Knowl Inf Syst (2011) 27:451–471
DOI 10.1007/s10115-010-0309-9

REGULAR PAPER

RP-Miner: a relaxed prune algorithm for frequent similar pattern mining

Ansel Yoan Rodríguez-González · José Francisco Martínez-Trinidad · Jesús Ariel Carrasco-Ochoa · José Ruiz-Shulcloper

Received: 14 August 2009 / Revised: 29 April 2010 / Accepted: 22 May 2010 /
Published online: 16 June 2010

Abstract Most of the current algorithms for mining frequent patterns assume that two object subdescriptions are similar if they are equal, but in many real-world problems some other ways to evaluate the similarity are used. Recently, three algorithms (*ObjectMiner*, *STreeDC-Miner* and *STreeNDC-Miner*) for mining frequent patterns allowing similarity functions different from the equality have been proposed. For searching frequent patterns, *ObjectMiner* and *STreeDC-Miner* use a pruning property called Downward Closure property, which should be held by the similarity function. For similarity functions that do not meet this property, the *STreeNDC-Miner* algorithm was proposed. However, for searching frequent patterns, this algorithm explores all subsets of features, which could be very expensive. In this work, we propose a frequent similar pattern mining algorithm for similarity functions that do not meet the Downward Closure property, which is faster than *STreeNDC-Miner* and loses fewer frequent similar patterns than *ObjectMiner* and *STreeDC-Miner*. Also we show the quality of the set of frequent similar patterns computed by our algorithm with respect to the quality of the set of frequent similar patterns computed by the other algorithms, in a supervised classification context.

Keywords Data mining · Frequent patterns · Mixed data · Similarity functions · Downward closure property

A. Y. Rodríguez-González (✉)
Data Mining Department, Advanced Technologies Application Center,
Siboney, Havana, Cuba
e-mail: arodriguez@cenatav.co.cu

A. Y. Rodríguez-González · J. F. Martínez-Trinidad · J. A. Carrasco-Ochoa
Department of Computer Science, National Institute of Astrophysics,
Optics and Electronics, Tonantzintla, Puebla, Mexico

J. Ruiz-Shulcloper
Advanced Technologies Application Center, Siboney, Havana, Cuba

1 Introduction

Data Mining emerged as an important research area at the beginning of the 1990s [1]. In Data Mining, the search of frequent patterns (frequent pattern mining) is a task in which lots of research have focused their attention. Since then, frequent pattern mining is usually the first and most expensive step of Association Rule Mining, another well-known and widely studied Data Mining task. Also, it is fundamental for many other data mining tasks, such as sequential patterns [4], episodes [16], associative classification [14], subspace clustering [2], causality [23] and partial periodicity [9]. Besides, frequent patterns describe regularities of a dataset and they could represent very important and hidden knowledge as user's profiles, modus operandi of some actions, syndromes, risk factors, etc. [10,12,13,15,26].

A *frequent pattern* is a combination of feature values of the objects of study that appears in a dataset with a frequency not less than a user-specified frequency threshold.

Most of the current algorithms for mining frequent patterns [8,5] assume that two object descriptions are similar when they are equal. However, in many real-world problems, object descriptions could be considered as similar whether they are equal or not. In those problems, the concept of similarity between object descriptions or its opposite, the concept of dissimilarity (not necessarily a distance) is usually employed to compare objects and count how many times an object appears in a dataset [11]. However, two real-world objects are not always exactly the same. In addition, and as a consequence, other concepts of similarity (not necessarily the opposite or the inverse of a distance, even not necessarily symmetric) are frequently used in soft sciences (geology [7], medicine [18,19], sociology [22], etc.) to make decisions. Furthermore, this concept has been employed in other areas such as information retrieval and document analysis [20,24,25].

A *frequent similar pattern* [5,21] is a combination of feature values of the object of study, such that, the accumulation of frequency of its similar patterns is no less than a user-specified frequency threshold. For example, if we have an *MP3 Player*, an *MP4 Player*, a *Mouse* and a *Keyboard* and the frequency threshold is 2 of 4 items, then the *MP3 Player* is a frequent similar pattern because the *MP4 Player* includes the functionalities of the *MP3 Player*. Therefore the *MP4 Player* can be considered similar to the *MP3 Player* in a certain sense. Consequently, 2 of the 4 items are similar to an *MP3 Player*. Notice that in this example the similarity is not symmetric because the *MP3 Player* does not include all functionalities of the *MP4 Player* and then, the *MP4 Player* is not a frequent similar pattern.

Another more complex example is the following. Given the dataset described by numerical and not numerical features (Mixed Data) shown in Table 1, assuming 0.6 as frequency threshold and using the traditional frequent pattern definition, the only frequent combination of feature values is $(Married = No)$, which appears 4 times in the 6 objects of the dataset.

However, if we use the frequent similar pattern definition considering that

Table 1 Example of a mixed dataset

Ω	Age (A)	Car (C)	Married (M)
O_1	23	$Compact$	No
O_2	25	Big	No
O_3	25	$Medium$	No
O_4	29	$Medium$	No
O_5	34	Big	Yes
O_6	38	$Fancy$	Yes

Table 2 Frequent similar patterns

Patterns	Frequency
$(Age = 25)$	0.66
$(Age = 29)$	0.66
$(Car = Medium)$	0.83
$(Car = Big)$	0.83
$(Married = No)$	0.66
$(Age = 25, Car = Medium)$	0.66
$(Age = 29, Car = Medium)$	0.66
$(Age = 25, Married = No)$	0.66
$(Car = Medium, Married = No)$	0.66
$(Age = 25, Car = Medium, Married = No)$	0.66

– two ages are similar if the absolute value of their difference is at most 5 years;
– compact cars are similar to medium cars, medium cars are similar to compact cars and big cars; big cars are similar to medium cars and fancy cars, and fancy cars are similar to big cars.

then the frequent similar patterns in the dataset of Table 1 and their frequencies would be those shown in Table 2.

As it can be appreciated, the use of different similarity functions (between feature values and object descriptions) produces patterns which are hidden for algorithms that assume that two object descriptions are similar when they are equal.

On the other hand, the cardinality of the search space for finding frequent similar patterns is exponential with respect to the number of features used for describing objects. In order to prune this space, a property named *Downward Closure* [21] has been applied. It says, in the context of frequent similar patterns, that all superdescriptions of a non-frequent similar pattern are also non-frequent similar patterns (Property 2.1).

For mining frequent similar patterns using Boolean (symmetric and non-symmetric) similarity functions that satisfy the *Downward Closure property*, two algorithms *ObjectMiner* [6] and *STreeDC-Miner* [21] have been proposed. On the other hand, for mining frequent similar patterns using Boolean (symmetric and non-symmetric) similarity functions that do not hold the *Downward Closure property*, the *STreeNDC-Miner* algorithm, which makes an exhaustive search [21], has also been proposed.

In this paper, we propose a frequent similar pattern mining algorithm for Boolean similarity functions that do not hold the *Downward Closure property*, which unlike *STreeNDC-Miner*, prunes the search space. Additionally, the proposed algorithm finds some similar patterns that are not obtained either by *ObjectMiner* or *STreeDC-Miner*.

The outline of this paper is as follows. Section 2 is dedicated to providing basic concepts. In Sect. 3, related works are reviewed. Section 4 describes the proposed algorithm. Finally, in Sects. 5 and 6, the experimental results and conclusions are exposed.

2 Basic concepts

Let $\Omega = \{O_1, O_2, \ldots, O_n\}$ be a dataset. Each object is described by a set of features $R = \{r_1, r_2, \ldots, r_m\}$ and represented as a tuple $(v_1, v_2, \ldots, v_m)$ where $v_i \in D_i$ (D_i is the domain of r_i, $1 \leq i \leq m$). A *subdescription* of an object O for a subset of features $S \subseteq R$ denoted as $I_S(O)$, is the description of O in terms of the features in S. $O[r]$ denotes the value of O on the feature $r \in R$. Each subset of features, $S \subseteq R$, $S \neq \emptyset$, has associated a Boolean *similarity function* f_S between subdescriptions of objects[1] [17]. Given two subdescriptions $I_S(O)$, $I_S(O')$, with $O, O' \in \Omega$, $f_S(O, O') = 1$ means that O is similar to O' with respect to S and $f_S(O, O') = 0$ means that O is not similar to O' with respect to S. Two examples of Boolean similarity functions are as follows:

$$f_S(O, O') = \begin{cases} 1 & \text{if } \forall r \in S | C_r(O[r], O'[r]) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$f_S(O, O') = \begin{cases} 1 & \text{if } \frac{|\{r \in S | C_r(O[r], O'[r]) = 1\}|}{|S|} \geq \alpha \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $C_r : D_r \times D_r \to \{0, 1\}$ is a comparison function between values of feature r. Two examples of comparison functions are as follows:

$$C_r(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$C_r(x, y) = \begin{cases} 1 & \text{if } |x - y| \leq \varepsilon \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Notice that the use of similarity function (1) with the comparison function (3) produces the equality function. However, the use of similarity functions (1) or (2), combining the comparison function (3) with at least one comparison function (4) setting $\varepsilon \neq 0$, produces similarity functions different from the equality.

Let $I_S(O)$ be a subdescription, $O \in \Omega$, $S \subseteq R$, $S \neq \emptyset$, and f_S be a Boolean similarity function; then the *frequency* of $I_S(O)$ in Ω for f_S is defined as:

$$f_S\text{-}freq(O) = \frac{|\{O' \in \Omega \,|\, f_S(O, O') = 1\}|}{|\Omega|} \tag{5}$$

A subdescription $I_S(O)$ is a *f_S-frequent subdescription* in Ω, also called *frequent similar pattern*, if its frequency $f_S\text{-}freq(O)$ is no less than a threshold $minFreq$ [21].

Given a dataset Ω, a Boolean similarity function f_S and a frequency threshold $minFreq$, the frequent similar pattern mining problem on mixed data using f_S, consists in finding all frequent similar patterns in Ω.

The Downward Closure property [21], in the context of frequent similar patterns, is defined as follows:

Property 2.1 (f_S-Downward Closure) *Let f_S be a similarity function, then f_S holds the Downward Closure property iff for all S_1, S_2, O, such that, $S_1 \subseteq S_2 \subseteq R$, $S_1 \neq \emptyset$, $O \in \Omega$:*

$$[f_{S_1}\text{-}freq(O) < minFreq] \Rightarrow [f_{S_2}\text{-}freq(O) < minFreq]$$

As we mentioned before, the f_S-Downward Closure property has been used for pruning the search space during the frequent similar pattern mining. However, not all similarity functions hold this property. For example, the similarity function (2) with $\alpha \neq 1$ and using the comparison function (3) does not hold the f_S-Downward Closure property.

We say that a similarity function f_S is non-increasing with respect to the cardinality of S iff for all $S_1 \subseteq S_2 \subseteq R$, $S_1 \neq \emptyset$, $O, O' \in \Omega$ $f_{S_1}(O, O') \geq f_{S_2}(O, O')$. Based on this we have:

[1] $f_S(O, O')$ denotes the similarity between O and O' using their subdescriptions $I_S(O)$ and $I_S(O')$.

Proposition 2.1 *If f_S is a non-increasing similarity function with respect to the cardinality of S, then f_S holds the f_S-Downward Closure property.*

Proof If for all S_1, S_2, O, O' such that $S_1 \subseteq S_2 \subseteq R$, $S_1 \neq \emptyset$, $O, O' \in \Omega$, $f_{S_1}(O, O') \geq f_{S_2}(O, O')$ then:

$$|\{O' \in \Omega \,|\, f_{S_1}(O, O') = 1\}| \geq |\{O' \in \Omega \,|\, f_{S_2}(O, O') = 1\}|$$
$$\frac{|\{O' \in \Omega \,|\, f_{S_1}(O, O') = 1\}|}{|\Omega|} \geq \frac{|\{O' \in \Omega \,|\, f_{S_2}(O, O') = 1\}|}{|\Omega|}$$
$$f_{S_1}\text{-}freq(O) \geq f_{S_2}\text{-}freq(O)$$

□

A subdescription $I_S(O)$ is considered a f_S-*interesting pattern* if $I_S(O)$ is a f_S-frequent subdescription or it contributes to the frequency of a f_S-frequent subdescription $I_S(O')$ (i.e., $f_S(O', O) = 1$).

Notice that if f_S holds the f_S-Downward Closure property and $I_S(O)$ is a non f_S-interesting pattern, then all superdescriptions of $I_S(O)$ are non f_S-interesting patterns too. Therefore, $I_S(O)$ and all its superdescriptions should not be considered for computing the frequency of frequent similar patterns. On the other hand, if f_S holds the f_S-Downward Closure property, but $I_S(O)$ is a f_S-interesting pattern, then its superdescriptions can contribute to the frequency of other patterns.

Two related processes for building frequent similar pattern candidates are the *combination* and *expansion* procedures. The *combination procedure* consists in obtaining a subdescription with $k+1$ features from two subdescriptions with k features, such that they share $k-1$ feature values. The *expansion procedure* consists in obtaining a subdescription with $k+1$ features from one subdescription with k features, by adding a feature value, such that the added feature is posterior to all features of the subset of features, taking into account a predefined order in the set of features R.

3 Related works

As we mentioned in the introduction, there are three algorithms for mining frequent similar patterns, all of them for Boolean similarity functions. *ObjectMiner* [6] and *STreeDC-Miner* [21] assume that the similarity function holds the Downward Closure property; and *STreeNDC-Miner* [21] does not assume it.

ObjectMiner was inspired in the Apriori Algorithm [3]. It works following a breadth first search strategy: first, for each feature, all the frequent similar values (frequent similar subdescriptions with only one feature), are determined. Afterward, for each pair (P_i, P_j) of frequent similar subdescriptions with $k-1$ features, found in the iteration $k-1$, such that P_i and P_j have a common subdescription with $k-2$ features, they are combined in order to create a new candidate subdescription with k features (see Fig. 1). In this step, for each pair (P_i, P_j) of frequent similar subdescriptions with $k-1$ features and a common subdescription with $k-2$ features, the next process is done:

- The combination P^* from P_i and P_j is obtained.
- The set of candidates to be similar to P^* is obtained by means of intersecting the set of subdescriptions similar to P_i and the set of subdescriptions similar to P_j.
- From these candidates, the set of subdescriptions similar to P^* is obtained.

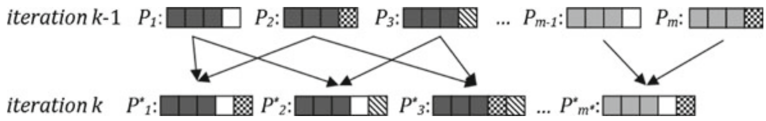

Fig. 1 Combination of subdescriptions with $k-1$ features into subdescriptions with k features

- The frequency of P^* is computed.
- If the frequency of P^* is equal or greater than $minFreq$ then P^* is a frequent similar subdescription.

This process finishes when a step does not produce any frequent similar subdescription with k features.

The main weakness of *ObjectMiner* is that although descriptions or subdescriptions of objects are usually repeated in the datasets, it does not use this fact in order to reduce the number of operations on subsequent steps. For this reason, the similarity between repetitions of the same subdescription are computed, causing an additional and unnecessary computational effort. Also storing the set of similar subdescriptions (including its repetitions) for each frequent subdescription affects *ObjectMiner* when it processes datasets with many objects.

STreeDC-Miner was focused on solving the main weakness of *ObjectMiner*. In order to do that *STreeDC-Miner* assumes an order between the features that describe the objects, and works following a depth first search strategy: first, it determines all the frequent similar values for each feature. In order to search for the frequent similar values for each feature (in general, the frequent similar patterns for a set of features *S*), the feature values are added to a structure called $STree$, in which equal subdescriptions are grouped. Thus, for computing the similarity between subdescriptions, only one subdescription for each group is considered, which reduces the number of similarity function evaluations. Afterward, through a recursive procedure, the frequent similar patterns obtained for each subset of features are expanded to frequent similar pattern candidates by adding a feature value, such that the added feature is posterior to all features of the subset of features, taking into account the feature order. The frequent similar pattern candidates and the expansions of the non- frequent similar patterns that are interesting patterns, are added to a $STree$ structure. Their similarities are computed from the similarities of the subdescriptions they come from. Thus, in the similarity computation, the similarity between two subdescriptions are only computed if the patterns they come from are similar. Also the frequency of all frequent similar pattern candidates is computed.

The structure called $STree$ (Fig. 2), is a tree in which each path from the root to a leaf represents a subdescription P. Each leaf contains: $P.objs$, the list of objects having the same subdescription that P; $P.similars$, the list of subdescriptions to which P is similar; $P.c_{=}$, the number of occurrences of the subdescription P in the dataset (length of *P.objs*); and $P.c_{\approx}$ the number of subdescriptions, which are similar to P, in the dataset (if the similarity function is symmetric then $P.c_{\approx} = |P.similars|$).

STreeDC-Miner, unlike *ObjectMiner*, reduces the number of times the similarity function is evaluated for repetitions of the same subdescription and it needs less memory than *ObjectMiner*. Nevertheless, both *STreeDC-Miner* and *ObjectMiner*, assume that the similarity function holds the Downward Closure property. For this reason, when the similarity function does not hold the Downward Closure property, both algorithms improperly prune the search space and lose many frequent similar patterns.

For mining frequent similar patterns, using Boolean similarity functions that do not hold the Downward Closure property, the *STreeNDC-Miner* [21] algorithm was proposed. *STreeNDC-Miner* does not prune the search space, it finds the similar frequent patterns for

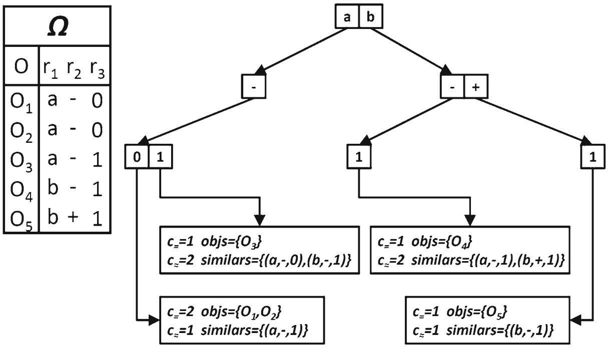

Fig. 2 Example of $STree_{\{r_1,r_2,r_3\}}$ for the dataset $\Omega = \{O_1, O_2, O_3, O_4, O_5\}$, the similarity function (2) with $\alpha = 0.5$ and the comparison function (3) for each feature

all $S \subseteq R$, $S \neq \emptyset$. However, the computational effort for searching frequent similar patterns is reduced using a top down strategy together with the $STree$ structure.

STreeNDC-Miner, like *STreeDC-Miner*, assumes an order among the features that describe the objects, and follows a depth first search strategy. But, unlike *STreeDC-Miner*, in order to search for frequent similar patterns, all reductions of R, by means of consecutive direct reductions, are obtained, where a direct reduction of a subset of features consists in removing a feature, such that the removed feature is posterior to all features already removed.

First, all objects are added to a $STree$ structure; their similarities and their frequencies are computed; and frequent similar patterns involving all the features are obtained. Later, for each reduced subset of R, the subdescriptions stored in the $STree$ structure corresponding to the subset from which the reduced subset comes from, are added to a new $STree$ structure, their similarities and their frequencies are computed; and the frequent similar patterns for the reduction are obtained too. As a consequence of the top down strategy, if two subdescriptions are equal for a subset of features S, then they are added to the $STree$ structure. Also, in the similarity computation, like *STreeDC-Miner*, only one subdescription by group is considered, which reduces the number of similarity function evaluations.

The main weakness of *STreeNDC-Miner* is that it does an exhaustive search of frequent similar patterns, which requires runtimes longer than the previous algorithms.

4 RP-Miner algorithm

If a similarity function fulfills the f_S-Downward Closure property, then combinations (on *ObjectMiner*) or expansions (on *STreeDC-Miner*) of non f_S-frequent subdescriptions for finding f_S-frequent subdescriptions have no sense. Therefore, the search space can be pruned. However, when the f_S-Downward Closure property is not held, then some superdescriptions of non- f_S-frequent subdescriptions could be f_S-frequent subdescriptions. To face this problem, we introduce a new algorithm that unlike *ObjectMiner*, which only combines f_S-frequent subdescriptions, and *STreeDC-Miner*, which adds to a f_S-frequent subdescription any other feature values following a predefined order; adds to a f_S-frequent subdescription any other feature value (without order), while the obtained subdescription has not been previously analyzed.

In Fig. 3, a dataset $\Omega = \{O_1, O_2, O_3, O_4, O_5\}$, the search space for this dataset and the frequency $f_S\text{-}freq$ of each subdescription, are showed. In order to compute $f_S\text{-}freq$, the

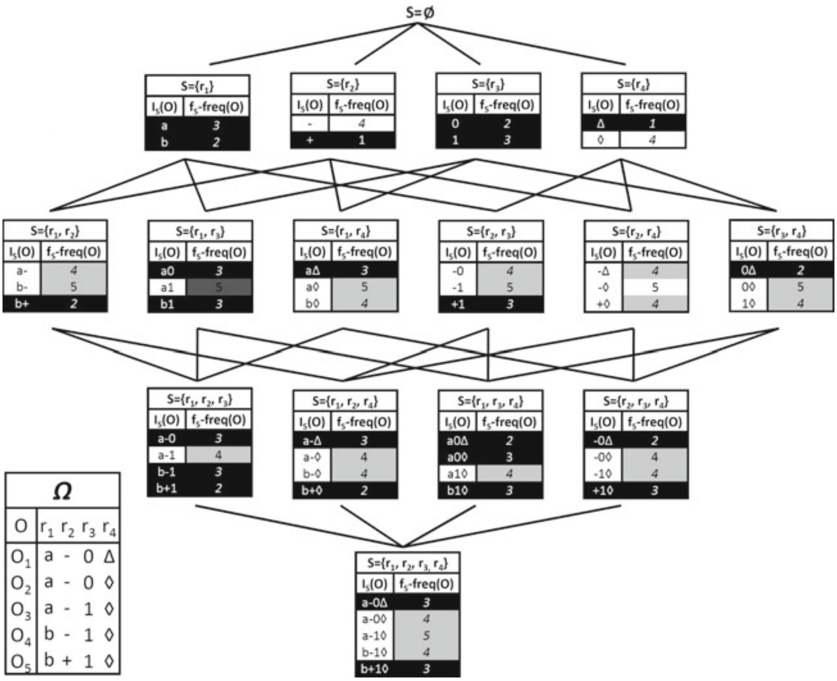

Fig. 3 Search space for the dataset $\Omega = \{O_1, O_2, O_3, O_4, O_5\}$

similarity function (2) with $\alpha = 0.5$ (that does not hold the f_S-Downward Closure property), and the comparison function (3) for each feature, were used. Considering $minFreq = 0.8$, in Fig. 3 the non-frequent similar patterns appear in black cells and the frequent similar patterns appear in white cells. For each frequent similar pattern $I_S(O)$, the color of the cell, in which its f_S-$freq$ value appears represents:

White. All subdescriptions $I_{S'}(O)$, such that $S' \subset S$, are frequent similar patterns too.

Light Gray. Not all subdescriptions $I_{S'}(O)$, such that $S' \subset S$, are frequent similar patterns, but there is at least one frequent similar pattern $I_{S'}(O)$, such that $S' \subset S$, $|S'| = |S| - 1$.

Dark Gray. Neither all subdescriptions $I_{S'}(O)$, such that $S' \subset S$, are frequent similar patterns nor there is at least a frequent similar pattern $I_{S'}(O)$, such that $S' \subset S$, $|S'| = |S| - 1$.

In this example, there are only 3 frequent similar patterns with white frequency cells, and there are 19 frequent similar patterns with light gray frequency cells and only one frequent similar pattern, with dark gray frequency cell.

The algorithms that explore and prune the search space using the f_S-Downward Closure property (*ObjectMiner* and *STreeDC-Miner*) would only find those frequent similar patterns with white frequency cells. On the other hand, since the non- frequent similar patterns that are f_S-interesting patterns are not pruned, *STreeDC-Miner* would find some of the frequent similar patterns with light gray frequency cells.

On the other hand, for each frequent similar pattern $I_S(O)$ with light gray frequency cell, there is at least one sequence $S_1, S_2, \ldots, S_k$, such that $S_k = S$; for all $i < k$, $S_i \subset S_{i+1}$; $|S_{i+1}| = |S_i| + 1$; and $I_{S_i}(O)$ is a frequent similar pattern with white or light gray frequency cell. Therefore, each frequent similar pattern $I_S(O)$ can be constructed by expanding another frequent similar pattern $I_{S'}(O)$, such that $S' \subseteq S$ and $|S'| = |S| - 1$, by adding one feature value.

4.1 Relaxed prune

In this section, we introduce a new expansion process for searching frequent similar patterns, starting from the frequent similar patterns with just one feature. The expanded patterns of each similar pattern $I_S(O)$ are obtained adding a feature $r \in R$ to S.

Obviously, there could exist more than one sequence $S_1, S_2, \ldots, S_k$ for building a pattern $I_{S_k}(O)$. Therefore, each pattern could be obtained by the expansion of several frequent similar patterns. For example, in Fig. 3, pattern (a, -, 0, $\Diamond$) can be obtained by expanding the patterns (-, 0, $\Diamond$), (a, 0, $\Diamond$), (a, -, $\Diamond$), and (a, -, 0).

However, if an expanded pattern has already been analyzed (to verify if it is a frequent similar pattern), and it is obtained again through another way, then it would be neither analyzed again nor expanded. Only, expanded patterns, that have not been analyzed, are considered as candidates to be frequent similar patterns.

In this process, the similarity between two expanded patterns is only computed if the patterns they were obtained from are similar.

The expansion process for the example of Fig. 3 is showed in Fig. 4. As it can be seen, none of the non-frequent similar patterns were expanded. However, each expansion $I_{\hat{S}}(O)$ of a non-frequent similar pattern $I_S(O)$ can be obtained through another way, if there is at least a frequent similar pattern $I_{S'}(O)$ such that $S' \subset \hat{S}$ and $|S'| + 1 = |\hat{S}|$. For example, the non-frequent similar pattern (a) is not expanded, but its expansion (a, -) is obtained when the frequent similar pattern (-) is expanded.

Following the expansion process described before, unlike *ObjectMiner* and *STreeDC-Miner* which prune all non-frequent similar patterns, only the non-frequent similar patterns $I_S(O)$ where there is no sequence $S_1, S_2, \ldots, S_k$, such that $S_k = S$; for all $i < k$, $S_i \subset S_{i+1}$; $|S_{i+1}| = |S_i| + 1$; and $I_{S_i}(O)$ is a frequent similar pattern, are pruned. We call this procedure *Relaxed Prune*.

4.2 Proposed algorithm

In this section, we introduce a new algorithm for mining frequent similar patterns using a similarity function that does not hold the Downward Closure property based on the relaxed prune introduced in Sect. 4.1. We call this algorithm *RP-Miner* (Algorithm 1).

The idea of this algorithm is the following: First, all single frequent similar feature values are calculated. Then, each obtained frequent similar pattern is successively expanded while it has not been previously analyzed.

At the beginning, the set of analyzed patterns (W), the set of frequent similar patterns ($frequents$), and the expanded set of features ($\hat{S}$) are empty sets; and $STree_S$ is *null*.

In order to search for the frequent similar patterns from each $\hat{S}$, we use an $STree_{\hat{S}}$ structure.

According to the size of $\hat{S}$, we distinguish three cases:

1. $\hat{S} = \emptyset$. The algorithm is recursively called for each $r \in R$ (lines 27–29)
2. $|\hat{S}| = 1$. All the objects in Ω are added to $STree_{\hat{S}}$ (lines 4–7). After that, the similarities between all subdescriptions in $STree_{\hat{S}}$ are computed, and for each subdescription P, the list $P.similars$ is updated (lines 8–10). Later, for each

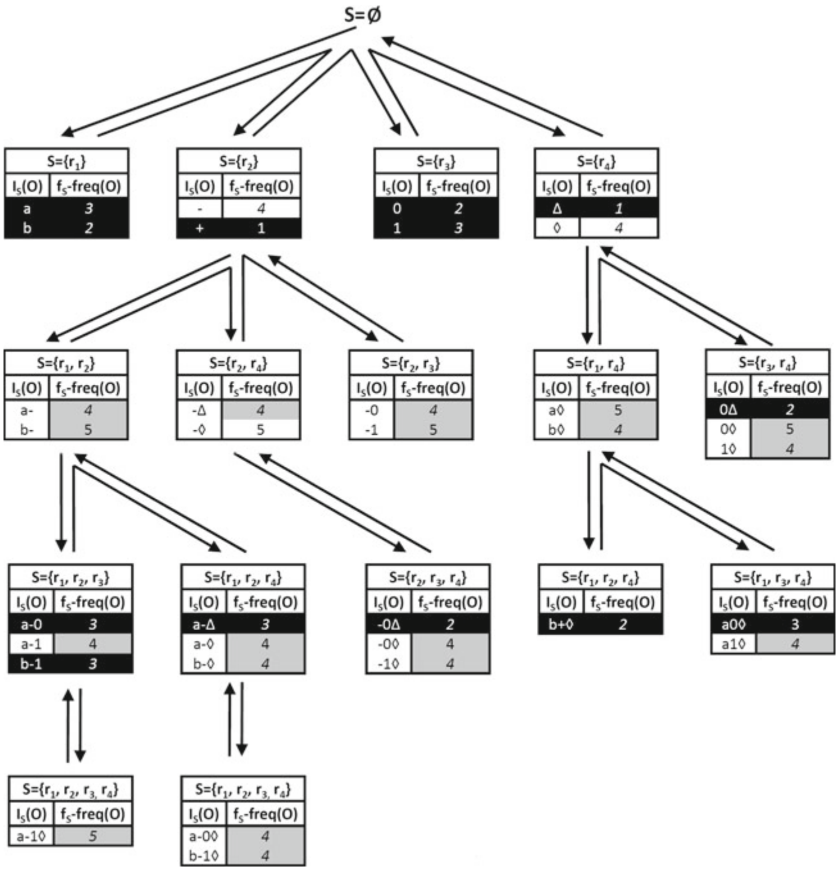

Fig. 4 Expansion process for mining frequent similar patterns on the dataset showed in Fig. 3

subdescription P in $STree_{\hat{S}}$, $P.c_{\approx}$ is computed; the frequent similar patterns in $STree_{\hat{S}}$ are also computed and the set of frequent similar patterns is updated (lines 22–26). Finally, if the set of frequent similar patterns in $STree_{\hat{S}}$ is not empty, the algorithm is recursively called for adding each $r \in R$ to $\hat{S}$ (lines 27–29)

3. $|\hat{S}| > 1$. For each f_S-*interesting pattern* P in $STree_S$, the objects in $P.objs$, which have not been analyzed, are added to $STree_{\hat{S}}$ and they are added to the set of analyzed patterns (lines 12–18). After that, only the similarity values between all subdescriptions in $STree_{\hat{S}}$, such that their similarities regarding S are different from zero, are computed. For each subdescription P, the list $P.similars$ is updated (lines 19–21). This reduces the number of similarity function evaluations. Later, for each subdescription P in $STree_{\hat{S}}$, $P.c_{\approx}$ is computed; the frequent similar patterns in $STree_{\hat{S}}$ are also computed; and the set of frequent similar patterns is updated (lines 22–26). Finally, if the set of frequent similar patterns in $STree_{\hat{S}}$ is not empty, the algorithm is recursively called for adding each $r \in R$ to $\hat{S}$ (lines 27–29).

Algorithm 1: $RP\text{-}Miner(STrees_S, \hat{S})$

```
if Ŝ ≠ ∅ then
    STree_Ŝ ←empty STree structure
    if |Ŝ| = 1 then
        foreach O ∈ Ω do
            if ¬STree_Ŝ.contain(I_Ŝ(O)) then
                STree_Ŝ.add(O)
            STree_Ŝ.I_Ŝ(O).c_= ← STree_Ŝ.I_Ŝ(O).c_= + 1
        foreach P, P′ ∈ STree_Ŝ do
            if f_Ŝ(P, P′) = 1 then
                P′.similars ← P′.similars ∪ P
    else
        foreach P ∈ STrees_S such that P is f_S-interesting do
            foreach O ∈ P.objs do
                if I_Ŝ(O) ∉ W then
                    if ¬STree_Ŝ.contain(I_Ŝ(O)) then
                        STree_Ŝ.add(O)
                        W ← W ∪ I_Ŝ(O)
                    STree_Ŝ.I_Ŝ(O).c_= ← STree_Ŝ.I_Ŝ(O).c_= + 1
        foreach P, P′ ∈ STree_Ŝ such that I_S(P) ∈ I_S(P′).similars do
            if f_Ŝ(P, P′) = 1 then
                P′.similars ← P′.similars ∪ P
    foreach P ∈ STree_Ŝ do
        foreach P′ ∈ P.similars such that P′ ≠ P do
            P′.c_≈ ← P′.c_≈ + P.c_=
    localFrequents ← {P ∈ STree_Ŝ : P.c_= + P.c_≈ ≥ minFreq}
    frequents ← frequents ∪ localFrequents
if Ŝ = ∅ or localFrequents ≠ ∅ then
    foreach r ∈ R − Ŝ do
        RP-Miner(STree_Ŝ, Ŝ ∪ {r})
```

5 Experimental results

In this section, we compare the proposed algorithm *RP-Miner* (RPM) against *ObjectMiner* (ObjMiner), *STreeDC-Miner* (STDC), and *STreeNDC-Miner* (STNDC) algorithms. We divided the experimental results in two sections. In the first section, a comparison in terms of the time needed to mine the frequent similar patterns and the number of frequent similar patterns mined by each algorithm was done. In the second section, we compare the quality of the set of frequent similar patterns obtained by each algorithm.

5.1 Efficiency and effectiveness of the algorithms

Since the frequent similar pattern mining problem consists in finding all frequent similar patterns, the effectiveness of an algorithm for frequent pattern mining is measured as the number of frequent similar patterns mined by it, whereas the efficiency is measured as the

Table 3 Description of the datasets used in the experiments

Dataset	Objects	Numerical features	Non numerical features
Car Evaluation	1728	2	5
Contraceptive Method Choice	1473	2	8
Census	32561	3	9
Poker Hand	1000000	5	6

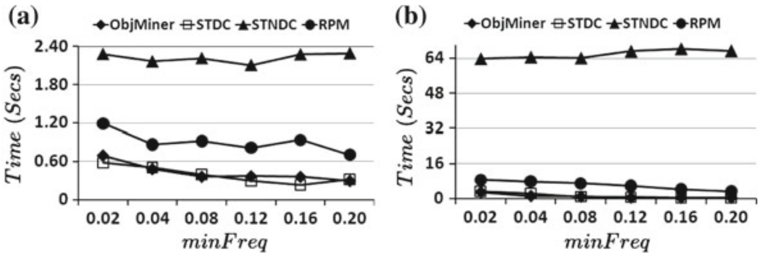

Fig. 5 Runtime for **a** *Car Evaluation* and **b** *Contraceptive Method Choice*

runtime spent for mining the frequent similar patterns. Additionally, some other measures like the number of similarity evaluations, as well as the ratio between the number of frequent similar patterns and the runtime, are also used. The comparison using each measure was done for different values of the $minFreq$ threshold. Table 3 gives a description of the datasets[2] used in our experiments. We consider that *Car Evaluation* and *Contraceptive Method Choice* datasets are not large datasets, but *Census* and *Poker Hand* are large datasets. First, we describe the results for non-large datasets and later for large datasets.

For the experiments, we used the similarity function (2) with $\alpha = 0.7$ for all datasets, which does not satisfy the f_S-Downward Closure property.

For *Car Evaluation*, we used the comparison function (4) with $\varepsilon = 2$ for features *Doors* and *Persons*, respectively; and for the remaining features, we used the function (3). For *Contraceptive Method Choice* and *Census*, we used the comparison function (4) with $\varepsilon = 5$ for feature *Age* and for the remaining features we used the function (3). For *Poker Hand*, we used the comparison function (3) for all features.

In Fig. 5a, b, the runtime of the algorithms for *Car Evaluation* and *Contraceptive Method Choice* are shown.

As we expected, in both datasets, the runtime of *RP-Miner* was longer than the runtime of *STreeDC-Miner* and *ObjectMiner* but it was shorter than the runtime of *STreeNDC-Miner*. This is a consequence of the number of frequent similar patterns found (see Fig. 7a, b) by *RP-Miner* and the number of similarity function evaluations computed (see Fig. 6a, b) by *RP-Miner*.

The number of frequent similar patterns found for *Car Evaluation* and *Contraceptive Method Choice* are shown in Fig. 7a, b. It is worthwhile to underline that *STreeNDC-Miner* finds all frequent similar patterns, while *ObjectMiner* and *STreeDC-Miner*, which assume that f_S fulfills the f_S-Downward Closure property, can do not find all frequent similar patterns. Also *RP-Miner* can does not find all frequent similar patterns. However, the use of the

2 http://archive.ics.uci.edu/ml/datasets.html.

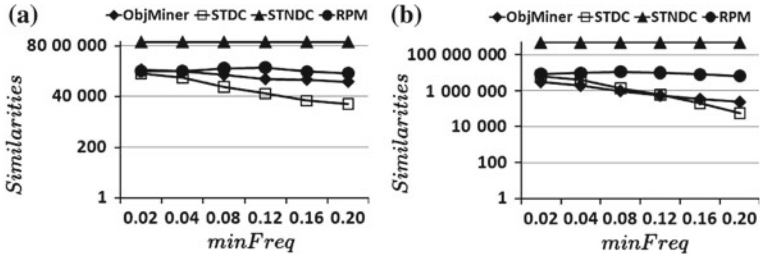

Fig. 6 Number of similarity function evaluations for **a** *Car Evaluation* and **b** *Contraceptive Method Choice*

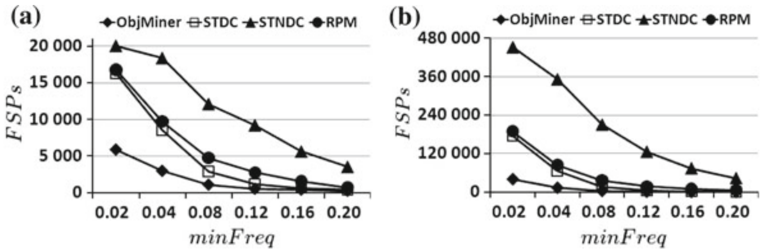

Fig. 7 Number of frequent similar patterns for **a** *Car Evaluation* and **b** *Contraceptive Method Choice*

relaxed prune allows *RP-Miner* to find many frequent similar patterns that *ObjectMiner* and *STreeDC-Miner* miss.

Notice that, in *Car Evaluation* for $minFreq = 0.02$, respect to the frequent similar patterns obtained by *STreeNDC-Miner*, *ObjectMiner* lost up to 14,168 (70.64%) frequent similar patterns and *STreeDC-Miner* lost up to 3,805 (18.97%) frequent similar patterns, while *RP-Miner* lost fewer frequent similar patterns (3,279, 16.35%) (see Fig. 7a). For the other values of $minFreq$, the number of frequent similar patterns lost by the algorithms is less, however, *RP-Miner* always lost fewer frequent similar patterns than *ObjectMiner* and *STreeDC-Miner*.

Analogously, in *Contraceptive Method Choice* for $minFreq = 0.02$, compared to the frequent similar patterns obtained by *STreeNDC-Miner*, *ObjectMiner* lost up to 412,977 (91.27%) frequent similar patterns and *STreeDC-Miner* lost up to 278,031 (61.45%) frequent similar patterns, while *RP-Miner* lost fewer frequent similar patterns (263,197, 58.17%) (see Fig. 7b). For the other values of $minFreq$, the number of frequent similar patterns lost by the algorithms is also less, however, *RP-Miner* always lost fewer frequent similar patterns than *ObjectMiner* and *STreeDC-Miner*.

Another relevant point is that, in both datasets *Car Evaluation* and *Contraceptive Method Choice*, the ratio between the number of frequent similar patterns and the runtime for our algorithm was greater (up to 1.6 and 3.1 times, respectively for $minFreq = 0.02$) than the ratio for *STreeNDC-Miner* (see Fig. 8a, b). In addition, our algorithm sometimes obtained values for this measure greater than those obtained by *ObjectMiner*.

The number of frequent similar patterns found for the large datasets *Poker Hand* and *Census*, are shown in Fig. 9a, b. These datasets contain much more objects than *Car Evaluation* and *Contraceptive Method Choice*. Although *STreeNDC-Miner* is the most effective algorithm (since it can find all frequent similar patterns), it is very slow because it performs an exhaustive search of frequent similar patterns. It is important to highlight that for *Poker Hand* and *Census* datasets using ($minFreq = 0.02$) *STreeNDC-Miner* was unable to finish after

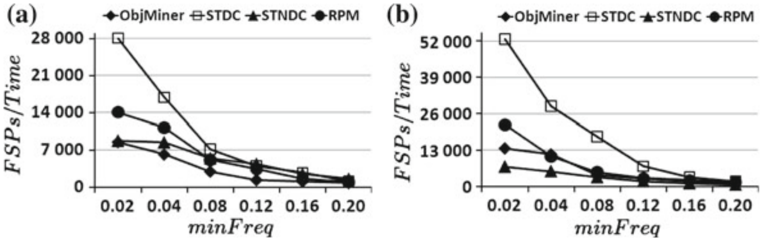

Fig. 8 Ratio between the number of frequent similar patterns and the runtime for **a** *Car Evaluation* and **b** *Contraceptive Method Choice*

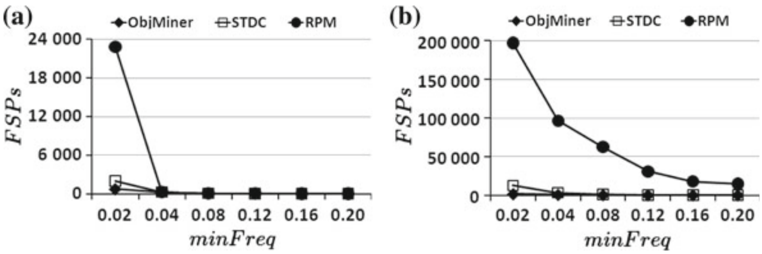

Fig. 9 Number of frequent similar patterns for **a** *Poker Hand* and **b** *Census*

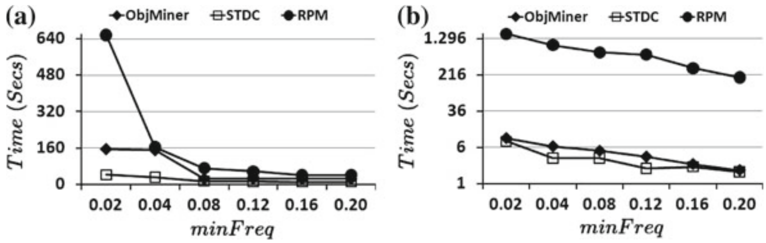

Fig. 10 Runtime for **a** *Poker Hand* and **b** *Census*

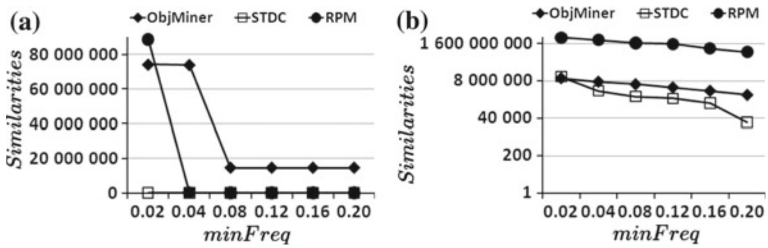

Fig. 11 Number of similarity function evaluations for **a** *Poker Hand* and **b** *Census*

10 days. For this reason, the *STreeNDC-Miner* results were not plotted and not compared with the other algorithms.

As in the previous datasets, in *Poker Hand* and *Census*, the runtime of *RP-Miner* was longer than the runtime of *ObjectMiner* and *STreeDC-Miner* (see Fig. 10a, b). Furthermore, this is a consequence of the number of frequent similar patterns found (see Fig. 9a, b) and the number of similarity function evaluations computed (see Fig. 11a, b) by *RP-Miner*.

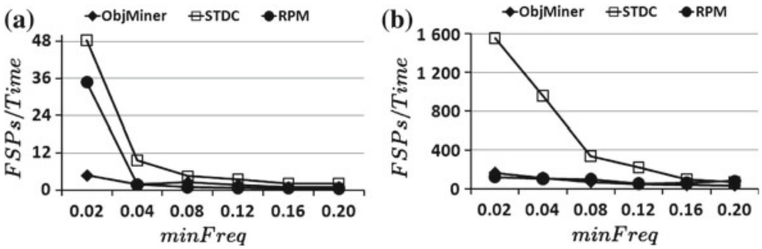

Fig. 12 Ratio between the number of frequent similar patterns and the runtime for **a** *Poker Hand* and **b** *Census*

However, in these datasets, the number of frequent similar patterns (see Fig. 9a, b) obtained by *RP-Miner* and lost by *ObjectMiner* and *STreeDC-Miner* was much bigger than the ones lost in the previous datasets.

It can be noticed that *ObjectMiner* and *STreeDC-Miner* lost more frequent similar patterns compared to the frequent similar patterns obtained by *RP-Miner* in *Poker Hand* for $minFreq = 0.02$ (see Fig. 9a). For the other values of $minFreq$, in this database the number of frequent similar patterns found by the algorithms was similar. Nevertheless, *RP-Miner* always lost fewer or the same frequent similar patterns than *ObjectMiner* and *STreeDC-Miner*.

Analogously, *ObjectMiner* and *STreeDC-Miner* lost more frequent similar patterns with respect to the frequent similar patterns obtained by *RP-Miner* in *Census* for $minFreq = 0.02$ (see Fig. 9b). For the other values of $minFreq$, like in previous dataset, the number of frequent similar patterns found by the algorithms was similar, and *RP-Miner* always lost fewer or the same frequent similar patterns than *ObjectMiner* and *STreeDC-Miner*.

In addition, in both datasets, the ratio between the number of frequent similar patterns and the runtime for our algorithm was sometimes greater than the ratio for *ObjectMiner* (Fig. 12a, b).

Based on these experiments, we can affirm that our algorithm is more efficient that *STreeNDC-Miner* and more effective than *ObjectMiner* and *STreeDC-Miner*.

Concerning the $minFreq$ threshold, we can see that for all the datasets when the $minFreq$ increases, the number of frequent similar patterns obtained by all the algorithms decreases, or remains equal (see Figs. 7, 9). Besides, small values of $minFreq$ favor *RP-Miner* if compared against *ObjectMiner* and *STreeDC-Miner*. Additionally, for *ObjectMiner*, *STreeDC-Miner* and *RP-Miner*, due to the prune, the number of candidates to frequent similar patterns and the number of similarity function evaluations tend to decrease when the $minFreq$ value increases, while for *STreeNDC-Miner*, due to the exhaustive search, the number of candidates to frequent similar patterns and the number of similarity function evaluations tend to remain similar when the $minFreq$ value increases.

5.2 Quality of the set of frequent similar patterns

Although the proposed algorithm (*RP-Miner*) obtains more frequent similar patterns than *ObjectMiner* and *STreeDC-Miner*, obtaining a bigger set of frequent similar patterns does not necessarily imply that this set is better. For this reason, in this section, we compare the quality of the set of frequent similar patterns obtained by each algorithm as the accuracy that a supervised classifier based on frequent similar patterns reaches when it uses this set of frequent similar patterns to classify unseen objects.

For this experiment, we used a simple supervised classifier based on frequent similar patterns, which in the training phase obtains the set of frequent similar patterns from each class and removes all the frequent similar patterns that appear in more than one class in order to keep only patterns that represent objects from a single class. In the classification phase, each object of the testing dataset is classified under the class where there are more frequent similar patterns similar to its subdescriptions. The accuracy is measured as the percentage of objects classified correctly.

For each dataset and for each value for the parameter $minFreq$, we repeated 10 times the experiment randomly selecting 50% of the dataset for training and using the remaining objects for testing. Since *STreeNDC-Miner* is not a feasible algorithm for large datasets, it is not used in this experiment for datasets *Poker Hand* and *Census*. Additionally to the datasets *Car Evaluation* and *Contraceptive Method Choice*, we included the no large datasets *Iris* and *Diabetes*, which have 150 and 768 objects and 4 and 8 numerical features, respectively.

For each dataset and for all the algorithms used for mining frequent similar patterns, the classification was done testing different values of the $minFreq$ threshold from $minFreq = 0.10$ until $minFreq = 0.20$ with steps of 0.01. In Tables 4 and 5, we show the accuracies achieved.

In the majority of these tests, the accuracy of the set of frequent similar patterns obtained with *RP-Miner* was better than the accuracy obtained by those patterns obtained through *STreeDC-Miner* and *ObjectMiner* respectively.

The last column in Tables 4 and 5 (*EQ-STDC*) contains the result of the *STreeDC-Miner* algorithm using the equality function as similarity function. From this column, it can be seen that using the equality function, as in the classical approach for frequent pattern mining, the classification accuracy obtained by the set of frequent patterns is usually lower than the classification accuracy obtained by the set of frequent similar patterns obtained using similarity functions different from the equality.

Regardless of the low accuracies in our experiments, in general, the classification accuracy reached through the frequent similar patterns found by our algorithm is better than the classification accuracy reached through the frequent similar patterns found by *ObjectMiner*, *STreeDC-Miner* and the classical approach. In fact, the best accuracies (gray cells in Tables 4, 5) for each dataset was achieved by our algorithm. Notice that the set of frequent similar patterns found by *RP-Miner* is a superset of the frequent similar patterns found by *ObjectMiner* and *STreeDC-Miner*. For this reason, we can affirm that the frequent similar patterns lost by *ObjectMiner* and *STreeDC-Miner* affect the classification accuracy, whereas those other additional patterns obtained by *RP-Miner* contribute to get better accuracies.

6 Conclusions

In this paper, we focused our attention on the problem of mining frequent similar patterns for Boolean similarity functions that do not satisfy the f_S-Downward Closure property. A Relaxed Prune for this kind of functions and a novel algorithm based on this new prune were proposed.

The experimental results have shown that the proposed algorithm, (*RP-Miner*), is more efficient than the *STreeNDC-Miner* algorithm and more effective than the *STreeDC-Miner* and *ObjectMiner* algorithms. Additionally, the quality of the set of frequent similar patterns found by *RP-Miner* was measured by means of a supervised classifier. From the experiments, we concluded that, in general, the proposed algorithm obtains better frequent similar patterns

Table 4 Accuracy of the sets of frequent similar patterns for *Car Evaluation*, *Contraceptive Method Choice*, *Iris*, *Diabetes*

Dataset	$minFreq$	*ObjMiner*	*STDC*	*STNDC*	*RP-Miner*	*EQ-STDC*
	0.10	64.86	65.96	51.01	66.27	65.43
	0.11	64.86	65.96	51.01	66.27	65.43
	0.12	64.86	65.96	51.01	66.27	65.43
	0.13	60.52	61.01	38.91	60.94	60.14
	0.14	60.52	61.01	38.91	60.94	60.14
Car Evaluation	0.15	60.52	61.01	38.91	60.94	60.14
	0.16	55.59	56.08	31.31	56.23	55.65
	0.17	55.59	56.08	31.31	56.23	55.65
	0.18	55.59	56.08	31.31	56.23	55.65
	0.19	55.39	55.77	28.61	55.71	55.39
	0.20	55.39	55.77	28.61	55.71	55.39
Max accuracies		64.86	65.96	51.01	**66.27**	65.43
	0.10	41.12	40.93	38.41	41.01	35.15
	0.11	41.36	41.29	37.19	40.84	34.79
	0.12	41.05	40.99	37.27	41.49	33.76
	0.13	41.02	40.69	35.70	39.98	33.23
	0.14	40.96	40.73	35.03	39.94	30.96
Contraceptive Method Choice	0.15	40.67	40.67	34.57	40.37	30.32
	0.16	40.76	40.34	34.08	40.25	29.65
	0.17	39.74	39.14	33.33	39.92	28.34
	0.18	40.36	39.58	33.02	40.27	28.63
	0.19	40.48	39.04	32.14	40.49	30.28
	0.20	41.03	39.41	32.14	40.37	30.23
Max accuracies		41.36	41.29	38.41	**41.49**	35.15
	0.10	92.93	93.33	93.33	93.47	66.52
	0.11	92.93	93.33	93.33	93.47	66.22
	0.12	92.93	93.33	93.33	93.47	65.74
	0.13	91.60	91.87	92.00	92.00	65.22
	0.14	91.60	91.87	92.00	92.00	65.19
Iris	0.15	91.60	91.87	92.00	92.00	64.55
	0.16	91.60	91.87	92.00	92.00	64.56
	0.17	89.87	90.00	90.40	90.23	63.91
	0.18	89.87	90.00	90.40	90.23	63.65
	0.19	89.87	90.00	90.40	90.23	63.86
	0.20	89.87	90.00	90.40	90.23	63.83
Max accuracies		92.93	93.33	93.33	**93.47**	66.52
	0.10	66.88	67.04	63.42	63.94	30.82
	0.11	66.66	66.58	63.28	63.86	28.80
	0.12	65.56	65.60	62.86	63.40	26.88
	0.13	66.12	65.90	63.20	63.50	26.04
	0.14	66.00	65.96	62.90	63.46	26.12
Diabetes	0.15	64.08	63.88	62.10	63.59	24.34

Table 4 continued

Dataset	*minFreq*	*ObjMiner*	*STDC*	*STNDC*	*RP-Miner*	*EQ-STDC*
	0.16	64.26	64.30	62.12	64.20	23.28
	0.17	64.46	64.48	62.28	65.08	22.26
	0.18	65.78	65.86	62.72	66.17	15.48
	0.19	66.68	66.64	63.66	66.80	14.30
	0.20	66.80	66.84	64.32	67.14	12.70
Max accuracies		66.88	67.04	64.32	**67.14**	30.82

Table 5 Accuracy of the sets of frequent similar patterns for *Poker Hand* and *Census*

Dataset	*minFreq*	*ObjMiner*	*STDC*	*RP-Miner*	*EQ-STDC*
	0.10	9.40	9.40	19.30	9.40
	0.11	8.18	8.18	18.43	8.18
	0.12	7.35	7.35	15.76	7.35
	0.13	8.91	8.91	14.97	8.91
	0.14	9.97	9.97	14.60	9.97
Poker Hand	0.15	12.79	12.79	14.33	12.79
	0.16	13.66	13.66	14.31	13.66
	0.17	14.16	14.16	14.28	14.16
	0.18	13.96	13.96	13.96	13.96
	0.19	13.96	13.96	13.96	13.96
	0.20	12.18	12.18	12.18	12.18
Max accuracies		14.16	14.16	**19.30**	14.16
	0.10	70, 20	72, 13	73, 47	70, 80
	0.11	70, 07	71, 33	73, 07	71, 40
	0.12	70, 07	71, 33	73, 07	71, 40
	0.13	69, 60	70, 80	72, 67	70, 87
	0.14	68, 47	70, 07	72, 60	70, 53
Census	0.15	66, 87	69, 73	72, 00	69, 80
	0.16	66, 87	69, 73	72, 00	69, 80
	0.17	66, 67	68, 07	71, 67	69, 60
	0.18	65, 00	67, 07	70, 40	69, 47
	0.19	63, 60	66, 40	70, 13	70, 87
	0.20	63, 60	66, 40	70, 13	70, 87
Max accuracies		70, 20	72, 13	**73, 47**	71, 40

than those obtained by *STreeDC-Miner*, *ObjectMiner*, *STreeNDC-Miner*, and the classical approach.

In future works, reducing the set of frequent similar patterns mined without losing information would be an interesting and imperative research direction. Another interesting research direction is mining frequent similar patterns for non-Boolean similarity functions.

Acknowledgments This work is partly supported by the National Council of Science and Technology of Mexico under the project CB-2008-01-106443 and grant 32086. The authors would like to thank the anonymous reviewers for their valuable suggestions and comments.

References

1. Agrawal R, Imielinski T, Swami A (1993) Mining association rules between sets of items in large databases. In: Proceedings of the 1993 ACM SIGMOD international conference on management of data, pp 207–216
2. Agrawal R, Gehrke J, Gunopulos D, Raghavan P (1998) Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of the 1998 ACM-SIGMOD international conference management of data, pp 94–105
3. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th international conference on very large data bases, pp 487–499
4. Agrawal R, Srikant R (1995) Mining sequential patterns. In: Proceedings of the 1995 international conference on data engineering, pp 3–14
5. Cheng J, Ke Y, Ng W (2008) A survey on algorithms for mining frequent itemsets over data streams. Knowl Inf Syst 16:1–27
6. Dánger R, Ruiz-Shulcloper J, Berlanga R (2004) Objectminer: a new approach for mining complex objects. In: Proceedings of the sixth international conference on enterprise information systems, pp 42–47
7. Gómez J, Rodríguez O, Valladares S, Ruiz-Shulcloper J et al (1994) Prognostic of gas-oil deposits in the Cuban Ophiological Association. Applying mathematical modeling. Geophys Int 33:447–467
8. Han J, Cheng H, Xin D, Yan X (2007) Frequent pattern mining: current status and future directions. Data Min Knowl Discov 15:55–86
9. Han J, Dong G, Yin Y (1999) Efficient mining of partial periodic patterns in time series database. In: Proceedings of the 1999 international conference data on engineering, pp 106–115
10. Iváncsy R, Vajk I (2006) Frequent pattern mining in web log data. Acta Polytechnica Hungarica. J Appl Sci Bp 1:77–90
11. Kelil A, Wang S, Jiang Q, Brzezinski R (2009) A general measure of similarity for categorical sequences. Knowl Inf Syst. doi:10.1007/s10115-009-0237-8
12. LaRosa C, Xiong L, Mandelberg K (2008) Frequent pattern mining for kernel trace data. In: Proceedings of the 2008 ACM symposium on applied computing, pp 880–885
13. Li J, Fu AW, Fahey P (2009) Efficient discovery of risk patterns in medical data. Artif Intell Med 45:77–89
14. Liu B, Hsu W, Ma Y (1998) Integrating classification and association rule mining. In: Proceedings of the 1998 international conference on knowledge discovery and data mining, pp 80–86
15. Lopez FJ, Blanco A, Garcia F, Cano C, Marin A (2008) Fuzzy association rules for biological data analysis: a case study on yeast. BMC Bioinform 9:107
16. Mannila H, Toivonen H, Verkamo AI (1997) Discovery of frequent episodes in event sequences. Data Min Knowl Discov 1:259–289
17. Martínez-Trinidad JF, Ruiz-Shulcloper J, Lazo-Cortés MS (2000) Structuralization of universes. Fuzzy Sets Syst 112:485–500
18. Ortiz-Posadas MR, Vega-Alvarado L, Toni B (2004) A similarity function to evaluate the orthodontic condition in patients with cleft lip and palate. Med Hypotheses 63:35–41
19. Ortiz-Posadas MR, Vega-Alvarado L, Toni B (2009) A mathematical function to evaluate surgical complexity of cleft lip and palate. Comput Methods Prog Biomed 94:232–238
20. Quan X, Liu G, Lu Z, Ni X, Wenyin L (2009) Short text similarity based on probabilistic topics. Knowl Inf Syst. doi:10.1007/s10115-009-0250-y
21. Rodríguez-González AY, Martínez-Trinidad JF, Carrasco-Ochoa JA, Ruiz-Shulcloper J (2008) Mining frequent similar patterns on mixed data. In: Ruiz-Shulcloper J, Kropatsch W (ed) Progress in pattern recognition, image analysis and applications, LNCS 5197, Springer, Berlin, pp 136–144
22. Ruiz-Shulcloper J, Fuentes-Rodrguez A (1981) A cybernetic model to analyze juvenile delinquency. Revista Ciencias Matemáticas 2:123–153
23. Silverstein C, Brin S, Motwani R, Ullman J (1998) Scalable techniques for mining causal structures. In: Proceedings of the 1998 international conference on very large data bases, pp 594–605
24. Wan X (2006) Beyond topical similarity: a structural similarity measure for retrieving highly similar documents. Knowl Inf Syst 15:55–73
25. Yang J, Cheungand WK, Chen X (2009) Learning element similarity matrix for semi-structured document analysis. Knowl Inf Syst 19:53–78

26. Zhang M, Kao B, Cheung DW, Yip KY (2007) Mining periodic patterns with gap requirement from sequences. ACM Trans Knowl Discov Data 1:7

Author Biographies

Ansel Yoan Rodríguez-González received his B.S. degree in Computer Sciences from Havana University, Cuba, in 2004 and his M.C. degree in Mathematics Sciences from Havana University, Cuba, in 2007. He is currently a Ph.D. student at the Department of Computer Science, National Institute of Astrophysics, Optics and Electronics, México.

José Francisco Martínez-Trinidad received his B.S. degree in Computer Science from Physics and Mathematics School of the Autonomous University of Puebla (BUAP), Mexico in 1995, his M.Sc. degree in Computer Science from the faculty of Computers Science of the Autonomous University of Puebla, Mexico in 1997 and his Ph.D. degree in the Center for Computing Research of the National Polytechnic Institute (CIC, IPN), Mexico in 2000. Professor Martinez-Trinidad edited/authored four books and over fifty journal and conference papers, on subjects related to Pattern Recognition.

Jesús Ariel Carrasco-Ochoa received his Ph.D. degree in Computer Science form the Center for Computing Research of the National Polytechnic Institute (CIC-IPN), Mexico, in 2001. He works as full time researcher at the National Institute for Astrophysics, Optics and Electronics of Mexico. His current research interests include Logical Combinatorial Patter Recognition, Data Mining, Testor Theory, Feature and Prototype Selection, Text Analysis, Fast Nearest Neighbor Classifiers and Clustering.

José Ruiz Shulcloper graduated from University of Havana in 1972, received his Ph.D. degree in mathematics from the Moscow State University in 1978. Since 1978, he is the leader of the research area on Logical Combinatorial Pattern Recognition and author of more than 100 scientific publications on pattern recognition and data.