Mining Multi-Relational Gradual Patterns

NhatHai Phan* Dino Ienco† Donato Malerba‡ Pascal Poncelet§
Maguelonne Teisseire¶

Abstract

Gradual patterns highlight covariations of attributes of the form *"The more/less X, the more/less Y"*. Their usefulness in several applications has recently stimulated the synthesis of several algorithms for their automated discovery from large datasets. However, existing techniques require all the interesting data to be in a single database relation or table. This paper extends the notion of gradual pattern to the case in which the co-variations are possibly expressed between attributes of different database relations. The interestingness measure for this class of "relational gradual patterns" is defined on the basis of both Kendall's τ and gradual supports. Moreover, this paper proposes two algorithms, named *τRGP Miner* and *gRGP Miner*, for the discovery of relational gradual rules. Three pruning strategies to reduce the search space are proposed. The efficiency of the algorithms is empirically validated, and the usefulness of relational gradual patterns is proved on some real-world databases.

1 Introduction

Nowadays, most of information systems are based on the relational database technology. The logical models of the data are sets of relations or tables possibly linked by foreign key constraints. This contrasts with the usual practice in Data Mining of organizing data in a single relation when they are analyzed. Relational data mining approaches [6] are characterized by both their direct applicability to "multi-relational data" (MRD) and their capability of looking for patterns which involve multiple database relations.

Most of the studies on relational data mining focus on relational patterns at the tuple level, i.e., they express relationships between tuples of different database relations. Relational association rules [4], relational naïve Bayesian classifiers [7], relational regression models [1] and relational subgroups [17], all express patterns as either SQL queries or first-order logic clauses with constraints between tuples or facts. Similarly, the probabilistic relational models [8] define a distribution over a set of instances of a schema, and consider the structure at the level of attribute values.

In this paper, we focus on patterns expressing the relational structure at the attribute level. We consider relational extensions of the class of gradual dependencies which express covariations of attributes of the form *"The more/less X, the more/less Y"* [10]. This class of patterns has a wide range of applications [3, 5, 18, 19, 20]. For instance, with reference to the financial database in Figure 1, the gradual pattern *"the higher the average salary in a district, the bigger the deals made by inhabitants living in the district"* could be useful for business planning. While the pattern *"the smaller the district, the longer the duration of the loans"* provides financial promoters with some useful insights. These two examples show gradual dependencies between attributes from different relations (i.e., the average salary, the size of districts, the deal, and the loan). Hence they could be discovered only in a MRD setting.

To discover this kind of patterns, we first introduce the concept of *multi-relational gradual pattern* (*multi-rgp*), and its associated support measures based on *Kendall's τ* [3] and *gradual support* [5]. Then we propose an algorithm, named *RGP-Miner*, to discover multi-rgps directly from MRDs. Mining the complete set of multi-rgps is a non-trivial task since the size of the search space is exponential in the number of numerical attributes of the multi-relational database. To tame computational complexity, we design three efficient rules named *Complementary Pruning*, *Apriori Pruning*, and *Backward Pruning* to shrink the search space and prevent unnecessary computations. Experiments conducted on real datasets demonstrate the pattern meaning, effectiveness, and efficiency of our proposed approaches.

2 Problem Statement

In this section, we formalize the data model, the notion of relational gradual patterns, and its examples.

*University of Oregon. haiphan@cs.uoregon.edu
†Irstea Montpellier, France. dino.ienco@teledetection.fr
‡University of Bari, Italy. donato.malerba@uniba.it
§University of Montpellier 2, France. pascal.poncelet@lirmm.fr
¶Irstea Montpellier, France. teisseire@teledetection.fr

2.1 Multi-Relational Data. In this work, we assume that a database *db* consists of a set of tables, $db = \{T^1, \ldots, T^n\}$, each of which has a schema $S(T^i) = (PK^i, \mathcal{FK}^i, \mathcal{A}^i)$ consisting of a *primary key* PK^i, possibly some *foreign keys* (set $\mathcal{FK}^i$), and at least one *attribute* (set $\mathcal{A}^i$). For instance, with reference to Figure 1, the *Loan* table has a key loan-ID, a foreign key *account-ID* and attributes *date, amount, duration,* etc.

Foreign keys define the only possible joins between two tables. Without loss of generality, we assume that any pair of tables has at most one foreign key linking them. Indeed, databases can always be losslessly recoded such that this assumption holds.

Moreover, we assume that all attributes have a numerical domain, since we are not interested in finding patterns among categorical attributes. The numerical domain of the attribute A^i_j is denoted as $Dom(A^i_j)$. Similarly, the domain of table T^i is denoted by $Dom(T^i)$, and corresponds to the Cartesian product of all domains involved, i.e.,

(2.1)
$$Dom(T^i) = Dom(PK^i) \prod_{\mathcal{FK}^i_j \in \mathcal{FK}^i} Dom(\mathcal{FK}^i_j) \prod_{A^i_j \in \mathcal{A}^i} Dom(A^i_j)$$

Each table is also associated with a set of tuples which also called the *table extension*. Henceforth, the distinction between the table and its extension is blurred, and we use the notation $t \in T^i$ to indicate that a tuple t is in extension of table T^i.

The database as whole should satisfy *referential integrity*, i.e., foreign-key values in a tuple refer to existing tuples in the table for which this foreign key refers to. In formal, if $t \in T^i$ and $S(T^i) = (PK^i, \mathcal{FK}^i, \mathcal{A}^i)$, then for each $\mathcal{FK}^i_j \in \mathcal{FK}^i$ whose domain is $Dom(PK^l)$, the following condition must hold: there exists a tuple $t' \in T^l$ such that $\pi_{\mathcal{FK}^i_j}(t) = \pi_{PK^l}(t')$, where π is the usual projection operator of relational algebra.

2.2 Multi-Relational Gradual Pattern. In the case of a (single) table, an itemset boils down to a selection [11] [15]. However, in relational gradual pattern context, a selection on an attribute cannot reflect the attribute graduality. The main reason is that the value of a specific attribute A^i_l (resp. set of attributes) must change following a regular rule (e.g., *"A^i_l is increasing or decreasing"*) over the tuples. Therefore, a common selection in which attribute values are selected by some simple operations (e.g., $>, <, =$) cannot present the attribute graduality in our context.

To address the issue, we propose the definition of relational gradual patterns in which the graduality is covered. Let us denote the increase and decrease of an attribute A^i_j as $A^{i>}_j$ and $A^{i<}_j$. Let E^i is the set of all single gradual attributes which are generated by $\mathcal{A}^i$ from table T^i, E^i can be defined as follows:

$$E^i = \bigcup_{j=1}^{|\mathcal{A}^i|} \{A^{i<}_j\} \cup \{A^{i>}_j\} \tag{2.2}$$

In a single table T^i, a relational gradual pattern q^i is a combination of gradual attributes in E^i. The set of all potential gradual patterns q^i are called Q^i. In addition, "proper" relational gradual patterns could be generated from gradual attributes which are from different tables. If the gradual attributes in a pattern are from a single table T^i, then the pattern is called a *mono-rgp*. If the gradual attributes are from two tables T^i and T^j then the pattern is called a *bi-rgp*. Finally, a *multi-rgp* can contain gradual attributes from multiple tables in MRDs. The definition of relational gradual patterns can be defined as follows:

DEFINITION 2.1. *(Relational Gradual Pattern). Let $db = \{T^1, \ldots, T^n\}$ be a database for which each table T^i has a schema $S(T^i) = (PK^i, \mathcal{FK}^i, \mathcal{A}^i)$.*

1. *Mono-relational gradual pattern (mono-rgp):* $p = q^i$ *where* $q^i \in Q^i$
2. *Bi-relational gradual pattern (bi-rgp) between T^i, T^j:*

   a. *if T^i:T^j is a 1:N relation then* $p = q^i q^j$ *where* $q^i \in Q^i$, $q^j \in Q^j$ *and* $PK^i \subseteq \mathcal{FK}^j$

   b. *if T^i:T^j is a M:N relation then* $p = q^i q^j$ *where* $q^i \in Q^i$, $q^j \in Q^j$ *and* $\exists T^c \in db$ *s.t.* $\left(PK^i \subseteq PK^c,\ PK^j \subseteq PK^c\right)$ *or* $\left(PK^i \subseteq \mathcal{FK}^c,\ PK^j \subseteq \mathcal{FK}^c\right)$
3. *Multi-relational gradual pattern (multi-rgp).* $p = q^i q^{i+1} \ldots q^m$ *where* $\forall j \in \{i, \ldots, m-1\} : q^j \in Q^j$ *and* $q^j q^{j+1}$ *is a bi-rgp*

In the Definition 2.1, we define the patterns that can be extracted in a single relation, in two relations, and in multi-relations. The combination of two mono-rgps from two relations become a bi-rgp. The relations from T^i to T^j in *2.a* and *2.b* respectively are 1:N and M:N relationships. Note that, in *2.b*, the table T^c is a connection relation between T^i and T^j. Finally, a multi-rgp is a generalization of the bi-rgp so that every consecutive pairs of mono-rgps is a bi-rgp.

For instance, in Figures 1 and 2, potential interesting multi-rgps are as follows:

Bi-rgp from a 1:N relation. In table Loan, we have a mono-rgp such as $q^2 = \{Amount^>, Duration^>\}$. Additionally, we can have a bi-rgp by connecting table Loan to table Account over the foreign key $account_{ID}$ (i.e. $Account.PK = Loan.\mathcal{FK}$). A potential bi-rgp is $p = q^1 q^2 = \{Frequency^>, Amount^>, Duration^>\}$ where

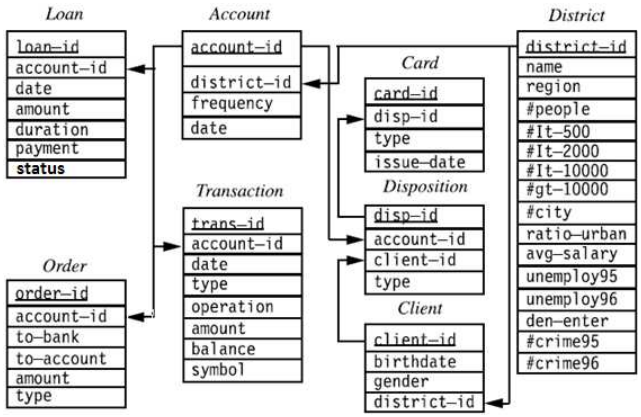

Figure 1: A Financial database (from PKDD CUP 99).

T¹ = ACCOUNT			T² = LOAN				
$account_{ID}$	Frequency	Date	$loan_{ID}$	$account_{ID}$	Date	Amount	Duration
10	2	06/2007	30	10	06/2008	10245	12
11	3	03/2006	31	10	09/2008	13722	24
12	3	06/2006	32	11	08/2006	27313	36
13	1	04/2006	33	12	09/2006	27147	28
14	5	05/2007	34	12	05/2008	27194	36
15	4	03/2008	35	13	06/2008	5203	6

Figure 2: An example database.

$q^1 = \{Frequency^{>}\} \in Q^{Account}$. The pattern can be explained as *"the more frequently the accounts are used, the more amount and the longer duration loans are"*.

Bi-rgp from a M:N relation. We have $PK^{Client} \subseteq \mathcal{FK}^{Disposition}$ and $PK^{Account} \subseteq \mathcal{FK}^{Disposition}$. Thus, the relation between table Client and table Account is M:N. In addition, the table Disposition is a connection table between them. A a result, a potential bi-rgp is $p = \{Birthdate^{<}, Frequency^{>}\}$ where $Birthdate^{<} \in Q^{Client}$ and $Frequency^{>} \in Q^{Account}$. The pattern can be explained as *"the older clients are, the more frequently they use their accounts"*.

Multi-rgp. A potential multi-rgp which could cross the three tables District, Account, and Loan is $p = \{\#People^{>}, Frequency^{>}, Amount^{>}\}$ where $\#People^{>} \in Q^{District}$, $Frequency^{>} \in Q^{Account}$, and $Amount^{>} \in Q^{Loan}$. The pattern can be explained as *"for the habitants living in the districts which have more inhabitants, the more frequently their accounts are used, and the more amount of loans they have"*.

3 Pattern Occurrences

The occurrences of common relational patterns can be a set of tuples (e.g., as in [15]) or a set of sets of sets of ... of tuples (e.g., as in [11]). However, defining occurrences of the multi-rgp can become more challenging. In essence, a single tuple cannot reflect the graduality of the patterns. It requires at least two tuples together to evaluate the graduality of the attributes. A gradual tuple pair can be defined as follows:

DEFINITION 3.1. *(Gradual Tuple Pair). Given a mono-rgp $q^i = \{A_j^{i\,*_j}, \ldots, A_n^{i\,*_n}\} \in Q^i$ with $*_j, \ldots, *_n \in \{<, >\}$. A pair of distinct tuples (t, t') s.t. $t, t' \in T^i$, is gradual in respect to q^i if:*

$$\forall A_l^{i\,*_l} \in q^i : \pi_{A_l^i}(t) *_l \pi_{A_l^i}(t') \tag{3.3}$$

Given a multi-rgp $p = q^i q^{i+1} \ldots q^m$, a pair of tuples (t, t') s.t. $t, t' \in (T^i \bowtie T^{i+1} \bowtie \ldots \bowtie T^m)$, is gradual in respect to p if:

$$\forall q^l \in p : \left(\pi_{T^l}(t), \pi_{T^l}(t')\right) \text{ respects } q^l \tag{3.4}$$

The occurrences of relational gradual patterns can thus contain sets of gradual tuple pairs.

DEFINITION 3.2. *(Pattern Occurrence). Let $db = \{T^1, \ldots, T^n\}$ be a MRD. Given a pattern $p = q^i q^{i+1} \ldots q^m$, the occurrence of p, denoted $occ(p)$, is a set of gradual tuple pairs (t, t') in respect to p. Note that $t, t' \in (T^i \bowtie T^{i+1} \bowtie \ldots \bowtie T^m)$.*

It is illustrative to consider what the domain of such patterns is. That is, *"what does an instance look like?"* In essence, the definition of these domains follows the structure of the pattern occurrence.

DEFINITION 3.3. *(Pattern Occurrence Domain). Let $db = \{T^1, \ldots, T^n\}$ be a database. Given a multi-rgp $p = q^i q^{i+1} \ldots q^m$, the domain of p's occurrences, denoted $DomOcc(p)$, is given by*

$$DomOcc(p) = (T^i \bowtie T^{i+1} \bowtie \ldots \bowtie T^m)^2 \tag{3.5}$$

For brevity, we only report the primary key and foreign keys in a pattern occurrence. For instance, given a mono-rgp q^1=$\{Frequency^{>}\} \in Q^{account}$ (Figure 3). One of the occurrences of q^1 is ($account_{ID}$, $account_{ID}$)=$\{10, 11\}$. The occurrences of q^1, denoted $occ(Frequency^{>})$, are illustrated in Figure 3. In similar, we have the occurrences of $q^2 = \{Amount^{>}\} \in Q^{Loan}$, denoted $occ(Amount^{>})$. If we have a bi-rgp $p = q^1 q^2$ then an occurrence of p is $\big((t_1, t_2)$=$(10, 11), (t'_1, t'_2)$=$(30, 32)\big)$ where (t_1, t_2)=$(10, 11) \in occ(q^1)$, (t'_1, t'_2)=$(30, 32) \in occ(q^2)$ and $\pi_{account_{ID}}(t_1)$=$\pi_{account_{ID}}(t'_1)$=10, $\pi_{account_{ID}}(t_2)$=$\pi_{account_{ID}}(t'_2)$=11. Obviously, we can compute the occurrences of p which are illustrated as the links between $occ(Frequency^{>})$ and $occ(Amount^{>})$.

4 Pattern Supports

There are different support definitions (i.e., *Kendall's* τ support [3], and *gradual* support [5]) for mono-rgps. In spite of differences, they share the same hypothesis which is pattern support, denoted σ, is the number of pattern occurrences over all possible cases:

$$\sigma(p) = \frac{|occ(p)|}{|all\ possible\ cases|} \tag{4.6}$$

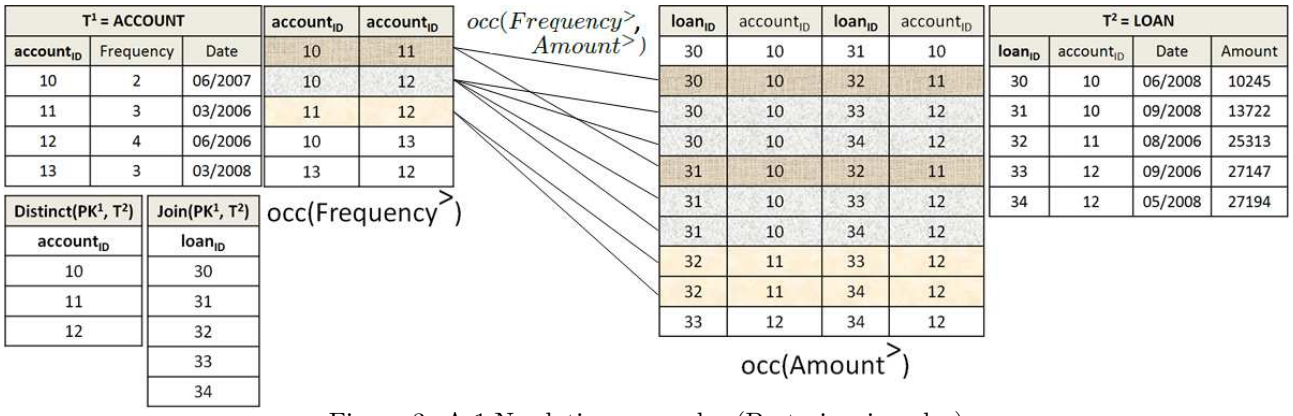

Figure 3: A 1:N relation example. (Best view in color)

Fortunately, the support measure for multi-rgps is not different from this point of view. Since we have already presented the pattern occurrences, the challenging becomes how we identify the "$|$*all possible cases*$|$" given a multi-rgp. Given a bi-rgp, the number of $|$*all possible cases*$|$ depends on the types of the relations (i.e., 1:N or M:N). It becomes even more complicated for multi-rgps which contain a series of bi-rgps.

To breakthrough the challenge, we propose a novel *Kendall's* τ-based support to measure the graduality of multi-rgps. Let us first propose a set of operations which will be used to define the support measure.

- $distinct(PK^i, T^j)$

  Given two tables T^i and T^j s.t. $PK^i \subseteq \mathcal{FK}^j$: 1) $distinct(PK^i, T^j)$ returns a set of *"distinct"* values $pk \in PK^i$ s.t. $\exists t' \in T^j : pk \subseteq \pi_{\mathcal{FK}^j}(t')$.
- $count(pk, T^j)$ returns the number of tuples $t' \in T^j$ s.t. $pk \subseteq \pi_{\mathcal{FK}^j}(t')$.
- $distinct(PK^m, p)$

  Given a multi-rgp $p = q^i \dots q^m$, $distinct(PK^m, p)$ is a set of *"distinct"* PK^m values which are involved in the occurrence domain of p.

In a single table context, $|$*all possible cases*$|$ of a mono-rgp $p = q^i$ actually is the number of randomly chosen pairs of tuples (t, t') s.t. $t, t' \in T^i$ and (t, t') possibly respects the pattern p. Indeed, the $|$*all possible cases*$| = \binom{|T^i|}{2}$ and not $|T^i|(|T^i| - 1)$ since it is not possible that (t, t') and (t', t) support p at the same time. The support of mono-rgps can be defined as follows:

Definition 4.1. *(Kendall's τ support of mono-rgps). Given a MRD db $= \{T^1, \dots, T^n\}$, and a mono-rgp $p = q^i$. The pattern support can be defined as follows:*

$$\sigma_\tau(p) = \frac{|occ(p)|}{\binom{|T^i|}{2}} \tag{4.7}$$

Regarding to bi-rgp $p = q^i q^j$ with the relation T^i:T^j is 1:N, the $|$*all possible cases*$|$ intuitively is $\binom{|distinct(PK^i, T^j)|}{2}$. This is because we need to join two tables via PK^i. In addition, only the values of PK^i which exist in table T^j can support p by respecting the mono-rgp q^j. The number of these primary key values is $|distinct(PK^i, T^j)|$. However, it is different from mono-rgps, there are some cases needed to be eliminated from $\binom{|distinct(PK^i, T^j)|}{2}$. In mono-rgps, the values of PK^i are distinct meanwhile they are not distinct after joining two tables T^i and T^j in bi-rgps. If two tuples in $T^i \bowtie T^j$ have the same value of PK^i then the combinations of them cannot respect the mono-rgp q^i in p. Therefore, we need to eliminate these combinations from the $|$*all possible cases*$|$. All of these combinations can be computed as $\sum_{pk \in distinct(PK^i, T^j)} \binom{count(pk, T^j)}{2}$.

Definition 4.2. *(Kendall's τ support of bi-rgps for 1:N relations). Given a MRD db $= \{T^1, \dots, T^n\}$, and a bi-rgp $p = q^i q^j$ s.t. T^i:T^j is 1:N. The pattern support can be defined as follows:*

$$\sigma_\tau(p) = \frac{|occ(p)|}{\binom{|distinct(PK^i, T^j)|}{2} - \sum_{pk \in distinct(PK^i, T^j)} \binom{count(pk, T^j)}{2}} \tag{4.8}$$

Regarding to bi-rgp $p = q^i q^j$ with the relation T^i:T^j is M:N, the $|$*all possible cases*$|$ can be simply computed as $\binom{|distinct(PK^i, T^c)|}{2} \times \binom{|distinct(PK^j, T^c)|}{2}$ which is total possible combinations of the values of PK^i and PK^j. T^c is the connection table between T^i and T^j.

Definition 4.3. *(Kendall's τ support of bi-rgps for M:N relations). Given a MRD db $= \{T^1, \dots, T^n\}$, and a bi-rgp $p = q^i q^j$ s.t. T^i:T^j is M:N. The pattern support can be defined as follows:*

$$\sigma_\tau(p) = \frac{|occ(p)|}{\binom{|distinct(PK^i, T^c)|}{2} \times \binom{|distinct(PK^j, T^c)|}{2}} \tag{4.9}$$

The support of multi-rgps can be considered as a generalization of bi-rgps. Given a multi-rgp $p = q^i \dots q^m$, p can be rewrote as $p = (q^i \dots q^{m-1})q^m$. In fact, $(q^i \dots q^{m-1})$ can be considered as a mono-rgp whose domain is $T^i \bowtie \dots \bowtie T^{m-1}$ and q^m is another mono-rgp. So, the $|$*all possible cases*$|$ clearly depends on the relationship between two tables T^{m-1} and T^m. As a result, the Definitions 4.2, 4.3 can be utilized to compute the support for multi-rgps. What we need is to identify values of the primary key PK^{m-1} which exist in the $occ(q^i \dots q^{m-1})$. The set of these values are denoted as $PK^{p'} = distinct(PK^{m-1}, p')$ where $p' = q^i \dots q^{m-1}$. The support of multi-rgps can be defined as follows:

DEFINITION 4.4. *(Kendall's τ support of multi-rgps). Let $db = \{T^1, \dots, T^n\}$ be a database, the support of multi-rgps $p = q^i \dots q^{m-1}q^m$ can be defined as follows:*

1) if T^{m-1}:T^m is 1:N

$$\sigma_\tau(p) = \frac{|occ(p)|}{\binom{|distinct(PK^{p'}, T^m)|}{2} - \sum_{pk \in distinct(PK^{p'}, T^m)} \binom{count(pk, T^m)}{2}}$$

2) if T^{m-1}:T^m is M:N and T^c is the connection table

$$\sigma_\tau(p) = \frac{|occ(p)|}{\binom{|distinct(PK^{p'}, T^c)|}{2} \times \binom{|distinct(PK^m, T^c)|}{2}}$$

In practice, to reduce the memory consumption we index only the tuples exist in the occurrences of $(q^i \dots q^{m-1})$. The following theorem gives the expected σ_τ, denoted $E[\sigma_\tau]$, in the case of statistically independent attributes. This expected support can be used as a reference point to assess the quality of a multi-rgp.

THEOREM 4.1. *($E[\sigma_\tau]$). Given a set of statistically independent gradual attributes $s = \{A_1, A_2, \dots, A_n\}$, $P_s = \{p_1, \dots, p_m\}$ is the set of all possible independent multi-rgps generated from s. The expected $supp_\tau$ of a pattern $p \in P_s$ is $E[\sigma_\tau](p) = \frac{1}{(2^{n-1}+n(3^{n-1}-1)+1)}$.*

where p and p' are independent multi-rgps if $(DomOcc(p) = DomOcc(p')) \wedge (occ(p) \cap occ(p') = \emptyset)$.

The proof of Theorem 4.1 is available at `https://sites.google.com/site/ihaiphan/`. Until now, we have proposed the definition of *Kendall's* τ support of multi-rgps. In [5], Di-Jorio et. al. propose a *gradual* support measure to emphasize the consecutiveness in changing of attributes over the values. Instead of a pair of tuples, the *gradual* support concerns on a list of tuples $\mathcal{L} = \{t_i, t_{i+1}, \dots, t_k\}$ in which $\forall t_j, t_{j+1} \in \mathcal{L} : (t_j, t_{j+1})$ is a gradual tuple pair (Def. 3.1). In the *gradual* support, $|occ(p)|$ becomes the size of the maximal list of tuples $\mathcal{L}$ which supports p in the MRD *db*. The $|$*all possible cases*$|$ becomes the possible longest $\mathcal{L}$ in the *db*. Based on this idea, we also propose the *gradual* support, denoted $\sigma_g(p)$, for multi-rgps.

5 Multi-Relational Gradual Pattern Miner

Extracting the complete set of multi-rgps is a non-trivial task. At first glance, the number of potential multi-rgps is exponential, i.e., approximately $2^{2\times|A_{db}|}$ where A_{db} is a set of all numerical attributes in the MRD *db*.

Key idea. Facing the huge potential search space, we propose an approach, named *RGP* Miner. In *RGP* Miner, we first extract *mono-rgp*s from single tables by applying *GRITE* algorithm [5]. Then the MRD *db* is transformed into a graph $G = (V, E)$ where V is a set of vertices and E is a set of edges (Figure 4). Each vertex $v_i \in V$ stands for $T^i \in db$ and each edge e_{ij} can be considered as a relationship between two tables T^i and T^j. Next, we apply a depth-first search on the set of vertices V following E to combine *mono-rgp*s together to be *multi-rgp*s. To avoid unnecessary computations and searches, we propose three pruning rules which are *Complementary Pruning*, *Apriori Pruning*, and *Backward Pruning*. Complementary Pruning is used to avoid the support computation for *complement patterns*. Then, Apriori Pruning is used to eliminate *infrequent patterns*. Finally, Backward Pruning avoids computing redundant patterns which have been traversed before.

Before presenting our algorithm, we first give definitions of *complement* and *frequent* patterns. Given two gradual attributes $A_j^{i>}$ and $A_j^{i<}$ which are from table T^i, we have $\sigma_\tau(A_j^{i>}) = \sigma_\tau(A_j^{i<})$. This is because $\forall t, t' :$ if $(t, t') \in occ(A_j^{i>})$ then $(t', t) \in occ(A_j^{i<})$ and vice versa. $A_j^{i>}$ and $A_j^{i<}$ are called complement each other. Two patterns p and p' are called complement each other if for all gradual attributes A_j^{i*} in p there is a complement attribute of A_j^{i*} in p' and vice versa. We also have $\sigma_\tau(p) = \sigma_\tau(p')$. Therefore, we can infer the support of a pattern from its complement.

In fact, the number of multi-rgps is exponential. So we only focus on frequent multi-rgps and not all of them. Let us define the frequent multi-rgps as follows:

DEFINITION 5.1. *(Frequent Pattern). Given a predefined minimal support threshold σ_{min}, and a multi-rgp $p = q^i q^{i+1} \dots q^m$. p is an frequent pattern if:*

$$\forall j \in \{i, \dots, m\} : \sigma_\tau(q^i \dots q^j) \geq \sigma_{min} \quad (5.10)$$

In order to start the depth-first search process, vertices in V are collected into two different categories, source vertices and leaf vertices. The source vertices are not pointed from the other vertices while the leaf vertices do not point to any other nodes. The search process is started with source vertices. Note that the vertex v_{m+1} can be added into the node $r = \{v_i, \dots, v_m\}$ if there is an edge from v_m to v_{m+1}.

Let us define some useful notations which will be used in our proposed algorithm as follows:

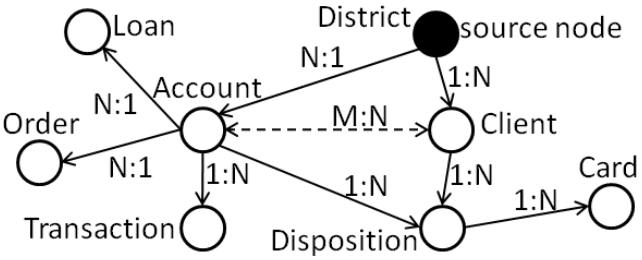

Figure 4: Graph of the Financial MRD in Fig 1.

- A set of frequent multi-rgps Q^r:

  Given a current node $r = \{v_i, v_{i+1}, \ldots, v_n\}$, Q^r returns the set of all frequent multi-rgps which are extracted from combining frequent mono-rgps in $Q^{v_i}, \ldots, Q^{v_n}$.

- Merging two nodes, $C^{r\cup r'} = (Q^r \bowtie Q^{r'}) \cup Q^r \cup Q^{r'}$:

  Given two nodes r and r', $C^{r\cup r'}=(Q^r \bowtie Q^{r'}) \cup Q^r \cup Q^{r'}$ contains all potential multi-rgps by merging r and r'.

- Attaching two multi-rgps, $p \bullet p'$:

  Given two multi-rgps $p = q^i, \ldots, q^m$ and $p' = q^j, \ldots, q^n$, we have $p \bullet p' = q^i, \ldots, q^m q^j, \ldots, q^n$.

PROPERTY 5.1. *(Complementary Pruning). Given two complement patterns p and p', we have: $\sigma(p) = \sigma(p')$.*

Proof. We have $\forall(t_1, t_2) \in occ(p)$ then $(t_2, t_1) \in occ(p')$. Thus, $|occ(p)| = |occ(p')|$. In addition, $DomOcc(p) = DomOcc(p')$. So the $|all\ the\ cases|$s of p and p' are the same. Consequently, $\sigma_\tau(p) = \sigma_\tau(p')$.

By applying Complementary Pruning, we can avoid computing the supports of complement patterns. Even though Complementary Pruning rule can eliminate the computation of a large number of candidates, there are many other candidates need to be ignored to shrink the search space. In essence, if a multi-rgp is not frequent then its extensions are not frequent as well. Therefore, we can eliminate those extensions. Apriori Pruning rule can be defined as follows:

PROPERTY 5.2. *(Apriori Pruning). Given two multi-rgps p and p', the following holds: if p is not frequent then $p \bullet p'$ is not frequent as well.*

Proof. We have if $p = \{q^i, \ldots, q^m\}$ is not frequent then $\exists j \in \{i, \ldots, m\}$ s.t. $\sigma_\tau(q^i \ldots q^j) < \sigma_{min}$. In addition, $p \subseteq p \bullet p'$ and thus $q^i \ldots q^j \subseteq p \bullet p'$. So, $\exists j \in \{i, \ldots, |p \bullet p'|\}$ s.t. $\sigma_\tau(q^i \ldots q^j) < \sigma_{min}$. As a result, $p \bullet p'$ is not frequent (Definition 5.1).

For instance, when we add new vertex *loan* into a current route $\{district, account\}$ (e.g., step 4^{th} in our depth-first search algorithm, Figure 5) we apply Complementary Pruning and Apriori Pruning to avoid unnecessary computations and infrequent patterns. Regarding to the route, there could be redundant routes in the search space. For instance, at step 10^{th}, the multi-rgp candidates in $Q^{district} \bowtie Q^{account} \bowtie Q^{disposition}$ had been evaluated in step 7^{th}. So they are redundant. Let us define *Backward Pruning rule* to ignore this type of redundant patterns.

PROPERTY 5.3. *(Backward Pruning). Given a current node $r = \{v_i, \ldots, v_n\}$, $r_a = r \setminus \{v_n\}$ is called ancestor of r, an already traversed node r'. A set of pattern candidates we need to evaluated $\mathcal{F}^r$ at the step r can be computed as follows:*

if $dest(r') = dest(r) \wedge comm(r_a, r') \neq \emptyset$
then $\mathcal{F}^r = (Q^{r_a \cap r'} \bowtie Q^{r_a \setminus r'} \cup Q^{r_a \setminus r'}) \bowtie Q^{v_n}$

Proof. We have $C^r=(Q^{r_a} \bowtie Q^{v_n}) \cup Q^{r_a} \cup Q^{v_n}$. Since Q^{r_a} and Q^{v_n} had been evaluated at previous steps, the candidates now become $\mathcal{F}^r=Q^{r_a} \bowtie Q^{v_n}=\big((Q^{r_a\cap r'} \bowtie Q^{r_a\setminus r'}) \cup (Q^{r_a\cap r'} \cup Q^{r_a\setminus r'})\big) \bowtie Q^{v_n} = (Q^{r_a\cap r'} \bowtie Q^{r_a\setminus r'} \cup Q^{r_a\setminus r'}) \bowtie Q^{v_n} \cup (Q^{r_a\cap r'} \bowtie Q^{v_n})$. We have $r_a \cap r' \cup v_n \subseteq r'$, thus $Q^{r_a\cap r'} \bowtie Q^{v_n}$ had been evaluated at the step r'. Consequently, $\mathcal{F}^r = (Q^{r_a\cap r'} \bowtie Q^{r_a\setminus r'} \cup Q^{r_a\setminus r'}) \bowtie Q^{v_n}$.

In the Backward Pruning rule, the first part of $\mathcal{F}^r$ (i.e., $Q^{r_a\cap r'} \bowtie Q^{r_a\setminus r'} \cup Q^{r_a\setminus r'}$) had been evaluated as the steps r_a and r'. Therefore, we do not need to reevaluated them at the step r.

By applying Backward Pruning, we can discard a number of redundant patterns when there is another route passing the same destination with the current route. For instance, at step 10^{th}, we have the current node $r = \{district, account, client, disposition\}$. In essence, $Q^{\{district,account,disposition\}}$ has been evaluated before at the node labeled (7). Thus the pattern candidates in $Q^{\{district,account\}} \bowtie Q^{disposition}$ are redundant. Therefore, these patterns need to be pruned.

Backward Pruning is efficient in the sense that we only need to do the pruning once at the step r and not for its subtrees. This is because all the redundant patterns have been discarded from Q^{r_a} and thus Q^r-based Cartesian product can be applied for all the subtrees of r. To obtain the patterns, at each node $r = \{v_i, \ldots, v_n\}$, the set of frequent patterns in Q^r which includes mono-rgps from Q^{v_n} will be reported. For instance, at the step 3^{rd}, only frequent patterns which contain mono-rgps from $Q^{account}$ will be reported.

All the Theorems are still true for the σ_g measure. We will omit the proofs for σ_g since they are very similar to the reported ones. In addition, the RGP Miner algorithm has two instances, i.e. τRGP Miner and gRGP Miner. The pseudo code of τRGP Miner is presented in Algorithm 5.1. The main difference in gRGP Miner is that we need to navigate a binary matrix of orders for each pattern to obtain the longest list of gradual tuples. In fact, we apply the matrix navigation function presented in the *GRITE* algorithm [5].

ALGORITHM 5.1. **τRGP Miner**

Input: MRD db, source nodes S, G=$\{V, E\}$, σ_{min}
Output: all frequent multi-rgps

```
1  begin
2      R := ∅;
   foreach v ∈ S do
3      r := {v};
       RGpattern(db, r, v, G, σ_min);
4  RGpattern(db, r, v, G, σ_min)
begin
5      foreach node l ∈ v.edges do
6          combining(r, v, l);
7      output Q^v; delete Q^v; R := R ∪ r;
8  combining(route r, node v, node l)
begin
9      if type(v, l) = 1 : N then
10         denom := (|join(PK^v, l)| choose 2);
11     else
12         denom := (|distinct(PK^v, k)| choose 2) × (|distinct(PK^l, k)| choose 2);
13     temp = ∅;
   foreach pattern q^i ∈ Q^v do
14         if Backward(q^i, l, r, R) = true then
15             if exist(db, q^i) = false then
16                 create(db, q^i);
17             foreach pattern q^j ∈ Q^l do
18                 if complement(q^i q^j, temp) = false then
19                     create(db, q^j);
                   occ := |q^i ⋈ q^j|;
                   σ_τ(q^i q^j) := occ / denom;
                   if σ_τ(q^i q^j) ≥ σ_min then
20                     temp := temp ∪ q^i q^j;
21                 delete q^j;
22                 else
23                     temp := temp ∪ q^i q^j;
24         delete q^i;
25     Q^l := Q^l ∪ temp;
   RGpattern(db, r ∪ l, l, G, σ_min);
```

26 $type(v, l)$ is the connection type v:l, $exist(db, q^i)$ returns *true* if q^i exists in db, return false otherwise, $complement(q^i q^j, temp)$ returns *false* if there is no complement of $q^i q^j$ in $temp$.

6 Experimental Results

A comprehensive experiment study has been conducted on real datasets which are the Financial data[1] (PKDD Cup 99) and the Thrombosis data[1] (PKDD Cup 2001). The Financial database scheme corresponds to the one given in Figure 1. There are 4,500 accounts, 5,369 clients, 5,369 objects in disposition, 6,471 objects in order, 2,500 objects in transaction, 682 objects in loan, 892 cards, and 77 districts. The experiments are carried out on a 2.8GHz Intel Core i7 cpu, 4GB memory.

6.1 Patterns. Interesting multi-rgps were discovered in the Financial database and Table 1 illustrates top strongest and longest patterns. We discuss some of them in the following.

Bases on σ_τ. The τRGP Miner returns the pattern $p = \{district.Unemploy96^{<}, order.Amount^{>}\}$ with $\sigma_\tau(p) = 58.32\%$. The pattern could be explained as: *"the less unemployed ratio districts are, the more expensive orders the habitants, who live in the districts, do"*. This pattern is very useful to recommend for business men that the bigger deals are from the less unemployed ratio districts. In addition, by applying the Theorem 4.1, the $E[\sigma_\tau](p)$ only is 14.3%. Thus, the pattern has an attractive support value.

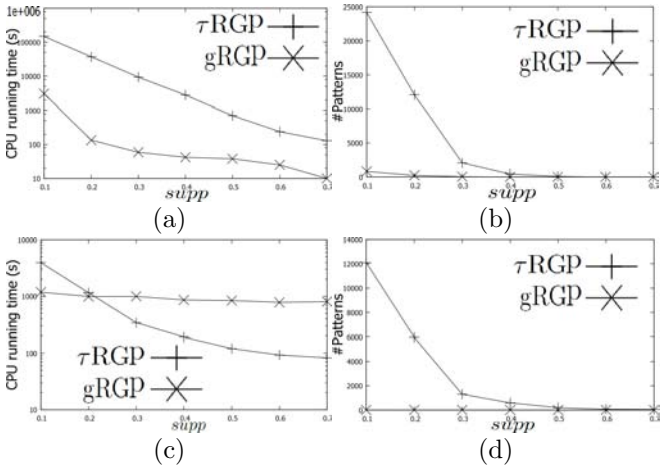

Figure 6: (a)(b)-Running time and #patterns on the Financial MRD. (c)(d)-Running time and #patterns on the Thrombosis MRD.

Another pattern related to the average salary is $p = \{district.Avg\text{-}Salary^{<}, orders.Amount^{<}\}$ with $\sigma_\tau(p) = 52.98\%$. The pattern can be explained as: *"the lower average salary the districts are, the less expensive orders the habitants, who live in the districts, do"*.

Bases on σ_g. The gRGP Miner also returns many interesting multi-rgps. Interestingly, the top strongest patterns based on σ_τ and σ_g are very different. For instance, the pattern $p = \{district.\#It\text{-}500^{>}, loan.Payment^{<}\}$ with $\sigma_g(p) = 44.444\%$. The pattern means: *"the more number of municipalities with inhabitants < 499 the districts have, the less the monthly payment is per inhabitant"*. This pattern is very interesting since it reveals unknown/unpublished loan policies.

Another interesting pattern is $p = \{district.\#It\text{-}500^{>}, loan.Duration^{>}\}$ with $\sigma_g(p) = 60\%$. The pattern means: *"the more number of municipalities with inhabitants < 499 the districts have, the longer debts the habitants have"*. This pattern shows that the habitants who are from the districts which have more number of municipalities with inhabitants < 499 are interested in borrowing money in longer time. Therefore, it could be useful to manage business strategies.

6.2 Efficiency and Pattern Distribution. To the best of our knowledge, there is no previous work which addresses multi-rgp issue. Therefore, in the efficiency experiments, we examine the proposed algorithms by varying the support threshold.

[1] http://lisp.vse.cz/challenge/.

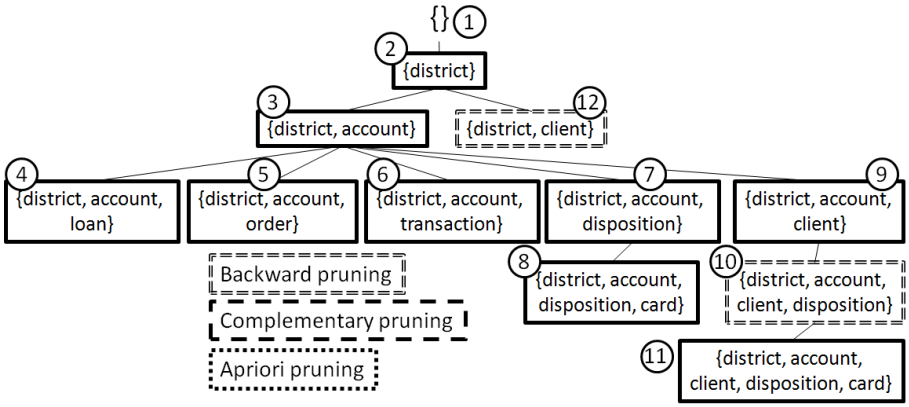

Figure 5: Multi-RGP search space of the graph in Figure 4.

Table 1: Frequent Multi-RGPs.

σ_τ	(%)	σ_g	(%)
$account.Date^{<}, loan.Date^{<}$	63.97	$district.\#It\text{-}500^{>}, loan.Duration^{>}$	60
$district.Unemploy96^{<}, orders.Amount^{>}$	58.32	$district.\#It\text{-}500^{>}, loan.Date^{<}$	55.556
$district.Avg\text{-}Salary^{<}, loan.Amount^{>}$	55.76	$district.\#People^{<}, loan.Duration^{>}$	50
$district.\#Crime95^{<}, loan.Amount^{>}$	54.42	$district.Unemploy95^{>}, loan.Payment^{>}$	50
$district.\#People^{>}, loan.Payment^{<}$	54.23	$district.\#Crime95^{<}, loan.Duration^{>}$	50
$district.\#Crime96^{<}, loan.Amount^{>}$	53.97	$district.\#Crime96^{<}, loan.Duration^{>}$	50
$district.Unemploy95^{<}, loan.Payment^{<}$	53.42	$district.Unemploy96^{<}, loan.Date^{>}$	45.455
$district.Avg\text{-}Salary^{<}, orders.Amount^{<}$	52.98	$district.\#It\text{-}500^{>}, loan.Payment^{<}$	44.444
$district.\#Entrepreneur^{<}, loan.Payment^{>}$	52.31	$district.Ratio\text{-}urban^{<}, loan.Payment^{>}$	44.444

Some of long patterns	σ_τ(%)
$district.\{\#It\text{-}2000^{<}, \#Crime95^{<}\}, account.Date^{>}, loan.Date^{<}$	36.91
$district.\{\#It\text{-}2000^{<}, \#Crime95^{<}, \#Crime96^{<}\}, account.Date^{>}, loan.Date^{<}$	36.53

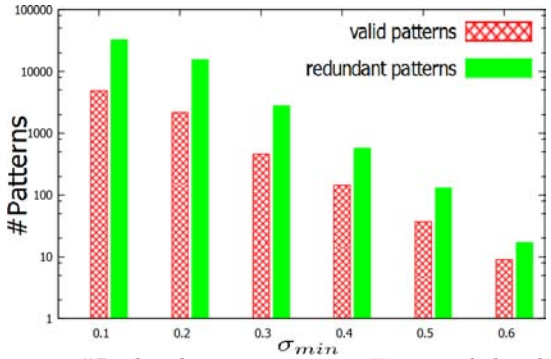

Figure 7: #Redundant patterns in Financial database.

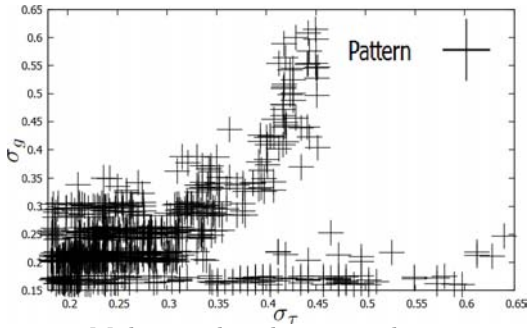

Figure 8: Multi-rgp distribution with $\sigma_{min} \geq 0.15$.

For Financial database, Figure 6a shows that τRGP Miner seems linear in terms of executing time. Furthermore, with the same support value, gRGP Miner is more efficient than τRGP Miner. This is because, by applying σ_g, there are less number of extracted patterns than σ_τ (Figure 6b). In essence, σ_g is tighter than σ_τ since the gradual tuple pairs are required to satisfy the consecutiveness constraint. However, there are differences in Thrombosis database. In most of cases, gRGP Miner is slower than τRGP Miner (Figure 6c). Since the matrix size (i.e., $32,935 \times 32,935$) which we need to navigate is significantly larger compare with the ones in Financial database (i.e., $6,471 \times 6,471$). The matrix navigation for computing the longest route is expensive. However, with low support values (i.e., σ=0.1), gRGP Miner is faster than τRGP Miner. This is because there are many patterns extracted by applying σ_τ (Figure 6d).

Let us consider the effectiveness of Backward Pruning rule (Figure 7). By applying the Backward Pruning, a large number of redundant patterns can be ignored in Financial database. In fact, the number of redundant patterns is much larger than the number of extracted ones. As a result, the process speeds up.

The distribution of multi-rgps extracted in Financial database is illustrated in Figure 8. We can consider that σ_τ and σ_g are converged at low value area ($\sigma_\tau, \sigma_g \leq 0.3$). However, they are diverse at the high value areas. It means the two support measures illustrate the graduality in different angles. They offer us a

powerful tool to discover more number of useful patterns which could benefit different real world applications. As a result, both σ_g and σ_τ are useful in mining multi-rgps.

7 Related Work

Most previous work on relational pattern mining can be categorized into methods that generalize ideas from frequent itemset mining to the relational setting and methods that are based on Inductive Logic Programming (ILP). In this section we discuss the differences between these approaches and our approach as well as other works that do not fall into these two categories.

Well known ideas and algorithms from frequent itemset mining can be used for multi-relational data if applied on the join of all tables. This kind of patterns essentially is itemsets such that items are attribute values and transactions are the tuples of the join table [9, 12, 13]. A different approach is taken by Smurfig [15] where the support is measured with respect to every table, as the relative number of keys that the items correspond to. Warmr [4] and Farmer [14] are ILP-based methods. The patterns are logic rules which can be regarded as local models of the database. The support is defined as the relative number of key values of one target table that satisfy the rule. The purpose of these methods is to extract the most frequent rules.

R-KRIMP [12] patterns are similar to the work of [2], who define these patterns as simple conjunctive queries of the form: $[p_0, \ldots, p_n]$. RDB-Krimp [11] is an improvement of R-KRIMP. RDB-Krimp and RMiner [16] are methods for mining relational databases by using information theoretic ideas for the assessment of patterns. RDB-Krimp focus on the total description length of the database joint with the patterns, and patterns are deemed more interesting if they are better at compressing this description length. RMiner insteads deem patterns more interesting if they describe surprising aspects of the database in a concise way.

Compare with the previous work, we are interested in mining patterns at a higher level. This type of patterns is the correlation between attributes from a graduality point of view.

8 Conclusion and Future Directions

In this paper, we propose the relational gradual pattern concept which enables us to examine the correlations between attributes from a graduality point of view in MRDs. To efficiently mine frequent patterns, we also define its associated support measures based on $Kendall's\ \tau$ and *gradual* supports in MRD context. Then, τRGP and gRGP Miner algorithms are proposed to extract all the frequent patterns. One of the future directions is to investigate top-k representative multi-rgp mining approach to avoid extracting huge amount of patterns. Another work is to define relational gradual pattern at tuple level. That helps us better understand the structure of MRDs from a graduality point of view.

References

[1] A. Appice, M. Ceci, and D. Malerba. Mining model trees: A multi-relational approach. In *ILP'03*, pages 4–21.

[2] H. Mannila B. Goethals, and W. L. Page. Mining association rules of simple conjunctive queries. In *SDM'08*, pages 96–107.

[3] T. Calders, B. Goethals, and S. Jaroszewicz. Mining rank-correlated sets of numerical attributes. In *KDD'06*, pages 96–105.

[4] L. Dehaspe and H. Toivonen. Discovery of frequent datalog patterns. *DMKD.*, 3(1):7–36, 1999.

[5] L. Di-Jorio, A. Laurent, and M. Teisseire. Mining frequent gradual itemsets from large databases. In *IDA'09*, pages 297–308.

[6] L. Džeroski and N. Lavrač. *Relational data mining.* Springer-Verlag, Berlin, 2001.

[7] P.A. Flach and N. Lachiche. Naive bayesian classification of structured data. *ML*, 57(3):233–269, 2004.

[8] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *IJCAI'99*, pages 1300–1309.

[9] B. Goethals, W. L. Page, and M. Mampaey. Mining interesting sets and rules in relational databases. In *SAC'10*, pages 997–1001.

[10] E. Hüllermeier. Association rules for expressing gradual dependencies. *DMKD*, volume 2431, pages 200–211, 2002.

[11] A. Koopman and A. Siebes. Characteristic relational patterns. In *KDD'09*, pages 437–446.

[12] A. Koopman and A. Siebes. Discovering relational item sets efficiently. In *SDM'08*, pages 108–119.

[13] E.K.K. Ng, A.W.-C. Fu, and K. Wang. Mining association rules from stars. In *ICDM'02*, pages 322–329.

[14] S. Nijssen and J. Kok. Efficient frequent query discovery in farmer. In *PKDD'03*, pages 350–362.

[15] W.L. Page. Mining patterns in relational databases. In *PhD Thesis, Universiteit Antwerpen, 2009.*

[16] E. Spyropoulou and T.D. Bie. Interesting multi-relational patterns. In *ICDM'11*, pages 675–684.

[17] F. Zelezný and N. Lavrac. Propositionalization-based relational subgroup discovery with rsd. *ML*, 62(1-2):33–63, 2006.

[18] N. Phan, D. Ienco, P. Poncelet, and M. Teisseire. Mining time relaxed gradual moving object clusters. In *ACM SIGSPATIAL'12*, pages 478–481.

[19] N. Phan, D. Ienco, P. Poncelet, and M. Teisseire. Mining Representative Movement Patterns through Compression. In *PAKDD'13*, pages 341–326.

[20] J. Nin, A. Laurent, and P. Poncelet. Speed Up Gradual Rule Mining from Stream Data! A B-Tree and OWA-based Approach. JIIS, 35.3: 447-463, 2010.