AN APPROACH TO DISCOVERING TEMPORAL ASSOCIATION RULES

Juan M. Ale
Facultad de Ciencias Exactas, UNLP, Argentina
and also at UNLM
Ambrosetti 255 –(1405)Buenos Aires, Argentina
(5411)4903-6735/(5411)4903-8751
E-mail: ale@acm.org

Gustavo H. Rossi
LIFIA, Facultad de Informática, UNLP, Argentina
and also at CONICET and UNLM
Calles 1 y 49 – La Plata, Argentina
(54221)422-8252/(54221)325-5393
E-mail: gustavo@sol.info.unlp.edu.ar

Keywords

Data Mining, Association Rules, Temporal Data Mining, Temporal Rules.

ABSTRACT

The goal of discovering association rules is to discover *all* possible associations that accomplish certain restrictions (minimum support and confidence and interesting). However, it is possible to find interesting associations with a high confidence level but with little support. This problem is caused by the way support is calculated, as the denominator represents the total number of transactions in a time period when the involved items may have not existed. If, on the other hand, we limit the total transactions to the ones belonging to the items' lifetime, those associations would be now discovered, as they would count on enough support. Another difficulty is the large number of rules that could be generated, for which many solutions have been proposed. Using age as an obsolescence factor for rules helps reduce the number of rules to be presented to the user. In this paper we expand the notion of association rules incorporating time to the frequent itemsets discovered. The concept of temporal support is introduced and, as an example, the known algorithm *A priori* is modified to incorporate the temporal notions.

1. INTRODUCTION

The problem of the discovery of association rules comes from the need to discover patterns in transaction data in a supermarket. But transaction data are temporal. For example, when gathering data about products purchased in a supermarket, the time of the purchase is registered in the transaction. This is called *transaction time,* in temporal databases jargon , which matches the *valid time*, corresponding to the time of the business transaction confirmation at the register. In [12], the author expresses: "*The time dimension is the one dimension virtually guaranteed to be present in every data warehouse, because virtually every data warehouse is a time series*".

In large data volumes, as used for data mining purposes, we may find information related to products that did not necessarily exist throughout the data gathering period. So we can find some products that, at the moment of performing that mining, have already been discontinued. There may be also new products that were introduced after the beginning of the gathering. Some of these new products should participate in the associations, but may not be included in any rule because of support restrictions. For example, if the total number of transactions is 30,000,000 and we fix as minimum support 0.5 %, then a particular product must appear in, at least, 150,000

transactions to be considered frequent. Moreover, suppose that these transactions were recorded during the last 30 months, at 1,000,000 per month. Now, take a product that has been sold during the 30 months and has just the minimum support: it appears on average in 5,000 transactions per month. Consider now another product that was incorporated in the last 6 months and that appears in 20,000 transactions per month. The total number of transactions in which it occurs is 120,000; for that reason, it is not frequent, even though it is four times as popular as the first. However, if we consider just the transactions generated since the product appeared in the market, its support might be above the stipulated minimum. In our example, the support for the new product would be 2%, **relative to its lifetime**, since in 6 months the total of transactions would be about 6,000,000 and this product appears in 120,000 of them. Therefore, these new products would appear in interesting and potentially useful association rules.

One way to solve this problem is by incorporating time in the model of discovery of association rules. We will call these rules *Temporal Association Rules.*

One subproduct of this idea is the possibility of eliminating outdated rules, according to the user criteria. Moreover, it is possible to delete obsolete itemsets as a function of their lifetime, reducing the amount of work to be done in the determination of the frequent itemsets and, hence, in the determination of the rules.

The temporal association rules introduced in this paper are an extension of the nontemporal model. The basic idea is to limit the search for frequent sets of items or *itemsets* to the lifetime of the itemset's members. For that reason, the concept of temporal support is introduced. Thus, each rule has an associated time frame, corresponding to the lifetime of the items participating in the rule . If the extent of a rule's lifetime exceeds a minimum stipulated by the user, we analyze if the rule is frequent in that period. This concept allows us to find rules that, with the traditional frequency viewpoint, it would not be possible to discover.

The remainder of this paper is organized as follows. Related work on discovery of association rules, temporal data mining in general, and discovery of temporal rules in temporal databases, in particular is given in Section 2. The temporal model is introduced in Section 3. In Section 4 we discuss the discovery of temporal rules, adapting the *A priori* method as an example. Also in section 4 we analyze changes to an existing algorithm for the rules' generation. Finally, in section 5 we conclude and briefly discuss future work.

2. RELATED WORK

The problem of discovering associations from data was introduced by Agrawal et al. in [1]. It was followed by successive refinements, generalizations and improvements ([5], [3], [9], [15], [17], [18]). Among these we can find improved algorithms for the discovery of frequent itemsets, generalized and quantitative association rules, and new measures for other types of data, different from the market basket.

Previous work about data mining that includes temporal aspects is usually related to the sequence of events' analysis ([2], [7], [8], [13]). The usual objective is to discover regularities in the occurrence of certain events and temporal relationships between the different events. In particular, in [13], the problem of recognizing frequent episodes in an event sequence is discussed; an episode is defined there as a collection of events that occur during time intervals of a specific size. Meanwhile [6] reviews the problem of discovering sequential patterns in transactional data bases. The solution consists in creating a sequence for every client and to look for frequent patterns into each sequence.

In [7] and [8] more complex patterns than in the cases mentioned above are considered. In these cases temporal distances with multiple granularities are treated.

Now we will analyze how the present work is related to others, specifically in mining temporal association rules. All of them have the same goals as ours: the discovery of association rules and their periods or interval time of validity. Our proposal was formulated independently of the others but shares with them some similarities. In [14] they study the problem of association rules that exist in certain time intervals and thus

display regular cyclic variations over time. They present algorithms for efficiently discover what they called "cyclic association rules". It is assumed that time intervals are specified for the user.

In [16] the authors study how the association rules vary over time, generalizing the work in [14]. They introduce the notion of calendar algebra to describe temporal phenomena of interest to the users and present algorithms for discovering "calendric association rules", that is, association rules that follow the temporal patterns set forth in the user supplied calendar expressions.

The third study([11]) also suggests calendar time expressions to represent temporal rules. They present only the basic ideas of the algorithms for discovering the temporal rules.

Our approach is based on taking into account the items' period of life or lifespan, this being the period between the first and the last time the item appears in transactions in the database. We compute the support of an itemset in the interval defined by its lifespan and define temporal support as the minimum interval width. Our approach differs from the others in that it is not necessary to define interval or calendars since the lifespan is intrinsic to the data. In addition, we describe in detail the temporal extension to the Apriori algorithm.

3. THE TEMPORAL MODEL

Let T = { . . ., t_0, t_1, t_2, . . . } be a set of times, countably infinite, over which a linear order $<_T$ is defined, where $t_i <_T t_j$ means t_i occurs before or is earlier than t_j([21]). We will assume that T is isomorphic to N (natural numbers) and restrict our attention to closed intervals $[t_i, t_j]$.

Definition 1: Let **R** = { A_1, . . ., A_p} where the A_i's are called *items*, **d** is a collection of subsets of **R** called the *transaction database*. Each transaction **s** in **d** is a set of items such that **s** $\subseteq$ **R**. The definition of **R** includes every item of **d**, independently of the moment in which it appears. Associated to **s** we have a timestamp t_s which represents the valid time of transaction **s**.
Every item has a period of life or *lifespan* in the database, which explicitly represents the temporal duration of the item information, *i.e.* the time in which the item is relevant to the user. The lifespan of an item A_i is given by an interval $[t_1, t_2]$, with $t_1 \leq t_2$.

Definition 2: Let A_i an item of **R**. With each item A_i and database **d,** we associate a *lifespan* defined by a time interval $[A_i.t_1, A_i.t_2]$ or simply $[t_1, t_2]$ if A_i is understood. $l : A_i \rightarrow 2^T$ is a function assigning a lifespan to each item A_i in **R**. We will refer to this lifespan as lAi. Then, we define $\mathbf{l_d}$, the lifespan of **d**, as $\mathbf{l_d} = \cup$ lAi , $\forall$ i.

Definition 3: Let $\boldsymbol{X} \subseteq$ **R** a set of items, **s** *contains* $\boldsymbol{X}$, or $\boldsymbol{X}$ *is verified* in **s**, if $\boldsymbol{X} \subseteq$ **s**. The set of transactions in **d** that contain $\boldsymbol{X}$ is indicated by **V**($\boldsymbol{X}$, **d**) = { **s** | **s** $\in$ **d** $\wedge$ $\boldsymbol{X} \subseteq$ **s**}. If the cardinality of $\boldsymbol{X}$ is k, $\boldsymbol{X}$ is called a k-itemset.
The lifespan of a k-itemset $\boldsymbol{X}$, with k > 1, is [t, t'] where t = max{t_1| $[t_1, t_2]$ is the lifespan of an item A_i in $\boldsymbol{X}$} and t'= min{ t_2| $[t_1, t_2]$ is the lifespan of an item A_i in $\boldsymbol{X}$ }.
As set operations are valid over lifespans, then the lifespan l_X of the k-itemset $\boldsymbol{X}$, where $\boldsymbol{X}$ is the union of the (k-1)-itemsets $\boldsymbol{V}$ and $\boldsymbol{W}$ with lifespans $l_{\boldsymbol{V}}$ and $l_{\boldsymbol{W}}$, respectively, is given by $l_{\boldsymbol{X}} = l_{\boldsymbol{V}} \cap l_{\boldsymbol{W}}$.

Definition 4: Let $\boldsymbol{X} \subseteq$ **R** be a set of items and $l_{\boldsymbol{X}}$ its lifespan. If **d** is the set of transactions of the database, then $\mathbf{d}_{l_{\boldsymbol{X}}}$ is the subset of transactions of **d** whose timestamps $t_i \in l_{\boldsymbol{X}}$. By $|\mathbf{d}_{l_{\boldsymbol{X}}}|$ we indicate the number of transactions of $\mathbf{d}_{l_{\boldsymbol{X}}}$.
In the nontemporal association rules model the following definition of support holds.

Definition 5: The *support* of $\boldsymbol{X}$ in **d**, denoted by $s(\boldsymbol{X}, \mathbf{d})$, is the fraction of the transactions in **d** that contains $\boldsymbol{X}$: $|\mathbf{V}(\boldsymbol{X}, \mathbf{d})| / |\mathbf{d}|$. The *frequency* of a set $\boldsymbol{X}$ is its support. Given a support threshold $\sigma \in [0, 1]$, $\boldsymbol{X}$ is *frequent* if $s(\boldsymbol{X}, \mathbf{d}) \geq \sigma$. In this case, it is said that $\boldsymbol{X}$ has *minimum support.*

In this paper we want to broaden the definition of support in order to include cases such as the proposed in the initial example. In other words, the incorporation of time would let us determine if an itemset is frequent by computing the ratio between the number of transactions that contain the itemset and the number of transactions in the database such that their valid time is included in the itemset's lifespan. Evidently, we need to filter items, and then the itemsets, with very short life as, for example, an item that has been sold just once would then have a support of 100%. For this reason we define **temporal support** as the amplitude of the lifespan of an itemset. We also define a *threshold* for the temporal support: if $\mathbf{l_d}$ is the lifespan of the database and $|\mathbf{l_d}|$ is its duration, then the threshold of the temporal support τ is a fraction of $|\mathbf{l_d}|$. Thus, for example, if the transactions correspond to a period of **n** months, τ **,** a fraction of **n** months**,** represents a lower bound for the temporal support of an itemset.

If the quantity of transactions of the database is $|\mathbf{d}|$, then $|\mathbf{d}| \, . \, \tau \, / \, |\mathbf{l_d}|$ would give us an aproximation to the minimum quantity of transactions to be considered as sample size. Then $|\mathbf{d}| \, . \tau \, / \, |\mathbf{l_d}|$ should be a statistically significant value, at the user's criteria.

On the other hand, the user could specify a time instant t_o, such that any item whose lifespan is $[t_1, t_2]$ and $t_2 < t_o$ is considered obsolete.

The new definition for support in the temporal model would now be:

Definition 6: The *support* of X in d over its lifespan l_X, denoted $s(X, l_X, \mathrm{d})$, is the fraction of transactions in d that contains X during the interval of time corresponding to l_X: $|V(X, \mathrm{d})| / |d_{l_X}|$. The *frequency* of a set X is its support. Given a threshold of support $\sigma \in [0, 1]$ and a threshold of temporal support τ , X is *frequent* in its lifespan l_X if $s(X, l_X, \mathrm{d}) \geq \sigma$ and $|l_X| \geq \tau$. In this case, it is said that X has *minimum support* in l_X.

The support threshold or frequency σ is a parameter given by the user and is dependent on the application. Likewise, the temporal support threshold τ is given by the user and also depends on the applications. Following we will introduce an example to clarify the definitions given up to now.

> **Example 1**: let **R** = { *A, B, C, D, E, F, G, H, I*} and **d** composed by the following six transactions:
>
> s_1 = {*A, C, F, H, I*}, t: 1
> s_2 = {*A, B, C, G*}, t: 2
> s_3 = {*B, C, D, G, I*}, t: 3
> s_4 = {*A, C, I*}, t: 4
> s_5 = {*C, D, E, H, I*}, t: 5
> s_6 = {*A, D, F, G*}, t: 6

If we establish as minimum support $\sigma = 0.45$ and minimum temporal support $\tau = 3$, we could now find the frequent $\boldsymbol{X}$, resulting in

$$\boldsymbol{X_1} = \{A\},\ l_{\mathbf{X}_1} = [1,6], \text{ since we find } A$$

between s_1 and s_6, which have stamped times 1 and 6, respectively. The support of $\{A\}$ is computed as $s(\{A\}, l_{\{A\}}, \mathbf{d}) = |\mathbf{V}(\{A\}, \mathbf{d})| / |\mathbf{d}_{[1,6]}| = 4 / 6 = 0.67$; the temporal support of A is $|l_A| = 6$. Then $\boldsymbol{X}_1 = \{A\}$ is frequent because $s(\{A\}, l_{\{A\}}, \mathbf{d}) = 0.67 > \sigma$ and $|l_A| = 6 > \tau$.

In the same way the following frequent itemsets are obtained; we indicate just their lifespan:

$X_2 = \{C\}$, $l_{X_2}=[1,5]$;
$X_3 = \{D\}$, $l_{X_3} = [3, 6]$;
$X_4 = \{G\}$, $l_{X_4}=[2,6]$;
$X_5 = \{I\}$, $l_{X_5}=[1,5]$
$X_6 = \{A, C\}$, $l_{X_6} = [1, 5]$;
$X_7 = \{C,D\}$, $l_{X_7} = [3, 5]$;
$X_8 = \{C, I\}$, $l_{X_8} = [1, 5]$.

The empty set ∅ is trivially frequent, so it is not considered since it is not interesting.
A temporal association rule expresses that a set of items tends to appear along with another set of items in the same transactions, in a specific time frame.

Definition 7: A *Temporal Association Rule* for **d** is an expression of the form $X \Rightarrow Y\ [t_1, t_2]$, where $X \subseteq \mathbf{R}$, $Y \subseteq \mathbf{R} \setminus X$, and $[t_1, t_2]$ is a time frame corresponding to the lifespan of $X \cup Y$ expressed in a granularity determined by the user**.**
A temporal association rule has three factors associated with it: *support*, *temporal support*, both already defined, and *confidence*, that will be defined next.

Definition 8: The *confidence of a rule* $X \Rightarrow Y\ [t_1, t_2]$, denoted by $conf(X \Rightarrow Y, [t_1,t_2], \mathbf{d})$ is the conditional probability that a transaction of **d**, randomly selected in the time frame$[t_1, t_2]$, that contains X also contains Y:

$$conf(X \Rightarrow Y,[t_1, t_2], \mathbf{d}) = s(X \cup Y, l_{X \cup Y}, \mathbf{d}) / s(X, l_{X \cup Y}, \mathbf{d}),$$

where $l_{X \cup Y} = \{[t_1, t_2]\}$.

Definition 9: The temporal association rule $X \Rightarrow Y\ [t_1, t_2]$ *holds* in **d** with *support s, temporal support* $|l_{X \cup Y}|$ and *confidence c* if s% of the transactions of **d** contain $\mathbf{X} \cup \mathbf{Y}$ and c% of the transactions of **d** that contain **X** also contain **Y,** in the time frame $[t_1, t_2]$.
Given a set of transactions **d**, and minimum levels of support, temporal support, and confidence, the problem of temporal association rule discovery is to generate all the association rules that have at least the given support, temporal support and confidence.

Example 2: following with the previous example, suppose that we establish the level of minimum confidence θ as being 0.7. From the frequent set $\{A, C\}$ we can consider two possible rules: $A \Rightarrow C$ [1,5] and $C \Rightarrow A$ [1,5]. The first has confidence $conf(A \Rightarrow C,[1,5], \mathbf{d}) = (3/5) / (3/5) = 1.0$ which is superior to the minimum θ = 0.7. The second has confidence $conf(C \Rightarrow A,[1,5], \mathbf{d}) = (3/5) / (5/5) = 0.6$, so it is discarded.

4. TEMPORAL RULES DISCOVERY

The discovery of all the association rules in a transaction set **d** can be made in two phases [AIS93]:

Phase 1. Find every set of items(*itemsets*) $X \subseteq \mathbf{R}$ that is frequent, *i.e.* their frequency exceeds the established minimum support σ .

Phase 2. Use the frequent itemsets X to find the rules: test for every $Y \subset X$, with $Y \neq \emptyset$, if the rule $X \setminus Y \Rightarrow Y$ satisfies with enough confidence, *i.e.* it exceeds the established minimum confidence θ .

In the following paragraph, we introduce suitable modifications to support temporal association rules discovery:

Phase 1T. Find every itemset $\boldsymbol{X} \subseteq \mathbf{R}$ such that $\boldsymbol{X}$ is *frequent* in its lifespan $l_{\boldsymbol{X}}$, *i.e.* $s(\boldsymbol{X}, \mathbf{l_X}, \mathbf{d}) \geq \sigma$ and $|l_{\boldsymbol{X}}| \geq \tau$.

Phase 2T. Use the frequent itemsets $\boldsymbol{X}$ to find the rules: verify for every $\boldsymbol{Y} \subset \boldsymbol{X}$, with $\boldsymbol{Y} \neq \varnothing$, if the rule $\boldsymbol{X} \setminus \boldsymbol{Y} \Rightarrow \boldsymbol{Y}$ $[t_1, t_2]$ is satisfied with enough confidence, in other words, exceeds the minimum confidence θ established in the interval $[t_1, t_2]$.

4.1 Generating Frequent Itemsets

Any of the proposed algorithms in the literature ([5], [9], [15]) for association rule discovery may be conveniently modified for its application for temporal association rules.

Let's see, for example, the *Apriori* algorithm of [5] into which we will introduce some slight changes to generate association rules taking time into consideration. As in the original notation, L_k represents the set of frequent k-itemsets. Each member of this set will have the following fields: i) itemset, ii) lower and upper limits of the life time of the item: t_1 and t_2, iii)count of support (Fr) of the itemset in $[t_1, t_2]$ and iv) total number of transactions (FTr) found in the interval $[t_1, t_2]$. C_k is the set of candidate k-itemsets; in other words, potentially frequent itemsets that have associated the same information as the members of L_k.

1. L_1 = { 1- frequent itemsets }; /*for each itemset of size 1we register the time of its first appearance in t_1 and the time of its last appearance in t_2 ; in FTr we count the amount of transactions registered in the interval $[t_1, t_2]$ and delete the itemset if it is obsolete*/
2. **for** (k = 2; $L_{k-1} \neq \varnothing$; k ++) **do begin**
3. C_k = *apriori-gen*(L_{k-1}); /*new candidates with their associated lifespan */
4. **foreach** transaction s ∈ d **do begin**
   $C_{\mathbf{s}}$ = *subset* (C_k , **s**); /*candidates c in **s** and such that timestamp t of **s** is in the interval $[t_1, t_2]$ of c */
5. **foreach** candidate c ∈ $C_{\mathbf{s}}$ **do**
6. c.Fr ++*;*
7. **foreach** candidate c ∈ C_k **do /*** such that timestamp t of **s** is in the interval $[t_1, t_2]$ de c */
8. update c.FTr;
9. **end**
10. $L_k = \{c \in C_k \,|(c.Fr \geq \sigma .FTr) \wedge (c.t_2 - c.t_1 \geq \tau)\}$
11. **end**
12. Answer = $\cup_k L_k$;

L_1 is obtained in the first pass, in which the items' occurrences are counted to determine the 1-frequent itemsets. For each itemset we store its lifespan $[t_1, t_2]$. Besides counting the absolute frequency for each itemset, Fr, we count the total number of transactions between t_1 and t_2, FTr. Then if Fr / FTr ≥ minimum support σ and if $t_2 - t_1 \geq$ minimum temporal support τ **,** we will say that the itemset X has minimum support in $[t_1, t_2]$.

Some items could be deleted from L_1 because they were obsolete, i.e., they have interval lifespans $[t_1, t_2]$ and $t_2 < t_o$. After deleting the obsolete items, the following lemma assures us that it is not necessary to check for obsolete k-itemsets, with k > 1, anymore.

Lemma: *An k- itemset, with k > 1, is obsolete if and only if contains an obsolete item.*

Proof: based on the definition of k-itemset lifespan, with k > 1.

Every following pass k consists of two phases: in the first are obtained the candidate itemsets C_k of size k, based on the frequent itemsets L_{k-1} of size k-1 obtained in the k-1 pass, by means of the function *apriori-gen*. The lifespan of a k-itemset with k > 1 is obtained in the following way: if the k-itemset **u** is obtained putting together k-1-itemsets **v** and **w,** then the lifespan of **u** is $[\mathbf{u}.t_1, \mathbf{u}.t_2]$, with $\mathbf{u}.t_1 = \max\{\mathbf{v}.t_1, \mathbf{w}.t_1\}$ and $\mathbf{u}.t_2 = \min\{\mathbf{v}.t_2, \mathbf{w}.t_2\}$. In the second phase we read the transactions' database to compute the support of the candidate itemsets of C_k, for which the function *subset* is used*;* it determines if each c member of C_k is contained in the transaction **s**. The timestamp t of **s** must satisfy $t \in l_c$. The algorithm does as many passes over the database as the maximal cardinality of a candidate itemset.

Generating a priori candidates

The candidates' generation is made by means of the function *apriori-gen*. This function takes as argument L_{k-1}, the set of all frequent (k-1)-itemsets, and returns a superset of the frequent k-itemsets, each one with their associated lifespan represented generically by the interval $[t_1, t_2]$. This function has been organized into a *Join Step and*a *Pruning Step*.

1. Join Step:

In SQL

insert into C_k
select $p.item_1, p.item_2, ..., p.item_{k-1}, q.item_{k-1}$
/* each resulting itemset will have an
associated interval $[t_1, t_2]$ such that
t_1= max { t_1 of the (k-1)-itemsets joined }
t_2 = min { t_2 of the (k-1)-itemsets joined} */
from L_{k-1} p, L_{k-1} q
where $p.item_1 = q.item_1$ *and ... and*
$p.item_{k-2} = q.item_{k-2}$ *and*
$p.item_{k-1} < q.item_{k-1}$;

In the next step (*pruning*) all the candidate itemsets $c \in C_k$ such that any subset of c with k-1 items is not in L_{k-1} are deleted. In the same way the itemsets c such that $|l_c| < \tau$ are deleted too.

2. Pruning Step:

foreach itemset $c \in C_k$ **do**
if $|l_c| < \tau$ **then**
delete c from C_k
else
foreach (k-1) subsets s in c **do**

if ($s \notin L_{k-1}$) **then**
delete c from C_k;

Example 3: Given **R** = {A, B, C, D, E, F, G, H, I} and a transactions database **d**, use the temporal version of *Apriori* to determine all the frequent sets of items of **R** in **d**. Assume that the minimum support is fixed at $\sigma = 0.4$ and minimum temporal support is $\tau = 3$. In the first pass L_1 is obtained; each original item has a defined lifespan, observed in the reading of the database. From now on, the itemsets' lifespan will be calculated as a function of the items' lifespan or component itemsets in the Apriori-gen function (join step).

Figure 1 shows the example in detail.

4.2 Generating Rules

To generate the rules, it is necessary to find all the subsets for every frequent itemset. Then, given a frequent itemset Z we must find, for each proper subset X of Z, the rules $X \Rightarrow (Z - X)\ [t_1, t_2]$ such that $s(Z,l_Z,\mathbf{d}) / s(X,l_Z,\mathbf{d}) \geq \theta$.

One of the problems we find in computing confidence, in accordance with the definition 8,

$$conf(\mathbf{X} \Rightarrow \mathbf{Y},[t_1, t_2], \mathbf{d}) = s(\mathbf{X} \cup \mathbf{Y}, l_{X \cup Y}, \mathbf{d}) / s(\mathbf{X}, l_{X \cup Y}, \mathbf{d}),$$

where $l_{X \cup Y} = \{[t_1, t_2]\}$, is the determination of $s(\mathbf{X}, l_{X \cup Y}, \mathbf{d})$. Evidently, $s(\mathbf{X}, l_{X \cup Y}, \mathbf{d})$ may not be equal to $s(\mathbf{X}, l_X, \mathbf{d})$, since $l_{X \cup Y} \subseteq l_X$. But in Phase1T we have calculated $s(\mathbf{X}, l_X, \mathbf{d})$, and not $s(\mathbf{X}, l_{X \cup Y}, \mathbf{d})$. Since, if XY is a frequent itemset of size k we will have 2^k possible subsets, we should recalculate the frequency for 2^k-2 itemsets in $l_{X \cup Y}$, and repeat that in each k-th pass, with $k > 1$. A way to avoid this, is to use an estimation in its place. In the simplest case, if we consider that all itemsets **X** have a uniform temporal distribution, then the chance of appearance in any subset of $l_{\mathbf{X}}$, in particular in $l_{\mathbf{X} \cup \mathbf{Y}}$, will be the same. Then we will be able to estimate $s(\mathbf{X}, l_{\mathbf{X} \cup \mathbf{Y}}, \mathbf{d})$ as $s(\mathbf{X}, l_{\mathbf{X}}, \mathbf{d})$. Then, modifying an algorithm as [ASr94a] to obtain all possible rules, given a frequent itemset Z, is immediate.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have introduced time in the problem of association rules discovery, given place to what we call Temporal Association Rules. Each item, itemset and rule has now an associated lifespan, which comes from the explicitly defined time in database transactions. We have also introduced the concept of temporal support. This gives way to the discovery of new rules that, due to the lack of necessary support, were not discovered with the traditional viewpoint. Now, with the concept of time, we consider the rules that have enough support in their lifespan, as long as they also have temporal support.

One of the problems related to the discovery of association rules that is often mentioned, is the great number of rules that can be generated. A solution is that the user may say which dates are old enough, so the rules with lifespan previous to those dates would be considered obsolete and not presented to the user. Furthermore, if the algorithm used to generate the frequent itemsets finds old items or itemsets, it may eliminate them directly, which it would be an additional pruning.

To show the incidence of time in the amount and quality of the obtained rules we extend, as an example, the A priori algorithm that generates the frequent itemsets.

We are currently implementing our algorithm for temporal association rule discovery. Besides, we will analyze the problem of the maintenance of temporal association In addition to what was considered in [10], we

will investigate the concept of *temporal border*; the temporal border includes itemsets that do not have enough temporal support, but such that the upper limit of their lifespan corresponds, at less than a Δt, with the temporal limit of the original database.

Database **d**		C1	L1
T Tid Items		Itemset support LS	Itemset support LS
1 100 A C F H I		A 0.67 [1,6]	A 0.67 [1,6]
2 200 A B C G		B 1.0 [2,3]	C 1.0 [1,5]
3 300 B C D G I	scan **d**	C 1.0 [1,5]	D 0.75 [3,6]
4 400 AC I	---------->	D 0.75 [3,6]	G 0.6 [2,6]
5 500 C DEHI		E 1.0 [5,5]	H 0.4 [1,5]
6 600 ADFG		F 0.33 [1,6]	I 0.8 [1,5
		G 0.6 [2,6]	
		H 0.4 [1,5]	
		I 0.8 [1,5]	

C_2		C_2	L_2
Itemset		Itemset support LS	Itemset support LS
{A,C}		{A,C} 0.6 [1,5]	A,C} 0.6 [1,5]
{A,D}		{A,D} 0.25 [3,6]	{A,G} 0.4 [2,6]
{A,G}		{A,G} 0.4 [2,6]	{A,I} 0.4 [1,5]
{A,H}		{A,H} 0.2 [1,5]	{C,D} 0.67 [3,5]
{A,I}		{A,I} 0.4 [1,5]	{C,G} 0.5 [2,5]
{C,D}		{C,D} 0.67 [3,5]	{C,H} 0.4 [1,5]
{C,G}	scan **d**	{CG} 0.5 [2,5]	{C,I} 0.8 [1,5]
{C,H}	----------→	{C,H} 0.4 [1,5]	{D,G} 0.5 [3,6]
{C, I}		{C, I} 0.8 [1,5]	{D, I} 0.67 [3, 5]
{D, G}		{D,G} 0.5 [3, 6]	{H, I} 0.4 [1,5]
{D,H}		{D, H} 0.33 [3, 6]	
{D, I}		{D,I} 0.67 [3,5]	
{G, H}		{G, H} 0.0 [2.5]	
{G, I}		{G,I} 0.25 [2,5]	
{H, I}		{H,I} 0.4 [1,5]	

C_3	C_3	L_3
Itemset	Itemset support LS	Itemset support LS

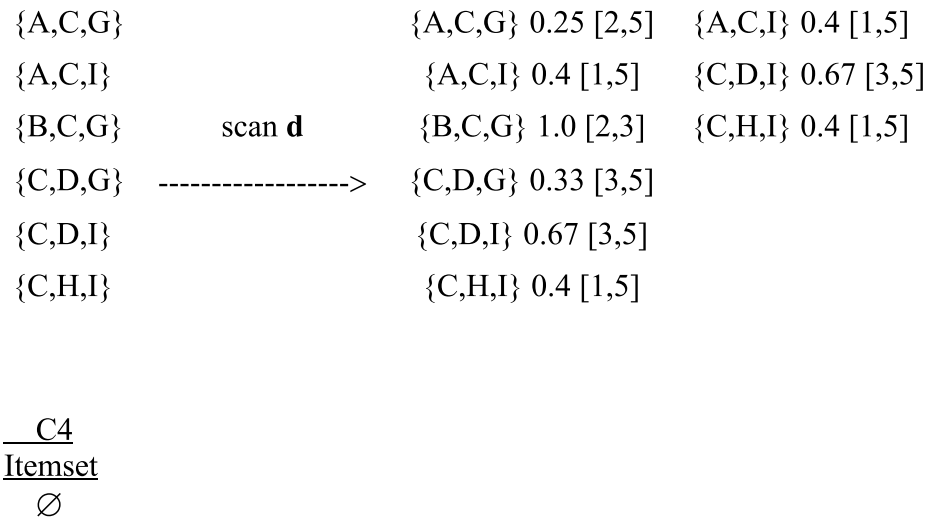

Figure 1: Example 3

6. REFERENCES

[1]. Agrawal, R.-Imielinski, T.-Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. Proc. ACM SIGMOD:207-216. 1993.

[2]. Agrawal, R.-Imielinski, T.-Swami,A.: Database mining: A performance perspective. IEEE TOKDE Vol.5 Nº5: 914-925. Oct.1994.

[3]. Agrawal, R.-Mannila, H.-Srikant, R.- Toivonen, H- Verkano, I.: Fast Discovery of Association Rules. In Advances in KD and DM: 307-328. The MIT Press. 1996.

[4]. Agrawal, R.-Srikent, R.: Fast Algorithms for Mining Association Rules. IBM Res. Rep. RJ9839, IBM Almaden. June 1994.

[5]. Agrawal, R.-Srikant, R.: Fast Algorithms for Mining Association Rules. Proc. of the 20th VLDB Conference: 478-499. 1994.

[6]. Agrawal, R.-Srikant, R.: Mining Sequential Patterns. Proc. IEEE Int'l.Conference on Database Engineering: 3-14. 1995.

[7]. Bettini, C-Wang, X.-Jajodia, S.: Testing Complex Temporal Relationships Involving Multiple Granularities and Its Application to Data Mining. Proc. of the ACM PODS'96: 68-78. 1996.

[8]. Bettini, C-Wang, X.-Jajodia, S.-Lin, J.: Discovering Frequent Event Patterns with Multiple Granularities in Time Sequences. IEEE TOKDE Vol.10 Nº 2: 222-237. April 1998.

[9]. Brin, S.-Motwani, R.-Ullman, J.-Tsur, S.: Dynamic Itemset Counting and Implication Rules for Market Basket Data. Proc. ACM SIGMOD: 255-264. 1997.

[10]. Cheung, D.-Han, J.-Ng, V.-Wong, C.: Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. Proc. Of 1996 Int'l Conf. On Data Engineering. Feb.1996.

[11]. Chen, X.-Petrounias, I.-Heathfield,H.: Discovering Temporal Association Rules in Temporal Databases. Proc. Int'l Workshop IADT'98. July 1998.

[12]. Kimball, Ralph: *The Data Warehouse Toolkit*. John Wiley & Sons. 1996.

[13]. Mannila, H.-Toivonen, H.-Verkamo, I: Discovering Frequent Episodes in Sequences. KDD'95. AAAI: 210-215. August 1995.

[14]. Ozden, B.-Ramaswamy, S.-Silberschatz, A.: Cyclic Association Rules. ICDE 1998.

[15]. Park, J.S.-Chen, M.S.-Yu, P.S.: An Effective Hash Based Algorithm for Mining Association Rules. Proc ACM SIGMOD: 175-186. 1995.

[16]. Ramaswami, S.-Mahajan, S- Silberschatz, A.: On the Discovery of Interesting Patterns in Associations Rules. Proc. 24th VLDB Conf. 1998.

[17]. Srikant, R.-Agrawal, R.: Mining Generalized Association Rules. Proc. 21st VLDB Conference: 407-419. Zurich. 1995.

[18]. Srikant, R.-Agrawal, R.: Mining Quantitative Association Rules In Large Relational Databases. Proc. ACM SIGMOD: 1-12. 1996.

[19]. Srikant, R.-Agrawal, R.: Mining Sequential Patterns: Generalization and Performance Improvements. In Advances in Database Technology-EDBT'96. Lectures Notes in CS 1057. Springer. 1996.

[20]. Tansel, A.-Ayan, N.: Discovery of Association Rules in Temporal Databases. Fourth Int'l Conference on KDD Workshop on Distributed Data Mining. August 1998.

[21]. Tansel, A. et al: *Temporal Databases: Theory, Design, and Implementation.* Benjaming/Cummings. 1993.

Juan M. Ale is full professor at La Matanza University and head of the db research group in the same University. He holds degrees in scientific computation, systems engineering and computer sciences from Buenos Aires University. Currently he is a PhD candidate at La Plata University. His current research interests include data mining, data warehousing and temporal databases.

Gustavo H. Rossi is full professor at La Plata University and head of LIFIA in the same University. He holds a PhD degree in computer science from PUC-Rio, Brazil. His current research interests are Web Information Systems and Navigation Patterns for E-commerce.

SAC 2000 March 19-21 Como, Italy
(c) 2000 ACM 1-58113-239-5/00/003>...>$5.00