Statistical Supports for Mining Sequential Patterns and Improving the Incremental Update Process on Data Streams

Pierre-Alain Laur[1], Jean-Emile Symphor[1], Richard Nock[1], and Pascal Poncelet[2]

[1] Grimaag-Dépt Scientifique Interfacultaire,
Université Antilles-Guyane, Campus de Schoelcher,
B.P. 7209, 97275 Schoelcher Cedex, Martinique, France
{palaur,je.symphor,rnock}@martinique.univ-ag.fr
[2] Lg2ip-Ecole des Mines d'Alès,
Site EERIE, parc scientifique Georges Besse,
30035 Nîmes Cedex, France
pascal.poncelet@ema.fr

Abstract. Recently the knowledge extraction community takes a closer look to new models where data arrive in timely manner like a fast and continous flow, *i.e.* data streams. As only a part of the stream can be stored, mining data streams for sequential patterns and updating previously found frequent patterns need to cope with uncertainty. In this paper, we introduce a new statistical approach which biaises the initial support for sequential patterns. This approach holds the advantage to maximiez either the precision or the recall, as chosen by the user, and limit the defradation of the other criterion. Moreover, these statistical supports help building statistical borders which are the relevant sets of frequent patterns to use into an incremental mining process. Theoretical results show that the technique is not far from the optimum, from the statistical standpoint. Experiments performed on sequential patterns demonstrate the interest of this approach and the potential of such techniques.

1 Introduction

A growing body of works arising from databases and data mining manage to deal with ordered sequence of items that arrives in timely manner, also known as Data Stream. Data stream have seen the emergence of crucial problems that were previously not as pregnant for databases, such as the accurate retrieval of information in a data flow that prevents its exact storage, and whose information may evolve through time. Emerging and real applications generate data streams: trend analysis, fraud detection, intrusion detection, click stream, among many others [13, 9, 12]. An important task in datamining that has recently attracted significant attention is to build the set of most frequent patterns encountered in the data stream. Addressing this issue faces a particular and highly non-trivial problem. The knowledge of the stream is only partial at any time. Basically, the data stored catches a glimpse of a data stream and thus the information we mine should take into account the uncertainty generated by this partial observation of the whole stream. From the frequent pattern standpoint, for instance, it is not enough to observe some pattern as frequent in the data stored, it is much more important to predict if it is really frequent or infrequent on the whole data stream. The first problem, we are facing is to make a statistical approximation of the true supports by observed supports. More over, in the case of data streams, where the database is subject to be updated regularly, maintaining frequent patterns has been addressed by various incremental algorithms, in order to mine frequent patterns without having to build repeatedly everything from scratch [33]. This is the second problem we are facing which consists in constructing and maintaining over the successive updating, a set of frequent patterns the most relevant as possible in an incremental mining process. The remainder of this paper is organised as follow. Section 2 goes deeper into presenting the problem statement. Our theoretical approach is presented and discussed in section 3. Section 4 reports the results of our experiments.We make some comparisons with related approaches in section 5. Finally, in section 6 we summarize our finding and conclude the paper with future avenues of research.

2 Problem statement

2.1 Mining sequential patterns into data streams

We define *items* as the unit information, *itemsets* to be sets of items [1], and *sequential patterns* to be sequences of items [25]. A pattern is *θ-frequent* if it occurs in at least a fraction θ of the data stream (called its support), where θ is a user-specified parameter.

Our problem is motivated by the fact that the data we store catches a glimpse of a data stream, and the information we mine should take into account the uncertainty generated by this *partial* observation of the whole stream. Our setting is thus a bit more downstream than those of [2, 4, 6, 15, 29]. Given the nature of the streaming data, there are two sources of error when estimating frequent patterns from the available part of the stream:

1. it is possible that some patterns observed as frequent might in fact not be frequent anymore from a longer observation of the data stream;
2. on the other hand, some patterns observed as not frequent may well in fact be frequent from a longer history of the data stream.

The point is that it is statistically hard to nullify both sources of error from the observation of a *subset*, even very large, of the whole data stream [28]. This unsatisfiable goal can be relaxed to the tight control of one source or error, while keeping the other one within reasonable bounds. This goal, which we address in this paper, can be summarized as follows; the user fixes some related parameters and chooses a source of error:

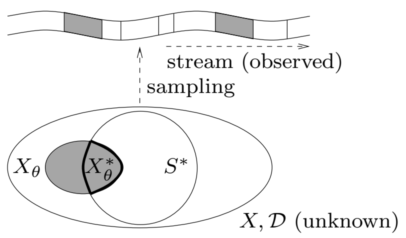

Fig. 1. Problem statement.

(a) the source of error chosen is nullified with high probability;
(b) the other one incurs a limited loss.

In this paper, we propose a solution to this problem which is statistically near optimal: any other technique that would yield a loss significantly smaller on (b) would not satisfy (a), regardless of its computation time.

2.2 Borders and Incremental updates

Let DB be the original database and *minSupp* the minimum support. Let *db* be the increment database where new sequences of items are added to DB. $DB' = DB \cup db$ is the updated database containing all sequences from DB and *db*. Let L_{DB} be the set of frequent patterns in DB. In incremental mining of sequential patterns, one important problem is to find frequent sequences in DB', noted $L_{DB'}$, with respect to a given minimum support θ provided by the user. Furthermore, the incremental approach has to take advantage of previously discovered patterns in order to avoid re-running mining algorithms when the data is updated. More precisely, here we will not address the whole problem of incremental mining here. In fact some algorithm does not only keep the L_{DB} set but for improving performance they keep a bigger set calls a border. Such border holds informations in order to avoid computing from scratch. We will address the following issue: which frequent patterns should be a member of this border when we have choosen to nullified with high probability one of the source of error from section 2.1.

3 Our approach

3.1 Definitions

The data stream is supposed to be obtained from the repetitive sampling of a potentially huge *domain* X which contains all possible data sequences, see figure 1. Each sequence is sampled independently through a distribution $\mathcal{D}$, see figure 1, for which we make absolutely *no* assumption, except that it remains fixed: there is no distribution drift through time. The reader may find relevant empirical studies on concept drift for supervised mining in [30]. The user specifies a real $0 < \theta < 1$, the *theoretical* support, and ideally wishes to recover all the *true* θ-frequent sequential patterns of X. This set is called X_θ, see figure 1.

Definition 1.

$$\forall 0 \leq \theta \leq 1, X_\theta = \{T \in X : \rho_X(T) \geq \theta\} \ , \qquad (1)$$

with $\rho_X(T) = \sum_{T' \in X : T \leq_t T'} \mathcal{D}(T')$*, and* $T \leq_t T'$ *means that* T *generalizes* T'*.*

The recovery of X_θ faces two problems. Apart from our statistical estimation problem, there is a combinatorial problem which comes from the fact that X is typically huge. The set of observed data sequences which we have sampled from X in the data stream (S) has a size $|S| = m$ $(|S| << |X|)$. In our framework, we usually reduce this difference with some algorithm returning a superset S^* of S, having size $|S^*| = m^* > m$. Typically, S^* contains additional generalizations of the elements of S. The key point is that S^* is usually still not large enough to cover X_θ, regardless of the way it is built, so that the pregnancy of our statistical estimation problem remains the same.

Our statistical estimation problem can be formalized as follows:

- approximate as best as possible the set :

$$X^*_\theta = X_\theta \cap S^* \ , \qquad (2)$$

  of true θ-frequent sequential patterns of X on S^*, for any S and S^* (see figures 1 and 2).

Remark that $\forall T \in S^*$, we cannot compute exactly $\rho_X(T)$, since we do not know X and $\mathcal{D}$. Rather, we have access to its best unbiased estimator, $\rho_S(T) = \sum_{T' \in S : T \leq_t T'} w(T')$ $(\forall T \in S^*)$. Here, $w(T')$ is the weight (observed frequency) of T' in S. We adopt the following approach to solve our problem:

- find some $0 < \theta' < 1$ and approximate the set X^*_θ by the set of *observed* θ'-frequent of S^*, that is:

$$S^*_{\theta'} = \{T \in S^* : \rho_S(T) \geq \theta'\} \ . \qquad (3)$$

Before computing θ', we first turn to the formal criteria appreciating the goodness-of-fit of $S^*_{\theta'}$, see figure 2. The two sources of error, committed with respect to X^*_θ, come from the two subsets of the symmetric difference with $S^*_{\theta'}$, as presented in figure 2. To quantify them, let us define:

$$TP = \sum_{T \in S^*_{\theta'} \cap X^*_\theta} \mathcal{D}(T) \qquad (4) \qquad\qquad FN = \sum_{T \in X^*_\theta \backslash S^*_{\theta'}} \mathcal{D}(T) \qquad (6)$$

$$FP = \sum_{T \in S^*_{\theta'} \backslash X^*_\theta} \mathcal{D}(T) \qquad (5) \qquad\qquad TN = \sum_{T \in S^* \backslash (S^*_{\theta'} \cup X^*_\theta)} \mathcal{D}(T) \ (7)$$

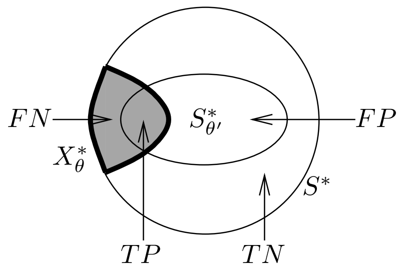

Fig. 2. The error estimation.

The *precision* allows to quantify the proportion of estimated θ-frequent patterns that are in fact not true θ-frequents, out of $S^*_{\theta'}$:

$$\mathtt{P} = TP/(TP+FP) \ . \tag{8}$$

Maximizing $\mathtt{P}$ leads to minimize our first source of error. Symmetrically, the *recall* allows to quantify the proportion of true θ-frequent patterns that are missed in $S^*_{\theta'}$:

$$\mathtt{R} = TP/(TP+FN) \ . \tag{9}$$

Maximizing $\mathtt{R}$ leads to minimize our second source of error. We also make use of a well known quantity in information retrieval, which is a weighted harmonic average of precision and recall, the $\mathtt{F}_\beta$-measure. Thus, we can adjust the importance of one source of error against the other by adjusting the β value:

$$\mathtt{F}_\beta = (1+\beta^2)\mathtt{PR}/(\mathtt{R}+\beta^2\mathtt{P}) \ , \tag{10}$$

Our approach is also represented using databases on figure 3.

- (a) (on this figure) represents a database that contains all possible data sequences of the stream X.
- (b) is the observed part of the stream S that could be store in a database. It has been build by sampling the stream through the distribution $\mathcal{D}$.
- (c) is the set of true θ-frequent sequential patterns of X in S^* that we want to approximate as best as possible.
- (d) represent the formal criteria for appreciating the goodness-of-fit of $S^*_{\theta'}$ by comparing results from (c) and the sets built from (e).
- (e) show the differents sets we will construct using well choosen θ' value on (b).

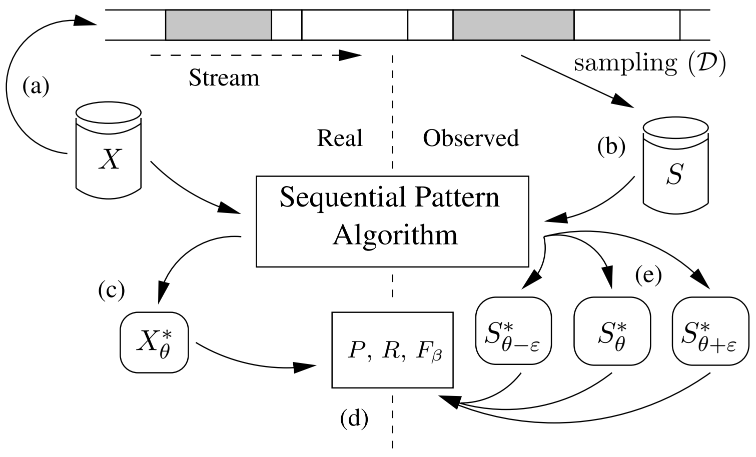

Fig. 3. Our databases framework.

3.2 About choosing θ'

A naive approach to approximate X^*_θ would typically be to fix $\theta' = \theta$. Unfortunately, the main and only interesting property of $S^*_{\theta'}$ is that it converges with probability 1 to X^*_θ as $m \to \infty$ from Borel-Cantelli Lemma [7]. Glivenko-Cantelli's Theorem gives a rate of convergence as a function of m, but it does not give the possibility to maximize $\texttt{P}$ or $\texttt{R}$.

Informally, our approach boils down to picking a θ' different from θ, so as to maximize either $\texttt{P}$ or $\texttt{R}$. Clearly, extremal values for θ' would do the job, but they would yield very poor values for F_β, and also be completely useless for data mining purposes. For example, we could choose $\theta' = 0$, and would obtain $S^*_0 = S^*$, and thus $\texttt{R} = 1$. However, in this case, we would also have $\texttt{P} = |X^*_\theta|/|S^*|$, a too small value for many domains and values of θ, and we would also keep all elements of S^* as true θ-frequents sequential patterns, a clearly huge drawback for mining issues. We could also choose $\theta' = 1$, so as to be sure to maximize $\texttt{P}$ this time; however, we would also have $\texttt{R} = 0$, and would keep *no* element of S^* as θ-frequent sequential patterns.

These extremal examples show the principle of our approach. Should we want to maximize $\texttt{P}$, we would pick a θ' larger than θ to guarantee with high probability that $\texttt{P} = 1$, yet while keeping large enough values for $\texttt{R}$ (or F_β), and a set $S^*_{\theta'}$ not too small to contain significant informations. There is obviously a statistical barrier which prevents θ' to be too close to θ to keep the constraint $\texttt{P} = 1$. The objective is to be the closest to this barrier, which statistically guarantees the largest $\texttt{R}$ values under the constraint.

3.3 Statistical borders

Statement In this section, we introduce two statistical borders (upper and lower) which are relevant in the choice of frequent sequential patterns to use into an incremental mining process. We adopt the concise probabilistic notation of [21], and define for some predicate P the notation $\forall^{\delta} P$ which means that P holds for all but a fraction $\leq \delta$ of the sets S sampled under distribution $\mathcal{D}$. Equivalently, P holds with probability $\geq 1-\delta$ over the sampling of S on distribution $\mathcal{D}$. The following definition is the cornerstone of our approach.

Definition 2. *$\forall 0 \leq \theta \leq 1$, $\forall 0 \leq \varepsilon \leq 1$, $\forall S \subseteq X$, we say that S^* is a upper statistical border of X iff $\forall T \in X^*_{\theta}$,*

$$\rho_S(T) \geq \rho_X(T) - \varepsilon \ . \tag{11}$$

Respectively, we say that S^* is a lower statistical border of X iff $\forall T \in S^* \backslash X^*_{\theta}$,

$$\rho_S(T) \leq \rho_X(T) + \varepsilon \ . \tag{12}$$

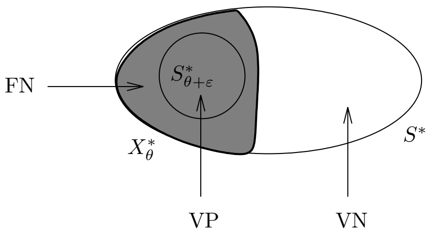

Fig. 4. Lower statistical border.

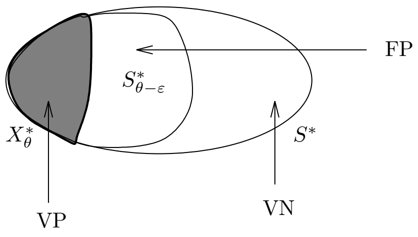

Fig. 5. Upper statistical border.

The way we use definition 2 is simple. Consider that the user has fixed both the theoretical support $0 \leq \theta \leq 1$, and a *statistical risk* parameter $0 < \delta < 1$. Suppose we can find ε such that:

$$\forall^{\delta}, S^* \text{ is a lower statistical border of } X \ . \tag{13}$$

Now, fix $\theta' = \theta + \varepsilon$, so that we keep $S^*_{\theta+\varepsilon}$. Because (13) holds, we observe $\forall T \in S^* \backslash X^*_{\theta}, \rho_S(T) \leq \rho_X(T) + \varepsilon < \theta + \varepsilon$. Thus, we obtain $\forall^{\delta}, S^*_{\theta+\varepsilon} \subseteq X^*_{\theta}$, which easily yields:

$$\forall^{\delta}, \mathtt{P} = 1 \ . \tag{14}$$

Thus, there is *no* first source of error, with high probability. All patterns of $S^*_{\theta+\varepsilon}$ are true *θ-frequent* of X^*_{θ}, but some are missing (see figure 4). $S^*_{\theta+\varepsilon}$ is the biggest possible set that only hold true *θ-frequent* patterns of X^*_{θ} after looking inside a sample of the stream. This set is define as the lower statistical border of X^*_{θ} obtained from the observed part of the stream S^* of X.

Now, suppose we can find ε such that $\forall^{\delta}, S^*$ is a upper statistical border of X, and fix this time $\theta' = \theta - \varepsilon$, so that we keep $S^*_{\theta-\varepsilon}$. Because of the property of

S^*, we observe $\forall T \in X^*_\theta, \rho_S(T) \geq \rho_X(T) - \varepsilon \geq \theta - \varepsilon$, which yields $\forall^\delta, X^*_\theta \subseteq S^*_{\theta-\varepsilon}$, and finally:

$$\forall^\delta, \mathtt{R} = 1 \ . \tag{15}$$

In that case, there is *no* second source of error with high probability. $S^*_{\theta-\varepsilon}$ hold patterns true *θ-frequent* of X^*_θ, but it also hold extra ones which are not (true *θ-frequent* figure 5). $S^*_{\theta-\varepsilon}$ is the smallest possible set that hold all the true *θ-frequent* patterns of X^*_θ after looking inside a sample of the stream. This set is define as the upper statistical border of X^*_θ obtained from the observed part S^* of the stream X.

Computationally speaking, both sets $S^*_{\theta+\varepsilon}$ and $S^*_{\theta-\varepsilon}$ can be easily built empirically from S^*. Finally, the way we address 3 is now reduced to finding an accurate value of ε such that S^* is an upper or lower statistical border of X with high probability.

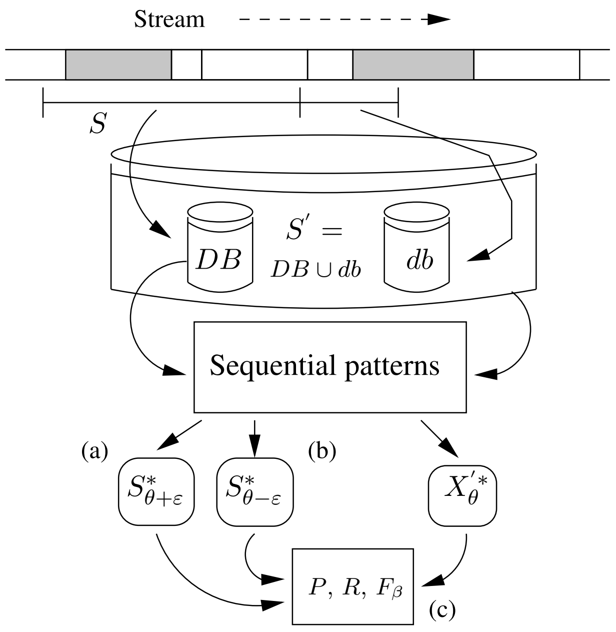

Fig. 6. Statistical borders for the incremental update process.

Incremental updates We show in figure 6 how we use the previously defined statistical borders into an incremental mining process. We search the *θ-frequent* sequential patterns from the statistical borders ((a) and (b) on the figure 6). These borders have been built for a θ' support, computed from the orginal

θ support and the ε value obtained from S (DB). These borders allow us to approximate as best as possible the set of all true θ*-frequent* sequential patterns for $S^{'}$, which represents the database DB updated with the increment db $(S^{'} = S \cup db)$. In the section X, we will compare the set of θ*-fréquent* sequential patterns obtained through the statistical borders using the set $X_{\theta}^{'*}$ ((c) on figure 6). This set represents all the true θ*-frequent* sequential patterns after the incremental update occured $(DB \cup db)$.

3.4 finding ε

The following Theorem gives a value ε which yields with high probability an upper statistical border of X.

Theorem 1. $\forall X, \forall \mathcal{D}, \forall m > 0, \forall 0 \leq \theta \leq 1, \forall 0 < \delta \leq 1$, *the following holds:* $\forall^{\delta}, S^{*}$ *is an upper statistical border of* X*, for any* ε *satisfying:*

$$\varepsilon \geq \sqrt{(1/(2m)) \ln(|X_{\theta}^{*}|/\delta)} \ .$$

Proof. A standard application of Chernoff bounds yields that the probability for any *fixed* itemset $T \in X$ to observe $\rho_S(T) \leq \rho_X(T) - \varepsilon$ is no more than $\exp(-2m\varepsilon^2)$. Using the union bound, the probability that this is observed for *some* itemset $\in X_{\theta}^{*}$ is no more than $|X_{\theta}^{*}| \exp(-2m\varepsilon^2)$. Solving for ε this quantity equal to δ yields the Theorem.

The same kind of result can be obtain for a lower statistical border, with the same proof. Hereafter, we give the statement of the Theorem.

Theorem 2. $\forall X, \forall \mathcal{D}, \forall m > 0, \forall 0 \leq \theta \leq 1, \forall 0 < \delta \leq 1$, *the following holds:* $\forall^{\delta}, S^{*}$ *is a lower statistical border of* X*, for any* ε *satisfying:*

$$\varepsilon \geq \sqrt{(1/(2m)) \ln(|S^{*} \backslash X_{\theta}^{*}|/\delta)} \ .$$

Theorems 1 and 2 say that finding (lower/upper) statistical borders is a fairly easy task. What they do *not* say is whether this simplicity can be replaced by another approach, may be more sophisticated, to find significantly *better* statistical borders. In other words, could there exists equivalents to Theorems 1 and 2 with a significantly smaller ε ? Let's show that the answer to this question is no.

The following argument shows that there are no significant better statistical borders than those proposed in Theorems 1 and 2. Informally, we build to this extent a skewed distribution $\mathcal{D}$ on some very simple X_{θ}^{*}, such that with probability $\geq \delta$ we "miss" the statistical border for some value of ε slightly smaller than that proposed in Theorems 1 or 2. The following Theorem proves the result for the upper statistical border of X.

Theorem 3. $\exists X, \exists \mathcal{D}, \exists m > 0, \exists 0 \leq \theta \leq 1, \exists 0 < \delta \leq 1$ *such that the following holds: with probability* $\geq \delta$*,* S^* *is* ***not*** *an upper statistical border of* X*, for any* ε *satisfying:*

$$\varepsilon \leq c\sqrt{(1/(2m))\ln(|X^*_\theta|/\delta)} \ ,$$

for some constant $c < 1$*.*

The proof of this Theorem is postponed to the appendix. Since failing to obtain an upper statistical border of X ultimately means failing to have maximal recall, our computation of ε is thus close to the best possible which keeps the guarantees we want on recall. Obviously, the same kind of Theorem holds for a lower statistical border of X, and its proof follows that of Theorem 3.

Theorem 4. $\exists X, \exists \mathcal{D}, \exists m > 0, \exists 0 \leq \theta \leq 1, \exists 0 < \delta \leq 1$ *such that the following holds: with probability* $\geq \delta$*,* S^* *is* ***not*** *a lower statistical border of* X*, for any* ε *satisfying:*

$$\varepsilon \leq c\sqrt{(1/(2m))\ln(|S^* \backslash X^*_\theta|/\delta)} \ ,$$

for some constant $c < 1$*.*

The criterion which is not controlled may suffer some loss, but what Theorems 3 and 4 say on this criterion is that the loss it incurs is also statistically near-optimal; a simple argument shows that the *value* of this loss behaves in a very reasonable manner: Theorems 1 and 2 guarantee that $\varepsilon \leq (1/m)\log m^*$ for reasonable δ; since generating S^* is as worst reasonably polynomial in m, we can expect $m^* \leq m^k$ for some small constant $k > 0$, which yields $\varepsilon \leq 1/m^{1-o(1)}$. In other words, $\theta \pm \varepsilon$ converges quite rapidly to θ, and since a similar rate of convergence of the observed frequencies to their expectations holds as well, we observe a fast convergence of $X^*_\theta \backslash S^*_{\theta+\varepsilon} \rightarrow \emptyset$ (for $\theta' = \theta + \varepsilon$) or $S^*_{\theta-\varepsilon} \backslash X^*_\theta \rightarrow \emptyset$ (for $\theta' = \theta - \varepsilon$), ensuring a reasonably fast maximization of the unconstrained criterion as well.

4 Experiments

Two kind of experiments were performed. Since our method for statistical supports does not depend on the algorithm used to build the frequent sequential patterns, and since its computational cost is not larger than that of computing ordinary frequencies, we do not evaluate it in terms of speed. Rather we focus on evaluating: how our statistical support can be helpful to mine sequential patterns on a data streams, given a fragment of this stream and how we can use statistical borders to help optimizing an incremental updates process.

4.1 Measures

We previously define in eqs (8), (9) and (10) the following information retrieval measeures : the precision P, the recall R and a weighted harmonic average of both, the F_β measure. These measures are widely spread to evaluate statistical predictive models. We have chosen to use them in our experiments when considering sequential patterns mining.

4.2 Datasets and sequential pattern algorithms

In order to evaluate our predective method and use of statistical borders into an incremental updates, we have chosen two real life datasets form Web servers. The first database, named “Dragons”, is obtained from an Internet web site[3] from March 21th 2005 to March 28th 2005. More specifically the data represent the behavior of this Web site usage. The Web logsize is about 2,54Go (132k transactions). A preprocess was done in order to prune irrevelant data (spiders, robots, etc.). In order to avoid traditional problems when considerating raw web logs, URL pages having same values for similar variables were grouped together. Finally we consider that the session time was set to 4 hours. The second database, named “BuAG”, is obtained from 3,48Go (54k transactions) web log server of the University[4] from january 1st to November 1st 2004. It shows how user access to this University Library Web Site. As previously a preprocess was done and the session time was set to 3 minutes.

As we previously advocate the independancy of our method with the used algorithm we have chosen to use two traditional sequential pattern algorithms `PSP` [19] for the evaluation of the statistical supports themselves and `SPADE` [32] while testing the use of statistical borders in the incremental update process.

4.3 Statistical support

To analyze the correctness of our statistical supports, we need to evaluate as many situation as possible, that is, we need to use our method with a range as large as possible for each of the parameters (the exception is δ, which was fixed to be .05). These parameters that vary during ours experiments are described in figure 7.

Better than using a real data stream, which would possibly skew the performance evaluations of the statistical supports, we have chosen to simulate data streams assuming the complete knowledge of the domains, thus allowing to compute exact values for the performance measurements. More precisely, we simulate data streams by sampling each database into fragments. For example, we could consider that data arrive in a timely manner from the “Dragons” database, and that only 20% of the whole data is stored. So we pick 20% of the transactions of

[3] www.elevezundragon.com.
[4] www.univ-ag.fr/buag/.

Database	θ	sampling1	sampling2
Dragons	[.07, .2] / .03	[.02,.1] / .01	[.15, .7] /.05
BuAG	[.08, .2] / .03	[.05,.1] / .01	[.15, .7] /.05

Fig. 7. Range of parameters for the experiments in the form $[a, b]/c$, where a is the starting value, c is the increment, and b is the last value.

this database, we consider that it is the data stored, and check the the precision `P`, the recall `R` and the weighted harmonic average of both F_β values with the whole database. Obviously, 20% is an arbitrary percentage, and we have in fact chosen to sample the database on a broad range of percentages using two scales. The first allows a fine sampling of the database, for small values ranging from 2% to 10% by steps of 1% ("sampling1" in figure 7), and typically gives an idea of what may happens for very large, fast data streams.We have completed this first range with a coarse range of samplings, from 15% to 70% by steps of 5% ("sampling2" in figure 7), which gives an idea of the average and limit behaviors of our method.

On the top of our experiments, we have chosen to use a traditional sequential pattern algorithm `PSP` [19]. Given the very large number of tests to do for each database, we have written a test generator, which automatically crosses the parameters, and make all experiments for all possible tuples of parameters.

Due to the very large number of experiments and the lack of space to report them all, we have chosen to report some plots we consider as representative, and synthesize the whole results.

Figure 8 shows result from experiments on choosen databases with $\delta = .05$. Each plot describes for one database and one support value, either the precision `P` or the recall `R` of the three methods which consist in keeping $S^*_{\theta-\varepsilon}, S^*_\theta$, and $S^*_{\theta+\varepsilon}$.

A first glance at these plots reveals that their behavior is almost always the same. Namely:

- the precision `P` increases with θ' (eq. 3), while the recall `R` decreases with θ',
- the precision `P` equals or approaches 1 for mostly storing sizes when $\theta' = \theta + \varepsilon$,
- the recall `R` equals or approaches 1 for mostly storing sizes when $\theta' = \theta - \varepsilon$.

These observations are in accordance with the theoretical results of section 3.2. There is another phenomenon we may observe: the recall `R` associated to $\theta' = \theta + \varepsilon$ is not that far from the recall `R` of $\theta' = \theta$. Similarly, the precision `P` associated to $\theta' = \theta - \varepsilon$ is not that far from the precision `P` of $\theta' = \theta$. This shows that the maximization of the precision `P` or the recall `R` is obtained at a reduced degradation of the other parameter. We also remark that the precision `P` plots tend to be better than that of the recall `R`. This is not really surprising, as advocated in section 3.2, since the range of values for the precision `P` is smaller than that

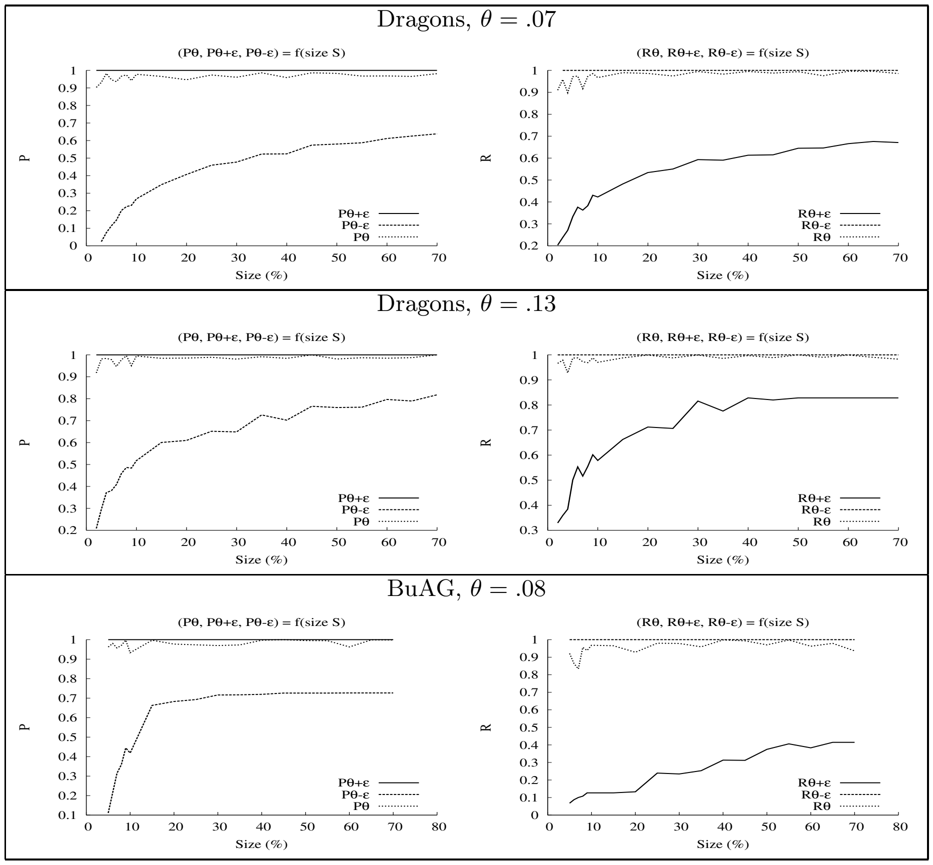

Fig. 8. Examples of plots with $\delta = .05$ and three θ values. For theses values we give the `P` (left plot) and `R` (right plot) for the three methods consisting in picking $S^*_{\theta-\varepsilon}, S^*_{\theta}, S^*_{\theta+\varepsilon}$.

of the recall `R`.

A close look at small storing sizes of the streams (before 10%) also reveals a more erratic behavior without straight convergence to maximal precision `P` or recall `R`. This behavior is not linked to the statistical support, but to the databases used. Indee, the data stream is simulated out of each database. So,small databases lead to even smaller storing sizes, and frequent sequential patterns kept out of small databases are in fact trickier to predict than for bigger databases. This point is important as, from a real-world standpoint, we tend to store very large databases, so we may expect this phenomenon to be reduced.

On these databases, another phenomenon seems to appear. First of all, because of the small values for θ, some tests have not be performed because $\theta-\varepsilon$ was

< 0. Furthermore, the greater difference observed between the curves seems to stem out from the different sizes of databases. For example, the BuAG database is smaller than the Dragons database by a factor 2.4. This, we think, explains the greater differences between the curves: they are mostly a small database phenomenon, and may not be expected from larger databases, or even real-world data streams.

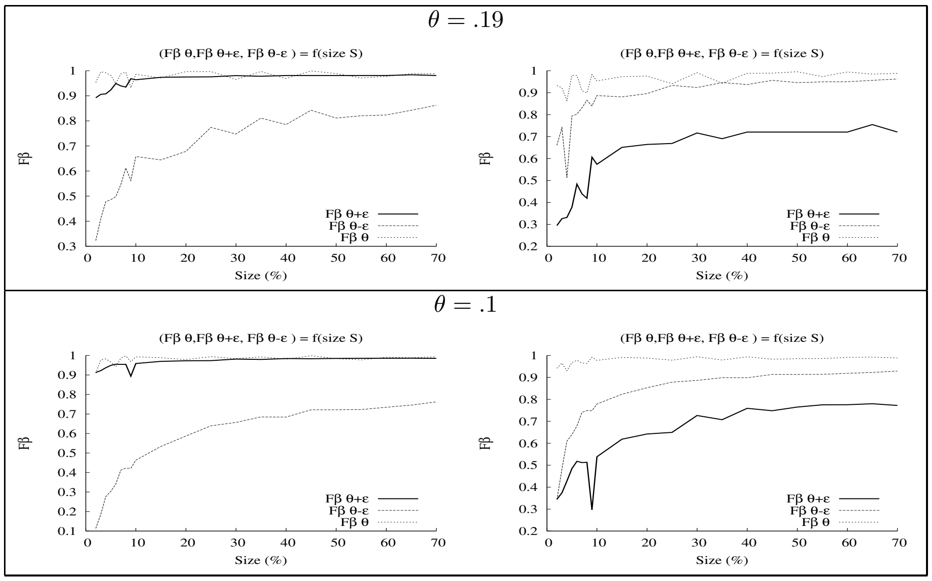

Fig. 9. Two sets of plots of the F_β value from the Dragons database, with $\beta = .2$ for the left plots and $\beta = 1.8$ for the right plots.

In figure 9, two sets of two plots taken from the Dragons database plot the weighted harmonic average of precision and recall F_β measure, against the size of the stream used (in %). The values of β have been chosen different from 1, since it would make no real sense to put the same weight into precision and recall given, that we put our primary emphasis on the maximization of a single criterion. The values have also been choosen not too small or too large to yield a reasonable prominence of one criterion (.2 and 1.8, see figure 9).

Different phenomenons appears on these plot. First of all, as seen on the left plots, the weighted harmonic average of precision and recall F_β value displays the advantage of choosing $\theta' = \theta + \varepsilon$ against the choice $\theta' = \theta$. Meanwhile, as shown on the right plots choosing $\theta' = \theta - \varepsilon$ against the choice $\theta' = \theta$ is not always a win / win strategy. Moreover, recall `R` that this is obtained while statistically guaranteeing the *maximal* value for whichever of the precision `P` or

Database	θ	size of DB	size of db
Dragons	[.07, .2] / .03	[.2,.6] / .1	[.1, .5] /.1
BuAG	[.08, .2] / .03	[.2,.6] / .1	[.1, .5] /.1

Fig. 10. Range of parameters for the experiments in the form $[a, b]/c$, where a is the starting value, c is the increment, and b is the last value.

recall `R` criterion, as chosen by the user. Such result throws out a question that we are working on in a near future: is it possible to build a new frequent sequential patterns set based on a mix of the frequent sets obtained with this statistical support, which will have a better weighted harmonic average of precision and recall F_β value ?

4.4 Incremental updates

We will focus here on the evaluation of the quality of our statistical borders using `P` and `R` measures more than their computation costs. To inforce that our method is independant from any algorithm we hace choosen here to use another well know sequential patterns algorithm `SPADE` [32] different from the one used in the previous section. To analyze the quality of our statistical bordes, we have evaluated as many situations as possible using a broad range of parameters (except for δ which will be set to .05). These variations are described in the figure 10. The first parameter shows the different support values used during these experiments ("θ"). The second one, define the ratio between the size of DB and the simulated data stream ("size of DB"). The last one, define the ratio between the size of db and the size of DB. In our case db will be at most 50% of the size of DB ("size of db"), this parameter allows us to control the size of the increment. In order to cross all these parameters, a generator has been built to run the tests.

As previously, we have chosen to simulate a data streams assuming the complete knowledge of the domains X. Thus to simulate the stream we sample each database into fragments DB (S). For example, we could consider that data arrive in a timely manner from the *Dragons* database, and that only 20% of the whole data could be stored. So we pick 20% of the transactions of this database, we consider that it is the data stored DB. We just now need to take an increment, db for example of size of 10%, of this database. Then we build the statistical borders (upper and lower) defined into section 3.3 for DB.

Figures 11 and 12 show results of experiments, with $\delta = 0.05$, for the databases *Dragons* and *BuAG*. In order to evaluate the quality of our borders, we plot their behaviour for `P` and `R` in relation to $S^{'}$ ($S^{'} = DB \cup db$). Thus, plots $P_{\theta+\varepsilon}$ and $P_{\theta-\varepsilon}$, represent the precision respectively for the lower statistical border and for the upper statistical border. As well as the $R_{\theta\pm\varepsilon}$ plots stand for the recall. Plots P_θ and R_θ stand for the naive choice of $\theta' = \theta$. Here again a first glance at these plots reveals that their behavior is almost always the same and moreover similar to the ones noticed in the experiments of section 4.3:

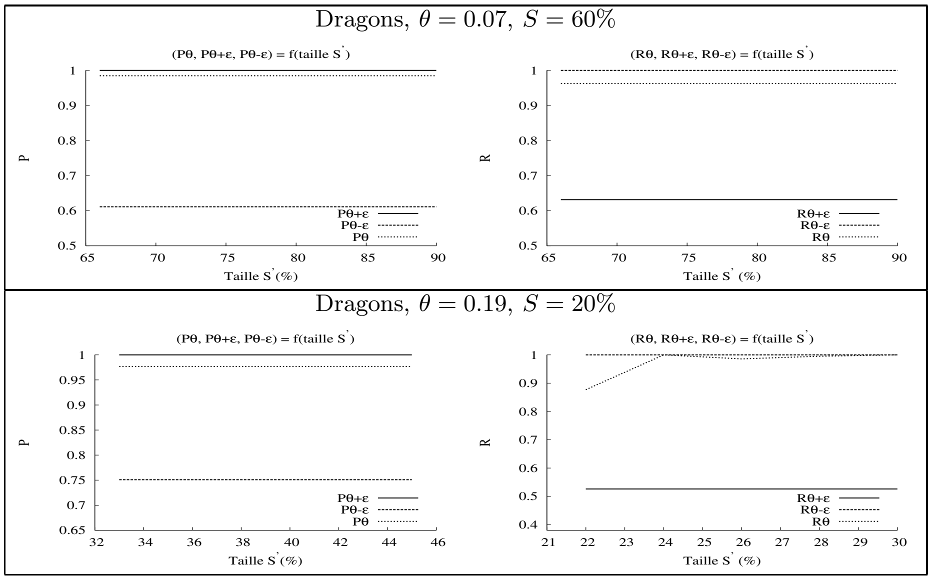

Fig. 11. shows precision P plots (on the left) and recall R plots (on the right) on the Dragon database for three values of θ' : $\theta - \varepsilon, \theta$ and $\theta + \varepsilon$, two values of θ and two size of S (% of—X—).

– the precision P equals or approaches 1 for mostly storing sizes when $\theta' = \theta + \varepsilon$,
– the recall R equals or approaches 1 for mostly storing sizes when $\theta' = \theta - \varepsilon$.

Again, these observations are in accordance with the theoretical results of section 3.2. The recall R associated to $\theta' = \theta + \varepsilon$ is not that far from the recall R of $\theta' = \theta$ as it was in section 4.3. Similarly, the precision P associated to $\theta' = \theta - \varepsilon$ is not that far from the precision P of $\theta' = \theta$. We also keep here our property of maximization of one criteria while the other is obtained at a reduced degradation cost of the other. Here also, we remark that, for the same reasons, the precision P plots tend to be better than that of the recall R especially on the figure 12.

Variation of the size of S (DB) has a non trivial impact on the plots and comfort a theorical result that sounds logical. Indeed if we have a deeper look at the recall R, we notice that, either on figure 11 or on figure 12, observed values for this quantity increase along with the size of S. In fact the more data are available the more quality the prediction is.

As in the experiment of section 4.3, for the same reasons advocate there, an other phenomenon seems to appear. First of all, because of the small values for θ, some tests have not be performed because $\theta - \varepsilon$ was < 0. Furthermore, the greater difference observed between the curves seems to stem out from the different sizes of databases.

On these plots (figures 11 and 12), the choice of $\theta' = \theta$ tends to give better results if we consider the mean of P and R than the choice of the two other values

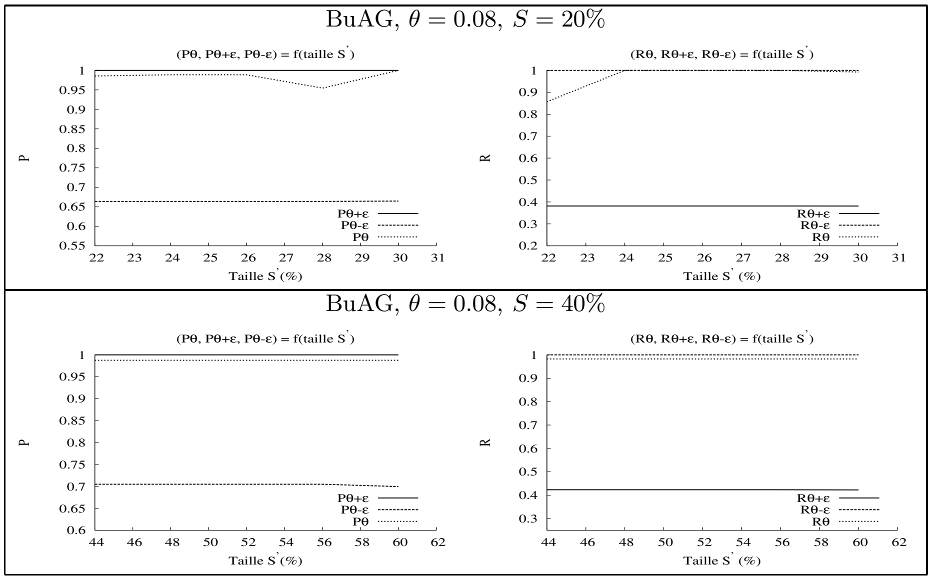

Fig. 12. shows precision `P` plots (on the left) and recall `R` plots (on the right) on the BuAg database for three values of θ' : $\theta - \varepsilon, \theta$ and $\theta + \varepsilon$, two values of θ and two size of S (% of—X—).

of θ', but in this case neither `P`, nor `R` are close to 1 with a high probability. With our approach, we can efficiently optimize the incremental update process. In the case where we have enough storage space, we would choose $\theta' = \theta - \varepsilon$, the upper border, which holds tough guarentees on the future frequents (`R` $= 1$). In these case the amount of complementary computations for the update would be low. However in the case where we have a limited storage space it would be wiser to keep a smaller border. $\theta' = \theta + \varepsilon$, the lower border, will there be the best choice as it holds the more revelant informations available (`P` $= 1$). Thus we can see our statistical borders as useful tools for the incremental update. Moreover they represent the limit from where it is useless to store more information in the case of $\theta' = \theta - \varepsilon$ (inferior value for $\theta^{'}$); and on other hand the limit from where the loss of information would be to important $\theta' = \theta + \varepsilon$ (superior value for $\theta^{'}$).

5 Related work

5.1 About mining sequential patterns from data streams

A significant body of previous works has addressed the accurate storing of the data stream history. This storage problem consists in finding compact data structures to reduce the size of the data kept out of the stream, while guaranteeing with high probability that the items *observed* as frequent from the stream are still observed frequent inside the data structure [2, 6, 15]. The first approach was proposed by [18] where they define the first single-pass algorithm. Li et al. [17] use a top-down frequent itemset discovery scheme. A regression-based algorithm is proposed in [26] to find frequent itemsets in sliding windows. Chi et al. [5] consider closed frequent itemsets. In [10], they propose a FP-tree-based algorithm [14] to mine frequent itemsets at multiple time granularities by a novel tilted-time windows technique. It should be more convenient, from a data mining standpoint, to try to reduce the storage uncertainty with an accurate forecasting on the data stream, rather than reducing it to the portion observed. This is the main difference with our framework.

A previous Chernoff-type analysis, due to [27], may be fit to handling data streams as well, but for slightly more restricted problems; in particular, while some of the bounds would typically not be applicable for large S^*, the others would be mainly addressed at controlling the precision of the support estimation, and not the maximization of our criteria (precision or recall). Finally, such results (and ours) do not rely on optimizing *the estimation* of these criteria (utility functions), like for example in [8], [24].

Perhaps the works closest to ours are some that have specifically focused in forecasting some properties on data due to a lack of information, either because the data are noisy [31], or because a constraint exists on the data storage that prevents to keep all the information [11]. A first difference with these works is that they focus on approximating 2 from section 3 without emphasis on the components of the solution's accuracy (precision and recall). Thus, they somewhat rely on the sole statistical hardness of the estimation task [28], without drilling down into its components. A second difference, very technical, is that all their bounds are pointwise, *i.e.* hold for a single itemset, and typically do not yield properties that hold uniformly, *i.e.* for a whole set of itemsets. That later case makes it necessary to bring some additional material, such as approximating cardinals or the concept of statistical borders , but at this price, we are able to show the statistical near-optimality of our approach (an important issue, not discussed in [11, 31]).

5.2 About incremental updates

Since 1996, numerous research works focus on maintaining frequent patterns sets obtained from static databases. Partition and FUP (Fast UPdate) [4] are two algorithms when the database is splitted in order to firstly seek for local frequents in each partition and then combining them in order to compute global frequents using cross validation technics. This rely on the hypothesis that global frequent patterns should be at least frequent in one partition. [23] developped an incremental update algorithm ISM (Incremental Sequence Mining) by maintaining a lattice of the database patterns build from all frequent patterns and the negative border. [34] have built an algorithm IUS(Incrementally Updating Sequence) by using a support value to limit the size of the stored negative border candidates. These approaches are subjected to all drawbacks linked to the use of the negative border:

- the space of the pattern candidates that have to be maintained is huge;
- It is necessary to consider the structural relationships that exist between patterns, especially in the case where patterns have a small support value.

[20] have proposed an incremental update algorithm ISE using a generating / pruning technic for the candidates that has the following drawbacks:

- the space of candidates could be huge, indeed the pruning phase (support counting) is very slow;
- this algorithm need many loops on the whole database. This phase costs a lots of time especially for long sequential patterns.

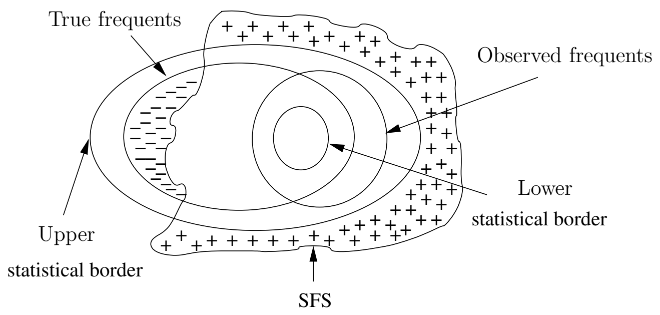

Fig. 13. Comparison.

IncSpan (Incremental Mining of Sequential Patterns in large database) algorithm from [3] rely on a statistical approach where they build a set of semi-frequents (SFS) by reducing the original value of the minimal support by a factor. Their main idea is that some of the frequent patterns in the updated database could come from the SFS or are already frequents in the database before the update occured. In another way, SFS is a kind of frontier area between frequent and non frequent patterns. On the figure 13, we show an exemple of the SFS set. This approach holds major insufficiency on a statistical stand point either on the estimation of the uncertainty due to the data streams as well as the way the set of pattern candidates that are used for the incremental update is built. Indeed, the choice of the factor used to reduce the minimal support value is heuristic and does no rely on any theorical assumption. This approach neither provide any certainty nor any information on the error made while building the SFS set and so on for the true *θ-frequent* patterns of the stream build on top of it (area with - symbol on the figure 13). Moreover, we don't have any guarentee on the fact that this set is minimum in matter of size (area with + symbol on the figure 13). This is very inconvenient for using it in the goal of optimizing the incremental update process.

6 Conclusion

There are five main contributions in this paper. First, we discuss the replacement of the conventional minimal support requirement for finding frequent sequential patterns by a statistical support, in case where storing the entire data is impossible (such as data streams), so as to keep some convenient properties over the data kept. Then, we provide a method to compute this statistical support, while keeping those revelant properties. The method exploits concentration inequalities for random variables a tool that has previously been helpful from both the theorical *and* pratical standpoints in other domains [16, 22, 28]. Moreover this method is near optimal from the statistical estimation standpoint. Our fourth contribution rely on how using these statistical supports to build borders in order to optimize an incremental update process. Finally we validate experimentally our approach. Experiments seem to display that the method is robust, regardless of the domain's complexity, the size of the data stored or the support values chosen. These are clear good points in favor of the applicability and scalability of the method.

There are a number of possible extensions to this work in many data mining researh fields. One very promising research direction would be to integrate our approach with those exploring data structures to maintains item that are observed as frequent with maximal recall [15]. Currently we are trying to address the following question: is it possible to build an intermediate set that keep as much as possible properties of these borders ? This set would represent a trade off between a too big statistical border to store and an other one for which needed database access while maintaining would be too important. In other words : how to build a border under a constraint of size that represent the best trade off between a recall `R` close to 1 and a precision `P` close to 1. A last research direction which is almost done will study how our method behaves when there is a *distribution drift* inside the stream.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large database. In *Proc. of ACM SIGMOD'93*, 1993.
2. M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Procs of the 29 th International Colloquium on Automata, Languages, and Programming*, pages 693–703, 2002.
3. X. Cheng, X. Yan, and J. Han. Incspan: Incremental mining of sequential patterns in large database. In *Proc. of KDD'04*, 2004.
4. D. Cheung, J. Han, V. Ng, and C. Wong. Maintenance of discovered association rules in large databases: an incremental update technique. In *Proc. of ICDE'93*, 1996.
5. Y. Chi, H. Wang, P.S. Yu, and R.R. Muntz. Moment: Maintaining closed frequent itemsets over a stream sliding window. In *Proc. of ICDM'04*, 2004.

6. G. Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proc. of the ACM International Conference on the Principles of Database Systems*, pages 296–306. ACM Press, 2003.
7. L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition.* Springer, 1996.
8. Carlos Domingo, Ricard Gavaldà, and Osamu Watanabe. Adaptive sampling methods for scaling up knowledge discovery algorithms. *Proc. of DMKD'02*, 6, 2002.
9. W. Fan, Y.-A. Huang, H. Wang, and P.-S. Yu. Active mining of data streams. In *Proc. of the SIAM International Conference on Data Mining*, pages 457–461, 2004.
10. G. Giannella, J. Han, J. Pei, X. Yan, and P. Yu. Mining frequent patterns in data streams at multiple time granularities. In *Next Generation Data Mining, MIT Press*, 2003.
11. P.-B. Gibbons and Y. Matias. New sampling-based summary statistics for improving approximate query answers. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 331–342, 1998.
12. L. Golab and M. Tamer Ozsu. Issues in data stream management. *ACM SIGMOD Records*, 2:5–14, 2003.
13. S. Gollapudi and D. Sivakumar. Framework and algorithms for trend analysis in massive temporal datasets. In *Proc. of the 13 th ACM International Conference on Information and Knowledge Management*, pages 168–177, 2004.
14. J. Han, J. Pei, B. Mortazavi-asl, Q. Chen, U. Dayal, and M. Hsu. Freespan: Frequent pattern-projected sequential pattern mining. In *Proc. of KDD'00*, 2000.
15. C. Jin, W. Qian, C. Sha, J.-X. Yu, and A. Zhou. Dynamically maintaining frequent items over a data stream. In *Proc. of CIKM'03*. ACM Press, 2003.
16. M. J. Kearns and Y. Mansour. A Fast, Bottom-up Decision Tree Pruning algorithm with Near-Optimal generalization. In *Proc. of ICML'98*, 1998.
17. H.-F. Li, S.Y. Lee, and M.-K. Shan. An efficient algorithm for mining frequent itemsets over the entire history of data streams. In *Proc. of the 1st Intl. Workshop on Knowledge Discovery in Data Streams*, 2004.
18. G. Manku and R. Motwani. Approximate frequency counts over data streams. In *Proc. of VLDB'02*, 2002.
19. F. Masseglia, F. Cathala, and P. Poncelet. The PSP approach for mining sequential patterns. In *Proc. of PKDD'98*, 1998.
20. F. Masseglia, P. Poncelet, and M. Teisseire. Incremental mining of sequential patterns in large databases. *Data and Knowledge Engineering*, 46, 2003.
21. D. McAllester. Some PAC-Bayesian theorems. *Machine Learning*, 37:355–363, 1999.
22. R. Nock and F. Nielsen. Statistical Region Merging. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2004. to appear.
23. S. Parthasarathy, M. Zaki, M. Orihara, and S. Dwarkadas. Incremental and interactive sequence mining. In *Proc. of CIKM'99*, 1999.
24. Tobias Scheffer and Stefan Wrobel. Finding the most interesting patterns in a database quickly by using sequential sampling. *Proc. of JMLR'02*, 3, 2002.
25. R. Agrawal R. Srikant. Mining sequential patterns. In *Proc. of ICDE'95 Conference*, 1995.
26. W.-G. Teng, M.-S. Chen, and P.S. Yu. A regression-based temporal patterns mining schema for data streams. In *Proc. of VLDB'03*, 2003.
27. Hannu Toivonen. Sampling large databases for association rules. In *Proc. of VLDB'96*, 1996.
28. V. Vapnik. *Statistical Learning Theory.* John Wiley, 1998.

29. A. Veloso, W. Meira, M. Carvalho, B. Possas, S. Parthasarathy, and M.-J. Zaki. Mining frequent itemsets in evolving databases. In *Proc. of the* 2^{nd} *SIAM International Conference on Data Mining*, pages 31–41, 2002.
30. D. Wettschereck and D. Aha. Mining concept-drifting data streams with ensemble classifiers. In *Proc. of KDD'03*, 2003.
31. J. Yang, W. Wang, P.-S. Yu, and J. Han. Mining long sequential patterns in a noisy environment. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 406–417, 2002.
32. M.J. Zaki. SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning Journal*, 42, 2001.
33. Q. Zheng, K. Ksu, S. Ma, and W. Lv. The algorithms of updating sequential patterns. In *Proc. of ICDM'02*, 2002.
34. Q. Zheng, K. Xu, S. Ma, and W. lv. The algorithms of updating sequential patterns. In *Proc. of ICDM'02*, 2002.

7 Appendix

We prove Theorem 3. We make the assumption that X^*_θ is a singleton, and θ will be chosen in $(1/2, 1]$: there exists a single θ-frequent itemset T. We also suppose that there are two itemsets in X with respective weight θ (this is T) and $1-\theta$. Given that we sample independently in S the data stream for m itemsets, there is a probability $\geq \eta$ to observe $\rho_S(T) < \rho_X(T) - \varepsilon$, with:

$$\eta = \binom{m}{m(\theta-\varepsilon)}(1-\theta)^{m(1-\theta+\varepsilon)}\theta^{m(\theta-\varepsilon)} \ , \tag{16}$$

and $\binom{m}{k} = m!/((m-k)!k!)$ the binomial coefficient. In fact, we could have used for η the tail of the Binomial distribution from the terms $k < m(\theta-\varepsilon)$, and this would yield a bound for η stronger than that of eq. (16). For the sake of readability, we abbreviate $f(m,\theta,\varepsilon)$ the right-hand side of eq. (16). We make use of the following well known Stirling-type inequalities:

$$\sqrt{2n\pi}(n/e)^n \leq n! \leq \exp(1/(12n))\sqrt{2n\pi}(n/e)^n \ .$$

We obtain the following lowerbound on $f(m,\theta,\varepsilon)$:

$$\begin{aligned} f(m,\theta,\varepsilon) \geq \exp\Bigg(&-\frac{1}{12my(1-y)} - \frac{1}{2}\ln(2\pi m y(1-y)) \\ &-m\left[(1-y)\ln\frac{1-y}{1-\theta} + y\ln\frac{y}{\theta}\right]\Bigg) \ . \end{aligned}$$

Here, we have made use of the shorthand $y = \theta - \varepsilon$, which we suppose to be $\in [0,1]$. The quantity inside the brackets is a Kullback-Leibler divergence, which can be upperbounded with the relationship $\ln(x) \leq x - 1$ by:

$$(1-y)\ln\frac{1-y}{1-\theta} + y\ln\frac{y}{\theta} \leq \frac{(\theta-y)^2}{\theta(1-\theta)} \ . \tag{17}$$

Provided m is not too small (in particular, $m \geq \max\{4\pi^2, 1 + 1/(3y(1-y))\}$), we may obtain:

$$f(m, \theta, \varepsilon) \geq \exp\left(-m\frac{\varepsilon^2}{\theta(1-\theta)} - \ln m\right) \ .$$

Now, provided:

$$\varepsilon \geq \sqrt{\frac{\theta(1-\theta)}{m} \ln m} \ , \tag{18}$$

we finally obtain $f(m, \theta, \varepsilon) \geq \exp(-2m\varepsilon^2/(\theta(1-\theta)))$. We shall clearly have $f(m, \theta, \varepsilon) \geq \delta$ provided:

$$\varepsilon = \sqrt{\frac{\theta(1-\theta)}{2m} \ln \frac{1}{\delta}} \ , \tag{19}$$

which satisfies eq. (18) whenever $\delta \leq 1/m^2$. Choosing θ close to 1/2 brings the statement of Theorem 3.