

Expressive generalized itemsets

Original

Expressive generalized itemsets / Baralis, ELENA MARIA; Cagliero, Luca; Cerquitelli, Tania; D'Elia, V.; Garza, Paolo. -
In: INFORMATION SCIENCES. - ISSN 0020-0255. - 278:(2014), pp. 327-343. [10.1016/j.ins.2014.03.056]

Availability:

This version is available at: 11583/2543388 since:

Publisher:

Elsevier

Published

DOI:10.1016/j.ins.2014.03.056

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

Expressive Generalized Itemsets

Elena Baralis, Luca Cagliero, Tania Cerquitelli, Vincenzo D'elia, Paolo Garza*

*Dipartimento di Automatica e Informatica, Politecnico di Torino,
Corso Duca degli Abruzzi 24, 10129, Torino, Italy*

Abstract

Generalized itemset mining is a powerful tool to discover multiple-level correlations among the analyzed data. A taxonomy is used to aggregate data items into higher-level concepts and to discover frequent recurrences among data items at different granularity levels. However, since traditional high-level itemsets may also represent the knowledge covered by their lower-level frequent descendant itemsets, the expressiveness of high-level itemsets can be rather limited. To overcome this issue, this article proposes two novel itemset types, called Expressive Generalized Itemset (EGI) and Maximal Expressive Generalized Itemset (Max-EGI), in which the frequency of occurrence of a high-level itemset is evaluated only on the portion of data not yet covered by any of its frequent descendants. Specifically, EGIs represent, at a high level of abstraction, the knowledge associated with sets of infrequent itemsets, while Max-EGIs compactly represent all the infrequent descendants of

*Corresponding author. Tel.: +39 011 090 7022. Fax: +39 011 090 7099.

Email addresses: elena.baralis@polito.it (Elena Baralis),
luca.cagliero@polito.it (Luca Cagliero), tania.cerquitelli@polito.it (Tania Cerquitelli), vincenzo.delia@polito.it (Vincenzo D'elia), paolo.garza@polito.it (Paolo Garza)

a generalized itemset. Furthermore, we also propose an algorithm to discover Max-EGIs at the top of the traditionally mined itemsets.

Experiments, performed on both real and synthetic datasets, demonstrate the effectiveness, efficiency, and scalability of the proposed approach.

Keywords: Generalized Itemset Mining, Data Mining, Expressiveness of generalized itemsets

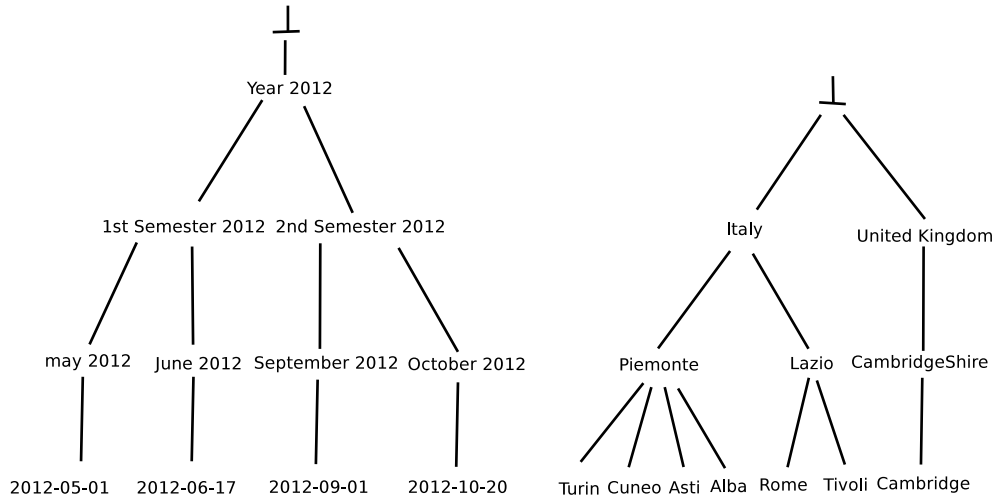
1. Introduction

Frequent itemset extraction [1] is a widely used exploratory technique to discover interesting correlations among data items. A frequent itemset is a set of data items, whose observed frequency of occurrence (support) in the source dataset is above a given threshold. A taxonomy (i.e., a set of is-a hierarchies) built over the data items can be used to aggregate items, based on granularity concepts, into higher-level items, called generalized items. Taxonomies enable the discovery of multiple-level patterns from the analyzed data. This process is called frequent generalized itemset mining [27]. Generalized itemsets are itemsets that may contain either data items or generalized items defined in the taxonomy. In most related works (e.g., [6, 19, 26]), the analyzed datasets are equipped with user-provided taxonomies and the corresponding generalized items are defined according to domain-specific knowledge. For example, in the healthcare domain examinations and drugs can be generalized as the corresponding categories, while in market basket analysis products (items) can be aggregated into the corresponding category or brand [27]. Frequent generalized itemsets are generalized itemsets whose support is above a given threshold. The support is defined as the ratio be-

tween the number of dataset records covered by the itemset and the total number of records in the dataset. The knowledge represented by a high-level (generalized) itemset is the same as a set of low-level descendants, including frequent and infrequent itemsets. However, a frequent high-level itemset is extracted even if its corresponding subset of frequent low-level itemsets covers almost the same dataset records. Hence, there is a need for improving the expressiveness of traditional generalized itemsets by proposing new types of multiple-level patterns.

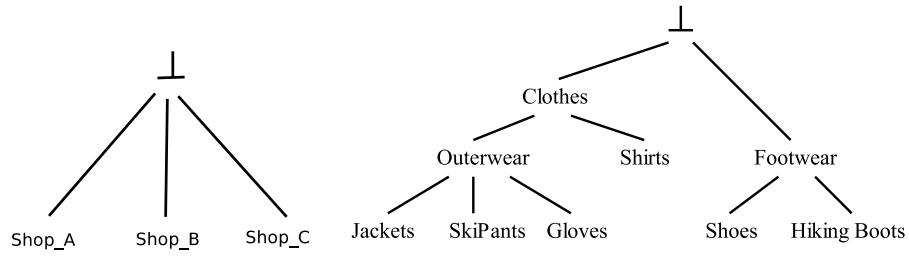
This article presents (i) two new itemset types, namely the Expressive Generalized Itemset (denoted by EGI) and the Maximal Expressive Generalized Itemset (Max-EGI). Both EGI and Max-EGI extend the notion of generalized itemset [27] by enhancing the pattern expressiveness with respect to its descendant set. (ii) An algorithm to discover Max-EGIs at the top of traditional itemsets. (iii) An in-depth experimental evaluation on many structured datasets to demonstrate the effectiveness and efficiency of the proposed approach to mine interesting and highly expressive patterns. (iv) An example of application of the proposed approach to a real-life application context, i.e., the analysis of network traffic captures. The proposed approach can be profitably exploited to analyze data coming from different application contexts if real data can be equipped with meaningful taxonomies (e.g., context-aware applications, network traffic data analysis, medical treatments).

As an example, Table 1 reports a running example dataset, where each record stores some information about the orders submitted to a chain of clothing shops. For each order the date, the name of the shop who received



(a) Order date - Generalized Tree GT_1

(b) City - Generalized Tree GT_2



(c) Clothing shop -
Generalized Tree GT_3

(d) Most discounted item - Generalized Tree
 GT_4

Figure 1: Taxonomy built on the attributes of the running example D

Order date	Clothing shop	City	Most discounted item
2012-06-17	Shop_A	Turin	Jackets
2012-09-01	Shop_A	Turin	Hiking boots
2012-10-20	Shop_B	Cambridge	Ski pants
2012-05-01	Shop_C	Rome	Hiking boots
2012-05-01	Shop_C	Tivoli	Jackets
2012-10-20	Shop_B	Asti	Hiking boots
2012-10-20	Shop_A	Cuneo	Jackets
2012-05-01	Shop_A	Cuneo	Hiking boots
2012-10-20	Shop_C	Alba	Gloves

Table 1: Example dataset.

the order, the city from which the order has been submitted, and the most discounted item are stored. Figure 1 shows a simple taxonomy built on the set of attributes of the running example. For the sake of clarity, in this section we consider only the subset of frequent generalized itemsets mined from the *most discounted item* attribute of the example dataset. Table 2(a) reports the set of frequent generalized itemsets mined by a traditional mining algorithm [27] by enforcing an absolute minimum support threshold equal to 2 and by exploiting the taxonomy in Figure 1. Both the “low-level” itemset (e.g., {(Most discounted item,Hiking boots)}) and the “high-level” (generalized) itemset (e.g., {(Most discounted item,Footwear)}) are mined even if, for example, the support value of {(Most discounted item,Footwear)} is equal to the one of its descendant {(Most discounted item,Hiking boots)}. Let us consider now generalized itemset {(Most discounted item,Outerwear)}. It covers both the knowledge associated with its infrequent descendants {(Most discounted item,Ski pant)} and {(Most discounted item,Gloves)} and the knowledge represented by its frequent descendant {(Most discounted item,Jackets)}.

Itemset	Sup
$\{(Most\ discounted\ item, Footwear)\}$	4
$\{(Most\ discounted\ item, Hiking\ boots)\}$	4
$\{(Most\ discounted\ item, Clothes)\}$	5
$\{(Most\ discounted\ item, Outerwear)\}$	5
$\{(Most\ discounted\ item, Jackets)\}$	3

(a) Generalized itemsets

Max-EGI	Sup
$\{(Most\ discounted\ item, Hiking\ boots)\}$	4
$\{(Most\ discounted\ item, Jackets)\}$	3
$\{(Most\ discounted\ item, Outerwear)\} \wr \{(Most\ discounted\ item, Jackets)\}$	2

(b) Max-EGIs

Table 2: Mined generalized itemsets and Max-EGIs. $\min_sup = 2$.

Hence, evaluating the interestingness of $\{(Most\ discounted\ item, Outerwear)\}$ is a challenging task. On the one hand, the pattern is interesting because it represents some knowledge that is not covered by any of its frequent descendants. However, on the other hand, the pattern expressiveness with respect to its descendants is rather limited, because the dataset records that contain jackets are already represented by the low-level itemset $(\{(Most\ discounted\ item, Jackets)\})$.

To select the high-level patterns that retain a significant degree of novelty with respect to their frequent descendants, we propose a novel type of high-level itemset, namely the EGI. While a traditional generalized itemset represents all of its descendant itemsets, both the frequent and the infrequent ones, each EGI represents, at a higher level of abstraction, the information represented by its subset of infrequent descendant itemsets. Hence, as thor-

oughly discussed in the following, EGIs are more expressive than traditional generalized itemsets because they consider, from a high-level viewpoint, only the rare knowledge that remains hidden at lower granularity levels.

EGIs are represented in the form $X \setminus S$, where X is a generalized itemset and S is a set of (generalized) itemsets that contains only frequent descendants of X ¹. $X \setminus S$ represents all the X 's descendants, except for those contained in S , and covers all the records that are matched by X except for those already covered by any itemset in S . For example, $\{(Most\ discounted\ item, Outerwear)\} \setminus \{(Most\ discounted\ item, Jackets)\}$ represents all the descendants of $\{(Most\ discounted\ item, Outerwear)\}$, except for $\{(Most\ discounted\ item, Jackets)\}$. Its support value is equal to 2, because it covers the same dataset records of $\{Outerwear\}$ except for those that are covered by $\{(Most\ discounted\ item, Jacket)\}$. Symbol \setminus , which is used to separate the X part from the S one, roughly recalls the symbol of complement between two sets (\setminus), because the knowledge covered by S is “excluded” from that covered by X . To enhance the readability and usability of the mined set, we consider a worthwhile EGI subset, i.e., the Max-EGIs. Max-EGIs are EGIs for which the S term has maximal length, i.e., the S term contains *all* the frequent descendants of X . Each Max-EGI compactly represents all the infrequent descendants of a generalized itemset X . Since the number of frequent descendants of X ($|S|$), is typically less than the number of infrequent descendants of X , the pattern $X \setminus S$ is a compact yet expressive representation of the infrequent knowledge covered by X . By convenient abuse of

¹For the sake of simplicity, in this preliminary example we neglect the case in which the S part may also contain other EGIs, which will be discussed in the following sections.

set theory notation, the “complement” between a generalized itemset X and its frequent descendant set S represents the (potentially large) set of infrequent descendants of X . For example, $\{(Most\ discounted\ item, Outerwear)\} \wr \{\{(Most\ discounted\ item, Jackets)\}\}$ represents itemsets $\{(Most\ discounted\ item, Ski\ pant)\}$ and $\{(Most\ discounted\ item, Gloves)\}$. Since they are individually infrequent but collectively frequent in the analyzed data, the expert could deem the above pattern to be interesting for advanced analysis. Table 2(b) reports the set of Max-EGIs extracted by enforcing an absolute minimum support threshold equal to 2 and by exploiting the taxonomy in Figure 1.

This article is organized as follows. Section 2 introduces the generalized itemset mining problem. Section 3 formally states the Max-EGI mining task, while in Section 4 an algorithm to efficiently perform Max-EGI mining is presented. Section 5 presents the experiments performed to evaluate the effectiveness and efficiency of the proposed approach. Section 6 compares our approach with previous works. Finally, Section 7 draws conclusions and presents future work.

2. Generalized itemset mining

In the context of structured data, a dataset consists of a set of records. Each record is a set of items, where an item is a pair (attribute_name, value). While attribute_name is the description of a data feature, value represents the associated information and it belongs to the corresponding attribute domain. Since continuous attributes are unsuitable for being employed in itemset mining, because their values are unlikely to occur frequently in the

analyzed data, attribute values are discretized using traditional preprocessing techniques [31]. Table 1 reports the structured dataset that will be used as running example throughout the section. To generalize items in a structured dataset at a higher level of abstraction, we introduce the notions of generalization tree and taxonomy. A generalization tree (see Figure 1(a)) is a rooted labeled tree², which represents the aggregations of the attribute domain values into higher-level concepts. Each leaf node of the tree in Figure 1(a) represents a distinct value of the domain of the *order date* attribute, whereas each non-leaf node represents a generalization (higher-level concept) of its children, which may be further generalized by its parent.

Definition 1. Generalization tree. *Let t_i be a data feature and Ω_i its domain. A generalization tree GT_i is a hierarchy of generalizations built over values in Ω_i and it is represented by a rooted labeled tree $GT_i(r, N, Label, E)$ where*

- *the set of labels $Label$ is a superset of Ω_i (i.e., $\Omega_i \subseteq Label$) and contains both the values in the attribute domain and their generalizations,*
- *leaf nodes in GT_i are labeled with values in Ω_i ,*
- *non-leaf nodes are aggregations of the values in Ω_i and are labeled with values not in Ω_i ,*

²A rooted labeled tree is an acyclic connected graph in which a node is selected as the root. A rooted labeled tree T could be denoted by $T(r, N, Label, E)$, where (1) N is the set of nodes; (2) $r \in N$ is the root node; (3) $Label$ is the set of node labels, for every node $n \in N$, $Label(n)$ is the label of node n ; and (4) $E = \{(x, y) \mid x, y \in N, x \neq y\}$ is the set of edges.

- the root node is labeled with the special value \perp (i.e., undefined).
- for each label $l \in \text{Label}$ there exists one and only one node in GT_i labeled with l .

Figure 1 reports three examples of generalization trees built over the attributes of the dataset D in Table 1. The *order date* attribute has the canonical form YY-MM-DD and might be generalized into its corresponding month, semester, and year (see Figure 1(a)), while the *city* attribute might be generalized into region/country and state (see Figure 1(b)). Since the *clothing shop* attribute values have no meaningful aggregations, the corresponding GT_3 aggregates all the leaves labeled with values in Ω_3 directly into the root node (see Figure 1(c)).

A taxonomy is a set of generalization trees defined on the attributes of a structured dataset. For example, the set of generalization trees in Figure 1 is a taxonomy defined on the attributes of the running example dataset.

Definition 2. Taxonomy. Let \mathcal{T} be a set of attributes. A taxonomy $\Gamma = \{GT_1, GT_2, \dots, GT_n\}$ is a forest of generalization trees, where GT_i is a generalization tree defined on attribute $t_i \in \mathcal{T}$.

Although a taxonomy may consist of an arbitrary number of generalization trees per attribute, for the sake of simplicity in this article we consider only taxonomies containing one generalization tree per attribute.

In the presence of taxonomies, itemsets are sets of data items belonging to distinct dataset attributes, for which the corresponding values are taxonomy node labels (disregarding the root label). Note that each item can be mapped

to the corresponding taxonomy node. If an itemset contains at least one item corresponding to a non-leaf taxonomy node (i.e., an item that does not appear in the source dataset) then it is also called *generalized* itemset. The generalization level of an item with respect to the given taxonomy is defined as the height of the subtree rooted in the corresponding taxonomy node. The itemsets whose items have all the same level are called *level-sharing* itemsets [15]. Similar to the approach proposed in [15], we focus on this subset of (generalized) itemsets. For example, $\{(\text{date}, \text{may 2012}), (\text{city}, \text{Latium})\}$ is an example of level-sharing itemset of level 2, because both the subtrees of GT_1 and GT_2 rooted in *may 2012* and *Latium*, respectively, have height 2 (see Figure 1). A k -itemset I (i.e., a set of k items) covers a given record r if all of its items are either contained in r or ancestors of an item in r with respect to the given taxonomy. Given a structured dataset \mathcal{D} , the I 's support in \mathcal{D} is defined as the ratio between the number of records in \mathcal{D} covered by I and the total number of records in \mathcal{D} . Itemsets whose support is above or equal to a given support threshold are said to be *frequent*. For example, since $\{(\text{date}, \text{may 2012}), (\text{city}, \text{Latium})\}$ covers the fourth and the fifth records of the running example dataset (see Table 1), its support value is $\frac{2}{9}$. Given a structured dataset \mathcal{D} , a taxonomy, and a minimum support threshold, the (generalized) frequent itemset mining problem entails discovering all the frequent (generalized) itemsets from \mathcal{D} .

3. Problem statement

To effectively represent the knowledge extracted from a structured dataset at different abstraction levels, we propose two novel itemset types, called EGI

and Max-EGI. As thoroughly discussed in the following, the newly proposed patterns are more expressive than traditional generalized itemsets.

Let us consider a k -itemset X . Every k -itemset $Y \neq X$ whose items are either contained in X or descendants of any item in X is said to be a *descendant* of X . Every portion of the source dataset \mathcal{D} covered by a X 's descendant is covered by X too. Hence, we argue that only the records covered by X and that are not covered by any of its frequent descendants represent a portion of data that is worth considering for X 's support count. In fact, the subset of records covered by any frequent descendant of X are already represented by a lower-level frequent itemset. With this in mind, we introduce the concepts of EGI and Max-EGI below.

EGIs are patterns in the form $X \wr S$, where X is a (generalized) itemset and S is a set of X 's descendants or, more generally, a set of other EGIs $Y \wr Q$ such that Y is a descendant of X . The set S will be also denoted as *descendant set* throughout the article. A more formal definition of EGI follows.

Definition 3. Expressive Generalized Itemset (EGI). *Let \mathcal{D} be a structured dataset, Γ a taxonomy, X a (generalized) itemset, and S a set of EGIs. $X \wr S$ is an EGI if and only if*

- A) S is empty (i.e., $S=\{\}$), or
- B) for each EGI $(Y \wr Q) \in S$, Y is a descendant of X in \mathcal{D} with respect to Γ .

A generalized itemset X is a special case of EGI $X \wr S$ for which $S=\emptyset$. For example, given the example dataset in Table 1 and the corresponding taxonomy in Figure 1, $\{(\text{city}, \text{Piedmont})\} \wr \{\}$ is an EGI, which is equivalent

to the traditional generalized itemset $\{(city, Piedmont)\}$. For the sake of readability, hereafter we will indicate the descendant set S only if it is not empty (e.g., $\{(city, Piedmont)\} \wr \{\}$ is abbreviated to $\{(city, Piedmont)\}$). According to Definition 3, if $S \neq \emptyset$ then it contains (a subset of) X 's descendants. For example, $\{(city, Piedmont)\} \wr \{(city, Turin)\}$ is an EGI where $X = \{(city, Piedmont)\}$, while S consists of a single descendant itemset, i.e., $\{(city, Turin)\}$. The above pattern represents all the cities of the Piedmont region except for Turin. In the following, we extend the ancestor and descent relationships to EGIs.

Definition 4. EGI ancestor and descendant. *Let $W = X \wr S$ and $Z = Y \wr Q$ be two EGIs. W is an ancestor of Z if and only if X is an ancestor of Y . If W is an ancestor of Z , then Z is a descendant of W .*

An EGI $X \wr S$ covers the dataset records covered by X *except* for those already covered by any of its descendants in S . A more formal definition follows.

Definition 5. EGI coverage. *Let \mathcal{D} be a structured dataset and $X \wr S$ an EGI. $X \wr S$ covers a record $r \in \mathcal{D}$ if and only if:*

- A) X covers r and
- B) \nexists an EGI $W_j \in S$ such that W_j covers r .

Based on the concept of EGI coverage, the EGI support is defined accordingly.

Definition 6. EGI support. *Let \mathcal{D} be a structured dataset and W an EGI. The W 's support in \mathcal{D} , denoted by $sup(W, \mathcal{D})$, is the ratio between the number of records in \mathcal{D} covered by W and the total number of records in \mathcal{D} .*

Recalling the running example, the support of $\{(city, Italy)\} \wr \{ \{(city, Piedmont)\} \}$ is $\frac{2}{9}$, because it covers 2 out of 9 dataset records. More specifically it covers Rome and Tivoli, located in Latium, whereas it does not cover any city located in Piedmont. Hence, the pattern represents all the Italian cities contained in the source dataset except for those located in the Piedmont region.

The level $L[X \wr S]$ and length of an EGI $X \wr S$ correspond to those of its X part. An EGI for which X has length k is also denoted by k -EGI. For example, $\{(city, Italy)\} \wr \{ \{(city, Piedmont)\} \}$ is characterized by length 1 and level 3.

The descendant set S of an EGI $X \wr S$ may potentially contain some of its descendant EGIs $Y \wr Q$. For example, the following EGI represents all the Italian cities occurring in the analyzed dataset except for those located in the Piedmont region other than Turin:

$$\{(city, Italy)\} \wr \{ \{(city, Piedmont)\} \wr \{ \{(city, Turin)\} \} \}$$

To reduce the amount of generated EGIs and thus ease the knowledge discovery process, we consider a worthwhile EGI subset. The EGI selection procedure is two-fold.

Level-sharing EGI selection: following an approach similar to the one proposed in [15], we consider only the EGIs for which X is a level-sharing itemset. We denote this type of EGIs as the *level-sharing* EGIs.

Max-EGI selection: to focus our attention on a subset of highly expressive EGIs, we select only the level-sharing EGIs $X \wr S$ for which:

- 1) the support in the source dataset is above or equal to a given threshold min_sup (i.e., the frequent EGIs) and
- 2) the descendant set S consists of all of the frequent descendant EGIs of $X \wr S$ (see Definition 4).

Constraints (1) and (2) ensure that the selected EGIs cover a significant portion of the analyzed data that is not already covered by any of their frequent descendants. EGIs that satisfy both (1) and (2) are called Max-EGIs. A more formal definition of Max-EGI follows.

Definition 7. Max-EGI. *Let \mathcal{D} be a structured dataset, Γ a taxonomy, $O = X \wr S$ a level-sharing EGI, and min_sup a minimum support threshold. O is a Max-EGI if and only if O is frequent (i.e., $\text{sup}(O, \mathcal{D}) \geq \text{min_sup}$) and at least one the following conditions hold:*

- $L[O]=1$ (i.e., O has no descendants) or
- there exists no $O_a = X_a \wr S_a$ such that $O_a \notin S$, $\text{sup}(O_a, \mathcal{D}) \geq \text{min_sup}$, and O_a is a frequent descendant of O with respect to Γ .

From the above definition, it follows that $O = X \wr S$ is Max-EGI if it is frequent according to min_sup and its S part contains all the frequent descendants of $X \wr S$.

Below we report the list of frequent EGIs mined from the city attribute of the example dataset in Table 1 by enforcing an absolute minimum support threshold equal to 2.

A) $\{(\text{city}, \text{Turin})\}$, $\text{support}=\frac{2}{9}$

- B) $\{(city, Cuneo)\}$, $support = \frac{2}{9}$
- C) $\{(city, Piedmont)\} \wr \{ \{(city, Turin)\}, \{(city, Cuneo)\} \}$, $support = \frac{2}{9}$
- D) $\{(city, Piedmont)\} \wr \{ \{(city, Turin)\} \}$, $support = \frac{4}{9}$
- E) $\{(city, Piedmont)\} \wr \{ \{(city, Cuneo)\} \}$, $support = \frac{4}{9}$
- F) $\{(city, Piedmont)\}$, $support = \frac{6}{9}$

EGIs (A), (B), and (C) are Max-EGIs. Instead, $\{(city, Piedmont)\} \wr \{ \{(city, Turin)\} \}$ is not a Max-EGI because its descendant set S does not contain the frequent Max-EGI descendant $\{(city, Cuneo)\}$. Hence, the records covered by $\{(city, Cuneo)\}$ are also covered by $\{(city, Piedmont)\} \wr \{ \{(city, Turin)\} \}$. Similar considerations hold for EGIs (E) and (F).

Given a structured dataset \mathcal{D} , a taxonomy, and a minimum support threshold min_sup , the Max-EGI miner extracts all the Max-EGIs that satisfy min_sup in \mathcal{D} (Cf. Definition 7).

4. The MAX-EGI MINER Algorithm

The MAX-EGI MINER (Maximal Expressive Generalized Itemsets Miner) algorithm takes as input a structured dataset D , a taxonomy Γ built on D , and a minimum support threshold min_sup . It extracts all the Max-EGIs from D that satisfy min_sup (see Definition 7). To accomplish this task, the input taxonomy is evaluated in a bottom-up fashion, i.e., Max-EGIs are generated in order of increasing generalization level.

The MAX-EGI MINER algorithm relies on a step-wise process which can be summarized as follows: (i) Traditional frequent level-sharing itemset mining and (ii) Max-EGIs extraction at the top of the previously extracted

Algorithm 1 MAX-EGI MINER - Maximal Expressive Generalized Itemset

$(X \wr S)$ Miner

Input: a structured dataset D , a taxonomy Γ , a minimum support threshold min_sup

Output: the set of frequent Max-EGIs \mathcal{L}

```
1:  $\mathcal{TI} = \text{mineTraditionalLevelSharing}(D, \Gamma, min\_sup)$  /* Frequent level-sharing itemset mining. */  
   /* Max-EGI mining */  
2:  $\mathcal{L} = \emptyset$   
   /* One frequent Max-EGI of level 1 is generated for every frequent level-sharing itemset of level 1 */  
3: for all  $W$  in  $\mathcal{TI}[1]$  do  
4:   insert  $(W \wr \{\})$  into  $C[1]$  /* Add a level-1 Max-EGI having  $W$  as its  $X$  part. */  
5: end for  
6:  $\mathcal{L}[1] = C[1]$   
   /* Generate Max-EGIs of level greater than 1 */  
7: for  $l=2$  to  $maxlevel$  do  
8:   /* One candidate Max-EGI of level  $l$  is generated for every frequent traditional level- $l$  itemset.  
     The  $S$  part of each candidate is initially empty. */  
9:   for all  $W$  in  $\mathcal{TI}[l]$  do  
10:    insert the candidate Max-EGI  $(W \wr \{\})$  into  $C[l]$   
11:   end for  
12:   /* Exploit candidate Max-EGIs of level  $l-1$  to populate the  $S$  part of candidate Max-EGIs of  
     level  $l$ . */  
13:   for all  $I$  in  $C[l-1]$  do  
14:      $GI = \text{retrieveAncestor}(C[l], I, l, \Gamma)$ ; /* Retrieve the candidate level- $l$  itemset  $GI$  that is ancestor  
       of  $I$  */  
15:     insert the set  $I.S$  into  $GI.S$  /* The Max-EGIs in  $I.S$  are descendants of  $GI$  */  
       /* If the level- $(l-1)$  candidate  $I$  is frequent then it must be inserted into  $GI.S$  too */  
16:     if support of  $I \geq min\_sup$  then  
17:       add  $I$  to  $GI.S$   
18:     end if  
19:   end for  
   /* Select the level- $l$  frequent Max-EGIs */  
20:   for all  $c$  in  $C[l]$  do  
21:     if support of  $c \geq min\_sup$  then  
22:       add  $c$  to  $\mathcal{L}[l]$   
23:     end if  
24:   end for  
25: end for  
26: return  $\mathcal{L}$ 
```

itemsets. A pseudo-code of the Max-EGI algorithm is given in Algorithm 1. A high-level description of each algorithm step is reported below.

Level-sharing itemset mining. Since, for each Max-EGI $O = X \wr S$, X is a frequent level-sharing itemset and S contains the frequent level-sharing X 's descendants, then traditional level-sharing itemset mining is used to drive the Max-EGI extraction process (see line 1 in Algorithm 1).

To efficiently extract frequent level-sharing itemsets, we adopted an FP-growth-like itemset miner, i.e., LCMv2 [14]. Similar to FP-Growth [16], LCM relies on a projection-based approach. It entails: (i) creating and storing in main memory an FP-tree-based dataset representation and (ii) mining the frequent itemsets by recursively visiting the conditional FP-tree projections. To suit the standard LCM implementation to generalized itemset mining, we adopted the strategy, first proposed in [27], of extending the dataset records by appending to each record all its item generalizations in Γ . Furthermore, the recursive projected FP-tree generation is tailored to level-sharing itemset mining. Specifically, the conditional FP-tree related to the level- l item i is generated by exclusively considering level- l items. In such a way, the generation of not level-sharing itemsets is prevented. The frequent level-sharing itemsets are stored in \mathcal{TI} (line 1).

Max-EGI mining. Once all the frequent level-sharing itemsets X are generated, MAX-EGI MINER associates with each of them a candidate Max-EGI $X \wr S$ and populates its descendant set S . level-1 Max-EGIs can be straightforwardly mapped to traditional level-1 itemsets (contained in $\mathcal{TI}[1]$) because they are characterized by an empty descendant set (lines 3-5). In contrast, high-level Max-EGIs are mined by following a level-wise approach, i.e., the

taxonomy is climbed up stepwise until the maximum generalization level is reached (lines 7-25). Level-wise taxonomy evaluation prevents the need for multiple itemset scans. In fact, level- l Max-EGIs are generated from the level- l itemsets $\mathcal{TI}[l]$ and the level- $(l - 1)$ candidate Max-EGIs $\mathcal{C}[l - 1]$. While the level- l itemsets in $\mathcal{TI}[l]$ are exploited to populate the X part of the level- l Max-EGIs $X \wr S$ (lines 9-11), the level- $(l - 1)$ candidate Max-EGIs in $\mathcal{C}[l - 1]$ are used to fill their S set (lines 13-19). Note that the traditional generalized itemsets with level less than l can be discarded early while mining level- l Max-EGIs. Finally, the frequent level- l Max-EGIs are added to the output set (lines 20-24).

Algorithm complexity. Itemset mining algorithms are commonly evaluated in terms of time complexity, because nowadays it is one of the most challenging issues [31]. The analysis of the time complexity of the MAX-EGI MINER algorithm can be divided into two steps. The first step concerns the analysis of the complexity of the frequent level-sharing itemset mining task. Since it has been accomplished by a LCMv2 algorithm, it is linear in the number of extracted itemsets [33]. The second step entails the analysis of the Max-EGI mining procedure, which is performed at the top of level-sharing itemsets. Since the Max-EGI extraction requires a level-wise scan of the list of extracted itemsets, its time complexity is again linear in the number of mined itemsets.

Algorithm completeness and correctness. In the following we will prove by contradiction that the MAX-EGI MINER algorithm is complete and correct according to the problem statement formalized in Section 3.

Violation of the completeness assumption. By contradiction, let us sup-

pose that a Max-EGI $X \wr S$ satisfying min_sup in \mathcal{D} is not extracted. Three unexpected behaviors may happen: (a) X is not a frequent level-sharing itemset, or (b) X is a frequent level-sharing itemset but it has not been extracted, or (c) X is a frequent level-sharing itemset but some Max-EGIs have wrongly been included in S .

Since, by construction, the support of X is above or equal to the one of $X \wr S$, then hypothesis (a) is false. Furthermore, all level-sharing itemsets X are extracted by means of an established level-sharing frequent itemset mining algorithm [33] (see line 1 in Algorithm 1). Hence, hypothesis (b) is false. Finally, according to lines 13-19 in Algorithm 1, the S part of each Max-EGI is populated with the frequent level- $(l-1)$ descendants of X solely. Therefore, hypothesis (c) is false as well. Contradiction.

Violation of the correctness assumption. By contradiction, let us suppose that a Max-EGI $X \wr S$ not satisfying min_sup in \mathcal{D} is wrongly extracted. Hence, X is a frequent level-sharing itemset but some Max-EGIs are missing in S . According to lines 13-19 in Algorithm 1, the S part of each Max-EGI is populated with the subset of all frequent level- $(l-1)$ descendants of X . Therefore, the aforementioned hypothesis is false. Contradiction.

5. Experimental results

A variety of experiments have been conducted on both real-life and synthetic datasets to evaluate MAX-EGI MINER performance and compare the Max-EGI expressiveness and usefulness with that of traditional generalized itemsets. Experiments were performed on a 3.30 GHz Intel(R) Xeon(R) CPU E31245 PC with 16 GB main memory running Linux (kernel 3.2.0).

As discussed in Section 4, the traditional level-sharing itemset extraction task is accomplished by an implementation of the LCMv2 algorithm [14] which has been extended to cope with multiple-level data. The implemented algorithm is a more efficient (projection-based) version of the ML_T2L1 algorithm, originally proposed in [15]. For the sake of brevity, the traditional generalized itemset mining algorithm will be denoted by ML_T2L1 throughout the section.

This section is organized as follows. Section 5.1 briefly describes the real and synthetic datasets that have been used in the performed experiments as well as the corresponding taxonomies. Section 5.2 compares the characteristics of the Max-EGIs mined by MAX-EGI MINER from real and synthetic datasets with that of the level-sharing itemsets mined by the ML_T2L1 algorithm. Sections 5.3 and 5.4 discuss the impact of the support threshold and the discretization method, respectively, on MAX-EGI MINER and ML_T2L1 performance. Finally, Section 5.5 analyzes the MAX-EGI MINER algorithm scalability.

5.1. Datasets and taxonomies

We tested the MAX-EGI MINER and ML_T2L1 algorithms on both real and synthetic datasets. Table 3 summarizes the main characteristics (i.e., the number of records and attributes) of the analyzed datasets.

Network traffic datasets. Network traffic data acquired in real-life contexts is typically analyzed at different abstraction levels. For example, for IP traffic monitoring experts could analyze either single IP addresses or IP subnets (i.e., IP address aggregations). For this reason, network traffic analysis has been considered to be a suitable application context for assessing

Dataset	Number of records	Number of attributes	Taxonomy height
<i>UCI datasets</i>			
adult	32,561	15	3
breast	699	11	3
cleve	303	14	3
crx	690	16	3
glass	214	11	3
heart	270	14	3
iris	150	5	3
labor	57	17	3
letter	20,000	17	3
nursery	12,960	9	3
pendigits	10,992	17	3
pima	768	9	3
shuttle	43,500	10	3
vehicle	846	19	3
waveformd	5,000	22	3
wine	178	14	3
yeast	1,484	10	3
<i>Network datasets</i>			
NetD1	3,802	6	4
NetD2	17,374	6	4
<i>Synthetic datasets</i>			
IBM_500K_15A_9L	500,000	15	9
IBM_500K_20A_5L	500,000	20	5

Table 3: Main dataset characteristics.

the effectiveness of the proposed approach.

Two real datasets with different characteristics were acquired by performing different capture sessions with the open-source Network Analyzer tool³ on a backbone link of our campus network⁴. The captured traffic has been aggregated into unidirectional traffic flows, i.e., records that summarize a group of similar and temporally contiguous packets. Each flow is a record that consists of six attributes: *SourceAddress* (source IP address), *DestAddress* (destination IP address), *SourcePort* (source port number), *DestPort* (destination port number), *FlowSize* (flow size expressed in bytes), and *NumPackets* (number of IP packets aggregated in that flow). We will refer to each dataset using the dataset name reported in the first column of Table 3 throughout the article.

Items occurring in the network datasets are aggregated according to the following generalization trees: (1) *SourceAddress* and *DestAddress* attributes are associated with the generalization tree reported in Figure 2(a). More specifically, IP addresses are generalized as the corresponding 24-bit *subnet* if they are local to our campus network or as *external addresses* otherwise. (2) *SourcePort* and *DestPort* are associated with the generalization tree in Figure 2(b), which aggregates ports into three well-known aggregation values (*well known*, *registered*, *dynamic*). (3) Both *FlowSize* and *NumPackages* have been discretized using the 4-bin unsupervised equi-frequency discretization, i.e., the same number of objects is (approximately) assigned to each interval. In Section 5.4 we analyze the impact of the discretization process

³Analyzer 3.0, <http://analyzer.polito.it>. Lastly accessed: December 13, 2013

⁴Due to privacy concerns, the network traffic datasets are not publicly available.

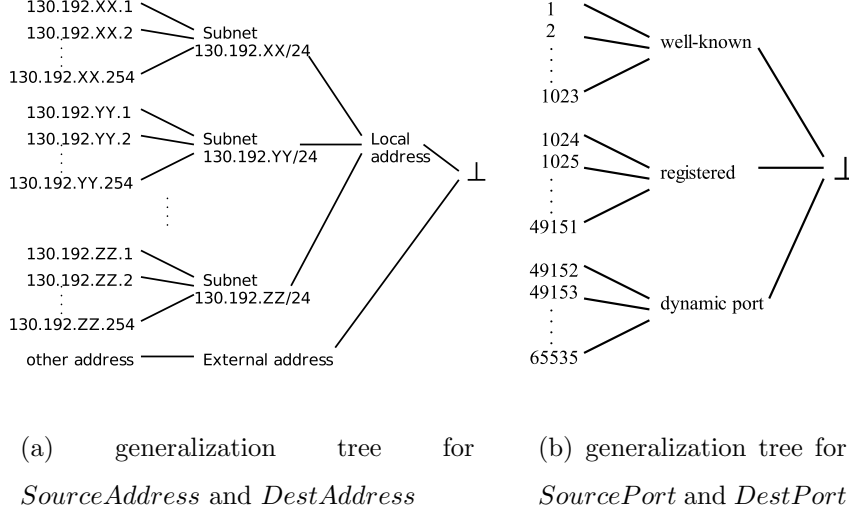


Figure 2: Network datasets. Generalization trees for IP addresses and ports.

on the mining result and we discuss the motivations behind the use of the equi-frequency discretization in the performed experiments. For *FlowSize* and *NumPackets* no high-level item aggregations have been defined. Hence, the corresponding items are directly aggregated into the root node.

UCI benchmark datasets. We also evaluated the performance of our approach on 17 real-life benchmark datasets, which were retrieved from the UCI repository [13]. Table 3 summarizes the main features of the tested datasets, which show rather different characteristics in terms of number of records and attributes.

For each tested dataset a 3-height taxonomy is generated. The taxonomy generation procedure over the real-life UCI datasets is performed as follows. To generate the generalization trees over the continuous data attributes (e.g., age, flnwgt, education-num, capital-gain, capital-loss, and hours-per-week), we applied a 10-bin equi-frequency discretization. Discretized item values

are exploited to aggregate pairs of consecutive item values. Instead, generalization trees over nominal data attributes are analyst-provided. If no meaningful aggregation is available, data items are directly aggregated into the root node. The UCI datasets and taxonomies used in the experimental evaluation are available for research purposes at [12]. A description of a representative UCI dataset coming from the census domain is reported below.

Adult dataset. Adult collects census data about American people (e.g., education, occupation, marital status, race, and sex). For the Education, Marital-status, and Native-country nominal attributes we defined the following generalization trees:

- Education (1st grade, ..., 12th grade, Bachelors, Masters, Doctorate, Preschool) \rightarrow No-College / College / ... / Post-College
- Marital-status (Divorced, Civil married, Church married, Separated, Widowed) \rightarrow Married / Non-Married
- Native-country \rightarrow Europe / America / Asia / Oceania / ... / South-Africa

For the remaining nominal attributes no meaningful item aggregations (disregarding the root node) are defined.

Synthetic datasets. To generate synthetic data we used the function 2 of the Quest IBM synthetic dataset generator [18], which has first been exploited by [23] in the context of data classification. The data generator automatically produces structured datasets that consist of a user-specified number of records and attributes. To automate the taxonomy generation procedure we extended the data generator source code as follows. Once a

user has specified the required taxonomy height h , for each attribute the item values are treated as taxonomy leaf nodes, sorted into lexicographical order, and clustered into a subset of equi-frequency bins. Each bin is associated with a generalized item which aggregates all the group members. Next, the high-level bins are further aggregated with each other and the procedure iterates until all the items are clustered into a single group (i.e., the root node). At each generalization level the bin frequency is automatically computed based on taxonomy height and attribute domain cardinality. For example, if the taxonomy height is equal to 3 then a 27-value attribute domain is partitioned into 9 equi-frequency bins at level 1, 3 equi-frequency bin at level 2, and a single level-3 bin. The extended generator version is available at [12]. Hereafter the synthetic datasets will be identified using the following notation: $IBM_num_records_num_attributesA_num_levelsL$, where $num_records$ and $num_attributes$ are the dataset cardinality and dimensionality, respectively, while num_levels is the taxonomy height.

5.2. Pattern analysis and expert-driven validation

We demonstrated the effectiveness of the proposed approach on both real and synthetic datasets with different data distributions and taxonomy characteristics. More specifically, we analyzed: (i) the usefulness of the extracted Max-EGIs for supporting the knowledge discovery process on the network traffic datasets (see Section 5.2.1), and (ii) the characteristics of the mined patterns on real and synthetic datasets (see Section 5.2.2).

Destination address subnet	Pattern ID	Max-EGI	Sup X ∩ S	Sup S part
130.192.XX/24	1	{(DestA,130.192.XX/24),(SourceA,external)} ∪ {(DestA,130.192.XX.DD),(SourceA,213.92.WW.LL)}	0.28%	0.10%
	2	{(DestA,130.192.XX.DD), (SourceA,213.92.WW.LL)}	0.10%	-
130.192.YY/24	3	{(DestA,130.192.YY/24), (SourceA,external)} ∪ {(DestA,130.192.YY.EE), (SourceA,195.210.AA.HH)} {(DestA,130.192.YY.GG), (SourceA,211.112.BB.FF)}	0.68%	0.32%
	4	{(DestA,130.192.YY.EE), (SourceA,195.210.AA.HH)}	0.22%	-
	5	{(DestA,130.192.YY.GG), (SourceA,211.112.BB.FF)}	0.10%	-
130.192.ZZ/24	6	{(DestA,130.192.ZZ.MM)}	0.25%	-

Table 4: Network dataset NetD2: examples of Max-EGIs relative to the subnets 130.192.XX/24, 130.192.YY/24, and 130.192.ZZ/24. min_sup=0.1%.

5.2.1. Max-EGI expert-driven validation

We validated the usefulness of the Max-EGIs mined from the *NetD2* dataset with the help of a domain expert. As an example, Table 4 reports a worthwhile subset of Max-EGIs selected by the analyst. They are related to the IP traffic directed to the campus subnets 130.192.XX/24, 130.192.YY/24, and 130.192.ZZ/24. To compare Max-EGIs with traditional itemsets, Table 5 also reports the corresponding set of frequent itemsets, generalized and not.

The expert deemed the aforementioned Max-EGIs to be valuable for targeted traffic flow analysis. For example, the Max-EGI with PID (1) in Table 4 describes the portion of traffic generated from external IP addresses that is directed to the campus subnet 130.192.XX/24. Item (DestAddress,130.192.XX/24) generalizes all the IP destination addresses that belong to subnet 130.192.XX/24, while (SourceAddress,external) represents all the external IP sources. The analysis of the characteristics of the incoming traffic generated by external

Destination A subnet	Pattern ID	Traditional itemset	Sup
130.192.XX/24	1	{(DestA,130.192.XX/24), (SourceA,external)}	0.38%
	2	{(DestA,130.192.XX.36), (SourceA,213.92.WW.LL)}	0.10%
130.192.YY/24	3	{(DestA,130.192.YY/24), (SourceA,external)}	1%
	4	{(DestA,130.192.YY.EE), (SourceA,195.210.AA.HH)}	0.22%
	5	{(DestA,130.192.YY.GG), (SourceA,211.112.BB.FF)}	0.10%
130.192.12/24	6	{(DestA,130.192.ZZ/24)}	0.33%
	7	{(DestA,130.192.ZZ.MM)}	0.25%

Table 5: Network dataset NetD2: examples of traditional generalized itemsets relative to the subnets 130.192.XX/24, 130.192.YY/24, and 130.192.ZZ/24. min_sup=0.1%

IP addresses could be helpful for performing campus network monitoring and service shaping. To delve into such traffic flows, the expert first examines the frequent descendant set S of the Max-EGI with PID (1), because it contains a reduced number of patterns. The traffic flows related to the pair of source and destination IP addresses 213.92.WW.LL and 130.192.XX.DD frequently occur in the analyzed trace. Source IP address 213.92.WW.LL turns out to be associated with a video streaming server, which provides public network services. Similarly, the analysis of the Max-EGI with PID (3) highlights a couple of other external public services. They are associated with the external IP addresses 195.210.AA.HH and 211.112.BB.FF, respectively, and they are reachable from the campus network subnet 130.192.YY/24. The traditional itemsets with IDs 1 and 3 in Table 5 correspond to the X part of the Max-EGIs (1) and (3) respectively. However, they provide a partial and somehow misleading information, because their support count considers both frequent and infrequent descendant contributions. From the analysis of the

Max-EGI mining results it appears that some IP external addresses should be analyzed apart from their common high-level aggregation. To perform these analyses, the expressiveness of traditional high-level itemsets is shown to be rather low compared to that of Max-EGIs.

Examples of real contexts of use for the mined Max-EGIs are: (i) the analysis and discovery of network service malfunctioning, which may trigger targeted reactions, (ii) the profiling of network user activities, and (iii) the shaping of the network service provision. For example, bandwidth could be shaped differently for frequently asked and underused public services. Similarly, the unexpectedly frequent traffic flows directed to a specific IP subnet could be monitored carefully because they may hide a potential system malfunctioning or a denial of service attack.

Pattern ID	Max-EGI	Sup X \cap S	Sup S part
1	$\{ \{ (SourceA, external), (DestA, local), (DestPort, well-known) \}$ \cap $\{ \{ (SourceA, external), (DestA, 130.192.LL), (DestPort, well-known) \} \cap$ $\{ \{ (SourceA, 213.209.XX.TT), (DestA, 130.192.LL.ZZ), (DestPort, 815) \} \}$ \dots $\{ (SourceA, external), (DestA, 130.192.YY), (DestPort, well-known) \}$ $\{ (SourceA, external), (DestA, 130.192.VV), (DestPort, well-known) \} \}$	0.28%	2.44%
2	$\{ (SourceA, external), (DestA, local), (DestPort, dynamic-port) \}$ \cap $\{ \{ (SourceA, external), (DestA, 130.192.XX), (DestPort, dynamic-port) \} \cap$ $\{ \{ (SourceA, 217.45.TT.PP), (DestA, 130.192.XX.ZZ), (DestPort, 57403) \} \}$ \dots $\{ (SourceA, external), (DestA, 130.192.YY), (DestPort, dynamic-port) \}$ $\{ (SourceA, external), (DestA, 130.192.VV), (DestPort, dynamic-port) \} \}$	0.30%	5.37%

Table 6: Network dataset NetD2: examples of complex Max-EGIs. min_sup=0.1%

Table 6 reports two examples of more “complex” Max-EGIs, which contain in their S set other Max-EGIs with $S \neq \emptyset$. The first Max-EGI describes the traffic generated from external IP addresses and directed to well-known ports. The network expert analyzes the frequent descendant set S of the Max-EGI with PID (1) and she identifies the most frequently accessed local subnets (e.g., DestA,130.192.YY and DestA,130.192.VV). Note that Max-EGI $\{(\text{SourceA}, \text{external}), (\text{DestA}, 130.192.LL), (\text{DestPort}, \text{well-known})\}$ contains $\{(\text{SourceA}, 213.209.XX.TT), (\text{DestA}, 130.192.LL.ZZ), (\text{DestPort}, 815)\}$ is also contained in S . This pattern indicates that a large number of external connections are directed to well-known ports of subnet 130.192.LL, but actually many of them are related to a specific IP address (130.192.LL.ZZ) corresponding to an external video streaming service provider. This information is worthy for network traffic monitoring. Similarly, the Max-EGI with PID (2) in Table 6 can be considered to shape external connections directed to dynamic ports of local IP addresses.

5.2.2. Characteristics of mined patterns

We performed different Max-EGIs and traditional generalized itemset mining sessions from real and synthetic datasets. The choice of the appropriate minimum support threshold value to enforce depends on the analyzed data distribution. During the experiments we noticed that both MAX-EGI MINER and ML_T2L1 produces interesting results by setting the minimum support threshold to 1% on the UCI and synthetic datasets and to 0.1% on the network datasets. Hence, we consider the above thresholds as reference configuration settings on the analyzed datasets. A more thorough discussion of the impact of the support threshold on the algorithm performance is given

in Section 5.3.

For each dataset Table 7 summarizes (i) the number of mined not generalized traditional itemsets, i.e., the level-1 Max-EGIs (see Column 3), (ii) the number of mined traditional level-sharing itemsets with level above 1, i.e., the traditional high-level itemsets (Column 4), and (iii) the number of mined Max-EGIs with level above 1 (Column 5). Since level-1 itemsets are in common between the two mined sets, their count is reported separately. Max-EGIs extend the traditional itemsets by also considering their corresponding frequent descendant set (i.e., the S part). Specifically, a high-level Max-EGI covers only the subset of dataset records that are associated with its infrequent descendants. Neglecting frequent descendant support contributions may affect the support counting of high-level patterns. Changes to the mining result are three-fold: (a) some of the traditional high-level itemsets are pruned, because they become infrequent with respect to the support threshold; (b) some others are enriched with a (not empty) descendant S ; (c) still others remain unchanged (i.e., their frequent descendant set S is empty). To compare Max-EGIs with traditional itemsets, Table 7 reports the number and percentage of traditional itemset changes occurred while extracting Max-EGIs (see Columns 6 and 7), where "changes" consists of itemset pruning (fold (a)) or enrichment (fold (b)).

Based on the results reported in Table 7, 10 UCI datasets out of 17 have a percentage of changed itemsets above 50%. Hence, the mined patterns represent, on average, a significantly different knowledge compared to the traditional level-sharing itemsets. The percentage of changed itemsets is more significant when coping with denser datasets (e.g., 99.05% for the Shut-

Dataset	min_sup (%)	Num. of itemsets of level 1	Num. of Level-sharing itemsets (level > 1)	Num. of Max-EGIs (level>1)	Num. of changed itemsets	Perc. of changed itemsets (%)	Avg. Num. of overlapped Level-sharing itemsets (level>1) per record	Avg. Num. of overlapped Max-EGIs (level>1) per record	Relative overlap reduction (%)
<i>UCI datasets</i>									
adult	1	249,413	368,926	193,384	192,364	52.1	9,216	2,789	69.7
breast	1	13,099	11,497	9,572	5,796	50.4	664	131	80.3
cleve	1	159,439	220,676	130,274	104,902	47.5	3,954	1,539	61.1
crx	1	1,069,561	1,430,270	877,711	734,228	51.3	32,287	10,475	67.6
glass	1	32,556	24,912	13,470	17,433	70.0	854	169	80.2
heart	1	459,708	614,661	334,996	334,400	54.4	8,243	2,803	66.0
iris	1	1,454	665	0	665	100	30	0	100
labor	1	7,027,978	6,546,242	0	6,546,242	100	127,308	0	100
letter	1	55,107	503,395	496,647	31,491	6.3	9,763	8,737	10.5
nursery	1	8,118	4,302	2,593	1,709	39.7	89	34	62.0
pendigits	1	37,978	437,442	431,020	24,849	5.7	7,085	6,400	9.7
pima	1	7,845	10,626	9,721	2,933	27.6	361	189	47.8
shuttle	1	10,847	6,764	1,608	6,700	99.1	1,013	27	97.4
vehicle	1	445,248	4,717,476	4,612,624	324,288	6.9	72,475	65,734	9.3
waveformd	1	786,443	13,589,520	13,494,548	475,716	3.5	236,850	223,243	5.7
wine	1	2,815,380	2,406,677	0	2,406,677	100	16,382	0	100
yeast	1	9,202	8,952	0	8,952	100	255	0	100
<i>Network datasets</i>									
NetD1	0.1	4,357	5,891	5,676	736	12.5	62	53	14.7
NetD2	0.1	1,576	4,603	4,542	349	7.6	61	57	6.6
<i>Synthetic datasets</i>									
IBM_500K_15A_9L	1	467	878,700	859,956	70,639	8.0	68,047	32,720	51.9
IBM_500K_20A_5L	1	467	1,327,092	1,325,794	10,344	0.8	1,054,383	1,048,562	0.6

Table 7: Number of mined Max-EGIs and level-sharing itemsets.

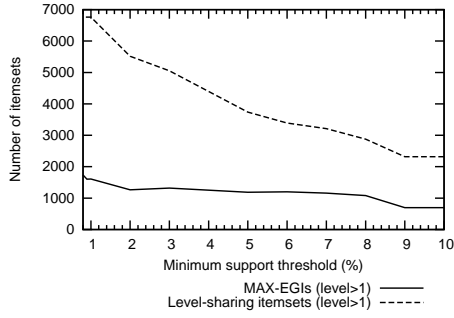
tle dataset), whereas it becomes less relevant on sparser ones (e.g., 7.60% for the *NetD2* network dataset). A dataset is usually considered to be *dense* if it contains a relatively large number of frequent itemsets even when averagely high support thresholds are enforced. Conversely, the dataset is said to be *sparse*. Data sparseness/density appears to be strongly correlated with the percentage of changed (high-level) itemsets. On the one hand, on denser datasets (e.g., Shuttle) a larger number of low-level itemset is likely to be frequent. Hence, the percentage of changed itemsets becomes significant and thus Max-EGI mining is much more effective than traditional itemset min-

ing. In 4 cases out of 17 (i.e., iris, labor, yeast, wine), no Max-EGI with level above 1 is mined, because low-level itemsets already cover most of the dataset records. On the other hand, itemsets generalization on sparser datasets (e.g., Letter, NetD2) prevents the discarding of a large number of infrequent but potentially relevant patterns. However, the percentage of changed itemsets is, on average, more limited.

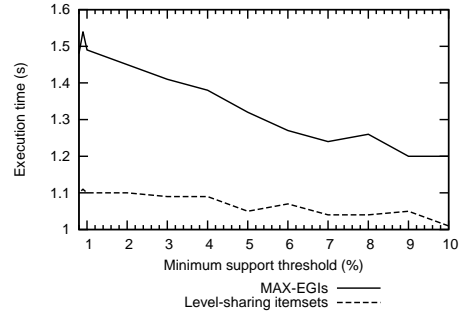
A key issue of traditional itemset mining is that each dataset record r could be covered by many extracted itemsets X_i (i.e., $X_i \subseteq r$) at the same time. In [24] the authors formulated the redescription mining problem as the task of finding sets of patterns that all cover the same set of records. These pattern sets are of interest as they point towards equivalences in the analyzed data facets. In the context of itemset mining the lower the number of “overlapped” itemsets covering the same record is, the more manageable the result becomes for manual inspection. Hence, in Table 7 we also quantitatively compared Max-EGIs with traditional itemsets in terms of the average number of overlapped Max-EGIs (Column 8) and traditional itemsets (Column 9) per record. The relative difference between the aforesaid measures, i.e., $\frac{Column(8)-Column(9)}{Column(8)}$, is also reported (see Column 10). The results empirically demonstrate that Max-EGIs are significantly less overlapped than traditional itemsets on most datasets, in particular on denser ones (e.g., 97.4% relative overlap reduction on Shuttle) for which the percentage of not extracted or changed high-level Max-EGIs is higher.

5.3. Performance analysis

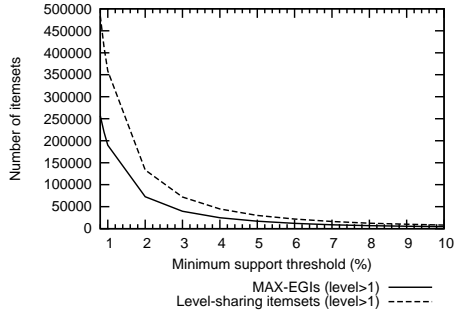
We thoroughly analyzed the performance of our approach in terms of (i) impact of the support threshold on the number of extracted Max-EGIs and



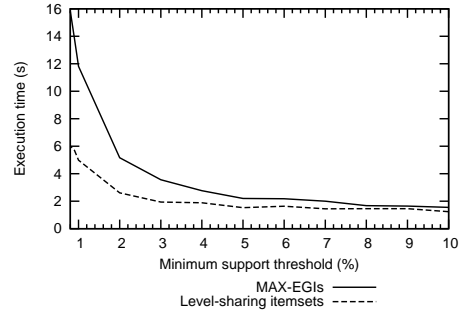
(a) Shuttle: Num. of itemsets



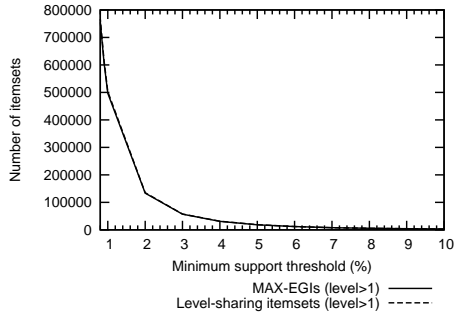
(b) Shuttle: Execution Time



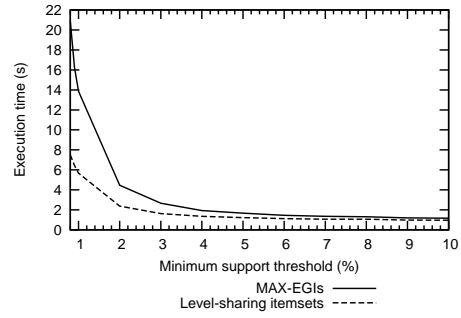
(c) Adult: Num. of itemsets



(d) Adult: Execution Time



(e) Letter: Num. of itemsets



(f) Letter: Execution Time

Figure 3: Effect of the minimum support threshold on the characteristics of the extracted itemsets with level>1 and execution time.

(ii) algorithm execution time.

Impact of the support threshold. Since the support threshold can significantly affect the performance of itemset mining algorithms, we also analyzed its impact on the MAX-EGI MINER and ML-T2L1 performance in terms of number of mined itemsets and algorithm execution time.

Figures 3(a)- 3(f) report the number of extracted high-level patterns (i.e., the Max-EGIs and the traditional level-sharing itemsets with level above 1) and the algorithm execution time spent on three representative UCI datasets by varying the support threshold value in the range [0.8%,10%]. We considered as representatives a very dense (Shuttle), an averagely dense (Adult), and a sparse dataset (Letter), because they yield high, medium, and low percentages of changed itemsets, respectively (see Table 7 at Column (7)).

For all datasets, when higher support thresholds (e.g., 7%) are enforced the selectivity of the support threshold is rather high (see Figures 3(a), 3(c), and 3(e)). In other words, many low-level itemsets become infrequent and, thus, are pruned, whereas some of their high-level ancestors are extracted. Therefore, the informative content of the mining result is rather low, because most of the extracted patterns represent rather generic correlations among data. In contrast, when setting relatively low support values (e.g., 1%) the mining result may contain potentially relevant knowledge. Since most of the high-level itemsets are kept, the curves achieved for the two algorithms have a similar trend (see Figures 3(a), 3(c), and 3(e)). However, the expressiveness of the traditional high-level itemsets degrades while setting low support thresholds, because many of the extracted high-level data correlations does not cover a significant number of uncovered records. Thanks to high-level

itemset pruning, MAX-EGI MINER yields a quite significant cardinality pattern set reduction compared to ML_T2L1. For example, setting `min_sup` to 1% the number of extracted Max-EGIs on Shuttle is 76% lower than the number of traditional level-sharing itemsets. The cardinality reduction is equal to 47% and 1% for Adult and Letter, respectively. Furthermore, an increasing number of traditional itemsets is enriched with a not empty S part. Hence, a higher percentage of changed itemsets is achieved (see Column (7) of Table 7). More specifically, when setting `min_sup` = 1%, the percentages of changed itemsets are 99.1%, 52.1%, and 6.3% on Shuttle, Adult, and Letter, respectively. As expected, the reduction in terms of number of mined itemsets is higher on denser datasets (e.g., Shuttle) and lower on sparser ones (e.g., Letter). For example, on Letter the number of level-sharing itemsets is close to the number of extracted Max-EGIs (the two curves in Figure 3(e) are substantially overlapped).

Algorithm execution time. We compared the execution times of MAX-EGI MINER and ML_T2L1 algorithms on the analyzed datasets. Table 8 reports the ML_T2L1 and MAX-EGI MINER execution time on all the considered datasets (both real and synthetic) by enforcing the reference support threshold. Although MAX-EGI MINER performs a post-processing step at the top of the traditional level-sharing itemsets, its execution time is comparable with that of ML_T2L1 with most of the tested support threshold values⁵ (see Figures 3(b), 3(d), and 3(f)).

⁵The gap in Figure 3(b) is negligible because the runs last less than 2 seconds.

Dataset	min_sup (%)	Execution time of ML.T2L1 (s)	Execution time of MAX-EGI MINER (s)
<i>UCI datasets</i>			
adult	1	5.0	11.8
breast	1	0.5	0.9
cleve	1	2.7	6.9
crx	1	11.4	38.0
glass	1	0.6	1.6
heart	1	3.2	8.5
iris	1	0.2	0.4
labor	1	71.1	295.2
letter	1	5.7	13.9
nursery	1	0.7	1.0
pendigits	1	4.2	10.6
pima	1	0.5	0.8
shuttle	1	1.1	1.5
vehicle	1	38.7	143.6
waveformd	1	112.2	876.5
wine	1	2.8	7.2
yeast	1	0.5	0.8
<i>Network datasets</i>			
NetD1	0.1	0.6	0.9
NetD2	0.1	0.8	1.4
<i>Synthetic datasets</i>			
IBM_500K_15a_9l	1	43.1	63.2
IBM_500K_20a_5l	1	42.4	59.5

Table 8: Execution time of MAX-EGI MINER and ML.T2L1.

5.4. Effect of the discretization method

Similar to traditional itemset mining, continuous data need to be discretized before executing the MAX-EGI MINER algorithm. Since the discretization process can affect the characteristics of the mining result, we analyzed the Max-EGI performance by adopting different well-known dis-

cretization methods. For the sake of brevity, in the following we reported the results achieved using two representative and established methods, i.e., equi-width and equi-depth [31]. Equi-width discretization focuses on partitioning the attribute domain into a fixed number of equi-width bins, whereas the equi-depth discretization method generates a set of bins which contain approximately the same number of objects.

We tested many datasets composed of continuous attributes solely (disregarding the class label). Since the effect of discretization is similar on all the tested datasets, we reported just the results achieved on a fairly complex dataset (Vehicle). Figures 4(a) and 4(b) report the number of Max-EGIs with level above 1 achieved using equi-width and equi-depth discretization, respectively. For each discretization, every plotted curve is related to a discretization process with a different number of bins. For example, the curve denoted by *equi-width X-Y* represents the mining results achieved on a source dataset discretized as follows: (i) a X-bin equi-width discretization step at level 1 (i.e., at the data item level) and (ii) a Y-bin equi-width discretization step to generate level-2 items (see Section 5.1).

As expected, increasing the number of bins the number of mined Max-EGIs decreases, because the average support of the data items decreases. By enforcing relatively low support threshold values (e.g., 1%), the equi-width discretization method yields an averagely higher number of Max-EGIs compared to the equi-depth discretization, because some of the generated bins (and their corresponding combinations) become on average frequent, whereas all the others are discarded early. In contrast, the equi-depth discretization method generates a large number of level-1 and level-2 items for which the

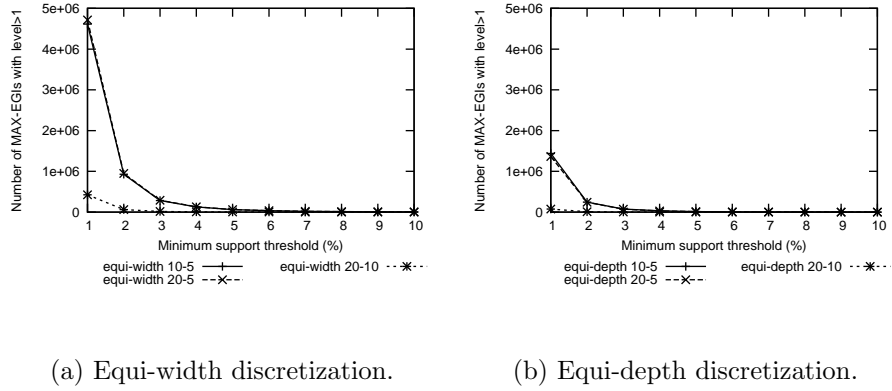


Figure 4: Vehicle dataset: Impact of the discretization step on the number of mined Max-EGIs (level>1). $\text{min_sup} = 1\%$.

observed frequency in the analyzed data is approximately the same. Hence, to some extent, the equi-depth discretization method appears to be more suitable for coping with relatively sparse datasets and thus it has been adopted as reference preprocessing step throughout the article.

5.5. Scalability

We analyzed the scalability, in terms of execution time, of the MAX-EGI MINER algorithm on synthetic datasets with (i) the number of dataset records, (ii) the number of attributes, and (iii) the taxonomy height. MAX-EGI MINER extractions were performed by setting the minimum support threshold to 1%. Note that, for most of the tested configurations, the number of generated patterns is in the order of 10^5 or even higher. Figure 5 reports the achieved results, which are discussed in the following.

Scalability with the number of records. The results on synthetic data were achieved varying the dataset cardinality in the range $[100,000, 1,000,000]$,

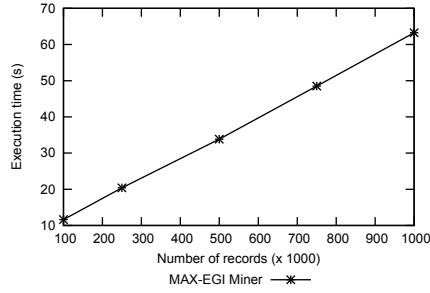
while setting the number of attributes and the taxonomy height to 15 and 5, respectively. The results, reported in Figure 5(a), show that the MAX-EGI MINER execution time scales roughly linearly with the number of records, because the data distribution does not vary substantially while the dataset cardinality increases.

Scalability with the number of attributes. The impact of the number of attributes on the MAX-EGI MINER execution time was tested by varying the dataset dimensionality in the range $[10, 20]$, while setting the number of records and the taxonomy height to 500,000 and 5, respectively. The results, reported in Figure 5(b), show that the execution time scales more than linearly with the number of attributes, because of the combinatorial increase in the number of generated combinations.

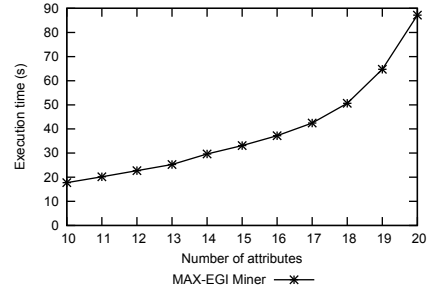
Effect of the taxonomy height. We varied the taxonomy height in the range $[2, 8]$ and we set the dataset cardinality and the number of attributes to 500,000 and 15, respectively. The results, reported in Figure 5(c), show that the MAX-EGI MINER execution time scales more than linearly with the taxonomy height. In fact, the more complex taxonomies you consider, the more high-level candidate Max-EGIs you generate.

6. Related work

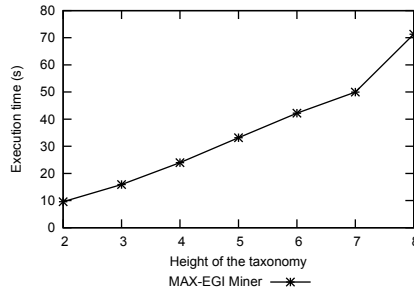
The problem of discovering generalized itemsets and association rules has first been addressed in [27] to perform market basket analysis. The authors propose a generalized itemset mining algorithm that extracts frequent itemsets by considering, for each item, all of its parents in a taxonomy supplied



(a) Scalability with the number of records. 15 attributes, 5 levels.



(b) Scalability with the number of attributes. 500,000 records, 5 levels.



(c) Scalability with the taxonomy height. 500,000 records, 15 attributes.

Figure 5: IBM dataset: MAX-EGI MINER algorithm scalability. $\text{min_sup} = 1\%$.

with the analyzed data. A similar problem has been addressed in [28] when dealing with quantitative data. However, since the candidate frequent itemsets are generated by evaluating the taxonomy exhaustively, a very large number of patterns is typically extracted. To overcome this issue, many related approaches [3, 10, 19, 29, 30] focus on reducing the complexity of the

mining process by preventing the generation of uninteresting or redundant candidates. For example, in [3] the authors propose to push user-provided boolean constraints, which enforce the presence or the absence of a given item combination, into the mining process. Similarly, the approach presented in [29] also takes subset-superset and parent-child item relationships into account. An attempt to constrain the generalized itemset extraction based on the cardinality of its descendant set has been made in [10]. To make the mining result practically manageable by domain experts, the authors propose to extract high-level itemsets only when their corresponding descendant set is so large that its manual inspection is practically unfeasible. In parallel, the approaches presented in [19, 30] focus on discovering closed and maximal itemsets, which are compact frequent itemset subsets [25], in the presence of taxonomies. Unlike [3, 10, 19, 29, 30], the approach presented in this article neither proposes novel itemset mining constraints nor focuses on selecting a worthwhile subset of traditional generalized itemsets. Instead, it proposes two novel generalized itemset types that are characterized by a higher expressive power compared to traditional high-level itemsets. A lazy support-driven approach to generalized itemset mining has also been presented [4]. The authors propose an algorithm that discovers the frequent itemsets and all of the (traditional) generalized itemsets that have at least one infrequent descendant. Furthermore, a recent extension of [4], presented in [5], addresses the pushing of analyst-provided constraints during the lazy itemset extraction process. The focus of this article differs, to a large extent, from that of the above-mentioned approaches because, rather than selecting a worthwhile itemset subset, in this article we propose two new pattern forms,

namely the EGI and the Max-EGI. The newly proposed generalized itemset types significantly improve the expressiveness of traditional high-level itemsets. Analyses of the item correlation changes across the taxonomy levels have been performed in [6, 9]. Both [6] and [9] consider high-level itemsets whose low-level descendants have contrasting item correlation, rather than mining expressive generalized itemsets.

A parallel effort have been devoted to proposing optimization strategies to accomplish the generalized itemset mining task efficiently [4, 15, 17, 26]. For instance, in [17] a faster support counting is proposed to compute the TID intersection in algorithms that exploit the vertical data format [35]. The authors in [15] proposed an optimization strategy based on a top-down hierarchy traversal. The proposed approach identifies in advance the itemsets that cannot be frequent in a transactional dataset by exploiting the Apriori principle [2]. To further prune the search space, the authors also propose to select a worthwhile subset of generalized itemsets, namely the level-sharing itemsets. More recently, in [20, 21] efficient generalized itemset and association rule mining in a fuzzy context has also been addressed. Similar to [15], we exploit the concept of level-sharing itemset to reduce the cardinality of the extracted patterns. However, since we address Max-EGI mining instead of traditional itemset mining, our focus is radically different.

In the last years, a large body of work has also been devoted to selecting succinct yet informative pattern sets (e.g., [7, 11, 22, 32]). They commonly evaluate the global quality of the mined set by means of entropy-based or statistics-based strategies. Unlike [7, 11, 22, 32], the approach presented in this article evaluates the interest of each individual pattern based on the

characteristics of a subset of descendant itemsets. Since pattern set selection is commonly applied as a postprocessing step, our approach may be considered to be orthogonal with respect to the previously mentioned ones.

7. Conclusions and future work

This article presents two new types of generalized itemsets, namely the EGI and the Max-EGI. The proposed patterns are more expressive than traditional ones, because the support count of high-level patterns only considers the dataset records not covered by any of their frequent descendants. Furthermore, the list frequent descendants is appended to each high-level itemset. To efficiently tackle the Max-EGI mining problem at the top of traditional itemsets, a novel algorithm has been proposed. The experimental results, achieved on both real and synthetic datasets, demonstrate the effectiveness of the proposed approach to discover interesting and expressive patterns from the analyzed datasets as well as the algorithm scalability.

Up to the present, EGIs and Max-EGIs only contain level-sharing itemsets [15]. Furthermore, their extraction is driven only by the minimum support constraint [27]. As future work, we plan to (i) mine EGIs associated with not level-sharing itemsets and (ii) push more complex pattern selection constraints (e.g., [8, 22, 34]) into the Max-EGI mining process, and (iii) tailor the Max-EGI mining process to continuous data to avoid the preliminary discretization step.

References

- [1] R. Agrawal, T. Imielinski, and Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD 1993*, pages 207–216, 1993.
- [2] R. Agrawal and R. Srikant. Fast algorithm for mining association rules. In *VLDB 1994*, pages 487–499, 1994.
- [3] R. Agrawal and R. Srikant. Mining association rules with item constraints. In *KDD 1997*, pages 67–73, 1997.
- [4] E. Baralis, L. Cagliero, T. Cerquitelli, V. D’Elia, and P. Garza. Support driven opportunistic aggregation for generalized itemset extraction. In *IEEE Conf. of Intelligent Systems*, pages 102–107, 2010.
- [5] E. Baralis, L. Cagliero, T. Cerquitelli, and P. Garza. Generalized association rule mining with constraints. *Inf. Sci.*, 194:68–84, 2012.
- [6] M. Barsky, S. Kim, T. Weninger, and J. Han. Mining flipping correlations from large datasets with taxonomies. *Proc. VLDB Endow.*, 5(4):370–381, Dec. 2011.
- [7] B. Bringmann and A. Zimmermann. The chosen few: On identifying valuable patterns. In *ICDM’07*, pages 63–72, 2007.
- [8] L. Cagliero. Discovering temporal change patterns in the presence of taxonomies. *IEEE Trans. Knowl. Data Eng.*, 25(3):541–555, 2013.
- [9] L. Cagliero, T. Cerquitelli, P. Garza, and L. Grimaudo. Misleading generalized itemset discovery. *Expert Syst. Appl.*, 41(4):1400–1410, 2014.

- [10] L. Cagliero and P. Garza. Itemset generalization with cardinality-based constraints. *Inf. Sci.*, 244:161–174, 2013.
- [11] T. Calders and B. Goethals. Mining all non-derivable frequent itemsets. In *PKDD'02*, pages 74–85, 2002.
- [12] DBDMG. Database and data mining group website: <http://dbdmg.polito.it/wordpress/research/expressive-generalized-itemsets/>. last accessed: 15/12/2013, 2013.
- [13] A. Frank and A. Asuncion. UCI machine learning repository. available at <http://archive.ics.uci.edu/ml>. last accessed: 30/09/2012, 2012.
- [14] T. F. Gharib. An efficient algorithm for mining frequent maximal and closed itemsets. *Int. J. Hybrid Intell. Syst.*, 6(3):147–153, Aug. 2009.
- [15] J. Han and Y. Fu. Mining multiple-level association rules in large databases. *IEEE TKDE*, 11(5):798–805, 1999.
- [16] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *ACM SIGMOD 2000*, pages 1–12, 2000.
- [17] J. Hipp, A. Myka, R. Wirth, and U. Guntzer. A new algorithm for faster mining of generalized association rules. In *PKDD'98*, pages 74–82, 1998.
- [18] IBM. IBM Quest Synthetic Data Generation Code, 2009.
- [19] D. Kunkle, D. Zhang, and G. Cooperman. Mining frequent generalized itemsets and generalized association rules without redundancy. *J. Comput. Sci. Technol.*, 23(1):77–102, 2008.

- [20] C. M. Kuok, A. Fu, and M. H. Wong. Mining fuzzy association rules in databases. *SIGMOD Record*, 27:41–46, 1998.
- [21] Y.-C. Lee, T.-P. Hong, and T.-C. Wang. Multi-level fuzzy mining with multiple minimum supports. *Expert Syst. Appl.*, 34(1):459–468, 2008.
- [22] M. Mampaey, N. Tatti, and J. Vreeken. Tell me what I need to know: succinctly summarizing data with itemsets. In *ACM SIGKDD’11*, pages 573–581, 2011.
- [23] M. Mehta, R. Agrawal, and J. Rissanen. SLIQ: A fast scalable classifier for data mining. In *EDBT’96*, pages 18–32, 1996.
- [24] L. Parida and N. Ramakrishnan. Redescription mining: Structure theory and algorithms. In *AAAI’05*, pages 837–844. AAAI Press, 2005.
- [25] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT’99*, pages 398–416, 1999.
- [26] I. Pramudiono and M. Kitsuregawa. Fp-tax: tree structure based generalized association rule mining. In *DMKD ’04*, pages 60–63, 2004.
- [27] R. Srikant and R. Agrawal. Mining generalized association rules. In *VLDB 1995*, pages 407–419, 1995.
- [28] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *ACM SIGMOD 1996*, pages 1–12, 1996.
- [29] K. Sriphaew and T. Theeramunkong. A new method for finding generalized frequent itemsets in association rule mining. In *Proceeding of*

the VII International Symposium on Computers and Communications, pages 420–431, 2002.

- [30] K. Sriphaew and T. Theeramunkong. Fast algorithms for mining generalized frequent patterns of generalized association rules. *IEICE Transactions on Information and Systems*, 87(3):761–770, 2004.
- [31] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.
- [32] N. Tatti and M. Mampaey. Using background knowledge to rank itemsets. *Data Min. Knowl. Discov.*, 21:293–309, 2010.
- [33] T. Uno, L. Kiyomi, and H. Arimura. LCM (ver 2): Efficient mining algorithms for frequent/closed/maximal itemsets. In *FIMI '04*, 2004.
- [34] M. J. Zaki. Generating non-redundant association rules. In *KDD'00*, pages 34–43, New York, NY, USA, 2000. ACM.
- [35] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *KDD*, pages 283–286, 1997.