

# Frequency-Based Views to Pattern Collections

Taneli Mielikäinen

*HIIT Basic Research Unit  
Department of Computer Science  
P.O.Box 68 (Gustaf Hällströmin katu 2b)  
FIN-00014 University of Helsinki, Finland*

---

## Abstract

Finding interesting patterns from data is one of the most important problems in data mining and it has been studied actively for more than a decade. However, it is still largely open problem which patterns are interesting and which are not.

The problem of detecting the interesting patterns (in a predefined class of patterns) has been attempted to solve by determining quality values for potentially interesting patterns and deciding a pattern to be interesting if its quality value (i.e., the interestingness of the pattern) is higher than a given threshold value. Again, it is very difficult to find a threshold value and a way to determine the quality values such that the collection of patterns with quality values greater than the threshold value would contain almost all truly interesting patterns and only few uninteresting ones.

To enable more accurate characterization of interesting patterns, use of constraints to further prune the pattern collection has been proposed. However, most of the constrained pattern discovery research has been focused on structural constraints for the pattern collections and the patterns. We take a complementary approach and focus on constraining the quality values of the patterns.

We propose quality value simplifications as a complementary approach to structural constraints on patterns. As a special case of the quality value simplifications, we consider discretizing the quality values. We analyze the worst case error of certain discretization functions and give efficient discretization algorithms minimizing several loss functions. In addition to that, we show that the discretizations of the quality values can be used to obtain small approximate condensed representations for collections of interesting patterns. We evaluate the proposed condensation approach experimentally using frequent itemsets.

*Key words:* Data Mining, Pattern discovery, Condensed Representations of Pattern Collections

---

## 1 Introduction

Finding interesting patterns from data is one of the most fundamental problems in data mining [29,48]. The pattern discovery task can be formulated more precisely as follows:

**Problem 1 (pattern discovery)** *Given a class  $\mathcal{P}$  of patterns and an interestingness predicate*

$$q : \mathcal{P} \rightarrow \{0, 1\}$$

*for the pattern class, find the collection*

$$\mathcal{P}_q = \{p \in \mathcal{P} : q(p) = 1\}$$

*of interesting patterns in  $\mathcal{P}$  with respect to  $q$ . Its complement  $\mathcal{P}_{\bar{q}} = \mathcal{P} \setminus \mathcal{P}_q$  is called the collection of uninteresting patterns in  $\mathcal{P}$  with respect to  $q$ .*

Thus, the pattern discovery problem, as defined above, consists of only two parts: the collection  $\mathcal{P}$  of possibly interesting patterns and the interestingness predicate  $q$ .

The pattern collection  $\mathcal{P}$  comprises *a priori* assumptions which patterns could be of interest. The collection is usually not represented explicitly since its cardinality can be very large, sometimes even infinite. (For example, a pattern collections consisting of all regular expressions on any non-empty alphabet is infinite.)

The interestingness predicate  $q$  reflects more careful conceptions about the interestingness of the patterns and it can be evaluated for any pattern in  $\mathcal{P}$ . For example, the predicate might be evaluated for some particular pattern by consulting an expert. The predicate is often based on data but the data is not always available for the user of the interestingness predicate. Furthermore, it is not necessary to have uniform algorithms for evaluating the interestingness predicate. For our purposes, the predicate can be assumed to be an oracle.

Defining a reasonable interestingness predicate is usually a very difficult task as the interestingness predicate should capture most truly interesting patterns and only few uninteresting ones. Due to these difficulties, a relaxation of an interestingness predicate, an *interestingness measure*

$$\phi : \mathcal{P} \rightarrow [0, 1]$$

---

*Email address:* [Taneli.Mielikainen@iki.fi](mailto:Taneli.Mielikainen@iki.fi) (Taneli Mielikäinen).

expressing the value  $\phi(p)$  of the interestingness for each pattern  $p \in \mathcal{P}$  (i.e., the *quality value* of  $p$ ) is often used instead of an interestingness predicate. Many kinds of interestingness measures have been studied in the pattern discovery literature; see, e.g., [58].

There are many reasons why interestingness measures are favored over interestingness predicates. An important reason is that it is often easier to suggest some degrees of interestingness for the patterns in the given collection than to partition the patterns into the groups of strictly interesting and uninteresting patterns. In fact, using an interestingness measure instead of an interestingness predicate partially postpones the difficulty of fixing a suitable interestingness predicate, since an interestingness measure implicitly determines an infinite number of interestingness predicates:

$$q_\sigma(p) = \begin{cases} 1 & \text{if } \phi(p) \geq \sigma \\ 0 & \text{otherwise} \end{cases}$$

for each  $\sigma \in [0, 1]$ .

The most prominent example of pattern discovery task using interestingness measures is discovering *frequent itemsets* from *transaction databases* [1,2,48].

**Definition 2 (itemsets and transaction databases)** *A set of possible items in data is denoted by  $\mathcal{I}$ . An itemset is a subset  $X$  of  $\mathcal{I}$ .*

*A transaction is a pair  $\langle i, X \rangle$  where  $i$  is a transaction identifier (e.g., a positive integer) and  $X$  is an itemset. A transaction database is a set  $\mathcal{D}$  of transactions such that no two transactions in  $\mathcal{D}$  have the same transaction identifier.*

The task of discovering frequent itemsets in a given transaction database can be defined as follows:

**Problem 3 (frequent itemset mining)** *Given a transaction database  $\mathcal{D}$  and a minimum frequency threshold  $\sigma \in [0, 1]$ , find all  $\sigma$ -frequent itemsets, i.e., the collection*

$$\mathcal{F}(\sigma, \mathcal{D}) = \{X \subseteq \mathcal{I} : fr(X, \mathcal{D}) \geq \sigma\}$$

where

$$fr(X, \mathcal{D}) = \frac{supp(X, \mathcal{D})}{|\mathcal{D}|}$$

is the frequency of  $X$  in  $\mathcal{D}$  and

$$\text{supp}(X, \mathcal{D}) = |\{\langle i, Y \rangle \in \mathcal{D} : X \subseteq Y\}|$$

is the support of  $X$  in  $\mathcal{D}$ .

Many kinds of data can be viewed as transaction databases and various data mining tasks arising in document analysis, web mining, computational biology, software engineering and so on can be modeled as frequent itemset mining.

There are many methods for mining frequent itemsets that are very efficient in practice, see [2,3,23,27,66]. Furthermore, many frequent itemset mining techniques have been adapted to other patterns such as sequences [60,67], episodes [12,26,50], trees [63,68], graphs [34,44,61,65] and queries [17,22,47].

The itemsets that are frequent in the transaction database  $\mathcal{D}$  are summaries of  $\mathcal{D}$  but they can be considered also as a side product of finding association rules. (Frequent association rules can be obtained efficiently from frequent itemsets by a straightforward algorithm [1]. Because of that, most of the research has been focusing on frequent itemset mining.)

**Definition 4 (association rules)** *An association rule is an implication of form  $X \Rightarrow Y$  such that  $X, Y \subseteq \mathcal{I}$ . The itemset  $X$  is called the body (or the antecedent) of the rule and the itemset  $Y$  is known as the head (or the consequent) of the rule.*

*The accuracy of an association rule  $X \Rightarrow Y$  in a transaction database  $\mathcal{D}$  is*

$$\text{acc}(X \Rightarrow Y, \mathcal{D}) = \frac{\text{fr}(X \cup Y, \mathcal{D})}{\text{fr}(X, \mathcal{D})}$$

*and the frequency of  $X \Rightarrow Y$  in  $\mathcal{D}$  is  $\text{fr}(X \Rightarrow Y, \mathcal{D}) = \text{fr}(X \cup Y, \mathcal{D})$ .*

Nevertheless, using interestingness measures instead of interestingness predicates does not solve the main problems with predicates: It is a highly non-trivial task to define an (anti-monotone) interestingness measure  $\phi$  such that there is a minimum threshold value  $\sigma$  capturing almost all truly interesting and only few uninteresting patterns in the collection. One way to augment the interestingness measure is to define additional constraints for the patterns. The use of constraints is a very important research topic in pattern discovery but the research has been concentrated mostly on structural constraints on the patterns and pattern collections [4,5,16,21,39,45,52,57]. Typical examples of structural constraints for patterns are constraints for items and itemsets: an interesting itemset can be required or forbidden to contain certain items or itemsets. Other typical constraints for pattern collections are monotone and

anti-monotone constraints such as minimum and maximum frequency thresholds, or minimum and maximum cardinality constraints for the itemsets.

**Example 5 (constraints in itemset mining)** *Let the set  $\mathcal{I}$  of items be products sold in a grocery store. The transaction database  $\mathcal{D}$  could then consist of transactions corresponding to purchases of customers that have bought something from the shop at least three times. As a constrained itemset mining task, we could be interested to find itemsets that*

- (1) *do not contain garlic,*
- (2) *consist of at least seven products,*
- (3) *contain at least two vegetables or bread and sour milk, and*
- (4) *cost at most ten euros.*

*These constraints attempt to characterize global travelers that are likely to become low-profit regular customers.*

*The first and the third constraint are examples of constraints to items or itemsets. The second and the fourth constraints are examples of anti-monotone and monotone constraints, respectively.*

*Clearly, all constraints could be expressed as boolean combinations of item constraints since that is sufficient to define any subcollection of  $2^{\mathcal{I}}$  and all constraints define a subcollection of  $2^{\mathcal{I}}$ . However, this would not be very intuitive and also it would be computationally more demanding to find all satisfying truth assignments (corresponding to the itemsets satisfying the constraint) for an arbitrary boolean formula.*

In this paper we propose a complementary approach to further restrict and sharpen the collection of interesting patterns. The approach is based on simplifying the quality values of the patterns and it can be seen as a natural generalization of characterizing the interesting patterns by a minimum quality value threshold  $\sigma$  for the quality values of the patterns. The quality value simplifications can be adapted easily to pattern classes of various kind since they depend only on the quality values of the interesting patterns and not on the structural properties of the patterns. Simplifying the quality values is suitable for interactive pattern discovery as a post-processing of a pattern collection containing the potentially interesting ones. For example, in the case of itemsets, the collection of potentially interesting patterns usually consists of the  $\sigma$ -frequent itemsets for the smallest possible minimum frequency threshold  $\sigma$  such that the frequent itemset mining is still feasible in practice.

In addition to making the collection more understandable in general, the simplifications of the quality values can be used to reduce the number of interesting patterns by discretizing the quality values and removing the patterns whose discretized quality values can be deduced (approximatively) from the

quality values of the patterns that are not removed. Although there might be more powerful ways to condense the collection of interesting patterns, the great virtue of discretization is its conceptual simplicity: it is relatively understandable how the discretization simplifies the structure of the quality values in the collection of interesting patterns.

The paper is organized as follows. In Section 2 we describe the general idea of simplifying quality values. In Section 3 we consider a special case of simplifications, namely discretizations of the quality values. We derive worst-case bounds for the errors of accuracies of association rules computed using discretized frequencies with certain maximum error guarantees and give efficient algorithms for discretizing quality values optimally with respect to several loss functions. In Section 4 we illustrate how discretizations can be used to reduce the number of patterns needed to represent the quality values of all  $\sigma$ -interesting patterns without exceeding a given error and show experimentally that discretizations can reduce considerably the number of itemsets needed to represent the frequencies of  $\sigma$ -frequent itemsets. Section 5 is a short conclusion.

For brevity, but without loss of generality, we consider frequencies instead of arbitrary quality values.

## 2 Frequency-Based Views

A *simplification of frequencies* is a mapping  $\psi : [0, 1] \rightarrow I$ , where  $I$  is a collection of non-overlapping intervals covering the interval  $[0, 1]$ , i.e.,

$$I \subset \{[a, b], [a, b), (a, b], (a, b) \subseteq [0, 1]\}$$

such that  $\cup I = [0, 1]$  and  $i \cap j = \emptyset$  for all  $i, j \in I$ . (In the rest of the paper, we often omit the brackets from singleton intervals, i.e., we write  $x$  instead of  $\{x\} = [x]$ .)

**Example 6 (frequent patterns)** *The collection  $\mathcal{F}(\sigma, \mathcal{D})$  of  $\sigma$ -frequent patterns can be defined using frequency simplifications as follows:*

$$\psi(fr(p, \mathcal{D})) = \begin{cases} fr(p, \mathcal{D}) & \text{if } fr(p, \mathcal{D}) \geq \sigma \quad \text{and} \\ [0, \sigma) & \text{otherwise.} \end{cases}$$

There are several immediate applications of frequency simplifications. They can be used, for example, to focus on some particular frequency-based property of the pattern class.

**Example 7 (focusing on some frequencies)** *First, Example 6 is an example of focusing on some frequencies.*

*As a second example, the data analyst might be interested only in very frequent (e.g., the frequency is at least  $1 - \epsilon$ ) and very infrequent (e.g., the frequency is at most  $\epsilon$ ) patterns. Then the patterns in the interval  $(\epsilon, 1 - \epsilon)$  can be neglected or their frequencies can be mapped all to interval  $(\epsilon, 1 - \epsilon)$ . Thus, the corresponding frequency simplification would be the mapping*

$$\psi(\text{fr}(p, \mathcal{D})) = \begin{cases} (\epsilon, 1 - \epsilon) & \text{if } \text{fr}(p, \mathcal{D}) \in (\epsilon, 1 - \epsilon) \text{ and} \\ \text{fr}(p, \mathcal{D}) & \text{otherwise.} \end{cases}$$

*As a third example, let us consider association rules. The data analyst might be interested in the rules with accuracy close to  $1/2$  (e.g., within some positive constant  $\epsilon$ ), i.e., the association rules  $p \Rightarrow p'$  with no predictive power. Thus, in that case the frequency simplification  $\psi(\text{acc}(p \Rightarrow p', \mathcal{D}))$  of the association rule  $p \Rightarrow p'$  is*

$$\psi(\text{acc}(p \Rightarrow p', \mathcal{D})) = \begin{cases} [0, 1/2 - \epsilon] & \text{if } \text{acc}(p \Rightarrow p', \mathcal{D}) < 1/2 - \epsilon, \\ (1/2 + \epsilon, 1] & \text{if } \text{acc}(p \Rightarrow p', \mathcal{D}) > 1/2 + \epsilon \text{ and} \\ \text{acc}(p \Rightarrow p', \mathcal{D}) & \text{otherwise.} \end{cases}$$

Frequency simplifications are useful also in condensing the collection of frequent patterns. For an example of this, see Section 4. Other potential applications are speeding up the pattern discovery algorithms, hiding confidential information about the data from the pattern users, correcting or indicating errors in data and in frequent patterns, and examining the stability of the collection of frequent patterns.

Although the frequency simplifications in general may require a considerable amount of interaction, defining simple mappings from the unit interval  $[0, 1]$  to its subintervals and applying the simplification in pattern discovery is often more tractable than defining complex structural constraints with respect to definability and computational complexity. Here are some examples of simple mappings:

- Points in a subinterval of  $[0, 1]$  can be replaced by the subinterval itself.
- The points can be discretized by a given discretization function.
- Affine transformations, logarithms and other mappings can be applied to the points.

Note that the simplification does not have to be applicable to all points in  $[0, 1]$  but only to the finite number of different frequencies  $fr(p, \mathcal{D})$  of the patterns in  $\mathcal{F}(\sigma, \mathcal{D})$  at hand.

The frequency simplifications have clearly certain limitations as they focus just on frequencies neglecting the structural aspects of the patterns and the pattern collection (although the structure of the pattern collection can be taken into account indirectly when defining the simplification). For example, sometimes interesting and uninteresting patterns can have the same frequency. Nevertheless, the frequency simplifications can be useful in constrained pattern discovery as a complementary approach to structural constraints. Furthermore the simplifications could be used to aid in the search for advantageous constraints by revealing properties that cannot be expressed by the frequencies.

### 3 Discretizing Frequencies

Discretization is an important special case of simplifying frequencies. In general, discretizations are used especially for two purposes: reducing noise and decreasing the size of the representation. As an example of these, let us look at  $k$ -means clusterings.

**Example 8 ( $k$ -means clustering)** *The  $k$ -means clustering of a (finite) point set  $P \subseteq \mathbb{R}^d$  tries to find a set  $O$  of  $k$  points in  $\mathbb{R}^d$  that minimize the cost*

$$\sum_{p \in P} \min_{o \in O} \sum_{i=1}^d (p_i - o_i)^2.$$

*This objective can be interpreted as trying to find the centers of  $k$  Gaussian distributions that would be the most likely to generate the point set  $P$ . Thus, each point in  $P$  can be considered as a cluster center plus some Gaussian noise.*

*The approximate representation of the set  $P$  by the set  $O$  of cluster centers is clearly smaller than the original point set  $P$ . Furthermore, if the centers of the Gaussian distributions are far enough from each other, then the points in  $P$  can be encoded in smaller space by expressing for each point  $p \in P$  the cluster  $o \in O$  where it belongs and the vector  $p - o$ .*

*Note that in practice, the  $k$ -means clusterings are not always correct ones, even if the assumption of  $k$  Gaussian distributions generating the set  $P$  is true, because the standard algorithm used for  $k$ -means clustering (known as the  $k$ -means algorithm) is a greedy heuristic. Furthermore, even if it were known to which cluster each of the points in  $P$  belongs to, the points in each*

cluster rarely provide the correct estimate for the cluster center. (For more details on  $k$ -means clustering, see e.g. [31,32,33,38,42].)

A discretization of frequencies can be defined as follows:

**Definition 9 (discretization of frequencies)** A discretization of frequencies is a mapping  $\gamma$  from  $[0, 1]$  to a (finite) subset of  $[0, 1]$  that preserves the order of the points. That is, if  $x, y \in [0, 1]$  and  $x \leq y$  then  $\gamma(x) \leq \gamma(y)$ . Points in the range of discretization function  $\gamma$  are called the discretization points of  $\gamma$ .

**Example 10 (discretization of frequencies)** Probably the simplest example of discretization functions is the mapping  $\gamma$  that maps all frequencies in  $[0, 1]$  to some constant  $c \in [0, 1]$ : Clearly, such a  $\gamma$  is a mapping from  $[0, 1]$  to a finite subset  $\{c\}$  of  $[0, 1]$  and  $x \leq y \Rightarrow \gamma(x) \leq \gamma(y)$  for all  $x, y \in [0, 1]$ .

One often very important requirement for a good discretization function is that it should not introduce much error, i.e., the discretized values should not differ too much from the original values. In the next subsections we prove data independent bounds for the errors in accuracies of association rules with respect to certain discretization functions of frequencies and give algorithms to minimize the empirical loss of several loss functions.

To simplify the considerations, the frequencies of the patterns are assumed to be non-zero for the rest of the paper.

### 3.1 Loss Functions for Discretization

The loss functions considered in this section are absolute error and approximation ratio.

The absolute error for a point  $x \in (0, 1]$  with respect to a discretization function  $\gamma$  is

$$\ell_a(x, \gamma) = |x - \gamma(x)|$$

and the maximum absolute error with respect to a discretization function  $\gamma$  for a finite set  $P \subset (0, 1]$  of points is

$$\ell_a(P, \gamma) = \max_{x \in P} \ell_a(x, \gamma). \tag{1}$$

In addition to the absolute error, also the relative error, i.e., the approximation ratio is often used to evaluate quality of the approximation. The approximation

ratio for a point  $x \in (0, 1]$  is

$$\ell_r(x, \gamma) = \frac{\gamma(x)}{x}$$

and the maximum approximation ratio interval with respect to a discretization function for a finite set  $P \subset (0, 1]$  is

$$\ell_r(P, \gamma) = \left[ \min_{x \in P} \ell_r(x, \gamma), \max_{x \in P} \ell_r(x, \gamma) \right]. \quad (2)$$

Let  $\ell(x, \gamma)$  denote the loss of a point  $x \in P$  with respect to a given discretization function  $\gamma$ . Sometimes the most appropriate error for a point set is not the maximum error  $\max_{x \in P} \ell(x, \gamma)$  but a weighted sum of the errors of individual points. If the weight function is  $w : P \rightarrow \mathbb{R}$  then the weighted sum of errors is

$$\ell_w(P, \gamma) = \sum_{x \in P} w(x) \ell(x, \gamma). \quad (3)$$

In the next few subsections we derive efficient algorithms for minimizing these loss functions defined by Equation 1, Equation 2 and Equation 3.

### 3.2 Data Independent Discretization

In this subsection we show that the discretization functions

$$\gamma_a^\epsilon(x) = \epsilon + 2\epsilon \left\lfloor \frac{x}{2\epsilon} \right\rfloor \quad (4)$$

and

$$\gamma_r^\epsilon(x) = (1 - \epsilon)^{1+2\lfloor (\ln x)/(2\ln(1-\epsilon)) \rfloor} \quad (5)$$

are worst case optimal discretization functions with respect to the maximum absolute error and the maximum approximation ratio, respectively. Furthermore, we bound the maximum absolute error and the intervals for approximation ratios for the approximation computed using the discretized frequencies.

Let us first study the optimality of the discretization functions. The discretization function  $\gamma_a^\epsilon$  is optimal in the following sense:

**Theorem 11** *Let  $P \subset (0, 1]$  be a finite set. Then*

$$\ell_a(P, \gamma_a^\epsilon) \leq \epsilon.$$

*Furthermore, for any other discretization function  $\gamma$  with less discretization points,  $\ell_a(x, \gamma) > \epsilon$  for some point  $x \in (0, 1]$ .*

**PROOF.** For any point  $x \in (0, 1]$ , the absolute error  $\ell_a(x, \gamma_a^\epsilon)$  with respect to the discretization function  $\gamma_a^\epsilon$  is at most  $\epsilon$  since

$$2\epsilon \left\lfloor \frac{x}{2\epsilon} \right\rfloor \leq x < 2\epsilon + 2\epsilon \left\lfloor \frac{x}{2\epsilon} \right\rfloor$$

and

$$\gamma_a^\epsilon(x) = \epsilon + 2\epsilon \left\lfloor \frac{x}{2\epsilon} \right\rfloor.$$

Any discretization  $\gamma$  can be considered as a covering of the interval  $(0, 1]$ . Each discretization point can cover an interval of length at most  $2\epsilon$  when the maximum absolute error is allowed to be at most  $\epsilon$ . Thus, at least  $\lceil 1/(2\epsilon) \rceil$  discretization points are needed to cover the whole interval  $(0, 1]$ . The discretization function  $\gamma_a^\epsilon$  uses exactly that number of discretization points.  $\square$

It can be observed from the proof of Theorem 11 that some maximum error bounds  $\epsilon$  are unnecessary high. Thus, the maximum absolute error bound  $\epsilon$  can be decreased without increasing the number of discretization points.

**Corollary 12** *The bound  $\epsilon$  for the maximum absolute error can be decreased to*

$$\frac{1}{2 \left\lceil \frac{1}{2\epsilon} \right\rceil}$$

*without increasing the number of discretization points when discretizing by the function  $\gamma_a^\epsilon$ .*

The worst case optimality of the discretization function  $\gamma_r^\epsilon$  can be shown as follows:

**Theorem 13** *Let  $P \subset (0, 1]$  be a finite set. Then*

$$\ell_r(P, \gamma_r^\epsilon) \subseteq \left[ 1 - \epsilon, \frac{1}{1 - \epsilon} \right].$$

Furthermore, for any other discretization function  $\gamma$  with less discretization points in the interval  $[x', 1]$  with  $x' \in (0, 1]$  we have

$$\ell_r(x, \gamma) \not\subseteq \left[1 - \epsilon, \frac{1}{1 - \epsilon}\right]$$

for some point  $x \in [x', 1]$ .

**PROOF.** Clearly,

$$\left\lfloor \frac{\ln x}{2 \ln(1 - \epsilon)} \right\rfloor \leq \frac{\ln x}{2 \ln(1 - \epsilon)} \leq 1 + \left\lfloor \frac{\ln x}{2 \ln(1 - \epsilon)} \right\rfloor$$

holds for all  $x \geq 0$  and we can write

$$x = (1 - \epsilon)^{(\ln x)/(\ln(1 - \epsilon))} = (1 - \epsilon)^{2(\ln x)/(2 \ln(1 - \epsilon))}.$$

Thus,

$$\begin{aligned} 1 - \epsilon &= \frac{(1 - \epsilon)^{1+2(\ln x)/(2 \ln(1 - \epsilon))}}{x} \\ &\leq \frac{(1 - \epsilon)^{1+2\lfloor (\ln x)/(2 \ln(1 - \epsilon)) \rfloor}}{x} \\ &= \frac{(1 - \epsilon)^{-1+2+2\lfloor (\ln x)/(2 \ln(1 - \epsilon)) \rfloor}}{x} \\ &\leq \frac{(1 - \epsilon)^{-1+2(\ln x)/(2 \ln(1 - \epsilon))}}{x} = \frac{1}{1 - \epsilon}. \end{aligned}$$

The discretization function  $\gamma_r^\epsilon$  is worst case optimal for any interval  $[x, 1] \subset (0, 1]$ , since it defines a partition of  $[x, 1]$  with maximally long intervals.  $\square$

Furthermore, the discretization function with the maximum absolute and the maximum relative approximation error guarantees gives guarantees for the maximum relative and the maximum absolute errors, respectively, as follows:

**Theorem 14** *A discretization function with the maximum absolute error  $\epsilon$  guarantees that a discretization of a  $x \in (0, 1]$  has the relative error in the interval  $[1 - \epsilon/x, 1 + \epsilon/x]$ .*

**PROOF.** By definition, the minimum and the maximum discretization errors of a discretization function with the maximum absolute error at most  $\epsilon$  are  $(x - \epsilon)/x = 1 - \epsilon/x$  and  $(x + \epsilon)/x = 1 + \epsilon/x$ .  $\square$

**Theorem 15** *A discretization function with the maximum relative error in the interval  $[1 - \epsilon, 1 + \epsilon]$  guarantees that a point  $x \in (0, 1]$  has the maximum absolute error at most  $\epsilon x$ .*

**PROOF.** The discretization  $\gamma(x)$  of  $x$  with the maximum relative error in the interval  $[1 - \epsilon, 1 + \epsilon]$  is in the interval  $[(1 - \epsilon)x, (1 + \epsilon)x]$ . Thus, the maximum absolute error is

$$\max \{x - (1 - \epsilon)x, (1 + \epsilon)x - x\} = \epsilon x$$

as claimed.  $\square$

An important use of frequent patterns is to discover accurate association rules. Thus, it would be very useful to be able to bound the errors in the accuracies of the association rules. For simplicity, we consider association rules over itemsets although all following results hold for any pattern collections and quality values.

Let us first study how well the maximum absolute error guarantees for frequency discretizations transfer to the maximum absolute error guarantees for the accuracies of association rules.

**Theorem 16** *Let  $\gamma^\epsilon$  be a discretization function with the maximum absolute error  $\epsilon$ . The maximum absolute error of the accuracy of the association rule  $X \Rightarrow Y$  when the frequencies  $fr(X \cup Y, \mathcal{D})$  and  $fr(X, \mathcal{D})$  are discretized by  $\gamma^\epsilon$  is at most*

$$\min \left\{ 1, \frac{2\epsilon}{fr(X, \mathcal{D})} \right\}.$$

**PROOF.** By definition, a discretization function preserves the order of points in the discretizations. Because  $fr(X \cup Y, \mathcal{D}) \leq fr(X, \mathcal{D})$ , we have  $\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) \leq \gamma^\epsilon(fr(X, \mathcal{D}))$ .

Since the correct accuracies are always in the interval  $[0, 1]$ , the maximum absolute error is at most 1.

The two extreme cases are

- (1) when  $fr(X \cup Y, \mathcal{D}) = fr(X, \mathcal{D}) - \delta > 0$  for arbitrary small  $\delta > 0$ , but  $\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) = fr(X \cup Y, \mathcal{D}) - \epsilon$  and  $\gamma^\epsilon(fr(X, \mathcal{D})) = fr(X, \mathcal{D}) + \epsilon$ , and
- (2) when  $fr(X \cup Y, \mathcal{D}) = fr(X, \mathcal{D}) - 2\epsilon + \delta > 0$  for arbitrary small  $\delta > 0$ , but  $\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) = \gamma^\epsilon(fr(X, \mathcal{D}))$ .

In the first case, the worst case absolute error is at most

$$\begin{aligned}
& \left| \frac{fr(X \cup Y, \mathcal{D}) - \epsilon}{fr(X \cup Y, \mathcal{D}) + \epsilon} - \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right| \\
& \leq \left| \frac{fr(X \cup Y, \mathcal{D}) - \epsilon}{fr(X, \mathcal{D}) + \epsilon} - \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right| \\
& = \frac{\epsilon fr(X, \mathcal{D}) + \epsilon fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})^2 + \epsilon fr(X, \mathcal{D})} \\
& \leq \frac{2\epsilon fr(X, \mathcal{D})}{fr(X, \mathcal{D})^2 + \epsilon fr(X, \mathcal{D})} \\
& \leq \frac{2\epsilon}{fr(X, \mathcal{D}) + \epsilon}.
\end{aligned}$$

In the second case, the absolute error in worst case is at most

$$1 - \frac{fr(X, \mathcal{D}) - 2\epsilon}{fr(X, \mathcal{D})} = \frac{2\epsilon}{fr(X, \mathcal{D})}$$

when  $fr(X, \mathcal{D}) \geq 2\epsilon$ .

Thus, the second case is larger and gives the upper bound.  $\square$

Note that in the worst case the maximum absolute error can indeed be 1 as shown by the following example:

**Example 17 (the tightness of the bound for  $\gamma_a^\epsilon$  and any  $\epsilon > 0$ )** Let

$$fr(X \cup Y, \mathcal{D}) = \frac{\delta}{2}$$

and

$$fr(X, \mathcal{D}) = 2\epsilon - \frac{\delta}{2}.$$

Then

$$\gamma_a^\epsilon(fr(X \cup Y, \mathcal{D})) = \gamma_a^\epsilon(fr(X, \mathcal{D})) = \epsilon.$$

Thus,

$$\left| 1 - \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right| = \left| 1 - \frac{\delta/2}{2\epsilon - \delta/2} \right| \rightarrow 1$$

when  $\delta \rightarrow 0$ .

If the maximum absolute error for the frequency discretization function is bounded, also the maximum relative error of the accuracies of the association rules computed from discretized and original frequencies can be bounded as follows:

**Theorem 18** *Let  $\gamma^\epsilon$  be a discretization function with maximum absolute error  $\epsilon$ . The approximation ratio in the accuracy of the association rule  $X \Rightarrow Y$ , when the frequencies  $fr(X \cup Y, \mathcal{D})$  and  $fr(X, \mathcal{D})$  are discretized using the function  $\gamma^\epsilon$ , is in the interval*

$$\left[ \max \left\{ 0, \frac{fr(X, \mathcal{D}) - \epsilon}{fr(X, \mathcal{D}) + \epsilon} \right\}, \frac{fr(X, \mathcal{D})}{fr(X \cup Y, \mathcal{D})} \right].$$

**PROOF.** The smallest approximation ratio is obtained when  $fr(X, \mathcal{D}) = fr(X \cup Y, \mathcal{D}) + \delta$  where  $\delta$  is an arbitrary small positive value, but  $\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) = fr(X \cup Y, \mathcal{D}) - \epsilon$  and  $\gamma^\epsilon(fr(X, \mathcal{D})) = fr(X, \mathcal{D}) + \epsilon$ . Then the approximation ratio is at most

$$\begin{aligned} & \frac{fr(X \cup Y, \mathcal{D}) - \epsilon}{fr(X, \mathcal{D}) + \epsilon} \left( \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right)^{-1} \\ &= \frac{fr(X \cup Y, \mathcal{D})fr(X, \mathcal{D}) - \epsilon fr(X, \mathcal{D})}{fr(X \cup Y, \mathcal{D})fr(X, \mathcal{D}) + \epsilon fr(X \cup Y, \mathcal{D})} \\ &= \frac{fr(X \cup Y, \mathcal{D})^2 + \delta fr(X \cup Y, \mathcal{D}) - \epsilon fr(X \cup Y, \mathcal{D}) - \delta \epsilon}{fr(X \cup Y, \mathcal{D})^2 + \delta fr(X \cup Y, \mathcal{D}) + \epsilon fr(X \cup Y, \mathcal{D})}. \end{aligned}$$

If  $\delta \rightarrow 0$ , then

$$\frac{fr(X \cup Y, \mathcal{D}) - \epsilon}{fr(X, \mathcal{D}) + \epsilon} \left( \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right)^{-1} \rightarrow \frac{fr(X \cup Y, \mathcal{D}) - \epsilon}{fr(X \cup Y, \mathcal{D}) + \epsilon}.$$

Note that in that inequality, we assume that  $fr(X, \mathcal{D}) \geq fr(X \cup Y, \mathcal{D}) \geq \epsilon$  because, by Definition 9, all discretized values are non-negative. Hence, we get the claimed lower bound.

By the definition of the approximation ratio, the upper bound is obtained when  $fr(X, \mathcal{D}) \neq fr(X \cup Y, \mathcal{D})$  but  $\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) = \gamma^\epsilon(fr(X, \mathcal{D}))$ . The greatest approximation ratio is obtained when  $fr(X \cup Y, \mathcal{D}) = fr(X, \mathcal{D}) - 2\epsilon + \delta$  for arbitrary small  $\delta > 0$  but  $\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) = \gamma^\epsilon(fr(X, \mathcal{D}))$ . Then the approximation ratio is

$$\frac{\gamma^\epsilon(fr(X \cup Y, \mathcal{D}))}{\gamma^\epsilon(fr(X, \mathcal{D}))} \left( \frac{fr(X, \mathcal{D}) - 2\epsilon + \delta}{fr(X, \mathcal{D})} \right)^{-1} \rightarrow \frac{fr(X, \mathcal{D})}{fr(X, \mathcal{D}) - 2\epsilon}$$

when  $\delta \rightarrow 0$ . If  $fr(X, \mathcal{D}) \rightarrow 2\epsilon$ , then the ratio increases unboundedly.  $\square$

The worst case the relative error bounds for the discretization function  $\gamma_a^\epsilon$  are the following.

**Example 19 (the worst case relative error bounds of  $\gamma_a^\epsilon$ )** *The smallest ratio is achieved when  $fr(X, \mathcal{D}) = 2k\epsilon$  and  $fr(X \cup Y, \mathcal{D}) = 2k\epsilon - \delta$  for arbitrary small  $\delta > 0$  and some  $k \in \{1, \dots, \lfloor 1/\epsilon \rfloor\}$ . The ratio*

$$\frac{(2k-1)\epsilon / (2k+1)\epsilon}{(2k\epsilon - \delta) / 2\epsilon}$$

*is minimized by choosing  $k = 1$ . Thus, the lower bound for the relative error is  $1/3$ .*

*The relative error cannot be bounded above since the frequencies  $fr(X, \mathcal{D}) = \delta$  and  $fr(X \cup Y, \mathcal{D}) = 2\epsilon - \delta$  give the ratio*

$$\frac{\epsilon/\epsilon}{\delta / (2\epsilon - \delta)} = \frac{2\epsilon}{\delta} - 1 \rightarrow \infty$$

*when  $\delta \rightarrow 0$  and  $\epsilon > 0$ .*

The relative errors for the accuracies of the association rules can be bounded much better when discretizing by the discretization function  $\gamma^\epsilon$  having the approximation ratio guarantees instead of the maximum absolute error guarantees.

**Theorem 20** *Let  $\gamma^\epsilon$  be a discretization function with the approximation ratio between  $(1 - \epsilon)$  and  $(1 - \epsilon)^{-1}$ . The approximation ratio in the accuracy of the*

association rule  $X \Rightarrow Y$  when the frequencies  $fr(X \cup Y, \mathcal{D})$  and  $fr(X, \mathcal{D})$  are discretized by  $\gamma^\epsilon$  is in the interval

$$\left[ (1 - \epsilon)^2, (1 - \epsilon)^{-2} \right].$$

**PROOF.** By choosing

$$\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) = (1 - \epsilon) fr(X \cup Y, \mathcal{D})$$

and

$$\gamma^\epsilon(fr(X, \mathcal{D})) = (1 - \epsilon)^{-1} fr(X, \mathcal{D})$$

we get

$$\frac{(1 - \epsilon) fr(X \cup Y, \mathcal{D})}{(1 - \epsilon)^{-1} fr(X, \mathcal{D})} = (1 - \epsilon)^2 \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})}.$$

By choosing

$$\gamma^\epsilon(fr(X \cup Y, \mathcal{D})) = (1 - \epsilon) fr(X \cup Y, \mathcal{D})$$

and

$$\gamma^\epsilon(fr(X, \mathcal{D})) = (1 - \epsilon)^{-1} fr(X, \mathcal{D})$$

we get

$$\frac{(1 - \epsilon)^{-1} fr(X \cup Y, \mathcal{D})}{(1 - \epsilon) fr(X, \mathcal{D})} = (1 - \epsilon)^{-2} \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})}.$$

It is easy to see that these are the worst case instances.  $\square$

Note that these bounds are tight also for the discretization function  $\gamma_r^\epsilon$ .

The discretization functions with the maximum absolute error guarantees give also some guarantees for the approximation ratios of accuracies:

**Theorem 21** *Let  $\gamma^\epsilon$  be a discretization function with the approximation ratio between  $(1 - \epsilon)$  and  $(1 - \epsilon)^{-1}$ . Then the maximum absolute error in the accuracy of the association rule  $X \Rightarrow Y$  when the frequencies  $fr(X \cup Y, \mathcal{D})$  and  $fr(X, \mathcal{D})$  are discretized by  $\gamma^\epsilon$  is at most*

$$1 - (1 - \epsilon)^2 = 2\epsilon(1 - \epsilon).$$

**PROOF.** There are two extreme cases. First, the frequencies  $fr(X, \mathcal{D})$  and  $fr(X \cup Y, \mathcal{D})$  can be almost equal but be discretized as far as possible from each other, i.e.,

$$\begin{aligned} & \left| \frac{(1 - \epsilon) fr(X \cup Y, \mathcal{D})}{(1 - \epsilon)^{-1} fr(X, \mathcal{D})} - \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right| \\ = & \left| (1 - \epsilon)^2 \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} - \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right| \\ = & \left(1 - (1 - \epsilon)^2\right) \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \end{aligned}$$

The maximum value is achieved by setting  $fr(X \cup Y, \mathcal{D}) = fr(X, \mathcal{D}) - \delta$  with an arbitrary small  $\delta > 0$ .

In the second case, the frequencies  $fr(X, \mathcal{D})$  and  $fr(X \cup Y, \mathcal{D})$  are discretized to have the same value although they are as apart from each other as possible. That is,

$$\begin{aligned} & \left| \frac{(1 - \epsilon)^{-1} fr(X \cup Y, \mathcal{D})}{(1 - \epsilon) fr(X, \mathcal{D})} - \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right| \\ = & \left| \frac{fr(X \cup Y, \mathcal{D})}{(1 - \epsilon)^2 fr(X, \mathcal{D})} - \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \right| \\ = & \left( \frac{1}{(1 - \epsilon)^2} - 1 \right) \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})} \\ = & \frac{1 - (1 - \epsilon)^2}{(1 - \epsilon)^2} \frac{fr(X \cup Y, \mathcal{D})}{fr(X, \mathcal{D})}. \end{aligned}$$

However, in that case  $fr(X \cup Y, \mathcal{D}) \leq (1 - \epsilon)^2 fr(X, \mathcal{D})$ . Thus, the maximum absolute error is again at most  $1 - (1 - \epsilon)^2$ .  $\square$

In this subsection we have seen that data independent discretization of frequencies with approximation guarantees can provide approximation guarantees also for the accuracies of the association rules computed from the discretized frequencies without any *a priori* information about the frequencies

(especially when the frequencies are discretized using a discretization function with the maximum approximation ratio guarantees).

### 3.3 Data Dependent Discretization

In practice, taking the actual data into account usually improves the performance of the approximation methods. Thus, it is natural to consider also data dependent discretization techniques. The problem of discretizing frequencies by taking the actual frequencies into account can be formulated as a computational problem as follows:

**Problem 22 (frequency discretization)** *Given a finite subset  $P$  of  $(0, 1]$ , a maximum error threshold  $\epsilon$  and a loss function  $\ell$ , find a discretization  $\gamma$  for  $P$  such that  $|\gamma(P)|$  is minimized and the error  $\ell(P, \gamma)$  is at most  $\epsilon$ .*

**Example 23 (frequency discretization)** *Let the set  $P \subset (0, 1]$  consist of points  $1/10, 3/10, 7/10$  and  $9/10$ , let the maximum error threshold  $\epsilon$  be  $1/10$ , and let the loss function  $\ell$  be the maximum absolute error (Equation 1).*

*Then the discretization function  $\gamma$  with the smallest number of discretization points and the maximum absolute error at most  $\epsilon$  is the mapping*

$$\gamma = \left\{ \frac{1}{10} \mapsto \frac{1}{5}, \frac{3}{10} \mapsto \frac{1}{5}, \frac{7}{10} \mapsto \frac{4}{5}, \frac{9}{10} \mapsto \frac{4}{5} \right\}.$$

*If  $\epsilon = 1/9$  instead, then there are several mappings with the maximum absolute error at most  $\epsilon$  and two discretization points. Namely, all mappings*

$$\gamma = \left\{ \frac{1}{10} \mapsto a, \frac{3}{10} \mapsto a, \frac{7}{10} \mapsto b, \frac{9}{10} \mapsto b \right\}$$

*with  $a \in [1/5 - 1/90, 1/5 + 1/90]$  and  $b \in [4/5 - 1/90, 4/5 + 1/90]$  have maximum absolute error at most  $\epsilon$ .*

Note that some variants of the discretization problem are closely related to density estimation. For example, if the loss function is the sum of squares, the best discretization with  $k$  discretization points is the maximum likelihood estimate for the expectations of  $k$  Gaussian distributions with the same variance describing the data [20,36]. Especially, estimating the density by piecewise constant functions, i.e., histograms, is closely related to discretization; see e.g. [35]. A histogram of a point set  $P \subseteq [0, 1]$  is a partition of the interval  $[0, 1]$  together with the counts of the points in each interval, that minimizes a certain loss function depending on the intervals and the point set. Although, constructing optimal histograms and discretizations sometimes coincide, an

important difference between them is the fact that discretizations concern only a (finite) subset  $P$  of the interval  $[0, 1]$  whereas histogram determines a discretization to the whole interval.

In this subsection we derive sub-quadratic algorithms for the maximum absolute error and polynomial-time solutions for also many other classes of loss functions.

### 3.3.1 Maximum absolute error

A discretization of a point set  $P \subseteq (0, 1]$  without exceeding the maximum absolute error  $\epsilon$  can be interpreted as an interval cover of the point set  $P$  with intervals of length  $2\epsilon$ , i.e., a collection of length- $2\epsilon$  sub-intervals of  $[0, 1]$  that cover all points in  $P$ .

A simple solution for the frequency discretization problem with the loss function being the maximum absolute error is to repeatedly choose the minimum uncovered point  $d \in P$  and discretize all the previously uncovered points of  $P$  in the interval  $[d, d + 2\epsilon]$  to the value  $d + \epsilon$ . This is described as Algorithm 1.

---

**Algorithm 1** A straightforward algorithm for discretization with respect to the maximum absolute error.

---

**Input:** A finite set  $P \subset [0, 1]$  and a real value  $\epsilon \in [0, 1]$ .

**Output:** A discretization function  $\gamma$  with  $\ell_a(P, \gamma) \leq \epsilon$ .

```

1: function INTERVAL-COVER( $P, \epsilon$ )
2:   while  $P \neq \emptyset$  do
3:      $d \leftarrow \min P$ 
4:      $I \leftarrow \{x \in P : d \leq x \leq d + 2\epsilon\}$ 
5:     for all  $x \in I$  do
6:        $\gamma(x) = d + \epsilon$ 
7:     end for
8:      $P \leftarrow P \setminus I$ 
9:   end while
10:  return  $\gamma$ 
11: end function

```

---

**Theorem 24** *Algorithm 1 finds a discretization function  $\gamma$  such that the error  $\ell_a(P, \gamma)$  is at most  $\epsilon$  and for all discretizations  $\gamma'$  with a smaller number of discretization points than  $|\gamma(P)|$  the error  $\ell_a(P, \gamma')$  is greater than  $\epsilon$ .*

**PROOF.** The maximum absolute error is at most  $\epsilon$  since all points are covered by intervals of length  $2\epsilon$  and the distance to the center of any covering interval is at most  $\epsilon$ .

To see that a smaller number of discretization points would have a larger

error, let  $x_1, \dots, x_m$  be the discretization points of the discretization  $\gamma$  found by Algorithm 1 for the point set  $P$ . By construction, there is a point  $x_i - \epsilon \in P$  for each  $1 \leq i \leq m$ . Furthermore,  $|x_i - x_j| \geq 2\epsilon$  for all discretization points  $x_i$  and  $x_j$  of  $\gamma$  such that  $1 \leq i < j \leq m$ , since otherwise the point  $x_j - \epsilon \in P$  is contained in the interval  $[x_i - \epsilon, x_i + \epsilon]$  or the point  $x_i - \epsilon \in P$  is contained in the interval  $[x_j - \epsilon, x_j + \epsilon]$ . Thus, no two points  $x_i - \epsilon, x_j - \epsilon \in P$  such that  $1 \leq i < j \leq m$  can share the same discretization point  $x_k$  where  $1 \leq k \leq m$ .  $\square$

The straightforward implementation of Algorithm 1 runs in time  $\mathcal{O}(|P|^2)$ . The bound is tight in the worst case as shown by Example 25.

**Example 25 (The worst-case running time of Algorithm 1)** *Let  $P = \{1/|P|, 2/|P|, \dots, 1 - 1/|P|, 1\}$  and  $\epsilon < 1/(2|P|)$ . Then at each iteration only one point is removed but all other points are inspected. There are  $|P|$  iterations and iteration  $i$  takes time  $\mathcal{O}(i)$ . Thus, the total time complexity is  $\mathcal{O}(|P|^2)$ .*

In the special case of  $\epsilon$  being a constant, the time complexity of the algorithm is linear in  $|P|$  because each iteration takes at most time  $\mathcal{O}(|P|)$  and there can be at most constant number of iterations: At each iteration, except possibly the last one, at least length  $2\epsilon$  subinterval of  $[0, 1]$  is covered. Thus, the number of iterations can be bounded above by  $\lceil 1/(2\epsilon) \rceil = \mathcal{O}(1)$  and the total time needed is  $\mathcal{O}(|P|)$ .

The worst case time complexity of the algorithm can be reduced to  $\mathcal{O}(|P| \log |P|)$  by constructing a heap for the point set  $P$ . A minimum element in the heap can be found in constant time and insertions and deletions to the heap can be done in time logarithmic in  $|P|$  [40].

The time complexity  $\mathcal{O}(|P| \log |P|)$  is not optimal, especially if some preprocessing of the point set  $P$  is allowed. For example, if the set  $P$  is represented as a sorted array, i.e., an array  $P$  such that  $P[i] \leq P[j]$  for all  $1 \leq i < j \leq |P|$ , then the problem can be solved in linear time in  $|P|$  by Algorithm 2.

The efficiency of Algorithm 2 depends crucially on the efficiency of sorting. In the worst case sorting real-valued points takes time  $\mathcal{O}(|P| \log |P|)$ , but sometimes, for example, when the points are almost in order, the points can be sorted much faster. For example, the frequent itemset mining algorithm APRIORI [2] finds the frequent itemsets in partially descending order in their frequencies. Note that also the generalization of the algorithm APRIORI, the levelwise algorithm (see [49]) can easily be implemented in such a way that it outputs frequent patterns in descending order in frequencies.

However, it is possible to find in time  $\mathcal{O}(|P|)$  a discretization function with the

---

**Algorithm 2** A linear-time algorithm for discretizing a sorted point set with respect to maximum absolute error.

---

**Input:** A finite set  $P \subset [0, 1]$  as an array in ascending order and a real value  $\epsilon \in [0, 1]$ .

**Output:** A discretization function  $\gamma$  with  $\ell_a(P, \gamma) \leq \epsilon$ .

```

1: function PREFIX-COVER( $P, \epsilon$ )
2:   for  $i = 1, \dots, |P|$  do
3:     if  $d < P[i] - \epsilon$  then
4:        $d \leftarrow P[i] + \epsilon$ 
5:     end if
6:      $\gamma(P[i]) \leftarrow d$ 
7:   end for
8:   return  $\gamma$ 
9: end function

```

---

maximum absolute error at most  $\epsilon$  and the minimum number of discretization points, even if the points in  $P$  are not in ordered in some specific way in advance. This can be done by first discretizing the frequencies without looking at the actual frequencies and then repairing the discretization. The high-level idea of the algorithm is as follows:

- (1) Put the points in  $P$  into bins  $0, 1, \dots, \lfloor 1/(2\epsilon) \rfloor$  corresponding to intervals  $(0, 2\epsilon], (2\epsilon, 4\epsilon], \dots, (2\epsilon \lfloor 1/(2\epsilon) \rfloor, 1]$ . Let  $B$  be the set of bins such that  $B[i]$  corresponds to bin  $i$ .
- (2) Find a minimal non-empty bin  $i$  in  $B$ . (A non-empty bin  $i$  is called minimal if  $i = 0$  or the bin  $i - 1$  is empty.)
- (3) Find the smallest point  $x$  in the bin  $i$ , replace the interval corresponding to the bin  $i$  by interval  $[x, x + 2\epsilon]$  and move the points of the bin  $i + 1$  that are in the interval  $[x, x + 2\epsilon]$  into the bin  $i$ .
- (4) Remove bin  $i$  from  $B$ .
- (5) Go to step 2 if there are still non-empty bins.

The algorithm can be implemented to run in linear time in  $|P|$ : The discretization into the bins can be computed in time  $\mathcal{O}(|P|)$  using a hash table for the set  $B$  [40]. A minimal non-empty bin can be found in amortized constant time by processing the consecutive runs of non-empty bins consecutively.

If the points in  $P$  are given in an arbitrary order, then Algorithm 3 is asymptotically optimal for minimizing the number of discretization points with respect to the given maximum absolute discretization error threshold  $\epsilon$  as shown by Theorem 26.

**Theorem 26** *No (deterministic) algorithm can find a discretization  $\gamma$  with the minimum number of discretization points without inspecting all points in  $P \subset (0, 1]$  when  $2\epsilon + \delta \leq 1$  for some  $\delta > 0$ .*

---

**Algorithm 3** A linear-time algorithm for discretization with respect to maximum absolute error.

---

**Input:** A finite set  $P \subset [0, 1]$  and a real value  $\epsilon \in [0, 1]$ .

**Output:** A discretization function  $\gamma$  with  $\ell_a(P, \gamma) \leq \epsilon$ .

```

1: function BIN-COVER( $P, \epsilon$ )
2:   for all  $x \in P$  do
3:      $i \leftarrow \lfloor x / (2\epsilon) \rfloor$ 
4:      $B[i] \leftarrow B[i] \cup \{x\}$ 
5:   end for
6:   for all  $B[i] \in B, B[i] \neq \emptyset$  do
7:     while  $i > 0$  and  $B[i - 1] \neq \emptyset$  do
8:        $i \leftarrow i - 1$ 
9:        $d \leftarrow \min B[i]$ 
10:    end while
11:    while  $B[i] \neq \emptyset$  do
12:       $I \leftarrow \{x \in B[i] : d \leq x \leq d + 2\epsilon\}$ 
13:      for all  $x \in I$  do
14:         $\gamma(x) \leftarrow d + \epsilon$ 
15:      end for
16:       $B[i] \leftarrow B[i] \setminus I$ 
17:      if  $\min B[i + 1] < d + 2\epsilon$  then
18:         $i \leftarrow i + 1$ 
19:      else
20:         $d \leftarrow \min B[i]$ 
21:      end if
22:    end while
23:  end for
24:  return  $\gamma$ 
25: end function

```

---

**PROOF.** Let  $P$  consist of points in the interval  $(0, \delta)$  and possibly the point 1. Furthermore, let the points examined by the algorithm be in the interval  $(0, \delta)$ . Based on that information, the algorithm cannot decide for sure whether or not the point 1 is in  $P$ .  $\square$

If the set  $P$  is given in ascending or descending order, however, then it is possible to find a set  $\gamma(P)$  of discretization points of minimum cardinality among those that determine discretization with the maximum absolute error at most  $\epsilon$ , in time  $\mathcal{O}(|\gamma(P)| \log |P|)$ ; see Algorithm 4. Although  $\gamma(P)$  is only an implicit representation of the discretization function  $\gamma : P \rightarrow \gamma(P)$ , the discretization of any point  $x \in P$  can be found in time  $\mathcal{O}(\log |\gamma(P)|)$  if the set  $\gamma(P)$  is represented, e.g., as a sorted array.

Note that the proposed techniques for discretizing with respect to the maximum absolute error guarantees (i.e., Algorithms 1, 2, 3 and 4) generalize

---

**Algorithm 4** A sublinear-time algorithm for discretization a sorted point set with respect to maximum absolute error.

---

**Input:** A finite set  $P \subseteq [0, 1]$  as an array in ascending order and a real value  $\epsilon \in [0, 1]$ .

**Output:** A discretization points  $\gamma(P)$  with  $\ell_a(P, \gamma) \leq \epsilon$ .

```

1: function LOG-COVER( $P, \epsilon$ )
2:    $i \leftarrow 1$ 
3:   while  $i \leq |P|$  do
4:      $d \leftarrow P[i] + \epsilon$ 
5:      $\gamma(P) \leftarrow \gamma(P) \cup \{d\}$ 
6:      $j \leftarrow |P| + 1$ 
7:     while  $j > i + 1$  do
8:        $k \leftarrow \lfloor (i + j) / 2 \rfloor$ 
9:       if  $P[k] \leq d + \epsilon$  then
10:         $i \leftarrow k$ 
11:       else
12:         $j \leftarrow k$ 
13:       end if
14:     end while
15:      $i \leftarrow j$ 
16:   end while
17:   return  $\gamma(P)$ 
18: end function

```

---

to maximum error functions that are strictly increasing transformations of the maximum absolute error function. Furthermore, the algorithms can be modified to minimize the maximum absolute error instead of the number of discretization points by a simple application of binary search.

### 3.3.2 Weighted sums of errors

Sometimes it would be more natural to evaluate the quality of discretizations using a weighted sum

$$\sum_{x \in P} w(x) \ell(x, \gamma)$$

of errors  $\ell(x, \gamma)$  instead of the maximum error  $\max_{x \in P} \ell(x, \gamma)$ . In that case, the algorithms described previously in this paper do not find the optimal solutions. Fortunately, the problem can be solved optimally in time polynomial in  $|P|$  by dynamic programming; see e.g. [20,36].

To describe the solution, we have to first define some notation. Let the point set  $P$  be represented as an array in ascending order, i.e.,  $P[i] \leq P[j]$  for all  $1 \leq i < j \leq |P|$ , and let  $P[i, j]$  denote the subarray  $P[i] \dots P[j]$ . The best

discretization point to represent the array  $P[i, j]$  is denoted by  $\mu_{i,j}$  and its error by  $\varepsilon_{i,j}$ . The loss of the best discretization  $P[1, i]$  with  $k$  discretization points with respect to the minimum sum of errors is denoted by  $\Delta_i^k$  and the  $k - 1$ th discretization point in that discretization is denoted by  $\omega_i^k$ .

The optimal error of  $P[1, i]$  for  $k$  discretization points can be defined by the following recursive formula:

$$\Delta_i^k = \begin{cases} \varepsilon_{1,i} & \text{if } k = 1 \text{ and} \\ \min_{k \leq j \leq i} \{ \Delta_{j-1}^{k-1} + \varepsilon_{j,i} \} & \text{otherwise.} \end{cases}$$

The optimal sum-of-errors discretization by dynamic programming can be divided into two subtasks:

- (1) Compute the matrices  $\mu$  of discretization points and  $\varepsilon$  of their errors:  $\mu_{i,j}$  is the discretization point for the subset  $P[i, j]$  and  $\varepsilon_{i,j}$  is its error.
- (2) Find the optimal discretizations for  $P[1, i]$  with  $k$  discretization points for all  $1 \leq k \leq i \leq |P|$  from the matrices  $\mu$  and  $\varepsilon$  using dynamic programming.

The optimal discretization function for  $P$  can be found from any matrix  $\varepsilon \in \mathbb{R}^{|P| \times |P|}$  of errors and any matrix  $\mu \in \mathbb{R}^{|P| \times |P|}$  of discretization points (although not all matrices  $\varepsilon$  and  $\mu$  make sense nor are they computable). For example, the matrices can be given by an expert.

Simple examples of error and discretization point matrices computable in polynomial time in  $|P|$  are the matrices  $\varepsilon$  and  $\mu$  for the weighted sum of absolute errors. They can be computed in time  $\mathcal{O}(|P|^3)$  as described by Algorithm 5. (Function MEDIAN computes the weighted median of  $P[i, j]$ .)

The discretization points  $\mu_{i,j}$  and the errors  $\varepsilon_{i,j}$  of  $P[i, j]$  for all  $1 \leq i \leq j \leq |P|$  can already be informative summaries of the set  $P$ . Besides of that, it is possible to extract from the matrices  $\varepsilon$  and  $\mu$  the matrices  $\Delta$  and  $\omega$  corresponding the partial sums of errors and the discretizations by Algorithm 6. (The matrices  $\Delta$  and  $\omega$  determine the optimal discretizations for each number of discretization points and each prefix  $P[1, i]$  of  $P$ .)

The time complexity of Algorithm 6 is  $\mathcal{O}(|P|^3)$ . The time consumption can be reduced to  $\mathcal{O}(k |P|^2)$  if we are interested only on discretizations with at most  $k$  discretization points. Furthermore, the method can be adapted to other kinds of loss functions, too. For some loss function the dynamic programming can be implemented with asymptotically better efficiency guarantees [18,36]. There are several ways to speed up the search in practice. For example, it is not necessary to compute the parts of the matrices that are detected not to be in

---

**Algorithm 5** An algorithm to compute the loss and discretization matrices  $\varepsilon$  and  $\mu$  for the point set  $P$  and a weight function  $w$ .

---

**Input:** A finite set  $P \subset [0, 1]$  and a weight function  $w : P \rightarrow \mathbb{R}$ .

**Output:** Matrices  $\varepsilon$  and  $\mu$ .

```

1: function VALUATE-ABS( $P, w$ )
2:   for  $i = 1, \dots, |P|$  do
3:     for  $j = i, \dots, |P|$  do
4:        $\mu_{i,j} \leftarrow \text{MEDIAN}(P[i, j], w)$ 
5:        $\varepsilon_{i,j} \leftarrow 0$ 
6:       for  $k = i, \dots, j$  do
7:          $\varepsilon_{i,j} \leftarrow \varepsilon_{i,j} + w(P[k]) |P[k] - \mu_{i,j}|$ 
8:       end for
9:     end for
10:  end for
11:  return  $\langle \varepsilon, \mu \rangle$ 
12: end function

```

---

**Algorithm 6** An algorithm to compute matrices  $\Delta$  and  $\omega$  from  $P$ ,  $\varepsilon$  and  $\mu$ .

---

**Input:** A finite set  $P \subset [0, 1]$ , and matrices  $\varepsilon$  and  $\mu$ .

**Output:** Matrices  $\Delta$  and  $\omega$ .

```

1: function TABULATOR( $P, \varepsilon, \mu$ )
2:   for all  $i \in \{1, \dots, |P|\}$  do ▷ Initialize the errors  $\Delta_i^k$ .
3:      $\Delta_i^1 \leftarrow \varepsilon_{1,i}$ 
4:   end for
5:   for all  $k, i \in \{2, \dots, |P|\}, k \leq i$  do
6:      $\Delta_i^k \leftarrow \infty$ 
7:   end for
8:   for  $k = 1, \dots, |P|$  do ▷ Find the best discretization of  $P[1, i]$  with  $k$ 
     discretization points.
9:      $\Delta' \leftarrow \infty$ 
10:    for all  $j, i \in \{k, \dots, |P|\}, j \leq i$  do
11:      if  $\Delta' < \Delta_i^k$  then
12:         $\Delta_i^k \leftarrow \Delta'$ 
13:         $\omega_i^k \leftarrow j - 1$ 
14:      end if
15:    end for
16:  end for
17:  return  $\langle \Delta, \omega \rangle$ 
18: end function

```

---

the best solutions.

Although the matrices  $\Delta$  and  $\omega$  contain the information about the optimal discretizations of all prefixes of  $P$  for each number of discretization points, the actual goal usually is to extract the optimal discretizations from these matrices.

The optimal discretizations of  $k$  discretization points can be found in time  $\mathcal{O}(|P|)$  by Algorithm 7. It can be adapted to find discretization with minimum number of discretization points and the error less than  $\epsilon$  in time linear in  $|P|$ .

---

**Algorithm 7** An algorithm to extract the best discretization of  $k$  discretization points from the matrices  $\Delta$  and  $\omega$ .

---

**Input:** A finite set  $P \subset [0, 1]$ , matrices  $\Delta$ ,  $\mu$  and  $\omega$ , and an integer  $k \in \{1, \dots, |P|\}$ .

**Output:** The discretization  $\gamma$  of  $k$  discretization points with the smallest error  $\Delta_{|P|}^k$ .

```

1: function FIND-DISCRETIZATION( $P, \mu, \omega, k$ )
2:    $i \leftarrow |P|$ 
3:   for  $l = k, \dots, 1$  do
4:     for  $j = i, \dots, \omega_i^l + 1$  do
5:        $\gamma(P[i]) \leftarrow \mu_{\omega_i^l, i}$ 
6:     end for
7:      $i \leftarrow \omega_i^l$ 
8:   end for
9:   return  $\gamma$ 
10: end function

```

---

Note that if it is sufficient to obtain just the set  $\gamma(P)$  of  $k$  discretization points, then the task can be conducted in time  $\mathcal{O}(k)$  by Algorithm 8.

---

**Algorithm 8** An algorithm to extract the best  $k$  discretization points from the matrices  $\Delta$  and  $\omega$ .

---

**Input:** A finite set  $P \subset [0, 1]$ , matrices  $\Delta$ ,  $\mu$  and  $\omega$ , and an integer  $k \in \{1, |P|\}$ .

**Output:** The set  $\gamma(P)$  of  $k$  discretization points with points with the smallest error  $\Delta_{|P|}^k$ .

```

1: function FIND-DISCRETIZATION-POINTS( $P, \mu, \omega, k$ )
2:    $\gamma(P) \leftarrow \emptyset$ 
3:    $i \leftarrow |P|$ 
4:   for  $l = k, \dots, 1$  do
5:      $j \leftarrow \omega_i^l + 1$ 
6:      $\gamma(P) \leftarrow \gamma(P) \cup \{\mu_{j, i}\}$ 
7:      $i \leftarrow \omega_i^l$ 
8:   end for
9:   return  $\gamma(P)$ 
10: end function

```

---

Instead of finding the best discretization with a certain number of discretization points, one could search for a hierarchical discretization suggesting good discretization of  $k$  discretization points for all values of  $k$ .

**Example 27 (hierarchical discretizations)** *Let the point set  $P$  consist of the points 0.1, 0.2, 0.5, 0.6, 0.9, and 1.0. Let us consider hierarchical dis-*

cretizations with respect to maximum absolute error. Two standard approaches to define hierarchical clusterings are divisive (or top-down) and agglomerative (or bottom-up) clusterings.

*Divisive hierarchical clustering starts from the whole point set and recursively divides it in such a way that the division always improves the solution as much as possible. For example, the divisive clustering of  $P$  would be the following:*

- *The first level of the clustering consists of only one cluster, namely the set  $\{0.1, 0.2, 0.5, 0.6, 0.9, 1.0\}$ .*
- *The maximum absolute error is decreased as much as possible by splitting the set into two parts  $\{0.1, 0.2, 0.5\}$  and  $\{0.6, 0.9, 1.0\}$*
- *In the third level no split improves the maximum absolute error. However, splitting  $\{0.1, 0.2, 0.5\}$  to  $\{0.1, 0.2\}$  and  $\{0.5\}$ , or splitting  $\{0.6, 0.9, 1.0\}$  to  $\{0.6\}$  and  $\{0.9, 1.0\}$  decreases most the maximum absolute error of one of the clusters with the maximum absolute error.*
- *The fourth level consists of the clusters  $\{0.1, 0.2\}$ ,  $\{0.5\}$ ,  $\{0.6\}$ , and  $\{0.9, 1.0\}$ .*
- *In the fifth level we have again two equally good splitting possibilities:  $\{0.1, 0.2\}$  to  $\{0.1\}$  and  $\{0.2\}$ , or  $\{0.9, 1.0\}$  to  $\{0.9\}$  and  $\{1.0\}$ .*
- *The last level consists of singletons  $\{0.1\}$ ,  $\{0.2\}$ ,  $\{0.5\}$ ,  $\{0.6\}$ ,  $\{0.9\}$ , and  $\{1.0\}$ .*

*Agglomerative hierarchical clustering starts from the singletons and merges the clusters by minimizing the error introduced by the merges. Thus, the agglomerative clustering of  $P$  would be the following: First level consists of singletons  $\{0.1\}$ ,  $\{0.2\}$ ,  $\{0.5\}$ ,  $\{0.6\}$ ,  $\{0.9\}$ , and  $\{1.0\}$ . In the next three level 0.1 and 0.2, 0.5 and 0.6, and 0.9 and 1.0 are merged in some order. Thus, the level four consists of clusters  $\{0.1, 0.2\}$ ,  $\{0.5, 0.6\}$ , and  $\{0.9, 1.0\}$ . In the level five either  $\{0.1, 0.2\}$  is merged with  $\{0.5, 0.6\}$ , or  $\{0.5, 0.6\}$  is merged with  $\{0.9, 1.0\}$ . The last level consists of the set  $P$ .*

*It depends on the actual use of the discretized values which one of these two approaches to hierarchical clustering is better.*

In addition to standard divisive and agglomerative hierarchical discretizations, it is possible to find hierarchical discretizations that are optimal with respect to a given permutation  $\pi : \{1, \dots, |P|\} \rightarrow \{1, \dots, |P|\}$  in the following sense: The discretization with  $\pi(1)$  discretization points has the minimum error among all discretizations with  $\pi(1)$  discretization points. The discretization with  $\pi(2)$  discretization points is the one that has the minimum error among all discretizations compatible with the discretization with  $\pi(1)$  discretization points. In general, the discretization with  $\pi(i)$  discretization points has the minimum error among the discretizations with  $\pi(i)$  discretization points that are compatible with the chosen discretizations with  $\pi(1), \pi(2), \dots, \pi(i-1)$  discretization points.

The time complexity of the straightforward dynamic programming implementation of this idea by modifying Algorithm 6 is  $\mathcal{O}(|P|^4)$ . Furthermore, for certain loss functions it is possible to construct hierarchical discretizations that are close to optimal for all values of the number of discretization points simultaneously [14].

The discretizations could be applied to association rules instead of frequent patterns. In that case, there are two values to discretize for each association rule: the frequency and the accuracy of the rule. This can be generalized for patterns with  $d$ -dimensional vectors of quality values. The problem is equivalent to clustering, and thus in general, the problem is NP-hard but many known approximation algorithms for clustering can be applied [8,15,19,37,42].

#### 4 Condensation by Discretization

A major problem in pattern discovery is that the collection of patterns that describe the relevant aspects of the data well can be very large. Thus, in addition to capturing most of the interesting patterns and only few uninteresting ones, it is important to be able to describe the relevant patterns concisely and accurately enough.

Frequency simplifications can be used to reach for that goal. Namely, discretization of frequencies can be used to simplify the collections of frequent patterns. The high-level schema is the following:

- (1) Discretize the frequencies of the frequent patterns.
- (2) Find a condensed representation for the pattern collection with the discretized frequencies.

The idea in condensed representations of pattern collections is to remove redundant patterns from the collections, i.e., to keep only the irredundant ones. (Redundancy is determined using some rules, e.g., assuming that the frequency of the pattern is equal to the minimum of the frequencies of its subpatterns.)

The condensed representations of frequent patterns, especially the condensed representations of frequent itemsets, have studied actively and several condensed representations, such as *maximal itemsets* [25], *closed itemsets* [55], *free itemsets* [7], *disjunction-free itemsets* [9], *disjunction-free generators* [41], *k-free itemsets* [11] *non-derivable itemsets* [10], *condensed pattern bases* [56], *pattern orderings* [54] and *pattern chains* [51], have been proposed. Furthermore, most condensed representations of itemset collections are readily applicable to several other collections of interesting patterns. In this paper, we need the definitions of only closed and maximal patterns.

**Definition 28 (maximal and closed  $\sigma$ -frequent patterns)** A pattern  $X$  is maximal in the collection  $\mathcal{F}(\sigma, \mathcal{D})$  of  $\sigma$ -frequent patterns in a database  $\mathcal{D}$  if for all  $A \in \mathcal{I} \setminus X$  holds:  $X \cup \{A\} \notin \mathcal{F}(\sigma, \mathcal{D})$ .

A patterns  $X$  is closed in  $\mathcal{F}(\sigma, \mathcal{D})$  with respect to the frequencies in  $\mathcal{D}$  if for all  $A \in \mathcal{I} \setminus X$  holds:  $fr(X, \mathcal{D}) > fr(X \cup \{A\}, \mathcal{D})$ .

As an example of condensation by discretization, the collection of closed frequent itemsets can be approximated by the closed frequent itemsets with respect to discretized frequencies.

**Example 29 (condensation by discretization and closed itemsets)** Let  $\mathcal{I} = \{1, \dots, \lfloor (1 - \sigma)n \rfloor\}$ ,  $\sigma \in (0, 1)$  and

$$\mathcal{D} = \{\langle 1, \{1\} \rangle, \dots, \langle \lfloor (1 - \sigma)n \rfloor, \{\lfloor (1 - \sigma)n \rfloor\} \rangle\} \\ \cup \{\langle \lfloor (1 - \sigma)n \rfloor + i, \mathcal{I} \rangle : i \in \{1, \dots, \lceil \sigma n \rceil\}\}.$$

Then

$$\mathcal{FC}(\sigma, \mathcal{D}) = \{\{1\}, \dots, \{n\}, \mathcal{I}\}$$

with  $fr(\mathcal{I}, \mathcal{D}) = \lceil \sigma |\mathcal{D}| \rceil / |\mathcal{D}|$  and  $fr(\{A\}, \mathcal{D}) = \lceil \sigma |\mathcal{D}| + 1 \rceil / |\mathcal{D}|$  for each  $A \in \mathcal{I}$ .

If we allow error  $1/|\mathcal{D}|$  in the frequencies, then we can discretize all frequencies of the non-empty  $\sigma$ -frequent closed itemsets in  $\mathcal{D}$  to  $\lceil \sigma |\mathcal{D}| \rceil / |\mathcal{D}|$ , i.e.,  $(\gamma \circ fr)(\emptyset, \mathcal{D}) = 1$  and  $(\gamma \circ fr)(X, \mathcal{D}) = \lceil \sigma |\mathcal{D}| \rceil / |\mathcal{D}|$  for all other  $X \subseteq \mathcal{I}$ .

Then the collection  $\mathcal{FC}(\sigma, \mathcal{D}, \gamma)$  of  $\sigma$ -frequent closed itemsets with respect to the discretization  $\gamma$  consists only of two itemsets  $\emptyset$  and  $\mathcal{I}$  with frequencies 1 and  $\lceil \sigma |\mathcal{D}| \rceil / |\mathcal{D}|$ .

Note that if the original transaction database is available, then a slightly similar approach to condense the collection of frequent itemsets is to take a random sample of the transactions and compute closed frequent itemsets in the sample. This reduces the number of closed itemsets but still results relatively good approximation for the frequencies of the frequent itemsets [53,59].

A major advantage of computing the closed frequent itemsets in a sample of transactions is that computing the closed frequent itemsets is potentially much faster than computing first the collection of (closed) itemsets in the original data and discretizing the frequencies. Disadvantages of this sampling approach are that the outcome of the closed itemset mining from the sample is also a random variable depending on the sample and the quality of the approximation provided by the closed frequent itemsets in the sample is at

most as good as the quality of the optimal approximation provided by the optimal discretization of the collection.

Of course, the sampling and discretization could be used in conjunction, by first taking a relatively large sample of transactions for obtaining the closed frequent itemsets efficiently and then discretizing the frequencies of the closed frequent itemsets in the sample. This should provide the computational efficiency and the approximation quality in between of sampling and discretizing. We focus, however, solely on discretizations.

**Example 30 (closed itemsets disappearing by discretization)** *Let us consider the collection  $\mathcal{FC}(\sigma, \mathcal{D})$  of 0.20-frequent itemsets in the course completion database of the department of Computer Science, University of Helsinki. The database consists of 2405 transactions corresponding to students and 5021 different items corresponding to courses.*

*The number of all, closed, and maximal 0.20-frequent itemsets in the course completion database are 2419, 2136, and 253, respectively. The 21 most popular courses in the database are shown in Table 1.*

*If the supports are discretized with the maximum absolute error 2 (that is less than 0.01 percent of the number of transactions in the database), then the number of closed itemsets with respect to the discretized supports is only 567, i.e., less than 24 percent of  $|\mathcal{FC}(\sigma, \mathcal{D})|$ .*

*In some parts of the itemset collection  $\mathcal{FC}(\sigma, \mathcal{D})$  the reduction in the number of the closed itemsets can be even greater than the average. For example, there are eight subsets of the itemset  $X = \{3, 5, 7, 13, 14, 15, 20\}$  than are closed with respect to exact supports but have the same discretized support as  $X$ . These itemsets are shown in Table 2.*

The number of discretization points determines the quality of the approximation: On one extreme — a discretization with only one discretization point — the frequent itemsets that are closed with respect to the discretized frequencies correspond to maximal itemsets. When the number of discretization points increases, also the number of closed frequent itemsets increases, the other extreme case being the collection of frequent closed itemsets without any discretization.

If the condensed representation depends on testing whether the frequencies of the patterns are equal (such condensed representations are, for example, the closed and the free patterns), then the number of discretization points can be used as an estimate the effectiveness of the discretization. In addition to simplifying the collections of frequent patterns, discretization can be used to make the discovery of some patterns more efficient.

Table 1

The 21 most frequent courses (i.e., items) in the course completion database. The columns are the number of courses that are more popular than the course corresponding to the row, the number of students that have passed the course, the official course code and the name of the course, respectively.

rank	count	code	name
0	2076	50001	Orientation Studies
1	1587	99270	Reading Comprehension in English
2	1498	58160	Programming Project
3	1210	58123	Computer Organization
4	1081	58128	Introduction to UNIX
5	1071	58125	Information Systems
6	1069	58131	Data Structures
7	1060	58161	Data Structures Project
8	931	99280	English Oral Test
9	920	58127	Programming in C
10	856	58122	Programming (Pascal)
11	803	99291	Oral and Written Skills in the Second Official Language, Swedish
12	763	58162	Information Systems Project
13	760	58132	Concurrent Systems
14	755	58110	Scientific Writing
15	748	58038	Database Systems I
16	744	57031	Approbatur in Mathematics I
17	733	581259	Software Engineering
18	709	57019	Discrete Mathematics I
19	697	581330	Models for Programming and Computing
20	695	50028	Maturity Test in Finnish

We evaluated the condensation abilities of discretizations using two data sets from UCI KDD Repository (<http://kdd.ics.uci.edu/>): Internet Usage data consisting of 10104 transactions and 10674 items, and IPUMS Census data consisting of 88443 transactions and 39954 items.

First we discretized the frequencies of the frequent itemsets in the Internet Usage and IPUMS Census databases and then computed which of the frequent itemsets are closed also with respect to the discretized frequencies. (In these

Table 2

The itemsets with the same discretized support as  $\{3, 5, 7, 13, 14, 15, 20\}$ . The column are as follows:  $supp(X, \mathcal{D})$  is the exact support of the itemset  $X$  in the completion database  $\mathcal{D}$ ,  $\gamma_a^2(supp(X, \mathcal{D}))$  is  $supp(X, \mathcal{D})$  discretized with maximum absolute error 2, and  $X \in \mathcal{FC}(\sigma, \mathcal{D})$  is the itemset  $X$ .

$supp(X, \mathcal{D})$	$\gamma_a^2(supp(X, \mathcal{D}))$	$X \in \mathcal{FC}(\sigma, \mathcal{D})$
488	490	$\{3, 5, 7, 13, 14, 15, 20\}$
489	490	$\{3, 5, 7, 13, 15, 20\}$
489	490	$\{3, 5, 13, 14, 15, 20\}$
490	490	$\{3, 5, 13, 15, 20\}$
490	490	$\{3, 7, 13, 14, 15, 20\}$
491	490	$\{3, 13, 14, 15, 20\}$
492	490	$\{3, 7, 13, 15, 20\}$
492	490	$\{5, 7, 13, 14, 15, 20\}$

experiments, we omitted the empty itemset from the itemset collections since its frequency is always 1.)

In the first series of experiments we were interested whether data dependent discretizations yield to smaller collections of closed itemsets than their data independent counterparts. We discretized the frequencies using discretization function  $\gamma_a^\epsilon$  (Equation 4) and the algorithm PREFIX-COVER (Algorithm 2) with different maximum absolute error thresholds  $\epsilon$  and removed the itemsets that were not closed with respect to the discretized frequencies.

The results for Internet Usage database with minimum frequency threshold 0.05 are shown in Table 3. The number of the 0.05-frequent itemsets, the number of the closed 0.05-frequent itemsets and the number of the maximal 0.05-frequent itemsets in Internet Usage database are 143391, 141568, and 23441, respectively.

The results for IPUMS Census database with minimum frequency threshold 0.2 Table 4, respectively. The results were similar to other minimum frequency thresholds. The number of the 0.2-frequent itemsets, the number of the closed 0.2-frequent itemsets and the number of the maximal 0.2-frequent itemsets in IPUMS Census database are 86879, 6689, and 578, respectively.

Clearly, the number of closed  $\sigma$ -frequent itemsets is an upper bound and the number of maximal  $\sigma$ -frequent itemsets is a lower bound for the number of frequent itemsets that are closed with respect to the discretized frequencies. The maximum absolute error is minimized in the case of just one discretization point by choosing its value to be the average of the maximum and the

Table 3

The number of closed itemsets in the collection of 0.05-frequent itemsets in the Internet Usage database with discretized frequencies for different maximum absolute error guarantees. The columns of the table are the maximum absolute error  $\epsilon$  allowed, the number of  $\sigma$ -frequent itemsets that are closed with respect to frequencies discretized using Equation 4 and the number of  $\sigma$ -frequent itemsets that are closed with respect to frequencies discretized using Algorithm 2.

$\epsilon$	fixed discretization	empirical discretization
0.0010	123426	123104
0.0050	72211	71765
0.0100	54489	45944
0.0200	34536	31836
0.0400	31587	25845
0.0600	26087	24399
0.0800	24479	23916
0.1000	23960	23705

Table 4

The number of closed itemsets in the collection of 0.2-frequent itemsets in the IPUMS Census database with discretized frequencies for different maximum absolute error guarantees. The columns of the table have the same interpretation as the columns of Table 3.

$\epsilon$	fixed discretization	empirical discretization
0.0010	3226	3242
0.0050	2362	2375
0.0100	1776	1772
0.0200	1223	1225
0.0400	1014	841
0.0600	932	725
0.0800	711	661
0.1000	627	627

minimum frequencies. The maximum absolute error for the best discretization with only one discretization point for the 0.05-frequent itemsets in the Internet Usage database is 0.4261. This is due to the fact that the highest frequency in the collection of the 0.05-frequent itemsets in the Internet Usage database (excluding the empty itemset) is 0.9022. The maximum absolute error for the best discretization with one discretization point for the 0.2-frequent itemsets in the IPUMS Census database is 0.4000. That is, there is an itemset with fre-

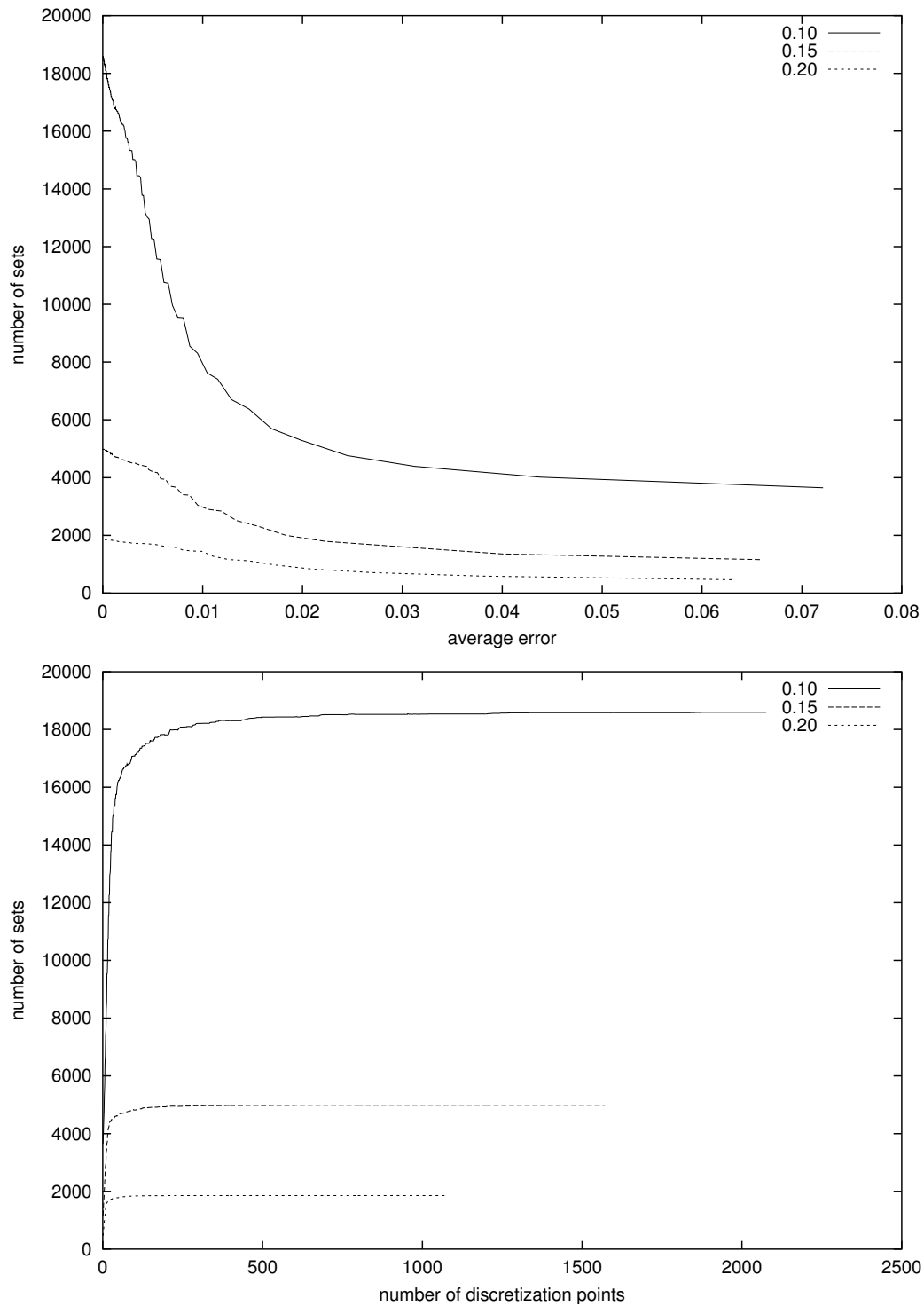


Fig. 1. Average absolute error discretization for Internet Usage data.

quency equal to 0.2 and an itemset with frequency equal to 1 in the collection of 0.2-frequent itemsets in the IPUMS Census database.

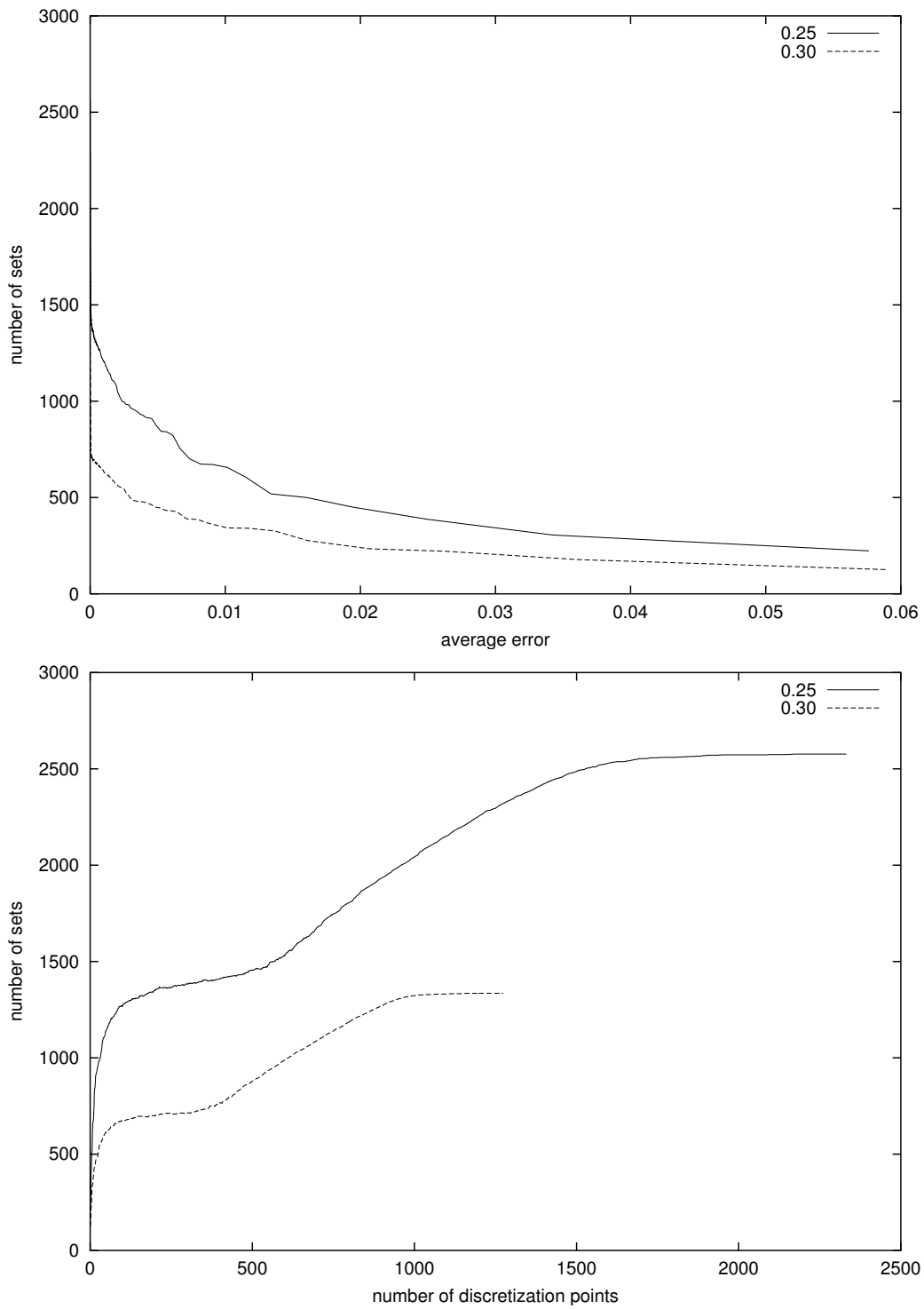


Fig. 2. Average absolute error discretization for IPUMS Census data.

In addition to minimizing the maximum absolute error, we computed the optimal discretizations with respect to the average absolute error using dynamic programming (Algorithms 5, 6 and 7). In particular, we computed the optimal discretizations for each possible number of discretization points. Practical feasibility of the dynamic programming discretization depends crucially on the number  $N = |\{fr(X, \mathcal{D}) : X \in \mathcal{F}(\sigma, \mathcal{D})\}|$  of different frequencies as its time complexity is  $\mathcal{O}(N^3)$ . Thus, the tests were conducted for smaller collections of frequent itemsets than in the case of discretization with respect to the maximum absolute error.

For the average absolute error a uniform weighting over the frequent itemsets were used. That is, the error of the discretization  $\gamma$  is

$$\frac{1}{|\mathcal{F}(\sigma, \mathcal{D})|} \sum_{X \in \mathcal{F}(\sigma, \mathcal{D})} \ell_a(fr(X, \mathcal{D}), \gamma).$$

The results are shown in Figure 1 and in Figure 2. The figures can be interpreted as follows. The labels of the curves are the minimum frequency thresholds  $\sigma$  for the collections of  $\sigma$ -frequent itemsets they correspond to. The upper figures show the number of frequent itemsets that are closed with respect to the discretized frequencies against the average absolute error of the discretization. The lower figures show the number of the closed  $\sigma$ -frequent itemsets for discretized frequencies against the number of discretization points.

On the whole, the results are encouraging, especially as the discretizations do not exploit directly the structure of the pattern collection but only the frequencies. Although there are differences between the results on different data sets, it is possible to observe that even with a quite small number of closed frequent itemsets and discretization points the frequencies of the frequent itemsets were approximated adequately.

## 5 Conclusions and Future Work

We have introduced the concept of frequency based views to frequent patterns as a complementary approach to defining structural constraints on patterns. We have given efficient algorithms for discretizing frequencies with respect to several loss functions and demonstrated that the discretization can be used for finding approximate condensed representations of frequent patterns.

However, there are still many open problems related to the frequency simplifications:

- Is it possible (and useful) to generalize frequency simplifications to higher

dimensions? For example, discretizations generalize (conceptually) trivially to higher dimensions as discussed in the end of the Section 3, but the generalizations of arbitrary simplifications seem to be more difficult.

- How should hierarchical frequency simplifications be defined and computed? Again, a simple approach is describe in Section 3 but it is not certain that it is the only and the best way to discretize hierarchically.
- What is the optimal number of discretization points? All kinds of criteria (MDL, BIC, etc.) could be used but the applicability of the assumptions in such complexity costs is somewhat questionable in the case of frequency discretizations.
- Are there simplification techniques that allow smaller approximate condensed representations of frequent patterns than the discretization? (For some alternative approaches, see [6,54,56,64].)
- What are the worst case error bounds for the condensed representations of pattern collections based on more complex deduction of frequencies? For example, the straightforward upper bounds from the maximum absolute error for non-derivable itemsets [10] grow exponentially, but for some discretization methods and loss functions the errors might still be reasonable.
- Can the frequency distribution estimated directly from data without actually computing the frequent sets? This question is closely connected also to counting the number of frequent itemsets without actually generating them.
- What is the computational complexity of discretization with respect to a wider class of loss functions?

Furthermore, frequency simplifications are closely connected to condensed representations of pattern collections, constrained pattern discovery and inductive databases which are important but still relatively open research topics in pattern discovery and data mining in general.

**Acknowledgments.** I wish to thank Jean-François Boulicaut, Gautam Das, Floris Geerts, Bart Goethals, Dimitrios Gunopulos, Heikki Mannila, Ari Rantanen, Janne Ravantti and the anonymous reviewers for helpful discussions and constructive comments.

## References

- [1] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, pages 207–216. ACM Press, 1993.

- [2] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. Fast discovery of association rules. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, chapter 12, pages 307–328. AAAI/MIT Press, 1996.
- [3] Roberto Bayardo, Bart Goethals, and Mohammed J. Zaki, editors. *Proceedings of the Workshop on Frequent Itemset Mining Implementations (FIMI-04), Brighton, UK, November 1, 2004*, volume 126 of *CEUR Workshop Proceedings*, 2004. <http://CEUR-WS.org/Vol-126/>.
- [4] Roberto J. Bayardo Jr., Rakesh Agrawal, and Dimitrios Gunopulos. Constraint-based rule mining in large, dense databases. *Data Mining and Knowledge Discovery*, 4(2/3):217–240, 2000.
- [5] Francesco Bonchi, Fosca Giannotti, Alessio Mazzanti, and Dino Pedreschi. ExAnte: Anticipated data reduction in constrained pattern mining. In Lavrač et al. [46], pages 59–70.
- [6] Jean-François Boulicaut and Arthur Bykowski. Frequent closures as a concise representation for binary data mining. In Takao Terano, Huan Liu, and Arbee L. P. Chen, editors, *Knowledge Discovery and Data Mining, Current Issues and New Applications, 4th Pacific-Asia Conference, PAKDD 2000, Kyoto, Japan, April 18-20, 2000, Proceedings*, volume 1805 of *Lecture Notes in Computer Science*, pages 62–73. Springer, 2000.
- [7] Jean-François Boulicaut, Arthur Bykowski, and Christophe Rigotti. Free-sets: a condensed representation of Boolean data for the approximation of frequency queries. *Data Mining and Knowledge Discovery*, 7(1):5–22, 2003.
- [8] Mihai Bădoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 250–257. ACM, 2002.
- [9] Artur Bykowski and Christophe Rigotti. A condensed representation to find frequent patterns. In *Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 21-23, 2001, Santa Barbara, California, USA*. ACM, 2001.
- [10] Toon Calders and Bart Goethals. Mining all non-derivable frequent itemsets. In Tapio Elomaa, Heikki Mannila, and Hannu Toivonen, editors, *Principles of Data Mining and Knowledge Discovery, 6th European Conference, PKDD 2002, Helsinki, Finland, August 19-23, 2002, Proceedings*, volume 2431 of *Lecture Notes in Artificial Intelligence*, pages 74–865. Springer, 2002.
- [11] Toon Calders and Bart Goethals. Minimal  $k$ -free representations of frequent sets. In Lavrač et al. [46], pages 71–82.
- [12] Gemma Casas-Garriga. Discovering unbounded episodes in sequential data. In Lavrač et al. [46], pages 83–94.

- [13] Nick Cercone, Tsau Young Lin, and Xindong Wu, editors. *Proceedings of the 2001 IEEE International Conference on Data Mining, 29 November - 2 December 2001, San Jose, California, USA*. IEEE Computer Society, 2001.
- [14] Sanjoy Dasgupta. Performance guarantees for hierarchical clustering. In Jyrki Kivinen and Robert H. Sloan, editors, *Computational Learning Theory, 15th Annual Conference on Computational Learning Theory, COLT 2002, Sydney, Australia, July 8-10, 2002, Proceedings*, volume 2375 of *Lecture Notes in Artificial Intelligence*, pages 351–363. Springer, 2002.
- [15] W. Fernandez de la Vega, Marek Karpinski, Claire Kenyon, and Yuval Rabani. Approximation schemes for clustering problems. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*. ACM, 2003.
- [16] Luc De Raedt, Manfred Jaeger, Sau Dan Lee, and Heikki Mannila. A theory of inductive query answering. In Kumar and Tsumoto [43], pages 123–130.
- [17] Luc Dehaspe and Hannu T.T. Toivonen. Discovery of relational association rules. In Sašo Džeroski and Nada Lavrač, editors, *Relational Data Mining*, pages 189–212. Springer, 2001.
- [18] Tapio Elomaa and Juho Rousu. On the computational complexity of optimal multisplitting. *Fundamenta Informaticae*, 47(1–2):35–52, 2001.
- [19] Tomás Feder and Daniel H. Greene. Optimal algorithms for approximate clustering. In *Proceedings of the twentieth annual ACM Symposium on Theory of Computing, Chicago, Illinois, May 2–4, 1988*, pages 434–444. ACM, 1988.
- [20] Walter D. Fisher. On grouping for maximum homogeneity. *Journal of the American Statistical Association*, 53(284):789–798, 1958.
- [21] Bart Goethals and Jan Van den Bussche. On supporting interactive association rule mining. In Yahiko Kambayashi, Mukesh K. Mohania, and A. Min Tjoa, editors, *DaWaK*, volume 1874 of *Lecture Notes in Computer Science*, pages 307–316. Springer, 2000.
- [22] Bart Goethals and Jan Van den Bussche. Relational association rules: Getting WARMeR. In Hand et al. [30], pages 125–139.
- [23] Bart Goethals and Mohammed J. Zaki, editors. *Proceedings of the Workshop on Frequent Itemset Mining Implementations (FIMI-03), Melbourne Florida, USA, November 19, 2003*, volume 90 of *CEUR Workshop Proceedings*, 2003. <http://CEUR-WS.org/Vol-90/>.
- [24] Gunter Grieser, Yuzuru Tanaka, and Akihiro Yamamoto, editors. *Discovery Science, 6th International Conference, DS 2003, Sapporo, Japan, October 17–19, 2003, Proceedings*, volume 2843 of *Lecture Notes in Computer Science*. Springer, 2003.
- [25] Dimitrios Gunopulos, Roni Khardon, Heikki Mannila, Sanjeev Saluja, Hannu Toivonen, and Ram Sewak Sharma. Discovering all most specific sentences. *ACM Transactions on Database Systems*, 28(2):140–174, 2003.

- [26] Robert Gwadera, Mikhail Atallah, and Wojciech Szpankowski. Reliable detection of episodes in event sequences. In Wu et al. [62], pages 67–74.
- [27] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, 2004.
- [28] D. Hand, D. Keim, and R. Ng, editors. *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23–26, 2002, Edmonton, Alberta, Canada*. ACM, 2002.
- [29] David J. Hand. Pattern detection and discovery. In Hand et al. [30], pages 1–12.
- [30] David J. Hand, Niall M. Adams, and Richard J. Bolton, editors. *Pattern Detection and Discovery, ESF Exploratory Workshop, London, UK, September 16–19, 2002, Proceedings*, volume 2447 of *Lecture Notes in Computer Science*. Springer, 2002.
- [31] David J. Hand, Heikki Mannila, and Padhraic Smyth. *Principles of Data Mining*. MIT Press, 2001.
- [32] Sarel Har-Peled and Bardia Sadri. How fast is the  $k$ -means method? *Algorithmica*, 41:185–202, 2005.
- [33] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer-Verlag, 2001.
- [34] Akihiro Inokuchi, Takshi Washio, and Hiroshi Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Machine Learning*, 50:321–354, 2003.
- [35] Yannis E. Ioannidis. The history of histograms (abridged). In Johann Christoph Freytag, Peter C. Lockemann, Serge Abiteboul, Michael J. Carey, Patricia G. Selinger, and Andreas Heuer, editors, *VLDB 2003, Proceedings of 29th International Conference on Very Large Data Bases, September 9–12, 2003, Berlin, Germany*, pages 19–30. Morgan Kaufmann, 2003.
- [36] H. V. Jagadish, Nick Koudas, S. Muthukrishnan, Viswanath Poosala, Kenneth C. Sevcik, and Torsten Suel. Optimal histograms with quality guarantees. In Ashish Gupta, Oded Shmueli, and Jennifer Widom, editors, *VLDB’98, Proceedings of 24rd International Conference on Very Large Data Bases, August 24–27, 1998, New York City, New York, USA*, pages 275–286. Morgan Kaufmann, 1998.
- [37] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM*, 51(3):497–515, 2004.
- [38] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for  $k$ -means clustering. *Computational Geometry: Theory and Applications*, 28:89–112, 2004.

- [39] Daniel Kifer, Johannes Gehrke, Cristian Bucila, and Walker White. How to quickly find a witness. In *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 9-12, 2003, San Diego, CA, USA*, pages 272–283. ACM, 2003.
- [40] Donald E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, second edition, 1998.
- [41] Marzena Kryszkiewicz. Concise representation of frequent patterns based on disjunction-free generators. In Cercone et al. [13], pages 305–312.
- [42] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time  $(1 + \epsilon)$ -approximation algorithm for  $k$ -means clustering in any dimensions. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 454–462. IEEE Computer Society, 2004.
- [43] Vipin Kumar and Shusaku Tsumoto, editors. *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan*. IEEE Computer Society, 2002.
- [44] Michihiro Kurakochi and George Karypis. Frequent subgraph discovery. In Cercone et al. [13], pages 313–320.
- [45] Laks V.S. Lakshmanan, Carson Kai-Sang Leung, and Raymond T. Ng. Efficient dynamic mining of constrained frequent sets. *ACM Transactions on Database systems*, 28(4):337–389, 2003.
- [46] Nada Lavrač, Dragan Gamberger, Hendrik Blockeel, and Ljupco Todorovski, editors. *Knowledge Discovery in Databases: PKDD 2003, 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003, Proceedings*, volume 2838 of *Lecture Notes in Artificial Intelligence*. Springer, 2003.
- [47] Jérôme Maloberti and Einoshin Suzuki. Improving efficiency of frequent query discovery by eliminating non-relevant candidates. In Grieser et al. [24], pages 220–232.
- [48] Heikki Mannila. Local and global methods in data mining: Basic techniques and open problems. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan Eidenbenz, and Ricardo Conejo, editors, *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, volume 2380 of *Lecture Notes in Computer Science*, pages 57–68. Springer, 2002.
- [49] Heikki Mannila and Hannu Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.
- [50] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289, 1997.

- [51] Taneli Mielikäinen. Chaining patterns. In Grieser et al. [24], pages 232–243.
- [52] Taneli Mielikäinen. Finding all occurring sets of interest. In Jean-François Boulicaut and Sašo Džeroski, editors, *2nd International Workshop on Knowledge Discovery in Inductive Databases*, pages 97–106, 2003.
- [53] Taneli Mielikäinen. Separating structure from interestingness. In Honghua Dai, Ramakrishnan Srikant, and Chengqi Zhang, editors, *Advances in Knowledge Discovery and Data Mining, 8th Pacific-Asia Conference, PAKDD 2004, Sydney, Australia, May 26-28, 2004, Proceedings*, volume 3056 of *Lecture Notes in Artificial Intelligence*, pages 476–485. Springer, 2004.
- [54] Taneli Mielikäinen and Heikki Mannila. The pattern ordering problem. In Lavrač et al. [46], pages 327–338.
- [55] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. In Catriel Beeri and Peter Buneman, editors, *Database Theory - ICDT '99, 7th International Conference, Jerusalem, Israel, January 10-12, 1999, Proceedings*, volume 1540 of *Lecture Notes in Computer Science*, pages 398–416. Springer, 1999.
- [56] Jian Pei, Guozhu Dong, Wei Zou, and Jiawei Han. On computing condensed pattern bases. In Kumar and Tsumoto [43], pages 378–385.
- [57] Ramakrishnan Srikant, Quoc Vu, and Rakesh Agrawal. Mining association rules with item constraints. In David Heckerman, Heikki Mannila, and Daryl Pregibon, editors, *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97), Newport Beach, California, USA, August 14-17, 1997*, pages 67–73. AAAI Press, 1997.
- [58] Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. Selecting the right interestingness measure for association patterns. In Hand et al. [28].
- [59] Hannu Toivonen. Sampling large databases for association rules. In T. M. Vijayaraman, Alejandro P. Buchmann, C. Mohan, and Nandlal L. Sarda, editors, *VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases, September 3-6, 1996, Mumbai (Bombay), India*, pages 134–145. Morgan Kaufmann, 1996.
- [60] Jianyong Wang and Jiawei Han. BIDE: Efficient mining of frequent closed sequences. In *Proceedings of the 20th International Conference on Data Engineering (ICDE 2004)*. IEEE Computer Society, 2004.
- [61] Xiong Wang, Jason T.L. Wang, Dennis Shasha, Bruce A. Shapiro, Isidore Rigoutsos, and Kaizhong Zhang. Finding patterns in three-dimensional graphs: Algorithms and applications to scientific data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(4):731–749, 2002.
- [62] Xindong Wu, Alex Tuzhilin, and Jude Shavlik, editors. *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003), 19-22 December 2003, Melbourne, Florida, USA*. IEEE Computer Society, 2003.

- [63] Yongqiao Xiao, Jenq-Foung Yao, Zhigang Li, and Margaret H. Dunham. Efficient data mining for maximal frequent subtrees. In Wu et al. [62], pages 379–386.
- [64] Xifeng Yan, Hong Cheng, Jiawei Han, and Dong Xin. Summarizing itemset patterns: A profile-based approach. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*. ACM, 2005.
- [65] Xifeng Yan and Jiawei Han. gSpan: Graph-based substructure pattern mining. In Kumar and Tsumoto [43], pages 721–724.
- [66] Mohammed J. Zaki. Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering*, 12(3):372–390, 2000.
- [67] Mohammed J. Zaki. SPADE: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42:31–60, 2001.
- [68] Mohammed J. Zaki. Efficiently mining frequent trees in a forest. In Hand et al. [28].