Discovering Knowledge from Local Patterns with Global Constraints

Bruno Crémilleux[1] and Arnaud Soulet[2]

[1] GREYC-CNRS, Université de Caen
Campus Côte de Nacre
F-14032 Caen Cédex France
bruno.cremilleux@info.unicaen.fr
[2] LI, Université François Rabelais de Tours
3 place Jean Jaurès
F-41029 Blois France
arnaud.soulet@univ-tours.fr

Abstract. It is well known that local patterns are at the core of a lot of knowledge which may be discovered from data. Nevertheless, use of local patterns is limited by their huge number and computational costs. Several approaches (e.g., condensed representations, pattern set discovery) aim at selecting or grouping local patterns to provide a global view of the data. In this paper, we propose the idea of global constraints to write queries addressing global patterns as sets of local patterns. Usefulness of global constraints is to take into account relationships between local patterns, such relations expressing a user bias according to its expectation (e.g., search of exceptions, top-k patterns). We think that global constraints are a powerful way to get meaningful patterns. We propose the generic Approximate-and-Push approach to mine patterns under global constraints and we give a method for the case of the top-k patterns w.r.t. any measure. Experiments show its efficiency since it was not feasible to mine such patterns beforehand.

Keywords: data mining, local patterns, constraint-based paradigm, global constraints, top-k patterns.

1 Introduction

In current scientific, industrial or business areas, the critical need is not to generate data, but to derive knowledge from huge datasets produced at high throughput. Extracting or mining knowledge from large amounts of data is at the core of the Knowledge Discovery from Data task, often also named "data mining". This involves different challenges, such as designing efficient tools to tackle data and the discovery of patterns of a potential user's interest. The constraint-based pattern mining framework is a powerful paradigm to discover new highly valuable knowledge [16]. Constraints provide a focus on the most promising knowledge by reducing the number of extracted patterns to those of a potential interest given by the user. There are now generic approaches to discover *local patterns*

O. Gervasi et al. (Eds.): ICCSA 2008, Part II, LNCS 5073, pp. 1242–1257, 2008.

under constraints [7, 20]ăand this issue is rather well-mastered, at least for the data described by items (i.e., boolean attributes). A Dagstuhl seminar has been devoted on the topic of local patterns in order to provide a clearer view of this new field [15] and a Definition of local patterns is given in Section 2.1.

Nevertheless, even if the number of produced local patterns is reduced thanks to the constraint, the output still remains too large for an individual and global analysis performed by the end-user. The most significant patterns are lost among too many trivial, noisy and redundant information. Many works propose methods to reduce the collection of patterns, like the so-called condensed representations [5] or the compression of the dataset by exploiting Minimum Description Length Principle [19] but most of them are dedicated to the particular case of frequent patterns (a pattern X is said *frequent* if the number of examples in the database supporting X exceeds a given threshold). Recent approaches - pattern teams [12], constraint-based pattern set mining [8] and selecting patterns according to the added value of a new pattern given the currently selected patterns [2] - aim at reducing the redundancy by selecting patterns from the initial large set of patterns on the basis of their usefulness in the context of the other selected patterns. Even if these approaches explicitly compare patterns between them (as the global constraints, see Section 3), they are mainly based on the reduction of the redundancy and they cannot take into account in a flexible way a bias given by the user to direct the final set of patterns toward a specific aim such as finding the best patterns according to an interestingness measure or the search of exceptions. In this paper, we call *global pattern* a set of local patterns satisfying a property involving several local patterns (cf. Section 2).

On the other hand, it should be a pity to consider the summarization of local patterns only from the point of view of the redundancy. We think that local patterns are also suitable to revisit classical data exploration tasks as for example classification or clustering and more generally to produce global patterns as models. Assuming that the user would like to produce a classifier based on rules. A complete and correct constraint-based data mining method ensures to produce a proper set of classification rules. It differs for instance from the decision trees technique where the attribute selection criterion ensures to pick the best attribute at each node but it does not guarantee the best whole tree [4]. Obviously, having a proper set of classification rules does not straightforwardly provide a relevant classifier. More generally, it is clear that moving from local patterns to global patterns as models like classifiers is still a challenge. Studying carefully the complementarity of the various pattern domains (e.g., clustering and local pattern discovery, local pattern discovery as feature construction for supervised classification) is needed.

This paper addresses this general issue by using the constraint paradigm. Our first contribution is to propose the notion of *global constraints* as a flexible and declarative way to design global patterns as sets of local patterns. This approach enables the user to express a bias and discover richer global patterns. Second, we provide the generic Approximate-and-Push approach to mine patterns satisfying

global constraints. We give a method for mining the top-k patterns w.r.t. any measure and we show its efficiency in practice.

This paper is organized as follows. Section 2 outlines basic definitions and several examples of global patterns as set of local patterns. We will see that the current techniques are specific to the kind of targeted global patterns. In Section 3, we propose the notion of global constraints and we show how ităenables us to address a lot of global patterns. We present in Section 4 our generic Approximate-and-Push approach to mine patterns under global constraints and the case of the top-k patterns w.r.t. any measure. In Section 5, we discuss of the usefulness of our approach to design global patterns as models. Finally, Section 6 concludes.

2 Basic Definitions and Examples of Global Patterns

We give below basic definitions used among this paper. Then we sketch examples of methods providing global patterns coming from local patterns.

2.1 Basic Definitions

Let $\mathcal{I}$ be a set of distinct literals called items, an itemset (also called *pattern*) corresponds to a non-null subset of $\mathcal{I}$. The language of patterns corresponds to $\mathcal{L}_\mathcal{I} = 2^\mathcal{I} \backslash \emptyset$. A transactional dataset $\mathcal{D}$ is a multi-set of patterns of $\mathcal{L}_\mathcal{I}$ usually called transactions. For instance, Table 1 gives a transactional dataset $\mathcal{D}$ where 9 transactions $t_1, \ldots, t_9$ are described by 6 items $A, \ldots, C_2$.

Table 1. Example of a transactional context $\mathcal{D}$

$\mathcal{D}$

Trans.	Items					
t_1	A	B			C_1	
t_2	A	B			C_1	
t_3			C		C_1	
t_4			C		C_1	
t_5			C		C_1	
t_6	A	B	C	D		C_2
t_7			C	D		C_2
t_8			C			C_2
t_9				D		C_2

Local patterns are regularities that hold for a particular part of the data. A local pattern is of special interest if it exhibits a deviating behavior w.r.t. the underlying global model of the data [11] because we are seeking for surprising knowledge which deviates from the already known background model. But local patterns can also be considered as fragmented and incomplete knowledge conveying some aspect of the data. As introduced in Section 1, a challenge is then

to gather the pieces of the puzzle to turn *global patterns*, i.e., sets of patterns satisfying a property involving several local patterns. Such global patterns provide summarizations of the data.

Let X be a pattern. The constraint-based pattern mining framework aims at discovering all the patterns of $\mathcal{L}_{\mathcal{I}}$ satisfying a given predicate q, named *constraint*, and occurring in $\mathcal{D}$. A well-known example is the frequency constraint focusing on patterns having a frequency in the database exceeding a given minimal threshold $minfr > 0$: $freq(X) \geq minfr$. For instance, AB is a frequent pattern with $minfr = 3$ because $freq(AB) = 3$. There are a lot of constraints to evaluate the relevance of local patterns. Examples of various and powerful constraints can be found in [16, 20]. We call *local constraint* such constraints because they concern only one pattern contrary to the global constraints that we propose in Section 3. The constraint paradigm also uses interestingness measures (the frequency is an example) to select local patterns. Another example of measure is the area of a pattern $area(X)$: it corresponds to the frequency of the pattern times its length (i.e., $area(X) = freq(X) \times count(X)$ where $count(X)$ denotes the cardinality of X). Previous examples are local constraints and Section 3 depicts examples of global constraints. We will come-back on the area as an example of an interestingness measure that can be used in the top-k global constraint (see Section 3).

Mining patterns under constraints requires the exploration of the search space depicted by $\mathcal{L}_{\mathcal{I}}$. Unfortunately, this space is generally huge and thus, pruning conditions are necessary. The property of monotonicity is well-used because pruning conditions are straightforwardly deduced [14]. A constraint q is anti-monotone w.r.t. the item specialization (resp. monotone) iff for all $X \in \mathcal{L}_{\mathcal{I}}$ satisfying q, any subset (resp. superset) of X also satisfies q. For instance, the minimal support constraint is anti-monotone. Whenever a pattern X is frequent, any subset of X is also frequent.

2.2 Global Patterns as Sets of Local Patterns

This section gives examples of methods producing the sets of all patterns satisfying a property involving several local patterns. Have a look on characterization rules which are an important issue in a lot of applications. The usefulness of the simplest rules (i.e., rules having minimal premises) for this task is shown in [6]. Basically, given a bounded number of exceptions δ, a rule has a minimal premise if any rule with a proper subset of its premise and less than δ exceptions does not enable to conclude on the same class value (Section 3 gives a formal definition highlighting the comparison between two local patterns). Such rules are inferred from a specific subset of patterns satisfying a property of δ-freeness [3]. Prototypes extracting the condensed representations of the δ-free patterns can be updated to extract the correct and complete collection of characterization rules as defined above [6]. Unfortunately, this approach cannot be extended if we are interested in other kinds of characterization rules.

The discovery of exception rules from a data set without domain-specific information is also of a great interest [22]. An exception rule is defined as a deviational

pattern to a strong rule and the interest of a rule is evaluated according to another rule. Again, the comparison between rules means that these exception rules are not local patterns. Suzuki proposes a method based on sound pruning and probabilistic estimation [22] to extract all the exception rules. But, as previously, this approach is devoted to this kind of patterns.

Another example is the top-k patterns [10]. These patterns are the k patterns optimizing an interestingness measure m (e.g., frequency, area). The collection of the top-k patterns is concise and gathers the most relevant patterns according to m. It can be seen as a global pattern. Discovering the top-k patterns optimizing a measure m becomes difficult as soon as m has no suitable property like anti-monotonicity. Naive post-processing approaches extract in a first step all patterns whose the value of m exceeds a threshold, then the k patterns optimizing m are selected. But, in practice, the choice of the threshold is too subtle. A too high threshold may lead to miss patterns satisfying the top-k constraint, a too small may generate thousands of patterns giving intractable computations. For instance, a naive method for mining the top-k patterns w.r.t. area requires to extract all present patterns because even patterns whose frequency value is 1 may satisfy the area constraint. A branch-and-bound algorithm for top-k patterns is proposed in [8], but only few measures (stemming from upper-boundable constraints) are tackled.

In the next section, we show that all of these global patterns (and many others) can be easily expressed with global constraints. In Section 4.2, we provide a method belonging to our generic Approximate-and-Push approach which efficiently mines the top-k patterns w.r.t. any measure (whereas the works on top-k patterns in the literature are limited to the frequency measure [10] and specific forms of the patterns).

3 Global Constraints for Mining Global Patterns

We propose in this section the notion of *global constraint* which formalizes the building of the global patterns previously introduced. Global constraints consider simultaneously at least two patterns contrary to local constraints which are checked individually on each pattern (see Section 2.1). By comparing several patterns, global characteristics from the database can be revealed.

Definition 1 (Global constraint). *A constraint q is said global if several patterns have to be compared to check if q is satisfied or not.*

A global constraint satisfaction is based on information coming from at least two patterns. This point is highlighted in the examples given below by the two patterns X and Y. For instance, several constraints (e.g., freeness, characterization rules, exception rules) require to check properties of the subsets Y of X. Peak constraint compares two neighbor patterns and top-k any pair of patterns. This definition of global constraints enables to write properties both minimizing the redundancy (e.g., freeness, characterization rules) and directed toward a bias given by the user (e.g., minimal premise with the characterization rules,

unexpected information with the exception rules, exceptional behavior with the peak constraint).

We present now a non-exhaustive list of global constraints illustrating Definition 1 in the context of itemset mining. All the examples given in Section 2.2 are written with the formalism of the global constraints. Some of them are very well-known, like freeness or closedness. We also propose the peak constraint which highlights an exceptional behavior of a pattern with respect to its neighbors.

- **Condensed representations (freeness and closedness):** A free (resp. closed) pattern is a minimal (resp. the maximal) pattern of its equivalence class of frequency [3, 17] (the frequency has been introduced in Section 2.1). This kind of patterns is useful to design condensed representations or to derive association rules. Freeness and closedness require to check the frequency of X with respect to the frequencies of its subsets.

$$freeness(X) \equiv \begin{cases} true & \text{if } \forall Y \in \mathcal{L}_{\mathcal{I}} \text{ such that } Y \subset X, \text{ one have } freq(X) < freq(Y) \\ false & \text{otherwise} \end{cases}$$

$$closedness(X) \equiv \begin{cases} true & \text{if } \forall Y \in \mathcal{L}_{\mathcal{I}} \text{ such that } X \subset Y, \text{ one have } freq(X) > freq(Y) \\ false & \text{otherwise} \end{cases}$$

  For instance, AC satisfies the $freeness$ constraint in dataset $\mathcal{D}$ given by Table 1 but not ABC since $freq(ABC) = freq(AB)$. Dually, $ABCD$ satisfies the *closedness* constraint but not ABC ($freq(ABC) = freq(ABCD)$).
- **Characterization rules:** A formulation of characterization rules is given in [6]. It can be defined by the following global constraint:

$$characterization(X \rightarrow c) \equiv \begin{cases} true & \text{if } \forall Y \in \mathcal{L}_{\mathcal{I}} \text{ such that } Y \subset X, \text{ one have} \\ & \quad X \rightarrow c \wedge \neg(Y \rightarrow c) \\ false & \text{otherwise} \end{cases}$$

  where $X \rightarrow c$ denotes a rule with $freq(X \rightarrow c) \geq \gamma$ and $freq(X \rightarrow \neg c) \leq \delta$. It means that $X \rightarrow c$ is a rule with a minimal premise to conclude to c with lower than δ exceptions. For instance, with $\gamma = 1$ and $\delta = 1$, the rule $A \rightarrow C_1$ is minimal and has only one exception in dataset $\mathcal{D}$ (see Table 1). Then, $A \rightarrow C_1$ satisfies the constraint *characterization*.
- **Exceptions:** Suzuki [22] defines exception rules which isolate unexpected information as the following set or rules (I is an item):

$$exception(X \rightarrow \neg I) \equiv \begin{cases} true & \text{if } \exists Y \in \mathcal{L}_{\mathcal{I}} \text{ such that } Y \subset X, \text{ one have} \\ & \quad X \backslash Y \rightarrow I \wedge X \rightarrow \neg I \wedge Y \not\rightarrow \neg I \\ false & \text{otherwise} \end{cases}$$

  The rule $X \rightarrow \neg I$ is an exception rule since usually if $X \backslash Y$ then I and if Y then not frequently $\neg I$. Let us illustrate this notion by assuming that a rule $X \rightarrow Y$ holds iff at least 2/3 of transactions containing X also contains Y. Following our running example, $AC \rightarrow \neg C_1$ is an exception rule because we jointly have $A \rightarrow C_1$ and $C \not\rightarrow \neg C_1$.

- **Top-k constraint:** Let $k > 0$ be an integer and $m : \mathcal{L}_{\mathcal{I}} \rightarrow \Re$ be a measure (e.g., frequency [10]). The top-k constraint w.r.t m are the k best patterns according to m and it equals to:

$$top_{k,m}(X) \equiv |\{Y \in \mathcal{L}_{\mathcal{I}} | Y \neq X \wedge m(Y) > m(X)\}| < k$$

  Typically, by ignoring class values C_1 and C_2 in dataset $\mathcal{D}$, the top-2 patterns according to the area measure are C ($area(C) = 6$) and AB ($area(AB) = 6$). Indeed, all the other patterns have an area lower than 6.
- **Peak constraint:** A peak pattern is a pattern whose neighbors have a value for a measure m lower than a threshold. It means that a peak pattern has an exceptional behavior compared to its neighbors. Let $m : \mathcal{L}_{\mathcal{I}} \rightarrow \Re$ be a measure, d be a distance (e.g., $d(X, Y) = |X \backslash Y| + |Y \backslash X|$), δ be an integer and ρ be a real, one have:

$$peak(X) \equiv \begin{cases} true & \text{if } \forall Y \in \mathcal{L}_{\mathcal{I}} \text{ such that } d(X,Y) < \delta, \text{ one have } m(X) \geq \rho \times m(Y) \\ false & \text{otherwise} \end{cases}$$

  For instance, according to Table 1 and not considering C_1 and C_2, the pattern AB is a peak with $\delta = 1$ and $\rho = 1.5$. More precisily, the area of AB ($= 6$) exeeds those of all its 4 neighbors (i.e., $area(A) = 3$, $area(B) = 3$, $area(ABC) = 3$ and $area(ABD) = 3$ and $3 \times \rho \leq 6$).

Among others, all examples of global patterns given in Section 2.2 can be straightforwardly expressed with global constraints. We think that this unified approach brought by the global constraints improves the understandability of the global patterns. Furthermore, this approach enables to easily design new global patterns like peaks, but many other examples can be addressed.

Global constraints provide a reduced collection of patterns forming a coverage or gathering the best ones according to a bias or a criterion expressing the interest of the user. The *covering constraints* synthesize the whole set of local patterns by deleting redundancies: freeness, closedness, characterization rules. The *optimizing constraints* optimize a given criterion over some patterns (i.e., exceptions, peak constraint) or over all the patterns (i.e., the $top_{k,m}$ constraint). There is no strict border between these two categories: for instance, a characterization rule also satisfies an optimizing constraint, i.e., a maximal number of exceptions.

4 The Approximate-and-Push Approach and the Case of the Top-k Patterns

This section proposes the generic Approximate-and-Push approach for mining patterns satisfying global constraints. We start by giving the key ideas of the approach and then we show how it performs in the case of the top-k patterns.

4.1 Overview of the Approximate-and-Push Approach

Basically, the idea is to automatically deduce local constraints from a global constraint. These local constraints are relaxations of the global one and can be pushed in the mining step. They are dynamically refined during the process to be more and more efficient. As suggested by its name, the Approximate-and-Push approach is divided in two steps which are iteratively repeated.

Approximate. This step provides a collection of candidate patterns which is an approximation of the final collection of patterns satisfying the global constraint. The initialization of the method has to guarantee that this approximation ensures to mine all patterns satisfying the global constraint. In the case of the top-k patterns, we will see in the next section that preserving *the* k best already extracted patterns is enough. Then, at each approximate step, a pattern X is tested to know if it has to be included in the collection of candidate patterns. X is added in the collection if it may satisfy the global constraint with regard to the other patterns already present in the collection. Candidate patterns of the collection are removed when a new tested pattern ensures that they cannot satisfy the global constraint.

Push. The aim of this step is to deduce from the global constraint, pruning conditions of the search space which benefit from the characteristics of the approximation, i.e., the collection of candidate patterns. Pruning conditions may be relaxations to get suitable properties like (anti-)monotonicity which are efficiently pushed by usual pattern mining algorithms. The next section details this process in the case of the top-k patterns. Moreover, when new patterns enter in the collection of candidate patterns, these pruning conditions become more efficient. This is clearly the main interest of the iterative process of the Approximate-and-Push approach.

The Approximate-and-Push approach can be performed with any pattern mining algorithm, provided that the pruning conditions which are inferred are useful for the selected algorithm. Furthermore, the Approximate-and-Push approach is easily extendable to other languages (sequences, trees, etc). In the next section, we show how an Apriori-like algorithm can be efficiently reused by the Approximate-and-Push approach to mine the top-k patterns w.r.t. any measure.

4.2 Mining the Top-k Patterns w.r.t. a Measure

This section shows how to push the $top_{k,m}$ constraint into the mining step thanks to the Approximate-and-Push approach. Besides its efficiency, one of the original features of our method is to deal with measures without properties of (anti)-monotonicity (i.e., such measures do not provide efficient pruning conditions for usual algorithms to mine patterns satisfying the constraint). To the best of our knowledge, our method is the first one to mine the top-k patterns w.r.t. any measure.

Principles of the Method. The definition of $top_{k,m}$ given in Section 3 does not straightforwardly provide a local constraint which can be pushed by an usual pattern mining algorithm. We reformulate the $top_{k,m}$ constraint to get a relevant local constraint. First, we introduce the threshold $\rho_{k,m}$ corresponding to $\min\{m(X)|X \in \mathcal{L}_{\mathcal{I}} \wedge top_{k,m}(X)\}$. Then, we can rewrite the $top_{k,m}$ constraint as $top_{k,m}(X) \equiv m(X) \geq \rho_{k,m}$. This last constraint is still a global one because the threshold $\rho_{k,m}$ implicitly implies comparisons between patterns. But, fixing this threshold makes possible the definition of relevant local constraints as we will see below. The key idea is to start by fixing an arbitrary threshold, then it will be refined to improve the efficiency of the pruning conditions coming from the local constraint thanks to the iterative process of the Approximate-and-Push approach.

Approximating the top-k patterns. As already said, this step preserves the approximation which is the k patterns $\mathcal{C}and$ optimizing the measure m from the patterns already mined. This ensures the correctness and the completeness of the result at the end of the scanning of the search space. We define an *adding threshold* ρ to decide if an applicant pattern has to be added or not in the collection $\mathcal{C}and$ (the following definition assumes that m has to be maximized, if m has to be minimized, it is enough to change the sign of m).

$$\rho = \begin{cases} -\infty, & \text{if } |\mathcal{C}and| < k \\ \min_{X \in \mathcal{C}and} m(X), & \text{otherwise} \end{cases}$$

An applicant pattern is added to $\mathcal{C}and$ if and only if its measure m is greater than or equal to the adding threshold ρ. At the beginning, there is a filling phase (i.e., $\rho = -\infty$) and all the mined patterns are added until $\mathcal{C}and$ has k candidate patterns. Thereafter, the addition of an applicant pattern depends on ρ which provides the minimum value of m of the k best current patterns. A pattern X is deleted from $\mathcal{C}and$ when k other patterns of $\mathcal{C}and$ have a better measure than $m(X)$. The great interest of this approach is that the adding threshold will increase ($\rho_{k,m}$ is its optimal value) according to the updating of the $\mathcal{C}and$ leading to more and more powerful pruning conditions in the push step.

Table 2. The top-2 pattern w.r.t the area with Apriori algorithm

Level 1

Pattern	$\mathcal{C}and$	ρ
A (3)	A	$-\infty$
B (3)	A, B	3
C (6)	C, A, B	3
D (3)	C, A, B, D	3

Level 2

Pattern	$\mathcal{C}and$	ρ
AB (6)	C, AB	6
AC (2)	C, AB	6
AD (2)	C, AB	6
BC (2)	C, AB	6
BD (2)	C, AB	6
CD (4)	C, AB	6

Table 2 depicts the modifications of candidate patterns $\mathcal{C}and$ during the mining of the constraint $top_{2,area}$ with a levelwise approach (as performed by Apriori [1]) in the transactional context provided by Table 1 (ignoring class values C_1 and C_2). The levelwise algorithm generates two successive sets of applicant patterns. For each level, patterns whose area is greater than ρ (bold patterns) enter in the collection of candidate patterns. The value of area is specified between brackets in the left column and the candidate patterns are gathered in the central column. As explained above, the threshold ρ (the right column) is progressively adjusted. While the number of patterns of $\mathcal{C}and$ is less than k, the threshold ρ remains $-\infty$. After, ρ corresponds to the minimal area satisfying by at least one pattern of $\mathcal{C}and$. The pattern A is not excluded when B enters in the collection because its area equals ρ. On the contrary, A, B and D are deleted when the pattern AB arrives in $\mathcal{C}and$ because then $\rho = 6$. At the end of the last level, $\mathcal{C}and$ contains exactly C and AB which are the two patterns satisfying $top_{2,area}$ (i.e., the two patterns having the best area).

Pushing the approximation. Suitable pruning conditions can be inferred from the collection of candidate patterns $\mathcal{C}and$ and the adding threshold. As $\mathcal{C}and$ preserves the $top_{k,m}$ constraint and any pattern in $\mathcal{C}and$ has a value of m greater than or equal to ρ, we can deduce that a pattern X must satisfy the local constraint $m(X) \geq \rho$ to be introduced into $\mathcal{C}and$. Unfortunately, this constraint does not straightforwardly provide suitable pruning conditions. For instance, if we come back to the area measure, $area(X) \geq \rho$ does not satisfy (anti-)monotone properties. For that purpose, we propose to relax the constraint $m(X) \geq \rho$ by a *relaxation* $m'(X) \geq \rho$ satisfying the two following conditions: $m'(X) \geq \rho$ is anti-monotone (condition $C1$) and $\forall X$, $m(X) \geq \rho \Rightarrow m'(X) \geq \rho$ (condition $C2$). Condition $C2$ ensures the completeness of the mining process. Finding an anti-monotone relaxation $m'(X) \geq \rho$ is not a trivial task. An automated and generic method taking into account any measure, which can be decomposed according to primitives, is given in [21]. Let us illustrate this step w.r.t. the area measure. If the k candidate patterns in the approximation have an area higher or equal to A and the size of the longest transaction in the database is L (so a pattern cannot be longer than L), we can deduce that the frequency of any top-k pattern X w.r.t. the area measure must be higher or equal to $\frac{A}{L}$ because $area(X) = freq(X) \times count(X)$. The key point is that we get a pruning condition (i.e., $freq(X) < \frac{A}{L}$) which corresponds to an anti-monotone constraint and thus can be efficiently pushed [16]. In other words, we have deduced from the global constraint a local constraint (here, $freq(X) \geq \frac{A}{L}$) leading to a pruning condition for the usual levelwise mining algorithms. Moreover, when new patterns enter in the collection of candidate patterns, the adding threshold can only increase (following our example, L is fixed and A can only increase) and this pruning condition becomes more and more efficient. Experiments described below demonstrate the practical efficiency of this process.

Let us come back on our running example of top-2 patterns w.r.t area given by Table 2. The used anti-monotone relaxation is $freq(X) \geq \rho/L$ (here $L = 4$ because the size of the longest transaction is 4). At the end of the first level,

$\rho = 3$ and the relaxation $freq(X) \geq 3/4$ eliminates no pattern. So, all the patterns containing two items and present in the context $\mathcal{D}$ are generated. On the contrary, at the end of the second level, $\rho = 6$ and the relaxation becomes $freq(X) \geq 6/4$. For the level 3, the generated patterns must have a frequency greater than or equal to 2 (none pattern has a sufficient frequency). The mining process stops at level 3 because no more pattern can be generated. Finally, the Approximate-and-Push approach avoids the generation of 4 patterns having a length of 3 and 1 pattern of length 4.

4.3 Experiments

The aim of the experiments is to show and quantify the efficiency of our method. Recalling that mining $top_{k,m}$ patterns w.r.t. any measure was not feasible beforehand, except by using a rather naive post-processing technique ($top_{k,m}$ patterns are filtered in a second time). Furthermore, as explained below, $top_{k,m}$ patterns may be missed by this technique!

In the following, **Approximate-and-Push** means our method described above and using an APRIORI-like algorithm to push the anti-monotone constraint automatically deduced. As there is no other generic approaches in the literature, we think that demonstrating the feasibility of mining the top-k patterns w.r.t. measures like area (which do not check good properties like (anti-)monotonicty) is itself a first interesting result of these experiments. Nevertheless, to go further and better understand the behavior and the efficiency of Approximate-and-Push, we compare it with two other specific strategies also based on APRIORI algorithm:

- **Optimal:** this strategy exploits the *optimal* anti-monotone relaxation of $m(X) \geq \rho_{k,m}$. We fix for the threshold $\rho_{k,m}$ the value optimizing the pruning condition without loss of top-k patterns. This is ideal but unfeasible in practice because this threshold cannot be known by the user. Nevertheless, the interest of this experiment is this strategy enables us to compare Approximate-and-Push w.r.t. the optimal situation pushing the constraint.
- **Post-processing:** until now, this is the single strategy to mine the top-k patterns w.r.t. measures like area, even if such an approach is not safe as we show below. In a first step, all patterns having a frequency higher than 10% are mined. Then, the k best patterns w.r.t. the measure are selected. The frequency threshold (here, 10%) is fixed by the user. In practice, the user makes a trade-off between tractability and completeness (avoid missing top-k patterns) according to his intuition. But, this method *cannot* be used safely. For instance, it can happen that a long pattern having a frequency below the frequency threshold (consequently not mining) has an area higher than a shorter and frequent pattern. Let us recall that Approximate-and-Push is sound and complete.

We use two datasets (`mushroom` and `letter` [1]) and two measures, the frequency and the area. We also performed experiments with other measures and

[1] `www.ics.uci.edu/~mlearn/MLRepository.html`

datasets, these experiments gave similar results. For each dataset, we mine the top-k patterns w.r.t these measures. All the experiments are done with the same APRIORI implementation and we can compare the different running-times. Experiments were conducted on an Xeon 2.2 GHz with 3GB of RAM memory running the Linux system. Figure 1 reports the extraction running times according to k, i.e., the number of desired patterns.

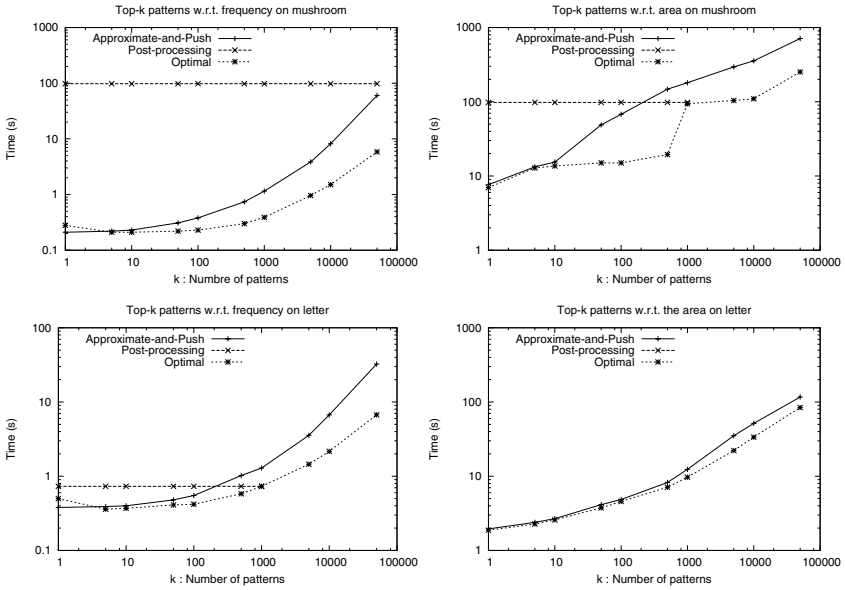

Fig. 1. Running-times for mining the top-k patterns

The Post-processing strategy has a specific behavior because its running-time is a constant which does not depend on the value k for a experiment. It suffers from two main drawbacks. When k is small, its running time is much longer than the other ones. More serious, this strategy fails for some values of k because it misses top-k patterns. In these cases, curves in Figure 1 are interrupted. With the area measure on `letter`, this strategy systematically returns false top-k patterns for any k! That it is why the curve for the Post-processing strategy is not plotted in this figure. These results recall that the Post-processing strategy is not safe.

Figure 1 indicates that the shapes of the Approximate-and-Push and Optimal curves are similar. The higher k is, the longer the running-time. Obviously, the Optimal strategy is better than the Approximate-and-Push strategy. But recalling that the Optimal strategy cannot be used in practice because it is impossible to guess the suitable threshold which has to be used to achieve the optimality. A major lesson is that the Approximate-and-Push curve is not so far from the Optimal curve (see for instance results on `letter`). Approximate-and-Push is especially efficient (and very close to the optimality) when k is small. This is due

to a fast filling phase of candidate patterns which provides quickly an efficient local constraint. This result is important because the user often fixes low values for k in order to obtain a small and analyzable output.

5 Future Issue: Designing Global Patterns as Models

A lot of global models which are the expected results of a data mining process can come from global patterns. Intuitively, we can define a *model* as a generalization of global patterns extracted from an application domain. A typical example is the associative classification. This technique proposes the integration of association rule mining and classification [13]. As the number of potential classification rules is very large, pruning techniques are used to generalize the information conveyed by the rules and get a well-suited classifier on new examples. Rules which are redundant from a functional point of view or may cause incorrect classification are deleted. Pruning is usually applied as a post-processing step on the extracted rules by using statistical parameters such as support, confidence and chi-square test. A more recent approach finds the best k correlated association rules for classification by using a measure which has suitable pruning properties [24]. A common characteristic of all these methods is to use heuristics techniques to select the rules from a complete collection of local patterns to produce a classifier.

Associations, like the frequent patterns, are also at the core of clustering works [23]. For instance, ECCLAT [9] is based on frequent closed patterns and has the originality to enable a slight overlapping between clusters. A global pattern produced by ECCLAT is performed by a greedy method in which the interest of a cluster is evaluated according to an interestingness measure.

Interestingly, global patterns can capture the joint effect of local patterns. The co-classification is a way of conceptual clustering and provides a limited collection of bi-clusters. These bi-clusters are linked for both objects and attribute-value pairs. In [18], the authors propose a generic framework for co-classification. Its great interest is that the bi-partition comes from a reconstruction of the objects and attributes defining the local patterns: the bi-clusters of the final bi-partition (i.e., a global pattern) are not necessary elements of the initial set of the local patterns. Nevertheless, a distance between the bi-sets which are at the origin of the bi-clusters has to be chosen.

We see that there are several techniques to infer global patterns and models from local patterns. Unfortunately, combinations of local patterns are ad hoc to a specific goal and often use heuristics or parameters which have to be set by the user. That is why the question of how to turn large collections of local patterns into global models deserves attention. We think that global constraints enable us to significantly go further in this direction by providing appropriate sets of patterns easier to manage than the local patterns currently used. Indeed, patterns addressing by global constraints reveal relationships and a global structure inside the data, they are noticeably less numerous and they can be oriented toward a bias given by the user. We think that global constraints enable us to get a sound basis of patterns to design models.

6 Conclusion

We think that designing and discovering global patterns and models are crucial goals for the end-users. In this paper, we have proposed the notion of global constraints to address this general issue. By comparing local patterns between them, global constraints take into account relationships in the data. We have shown that global constraints are a flexible and declarative way to define a lot of global patterns (e.g., condensed representations, exceptions, top-k patterns, characterization rules). This approach also enables the user to express a bias and discover meaningful global patterns. In the literature, only few global patterns can be mined and only by using dedicated algorithms mainly relying on anti-monotone pruning. In this paper, we have proposed the generic Approximate-and-Push approach and we have given an efficient method for mining the top-k patterns w.r.t. a measure.

In the future, we would like to perform further experiments with other global constraints using the Approximate-and-Push approach. It would also be interesting to evaluate the quality of such new discovered patterns in real-world applications. Furthermore, a promising open avenue is to design high-level constraints to directly build global models, it would avoid the choice of heuristics, as it is the case in the current methods.

Acknowledgments. The authors would like to thank the participants of the Dagstuhl seminar on Parallel Universes and Local Patterns for enlightening on the subjects of this paper. Special thanks to Johannes Fürnkranz, Arno J. Knobbe and Martin Scholz for the fruitful discussions on the use of local patterns to design models. This work is partly supported by the ANR (French Research National Agency) funded project Bingo2 ANR-07-MDCO-014, which is a follow-up of the first Bingo project (2004-2007).

References

[1] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast discovery of association rules. In: Advances in Knowledge Discovery and Data Mining, ch.12, AAAI/MIT Press (1996)

[2] B.B., Zimmermann, A.: The chosen few: On identifying valuable patterns. In: Perner, P. (ed.) ICDM 2007. LNCS (LNAI), vol. 4597, pp. 63–72. Springer, Heidelberg (2007)

[3] Boulicaut, J.-F., Bykowski, A., Rigotti, C.: Free-sets: A condensed representation of boolean data for the approximation of frequency queries. Data Min. Knowl. Discov. 7(1), 5–22 (2003)

[4] Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees. Statistics probability series. Wadsworth, Belmont (1984)

[5] Calders, T., Rigotti, C., Boulicaut, J.-F.: A survey on condensed representations for frequent sets. In: Boulicaut, J.-F., De Raedt, L., Mannila, H. (eds.) Constraint-Based Mining and Inductive Databases. LNCS (LNAI), vol. 3848, pp. 64–80. Springer, Heidelberg (2006)

[6] Crémilleux, B., Boulicaut, J.-F.: Simplest rules characterizing classes generated by delta-free sets. In: 22nd Int. Conf. on Knowledge Based Systems and Applied Artificial Intelligence (ES 2002), Cambridge, UK, December 2002, pp. 33–46. Springer, Heidelberg (2002)

[7] De Raedt, L., Jäger, M., Lee, S.D., Mannila, H.: A theory of inductive query answering. In: Proceedings of the IEEE Conference on Data Mining (ICDM 2002), Maebashi, Japan, 2002, pp. 123–130 (2002)

[8] De Raedt, L., Zimmermann, A.: Constraint-based pattern set mining. In: Proceedings of the Seventh SIAM International Conference on Data Mining, Minneapolis, Minnesota, USA, April 2007, SIAM, Philadelphia (2007)

[9] Durand, N., Crémilleux, B.: ECCLAT: a New Approach of Clusters Discovery in Categorical Data. In: 22nd Int. Conf. on Knowledge Based Systems and Applied Artificial Intelligence (ES 2002), Cambridge, UK, December 2002, pp. 177–190. Springer, Heidelberg (2002)

[10] Fu, A., R.W., Kwong, W., Tang, J.: Mining n-most interesting itemsets. In: Ohsuga, S., Raś, Z.W. (eds.) ISMIS 2000. LNCS (LNAI), vol. 1932, pp. 59–67. Springer, Heidelberg (2000)

[11] Hand, D.J.: ESF exploratory workshop on Pattern Detection and Discovery in Data Mining. In: Pattern Detection and Discovery. LNCS, vol. 2447, pp. 1–12. Springer, Heidelberg (2002)

[12] Knobbe, A., Ho, E.: Pattern teams. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 577–584. Springer, Heidelberg (2006)

[13] Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rules mining. In: Proceedings of Fourth International Conference on Knowledge Discovery & Data Mining (KDD 1998), New York, August 1998, pp. 80–86. AAAI Press, Menlo Park (1998)

[14] Mannila, H., Toivonen, H.: Levelwise search and borders of theories in knowledge discovery. Data Mining and Knowledge Discovery 1(3), 241–258 (1997)

[15] Morik, K., Boulicaut, J.-F., Siebes, A. (eds.): Local Pattern Detection. LNCS (LNAI), vol. 3539. Springer, Heidelberg (2005)

[16] Ng, R.T., Lakshmanan, V.S., Han, J., Pang, A.: Exploratory mining and pruning optimizations of constrained associations rules. In: Proceedings of ACM SIGMOD 1998, pp. 13–24. ACM Press, New York (1998)

[17] Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 398–416. Springer, Heidelberg (1998)

[18] Pensa, R., Robardet, C., Boulicaut, J.-F.: A bi-clustering framework for categorical data. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 643–650. Springer, Heidelberg (2005)

[19] Siebes, A., Vreeken, J., van Leeuwen, M.: Item sets that compress. In: Proceedings of the Sixth SIAM International Conference on Data Mining, Bethesda, MD, USA, April 2006, SIAM, Philadelphia (2006)

[20] Soulet, A., Crémilleux, B.: An efficient framework for mining flexible constraints. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 661–671. Springer, Heidelberg (2005)

[21] Soulet, A., Crémilleux, B.: Mining constraint-based patterns using automatic relaxation. Intelligent Data Analysis 13(1) (to appear). IOS Press

[22] Suzuki, E.: Undirected discovery of interesting exception rules. International Journal of Pattern Recognition and Artificial Intelligence 16(8), 1065–1086 (2002)
[23] Wang, K., Chu, X., Liu, B.: Clustering transactions using large items. In: Proceedings of ACM CIKM 1999 (1999)
[24] Zimmermann, A., De Raedt, L.: Corclass: correlated association rule mining for classification. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS (LNAI), vol. 3245, pp. 60–72. Springer, Heidelberg (2004)