Using relational operations to express association rules

© Michal Burda, Marian Mindek, Jana Šarmanová

Dept. of Computer Science, FEI, VŠB – Technical university of Ostrava,
17. listopadu 15, 708 33 Ostrava–Poruba, Czech Republic
{Michal.Burda, Marian.Mindek, Jana.Sarmanova}@vsb.cz

Abstract

Association rules are essential data mining tool. There exist many types of them. Indeed, association rules of different types have often completely dissimilar syntax. In this paper, we try to design a framework based on relational operations enabling one to express various association rules using the same syntax. Doing so we will be able to study association rules at high level – it will be possible to formally describe common characteristics of different rule types.

1 Preface

This paper focuses on the possibility of representing the association rules using the relational operations. We do a lot of motivating in the beginning to make the reader clear why it is important to introduce precise formal language suitable for writing down various association rules in the same way.

After that, we get through many precise definitions which result in a formal logic called *Probabilistic Logic of Typed Relations* (PLTR).

At last, we reveal the reader some challenging task our research faces up today.

This paper expects the reader to be familiar with basics of association rule mining, mathematical logic, and with most important notations of statistical science.

2 Quick tour over association rule types

Association rule is an explicit representation of knowledge about some possibly interesting relationship that holds good in data. There exist many approaches to get such rules from data. They use sophisticated statistical methods or empirical procedures to measure the relevance of the rule. The best-known type of such relationship are rules of the type:

$$a_1 \wedge a_2 \wedge a_3 \Rightarrow b. \tag{1}$$

It express the fact that when an object has got the attributes a_1, a_2 and a_3 it is very probably that it has got b, too. For example, the Market-basket analysis (see [1], [9]) works mainly on data describing shopping activities of the customers and can produce the subsequent rule:

$$\mathsf{tequila} \wedge \mathsf{salt} \Rightarrow \mathsf{lemon} \tag{2}$$

Proceedings of the Spring Young Researcher's Colloquium on Database and Information Systems SYRCoDIS, St.-Petersburg, Russia, 2005

indicating that customers buying tequila and salt often buy lemons too. Such rules may be generalized to arbitrary categorial data. For example, let's have got a database of patients suffering certain disease. We can mine the following rule:

$$\mathsf{weight} > 100\mathrm{kg} \wedge \mathsf{smoker} \wedge \neg\mathsf{sport} \Rightarrow \mathsf{heart\text{-}failure}. \tag{3}$$

The rule says, there is strong evidence of getting heart failure for non-sporting smokers heavy at least 100 kg. Please note, the rule is rather rational and it tells probably nothing to the experts. However, many surprising rules may be mined, in practice.

We are not restricted on implicational rules only. Book [7] introduces for example the rules of the form:

$$\mathsf{mathematics} = \text{“excellent”} \sim \mathsf{physics} = \text{“excellent”} \tag{4}$$

saying that in database about pupils the excellentness in mathematics appears together with excelentness in physics. In the other words, the excellenteness in mathematics *is associated with* the excellenteness in physics.

Let's get more complicated. Work [7] presents further the subsequent rules:

$$X \text{ corr } Y \;/\; C. \tag{5}$$

The interpretation is: "The values of attributes A and B are correlated when looking only on objects fulfilling condition C." Rules of that type are well applicable on quantitative data.

Authors of paper [2] have developed the consecutive rules:

$$\mathsf{sex} = \text{“female”} \Rightarrow \mathsf{wage}\text{: mean = \$7.90/hr} \quad (\text{overall mean } \mathsf{wage} = \$9.02) \tag{6}$$

indicating that the women's wage mean is significantly different to the rest of examined objects.

We can be even more complicated and ask for the significance of the mean difference in data D between two subsets $A_1, A_2 \subset D$ whose union isn't the set D itself ($A_1 \cap A_2 = \emptyset$, $A_1 \cup A_2 \subset D$) as in [12], [13]. E.g. we may to compare the mean wage of people with age lower than 30 and people older than 50 etc.

Even the rule saying that there is *some* difference between two sets of objects in an array of attributes without telling us the kind of the dissimilarity (motivated by multi-dimensional statistical tests, see [11]) is better than knowing nothing.

2.1 Employing statistics in association rule mining

One of the ways to increase the power of data mining software is to employ the results of the statistic science in association rules mining process. The rule types mentioned above are only a few of many possibilities when considering the large amount of statistical tests that may be conceivable for rules mining.

Statistical tests of hypotheses are very powerful tool. In statistics, wide range of tests was designed. The tests are used to make decisions about miscellaneous characteristics; so, each test could be used to mine a rule of different type.

Moreover, statistical tests of hypotheses provide a basic measure of rule interest. Every statistical test is based on the computation of probability, at which the tested characteristic is valid. Thus, the basic measure of rule interest could be that probability. The more the tested characteristic is probable, the stronger tested relationship is, and therefore the more interesting the resulting rule could be.

A perfunctory understanding of statistical tests of hypotheses is assumed. We denote a zero hypothesis H_0 and alternative hypothesis H_A. This subsection tries to summarize basic characteristics of statistical tests and make clear, what can be tested at all.

Partitioning of statistical tests according to distribution dependence:

1. *Parametric* – these tests rely on a special type of distribution of sample data. These tests should be used with care in association rule mining process since we can not say anything specific to the analyzed data, in general.

2. *Non-parametric* – often called *rank tests*, too. These tests does not depend on a specific shape of sample's distribution. Weak apriory assumptions make them more useful in association rule mining process. However, weakness of assumptions is paid with lesser strength of tests and the conversion of quantitative data to ranks is sometimes relatively complex because of the need to sort the data, etc.

Partitioning of statistical tests according to the number of compared samples:

1. *One-sample tests* – are used to test, whether a specific sample characteristic ψ is equal to a given value ψ_0. For example, we can test, whether the weights of flour bags filled by automatic machine are in average equal to $\psi_0 = 1$ kg ($H_0 : \psi = \psi_0$) or if there is present a systematic error which causes under-weights or over-weights ($H_A : \psi \neq \psi_0$). Such tests may be utilized in pair-comparing rules; for example, whether the salary of people increased after a training course.

   Furthermore, one kind of the well-known association rule (3) is defined as one-sample test on probability of binomial distribution; it tests that the simultaneous occurrence of condition $A \wedge S$ in the rule is very probable in contrast to $A \wedge \neg S$ (see [8]).

2. *Two-sample tests* – are more general. They are able to compare a characteristic of a sample against characteristic of second sample. For example, we can test, whether the mean weight of some race A of men is significantly different from race B ($H_0 : \psi_A = \psi_B$, $H_A : \psi_A \neq \psi_B$). Such tests are just ideal to be utilized in association rule mining.

3. *Multi-sample tests* – are used to compare a specific characteristics of finite number of samples A, B, ..., Z. ($H_0 : \psi_A = \psi_B = \ldots = \psi_Z$ is tested against $H_A : \exists i, j \in \{A, B, \ldots, Z\} : \psi_i \neq \psi_j$.) These tests could be used in rule mining too.

Partitioning of statistical tests according to the number of compared quantities:

1. *One-dimensional* – these tests work only with one quantity (e.g. tests used to compare weight, wage or density of something).

2. *Multi-dimensional* – does not compare the only quantity, but in general the vector of quantities. We can compare, for example, weight, height, blood pressure etc. of two samples with one test trying to state whether there is certain difference in at least one attribute.

Both one-dimensional and multi-dimensional tests may be used to discover rules.

Partitioning of statistical tests according to the type of tested characteristics:

There exist tests dealing with *mean*, *variance (dispersion)*, *correlation*, and so on. The type of tested characteristic denotes in fact the type of mined rule...

2.2 Where is the problem?

What do we want to tell by this rule types enumeration? We can claim, the actually well-known rule types provide the possibility to mine various knowledge and thus are very different and the rule notation is very dissimilar too.

However, to be able to study such different rule types deeply, to be able to explore the similarities and relationships between various rule types and to infer formal conclusions about the rules, we should have got a tool for uniform rule notation. That is, we need a formal language capable to express association rules of any type – and just this topic is addressed in this paper.

We do so in the hope of uncovering some hidden truths about association rules which gives us the ideas of further improvements in the array of association rule mining algorithms, visualization and so on. We believe that without good understanding of causalities and relationships existing between various complex association rule types we won't be able to do this. (In fact, the usage of the framework presented here brings us immediately the benefit of finding the idea of *cosymmetric association rules*, which are discussed shortly in conclusion.)

Some work at the main topic of this paper was also done in [7]. However, we present diametrally different approach here and we think it is easier and more versatile.

3 Representing association rules using projection and selection

We can see from the previous chapter that association rules express some strong relationship in data. Our idea of how to design a formal logic language suitable for association rule notation is based on considering a rule as a relationship between two or more sub-tables of data table R.

For example, rule (1) can be taken as a relationship between a sub-table A that consists of all R's rows satisfying criterion $a_1 \wedge a_2 \wedge a_3$ and sub-table B satisfying criterion b (see also figure 1(a)).

When looking on rule (5) we see that it expresses relationship between two sub-tables A and B. Both A and B have got just rows that satisfy condition C, but table A consists of R's attribute X only and B of attribute Y only (vide also figure 1(b)).

Rule of type (6) models a relationship between two disjunctive sub-tables having one identical column. Such relationships are also depicted in figure 1(c).

We can continue the same way and transform every rule mentioned in section 2 to the proposition based on relationship between R's sub-tables.

Okay, you might say, we can treat association rules as relationships between sub-tables of analysed data table, but how to describe such sub-tables to obtain formal and easy-to-understand formulae? The answer inheres in commonly known relational-database operations: *selection* and *projection*. While selection is a tool for choosing a certain rows satisfying given condition, the projection takes out only specified columns of data table. Combining these two operations together, we can describe almost every sub-table of analysed data table R (see also figure 2).

To provide some concrete motivating examples, we must be a little more formal. Thus, let R be the data table. The expression of the form

$$R(C) \tag{7}$$

means selection of R's rows that satisfy given condition C, and the expression of the form

$$R[A_1, A_2, \ldots, A_n] \tag{8}$$

states projection on columns (attributes) A_1, A_2, …, A_n.

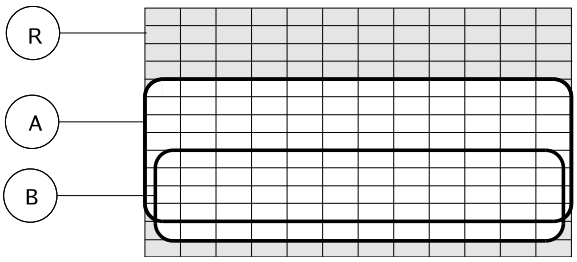
It is important for our topic to treat projection as the operation which holds duplicities. This is a small difference in contrast to commonly known projection on relations, since usual relations are sets and sets can not contain the same item twice. Hence, such relational projection may cause some information lost – duplicities are removed. Please see table 1 for illustrative projection and selection results. Precise definition of notions such as *data table*, *attribute*, *projection* or *selection* will be provided later in section 4.

Relationships between data can be modelled as usual mappings assigning a truth value to a tuple of sub-tables given as arguments. That way we are able to write every rule of section 2 in the new style. To show it, we will start with rule (5). We can write it in our new notation as follows:

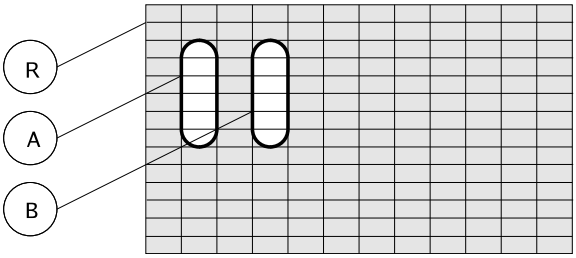
$$\mathrm{corr}\big(R(C)[X],\ R(C)[Y]\big) \tag{9}$$

Figure 1: Some types of relationships between two sub-tables A and B of table R

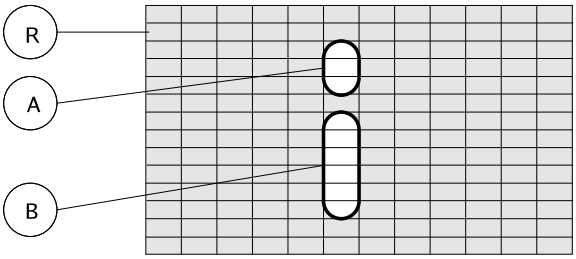
(a) Row-based relationship



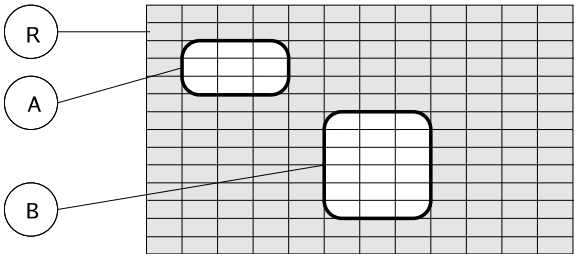
(b) Relationship among two columns of the same rows



(c) Relationship among two sub-tables of one column



(d) General relationship among two sub-tables



or infixually

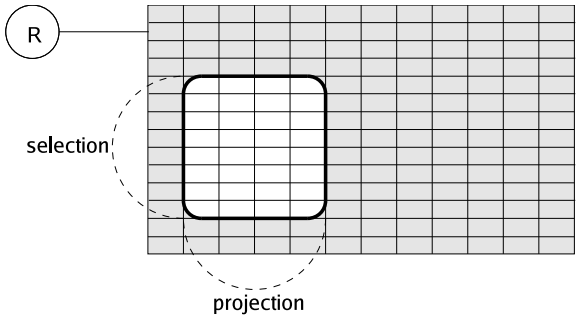
$$R(C)[A] \text{ corr } R(C)[B]. \tag{10}$$

In formula (9) (resp. (10)), we have prepared two sub-tables using selections and projections ($R(C)[A]$ and $R(C)[B]$). These sub-tables act as arguments to a mapping named corr which results to "true" when the two columns given as its arguments are strongly correlated.

Rule (6) can be represented by subsequent formula:

$$R(\mathsf{sex} = \text{“female”})[\mathsf{wage}] <_{\text{Z–test}} R(\mathsf{sex} \neq \text{“female”})[\mathsf{wage}]. \tag{11}$$

Figure 2: Selection and projection defines a sub-table of table R



It says that the statistical Z–test (this is the test which Aumann and Lindell used in [2] to mine rule (6)) showed significant difference in wage between the set of women and the rest of data table.

The advantage of our approach also lies in the fact that the mapping $<_{\text{Z–test}}$ and others may be defined as to result not in only true or false, but generally in arbitrary values – for example in the p–value of the statistical test on the background. That way we can construct a multi–valued logic based on probability.

Our research shows that representation of rule (4) is not such straightforward as in case of preceding rules. The benefit of a small complications is rise of an idea of new association rules type. A rule of type (4) we can write in new notation this way:

$$\sim \big(R,\ R(C_1),\ R(C_2)\big) \tag{12}$$

where C_1 is condition $\mathsf{mathematics} =$ "excellent" and C_2 is $\mathsf{physics} =$ "excellent". The reason of the need the mapping $\sim$ to be ternary is as follows. To determine (4)'s truth value we must know the number of records satisfying also conditions $C_1 \wedge \neg C_2$, $\neg C_1 \wedge C_2$ and $\neg C_1 \wedge \neg C_2$; it equals to the number of rows in $R(C_1) - R(C_2)$, $R(C_2) - R(C_1)$ and $R - R(C_1) - R(C_2)$, respectively. (For more details of how the (4)'s confidence is computed see [7], [8], [15].) The rule (12) can be interpreted as follows: "In the context of data table R the conditions C_1 and C_2 are mostly satisfied both or neither." The new type of association rules arises when we put selection at condition C_0 to R in the first argument – that is, when we modify the *context*:

$$\sim \big(R(C_0),\ R(C_1),\ R(C_2)\big). \tag{13}$$

Such rule means: "In the context of R's records satisfying condition C_0..." The equivalent to this rule type is mining rules (4) on sub-table R' that is created from R by omitting all rows that do not satisfy condition C_0.

The representation of rules (1) or (3) is straightforward. We can apply the same principle as with (4):

$$\Rightarrow \big(R,\ R(a_1 \wedge a_2 \wedge a_3),\ R(b)\big), \tag{14}$$

Please note the possibility of representing more general rules than described former. We are not limited to use only one column sub-tables as in figure 1(c). When we employ some multi-dimensional statistical test (see subsection 2.1) we can compare many columns at a time.

Table 1: Concrete example: selection and projection on data table R.

(a) Data table R

	a_1	a_2	a_3
k_1	1	0,25	Tom
k_2	5	0,65	Jack
k_3	7	0,72	Bill
k_4	8	0,85	John
k_5	8	0,31	Tom
k_6	9	0,25	Tom

(b) $R(a_1 > 6)$

	a_1	a_2	a_3
k_3	7	0,72	Bill
k_4	8	0,85	John
k_5	8	0,31	Tom
k_6	9	0,25	Tom

(c) $R[a_3]$

	a_3
k_1	Tom
k_2	Jack
k_3	Bill
k_4	John
k_5	Tom
k_6	Tom

Moreover, sometimes even the rules relating two sub-tables of type depicted in figure 1(d) are profitable. Suppose sub-table A contains quality characteristics of some metal a measured *before* some process of quality improvement ($A = R(\mathsf{type} = a)[Q_1, Q_2, Q_3]$) and sub-table B represents the same quality characteristics of metal b measured *after* the process of quality improvement ($B = R(\mathsf{type} = b)[Q'_1, Q'_2, Q'_3]$). Then a rule $A > B$ denotes the metal of type a is better even before the improvement process than metal of type b after improvement is performed. (However, this is rather tricky example, we know it.)

We hope the preceding examples give the reader much motivation to accept our proposition of treating the association rule mining process as follows. "Mining association rules of certain type from data table R is a process of searching rather simple mixtures of selections and projections on data table R for which the mapping representing the certain rule type gives truth value of our demand."

4 Basic detailed definitions

We see that the language for writing association rules using relational operations is very expressible. We won't remain only in coarse motivation examples. In this section, we provide a detailed definitions of the *Probabilistic logic of typed relations* (PLTR) which we propose to represent association rules in.

4.1 Typed relations

We start with basic definitions of the mathematical representations of data tables, attributes etc. Definitions of this sub-section are mainly based on the work [14]. However, we had to make some important modifications to adapt the notations to our needs.

In the subsequent, we sometimes use the phrase "set of abstract elements". It is simply a set whose elements are not concrete, e.g. $X = \{x_1, x_2, x_3\}$ is abstract set of three items.

Definition 1 *Let Ω be an infinite set of abstract elements which we will call as* attributes*. Let each $a \in \Omega$ has assigned a non-empty set $\delta(a) = D_a$ called* domain of attribute a*.* Type A (of relations) *is any finite subset of the set Ω. We denote $\mathcal{T}_\Omega$ a set of all types.*

A type A of relation is something like description of the data table. It says, what attributes (columns) are present in the data table and what data can be stored in that attributes (attribute domain).

Example 2 Let $a_1, a_2, a_3 \in \Omega$, $\delta(a_1) = \mathbb{N}$, $\delta(a_2) = \mathbb{Q}$ and $\delta(a_3)$ is equal to a set of all words made from English letters of length maximum 30, then a set $A = \{a_1, a_2, a_3\}$ is type. ∎

It is obvious that if A and B are types then $A \cup B$, $A \cap B$ and $A - B$ are types.

Definition 3 *Let Υ is infinitive set of abstract elements which we will call as* objects*. Let $A \in \mathcal{T}_\Omega$ is type. Let's denote $D_A = \{D_a = \delta(a) : a \in A\}$ (specially $D_\emptyset = \emptyset$).* A tuple of type A *is pair $\langle k, l\rangle$, where $k \in \Upsilon$ and l is such mapping $l : A \to D_A$ that $(\forall a \in A)(l(a) \in D_a)$.* A set of all tuples of type A *will be denoted as 1^A. A set of all tuples of type $\{a\}$ ($a \in \Omega$) will be denoted as 1^a.*

A tuple of type A is intuitively a representation of one row of data table. We can observe that $(\forall a \in \Omega)(1^a \neq \emptyset)$ and $(\forall A \in \mathcal{T}_\Omega)(1^A \neq \emptyset)$ because definitions 1 and 3 say that $D_a \neq \emptyset$, $D_A \neq \emptyset$.

Example 4 Consider type A from example 2. Then $u = \Big\langle k,\ \{\langle a_1,\ 1\rangle, \langle a_2,\ 0,25\rangle, \langle a_3,\ \mathsf{James}\rangle\}\Big\rangle$ where $k \in \Upsilon$ is a tuple of type A. ∎

Definition 5 (typed relation) *Let $A \in \mathcal{T}_\Omega$ be type.* A relation R of type A *is any finite subset of the set 1^A. Symbol $\mathcal{R}_A$ denotes a set of all relations of type A. Similarly, a set of all relations of type $\{a\}$ where $a \in \Omega$ is denoted with symbol $\mathcal{R}_a$.*

As one can see, the notion of typed relations is simply a representation of data table. Now, we introduce definitions of the important relational operations called selection and projection.

Definition 6 (selection) *Let $A \in \mathcal{T}_\Omega$ and let $R \in \mathcal{R}_A$.* A selection from relation R of type A according to condition C *is relation*

$$R(C) = \{u : u \in R \wedge C(u)\} \tag{15}$$

of type A, too. The notation $C(u)$ denotes a selection condition *and it constitutes the fact that condition C holds on tuple u.*

Definition 7 (projection) *Let $A \in \mathcal{T}_\Omega$, $B \subseteq A$ and let $R \in \mathcal{R}_A$.* A projection of relation R to the type B *is relation*

$$\begin{aligned} R[B] \quad = \quad & \Big\{u = \langle k_u, l_u\rangle \in 1^B : \\ & \big(\exists v = \langle k_u, l_v\rangle \in R\big) \\ & \big(\forall b \in B\big)\big(l_u(b) = l_v(b)\big)\Big\} \end{aligned} \tag{16}$$

of type B.

Example 8 As an example of relation R of type A (see example 2) we can take table 1(a) (k_i on the table are *objects*, that is, $k_i \in \Upsilon$). The selection $(a_1 > 6)$ on that relation is presented in table 1(b). Projection $[a_3]$ one can see in table 1(c). ∎

Please note that projection is defined as to hold duplicities. Why this is a matter see section 3.

4.2 Modelling relationships

With these basic definitions we can go forth to define general notion of a *predicate of relationship*. Predicate of relationship is simply a mapping that assigns truth value to relations given as arguments. Since we are building *probabilistic* logic, the truth value will be a probability. Concretely, truth value will be defined as a pair of values $\langle l, h\rangle$ where $0 \leq l \leq h \leq 1$ that form an interval of probability (l is its lower bound and h its upper bound).

Definition 9 Probability interval *is a pair $i = \langle l,\ h\rangle$, where $l, h \in \mathbb{R}$ and $0 \leq l \leq h \leq 1$.* A set V of truth values *is a set of any probability interval i. By notation $q \in' i = \langle l, h\rangle$ where $q \in \mathbb{R}$ we mean $l \leq q \leq h$.*

Definition 10 *Let $A_1, A_2, \ldots, A_n \in \mathcal{T}_\Omega$ and let V be a set of truth values then* n-ary relationship predicate *is a mapping $p^\star : D_{p^\star} \to V$ where $D_{p^\star} \subseteq \mathcal{R}_{A_1} \times \mathcal{R}_{A_2} \times \ldots \times \mathcal{R}_{A_n}$ is a set called* domain of relationship predicate $p^\star$.

So, relationship predicate is a mapping that assigns to certain relations a truth value. It is obvious, we can model various relationships that way. The definition presented above assumes the predicate to result in the interval of probability. However, we can modify the definitions to suit classical two-valued logic.

5 Intermezzo: Simultaneous statistical inference and data mining

Before going further we should do a little discussion on data-mining from the statistician's point of view.

Nowadays usual practice is to use statistical tests in association rules mining process. Statistical tests are based on hypothesis testing. One formulates a zero hypothesis H_0 and appropriate alternative hypothesis H_A such that the simultaneous turning up of H_0 and H_A is impossible; that is, $P(H_0 \cap H_A) = 0$ where P states

probability. Hypotheses are constructed as to H_A be the event whose validity we want to be convinced of. The appropriate statistical test then computes the probability α at which the zero hypothesis H_0 is valid. If this probability is very small ($\alpha < 0,05$, $\alpha < 0,01$ or even $\alpha < 0,001$) we are almost sure of H_0's invalidity. In the other words, we know at least with $1 - \alpha$ certainty that H_A is valid.

Please take into account the following important note: when α isn't small, we aren't sure of H_0's validity; we simply know nothing – vide e.g. [10].

Thus, stating whether a rule is important or not is done by the certain test of hypotheses. The importance of the rule is then derived from the level of significance at which the zero hypothesis can be rejected – that is, the truth value of the rule is simply $1 - \alpha$. One can see that the same approach is enabled in this work; we use interval of probability as the rule's truth value.

However, when performing many tests of hypotheses at the time and rejecting many H_0 hypotheses at very small level of significance α, the resultant probability of validity is greater than α. In fact, the probability of erroneous reject of at least one H_0 tends rapidly to 1.

This phenomenon is often called as *simultaneous statistical inference* (see [10] etc.). We can explain it using Bonferroni's theorem (vide [11]) which sounds as follows.

Consider random events $A_1, A_2, \ldots, A_n$. Each of them has got some certain probability $P(A_i)$. Bonferroni's theorem says that for the probability of the event $A_1 \cap A_2 \cap \ldots \cap A_n$ (the simultaneous occurrence of every event A_i) holds the following:

$$P(A_1 \cap \ldots \cap A_n) \leq \max\left(0,\ 1 - \sum_{i=1}^{n}\big(1 - P(A_i)\big)\right). \tag{17}$$

Take A_i to be the validity of any i-th H_A. So $P(A_i) = 1 - \alpha_i$, where α is level of significance at which we have rejected corresponding zero hypothesis H_0. Thus the probability that every H_0 was rejected rightfully is

$$P(A_1 \cap \ldots \cap A_n) \leq \max\left(0,\ 1 - \sum_{i=1}^{n}(\alpha_i)\right). \tag{18}$$

The sum tends fast to 1 so the right side gets to zero very quickly. For example, when performing 20 tests and rejecting zero hypotheses at $\alpha = 0,05$ in each test, the overall probability of at least one unauthorized rejection is greater or equal to $20 \cdot 0,05 = 1$.

What is the consquence? In contrast to common statistical analyses, data-mining results can not be treated as natural law or undisputed truth. Data-mining gives answers to the questions of type "what *maybe* holds in the area described by my data?" E.g. work [7] proposes to understand the association rules as enumeration of hypotheses that are supported with data or as hypotheses the data tend to.

6 Advanced definitions

Although we have vindicate the possibility to ignore simultaneous statistical inference (vide section 5), sometimes it is useful to prevent it even in data-mining results. The concrete task we are accually working on is the situation when some mined rules belong *somehow* together.

For example, let attribute A be categorical with domain $\delta(A) = \{1, 2, 3, 4\}$ and let attribute B be quantitative, that is $\delta(B) = \mathbb{R}$. Consider the following rules:

$$R(A = 1)[B] >_{\text{Z–test}} R(A = 2)[B], \tag{19}$$

$$R(A = 2)[B] <_{\text{Z–test}} R(A = 4)[B], \tag{20}$$

$$R(A = 3)[B] <_{\text{Z–test}} R(A = 4)[B]. \tag{21}$$

These rules logically belong together because they express strong differences in attribute B between objects that differ in attribute A. We are working on a method that treats such rule sets specially and since we manage a set of rules as one complicated rule, it is useful for us to know the overall p–value. Videlicet, we are interested what is the probability of the truthfulness of the rule

$$\begin{aligned}&\big(R(A = 1)[B] >_{\text{Z–test}} R(A = 2)[B]\big) \wedge \\ \wedge&\big(R(A = 2)[B] <_{\text{Z–test}} R(A = 4)[B]\big) \wedge \\ \wedge&\big(R(A = 3)[B] <_{\text{Z–test}} R(A = 4)[B]\big).\end{aligned} \tag{22}$$

6.1 Probabilistic connectives

In this sub-section we will define logical connectives able to work with probability intervals as presented in definition 9.

Definition 11 *Let V be a set of truth values.* Probabilistic connective *is a mapping $\iota^\star : V^n \to V$ where n is arity of probabilistic connective $\iota^\star$.*

Definition 12 *Let V be a set of truth values.* Probabilistic negation *is unary probabilistic connective $\neg^\star : V \to V$ such that $\neg^\star(\langle l,\ h\rangle) = \langle 1 - h,\ 1 - l\rangle$.*

We can see that the traditional feature of negations holds for our probabilistic negation too:

$$\neg^\star\big(\neg^\star(i)\big) = i. \tag{23}$$

Definitions of probabilistic conjunctions and disjunctions will be based on the commonly known notations of t-norm and t-conorm.

Definition 13 *Let V be a set of truth values.* Triangular norm (t-norm) *is any binary probabilistic connective $T : V \times V \to V$ where $\forall x, y, z \in V$ holds:*

1. $T(x, y) = T(y, x)$; *(commutativity)*
2. $T\big(x, T(y, z)\big) = T\big(T(x, y), z\big)$; *(asociativity)*
3. *if* $\min(x) \leq \min(y)$ *then*

$$\min\big(T(x, z)\big) \leq \min\big(T(y, z)\big)$$

and if $\max(x) \leq \max(y)$ *then*

$$\max\big(T(x, z)\big) \leq \max\big(T(y, z)\big);$$

(monotony)

4. $T\big(\langle 0, 0\rangle,\ x\big) = \langle 0, 0\rangle$ *and* $T\big(\langle 1, 1\rangle,\ x\big) = x$. *(restriction)*

Definition 14 *Let V be a set of truth values.* Triangular co-norm (t-conorm) *is such binary probabilistic connective $S : V \times V \to V$ where $\forall x, y \in V$ holds:*

$$S(x,y) = \neg^\star\Big(T\big(\neg^\star(x),\ \neg^\star(y)\big)\Big) \tag{24}$$

and T is t-norm.

Concrete t-norms are interpreted as probabilistic conjunctions and t-conorms as probabilistic disjunctions. We can easily check the validity of the subsequent equation:

$$T(x,y) = \neg^\star\Big(S\big(\neg^\star(x),\ \neg^\star(y)\big)\Big). \tag{25}$$

Specific t-norm and t-conorm is defined accordingly to Bonferroni's theorem about probability of random event conjunction (see [11]).

Definition 15 *Let $i_1, i_2 \in V$ and $i_1 = \langle l_1,\ h_1\rangle$, $i_2 = \langle l_2,\ h_2\rangle$. Binary probabilistic connective $\wedge^\star_{safe}$ defined as*

$$\wedge^\star_{safe}(i_1,i_2) = \Big\langle \max(0,\ l_1 + l_2 - 1),\ \min(h_1,\ h_2)\Big\rangle \tag{26}$$

is called as safe conjunction.

Binary probabilistic connective $\vee^\star_{safe}$ defined as

$$\vee^\star_{safe}(i_1,i_2) = \Big\langle \max(l_1,\ l_2),\ \min(1,\ h_1 + h_2)\Big\rangle \tag{27}$$

is called as safe disjunction.

Theorem 16 *Safe conjunction $\wedge^\star_{safe}$ is t-norm and safe disjunction $\vee^\star_{safe}$ is t-conorm of $\wedge^\star_{safe}$.*

Proof. Evident. □

With connectives defined in this sub-section we can even join various relationship predicates together.

7 Probabilistic Logic of Typed Relations

Now we have defined all notions necessary for creation of language of Probabilistic Logic of Typed Relations (PLTR).

Definition 17 (alphabet) *The PLTR language alphabet consists from the following symbols:*

- *Symbols for typed relations: $R, S, T, \ldots$*
- *Symbols for attributes: $a, b, c, \ldots$*
- *Logical connectives for creation of selection conditions: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$*
- *Relational operations: $-, \cup, \cap$*
- *Symbols of n-ary relationship predicates: $p^\star$, $q^\star$, $r^\star, \ldots, \leq^\star, \ldots$*
- *Probabilistic logical connectives: $\neg^\star$, $\wedge^\star$, $\vee^\star$*
- *Auxiliary symbols (parentheses, brackets): $(,)$, $[,]$*

Definition 18 (language) *Language of PLTR is a set $\mathcal{J}$ of formulae created from the alphabet of the PLTR language accordingly to the following conditions. Let $A_1, A_2, A_3, A, B \in \mathcal{T}_\Omega$ be types of relations and let $A \subseteq B$. We define:*

- *selection condition of type A:*
  1. *Any symbol $a \in A$ of attribute's name is a selection condition of type A.*
  2. *If C_1, C_2 are selection conditions of type A then also $\neg C_1$, $(C_1 \vee C_2)$, $(C_1 \wedge C_2)$, $(C_1 \Rightarrow C_2)$ and $(C_1 \Leftrightarrow C_2)$ are selection conditions of type A.*
  3. *There is no other selection condition of type A.*
- *term of type A:*
  1. *Symbols of relations of type A are terms of type A.*
  2. *If T_1, T_2 are terms of type A then also $(T_1 \cup T_2)$, $(T_1 \cap T_2)$ and $(T_1 - T_2)$ are terms of type A.*
  3. *If T is term of type A and C is selection condition of type A then also $T(C)$ is term of type A.*
  4. *If $A = \{a_1, a_2, \ldots, a_k\} \subseteq B$ and T is term of type B then $T[a_1, a_2, \ldots, a_k]$ is term of type A.*
  5. *There is no other term of type A.*
- *formula PLTR:*
  1. *If T_1, T_2, …, T_n are terms of type A_1, A_2, …, A_n and $p^\star$ is n-ary relationship predicate then $p^\star(T_1, T_2, \ldots, T_n)$ is formula.*
  2. *If F_1, F_2 are formulae PLTR then also $\neg^\star F_1$, $(F_1 \vee^\star F_2)$ and $(F_1 \wedge^\star F_2)$ are formulae PLTR.*
  3. *There is no other formula PLTR.*

Definition 19 (model) Model *of PLTR is a pair $M = \langle Q, g\rangle$ where Q is a set of typed relations $R_1, R_2, \ldots, R_n$ of types $A_1, A_2, \ldots, A_n$ and g is a mapping assigning an element from Q to every symbol of typed relations.* A set of all models *is denoted $\mathcal{M}$.*

The model definition assumes the relationship predicates to have fixed interpretations all the time. (That is, the relationship predicates interpretations are the same for all models.)

Definition 20 *Let V be a set of truth values (that is, V is a set of all probability intervals).* The evaluating function *Val is a function which maps formula F and model $M = \langle Q, g\rangle$ to the interval of probability. That is, $Val : \mathcal{J} \times \mathcal{M} \to V$. Domain of function Val depends on domains of relationship predicates used in formula F.*

The receipt to get the value of function Val is as follows. Let $M = \langle Q, g\rangle$ be a model of PLTR then:

1. *One must find out values of all terms recursively. (For that case let us introduce a function Val' for term evaluation):*

(a) Let R be the symbol of typed relation then

$$Val'(R) = g(R).$$

(b) Let R, S be the terms then

$$\begin{aligned} Val'(R \cup S) &= Val'(R) \cup Val'(S), \\ Val'(R \cap S) &= Val'(R) \cap Val'(S), \\ Val'(R - S) &= Val'(R) - Val'(S). \end{aligned}$$

(c) Let T be the term and $a_1, \ldots, a_n \in \Omega$ then

$$Val'(T[a_1, \ldots, a_n]) = Val'(T)[a_1, \ldots, a_n].$$

(d) Let T be the term and C be the selection condition then

$$Val'\big(T(C)\big) = \big(Val'(T)\big)(C).$$

2. *One must figure out the formula's truth value recursively:*

(a) Let $T_1, \ldots, T_n$ be the terms and let $p^\star$ be the n-ary relationship predicate (defined on $\langle Val'(T_1), \ldots, Val'(T_n)\rangle$) then

$$\begin{aligned} &Val\big(p^\star(T_1, \ldots, T_n)\big) = \\ &= p^\star\big(Val'(T_1), \ldots, Val'(T_n)\big). \end{aligned}$$

(b) Let F, G be formulae then

$$\begin{aligned} Val\big(\neg^\star F\big) &= \neg^\star\big(Val(F)\big), \\ Val(F \wedge^\star G) &= \wedge^\star\big(Val(F), Val(G)\big), \\ Val(F \vee^\star G) &= \vee^\star\big(Val(F), Val(G)\big). \end{aligned}$$

One can find many well formed formulae of PLTR language in section 3.

8 Semantic system of PLTR

This section provides definitions of *logical followings* which are necessary for writing down considerations about PLTR formulae. We start with definition taken from book [7]:

Definition 21 A semantic system *is determined by an non-empty set $Sent$ of formulae, a non-empty set $\mathcal{M}$ of models, a non-empty set V of truth values and an evaluating function $Val : (Sent \times \mathcal{M}) \to V$.*

We need one more definition from book [7]:

Definition 22 *Let $\mathcal{S} = \langle Sent, \mathcal{M}, V, Val\rangle$ be a semantic system and let $V_0 \subseteq V$ be a set of* designated values.

- *A formula $\varphi \in Sent$ is V_0-*true *in a model M (notation $M \models_{V_0} \varphi$) if*

$$Val(\varphi, M) \in V_0.$$

- *A formula $\varphi \in Sent$ is V_0-*tautology *(notation $\models_{V_0} \varphi$) if it is V_0-true for each model $M \in \mathcal{M}$ such that the formula φ is defined on it.*

- *Formula $\varphi \in Sent$ is a* logical V_0-consequence *of a set $\Psi \subseteq Sent$ of formulae (notation $\Psi \models_{V_0} \varphi$) if $\forall M \in \mathcal{M}$ the following holds:*

$$\big((\forall \psi \in \Psi)(M \models_{V_0} \psi)\big) \Rightarrow (M \models_{V_0} \varphi).$$

- *Formulae $\varphi \in Sent$ and $\psi \in Sent$ are* logically V_0-equivalent *(notation $\varphi \equiv_{V_0} \psi$) if*

$$(\varphi \models_{V_0} \psi) \wedge (\psi \models_{V_0} \varphi).$$

For new we introduce even stronger concepts:

Definition 23 *Let $\mathcal{S} = \langle Sent, \mathcal{M}, V, Val\rangle$ be a semantic system.*

- *Formula $\varphi \in Sent$ is* logical consequence *of a set $\Psi \subseteq Sent$ of formulae (notation $\Psi \models \varphi$) if*

$$(\forall V_0 \subseteq V)(\Psi \models_{V_0} \varphi).$$

- *Formulae $\varphi \in Sent$ and $\psi \in Sent$ are* logically equivalent *(notation $\varphi \equiv \psi$) if*

$$(\forall V_0 \subseteq V)(\varphi \equiv_{V_0} \psi).$$

Example 24 Here are examples of some simple logical equivalences:

$$\begin{aligned} p^\star\big(R(C_1) \cup R(C_2)\big) &\equiv p^\star\big(R(C_1 \vee C_2)\big), \\ p^\star\big(R(C_1) \cap R(C_2)\big) &\equiv p^\star\big(R(C_1 \wedge C_2)\big), \\ p^\star\big(R(C_1) - R(C_2)\big) &\equiv p^\star\big(R(C_1 \wedge \neg C_2)\big). \end{aligned}$$

■

9 Related work

Association rules have been well researched. First mention about them is dated in sixtieth years of the twentieth century. Czech authors Hájek, Havel and Chytil in 1966 presented the work [6] about automated hypothesis testing in method called GUHA. Their work appeard long time before concepts such as data mining or knowledge discovery becomes familiar. Although their work was presented many times in the whole world, many people interested in data mining often erroneously present Agrawal, Imielinski and Swami ([1]) as the first people who tried to mine association rules. (See e.g. [9] on page 276.)

Hájek and Havránek have written the book [7] where the concepts of the automatized hypotheses generation were very well described. They based the thinking of the rules as of formulae interconnected with *generalized quantifiers*.

Generalized quantifiers is natural generalization of classical quantifiers $\forall$ (universal) and $\exists$ (existential). For example, Rescher's (1962) plurality quantifier W says that "most objects satisfy the formula". The second example, the Church's (1951) quantifier of implication (not to be confused with the logicall connective of implication) $\Rightarrow o\,(\varphi_1(o), \varphi_2(o))$ says that the φ_2 formula is true for all objects for which the formula φ_1 is true.

Authors of the GUHA method have introduced many such generalized quantifiers that model various relationships.

This work proposes the other look at association rules. We don't treat association rules as formulae interconnected with quantifiers but rather as pieces of data described with relational operations interconnected with predicates. The difference is in the level of logical notions where the knowledge is presented. GUHA uses predicates simply to denote attributes and it models relationships with quantifiers. We hide the fashion of describing the objects figuring in a rule in functional symbols and the relationship modelling arises in predicates, already.

10 Conclusion and future work

In this paper, we have presented summary overview of the best-known association rule types. We have designed a new language usable for rule representations at that basis and we have shown many examples of how to use this language (see also our work [4]). The usage of the presented language has directly led to some new types of association rules (vide rule (13)).

Please note that our primary goal of the language design was to create logic suitable for our further research on similar properties of various association rules. We confess the fact that our language may not be ideal instrument for rule representations to the end-user (analyst).

Our future work is focused to the study of properties of so-called *cosymmetric rules*. Actually, we don't have precise definition of what a cosymmetric rule is. (Some notes may be found in [5].) Intuitively, we can say that cosymmetric rule is a rule of the form

$$D(C_1)[A_1, \ldots, A_n] >^\star D(C_2)[B_1, \ldots, B_m] \quad (28)$$

where two sub-tables are compared and the mapping $>^\star$ tells us whether the first sub-table is *somehow* greater than the second one. For example, (11) is a typical member of cosymmetric rule class. It comes to light that the class of cosymmetric rules is very large. Our research shows that even the rules (1), (2), (3) could be treated as cosymmetric.

We are also working on fast algorithms for mining the cosymmetric rules. See [3] for the implementation of the generic framework for mining frequent conjunctions.

In the centre of interest stands the problem of visualization of cosymmetric rules – the usage of conceptual lattices for such purposes is elaborated. We also study the benefits of displaying the cosymmetric rules as quasi–ordered set in slightly modified Hasse's diagram.

References

[1] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining associations between sets of items in massive databases. In *ACM SIGMOD 1993 Int. Conference on Management of Data*, pages 207–216, Washington D.C., 1993.

[2] Yonatan Aumann and Yehuda Lindell. A statistical theory for quantitative association rules. In *Knowledge Discovery and Data Mining*, pages 261–270, 1999.

[3] Michal Burda, Martin Hynar, and Jana Šarmanová. Object-oriented interface to algorithms searching frequent conjunctions. In *ISIM*, Hradec nad Moravicí, 2005.

[4] Michal Burda, Martin Hynar, and Jana Šarmanová. Probabilistic logic of typed relations (in czech). In *Znalosti, poster proceedings*, pages 13–16, Vysoké Tatry, 2005.

[5] Michal Burda, Marian Mindek, and Jana Šarmanová. Characteristics of cosymmetric association rules. In *Dateso*, 2005.

[6] Petr Hájek, Ivan Havel, and Metoděj Chytil. The GUHA method of automatic hypotheses determination. In *Computing 1*, pages 293–308, 1966.

[7] Petr Hájek and Tomáš Havránek. *Mechanizing Hypothesis Formation*. Springer–Verlag, Berlin, 1978. Internet: `http://www.cs.cas.cz/~hajek/guhabook/` (May 2004).

[8] Petr Hájek, Tomáš Havránek, and M. K. Chytil. *The GUHA method – automatized hypothesis creation (in czech)*. Academia, Praha, 1983.

[9] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, USA, 2000.

[10] Tomáš Havránek. *Statistics for biological and medical sciences (in czech)*. Academia, Praha, 1993.

[11] Petr Hebák and Jiří Hustopecký. *Multi-dimensional statistical methods and its applications (in czech)*. SNTL, Praha, 1987.

[12] Tomáš Karban. SDS–rules (in czech). In *Znalosti, poster proceedings*, pages 17–20, Brno, 2004.

[13] Tomáš Karban and Milan Šimůnek. SDS–rules and association rules. In *Track on Data Mining (DM), ACM Symposium on Applied Computing*, 2004.

[14] Wendy MacCaull and Ewa Orłowska. A calculus of typed relations. In *RelMiCS/Kleene–Algebra Ws 2003, LNCS 3051*, pages 191–201, 2004.

[15] Jan Rauch. Association rules and mathematical logic (in czech). In *Znalosti*, pages 114–125, Brno, 2004.