Control and Cybernetics

vol. **38** (2009) No. 1

An experimental evaluation of two approaches to mining context based sequential patterns*

by

Jerzy Stefanowski and Radosław Ziembiński

Institute of Computing Science, Poznan University of Technology
ul. Piotrowo 2, 60–965 Poznań
e-mail: {jerzy.stefanowski, radoslaw.ziembinski}@cs.put.poznan.pl

Abstract: The paper discusses the results of experiments with a new context extension of a sequential pattern mining problem. In this extension, two kinds of context attributes are introduced for describing the source of a sequence and for each element inside this sequence. Such context based sequential patterns may be discovered by a new algorithm, called Context Mapping Improved, specific for handling attributes with similarity functions. For numerical attributes an alternative approach could include their prediscretization, transforming discrete values into artificial items and, then, using an adaptation of an algorithm for mining sequential patterns from nominal items. The aim of this paper is to experimentally compare these two approaches to mine artificially generated sequence databases with numerical context attributes where several reference patterns are hidden. The results of experiments show that the Context Mapping Improved algorithm has led to better re-discovery of reference patterns. Moreover, a new measure for comparing two sets of context based patterns is introduced.

Keywords: knowledge discovery, sequential patterns mining, context patterns, similarity of patterns.

1. Introduction

The sequential pattern mining is one of essential tasks of data mining. Its definition was introduced by Agrawal and Srikant (1995) and can be shortly presented in the following way – for a given *database of sequences* find all sequential patterns with a user-specified *minimum support threshold*. The database contains a set of sequences where each sequence is a list of elements (referring to transactions) ordered by an associated identifier. The element of a sequence is a set of *items* (shortly called an itemset). The support of the pattern (a sub-sequence) is counted as a number of sequences in the database including this sub-sequence.

*Submitted: June 2008; Accepted: October 2008.

An inclusion means that itemsets in the pattern are subsets of appropriate itemsets in the supporting sequence with preserving a pattern element order. This problem is illustrated in Fig. 2 by a simple shopping example of buying some goods.

Mining sequential patterns has led to many applications, e.g. analysing buying activities of customers in shops, behaviour of users of WWW servers, telecommunication, medicine; for a review see, e.g., Dong and Pei (2007), Morzy (2004). Moreover, it is a challenging research problem for the data mining community. Up to now, a lot of research concerned the most efficient approaches to mining sequential patterns from large data bases. So, several algorithms have been proposed, e.g. modifications of the Apriori algorithm, PrefixSpan, SPADE, GSP algorithms, see reviews in Han, Pei (2001). Simultaneously, the problem itself has been generalized to take into account other aspects of data characteristics, such as time constraints, time windows, quantitative mining and hierarchical patterns mining (Srikant and Agrawal, 1996, and Kim, Lim, Ng, Shim, 2004). Other approaches have also exploited structural dependencies between elements, mining frequent sub-trees and sub-graphs.

However, in the majority of these approaches itemsets containing sets of nominal values only are used and rather limited information about sources of transactions is exploited. This is also reflected by the inclusion operation while comparing sequences which is feasible for nominal values. These properties may limit modelling of some more complex real-life problems. In many cases sources of transactions can provide additional non-nominal information associated with either circumstances of transactions or the sequence itself (e.g. a description of the place, environment factors, duration, engaged tools or persons, etc.). Handling such information can be done by introducing two different sets of so called *context attributes*, attached both at the level of the sequence and its elements. These attributes may be defined on various scales, not only nominal but also numerical ones. In general, such context attributes allow to mine "richer" sequence patterns in the sense of providing more descriptions of circumstances for occurring frequent sub-sequences of events. Let us repeat that existing approaches to sequential patterns (shortly called traditional ones or abbreviated as TSPM) cannot handle these additional pieces of information. This difference is illustrated in Fig. 1, where one can notice additional data sources characterizing source of the sequence and other data referring to the context where the given transaction occurred.

Shortcomings of this traditional problem have led us to formulation of its generalization, called *context based sequential pattern mining* (shortly CBSPM), see Stefanowski, Ziembiński (2005). An important property is the ability to handle directly context attributes (also numerical ones) which implies using special functions measuring similarity of their values in contexts of a pattern and a sequence instead of the set inclusion operation used in TPSM. As CBSPM can not be solved by simple extensions of traditional mining algorithms, a new algorithm, called *ContextMappingImproved*, was developed in Ziembiński (2007).

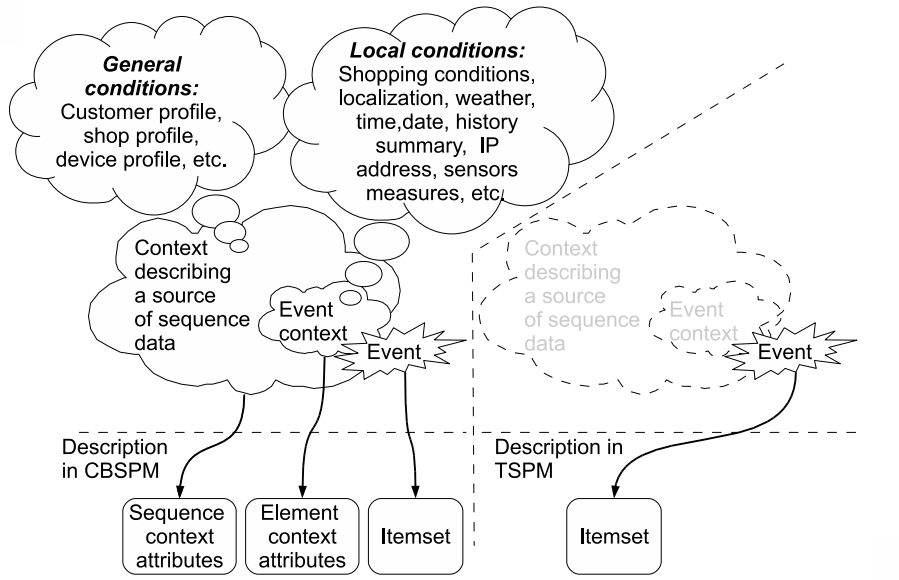

Figure 1. Differences between context data and typical data in CBSPM and TSPM problems.

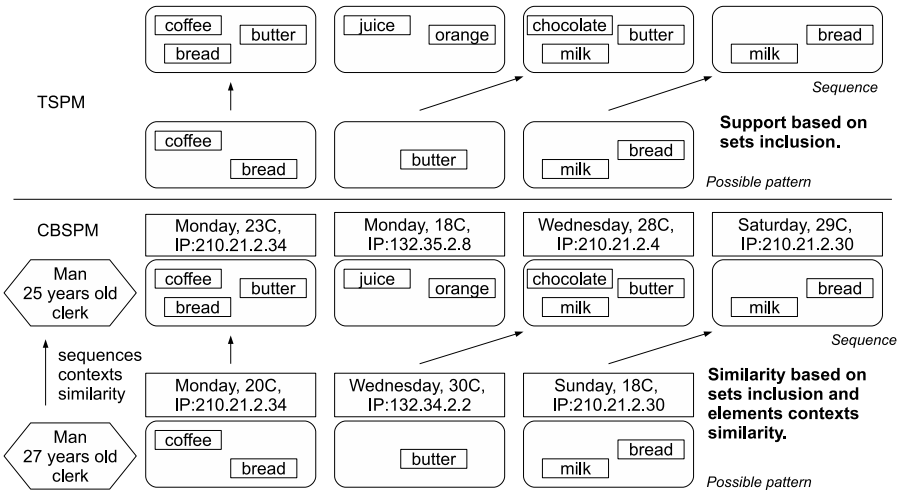

Figure 2. Comparing a pattern and a sequence in TSPM (an upper part of the figure) and CBSPM (a lower part).

Previous experiments showed that looking for context patterns may require more computational costs than for traditional sequential patterns (although the computational cost still depends on proper tuning values of the minimal support threshold). Thus, we could ask a question whether it is possible to discover context patterns in another way by their proper adaptation to the traditional problems. Following some motivations from Pinto at al. (2001), referring to another "multi-dimensional" sequential patterns mining problem, one can think about artificially introducing attributes as additional elements into items – though the authors referred to considered only nominal attributes characterizing the complete sequence. Although the traditional approach does not allow to process numerical data directly, it could be possible to consider yet another approach to handle context numeric data as a "*transformation approach*". In this approach all numeric attributes are pre-discretized and their discrete values could be transformed into artificial nominal items suitable to be processed by an extension of a "traditional" algorithm like PrefixSpan.

However, this leads us to another question about differences of results (let us say, the "quality" of patterns) offered by both approaches: ContextMappingImproved and the transformation of PrefixSpan with pre-disretization (the equal width and the equal frequency local methods). We propose to compare them by studying their ability to re-discover the original context based patterns hidden in an artificially generated sequences database. We are interested in verifying differences between results obtained from both algorithms and checking whether the approach with pre-discretization could be an alternative to the ContextMappingImproved algorithm. Measuring the patterns re-discovery degree also requires some research on a new measure as the problem of calculating similarities between two sets of context based sequential patterns has not been studied yet. A presentation of the results of this comparative, experimental study and an introduction of a new measure of similarity between context patterns are main goals of this paper.

The paper is organized in a following way. Section 2 contains a brief presentation of the problem of CBSPM and a general scheme of algorithms for mining patterns coherent to the definition of CBSPM. In Section 3, we introduce a new measure for comparing two sets of context patterns. Then, we describe a conditions of experiments and their results in Section 4. Moreover, we briefly present a generator of databases. The last section contains discussion of results of experiments performed and final conclusions.

2. Basic concepts of mining context sequential patterns

2.1. A definition of the CBSPM Problem

We briefly describe the CBSPM problem – for more details see Ziembiński (2007) and Stefanowski, Ziembiński (2005).

As in the TPSM problem $L = \{i_1, i_2, \ldots, i_n\}$ is a set of items and X is a non-empty subset of items. A *sequence* is an ordered list of elements $s = < t_1, t_2, \ldots, t_m >$, where $t_i \subseteq L, i = 1, 2, \ldots, m$. An element t_i of sequence is an itemset (X). A sequence $\alpha =< \alpha_1, \alpha_2, \ldots, \alpha_r >$ is *included* in another sequence $\beta =< \beta_1, \beta_2, \ldots, \beta_s >$ (what is denoted as $\alpha \sqsubseteq \beta$) if there exist integers $1 \leq j_1 < j_2 < \ldots < j_r \leq s$ such that $\alpha_1 \subseteq \beta_{j_1}, \alpha_2 \subseteq \beta_{j_2}, \ldots, \alpha_r \subseteq \beta_{j_r}$. In the set of sequences any sequence s is *maximal* if s is not included in any other sequence from the same set.

In the CBSPM problem two kinds of sets of context attributes are additionally introduced into the structure of sequences. The first set, called *sequence context*, denoted as $D = \langle D_1, ...D_v \rangle$, is a set of attributes describing complete sequence. It usually reflects some properties of the source of a sequence (e.g. a user profile). The second set of different attributes $C = \langle C_1, ..., C_w \rangle$ is called the *element context*. It is assigned to each element in the sequence and it describes circumstances of an event / transaction referring to the itemset in this element. The structure of context attributes is homogeneous for all objects of the same type (e.g. sequences, elements). It is also assumed that attributes may be defined on either nominal, ordinal or numerical scales.

Unlike the TSPM, in the context based problem, besides the inclusion of sequence elements, it is necessary to consider a *similarity* between a sequence (being a candidate for a pattern) and other sequences in the data base. This leads to calculating similarity between values of attributes, both for the sequence set D and for the transaction sets C. Thus, for each attribute a dedicated *similarity function* σ should be defined. For nominal attributes it can be based on a simple indiscernibility relation. However, for ordinal or numerical attributes more sophisticated functions should be used, e.g. based on distance measures, interval comparisons or even using some richer forms resulting from the domain knowledge about the problem at hand. Such similarity functions have to be pre-defined for the sequence context attributes D (denoted as $(\sigma^k(d_1^k, d_2^k),\ d_1^k, d_2^k \in D_{k \in 1..v})$) and element context attributes (denoted as $\sigma^k(c_1^k, c_2^k),\ c_1^k, c_2^k \in C_{k \in 1..w}$). The value of each similarity function is normalized to the range from 1.0 to 0.0 where 1.0 means identity of the compared attribute values pair and 0.0 means dissimilarity.

Then, values of similarities of values of single attributes are aggregated to a global evaluation of set of attributes using additional aggregation operators:

$$\Theta^C(CI_1, CI_2) = \Theta^C(\sigma_1^C(c_1^1, c_2^1), \sigma_2^C(c_1^2, c_2^2), ..., \sigma_w^C(c_1^w, c_2^w)), CI_1, CI_2 \in C,$$
$$\Theta^D(DI_1, DI_2) = \Theta^D(\sigma_1^D(d_1^1, d_2^1), \sigma_2^D(d_1^2, d_2^2), ..., \sigma_v^D(d_1^v, d_2^v)), DI_1, DI_2 \in D$$

for sequence context, and analogously for element contexts. The aggregation may be done using various operators like minimum, maximum, weighted sum or other aggregation functions (OWA). The value of this aggregation is normalized to the range [0,1]

Then, counting the support of a pattern by a sequence requires changing this continuous value into a binary one - support or not. So, if an aggregated

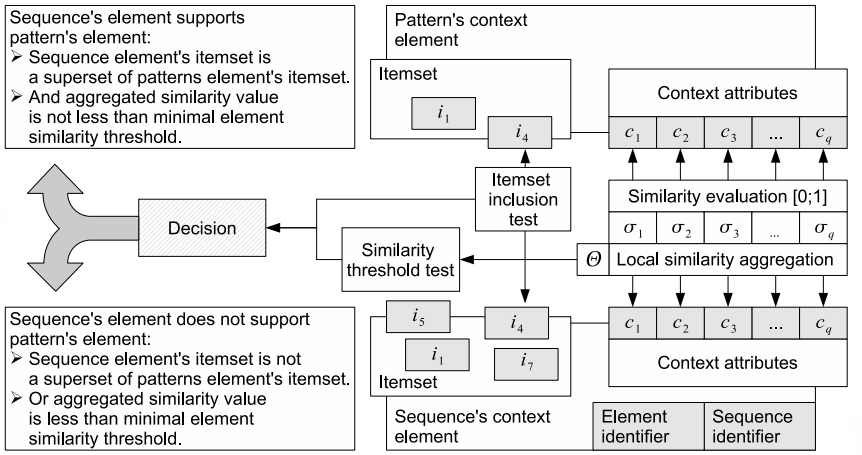

Figure 3. The support of a pattern by a sequence in the context based problem.

similarity value Θ^C (or Θ^D) is equal or greater than a *user defined threshold of minimal similarity* level θ^C (or θ^D) the context attribute sets are similar, otherwise they are considered dissimilar.

The essential concept of the support is extended in CBSPM in the following way. An element α is supported by another element β if the context of α is similar to the context of β (with respect to the predefined threshold level of similarity θ^C) and the itemset in the element α is included in the itemset of element β. This concept of calculating support is illustrated in Fig. 3.

A *context based pattern is supported by a context sequence* iff:

1. The context of the pattern is similar to the context of the sequence (with respect to the thresholds θ^D);
2. Each element in the pattern is supported by the respective element the sequence.

The problem of Context Based Sequential Pattern Mining is defined as: given a database of context sequences, where sets of context attributes D, C with their similarity functions σ and threshold values are defined, the task is to find all maximal context based patterns among all sequences supported by at least *min_ support* of sequences in the database of sequences.

Thresholds θ^D , θ^C and the *min_ support* are exploration parameters to be set by a user for a given problem.

2.2. Algorithms for discovery of context based patterns

Algorithms specific for CBSPM have been introduced in Ziembiński (2007). A limited space of this paper forbids us to present all details of the ContextMappingImproved algorithm – we briefly introduce the concept of *mapping similar values of attributes*, which is the essential step in it.

The definition of the problem assumes that an element of a sequence is frequent if the number of other sequences containing similar elements is greater than the minimal support threshold required by the user. In the first phase of the ContextMappingImproved algorithm a list of contexts (set of attributes) similar to other contexts in the database is constructed. Similarity between sets of attributes is calculated in the way defined in the previous sub-section with respect to appropriate similarity functions σ and threshold values θ^D or θ^C. If the number of similar contexts in the list, i.e. corresponding to different sequences, is greater than the value of the threshold *min_ support*, then this context is considered as frequent. Each frequent context is transformed into two interconnected artificial items, called A and B, respectively. In the original sequence item A replaces the frequent context (i.e. the set of attributes is replaced by it) and the item B is added to itemsets of all elements from other sequences on the list. It is assumed that only the item of type B supports the item of type A. Moreover, the sequence context is mapped to an additional artificial element added to the sequence - similarly to contexts of elements by artificial items. This mapping transforms similar context sequences into traditional-like database of sequences with nominal items only. In the next stage context patterns can be discovered by a specific adaptation of the PrefixSpan algorithm (Han et al. 2001). The algorithm ensures that all patterns must contain an additional element representing the sequence context and further sequence of elements being a pattern body. Each pattern element must contain an artificial item representing the frequent context and frequent itemset. The final step is connected to the reverse mapping of mined patterns. The replacement of artificial items with corresponding context values gives the resulting set of context patterns.

Some experiments from Ziembiński (2007) showed that this algorithm may be slower than PrefixSpan running on the same database with removed context attributes. As the mapping can be memory consuming, a heuristic version of the algorithm was also proposed. Its key point is preserving a limited number of the most similar mappings only and avoiding "explosion" of the number of additional items to be mined.

Another approach to handling values of numerical context attributes while discovering context pattern is briefly described below. Unlike in the above approaches, dedicated similarity functions are not considered, but domains of each numerical attributes are *discretized*, i.e. a range of a numerical attribute is automatically divided into a number of disjoint sub-intervals (folds) and original numerical values are transformed into discrete values (codes). Let us remark that a pre-discretization is quite often used in pre-processing before using many mining algorithms and can be performed by many algorithms, for a review see, e.g., Grzymala (2002), Yang, Webb and Wu (2005). Some pre-discretization algorithms more specific for association rules and other patterns have also been discussed in Agrawal et al. (1998). In our experiments, we decided to use low cost local discretization methods, namely: equal width or equal frequency folds.

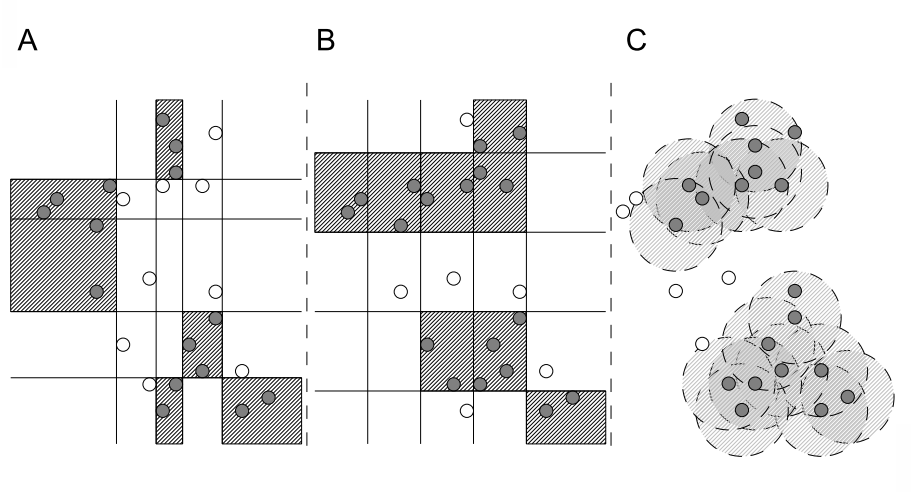

Figure 4. Discretizations of domains of numerical context attributes with equal frequency folds (A), equal width folds (B) – compared to a "similarity" based approach used in the CBSPM problem and the ContextMappingImproved algorithm (C).

Fig. 4 illustrates the difference in handling numerical attributes by this and the former approach. Differences in fold size and the layout for both discretization approaches are visible. One can also notice how the ContextMappingImproved algorithm constructs a local neighborhood of the value of a given context attribute while looking for similar values of this attribute in other sequences. It clearly illustrates the difference to the direct pre-discretization.

After the pre-discretization, each discrete value of an attribute corresponding to a particular discretization fold is transformed into an artificial item. Let us notice that in this transformation we radically change data granulation as one moves from numerical to nominal scale (as items do not carry any ordinal information). However, having nominal items it is possible to introduce them as additional components into the sequence and again adopt the Prefixspan algorithm to process such data structure. It should be noticed, however, that such patterns discovered from transformed sequences do not contain single attribute values as in the definition of the pattern in the CBSPM problem. Instead, they contain items representing folds – sub-intervals of context attributes.

3. Measuring similarity between context patterns

We want to evaluate the degree of similarity between two sets of context patterns finding out how much discovered patterns are similar to reference patterns hidden in the database containing context attributes. Some researchers already

studied similarity of sequences, e.g. for a clustering problem – for more details see Ronkainen (1998), Morzy, Wojciechowski, Zakrzewicz (1999), Guralnik, Karypis (2001), Yang, Wang (2003), but here we need to take into account sets of context attributes, which have not been studied before. So, a new measure of the similarity for sets of patterns has to be developed. As several aspects of context based pattern should be taken into account, we list below the requirements to the new measure:

1. Its value is normalised to the range of 1.0 to 0.0 where 1.0 means that both sets of patterns are exactly the same and 0.0 means that there is no similar information related to the context, itemsets and element order between both pattern sets.
2. The measure should "punish" two undesirable situations: when the set of discovered patterns is larger than the reference set (the algorithm may produce too many patterns) and when the algorithm discovers a smaller set of patterns than in the reference set (e.g. as a result of inappropriate discretization).
3. It should reflect a case when a shorter discovered pattern could be compared to a different combination of elements from the longer reference pattern. This may happen in a situation, where a longer pattern from the reference set contains few repetitions of the shorter discovered pattern.
4. A non-similar element in comparing patterns is treated as a kind of "noise" and should decrease the value of the measure. Resepectively, occurrence of unused elements in reference patterns should be negatively reflected.
5. It is necessary to handle the values of context attributes from the ContextMappingImproved algorithm and "sub-intervals" from TSPM with a discretization while comparing them to the reference pattern.

Taking into account all these requirements leads to quite a complex aggregation of basic similarities - so we decided to construct a similarity measure as an aggregation function (not a simple measure based on a kind of distance metric as often considered in data analysis).

3.1. Comparing two context patterns

Each element E from a pattern $mp = (D, S = \langle E_1, E_2, ..., E_k \rangle)$ contains a context (values of attributes) c and an itemset X. To evaluate similarity of itemsets Jaccard's coefficient can be used as in Guralnik, Karypis (2001). Assuming that element $E_1(C_1, X_1)$ is compared to the element from the second pattern $E_2(C_2, X_2)$ the Jaccard's coefficient is defined as:

$$\Theta^X(X_1, X_2) = \frac{|X_1 \cap X_2|}{|X_1 \cup X_2|}.$$

If both elements have no common items, the measure returns 0.0. Respectively, if both itemsets are identical, it gives 1.0. Similarity of context attributes

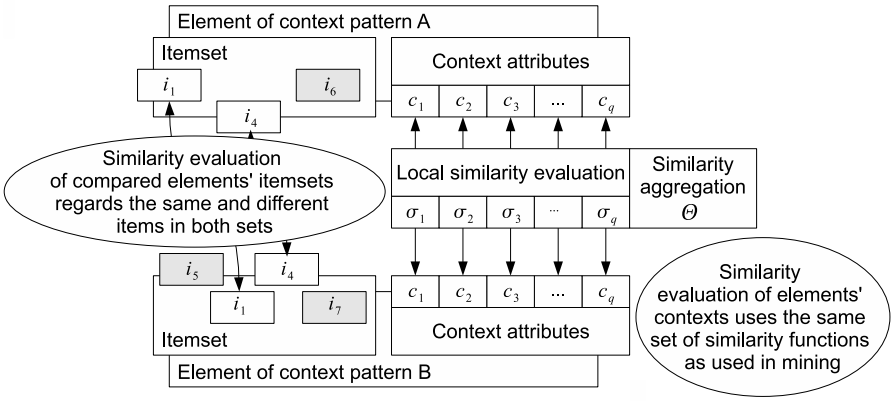

Figure 5. Calculating similarity of respective elements in two patterns.

is calculated with similarity functions in the same way as it was described in Section 2. However, thresholds of minimal similarities are not used here and the similarity between values of the sets of context attribute is calculated as continuous value using similarity functions: $\sigma^C(c_1, c_2)$ for element contexts or $\sigma^D(d_1, d_2)$ for sequence contexts. This concept is illustrated in the Fig. 5.

In a more difficult case of TSPM, the pre-discretization fold is compared to a value of the context attribute. For a pre-discretization fold an average similarity value is calculated using the upper $\overline{X_2^{D,C}}$ and lower $\underline{X_2^{D,C}}$ boundary of the pre-discretization fold. The aggregated similarity value is calculated using the same similarity aggregation functions as used in CBSPM to obtain comparable results in the same database:

$$\Theta^C(C_1, c_2) = \Theta^C(\sigma_1^{CX}(c_1^1, X_2^C), \sigma_2^{CX}(c_1^2, X_2^C), ..., \sigma_w^{CX}(c_1^w, X_2^C))$$
$$\Theta^D(D_1, d_2) = \Theta^D(\sigma_1^{DX}(d_1^1, X_2^D), \sigma_2^{DX}(d_1^2, X_2^D), ..., \sigma_v^{DX}(d_1^v, X_2^D)).$$

In experiments, presented later in the paper, the average function was used for aggregation. The total similarity of the compared pair of elements is calculated according to the following formulas:

$$\Psi^C(E_1(C_1, X_1), E_2(C_2, X_2)) = \Theta^C(C_1, c_2) \cdot \Theta^X(X_1, X_2)$$
$$\Psi^D(mp_1(D_1, S_1), mp_2(D_2, S_2)) = \Theta^D(D_1, D_2)$$

where $E_1 \in mp_1$ and $E_2 \in mp_2$ in both CBSPM and TSPM approaches. We thus take into account similarity values of compared sequence contexts, element contexts and element itemsets.

While comparing two patterns, mp_1 and mp_2, we often encounter a situation where compared patterns contain only a limited number of common similar ele-

ments. Such common elements create a core mr_{mp_1,mp_2}, identified as a common maximal sub-sequence of both patterns. Then, the core sequence must contain elements fulfilling the condition $\Psi^C(E_1(C_1, X_1), E_2(C_2, X_2)) > 0$. The core coefficient is defined as:

$$\Lambda(mp_1, mp_2) = \frac{|mr_{mp_1,mp_2}|}{max(|mp_1|, |mp_2|)}.$$

to take into account common elements and existing non paired elements between the reference pattern mp_1 and the discovered pattern mp_2.

The similarity of two patterns mp_1, mp_2 with the respect to a core mr_{mp_1,mp_2} is calculated according to the following equation:

$$mrsim(mp_1, mp_2, mr_{mp_1,mp_2}) = \frac{\Lambda(mp_1, mp_2) \cdot \Psi^D \cdot (\Psi_1^C + \Psi_2^C + ... + \Psi_r^C)}{r}$$

where $r = |mr_{mp_1,mp_2}|$.

If patterns are not of the same size, then it is possible to encounter a case where several different combinations of similar elements exist (multiple cores). MR denotes a set of such cores. The total similarity of two patterns mp_1, mp_2 is calculated by averaging similarities calculated for all possible cores in both compared patterns:

$$mpsim(mp_1, mp_2) = \frac{\sum\limits_{mr_{mp_1,mp_2} \in MR_{mp_1,mp_2}} mrsim(mp_1, mp_2, mr_{mp_1,mp_2})}{|MR_{mp_1,mp_2}|}.$$

3.2. Comparing two sets of context patterns

Let us consider two sets: of the reference patterns MP_r and of the discovered patterns MP_d. Similarity values of all pairs of patterns from the sets MP_r, MP_d can be stored in a special similarity matrix. Using them we can define some basic measures.

The reconstruction measure evaluates a re-discovery degree of hidden patterns and is defined as:

$$RMSim(MP_r, MP_d) = \frac{\sum_{i=1,...,k} Lsim(i)}{|MP_r|}$$

where $k = |MP_r|$ is the number of patterns in the reference set,

$$Lsim(i) = \sum_{mp_f \in L(mp_i)} mpsim(mp_f, mp_i)/|L(mp_i)|$$

and $L(mp_i)$ is the list of the most similar discovered patterns mp_f to the given i-th reference pattern mp_i. Technically, this list is constructed in the following

way: for each discovered pattern we find the most similar reference pattern and put the discovered pattern on the list attached to the reference pattern. If the similarity value of the next less similar reference pattern is very close to the first value (with respect to an assumed acceptance threshold) then the discovered pattern is also added to the list of this reference pattern. The process repeats for less similar reference patterns until threshold condition is met. When all discovered patterns have been processed, similarities between the reference pattern and those stored on its list are averaged, yielding $Lsim(i)$. If a list of a reference pattern is empty, then $Lsim(i)$ is equal to 0.0. Concluding, the purpose of the reconstruction measure is to find the degree of coverage of reference patterns by mined patterns.

Yet another possible measure is *the average similarity measure*. Its intuitive goal is to estimate an overall content of information in set of discovered patterns related to reference patterns with respect to any "noise" elements that discovered patterns may contain. By comparing to the previous basic measure this one is calculated simply by adding similarity values between a reference pattern and all mined patterns to the reference pattern list. List aggregation procedure is exactly the same – it is based on averaging. This measure allows for comparing all discovered patterns with each reference pattern, considering even completely dissimilar pairs. Therefore, values of the measure are nominally lower than values of the reconstruction measure, because the latter considers a small fraction of the most similar pairs of patterns.

4. Experiments

The main aim of the experiments carried out to evaluate the ability of two compared algorithm to re-discover the given number of context based patterns hidden in artificially generated sequence data bases. These patterns will further be called *reference patterns*.

As there were no ready benchmarks for context based sequence data bases we had to construct a special generator of "artificial" data, which could be parametrized to create required itemsets structure, context size and reference pattern support. Due to the paper size, we only give a general idea of this generator.

4.1. Generating a sequence data base

In general it is necessary to get a certain number of reference context patterns which are characterized by defined length, itemset structure, context size and support. It is assumed that the data base should contain a number (corresponding to the support) of slightly diversified context based sequences which should be sufficiently similar to an appropriate reference pattern.

There are two phases of generating such sequences: constructing itemsets and generating values of context attributes.

In the first phase for each reference pattern a required number of unique itemsets is randomly constructed from the set of available items. Itemsets are mutually dissimilar. Then, a required number of copies of one pattern are replicated to sequences in the database according to the assumed support threshold in such a way that these sequences are super-sets of the reference pattern, i.e. their itemsets may be additionally increased. If the length of sequences should be greater than the pattern length, additional random itemsets are randomly introduced as elements into these sequences.

In the second phase of generation of data bases, values of context attributes are assigned to these sequences. For each reference pattern, these contexts are generated using a structure of the kind of a "grid" in the multi-dimensional real number space, i.e. each distinct set of values of attributes corresponds to one node in this grid. Distances between nodes of this hypergrid can be changed to make contexts more distant or closer, i.e. reference patterns may be more or less similar to each other. The context values of each reference pattern are copied to "its" sequences in the data base. However, their values are slightly changed, i.e. randomly distorted within a defined hypersphere in a real number space. This ensures more realistic distribution of context values, not just simple copies of values – which could make discovery not so trivial. The generator performs rotations of the grid nodes according to randomly selected planes to eliminate linear distribution of nodes along axes. We can say that instances of context values in the data base create a kind of "cloud" around the context of the hidden pattern. Contexts describing patterns and their elements are independently generated using two different grids.

4.2. Conditions of experiments

The sequence of steps for each experiment is shown in the Fig. 6. The first step involves the generation of an artificial data base with the set of hidden reference patterns and chosen values of parameters. The same artificial data base is mined using both ContextMappingImproved algorithm and "transformation and discretization" approach. Both approaches discover a set of context patterns. This set of discovered patterns is compared against the set of reference patterns using the measure described in the previous section.

For the transformation approach the PrefixSpan algorithm was run with the equal frequency pre-discretization and with the equal folds width pre-discretization. Shortly speaking, the equal frequency pre-discretization divides the context attribute domain into folds containing the same number of contexts. The equal width pre-discretization creates folds with an equal width but containing different number of contexts. More details are given in Grzymala (2002).

For the application of the ContextMappingImproved algorithm we will use a similarity function assigned to each context attribute based on the following

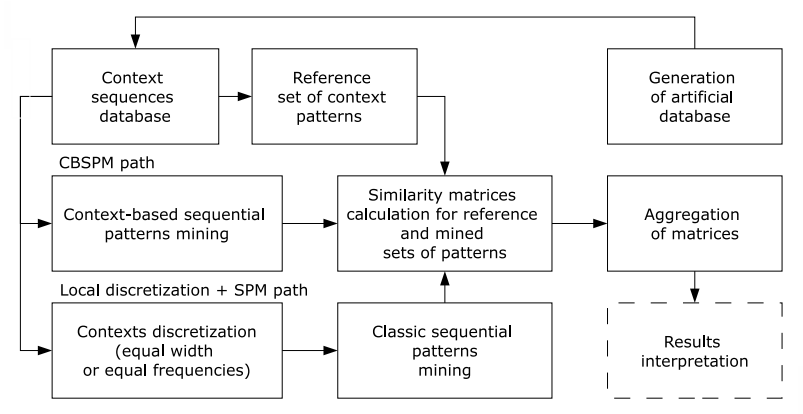

Figure 6. Basic steps in experiments.

formula:

$$\sigma^{C,D}(c_1, c_2) = \begin{cases} 1.0 - |c_1 - c_2| & \Leftrightarrow \quad |c_1 - c_2| < 1.0 \\ 0.0 & \Leftrightarrow \quad |c_1 - c_2| \geq 1.0 \end{cases}.$$

The ContextMappingImproved algorithm has been started with context similarity thresholds θ^C and θ^D equal to 0.4 (CPT1 label in figures), 0.6 (CPT2) and 0.8 (CPT3). A justification of these parameters is discussed later. The PrefixSpan pre-discretization divided each attribute domain values into 3, 5 and 7 folds. These "settings" are denoted in the figures as DES1, DES2, DES3 for the equal width pre-discretization and DEF1, DEF2, DEF3 for the equal frequency pre-discretization. The results presented were obtained for database containing eight hidden reference patterns containing four elements with support equal 0.25. Each database sequence has length of 12 elements and it contains subsequences that belongs to two hidden reference patterns and four additional randomly created elements. Random itemsets in elements do not overlap with itemsets used in reference pattern itemsets. A number of sequences in generated databases was 500. Sequence contexts and element contexts contain two attributes created around hypergrid nodes. The distance between nodes in three first experiments were greater than distortion spheres around nodes so the contexts from different elements were not too close to each other. In the fourth experiment the distance was gradually decreased to zero. Some results of other experiments correspond to very low values of a minimal support threshold – what in the case of consideration of a much higher numbers of sequences and the sequence length could make experiment infeasible. Moreover, we noticed that increasing too much the number of sequences or the length of sequences could make experiment infeasible when low values of minimal support or similarity thresholds were considered, so we skipped them. Due to the space limit,

we do not show results for other combinations of parameters, i.e. characteristics of reference patterns, as they are consistent with the presented ones.

4.3. Results of experiments

First results, presented in Fig. 7A, illustrate the influence of the minimal support threshold on the value of the re-discovery measure for both algorithms. The second experiment was focused on studying the relation between the number of pre-discretization folds and both measures for the transformation approach and traditional algorithms – see Fig. 8A. The value of the minimal support during mining was decreased to 0.04, because the TSPM approach with pre-discretization showed quite poor performance for higher threshold values (notice this in Fig. 7A). The third experiment aimed at studying an impact of the change of minimal similarity thresholds in CBSPM on the reconstruction ability, see Fig. 8B.

The final experiment verified the influence of the distance between generator grid nodes (i.e. the distance between appropriate contexts of reference patterns) on the quality of discovered patterns. The distance between nodes was gradually decreased (but with maintenance of the same radiuses of the distortion spheres). In consequence, the contexts that belonged to different patterns started to be to similar to each other, so creating a kind of "noise" for mining algorithms – see Fig. 7B.

5. Discussion and final remarks

Let us comment on the presented results of our experiments. One can easily notice that in the first experiment the values of the reconstruction measure for ContextMappingImproved algorithm (more specific for CBSPM) are at least five times higher than the results for TSPM with pre-discretization (see Fig. 7A). Moreover, only ContextMappingImproved algorithm could discover patterns at the value of a minimal support threshold comparable to the value of support of reference patterns applied while generating a data base.

On the other hand, we could say that the pre-discretization divided the space of values of context attributes into quite small folds containing usually small numbers of context instances. Therefore, the TSPM algorithms could find patterns only when the "density" of context values (referring to sequences) in folds is greater than the minimal support threshold. This occurs for much smaller value of the minimal support threshold than it was in the case of ContextMappingImproved. It seems that the ContextMappingImproved algorithm may work better because it recognizes much better "dense neighborhoods" around values of context attributes from the reference set of patterns due to the direct use of similarity functions, which is a kind of a characteristic feature of the CBSPM problem. The "transformation" approach with the equal width pre-discretization detects dense clusters of contexts quite poorly. The worst quality was obtained

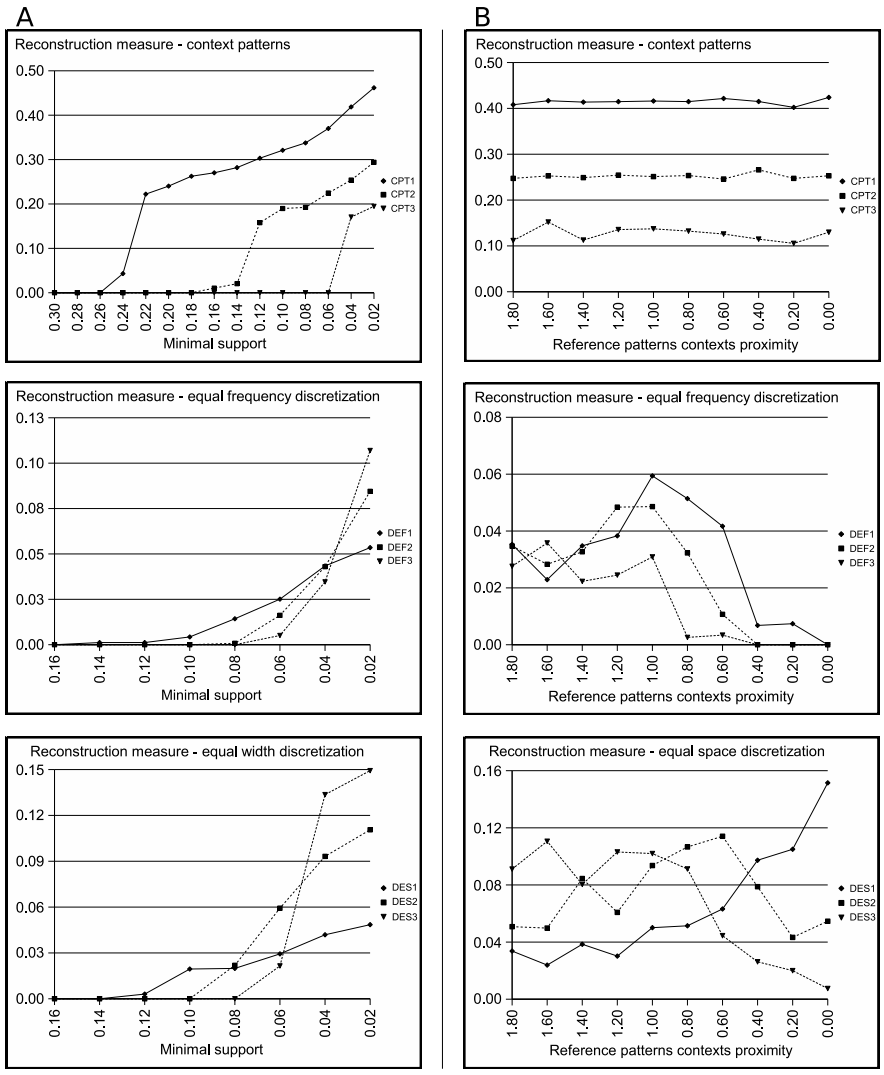

Figure 7. (A) The reconstruction measure for patterns discovered by both algorithms with respect to the minimal support threshold and (B) the "proximity" between values of context attributes in the reference patterns.

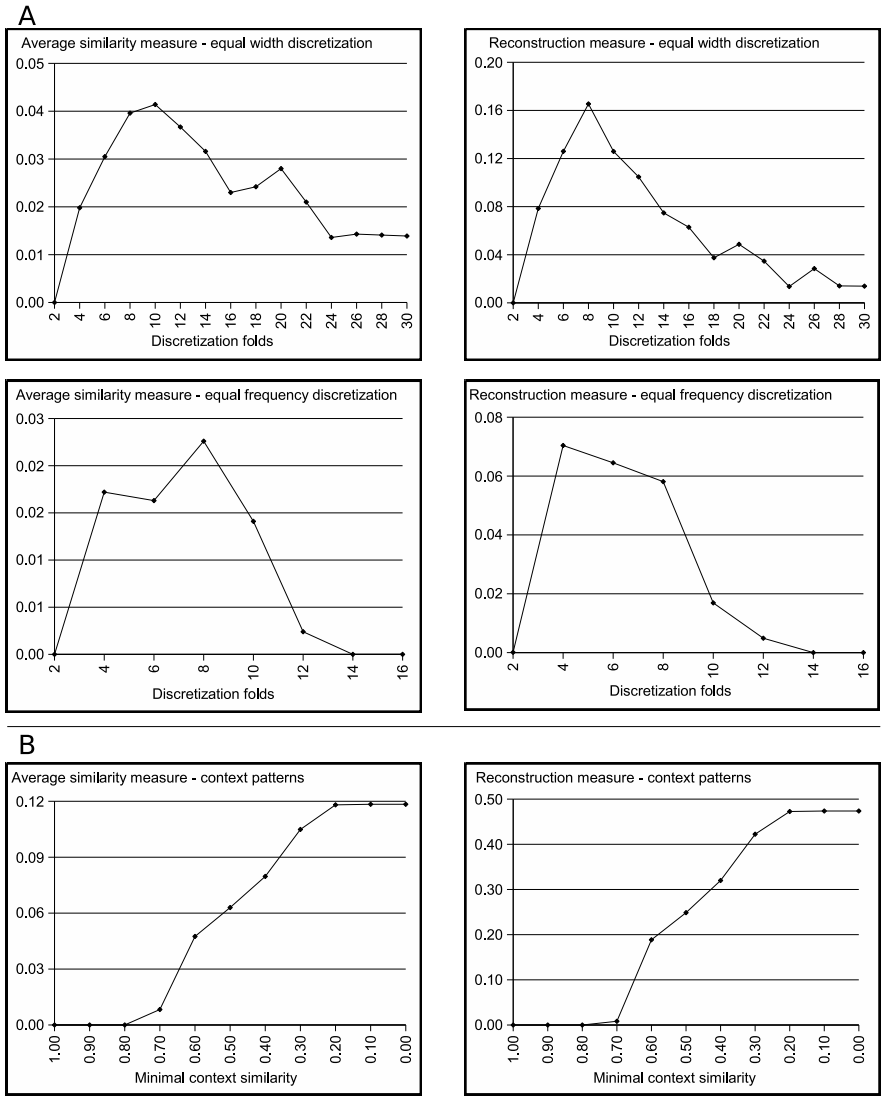

Figure 8. (A) The evaluation measures for patterns discovered by TSPM algorithm with respect to a number of pre-discretization folds. (B) The quality of patterns discovered by Context Mapping algorithm related to values of minimal context similarity thresholds (both are equal for sequence and element contexts).

by this approach when integrated with the equal frequency pre-discterization. We could say that a greater number of pre-discretization folds could improve the performance of the transformation approach. However, after increasing the number of folds again a lower value of the minimal support threshold is required to find patterns (see Fig. 7A where EFD3 finds patterns with a lower minimal support than EFD1). Additionally, the number of folds cannot be also set too high – see Fig. 8A – because a low value of the minimal support threshold may result in discovering a high number of less "accurate" patterns containing "noise" itemsets.

Another experiment with the ContextMappingImproved algorithm showed that minimal similarity thresholds in this case should be tuned in the range 0.3-0.7 – see Fig. 8B. Minimal similarity thresholds determine the size of reference context "neighborhood", where supporting instances of values of context attributes can be found. This algorithm should discover more patterns if similarity thresholds are getting lower (and the "neighborhood" is getting larger). However, such patterns may have worse quality, at the same, time because they may contain some "noise" elements that do not belong to reference patterns. Finally, the fourth experiment showed that reducing distance between values of reference context attributes has much weaker impact on results of the ContextMappingImproved algorithm than on the results of the transformation approach – see Fig. 7B.

To sum up, all experiments with re-discovery of hidden patterns clearly showed that the ContextMappingImproved algorithm worked much better than previously known algorithms using a transformation approach to pre-discretization. The computational costs were only slightly higher in some cases and even smaller in others, compared to pre-discretization with too many intervals.

Acknowledgment

The research of the first author was partially supported by the grant N N519 3505 33.

References

Agrawal, R. and Srikant, R. (1994) Fast algorithms for mining association rules. *Proc. of 20th International Conference on Very Large Data Bases*. Morgan Kaufmann, 487–499.

Agrawal, R., Gehrke, J., Gunopulos, D. and Raghavan, P. (1998) Automatic subspace clustering of high dimensional data for data mining applications. *Proc. of the 1998 ACM SIGMOD international conference on Management of data*. ACM Press, 94–105.

Dong, G. and Pei, J. (2007) *Sequence Data Mining*. Springer-Verlag, 1–12.

Grzymala-Busse, J. (2002) Discretization of Numerical Attributes. In: W. Klosgen and J. Zytkow, eds., *Handbook of Data Mining and Knowledge Discovery*. Oxford University Press, 218–225.

Guralnik, V. and Karypis, G. (2001) A Scalable Algorithm for Clustering Sequential Data. *Proc. of the 2001 IEEE International Conference on Data Mining*, San Jose, California. IEEE Computer Society Press, 179–186.

Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q. et al. (2001) PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. *Proceedings of the 17th International Conference on Data Engineering*. IEEE Computer Society, 215–224.

Kim, C., Lim, J., Ng, R.T. and Shim, K. (2004) SQUIRE: Sequential Pattern Mining with Quantities. *Proc. of 20th International Conference on Data Engineering (ICDE'04)*. IEEE Computer Society, 215–224.

Morzy, T. (2004) *Discovery associations: algorithms and data structures*. PAN Press, OWN Poznan.

Morzy, T., Wojciechowski, M. and Zakrzewicz, M. (1999) Pattern-Oriented Hierarchical Clustering. *Proc. of the Third East European Conference on Advances in Databases and Information Systems*, ADBIS'99, Maribor, Slovenia. **LCNS 1691**. Springer, 179–190.

Pinto, H., Han, J., Pei, J., Wang, K., Chen, Q. and Dayal, U. (2001) Multi-dimensional sequential pattern mining. *Proceedings of the 10th International Conference on Information and Knowledge Management*, Atlanta, Georgia. ACM Press, 81–88.

Srikant, R. and Agrawal, R. (1996) Mining Sequential Patterns: Generalizations and Performance Improvements. *Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology*. **LNCS**, **1057**, Springer-Verlag, 3–17.

Srikant, R. and Agrawal, R. (1996) Mining Quantitative Association Rules in Large Relational Tables. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*. ACM Press, 1–12.

Stefanowski, J. and Ziembiński, R. (2005) Mining Context Based Sequential Patterns. *Proceedings of the Third International Atlantic Web Intelligence Conference: Advances in Web Intelligence*. **LNCS 3528**, Springer-Verlag, 401–407.

Yang, Y., Webb, G. and Wu, Xindong (2005) Discretization Methods. In: O. Maimon and L. Rokach, eds., *Data Mining and Knowledge Discovery Handbook*. Springer, 113–128.

Ziembiński, R. (2007) Algorithms for Context Based Sequential Pattern Mining. *Fundamenta Informaticae*, **76** (4), 495–510.