T.R.
TURKISH NAVAL ACADEMY
NAVAL SCIENCE AND ENGINEERING INSTITUTE
DEPARTMENT OF COMPUTER ENGINEERING

GRAPH BASED CONCEPT DISCOVERY FOR MULTI RELATIONAL DATA MINING

A Master Thesis

Mahmut İĞDE

Advisor: Asst.Prof.Dr.Yusuf KAVURUCU

İstanbul, 2015

T.R.
TURKISH NAVAL ACADEMY
NAVAL SCIENCE AND ENGINEERING INSTITUTE
DEPARTMENT OF COMPUTER ENGINEERING

GRAPH BASED CONCEPT DISCOVERY FOR MULTI RELATIONAL DATA MINING

A Master Thesis

Mahmut İĞDE

Advisor: Asst.Prof.Dr.Yusuf KAVURUCU

İstanbul, 2015

GRAPH BASED CONCEPT DISCOVERY

FOR MULTI RELATIONAL DATA MINING

MAHMUT İĞDE

Submitted in partial fulfillment of the requirement for degree of

MASTER OF SCIENCE IN COMMAND, CONTROL, COMMUNICATIONS,

COMPUTER AND INTELLIGENCE (C4I)

Turkish Naval Academy

Naval Science and Engineering Institute

Author:______________________

Mahmut İĞDE

Defense Date: 11.06.2015

Approved by:

____

Asst.Prof.Dr. Yusuf KAVURUCU (Thesis Adviser)

____

Asst.Prof.Dr. Okan TOPÇU (Defense Committee Member)

____

Asst.Prof.Dr.Alev MUTLU (Defense Committee Member)

____

Asst.Prof.Dr. Gürkan KOLDAŞ (Defense Committee Member)

____

Asst.Prof.Dr. Tolga ÖNEL (Defense Committee Member)

DEDICATION

Dedicated to my family...

ACKNOWLEDGEMENT

I would like to render my special thanks to

My advisor, Asst.Prof.Dr.Yusuf KAVURUCU, for giving me ultimate guidance and support,

My instructors, Asst.Prof.Dr.Okan TOPÇU and Asst.Prof.Dr.Tolga ENSARİ for valuable contributions to my scientific development,

Institute director, Dr.Mustafa KARADENİZ, for his active collaboration,

Especially, Asst.Prof.Dr.Alev MUTLU for his contribution to my conference paper and guidance for implementation of proposed system.

DISCLAIMER STATEMENT

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Turkish Naval Forces, Turkish Naval Academy and Naval Science and Engineering Institute. I declare that all materials in this thesis are used according to academical rules and ethics. All accomodated out-sources are included in the references section and these out-sources are referred inside the thesis where they are included.

PLAGIARISM DECLARATION

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

ÖZET

ÇOK İLİŞKİLİ VERİ MADENCİLİĞİ İÇİN ÇİZGE TABANLI KONSEPT TANIMLAMA

Mahmut İĞDE

Komuta, Kontrol, Muhabere, Bilgisayar ve İstihbarat
Yüksek Lisans Tezi, 2015

Danışman: Yrd.Doç.Dr.Yusuf KAVURUCU

Anahtar Kelimeler: Çok İlişkili Veri Madenciliği, Çizge, Tümevaran Mantıksal Programlama(TMP), Konsept Tanımlama

Teknolojinin ve özellikle de bilgisayarın hayatın her alanına girmesi ile birlikte birçok alanda verilerin depolanması ihtiyacı ortaya çıkmıştır. Bu ihtiyaç verilerin kullanışlı şekilde depolandığı veritabanı kavramının doğmasına yol açmıştır. Veritabanında veriler aralarındaki ilişkiye göre mantıksal olarak dosya/dosyalar halinde bir araya getirilmiştir. Hastaneler, labaratuvarlar, bankalar, sanal alışveriş siteleri veritabanlarından faydalanılan yerlere birkaç örnek olarak gösterilebilir.

Önemli hususlardan birisi de verileri kullanışlı ve eksiksiz bir şekilde tutan veritabanlarından bilginin çıkarılmasıdır. Bu süreç veri madenciliği olarak adlandırılır. Veri madenciliğinin ana hedefi veritaban(lar)ında depolanan muazzam ve ilk bakıldığında anlamsız veri topluluğundan üstü kapalı ve yararlı bilginin çıkarılmasıdır.

Çok ilişkili veritabanları verilerin birden fazla tabloda (ilişkide) gösterildiği veritabanlarıdır. Bu tablolar arasındaki ilişkiler veri tabanında ayrıca tablo (ilişki) olarak tutulur. Daha etkili ve yaygın bilinen Çok İlişkili Veri Madenciliği (ÇİVM) yaklaşımı Tümevaran Mantıksal Programlama (TMP) üzerine kurulmuştur. TMP tümevaran öğrenme ve mantıksal programlama konseptlerini içermektedir. Bu açıdan bakıldığında, ÇİVM'nin asıl amacı üstü kapalı ve az önemli görünen bilginin TMP yaklaşım ve teknikleri kullanılarak ilşkisel veritabanlarından ortaya çıkarılmasıdır.

Çizge tabanlı yaklaşım veritabanlarında bilgi keşfi için diğer bir metottur. Bu yaklaşımda veriler çizge yapısında tutulur ve çizge madenciliği teknikleri ile bilgi keşfi yapılır.

Çok ilişkili veri madencilinde Konsept tanımlamanın hedefi ilişki hedef ilişki olarak adlandırılan, bir ilişkiyi ifade eden ilişkisel kuralların geri plan bilgisi olarak adlandırılan veritabanındaki diğer ilişkiler açısından tanımlanmasıdır. Bu çalışmada konsept tanımlama için çizge tabanlı konsept tanımlama metodu anlatılmıştır. Ortaya konulan metod, Ç-KTS(Çizge tabanlı Konsept tanımlama Sistemi), hem altyapı tabanlı hem de yol bulma tabanlı yaklaşımları kullanmakta olduğundan hibrid metod olarak kabul edilebilir. Ç-KTS başlangıçta ilişkisel bir veritabanında tutulan her bir hedef ilişki ve onunla bağlantılı arka plan gerçekleri için bağımsız çizge yapıları oluşturur ve bir özet çizge oluştururken onlardan faydalanır. Özet çizge yapısı konsept tanımlayıcıları bulma amaçlı taranır.

Farklı öğrenme problemlerine ait veri setleri üzerinde birkaç set deney yapılmıştır. Deney sonuçları Ç-KTS nin farklı öğrenme problemlerine ait hedef ilişkilerin tanımlamalarını öğrenebilmekte yeterli olduğunu göstermiştir.

ABSTRACT

GRAPH BASED CONCEPT DISCOVERY FOR MULTI RELATIONAL DATA MINING

Mahmut İĞDE

Command, Control, Communications, Computer and Intelligence
Master of Science Thesis, 2015

Adviser:Asst.Prof.Dr.Yusuf KAVURUCU

Key Words: Multi-Relational Data Mining, Graph, Inductive Logic Programming (ILP), Concept Discovery

Developments in technology, especially in computer science created the need of storing data in variety of areas. This need created the term database where the data is stored in a useful form. In the database data is logically integrated in a file/files according to relations among them. Hospitals, laboratories, banks, online shopping sites are just a few examples where databases are necessarily used.

One of the important issue is to extract knowledge from these databases that hold data in a useful and complete form. This process is called as data mining. The main objective of data mining is to extract implicit and useful knowledge from huge and at first glance meaningless mass of data that is stored in database(s).

Multi-Relational databases are the ones in which the data is stored in multiple tables (relations). The relationships between those tables are also stored as tables (relations) in the database. The more effective and commonly known approach for Multi-Relational Data Mining (MRDM) techniques are based on Inductive Logic Programming (ILP). ILP contains concepts from Inductive Learning and Logic Programming. From this point, the main purpose of MRDM is extracting implicit and trivial knowledge from relational database(s) using ILP approaches and techniques.

Graph based approach is another method for knowledge discovery in databases. In this approach, data is stored in graph structures and graph mining techniques used for knowledge discovery.

Concept discovery in multi-relational data mining aims to find relational rules that best describe a relation, called target relation, in terms of other relations in the database, called background knowledge. In this study, a graph-based concept discovery method for concept discovery is presented. The proposed method, namely G-CDS (Graph-based Concept Discovery System), utilizes methods both from substructure-based and path-finding based approaches, hence it can be considered as a hybrid method. G-CDS generates disconnected graph structures for each target relation and its related background knowledge, which are initially stored in a relational database, and utilizes them to guide generation of a summary graph. The summary graph is traversed to find concept descriptors.

A set of experiments is conducted on datasets that belong to different learning problems. The experimental results show that G-CDS is capable of learning definitions of target relations that belong to different learning problems.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS

Abbreviations :	**Explanation :**
MRDM	Multi-Relational Data Mining
ILP	Inductive Logic Programming
G-CDS	Graph-Based Concept Discovery System
C^2D	Confidence-based Concept Discovery
CRIS	Concept Rule Induction System
FOIL	First Order Inductive Learner
RPBL	Relational Paths Based Learning
RBL	Relational Path Finding
GBI	Graph Based Induction
MIS	Model Inference System
KDD	Knowledge Discovery in Databases
DHDB	Deductive Hierarchical Database
FOPC	First-Order Predicate Calculus
MDL	Medium Description Length
DL	Description Length
SQL	Structured Query Language
HOC	Head Output Connected
RLGG	Relative Least General Generalization

CHAPTER 1 INTRODUCTION

Due to increase of complex data usage in information systems, the amount of data collected in relational databases is also increasing. This increase forced the development of multi-relational learning algorithms that can be applied to directly multi-relational data stored in databases [1]. Generally, first-order predicate logic is employed as the representation language for such learning systems. The learning systems, which find logical patterns valid for given background knowledge, have been investigated under a research area which is called Inductive Logic Programming (ILP) [2].

Concept is defined as a set of frequent patterns that are embedded in the features of the concept instances in the form of relations among objects [3]. Concept discovery is the problem of learning definitions of a specific relation, called target relation, in terms of other relations provided as background knowledge [4].

Concept discovery in relational databases is a predictive learning task. There is a specific target concept to be learned in the light of the past experiences. In ILP-based concept learning methods, logical patterns for the target concept are induced that are validated against the background facts. Association rule mining is a technique that is employed in the proposed algorithm for relational concept discovery. Association rule mining is finding frequent patterns, associations or correlations among sets of items or objects in databases. Relational association rules are expressed as query extensions in first-order logic [5].

The concept discovery problem has extensively been studied by the ILP community with successful applications in several domains such as bioinformatics, engineering, and environmental sciences.

Graph-based approach is another concept discovery method which is based on graph structure. Graph-based concept discovery methods can be classified into two main categories: Substructure-based approaches and path finding-based approaches [7]. In a graph if a substructure is seen frequently then there should be a concept which constructs that substructure. On the other hand, path finding-based approaches [8] assume that a concept should appear as frequent and finite length paths that connect some arguments of positive target instances.

In this work a hybrid framework is proposed, namely G-CDS (Graph-based Concept Discovery System), for graph-based concept discovery. Directed and labeled graph is employed where nodes represent target and background relations, and edges connect those nodes that have at least one common argument. The proposed approach inputs the data in relational format, generates a distinct graph for each target instance, generates the summary graph from those distinct graphs, extracts concept definitions from the summary graph, and outputs concept descriptors in the form of relations. Similar to substructure-based approaches, it groups similar relations and represents them as a single node. Similar to path finding-based approaches, it infers the concept descriptors by finding paths that connect relations.

The experimental results of G-CDS revealed promising performance on the quality of concept discovery in comparison with similar works [9, 10].

1.1. MOTIVATION

Challenging with huge amount of data has been a serious problem in the last decade. Although this huge data can be stored in relational databases with the help of developing technology, people may not find certain and useful information when they need. This is the main problem of data mining and in this thesis our objective is to find an efficient and effective solution.

There are several knowledge discovery systems which are used for MRDM. Most of them are based on ILP and use conventional association rule mining techniques (such as Apriori) during concept discovery process. FOIL [4], PROGOL [21], WARMR [40], GOLEM [33], C^2D [11] and CRIS [9] are some of the well-known ILP-based systems in the literature.

FOIL, PROGOL and WARMR need mode declarations and negative data to prune the search space efficiently and work correctly. Negative data and mode declaration requirements have some difficulties at the beginning and user may not solve these problems - especially mode declaration problem - if not interested much in that area.

On the other hand, C^2D and CRIS do not require mode declarations and negative data. Instead of them, type specification of variables are the only requirement that already exist within the database.

Graph based concept discovery is another competitor in MRDM. The data is given/represented in a graph data structure and various techniques of graph mining is used for concept discovery. SubdueCL [41], Graph Based Induction (GBI) [42] and Relational Paths Based Learning (RPBL) [8] are some examples of graph based approach.

In this thesis, main motivation is to overcome the drawbacks of C^2D. C^2D sometimes prunes important rules in the concept discovery process because of confidence-based pruning mechanism. In addition to this, the process order of target instances may change the result hypothesis in some datasets, and this affects the quality of the overall process. Finally, user has to choose the appropriate option to induce transitive rules in C^2D. The proposed concept discovery system, namely G-CDS (Graph-based Concept Discovery System) generates graph structures with respect to data that is initially stored in a relational database and utilizes them to guide the concept induction process. G-CDS has solutions to above-mentioned drawbacks. A set of experiments is conducted on data sets that belong to different learning problems. The results show that G-CDS has promising results in comparison to state of the art methods and C^2D.

1.2. PROBLEM DEFINITION AND RESEARCH QUESTIONS

The research steps in this study are (i) to learn how to extract useful and unknown information by using concept discovery systems, (ii) how to develop a concept discovery system based on Inductive Logic Programming, (iii) with usage of graph based approaches how can we develop a concept discovery system that is more efficient and effective in terms of experimental results.

This study deals with the understanding of the main principles of the MRDM, ILP and graph based concept discovery. After studying prior concept discovery systems, a new concept discovery system namely G-CDS is introduced. Here are some questions that lead us to study on this subject;

- What are the main problems of those early presented concept discovery systems?
- Can we develop a concept discovery system that can process relational databases?
- Are there any graph based approaches for concept discovery?

- What are the advantages of graph structure usage in concept discovery systems?
- How to implement the proposed graph based concept discovery system?

1.3. THESIS OBJECTIVES

There are a few concept discovery systems developed before beginning our study. The objective of this thesis is not to implement something novel but to present a system that is more efficient and accurate than the previous systems.

Since graph structure is appropriate both visually and functionally than ILP, we started to study on a graph based concept discovery. In this study, some drawbacks of Confidence based Concept Discovery System (C^2D) [11] are solved. The new system is faster than C^2D and presents better hypothesis.

1.4. THESIS OUTLINE

Chapter 1 gives preliminary information about multi-relational data mining, ILP-based learning, Graph based learning and concept discovery in introduction section. Chapter 2 explains the basis of MRDM, ILP, Graph Based learning, Association Rule Mining (ARM). It describes the basic techniques in ILP and in graph based approaches which are used in the proposed method. In addition to this, the popular metrics used in ARM, which are support and confidence is explained. ILP based systems LINUS, GOLEM, CIGOL, MIS, FOIL, PROGOL, ALEPH, WARMR, SAHILP and Graph based systems, RBL, RBPL, SUBDUE, SUBDUECL, GBI are also described in this chapter. At the end of the chapter, these systems are compared with the proposed method. Chapter 3 presents a comparative overview of C^2D and G-CDS on a running example. Also the algorithm of these two methods are introduced. Chapter 4 discusses the experimental results of the proposed method on real world problems such as Learning Recursive Rules in Same-Generation Problem, Finding Transitive Rules Using Unrelated Relations and Constructing Transitive Rules Under Missing Background Information. Chapter 5 includes concluding remarks and possible improvements as a future work.

CHAPTER 2 LITERATURE SURVEY

In this chapter, firstly, multi relational data mining and its basic components are explained. Then the methods and the steps of knowledge discovery process are described. Inductive Logic Programming and Graph Based Approach that are subjected to Knowledge Discovery are presented consecutively. Afterwards, differences and similarities between these approaches are discussed. Finally, examples systems based on these approaches are mentioned in comparison with each other.

2.1. MULTI-RELATIONAL DATA MINING

Technological development in computer science allows us to store much more data than ever in different forms. But the more important issue here is to be able to extract useful information like relational analysis, reports, shopping habits, advertorial goals etc. These are the subjects of data mining. With data mining tools we can make some searches in databases. After some interpretation, information evolves to knowledge and the evolution makes data mining go further to a new field called Knowledge Discovery in Databases (KDD). Knowledge discovery is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data [12]. Given;

- F as set of facts,
- L as language
- C as certainty
- F_S as a subset of F
- Pattern S is defined as a statement in L which describes relationships among F_S with a certainty c, such that S is simpler to represent facts than scanning all the facts in F_S. A pattern that is interesting and certain enough is called knowledge. The system investigating set of facts in a database and extract patterns simply outputs discovered knowledge. Although KDD and Data Mining are sometimes used indistinctly, data mining is a step in knowledge discovery. Banks, shopping companies, airline companies, automobile

manufacturers, medical laboratories use knowledge discovery systems to achieve their goals in their domains.

Steps of KDD process is shown in Figure 2.1 [13] and explained below.

1) In the selection step the data set is selected in which the knowledge discovery process will performed.

2) In the preprocessing step data cleaning is done. Errors and outlier values in data is called noise. Also missing values are another problem that prevent data mining algorithm work smoothly. In this step both missing values and noisy data are filled or fixed.

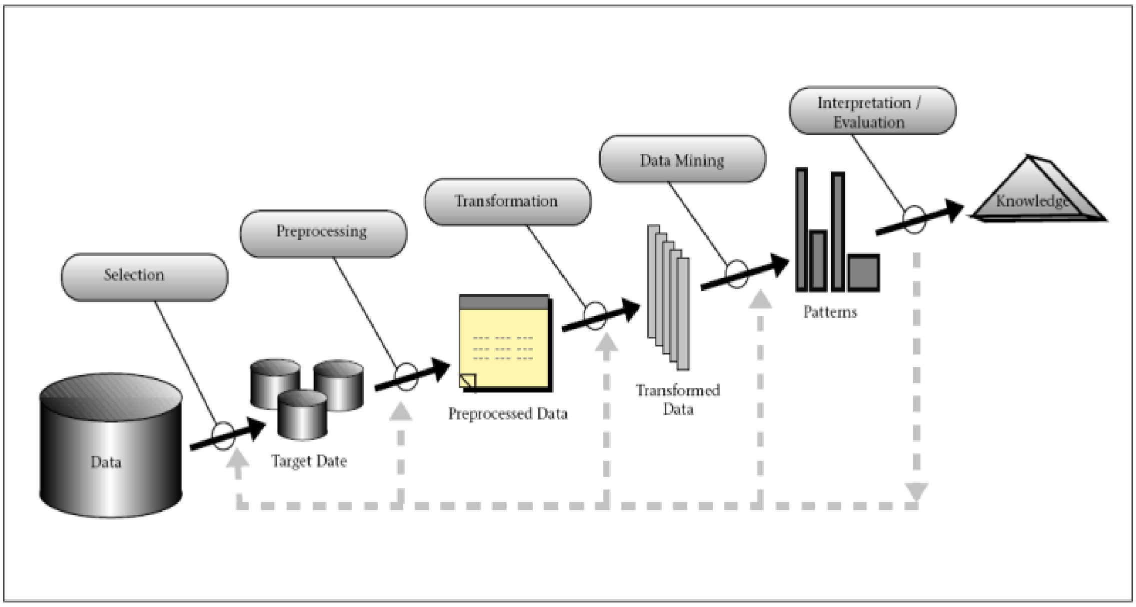

Figure 2.1: Steps That Compose the KDD Process

3) Transformation step converts the data into suitable form to help data mining algorithm run smoothly.

4) Data mining step is the main step where the user get raw results. After data mining step hidden patterns are extracted with the help of data mining techniques and presented in a desired form.

5) Interpretation and evaluation process is the last step in knowledge discovery process. The result of data mining are interpreted and if necessary checked by returning previous steps. At the end of this step extracted knowledge can be used as a report, as an analytical input or as a concept.

Knowledge discovery is the main process but until the interpretation and evaluation step data mining techniques and methods are applied and the patterns or regularities are extracted. It would not be an exaggeration to say that data mining is the most important sub-process in whole knowledge discovery process. Difference between database terminology and logic programming terminology is shown in table Table 2.1 [14] since they will be needed while making descriptions.

Table 2.1: Database and Logic Programming Terms

DB Terminology	LP Terminology
relation name p	predicate symbol p
attribute of relation p	argument of predicate p
tuple (a1....... an)	ground fact p(a1........... an)
relation p a set of tuples	predicate p defined extensionally by a set of ground facts
relation q defined as a view	predicate q defined intensionally by a set of rules clauses

Initial data mining techniques were developed to deal with propositional representation of databases. Propositional representation means single table for each target relation, and for the background facts. Propositional representation is lacking in adequacy that an instance may have different relations with different background facts. In this case algorithm execution could fail to describe exact relations, and concept descriptor may fall back in quality.

Since single-table data representation does not provide an available structure for storing complex data, multi-relational databases are appropriate for complex data storing. After multi-relational representation structures are described, previous data mining techniques become less important. Afterwards, multi-relational learning algorithms are developed and gain importance with respect to former methods. Relational learning systems can be realized by using two different approaches;

1) The first approach consists of pre-processing, hypothesis construction and post-processing steps. Since classical data mining algorithms could not work on relational structure, transformation from relational to propositional structure is needed. Now available attribute-value learning system can be used according to relevant task. After rule extraction, results could be transformed into relational form again. During the transformations both information and meaning loss is possible which is not desirable. LINUS is an example ILP based learner working with first

method. It induces hypothesis in the form of constrained deductive hierarchical database (DHDB) clauses [15]. The main idea in LINUS is to transform the problem of learning relational DHDB descriptions into a propositional learning task [16]. Then using any of three propositional learning systems ASSISTANT [17], NEWGEM [19] and CN2 [18] is applied.

2) The second approach is improves attribute-value learning system into multi relational learning system. In fact there is not much difference between two different learning systems. They fall apart from each other in terms of rule coverage testing, search techniques and of data representation. Upgraded data mining systems use first order predicate logic for background knowledge as input data. Horn clause form is used for the induced hypothesis. The learning systems which induce logical patterns valid for given background knowledge, have been investigated under a research area, called Inductive Logic Programming (ILP) [14]. The primitives of ILP is hidden in its name. So ILP based learning systems use the upgraded machine learning techniques such as prediction, classification and association rule mining. While upgrading these propositional data mining algorithms into relational ones it is enough to improve only the key notions. As an example in rule induction process refinement operator is a key notion. The well known FOIL [4] system can be seen as an upgrade of the propositional rule induction program CN2 [20]. Another well known ILP system, PROGOL [21] can be viewed as upgrading the AQ approach [22] to rule induction.

Concept is defined as a set of frequent patterns that are embedded in the features of the concept instances in the form of relations among objects [3]. Concept discovery is the problem of learning definitions of a specific relation, called target relation, in terms of other relations provided as background knowledge [4]. The concept discovery problem has extensively been studied by the ILP community with successful applications in several domains such as bioinformatics, engineering, and environmental sciences. Graph-based approach is another concept discovery method which is based on graph structure. Graph-based concept discovery methods can be classified into two main categories: Substructure-based approaches and path finding-based approaches [7].

Main purpose of Concept Learning is to develop search techniques that efficiently traverse target concept description space consisting of logical Horn clauses. There are various methods designed to solve this problem [23]:

- Top-down approach using information gain as search heuristics
- Top-down approach utilizing higher-order rule schemas to constrain search
- Bottom-up approach constraining search by generalizing from concept instances using inverse resolution operators
- Bottom-up approach making search using relative least general generalization (RLGG) operator.

Relational learning algorithm FOIL [4] was the first one that uses information gain based search heuristics. It uses an AQ-like covering approach [24] and it inherits the top-down 4 search strategy from an early concept learning system MIS [25]. Recently, many systems such as FOCL that extend FOIL in various aspects have been introduced [26]. CIA [24], MODELER [28] and RDT [27] are among the methods that use higher-order rule schemas in order to guide search for learning logical clauses. CIA learns higher-order rule schemas from induced Horn clauses via substituting variables for both terms and predicates. These schemas are employed by the system in order to explain newly introduced concept instances. In case no schema is found to explain the new instance, new rules are introduced by the system via the rule learning system CLINT. MODELER uses a different method. Instead of learning rules it employs pre-defined higher-order rule schemas. Then for the purpose of explaining the concept instance it declares additional constraints as an instantiation of rule schemas. For higher-order rule instantiation RDT utilizes the topology of clauses as an extra constraint. Relational patterns involve multiple relations from a relational database. They are typically expressed in subsets of first-order logic. As given in Table 2.1 a relation in a database corresponds to a predicate in first-order logic and the attributes of the relation correspond to the arguments of the predicate.

The search heuristics, information gain and higher-order rule schemas, have no proof-theoretic basis; therefore the search space of possible concept descriptions is not complete. The resolution rule that forms the basis of the logic programming paradigm is a sound and complete inference rule. Inverting this inference rule results in induction of refutation trees in a bottom-up fashion and systems employing inverse resolution operators have a proof-theoretic search strategy [24].

MARVIN [29] is the first ILP system inducing Horn clauses using an inverse resolution generalization operator. The language used to express the hypothesis does not contain clauses with existential quantified variables. Consequently no new predicates introduced by the system. The oracle evaluates the quality of induced clauses instead of using search heuristics to direct the search.

CIGOL [30] employs three generalization operators based on inverse resolution, which are relational upgrades of absorption, intra-construction and truncation operators used in DUCE [31], whereas MARVIN utilizes only absorption operator. With these extra operators, CIGOL extends the learning capability of MARVIN with generating new predicate definitions. CIGOL also needs oracle knowledge to direct the induction process.

PROGOL [21] is a bottom-up Horn clause induction system that uses the inverse entailment operator in induction phase. In the system, firstly the positive instance is selected to be generalized. And then, the most specific clause within the language constraints that entails the selected positive instance is constructed and the hypothesis space of clauses that are more general than this most specific clause is searched to find a qualified concept description.

GOLEM [33] is a bottom-up ILP that is based on the relative least general generalization (rlgg) operator which constraints the search space. First it randomly picks several pairs of positive examples, computes their rlgg clauses and chooses the one with greatest coverage. Covered instances are removed from inpu examples. If the final clause does not cover all positives, the algorithm will be applied to the remaining positives.

C^2D [9, 11] is a concept discovery system that uses first-order logic as the concept definition language and generates a set of concept rules having the target concept in the head. Since the predicate calculus is capable of representation of relations between objects via predicates and relations between predicates via shared variables among predicate arguments first-order logical framework is chosen as the concept definition language where concepts/relational patterns are represented by function-free concept rules.

MRDM tools are developed for discovering knowledge from multi-relational data. They can be applied directly to find relational patterns that involve multiple relations.

However, most of the previous data mining methods assume that the data represented in a single table and require preprocessing of multi-relational data to a single table before they can be applied. Yet this approach is not efficient that it can cause loss of meaning of information.

Just as many data mining algorithms come from the field of machine learning, many MRDM algorithms come from the field of Inductive Logic Programming (ILP) [14]. ILP has been concerned with finding patterns expressed as logic programs. In fact, a number of the ILP-based techniques proposed for MRDM rely on their propositional counterparts. Figure 2.2 [34] shows the relationship between propositional and MRDM algorithms proposed so far.

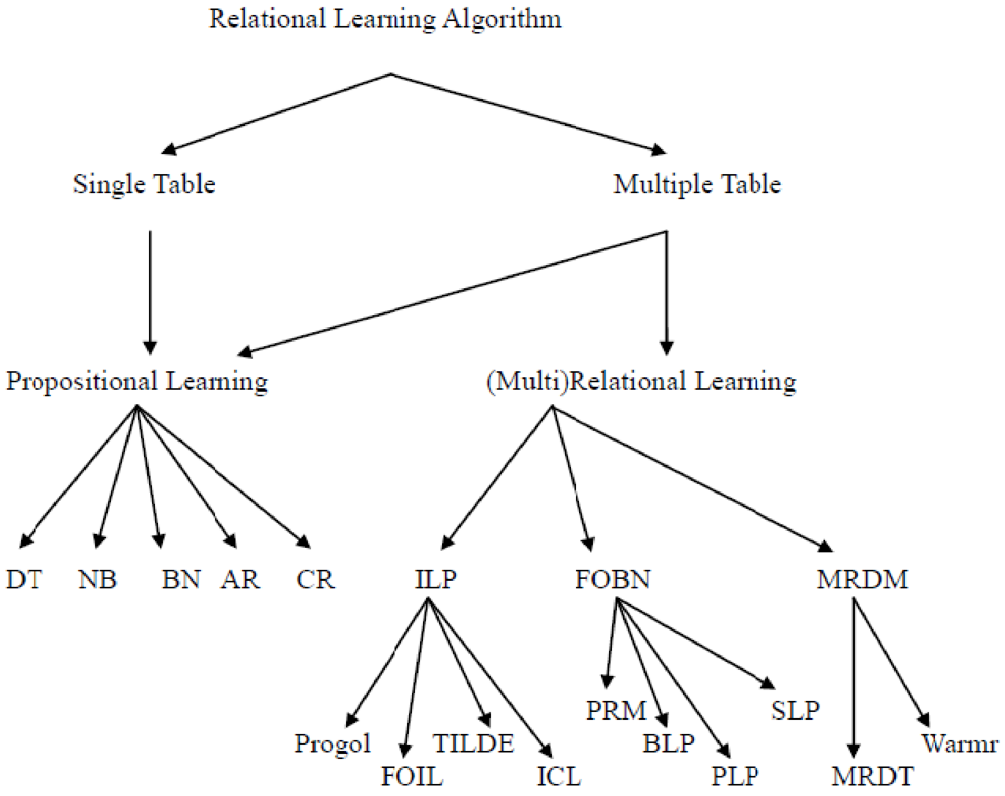

Figure 2.2: Relationships between Propositional and Multi-Relational Learning Algorithms

Graphs are another available data structures for representation of structural relations. Graph based learning systems can use data which is in the form of First-Order Predicate Calculus (FOPC). These systems use graph structure for the representation of examples which belongs to the concept to be learned. Their techniques are graph mining techniques and the search space consist of graphs. Graph-based concept discovery methods can be classified into two main categories: Substructure-based approaches and path finding-based approaches [7]. In a graph if a substructure is seen frequently then there should be a concept which constructs that substructure. On the other hand, path finding-based approaches [8] assume that a concept should appear as frequent and finite length paths that connect some arguments of positive target instances.

SubdueCL [41] represent data as a directed, labelled graph. In that graph, nodes store arguments of the facts, and labelled edges are the relation names connecting the arguments of the facts. In SubdueCL, substructures are evaluated according to the number of positive and negative target instances they explain.

Another concept learning system based on substructure discovery is Graph Based Induction (GBI) [42]. It employs colored digraph as the representation framework where colors attached to the nodes represent the attributes of the facts. GBI examines each connected pair of nodes, and merges the frequent typical ones. The final merged substructures are labeled as concepts.

Relational Path Finding [43] is one of the earliest path finding-based approaches which aims to overcome the local plateau problem of ILP-based concept discovery systems. In Relational Path finding, similar to SubdueCL, nodes represent fact arguments. Edges are labelled after the relation names and connect such pairs of nodes that they form a fact. It employs bidirectional breadth first search to discover the concept descriptors.

Relational Paths Based Learning (RPBL) [8] is yet another concept discovery system based on path finding. In RPBL, nodes represent binary facts, and edges connect nodes that share some arguments in common. To learn recursive concept descriptors, extended version of RPBL treats the target instances also as background knowledge. To apply domain theories into the learning process, they extend the graph in accordance with domain theories, i.e. by connecting nodes that hold with the domain theories.

2.2. AN OVERVIEW OF INDUCTIVE LOGIC PROGRAMMING BASED LEARNING

Most of data mining algorithms use flat/single table databases. This is known as propositional or attribute-value learning. But rapidly growing amount of data in multi-relational (multi-table) databases created the need of some other algorithm which can be used in relational data. Then multi-relational data mining algorithms has emerged. These algorithms are based on ILP and able to work directly on relational databases. Using ILP different learning systems are discovered. First-order predicate logic is employed as the representation language by the learning systems. Those learning systems which induce logical patterns valid for given background knowledge, have been investigated under a research area, called Inductive Logic Programming (ILP) [14].

Since ILP comes from both inductive learning and logic programming its techniques also inherited from these fields. Inductive learning is a subfield of machine learning and uses prediction and classification techniques. These techniques bring ILP the ability to produce a hypothesis from examples. On the other hand logic programming as a declarative programming language provide the logical representation model of hypothesis considering the facts in database.

ILP systems can be classified into two general categories according to the learning technique: predictive and descriptive learning systems. In predictive ILP systems, there is a specific target concept to be learned in the light of past experiences; however, there is no specific goal in descriptive learning and the task is to identify patterns in the data [35].

2.2.1 Predictive Inductive Learning

Predictive learning aims to learn a specific target concept in the light of past experiences. This is a kind of concept discovery task. Predictive learning can be applied to any classification or prediction problem, such as predicting carcinogenic activity of chemical compounds based on their chemical structures [36]. In this problem, the concept instance space is chemical compounds, the concept is whether a compound is carcinogenic or not and the task is finding correct classification rules that map positive instances to carcinogenic class and negative ones to non-carcinogenic class.

The problem setting of the predictive learning task introduced by Muggleton in [37] can be stated as follows:

Given:

- Target class/concept C,

- A set E of positive and negative example of the class/concept C,

- A finite set of background facts/clauses B,

- Concept description language L (language bias).

Find:

- A finite set of clauses H, expressed in concept description language L, such that H together with the background knowledge B entail all positive instances E(+) and none of the negative instances E(-). In other words, H is complete and consistent with respect to B and E, respectively.

Two criteria namely completeness and consistency are the quality mesaures in this problem setting while selecting the induced hypotheses; however the definitions of these terms require the hypotheses %100 fit the given instances, which is too strict for hypothesis to have predictive power. There may be errors in the background knowledge and training concept instances; or training examples can be sparse to reflect the general regularities hidden in the concept [15]. Success measurement of a predictive learning system depends on the ability to generalize for implicit concept instances correctly, predictive ILP systems should commision more relaxed quality criterion that allow some training examples remain misclassified.

Predictive ILP systems learn the target concept via searching hypothesis space in one of two directions: top-down and bottom-up. Bottom-up approaches start with the most specific clause containing a given positive example and generalize the hypothesis until the concept description with the background knowledge implies all positive instances. On the other hand, top-down ILP systems begin with the most general hypothesis which covers all instances and noninstances of the concept and diminish the borders of the hypothesis such that the final hypothesis covers no negative instance of the concept. Besides, top-down (bottom-up) systems may employ a generalization (specialization) operator in order to adapt the hypothesis according to given concept instances [14].

2.2.2 Descriptive Inductive Learning

Difference between descriptive ILP and the predictive data mining is the guidance of the search for concept discovery. Descriptive learning is not directed by a target concept during the search within the database. Descriptive ILP systems do not know which class or concept they are looking for in the underlying database; instead they search for interesting frequent patterns with no single target attribute, i.e. the consequent of the rules can be any attribute or relation in the data [39]. In other words, descriptive data mining systems explore relationships between the tendency of domain subjects in doing an action/having a property (buying a specific product/having cancer genetic effect) and domain-related features of the subjects (being female/having a specific molecular structure).

Descriptive data mining mainly aims to find useful/interesting and understandable patterns/regularities. Therefore, the pattern representation language and the interestingness criterion have the main role in the success of descriptive data mining systems.

Predictive ILP systems generally do not utilize the “support” concept in pruning the search space. However descriptive ILP systems like WARMR [40], which aims at identifying patterns without any specific target, utilize the general support rule, namely, APRIORI rule, as a strong search heuristics [38].

There are two important criteria for induced hypothesis in ILP. They are completeness and consistency. A hypothesis is complete with respect to background knowledge and training examples, if all the positive examples are covered. Also a hypothesis is consistent with respect to background knowledge and training examples, if none of the negative examples are covered. Given in [15] a hypothesis is;

- A hypothesis H
- Background knowledge B
- Examples ε

complete: if $covers(B,H,\varepsilon^{+})=\varepsilon^{+}$

consistent: if $covers(B,H,\varepsilon^{-})= \varepsilon^{-}$

2.3. ASSOCIATION RULE MINING

Organizations have huge amount of data about their customers, products, sales numbers etc. The aim of data mining systems is to discover knowledge about the stored data in favor of data owner for improving business gain. As an example in a supermarket product manager of the market should decide on the placement of the products on the shelves. The placement includes which products not to put on the sale or which products to be sold at a discount in order to maximize the attractiveness of the products. The best decision depends on the uncovered relations between items in the customer transaction data collected so far. For example, the information about the products that people tends to buy together can help to locate items. Also the products which are not very attracted to customers can be sold at a discount or could be removed from the shelves. The general objective of association rule mining is to find frequent associations built-in the subsets of the data and enhance the functionality of databases in a way that decision makers can query such associations. The most popular application area of association rule extracting systems is market basket type transactional databases. Many algorithms have been developed in order to find interesting Boolean association rules between sets of basket items [53,54,55,56].

Boolean association rules used for representing the relations between the attributes of a single table. Boolean association rules are in fact propositional classification rules with no single target attribute, in the form of $X \Rightarrow Y$ where X and Y are a set of conditions with no restriction on the consequent Y [35]. For example, assume that there is a table of personnel working in a company with attributes age, sex, # of children, income, title, graduate degree etc. An example association rule is "graduate degree $\geq$ master $\Rightarrow$ title = specialist" about the Personnel relation. This rule shows that personnel who have a master degree or higher tend to be in a position Specialist or higher in the company [51].

The most popular application area of the boolean association rule mining is the market basket problem. In the market basket problem, the database consists of a transaction table with columns, each of which represents a product type on sale, and rows representing baskets that store items purchased on a transaction. The goal of market-basket mining is to find strong association rules between frequent item sets

[51]. An example association rule is “Beer ⇒ Diaper”, which can be interpreted as “Beer causes diaper to be bought”.

A frequent item set is defined as an item set with support value greater than a support threshold. Support of an item set is the frequency of the item set, defined in terms of the fraction of the baskets including item set. For example, if most of the baskets in the transaction base consist of Beer and Diaper, then Beer, Diaper is a frequent item set. It can be concluded that support of an association rule A ⇒ B is the support of the item set A ∪ B.

Support of a rule shows its coverage on database. If it is smaller than the threshold, then the rule can only explain the tendency for very small part of the transactions and it is useless to take it into consideration for future business plans [51].

A strong association rule is a rule that has confidence greater than a confidence threshold. Confidence of an association rule A ⇒ B is evaluated by the probability of the baskets, having the item set A, also have items B. In other words, the confidence of an association rule is the ratio of the number of baskets that contain the item set A ∪ B to the number of baskets including the item set A [57]. If most of the baskets including Beer also include Diaper, then Beer ⇒ Diaper is a strong association rule.

Support shows the generality of the rule and the confidence designates the validity of the rule [51]. If Beer, Diaper is a frequent item set and Beer ⇒ Diaper is a strong association rule, then Beer ⇒ Diaper is valid for most of the baskets in database.

The common approach in market-basket area is first finding frequent item sets and then deriving strong association rules from these frequent item sets. In this approach, the main critical point is to extract frequent item sets from transaction database efficiently. The naive method is to count all combinations of items in the portfolio that appear in the transactions. If the number of item types in the shelves is n, then each of $2^n - 1$ item sets should be counted against the database, which is exponential and costly. Therefore, most association rule mining algorithms develop smart methods in order to reduce the number of item sets to be counted [57]. APRIORI [54] is the most popular and well known association rule mining algorithm.

2.3.1 Relational Association Rule Mining

Target of association rule mining is to extract patterns or regularities in data. In boolean association rule discovery, one object type and one table exist defining different attributes of this object type. The patterns mined are feature sets that are common for number of objects exceeding a frequency threshold. For example, in the market-basket problem, the objects are baskets, each item is one feature of the basket and the patterns are the frequent item sets common in baskets.

In relational association rule mining, there are more than one object type and the patterns are not only feature sets but also they consist of relations between objects. Relational association rule mining can be described as discovering recurrent relational patterns in a relational database [51].

The features of different objects and relations among them can not be represented by feature/item sets. Instead of propositional representation item sets should be upgraded to first-order logic form. Each relation, in first-order logic, is represented by a predicate and the relations of objects are represented by variables. In relational association rule mining the predicate sets are actually first-order queries; and the main task is to discover the interesting queries which best match the database.

For example, we have a relational database including five relations represented by Prolog facts in Table 2.2.

Table 2.2 : An Example Prolog Database

person(mahmut) person(mehmet) person(yunus)	sister(elif,mahmut) sister (tugba,mahmut) sister (sibel,mehmet) sister (gamze, yunus)	nurse(gamze) student(sibel) manager(tugba) teacher(elif)

Using the database in Table 2.2 relational pattern “people whose sister is a teacher” can be stated as query in a predicate set as follows: person(X), sister(Y, X), teacher(Y).

It can be derived that the queries are in fact item sets in which items are related by variables. In other words, they are “relational item sets” [40].

Consequently first-order query and the key predicate with its key fields can represent a relational pattern.

2.3.2 First-order Association Rules

In relational association rules meaning of the association rule, as in relational patterns, is determined by the key object of the rule. There is causality relations between objects of different relational association rules.

In boolean association rule mining, the meaning of a Boolean association rule is unique since there is only one object in database. However, in relational association rule mining, it is possible to generate association rules that contain objects other than the key objects and the head of the rule does not include key object or objects. For instance, we can conclude the following association rule from the predicate set person(X),sister(Y,X),teacher(Y) where “person” is the key predicate:

person(X), sister(Y,X) ⇒ teacher(Y) (Prolog form)

The object that will be counted here is X, since the key predicate is “person”. The rule includes sister Y as an object different from key object and the head of the rule does not include key object X. As a Prolog clause, it can be interpreted as “If a person has a sister, then this sister is a teacher”; however, the rule, as an association rule, relates a person’s feature of having a sister with the feature of the same person having a sister who is a teacher. In the first interpretation the key object is the sister; in the second, it is the person.

The difference in meaning of relational rules in predictive and descriptive framework results in difference in the notation of relational rules. While relational rules are shown as X ⇒ Y in Prolog form, the association rules, called “query extensions”, are shown as X ∼> Y [5].

In case the key predicate is the predicate in the head of the rule and the key variables of the association rules occur in the head the meanings of relational association rules and Prolog definite clauses can be same. Therefore, an association rule finder can be used as a predictive data mining system with support concept if the system guarantees that the head predicate of the rule is the key predicate and a subset of the variables in the head consists of the key variables [51]. In this study, since attributes of the head predicate are always considered as key objects, the issue is not discussed.

2.4. APRIORI

APRIORI is most popular and well known association rule mining algorithm. It has the following property, "All subsets of a frequent item set must be also frequent", in order to prune candidate item set space:

The property contributes the rule induction process in a positive way that if an item set is not frequent then any superset of this set is also not frequent. It can be concluded that the item set space should be traversed from small size item sets to large ones in order to discard any superset of infrequent item sets from scratch. In order to apply this reasoning, APRIORI reorganizes the item set space as a lattice based on the subset relation, as shown in Figure 2.3 [52].

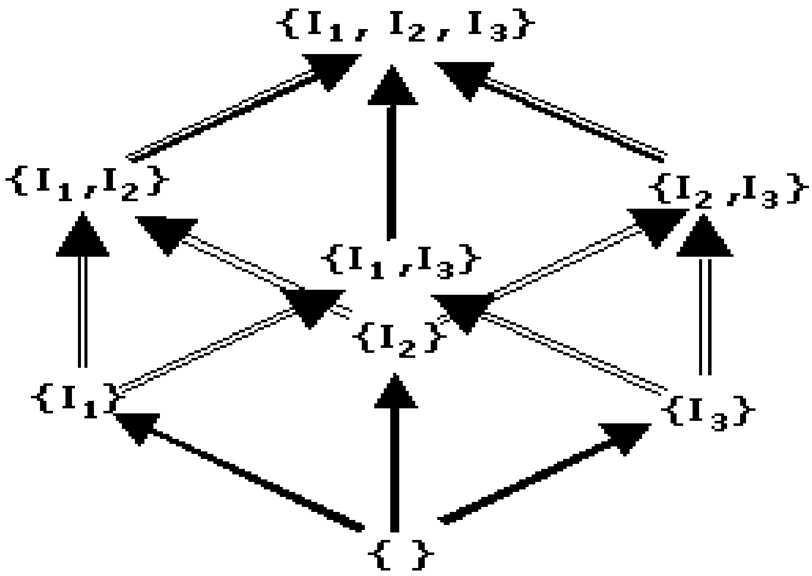

Figure 2.3: The APRIORI Lattice with Three Items

The item set lattice in Figure 2.3 is composed of possible large item sets for items I_1, I_2, I_3. The directed lines in the lattice represent the subset relationships, and the frequent item set property says that any set in a path below an item set is infrequent if the original item set is infrequent. For instance, if the item I_1 is not found frequently in transaction baskets, then the item sets { I_1, I_2}, { I_1, I_3} and { I_1, I_2, I_3} are not frequent, either.

If all subsets of an item is frequent then this item set can be accepted as a candidate. Here frequent item set means the number of occurrences of it in transactions is greater than the support threshold value.

APRIORI algorithm proceeds levelwise in the lattice as follows [51]:

Step 1. All item sets of size 1 (items itself, I_1, I_2, I_3) are used as candidate item sets, C_1, in the first step. Find large item sets from C_1 that appear at least fraction min_sup (support threshold) of baskets. This set of large item sets is expressed as L_1.

Step 2. Generate n+1-element candidate item sets C_{n+1} from n-element large item sets Ln by combining n-element large item sets that have n-1 items in common.

Step 3. Scan the database to count (n+1)-element candidate item sets in transactions and decide if they are large. The resultant set of (n+1)-element large item sets is L_{n+1}. Go to step 2 if L_{n+1} is not empty set, otherwise go to step 4.

Step 4. Output $L_1 \cup L_2 \cup \cdot \cdot \cdot \cup L_n$.

APRIORI scan the database per level. This results in n+1 database scans, and if item set lattice is too deep it becomes costly. In order to prevent this weakness of the APRIORI algorithm, most data miners limit the maximum length i.e number of literals in a rule of a possible frequent item set.

2.5. AN OVERVIEW OF GRAPH-BASED LEARNING

Increasing need of mining structured data has been satisfied by ILP based approaches for years. Then the use of graph data structures and graph mining techniques emerged to meet this demand. Since graph topology is one of the most fundamental structures studied in mathematics, and has a strong relation with logical languages, graph mining is expected to contribute to the development of new principles in data mining. Furthermore, graph mining has a high potential to provide practical applications because the graph structured data widely occurs in various practical fields including biology, chemistry, material science and communication networking [44].

A graph is a structure of objects where some connections exist between these objects. In graph theory the more formal definition is as follows: G is a graph in which a set of vertices or nodes V, and as connector of the vertices a set of edges E are the elements of it. There is also a mapping function between edges and node pair which indicates the which edge belongs to which vertices. More interpretable representation of graph structure is:

$$G(V,E,f) = \begin{cases} V; \text{ is a set of vertices} \\ E; \text{ a set of edges connecting some vertex pairs in } V \\ f \text{ ;a mapping } f : E \quad V \times V . \end{cases}$$

The popularity of Graph-based data mining is increasing in recent years but graph structure has a long history behind. Although graph-based data mining approaches are new, five theoretical categories are inherited from graph research area in recent years. These theoritical aspects are subgraph categories, subgraph isomorphism, graph invariants, mining measures and solution methods.

- **Subgraph categories**: Graph-based data mining aims at finding substructures in whole graph structure. Target of search may be various characteristic substructures. It could be induced subgraph, connected subgraph, ordered tree or a path.

- **Subgraph isomorphism**: There could be topologically identical subparts in different graphs. The similarity of subgraphs of two different graphs is checked with a bijection mapping. This mapping crosschecks the vertices and edges of both subgraphs.

- **Graph invariants**: Proporties which indicate topological structure of a graph are called graph invariants. Number of loops, number of vertices and number of edges connecting vertices are some examples of graph invariants. Naturally it can be said that isomorphic graphs have exactly the same graph invariants. But vice versa, if two graphs or subgraphs have identical graph invariants values it is not certain that they are being isomorphic.

- **Mining measures**: Detecting frequent substructures in a graph is the main stage in graph mining but there is also various measurements similar to classical data mining measures.

- **Solution methods**: Searching graph structure for graph isomorphism detection is the main subject of solution methods. There are five search methods in graph mining. greedy search based approach, inductive logic programming (ILP) based approach, inductive database based approach, mathematical graph theory based approach and kernel function based approach [44].

Among several problems in concept discovery, a common problem faced by ILP-based concept discovery systems is the so called local plateau problem [6] . In such cases classical operators of ILP that refine concept descriptors by one literal at a time are insufficient to improve the quality of the concept descriptors and the systems perform a blind search as shown in Figure 2.4 [43].

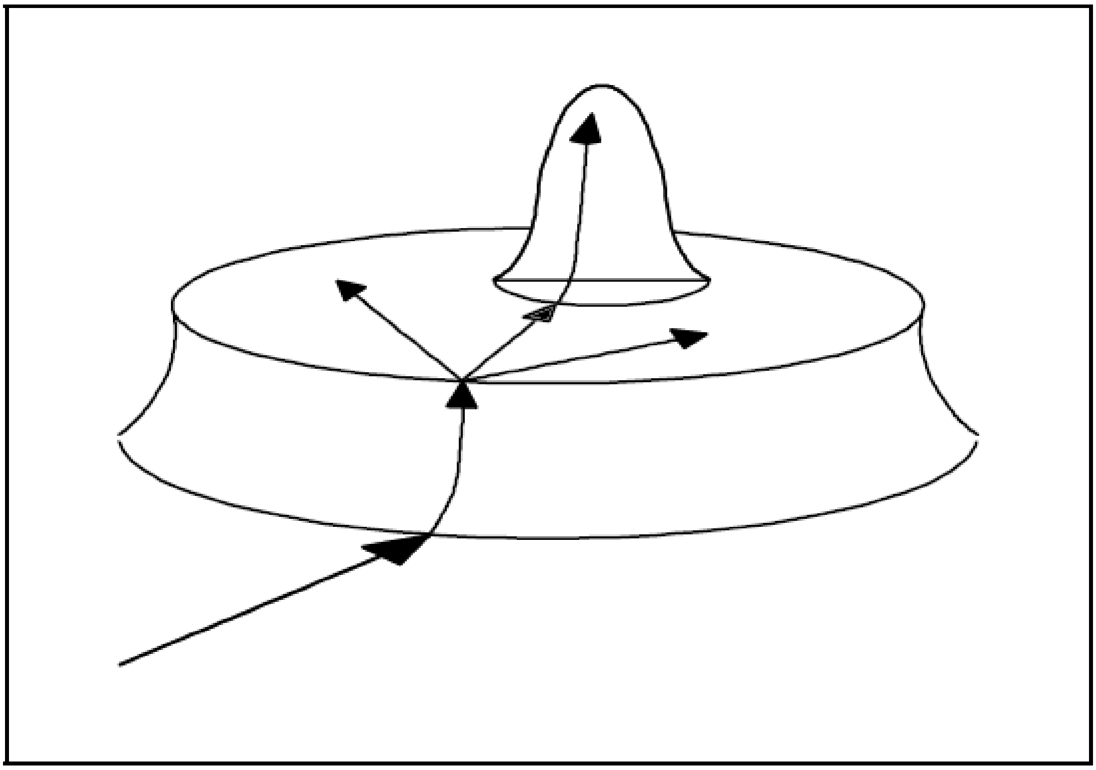

Figure 2.4 : The Local Plateau Problem

FOIL [4] is an examle of learning systems which use first-order Horn clauses and employ hill-climbing technique to learn clauses one literal at a time. Local plateau problem is the result of the systems which use hill-climbing heuristic, i.e. the learning is progressed by adding literals one by one in every step. Categorization of graph mining algorithms is given in Figure 2.5 [45].

Graph based approach is introduced by Richards and Mooney [43] to cope with Local Plateau problem. Beginning of the problem is, in order to improve the classification accuracy of a rule, two or more literals must be added simultaneously. There may be many single relations which do not decrease the category accuracy, but which of these will lead to an eventual improvement? This approach is based on the assumption that, in relational domains, there usually exists a fixed-length path of relations linking the set of terms satisfying the goal concept [43].

Discrimination between graph based approaches rely on their search strategy, data representation, algorithm approach and hypothesis evaluation.

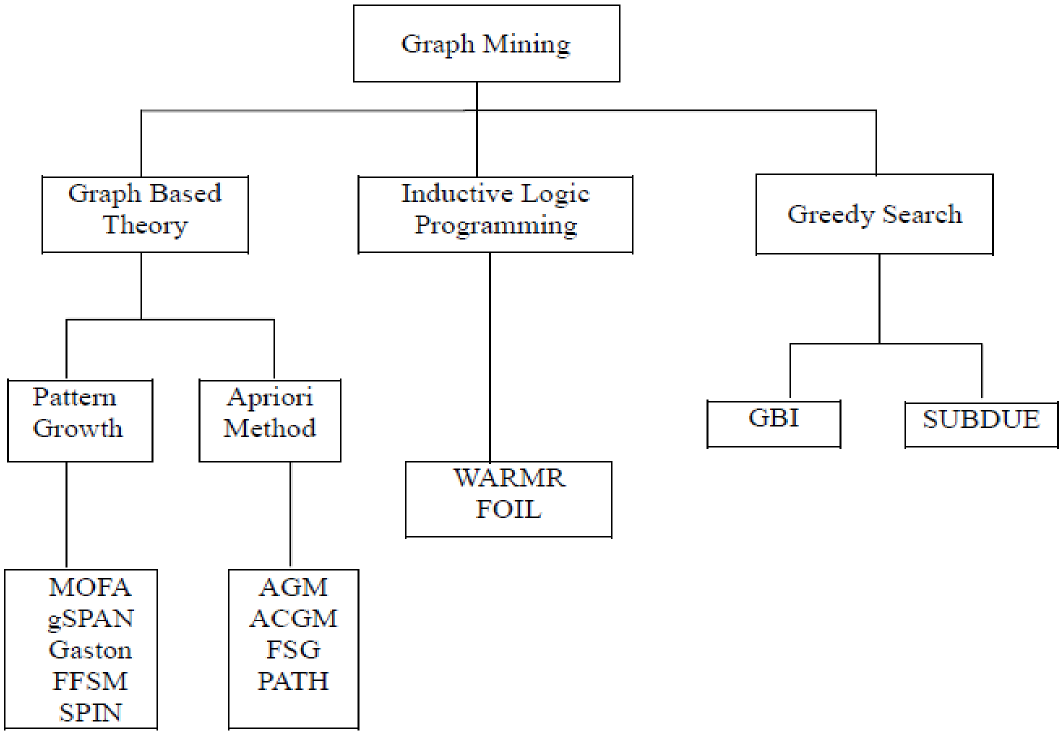

Figure 2.5 : Categorization of Graph Mining Algorithms.

Graph-based concept discovery methods can be classified as Substructure-based approaches and path finding-based approaches. According to substructure based approach; in a graph if a substructure is seen frequently then there should be a concept which constructs that substructure. On the other hand, the idea of path finding-based approaches assume that a concept should appear as frequent and finite length paths that connect some arguments of positive target instances.

2.5.1. Substructure-Based Approaches

Substructure-based concept discovery is simply the process of identifying similar and common sub-structures within a structure of graph. The aim in here is to discover such structures that compress the data and use them as the descriptors of the dealing concept. Discovered substructure help the user for data structure simplification and data interpretation.

SUBDUE [46] is one of the well known substructure based concept discovery system which finds repetitive substructures within a graph representational data. Discovery algorithm of SUBDUE takes directed and labelled multigraphs as input. In the structure arguments are represented as nodes and relations are shown as labels on edges that connect the vertices. As stated in Figure 2.4 [45] SUBDUE uses greedy based search. Medium description length (MDL) [47] value is used for the subgraph discovery that compress the initial graph structures best. MDL value is calculated with the equation given in Equation 2.1.

$$compression = \frac{DL(S)+DL(G|S)}{DL(G)} \qquad \textbf{(Equation 2.1)}$$

In the equation DL(S) is the description length of the substructure, DL (G|S) is the description length of the graph after being compressed by the substructure and DL(G) is the description length of the original graph. The smaller compression ratio indicates the better substructure. Here description length refers to number of bits required for substructure/graph description.

At the beginning a single vertex is accepted as input substructure. Then algorithm expands this single node with the edges. New nodes added to the other point of this edges. Substructure expansion goes on according to MDL value and this continues with each iteration via a new edge/edges or nodes. After having reached the given threshold length or no possible new substructure is left the algorithm ends. Discovered subgraphs are used as input for following iteration during the whole process.

Another substructure based concept descriptor is SubdueCL [41]. CL means concept learner and this system is an extension of previously explained Subdue. Different from its predecessor Subdue includes negative examples into process. It takes advantages of the same function which Subdue uses but with the inclusion of negative examples learning methodology changes so does the evaluation step.

SubdueCL works as a supervised learner by differentiating positive and negative examples using a set-covering approach instead of graph compression. The hypothesis found by SubdueCL is a disjunction of conjunctions (substructures), i.e., the concept may contain several rules. SubdueCL forms one of these conjunctions (rules) in each iteration. Positive example graphs that are described by the substructure found in a previous iteration are removed from the graph for subsequent iterations [48].

In the evaluation process the inclusion of the substructure into the concept is decided after looking at the evaluation formula. This formula indicates the goodness of the substructure which means covering ratio of positive example without covering negative ones. Before the goodness value calculation the error value is computed. Error consists of uncovered positive examples and covered negative examples. The value of substructure is given in Equation 2.2a. [48] with respect to error value given in Equation 2.2b. [48];

value = 1- Error **(Equation 2.2)**

$$\text{Error} = \frac{PosEgsNotCovered + NegEgsCovered}{PosEgs + NegEgs}$$ **(Equation 2.3)**

Referring to dealing substructure the arguments of the formula defined as; #PosEgsNotCovered is the not uncovered positive examples , #NegEgsCovered is the number of covered negative examples, #PosEgs is the number of remaining positive examples in the training set after the ones which covered in a previous iteration removed and #NegEgs is the number of all negative examples.

Algorithm of SubdueCL [48] use beam search method and starts with positive graph examples, negative graph examples, beam length and the number of the substructure as a limit. A SubdueCL function is used in order to form the hypothesis H that describes the positive examples. H is expanded each time with a substructure after SubdueCL function is called. If SubdueCL returns NULL, the beam is increased for larger search space. Positive examples covered by substructure which is found by SubdueCL, are removed from the positive graph. SubdueCL function creates a substructure for each nodes and builds a ParentList. The number of substructures to be kept by ParentList is settled according to Beam size. Every substructure in the list is expanded by one edge or one vertex and an edge in all possible ways and evaluated according to equation 3 upwards. The substructure which cover at least one positive example are hold in the BestList considering the Beam size. The substructures that were obtained from the expansion of the substructures in the Parent-

List are kept in ChildList. The Limit indicates the expansion quantity, but if the there is no substructure in BestList is after expanding, the limit is increased until one is found. At the end BestList with extarcted substructures is returned by the SubdueCL function.

Graph Based Induction (GBI) [42] is another Concept Learning system relies on substructure based approach. It continuously break joined pairs of nodes into pieces to find typical patterns in graph data. Greedy search is used during the process and no information loss is appears despite the chunking of two adjoined nodes. Tree structured data with node and link labels are accepted as input. General graph data having directed or undirected with colored or uncolored nodes and loops could be handled by GBI [49]. GBI looks for the substructure in graph and represent found pattern with one node. This compression is based on the frequency of the subgraph. After the merge of nodes into one node this typicality is called as concept.

2.5.2 Path Finding Based Appraches

The motivation behind this approach is to find similar frequent paths which links the arguments of target concepts via graph traversal algorithms.

Among several problems in concept discovery, a common problem faced by ILP-based concept discovery systems is the so called local plateau problem [6]. In such cases classical operators of ILP that refine concept descriptors by one literal at a time are insufficient to improve the quality of the concept descriptors and the systems perform a blind search.

Relational Pathfinding [43] is one of the earliest pathfinding-based approaches which aims to overcome the local plateau problem of ILP-based concept discovery systems. In Relational Pathfinding, similar to SubdueCL, nodes represent fact arguments. Edges are labeled after the relation names and connect such pairs of nodes that they form a fact. It employs bi-directional breadth first search to discover the concept descriptors. In this approach while background facts could be reprsented as binary and unary relations, target concept can be in n-ary relation form. Firstly disjoint graphs are created as much as the arguments of target relation each of which is a single node. These graphs also has a endvalue representing the expansion of every node after they are added to previous graph structure. At the end of each expansion intersection of endvalues are crosschecked and if any intersection is found then expansion stops. Another interruption reason is reaching user defined expansion bound. Exploration of intersection between two endvalues indicates there exist a path. Every subpaths finally merged and the final path is formed. In case of more than one final path occurunce the best one based on their accuracies is selected.

Relational Paths Based Learning (RBPL) is another path finding based approach having data in the form of labeled directed graph. Different than Relational Pathfinding it stores relations in the nodes not the arguments of facts. Edges provides the connection between the vertices which have common arguments and values.

Relational Path Finding uses relational paths to escape from local maxima and local plateaus. However, RPBL uses relational paths as explanations of positive examples, and avoid local maxima and local plateaus. In essence, RPBL is a combination of inductive learning with analytical learning, and Relational Path Finding belongs to inductive learning. RPBL generalizes relational paths to get hypothesis, instead of using relational paths as heuristics for adding literals. In addition, RPBL is a simultaneous covering approach, and Relational Path Finding is a sequential one [50]. Structures used in RBPL approach is defined below.

- **Structured Instance Space (SIS):** This graph structure keeps the nodes as binary facts, and edges as constants between vertices.
- **Expanded SIS (E-SIS):** After applying domain theories to SIS this enhances.
- **Relational Path (RP):** Any path in SIS which connects arguments of positive target concept.
- **Specialized RP (S-RP):** Specialization of RP having unary relations that respresent constants on RP.
- **Generalized S-RP (G-S-RP):** Generalization of S-RP that obtained by variablizing constants on S-RP.

A target instance and a maximum length is required at the beginning. The SIS is searched for the RPs which connect arguments of target instance. Unary facts are added for the constant arguments of RPs to obtain S-RP. After generalization G-SRPs appears and combinations of G-S-RPs composes the hypothesis in the form of Horn clauses.

Mode Directed Path Finding [61] is a concept discovery method that uses saturated bottom clause. Saturated bottom clause is bottom up ILP based concept discovery approach. Distinguishing property of Mode Directed Path Finding compared to other path finding based approaches is both using hypergraph instead of graph and working on a saturated bottom clauses. In the body of the clause each literal is represented as a node of the hypergraph. Input/output mode declarations are used to create the edges between the arguments of the literals. As the origin of the path head literal is represented by n distinct vertices i.e. n is the arity of the head literal. Path length is 0 at the beginning and in a depth first manner expanded nodes are added to the list that is associated with the path. Every found paths are compared to each other and if a similarity is discovered the paths are merged with their lists and vertices. The result of the merging gives the concept descriptor.

Head Output Connected (HOC) predicate learning is an other study working on saturated bottom clause. It is a special clause of the clauses induced by path finding learning systems. Two cases indicates the clause as a HOC:

1) In each body literal input variable has to be either in the head literal or in the previous body literal and

2) All head output literals has to be instantiated in the body of the clause.

2.5.3 A Hybrid Graph Based Consept Discovery Method

Hybrid Graph based concept discovery method employs both techniques from path finding and substructure based approaches. It searches for paths which form the concept descriptors on a compressed graph [7]. In this approach concept descriptions are induced during the construction process of the graph structure.

The method has some similarities and distinguishing points in comparison to other state of art graph based systems. Similar to substructure based approach it looks for similar facts and merge them as a single vertex. It also infers the concept descriptors by inducing paths which connects arguments of the target instances. The method differs from other graph based concept discovery methods in such a way that induction of concept descriptors is completed during the graph construction process.

Hybrid graph based concept discovery method is limited by working on binary relations. The nodes of the graph structure in the method stores arguments of the relations. The connection of the vertexes is created by Head Output Connected (HOC) class of learning probelms.

2.6. A COMPERATIVE STUDY ON GRAPH BASED LEARNING AND ILP BASED LEARNING

Graph mining has a strong relation with the aforementioned multi-relational data mining techniques. However, the main objective of graph mining is to provide new principles and efficient algorithms to mine topological substructures embedded in graph data, while the main objective of multi-relational data mining is to provide principles to mine and/or learn the relational patterns represented by the expressive logical languages. The former is more geometry oriented and the latter more logic and relation oriented.

ILP based approaches dominated the multi relational learning area for decades. Then graph based approaches are firstly introduced to solve local plateau problem of ILP based learning. Recent years were a very fertile era for graph based approaches. The simplicity of graph representation of data and being more interpretable are the advantages of graph based concept learning. Although tools of ILP methods are still being used improvement in search space pruning and bias help graph based learning step forward.

2.7. PRELIMINARIES

Some of the common graph terminology used in this thesis before describing the technical details of graph-based concept discovery process is presented.

Definition 1 (Graph): A graph G = (V, E) consists of a set of vertices V and a set of edges $E \subseteq (V \times V)$. An edge $e = (u, v) \in E$ starts from vertex u (source) and ends at vertex v (target). The vertices u and v are incident with the edge e and adjacent to each other.

Definition 2 (Subgraph): A subgraph of a graph G = (V, E) is a graph Gs =(Vs, Es) where $Vs \subseteq V$ and $Es \subseteq (Vs \times Vs) \cap E$.

Definition 3 (Path): Let G = (V, E) be a graph. Assume that $v_i \in V$ for i = 0, 1,......, t. We say that there is a path from v_0 to v_t if v_i and v_{i+1} are adjacent to each other for i = 0, 1,...., t - 1.

Definition 4 (Connected Graph): A graph G = (V, E) is connected if there is a path between any pair of nodes u, v $\in$ V .

In this thesis only connected subgraphs is considered. Given a positive integer k, we use the term size-k subgraph to denote a connected subgraph with k edges.

Assume that we are given a collection D of m graphs represented as D = $\{G_1,G_2,.....G_m\}$. Here, all the G_is have the same set of labeled vertices, V . Hence, we represent each graph as $G_i = (V , E_i)$ where $E_i \subseteq (V \times V)$. The frequency of a subgraph G' over D denotes the number of graphs $G_i \in D$ that contains G'. The following function computes this:

$$\text{frequency}(G',D) = \sum_{G_i \in D} \text{exists}(G' , G_i)$$

Frequency of an edge is defined, e in D (denoted with frequency (e, D)) as the number of $G_i \in D$ that contain e. We normalize the frequency of each edge to compute another function denoted by fr(e,D) as $\text{fr}(e,D) = \frac{\text{frequency}(e,D)}{|D|} = \frac{\text{frequency}(e,D)}{m}$

Definition 5 (Template Graph): Let D be a collection of graphs G_i with $G_i = (V, E_i)$. The template graph of D is an edge weighted graph $T = (V, E_T , \psi())$, where $E_T = \bigcup_i E_i$ and ψ () is a function $\psi : E_T \rightarrow R$ such that $\forall e \in E_T$, $\psi (e) = - \log(\text{fr} (e, D))$.

Intuitively, the template graph summarizes the frequencies of all edges in the graph dataset. A frequent size-k subgraph G appears in many graphs in the graph dataset. As a result, it is conjectured that it consists of frequent edges of the template graph.

Definition 6 (Graph Isomorphism): Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs. G_1 is isomorphic to G_2 if there exists a mapping $\emptyset : G_1 \rightarrow G_2$ such that for each vertex $v \in V_1$, $\emptyset (v) \in V_2$ and for each edge $e = (u, v) \in E1$,$\emptyset (e) = (\emptyset (u), \emptyset (v)) \in E_2$.

Definition 7 (Subgraph Isomorphism): Let G_1 and G_2 be two graphs. There exists a subgraph isomorphism between G_1 and G_2 if there exists a graph $H \subseteq G_2$ which is isomorphic to G_1.

Definition 8 (Concept): A concept is a set of frequent patterns, embedded in the features of the concept instances and relations of objects belong to the concept with other objects.

Concept rules are expressed in definite clause format (h⟵b) in this thesis this, in which h is a single positive literal and b consists of positive literals.

2.8. SUPPORT AND CONFIDENCE

Two criteria are important in the evaluation of a candidate concept rule: how many of the concept instances are captured by the rule (coverage) and the proportions of the objects which truly belong to the target concept among all those that show the pattern of the rule (accuracy); support and confidence, respectively. Therefore, the system should assign a score to each candidate concept rule according to its support and confidence value.

The **support** value of a concept rule C is defined as the number of different bindings for the variables in the head relation that satisfy the rule, divided by the number of different bindings for the variables in the head relation. In other words, it is the ratio of number of positive target instances captured by the rule over number of target instances.

Let C be $h \longleftarrow b$,

$$\mathrm{support}(h \longleftarrow b) = \frac{|\text{bindings of variables for h that satisfy } h \longleftarrow b|}{|\text{bindings of variables for h that satisfy h}|}$$

The **confidence** of a concept rule C is defined as the number of different bindings for the variables in the head relation that satisfy the rule, divided by the number of different bindings for the variables in the head relation that satisfy the body literals. In other words, it is the ratio of number of positive target instances captured by the rule over number of instances that are deducible by the body literals in the rule.

Let C be $h \longleftarrow b$,

$$\mathrm{confidence}(h \longleftarrow b) = \frac{|\text{bindings of vars for h that satisfy } h \longleftarrow b|}{|\text{bindings of vars for h that satisfy b}|}$$

CHAPTER 3 G-CDS: GRAPH-BASED CONCEPT DISCOVERY SYSTEM

Concecpt discovery, from relational database point of view, is a predictive learning task that aims to discover logical definitions of relation in terms of other relations provided. ILP-based systems have long dominated the concept discovery research. In such systems, data, target relation and the background knowledge, are represented in first-order logic and induced concept descriptors are represented as Horn clauses where the negated literal is the concept to be learned and the other liters come from the background knowledge. Althoug ILP-based systems have successfully applied in several domains, such systems face scalability issues and are vulnerable to local optima problem. Another trending direction in concept discovery is based on graphs which aims to overcome the shortcomings of ILP-based systems.

Graph-based concept discovery systems demonstrate promising results as relational data can easily be represented within graph structure and graphs have deeply studied algorithms that can be utilized for concept discovery.

In this section we introduce a hybrid graph based system for concept discovery. The system is called as Graph-based Concept Discovery System (G-CDS). In the proposed approach, nodes represent the tuples of the relations and edges connect nodes that containing common arguments. The system first builds individual graphs for each target instance and then forms a summary graph from those individual graphs. Concept descriptors are extracted from the summary graph. G-CDS is introduced to overcome the limitations of an ILP-based system called C^2D. These limitations include:

- Target instance ordering problem,
- Large search space problem,
- Extra declarations to induce transitive rules.

The proposed system overcomes these issues by;

- Considering all of the target instances at once,
- Graph sturcture prevents the generation of infeasible concept descriptors,
- Graph structure allows inducing transitive rules by interleaving other back-ground relations to the rule induction process.

In the rest of this section, we introduce a running example data set, describe C^{2}D in detail and present the proposed system.

3.1. EXAMPLE DATA SET

The elti relation given in Table 3.1, is used as a running example throughout this thesis. In this example, elti (e) is the concept to be learned, and three concept instances are given. Background knowleedge consists of three relations, namely brother (b), husband (h), and wife (w). elti is a kinship relation in Turkish and holds between two women if they are wives of two male brothers.

In the example data set, there are three positive concept instances. From relational database perspective, the negative concept instances do not exist in the elti table (the proposed techniques do not process negative data). The task is to define target relation elti(A, B), which states that person A is an elti of person B, in terms of the background knowledge knowledge, i.e. in terms of brother, husband, and wife relations.

The example data set is a part of entire elti data set which is choosen to show levels of graph construction and concept description processes. Techniques of both C^{2}D and G-CDS is explained with examples using this dataset.

Table 3.1: The Elti Data Set

Target Instance	**Background Data**	
t1:e (cemile, ayse)	b1:b(mehmet,ismail)	b2: b (mehmet, veli)
t2:e (cemile, fatma)	b3:b(sadullah,yildirim)	b4:h (sadullah,nalan)
t3:e (nalan,bedriye)	b5: h (veli,ayse)	b6:h (yildirim,bedriye)
	b7:h (mehmet,cemile)	b8: h (ismail,fatma)
	b9:w(bedriye,yildirim)	b10: w (fatma,ismail)
	b11:w (ayse,veli)	b12: w(nalan,sadullah)
	b13:w(cemile,mehmet)	

3.2. C^{2}D: CONFIDENCE-BASED CONCEPT DISCOVERY

Concept is defined as a set of frequent patterns that are embedded in the features of the concept instances in the form of relations among objects [3]. C^{2}D [9] is a predictive concept discovery system that uses first-order logic as the concept representation language and generates a set of concept rules having the target concept in the head. Since the predicate calculus is capable of representating relations between objects via predicates and relations between predicates via shared variables among predicate arguments, in C^{2}D, first-order logical framework is chosen as the concept definition language where concepts/relational patterns are represented by function-free concept rules.

In the concept discovery algorithms, a concept rule can be interpreted as a partial definition for a concept, where the head predicate identifies the defined concept and the predicates in the body of the rule represent the required features and relations for an object that belongs to this concept.

To evaluate the quality of induced concept descriptors, C^{2}D calculates support and confidence values of the rules. C^{2}D translates cocenpt descriptors into SQL queries to calculate their support and confidence values. The methods employed for transformation are illustrated in Table 3.2 and Table 3.3.

Table 3.2: The SQL Queries for Support Calculation

```
Support = COUNT1 / COUNT2
COUNT1:
SELECT COUNT(*) FROM
            SELECT DISTINCT(key fields in the head predicate)
            FROM relations in C
            WHERE conditions expressed in the rule C
COUNT2:
SELECT COUNT(*) FROM
          SELECT DISTINCT(key fields in the head predicate)
          FROM the head relation in C
```

Table 3.3: The SQL Queries for Confidence Calculation

```
Confidence = COUNT3 / COUNT4
COUNT3:
SELECT COUNT(*) FROM
        SELECT DISTINCT(key fields in the body predicates that are bound in the
                              head predicate)
        FROM relations in C
        WHERE conditions expressed in the rule C
COUNT4:
SELECT COUNT(*) FROM
        SELECT DISTINCT(key fields in the body predicates
                              that are bound in the head predicate)
        FROM relations in the body of C
        WHERE conditions expressed in the body of C
```

In order to calculate the support and confidence values for the elti example, the SQL queries for the below rule are shown in Table 3.4 and Table 3.5:

elti(A, B)⟵h(C, A).

Table 3.4: The SQL Queries for Support of Elti(A, B)⟵h(C, A).

```
Support = COUNT1 / COUNT2
COUNT1:
SELECT COUNT DISTINCT (e.arg1, e.arg2)
   FROM elti e, husband h
   WHERE e.arg1 = h.arg2
COUNT2:
SELECT COUNT DISTINCT(e.arg1)
   FROM elti e;
```

Table 3.5: The SQL Queries for Confidence of Elti(A, B)⟵ h (C, A).

```
Confidence = COUNT3 / COUNT4
COUNT3:
SELECT COUNT DISTINCT(h.arg1, h.arg2)
    FROM elti e, husband h
    WHERE e.arg1 = h.arg2
COUNT4:
SELECT COUNT DISTINCT(h.arg2)
    FROM husband h
```

The algorithm of C^2D, given in Table 3.6, starts with selecting a positive concept instance. The most general rules with two literals, one in the head and one in the body that entail the positive example are generated and then the concept rule space is searched with an APRIORI-based specialization operator. In the refinement graph, if support parameter is on and the frequency of a rule is below the support threshold, it is pruned as an infrequent rule. In addition to this, rules whose confidence values are not higher than their parents' confidence values are also eliminated. When the maximum depth is reached or no more candidate rules can be found, if confidence parameter is on, then the rules that have less confidence value than the confidence threshold are eliminated for the solution set. Among the produced strong and frequent rules, the best rule is selected and the rule search is repeated for the remaining concept instances that are not in the coverage of the selected hypothesis rules. If there is no possible best rule found for the selected positive concept instance, then the algorithm will select another positive concept instance and start from the beginning. At the end, some uncovered positive concept instances may exist because of the user settings for the thresholds.

Table 3.6: C^{2}D Algorithm

```
- Input: I: concept instance set (from DB), BF: background facts
- Output: H: hypothesis set (initially H=;)
- Parameters: min-sup: support threshold, min-conf: confidence threshold
      B (used in f-metric), md: maximum depth (body literal count)
- Local Variables: Ci: candidate rules set at level i, d: level
      G: generalization rules set, FSC: frequent and strong rules set
- Repeat until I is covered by H (until I = ;) OR
  No more possible G can be found according to given parameters:
    1. Select p from I.
    2. GENERALIZATION: Generate G of p by using BF
    3. Initialize C1:=G, FSC:=;, d:=1
    4. REFINEMENT OF GENERALIZATION:
      While Cd , ; and d _md
        a. FSC = FSC [ FREQUENT STRONG RULES(Cd, min-sup)
        b. Cd+1 = CANDIDATE GENERATION(FSC, min-sup)
          i. UNION: For each pair of the rules in level d
             compute each possible union rule.
          ii. For each union rule satisfying min-sup;
             -SPECIALIZATION: Generate rules by unifying existential variables
             -FILTERING: Discard non-frequent and less-confident rules
       c. d:=d+1.
     5. EVALUATION:
        Eliminate rules from FSC according to min-conf
        Select cbest from FSC according to selected evaluation criteria
     6. COVERING: Compute Ic _ I covered by cbest
     7. H := H [ cbest
     8. I := I - Ic
-Return H.
```

In the elti example, C^{2}D selects the first concept instance “elti(cemile, ayse)” and then finds the related fact set of the current concept instance {husband(veli,ayse), h(mehmet,cemile),w(ayse,veli),w(cemile,mehmet)}. In the generalization phase, the system generalizes all concept instances together with all related facts. For instance, by applying absorption operator to the concept instance “elti(cemile,ayse)” and to the related fact “husband(veli,ayse)”, the concept descriptions of the form

$$\{\, elti(cemile, ayse) \longleftarrow husband(veli,ayse)\}\, \theta_2^{-1}$$

are derived.

After the generalization phase, C^2D populates first level of the APRIORI lattice with these two literal concept descriptions obtained in the generalization phase. In the elti example, for the concept instance elti(cemile, ayse), two literal generalizations are generated in the presence of related facts and the first level of the search lattice is populated with these generalizations. With the minimum support 0.2 and minimum confidence 0.6 C^2D finds the following rules that can capture the description of elti.

R1: elti(A, B)⟵husband(C,A), husband(D, B), brother(C, D).
R2: elti(A, B)⟵husband(C, A), husband(D, B), brother(D, C).
R3: elti(A, B)⟵husband(C, A), wife(B, D), brother(C, D).
R4: elti(A, B)⟵husband(C, A), wife(B, D), brother(D, C).
R5: elti(A, B)⟵husband(C, B), wife(A, D), brother(C, D).
R6: elti(A, B)⟵husband(C, B), wife(A, D), brother(D, C).
R7: elti(A, B)⟵wife(A, C), wife(B, D), brother(C, D).
R8: elti(A, B)⟵wife(A, C), wife(B, D), brother(D, C).

Coverage: After the best rule is selected, concept instances covered by this rule are determined and removed from the concept instances set. The main iteration continues until all concept instances are covered or no more possible candidate rule can be found for the uncovered concept instances.

In the elti example, the search tree constructed for the instance elti(cemile, ayse) is traversed for the best rule. The rule elti(A, B)⟵husband(C, A), wife(B,D), brother(C,D) with support value of 1 and the confidence value of 1 is selected and added to the hypothesis. Since all the concept instances are covered by this rule, the algorithm terminates and outputs the following hypothesis:

elti(A, B)⟵husband(C, A), wife(B, D), brother(C, D)

3.3. PROPOSED SYSTEM

The proposed system G-CDS is a hybrid framework for graph-based concept discovery. Directed, labeled graph is employed during the construction of the system where nodes represent target and background instances, and edges connect those nodes that have at least one common argument. The proposed approach inputs the data in relational format, generates distinct graphs for each target instance, generates the summary graph from those distinct graphs, extracts concept definitions from the summary graph. Paths of length *l* that originate from a root node, i.e. a node that represents a target instance, and are existent in many individual graphs are considered as a concept descriptor.

The proposed method starts with building individual graphs for each target instances. Then these individual graphs are merged to form a summary graph. Concept descriptors are extracted from the summary graph. Construction of the summary graph resembles the propoerties of the substructure-based graph concept discovery and path extraction resembels path finding-based concept discovery.

3.3.1 The Algorithm

The proposed method, namely G-CDS, takes a set of target instances, a set of background knowledge, minimum support, minimum confidence, and maximum rule length parameters as input and outputs concept rules that describe the target relation. The target instances and the background knowledge are initially stored in a relational database. The proposed method is composed of following components:

- **Individual graph construction**: In this step, distinct graphs are created for each target instance. Vertices contain tuples from the relational database, and edges connect vertices that have at least one argument in common. Edges are labeled after relation names and they also include information about how two vertices are related. Using the elti data set given in Table 3.1. individual graph construction steps are shown in Figures below. The maximum rule length parameter is accepted as 3.

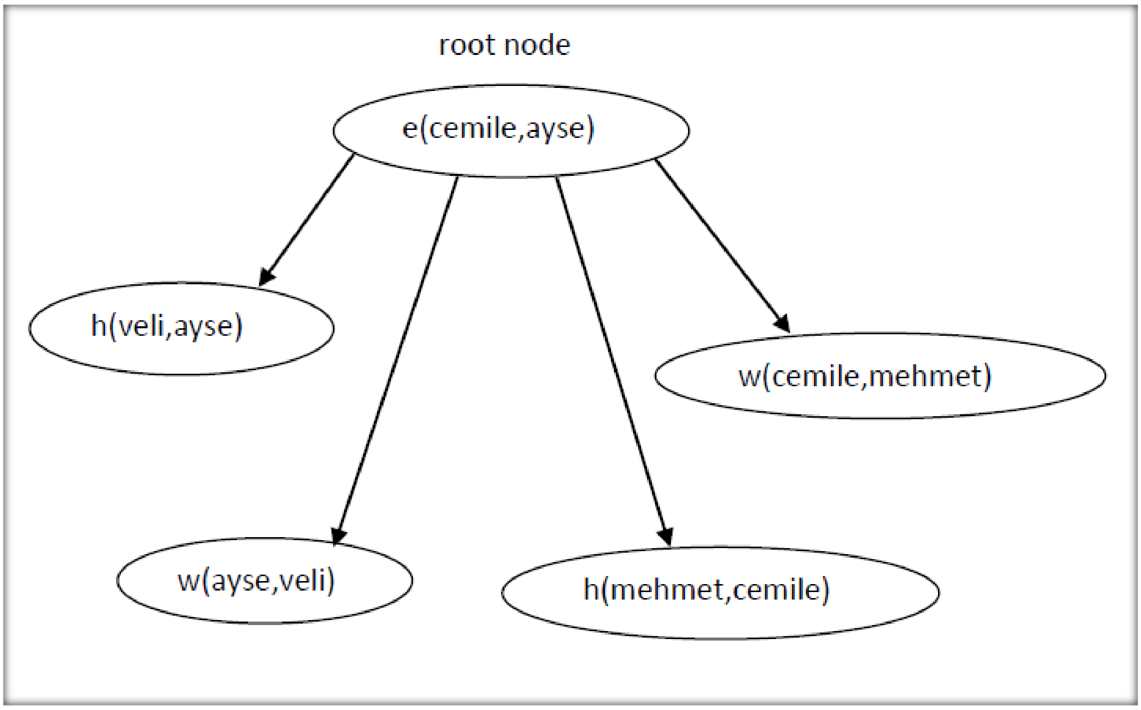

Figure 3.1: First Length Individual Graph Construction

In Figure 3.1 example target instance e(cemile,ayse) is expanded with respect to background knowledge.

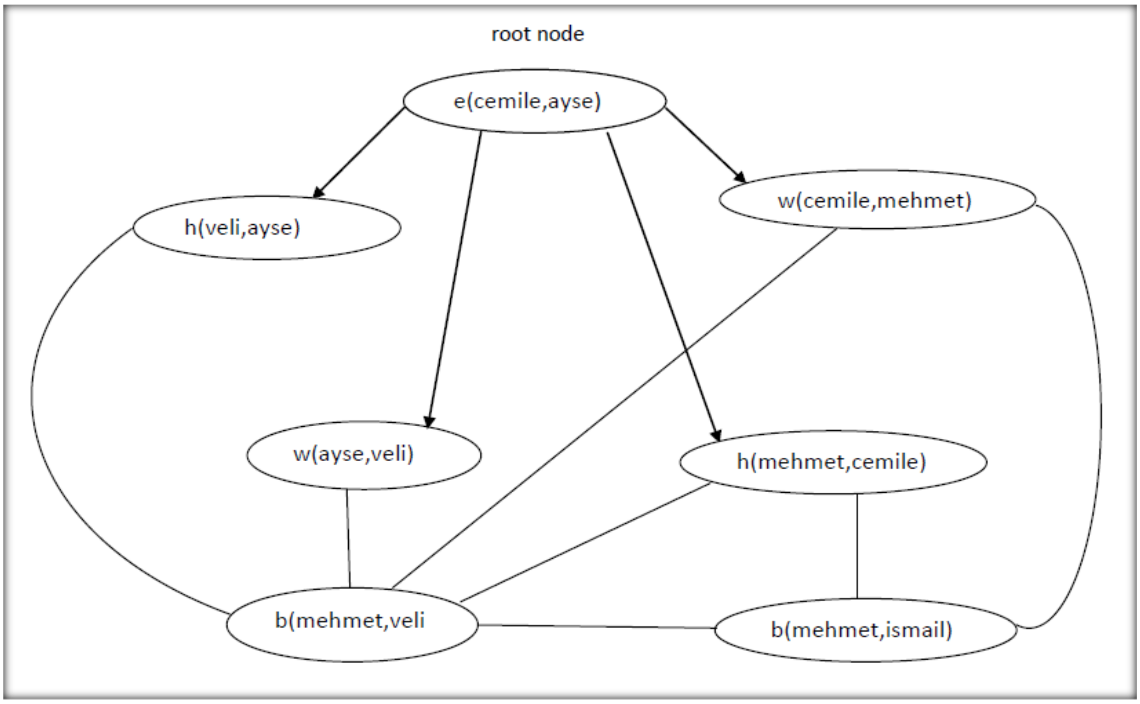

Figure 3.2: Second Length Individual Graph Construction

In Figure 3.2 graph structure is expanded using background facts which includes common argument with length one background relations.

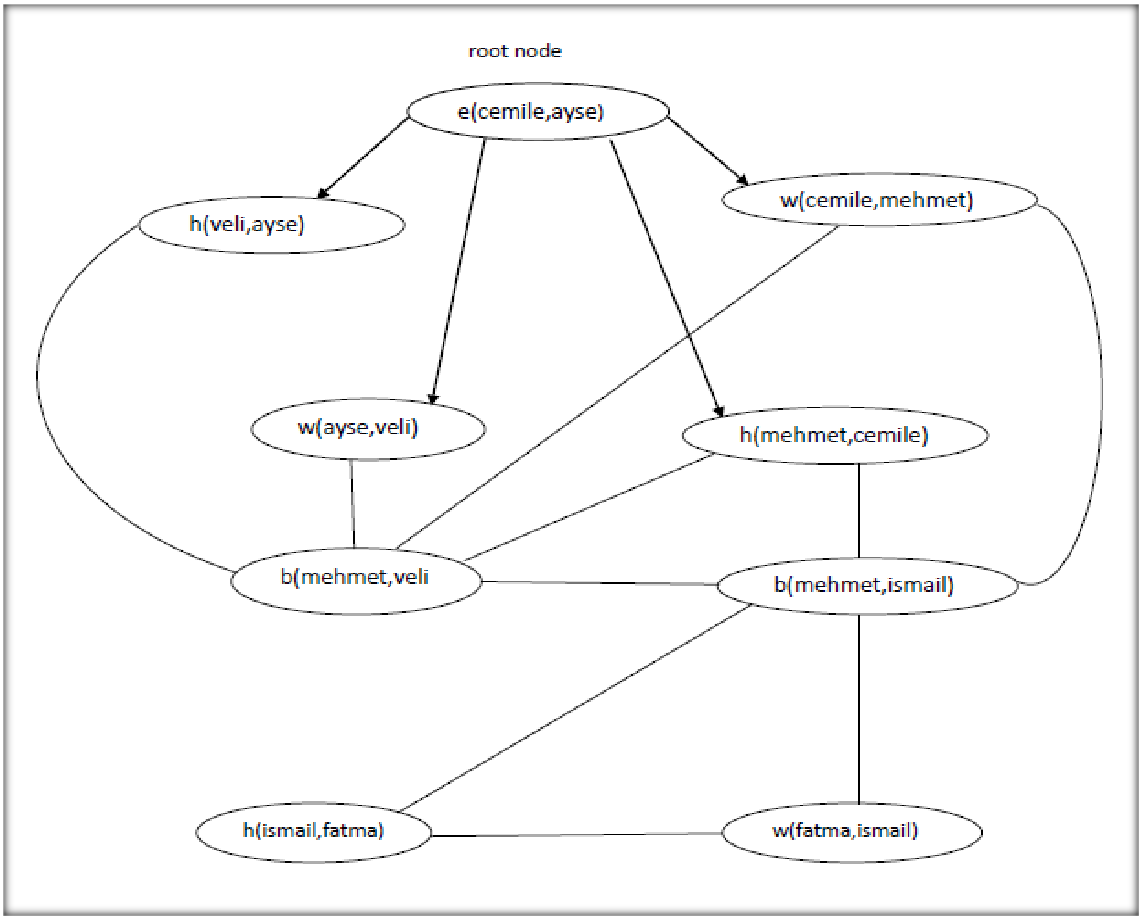

Figure 3.3: Third Length Individual Graph Construction

In Figure 3.3 graph structure is expanded with respect to background knowledge which includes the common argument with length two background relations. Since the maximum rule lenth is 3, this expansion is the last step of individual graph construction.

After constructing all the individual graphs, they are compared to obtain the summary graph. As an example, assume that v1 and v2 are two vertices of a graph that represent, respectively, e(cemile, ayse) and husband(mehmet, cemile). The edge that connects these two vertices is labeled as elti#A#B!husband#C#A. The variables in the edge label, namely A, B, C, represent the argument values, and the same variable means that an identical argument partakes in relations. Subfigures a and b of Figure 3.4 show the partial graphs of elti(cemile,ayse) and elti(cemile, fatma).

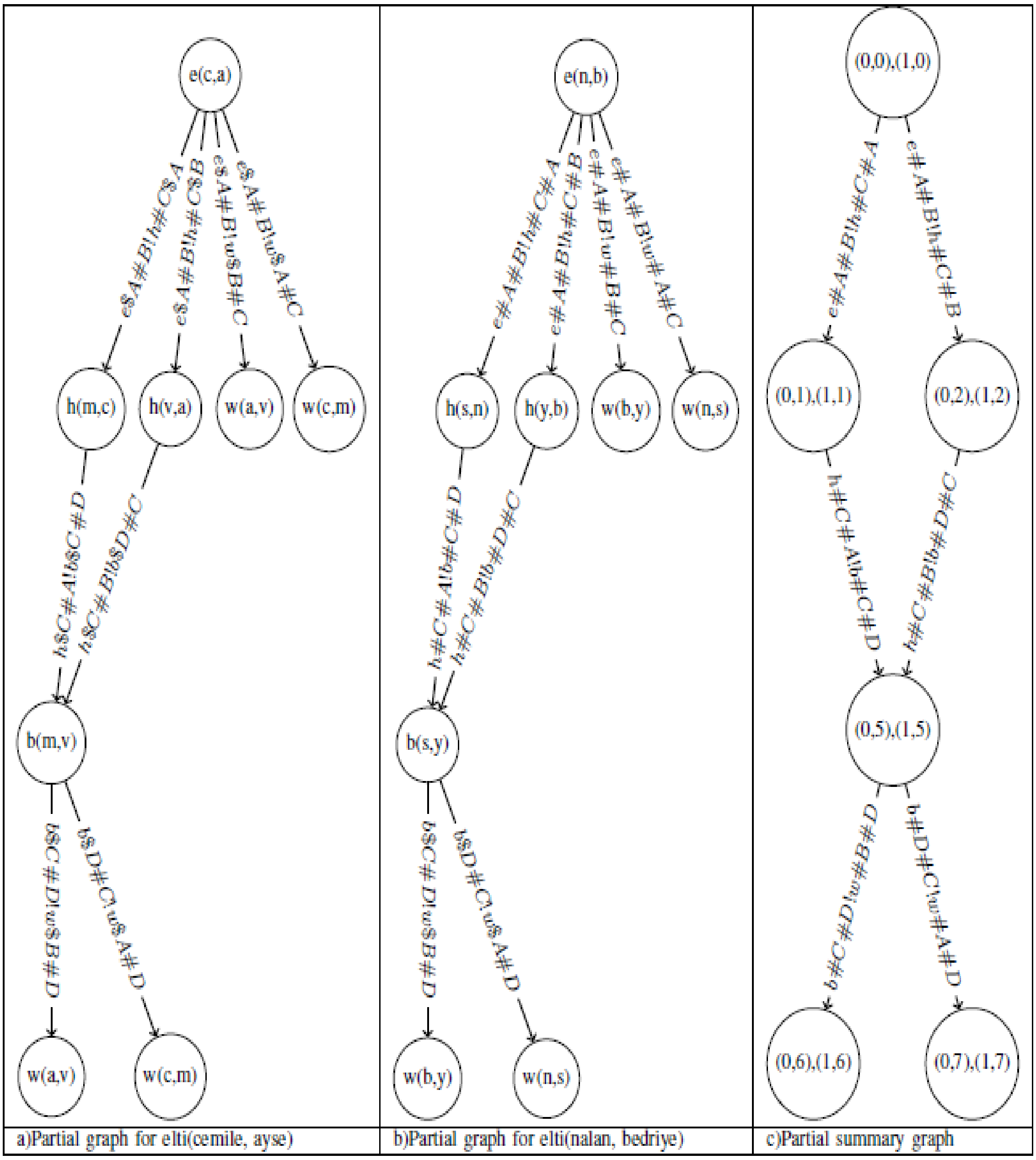

Figure 3.4: Partial Representations of Individual and Summary Graphs

- **Graphs comparison for summary graph construction**: In this step, a summary graph that represents the structure of the entire dataset is constructed from individual graphs. For this purpose we perform a level-wise deepening traversal on the individual graphs and represent nodes that are identical with respect to their parents as a single vertex in the summary graph. Two nodes are called identical if they have exactly same labeled incoming edges. As an example node h(m, c) of the first individual graph is identical to node h(m, c) of the second individual graph as they are related to their parents, i.e. e(c,a) and e(c,f) in the same way - child nodes have their parents' first argument as their second argument, and their first arguments do not appear in their parents. Subfigure c of Figure 3.4 shows partial summary graph.

The summary graph is a directed, labeled graph. Vertices contain identical nodes in the individual graphs, and edges are labeled after the relations that hold between identical nodes. The algorithm to construct the summary graph inputs a vector of nodes of the summary graph that are yet to be explored.

Algorithm to construct the summary graph is given in Algorithm 1. The algorithm examines every outgoing edge of the nodes that are associated with of the summary graph under consideration and gets their edge labels. Then, edges are grouped and those groups that have fewer elements that the threshold value are pruned. If the summary graph does not contain a vertex with an incoming edge labeled with a group label, a new node is created and the target node along with its graph id is inserted into the vector of the newly created node. Else the graph id and target node id values are appended to the node. The group size control on line 7 prunes those groups that contain fewer elements than the threshold. The algorithm continues to build the summary graph until all nodes of the summary graph are explored. The summary graph construction algorithm also avoids construction of cycles - if a < graph id; node id > pair appears in an ancestor of a node, the pair is not added to the descendants of the node. In Figure 3.5 summary graph is shown.

<u>**Algorithm 1** Summary Graph Construction</u>

Require: vector of <graph id, node id> pairs to examine: p<ps>
Ensure: Partially summarized graph
1**: while** (p is not empty) do
2**:** **for each** (p in ps) do
3: labels outedges of vertex node id in the graph constructed from target relation graph id
4**:** **end for**
5: lblGroups group labels based on their labels, each group contains label, graph id, target node id
6**:** **for each** (group in lblGroups) do
7**:** **if** (size of the group threshold) then
8**:** **if** (no vertex associated with the label is created) then
9: create new vertex, insert associated graph id; target node id
10: temp p id of the newly created vertex
11**:** **else**
12: insert graph id; target node id to already created vertex
13: if the id of the vertex is not already in temp p, insert it into temp p

```
            end if
         end if
         p = temp p
    end for
end while
```

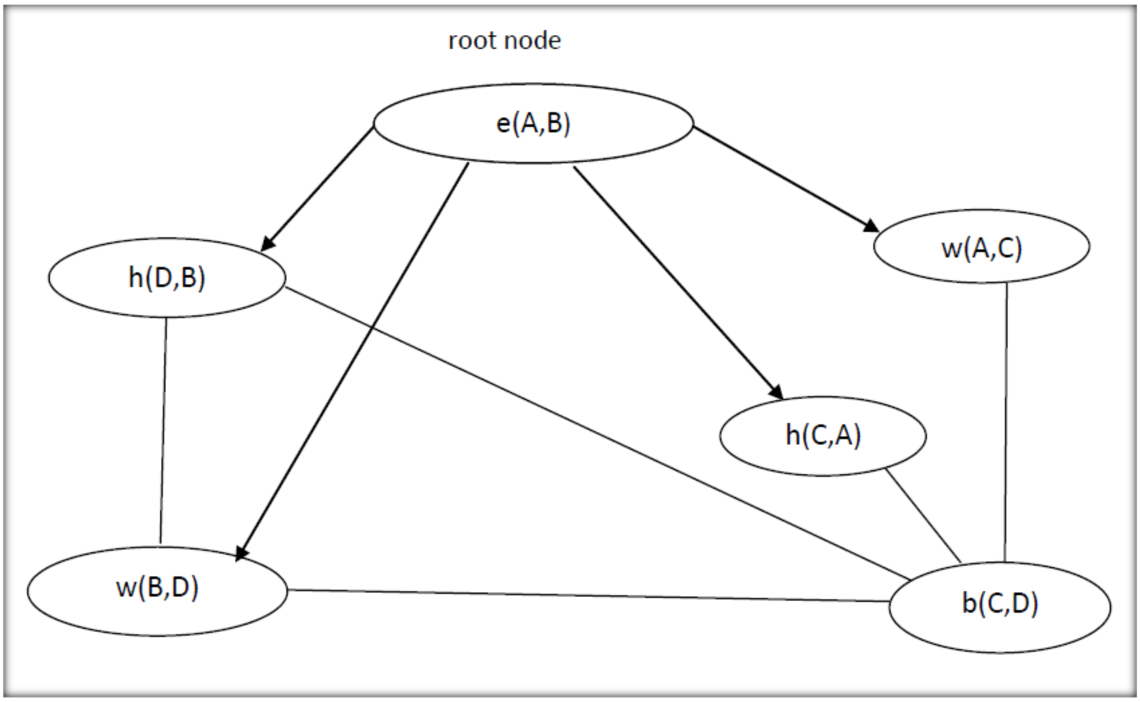

Figure 3.5: Summary Graph

- **Rule extraction**: In this step the summary graph is traversed in a depth-first manner. The depth of the search is limited by the maximum rule length parameter, which the system inputs. As an example the path that goes from edges e#A#B!h#C#A, h#C#B!h#D#C, h#C#C!w#D#A forms a concept rule which is e(A,B):- h(C,A), b(D,C),w(D,A).

- **Evaluation and pruning**: Paths extracted in the previous step are translated into SQL queries to calculate support and confidence values of the rules. Those rules that have support and confidence values higher than the threshold values value are accepted as solution clauses. Please note that this algorithm runs a two-step pruning mechanism, first is during the construction of the summary graph, i.e. pruning based on the frequency of the similar nodes, and the pruning performed at this step**.**

- **Covering**: In this step target instances explained by the solution clauses are marked as covered. If the number of the remaining uncovered target instances is below minimum support times number of uncovered target instances the concept induction process terminates, else nodes in the summary graph that contain tuples related to the covered target instances are updated as to not include those <graph id, node id> pairs. And the summary graph is again traversed for possible new solution clauses.

3.3.2 Time Complexity Analysis

The construction of the individual graphs is linear to the number of instances in the data set. Construction of the summary graph is O(nbd), where n is the number of individual graphs, b is the branching factor of the nodes, and d is the maximum rule length. Complexity of rule extraction from summary graph is O(bd), where b and d are as defined as above.

CHAPTER 4 EXPERIMENTAL RESULTS

A set of experiments are conducted to test the performance of G-CDS in comparison with C^2D and Hybrid graph based concept discovery method on well-known data sets in terms of coverage and predictive accuracy.

The experiments were conducted on Middle East Technical University Computer Engineering Department's (METU Ceng) high performance computing facility NAR [58]. NAR has 46 computational nodes (HP ProLiant BL460c) with total of 368 cores, 736 GB of memory and 6,5 TB harddisk space.

The proposed system is implemented in C++. MySQL version 5.0.95 [59] is used as data store and management tool. To implemet the method in a graph structure Boost.MPI library version 1.37.0 [60] is used.

Performance of G-CDS is compared to other systems in terms of coverage, accuracy, running time, number of induced concept descriptors and lengths of the induced concept descriptors.

1) Coverage: Coverage donates the number of the test data that is covered by induced hypothesis over the number of all taget instances. It is similar to support in C^2D.

2) Accuracy: Accuracy donates the fraction of of the sum of correctly covered true positive and true negative target instances over the total number of true positive, true negative, false positive and false negative target instances. It is similar to confidence in C^2D.

3) Running time: Running time includes both time needed for graph construction and rule generation.

4) Number of induced concept descriptors: This indicates the number of concept descriptors discovered.

5) Rule length: Number of literals in the body of the concept descriptor.

The experiments are conducted on data sets that belong to different learning problems, e.g. learning concept descriptors in highly connected data sets, learning in recursive concept descriptors, learning transitive rules.

4.1. LEARNING IN HIGHLY CONNECTED DATASETS

We firstly analyzed the performance of the proposed method on datasets that map into almost fully connected graphs. For this purpose, we employed the Elti and the Dunur datasets. Each dataset contains 9 target relations and 224 background instances. Result are shown in Table 4.1 for elti data set and shown in Table 4.3 for dunur data set.

Table 4.1: The Experimental Results for Elti Data Set

Elti Data Set	C^2D	Hybrid G-CDM	G-CDS
Accuracy	1	1	1
Coverage	1	1	1
Time(seconds)	110	0,51	7

The experimental results show that G-CDS discovered all of the rules that describe the target relations. G-CDS outperformed C^2D in a sense that it did not output permutations of the concept descriptors.

Table 4.2: Concept Descriptors Induced by C^2D and G-CDS

The rules induced by C^2D
R1: elti(A, B)⟵husband(C,A), husband(D, B), brother(C, D). R2: elti(A, B)⟵husband(C, A), husband(D, B), brother(D, C). R3: elti(A, B)⟵husband(C, A), wife(B, D), brother(C, D). R4: elti(A, B)⟵husband(C, A), wife(B, D), brother(D, C). R5: elti(A, B)⟵husband(C, B), wife(A, D), brother(C, D). R6: elti(A, B)⟵husband(C, B), wife(A, D), brother(D, C). R7: elti(A, B)⟵wife(A, C), wife(B, D), brother(C, D). R8: elti(A, B)⟵wife(A, C), wife(B, D), brother(D, C).
The rules induced by G-CDS
R1: elti(A, B)⟵husband(C,A), brother(C, D), husband(D, B). R2: elti(A, B)⟵husband(C, A), brother(C, D), wife(B, D). R3: elti(A, B)⟵wife(A, C), brother(C, D), husband(D, B). R4: elti(A, B)⟵ wife(A, C), brother(C, D), wife(B, D).

Table 4.3: The Experimental Results for Dunur Data Set

Elti Data Set	C^2D	Hybrid G-CDM	G-CDS
Accuracy	1	1	1
Coverage	1	1	1
Time(seconds)	9	0,11	8

G-CDS discovered exactly the same concept descriptors C^2D and Hybrid G-CDM. When compared to C^2D, the proposed system has a slightly short running time. Hybrid G-CDM has significantly shorter running time compared to the proposed system. We believe this is due to the fact that Hybrid G-CDM builds a single graph and discovers the rules while constructing the graph.

In Table 4.4 we compare the proposed system to RBPL and Hybrid G-CDM in terms of number of concept descriptors discovered and average concept descriptor lenghth. As the experimental results show, G-CDS discovers concept descriptors with similar properties when compared to Hybrid G-CDM. RBPL discoveres less number of concept descriptors which are longer than those discovered by G-CDS and Hybrid G-CDM. Longer rules are hard to interpret and are subject to overfitting problem. This sugegsts that the proposed method discovered better rules than RBPL.

Table 4.4: The Experimental Results for Family Data Set

Target relation	System	Num. of Clauses	Average Length	Time(seconds)
Wife	RBPL	1	3	NA
	Hybrid G-CDM	1	2	NA
	G-CDS	1	3	10,7
Aunt	RBPL	2	6	NA
	Hybrid G-CDM	10	2	NA
	G-CDS	10	3	10,8
Father	RBPL	2	6	NA
	Hybrid G-CDM	6	2	NA
	G-CDS	6	3	14
Brother	RBPL	2	6	NA
	Hybrid G-CDM	2	2,5	NA
	G-CDS	2	2,5	11

Table 4.4: The Experimental Results for Family Data Set (Continuation)

Target relation	System	Num. of Clauses	Average Length	Time(seconds)
Daughter	RBPL	2	6	NA
	Hybrid G-CDM	3	2	NA
	G-CDS	4	2,5	25
Husband	RBPL	1	3	NA
	Hybrid G-CDM	1	2	NA
	G-CDS	3	3	7,49
Mother	RBPL	2	6	NA
	Hybrid G-CDM	6	2	NA
	G-CDS	6	3	25
Sister	RBPL	2	6	NA
	Hybrid G-CDM	4	2,7	NA
	G-CDS	4	2,5	21
Uncle	RBPL	2	6	NA
	Hybrid G-CDM	9	2	NA
	G-CDS	9	2	21
Son	RBPL	2	6	NA
	Hybrid G-CDM	3	2	NA
	G-CDS	4	2,5	26

4.2. LEARNING RECURSIVE RULES

Recursive concept descriptors are those that contain the target relation in the body of the concept descriptor. In order to evaluate the performance of the proposed method we conducted experiments on the same generation data set. In the data set, 344 pairs of actual family members are given as positive examples of same-generation (sg) relation. Additionally, 64 background facts are provided to describe the parental (p) relationships in the family. We set the support threshold as 0.3, confidence threshold as 0.6 and maximum depth as 3.

G-CDS finds the following clauses (similar to C^2D) for this data set:

sg(A, B) ⟵ p(C, A), sg(C, D), p(D, B).

sg(A, B) ⟵ p(C, A), sg(C, D), p(D, A) .

sg(A, B) ⟵ p(C, A), p(C, B).

Table 4.5: The Experimental Results for Same-Gen Data Set

Elti Data Set	C^2D	Hybrid G-CDM	G-CDS
Accuracy	1	1	1
Coverage	NA	0,84	0,84
Time(seconds)	24	0,76	8

4.3. CONSTRUCTING TRANSITIVE RULES UNDER THE EXISTENCE OF INDIRECTLY RELATED FACTS

Michalski's trains problem [32] is a typical case in which the most background facts are indirectly related to target instances. In this data set, the target relation eastbound (train) is only related with has_car(train, car) relation. The other background relations have an argument of type car and are only related with has_car relation. In the data set there are 12 target relations and 196 background facts.

The eastbound data set shown in Table 4.6 is used in the experiments. The eastbound relation has 5 records which are (east1, east2, east3, east4, east5). The target relation has one parameter and its type is train. One of the background relations (has_car) has only related column type and facts. The other background relations are not related.

By adding indirectly related facts in the discovery process, C^2D finds the following rule for this data set:

eastbound(A)⟵has car(A, B), closed(B) (s=5/5, c=5/7).

Table 4.6: The Eastbound Data Set

Table	Arguments
eastbound	train
has_car	train,car
short	car
closed	car
long	car
open_car	car
double	car
jagged	car
shape_car	car,shape
load	car,shape,number
wheel	car,number

The best rule for this data set is actually different than what C^2D found. Because of the pruning mechanisms of C^2D, it is unable to find the best rule. However, G-CDS generates five graphs and the summary graph and by defining maximum rule length as three, it finds the following rule:

eastbound(A) ⟵ has car(A, B),closed(B), short(B) (s=5/5, c=5/5).

eastbound(A) ⟵ has car(A, B), double(B) (s=5/5, c=2/3).

eastbound(A) ⟵ has_car(A,B),has_car(A,C),closed(C) (s = 5/5 c = 5/7)

eastbound(A) ⟵ has_car(A,B),load(B,C,D),closed(B) (s = 5/5 c = 5/7)

The rules are discovered in 29 seconds.

CHAPTER 5 CONCLUSIONS AND FUTURE WORK

Multi-relational data mining approaches has become popular in computer science due to increase in the use of relational data and due to limitations of classical data mining algorithms working on a structured data. Various systems were developed having different characteristics like search direction, search heuristic, search technique etc. Depending on the input data, every system has both advantages and disadvantages. In this study, aspects of multi-relational data mining are examined and discussed and a graph based concept discovery system is introduced. The aim is to improve the results produced by well known ILP based concept discovery system C^2D and graph based concept discovery system, in particular path finding based, RPBL on well known data sets. As an outcome, a hybrid graph based concept discovery system namely G-CDS is developed.

Concept discovery is concerned with inducing logical definitions of a relation, called the target relation, in terms of other relations provided, called background knowledge. The concept discovery problem has been investigated under Inductive Logic Programming research in depth and systems evaluated on several data sets that belong to different learning problems. Although these systems showed important achievements they are subject to long running times and several problems such as local plateau problem.

Graph-based concept disvovery syatems are introduced particularly to solve the local plateau problem of ILP-based systems. Due to the graph structure, graph-based approaches also outperformed ILP-based ones in terms of running time. The success behind graph based systems is due to the deeply studied algorithms that can be adoptted for concept discovery problem. Also the transformation between relational database and graph is an easy and well formulated task.

In this work we present a hybrid graph-based concept discovery for data stored in a multi-relational databases. The method is hybrid, as it is similar to substructure-based approaches it is creates a summary graph by grouping similar nodes, and is similar to path-finding methods as it extracts concept descriptors by traversing the summary graph. The experimental results show that the proposed method is capable of inducing correct concept descriptors for datasets that belong to different learning problems.

When compared to ILP-based systems, the proposed method has a shorter running time with higher coverage and accuracy values. When compared to state-of-the-art graph based concept discovery system RBPL, the proposed method discoveres more easily interpretable concept descriptors in comparable running time.

As a future work, we plan to further investigate the performance of G-CDS on n-ary datasets such as Mutagenesis [21] and PTE [36]. The proposed approach is well suited for parallel execution, hence another future direction is parallelizing the proposed method.

REFERENCES

[1] Džeroski, S. (2003). Multi-Relational Data Mining: An Introduction. *ACM SIGKDD Explorations Newsletter*, *5*(1), 1-16.

[2] Muggleton, S. (1999). Inductive Logic Programming: Issues, Results and the Challenge of Learning Language in Logic. *Artificial Intelligence*, *114*(1), 283-296.

[3] Huchard, M., Hacene, M. R., Roume, C., & Valtchev, P. (2007). Relational Concept Discovery in Structured Datasets. *Annals of Mathematics and Artificial Intelligence*, *49*(1-4), 39-76.

[4] Quinlan, J. R. (1990). Learning Logical Definitions from Relations. *Machine Learning*, *5*(3), 239-266.

[5] Dehaspe, L., & Toivonen, H. (2001). Discovery of Relational Association Rules. In *Relational Data Mining* (pp. 189-212). Springer Berlin Heidelberg.

[6] Alphonse, É., & Osmani, A. (2008). On The Connection Between the Phase Transition of the Covering Test and the Learning Success Rate in ILP. *Machine Learning*, *70*(2-3), 135-150.

[7] Mutlu, A., & Karagoz, P. (2013). A Hybrid Graph-Based Method for Concept Rule Discovery. In *Data Warehousing and Knowledge Discovery* (pp. 327-338). Springer Berlin Heidelberg.

[8] Gao, Z., Zhang, Z., & Huang, Z. (2009, March). Learning Relations by Path Finding and Simultaneous Covering. In *Computer Science and Information Engineering, 2009 WRI World Congress on* (Vol. 5, pp. 539-543). IEEE.

[9] Kavurucu, Y., Senkul, P., & Toroslu, I. H. (2009). ILP-Based Concept Discovery in Multi-Relational Data Mining. *Expert Systems with Applications*, *36*(9), 11418-11428.

[10] Kavurucu, Y., Senkul, P., & Toroslu, I. H. (2010). Concept Discovery on Relational Databases: New Techniques for Search Space Pruning and Rule Quality Improvement. *Knowledge-Based Systems*, *23*(8), 743-756.

[11] Kavurucu, Y., Senkul, P., & Toroslu, I. H. (2008). Confidence-Based Concept Discovery in Multi-Relational Data Mining. In *Proceedings of the International MultiConference of Engineers and Computer Scientists* (Vol. 1).

[12] Frawley, W. J., Piatetsky-Shapiro, G., & Matheus, C. J. (1992). Knowledge Discovery in Databases: An Overview. *AI magazine*, *13*(3), 57.

[13] Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. *AI magazine*, *17*(3), 37.

[14] Otero, R., & Tamaddoni-Nezhad, A. (1992). *Inductive Logic Programming* (Vol. 168). S. Muggleton (Ed.). London: Academic Press.

[15] Lavrac, N., & Dzeroski, S. (1994). Inductive Logic Programming. In *WLP* (pp. 146-160).

[16] Lavrač, N., Džeroski, S., & Grobelnik, M. (1991, January). Learning Nonrecursive Definitions of Relations with LINUS. In *Machine Learning—EWSL-91* (pp. 265-281). Springer Berlin Heidelberg.

[17] Cestnik, B., Kononenko, I., & Bratko, I. (1987, May). ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users. In *EWSL* (pp. 31-45).

[18] Clark, P., & Boswell, R. (1991, January). Rule Induction with CN2: Some Recent Improvements. In *Machine learning—EWSL-91* (pp. 151-163). Springer Berlin Heidelberg.

[19] Mozetic, I. (1985). NEWGEM: Program for Learning from Examples. *Technical Documentation and User's Guide. Reports of Intelligent Systems Group UIUCDCS-F-85-949, Department of Computer Science, University of Illinois. Urbana Champaign, IL*.

[20] Clark, P., & Niblett, T. (1989). The CN2 Induction Algorithm. *Machine learning*,*3*(4), 261-283.

[21] Muggleton, S. (1995). Inverse Entailment and Progol. *New generation computing*, *13*(3-4), 245-286.

[22] Michalsky, R. S., Mozetic, I., Hong, J., & Lavrac, N. The Multi-Purpose Incremental Learning System AQ15. *Proc. 5thNCAI*, *1041*, 1045.

[23] Aha, D. W. (1992). Relating Relational Learning Algorithms. *Inductive Logic Programming*, *38*, 233-254.

[24] De Raedt, L., & Bruynooghe, M. (1989, December). Constructive Induction by Analogy. In *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 476-477). Morgan Kaufmann Publishers Inc.

[25] Shapiro, E. Y. (1983). Algorithm Program Debugging.

[26] Pazzani, M., & Kibler, D. (1992). The Utility of Knowledge in Inductive Learning.*Machine learning*, *9*(1), 57-94.

[27] Kietz, J. U., & Wrobel, S. (1992). Controlling the Complexity of Learning in Logic Through Syntactic and Task-Oriented Models. In *Inductive logic programming*.

[28] Wrobel, S. (1988). Automatic Representation Adjustment in an Observational Discovery System. In *EWSL* (pp. 253-262).

[29] Sammut, C., & Banerji, R. B. (1986). Learning Concepts by Asking Questions. *Machine learning: An Artificial Intelligence Approach*, *2*, 167-192.

[30] Muggleton, S., & Buntine, W. (1992). Machine Invention of First-Order Predicates by Inverting Resolution. In *Proceedings of the Fifth International Conference on Machine Learning* (pp. 339-352).

[31] Muggleton, S. (1987, August). Duce, An Oracle-based Approach to Constructive Induction. In *IJCAI* (pp. 287-292).

[32] Larson, J., & Michalski, R. S. (1977). Inductive Inference of VL Decision Rules. *ACM SIGART Bulletin*, (63), 38-44.

[33] Muggleton, S., & Feng, C. (1990). *Efficient Induction of Logic Programs*. Turing Institute.

[34] Leiva, H. A., Atramentov, A., & Honavar, V. (2002). A Multi-Relational Decision Tree Learning Algorithm. *Deparment of Computer Science, Iowa State University*.

[35] Giannotti, F., Manco, G., & Wijsen, J. (2004). Logical Languages for Data Mining. In *Logics for Emerging Applications of Databases* (pp. 325-361). Springer Berlin Heidelberg.

[36] Srinivasan, A., King, R. D., Muggleton, S. H., & Sternberg, M. J. (1997, August). The Predictive Toxicology Evaluation Challenge. In *IJCAI (1)*(pp.4-9).

[37] Muggleton, S. (1991). Inductive Logic Programming. *New Generation*

Computing,*8*(4), 295-318.

[38] Dehaspe, L., & Toivonen, H. (1999). Discovery of Frequent Datalog Patterns.*Data Mining and Knowledge Discovery*, *3*(1), 7-36.

[39] De Raedt, L., Blockeel, H., Dehaspe, L., & Van Laer, W. (2001). Three Companions for Data Mining in First Order Logic. In *Relational data mining* (pp. 105-139). Springer Berlin Heidelberg.

[40] Dehaspe, L., & De Raedt, L. (1997). Mining Association Rules in Multiple Relations. In *Inductive Logic Programming* (pp. 125-132). Springer Berlin Heidelberg.

[41] Gonzalez, J. A., Holder, L. B., & Cook, D. J. (2000). Graph Based Concept Learning. *AAAI/IAAI*, *1072*.

[42] Yoshida, K. I., & Motoda, H. (1995). CLIP: Concept Learning From Inference Patterns. *Artificial Intelligence*, *75*(1), 63-92.

[43] Richards, B. L. (1992). Learning Relations by Bathfinding.

[44] Washio, T., & Motoda, H. (2003). State of the Art of Graph-Based Data Mining. *Acm Sigkdd Explorations Newsletter*, *5*(1), 59-68.

[45] Gholami, M., & Salajegheh, A. (2012). A Survey on Algorithms of Mining Frequent Subgraphs. *International Journal of Engineering Inventions*, *1*(5), 60-63.

[46] Cook, D. J., & Holder, L. B. (1994). Substructure Discovery Using Minimum Description Length and Background Knowledge. *Journal of Artificial Intelligence Research*, 231-255.

[47] Cook, D.J.; Holder, L.B. (2000).Graph-Based Data Mining, *IEEE Intelligent System*, (pp. 32-41).

[48] Gonzalez, J., Holder, L. B., & Cook, D. J. (2001, September). Application of Graph-Based Concept Learning to the Predictive Toxicology Domain. In *Proceedings of the Predictive Toxicology Challenge Workshop*.

[49] Matsuda, T., Motoda, H., & Washio, T. (2002). Graph-Based Induction and its Applications. *Advanced Engineering Informatics*, *16*(2), 135-143.

[50] Gao, Z., Zhang, Z., & Huang, Z. (2009, April). Extensions to the Relational Paths Based Learning Approach RPBL. In *2009 First Asian Conference on Intelligent Information and Database Systems* (pp. 214-219). IEEE.

[51] Toprak, S. D. (2005). *A New Hybrid Multi-Relational Data Mining Technique* (Doctoral dissertation, Middle East Technical University).

[52] Kavurucu, Y. (2009). An ILP-Based Concept Discovery System for Multi-Relational Data Mining.

[53] Agrawal, R., Imieliński, T., & Swami, A. (1993, June). Mining Association Rules Between Sets of Items in Large Databases. In *ACM SIGMOD Record* (Vol. 22, No. 2, pp. 207-216). ACM.

[54] Agrawal, R., & Srikant, R. (1994, September). Fast Algorithms for Mining Association Rules. In *Proc. 20th Int. Conf. very Large Data Bases, VLDB* (Vol. 1215, pp. 487-499).

[55] Han, J., Pei, J., & Yin, Y. (2000, May). Mining Frequent Patterns without Candidate Generation. In *ACM SIGMOD Record* (Vol. 29, No. 2, pp. 1-12). ACM.

[56] Sarawagi, S., Thomas, S., & Agrawal, R. (1998). *Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications* (Vol. 27, No. 2, pp. 343-354). ACM.

[57] Dunham, M. H. (2006). *Data Mining: Introductory and Advanced Topics*. Pearson Education India.

[58] URL, http://www.ceng.metu.edu.tr/hpc/index/.[Last Accessed June, 2015]

[59] URL, http://dev.mysql.com/doc/refman/5.0/en/index.html. [Last Accessed April, 2015]

[60] URL, http://www.boost.org/users/history/version_1_37_0.html. [Last Accessed December, 2014]

[61] Ong, I. M., de Castro Dutra, I., Page, D., & Costa, V. S. (2005). Mode Directed Path Finding. In *Machine Learning: ECML 2005* (pp. 673-681). Springer Berlin Heidelberg.