Universität des Saarlandes
Max-Planck-Institut für Informatik
AG5

Towards Nonmonotonic Relational Learning from Knowledge Graphs

Masterarbeit im Fach Informatik
Master's Thesis in Computer Science
von / by

Hai Dang Tran

angefertigt unter der Leitung von / supervised by

Dr. Daria Stepanova

begutachtet von / reviewers

Dr. Daria Stepanova

Prof. Dr. Gerhard Weikum

Saarbrücken, June 2017

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Statement in Lieu of an Oath

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, June 2017 Hai Dang Tran

Abstract

These days witness the development of information extraction, which leads to the birth of knowledge graphs (KGs), i.e., large collections of relational facts. Since these KGs are automatically built, it is not guaranteed that they are precise and complete. Hence, Open World Assumption (OWA) should be applied to them, that is, facts which are not given can be treated as true or false. Rule mining approaches can be applied to tackle the problem of KG completion. However, the existing approaches focus on positive rules, which do not take exceptions or negated atoms into consideration and may therefore infer wrong facts. Recently a rule-based technique that bridges this gap has been proposed, but it only works on a flattened presentation of a KG, that is, a collection of unary facts. This work extends the previous results and presents an approach for mining nonmonotonic rules (i.e., rules with exceptions in the body) from KGs in the relational format. The contributions of this thesis are:

- We have introduced a framework to revise and improve quality of positive rules.
- We have proposed a method for finding exception candidates, assessing their quality and ranking them based on chosen measure with a novel concept of partial materialization.
- We have implemented a system and conducted experiments to test our methodology. The obtained results are promising and they demonstrate the effectiveness of our approach.

Acknowledgements

First and foremost, I want to give my special thanks to Dr. Daria Stepanova for her enthusiastic advisory during my master thesis and ILP paper. I feel indebted to Daria for her guidance about Logic Programming and scientific writing skills. She always fully support me to pursue my dream and I will not forget it.

I also want to say thanks to Prof. Gerhard Weikum for giving a chance to work in the Database and Information System Group. The group is a good environment for me to foster my research skills.

It is my pleasure to work with Mohamed H. Gad-Elrab in the ILP paper, I really appreciate his advice for conducting experiment and implementing system.

A sincere thanks to Thinh and Marko for helping me to develop my career path. Besides, I will remember the time that I train with Ralf and play for FC Saarbrücken Table Tennis Team III. Thanks to the team for helping me to find a new passion along with programming.

Last but not the least, I am grateful to my parents and my brother who are always besides me with an unconditional love. They are the reason why I try my best in studying and working day by day.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

These days witness the development of information extraction methods, which are fruitfully exploited for automatic construction of Knowledge Graphs (KGs). These are large collections of relational facts in the $\langle subject \rangle \langle predicate \rangle \langle object \rangle$ format. Such triples reflect facts about the real world, which can be converted to facts over unary and binary relations in predicate calculus. Every such triple can be represented as a binary fact *predicate(subject, object)* if *predicate* is not *type* and a unary fact *object(subject)* otherwise. For example, $\langle Brad \rangle \langle livesIn \rangle \langle Berlin \rangle$ can be rewritten as $livesIn(Brad, Berlin)$ while $\langle Mat \rangle \langle type \rangle \langle artist \rangle$ can be converted to $artist(Mat)$. Examples of KGs include YAGO [45], Wikidata [1], FreeBase [2], etc (see Chapter 2 for details).

Unfortunately, KGs are normally incomplete due to automatic construction. Hence, they are processed under the Open World Assumption (OWA) where missing facts can be true or false. Therefore, the *knowledge completion problem* (also known as *link prediction*) plays an important role in improving the quality of KGs. To achieve this, rule learning approaches [8, 17] are used to create rules which can infer new potentially missing facts. However, the existing methods only pay attention to positive rules, which do not take exceptions (negated atoms) into consideration and may therefore make wrong predictions. For instance, a rule:

$$r1 : livesIn(X, Z) \leftarrow isMarriedTo(X, Y), livesIn(Y, Z) \tag{1.1}$$

[1] https://www.wikidata.org
[2] https://developers.google.com/freebase/

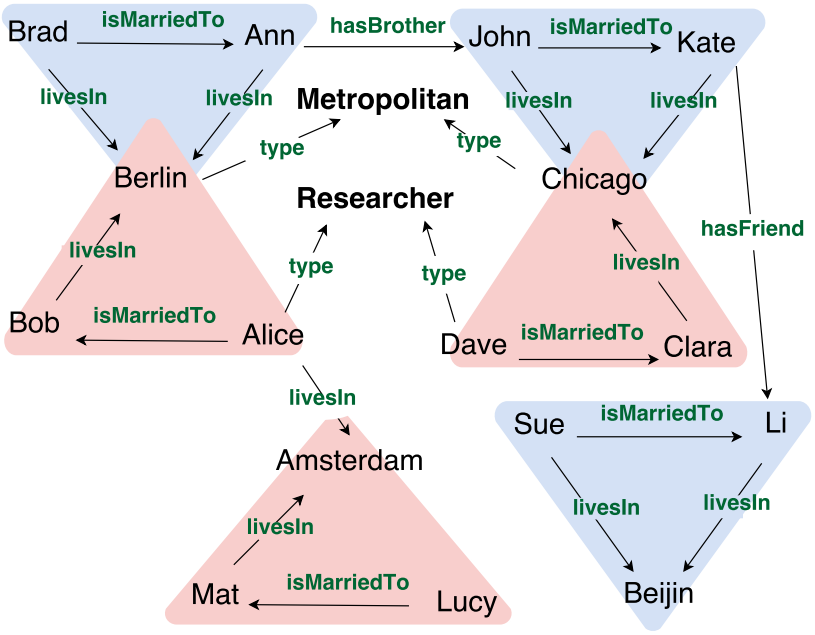

FIGURE 1.1: A Visualization of a Knowledge Graph

can be discovered from the graph in Figure 1.1 and exploited to predict new facts $livesIn(Alice, Berlin), livesIn(Dave, Chicago)$ and $livesIn(Lucy, Amsterdam)$. However, in reality the first two facts might be incorrect, since *Alice* and *Dave* are researchers and the rule $r1$ might have researcher as an exception. Exceptions should be taken into account in rule learning to improve both the rule quality and the quality of facts it produces.

1.2 State of the Art and Its Limitations

Nonmonotonic rule learning [40, 21, 43, 9, 26] focuses on discovering a set of rules with exceptions (negated atoms) in their bodies (see Chapter 3 for details). However, the state-of-the-art nonmonotonic ILP algorithms cannot be directly applied to our problem due to the following reasons:

- The *target relations* cannot be explicitly defined because we do not know which parts of the graph need to be extended. A naive solution to this issue is to mine all possible rules for the available predicates in the KG. However, this solution is computationally expensive due to the large number of facts in the original KGs.
- Second, ILP systems usually exploit positive and *negative examples*. While positive examples are available in our work, negative ones are not given. Besides, the latter are difficult to collect because we cannot differentiate between wrong and unseen

triples. Thus, a simple approach to overcome this is to discover rules from solely positive facts.

- Finally, the *language bias* is not easy to define since the schema of the original data is not given.

To overcome the above obstacles, casting our problem into an exploratory data analysis is a suitable solution. The authors in [16] propose association rule mining methods to explore Horn positive rules, and then revise them by inserting exceptions or negated atoms in their bodies to improve the predictive quality of the rules. However, this approach only works on a flattened presentation of a KG, i.e., a collection of unary facts.

1.3 Goals

The aim of this thesis is to:

- Propose a theory to explore interesting and informative nonmonotonic rules.
- Build a scalable system in terms of run time to generate rules with exceptions from a large collections of facts.
- Extensively evaluate the developed system by performing experiments on the real world KGs.

1.4 Contributions

This thesis is an extension of the work [16] to handle KGs in their original form. More specifically, we want to mine rules with exceptions from a set of relational facts in KGs treated under the OWA. We transform the knowledge completion problem into *theory revision* task, where given a KG and a set of previously learned Horn rules, the goal is to revise Horn rules to nonmonotonic ones by introducing exceptions. The positive rules can be found by using off-the-shell tools for association rule learning. The revision should be done with the purpose of improving the quality of the revised rules compared to their original versions. To estimate the quality, we exploit the conviction measure [6].

Our approach consists of four steps. First, for each positive rule, we try to find normal and abnormal sets, that is, set of instances that follow and do not follow a given rule, respectively. Second, we mine exception witness sets, that is, set of unary or binary

relations that can eliminate abnormal instances. For instance, *Researcher* is a positive exception for the rule $r1$ based on the KG in Figure 1.1. Third, we add exceptions to the Horn rules in the form of a single negated atom and define a measure to assess the quality of the obtained revisions. Importantly, the interaction between revised rules is taken into consideration in our approach by a novel concept of *partial materialization*. Finally, exceptions are ranked based on the proposed measures, and the best revision is selected as the final solution. The contributions of this thesis can be summarized as follows:

- We introduce a framework to revise and improve quality of positive rules by incorporating negated atoms into their bodies.
- We propose a method for finding exception candidates, assessing their quality and ranking them based on the defined measures. With the novel concept of *partial materialization*, the cross-talk between the rules is taken into account during the revision process.
- We implement a system that mines Horn rules from a KG and enriches them with exceptions.
- We construct benchmarks and conduct experiments with YAGO3 and IMDB datasets to test the above-mentioned methodology and the partial materialization concept.

1.5 Structure

The outline of this thesis is as follows. Chapter 2 describes related work and presents the literature review. Chapter 3 provides background and preliminaries for nonmonotonic logic programs and relational association rule learning. Chapter 4 presents the rule learning problem that we are tackling and describes our methodology. Chapter 5 contains the system overview, implementation details and optimization strategies. Chapter 6 describes the benchmark construction process, experimental results, their analysis and interpretation. Finally, Chapter 7 concludes the thesis and outlines approach directions.

Chapter 2

Related Work

The area of relational learning in the context of Semantic Web has recently attracted interest from many researchers. The approaches that aim at tackling this problem can be mainly classified into three groups: statistics-based, text-based and logic-based. In this chapter we review Semantic Web and the relevant works from these groups.

2.1 Semantic Web

The Semantic Web [4] is a paradigm that enables computers to interpret the meaning of the data from the Internet. Alternatively, it is defined as "a web of data that can be processed directly and indirectly by machines".

The World Wide Web Consortium (W3C) defines the Semantic Web and the Resource Description Framework (RDF) as a platform for Linked Data [4]. The W3C proposes other technologies such as SPARQL and OWL for data manipulation. Architecture of the W3C is presented as a stack in Figure 2.1 [1].

The RDF is widely used for representing KGs and this is the fundamental format for many KGs such as YAGO [45], FreeBase [2], Wikidata[3], etc. There are several prominent examples of KGs and their sizes in Table 2.1 [36]. Google's Knowledge Graph is the largest among all the KGs mentioned in the table with 18 billions facts.

Though having different number of predicates, entities and facts, these graphs share the following common properties [36]:

[1] https://www.w3.org/DesignIssues/diagrams/sweb-stack/2006a.png
[2] https://developers.google.com/freebase/
[3] https://www.wikidata.org

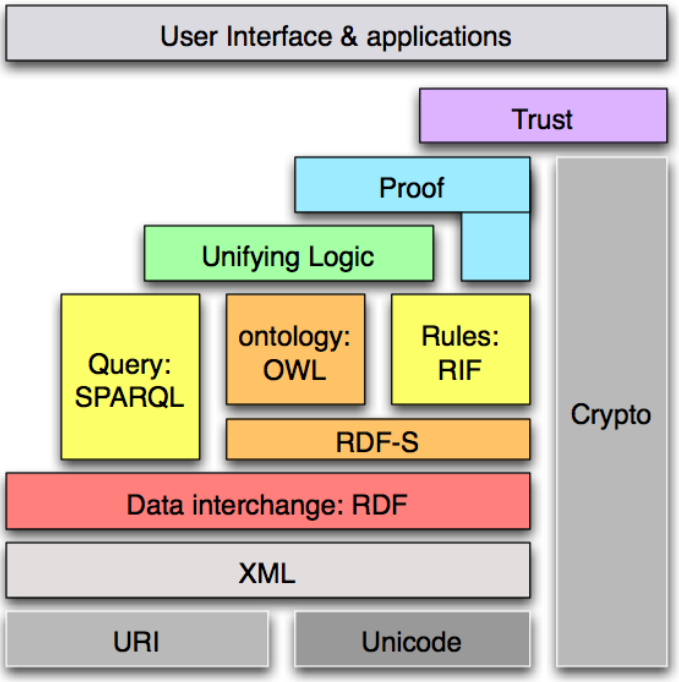

FIGURE 2.1: The Semantic Web Technologies

Name	Instances	Facts	Types	Relations
DBpedia (English)	4,806,150	176,043,129	735	2,813
YAGO	4,595,906	25,946,870	488,469	77
Freebase	49,947,845	3,041,722,635	26,507	37,781
Wikidata	15,602,060	65,993,797	23,157	1,673
NELL	2,006,896	432,845	285	425
OpenCyc	118,499	2,413,894	45,153	18,526
Google's Knowledge Graph	570,000,000	18,000,000,000	1,500	35,000
Google's Knowledge Vault	45,000,000	271,000,000	1,100	4,469
Yahoo! Knowledge Graph	3,443,743	1,391,054,990	250	800
IMDB	117,551	134,639	17,049	39

TABLE 2.1: List of Typical Knowledge Graphs

- KGs reflect facts from real world.
- KGs cover different domains, for example, YAGO is a general purpose KG while IMDB focuses on the movie domain.

2.2 Statistics-based Approaches for Relational Learning

Relational Learning is a research field that focuses on "representing, reasoning and learning in domains with complex relational structure" [19]. Many frameworks in this

field can be fruitfully applied in the Semantic Web context. In this section, we discuss Relational Learning from statistics view. The statistics-based approaches focus on building models with latent features which are not directly observable from the original data [33]. The core idea is to infer correlation between objects based on selected hidden features.

RESCAL is one of principal algorithms in this direction where relations of all hidden feature pairs are taken into consideration to measure likelihood of true facts [34, 35]. This method is extended to classical tensor decomposition algorithm [25] and neural tensor network [44].

While many tensor factorization methods use subject, predicate and object as three dimensions to model a KG as a cube, some algorithms transform this cube into two dimensional data in a preprocessing step. For instance, in [22, 41] subject and object in every triple are merged into a single element representing a pair of entities.

Distance-based models use the intuition that entities have high chance to be related to each other if their hidden representation features are close to each other. The closeness can be checked based on some predefined distance measures. This approach is expanded to structure embedding models in [5].

2.3 Logic-based Approaches for Relational Learning

The logic-based approaches aim at finding observable patterns in order to infer new links in the KG [33]. Since the patterns can be directly seen in the data, these algorithms are more interpretable than the ones based on statistical approaches. For instance, a pattern extracted from the graph in Figure 1.1 can be represented as rule 1.1 ($r1$). This rule reflects a triangular pattern in the KG, and since the rules are extracted from the graph, it should be vice versa, i.e., predicates correspond to the labels. Because this pattern is frequently observed, it can be extracted to predict new links in the KG. For instance, if we apply the rule $r1$ to a KG in Figure 1.1, we can deduce the facts *livesIn(Lucy, Amsterdam)* and *livesIn(Dave, Chicago)*.

In the rest of this section, we review some of the most prominent logic-based relational learning systems in details and illustrate them with typical tools.

2.3.1 Inductive Logic Programming Systems

Inductive Logic Programming [32] (ILP) is a combination of Machine Learning and Logic Programming fields whose goals is to generate hypothesis from background knowledge

and specific examples, which are normally classified into positive and negative ones. The aim is to find a hypothesis that covers all positive examples and none of the negative ones.

Rule mining is a core problem in ILP, however, applying ILP algorithms to Semantic Web data is problematic for the following reasons. First, ILP tools are not scalable for large knowlege graphs such as YAGO, Freebase, Wikidata [17]. In some experiments conducted in [17], it has been shown that the cutting edge ILP tools, for example, ALEPH [31, 17], QuickFoil [50, 17] require several days to process the YAGO KG.

Second, ILP methods use closed world assumption (CWA), under which a given KG is assumed to be complete and missing facts are treated as false rather than unknown. ILP methods require both positive and negative examples like traditional machine learning algorithms. One cannot rely on CWA in our problem since in the KGs the positive examples are incomplete and negative ones are unknown [17].

Third, most ILP algorithms generate positive Horn rules without exceptions [40]. These rules often have low precision, one of the possible ways to improve the quality of the facts the rules predict is to add exceptions. Therefore, mining exceptions from rules is an important task which we study in this work. In the following paragraphs, some popular ILP systems are described in details.

ALEPH. This name stands for A Learning Engine for Proposing Hypotheses which is one of the typical systems in ILP. The tool is developed and extended from P-Progol [4]. Like many other ILP systems, as input ALEPH receives the background knowledge and sets of positive and negative examples. After that, it generates a set of rules as a hypothesis set. More specifically, the tool processes data in the following steps:

- ALEPH picks an example to hypothesize and terminates if no such example is found.
- Then the most specific rule that infers the example is computed.
- Among all subsets, i.e., combinations of atoms in the most specific rule, the one with the highest score is added to the hypothesis set.
- All examples inferred by the chosen hypothesis are removed from the data and the process starts again from the first step.

Though this algorithm is among the most popular ones in ILP, experiments in [17] show that its run time is extremely slow. This demonstrates the unsuitability of ALEPH for large datasets.

[4] http://www.cs.ox.ac.uk/activities/machinelearning/Aleph/aleph

QuickFOIL. To address the scalability issues of ILP systems, an optimized version of FOIL [38], i.e., the system QuickFOIL [51] has been developed with two following steps:

- First, QuickFOIL produces a new Horn rule that maximizes the difference between the number of covered positive and negative examples in the training data. This criteria is different from traditional ILP where among the rules that do not infer negative examples, the one covering highest number of positive examples is chosen.
- Second, whenever the chosen rule covers an example, the latter is removed from the set of positive ones [17]. Intuitively, the collection of found rules at each step can be used to summarize original dataset. QuickFOIL uses pruning techniques where duplicates (i.e., rules are equivalent to ones generated before) are removed to process large-scale input with millions of facts.

2.3.2 Relational Learning Systems

Relational Learning systems do not require negative examples in the data, and they also work well under the Open World Assumption (OWA). This makes them attractive for our setting. In the following paragraphs, some Relational Learning tools are described in details.

WARMR. The WARMR [13, 14] system represents a combination of traditional ILP and Relational Learning approaches. It extends APRIORI algorithm [2] to the relational setting and exploits a pruning technique in level-wise search to mine patterns. More specifically, at each level, infrequent queries are removed and the remaining frequent ones are expanded using the above operation. New queries are deleted if they can be expanded from an infrequent pattern in the previous level. This step is repeated until we cannot generate new queries. To check the runtime performance of WARMR, the authors of AMIE+ [17] test it on the YAGO dataset. The tool processes more than a day with YAGO2. WARMR is developed with ProLog, and thus, this makes it slow in working with large datasets [17].

AMIE(+). The scalability issues as well as CWA assumption issues of ILP are tackled in the AMIE(+) [17] system. This tool receives a large KG as input and generates top positive Horn rules ranked based on a certain association rule score. Initially, AMIE begins with a list of rules with empty body and a binary head predicate. After that, the rules are expanded with additional relations and variables using three mining operators. These operators are executed by query manipulation to explore new predicates and

instances. As regards the implementation, AMIE uses in-memory database with fact indexes to run select and existence queries.

AMIE+ is an optimized version of AMIE. The authors introduce three main pruning strategies: maximum length constraint, quality condition and simplifying query. More specifically, in the first strategy, a rule is not expanded if its length reaches a threshold. In the second strategy, if confidence of a rule is already 100%, the rule cannot be better and does not need to be extended. In the final strategy, approximation techniques are applied to assess the confidence of a given rule. Experimental work in [17] shows that this platform can process millions of triples in RDF graph and it surpasses other methods in terms of the run time.

RDF2Rules. The RDF2Rules [46] focuses on mining cyclic patterns from KGs, which are postprocessed and cast into rules. This tool improves rule quality by inserting entity types to their bodies, for example, $diedIn(X,Y) \leftarrow wasBornIn(X,Y), people(X), country(Y)$ is expected to be more accurate than $diedIn(X,Y) \leftarrow wasBornIn(X,Y)$. Furthermore, they introduce a new confidence measure that exploits types of entities to verify rule quality. Experiments show that this tool is effective, especially with rules containing entity types, and it often outperforms AMIE+. However, the form of the mined rule is restricted to cycles, which makes the system more restrictive than the AMIE+ system.

Like other ILP platforms, Relational Learning tools only focus on positive (Horn) rules and they do not learn nonmonotonic rules, i.e., rules with exceptions. In this work, we aim at bridging this gap and develop method to learn the latter types of rules.

2.3.3 Nonmonotonic Rule Mining Systems

Rules with exceptions are traditionally learned by nonmonotonic rule learning systems. The research work in [9] is devoted to the problem of learning nonmonotonic rules. The idea behind is to convert an ILP task into an ALP [24] instance, and then, the way the ALP instance is solved can be mapped to the ILP one 's solutions. However, this system adopts CWA, and therefore cannot be applied in our work. There is another nonmonotonic rule learning method that accounts for incomplete data [23] where a coordination of Semantic Web ontology and positive/negative rules is exploited to explore patterns. The work [23] pays attention to complicated communication between different reasoning components. This is different from the current work where we focus on the quality of the predicted data.

	bornInUS	livesInUS	immigratesToUS	livesInUK
s1	1	1	0	0
s2	1	0	1	1
s3	1	1	0	1
s4	0	0	1	0
s5	1	0	0	1
s6	0	0	1	1
s7	0	0	1	1
s8	1	0	0	0
s9	1	1	1	1
s10	0	1	0	1

TABLE 2.2: Transaction Table as Flattened Knowledge Graph Data.

Nonmonotonic Rule Mining under OWA over the flattened data. The work [16] made some steps towards learning nonmonotonic programs from incomplete KGs. However, the developed methods work only on flattened KGs, i.e., the graphs in which all RDF triples are converted to unary facts by concatenating predicates and objects. More specifically, a binary fact such as *bornIn(s1, US)* is transformed to a fact in the unary form *bornInUS(s1)*. This way, a KG is converted to a binary transaction table (for example, see Table 2.2) where 1 (resp., 0) appears in the intersection of a column c and a row r if the pact $c(r)$ is present (resp., not present) in the KG. For instance, Table 2.2 reflects that *bornInUS(s1)* is included in the KG while *livesInUS(s6)* is not. This representation allows one to apply highly optimized Item Set Mining algorithms [49] to extract data patterns which can then be used to create positive association rules [20].

In the next step, the exceptions for each rule are found and ranked based on several score measures and innovative concept of Partial Materialization (PM) [16]. The idea behind the is to enable the interaction between rules. A difference between this work and [16] stands from the format of the data. While in [16] binary facts are translated into unary ones, in this thesis we keep the facts in their natural relational format.

2.4 Text-based Approaches for Relational Learning

Both statistics-based and logic-based approaches are internal methods, i.e., only data inside the graph is used to infer new relations. We now discuss other approaches that require data outside the KG, i.e., web pages linking to objects or big collection of documents.

Wikipedia pages can be used to identify relations between entities [48]. On a larger scale, an algorithm in [47] is developed to learn lexical predicate patterns, for example,

isMarriedTo predicate in Figure 1.1 can be expressed as "married to", "engaged with", etc in some documents. Then these patterns are searched all over the Internet to find subject-object pairs corresponding to them. These new pairs can be added to the original graph. For example, in [47] a big text corpus is used to learn relations.

In [30] the authors refine KGs by exploiting the assumption that entities appearing in the same table should have the same relations. Similarly, [37] and [42] exploit page lists and HTML tables, resp.

Instead of documents, interlinks between the same entities in different KGs can be used to add new facts [1, 7]. More specifically, relations of two entities in FreeBase can be inserted into YAGO if they also appear in the latter KG. In [15] this approach has been extended to a probabilistic setting, where for every deduced relation a corresponding probabilistic weight is computed.

Chapter 3

Background

In this chapter, we introduce some preliminary knowledge for the rest of the thesis with the following organization. First, nonmonotonic logic programs under answer set semantics and KG completion problem are presented. Second, we address some definitions of relational association rule mining such as *head support, absolute support, confidence* and *conviction*. These definitions are exploited in the main part of this work (Chapter 4 and 5).

3.1 Nonmonotonic Logic Program

In the current work, we rely on the standard definitions of logic programs [28]. Formally, *nonmonotonic logic program* P is a rule set where each rule has the form:

$$r : H \leftarrow B, notE \tag{3.1}$$

In details, H stands for *head(r)* which is a head of the rule r, i.e., a first-order atom in the format $a(\boldsymbol{X})$. Besides, *B, not E* is a conjunction: $b_1(\boldsymbol{Y}_1)$, $b_2(\boldsymbol{Y}_2)$, ..., $b_k(\boldsymbol{Y}_k)$ and *not* $b_{k+1}(\boldsymbol{Y}_{k+1})$, *not* $b_{k+2}(\boldsymbol{Y}_{k+2})$, ..., *not* $b_n(\boldsymbol{Y}_n)$, resp. B and *not* E are used as a short form of $body^+(r)$ and $body^-(r)$, resp. The *not* operator is called the *negation as failure (NAF) or default negation*. If $body^-(r) = \emptyset$ then r is a positive (Horn) rule. $\boldsymbol{X}$, $\boldsymbol{Y}_1$, $\boldsymbol{Y}_2$, ..., $\boldsymbol{Y}_n$ are tuples of arguments, i.e., variables and/or constants and their sizes are the arity of a, b_1, b_2, ..., b_n, resp. The signature of the program P is denoted as $\Sigma_P = \langle \mathbf{P}, \mathcal{C} \rangle$ where $\mathbf{P}$, $\mathcal{C}$ are the sets of predicates and constants in P, resp.

A logic program P is *ground* if it does not contain any variables, i.e., only constants and predicates can appear in each rule r. For a non ground program P, $Gr(P)$ is a ground instantiation of P which is obtained by replacing variables with constants in all possible ways. The *Herbrand Universe HU(P)* of P is the set of constants $\mathcal{C}$ appearing in the program P and *Herbrand Base HB(P)* contains all possible ground atoms constructed by predicates in P and constants in $\mathcal{C}$, resp. Any subset of $HB(P)$ is a *Herbrand Interpretation* of a program P. An interpretation I is a model of a rule r if for every possible substitution of variables with constants for which $body^+(r), body^-(r)$ are true, $head(r)$ is also true w.r.t. I. I is defined as a model of a program P if it satisfies all rules in P. In addition, $MM(P)$ denotes a set of a subset-inclusion minimal models of P.

An *answer set* I of P is a Herbrand interpretation of P s.t. $I \in MM(P^I)$. Here, P^I denotes the Gelfond-Lifschitz (GL) reduct [18] of P which is generated by deleting any rule r s.t. $body^-(r)$ intersects with I and then removing all NAFs in the rest of the rules. $AS(P)$ stands for the set of all answer sets for P.

Example 3.1. *Consider the following nonmonotonic program as an example:*

$$P = \left\{ \begin{array}{l} \textit{(1) livesIn}(\textit{brad}, \textit{berlin}); \ \textit{(2) isMarriedTo}(\textit{brad}, \textit{ann}); \\ \textit{(3) livesIn}(Y, Z) \leftarrow \textit{isMarriedTo}(X, Y), \textit{livesIn}(X, Z), \textit{not} \quad \textit{researcher}(Y) \end{array} \right\}$$

We obtain the ground instantiation $Gr(P)$ of P by replacing X, Y, Z with brad, ann and berlin, resp. Consider the interpretation $I = \{$isMarriedTo(brad,ann), livesIn(ann,berlin), livesIn(brad,berlin)$\}$. Based on the above definition, the GL-reduct P^I of P consists of a rule livesIn(ann,berlin) $\leftarrow$ livesIn(brad,berlin), isMarriedTo(brad,ann) and the ground terms (1), (2). Since I is a minimal model of P^I, by definition, we have that $I \in AS(P)$. □

3.2 Knowledge Graph Completion

In this section, we provide a definition of a KG completion problem based on the above concept of Answer Set Programming. The factual representation of a KG $\mathcal{G}$ is the set of ground atoms over the signature $\Sigma_{\mathcal{G}} = \langle \mathbf{C}, \mathbf{R}, \mathcal{C} \rangle$, in which $\mathbf{C}$, $\mathbf{R}$ and $\mathcal{C}$ denote the sets of unary predicates, binary predicates and constants, resp. By $\mathcal{G}^i$, we denote a KG that contains all correct facts with predicates and constants from $\Sigma_{\mathcal{G}^a}$ that are true in the real world. Based on [10], the gap between the *available graph* $\mathcal{G}^a$ and $\mathcal{G}^i$ is defined as follows.

Definition 3.1 (Incomplete data source)**.** A pair $G = (\mathcal{G}^a, \mathcal{G}^i)$ of two KGs is an incomplete data source, where $\mathcal{G}^a \subseteq \mathcal{G}^i$ and $\Sigma_{\mathcal{G}^a} = \Sigma_{\mathcal{G}^i}$.

We aim at learning a nonmonotonic rule set $\mathcal{R}$ from the $\mathcal{G}^a$ s.t. application of $\mathcal{R}$ to $\mathcal{G}^a$ results in a good approximation of $\mathcal{G}^i$. Application of $\mathcal{R}$ to $\mathcal{G}^a$ corresponds to the

calculation of $I \in AS(\mathcal{R} \cup \mathcal{G})$. More specifically, we define the rule based KG completion as follows:

Definition 3.2 (Rule-based KG completion)**.** Given a factual representation of a KG $\mathcal{G}$ over the signature $\Sigma_{\mathcal{G}} = \langle \mathbf{C}, \mathbf{R}, \mathcal{C} \rangle$ and a set of rules $\mathcal{R}$ mined from $\mathcal{G}$. The *completion of* $\mathcal{G}$ *w.r.t.* $\mathcal{R}$ is a graph $\mathcal{G}_{\mathcal{R}}$ constructed from any answer set in $AS(\mathcal{R} \cup \mathcal{G})$.

3.3 Association Rule Mining in Relational Setting

Association rule mining concerns the extraction of frequent patterns from the data and their subsequent casting into rules. Originally, association rules were studied in the market basket context, where interesting relations between products from transaction database of customer purchases were extracted, e.g., $\{onions,\ potatoes\} => \{burger\}$ [52]. Recently, association rule learning methods were extended to relational settings, which attracts research interests of both ILP [11] and KG [17] scientists. In the following description, we present typical rule measures for association rules in the relational setting.

A *conjunctive query* Q w.r.t $\mathcal{G}$ is the expression of the form $Q(\vec{X}) : -p_1(\vec{X_1}), \ldots, p_m(\vec{X_m})$. The body (i.e., right part) of the query is a list of positive or negative atoms over $\mathcal{G}$. The head (i.e., left part) is a tuple of variables appearing in the right part. The *answer* of Q w.r.t $\mathcal{G}$ is defined as a set $Q(\mathcal{G}) := \{\text{substitutions } \theta \text{ of } \vec{X}\text{: } p_i(\vec{X_i}\theta) \in \mathcal{G} \text{ with } i \in [1..m]\}$. Based on [12], the *(absolute) support* of a conjunctive query Q w.r.t the KG $\mathcal{G}$ is the cardinality of the set $Q(\mathcal{G})$. For instance, the query

$$Q(X, Y, Z) : -isMarriedTo(X, Y), livesIn(Y, Z) \tag{3.2}$$

has the absolute support 6 w.r.t $\mathcal{G}$ in Figure 1.1 since there are 6 substitutions for the triple $\langle X, Y, Z \rangle$ that satisfy *isMarriedTo(X,Y), livesIn(Y,Z)* w.r.t $\mathcal{G}$.

An *association rule* is of the format $Q_1 => Q_2$, where Q_1 and Q_2 are conjunctive queries and all atoms in the body of Q_1 also appear in that of Q_2. For instance, from $Q(X, Y, Z)$ in the above example and

$$Q'(X, Y, Z) : -isMarriedTo(X, Y),\ livesIn(X, Z), livesIn(Y, Z) \tag{3.3}$$

the association rule $Q => Q'$ can be built.

In the current work, association rules are exploited with the aim to predict new facts, hence, they should be translated into logical format. More specifically, we convert the association rule $Q_1 => Q_2$ to the logical one $Q_2 \backslash Q_1 \leftarrow Q_1$, where $Q_2 \backslash Q_1$ is the set of

atoms which are in Q_2 but not in Q_1. For example, it can be seen that the above rule $Q => Q'$ can be transformed to *r1* in Section 1.

In this work, the conviction measure [6] is exploited for estimating the rule quality, since it is guaranteed to have high predictive power [3]. Hence, this measure is useful for KG completion introduced in Chapter 4. With the rule $r: \ H \leftarrow B, not \ \ E$, where $H = h(X, Y)$ and B, E contain a set of variables $\vec{Z} \supseteq X, Y$, we can find the *conviction* by the following formula:

$$conv(r, \mathcal{G}) = \frac{1 - supp(h(X, Y), \mathcal{G})}{1 - conf(r, \mathcal{G})} \tag{3.4}$$

in which $supp(h(X, Y), \mathcal{G})$ stands for *relative support* of the head $h(X, Y)$, defined as:

$$supp(h(X,Y), \mathcal{G}) = \frac{\#(X,Y) : h(X,Y) \in \mathcal{G}}{(\#X : \exists Y \ h(X,Y) \in \mathcal{G}) * (\#Y : \exists X \ h(X,Y) \in \mathcal{G})} \tag{3.5}$$

besides, *conf* denotes the *confidence* of r:

$$conf(r, \mathcal{G}) = \frac{\#(X,Y) : H \in \mathcal{G}, \exists \vec{Z} \ B \in \mathcal{G}, E \notin \mathcal{G}}{\#(X,Y) : \exists \vec{Z} \ B \in \mathcal{G}, E \notin \mathcal{G}} \tag{3.6}$$

Example 3.2. *Applying these definitions to the KG in Figure 1.1, we obtain the following results. Given a KG $\mathcal{G}$ in Figure 1.1 and a rule $r1$ in the form 1.1, there are three ways to substitute variables in $r1$ based on $\mathcal{G}$. Thus, the absolute support of $r1$ w.r.t. $\mathcal{G}$ is 3. Besides, the relative head support of predicates livesIn and hasFriend are $supp(livesIn(X, Z), \mathcal{G}) = \frac{9}{8 \times 4} \approx 0.3$ and $supp(hasFriend(X, Z), \mathcal{G}) = \frac{1}{1 \times 1} = 1.0$, resp. The confidence of rule $r1$ is $conf(r1, \mathcal{G}) = \frac{2}{5} = 0.4$, thus, the conviction measure is as follows $conv(r1, \mathcal{G}) = \frac{1 - 0.3}{1 - 0.4} = \frac{7}{6} = 1.17$*

Chapter 4

A Theory Revision Framework for Rule-based KG Completion

This chapter discusses the theoretical framework for learning nonmonotonic rules from KGs. First, the problem statement is formally defined with the introduction of theory revision and KG completion concepts. Second, we propose our methodology for tackling this problem. The theoretical developments that we present are realized within the RUMIS system described in Chapter 5.

4.1 Problem Statement

This section formally presents the problem statement and the main goal of the thesis. Assume that $\mathcal{G}^i$ is an ideal completion of the available KG $\mathcal{G}$, i.e., a KG that contains every true fact over the signature $\Sigma_{\mathcal{G}}$. As input, we are given an available incomplete KG $\mathcal{G}$ and a set $\mathcal{R}_H$ of positive rules mined from $\mathcal{G}$. Our aim is to insert NAF atoms (exceptions) to the positive rules to obtain a nonmonotonic ruleset $\mathcal{R}_{NM}$ s.t. the difference between $\mathcal{G}_{\mathcal{R}_{NM}}$ and $\mathcal{G}^i$ is minimized. The $\mathcal{R}_{NM}$ is a good choice for the rule revision of $\mathcal{R}_H$ if it deletes many incorrectly predicted facts from $\mathcal{R}_H$, and still retains many true ones from $\mathcal{R}_H$.

Obviously, $\mathcal{G}^i$ is *unavailable*, hence the truthfulness of predictions cannot be estimated. Consequently, standard ILP measures that rely on CWA cannot be exploited in our work to assess how good a rule revision is. To tackle this issue, we propose to use measures from association rule mining introduced in the previous chapter. Based on our theory revision framework, a ruleset revision is good if its total rule measure is as high

as possible, and the inserted NAFs are not over-fitting the KG, i.e., they are relevant exceptions rather than noise.

To fulfill these constraints, we introduce two functions for assessing the quality, q_{rm} and $q_{conflict}$, that take a set of rules $\mathcal{R}$ and a KG $\mathcal{G}$ and generate a value which leverages the quality of $\mathcal{R}$. More specifically, q_{rm} is defined as follows:

$$q_{rm}(\mathcal{R}, \mathcal{G}) = \frac{\sum_{r \in \mathcal{R}} rm(r, \mathcal{G})}{|\mathcal{R}|}, \tag{4.1}$$

where rm is a standard association rule measure, while $q_{conflict}$ calculates the quantity of conflicting facts predicted by applying the rules in $\mathcal{R}$ to $\mathcal{G}$. To find $q_{conflict}$, the given rule set in extended with additional auxiliary rules $\mathcal{R}_{aux}$ constructed for every revised rule $r \in \mathcal{R}$ by removing *not* from $body^-(r)$, and subsequently substituting the head predicate h of r by a dummy predicate not_h. The newly created predicate not_h intuitively corresponds to the negation of h. The formula of $q_{conflict}$ is then defined as follows.

$$q_{conflict}(\mathcal{R}, \mathcal{G}) = \sum_{p \in pred(\mathcal{R})} \frac{|\{\vec{c} \,|\, p(\vec{c}), not\_p(\vec{c}) \in \mathcal{G}_{\mathcal{R}^{aux}}\}|}{|\{\vec{c} \,|\, not\_p(\vec{c}) \in \mathcal{G}_{\mathcal{R}^{aux}}\}|}, \tag{4.2}$$

where $pred(\mathcal{R})$ is the set of relations occurring in $\mathcal{R}$, and $\vec{c}$ contains at most two constants in $\mathcal{C}$. Intuitively, $q_{conflict}$ is created to differentiate actual exceptions from noise, by taking the interaction between the rules from $\mathcal{R}$ into consideration. The following example demonstrates this core idea.

Example 4.1. *The occurrence of the unary relation researcher is necessary to improve the quality of r1 mined from the KG in Figure 1.1 based on standard association rule measures. Indeed, thanks to this, the revision may explain why* $livesIn(Dave, Chicago)$ *and* $livesIn(Alice, Berlin)$ *do not appear in the figure. However, the addition of a new nonmonotonic rule to the rule set may spoil this. For instance, after* $livesIn(Alice, Berlin)$ *is predicted by r2:* $livesIn(X, Y) \leftarrow workIn(X, Y)$, *not* $artist(X)$, *the impact of the exception researcher is weakened.*

Now we are ready to present our revision framework based on the above quality functions.

Definition 4.1 (Quality-based Horn theory revision (QHTR))**.** Let $\mathcal{G}$ be a KG over the signature $\Sigma_{\mathcal{G}}$, $\mathcal{R}_H$ is a set of positive rules mined from $\mathcal{G}$. Besides, let q_{rm}, $q_{conflict}$ be the quality functions in 4.1 and 4.2 resp., the *quality-based Horn theory revision problem*

is to find a revision set $\mathcal{R}_{NM}$ over $\Sigma_{\mathcal{G}}$ obtained by inserting exceptions to bodies of rules in $\mathcal{R}_H$, s.t. $q_{rm}(\mathcal{R}_{NM}, \mathcal{G})$ is maximized and $q_{conflict}(\mathcal{R}_{NM}, \mathcal{G})$ is minimized.

In the rest of this section we introduce the concepts of r-(ab)normal substitutions and Exception Witness Sets (EWSs).

Definition 4.2 (r-(ab)normal substitutions)**.** Given a KG $\mathcal{G}$, let r and $\mathcal{V}$ be a positive rule learned from $\mathcal{G}$ and a variable set appearing in r, resp. For the substitutions $\theta, \theta' : \mathcal{V} \to \mathcal{C}$, we have:

- The r-normal set of substitutions is $NS(r, \mathcal{G}) = \{\theta \mid head(r)\theta, body(r)\theta \subseteq \mathcal{G}\}$.
- The r-abnormal set of substitutions is $ABS(r, \mathcal{G}) = \{\theta' \mid body(r)\theta' \subseteq \mathcal{G}\ , head(r)\theta' \notin \mathcal{G}\}$.

Example 4.2. *In Figure 1.1 the blue and red triangles indicate the* $NS(r1, \mathcal{G})$ *and* $ABS(r1, \mathcal{G})$*, resp. More specifically, the normal set is given as* $NS(r1, \mathcal{G}) = \{\theta_1, \theta_2, \theta_3\}$ *with* $\theta_1, \theta_2, \theta_3$ *being* $\{X/Brad, Y/Ann, Z/Berlin\}$, $\{X/John, Y/Kate, Z/Chicago\}$ *and* $\{X/Sue, Y/Li, Z/Beijin\}$*, resp. Similarly, we have three substitutions for the abnormal set* $ABS(r1, \mathcal{G})$*:* $\{X/Alice, Y/Bob, Z/Berlin\}$, $\{X/Dave, Y/Clara, Z/Chicago\}$ *and* $\{X/Lucy, Y/Mat, Z/Amsterdam\}$.

The intuition is that given the ideal KG $\mathcal{G}^i$, the r-(ab)normal substitutions correspond to the ground rules which are satisfied (resp. not satisfied) in $\mathcal{G}^i$. Observe that, due to the incompleteness of the data under the OWA, r-abnormal substitutions are not guaranteed to be such in the ideal graph. To differentiate the "incorrectly" and "correctly" detected substitutions in the r-abnormal set, we exploit the notions of *exception witness sets* (EWS) defined as follows.

Definition 4.3 (Exception Witness Set)**.** Given a KG $\mathcal{G}$ and a rule r extracted from it, let $\mathcal{V}$ be a set of all variables appearing in r. Let $\vec{X}$ be a variable subset of $\mathcal{V}$ and, let $EWS^+(r, \mathcal{G}, \vec{X}), EWS^-(r, \mathcal{G}, \vec{X})$ be sets of predicates $EWS^+(r, \mathcal{G}, \vec{X}) = \{e : \exists\theta \in ABS(r, \mathcal{G}) : e(\vec{X}\theta) \in \mathcal{G}\}$ and $EWS^-(r, \mathcal{G}, \vec{X}) = \{e : \exists\theta \in NS(r, \mathcal{G}) : e(\vec{X}\theta) \in \mathcal{G}\}$. Exception witness set of r w.r.t. $\mathcal{G}$ and $\vec{X}$ is a set difference $EWS(r, \mathcal{G}, \vec{X}) = EWS^+(r, \mathcal{G}, \vec{X}) \setminus EWS^-(r, \mathcal{G}, \vec{X})$. In other words, it is a set of predicates covering some substitutions in $ABS(r, \mathcal{G})$ but none in $NS(r, \mathcal{G})$.

Example 4.3. *Consider the rule* $r1$*, the KG* $\mathcal{G}$ *in Figure 1.1 and* $\mathcal{V} = \{X\}$*. Based on the Definition 4.3, we have* $EWS^+(r1, \mathcal{G}, X) = \{Researcher\}$ *and* $EWS^-(r1, \mathcal{G}, X) = \{\}$*, then* $EWS(r1, \mathcal{G}, X) = \{Researcher\}$*. Besides, we have no exception candidates for* $\mathcal{V} = \{Z\}$ *since* $EWS^+(r1, \mathcal{G}, Z) = EWS^-(r1, \mathcal{G}, Z) = \{Metropolitan\}$.

Since the binary exceptions might appear in the rules, generally the cardinality of *EWS*s can be large. Indeed, for a rule with n distinct variables, there can be up to n^2 elements in *EWS*s. In addition, a good rule revision may have many exceptions in its body, thus our QHTR problem can have exponentially many solutions. To avoid the explosion of the revision search space, in the current work, we consider only rules with a single negated atom. Handling exception combination is left for future work.

Definition 4.4 (Safely Predicted Facts)**.** Given a KG $\mathcal{G}$, a positive rule r extracted from it and all exception candidates, the safely predicted facts of r w.r.t. $\mathcal{G}$ are all facts from $\mathcal{G}_{r'} \setminus \mathcal{G}$ where r' is constructed from r by adding all exception candidates of r to its body at once.

Example 4.4. *Consider the rule* $r1$ *and the KG* $\mathcal{G}$ *in Figure 1.1, applying* $r1$ *we obtain the following set* {*livesIn(Alice, Berlin), livesIn(Dave, Chicago), livesIn(Lucy, Amstersam)*}*. The rule* $r1$ *revised with exceptions results in* {*livesIn(Lucy, Amstersam)*}*. Thus, based on Definition 4.4, the fact safely predicted by* $r1$ *w.r.t.* $\mathcal{G}$ *is* {*livesIn(Lucy, Amstersam)*}*.*

4.2 Methodology

Since the number of facts in the KG is large, there can be many candidates in the EWSs to process. Hence, finding the globally best solution for QHTR problem is not feasible. Therefore, in this work we are aiming at finding an approximately best revision. The core idea of our solution is to search for the locally best exception for each rule iteratively, while still taking into account other rules in a set. Our methodology for finding such approximate solutions comprises the following four steps.

Step 1. Given a KG $\mathcal{G}$, first, top Horn rules $\mathcal{R}_H$ are mined from $\mathcal{G}$ based on the *absolute support* measure introduced in Chapter 3. To execute this step, any top notch relational association rule learning tool can be exploited (e.g., AMIE(+), WARMR, etc). After that, the *r-(ab)normal* substitutions for all rules $r \in \mathcal{R}_H$ are computed.

Step 2 and 3. For each $r \in \mathcal{R}_H$ with the head $h(X, Y)$, the candidates in $EWS(r, \mathcal{G}, X)$, $EWS(r, \mathcal{G}, Y)$ and $EWS(r, \mathcal{G}, \langle X, Y \rangle)$ are found. First, $E^+ = \{not\_h(a, b) : \exists\theta = \{X/a, Y/b, ...\} \in ABS(r, \mathcal{G})\}$ and $E^+ = \{not\_h(a, b) : \exists\theta = \{X/c, Y/d, ...\} \in NS(r, \mathcal{G})\}$ are computed. Second, we exploit a typical ILP function $learn(E^+, E^-, \mathcal{G})$ (e.g., from [39]), which finds hypothesis containing the head $not\_h(X, Y)$ and an atomic predicate in the format $p(X), p'(Y)$ or $p''(X, Y)$ as the body. We want to find hypothesis that do not infer any elements in E^-, while inferring some element in E^+. The EWS is the collection of relations appearing in the body of the resulting hypothesis.

Second, potential revisions of each rule r in $\mathcal{R}_H$ are constructed by alternatively inserting one exception from $EWS(r)$ to the rule body. In total, with every rule r, we have $|EWS(r, \mathcal{G}, X)| + |EWS(r, \mathcal{G}, Y)| + |EWS(r, \mathcal{G}, \langle X, Y \rangle)|$ exceptions to consider.

Steps 4. For each rule, its revision candidates are ranked and the best one is added to the final result $\mathcal{R}_{NM}$. Since the KG $\mathcal{G}$ and EWSs can be very large, the search space for $\mathcal{R}_{NM}$ is huge, and subsequently, searching for the globally optimal result is not feasible in practice. Hence, we propose to gradually construct $\mathcal{R}_{NM}$ by finding the locally optimal revision r_i^j for each rule $r_i \in \mathcal{R}_H$. To this end, we propose three ranking strategies Naive, PM, OPM. While the first one processes the rules independently, the other two exploit a novel concept of *partial materialization*. More specifically, the core idea of this concept is to assess potential revisions of the rule r_i not w.r.t. $\mathcal{G}$, but w.r.t. $\mathcal{G}_{\mathcal{R}'}$ where $\mathcal{R}'$ is a set of some rules different from r_i. This approach enables communication among the rules which differs from the Naive ranking. We now describe each ranking in details.

The **Naive (N)** ranking strategy does not take the cross-talk between the rules into consideration. For each rule r_i in R_H, it selects the optimal revised rule r_i^j only based on the chosen measure rm, i.e., the revision should have the highest $rm(r_i^j, \mathcal{G})$ value to be in the solution set. Hence, the revision R_{NM} obtained by applying the Naive ranking satisfies only (i) of Definition 4.1. However, it does not take the criteria (ii) into account at all, and hence may produce noisy revisions that overfit the data.

The **Partial Materialization (PM)** ranking strategy selects the optimal revision r_i^j based on the largest value of

$$score(r_i^j, \mathcal{G}) = \frac{rm(r_i^j, \mathcal{G}_{\mathcal{R}'}) + rm(r_i^{j\,aux}, \mathcal{G}_{\mathcal{R}'})}{2} \tag{4.3}$$

where $\mathcal{R}'$ contains rules r_l' generated from each r_l in $\mathcal{R}_H \backslash r_i$ by inserting all exceptions of r_l into $body(r_l)$ at once. In other words, for every r_i, $\mathcal{G}_{\mathcal{R}'}$ contains facts from $\mathcal{G}$ and *safe predictions* of all rules from $\mathcal{R} \backslash r_i$. By injecting all exceptions into every rule, we decrease the possibility of wrong predictions.

Example 4.5. *Consider the following KG $\mathcal{G}$ and a revised set $\mathcal{R}$ where each Horn rule has one exception candidate:*

$$\mathcal{G} = \left\{ \begin{array}{l} (1)\ livesIn(ann, berlin);\ (2)\ isMarriedTo(brad, ann) \\ (3)\ artist(ann);\ (4)\ bornIn(ann, berlin) \end{array} \right\}$$

$$\mathcal{R} = \left\{ \begin{array}{l} r_1 : livesIn(X, Z) \leftarrow isMarriedTo(X, Y), livesIn(Y, Z), not \quad researcher(X) \\ r_2 : bornIn(X, Y) \leftarrow livesIn(X, Y), not \quad artist(X) \end{array} \right\}$$

Consider the rule r_2 *and a set of other rules* $\mathcal{R}' = \mathcal{R} \setminus \{r_2\} = \{r1\}$. *Based on the definition, we have* $\mathcal{G}_{\mathcal{R}'} = \mathcal{G} \cup \{livesIn(brad, berlin)\}$, $conf(r_2, \mathcal{G}_{\mathcal{R}'}) = \frac{0}{1} = 0$ *and* $conv(r_2, \mathcal{G}_{\mathcal{R}'}) = \frac{1-0}{1-0} = 1$. *Similarly, we have* $conv(r_2^{aux}, \mathcal{G}_{\mathcal{R}'}) = \frac{1-1}{1-0} = 0$, *hence,* $score(r_2, \mathcal{G}_{\mathcal{R}'})$ *is the average of two above conviction values and equals to* 0.5. □

The **Ordered Partial Materialization (OPM)** ranking strategy is analogous to ***PM***, but here the word "Ordered" means that the chosen ruleset $\mathcal{R}'$ consists of merely rules r'_l where the corresponding r_l is revised before the current rule r_i. In other words, in this strategy the rule order in R_H matters, and it is determined by some positive rule measure such as *confidence* or *conviction*.

Chapter 5

RUMIS - Nonmonotonic Rule Mining System

RUMIS stands for Nonmonotonic Rule Mining System which is the tool developed within the current thesis. The RUMIS system aims at revising Horn rules to nonmonotonic ones under the OWA. This chapter describes in details the practical implementation of the theory framework presented in Chapter 4 and uses background definitions in Chapter 3. First, we present the system overview. Second, the implementations of main components in the system are described. Finally, we discuss the system usage.

5.1 System Overview

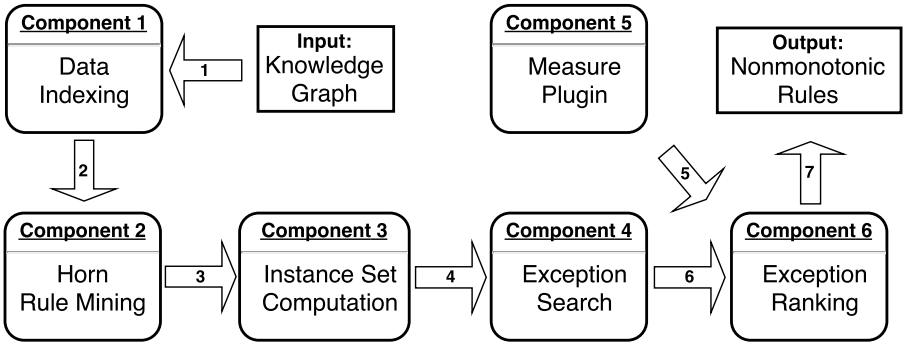

FIGURE 5.1: Components of RUMIS

There are six components in RUMIS as presented in Figure 5.1 with arrows indicating the data flow from input to output. RUMIS mines the nonmonotonic rules from the original graph in the following steps:

- In (1), a KG $\mathcal{G}$ as input is passed to RUMIS. It is then stored and indexed by Component 1 which is exploited to speed up computation in the next steps.
- In (2), Horn rules are mined by Component 2 from facts indexed into RUMIS.
- In (3), normal and abnormal instance sets are found by Component 3 based on the KG $\mathcal{G}$ and the Horn rules mined in the previous step.
- Given normal and abnormal sets known for each Horn rule, (4) represents for finding exception witness sets in Component 4.
- In (5) and (6), exception candidates are ranked in Component 6 using a measure provided by Component 5. RUMIS allocates a separate component for measure plugin to enable flexibility of ranking criteria.
- In (7), RUMIS returns the best revision for the Horn rule set as output.

5.2 Implementation

In this section, the main components mentioned in Section 5.1 are described in details.

5.2.1 Data Indexing

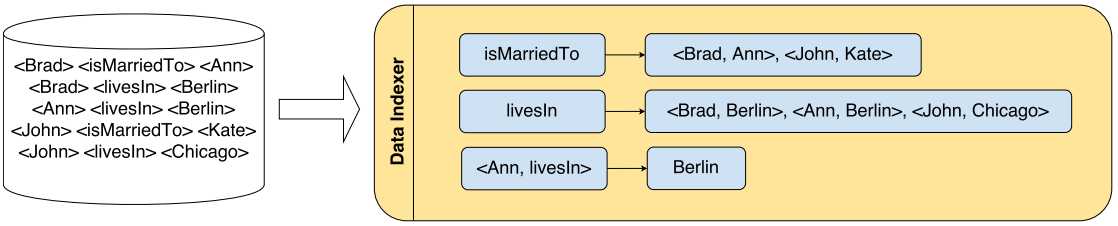

FIGURE 5.2: RUMIS Data Indexer

Since the KG $\mathcal{G}$ can be large, computation of (ab)normal instances as well as exception witness sets is time consuming. To overcome this issue, data indexing is exploited. In a traditional search engine setting, terms such as words, n-grams are indexed into the system and their corresponding posting lists [29] are a collection of documents containing them. We exploit the same intuition in our work. More specifically, RUMIS treats every fact as a document and the terms are subjects, predicates, objects or combinations of them.

For example, Figure 5.2 presents the data indexed from part of a KG in Figure 1.1. Hence, given the predicate *isMarriedTo*, one can determine a set of subject-object pairs

corresponding to it, i.e., <*Brad, Ann*> and <*John, Kate*>. Similarly, given a subject and a relation, the set of objects can be retrieved. For example, we can get "Berlin" from the question "Where does Ann live" based on the data indexing model in Figure 5.2. In this example, the term is the combination of subjects and predicates. More formally, the Data Indexing provides the following functions that are exploited in the rest of the components:

- *getPredicateSubjectSet($\mathcal{G}$, object)* returns a set of predicate-subject pairs corresponding to a given *object* in the KG $\mathcal{G}$.
- *getPredicateSet($\mathcal{G}$, subject, object)* returns a set of predicates corresponding to the input *subject, object* entities.
- *getSubjectSet($\mathcal{G}$, predicate, object)* retrieves a set of subjects for a given *predicate* and an *object*.
- *getSubjectObjectSet($\mathcal{G}$, predicate)* outputs a set of *subject-object* pairs for a given *predicate*.

5.2.2 Positive Rule Mining

Algorithm 1: Positive Rule Mining

Input : KG $\mathcal{G}$
Output : Set of positive rules of the form *h(X, Z) ← p(X, Y), q(Y, Z)*

```
1  absSupp ← {}
2  foreach Triple YqZ in G do
3      pXSet ← getPredicateSubjectSet(G, Y)
4      foreach pX in pXSet do
5          hSet ← getPredicateSet(G, X, Z)
6          foreach h in hSet do
7              absSupp[hpq] + +
8          end
9      end
10 end
11 Sort hpq in a decreasing order of absSupp[hpq]
12 return absSupp
```

Positive Rule Mining component computes a set of the form *h(X, Z) ← p(X, Y), q(Y, Z)* whose *absolute support* exceeds a given threshold. Existing tools from the literature can be exploited for this component, however, due to technical issues we propose to re-implement Horn rule learning as in Algorithm 1. The steps of this algorithm are described as follows. First, *absSupp* defined in line 1 is intended to store the absolute

support of patterns. In the loop (2), for each triple $<Y\ q\ Z>$ from the KG $\mathcal{G}$, using the *getPredicateSubjectSet* function in the Data Indexing component, a set of pairs *X, p* is found in (3) such that $<X\ p\ Y>$ is in $\mathcal{G}$. Then, from line 5 to 6, with *X, Z* found in the previous step, RUMIS searches for every relation h s.t. $<X\ h\ Z>$ is in $\mathcal{G}$. At this point, it is guaranteed that *h(X, Z), p(X, Y), q(Y, Z)*. After that, in line 7, the absolute support of the considered rule is increased by 1. Finally, all rules are sorted in decreasing order of the absolute support (line 11).

5.2.3 Normal and Abnormal Set Mining

Algorithm 2: Normal and Abnormal Set Mining

Input : KG $\mathcal{G}$, h, p, q predicates in the rule $r2$
Output : Normal and abnormal sets of the rule $r2$

```
1  NS ← {}
2  ABS ← {}
3  YZSet ← getSubjectObjectSet(G, q)
4  foreach Pair YZ in YZSet do
5  |   XSet ← getSubjectSet(G, p, Y)
6  |   foreach X in XSet do
7  |   |   if XhZ is in G then
8  |   |   |   Add XZ to NS
9  |   |   else
10 |   |   |   Add XZ to ABS
11 |   end
12 end
13 return NS and ABS
```

This component computes normal and abnormal instance sets for a given rule and a KG as described in Algorithm 2. First, in lines 1 and 2, variables for (ab)normal sets are initialized. Second, for each pair *Y, Z* s.t. $<Y\ q\ Z>$ is in $\mathcal{G}$, a set of X for which $<X\ p\ Y>$ is in the KG is found based on the Data Indexing component (line 3 to 5). At this point, it is guaranteed that *X, Y, Z* satisfy the body of the given rule. Finally, for every X found in the previous step, we verify whether $<X\ h\ Z>$ is in the KG or not. If it is in $\mathcal{G}$, then $<X\ Z>$ is added to the normal set, otherwise it is added to the abnormal set.

5.2.4 Exception Witness Set Mining

We now describe the algorithm for exception witness set construction (Algorithm 3). Given a KG and a positive rule, this component computes all unary and binary exception candidates. To simplify the presentation, Algorithm 3 focuses only on computing binary

Algorithm 3: Exception Witness Set Mining

Input : KG $\mathcal{G}$, h, p, q predicates of the rule $r2$
Output : Exception witness set of the rule $r2$

1 $NS \leftarrow getNormalSet(\mathcal{G}, h, p, q)$
2 $ABS \leftarrow getAbnormalSet(\mathcal{G}, h, p, q)$
3 $EWS^{+} \leftarrow \{\}$
4 $EWS^{-} \leftarrow \{\}$
5 **foreach** *Pair XZ in ABS* **do**
6 $\quad pSet \leftarrow getPredicateSet(\mathcal{G}, X, Z)$
7 $\quad EWS^{+} \leftarrow EWS^{+} \cup pSet$
8 **end**
9 **foreach** *Pair XZ in NS* **do**
10 $\quad pSet \leftarrow getPredicateSet(\mathcal{G}, X, Z)$
11 $\quad EWS^{-} \leftarrow EWS^{-} \cup pSet$
12 **end**
13 $EWS \leftarrow EWS^{+} \setminus EWS^{-}$
14 **return** EWS

exceptions, analogously, unary ones can be mined. More specifically, our goal is to find exceptions of the form *e(X, Z)* which are then inserted into the body of *h(X, Z) :- p(X, Y), q(Y, Z)* to get its revised rule *h(X, Z) :- p(X, Y), q(Y, Z), not e(X, Z)*. Algorithm 3 proceeds as follows. First, normal and abnormal instance sets are found in line 1 and 2 by exploiting Algorithm 2. Then the variables EWS^{+} and EWS^{-} which store a set of relations between $<X\ Z>$ in the abnormal and normal sets resp. are created (line 3 and 4). Second, for every pair $<X\ Z>$ in the abnormal set, all relations between X and Z are added to the EWS^{+} (line 5 to 8). After that, a similar procedure is applied to EWS^{-} in lines 9 to 12. Finally, exception witness set EWS is constructed as the difference between EWS^{+} and EWS^{-} (line 13), which forms the output of the algorithm in line 14.

5.2.5 Measure Plugin

The RUMIS system is supplied with the measure plugin which allows the user to conveniently specify various rule quality criteria. In the current implementation, confidence and conviction are supported. While the former is well-suited for descriptive purposes, the latter is accepted as a possible measure for estimating rule's predictive capabilities [3]. Since the main concern of this thesis is the Horn rule revision and subsequent application of the revised set for predicting possibly missing facts in the original KG, the conviction is the most suitable measure, which is implemented within the RUMIS system.

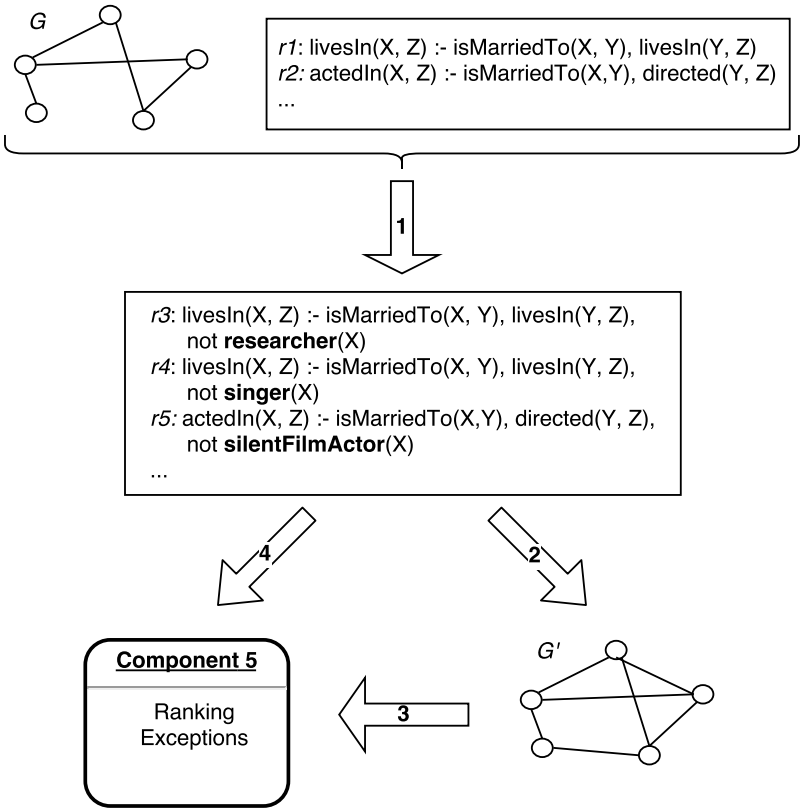

FIGURE 5.3: (Ordered) Partial Materialization Ranking

5.2.6 Exception Ranking

We now describe the exception ranking procedure in details which is illustrated in Figure 5.3 on the PM ranker.

- From a given KG $\mathcal{G}$ and a set of Horn rules, EWS mining (Algorithm 3) is executed to find exception candidates for each Horn rule in (1). As shown in Figure 5.3, a Horn rule may have several exceptions.
- (2) shows that revisions with exceptions (rules $r3, r4, r5, ...$) are used to infer new facts from the original KG.
- $\mathcal{G}'$, i.e., the original KG $\mathcal{G}$ with new predicted facts, is exploited to rank exceptions for each Horn rule in (3). The combination of steps (2) and (3) describes the

Algorithm 4: PM Ranking

Input : KG $\mathcal{G}$, set of positive rules $\mathcal{R}_H$
Output : Set of best revisions R_{NM} for the given positive rules R_H

1 $R_{NM} \leftarrow \{\}$
2 **foreach** *Rule* r *in* R_H **do**
3 $\quad \mathcal{G}' \leftarrow \mathcal{G}$
4 $\quad$ **foreach** *Rule* r'' *in* R_H, r'' *is different from* r **do**
5 $\quad\quad$ Generate safely predicted facts of r'' w.r.t. $\mathcal{G}$ then index these facts to $\mathcal{G}'$
6 $\quad$ **end**
7 $\quad$ Rank exceptions of r based on $\mathcal{G}'$, choose r' as the best revision of r
8 $\quad$ Add r' to R_{NM}
9 **end**
10 **return** R_{NM}

interaction of different rules, i.e, facts generated by other revisions are taken into account to measure quality of a particular nonmonotonic rule.

- (4) is a process that exceptions are ranked by Component 5 according to $\mathcal{G}'$ from (3), the best exception is chosen for the addition to the final revision set.

PM Ranking. Algorithm 4 describes PM ranking procedure, which takes as input a KG and a set of positive rules R_H and outputs a set R_{NM} of their revisions. In line 1, a set of the revised rules is initialized as an empty set. After that, for each positive rule r in R_H, we clone the original KG $\mathcal{G}$ to a new KG $\mathcal{G}'$, then all other rules in R_H are exploited to generate their safely predicted facts which are subsequently added to $\mathcal{G}'$ (line 2 to 6) by Data Indexing component. RUMIS is conservative in revising rules, since the KG $\mathcal{G}$ instead of $\mathcal{G}'$ is exploited to rank exceptions in every iteration. Next, in lines 7 and 8, based on the new KG $\mathcal{G}'$, exceptions of r are ranked and the best revision is added to R_{NM}. Finally, we return R_{NM} as an output of the algorithm.

OPM Ranking. This function is similar to PM ranking, the major difference is the way how the revisions are selected for expansion. At the initial step, Horn rules are sorted in decreasing order of conviction measure. After that, only safely predicted facts from previous rules are exploited to assess the quality of a given rule r. In the OPM ranking, the order of Horn rules matters, i.e., the higher is the conviction of a Horn rule, the more prominent is its impact on other rules.

5.3 Optimization

In Algorithm 4, the KG $\mathcal{G}$ cloning and indexing procedures of the safely predicted facts are executed $|R_H|$ and $|R_H|^2$ times, resp. Since the number of facts in $\mathcal{G}$ and

Algorithm 5: PM Ranking

Input : KG $\mathcal{G}$, set of positive rules $\mathcal{R}_H$
Output : Set of best revisions R_{NM} for the given positive rules R_H

1 $\mathcal{G}' \leftarrow \mathcal{G}$
2 $R_{NM} \leftarrow \{\}$
3 **foreach** *Rule* r *in* R_H **do**
4 | Generate safely predicted facts of r w.r.t. $\mathcal{G}$ then index these facts to $\mathcal{G}'$
5 **end**
6 **foreach** *Rule* r *in* R_H **do**
7 | Generate safely predicted facts of r w.r.t. $\mathcal{G}$ then remove these facts' indexes from $\mathcal{G}'$
8 | Rank exceptions of r based on $\mathcal{G}'$, choose r' as the best revision of r
9 | Add r' to R_{NM}
10 | Generate safely predicted facts of r w.r.t. $\mathcal{G}$ then index these facts to $\mathcal{G}'$ (reverses step in line 7)
11 **end**
12 **return** R_{NM}

positive rules can be large, these operations are time-consuming. We now discuss possible optimizations of Algorithm 4, which concern the KG cloning and data indexing operations. In Algorithm 5, we present possible refinements of Algorithm 4 for the PM ranking.

In lines 1 and 2 of the Algorithm 5, we clone the original graph $\mathcal{G}$ to $\mathcal{G}'$ and create an empty nonmonotonic rule set R_{NM}. After that, from line 3 to 5, for every Horn rule r in R_H, its safely predicted facts are added to $\mathcal{G}'$ using the Data Indexing component.

Now RUMIS is ready to refine PM ranking. In line 7, for each Horn rule r, indexes of its safely predicted facts are removed from $\mathcal{G}'$. This step is needed in order to make sure that safely predicted facts of all other rules apart from r are exploited to determine the quality of r (line 8). This witnesses that the interaction between nonmonotonic rules is taken into consideration during the ranking. After that, the best revision of r is added to the final result in line 9. Line 10 shows a step that reverses what is done in line 7, i.e., safely predicted facts of the rule r based on $\mathcal{G}$ are added to $\mathcal{G}'$ again. This guarantees that the same state in the next iteration can be processed with a new rule.

With the optimized algorithm, we only need $O(|R_H|)$ operations for indexing new predicted facts and one operation for cloning the KG. This is a significant improvement compared to the original version of the algorithm. The difference is visible in practice if the KG is large and many rules are considered.

5.4 Usage

RUMIS [1] is developed and currently tested in Linux, we may extend it to Windows in the future. The system requires the installation of Java 8 as well as the DLV [2] tool for conducting experiments. RUMIS supports the following main tasks:

- *Training data generation:* Given the ideal KG, the training KG is constructed automatically by removing 20% of facts from $\mathcal{G}$ for every binary predicate.
- *Horn rule mining:* With the training KG, a list of Horn rules of the form *h(X, Z) ← p(X, Y), q(Y, Z)* can be learned exploiting Algorithm 1.
- *Nonmonotonic rule mining:* Based on the training KG, exceptions for each positive rule are ranked and the best one is chosen to generate the revision.
- *Automatic experiment:* With the KGs $\mathcal{G}$ and $\mathcal{G}^i$, the experiment can be executed to measure the quality of revision sets and predict new facts from the training data.

In the rest of this section, we list the command line options of RUMIS and explain them in details.

Command Line Options. The RUMIS system supports the following options:

- *-d:* This flag enables DLV in order to extend the KG.
- *-e=[execution function]:* This requires a string as a function for execution, i.e., *new, pos, neg, exp* correspond to creating a new learning KG, positve and nonmonotonic rule mining and conducting the experiment, resp.
- *-f=[working folder path]:* With this option, experiment folder path can be specified.
- *-h:* Option used to retrieve the help menu of the system.
- *-l=[KG file path]:* This requires a file path to the graph in the SPO format, which enables users to choose the learning data.
- *-o=[predicate ratio]:* With this option, one can fix the percentage of facts to be removed for the creation of a learning KG.
- *-p=[Horn rule file path]:* This requires a string as an Horn rule file path, each line in this file is a positive rule in the form *h(X, Z) ← p(X, Y), q(Y, Z).*

[1] https://github.com/htran010589/nonmonotonic-rule-mining
[2] http://www.dlvsystem.com/html/DLV_User_Manual.html

- *-r=[ranking]:* Option allows one to specify the ranking type, i.e., 0, 1, 2 standing for Naive, PM, OPM ranking, resp.
- *-s:* This flag is used to enable sampling of the positive rules.
- *-t=[number of top Horn rules]:* This requires an integer as a number of positive rules with the top absolute support that will be considered for revision.

Command Examples. First of all, please download the repository[3], and then uncompress *data/sample.imdb.txt.zip* to get *sample.imdb.txt* file. In the next step, the repository root folder should be located: *$ cd nonmonotonic-rule-mining.* Now the preparation is finished, and we are ready to present some command examples for RUMIS features.

Training data generation. A learning KG of the IMDB sample dataset can be generated with predicate ratio being 80%: *$ java -jar rumis-1.0.jar -e=new -l=data/sample.imdb.txt -o=0.8 1>training.sample.imdb.txt.* Then *training.sample.imdb.txt* file is the learning KG that we want to generate.

Horn rule mining. Please run the following command for executing IMDB Horn rule mining: *$ java -jar rumis-1.0.jar -e=pos -l=data/sample.imdb.txt.* Then *horn-rules.txt* and can be seen in the same folder with RUMIS jar file where explored positive rules are listed.

Nonmonotonic rule mining. Please run the following command for executing IMDB nonmonotonic rule mining with OPM ranking: *$ java -jar rumis-1.0.jar -e=neg -p=horn-rules.txt -l=data/sample.imdb.txt -r=2.* One may just want to revise top 10 Horn rules with the *-t* option: *$ java -jar rumis-1.0.jar -e=neg -p=horn-rules.txt -l=data/sample.imdb.txt -r=2 -t=10.* After these commands, *revised-rules.txt* in the root folder is the list of nonmonotonic rules.

Automatic experiment. The working folder can be built as a directory *data/experiment/IMDB* that contains all of the following files. First, please rename *sample.imdb.txt* file to *ideal.data.txt* in the directory. Second, please sample the learning data of the ideal KG file and get *training.data.txt.* Third, one should generate *horn-rules.txt* as an output of positive rule mining function applied to the learning data. If only some positive rules need to be revised, we can list them in *selected.horn-rules.txt.* Finally, DLV binary file should be downloaded to the directory and renamed as *dlv.bin.*

The command that executes experiment with OPM ranking and top 10 positive rules (without DLV) is: *java -jar rumis-1.0.jar -e=exp -f=data/experiment/IMDB/ -r=2 -t=10 1>experiment.txt 2>log.*

[3] https://github.com/htran010589/nonmonotonic-rule-mining

Chapter 6

Evaluation

In this chapter, we present the evaluation results of the RUMIS system. We focus on testing the quality of rule revisions in terms of the conviction measure, as well the prediction quality. The chapter is organized as follows. First, the setting of the experiment is discussed. Second, the quality of rulesets and the facts they predict are assessed. Finally, we present the runtime performance of the system RUMIS on different real-world datasets.

6.1 Setting

Dataset. To measure the quality of rules and their predictions, the ideal graph $\mathcal{G}^i$, i.e. the KG containing all true facts about the real world, is required. However, constructing $\mathcal{G}^i$ is obviously not possible. Hence, instead we treat the given graph as an approximation of the ideal KG $\mathcal{G}^i_{appr}$. To obtain the available (training) KG $\mathcal{G}^a$ we remove from $\mathcal{G}^i_{appr}$ 20% of the facts for every binary relation, and retain all unary facts in $\mathcal{G}^i_{appr}$. It is guaranteed that there is no isolated vertex in $\mathcal{G}^a$, i.e the node which is not connected to any other nodes in the graph. In our experiment, YAGO3 [45], Wikidata Football and IMDB [1] datasets are used as the ideal KGs. YAGO3 covers a variety of domains and contains roughly 1.8 million entities, 38 predicates, and 20.7 million triples. Meanwhile, IMDB only focus on movie content collected from the IMDB website [2], there are 112 thousand entities, 38 predicates and 583 thousand triples in this KG. Besides, to construct the Wikidata Football for our experiments, we sample 1 million facts with 238 thousand entities and 443 predicates from football domain of the original Wikidata KG [3].

[1] http://people.mpi-inf.mpg.de/~gadelrab/downloads/ILP2016
[2] http://imdb.com
[3] https://www.wikidata.org

Figure 6.1 depicts the ideal, approximated and available KGs as well the extended KGs $\mathcal{G}_{\mathcal{R}_H}$, $\mathcal{G}_{\mathcal{R}_{NM}}$ obtained by resp. applying $\mathcal{R}_H$ and its revision $\mathcal{R}_{NM}$ to $\mathcal{G}^a$. The RUMIS system aims to tackle the KG completion problem by narrowing the difference between the $\mathcal{G}_{\mathcal{R}_{NM}}$ and $\mathcal{G}^i$.

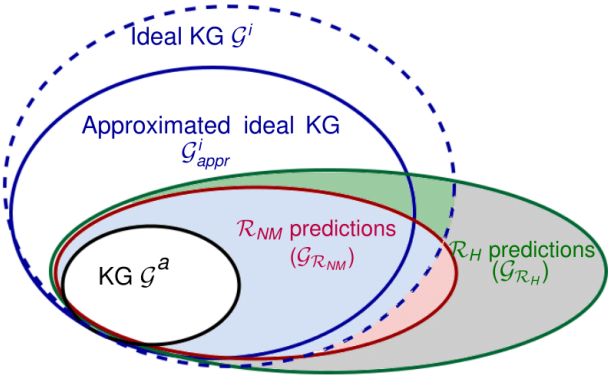

FIGURE 6.1: The Ideal, Available and Extended KGs.

Experimental Setup. In our experiments, RUMIS is executed on a server which has Linux OS, 40 cores and 400GB RAM memory. As the data preparation, positive rules in the format *h(X, Z)* ← *p(X, Y), q(Y, Z)* are extracted from $\mathcal{G}^a$ and ranked according to the *absolute support* measure. The positive rule mining function of RUMIS (see Chapter 5) is exploited in this step. After that, the positive rules are revised following the approach presented in Chapter 4. The *conviction* measure is used as the *rm* rule measure. Various exception ranking earlier described strategies are realized in RUMIS. The resulting rule revisions are stored in $\mathcal{R}_N$, $\mathcal{R}_{PM}$, and $\mathcal{R}_{OPM}$, resp.

6.2 Ruleset Quality

We present the evaluation results of the rule quality assessment in Table 6.1 and 6.2 for YAGO, IMDB and Wikidata Football, resp. For every row in the tables, we fix the top-k ($k = 5, 30, \ldots 100$) positive rules $\mathcal{R}_H$ that will be subsequently revised. Then the *average conviction* of the rules and their revisions are found for YAGO, IMDB and Wikidata Football by using RUMIS. Naturally the quality of the revised rules is better than that of their positive versions w.r.t. conviction. Besides, in general while the average quality of every column has a decreasing trend with the appearance of rules having lower precision levels, the enhancement ratio between positive rules and their revisions increases and reaches the peak of 7.8% and 10.3% for top-100 IMDB and top-30 Wikidata Football rules, resp. The results show that the introduction of negated atoms significantly boost up the rule precision.

topk	**YAGO**				**IMDB**			
	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$
5	1.3784	1.3821	1.3821	1.3821	2.2670	2.3014	2.3008	2.3014
30	1.1207	1.1253	1.1236	1.1237	1.5453	1.5644	1.5543	1.5640
50	1.0884	1.0923	1.0909	1.0913	1.3571	1.3749	1.3666	1.3746
60	1.0797	1.0837	1.0823	1.0829	1.3063	1.3221	1.3143	1.3219
70	1.0714	1.0755	1.0736	1.0744	1.2675	1.2817	1.2746	1.2814
80	1.0685	1.0731	1.0710	1.0720	1.2368	1.2499	1.2431	1.2497
100	1.0618	1.0668	1.0648	1.0659	1.3074	1.4100	1.3987	1.4098

TABLE 6.1: The Average Quality of the Top Positive and Nonmonotonic Rules for YAGO, IMDB.

topk	**Wikidata Football**			
	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$
5	3.2282	3.2342	3.2340	3.2342
30	3.1118	3.4315	3.4194	**3.4271**
50	2.7115	2.9193	2.9070	2.9135
60	2.4930	2.7101	2.6986	2.7046
70	2.3395	2.5272	2.3931	2.5219
80	2.4071	2.5781	2.4597	2.5734
100	2.3258	2.4847	2.3859	2.4806

TABLE 6.2: The Average Quality of the Top Positive and Nonmonotonic Rules for Wikidata Football.

The enhancement ratio between revisions of the three ranking methods and the top positive rules is shown in Figure 6.2, 6.3 and 6.4 for IMDB, YAGO and Wikidata Football, resp. In these figures, the height and the width are corresponding to the top-k and the improvement rate of the average conviction. One can observe that the rate has an uptrend, in general the more low quality Horn rules are added to the top ruleset, the higher is the improvement rate. The Naive ranking shows the best results w.r.t. the rule quality, which is obviously expected.

With IMDB dataset, the results of Naive and OPM ranking are approximately the same and slightly better than PM ranking. For the top-100 rules, the best average improvement of 7.8% is achieved. It can be seen that the quality of positive rules around top-100 are much worse than the rest, resulting in the sharp increase between top-80 and top-100 in the Figure 6.2.

In general, the improvement rates for the average conviction in YAGO increase. However, the contrast between the highest and the lowest in this figure is not as large as in IMDB data due to the qualities of top-100 Horn rules in YAGO are not highly different.

As regards the Wikidata Football, Figure 6.4 witnesses an interesting pattern where the improvement rate significantly climbs from top-5 to top-30. It can be seen that

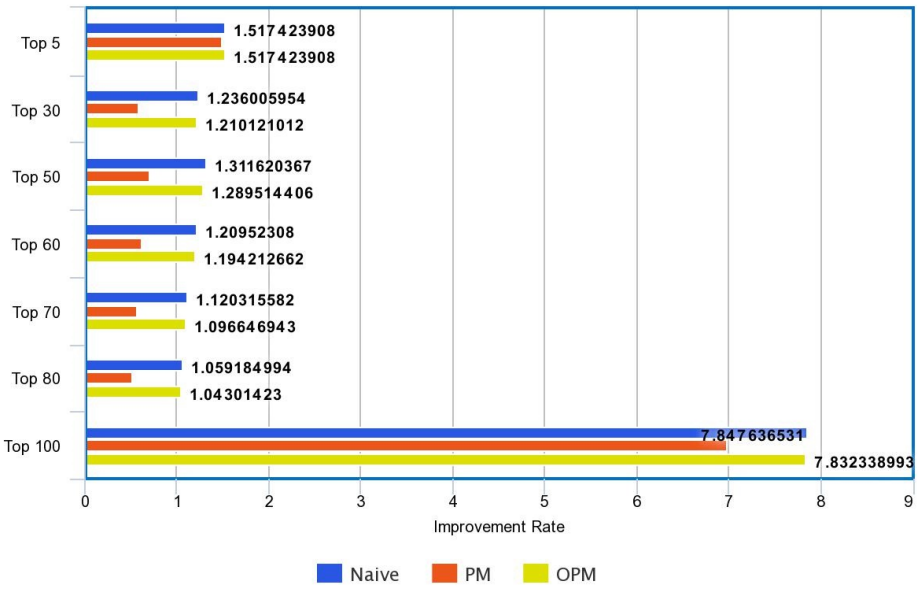

FIGURE 6.2: The Average Conviction Improvement Rate (%) of Rules Revised using Our Methods and IMDB Dataset.

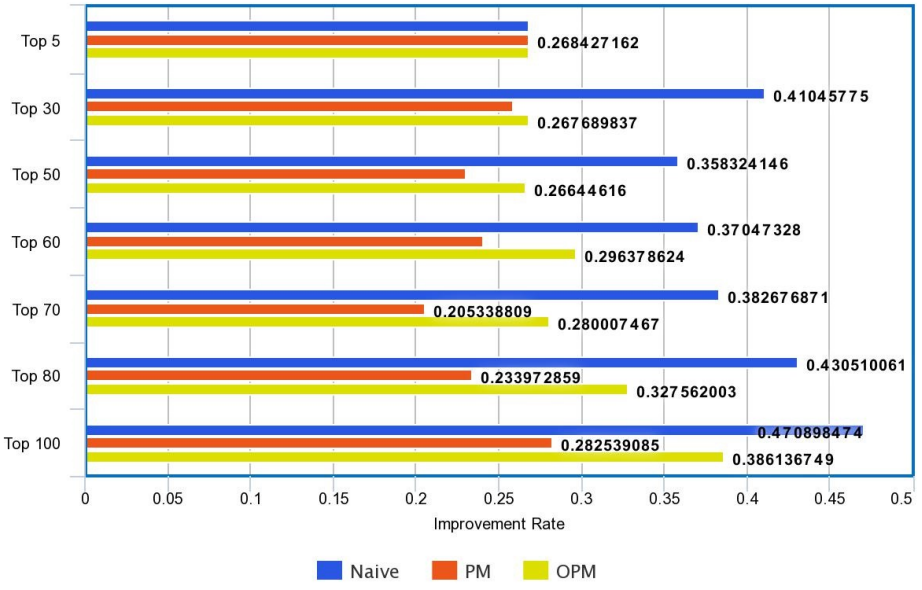

FIGURE 6.3: The Average Conviction Improvement Rate (%) of Rules Revised using Our Methods and YAGO Dataset.

RUMIS mines very good exceptions for positive rules around top-30, resulting in a peak of more than 10% enhancement ratio for this dataset.

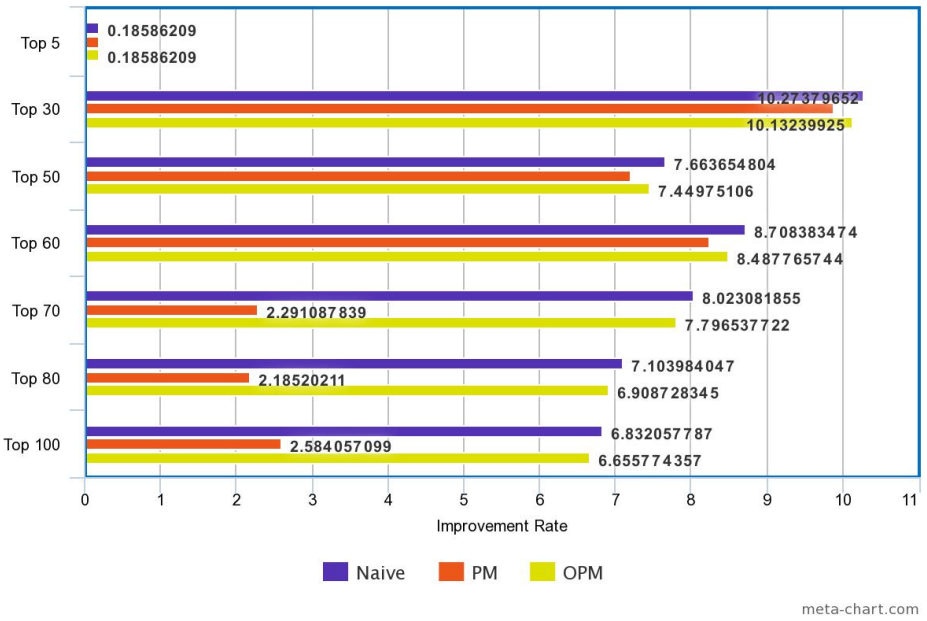

FIGURE 6.4: The Average Conviction Improvement Rate (%) of Rules Revised using Our Methods and Wikidata Football Dataset.

6.3 Prediction Quality

We now describe the evaluation procedure for estimating the predictive quality of the revised rules. Among top-50 (for IMDB, YAGO3) and top-300 (for Wikidata Football) positive rules mined from the KG, 5 are sampled as $\mathcal{R}_H$ and then the revision procedure is applied for these rules. After that, the predictions of the rules from $\mathcal{R}_H$ and their revisions are analyzed to evaluate our approaches.

To this end, these rulesets are applied to the learning KG $\mathcal{G}^a$ and the corresponding predicted facts are generated by DLV tool [27]. Subsequently, we achieve $\mathcal{G}_{\mathcal{R}_H}$, $\mathcal{G}_{\mathcal{R}_N}$, $\mathcal{G}_{\mathcal{R}_{PM}}$ and $\mathcal{G}_{\mathcal{R}_{OPM}}$ for $\mathcal{R}_H$ and revisions of $\mathcal{R}_H$, resp. The statistics is shown in Table 6.3, where the first three columns indicate the head relations of the rules in $\mathcal{R}_H$, the number of new predictions, i.e. predicted facts not included in $\mathcal{G}^a$ and the part of these predictions which are outside $\mathcal{G}^i_{appr}$, resp. The statistics in the second and third columns is available for positive rules and all of their revisions.

One can see that not many predicted facts are included in $\mathcal{G}^i_{appr}$ ($\approx$9% for IMDB and $\approx$2% for YAGO ideal KGs). This can be explained by the fact that YAGO is a highly incomplete general purpose KG. Moreover, it is crucial to note that the sampled Horn rules and their revisions generate approximately the same number of correctly predicted facts which are present in $\mathcal{G}^i_{appr}$. More specifically, in all three datasets $\mathcal{G}_{\mathcal{R}_H} \backslash \mathcal{G}_{\mathcal{R}_{PM}} \cap \mathcal{G}^i_{appr} = \emptyset$ means that the grey region has nothing in common with the approximated ideal KG in

predicate	predictions				outside approx. ideal KG				corr. removed %		
	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$
I:*actedIn*	1231	1214	1230	1214	1148	1131	1147	1131	90	100	90
I:*genre*	629	609	618	609	493	477	482	477	50	20	50
I:*hasLang*	173	102	125	102	163	92	115	92	60	100	60
I:*prodIn*	2489	2256	2327	2327	2488	2255	2326	2326	10	10	30
									52.50	45.16	57.75
Y:*direct*	41079	39174	39174	39174	41021	39116	39116	39116	100	100	100
Y:*grFrom*	3519	3456	3456	3456	3363	3300	3300	3300	100	100	70
Y:*citizOf*	3407	2883	2883	2883	3360	2836	2836	2836	50	50	70
Y:*bornIn*	110283	108317	109846	108317	109572	107607	109137	107607	90	90	100
									85	85	85
W:*occupa*	530	483	528	528	509	462	507	507	80	100	100
W:*citizOf*	606	580	604	603	528	503	526	525	100	100	100
W:*diedIn*	16483	14516	16174	16253	16395	14429	16086	16165	100	100	100
W:*bornIn*	3669	3625	3620	3620	3604	3561	3555	3555	100	90	90
									95	**97.5**	**97.5**

TABLE 6.3: New Facts Predicted by the Rulesets for IMDB (I), YAGO (Y) and Wikidata Football (W).

Figure 6.1, in other words, the addition of exceptions does not lead to the removal of correct predictions from $\mathcal{G}^i_{appr}$.

To guarantee the fairness of the comparison between predictions generated by different rulesets, it is necessary to keep the $\mathcal{R}_H$ not totally incorrect. Indeed, provided that the sampled positive rules always generate inaccurate facts, inserting arbitrary negated atoms may filter out some incorrect predictions, resulting in the rule enhancement, yet the rules themselves would still be of poor quality. Furthermore, observe that the number of predictions made by $\mathcal{R}_H$ outside $\mathcal{G}^i_{appr}$ (third column of Table 6.3) is rather large. To verify these predictions, unfortunately no ground truth is available. Thus, we have to manually check the generated facts using the Internet resource. Since the number of the facts to be checked is huge, we propose to randomly select maximum 20 new predictions for each head predicate in $\mathcal{R}_H$ and verify them. For the IMDB, YAGO and Wikidata Football, 70%, 30% and 55% of predictions respectively turned out to be indeed correct. This shows that the quality of the positive rules that we start with is acceptable.

Since the size of the set difference between predictions made by $\mathcal{R}_H$ and extended by applying $\mathcal{R}_H$ and its revisions is also huge, we have to proceed further with sampling to evaluate the predictive quality of the revision. Here for each head relation from the set differences $\mathcal{G}_{\mathcal{R}_H} \setminus \mathcal{G}_{\mathcal{R}_N}$, $\mathcal{G}_{\mathcal{R}_H} \setminus \mathcal{G}_{\mathcal{R}_{PM}}$ and $\mathcal{G}_{\mathcal{R}_H} \setminus \mathcal{G}_{\mathcal{R}_{OPM}}$, 10 facts have been randomly sampled for manual check. In the last column of the Table 6.3, the proportion of incorrect facts in the difference sets are presented. These facts are called "correctly removed", since they correspond to false prediction made by $\mathcal{R}_H$ but avoided by the respective revisions (the grey region in in Figure 6.1). For the IMDB dataset, among all the revision strategies, OPM ranking always performs best with 57.75% and 97.5% correctly removed predictions for IMDB and Wikidata Football, resp. Meanwhile, all the rankers demonstrate the same results (85%) for the YAGO KG. Since the predictive power of the positive rules

$r_1 : writtenBy(X, Z) \leftarrow hasPredecessor(X, Y), writtenBy(Y, Z),$ **not** $american\_film(X)$
$r_2 : actedIn(X, Z) \leftarrow isMarriedTo(X, Y), directed(Y, Z),$ **not** $silent\_film\_actor(X)$
$r_3 : isPoliticianOf(X, Z) \leftarrow hasChild(X, Y), isPoliticianOf(Y, Z),$**not** $vicepresidentOfMexico(X)$
$r_4 : hasCitizenship(X, Z) \leftarrow hasFather(X, Y), hasCitizenship(Y, Z),$**not** $countryOfTheUK(Z)$

FIGURE 6.5: Examples of the Revised Rules

in IMDB is better than those in YAGO, the revision of the latter makes a more visible impact than the former.

6.4 Running Times

In our experiment, top-100 positive rules are mined from IMDB and YAGO while this number of Wikidata Football is 300. Table 6.4 provides statistics about running times (in seconds) of three different steps in the RUMIS system. More specifically, the second row of this table indicates how long for mining Horn rules and exception witness sets. In addition, the third and fourth rows present the average running times (over three ranking methods Naive, PM, OPM) of resp. ranking exceptions and extending KGs with DLV.

Steps	IMDB	YAGO	Wikidata Football
Horn Rule and EWS Mining	7	68	193
Exception Ranking	32	111	2940
Extension with DLV	8	310	180

TABLE 6.4: Running Times for Each Step of the Three Datasets

It can be seen that the numbers of Wikidata Football for the first two steps are the largest since the quantity of positive rules mined from this dataset is much bigger than that of IMDB or YAGO. Meanwhile, among the three datasets, YAGO has the most number of facts, resulting in the longest time to extend the KG using DLV (310 seconds).

Example rules. Some interesting examples are presented in the Figure 6.5. Here the rule r_1 mined from IMDB dataset indicates that normally movies in the same series are written by the same writer except the American movies. The rule r_3 generated from YAGO reveals an interesting pattern from domain politics, i.e, typically fathers and sons are politicians in the same country unless the fathers are Mexican vice-presidents. Being mined from Wikidata Football, the rule r_4 shows that father and son usually have the same citizenship, except the case that the father holds a citizenship in the UK (countries such as England, Northern Ireland, Wales and Scotland).

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Advances in Information Extraction have led to the construction of large KGs in the format $\langle subject\ predicate\ object \rangle$ triples. However, due to the automatic construction, these KGs can be incomplete. Horn rule mining is a popular approach to address this issue [17].

In this thesis we have looked at the problem of enhancing the quality of positive rules and subsequently improve the accuracy of their predictions, by revising the mined rules into the nonmonotonic ones. We follow the ideas from [16] where the same problem was studied for KG containing only unary facts. Meanwhile, RUMIS tool implemented in our research generates negative rules from KG in nature format.

We have extended the results from [16] to KG revision in their original relational form and developed the RUMIS system, for learning nonmonotonic rules in KGs under the OWA. Subsequently, the chosen revisions are exploited to extend the original data, and thus, tackle the KG completion problem. Some experiments in the thesis are conducted for testing quality of rules generated by RUMIS and the results demonstrate that the proposed approach outperforms the state of the art KG rule learning systems w.r.t. to the quality of the made predictions.

7.2 Future Directions

We now discuss possible future directions.

- **Tackle language bias challenge.** In this work we fixed the language bias of the rules to be mined. Certainly, a promising and natural direction is to extend our results to further rule forms and make the language bias more flexible. More forms for the positive rules can be implemented, not only the specific form introduced in Chapter 5. The language bias can be manually specified by users, this setting will make the RUMIS less restrictive and diversify possible revisions. Another possible future direction is to add existential operators to the heads of the rules, e.g., $\exists\ Y$ *hasParent(X, Y)* $\leftarrow$ *person(X)* can be taken into consideration.

- **Enriching exception forms.** Currently RUMIS only supports one exception in the body and this exception is the relation between variables in the head or a unary atom. To extend the form of nonmonotonic rules, we can increase the number of exceptions in the body of the revision. E.g., some interesting rule like *isCitizenOf(X, Y)* $\leftarrow$ *bornIn(X, Y), not manager(X), not isAsiaCountry(Y)* can be mined in the updated version of the system.

- **Optimizations.** Due to the large size of the KGs, data indexing is a time burden step in our implementation. Thus, a natural optimization direction is storing the facts in a database and building a web service to find (ab)normal sets of conjunctive relational queries. Thanks to this technique, we only build the service once at the beginning, and the time for experiment is significantly reduced.

- **Try more rule measures.** Other predictive measures and exception evaluation methods can be tested to search for interesting nonmonotonic rules. A survey in [3] specifies a variety of choices for predictive rule measure where conviction is only one of them.

- **N-ary facts.** In YAGO and IMDB datasets, the facts are three-dimensional, i.e., each of them contains a subject, a predicate and an object. The Wikidata Football dataset exploited in the experiments has the same property, i.e., it is extracted from simplified version of original Wikidata where all facts are projected to three-dimensional space. One possible future extension takes the n-ary facts, i.e., facts contain n-parameters into consideration. E.g., the five-dimensional fact *<Ronaldo> <playsFor> <Manchester United> <in> <2008>* may appear in full Wikidata KG.

- **Sample training KG randomly.** In our current experiment, every learning KG is chosen from the ideal KG by removing 20% of facts for each binary predicate. This setting is restrictive because it tries to maintain the distribution of facts over the predicates. It is worth testing the quality of predictions generated by RUMIS if the training data is chosen randomly where the percentage of the retained facts is different for every relation.

- **Optimize DLV.** DLV running time is a burden in the current system. Indeed, with very large KGs and 10 revised rules, it takes up to days to extend KG. Since completing a given KG is one of the main goals of the tool, optimizations of DLV tool should be studied to make sure the total run time performance is acceptable.

- **Facts with probability.** For our experiment, the facts in the given KG are always true, this is not suitable in practice where some facts can be totally false. Indeed, modern KGs might contain incorrect triples, since their large parts are constructed automatically by information extraction techniques. As a possible future direction, probability can be assigned to facts in the training KG as the weight, and taken into account during the rule learning and revision. This results in new predicted facts with a certain confidence, represented as weights assigned to them.

Bibliography

[1] A survey of current link discovery frameworks. *Semantic Web*, (Preprint):1–18, 2015.

[2] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. Advances in knowledge discovery and data mining. chapter Fast Discovery of Association Rules, pages 307–328. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996.

[3] Paulo J. Azevedo and Alípio M. Jorge. *Comparing Rule Measures for Predictive Association Rules*, pages 510–517. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[4] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):122, 2009.

[5] A. Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *AAAI*, 2011.

[6] Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. *SIGMOD Rec.*, 26(2):255–264, June 1997.

[7] Volha Bryl and Christian Bizer. Learning conflict resolution strategies for cross-language wikipedia data fusion. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 1129–1134, New York, NY, USA, 2014. ACM.

[8] Yang Chen, Sean Goldberg, Daizy Zhe Wang, and Soumitra Siddharth Johri. Ontological Pathfinding: Mining First-Order Knowledge from Large Knowledge Bases. In *in Proc. of SIGMOD 2016*, page to appear, 2016.

[9] Domenico Corapi, Alessandra Russo, and Emil Lupu. Inductive Logic Programming as Abductive Search. In Manuel Hermenegildo and Torsten Schaub, editors, *Technical Communications of the 26th International Conference on Logic Programming*, volume 7 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54–63, Dagstuhl, Germany, 2010. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[10] Fariz Darari, Werner Nutt, Giuseppe Pirrò, and Simon Razniewski. Completeness Statements about RDF Data Sources and Their Use for Query Answering. In *Proceedings of ISWC*, pages 66–83, 2013.

[11] Luc Dehaspe and Luc De Raedt. Mining association rules in multiple relations. In Nada Lavrac and Saso Dzeroski, editors, *Inductive Logic Programming, 7th International Workshop, ILP-97, Prague, Czech Republic, September 17-20, 1997, Proceedings*, volume 1297 of *Lecture Notes in Computer Science*, pages 125–132. Springer, 1997.

[12] Luc Dehaspe and Luc De Raedt. Mining association rules in multiple relations. In Nada Lavrac and Saso Dzeroski, editors, *Inductive Logic Programming, 7th International Workshop, ILP-97, Prague, Czech Republic, September 17-20, 1997, Proceedings*, volume 1297 of *Lecture Notes in Computer Science*, pages 125–132. Springer, 1997.

[13] Luc Dehaspe and Hannu Toironen. Relational data mining. chapter Discovery of Relational Association Rules, pages 189–208. Springer-Verlag New York, Inc., New York, NY, USA, 2000.

[14] Luc Dehaspe and Hannu Toivonen. Discovery of frequent datalog patterns. *Data Mining and Knowledge Discovery*, 3(1):7–36, 1999.

[15] Arnab Dutta, Christian Meilicke, and Simone Paolo Ponzetto. *A Probabilistic Approach for Integrating Heterogeneous Knowledge Sources*, pages 286–301. Springer International Publishing, Cham, 2014.

[16] Mohamed H. Gad-Elrab, Daria Stepanova, Jacopo Urbani, and Gerhard Weikum. *Exception-Enriched Rule Learning from Knowledge Graphs*, pages 234–251. Springer International Publishing, Cham, 2016.

[17] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. Fast rule mining in ontological knowledge bases with amie $$+$$ +. *The VLDB Journal*, 24(6):707–730, 2015.

[18] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. pages 1070–1080. MIT Press, 1988.

[19] Lise Getoor and Ben Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.

[20] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, 2004.

[21] Katsumi Inoue. Learning extended logic programs. In *In Proceedings of the 15th International Joint Conference on Artificial Intelligence*, pages 176–181. Morgan Kaufmann, 1997.

[22] Xueyan Jiang, Volker Tresp, Yi Huang, and Maximilian Nickel. Link prediction in multi-relational graphs using additive models. In *Proceedings of the 2012 International Conference on Semantic Technologies Meet Recommender Systems & Big Data - Volume 919*, SeRSy'12, pages 1–12, Aachen, Germany, Germany, 2012. CEUR-WS.org.

[23] Joanna Józefowska, Agnieszka Lawrynowicz, and Tomasz Lukaszewski. The role of semantics in mining frequent patterns from knowledge bases in description logics with rules. *CoRR*, abs/1003.2700, 2010.

[24] A. C. KAKAS, R. A. KOWALSKI, and F. TONI. Abductive logic programming. *Journal of Logic and Computation*, 2(6):719–770, 1992.

[25] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, August 2009.

[26] Mark Law, Alessandra Russo, and Krysia Broda. The ILASP system for learning answer set programs, 2015.

[27] Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Thomas Eiter, Georg Gottlob, Simona Perri, and Francesco Scarcello. The dlv system for knowledge representation and reasoning. *ACM Transactions on Computational Logic (TOCL)*, 7(3):499–562, 2006.

[28] John W. Lloyd. *Foundations of Logic Programming, 2nd Edition*. Springer, 1987.

[29] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[30] Emir Muñoz, Aidan Hogan, and Alessandra Mileo. Triplifying wikipedia's tables. In *Proceedings of the First International Conference on Linked Data for Information Extraction - Volume 1057*, LD4IE'13, pages 26–37, Aachen, Germany, Germany, 2013. CEUR-WS.org.

[31] Stephen Muggleton. Inverse entailment and progol. *New Generation Computing*, 13(3):245–286, 1995.

[32] Stephen Muggleton and Luc de Raedt. Special issue: Ten years of logic programming inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19:629 – 679, 1994.

[33] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

[34] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 809–816, New York, NY, USA, June 2011. ACM.

[35] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 271–280, New York, NY, USA, 2012. ACM.

[36] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web Journal*, 2015.

[37] Heiko Paulheim and Simone Paolo Ponzetto. Extending dbpedia with wikipedia list pages. In *NLP-DBPEDIA 2013 : Proceedings of the NLP & DBpedia workshop co-located with the 12th International Semantic Web Conference (ISWC 2013)*, volume 1064, pages 1–6, Aachen, 2013. RWTH.

[38] J. R. Quinlan and R. M. Cameron-jones. Foil: A midterm report. In *In Proceedings of the European Conference on Machine Learning*, pages 3–20. Springer-Verlag, 1993.

[39] J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266, 1990.

[40] Oliver Ray. Nonmonotonic abductive inductive learning. *Journal of Applied Logic*, 7(3):329 – 340, 2009. Special Issue: Abduction and Induction in Artificial Intelligence.

[41] Sebastian Riedel, Limin Yao, Benjamin M. Marlin, and Andrew McCallum. Relation extraction with matrix factorization and universal schemas. In *Joint Human Language Technology Conference/Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL '13)*, June 2013.

[42] Dominique Ritze, Oliver Lehmberg, and Christian Bizer. Matching html tables to dbpedia. In *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics*, WIMS '15, pages 10:1–10:6, New York, NY, USA, 2015. ACM.

[43] Chiaki Sakama. Induction from answer sets in nonmonotonic logic programs. *ACM Trans. Comput. Log.*, 6(2):203–231, 2005.

[44] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. Reasoning With Neural Tensor Networks For Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26*. 2013.

[45] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM.

[46] Zhichun Wang and Juan-Zi Li. Rdf2rules: Learning rules from RDF knowledge bases by mining frequent predicate cycles. *CoRR*, abs/1512.07734, 2015.

[47] Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 515–526, New York, NY, USA, 2014. ACM.

[48] Fei Wu, Raphael Hoffmann, and Daniel S. Weld. Information extraction from wikipedia: Moving down the long tail. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 731–739, New York, NY, USA, 2008. ACM.

[49] Mohammed J. Zaki and Wagner Meira Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press, New York, NY, USA, 2014.

[50] Qiang Zeng, Jignesh M. Patel, and David Page. QuickFOIL: Scalable Inductive Logic Programming. *PVLDB*, 8(3):197–208, 2014.

[51] Qiang Zeng, Jignesh M. Patel, and David Page. QuickFOIL: Scalable Inductive Logic Programming. *PVLDB*, 8(3):197–208, 2014.

[52] Chengqi Zhang and Shichao Zhang. *Association Rule Mining: Models and Algorithms*. Springer-Verlag, Berlin, Heidelberg, 2002.