IMPROVING THE EFFICIENCY OF ILP-BASED AND GRAPH-BASED CONCEPT DISCOVERY SYSTEMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

NAZMIYE CEREN ABAY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

DECEMBER 2014

Approval of the thesis:

IMPROVING THE EFFICIENCY OF ILP-BASED AND GRAPH-BASED CONCEPT DISCOVERY SYSTEMS

submitted by **NAZMIYE CEREN ABAY** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Adnan Yazıcı
Head of Department, **Computer Engineering** _____

Assoc. Prof. Dr. Pınar Karagöz
Supervisor, **Computer Engineering Department, METU** _____

Examining Committee Members:

Prof. Dr. İsmail HakkıToroslu
Computer Engineering Department, METU _____

Assoc. Prof. Dr. Pınar Karagöz
Computer Engineering Department, METU _____

Prof. Dr. Ahmet Coşar
Computer Engineering Department, METU _____

Assoc. Prof. Halit Oğuztüzün
Computer Engineering Department, METU _____

Assist. Prof. Dr. Alev Mutlu
Computer Engineering Department, Kocaeli University _____

Date: *10/12/2014*

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.

Name, Last Name: NAZMIYE CEREN ABAY

Signature :

ABSTRACT

IMPROVING THE EFFICIENCY OF GRAPH-BASED CONCEPT DISCOVERY SYSTEMS

Abay, Nazmiye Ceren
MS., Department of Computer Engineering
Supervisor : Assoc. Prof. Dr. Pınar Karagöz

December 2014, 83 pages

Concept discovery is a process for finding hidden relations from the given set of experiences named as background knowledge [27]. Concept discovery problems are investigated under Inductive Logic Programming (ILP)-based approaches and graph-based approaches [28]. Although ILP-based systems dominate the area, these systems have some problems such as local maxima and local plateaus [15]. Recently, graph based system becomes more popular due to its flexible structure, clear representation of data and ability of overcoming problems of ILP-based systems.

Graph based approaches can be classified into two parts defined as structure-based approaches and path-finding approaches according to their methods they use for discovering concepts.

The proposed approach can be classified as a combination of both path-finding approaches and methods of association rule mining. It finds paths between the arguments of target instances for concept discovery from the given graph. In addition

to path-finding from given graph, association rule mining techniques are used for pruning based on support and confidence. The proposed method is different from the other path-finding approaches because association rule mining techniques are used for reducing search space and finding frequent and strong concept definitions.

Keywords: Concept Discovery, Graph, Path, Support, Confidence

ÖZ

ÇİZGE TABANLI KAVRAM KEŞFİ SİSTEMLERİNİN VERİMLİLİĞİNİN ARTIRILMASI

Abay, Nazmiye Ceren
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Pınar Karagöz

Aralık 2014, 83 sayfa

Bir kavramın keşfi o kavramla ilgili arka planda tanımlanmış ilişkilerin üzerinden kavramın tanımlanmasıyla yapılır. Genel olarak kavram keşiflerini bulma problemlerinde Tümevaran Mantık Programlama (TMP) ve çizge tabanlı yöntemler kullanılır. Tümevaran Mantık Programlama yöntemleri kavram keşif problemlerinde yoğun olarak kullanılsa da bu sistemlerde yerel maxima ve yerel plato problemleri içermektedir. Sonuç olarak son zamanlarda çizge tabanlı yöntemler esnek yaklaşımlarından, açık veri gösterimlerinden ve Tümevaran Mantık Programlama problemlerini ortadan kaldırdığından dolayı popülerlik kazanmıştır.

Çizge tabanlı yöntemler genel olarak kavram keşfi yaparken izledikleri yöntemlere göre ortak bileşen bulma ve yol bulma sistemleri olarak ikiye ayrılırlar.

Bu çalışmada uygulanan yöntemler yol bulma ve bağlantılı kural madenciligi yöntemlerinin birleştirilmesidir. Önerilen yöntem, kavram keşfinde verilen çizgede hedef örneğinin parametreleri arasındaki tüm yolları bulur. Verilen çizgedeki tüm

yolları bulmaya ek olarak, bağlantılı kural madenciligi tekniklerinden kapsam ve doğruluk tekniklerine bağlı olarak eleme yöntemleri kullanılır. Önerilen yöntem diğer yol bulma yöntemlerinden farklıdır çünkü bağlantılı kural madenciligi yöntemleri kavram araması yapılan alanların azaltılmasında ve yaygın ve güçlü kavram tanımları bulunmasında kullanılır.

Anahtar Kelimeler: Kavram Keşfi, Çizge, Yol, Kapsam, Doğruluk

To My Parent

ACKNOWLEDGMENTS

I express my sincere appreciation to my supervisor, Assoc. Prof. Pınar Karagöz, for her guidance, insight, encouragement, motivation and positive attitudes throughout the study. She always supports me for both researching and the other issues in life. I am thankful for having the chance of working with her in my dissertation.

I would like to thank Professor İsmail Hakkı Toroslu, Professor Ahmet Coşar, Assoc. Prof. Halit Oğuztüzün for accepting to be members of my dissertation committee. Their valuable feedback was of great value to finalize this work.

I wish to express a lot of thanks to Assist. Prof. Dr. Alev Mutlu, for his valuable suggestions and comments at crucial points in the development of the thesis. He always supports my work even in different cities.

I wish to express a special thanks to Dr. Dilek Türkoğlu for her invaluable patience, kindness and encouragement. Without her moral support, I never gather my strength for completing my dissertation. She reforms my perspective, idea and attitudes towards life patiently.

I would like to thank to my family, for their significant love, support and patience. They always prepare best environment for me to complete my thesis.

And finally, I thank my friends, Ceyda, Tuğba, Irem, Barış, Yusuf, Ali, Osman, Doruk and Sait for their helps and supports to complete my master degree.

TABLE OF CONTENTS

ABSTRACT…….…....v
ÖZ……..….vii
ACKNOWLEDGMENTS…….…...x
TABLE OFCONTENTS…….…….xi
LIST OF TABLES…….....xiii
CHAPTERS
1 INTRODUCTION 1
2 BACKGROUND 3
2.1.Inductive Logic Programming (ILP) 3
2.2.Studies of ILP-based Concept Discovery Systems 5
2.2.1.ALEPH 5
2.2.2.GOLEM 6
2.2.3.WARMR 6
2.3.Studies on Graph-based Concept Discovery 7
2.3.1.Subdue 7
2.3.2.SubdueCL 8
2.3.3.Relational Path-Finding 9
2.3.4.Hybrid Graph-based Method for Concept Rule Discovery 10
2.3.5.Concept Rule Induction System (CRIS) 12
2.3.5.1.Basic Techniques Used in CRIS 12
2.4.Association Analysis 16
2.4.1.Association Rule 16
2.4.2.Apriori Method 18
2.4.2.1.Computational Complexity 18
2.5.Neo4j 19
2.6.WEKA 21
3 GRAPH DATABASE FOR GRAPH BASED CONCEPT DISCOVERY 25

3.1. The Proposed Method 25
3.2. Running Example 33
3.2.1.Concept Discovery on Relation-Based Datasets 33
3.2.2.Concept Discovery on Attribute-Based Datasets 42
3.2.3.Preprocessing Constants 42
4 EXPERIMENTAL RESULTS 57
4.1.Data sets 57
4.2.Experimental Results 59
4.2.1.Time Comparison for Relation-based Datasets 59
4.2.2.Time Comparison for Attribute-based Datasets 61
4.3.Concept Discovery on Relation-based Datasets 62
4.3.1.Concept Discovery on Same-Generation Data Set 62
4.3.2.Concept Discovery on Elti Data Set 64
4.3.3.Concept Discovery on Dunur Data Set 66
4.4.Concept Discovery on Attribute-based Datasets 67
4.4.1.Concept Discovery on Eastbound Data Set 67
4.4.2.Concept Discovery on Mutagenesis Data Set 68
4.4.3.Concept Discovery on PTE-1 Data Set 70
4.4.4.Concept Discovery on Mesh Data Set 72
5 CONCLUSION AND FUTURE WORK 75
REFERENCES 77

LIST OF TABLES

TABLES

Table 1: Same-Generation data set 13
Table 2: The pseudo code of algorithm of the proposed method 28
Table 3: Nominator of Confidence Query 32
Table 4: Denominator of Confidence Query 32
Table 5: Target instances of relation-based datasets 34
Table 6: Concept Instances and type declarations for *elti* dataset 35
Table 7: Background facts for *elti (cemile, ayse)* 35
Table 8: Candidate Solution Clauses and their frequencies in *elti* graph 37
Table 9: Candidate Solution Clauses of *elti (cemile, ayse)* 39
Table 10: Candidate Solution Clauses and Frequencies of *elti* after Renaming 40
Table 11: Parameters for evaluation and pruning on *elti* graph 40
Table 12: Candidate solution clauses and their support values of *elti* graph 41
Table 13: Candidate solution clauses and their confidence values of *elti* graph 41
Table 14: Charge ranges and differences classified by J48 Pruned Decision Tree 45
Table 15: Division of charge values based on support threshold 46
Table 16: SQL template query for alphabetical constant 46
Table 17: SQL query for alphabetical constant 47
Table 18: Candidate solution clauses from preprocessing step 47
Table 19: Example of relations for attribute-based datasets 48
Table 20: SQL template query for infrequent table 48
Table 21: SQL query for infrequent table 49
Table 22: Candidate solution clauses and their frequencies 54
Table 23: Parameters for evaluation and pruning on *muta* graph 55
Table 24: Candidate solution clauses and their support values of *muta* graph 56
Table 25: Summary of the benchmark dataset used in the experiments 58

Table 26: Properties and experimental settings for relation-based dataset..... 58
Table 27: Evaluation parameters for data sets 59
Table 28: Graph properties of relation-based datasets 59
Table 29: Graph construction time for relation-based datasets 59
Table 30: Graph construction time for attribute-based datasets 61
Table 31: Graph properties of attribute-based datasets 62
Table 32: Graph properties of attribute-based datasets 62
Table 33: Coverage and Accuracy Results for *same-gen* Dataset 63
Table 34: Running Time for *same-gen* Dataset 63
Table 35: Coverage and Accuracy Results for *elti* dataset 65
Table 36: Running Time for *elti* dataset 65
Table 37: Coverage and Accuracy Results for *dunur* dataset..... 66
Table 38: Running Time for *dunur* dataset..... 67
Table 39: Coverage and Accuracy Results for *eastbound* dataset 68
Table 40: Running Time for *eastbound* dataset 68
Table 41: Coverage and Accuracy Results for *muta* dataset 69
Table 42: Running Time for *muta* dataset 70
Table 43: Coverage and Accuracy Results for *PTE-1* dataset..... 72
Table 44: Running Time for *PTE-1* dataset..... 72
Table 45: Mesh Concept Discovery Rules and f-metric Rules..... 73
Table 46: Coverage and Accuracy Results for *mesh* dataset..... 73
Table 47: Running Time for *mesh* dataset..... 74

LIST OF FIGURES

FIGURES

Figure 1: Neo4j Web Interface 20
Figure 2: WEKA Explorer 22
Figure 3: The flowchart of the proposed method 27
Figure 4: Example for conceptual graph of Sowa 29
Figure 5: Dataset Types used for concept discovery process 33
Figure 6: Subgraph of *elti* dataset related with *cemile* and *ayse* 36
Figure 7: Paths that contain one relation *elti* graph 38
Figure 8: Paths that contains loop in *elti* graph 38
Figure 9: WEKA Explorer for J48 Pruned Tree 43
Figure 10: WEKA J48 Pruned Tree Visualizer 44
Figure 11: Connected graph of *muta* dataset based on mutagenicity of drugs 50
Figure 12: Sub-graph of *muta* graph for *d5* drug 51
Figure 13: Creating unique nodes for preventing confusion 52
Figure 14: Non-mutagenesis drugs of *muta* dataset 53
Figure 15: Construction time versus total nodes and relations for relation-based 60
Figure 16: Construction time versus total nodes and relations for attribute-based 61

CHAPTER 1

INTRODUCTION

It is known that computer is an inevitable part of our daily life and it is extensively used in every age of people. The data that is stored in computer becomes more complex and bigger in real life and relational databases are used for storing data in multiple tables instead of single table. At the same time, this situation emphasizes multi relational data mining, for analyzing and extracting information on multiple related tables directly because interpreting these huge databases manually is impractical. Learning from intractably large search space is a central topic for multi relational data mining because extraction of previously unknown and potentially useful information from data is crucial issue for science of today. For such learning systems, ILP proposes inducting of first order predicate logic from experiences in the form of Prolog Logic Program [2]. ILP is used commonly for multi relational data mining in various kinds of domains of real world [11, 27, 7, 12, 4, 13, 14, 15, 16, 23, 24, 25, 29, 30, 31, 32, 33, 34, 35, 36].

ILP dominates multi relational data mining because it is accurate and it outputs understandable expressions which are easily makes sense for domain experts [1]. However, it also gives negative side effects because ILP uses resources heavily and it has long execution time [39]. Moreover, ILP is inappropriate some kind of domains that uses single table [47]. Others frequent problems that ILP encounters are the local maxima and plateaus which affect the quality of learning [15]. To cope with these obstacles, graph-based data mining is introduced. It is stated that graphs represents

concepts into first order predicate logic by using Conceptual Graphs [3]. Multi relational is divided by two parts in terms of logic based approaches and graph-based-approaches. Graph-based approaches are different from logic based approaches. The main difference is the representation of data. Graph representation is powerful tool for concept discovery because it is flexible and understandable representation for data.

In this dissertation, we will propose new ILP-based approach to overcome the efficiency problem of ILP. This new method finds all paths between instances of target relations. And it uses confidence based pruning defined in CRIS for finding best and frequent concepts that cover its concept instances.

This dissertation is divided into 6 chapters. Chapter 1 introduces inductive logic programming (ILP) and proposes new ILP-based method. Chapter 2 presents the related works of well-known ILP-based methods. Chapter 3 describes the proposed method for graph based concept discovery. Chapter 4 shows the experimental results of the proposed method. Experimental results are compared with Concept Rule Induction System (CRIS), Hybrid Graph-based Method for Concept Rule Discovery and other state of the art studies. Chapter 5 concludes dissertation and possible improvements.

CHAPTER 2

BACKGROUND

This section is intended to introduce concept of the inductive logic programming (ILP) and concept discovery methods. Firstly, we give definition and main algorithm of ILP, and then we explain two types of concept discovery systems namely, ILP-based concept discovery systems and graph-based concept discovery systems.

2.1. Inductive Logic Programming (ILP)

Inductive Logic Learning is defined as an intersection of both logic programming and inductive learning [4]. It also takes approaches of machine learning and logic programming. It induces hypothesis that from examples and background knowledge in the form of first order predicate logic (FOPC) [2]. Using logic in Inductive Logic Programming is beneficial in many aspects. ILP formulates the knowledge in the form of FOPC which makes easily understood for domain experts. Moreover, machine learning programming manipulates logic programs in a regular way [4].

B: Background knowledge in logic form

E^+: Positive examples

E^- : Negative examples

E: Finite set of examples which consists of positive evidence (E^+) and negative evidence (E^-), $E = E^+ \cup E^-$

H: Hypothesis which is discovered from examples (E) and background knowledge (B) such that $H \in L$ and H are consistent with I.

L: Language Specification that describes the hypothesis space.
I: An optional set of constraints on acceptable hypotheses (H).

Mainly, ILP system takes input in terms of logic form of positive examples (E^+) and negative examples (E^-) of the concept to be discovered with the background information [5]. ILP system tries to induce hypothesis (H) that covers all positive instances and none of the negative instances. It means hypothesis is a complete and consistent with background knowledge and examples.

Inductive Logic Programming is defined with the following formulations [4] :

Prior Satisfiability: $B \nvDash E^-$

Posterior Satisfiability: $B \wedge H \nvDash E^-$ (Consistency)

Prior Necessity: $B \nvDash E^+$

Posterior Sufficiency: $B \wedge H \vDash E^+$ (Completeness)

Posterior Sufficiency is described as *completeness* with regard to positive evidence and Posterior Satisfiability is named as *consistency* with the negative evidence [39].

The consistency condition is flexible and sometimes the hypothesis is allowed to be inconsistent by covering some negative evidences. However, the experiment states that expected error of a learner when learning from only positive examples is close value when compared to a learner that learns from both positive and negative examples [6]. Although covering the negative evidences is tolerable in large data, relational databases that the proposed method works on only cover positive examples. This means that proposed method will only work on positive instances.

Most ILP-based systems use the sequential covering algorithm that depends on finding one rule and extract the data it covers, and then iterates same process. However, in large datasets, search space may be very huge and ILP-based systems may not return hypotheses in a reasonable time. To avoid the dependency of the amount of search space, ILP-based systems are commonly uses hill-climbing

heuristic methods in order to avoid the combined explosion search space from large datasets [16]. However, hill-climbing heuristic methods encounter locality problems such as local maxima and local plateaus [48]. Researches has been still conducted to increase the efficiency of ILP-based systems reducing the sequential execution time such as reducing the number of hypothesis, efficiently testing candidate hypothesis or parallelization of ILP-based systems [50].

2.2. Studies of ILP-based Concept Discovery Systems

ILP search strategies are mainly separated into two parts namely top-down (general-to-specific) search and bottom-up (specific-to-general) search while constructing concept discovery rules [9].

Bottom up algorithm is mainly based on inverting resolution conversely and with the use of background knowledge, examples are iteratively generated by applying refinement operations. Conversely, top-down method searches hypothesis space for inducing concept rules from the most general clause to most specific clause. In this search, initial hypothesis is more general hypotheses. In later steps, this general hypothesis is specialized through the use of refinement operator in order to be consistent [1].

2.2.1. ALEPH

A Learning Engine for Proposing Hypotheses (ALEPH) is a top-down ILP system [12]. It is written in Prolog principally. The algorithm of ALEPH is similar to Progol, but ALEPH is more flexible in searching hypothesis space, evaluation and refinement. ALEPH takes minimum support, minimum confidence, search strategies, evaluation functions, and refinement operator as parameters.

ALEPH has been applied to a different type of real-world problems. ALEPH shows remarkable results in construction of structure-activity relations, mutagenic and carcinogenic activity in biological search. ALEPH follows four main steps in its simple algorithm.

Step 1: An example is selected for generalization step. If there is no example for selection program is ended. Otherwise, it applies **Step 2**.

Step 2: In this step, the most specific clause that covers the selected example is generated. This most specific clause is named as bottom clause and it is a definite clause that contains many literals [11].

Step 3: In this step, the more general clause than the bottom clause is searched in the bottom clause that has the "best" score.

Step 4: In "cover removal" step, the current theory is expanded by the clause with the best score. All examples made redundant are removed. Then, return to Step 1.

2.2.2. GOLEM

Golem is bottom up ILP-based concept discovery method. In intractability of large search space, inducing the accurate first order logic programs are difficult. And Golem is based on *Relative Least General Generalization* (rllg) to avoid intractable large search space. By Plotkin's *Relative Least General Generalization* [10], Golem limits the hypothesis search space. The hypothesis is constructed from only positive examples which make Golem both insufficient and incorrect [8, 9]. It uses negative examples for reducing the literals of body. Golem is tested on the satellite problem [52], mesh analyses [51] and structures activity prediction for drugs [53].

2.2.3. WARMR

WARMR is a general purpose data mining tool for frequent pattern discovery in relational databases [13]. It is used different kind of domains in terms of telecommunication [21] and carcinogenesis of chemicals [13] that answers some efficiency problems of ILP. WARMR is a level-wise approach based on Apriori algorithm [37]. It consists of two main steps for finding the frequent pattern namely, candidate generation and candidate evaluation. In candidate generation, most general pattern is selected. If this pattern is infrequent, it will be pruned. In candidate evaluation, frequencies of candidate patterns are computed according database.

2.3. Studies on Graph-based Concept Discovery

Graph-based concept discovery systems are mainly classified as substructure-based approaches and path finding-based approaches according to the applied methods in concept discovery process.

Concept discovery systems classified as substructure-based approaches search iteratively similar substructures in graph. These similar substructures are replaced and presents as a single node in a graph. The process continues until all substructures are considered or the total amount of computation exceeds a given limit.

The systems that fall into the second group search finite length paths that connect arguments of the positive target instances and output these paths as concept descriptors.

2.3.1. Subdue

Subdue is general tool and can be applied to several domains represented as graph [14]. Subdue is tested in many domains such as earthquake activity [42], circuit analysis [43] and chemical toxicology domain [44]. It uses Minimum Encoding as a model evaluation method for minimizing entire data set [45]. And it uses constrained beam search as a search technique for finding best substructure. This approach does not encounter computational complexity because it uses greedy search strategies. However, it may miss some important patterns [49].

Subdue promises identification of common substructure in a relational data is important process for interpreting data. These common substructures can compress data in structural concepts. This process simplifies representation of data that makes easier the discovering interesting patterns. It is a crucial process because it provides to find of the cause of chemical cancers by obtaining related SARs despite the diversity among the compounds [54].

Subdue system takes input of the graphical representation of chemical compounds with the help of domain knowledge. By using this graph, Subdue system discovers the best substructure that is repetitive in data. In subdue algorithm, substructure begins with a vertex. Then it expands substructure until considering all possible substructures or a computation limit. Best substructure is found after multiple iterations by expanding the previously found substructures. After finding best substructure, it represents data in terms of structural concepts by compressing data based on the best substructures.

2.3.2. SubdueCL

Subdue concept learner (SubdueCL) is a graph based relational concept learner system [41]. It is an extension to the Subdue system. Subdue only works in on positive examples while SubdueCL learns from both negative and positive examples. Main algorithm and search strategies of SubdueCL are based on the Subdue but the learning process is different than Subdue. Subdue uses a graph compression approach but Subdue uses a set of covering approach by differentiating positive and negative examples. Due to negative examples, Subdue want to cover all positive examples but very little negative examples as soon as possible. For this aim, Subdue uses an evaluation formula to give a value to all generated substructures [14].

The negative examples covered by substructure are considered as errors.

$$\text{Value} = 1 - \text{Error}$$

By this formula, it tests covered positive examples and negative examples by substructure. The substructure takes higher score by covering positive examples while it takes lower score by covering negative examples.

$$\text{Error} = \frac{\#\text{PosEgsNotCovered} + \#\text{NegEgsCovered}}{\#\text{PosEgs} + \#\text{NegEgs}}$$

SubdueCL is a parametric function. It takes positive examples G_p, the negative examples G_n, the beam length and limit on the number of substructures to include in its search. Until all positive examples are covered, it repetitively searches for best substructures. If the best structure cannot be produced, it extends the search space. SubdueCL is tested on various types of domains [55, 56, 57].

2.3.3. Relational Path-Finding

Relational path-finding is a new approach in first order learning systems. It is proposed to avoid locality problems such as local maxima and local plateaus [15]. It is based on an assumption that in relational domains, there will be always fixed-length path of relations that satisfies target concepts.

Relational path-finding supposes that the relational domain can be represented as graph of constants and these constants are connected by relations with respect to content of relational domains if these constants interact with each others. Also, it supposes important concepts should be represented by a fixed-length of paths consisting definite number of relations.

Relational path-finding arbitrarily selects a positive example and uses it to find initial rule. According to constants of selected positive example, it finds paths by expanding to nodes associated with constants. Relational path-finding states that these constants are linked each other by relations. And each constant forms a sub-graph. Sub-graphs among these constants are isolated to others. Examining each sub-graph, they are expanded to other nodes. Each node is examined according to finding intersection between these sub-graphs. If the intersection is found there exists the fixed-length path between constants and the rule is instantiated. If the intersection is not found, expanding is continued by adding relations and nodes to sub-graphs. If the depth limit exceeds or no intersection is found, the rule will be rejected.

Relational path-finding searches relational paths in structured instance space instead of a hypothesis space to learn first order relations [38].

2.3.4. Hybrid Graph-based Method for Concept Rule Discovery

A Hybrid Graph-based is a new method that combines both substructure-based approach and path finding approach [23]. It has similar features of both these approaches but it has differences.

A Hybrid Graph-based has different than substructure based approach because the graph is not compressed according to substructure. But similar to substructure based approach, it groups similar arguments by a single node.

A Hybrid Graph-based is different from path finding because it does not look for paths for finding concept descriptors. It takes paths while constructing graph from relational format.

In Hybrid Graph-based method, relational format is taken and the graph is constructed in the light of relational format. Then, it extracts concept definitions and output concept descriptors in the form of first order predicate logic.

A Hybrid Graph-based is a parametric method that takes a set of target instances, a set of background data, minimum support, minimum confidence and maximum rule length as parameters. This approach contains 7 steps.

1. **Initialization:** In this step, two nodes are creating named as source and target nodes. A disconnected graph is constructed with these nodes. One node holds a first argument of the concept instances. And other node holds the second argument of concept instances.

2. **Expansion:** According to background knowledge, graph is expanded by adding nodes and relations to constants. In this step, the process is searching relations that have oncoming edges for avoiding loops. For two nodes, isolated sub-graphs that only include the relations and nodes connected to constants are constructed.

3. **Merge:** In this step, the nodes that have some arguments of concept instances are merged and edges are synchronized according to merge operations.

4. **Evaluation and Pruning:** In evaluation and Pruning step, current descriptors are evaluated according to minimum support and minimum confidence values. If they are below the threshold they are pruned. Otherwise, expanding adding nodes and relations to constants is continued. This calculation is applied in relational domain by SQL queries.

5. **Check for Intersection:** In this step, intersection of two sub-graphs is searched. The intersection means that there exists a path between the arguments of the concept instances. If the intersection is found, candidate concept rule is found and it passes to other step. If no intersection is found and the current path exceeds the value of [maximum_depth / 2], the candidate concept rule is rejected and the process is stopped.

6. **Update Path Variables:** After intersection is found and candidate concept rules are defined, variables of these rules are updated to be consistent with the content of nodes. After arranging candidate concept rules, it will be tested with respect to minimum support and confidence value. If their frequency is below the threshold value, they are pruned. Also, the candidate solution set is ignored if it is found previously.

7. **Covering:** In this step, the ratio of covered positive examples is calculated and they are marked as covered. If uncovered positive instances are below the value of minimum support x #target instances, induction process is stopped, otherwise the process is restarted.

Experimental results are tested by A Hybrid Graph-based results and they show that it is promising method when compared to other ILP-based methods in terms of efficiency and accuracy.

2.3.5. Concept Rule Induction System (CRIS)

Concept Rule Induction System is a predictive concept learning ILP system which uses relational association rule mining techniques for finding strong and frequent concept definitions with respect to target relations and background knowledge [7].

In descriptive learning, there is a target concept to be discovered from background knowledge. With techniques of associative rule mining, CRIS induces association rules which consist of only target concepts as the head relations. Background knowledge appears only in the body clauses.

2.3.5.1. Basic Techniques Used in CRIS

CRIS induces a set of concept rules having the target relation in the head clause. While inducing this theory, it uses 3 basic strategies for pruning techniques.

Strategy 1: In CRIS, candidate concept rules are induced according to the definition of θ-subsumption. θ-subsumption is defined as following:

A definite clause *C,* θ-subsumes a definite clause C' if and only if such that:

$$head\ (C) = head\ (C')\ and\ body\ body(C) \subseteq body\ (C')$$

Assume that CRIS induces concept rules of database of the example of *same_gen* with type declarations. In this example, *same_gen* is the concept to be learned, and three concept instances are given. Background facts with respect to one relation namely parent is given. And types of the attributes of relations are provided.

Table 1: Same-Generation data set

Concept Instances	**Background Facts**	**Type Declarations**
same_gen(mert, kubra) same_gen(mert, dilek) same_gen(kubra, erdem) same_gen(mert, erdem) same_gen(erdem, dilek) same_gen(dilek, erdem)	parent(mert, kubra) parent(mert, dilek) parent(kubra, erdem)	parent(person, person) same_gen(person, person)

Consider that CRIS induces two concept rules from the *same_gen* example

$$C_1: same\_gen\ (A, B) \leftarrow parent\ (A, B)$$
$$C_2: same\_gen\ (A, B) \leftarrow parent\ (A, B), same\_gen\ (A, C)$$

In this example, it is seen that head of C_1 and C_2 are the same and C_1 is more general than C_2 because the body of C_1 is a subset of C_2. C_1 θ-subsumes C_2 because it is consistent with definition of θ-subsumes.

Strategy 2: In CRIS, non-promising rules are pruned for saving calculation for infrequent clauses. This strategy states that confidence value of *parent* is lower than the specialized rules.

It is shown in the example of *same_gen*, two rules of are considered for union because they have same head literals and they have difference of one literal in body clause.

$$C_1: same\_gen\ (A, B) \leftarrow parent\ (A, C)\ (c=0.4)$$
$$C_2: same\_gen\ (A, B) \leftarrow parent(C, B)\ (c=0.5)$$

The unification of C_1 and C_2:

C_3: *same_gen (A, B) ← parent (A, C), parent(C, B) (c=0.3)*

C_4: *same_gen (A, B) ← parent(C, B), parent (A, C) (c=0.6)*

C_3 is pruning because it has a lower confidence value of both C_1 and C_2. However, C_4 is not pruned because its confidence value is higher of both C_1 and C_2.

Strategy 3: If concept rule includes foreign key constraint between the head and body clause, the primary key literal of head clause and foreign key constraint of body clause will be same in generalization step.

To illustrate that in the example of *same_gen*, consider *C* as a candidate concept rule:

Assume that data holds foreign key constraint between head and body clause such that primary id of *same_gen* presented by literal A and is connected to *parent* by foreign key constraint presented by A illustrate below by C_1 clause.

C_1: *same_gen (A, B) ← parent (A, C)*

In generalization step, CRIS cannot generate rule such that

C_2: *same_gen (A, B) ← parent (D, C).*

CRIS works on relational databases which contains target relation and background facts. CRIS has a parametric function which takes minimum support (*min_sup*), minimum confidence (*min_conf*) and maximum rule depth (*max_depth*) as parameters for inducing hypothesis for concept discovery.

Algorithm of CRIS includes four main parts namely, generalization, specialization, evaluation and coverage.

1. Generalization: In generalization step, most general concept rules are induced with a head and single predicate body clause with the guide of Strategy 3

with respect to considering all target instances. After constructing general concept rules, search space is analyzed by the Apriori-based approaches. Strategy 1 and Strategy 2 are used for specialization and infrequent and weak candidate concept rules are pruned.

When the maximum depth which is defined as a parameter is reached, according to confidence values, candidate concept rules are eliminated. In this way, strong and best fit rules are selected with respect to given data.

In generalization step, the important point is to determine an argument of a head of an induced rule is a variable or a constant. To overcome this issue some queries are executed in SQL statement to find frequencies of an argument.

Queries are executed to find appearance of an argument to compare the value of *min_sup * number_of_uncovered_instances*. An argument is determined as a constant if it is appearance is higher than *min_sup * number_of_uncovered_instances* which means it is frequent value. If the appearance of an argument is lower than this value it is determined as a variable in the head of candidate concept rules.

In some datasets, for numeric attributes, it is feasible to give ranges on constants according to support thresholds.

2. Specialization: CRIS specializes the general concept rules by Apriori-based specialization operator by searching space from general to specific defined as top down approaches. In this step, concept descriptors of length k are combined to every other concept descriptors for refining concept descriptor of length k+1.

3. Evaluation and Filtering: After specialization step, CRIS eliminates rules whose confidence values are below the confidence threshold. The remaining strong rules are examined again, for finding the best concept rule. The best concept rule is selected according to hypothesis evaluation criteria that are defined in the f-metric definition (adapted from f-score formula [46]). f-metric emphasizes importance of

support or confidence value with respect to changing value of B. If B is defined as greater than 1, the effect of confidence will be higher. Otherwise, if B is defined as less than 1, then support will be emphasized.

$$f\text{-}metric = \frac{((B2 + 1)x\ conf\ x\ supp)}{((B\ x\ conf) + supp}$$

4. Coverage: After the best rule is selected, CRIS detects target instances that are covered by induced concept rule. Each concept instances that is covered by induced rules are removed. This loop continues until all concept instances are covered and no more candidate rules may be generated for the uncovered concept instances.

CRIS is an efficient ILP-based system. Its performance is tested on different types of domains in terms of coverage and accuracy.

2.4. Association Analysis

Association analysis is a method which is commonly used in many areas for discovering remarkable relationships which are underhanded in large data sets. These newly discovered relationships are named as association rules or sets of frequent items. In this process, interesting and strong connections are revealed.

Association analysis may reveal two problems. One problem is about the cost of computation. Discovering patterns in a large datasets may be very expensive and generally it may not return patterns in a reasonable time. Second problem is about the accuracy. In some datasets, association analyses may not promise to obtain the real representation of datasets, some rules may reveal randomly.

2.4.1. Association Rule

An association rule may be expressed in the form of $X \rightarrow Y$, where X and Y are disjoint itemsets.

Support and confidence values measures the strength of an association rules. By support and confidence values, uninteresting rules may be eliminated which are potentially unrelated with the given dataset.

Support measures the frequency of a rule in a dataset and it is formulated as:

$$Support,\ s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

Support values are an important property for discovering association rule because it eliminates rules which have low support values. These rules that have low support values decrease the efficiency of discovering process. Because computation of these rules are needless and rules are eliminated. By eliminating these rules, general rules are emphasized which are real representation for dataset. Confidence tests the appearance of Y in transactions which holds X and it is formulated as:

$$Confidence,\ c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

Confidence value is also an important feature and shows the reliability of the inference made by a rule. This shows the strong of connections between items.

For discovering frequent pattern in a dataset, some restrictions are defined.

For a set of transactions defined as T, all rules are analyzed according to minimum support and minimum confidence values. Minimum support and minimum confidence values are defined as thresholds. For each rule support and confidence values are calculated and evaluated. The rules having support value and confidence value are below the thresholds are pruned and labeled as weak rules as defined support $\geq$ min_sup and confidence $\geq$ min_conf.

For brute-force approach, calculation of confidence and support values of every rule may be very expensive in large datasets because of the explosion of possible rules. Possible rules revealed from dataset that holds k item is calculated by $3^k - 2^{k+1} + 1$

which means exponential growth of rules. And it may be very large in most applications. Also, it is clear that many of computations contain uninteresting rules which will be needless. This affects the efficiency of a process and an approach previously pruning rules without computing all support and confidence values is useful.

2.4.2. Apriori Method

Apriori is an association rule mining method that promises to reduce the complexity of computation and the number of comparisons for candidate item-sets [18].

For avoiding these exhaustive computations, it is accepted that if the item-set is frequent, the candidate rules which is superset of this frequent item is frequent, too.

Conversely, if the item-set is infrequent, the candidate rules which is superset of this frequent item is infrequent and this rules are pruned without calculating their confidence values.

Apriori uses minimum support for pruning candidate rules. Support-based pruning is the pruning strategy that is used to reduce search space. It uses support measure for pruning. It also states that the support value for an item-set is never higher than the support value of its subsets. This is defined as anti-monotone feature of a support-based pruning.

f is anti-monotone if X is a subset of Y, then f(Y) must not exceed f(X).

$$\forall X, Y \in J : (X \subseteq Y) \rightarrow f(Y) \leq f(X)$$

2.4.2.1. Computational Complexity

Complexity of Apriori algorithm depends on different factors such as support threshold, the number of items, transactions and average transaction width.

If the support threshold is selected as lower value, then the number of frequent items will be higher. It means the increasing of complexity because generated candidate item-sets increases and calculating support value of these generated rules will be more exhaustive. Moreover, when maximum size of frequent item-sets increases, the algorithm repeats the steps for generation and pruning numerously.

If the number of items is high, more space will be needed to hold increased number of candidate item-sets. Also, it consumes space because more computation is needed as the generation of candidate item-sets increase.

Number of transaction is another factor that affects computational complexity. Apriori compares the frequency of candidate item-sets in transactions. If this number increases, the number of passes will be higher and the running time also increases.

The number of items that are stored in transaction is defined as transaction width. Average transaction width affects the complexity of the Apriori algorithm. Because maximum size of the frequent item-sets tends to increase which cause to generation of more candidate item-sets.

2.5. Neo4j

Neo4j is an open-source, highly scalable graph database which is supported by Neo Technology by implementing in Java language [5]. It has an interactive and friendly Web-Interface.

Neo4j graph database is based on graph theory and graph is constructed by nodes (vertexes) and relationships (edges) between them.

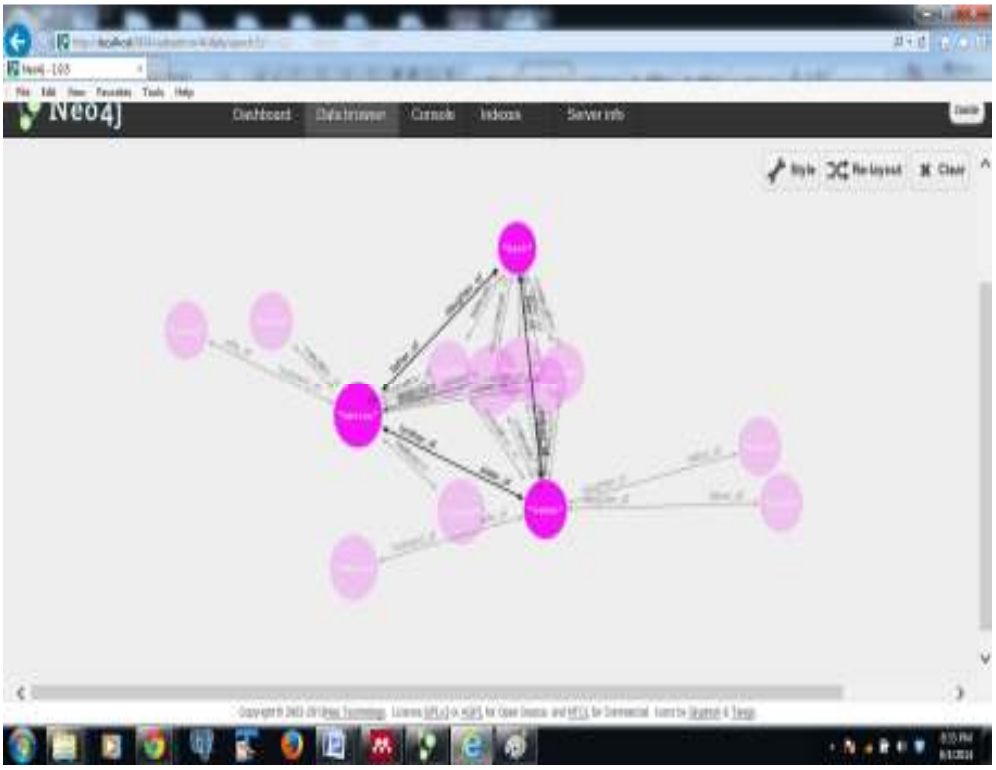

Figure 1: Neo4j Web Interface

Neo4j allows user to import data which is stored in the CSV format. This provides to import data from relational databases to neo4j graph databases. Neo4j graph databases uses two files for this conversion. One file stores node and its properties and the other file is for relationships and properties between nodes.

Neo4j can be integrated to your JVM processes by embedded Neo4j library jars in your builds. Neo4j is also offers object oriented approaches to your graph database by Node, Relationship and Path classes and implementations. Moreover, it promises high-speed traversals and graph algorithm such as shortest path, Dijkstra's algorithms.

In this thesis, neo4j is used for conversion of relational databases to graph databases. Also, it is used for querying for all paths between arguments of the target instances to induce concept rules.

2.6. WEKA

WEKA (Waikato Environment for Knowledge Analysis) is a machine learning tool that is developed by University of Waikato [20]. It is written in java and the machine learning algorithms can either be applied directly to a dataset or called from your own Java code. Moreover, it may access to databases using Java Database Connectivity and process the result returned by a database query.

WEKA includes tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.

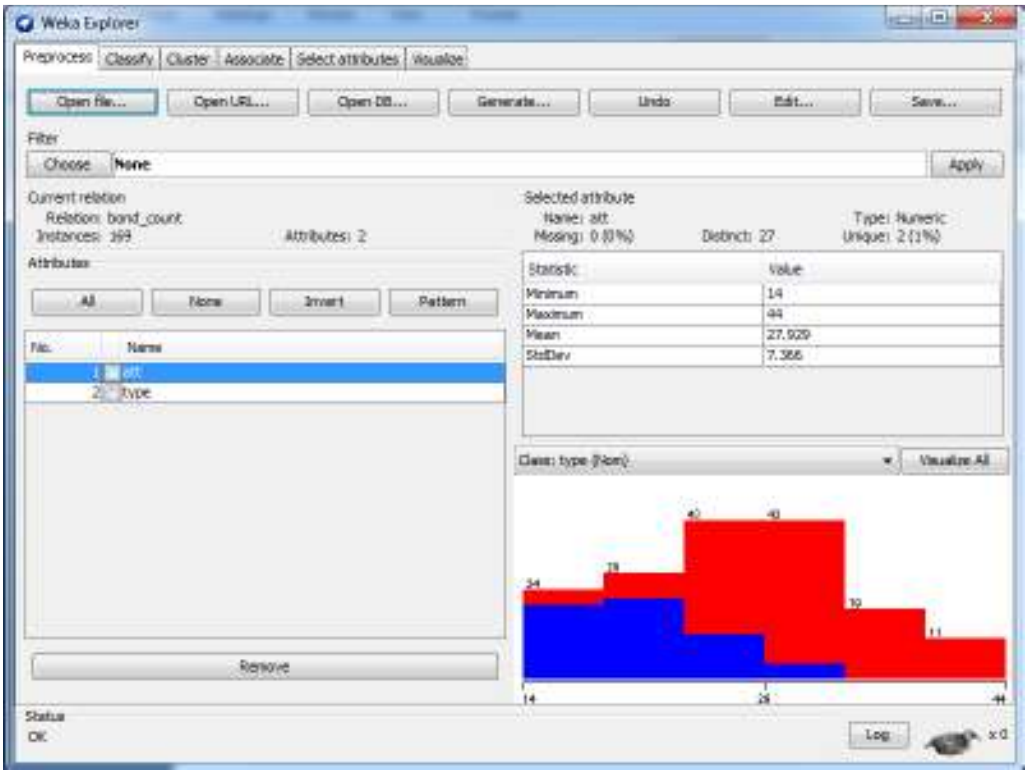

Figure 2: WEKA Explorer

User can manipulates learning algorithms from java code or from files such as .csv, .arttf files.

In this dissertation, WEKA is used for classifying of numeric attributes. For classifying, the parameters such as test options, percentage of split and cross validation can be given manually from the explorer of WEKA. It has many types of decision trees that help to classify values according to given attributes. Decision trees of *J48* and *Decision Stump* are commonly used for classifying of numeric attributes.

J48 is an open source Java implementation of the C4.5 algorithm in the WEKA. Ross Quinlan develops C4.5 algorithm used to generate a decision tree [21]. C4.5 constructs decision trees from a set of training data in the same way as ID3 algorithm using information entropy [22]. This decision tree is typically used in the machine learning and natural language processing domains.

A decision stump is a simpler decision tree that contains only a one-level decision tree where the split at the root level is based on a specific attribute/value pair [26]. It is machine learning model consists one root and one level leaves. Decision Stump performs best in terms of speed [40].

CHAPTER 3

GRAPH DATABASE FOR GRAPH BASED CONCEPT DISCOVERY

The system which is presented in this dissertation uses graphs for concept discovery. Knowledge can be represented in many ways and Sowa's conceptual graph proposes that knowledge can be represented as a labeled graph and can be transformed to first order predicate logic based on the existential graphs of Charles Sanders Peirce [17]. Sowa's conceptual graph is a logic based design used to represent conceptual schemas that are applied to database systems. Conceptual graphs are applied to a wide range of topics because all kinds of knowledge are labeled graphs which are intuitive and human readable.

In this dissertation, the proposed approach is based on the combination of path-finding approaches and Apriori-based pruning. It accepts the assumption that if two nodes are related, there must be finite length path that links these two nodes [15]. After finding paths, we follow the confidence and support based pruning approaches as CRIS offers for finding high quality concept rules.

3.1. The Proposed Method

The proposed method contains five steps. The proposed method takes a set of target instances, a set of background facts, minimum support, minimum confidence, and maximum rule length parameters. The target instances and background facts are initially stored in relational databases.

Before constructing graph, the datasets are preprocessed for constants or eliminating infrequent tables. After construction, the paths are queried from graph database and processed for increasing the quality of concept descriptors.

After finding all paths, the proposed approach concept prunes concept descriptors according to their support and confidence values. According to *f-metric* score the best rule is selected and declared as a solution for given dataset.

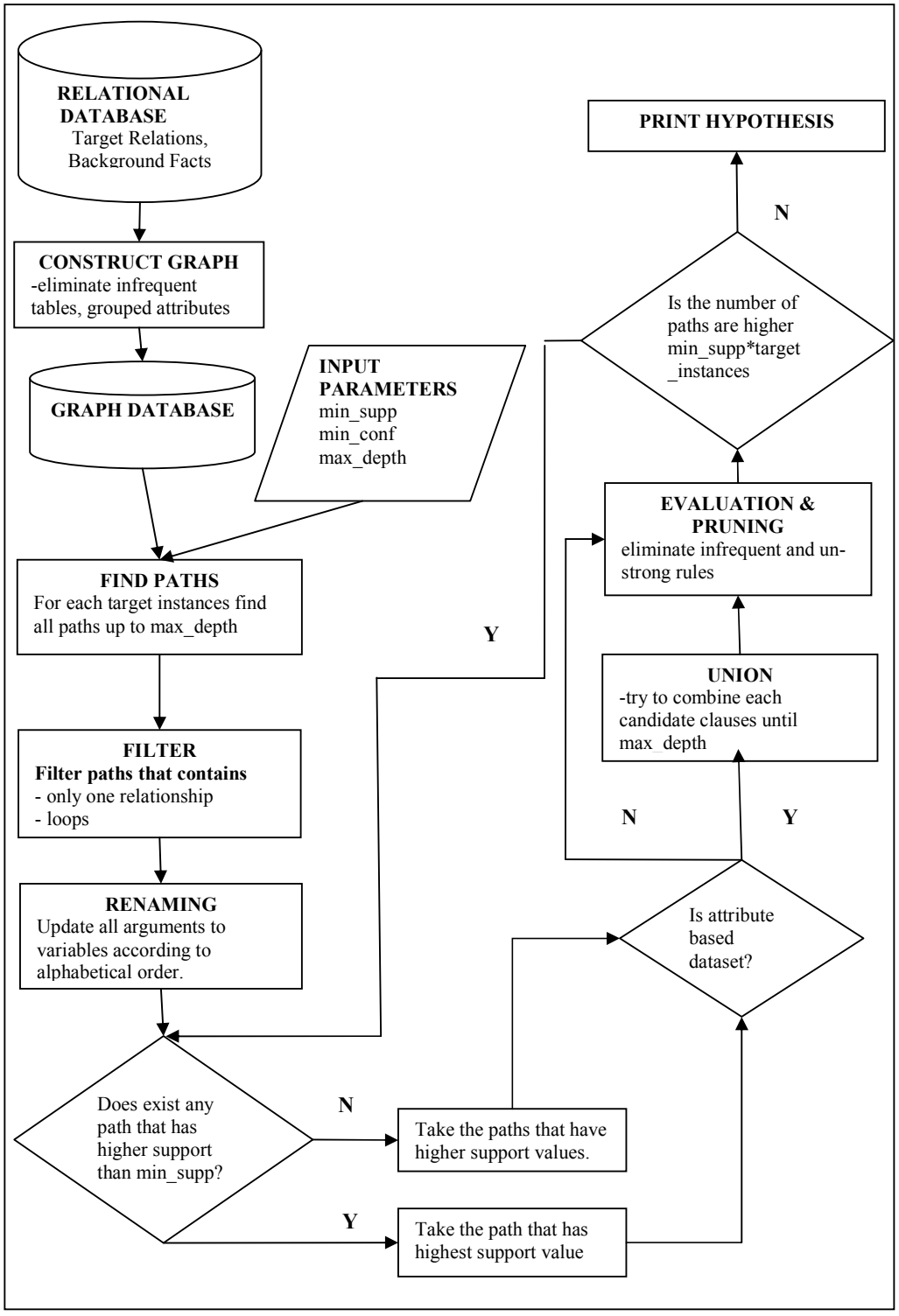

Figure 3: The flowchart of the proposed method

Table 2: The pseudo code of algorithm of the proposed method

```
Input: Graph Database G, Target Instances I
Output: Hypothesis Set H
Parameters: min_sup, min_conf, max_depth, B (used in f-metric),
            confidenceHashMap, supportHashMap
Local Variables: candidateSolutionClausesList, solutionSetList,
            totalNumberOfPaths, allPathList, processedPathList, eliminatedPathList

for each target instance in TargetInstanceSet
      allPathList=find all paths between arguments of targetInstance
      eliminatedPathList=eliminates pathList for paths contains loops and one
                  relations
     renamedPathList=replace arguments of eliminatedPathList
end for

if graph is attribute-based graph
      if graph contains numeric constants
            WEKAPathList = paths classified in WEKA
            add WEKAPathList to candidateSolutionClausesList
      end if
end if

while totalPathNumber > min_support * number of target instances
      for each path in renamedPathList
            pathSupport = frequencyOfPath/totalNumberOfPath
            if pathSupport > min_support
                  add path to candidateSolutionClausesList
            else
                  add path has highest pathSupport to candidateSolutionClausesList
            end if
      end for

      if graph is attribute-based graph
            unifiedList = union for all elements for candidateSolutionClausesList
            add unifiedList to candidateSolutionClausesList
      end if
       for each path in candidateSolutionSet
            if confidenceHashMap contains confidence of path
                  confidenceOfPath = calculateConfidence
            else
                  confidenceOfPath = confidenceHashMap.get(path)
            end if

            if confidenceOfPath > minimum_confidence
                add path to solutionSetList
            else
                remove from candidateSolutionClausesList
            end if
 end while
```

Table 2 (continued)

```
for each path in solutionSetList
    if supportHashMap contains support of path
        supportOfPath = calculateSupport from Relational Database
    else
        supportOfPath = supportHashMap.get(path)
    end if

    if supportOfPath <= minimum_confidence
        remove from solutionSetList
    end if
end for
```

1. Construction Graph: In this step, we use directed, labeled, connected and simple graph for representation of data set. In our data representation, the main idea is similar to Sowa's conceptual graphs.

Sowa's conceptual graph defines concepts as nodes. These nodes are linked by edge that represents relation. It is illustrated with an example used on Sowa's Conceptual Graph.

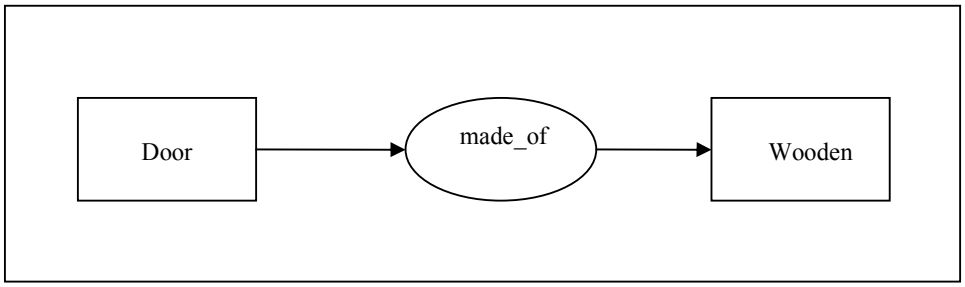

Figure 4: Example for conceptual graph of Sowa

In this graph, concepts are represented by square vertices and relation is represented by oval vertices between these concepts. It shows relation between nodes named as

Door and *Wooden*. They are connected by *made_of* relation. This graph describes expression of "A door is made_of Wooden".

Using conceptual graph of Sowa, each argument of relations is represented as a distinct node with a unique id in our data sets. And each relation is shown as an edge between nodes.

2. Find Paths: Neo4j has advanced search abilities for path finding in terms of speed. In this step, the proposed method uses Java implementation of path finder of Neo4j. For each target instances, the proposed method gives parameters for finding all paths between the arguments of the target instances.

3. Filter: In filter steps, the processing is same for both relation-based graph and attribute-based graph. All paths are investigated because of reducing candidate item-sets. By this way, we reduce the computation of infrequent and un-strong rules.

However, finding paths up to *max_depth* includes paths that have only one relationship. We eliminate them because one-length-path means that paths contain only target relation.

Moreover, all paths are analyzed again because it may traverse between two nodes in same relationship type that cause loop. The proposed method also eliminates paths that contain loop because they can loop up to *max_depth* without containing any other relations. Furthermore, they increase the total number of paths and they affect complexity of calculation. As a result, concept learning may be disrupted and the quality of rules is decreased. So the proposed method eliminates paths contains loops.

4. Rename Arguments with Variables or Intervals: In renaming step, each argument is handled by independently. Arguments are renamed by variables by alphabetical ordering. However, relations in concept learning may contain numeric constants. Numeric constants are renamed with different approaches because it is not feasible to seek acceptable constants for numeric constants distributed in a wide

range. So, the proposed method uses WEKA for these constants to classify them in feasible ranges.

5. Union: Union processing is only applied to attribute-based datasets. Because, each property is defined as separate path and these properties must be unified for refining the clauses. Conversely, it is not appropriate for relation-based datasets because they have finite-length path if two nodes are related and adding relation is useless.

The proposed method tries to combine each candidate clauses of length k to refine candidate clauses of length k+1 for clauses that contains same head literals. These newly unified clauses are accepted as a candidate solution clause and they are evaluated in Evaluation and Pruning step according to confidence values.

6. Evaluation and Pruning: Before this step, all paths are clarified. To find the strongest rules, Apriori-based pruning is applied to given paths defined in CRIS and Hybrid Graph-based Method.

In this step, each path is considered as candidate solution clauses. Each candidate solution clause is evaluated according to its support values. Support value is calculated according to total number of paths. For example, for a candidate solution clause defined as C_1 its support value is calculated as follows:

$$\text{Support} = \frac{Frequency\ of\ candidate\ clause}{Total\ number\ of\ paths}$$

After eliminating, confidence values are calculated from the databases as it is applied in CRIS and Hybrid Graph-based method. To illustrate, a query is given an example of *elti* dataset that calculates confidence values. For a candidate solution clause defined as C_1.

C_1: elti (A, B):-husband (B, A)

$$\text{confidence} = \frac{Count1}{Count2} \text{ where}$$

Table 3: Nominator of Confidence Query

Count1:
SELECT COUNT DISTINCT (h.arg1, h.arg2) FROM elti e, husband h WHERE e.arg1=h.arg2 AND e.arg2=h.arg1

Table 4: Denominator of Confidence Query

Count2:
SELECT COUNT DISTINCT (h.arg1, h.arg2) FROM husband h

For each iteration, the paths that have higher support values are selected and evaluated. If there exists no path that has higher support value, the path that has highest support value is selected and evaluated.

If confidence value of selected path is below the threshold, it will be pruned. If it has higher confidence value it partakes for next iteration. The proposed method runs until total number of paths are higher value than *target_instances* * *minimum_support_threshold*. If the process stops, it means total number of paths has lower value for generating candidate clauses. After elimination of infrequent candidate solution clauses, there may be more than one candidate solution clauses. The proposed method selects the best clause according to f-metric score [19].

$$f\text{-}metric = \frac{(B^2+1)x\ confidence\ x\ support}{(B\ x\ confidence)+support}$$

B can be changed for modify the effect of confidence or support.

3.2. Running Example

In this dissertation, structures of the datasets are different and different approaches are needed for concept discovery. We divided datasets into two parts according to our approaches that we follow in terms of attribute-based datasets and relation-based datasets.

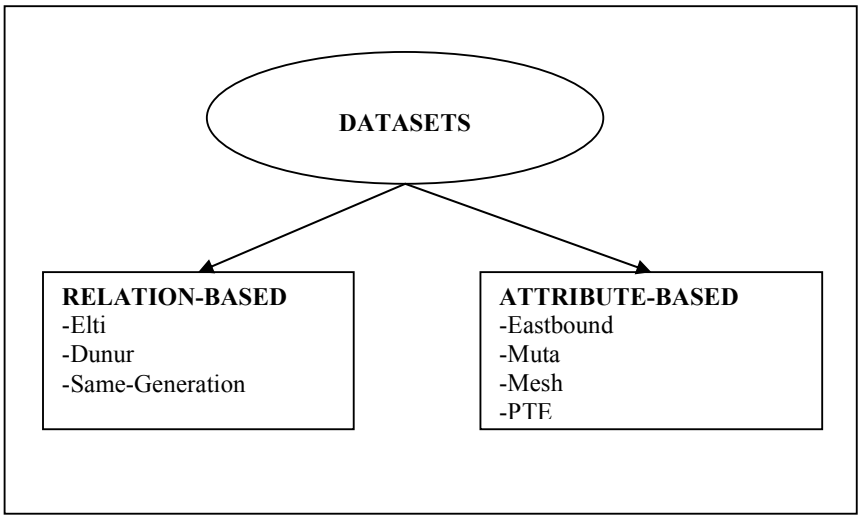

Figure 5: Dataset Types used for concept discovery process

According to these differences, graph representation and steps of the proposed method will be different. This section illustrates steps of the proposed method of concept discovery on both relation-based datasets and attribute-based datasets.

3.2.1. Concept Discovery on Relation-Based Datasets

elti dataset is classified as a relation-based dataset by the proposed method. We illustrate the steps of concept learning on *elti* dataset as a running example of concept discovery on relation-based datasets.

1. Relation-based Graph Construction: Datasets such as elti, dunur and same_gen contain binary relations between arguments of target concepts. It means that they contain finite length paths between the arguments of target relations. The proposed approach groups these datasets as relation-based datasets.

Table 5: Target instances of relation-based datasets

Target Instances of Relation-based Datasets
dunur(person, person)
elti(person, person)
same_gen(person, person)

There is no alphabetical or numerical constant in relation-based datasets. The proposed method can be applied directly to these datasets.

For constructing graph from relational database to graph database, two relational tables are created. First table represents all nodes and its properties. And each node is defined with a unique id for querying. Second file represents relations between nodes. This table contains nodes and relations between them. Although table can contain properties of both relations and nodes, in relation-based datasets it is not needed because nodes and relations do not have properties. For transferring these tables to graph databases, they are converted to CSV file and imported to Neo4j.

Type declarations and concept instances of *elti* dataset are given. All arguments are type of *person*.

Table 6: Concept Instances and type declarations for *elti* dataset

Concept Instances	Type Declaration
elti (cemile,ayse)	wife (person, person)
elti (cemile,ayten)	sister (person, person)
elti (ayse, cemile)	daughter (person, person)
elti (ayse, ayten)	husband (person, person)
elti (ayten, cemile)	mother (person, person)
elti (ayten, ayse)	father (person, person)
elti (nalan, bedriye)	brother (person, person)
elti (bedriye, nalan)	elti (person, person)

In *elti* dataset, we have 8 target instances and 224 background facts. From eight concept instances defined in *elti* dataset, each concept instance is analyzed independently. We pick first concept instance for simplicity, *elti (cemile, ayse).*

Table 7: Background facts for *elti (cemile, ayse)*

Concept Instance	Background Facts
elti (cemile, ayse)	wife (ayse, altan) sister (kubra, dilek) daughter (kubra, cemile) husband (altan, ayse) sister (dilek, kubra) mother (cemile, dilek) brother (altan, osman) husband (osman, cemile) daughter (dilek, cemile) father (osman, dilek) wife (cemile, osman) father (osman, kubra) daughter (dilek, osman) mother (cemile, kubra) daughter (kubra, osman)

The sub-graph of *elti* dataset is shown. All background facts related to arguments of target instances are used for constructing sub-graph.

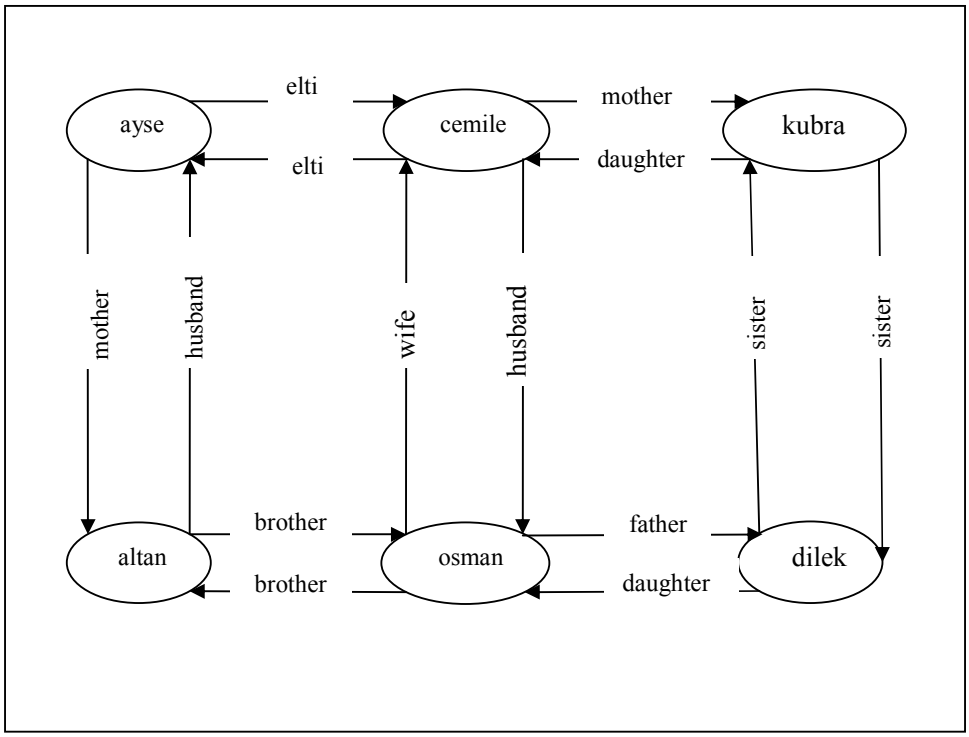

Figure 6: Subgraph of *elti* dataset related with *cemile* and *ayse*

2. Relation-based Path Finding: In relation-based path finding, path finding process is searched through between two nodes in a simple graph because arguments of target instances are directly related to each others.

The proposed method uses Neo4j path finder and it takes parameters as *max_depth* and id of two nodes. Path finding is iterated for every concept instances and all paths are stored for filter step to reduce complexity of calculation.

To make more understandable this step, the proposed method shows all paths between the concept instances of *elti* dataset.

Totally, 104 paths are found for concept learning on *elti* dataset. For simplicity, we will show only the paths between *cemile*, *ayse* for concept instance of *elti(cemile, ayse)*. For concept instance of *elti(cemile, ayse)*, we have 14 paths.

Table 8: Candidate Solution Clauses and their frequencies in *elti* graph

Candidate Solution Clauses	**Freq.**
elti(ayten,cemile);elti(ayten,ayse)	1
husband(osman,cemile);brother(altan,osman);wife(ayse,altan)	1
husband(osman,cemile);brother(osman,altan);wife(ayse,altan)	1
wife(cemile,osman);brother(altan,osman);wife(ayse,altan)	1
elti(ayse,cemile)	1
husband(osman,cemile);brother(osman,altan);husband(altan,ayse)	1
husband(osman,cemile);brother(altan,osman);husband(altan,ayse)	1
elti(cemile,ayten);elti(ayten,ayse)	1
elti(cemile,ayse)	1
elti(cemile,ayten);elti(ayse,ayten)	1
wife(cemile,osman);brother(osman,altan);wife(ayse,altan)	1
wife(cemile,osman);brother(osman,altan);husband(altan,ayse)	1
elti(ayten,cemile);elti(ayse,ayten)	1
wife(cemile,osman);brother(altan,osman);husband(altan,ayse)	1

3. Relation-based Filter Paths: In this step, we eliminate the paths that disrupt the rule quality.

We employ the *elti* data set as a running example to show the unrelated paths. *elt*i relation is the concept to be learned in this dataset.

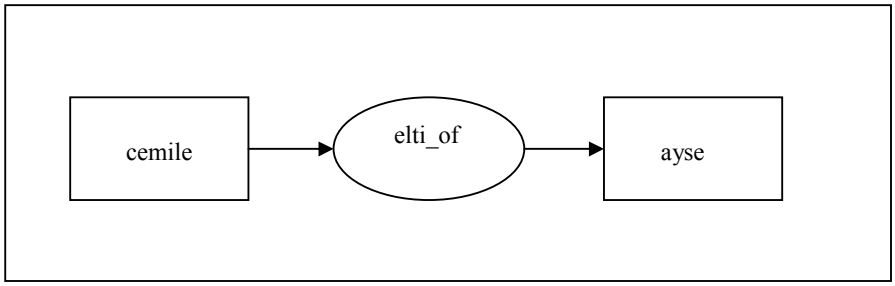

Figure 7: Paths that contain one relation *elti* graph

We eliminate the paths that contain one relation because it is same as the given concept instance and it gives no result for concept learning process.

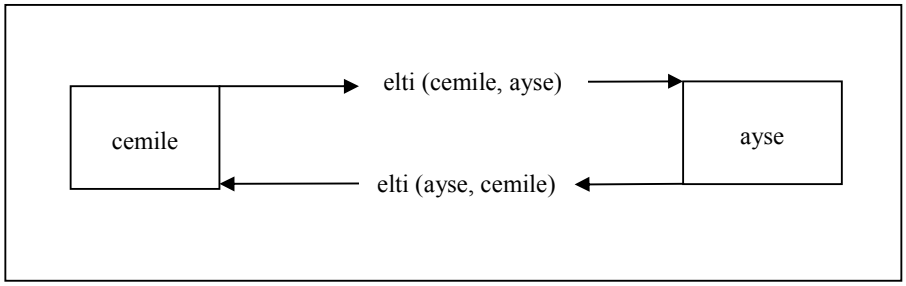

Figure 8: Paths that contains loop in *elti* graph

Also, the recursion is not allowed and the paths that contain concept relations will be also eliminated. As it is seen in the example, *'cemile is elti of ayse'* and at the same time *'ayse is elti of cemile'* and it contributes no result for concept learning.

After elimination of these paths we have 64 paths for *elti* dataset. We show the eliminated paths of *cemile*, *ayse* for concept instance of *elti (cemile, ayse)*. After elimination we have 8 paths for next step. We have eliminated 6 paths and we reduce the complexity of calculation.

Table 9: Candidate Solution Clauses of *elti (cemile, ayse)*

Candidate Solution Clauses
husband(osman,cemile);brother(altan,osman);wife(ayse,altan)
husband(osman,cemile);brother(osman,altan);wife(ayse,altan)
wife(cemile,osman);brother(altan,osman);wife(ayse,altan)
husband(osman,cemile);brother(osman,altan);husband(altan,ayse)
husband(osman,cemile);brother(altan,osman);husband(altan,ayse)
wife(cemile,osman);brother(osman,altan);wife(ayse,altan)
wife(cemile,osman);brother(osman,altan);husband(altan,ayse)
wife(cemile,osman);brother(altan,osman);husband(altan,ayse)

4. Rename Arguments with Variables or Intervals: In relational datasets, we have paths that contain only arguments and no constants are defined so renaming can be applied to paths directly.

Renaming is applied to all head and body clauses with unique variable. Beginning with the head clauses all arguments are replaced by alphabetical order until the end of clause and renaming is coherent for each path.

For illustration, one path is selected for concept instance of *elti (cemile, ayse)*.

C_1: husband (osman, cemile), brother (osman, altan), wife (ayse, altan)

The proposed method changes into formal clauses as follows:

C_1: elti (A, B):-husband(C, A), brother(C, D), wife (B, D)
where A is defined for cemile,
B is defined for ayse,
C is defined for osman,
D is defined for altan

Renaming is applied to all paths of *elti* dataset and similarities are emerged.

Table 10: Candidate Solution Clauses and Frequencies of *elti* after Renaming

Candidate Solution Clauses	**Frequencies**
wife(A,C);brother(C,D);wife(B,D)	8
wife(A,C);brother(C,D);husband(D,B)	8
husband(C,A);brother(C,D);husband(D,B)	8
wife(A,C);brother(D,C);husband(D,B)	8
husband(C,A);brother(C,D);wife(B,D)	8
husband(C,A);brother(D,C);wife(B,D)	8
husband(C,A);brother(D,C);husband(D,B)	8
wife(A,C);brother(D,C);wife(B,D)	8

5. Evaluation and Pruning: For *elti* dataset, for evaluation and pruning the parameters are defined for concept discovery process.

Table 11: Parameters for evaluation and pruning on *elti* graph

Parameters	
number of total paths	64
minimum support threshold	0.2
minimum confidence threshold	0.6
maximum clause depth	3

According to idea of Apriori algorithm, each support and confidence values of distinct clauses are calculated. Candidate solution clauses and frequencies are given for concept instances of *elti (A, B)*.

Table 12: Candidate solution clauses and their support values of *elti* graph

Candidate Solution Clauses	**Support Value**
wife(A,C);brother(C,D);wife(B,D)	0.125
wife(A,C);brother(C,D);husband(D,B)	0.125
husband(C,A);brother(C,D);husband(D,B)	0.125
wife(A,C);brother(D,C);husband(D,B)	0.125
husband(C,A);brother(C,D);wife(B,D)	0.125
husband(C,A);brother(D,C);wife(B,D)	0.125
husband(C,A);brother(D,C);husband(D,B)	0.125
wife(A,C);brother(D,C);wife(B,D)	0.125

No candidate solution clauses have higher support value than minimum support threshold. So, the proposed method selects the path that has highest support value. After selection, the proposed method calculates confidence values from relational databases. In this example all paths have same support value, so all of them are selected for evaluation of confidence values.

Table 13: Candidate solution clauses and their confidence values of *elti* graph

Candidate Solution Clauses	**Confidence Value**
wife(A,C);brother(C,D);wife(B,D)	1.0
wife(A,C);brother(C,D);husband(D,B)	1.0
husband(C,A);brother(C,D);husband(D,B)	1.0
wife(A,C);brother(D,C);husband(D,B)	1.0
husband(C,A);brother(C,D);wife(B,D)	1.0
husband(C,A);brother(D,C);wife(B,D)	1.0
husband(C,A);brother(D,C);husband(D,B)	1.0
wife(A,C);brother(D,C);wife(B,D)	1.0

According to confidence values, no candidate solution clauses are pruned. They all have higher confidence values from minimum confidence threshold. 8 newly solutions are found.

Rule 1: elti (A, B):- wife (A, C); brother(C, D); husband (D, B);

Rule 2: elti (A, B):- husband(C, A); brother(C, D); husband (D, B);

Rule 3: elti (A, B):- wife (A, C); brother (D, C); husband (D, B);

Rule 4: elti (A, B):- husband(C, A); brother (D, C); wife (B, D);

Rule 5: elti (A, B):- husband(C, A); brother (D, C); husband (D, B);

Rule 6: elti (A, B):- wife (A, C); brother (D, C); wife (B, D);

Rule 7: elti (A, B):- husband(C, A); brother(C, D); wife (B, D);

Rule 8: elti (A, B):- wife (A, C); brother(C, D); wife (B, D);

3.2.2. Concept Discovery on Attribute-Based Datasets

Muta dataset is classified as a relation-based dataset by the proposed method. We illustrate the steps of concept learning of *muta* dataset. However, some additional steps are also represented in attribute-based dataset such as preprocessing constants and unification of paths. In this part, these steps are also illustrated on concept discovery on attribute-based datasets.

3.2.3. Preprocessing Constants

However, preprocessing is required because in attribute-based datasets, their relations contain both numeric and alphabetical constants. These constants may be problem for concept learning because it is not always feasible to give different variable for each constant. In attribute-based datasets, we follow different approaches for both numeric and alphabetical constants.

Numeric Constants: Numeric constants in dataset may be problem because it may be distributed in a wide range and we prefer to group them instead of giving different naming for each constant. So, the proposed method tries to group these if possible by using classification tool of WEKA.

If the relation contains one constant, the characteristics can be distributed in distinctive cut because the proposed method refines the connection between the argument of target instance and constants. If the relation contains more than one constant, it may not be clear which constants has affect the characteristic of argument of target instance. To illustrate, in muta data set *muta_bond_count* relation is exemplified.

In muta dataset, the mutagenicity of drugs is analyzed. The proposed method searches the effect of muta bond count number to mutagenicity of drugs. WEKA uses decision tree of J48 and Decision Stump for classifying the relation of mutagenicity and muta bond count number.

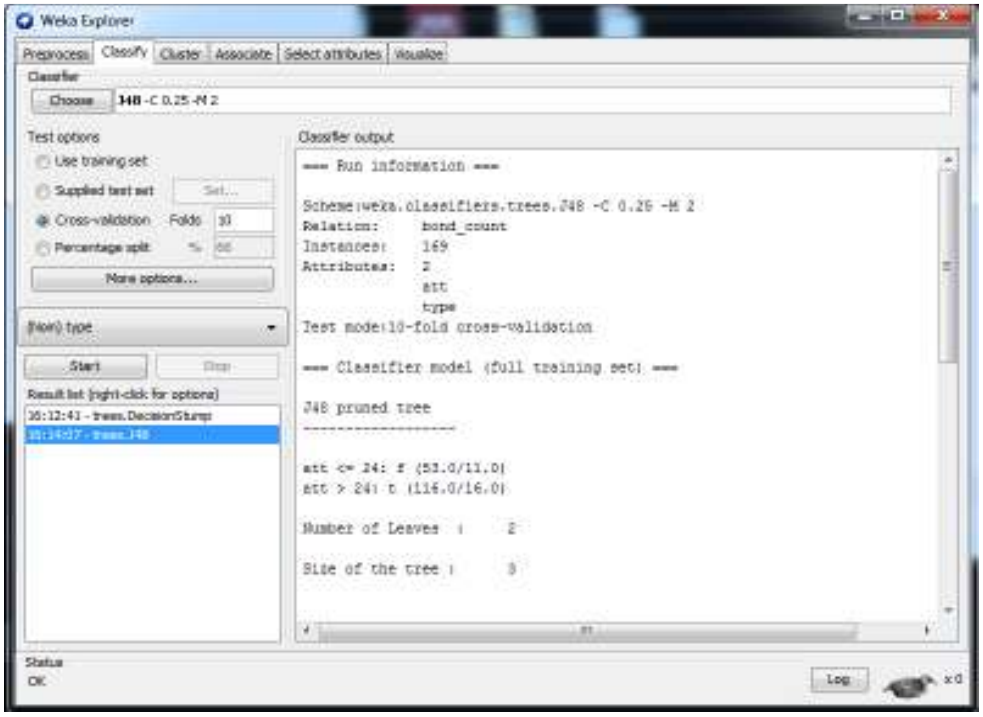

Figure 9: WEKA Explorer for J48 Pruned Tree

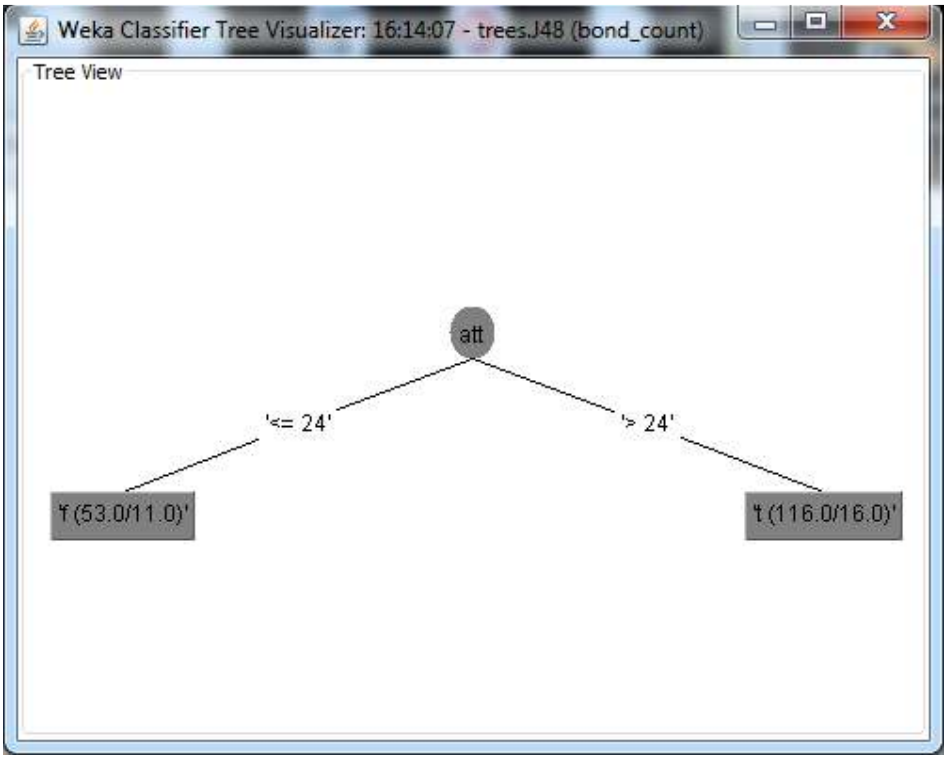

Figure 10: WEKA J48 Pruned Tree Visualizer

C_1: *muta1_train (A, F):- muta_bond_count (A, <=24) support value:* $\frac{42}{58}$

C_2: *muta1_train (A, T):- muta_bond_count (A, >=25) support value:* $\frac{100}{111}$

According to given J48 Pruned Tree, support values of given clauses are calculated and it confirms WEKA distribution for given minimum support threshold.

However, for decimal number, different strategy will be processed because decimal numbers may scattered in expansive range. The proposed method will not use the classification tool of WEKA namely *J48 or Decision Stump* decision trees because they divide numbers into two parts which is inappropriate for expansive ranges.

In CRIS, a strategy is applied for grouping decimal number. Feasible ranges are defined according to *min_support_threshold* and row number of table. To illustrate,

muta_atm relation that has charge value defined as a decimal number is given. The proposed method divides charge values according to number that is calculated by *min_support_threshold * row number* of *muta_atm* relation.

To illustrate this, table of *muta_atm* relation in muta dataset is given. For *muta_atm* relation for drugs that has *muta1_train* is false, 1204 rows are given range from [-0.781, 0.864]. For 1204 numbers, using decision tree is not sensible. Because when it is divided into two parts, accuracy may be lost.

Table 14: Charge ranges and differences classified by J48 Pruned Decision Tree

Ranges	**Difference**
[-0.781, -0.038]	743
[-0.012, 0.864]	876

However, if it is divided into parts according to *min_support_threshold* row number,* boundaries of ranges will be more clear and tendency of charge can be understandable.

For example, the rows are grouped into 10 parts for minimum support threshold of 0.1.

Table 15: Division of charge values based on support threshold

Ranges	Difference	Range Name
[-0.563, -0.384]	0.179	x1
[-0.383, -0.128]	0.255	x2
[-0.127, -0.121]	0.006	x3
[-0.120, -0.116]	0.004	x4
[-0.115, -0.111]	0.004	x5
[-0.110, 0.056]	0.054	x6
[0.057, 0.131]	0.074	x7
[0.132, 0.140]	0.008	x8
[0.141, 0.146]	0.005	x9
[0.147, 0.516]	0.369	x10

Each range contains 120 drugs and difference of boundaries of range shows that the tendency of drugs. Because minor difference means that drugs have similar or close charge values. It is seen from the table, charge arguments of *muta_atm (drug1, atom, element, int, charge)* relation show tendency to the intervals [-0.120, -0.116], [-0.115, -0.111].

Alphabetical Constants: For alphabetical constants, we classify these constants according to their frequencies in terms of minimum support threshold and total row number of given table.

Table 16: SQL template query for alphabetical constant

```
SELECT constant FROM given_table GROUP BY constant HAVING
COUNT(*) >= total_row_number * minimum_support_threshold
```

To illustrate, for *muta* dataset, *muta_atm* table has alphabetical constant as element attribute. *muta_atm* table has totally 5894 rows and minimum support threshold is defined as 0.1.

Table 17: SQL query for alphabetical constant

```
SELECT element FROM muta_atm GROUP BY element HAVING
COUNT(*) >= 5894 * 0.1
```

In *muta* dataset, *muta_atm* relation only takes element attribute as *c,h,o*.

In preprocessing step, we add 7 candidate solution clauses for concept learning.

Table 18: Candidate solution clauses from preprocessing step

Candidate Solution Clauses
muta_atm_count (A,<=21)
muta_atm_max_charge (A,>0.8305);
muta_atm_min_charge (A,<=-0.3925);
muta_bond_count (A,<=24);
muta_ind1 (A,<=0.5);
muta_logp (A,<=2.265);
muta_lumo (A,>-1.0855);

1. Attribute-based Graph Construction: Schemas of attribute-based datasets such as *muta, PTE-1, mesh and eastbound* is different in many aspects. Commonly, they hold attributes of one node in relations and there is no link between the arguments of target instances. To illustrate, *pte_bond_count* relation holds two arguments in terms of drug and its properties. There is no path or link between these arguments of relation. So, arguments should be stored as an attribute according to domain knowledge. We name these datasets as attribute-based relations.

Moreover, there exists n-ary relations that contain numeric constants and these constants should be grouped according to defined feasible ranges by the help of classifying tools of WEKA. To illustrate, *muta_atm* relation is given. *muta_atm*

relation shows one drug and its attributes. This relation holds one alphabetic constant and two numeric constants with different ranges.

Table 19: Example of relations for attribute-based datasets

Relations of Attribute-based Datasets
pte_bond_count(d1, 28)
muta_atm(d1, d1_1, c, 22, -0.113)

According to basis of Apriori algorithm, data preprocessing is applied for attribute-based datasets. The proposed method evaluates the tables of relational database in terms of frequency, and eliminates infrequent tables.

For instance, 340 drugs and their properties are defined in PTE-1 dataset. In each table, drugs and their related attributes are given. For avoiding the complexity of computation, the proposed system eliminates infrequent tables. The proposed method runs this clause in relational database for each table given in muta dataset.

Table 20: SQL template query for infrequent table

```
SELECT DISTINCT DRUG FROM GIVEN_MUTA_TABLE;
```

For given support threshold, tables that have number of rows below the support threshold are eliminated. For example for *pte_alcohol* table, we run this query and analyze its result.

Table 21: SQL query for infrequent table

```
SELECT DISTINCT ARG0 FROM PTE_ALCOHOL;
```

It returns 8 rows. According to given support threshold which is defined as 0.1, we calculate table threshold as *number_of_drugs * support_threshold*. For this query, result is below 34 and this table is eliminated while constructing graph. Because it is a signal that *pte_alcohol* table cannot generate frequent items. After eliminating these infrequent tables, graph construction is started.

In attribute-based graphs, there will be no direct paths between the arguments of target concepts. Target instances define the features of elements. For example, in muta dataset target instances show whether it is mutagenesis or not. And according to its features, some common attributes are grouped.

Because as it is shown below, this attributes are not related with each others. It is only linked the first arguments of the target instances defined as *muta1_train (drug, boolean)*. This is an example of drug that is placed in muta dataset.

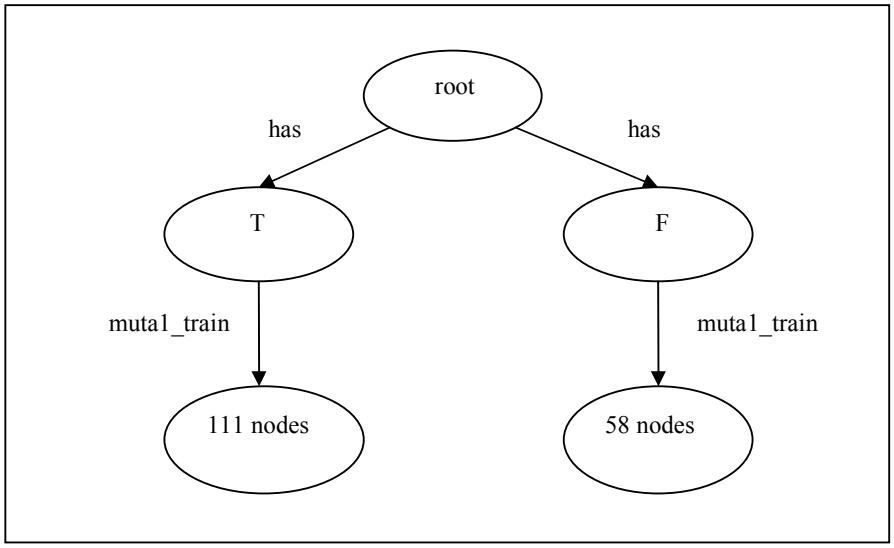

Figure 11: Connected graph of *muta* dataset based on mutagenicity of drugs

In proposed method, for constructing the connected graph 3 nodes are added to *muta* dataset. *T* node and *F* node are collected under the *root*. *Root* node connected to others by *has* relation. This relation is not defined in muta dataset and it is used for only constructing connected graph. Each drug defined as *true* in *muta1_train* relation, is created as node and united under the node named as *T* in proposed method. Conversely, each drug defined as *false* in *muta1_train* relation, is created as node and united under the node named as *F*.

It is a representation of one target instance that is defined in *muta* dataset. This *d5* drug has 14 muta_*atm* relations. However, in this representation 2 *muta_atm* relations are shown for clear understanding and visual quality.

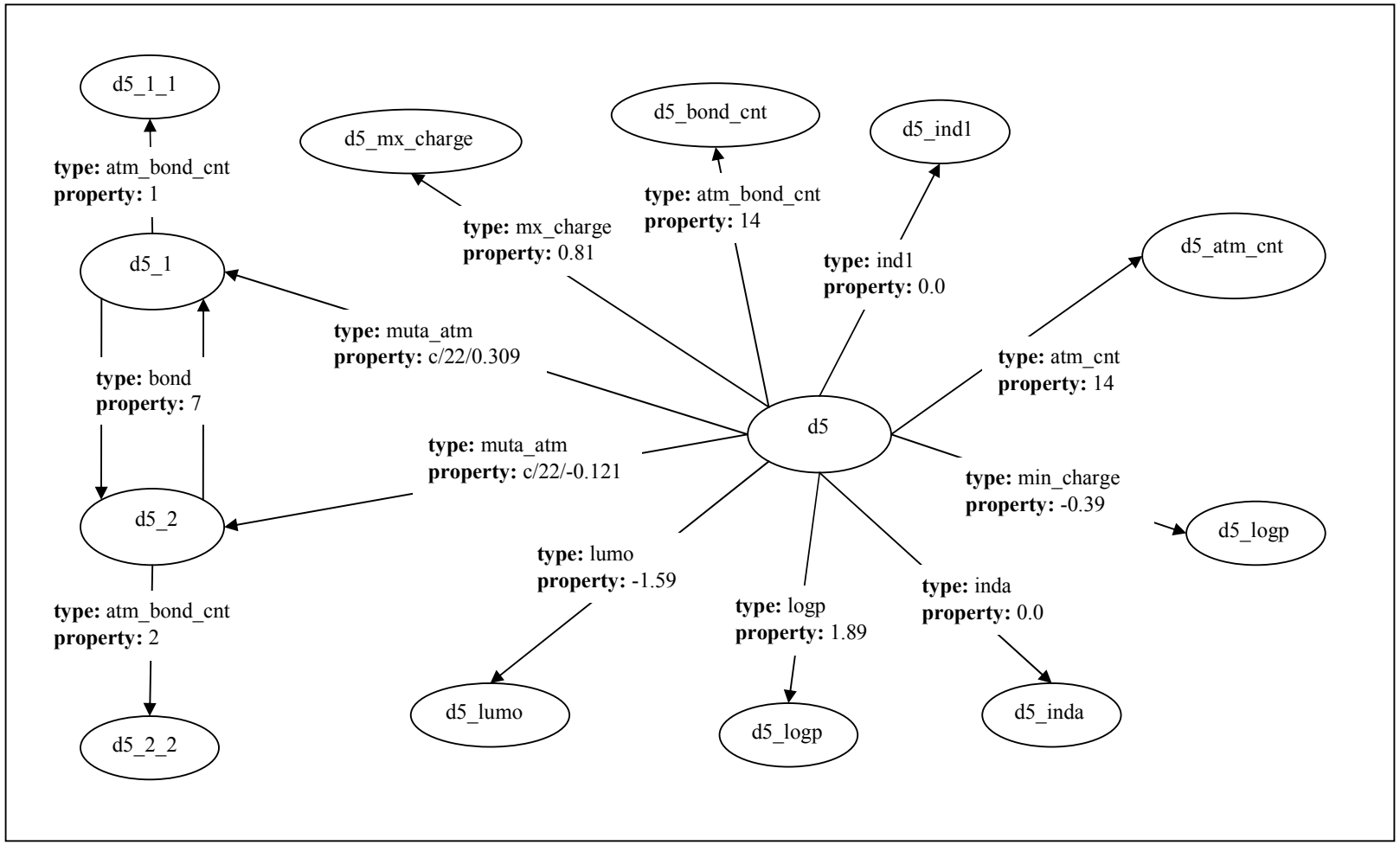

Figure 12: Sub-graph of *muta* graph for *d5* drug

For preventing the confusion, we define new nodes according to tables that contain attributes. For example, for *muta_bond_count* relation of *d169* drug, new unique node is created. Because if the graph holds value of 16 as a node, many drugs related to relation that contains 16, try to connect this new common node. In attribute-based graph, paths are found in an algorithm by backtracking and these may disrupt concept learning.

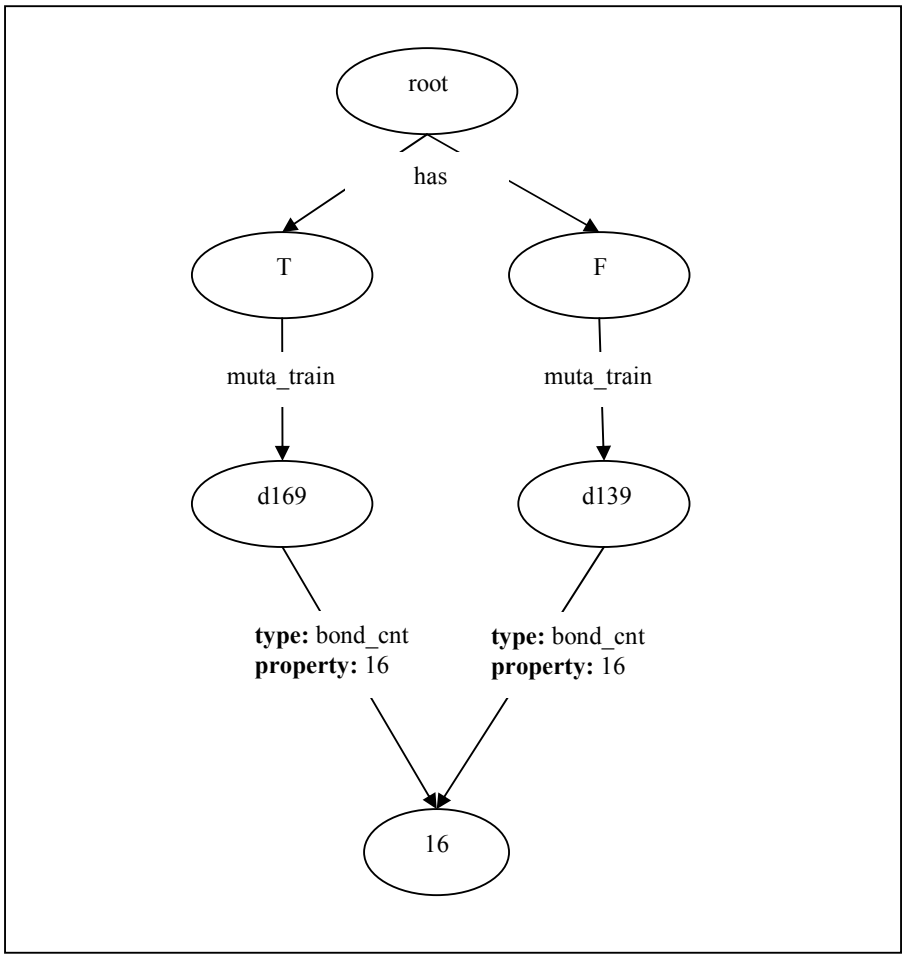

Figure 13: Creating unique nodes for preventing confusion

2. Attribute-based Path Finding: In attribute-based path finding, target instances represent the features of datasets. There is no link between arguments of target instances. In attribute-based graph, we cannot traverse between two nodes. So, we focus on the all paths that contain given instances.

We examine each concept instance independently. To illustrate, *d5 drug* of muta dataset is analyzed. In this sub-graph, all paths that contain *d5 drug* are considered. When constructing rule, properties of relations are used for filling other arguments of rule. For example, the property of *muta_atm* relation is parsed and used for filling *element, int* and *charge* arguments.

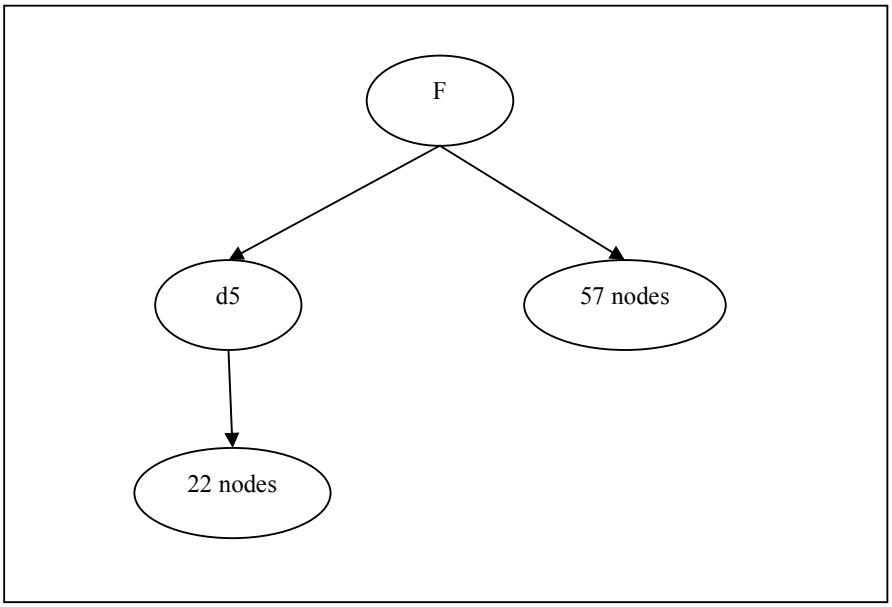

Figure 14: Non-mutagenesis drugs of *muta* dataset

For each 57 nodes, path finding process is repeated.

3. Attribute-based Filtering: In attribute-based datasets, filter processing is applied for only eliminating paths that contain only one relation. Because path finding strategy is different. We take paths that contain given arguments. In relation-based

datasets, we search for paths between two nodes. The proposed method uses different implementation of path finder's of Neo4j and we can give parameters for eliminating loop and concept relation when search process is applied.

4. Attribute-based Renaming: In attribute-based datasets, there is a little change in renaming process. All arguments except constants are renamed again according to alphabetical ordering. However, numeric constants distributed in a wide range are renamed according to ranges defined in preprocessing part of path finding step.

To illustrate *muta_atm* relation of muta dataset is given. In table 15, ranges and their names are defined based on minimum support threshold. Drugs related with *muta_atm* are grouped according to their charge range. For example, paths that have charge values higher than -0.110 and lower than 0.056 has grouped and renamed path as *muta_atm(A,B,c,22,x6)*. After renaming process we have 2399 paths for *muta1_train(A, true)*.

Table 22: Candidate solution clauses and their frequencies

Candidate Solution Clauses	**Frequencies**
muta_atm(A,B,c,22,x6);	152
muta_atm(A,B,h,3,x6);	31
muta_atm(A,B,c,22,x2);	100
muta_atm(A,B,c,22,x5);	298
muta_atm(A,B,c,22,x10);	44
muta_atm(A,B,h,3,x9);	348
muta_atm(A,B,h,3,x10);	170
muta_atm(A,B,c,22,x4);	267
muta_atm(A,B,o,40,x2);	96
muta_atm(A,B,h,3,x8);	222
muta_atm(A,B,c,22,x3);	223
muta_atm(A,B,c,22,x7);	8
muta_atm(A,B,h,3,x7);	184
muta_atm(A,B,o,40,x1);	256

5. Attribute-based Union: Union process is only applied to attribute-based datasets. Unification is different from the union strategy of CRIS. We have only one criterion for unification of two candidate solution clauses. If candidate solution clauses have the same head literals, unification is feasible.

In CRIS unification, new generated clause refined by unification is accepted as a candidate solution clause if it has higher confidence value from the candidate solution clauses that used for unification. However, in this step, it is sufficient for acceptance of unified clause as a candidate solution clause, if it has higher confidence of minimum confidence threshold.

6. Attribute-based Evaluation and Pruning: For *muta* dataset, for evaluation and pruning the parameters are defined for concept discovery process.

Table 23: Parameters for evaluation and pruning on *muta* graph

Parameters	
number of total paths	2399
minimum support threshold	0.1
minimum confidence threshold	0.7
maximum clause depth	3

According to idea of Apriori algorithm, each support values of distinct clauses are calculated based on their path number. Candidate solution clauses and support values are given for concept instances of *muta1_train (A, true)*.

Table 24: Candidate solution clauses and their support values of *muta* graph

Candidate Solution Clauses	**Support Values**
muta_atm(A,B,c,22,x6);	0.06
muta_atm(A,B,h,3,x6);	0.01
muta_atm(A,B,c,22,x2);	0.04
muta_atm(A,B,c,22,x5);	0.124
muta_atm(A,B,c,22,x10);	0.018
muta_atm(A,B,h,3,x9);	0.145
muta_atm(A,B,h,3,x10);	0.07
muta_atm(A,B,c,22,x4);	0.111
muta_atm(A,B,o,40,x2);	0.04
muta_atm(A,B,h,3,x8);	0.09
muta_atm(A,B,c,22,x3);	0.09
muta_atm(A,B,c,22,x7);	0.003
muta_atm(A,B,h,3,x7);	0.08
muta_atm(A,B,o,40,x1);	0.107

We have 4 candidate solution clauses after support based pruning in first iteration. Moreover, 7 candidate solution clauses are also added from preprocessing step of *muta* dataset and candidate solution clauses are added from union step. In last iteration, after confidence based pruning we have totally 77 candidate solution clauses. According to f-metric calculation, we have best rule for *muta1_train (A, true)*.

muta_atm_max_charge(A,<=0.83); muta_logp(A,>2.27); muta_lumo(A,<=-1.09)
support = 0.95, confidence = 0.71

CHAPTER 4

EXPERIMENTAL RESULTS

In this section, firstly we describe the datasets of well-known problems tested in the experiments. After describing, we present the performance of the proposed approach in terms of coverage and accuracy. The experimental results are compared other ILP-based approaches. We compare our results with CRIS and Hybrid Graph Based method. Because Hybrid Graph based method is similar to proposed approach in many aspects. It finds paths on graph for concept discovery. Also, it processes pruning based on support and confidence values. Conversely, CRIS discovers concepts on non graphs. However, it uses similar methods of Apriori algorithm for pruning based on support and confidence thresholds.

Coverage is defined the ratio of the number of target instances of test data set covered by the induced hypothesis over the all target instances. Also, accuracy is defined as the ratio of sum of instances both negative and positive that is correctly covered over the sum of true positive, true negative, false negative and true negative instances [7].

4.1. Data sets

The proposed method is tested on five different datasets as a learning problem. We group these data sets into two parts according to their graph structures. We run these experiments on a computer that has Intel Core 2.53GHz processor and 4 GB memory. In table, we give introduction for information and properties of benchmark data sets.

Table 25: Summary of the benchmark dataset used in the experiments

Data Set	Description
Elti	A real world family data set which contains transitive relations directly related to target instances.
Dunur	A real world family data set which contains indirectly relations to target instances.
Same-Gen	A real world kinship data set which contains recursive relations directly to target instances.
Eastbound	A data set contains information about trains and their related facts.
Mesh	A data set about determination of the number of elements on each edge of the mesh.
Muta	A data set about chemicals which aims to learn mutagenicity of each drug that contains.
PTE-1	A biochemical data sets about learning whether a chemical is carcinogenic or not.

For each data set, total number of relationships and facts are shown.

Table 26: Properties and experimental settings for relation-based dataset

Data Set	#Num. Pred.	#Num. Ins.
Elti	9	224
Dunur	9	224
Same-Gen	2	408
Eastbound	12	196
Mesh	26	1749
Muta	26	15003
PTE-1	32	29267

In this section, the parameters of the proposed approach are given such as minimum support, minimum confidence and maximum length.

Table 27: Evaluation parameters for data sets

Data Set	Min. Support	Min. Conf.	Max. Length	B
Elti	0.2	0.6	3	1
Dunur	0.3	0.7	3	1
Same-Gen	0.3	0.6	3	1
Eastbound	0.1	0.6	3	1
Mesh	0.1	0.7	3	1
Muta	0.1	0.7	3	1
PTE-1	0.1	0.7	3	1

4.2. Experimental Results

4.2.1. Time Comparison for Relation-based Datasets

Total nodes and relations that are created while constructing graphs are shown below.

Table 28: Graph properties of relation-based datasets

Datasets	Number of Nodes	Number of Relations
Dunur	24	128
Elti	47	224
Same_gen	47	408

Construction times according to relation-based datasets are given below.

Table 29: Graph construction time for relation-based datasets

Datasets	Graph Construction Time (sec)
Dunur	2
Elti	1
Same_gen	1

This chart represents total number of nodes and relations for each dataset in constructing graph. As it is seen, construction time is directly related to total number of relations and nodes.

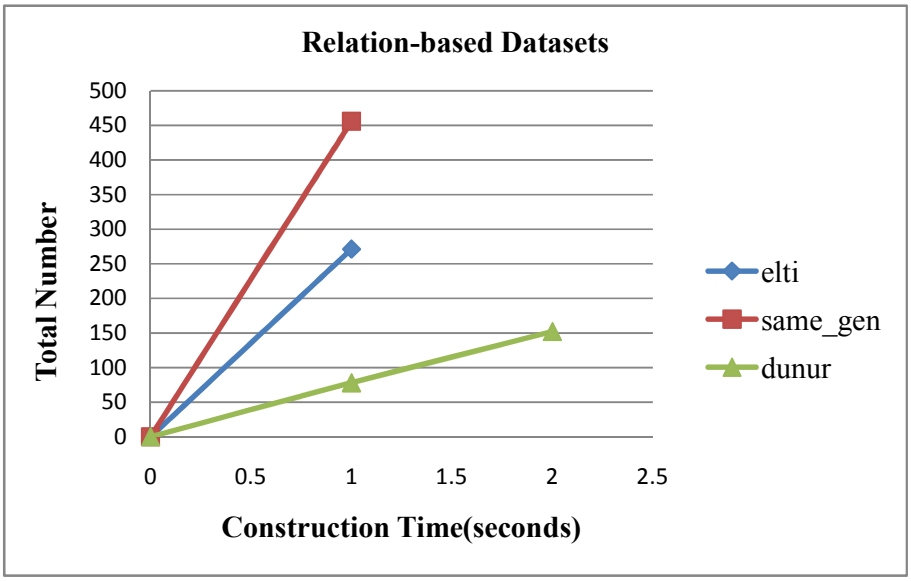

Figure 15: Construction time versus total nodes and relations for relation-based

4.2.2. Time Comparison for Attribute-based Datasets

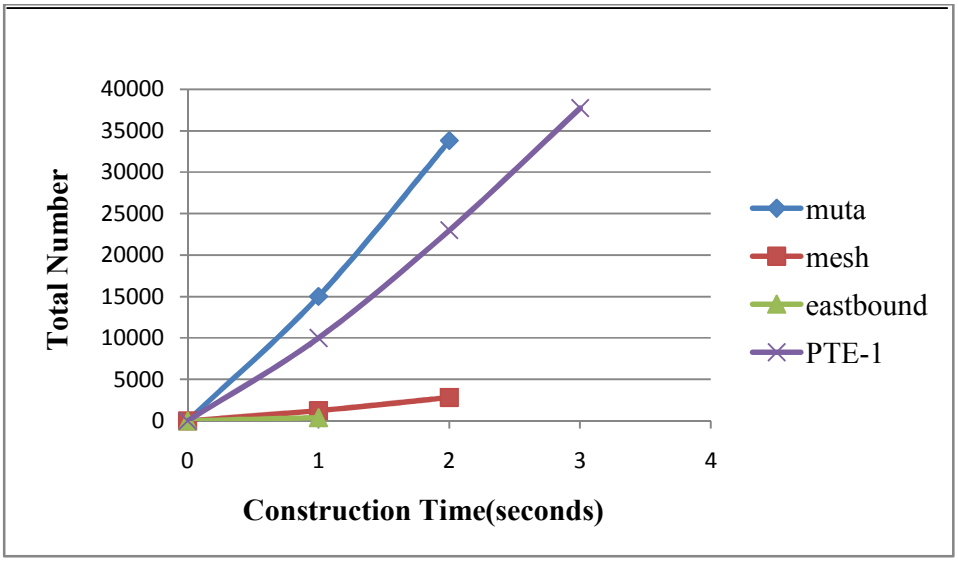

Figure 16: Construction time versus total nodes and relations for attribute-based

As it is seen from the chart, there is no explicit relation between construction time and total nodes. However, it may be stated that the construction time of attribute-based graphs is lower than the construction time of relation-based graphs when compared to total of number of nodes and number of relations.

Table 30: Graph construction time for attribute-based datasets

Datasets	Graph Construction Time (sec)
muta	2
mesh	2
eastbound	1
PTE-1	3

As it is seen number of relations and nodes of the attribute-based datasets are higher, because the proposed method handles constants by making them distinct nodes.

Table 31: Graph properties of attribute-based datasets

Datasets	**Number of Nodes**	**Number of Relations**
muta	11388	22401
mesh	1134	1694
eastbound	197	196
PTE-1	15509	22234

In attribute-based graphs, information is carried with properties of relationships. It is seen that there is a connection between both property numbers and relationship numbers.

Table 32: Graph properties of attribute-based datasets

Datasets	**Relationship Type**	**Number of Properties**
muta	13	53697
mesh	25	3172
eastbound	13	635
PTE-1	28	56879

4.3. Concept Discovery on Relation-based Datasets

4.3.1. Concept Discovery on Same-Generation Data Set

In same-gen data set, 344 pairs of person are given as positive examples of *same_gen*. Additionally, 64 background facts are provided to describe the *parent* relationships in the family. And there are 47 persons in the examples for describing the argument of relations. Relations of *same-gen* and *parent* have two arguments having person.

According to parameters of confidence threshold as 0.6, support threshold as 0.3 and maximum depth as 3, the proposed method discovers concepts. This concept discovery is slightly different that the other datasets because recursion is allowed.

Rule 1:same_gen(A, B):-parent(C, A); parent(C, D); same_gen(D,B)
Rule 2: same_gen(A, B):-parent(C, A); parent(C, D);same_gen(B,D)
Rule 3: same_gen(A, B):-same_gen(A, C); parent (D, C); parent (D, B)
Rule 4: same_gen(A, B):-same_gen(C, A); parent (D, C); parent (D, B)

This recursive rule shows that the proposed method was successful for learning the target relations of same-gen dataset when compared to coverage and accuracy results of the other art of concept discovery systems. The proposed method has same accuracy and higher percentage of coverage same as CRIS and Hybrid Graph Based method.

Table 33: Coverage and Accuracy Results for *same-gen* Dataset

Dataset	**Coverage**	**Accuracy**
Hybrid Graph based	0.84	1.0
CRIS	0.84	1.0
Proposed Method	1.0	1.0

The experiment shows that the proposed method is slightly slower than both CRIS and Hybrid Graph based method.

Table 34: Running Time for *same-gen* Dataset

Dataset	**Running Time(hh:mm:ss.s)**
Hybrid Graph based	00:00:76
CRIS	00:00:12
Proposed Method	00:00:24

For this *same_gen* data set, ALEPH, PROGOL and GOLEM cannot find any solution under given parameters. ALEPH finds the following hypothesis:

Rule 1: same_gen(A, B):- parent(C, A); parent(C, B)
Rule 2: same_gen (A, B):- same_gen (A, C); same_gen (C, B)
Rule 3: same_gen (A, B):- parent(C, A);same_gen(C,D); parent(D, B)

However, PROGOL can only find *same_gen(A,B):-same_gen(B, C);same_gen (C ,A)* as a solution. Similarly, GOLEM could not find any solution under strong mode declarations.

4.3.2. Concept Discovery on Elti Data Set

In this experiment, *elti* relation represents the family relation between the wives of two brothers. And *elti* is a commonly used term which is a Turkish word for family relationship.

In the data set, *elti* is selected as target relation. For *elti* data set, brother instances are unrelated facts of *elti* data set because the people in *elti* relation have no brothers. Therefore, brother instances are defined as unrelated facts. The proposed method runs on the data set for minimum support as 0.2 and minimum confidence as 0.6. Four rules are found from proposed approach.

Rule 1: elti (A, B):- wife (A, C); brother(C, D); husband (D, B);
Rule 2: elti (A, B):- husband(C, A); brother(C, D); husband (D, B);
Rule 3: elti (A, B):- wife (A, C); brother (D, C); husband (D, B);
Rule 4: elti (A, B):- husband(C, A); brother (D, C); wife (B, D);
Rule 5: elti (A, B):- husband(C, A); brother (D, C); husband (D, B);
Rule 6: elti (A, B):- wife (A, C); brother (D, C); wife (B, D);
Rule 7: elti (A, B):- husband(C, A); brother(C, D); wife (B, D);
Rule 8: elti (A, B):- wife (A, C); brother(C, D); wife (B, D);

Same as the same-gen dataset, the proposed method finds same solution sets of CRIS and Hybrid Graph based method. The proposed method is successful on *elti* data set for concept learning.

Table 35: Coverage and Accuracy Results for *elti* dataset

Dataset	**Coverage**	**Accuracy**
Hybrid Graph based	1.0	1.0
CRIS	1.0	1.0
Proposed Method	1.0	1.0

The experiment shows that the proposed method is the fastest method when compared to other approaches.

Table 36: Running Time for *elti* dataset

Dataset	**Running Time(hh:mm:ss.s)**
Hybrid Graph based	00:00:51
CRIS	00:00:35
pCRIS	00:00:99
Tabular CRIS	00:02:12
Proposed Method	00:00:04

For the *elti* dataset, GOLEM cannot find a rule under several mode declarations. Moreover, PROGOL also cannot find a rule for *elti* dataset. However, if only *husband*, *wife* and *brother* relations are given as background knowledge, it finds only one of the transitive rule under strict mode declarations:

elti(A, B) husband(C, A), husband(D, B), brother(C, D).

ALEPH can only find one of the transitive rules for this experiment:

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

4.3.3. Concept Discovery on Dunur Data Set

In this experiment, *dunur* relation represents the family relationship of two persons who are the parents of a married couple. In the data set, the *dunur* relation is selected as the target relation, and we run proposed method on *dunur* data set for minimum support as 0.2, minimum confidence as 0.6. Four rules are found from proposed approach.

Rule1: dunur (A, B):- son(C, A); wife (D, C); daughter (D, B)
Rule2: dunur (A, B):- son(C, A); husband(C, D); daughter (D, B)
Rule3: dunur (A, B):- daughter(C, A); husband (D, C); son (D, B)
Rule4: dunur (A, B):- daughter(C, A); wife(C, D); son (D, B)

The proposed method finds same solution sets of CRIS and Hybrid Graph based method. The proposed method is also successful on learning from *dunur* data set.

Table 37: Coverage and Accuracy Results for *dunur* dataset

Dataset	**Coverage**	**Accuracy**
Hybrid Graph based	1.0	1.0
CRIS	1.0	1.0
Proposed Method	1.0	1.0

The experiment shows that the proposed method the fastest when compared to other approaches.

Table 38: Running Time for *dunur* dataset

Dataset	**Running Time(hh:mm:ss.s)**
Hybrid Graph based	00:00:11
CRIS	00:00:26
pCRIS	00:00:90
Tabular CRIS	00:02:88
Proposed Method	00:00:08

The *dunur* experiments are conducted on PROGOL, ALEPH and GOLEM systems. Neither of the systems could find any rules in both of the experiments either under strict mode declarations.

4.4. Concept Discovery on Attribute-based Datasets

4.4.1. Concept Discovery on Eastbound Data Set

The eastbound data set is a kind of attribute-based graph. The eastbound has five target instances which are {east1, east2, east3, east4, east5} [60]. The eastbound instances only related to *has_car* background relation. The other is indirectly related to *eastbound* target instances. The proposed method run on *eastbound* data set under parameters of minimum support 0.2 and min confidence 0.6. The proposed method finds best rule defined as:

Rule1: eastbound (A):-has_car (A, B), closed (B), tr_short (B) (s=1.0, c=1.0)

CRIS finds rule defined which has f-metric score as 0.83 which has lower score than the proposed method.

eastbound (A):-has car (A, B), closed (B)(s=5/5, c=5/7)

PROGOL finds only the following rule:

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

ALEPH cannot find a rule without negative instances. When negative instances are provided, it finds the following best for this experiment:

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

However; GOLEM cannot find a rule for this experiment.

Accuracy and coverage of the proposed method is higher than CRIS.

Table 39: Coverage and Accuracy Results for *eastbound* dataset

Dataset	**Accuracy**	**Coverage**
CRIS	0.7	1.0
Proposed Method	1.0	1.0

It is seen that running time of the proposed method is better than CRIS explicitly.

Table 40: Running Time for *eastbound* dataset

Dataset	**Running Time(hh:mm:ss.s)**
CRIS	00:11:36
pCRIS	00:01:88
Tabular CRIS	00:06:54
Proposed Method	00:00:12

4.4.2. Concept Discovery on Mutagenesis Data Set

Concept discovery on mutagenesis is about the problem of predicting the mutagenic activity of small molecules in terms of a property that is related to carcinogenicity. In

this dataset, we have 230 compounds and we use the *regression-friendly* data set which has 188 compounds [58].

The target relation *muta1_train* has two arguments having drug and Boolean (true/false) type. *muta1_train* table shows the mutagenicity of 188 drugs.

The proposed method finds concept discovery rules which have higher f-metric values than CRIS.

For non-mutagenicity drugs:

muta1_train (A, false):-muta_atm(A,B,c,22,x10);muta_ind1(A,<=0.5)

For mutagenicity drugs:

muta_atm_max_charge(A,<=0.83);muta_logp(A,>2.27);muta_lumo(A,<=-1.09)

CRIS has the higher accuracy of the proposed method; however, its coverage is lower than the proposed method.

Table 41: Coverage and Accuracy Results for *muta* dataset

Dataset	**Accuracy**	**Coverage**
CRIS	0.85	0.53
Proposed Method	0.85	0.79

It is seen that running time of the proposed method is far better than CRIS explicitly. Although the accuracies are same, the proposed method has higher coverage because we have rule that covers negative instances when compared the solution found from CRIS.

For example, the rule found from CRIS for *muta1_train(A,true)* is shown below with support and confidence values:

Rule: muta_atm(A,B,C,22,<=-0.031),muta_bond_count(A,>=27)
s=0.54, c=0.71, f-metric score = 0.61

The rule found from the proposed method for *muta1_train(A,true)* is shown below with support and confidence values:

Rule:muta_atm_max_charge(A,<=0.83);muta_logp(A,>2.27);muta_lumo(A,<=-1.09)
s=0.95, c=0.71, f-metric score = 0.81

Although their confidence values are equal, the proposed method finds solution having higher support value. And higher support value increases the coverage percentage.

Table 42: Running Time for *muta* dataset

Dataset	**Running Time(hh:mm:ss.s)**
CRIS	03:42:00
pCRIS	00:04:30
Tabular CRIS	00:12:05
Proposed Method	00:02:45

The proposed method may be alternative to CRIS when the accuracy and time concepts are more important than coverage.

4.4.3. Concept Discovery on PTE-1 Data Set

Concept discovery on Predictive Toxicology Evaluation (PTE) dataset is about the problem of predicting the frequent substructures in chemical compounds that cause carcinogenicity. Cancer is a common disease and it is related to environmental factors

in terms of exposure to carcinogenic chemicals. Testing compounds for analyzing carcinogenesis is a very expensive and time consuming process so researches require computer based methods for it. The National Toxicity Program (NTP) of the U.S.

National Institute for Environmental Health Sciences guides bioassays of chemicals on rodents to predict the carcinogenetic effects on human health [59]. The problem is feasible for inductive logic programming because carcinogenicity of chemicals can be predicted based on previous researches using machine learning methods.

More than 300 compounds are classified in terms of carcinogenicity according to the tests guided on the rodents in the NTP program. In this experiment, 298 of them are separated as training set, 39 of them formed test set of first PTE challenge (PTE-1).

The target relation *pte_active* has two arguments having drug and *boolean* type. The type table drug and *boolean* are created having 340 and 2 (*true/false*) records respectively.

The proposed method finds rules according to carcinogenicity of chemicals.

For carcinogenic compounds:

pte_active(A, true):-pte_ames(A);pte_atm(A,B,h,3,x7)

where x7<=0.095 and x7>=0.068

For non-carcinogenic compounds:

pte_atm(A,B,h,3,x7);pte_atm_count(A,>=27);pte_has_property(A,salmonella,n)

where *x7<=0.095 and x7>=0.068*

Table 43: Coverage and Accuracy Results for *PTE-1* dataset

Dataset	**Accuracy**	**Coverage**
CRIS	0.88	Not Supported
Proposed Method	0.85	0.87

It is seen that running time of the proposed method is far better than CRIS explicitly.

Table 44: Running Time for *PTE-1* dataset

Dataset	**Running Time(hh:mm:ss.s)**
CRIS	05:17:00
pCRIS	04:26:00
Tabular CRIS	36:13:00
Proposed Method	01:18:41

CRIS has the higher accuracy of the proposed method; however, its coverage is lower than the proposed method.

4.4.4. Concept Discovery on Mesh Data Set

In mechanical engineering, finite elements methods are excessively used in studying on stressing of physical structures. Although differential equations connect the link between physical structures and the effects of internal and external pressures on these physical structures, the computations of these differential equations do not give result in a reasonable time. As a result, physical structures are represented by finite number, called as mesh, to analyze the errors in the calculated deformation values.

Mesh is a mechanical engineering problem dataset that try to determine the mesh resolution for a given structure that results in accurate deformation values. In mesh dataset, there are 223 training examples and 1474 background facts. The target relation mesh train has two arguments defined by *element* and *integer* type. The type

tables *element* and *integer* are created having 278 and 13 records. The test relation mesh test has 55 examples. We have found 15 rules for *mesh* dataset. Rules and f-metric values are shown below.

Table 45: Mesh Concept Discovery Rules and f-metric Rules

Concept Rules	**f-metric value**
mesh(A,1):-sshort(A)	0.46
mesh(A,2):-neighbour_yz(B,A);usual(A)	0.60
mesh(A,3):-neighbour_xy(A,B);noload(A);usual(A)	0.41
mesh(A,4):-neighbour_xy(B,A);usual(A)	0.22
mesh(A,5):-free(A);neighbour_xy(B,A);usual(A)	0.41
mesh(A,6):-neighbour_zx(A,B);two_side_fixed(A)	0.57
mesh(A,8):-circuit(A);opposite(B,A)	0.5
mesh(A,8):-circuit(A);cont_loaded(A)	0.5
mesh(A,8):-circuit(A);cont_loaded(A);equal(B,A)	0.5
mesh(A,9):-neighbour_xy(B,A);quarter_circuit(A)	1.0
mesh(A,9):-neighbour_yz(B,A);quarter_circuit(A)	1.0
mesh(A,9):-neighbour_yz(A,B);quarter_circuit(A)	1.0
mesh(A,9):-neighbour_xy(A,B);quarter_circuit(A)	1.0
mesh(A,9):-quarter_circuit(A)	1.0
mesh(A,12):-circuit(A);free(A)	0.86

Table 46: Coverage and Accuracy Results for *mesh* dataset

Dataset	**Coverage**	**Accuracy**
CRIS	0.29	0.49
Proposed Method	0.07	0.27

Although the proposed method finds best rules with higher f-metric scores when compared to rules that CRIS found, rules of the proposed method has lower percentage for coverage and accuracy. However, the test dataset is randomly selected and it does not define absolute judgment. When we calculate accuracy on *mesh* table

instead of *mesh_test* table, the proposed method shows percentage of *0.61* over 223 examples.

Table 47: Running Time for *mesh* dataset

Dataset	**Running Time(hh:mm:ss.s)**
CRIS	02:39:34
pCRIS	00:17:12
Tabular CRIS	00:37:36
Proposed Method	00:01.33

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this thesis, we focused on the graph based concept learning approaches and its implementations. Graph databases are recently used because of its advantages. Inevitably, it penetrates ILP-based systems for learning process. So, we propose a new method for graph based concept learning. Although graph databases many advantages of querying, for huge datasets some constraints are needed. Therefore, we combine our approach with Apriori-based pruning mechanism. For graph concept discovery, confidence-based pruning algorithms are conducted [24, 25, 7]. These methods present that eliminating at early steps provide efficiency on running time and accuracy. According to these results we compare our approach. We have tested our approaches on several benchmark problems and comparisons show that the proposed method is successful in terms of accuracy, coverage and time. The proposed method is compatible with current state-of-the-art knowledge discovery systems.

We have divided datasets into two parts according to their structures namely relation-based and attribute-based datasets. According to type of datasets, our approach has differences in concept discovery steps.

For relation-based datasets, the proposed method searches for paths between two nodes that represent argument of the target instances. Our approach has different view when calculating the support values of paths. Instead of querying from relational databases, it calculates support values based on frequency of paths and total number of paths. After calculation support values, paths are tested based on their confidence values by querying in relational databases. The paths that have higher confidence

value from confidence threshold are considered as candidate solution clauses. The other paths are pruned based on Apriori method. By this way, we intend to reduce the computational complexity. After eliminating paths based on the confidence pruning, best rule is selected according to f-metric value. Using f-metric value for best rule selection conducts hypothesis with higher quality. The experiments show that the proposed method learns concepts on relation-based datasets regularly in terms of accuracy, coverage and running time.

For relation-based datasets, before construction of graph, preprocessing is required. We preprocess datasets with the help of WEKA because it contains both alphabetical and numeric constants. The paths that have constants are classified and considered as candidate solutions in the beginning step of the proposed method. Apart from preprocessing, concept learning on attribute-based dataset differs by adding Unification step to proposed method. After unification step, evaluation and pruning applied to unified paths for determining whether they are infrequent paths or not. The experiments show that the proposed method of concept learning on attribute-based datasets has better than most of the state-of-the-art knowledge discovery systems in terms of accuracy, coverage and running time.

In future work, the proposed method can be improved in terms of several directions. Firstly, we may focus on index mechanism of Neo4j for accelerating the running time of concept learning, because concept learning on relation-based datasets is slightly slower. Moreover, we may try to different approaches for constructing attribute-based datasets. In this approach, nodes hold attributes instead of making them distinct nodes. By this way, we may reduce the number of relations and nodes in graph. Another improvement may be applied to calculation of confidence values. We may try to calculate confidence values by querying in Neo4j as the future work. By this way, we may break the dependency of relational databases for proposed method.

REFERENCES

[1] Fonseca, N., Costa, V. S., Silva, F., Ciˆ, D. De, Fonseca, N., Costa, V. S. … Camacho, R. (2003). On the Implementation of an ILP System with Prolog Camacho encia de Computadores on the Implementation of an ILP System with Prolog.

[2] Muggleton, S., & de Raedt, L. (1994). Inductive Logic Programming: Theory and methods. *The Journal of Logic Programming*, *19-20*, 629–679. doi:10.1016/0743-1066(94)90035-3

[3] Sowa, J. F. (1984). *Conceptual structures: information processing in mind and machine*. Addison-Wesley Longman Publishing Co., Inc.

[4] Muggleton, S. (1991). Inductive logic programming. *New generation computing*,*8* (4), 295-318.

[5] Miller, J. J. (2013). Graph Database Applications and Concepts with Neo4j.

[6] Muggleton, S. (1997). Learning from positive data. In *Inductive logic programming* (pp. 358-376). Springer Berlin Heidelberg.

[7] 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).

[8] F. Bergadano. Inductive Data Base Relations. To appear in IEEE Trans. on Data and Knowledge Engineering, 1993.

[9] Bergadano, F., & Gunetti, D. (1993). Learning relations: Basing top-down methods on inverse resolution. In *Advances in Artificial Intelligence* (pp. 190-201). Springer Berlin Heidelberg.

[10] Plotkin, G. D. (n.d.). A Further Note on Inductive Generalization, (c).

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

[12] Srinivasan, A. (2001). The ALEPH manual. *Machine Learning at the Computing Laboratory, Oxford University*.

[13] King, R. D., Srinivasan, a, & Dehaspe, L. (2001). WARMR: a data mining tool for chemical data. *Journal of Computer-Aided Molecular Design*, *15*(2), 173–81. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/11272703 [last accessed on 23.10.2014].

[14] Gonzalez, J., Holder, L., & Cook, D. (n.d.). Application of Graph-Based Concept Learning to the Predictive Toxicology Domain.

[15] Richards, B. L. (1992). Learning Relations by Path-finding, 50–55.

[16] Ong, I. M., Dutra, D. C., Page, D., & Costa, S. (n.d.). Mode Directed Path Finding.

[17] Sowa, J. F. (1984). *Conceptual structures: information processing in mind and machine*. Addison-Wesley Longman Publishing Co., Inc...

[18] Tan, P. N., & Steinbach, M. (2006). Vipin Kumar, Introduction to Data Mining.

[19] Goutte, C., Gaussier, E.: A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: ECIR'05: Proceedings of the 27th European Conference on Information Retrieval, Springer (2005) 345-359

[20] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

[21] Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

[22] Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81-106.

[23] 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.

[24] S. D. Toprak, P.Senkul, Y.Kavurucu, and I. H. Toroslu. A new ILP-based concept discovery method for business intelligence. In ICDE *Workshop on Data Mining and Business Intelligence,* April 2007.

[25] S. D. Toprak. A new hybrid multi-relational data mining technique. Master's thesis, Middle East Technical University, Computer Engineering Department, Ankara, Turkey, May 2005.

[26] Iba, W., & Langley, P. (1992, July). Induction of One-Level Decision Trees. In*ML* (pp. 233-240).

[27] Quinlan, J. R. (2006). Learning Logical Definitions from Relations, 266(1990), 239–266.

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

[29] Yoshida, K. I., & Motoda, H. (1995). CLIP: Concept learning from inference patterns. *Artificial Intelligence*, *75*(1), 63-92.

[30] Santos, J. C., Tamaddoni-Nezhad, A., & Muggleton, S. (2009). An ILP system for learning head output connected predicates. In *Progress in Artificial Intelligence* (pp. 150-159). Springer Berlin Heidelberg.

[31] Srinivasan, A., King, R. D., Muggleton, S. H., & Sternberg, M. J. E. (n.d.). Carcinogenesis Predictions Using ILP.

[32] Džeroskil, S., Jacobs, N., Molina, M., & Moure, C. (1998). *ILP experiments in detecting traffic problems* (pp. 61-66). Springer Berlin Heidelberg.

[33] Van Assche, A., Vens, C., Blockeel, H., & Džeroski, S. (2006). First order random forests: Learning relational classifiers with complex aggregates.*Machine Learning*, *64*(1-3), 149-182.

[34] Chang Chien, Y. W., & Chen, Y. L. (2009). A phenotypic genetic algorithm for inductive logic programming. *Expert Systems with Applications*, *36*(3), 6935-6944.

[35] Dolšak, B. (2002). Finite element mesh design expert system. *Knowledge-based systems*, *15*(5), 315-322.

[36] Doncescu, A., Waissman, J., Richard, G., & Roux, G. (2002). Characterization of bio-chemical signals by inductive logic programming. *Knowledge-Based Systems*, *15*(1), 129-137.

[37] Dehaspe, L., & De Raedt, L. (1997). Mining association rules in multiple relations. In *Inductive Logic Programming* (pp. 125-132). Springer Berlin Heidelberg.

[38] 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.

[39] Fonseca, N., Costa, V. S., Silva, F., & Camacho, R. (2003). *On the implementation of an ilp system with prolog*. Technical report, DCC-FC & LIACC, UP.

[40] Zhao, Y., & Zhang, Y. (2008). Comparison of decision tree methods for finding active objects. *Advances in Space Research*, *41*(12), 1955-1959.

[41] Gonzalez, J. A., Holder, L. B., & Cook, D. J. (1998). Graph-Based Concept Learning, (Valiant 1985).

[42] Gonzalez, J. A., Holder, L. B., & Cook, D. J. (2000, May). Structural Knowledge Discovery Used to Analyze Earthquake Activity. In *FLAIRS Conference* (pp. 86-90).

[43] Djoko, S., Cook, D. J., & Holder, L. B. (1995, August). Analyzing the Benefits of Domain Knowledge in Substructure Discovery. In *KDD* (pp. 75-80).

[44] Su, S., Cook, D. J., & Holder, L. B. (1999). Application of Knowledge Discovery to Molecular Biology: Identifying Structural Regularities in Proteins. In *Pacific Symposium on Biocomputing* (Vol. 4, pp. 190-201).

[45] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific Publishing Company, 1989.

[46] C. Goutte and E. Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. *In European Colloquium on IR Research (ECIR'05),* pages 345-359, Xerox Research Centre Europe 6, chemin de Maupertuis F-38240 Meylan, France, 2005. Springer.

[47] Wrobel, S. (2001). Inductive logic programming for knowledge discovery in databases. In *Relational data mining* (pp. 74-101). Springer Berlin Heidelberg.

[48] Gao, Z., Zhang, Z., & Huang, Z. (2009, April). Extensions to the Relational Paths Based Learning Approach RPBL. In *ACIIDS* (pp. 214-219).

[49] Inokuchi, A., Washio, T., & Motoda, H. (2000). An apriori-based algorithm for mining frequent substructures from graph data. In *Principles of Data Mining and Knowledge Discovery* (pp. 13-23). Springer Berlin Heidelberg.

[50] Fonseca, N., Rocha, R., Camacho, R., & Silva, F. (2003). Efficient data structures for inductive logic programming. In *Inductive Logic Programming* (pp. 130-145). Springer Berlin Heidelberg.

[51] Muggleton, S. (1992). The Application of Inductive Logic Programming to Finite Element Mesh Design. *Inductive Logic Programming, Academic Press, New York*.

[52] Feng, C. (1992). Inducing temporal fault dignostic rules from a qualitative model.*Inductive Logic Programming*, 473-486.

[53] King, R. D., Muggleton, S., Lewis, R. A., & Sternberg, M. J. (1992). Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proceedings of the national academy of sciences*,*89*(23), 11322-11326.

[54] Chittimoori, R. N., Holder, L. B., & Cook, D. J. (1999, May). Applying the Subdue Substructure Discovery System to the Chemical Toxicity Domain. In*FLAIRS Conference* (pp. 90-94).

[55] Cook, D. J., Holder, L. B., Su, S., Maglothin, R., & Jonyer, I. (2001). Structural mining of molecular biology data. *Engineering in Medicine and Biology Magazine, IEEE*, *20*(4), 67-74.

[56] Chittimoori, R. N., Gonzalez, J. A., & Holder, L. B. (1999, July). Structural knowledge discovery in chemical and spatio-temporal databases. In *AAAI/IAAI*(p. 959).

[57] Gonzalez Bernal, J. A. (2001). *Empirical and theoretical analysis of relational concept learning using a graph-based representation*. The University of Texas at Arlington.

[58] Srinivasan, A., Muggleton, S. H., Sternberg, M. J., & King, R. D. (1996). Theories for mutagenicity: A study in first-order and feature-based induction. *Artificial Intelligence*, *85*(1), 277-299.

[59] Dehaspe, L., Toivonen, H., & King, R. D. (1998, August). Finding Frequent Substructures in Chemical Compounds. In *KDD* (Vol. 98, p. 1998).

[60] R. Michalski and J. Larson. Inductive inference of VL decision rules. In *Workshop on Pattern-Directed Inference Systems*, volume 63, pages 33–44, Hawaii, 1997. SIGART Newsletter, ACM.

[61] Muggleton, S. (1992). The Application of Inductive Logic Programming to Finite Element Mesh Design. *Inductive Logic Programming, Academic Press, New York*.