0.22 & 0.45 & -0.72 & -0.28-0.24i & -0.28+0.24i \\ 0.22 & 0.45 & 0.29 & -0.22-0.22i & -0.22+0.22i \\ 0.22 & 0.45 & -0.17 & -0.21+0.46i & -0.21-0.46i \end{array}$$

$$\mathbf{\Lambda} = diag\,(\,[\;\; 0.2 \;\; 0 \;\; 1 \;\; 1 \;\; 1 \;\;]\,)$$

$$\mathbf{X}' = \mathbf{\Lambda}^{1/2}\mathbf{U}^T = \begin{array}{ccccc} \mathbf{x}'_1 & \mathbf{x}'_2 & \mathbf{x}'_3 & \mathbf{x}'_4 & \mathbf{x}'_5 \\ \hline -0.40 & 0.10 & 0.10 & 0.10 & 0.10 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.60 & -0.72 & 0.29 & -0.17 \\ 0.00 & 0.71 & -0.28+0.25i & -0.22+0.22i & -0.21-0.46i \\ 0.00 & 0.71 & -0.28-0.25i & -0.22-0.22i & -0.21+0.46i \end{array}$$

FIGURE 5.3. An example star graph and its eigenvalues/eigenvectors of the $\mathbf{L}+$, together with node vectors in the transformed space.

Figure 5.3 shows an example of a simple star graph with five nodes. The eigenvectors $\mathbf{U}$ (= $[\mathbf{v}_1, \ldots, \mathbf{v}_5]$) and eigenvalues $\mathbf{\Lambda}$ of $\mathbf{L}^{+}$ are shown in the upper half of the graph. The values of $\mathbf{\Lambda}$ agrees with Lemma 5.1(2), and the eigencomponents of the three eigenvectors $(\mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5)$ of the eigenvalue 1 in the node vector $\mathbf{x}_1$ corresponding to the central vertex are all zeros, satisfying Lemma 5.1(3).

The lower half of Figure 5.3 shows the node vectors ($\mathbf{x}_1, \ldots, \mathbf{x}_5$) in the transformed space. As mentioned above, the commute time distance between nodes in the original graph becomes the Euclidean distance between nodes in the transformed space, and the elements of $\mathbf{L}^+$ is the inner product of the corresponding nodes in the transformed space.

It is obvious that for a large graph with many local star structures, the commute time distance between any two center nodes is small since there are many zeros as the eigencomponents in their corresponding transformed node vectors (such as $\mathbf{x}'_1$ in the example). Together with the fact that the transformed node vectors $\mathbf{x}'_i$ are centered ($\sum_{i=1}^n \mathbf{x}'_i = \mathbf{0}$), we can also conclude that for any leaf node connecting directly to two center nodes, the commute time distance is smaller between the leaf and the one with a larger degree.

The fact that transformed node vectors are centered can be shown from $\sum_{i=1}^n \mathbf{x}'_i = \mathbf{\Lambda}^{1/2} \sum_{i=1}^n \mathbf{x}_i = \mathbf{\Lambda}^{1/2}\mathbf{U}^T \sum_{i=1}^n \mathbf{e}_i = \mathbf{\Lambda}^{1/2}\mathbf{U}^T\mathbf{e}$. And from $\mathbf{\Lambda} = \mathbf{U}^T\mathbf{L}^+\mathbf{U}$, we have $\mathbf{\Lambda}^{1/2}\mathbf{U}^T = \mathbf{\Lambda}^{-1/2}\mathbf{U}^T\mathbf{L}^+$. Therefore $\sum_{i=1}^n \mathbf{x}'_i = (\mathbf{\Lambda}^{1/2}\mathbf{U}^T)\mathbf{e} = (\mathbf{\Lambda}^{-1/2}\mathbf{U}^T\mathbf{L}^+)\mathbf{e} = \mathbf{0}$ since $\mathbf{L}^+\mathbf{e} = \mathbf{0}$.

A more detailed example is shown in Figure 5.4. A connection graph between people and movies is used to encode the information of movie viewerships where movies are illustrated as big solid circles and people as small circles. An arc is drawn between a movie and a person if the movie is watched by the person. From the graph, we observe that there exists three star substructures with A, B and C being center vertices respectively.

Given such a graph, finding similar movies can be naturally solved by a graph-based similarity (such as s_{CT} or s_{L+}). At the first glance, since movie A and B have more common viewers than A and C, we should conclude that A and

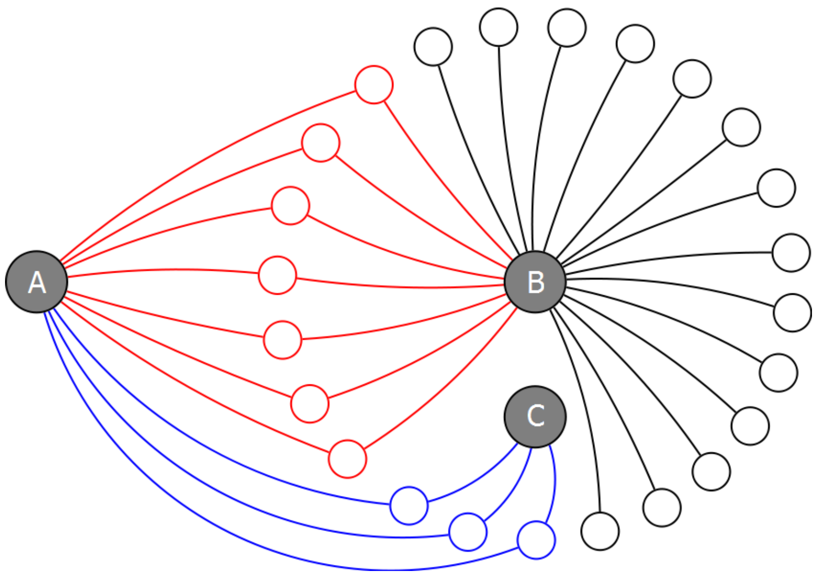


FIGURE 5.4. . An example connection graph between people and movies depicting the movie viewership.

B are more similar to each other than A and B. However, it is soon evident that the group of people who watch C also exclusively watch A. And while movie A and B are commonly viewed by more, it is simply because B is popular and in fact many more people who watch B neither watch A and C. While it might be very legitimate for a system to rank B higher than C in the recommendation to a person who has viewed A, the viewership distribution in this scenario suggests a closer bond between A and C in terms of relevance (imagine A and B are movies from completely different genres and C being a director's cut version of A).

To capture such relevance, s_{L+} would perform better than s_{CT}, as we have pointed out and show-cased in the experiments that the commute time distance is biased towards high-degree nodes. This is especially pronounced in graphs with skewed degree distribution. The connection graph in this example is a bit skewed in that movie B has much more viewers than A and C. Indeed, the calculation below illustrates this point.

$n(A,B) = 0.535$, $n(A,C) = 0.817$, $l^{+}(A,B) = -0.031$, $l^{+}(A,C) = 0.098$. Therefore, $S_{CT}(A,B) > S_{CT}(A,C)$, while $S_{L+}(A,B) < S_{L+}(A,C)$.

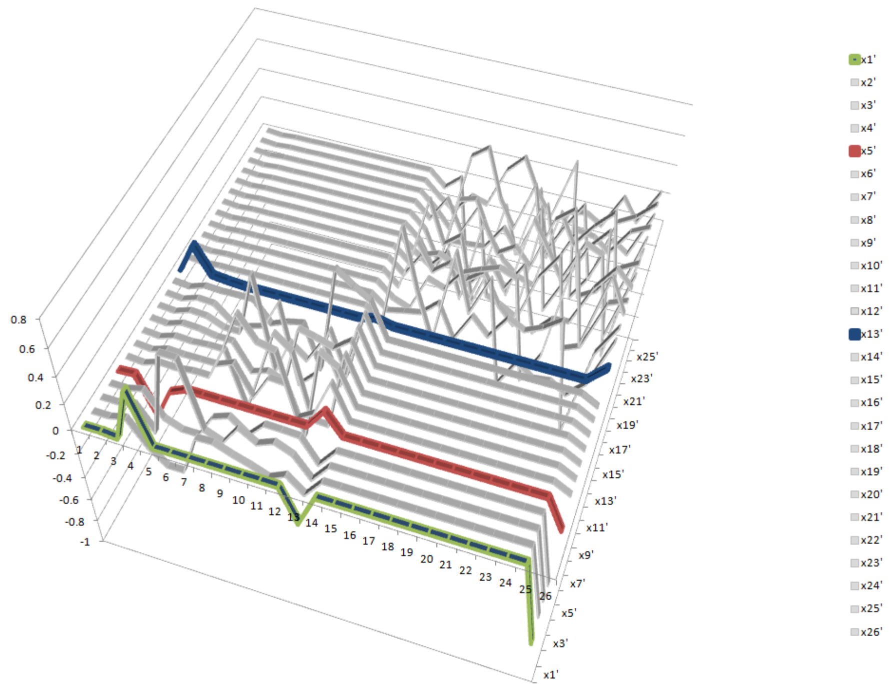


FIGURE 5.5. The 3-D plot of node vectors in the transformed space for the graph depicted in Figure 5.4.

To give an intuition of the calculation of s_{CT} and s_{L+} in this example, Figure 5.5 shows a 3-D plot of the transformed node vectors, where $\mathbf{x}'_1$ corresponds to node B in Figure 5.4, $\mathbf{x}'_5$ to A, $\mathbf{x}'_{13}$ to C, and they are color-coded differently. Additionally, people who watch both A and C (blue nodes in the original graph) correspond to the transformed node vectors $\mathbf{x}'_2 - \mathbf{x}'_4$; people who watch both A and B (red nodes in the original graph) correspond to $\mathbf{x}'_6 - \mathbf{x}'_{12}$; and people who watch B only correspond to $\mathbf{x}'_{14} - \mathbf{x}'_{26}$.

It is obvious that $\mathbf{x}'_1$, $\mathbf{x}'_5$ and $\mathbf{x}'_{13}$ appear much flatter than others because they are center nodes in their respective stars. Moreover, notice that last eigencomponents from all node vectors form an eigenvector $\mathbf{v}'_{26}$ corresponding to the largest eigenvalue of $\mathbf{L}^{+}$, which is the second eigenvalue of $\mathbf{L}$, therefore $\mathbf{v}'_{26}$ is the *Fiedler vector* and can be used to partition the graph. The most straightforward way is to use the sign of the eigencomponents to partition the graph into two clusters: $\{\mathbf{x}'_1 - \mathbf{x}'_{12}\}$ and $\{\mathbf{x}'_{13} - \mathbf{x}'_{26}\}$. The inner product similarity s_{L+} accounts for this partition and is mainly decided by the product at these eigencomponents.

CHAPTER VI

MINING SEMANTICALLY ASSOCIATED ITEMSETS WITH ONTOLOGIES

The RDF bipartite graph is able to represent data and domain knowledge encoded in ontologies in the same way so that analysis approaches on the RDF bipartite graph can benefit from the combined source of information. In the last chapter, I have described in detail the mining method for cases where ontologies are not necessary to be included based on the coarsened RDF hypergraphs. In this chapter, I cover cases where ontologies are present and incorporated so that mining semantically associated itemsets can be more effective with the help of encoded domain knowledge.

This chapter makes the following main contributions: First, I employ the RDF bipartite graph representation to capture both ontologies and data. Each edge can be weighted so that certain links (such as *is_a* or *may_treat* relationships) can carry appropriate strength. Then, I implement highly efficient and scalable random walks with restart over the RDF bipartite graph to generate semantically associated itemsets. Finally, I evaluate the correctness of the results on well-known shopping cart datasets, and the scalability of the method on our large electronic health dataset.

The study described in this chapter received contributions from several individuals. Dr. Dejing Dou and Dr. Ruoming Jin provided valuable insights on the design of the similarity measure based on the RDF bipartite graphs. Dr. Paea LePendu and Dr. Nigam Shah provided the electronic health dataset and helped evaluate the experimental results.

6.1. Method

To enable the incorporation of ontologies in mining semantically associated itemsets, We use the RDF bipartite representation described in Chapter III. We distinguish different semantic relationships in the RDF bipartite graph by assigning weights to those corresponding paths. The various semantic relationships include, for example, class subsumption, part_of, and other general or domain–specific properties.

Formally, the RDF bipartite graph as a combined representation for both data and ontologies is defined as $G = \langle V_v \cup V_s, E \rangle$, where V_v denotes *value nodes* corresponding to components of RDF statements (*i.e.*, subject, predicate, or object), and V_s denotes *statement nodes* corresponding to RDF statements. More specifically, statement nodes can be further divided according to whether they are from data or ontology, *i.e.*, $V_s = V_d \cup V_o$; Value nodes can be divided according to whether they represent rows (records) or columns (attributes) in data, *i.e.*, $V_d = V_r \cup V_a$. The graph G can be represented in a biadjacency matrix $\mathbf{M}$, where $\mathbf{M}(i, j)$ is non-zero if there is an edge between $\langle V_{v_i}, V_{s_j} \rangle$. For an unweighted graph, the value can be 0/1, and for a weighted graph, any non-negative value.

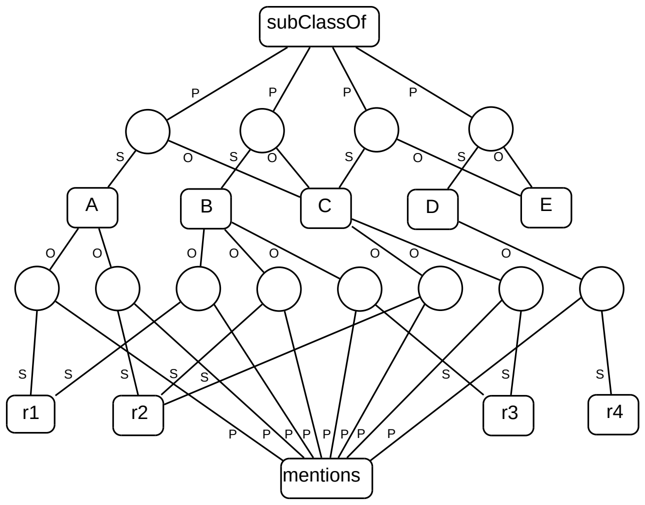
Example 6.1 (**An example RDF bipartite graph and its biadjacency matrix**)**.** In Figure 6.1 we show an example of an RDF bipartite graph. This graph has been used in Chapter III to demonstrate how a data graph and an ontology graph can be combined into a single RDF bipartite graph. In this example, we describe its biadjacency matrix. The upper half of the bipartite graph in Figure 6.1(A) is constructed from information of a domain ontology, which is corresponding to RDF statements s_1–s_4 in Figure 6.1(B). The lower half of the bipartite graph is from a transaction table, which can be represented by statements

s_5–s_{12}. Figure 6.1(C) shows the biadjacency matrices $\mathbf{M}_d$ and $\mathbf{M}_o$ for the data and ontology part of the RDF bipartite graph respectively. We can see that rows of $\mathbf{M}_d$ and $\mathbf{M}_o$ correspond to *value nodes*, (V_v), which can be further divided into row nodes V_r and attribute nodes V_a. On the other hand, columns of $\mathbf{M}_d$ are nodes that correspond to RDF statements about data (V_d), and columns of $\mathbf{M}_o$ correspond to the ontology (V_o). The union of V_d and V_o constitutes the whole set of statement nodes V_s (circle nodes in Figure 6.1(A).

From this example we notice that the biadjacency matrix $\mathbf{M}$ can be split into vertical stripes by statement nodes V_s. To obtain the biadjacency matrix $\mathbf{M}$ of the combined RDF bipartite graph in Figure 6.1(A), we can simply concatenate $\mathbf{M}_d$ and $\mathbf{M}_o$ horizontally: $\mathbf{M} = [\mathbf{M}_d\ \mathbf{M}_o]$. This gives us a way to construct the matrix modularly from its independent components. In general, if there are k different semantic relationships in ontologies, $\mathbf{M}_o$ can be divided into more vertical stripes $\{\mathbf{M}_{o_i}, i = 1 \dots k\}$, where $\mathbf{M}_{o_i}$ may represent, for example, the "part_of" lattice. Each $\mathbf{M}_{o_i}$ can be distinguished from others by different weights assigned to it. In short, $\mathbf{M}$ is the horizontal concatenation of all weighted vertical stripes as shown in Equation 6.1. The internal block structure of the concatenated biadjacency matrix $\mathbf{M}$ is shown in Equation 6.2.

$$\mathbf{M} = \begin{bmatrix} w_d\mathbf{M}_d & w_{o_1}\mathbf{M}_{o_1} & w_{o_2}\mathbf{M}_{o_2} & \dots \end{bmatrix} \qquad \text{(Equation 6.1.)}$$

$$\mathbf{M} = \begin{array}{c} \\ r \\ a \end{array} \begin{array}{c|c|c|c} ds & os_1 & os_2 & \dots \\ \mathbf{M}_{dr} & \mathbf{0} & \mathbf{0} & \dots \\ \hline \mathbf{M}_{da} & \mathbf{O}_1 & \mathbf{O}_2 & \dots \end{array} \qquad \text{(Equation 6.2.)}$$



	S	P	O
s_1:	`<A>`	`<subClassOf>`	`<C>`
s_2:	`<B>`	`<subClassOf>`	`<C>`
s_3:	`<C>`	`<subClassOf>`	`<E>`
s_4:	`<D>`	`<subClassOf>`	`<E>`
s_5:	`<r1>`	`<mentions>`	`<A>`
s_6:	`<r1>`	`<mentions>`	`<B>`
s_7:	`<r2>`	`<mentions>`	`<A>`
s_8:	`<r2>`	`<mentions>`	`<B>`
s_9:	`<r2>`	`<mentions>`	`<C>`
s_{10}:	`<r3>`	`<mentions>`	`<B>`
s_{11}:	`<r3>`	`<mentions>`	`<C>`
s_{12}:	`<r4>`	`<mentions>`	`<D>`

(A) (B)

$$\overbrace{\overbrace{s_5\quad s_6\quad s_7\quad \dots\quad s_{12}}^{V_d}\qquad\qquad\qquad \overbrace{s_1\quad s_2\quad s_3\quad s_4}^{V_o}}^{V_s}$$

$$\mathbf{M}_d = \begin{array}{c|ccccc|} & s_5 & s_6 & s_7 & \dots & s_{12} \\ \hline r_1 & 1 & 1 & 0 & & 0 \\ r_2 & 0 & 0 & 1 & & 0 \\ r_3 & 0 & 0 & 0 & \dots & 0 \\ r_4 & 0 & 0 & 0 & & 1 \\ A & 1 & 0 & 1 & & 0 \\ B & 0 & 1 & 0 & & 0 \\ C & 0 & 0 & 0 & \dots & 0 \\ D & 0 & 0 & 0 & & 1 \\ E & 0 & 0 & 0 & & 0 \end{array} \left.\begin{array}{c} V_r \\ V_a \end{array}\right\} V_v \qquad \mathbf{M}_o = \begin{array}{c|cccc|} & s_1 & s_2 & s_3 & s_4 \\ \hline r_1 & 0 & 0 & 0 & 0 \\ r_2 & 0 & 0 & 0 & 0 \\ r_3 & 0 & 0 & 0 & 0 \\ r_4 & 0 & 0 & 0 & 0 \\ A & 1 & 0 & 0 & 0 \\ B & 0 & 1 & 0 & 0 \\ C & 1 & 1 & 1 & 0 \\ D & 0 & 0 & 0 & 1 \\ E & 0 & 0 & 1 & 1 \end{array} \left.\begin{array}{c} V_r \\ V_a \end{array}\right\} V_v$$

(C)

FIGURE 6.1. An example RDF bipartite graph and a detailed anatomy of its biadjacency matrix.

With the RDF bipartite graph and the form of its biadjacency matrix defined, in the next section, we move on to describe a similarity measure based on random walk with restart over this graph to discover semantically associated itemsets.

6.1.1. Similarity Ranking by Random Walk with Restart

Similar to the relevance score [97], we believe that two items have a strong semantic association if they are related to many similar objects. We denote the similarity score between entities e_1 and e_2 by $s(e_1, e_2)$, where $s(e_1, e_2) \in [0, 1]$ and $s(e_1, e_2) = 1$ if $e_1 = e_2$. Now the problem of ranking semantic associations in the unified graph can be described as follows.

Given an attribute node a in the unified graph $G = G_d \cup G_o$ and $a \in G_d \cap G_o$ we want to compute a similarity score $s(a, b)$ for all nodes $b(\neq a) \in G_d \cap G_o$. The result is a one-column vector containing all similarity scores with respect to a [98]. We choose to apply random walks with restart (RWR) from the given node a, and use the steady-state probability of each other node at convergence as the similarity measure. In other words, the similarity score of node b is defined as the probability of visiting b via a random walk which starts from a and goes back to a with a probability c.

RWR is closely related to the two similarity measures, *i.e.*, s_{CT} and s_{L+}, that are presented in Chapter V on RDF hypergraphs, since they are all derived from the random walk model. It is sometimes a desirable property of a similarity measure for many applications, if it is able to discount nodes with large degrees like the s_{L+} measure. The adaptation of s_{L+} to RDF bipartite graph is a topic worth exploring in future work. However, the scores of RWR on bipartite graphs are easier to compute, especially, when the number of nodes in the two sides is highly unbalanced. The RDF bipartite graph is unbalanced because there are generally many more statement nodes than value nodes on large graphs. Therefore we focus on studying RWR over the RDF bipartite graph in this chapter.

In more detail, RWR in a bipartite graph works as follows: assume we have a random walker that starts from node a. For each step, the walker chooses randomly among the available edges from the current node. After each iteration, with probability c, it resets its position back to node a. The final steady-state probability that the random walker reaches node b is the similarity score of b with respect to a. We choose the random walk approach to compute the relevance score because it gives node b high ranking if b and a are connected by many nodes; this is due to the random walker having more paths to reach b from a. The purpose of the periodic restart of the random walk is to raise the chance that close related nodes are visited more often than other nodes.

In the following, we describe how to calculate algorithmically the similarity ranking based on random walk with restart on the unified RDF bipartite graph. The algorithm can be used in situations where, for example, users are interested in knowing products that are usually bought together in the same transactions by different customers, or common side effects of the same drugs prescribed to different patients, etc.

Given a biadjacency matrix $\mathbf{M}$ in Equation 6.1 for the combined RDF bipartite graph G, we can construct the adjacency matrix $\mathbf{A}$ of G as following:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M}^T & \mathbf{0} \end{bmatrix}.$$

The probability of a random walker taking a particular edge $\langle a, b\rangle$ from a node a while traversing the graph is proportional to the edge weight over the total weight of all outgoing edges from a, i.e., $\mathbf{P}(a, b) = \mathbf{A}(a, b)/\Sigma_{i=1}^{m+n}\mathbf{A}(a, i)$. Therefore, the Markov transition matrix $\mathbf{P}$ of G is constructed as: $\mathbf{P} = normc(\mathbf{A})$, where $normc(\mathbf{A})$ normalizes $\mathbf{A}$ such that every column sum up to 1.

Given the transition matrix $\mathbf{P}$, we can calculate the similarity scores using the following steps. First, we transform the input attribute node a into a $(k + n) \times 1$ query vector $\mathbf{q}_a$ with 1 in the a-th row and 0 otherwise. Second, we need to compute a $(k + n) \times 1$ steady-state probability vector $\mathbf{u}_a$ over all nodes in G. Last we extract only the steady-state probabilities of row nodes in $\mathbf{M}$ (corresponding to value nodes in the RDF bipartite graph) as the output similarity score vector. Notice that $\mathbf{u}_a$ can be computed by an iterated method from the following iterative equation.

Let c be the probability of restarting random-walk from the node a. Then the steady-state probability vector $\mathbf{u}_a$ satisfies

$$\mathbf{u}_a = (1 - c)\mathbf{P}_A\mathbf{u}_a + c\mathbf{q}_a \ . \qquad \text{(Equation 6.3.)}$$

The iterative update of $\mathbf{u}_a$ can be performed as shown in Algorithm 2. The while loop is modified from Equation 6.3 to avoid materializing $\mathbf{A}$ and $\mathbf{P}$ for scalability.

6.2. Case Studies

In this section, we evaluate the method of random walk with restart on the combined RDF bipartite graph for discovering semantic associations. We conducted

Algorithm 2 Calculate Semantic Association

```
Input: query attribute $a$, bipartite matrix $M$, restarting probability $c$, tolerant
    threshold $\epsilon$
Output: similarity vector $\mathbf{u}_a(1:k)$
  $\mathbf{q}_a \Leftarrow \mathbf{0}$
  $\mathbf{q}_a(a) = 1$ (set $a$-th element of $\mathbf{q}_a$ to 1)
  while $|\Delta \mathbf{u}_a| > \epsilon$ do
      $$\mathbf{u}_a = (1-c)\begin{bmatrix} normc(\mathbf{M})\mathbf{u}_a(k+1:k+n); \\ normc(\mathbf{M}^T)\mathbf{u}_a(1:k) \end{bmatrix} + c\mathbf{q}_a$$
  end while
  return $\mathbf{u}_a(1:k)$
```

a series of experiments to highlight the effect of the incorporating the ontologies in the mining task, and to explore the impact of different ratios of weights assigned to various kinds of relationships in the graph. First, to illustrate the power of combined RDF bipartite graph in finding semantic associations while taking into account seamlessly the ontological information, we evaluated our methods on a commonly used *shopping cart* dataset together with a manually created ontology describing the subsumption hierarchy for grocery items. Finally, we applied our method to actual *electronic health records* to highlight its scalability and applicability to the medical domain.

Below we first summarize the sizes of the datasets used in our experiments in Table 6.1.

	# data stmts	# isa stmts	# other stmts[†]
Shopping cart	8,481	127	0
Electronic health	148,690,056	1,048,604	43780

TABLE 6.1. Dataset overview ("stmts" stands for RDF statements). [†] In the electronic healths test, we explore the "may_treat" relationship between drugs and diseases defined in the National Drug File.

6.2.1. Shopping Cart

6.2.1.1. Dataset

The shopping cart dataset is the same as we used in the case study of Chapter V. It contains purchase information on 100 grocery items (represented by boolean column headers) for 2,127 shopping orders (corresponding to tuples) from a Foodmart. We first construct an RDF bipartite graph from the dataset by transforming the table to 8481 RDF statements.

Besides, we manually create an ontology to organize the grocery items into a subsumption hierarchy. In this process, we introduce 28 parent nodes (the 100 grocery items appeared in the data are mostly at the leaf level) from which derive a total of 127 RDF statements. As the size of this dataset is fairly small, the calculation of similarity ranking for a given term is fast. In the following we highlight the effect of incorporation of ontology by comparing results obtained with and without ontologies.

6.2.1.2. Results

In Table 6.2, results of items ranked by the strength of semantic association with regard to a query term "Toothbrush" under various combinations of parameters are demonstrated side-by-side for comparison. We first show the result ranked by co-frequency in Table 6.2(A) as a baseline. Then, we observe that without using ontology, performing random walk with restart on the data graph (Table 6.2(A)) starting from "toothbrush" yields similar results to the work reported in [90] based on random walk commute time similarity. Items ranked high in this setting where only the data graph is considered are typically either hub nodes (with many edges linking to other items) or co-frequent with the query item (many edges connecting them). Second, applying the same similarity ranking method solely on the ontology graph (Table 6.2(C)) gives a list of association based on the graph-configuration of the ontological structure (in this case, the rdfs:subClassOf lattice). The items that are considered most similar to the query term "Toothbrush" is its immediate parent class "PersonalHygiene," followed by some most derived classes at the same level of "PersonalHygiene" and then siblings of "Toothbrush" itself. Next, Table 6.2(D)–(F) demonstrate the results of mining on the combined graph with different ratios of weights assigned to ontology edges and data edges respectively. It is obvious that these results can be seen as a mix of the data-only and ontology-only results with various emphasis on the data or ontology. We can observe that when $w_o/w_d = 20$ the ontology and data appear to have equal significance in determining the ranking (w_o is the weight of ontology edges (*i.e.*, rdfs:subClassOf) and w_d is the weight of data edges). In a rough sense, it conforms to the ratio of the size of ontology graph and data graph as well (see Table 6.1). In reality, the appropriate ratio for the edge weights is not only dependent on the size of graphs but also the specific configuration of the graph (depth, average degree, etc). Moreover, specifying the ratio of prior knowledge in ontologies and inductive evidences in data that one wants to employ for discovering new patterns is a highly empirical process. Multiple pilot trials may need to be carried out for the optimal ratio before it is applied to the real application.

We notice that without any filtering on the ranked semantic associations from the combined graph, the list includes items that never appear in the transactional data. This is because typically the semantic annotation process links table attributes to their most specific matching concept in the ontology which are

ranked by co-frequency		w/ data only		w/ ontology only	
item	freq	item	p(%)	item	p(%)
PaperWipes	8	Soup	0.42	PersonalHygiene	12.55
Popcorn	7	Cookies	0.41	Snack	0.86
Soup	6	NasalSprays	0.38	Health	0.64
NasalSprays	6	Popcorn	0.32	Sponges	0.57
Cookies	6	PaperWipes	0.29	Soap	0.57
Spices	5	FrozenVegetables	0.29	Shampoo	0.57
Soda	4	PersonalHygiene	0.26	NasalSprays	0.57
Shrimp	4	DriedFruit	0.25	Mouthwash	0.57
FlavoredDrinks	4	Milk	0.25	Conditioner	0.57
Dips	4	Mouthwash	0.24	MealCourse	0.54

(A) (B) (C)

$w_o = 1$, $w_d = 1$		$w_o = 10$, $w_d = 1$		$o_w = 20$, $o_d = 1$	
item	p(%)	item	p(%)	item	p(%)
PersonalHygiene	0.74	PersonalHygiene	3.97	PersonalHygiene	6.27
Soup	0.41	NasalSprays	0.41	NasalSprays	0.5
Cookies	0.4	Soup	0.34	Mouthwash	0.41
NasalSprays	0.37	Cookies	0.34	Shampoo	0.31
Popcorn	0.31	Mouthwash	0.3	Soup	0.29
FrozenVegetables	0.29	Popcorn	0.25	Cookies	0.29
PaperWipes	0.28	FrozenVegetables	0.24	Sponges	0.28
DriedFruit	0.25	PaperWipes	0.23	Health	0.27
Milk	0.25	DriedFruit	0.22	Conditioner	0.27
Mouthwash	0.23	Milk	0.21	Soap	0.25

(D) (E) (F)

TABLE 6.2. Foodmart items ranked by the strength of semantic association (i.e., $p(\%)$, the steady-state probability), given the query term “Tooth Brush.”

close to the leaf level. The incorporation of ontology is to aid the mining process, therefore including in the result those parent nodes (*e.g.*, “PersonalHygiene”) that never appear in the data is counterintuitive. To overcome this, we can simply filter out those items exclusive to the ontology. Table 6.3 shows an example of filtered result given a query term “soup.” The co-frequency of items are also listed for comparison.

w/ data only					
item	p(%)	freq	item	p(%)	freq
Cheese	0.38	98	Preserves	0.19	65
Cookies	0.32	96	Juice	0.17	47
DriedFruit	0.32	87	Lightbulbs	0.17	47
Wine	0.24	63	PaperWipes	0.16	55
CannedVegetables	0.23	67	Pizza	0.16	46
FrozenVegetables	0.23	79	Nuts	0.16	60
Cereal	0.22	56	Popcorn	0.16	39
Milk	0.22	53	Chips	0.16	46
ChocolateCandy	0.19	16	Eggs	0.16	51
Waffles	0.19	51	TVDinner	0.15	40

w/ onto only					
item	p(%)	freq	item	freq	p(%)
TVDinner	0.46	40	Sponges	21	0.06
Pizza	0.46	46	Soap	0	0.06
Pasta	0.46	29	Shampoo	34	0.06
HotDogs	0.46	30	NasalSprays	21	0.06
Hamburger	0.46	19	Mouthwash	28	0.06
FrenchFries	0.46	37	Conditioner	12	0.06
DeliSalads	0.46	31	Ibuprofen	18	0.06
DeliMeats	0.46	37	ColdRemedies	33	0.06
Sunglasses	0.07	12	Aspirin	22	0.06
Toothbrushes	0.06	13	Acetominifen	12	0.06

TABLE 6.3. Semantically associated items for the query term "Soup," by filtering out those items exclusive to the Foodmart ontology.

6.2.2. Electronic Health Records

6.2.2.1. Dataset

In our second evaluation, we analyzed the electronic health records of real patients. The patient clinical note data are from Stanford Hospital's Clinical Data Warehouse (STRIDE). These records archive over 17-years worth of patient data comprising of 1.6 million patients, 15 million encounters, 25 million coded ICD9 diagnoses, and a combination of pathology, radiology, and transcription reports totaling over 9 million clinical notes (i.e., unstructured text). We obtained the set of drugs and diseases for each patient's clinical note by using a new tool, the *Annotator Workflow*, developed at the National Center for Biomedical Ontology (NCBO), which annotates clinical text from electronic health record systems and extracts disease and drug mentions from the electronic health records.

From this set of 1.6 million patients with annotated records, we vectorize texts and turned them into a huge bag-of-word representation, from which an RDF bipartite graph is constructed (including 148 million RDF statements, see Table 6.1). we applied our algorithms to all previous records in the patient's timeline, looking at just the set of drugs and their semantically related diseases. Therefore, at a very simplistic level, the experiment result shows that strong semantically associated items in this context could possibly represent sets of drugs that could lead toward certain diseases.

One strength of the Annotator is the highly comprehensive and interlinked lexicon that it uses. It can incorporate the entire NCBO BioPortal ontology library of over 250 ontologies to identify biomedical concepts from text using a dictionary of terms generated from those ontologies. Terms from these ontologies are linked together via mappings. For this study, we specifically configured the workflow to use a subset of those ontologies that are most relevant to clinical domains, including Unified Medical Language System (UMLS) terminologies such as SNOMED-CT, the National Drug File (NDFRT) and RxNORM, as well as ontologies like the Human Disease Ontology. The resulting set of ontologies contains 1 million subsumption statements.

To highlight the capability of our method for incorporating multiple types of relationships, we also explore the "may_treat" relationship between drugs and diseases defined in NDFRT, for example, Thiabendazole may_treat Larva Migrans. Since we are interested in learning the interaction between drugs and diseases, may_treat is naturally a better indicator relationship to include while mining semantic associations than the subsumption relationship. Our results below illustrate this point.

6.2.2.2. Results

Before studying the drug-disease association, we carried out a similar test to that on the shopping cart dataset, in which we focus on studying the drug-drug and disease-disease association. To this purpose, we combine the subsumption hierarchy in the ontology graph with the data graph. Table 6.4 shows the ranked semantic association for the query term "Rofecoxib" (an active ingredient of some anti-inflammatory drugs) given different weight configuration to combine graphs. Without any preprocessing and prior knowledge about how the clinical notes are prescribed, the incorporation of subsumption relationship can be seen as a mean for denoising and enhancement of the data. Given the ratio of the size of the ontology to the size of data, the data graph in this test is more dominant in determining the ranking than in the shopping cart experiment. One can gradually change the ratio of w_o to w_d to strike a balance and achieve the optimal result.

rank	w/ data only	w/ onto only	$w_o = 10000, w_d = 1$
1	reflux	valdecoxib	reflux
2	medical history	meloxicam	obstruction
3	history of previous events	celecoxib	injury
4	diagnosis	parecoxib	valdecoxib
5	pharmaceutical preparations	etoricoxib	medical history
6	blood and lymphatic system disorders	deracoxib	foreign body sensation
7	disease	lumiracoxib	history of previous events
8	infantile neuroaxonal dystrophy	firocoxib	adverse effects
9	today	nabumetone	celecoxib
10	hypersensitivity	macrolides	actual hypothermia

TABLE 6.4. Results of Health items ranked by the strength of semantic association, given the query term "Rofecoxib."

To verify the drug–disease association and study the impact of different semantic relationships on finding such association, we carry out the following experiment. Table 6.5 illustrates the rankings of three associations (one per row) under different settings. The first element in the pair is the query item, which are all active ingredients of some prescription drugs, and the ranking shown in the table is for the second item, which are diseases. For example, arthritis is ranked as the 527th semantically associated item to Rofecoxib according to similarity ranking based only on data graph. All these item pairs are actually gold standard associations backed by known drug–disease relationships, we know the strength of semantic associations between them should be strong.

We observe that the ranking based on data graph alone is fairly high already, consider there are approximately 1 million concepts of interest. However, the results based on the combination of data and subsumption ("isa") graph are worse. It is because the subsumption hierarchies for drugs and diseases are largely separate structures. Therefore the subsumption relationships can only boost the association within the drug and disease hierarchies respectively, but obfuscate the cross-hierarchy associations that we aim to find between drugs and diseases. On the other hand, however, the association between these pairs can be exactly captured by the NDFRT "may_treat" relationship (*e.g.*, NDFRT explicitly defines that Rofecoxib may_treat arthritis). When the "may_treat" graph is incorporated into the mining process, the ranking for the association is greatly boosted.

	w/ data only		w/ data and "isa"		w/ data and "may_treat"	
	p(%)	rank	p(%)	rank	p(%)	rank
$\langle Rofecoxib, degenerative\ polyarthritis\rangle$	0.006	527	0.004	632	0.51	13
$\langle valdecoxib, degenerative\ polyarthritis\rangle$	0.007	613	0.005	695	0.63	17
$\langle troglitazone, diabetes\rangle$	0.006	478	0.005	514	0.44	11

TABLE 6.5. Rankings of three semantic associations in health data under different settings.

Conversely, we are also interested in learning whether the data graph can help discover patterns in the ontology graph. Figure 6.2 (left) shows a subgraph of the NDFRT "may_treat" relationship. Rofecoxib is asserted to treat two diseases, namely, dysmenorrhea and degenerative polyarthritis. And there are altogether

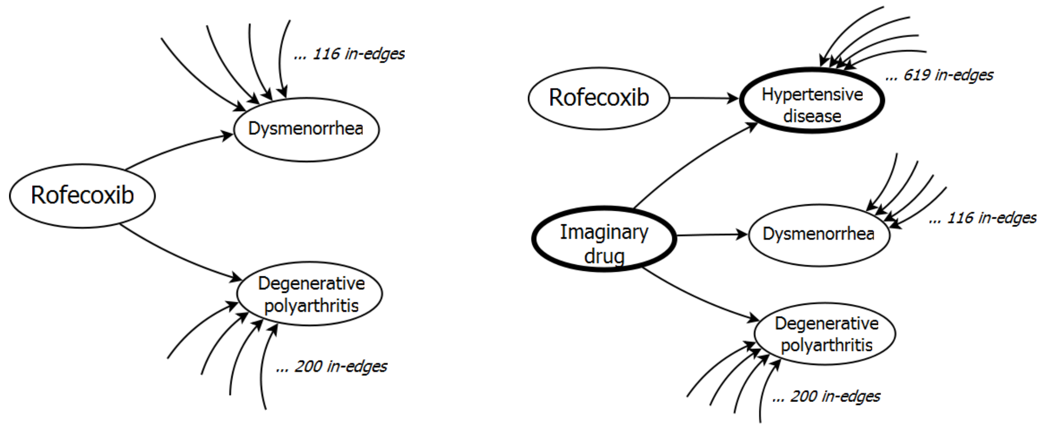


FIGURE 6.2. The may_treat subgraph before and after distortion: The left-hand side of the figure shows the may_treat subgraph of ground truth relationships between the drug Rofecoxib and two diseases. The right-hand side shows the may_treat subgraph with some deliberately distorted information.

	w/ noisy may_treat only		w/ data and noisy may_treat	
	p(%)	rank	p(%)	rank
$\langle Rofecoxib, degenerative\ polyarthritis\rangle$	3.60e-3	555	8.14e-3	263
$\langle Rofecoxib, dysmenorrhea\rangle$	1.54e-2	246	1.26e-3	1703

TABLE 6.6. Rankings of associations on the noisy may_treat graph (Figure 6.2 right) between Rofecoxib and two diseases derived with and without data.

116 and 200 drugs that are known to treat dysmenorrhea and degenerative polyarthritis respectively (hence the in-degrees of the nodes). Applying our method on this graph with the query term "Rofecoxib" yields a similarity-ranked list having degenerative polyarthritis and dysmenorrhea as the top two items. Since this result is the exact ground truth, there is no improvement to be made with the incorporation of the data graph. Therefore, we alter the ground truth graph with some deliberately distorted information, as is shown in Figure 6.2 (right), so that the may_treat graph alone produces only inferior result. More specifically, we specify that Rofecoxib should treat hypertensive disease, the very diseases that is asserted to be treated by the most drugs (a total of 619). Then we add an imaginary drug to treat degenerative polyarthritis, dysmenorrhea, and

hypertensive disease. In this way, the original direct connections between Rofecoxb and degenerative polyarthritis and dysmenorrhea become erroneously indirect and are obfuscated by some the noise of high degree nodes along the path. With this scenario, we hope to learn if the incorporation of data graph can correct the misinformation in ontologies.

Table 6.6 shows the result of ranks of the associations between Rofecoxib and degenerative polyarthritis and dysmenorrhea. The ranks of the associations drastically drop to the 555th and 246th respectively on the noisy graph from the top two on the original ground truth graph. This is mainly due to the large node, hypertensive disease, in the middle of the connections. However, with the combined data and may_treat graph, we notice that the rank of Rofecoxib and degenerative polyarthritis increases to 263rd, while the rank of Rofecoxib and dysmenorrhea decreases to 1703rd. This shows that the data graph endorses more strongly the association between Rofecoxib and degenerative polyarthritis. Indeed, although Rofecoxib are known to treat both degenerative polyarthritis and dysmenorrhea, the former is a much more popular usage. A search on the National Library of Medicine's PubMed database[1] for "Rofecoxib and polyarthritis" returns 518 results, while "Rofecoxib and dysmenorrhea" only returns 29. This result shows that the data graph can help correct misinformation in ontologies to some extent, and in a sense, it also gives a clue of how prior beliefs fit with reality.

[1] `http://www.ncbi.nlm.nih.gov/`

CHAPTER VII

CONCLUSION

This dissertation proposes the framework of semantic data mining, a novel direction for the field of data mining with a focus on the incorporation of domain knowledge encoded in ontologies. The enabling technologies are based on three contributions presented in the dissertation.

first, we propose a graph-based formalism that allows a coherent representation for both data and ontologies. The key concept of the approach is to use RDF as a common ground for both data and domain knowledge encoded in ontologies and then to employ the RDF hypergraph or bipartite graph as the unified representation.

Second, when mining tasks require accessing disparate, heterogeneous data sources, we develop a method based on metaheuristic optimization to automatically resolve schema heterogeneities as well as to achieve pattern comparison for meta-analysis.

Finally, we demonstrate analysis techniques that can be carried out based on the RDF hypergraph or bipartite graph to tackle common data mining tasks. We showcase the utility of such techniques on a mining problem called semantically associated itemset discovery, a particular kind of frequent itemset mining focusing on finding indirect connections. For this purpose, several graph-based similarity measures are provided as key components of mining algorithms. Their capacities, limitations and trade-offs are studied. The concept of random walk on hypergraphs while traversing and calculating similarities among nodes are used in designing these measures.

This dissertation also presents the details of some case studies that have validated the hypotheses used in designing the graph-based semantic data mining framework.

7.1. Future Work

Semantic data mining is an emerging field, and many interesting research directions related to it are yet to be explored. The research work presented in this dissertation can be extended in several directions. The following are some of the most important ones we have identified.

7.1.1. Automatic and Robust Semantic Annotation

Semantic annotation is crucial in realizing semantic data mining by bridging formal semantics in Semantic Web meta-data with data. It aims at assigning semantic descriptions to elements of data. The annotation process can be generally divided into two steps. The first is to establish mappings between existing Semantic Web terms and terms need to be annotated in the data. The second step is to come up with a local ontological structure constituting the Semantic Web terms to model the data. Most of previous work focus on the second step. Some skip the first step and bootstrap the ontological terms and structure from the local data itself. For example, a number of systems that map data in RDB to RDF leverage a set of rules such as "table to class and column to predicate." Other examples of rules involved in mapping RDB schema to OWL ontologies include "foreign keys to object property and non-key attributes to datatype property." Similar ideas have been adopted in annotating semi-structured data. Existing spreadsheet-to-rdf tools typically map spreadsheets to star-shaped RDF graphs. Some tools try to express richer spreadsheet semantics, e.g., Han et al. developed a spreadsheet-to-rdf tool called RDF123 [99] that allows users to define mappings to arbitrary graphs.

We argue that mapping RDB or spreadsheet to linked data (e.g., RDF) without reference to existing semantic descriptions does not lend itself well to aiding semantic data mining. The automatically constructed self-contained local ontology may be applicable to describe a specific dataset but is most likely too rough to capture the full domain semantics that is necessary to express meaningful domain knowledge. Moreover, with the advent of the Semantic Web and pervasive connectivity, an increasing number of ontologies have been made widely available for reuse. These ontologies are created by thorough knowledge engineering process and should serve as better models for annotation. However, on the other hand, the sheer number of Semantic Web ontologies and lack of effective search functionality can lead to a huge hidden barrier for common users. Choosing proper Semantic Web ontologies and terms (classes and properties) requires familiarity with appropriate ontologies and the terms they define. There is very few system that is able to provide automatic suggestions.

To solve this problem, we proposed a preliminary idea of learning-based semantic search algorithm reported in [100] to suggest proper Semantic Web terms and ontologies for annotation given semantically related words and general domain and context information. The evaluation of the algorithm is a matter of ongoing research.

7.1.2. Learning Weights Automatically

The RDF bipartite graph representation relies on the assignment of weights to different types of hyperedges, or paths in bipartite graphs, to distinguish the underlying semantics they represent. When ontologies are involved in the mining process, there are at least two types of connections involved, corresponding to RDF statements coming from data and ontologies respectively. Thus at least two different weights have to be assigned with respect to this distinction. In the current work, the (ratio of) weights can be decided purely empirically, or through a series of trial and error experiments on a sampled sub-dataset, which is hardly guaranteed to be accurate or generalizable. Such difficulty is even more noticeable when there are multiple semantic relationships present in the data and ontologies that one hopes to distinguish so as to achieve finer-grained control over their respective contributions to the mining result. Therefore how to automatically derive suitable weights is an important research question.

One technique that can be used is to train a prediction model from labeled data. This approach suffers from the difficulty of acquiring the gold-standard training sample. Tian *et al.* [101] proposed a semi-supervised approach for classifying nodes in a graph based on a relatively small labeled set. The main idea is to formalize the weight assignment and label propagation in one constraint optimization problem while the two objectives can be alternately solved using a two-step iterative method. While this approach is promising, how to extend it to other mining tasks such as frequent pattern mining is worth of further investigation.

7.1.3. Handling Continuous Features

The RDF bipartite graph is straightforward to represent binary-valued data and is also able to represent nominal-valued data through RDB nominal value expansion. However, there is no immediate solution to make numerical (continuous) data representable. If we enumerate all values present in a numerical feature and create one node in the graph for each of such values, the size of the resulting graph is bound to become intractable. A common way to handle continuous feature is discretization, or binning, as in making histograms. Typical discretization methods include equal interval/frequency partitioning, or more sophisticated ones such as Fayyad and Irani's supervised method called MDL [102] that uses information gain to recursively define the best bins.

The more interesting part comes when discretization has to be guided by domain knowledge. For example, certain patterns may make better sense when a column of dates is present and discretized into seasons or quarters rather than arbitrary time intervals. How to represent and execute such domain knowledge in a way that is adaptable to the graph-based semantic data mining framework is a matter of ongoing research. For example the process of domain knowledge guided discretization may hint at a need for a set of rule-based data transformation routines under a well-defined protocol that can be treated as a preprocessing step before converting data to graphs. More ways to handle domain knowledge in a standardized way are discussed later in this section.

7.1.4. Scalability Issues

The presented graph-based semantic mining framework heavily relies on the notion of graph-based similarity. We describe the use of several random walk-based measures. The matrix operations required to derive the similarity measures and is very expensive on large graphs. Non-trivial practical problems are often associated with large scales. For instance, there are more than 30,000 classes in the well-known Gene Ontology, and big online social networks have hundreds of millions of users. Therefore scalability of the graph-based semantic mining methods are of critical importance.

The general solution is to employ approximation and develop parallelizable algorithms. Lin and Cohen [103] proposed an approximation to an eigenvalue-weighted linear combination of all the eigenvectors, which can be achieved by performing a small number of matrix-vector multiplications. Such procedure results in a simple and scalable method, called power iteration clustering, that finds a very low-dimensional data embedding using truncated power iteration on a normalized pair-wise similarity matrix of the data points. Zhao *et al.* [104] described the idea of embedding graph nodes into points on a coordinate system. By allowing lower distance distortion errors, they were able to develop a practical system that provides fast embedding of large graphs in a hyperbolic space. The embedding algorithm can be parallelized to allow the cost of the embedding process to be spread across multiple servers. Furthermore, they presented a method to use graph coordinates to efficiently locate shortest paths between node pairs. Such a concept can be naturally extended to embed graph nodes according to their commute time distance. Savas and Dhillon [105] introduced a novel framework called clustered low rank matrix approximation for massive graphs. The first step is to partition the vertices into a set of disjoint clusters with some fast procedure to preserve important structural information of the original graph. Then a low rank approximation of each cluster is computed independently. Finally the different cluster-wise approximations are combined using an optimal projection step to obtain a low rank approximation of the entire graph, thus including connections or edges between vertices from different clusters. While all these techniques are promising, we will need to extend them to the RDF bipartite graph, and the stratification (between data and ontologies and among different semantic types) of the graph may require further adaptions and modifications to those algorithms.

7.1.5. New Ways of Representing Complex Domain Semantics

The RDF bipartite graph can represent concrete semantics such as the "*is_a*" or "*located_in*" relationship. However, meta semantics such as domain/range and cardinality constraints are not so straightforward to be modeled.

One possible approach that can be used to enhance the ability of handling more complex domain semantics in certain applications is to model domain constraints by explicitly describing the desired or acceptable walk (traversal sequence) in the RDF hypergraph or bipartite graph. In this case, the recently proposed regular traversal expression [106] is worth investigation. In the basic case, we can specify only certain types of nodes in a given random walk. The regular traversal expression can allow us to even specify acceptable path segments or sequences. However, the fast power-iteration approach for computing the stationary probability may not be applicable any more due to the label sequence constraint. To address this problem, we can apply the Monte-Carlo simulation of the random walk to approximate the similarity measure. Notice that this approach can be rather scalable as the simulation can be in general constrained in those nodes linking two targeted nodes.

7.2. Concluding Remarks

Although it is widely acknowledged that the use of domain knowledge is important in all stages of the data mining process, research on systematic fusion of the knowledge and data mining still remains in its early stages. The conventional way of using domain knowledge often causes a tight coupling of assumptions and algorithms, and hence may hinder the maintainability and interoperability of mining systems. We propose a new angle to attack this problem by introducing a graph-based formalism. Domain knowledge is encoded in ontologies which are in turn represented as RDF hypergraphs, the same representation we can use to model data. In this way, domain knowledge is treated as first-class objects in the mining algorithm, and the central task becomes first finding good quality ontologies that captures domain semantics and then determining a relevant subset of the ontologies that should be part of the mining process. The relevant strength of the relationships in ontologies can be defined by assigning proper weights to them. Several graph-based similarity measures are provided as key components of mining algorithms on the unified graph representation. Their capacities, limitations, trade-offs, and possible directions for improvement are studied on a series of synthetic and real-world case studies.

We believe that the designed principles for semantic data mining with graph-based approaches provide a novel and useful way to incorporate domain knowledge. As such they should be considered the most important contributions of this dissertation.

REFERENCES CITED

[1] W. J. Frawley, Piatetsky G. Shapiro, and C. J. Matheus. Knowledge discovery in databases - an overview. *AI Magazine*, 13:57–70, 1992.

[2] Usama Fayyad, Gregory Piatetsky-shapiro, and Padhraic Smyth. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17:37–54, 1996.

[3] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003. ISBN 0137903952.

[4] Hayam Hirsh and Michiel Noordewier. Using Background Knowledge to Improve Inductive Learning: A Case Study in Molecular Biology. *IEEE Expert: Intelligent Systems and Their Applications*, 9:3–6, October 1994. ISSN 0885-9000.

[5] Gary Weiss and Foster Provost. The Effect of Class Distribution on Classifier Learning: An Empirical Study. Technical report, Department of Computer Science, Rutgers University, 2001.

[6] Hennie Daniels, Ad Feelders, and Marina Velikova. Integrating Economic Knowledge in Data Mining Algorithms. Discussion Paper 2001-63, Tilburg University, Center for Economic Research, 2001.

[7] Ioannis Kopanas, Nikolaos M. Avouris, and Sophia Daskalaki. The Role of Domain Knowledge in a Large Scale Data Mining Project. In *Proceedings of the Second Hellenic Conference on AI: Methods and Applications of Artificial Intelligence*, SETN '02, pages 288–299, London, UK, 2002. Springer-Verlag. ISBN 3-540-43472-0.

[8] Helge Langseth and Thomas D. Nielsen. Fusion of domain knowledge with data for structural learning in object oriented domains. *J. Mach. Learn. Res.*, 4: 339–368, December 2003. ISSN 1532-4435.

[9] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web: Scientific American. *Scientific American*, May 2001.

[10] Deborah L. McGuinness and Frank van Harmelen. OWL Web Ontology Language Overview. W3C Recommendation, World Wide Web Consortium, February 2004.

[11] Thomas R. Gruber. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. In *In Formal Ontology In Conceptual Analysis And Knowledge Representation, Kluwer Academic Publishers, In Press. Substantial Revision Of Paper Presented At The International Workshop On Formal Ontology*. Kluwer Academic Publishers, 1993.

[12] Alexander Maedche, Boris Motik, Ljiljana Stojanovic, Rudi Studer, and Raphael Volz. Ontologies for Enterprise Knowledge Management. *IEEE Intelligent Systems*, 18:26–33, March 2003. ISSN 1541-1672.

[13] Natalya F. Noy and Deborah L. Mcguinness. Ontology Development 101: A Guide to Creating Your First Ontology. Technical report, Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.

[14] A. J. Duineveld, R. Stoter, M. R. Weiden, B. Kenepa, and V. R. Benjamins. Wondertools: A Comparative Study Of Ontological Engineering Tools. *Int. J. Hum.-Comput. Stud.*, 52(6):1111–1133, June 2000. ISSN 1071-5819.

[15] Holger Knublauch, Ray W. Fergerson, Natalya F. Noy, and Mark A. Musen. The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. In *The Semantic Web – ISWC 2004*, volume 3298 of *Lecture Notes in Computer Science*, chapter 17, pages 229–243. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-540-23798-3.

[16] The Gene Ontology Consortium. Gene Ontology: Tool For The Unification Of Biology. In *Nat. Genet.*, volume 25, pages 25–29, May 2000.

[17] Olivier Bodenreider. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32(suppl 1): D267–D270, January 2004. ISSN 1362-4962.

[18] Gwen A. Frishkoff, Robert M. Frank, Jiawei Rong, Dejing Dou, Joseph Dien, and Laura K. Halderman. A Framework to Support Automated Classification and Labeling of Brain Electromagnetic Patterns. *Computational Intelligence and Neuroscience (CIN), Special Issue, EEG/MEG Analysis and Signal Processing*, 7(3):1–13, 2007.

[19] Gwen Frishkoff, Paea LePendu, Robert Frank, Haishan Liu, and Dejing Dou. Development of Neural Electromagnetic Ontologies (NEMO): Ontology-based Tools for Representation and Integration of Event-related Brain Potentials. In *Proceedings of the International Conference on Biomedical Ontology (ICBO)*, pages 31–34, 2009.

[20] Jonathan Hayes and Claudio Gutierrez. Bipartite Graphs as Intermediate Model for RDF. Master's thesis, Universidad de Chile, 2004.

[21] Lise Getoor and Christopher P. Diehl. Link Mining: A Survey. *SIGKDD Explor. Newsl.*, 7:3–12, December 2005. ISSN 1931-0145.

[22] Atanas Kiryakov, Borislav Popov, Ivan Terziev, Dimitar Manov, and Damyan Ognyanoff. Semantic annotation, indexing, and retrieval. *J. Web Sem.*, 2(1): 49–79, 2004.

[23] Satya S. Sahoo, Wolfgang Halb, Sebastian Hellmann, Kingsley Idehen, Ted Thibodeau Jr, S02ren Auer, Juan Sequeda, and Ahmed Ezzat. A survey of current approaches for mapping of relational databases to rdf, 01 2009.

[24] Don R Swanson. Two medical literatures that are logically but not bibliographically connected. *Journal of the American Society for Information Science*, 38(4):228–233, January 1999. ISSN 1097-4571.

[25] Michael Pazzani and Dennis Kibler. The Utility of Knowledge in Inductive Learning. *Mach. Learn.*, 9:57–94, June 1992. ISSN 0885-6125.

[26] R. Ambrosino and B.G. Buchanan. The use of physician domain knowledge to improve the learning of rule-based models for decision-support. In *Proceedings of the Annual Fall Symposium of the American Medical Informatics Association*, pages 192–196, 1999.

[27] Atish P. Sinha and Huimin Zhao. Incorporating Domain Knowledge Into Data Mining Classifiers: An Application In Indirect Lending. *Decis. Support Syst.*, 46:287–299, December 2008. ISSN 0167-9236.

[28] Carsten Pohle. Integrating and Updating Domain Knowledge with Data Mining. In *VLDB PhD Workshop*, 2003.

[29] Suk-Chung Yoon, Lawrence J. Henschen, E. K. Park, and Sam Makki. Using domain knowledge in knowledge discovery. In *Proceedings of the eighth international conference on Information and knowledge management*, CIKM '99, pages 243–250, New York, NY, USA, 1999. ACM. ISBN 1-58113-146-1.

[30] Vikram Singh and Sapna Nagpal. Integrating User's Domain Knowledge with Association Rule Mining. *CoRR*, abs/1004.3568, 2010.

[31] Steffen Staab and Andreas Hotho. Ontology-based text document clustering. In *Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM'03 Conference held in Zakopane*, pages 451–452, 2003. ISBN 3-540-00843-8.

[32] Jian Wen, Zhoujun Li, and Xiaohua Hu. Ontology Based Clustering for Improving Genomic IR. In *Computer-Based Medical Systems, 2007. CBMS '07. Twentieth IEEE International Symposium on*, pages 225–230, 2007.

[33] Jun Fang, Lei Guo, XiaoDong Wang, and Ning Yang. Ontology-based automatic classification and ranking for web documents. In *Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery - Volume 03*, pages 627–631, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2874-0.

[34] George A. Miller. Wordnet: a lexical database for english. *Commun. ACM*, 38 (11):39–41, November 1995. ISSN 0001-0782.

[35] Ching Kang Cheng, Xiaoshan Pan, and Franz J. Kurfess. Ontology-Based Semantic Classification of Unstructured Documents. In Andreas Nürnberger and Marcin Detyniecki, editors, *Adaptive Multimedia Retrieval*, volume 3094 of *Lecture Notes in Computer Science*, pages 120–131. Springer, 2003. ISBN 3-540-22163-8.

[36] Kazem Taghva, Julie Borsack, Jeffrey Coombs, Allen Condit, Steve Lumos, and Tom Nartker. Ontology-based Classification of Email. In *Proceedings of the International Conference on Information Technology: Computers and Communications*, ITCC '03, pages 194–, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1916-4.

[37] L. Tenenboim, B. Shapira, and P. Shoval. Ontology-based Classification of News in an Electronic Newspaper. In *Proceedings of INFOS 2008*, pages 89–97, Varna, Bulgaria, 2008.

[38] PawełLula and Grażyna Paliwoda-Pekosz. An ontology-based cluster analysis framework. In *Proceedings of the first international workshop on Ontology-supported business intelligence*, OBI '08, pages 7:1–7:6, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-219-1.

[39] Juan Li and Son Vuong. Ontology-Based Clustering and Routing in Peer-to-Peer Networks. In *Proceedings of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies*, PDCAT '05, pages 791–795, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2405-2.

[40] Boris Adryan and Reinhard Schuh. Gene-Ontology-based clustering of gene expression data. *Bioinformatics*, 20:2851–2852, November 2004. ISSN 1367-4803.

[41] Bin Shen, Min Yao, Zhaohui Wu, Yangu Zhang, and Wensheng Yi. Ontology-based Association Rules Retrieval using Protege Tools. In *Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, pages 765–769, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2702-7.

[42] Michel Chein and Marie-Laure Mugnier. *Graph-based Knowledge Representation: Computational Foundations of Conceptual Graphs*. Springer, London, 2008. ISBN 978-1-84800-285-2.

[43] John Sowa, editor. *Principles of Semantic Networks: Explorations in the Representation of Knowledge (Morgan Kaufmann Series in Representation and Reasoning)*. Morgan Kaufmann Pub, May 1991.

[44] William A. Woods. What's in a Link: Foundations for Semantic Networks. In Daniel G. Bobrow and Allan M. Collins, editors, *Representation and Understanding: Studies in Cognitive Science*, pages 35–82. Academic Press, 1975.

[45] Ronald J. Brachman, Deborah L. McGuinness, Peter F. Patel-Schneider, Lori A. Resnick, Lori Alperin Resnick, and Alexander Borgida. Living with CLASSIC: When and How to Use a KL-ONE-Like Language. In *Principles of Semantic Networks*, pages 401–456. Morgan Kaufmann, 1991.

[46] Ian Horrocks. Daml+oil: a description logic for the semantic web. *IEEE Data Engineering Bulletin*, 25:4–9, 2002.

[47] RDF semantics, W3C recommendation, 2004. URL `http://www.w3.org/TR/rdf-mt/`.

[48] Renzo Angles, Claudio Gutierrez, and Jonathan Hayes. Rdf query languages need support for graph properties. Technical report, 2004.

[49] C. Berge. Hypergraphs. *Bull. Symbolic Logic*, 1989.

[50] Ian Hodkinson and Martin Otto. Finite conformal hypergraph covers and Gaifman cliques in finite structures. *Bull. Symbolic Logic*, 9:387–405, 2002.

[51] Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 19–26, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1.

[52] Adrian Corduneanu and Tommi Jaakkola. On information regularization. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, UAI'03, pages 151–158, San Francisco, CA, USA, 2003. Morgan Kaufmann Publishers Inc. ISBN 0-127-05664-5.

[53] Olivier Bousquet, Olivier Chapelle, and Matthias Hein. Measure based regularization. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schkopf, editors, *NIPS*. MIT Press, 2003. ISBN 0-262-20152-6.

[54] L. Lovasz. Random Walks on Graphs: A Survey. In *Combinatorics*, pages 353–397, Budapest, 1993. Janos Bolyai Math. Soc.

[55] D.J. Klein and M. Randic. Resistance Distance. *J. Math. Chemistry*, 12:81–95, 1993.

[56] F. Gobel and A. Jagers. Random Walks on Graphs. *Stochastic Processes and Their Applications*, 2:311–336, 1974.

[57] S. Barnett, editor. *Matrices: Methods and Applications.* Oxford Univ. Press, 1992.

[58] Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. Random-Walk Computation Of Similarities Between Nodes Of A Graph, With Application To Collaborative Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19:2007, 2006.

[59] Jia-Yu Pan, Hyungjeong Yang, Christos Faloutsos, and Pinar Duygulu. Cross-modal Correlation Mining Using Graph Algorithms. *Knowledge Discovery and Data Mining: Challenges and Realities with Real World Data.*, 2006.

[60] L. Yen, L. Vanvyve, D. Wouters, F. Fouss, F. Verleysen, and M. Saerens. Clustering Using A Random-Walk Based Distance Measure. In *Proceedings of ESANN'2005*, 2005.

[61] Yang Zhou, Hong Cheng, and Jeffrey Xu Yu. Graph Clustering Based On Structural/Attribute Similarities. *Proc. VLDB Endow.*, 2:718–729, August 2009. ISSN 2150-8097.

[62] B Suman and P Kumar. A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of The Operational Research Society*, 57:1143–1160, 2006.

[63] Eckart Zitzler and Lothar Thiele. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. pages 292–301. Springer, 1998.

[64] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi. Readings in computer vision: issues, problems, principles, and paradigms. chapter Optimization by simulated annealing, pages 606–615. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987. ISBN 0-934613-33-8.

[65] John H. Holland. *Adaptation in natural and artificial systems.* MIT Press, Cambridge, MA, USA, 1992. ISBN 0-262-58111-6.

[66] Kalyanmoy Deb. *Multi-Objective Optimization Using Evolutionary Algorithms.* John Wiley & Sons, Inc., New York, NY, USA, 2001. ISBN 047187339X.

[67] Erhard Rahm and Philip A. Bernstein. A Survey of Approaches to Automatic Schema Matching. *VLDB JOURNAL*, 10:2001, 2001.

[68] J. A. Larson, S. B. Navathe, and R. Elmasri. A Theory of Attributed Equivalence in Databases with Application to Schema Integration. *IEEE Trans. Softw. Eng.*, 15:449–463, April 1989. ISSN 0098-5589.

[69] Amit P. Sheth, James A. Larson, Aloysius Cornelio, and Shamkant B. Navathe. A Tool for Integrating Conceptual Schemas and User Views. In *Proceedings of the Fourth International Conference on Data Engineering*, pages 176–183, Washington, DC, USA, 1988. IEEE Computer Society. ISBN 0-8186-0827-7.

[70] Wen-Syan Li and Chris Clifton. Semint: A Tool for Identifying Attribute Correspondences in Heterogeneous Databases Using Neural Networks, 2000.

[71] Anhai Doan, Pedro Domingos, and Alon Y. Levy. Learning Source Description for Data Integration. In *WebDB (Informal Proceedings)*, pages 81–86, 2000.

[72] Robin Dhamankar, Yoonkyong Lee, Anhai Doan, Alon Halevy, and Pedro Domingos. iMAP: Discovering Complex Semantic Matches between Database Schemas. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, ACM.* Press, 2004.

[73] Haishan Liu, Gwen Frishkoff, Robert Frank, and Dejing Dou. Ontology-based Mining of Brainwaves: A Sequence Similarity Technique for Mapping Alternative Descriptions of Patterns in Event Related Potentials (ERP) Data. In *Proceedings of the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 43–54, 2010.

[74] Eric Bae, James Bailey, and Guozhu Dong. A Clustering Comparison Measure Using Density Profiles and Its Application to The Discovery of Alternate Clusterings. *Data Min. Knowl. Discov.*, 21:427–471, November 2010. ISSN 1384-5810.

[75] Yvonne Bradford, Tom Conlin, Nathan Dunn, David Fashena, Ken Frazer, Douglas G. Howe, Jonathan Knight, Prita Mani, Ryan Martin, Sierra A. T. Moxon, Holly Paddock, Christian Pich 0002, Sridhar Ramachandran, Barbara J. Ruef, Leyla Ruzicka, Holle Bauer Schaper, Kevin Schaper, Xiang Shao, Amy Singer, Judy Sprague, Brock Sprunger, Ceri E. Van Slyke, and Monte Westerfield. Zfin: enhancements and updates to the zebrafish model organism database. *Nucleic Acids Research*, 39(Database-Issue):822–829, 2011.

[76] Tim Berners Lee. Relational databases on the semantic web, 09 1998. Design Issues (published on the Web).

[77] Clement Jonquet, Paea LePendu, Sean Falconer, Adrien Coulet, Natalya F. Noy, Mark A. Musen, and Nigam H. Shah. Ncbo resource index: Ontology-based search and mining of biomedical resources. *Web Semant.*, 9 (3):316–324, September 2011. ISSN 1570-8268.

[78] Lawrence Reeve and Hyoil Han. Survey of semantic annotation platforms. In *Proceedings of the 2005 ACM symposium on Applied computing*, SAC '05, pages 1634–1638, New York, NY, USA, 2005. ACM. ISBN 1-58113-964-0.

[79] Haishan Liu, Dejing Dou, and Hao Wang. Breaking the Deadlock: Simultaneously Discovering Attribute Matching and Cluster Matching with Multi-Objective Metaheuristics. *Journal on Data Semantics*, 1(2):133–145, 2012.

[80] Haishan Liu, Gwen Frishkoff, Robert Frank, and Dejing Dou. Sharing and Integration of Cognitive Neuroscience Data: Metric and Pattern Matching across Heterogeneous ERP Datasets. *Neurocomputing*, 92(September): 156–169, 2012.

[81] H. W. Kuhn. The Hungarian Method for The Assignment Problem. *Naval Research Logistic Quarterly*, 2:83–97, 1955.

[82] B Suman. Simulated annealing based multiobjective algorithm and their application for system reliability. *Engin Optim*, pages 391–416, 2003.

[83] Isabelle Guyon, Asa Ben Hur, Steve Gunn, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge. In *Advances in Neural Information Processing Systems 17*, pages 545–552. MIT Press, 2004.

[84] Joseph Dien. The ERP PCA Toolkit: An Open Source Program for Advanced Statistical Analysis of Event-Related Potential Data. *Journal of Neuroscience Methods*, 187(1):138 – 145, 2010. ISSN 0165-0270.

[85] Carlos M. Fonseca and Peter J. Fleming. Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 416–423, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc. ISBN 1-55860-299-2.

[86] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553, 1998.

[87] William M. Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. ISSN 01621459.

[88] L. Hamers, Y. Hemeryck, G. Herweyers, M. Janssen, H. Keters, R. Rousseau, and A. Vanhoutte. Similarity Measures In Scientometric Research: The Jaccard Index Versus Salton's Cosine Formula. *Inf. Process. Manage.*, 25: 315–318, May 1989. ISSN 0306-4573.

[89] Ana L.N. Fred and Anil K. Jain. Robust Data Clustering. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on.*, 2:128, 2003. ISSN 1063-6919.

[90] Haishan Liu, Paea Le Pendu, Ruoming Jin, and Dejing Dou. A hypergraph-based method for discovering semantically associated itemsets. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*, ICDM '11, pages 398–406, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-0-7695-4408-3.

[91] Amit P. Sheth, Boanerges Aleman-Meza, Ismailcem Budak Arpinar, Clemens Bertram, Yashodhan S. Warke, Cartic Ramakrishanan, Chris Halaschek, Kemafor Anyanwu, David Avant, F. S. Arpinar, and Krys Kochut. Semantic association identification and knowledge discovery for national security applications. *J. Database Manag.*, 16(1):33–53, 2005.

[92] Dengyong Zhou, Jiayuan Huang, and Bernhard Scholkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems (NIPS) 19*, page 2006. MIT Press, 2006.

[93] R. Baeza-Yates and B. Ribeiro-Neto, editors. *Modern Information Retrieval.* Addison-Wesley, 1999.

[94] I. Herstein and D. Winter, editors. *Matrix Theory and Linear Algebra.* Maxwell Macmillan International Editions, 1988.

[95] Rakesh Agrawal and Ramakrishnan Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc. ISBN 1-55860-153-8.

[96] Knkar Ch. Das. Some Properties Of Laplacian Eigenvalues For Generalized Star Graphs. *Kragujevac J. Math.*, 27:145–162, 2005.

[97] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, ICDM '05, pages 418–425, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2278-5.

[98] Jiyang Chen, Osmar R. Zaiane, Randy Goebel, and Philip S. Yu. Tuplerank: Ranking relational databases using random walks on extended k-partite graphs.

[99] Lushan Han, Tim Finin, Cynthia Parr, Joel Sachs, and Anupam Joshi. Rdf123: From spreadsheets to rdf. In *ISWC '08: Proceedings of the 7th International Conference on The Semantic Web*, pages 451–466, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-88563-4.

[100] Haishan Liu. Towards Semantic Data Mining. In *Proceedings of the 9th International Semantic Web Conference (ISWC2010)*, November 2010.

[101] Ze Tian, TaeHyun Hwang, and Rui Kuang. A hypergraph-based learning algorithm for classifying gene expression and arraycgh data with prior knowledge. *Bioinformatics*, 25(21):2831–2838, November 2009. ISSN 1367-4803.

[102] Usama M. Fayyad and Keki B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *IJCAI*, pages 1022–1029, 1993.

[103] Frank Lin and William W. Cohen. Power iteration clustering. In Johannes Fürnkranz and Thorsten Joachims, editors, *ICML*, pages 655–662. Omnipress, 2010. ISBN 978-1-60558-907-7.

[104] Xiaohan Zhao, Alessandra Sala, Haitao Zheng, and Ben Y. Zhao. Efficient shortest paths on massive social graphs. In Dimitrios Georgakopoulos and James B. D. Joshi, editors, *CollaborateCom*, pages 77–86. IEEE, 2011. ISBN 978-1-4673-0683-6.

[105] Berkant Savas and Inderjit S. Dhillon. Clustered low rank approximation of graphs in information science applications. In *SDM*, pages 164–175. SIAM / Omnipress, 2011. ISBN 978-0-898719-92-5.

[106] Marko A. Rodriguez and Peter Neubauer. The graph traversal pattern. *CoRR*, abs/1004.1001, 2010.