KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Modeling Relational Data Mining

Sergey Paramonov

Supervisors:
Prof. dr. Luc De Raedt
Prof. dr. Marc Denecker

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Computer Science

August 2017

Modeling Relational Data Mining

Sergey PARAMONOV

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Computer Science

August 2017

Preface

...

Instructies van de faculteit:

In het voorwoord wordt de algemene doelstelling van het werk samengevat in enkele regels en worden personen, diensten of firma's bedankt voor hun medewerking bij het tot stand komen van het werk.

De naam van firma's en personen uit deze firma's mogen slechts worden vermeld mits hun uitdrukkelijke toelating én na overleg met de supervisor(en)! Steeds wordt de supervisor(en) vermeld, de verantwoordelijke en eventueel de personen die rechtstreeks geholpen hebben bv. door het ter beschikking stelling van meetresultaten, faciliteiten. Ook de instantie die eventueel een doctoraatsbeurs heeft toegekend wordt bedankt (bv. FWO, IWT, ...).

Abstract

< What data mining is >
Data Mining is the process of discovering new knowledge from the data. A significant attention in the research community is devoted to data analysis in the presence of the data independence assumption, i.e., when the collected data points are independent from each other.

< What relational means >
However, it is often the case that the objects are connected and related to each other by means of relations. This setting is called *relational* and the associated task is referred as *relational reasoning*.

< Relational problems >
Electronic tables, spreadsheets, and databases are all examples of relational data that is in a wide use today. The objects are connected by means of relations and constraints. In spreadsheets these are tables and formulae, and in databases schema relations and integrity constraints.

< Key issue >
We argue that a general approach for modeling and solving data mining problems in the relational setting is missing. The goal of this thesis is to fill in this gap.

< Contribution >
Firstly, we demonstrate how the problem of learning in relation setting is different from the classical machine learning approach and propose a system named TaCLe as the first working in this setting.

Secondly, we demonstrate how the relational approach generalizes a classical problem of boolean matrix factorization into the Relational Data Factorization, which allows to model a spectrum of classical data mining problems and introduces new ones as well.

Thirdly, we demonstrate how existing relational reasoning formalisms, such as Answer Set Programming, can be enhanced by relational learning techniques known as sketching.

Last but not least, we demonstrate how relational approach can be used to mine relational patterns, known as structured pattern mining in data mining community.

Beknopte samenvatting

...

Instructies van de faculteit:

In een beknopte tekst van maximum 2 pagina's worden de belangrijkste doelstellingen en besluiten geformuleerd, zowel in het Nederlands als in het Engels. Zulke samenvattingen kunnen worden gebruikt in wetenschappelijke verslagen van het departement of de faculteit. Het Engels moet vlekkeloos zijn.

List of Abbreviations

ASP Answer Set Programming. 8

DCA Domain Closure Assumption. 8

FOL First Order Logic. 5, 7

UNA Unique Name Assumption. 8

List of Symbols

I	A relational structure

Contents

List of Figures

List of Tables

Instructies van de faculteit:

De hoofdstukken: Elk hoofdstuk is ingelast met een bepaald doel voor ogen. Dit doel wordt vermeld in de eerste paragraaf van elk hoofdstuk. Naargelang de aard van de tekst (experiment, uitvoering, theoretische ontwikkeling, . . .) volgen de paragrafen elkaar op. Beweringen worden altijd gestaafd, hetzij door eigen experimenten, hetzij door een theoretische afleiding, hetzij door verwijzingen naar de literatuur. Elk hoofdstuk eindigt met een kort samenvattend besluit waarbij nagegaan wordt in hoeverre de doelstelling van het betrokken hoofdstuk verwezenlijkt is. De deelbesluiten moeten de lezer automatisch leiden naar het algemeen besluit aan het einde van het werk.

Chapter 1

Introduction

In his seminal work Sir Bob Kowalski (1979) proclaimed:

$$\text{Algorithm} = \text{Logic} + \text{Control}.$$

1.1 Structure of the Text

Sergey: talk about chapters here

1.2 Datasets, code and experimental results

Sergey: talk about github here: TaCLe, SkASP, etc

Chapter 2

Background

From now on you shall be called
Brian that is called Brian.

The Life of Brian

In this chapter, we introduce commonly used formalisms and definitions.

2.1 First Order Logic

In this section, we describe the syntax and semantics of FOL, for an extensive overview of FOL, we refer to Enderton (2001).

A formal language is a triple: *vocabulary*, *syntax* and *semantics*. Vocabulary is the set of symbols that can be used. Syntax is the set of rules on how these symbols can be combined together. And semantics is the way to interpret the statements. We omit the vocabulary, if it is clear from the context.

For each predicate p and each function symbol f in the vocabulary, we assume a natural number n called *arity* to be given, written as p/n and f/n. This number indicates the number of parameters it takes, we often omit the arity if it is clear from the context. Propositional symbols are zero-arity predicates and constants are zero-arity functions. We assume propositional symbols *true*, $\top$ and *false*, $\bot$ to be always in the vocabulary.

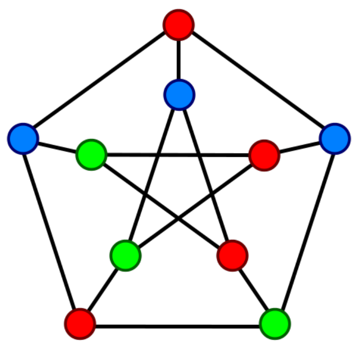

Figure 2.1: Graph coloring of the Petersen's graph using three colors

Example 2.1 (Predicates and functions)**.** Consider the famous problem of coloring a map, as in Figure 2.1. The vocabulary would contain a predicate symbol *border/2* and a function *coloring/1*, together with a set of constants representing countries, such as *belgium*, *netherlands*, etc.

A *structure* associates values with the predicates and functions from the vocabulary. A structure I consists of

- A domain D^I, which we also refer as the *universe*. The domain defines what are the possible values to be used.
- For each predicate p/n, the set of its values $p^I \subseteq \overbrace{D^I \times \cdots \times D^I}^{n}$
- For each function symbol f/n, f^I the mapping $\overbrace{D^I \times \cdots \times D^I}^{n} \mapsto D^I$

We call the set of values p^I for a predicate p and the mapping f^I for a function f an *interpretation.*

We assume the set of inequalities $\{=, \neq, <, >, \leq, \geq\}$, arithmethic $\{+, -, *, \div\}$ and true $\top$, false $\bot$ to be interpreted by any interpretation in a standard way.

Example 2.2 (Interpretaion of map coloring)**.** Consider the predicate and the function from Example 2.1. Then, I can be the following:

- Domain $D^I = \{$*red, green, blue, belgium, netherlands, germany, france*$\}$
- The interpretation $border^I = \{(belgium, netherlands), \dots\}$
- The interpretation $coloring^I = \{belgium \mapsto green, \dots\}$

Syntax The syntax of First Order Logic, such as valid terms and formulae, is defined inductively.

A *term* is defined as follows

- a *variable* is a term
- if $t_1, \ldots, t_n$ are terms and f/n is a function, then $f(t_1, \ldots, t_n)$ is term

We say a term t is a *domain term*, if t is in the domain.

A *formula* is defined as follows

- if p/n is a predicate and $t_1, \ldots, t_n$ are terms, then $p(t_1, \ldots, t_n)$ is a formula, called an *atom*
- if ϕ is a formula, then $\neg\phi$ is a formula
- if ϕ is a formula and x is a variable, then $\forall x : \phi$ and $\exists x : \phi$ are formulae
- if ϕ and ψ are formulae, then $\phi \wedge \psi$ and $\phi \vee \psi$ are formulae

We call a *literal* an atom or a negation of an atom. An expressions of the form $a \rightarrow b$ is a shorthand for $\neg a \vee b$, and $a \leftrightarrow b$ is a shorthand for $a \rightarrow b \wedge b \rightarrow a$. A *sentence* is a formula without free, i.e., non-quantified, variables. A set of sentences is called a *theory.*

Example 2.3 (Formula)**.** As in previous examples, we consider the problem of map coloring. The following formula:

$$\forall X, Y\colon \mathit{border}(X, Y) \rightarrow \mathit{coloring}(X) \neq \mathit{coloring}(Y)$$

has an indented meaning of enforcing the country colours to be different.

Semantics The semantics is defined inductively over the structure of the terms and of the formulae.

Let I be a structure and t be a domain term, then the *value* of t is $t^I = d$, where d is the domain element. If t is an expression of the form $f(t_1, \ldots, t_n)$, then its value is $f^I(t_1^I, \ldots, t_n^I) = d$, where d is the domain element.

The truth assignment of a structure I over formulae is defined inductively as follows:

- $p(t_1, \ldots, t_n)^I$ is true, iff $p^I(t_1^I, \ldots, t_n^I)$ is true

- $(\neg\phi)^I$ is true, iff ϕ^I is false
- $(\phi \wedge \psi)^I$ are true, iff ϕ^I and ψ^I are true
- $(\phi \vee \psi)^I$ are true, iff ϕ^I or ψ^I are true
- $\forall X : \phi$ is true, iff for each d in D^I, $\phi[X/d]^I$ is true[1]
- $\exists X : \phi$ is true, iff there is d in D^I such that $\phi[X/d]^I$ is true

We say that I is a *model* of or I *satisfies* a sentence ϕ iff ϕ^I is true, written as $I \models \phi$.

Example 2.4 (Formula evalution)**.** Assume, that domain contains only two countries *belgium* and *netherlands* and two colors *green* and *red*. Let us show that I with *border* being true for (*belgium,netherlands*) and *coloring* mapping *belgium* to *green* and *netherlands* to *red* is a model of the formula in Example 2.3.

$(\forall X, Y\colon \mathit{border}(X,Y) \rightarrow \mathit{coloring}(X) \neq \mathit{coloring}(Y))^I$ iff
for all $d_1, d_2 \in D$ holds $(\mathit{border}(d_1,d_2))^I \rightarrow (\mathit{coloring}(d_1) \neq \mathit{coloring}(d_2))^I$ iff
$d_1 \neq \mathit{belgium}$ and $d_2 \neq \mathit{netherlands}$ the formula holds trivially and if
$d_1 = \mathit{belgium}$ and $d_2 = \mathit{netherlands}$, then $(\mathit{belgium}, \mathit{netherlands})$ in border^I and
$f(\mathit{belgium})^I = \mathit{green} \neq \mathit{red} = f(\mathit{netherlands})^I$ holds since
$\mathit{green} \neq \mathit{red}$ is true

Herbrant Interpretation The *Herbrant Universe* is the set of all terms over the vocabulary without variables. A *Herbrant Interpretaion* has the Herbrant Universe as its domain and interprets each symbol and function by itself. Any model that contains Unique Name Assumption (UNA) (Reiter 1984) and Domain Closure Assumption (DCA) (Reiter 1980) are equivalent to a Herbrant model.

2.2 Answer Set Programming

Answer Set Programming (ASP) is a logic programming paradigm for solving combinatorial and constraint optimization problems (Lifschitz 2008a).

Contrary to the programming language Prolog, which is based on a proof-theoretic approach to answer queries, ASP follows a model generation approach. It has been shown to be effective for a wide range of constraint satisfaction problems (Gebser et al. 2012).

[1] $\phi[X/d]$ is a formula ϕ, where each occurrence of X is substituted with d

The remainder of this section introduces the essentials of ASP in a rather informal way. ASP is a rich (and technical) research area, so we do not focus on technical issues as these would complicate the presentation, but rather refer the interested reader to Eiter et al. (2009), Gebser et al. (2012), Leone et al. (2002), and Lifschitz (2008b) for more details on this. For the actual implementation, we will use the clasp system (Brewka et al. 2011; Gebser et al. 2012).

Definition 2.2.1 (Disjunctive datalog program)**.** A disjunctive datalog program is a finite set of rules of the form:

$$a_1 \vee a_2 \vee \cdots \vee a_n \leftarrow b_1, \ldots, b_k,\ \mathit{not}\ c_1, \ldots,\ \mathit{not}\ c_h$$

where $a_1, \ldots, a_n, b_1, \ldots, b_k, c_1, \ldots c_h$ are atoms of a function-free First Order language L. Each atom is an expression of the form $p(t_1, \ldots, t_n)$, where p is a predicate name and t_i is either a constant or a variable. We refer to the head of rule r as $H(r) = \{a_1, \ldots, a_n\}$ and to the body as $B(r) = B^+(r) \cup B^-(r)$, where $B^+(r) = \{b_1, \ldots, b_k\}$ is the positive part of the body and $B^-(r) = \{c_1, \ldots, c_h\}$ the negative.

If a disjunctive datalog program P has variables, then its semantics are considered to be the same as that of its grounded version, written as *ground(P)*, i.e. all variables are substituted with constants from the Herbrand Universe H_P (the constants occurring in the program). The semantics of a program with variables is defined by the semantics of the corresponding grounded version.

An interpretation I w.r.t. to a program P is a set of ground atoms of P. Let P be a positive disjunctive datalog program (i.e. without negation), then an interpretation I is called closed under P, if for every $r \in \mathit{ground}(P)$ it holds that $H(r) \cap I \neq \emptyset$ whenever $B(r) \subseteq I$.

Definition 2.2.2 (Answer set of a positive program (Eiter et al. 2009))**.** An answer set of a positive program P is a minimal (under set inclusion) interpretation among all interpretations that are closed under P.

Let us introduce the concept of a reduct (Lifschitz 2008b).

Definition 2.2.3 (Gelfond-Lifschitz reduct)**.** A reduct of a ground program P w.r.t. an interpretation I, written as P^I, is a positive ground program P^I obtained by:

- removing all rules $r \in P$ for which $B^-(r) \cap I \neq \emptyset$;
- removing the literals “*not a*” from all remaining rules.

Intuitively, the reduct of a program is a program where all rules with bodies contradicting I are removed and in all non-contradicting all negative ones are ignored. The interpretation I is a guess as to what is true and what is false.

Definition 2.2.4 (An answer set of a disjunctive program)**.** An answer set of a disjunctive program P is an interpretation I such that I is an answer set of positive ground program $\mathit{ground}(P)^I$.

Example 2.5. Consider the following disjunctive datalog program P.

$$a \vee c \leftarrow b. \qquad b \leftarrow a,\ \mathit{not}\ c. \qquad a.$$

If we take the interpretation $I = \{a, b\}$ of P as candidate answer set, then the reduct P^I is

$$a \vee c \leftarrow b. \qquad b \leftarrow a. \qquad a.$$

and it is easily seen that I is a minimal interpretation closed under P^I, and therefore an answer set.

We also use a special form of disjunctive rules called *choice rules* (Gebser et al. 2012):

$$v_1\ \{a_1, a_2, \dots a_n\}\ v_2 \leftarrow b_1, \dots, b_k,\ \mathit{not}\ c_1, \dots,\ \mathit{not}\ c_h$$

where v_1 and v_2 are integer constants. The semantics are as follows: if the body is satisfied, then the number of true atoms in $\{a_1, a_2 \dots a_n\}$ is from v_1 to v_2.

An aggregate atom is an atom that has the following form: $l\#\{a_1, \dots, a_n\}u$ where l and u are constant numbers, each a_i is a literal. The atom is true in an answer set A iff there are from l to u literals a_i that are true in A.

Another construct is *maximization* (Gebser et al. 2012; Leone et al. 2002) (*minimization* is defined analogously) stated as $\#\mathit{maximize}\{a_1 = k_1, \dots, a_n = k_n\}$, where $a_1, \dots, a_n$ are classic literals and $k_1, \dots, k_n$ are integer constants (possibly negative). The semantics of this constraint are as follows: a model I is selected if the weighted sum of $[a_i] * k_i$ is maximal in I, where $[\cdot]$ are Iverson brackets, i.e. $[a]$ is equal to 1 iff a is true in I and 0 otherwise.

Example 2.6 (Map Coloring in ASP)**.** Let us demonstrate how the problem of map coloring discussed before can be expressed in ASP.

The first constraint ensures that the binary predicate *coloring/2* is a function from nodes to colors and the second that if there are two nodes of the same color joined by an edge, then it is not a model.

First Order	**Constraint Programming**	**Answer Set Programming**
variable	–	variable
constant	variable	constant
sentence	constraint	rule/constraint
precicate symbol	Boolean variable array	predicate symbol
function symbol	variable array	function symbol
atomic sentence	–	fact
interpretation	assignment	set of atoms
theory	model	program
model	solution	answer set

Table 2.1: Mapping between terms in ASP, CP and FO

Listing 2.1: ASP encoding of map coloring constraints

```
1 { coloring(X,C) : colors(C) } 1 :- node(X).

:- edge(X,Y), coloring(X,C1), coloring(Y,C2), C1 = C2.
```

2.3 Constraint Programming

A detailed introduction to constraint programming can be found in the ”Handbook of Constraint Programming“ **cp_handbook**

2.4 Comparison between terminologies

Since some readers might be more familiar with Constraint Programming than with Answer Set Programming or First Order Logic, in Table 2.1 we present here terminology comparison **phd_broes**

In this thesis we use Answer Set Programming terminology, unless specified otherwise.

2.5 FO$(\cdot)$ and the IDP system

2.6 Inductive Logic Programming and Relational Pattern Mining

Chapter 3

This is conclusion

...

Instructies van de faculteit:

Algemene besluiten: Verwijzend naar de inleiding en naar de besluiten van de afzonderlijke hoofdstukken worden op het einde van het proefschrift de voornaamste besluiten gebundeld. Hier wordt de nadruk gelegd op de eigen inbreng, de verworven resultaten, de ‘stellingen’ van het proefschrift en de originele bijdragen tot het onderzoeksdomein. De onopgeloste problemen worden aangestipt en suggesties voor eventueel verder onderzoek worden gemaakt.

Appendix A

This is myappendix

...

Instructies van de faculteit:

De appendices: ze omvatten alle gedeelten uit de tekst die weliswaar essentieel zijn voor het proefschrift, maar waarvan de inlassing in de tekst de leesbaarheid ervan nadelig zouden beïnvloeden bv. omwille van hun lengte. Zo kunnen bv. de brute meetresultaten of een computerprogramma met zijn bron, commentaar en voorbeelden beter thuishoren in een appendix dan in de tekst zelf. De appendices kunnen desgevallend worden gebundeld in een apart boekdeel.

Bibliography

Brewka, G., T. Eiter, and M. Truszczyński (2011). "Answer set programming at a glance". In: *Communications of the ACM* 54.12, pp. 92–103.

Eiter, T., G. Ianni, and T. Krennwallner (2009). "Answer Set Programming: A Primer". In: *5th International Reasoning Web Summer School (RW 2009), Brixen/Bressanone, Italy, August 30 – September 4, 2009.* Vol. 5689. LNCS. Springer.

Enderton, H. B. (2001). *A mathematical introduction to logic.* 2nd ed. Harcourt/Academic Press.

Gebser, M., R. Kaminski, B. Kaufmann, and T. Schaub (2012). *Answer Set Solving in Practice.* Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers.

Kowalski, R. (1979). "Algorithm = Logic + Control". In: *Commun. ACM* 22.7, pp. 424–436. DOI: 10.1145/359131.359136.

Leone, N., G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello (2002). "The DLV System for Knowledge Representation and Reasoning". In: *ACM Transactions on Computational Logic* 7, pp. 499–562.

Lifschitz, V. (2008b). "What Is Answer Set Programming?" In: *Association for the Advancement of Artificial Intelligence*, pp. 1594–1597.

– (2008a). "What Is Answer Set Programming?." In: *AAAI.* Vol. 8, pp. 1594–1597.

Reiter, R. (1980). "Equality and Domain Closure in First-Order Databases". In: *J. ACM* 27.2, pp. 235–249.

– (1984). "Towards a Logical Reconstruction of Relational Database Theory". In: *On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases, and Programming Languages.* Ed. by M. L. Brodie, J. Mylopoulos, and J. W. Schmidt. New York, NY: Springer New York, pp. 191–238.

Instructies van de faculteit:

De bibliografie. Departementale richtlijnen terzake te volgen.

This is curriculum

...

Instructies van de faculteit:

Beknopt CV van de doctorandus.

List of publications

Input file chapters/publications/publications.tex does not exist. Make sure its starts with “\chapter{List of publications}”. To not include this chapter in the table of contents, use the starred version of the \chapter command...

Instructies van de faculteit:

Lijst van de publicaties door de doctorandus/a (auteur of co-auteur).

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
DTAI
Celestijnenlaan 200A
B-3001 Leuven
sergey.paramonov@kuleuven.be
sergey-paramonov.com