Generalized itemset discovery by means of opportunistic aggregation

Elena Baralis, Tania Cerquitelli, Vincenzo D'Elia

Politecnico di Torino

Abstract:

Association rule extraction is a widely used exploratory technique which has been exploited in different contexts (e.g., network traffic characterization, biological data, medical images). However, association rule extraction, driven by support and confidence constraints, entails (i) generating a huge number of rules which are difficult to analyze, or (ii) pruning rare itemsets, even if their hidden knowledge might be relevant. To address the above issues, this chapter presents a novel algorithm, called GenIO (GeNeralized Itemset DiscOverer), to analyze correlation among data by means of generalized itemsets, which provide a powerful tool to efficiently extract hidden knowledge, discarded by previous approaches. The proposed technique exploits (user provided) taxonomies to drive the pruning phase of the extraction process. Instead of extracting itemsets for all levels of the taxonomy and post-pruning them, the GenIO algorithm performs a support driven opportunistic aggregation of itemsets. Generalized itemsets are extracted only if items at a lower level in the taxonomy are below the support threshold. Experiments performed in the network traffic domain show the efficiency and the effectiveness of the proposed algorithm.

Keywords: Generalized itemset mining, knowledge discovery, association rule mining, data mining techniques, network traffic data analysis.

INTRODUCTION

Data mining is a relatively new research field focused on automatic extraction of useful information hidden in huge databases. It combines traditional database system techniques with information retrieval, statistics, and artificial intelligence algorithms. The goal of data mining is to automatically identify interesting patterns in data which highlight useful and previously unknown information. The extracted information may be used to support business decisions. For example, association rule extraction (Agrawal et al., 1993) allows the identification of correlations among sold items which can be used by marketers to implement more efficient marketing strategies.

Association rules are extracted from a database D. D is a collection of transactions, where each transaction is a set of data items. Association rules are usually represented in the form $A \Rightarrow B$, where A and B are itemsets, i.e., sets of data items. Each rule is usually characterized by the support and confidence quality indices (Han and Kamber, 2006). The support is the prior probability of A and B (i.e., its observed frequency in the data set). The confidence is the conditional probability of B given A and characterizes the "strength" of a rule. Since association rules identify collections of itemsets that are statistically related (i.e., frequent), they find application in a wide range of different domains, including medical images (Antonie et al., 2001), biological data (Cong et al., 2004), and network traffic data (Baldi et al., 2005). Association rule mining (Agrawal et al., 1993; Agrawal and Srikant, 1994; Han et al., 2000; Mannila et al., 1994; Orlando et al., 2003; Toivonen, 1996) is a two-step process: (i) Frequent itemset extraction and (ii) association rule generation from frequent itemsets. Research activity usually focuses on defining efficient algorithms for itemset extraction, which represents the most computationally intensive knowledge extraction task in association rule mining (Agrawal and Srikant, 1994).

The traditional itemset mining problem can be described as follows. Given a database of transactions and a minimum support threshold, find all itemsets whose support is above the given threshold. However, such threshold may prevent infrequent, but potentially relevant, itemsets from being extracted, while enforcing a very low threshold makes the task unfeasible. To address the above issues, this chapter proposes a novel algorithm, called GenIO (GeNeralized Itemset DiscOverer) to support data analysis by automatically extracting higher level, more abstract correlations from data. GenIO extends the concept of multi-level itemsets (Han and Kamber, 2006) by performing an opportunistic extraction of generalized itemsets. The proposed algorithm exploits (user provided) taxonomies to drive the pruning phase of the extraction process and efficiently extract these itemsets. Instead of extracting itemsets for all levels of the taxonomy and post-pruning them (Han and Fu, 1999), the proposed generalization technique over the taxonomy is support driven, i.e., it generalizes an itemset only if the items composing it (at a lower level in the taxonomy) are below the support threshold. If the generalized itemset is above the support threshold, the generalization process stops, otherwise it climbs up the taxonomy again. Thus, GenIO automatically performs generalization only when needed to exceed the support threshold. Differently from post-pruning approaches (Han and Fu, 1999), which require setting an appropriate support threshold, our approach prevents pruning rare itemsets.

GenIO is implemented by customizing the Apriori algorithm (Agrawal and Srikant, 1994) to efficiently extract generalized itemsets, with variable support thresholds, from structured data. The proposed algorithm has been tested on network traffic data for two reasons: (i) A huge amount of data may be collected in a short time, (ii) in very large traffic network traces it is hard to detect correlations among data and identify anomalies because of the excessively detailed granularity of information (e.g., IP addresses). In this context, the generalization approach proposed in GenIO proves to be particularly effective. Experiments, performed on different network dumps, show the efficiency and the effectiveness of the proposed algorithm.

MOTIVATING EXAMPLE

Consider an ftp server on port 21 having address *130.192.15.17*. To describe the activity of a client connecting to this server, an itemset in the form

$$(source-address = 154.125.3.2, destination-address = 130.192.15.17, destination-port = 21)$$

should be extracted. Since such triplet is infrequent in a very large traffic network trace, extracting this itemset would require enforcing a very low support threshold, which makes the task unfeasible. However, a higher level view of the network may be provided by the following generalized itemset

$$(source-address = 154.125.3.0/17, destination-address = 130.192.15.17, destinatin-port = 21)$$

which shows a generalization of the source-address attribute (i.e., a subnet including IP addresses 154.125.3.0/17). This itemset is characterized by a larger support and shows that the subnet detected by the generalization process is actually generating most of the traffic. Thus, it provides valuable knowledge for network monitoring. Furthermore, the number of different items (e.g., the specific address *source-address*=154.125.3.0) in network traffic may be very large. Hence, itemsets at this detail level may provide hardly interpretable knowledge. Our generalization approach allows preserving a detailed view of frequent items (e.g., *destination-address*=130.192.15.17), while generalizing at a higher taxonomy level infrequent items (e.g. *source-address*=154.125.3.0).

PROBLEM STATEMENT

Generalized rules extend the notion of multi-level association rules (Han and Fu, 1999; Kaya and Alhajj, 2004), which provide a higher level domain abstraction. In the following, we formally define the generalized itemset extraction problem, which represents the most computationally intensive knowledge extraction task in association rule mining (Agrawal and Srikant, 1994). Examples are selected from the network traffic domain.

Definition 1. *Item.* Let $T = \{t_1, t_2, \ldots, t_n\}$ be a set of data features, called attributes. An item $t_i = value_i \in \Omega_i$ assigns the value $value_i$ to attribute t_i. Ω_i is the domain of attribute t_i. Ω_i may be either a categorical or a continuous domain.

Definition 2. *Itemset.* Let $I = \{t_1 = value_1, t_2 = value_2, \ldots, t_n = value_n\}$ contain the enumeration of all items in the dataset. An itemset $X \subseteq I$ is a set of items.

Definition 3. *Structured dataset.* Let $T = \{t_1, t_2, \ldots, t_n\}$ be a set of attributes and $I = \{t_1 = value_1, t_2 = value_2, \ldots, t_n = value_n\}$ a set of items. A structured dataset D is a collection of records, where each record r is a collection of items $t_i = value_i$, one for each attribute in T.

A trace in the network traffic domain is a structured dataset which holds information on the stream of packets. Each record of the trace is a set of items $t_i = value_i$. For example, the attribute t_i may be the IP destination address, while its value $value_i$ is the value captured from the net (e.g., 130.192.3.17).

Definition 4. *Support.* Let D be a structured dataset and X an itemset. The support of X is the (observed) frequency of X in D.

An aggregation hierarchy may be defined by means of rewriting rules.

Definition 5. *Rewriting Rule.* Let t_i be an attribute and Ω_i its domain. A rewriting rule RR_i is a pre-determined hierarchy of aggregations over values in Ω_i. RR_i is a tree whose leaves are values in Ω_i. Each non-leaf node in the tree is an aggregation of its children, which may be further generalized by its father. Children nodes are mutually exclusive and collectively exhaustive.

Consider, for example, the destination-port address attribute. A simple $RR_{destination-port}$ is shown in **Figure 1**. The $RR_{destination-port}$ assigns *well known* if the destination-port value is lower than 1024, *registered* if in the range 1024-49151, *dynamic* otherwise. The root of $RR_{destination-port}$ aggregates all values allowed for the destination-port address attribute.

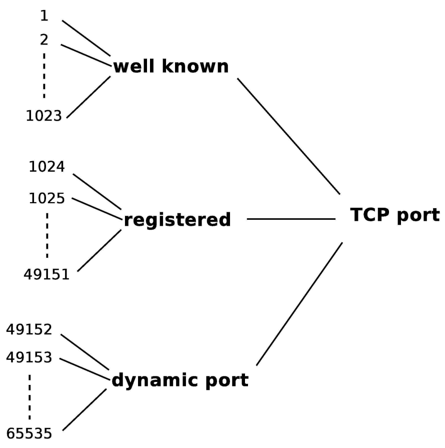


Figure 1. A rewriting rule $RR_{destination-port}$ for the destination port attribute

A taxonomy is a collection of rewriting rules. It needs to be unambiguous, to guarantee aggregation correctness and termination of the rewriting process. This property is proved by Theorem 1 below.

Definition 6. *Taxonomy.* Let $T = \{t_1, t_2, \ldots, t_n\}$ be a set of data features, $\Theta = \{RR_1, RR_2, \ldots, RR_n\}$ a set of rewriting rules defined on T, and $I = \{t_1 = value_1, t_2 = value_2, \ldots, t_n = value_n\}$ a set of items. A taxonomy $\Gamma = \bigcup_{i \in k} RR_i$, where $\Gamma \subseteq \Theta$ and $k < n$, is a forest of rewriting rules. Γ contains at most one rewriting rule RR_i for each data feature t_i in T.

Theorem 1. Let Γ be a taxonomy defined by Definition 6. Γ is not ambiguous.

Proof. Since each rewriting rule RR_i in Γ is defined on a different attribute t_i, it trivially follows that no attribute has an ambiguous aggregation. ■

In the network traffic domain many different rewriting rules may be defined for the *source-address, destination-address, source-port, destination-port* attributes. To guarantee aggregation correctness and termination of the rewriting process, any taxonomy in this domain may contain at most a single rule for each attribute.

Definition 7. *Generalized item.* Let t_i be an attribute, Ω_i its domain, and RR_i a rewriting rule defined over values in Ω_i. A generalized item $t_i = \exp ression_i$ assigns the value $\exp ression_i$ to attribute t_i. $\exp ression_i$ is an aggregation value defined in RR_i.

Consider again the $RR_{destination-port}$ shown in Figure 1. $RR_{destination-port}$ defines three new aggregation values (i.e., *well known, registered, dynamic*) which can appear in generalized items $destination-port = \exp ression_{destination-port}$. $\exp ression_{destination-port}$ may take the values *well known, registered, dynamic* as defined by the rewriting rule $RR_{destination-port}$.

Definition 8. *Support of a generalized item.* Let D be a structured dataset, $t_i = \exp ression_i$ a generalized item, and RR_i the corresponding rewriting rule defined over the t_i domain (Ω_i). The support of $t_i = \exp ression_i$ is the (observed) frequency of its aggregation value defined by RR_i in all the records in D.

The support of a generalized item $t_i = \exp ression_i$ may also be counted by adding the frequency of all leave values defined by RR_i in D.

A generalized itemset may include both items and generalized items, as stated by the following definition.

Definition 9. *Generalized itemset.* Let $I = \{t_1 = value_1, t_2 = value_2, \ldots, t_n = value_n\}$ be a set of items and $E = \{t_1 = \exp ression_1, t_2 = \exp ression_3, \ldots, t_n = \exp ression_n\}$ be a set of generalized items. A generalized itemset Y is a subset of $I \cup E$.

Definition 10. *Support of a generalized itemset.* Let D be a structured dataset and Y a generalized itemset. The support of Y is the frequency of Y in all the records in D.

Given a structured dataset D, a taxonomy Γ, and a minimum support threshold s, frequent generalized itemset mining is the process to extract all generalized itemset whose support is above support threshold s.

THE GenIO ALGORITHM

The GenIO (Generalized Itemset Discoverer) algorithm extracts generalized itemsets. It exploits a taxonomy Γ (i.e., a set of rewriting rules) to generalize concepts defined in the structured dataset under analysis. GenIO takes in input a set of RR_i, the structured dataset D, and a minimum support threshold.

The pseudo-code of GenIO is given by Algorithm 1. GenIO implementation is based on *Apriori* (Agrawal and Srikant, 1994). *Apriori* is a level-wise algorithm, which, at each iteration, generates all frequent itemsets of a given length. At arbitrary iteration *k*, two steps are performed: (i) Candidate generation, the most computationally and memory intensive step, in which all possible *k*-itemsets are generated from *(k-1)*-itemsets, (ii) candidate pruning, which is based on the property that all subsets of frequent itemsets must also be frequent, to discard candidate itemsets which cannot be frequent. Finally, actual candidate support is counted by reading the database.

Our approach follows the same level-wise pattern. However, the GenIO algorithm (i) exploits context knowledge to further prune candidates (line 7), and (ii) manages rare itemsets by means of RR_c (lines 10-16). Further candidate pruning is based on uniqueness of attributes in a given record of a structured dataset (e.g., in the traffic network domain a flow, i.e., a record of the network trace, with multiple *destination-address* cannot be defined). For example, suppose that after the first dataset traversal, we have *m* frequent 1-itemsets tagged *source-address* and *n* tagged *destination-address*. Apriori exhaustive candidate generation would produce $\binom{m+n}{2}$ possible combinations. Since each attribute is allowed only once in each record, only $m \cdot n$ 2-itemsets (obtained by combining one item tagged *source-address* and one item tagged *destination-address*) are relevant combinations. Hence, we only generate these candidates, thus reducing the required computational and memory cost. Once the support of each itemset has been computed (lines 5 and 8), a generalized version of each one is generated (line 11). Redundant generalized itemsets (i.e., same generalized itemsets generated by two different unfrequent values of the same attribute) are discarded by the update procedure in line 12, which also computes the current support value of the generalized itemset. Finally, only generalized itemsets above the support threshold and derived from rare itemset rewritings are kept (line 17). When a generalized itemset is generated only by frequent itemsets, it is discarded because its knowledge is already provided by the corresponding (frequent) non generalized itemsets.

Algorithm 1. GenIO Generalized Itemset Discoverer

```
Input: minimum support min_sup, $\Gamma$, dataset $D$
Output: set of generalized frequent itemsets L
1: $k=1$, $L=\emptyset$
2: repeat
3: $Gen=\emptyset$ // generalized itemset container
4:  if $k=1$ then
5:     $C_1$ =scan $D$ and count support for each item
6:  else
7:     $C_k$ = candidate_generation( $L_{k-1}$ )
8:      scan $D$ and count support for each $c \in C_k$
9:   end if
10:  for all $c$ in $C_k$ do
11:     new_itemset = apply $RR_c$ on $c$
12:     update $Gen$ with new_itemset
13:     if support of $c$ < min_sup then
14:        mark new_itemset
15:     end if
16:  end for
17: $L_k$ = {items in $C_k$ whose support ≥ min_sup} $\cup$ {marked itemsets of $Gen$
whose support ≥ min_sup }
18: $k=k+1$
19: until $L_k \neq \emptyset$
20: return $L$
```

EXPERIMENTAL RESULTS

We evaluated the performance of the GenIO algorithm by means of a large set of experiments which analyze (i) the performance of frequent generalized itemset extraction in terms of execution time, and (ii) the number and relevance of the extracted generalized itemsets.

Experimental setting

Experiments have been performed on an AMD Sempron(tm) 2400+ PC with 1666 MHz CPU and 512 Mb main memory, Linux operating system and WEKA version 3.5.2. Eight real datasets have been exploited to validate the efficiency and effectiveness of the GenIO algorithm. These datasets have been obtained by performing different capture sessions using the Analyzer tool (NetGroup, Politecnico di Torino) on a backbone link of the campus network of the Politecnico di Torino. Captured traffic has been aggregated in traffic flows, i.e. records which summarize a group of similar and temporally contiguous packets. Each flow is a transaction characterized by six attributes: Source IP address, destination IP address, source port, destination port, flow size (i.e., the size of the flow expressed in byte), and number of IP packets aggregated in that flow. We will refer to each dataset using the ID shown in the y-axis of **Figure 2**. **Figure 2** reports the number of records and the number of different items for each datasets. All datasets are characterized by a very large number of different items. Dataset #1 is the least sparse, while the sparsest is dataset #8. Dataset ids are ordered by increasing sparseness. All datasets are characterized by a skewed data distribution (see **Figure 2**).

The generalization process, performed during the generalized itemset mining, is affected by the data distribution. Sparser data distributions may require a higher number of generalizations, and thus more computational time to perform the mining process.

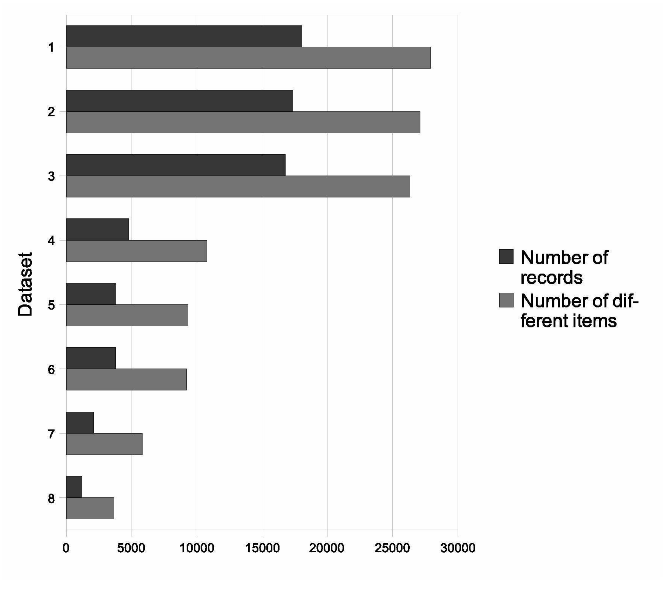


Figure 2. Characteristics of the datasets

The taxonomy used in the performed set of experiments aggregates unfrequent items according to the following rewriting rules. (1) Source and destination ports are aggregated by exploiting the rewriting rule shown in **Figure 1**, which introduces three new aggregation values (i.e., *well known*, *registered*, *dynamic*). (2) Source and destination IP addresses are aggregated by exploiting the rewriting rule shown **Figure 3**. IP addresses are aggregated in *subnet* if they are local to the campus network. IP addresses which do not belong to the campus network are aggregated in a general *external network* node. Furthermore, both the flow size and number of IP packets attributes are discretized in 4 bins.

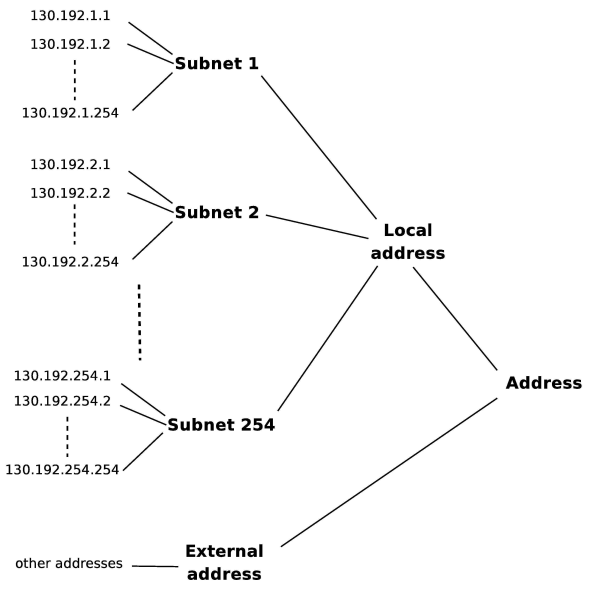


Figure 3. A rewriting rule $RR_{IP-address}$ exploited for both source and destination IP address attributes

Frequent generalized itemset extraction performance

To evaluate the performance of the GenIO algorithm, different generalized itemset extraction processes have been run. **Figure 4** shows the execution time with various support thresholds. The execution time includes both the generalization process and the mining step. Sparser data distributions require a higher number of generalization steps. However, the generalization process has a small impact on the performance of the algorithm, which is mostly affected by the number of transactions (i.e., flow records). For example, dataset #1, which is the least sparse dataset, but the biggest by considering the transaction number, requires the highest run time for any support thresholds.

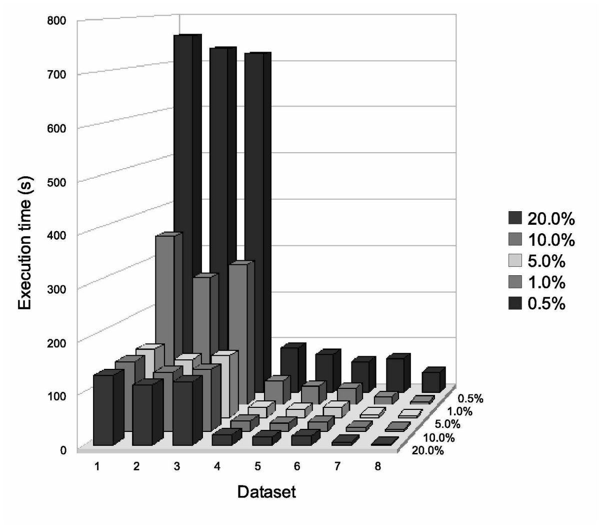


Figure 4. Performance of GenIO algorithm

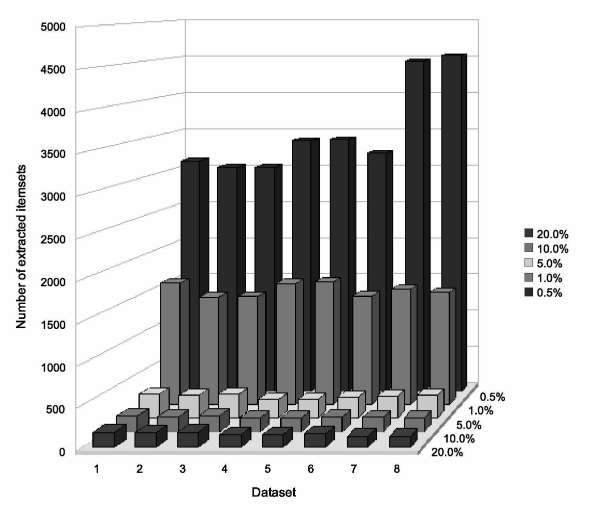
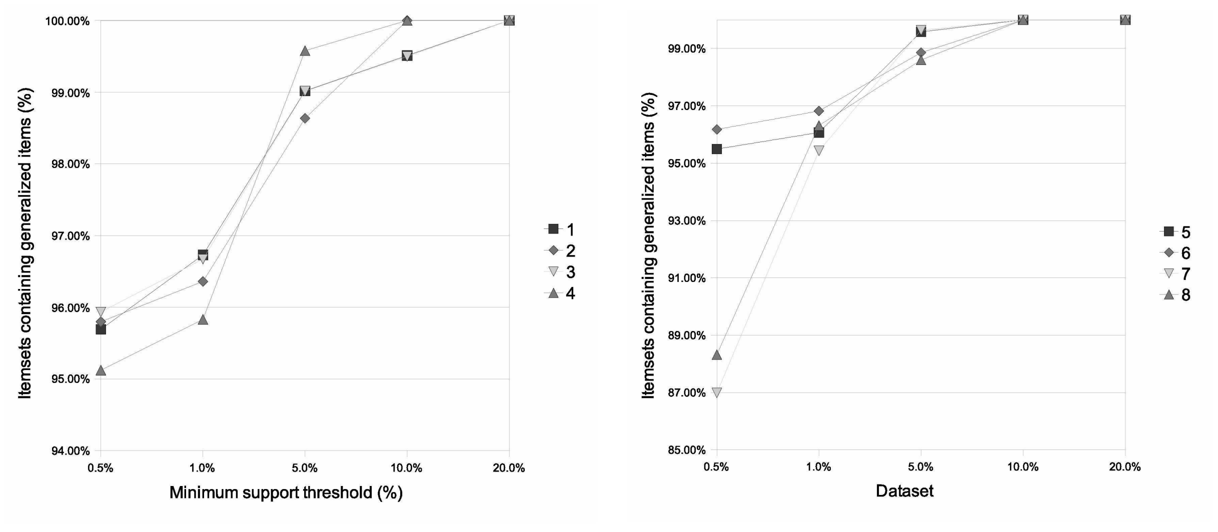


Figure 5. Number of itemsets extracted by GenIO algorithm

Figure 5 shows the number of itemsets extracted by enforcing different support thresholds. When low support thresholds are enforced the number of mined itemsets always increases. For high support thresholds, the number of generalized itemsets increases when the data distribution becomes less sparse. When low support thresholds are enforced, the trend is reversed. In particular, the number of generalized itemsets increases significantly when the data distribution becomes more sparse (see Figure 5). This effect is given by the high number of low frequency items characterizing a sparse dataset. These items become frequent when the mining process is performed by enforcing a low support threshold.

Analysis of the extracted generalized itemsets

In the following we analyze the percentage of extracted itemsets arising from the generalization process. Furthermore, the meaning of the extracted generalized rules is analyzed in the network traffic domain.



(a) Datasets: 1,2,3, and 4 (b) Datasets: 5,6,7, and 8

Figure 6. Percentage of extracted itemsets arising from the generalization process

Figure 6 shows the impact of the aggregation process on the number of mined itemsets for each dataset. In particular, Figure 6 reports the percentage of itemsets arising from the generalization process with respect to the total number of extracted itemsets by varying the minimum support threshold. Since generalization is a support driven process, the number of itemsets containing terms at higher level of the taxonomy increases by enforcing higher support thresholds.

Traditional itemset mining approaches, which do not perform generalization, for the same support threshold, will not extract the percentage of itemsets reported in Figure 6. Furthermore, traditional generalized itemset mining could obtain the same knowledge mined by the GenIO algorithm only through a very expensive post-processing analysis over a very large set of low support itemsets.

Figure 7 shows the number of the generalized 2-itemsets obtained from dataset #1. For better visualization, results have been restricted to addresses of the campus network. Thus, no external IP address has been considered. The 2-itemsets are in the form *(destination-address, destination-port),* where the address is automatically aggregated to the subnet when single IP support is under the minimum support threshold (set to 1%). Figure 7 provides a characterization of the traffic on the campus network. Many extracted itemsets describe general network features. For example, the first top support itemset identifies the VPN concentrator of the campus network. Larger itemsets allow focusing on specific traffic behaviors. For example, the itemsets (*destination-address=130.192.e.e, destination-port=57403, source-address=x.x.x.x, source-port=registered-port*) with support equal to 2.3% highlights an unconventional high-volume traffic toward a specific host of the campus network, whereas the itemsets (*source-address=y.y.y.y, destination-address=130.192.a.a, destination-port=registered-port, source-port=well-known*) with support equal to 2% identifies connections to the VPN concentrator by means of a client using well-known source ports.

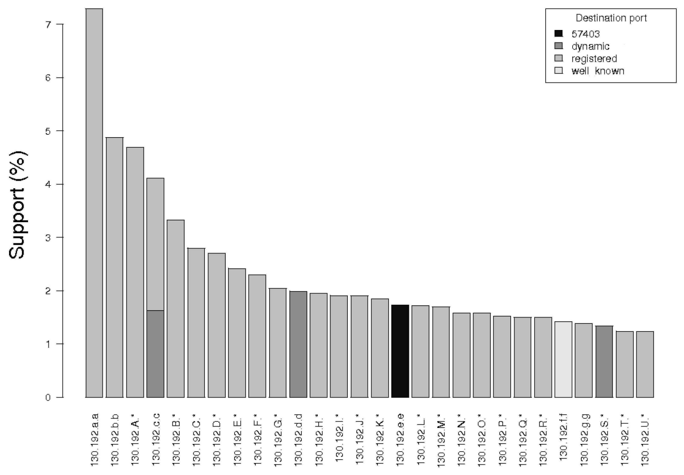


Figure 7. Number of extracted itemsets for different destination IP addresses and ports

RELATED WORK

The problem of generalized association rules has been introduced in (Srikant et al., 1997). Given a large database of transactions, where each transaction is composed by a set of items, and a taxonomy on the items, generalized association rules mining consists in finding associations between items at any level of the taxonomy. Proposed algorithms generate itemsets by considering, for each item, its parents in the hierarchy. By means of this approach candidate frequent itemsets are generated exhaustively, thus producing a large amount of unnecessary combination of items.

A parallel and independent definition of the problem is presented in (Han and Fu, 1995), where a top-down based approach has been presented to extract generalized itemsets. The aim of the proposed strategy is the reduction of the number of unnecessarily generated itemsets. Hence, in the first step of itemset extraction only itemsets containing terms at higher level in the taxonomy, are generated. If a more general term is unfrequent, then its descendants can be ignored during candidate generation.

Both works (Srikant et al., 1997; Han and Fu, 1995) introduce generalized association rules only for categorical data, while (Srikant and Agrawal, 1996) proposes the use of generalization both for categorical and quantitative data, by extending the concept of boolean association rules introduced in (Agrawal et al., 1993). While categorical attributes are aggregated by following a user provided taxonomy, quantitative data are aggregated by means of a data-dependent information loss measure. In (Srikant et al., 1997), the extraction process is further enriched with the possibility of defining boolean rules over the presence or absence of items.

One step further towards a more efficient extraction process for generalized association rules was based on new optimization strategies (Han and Fu, 1995; Han and Fu, 1999; Hipp et al., 1998). A basic optimization involves top-down hierarchy traversal to prematurely identify items which cannot be frequent in the dataset (Han and Fu, 1995; Han and Fu, 1999). Other techniques involve faster support counting (Hipp et al., 1998) by using the TID intersection computation which is common in algorithm designed for vertical data format (Zaki et al., 1997).

A parallel effort was devoted to the integration of association rules mining in relational database systems (Sarawagi et al., 2000) which led to the development of DBMS-centric approach for association rules mining. This approach is usually based on the definition of ad-hoc queries (Thomas and Sarawagi, 1998). In (Thomas and Sarawagi, 1998) different kind of association among data are extracted by complex SQL queries. Association among data can be boolean predicated, sequential pattern, and generalized association rules.

All the proposed approaches which extract rules at higher level of abstraction exploit user provided taxonomies to obtain more abstract representations of items. These techniques always extract all the possible rules at all the levels of abstraction derived from the defined taxonomy. By means of these techniques a huge amount of rules is generated, and the selection of the interesting rules is left as a post-processing step. Hence, for low support thresholds, both the extraction process and the post-processing step may require an excessive CPU time. Furthermore, a very large space is required to store the huge number of rules on which the post-processing step is performed.

GenIO performs an opportunistic extraction of generalized itemsets by exploiting a user provided taxonomy to drive the pruning phase of the extraction process. Instead of extracting itemsets for all levels of the taxonomy and post-pruning them, the proposed generalization technique over the taxonomy is support driven, i.e., it generalizes an itemset only if the items composing it (at a lower level in the taxonomy) are below the support threshold. If the generalized itemset is above the support threshold, the generalization process stops, otherwise it climbs up the taxonomy again. Thus, GenIO automatically performs generalization only when needed to exceed the support threshold. Differently from post-pruning approaches, which require setting an appropriate support threshold, our approach prevents pruning rare itemsets.

CONCLUSION

Association rule extraction is a widely used exploratory technique which is able to discover correlations among data without requiring previous knowledge of the application domain. We proposed the GenIO algorithm to efficiently address generalized itemset extraction. User provided data taxonomies are exploited to merge rare patterns in more general expressions. The behavior of the GenIO algorithm has been tested on real network traffic data. Different sets of *Rewriting Rules* have been considered to assess the capability of our approach to generate aggregated knowledge in heterogeneous operating conditions. The experimental results show the effectiveness of the proposed approach in discovering unexpected and generalized hidden knowledge in traffic data.

We have implemented the GenIO algorithm by modifying the Apriori algorithm. The same approach could be implemented in the FP-Growth (Han et al., 2000) algorithm, which is more efficient than Apriori.

REFERENCES

Agrawal, R., Imielinski, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. *ACM SIGMOD Record*, 22(2):207–216.

Agrawal, R. and Srikant, R. (1994). Fast algorithm for mining association rules. In *VLDB’94*, Santiago, Chile.

Agrawal, R. and Srikant, R. (1997). Mining generalized association rules. *FGCS. Future generations computer systems*, 13(23):161–180.

Antonie, M. L., Zaiane, O. R., and Coman, A. (2001). Application of data mining techniqeus for medical image classification. In *MDM/KDD 2001*.

Baldi, M., Baralis, E., and Risso, F. (2005). Data mining techniques for effective and scalable traffic analysis. *International Symposium on Integrated Network Management*, pages 105–118.

Cong, G., Tung, A. K. H., Xu, X., Pan, F., and Yang, J. (2004). Farmer: finding interesting rule groups in microarray datasets. In *ACM SIGMOD’04*, Paris, France.

Han, J. and Fu, Y. (1995). Discovery of multiple-level association rules from large databases. *Proceedings of the 21th International Conference on Very Large Data Bases*, pages 420–431.

Han, J. and Fu, Y. (1999). Mining Multiple-Level Association Rules in Large Databases. *IEEE Transactions on knowledge and data engireering*, 11(7):798–805.

Han, J., Pei, J., and Yin, Y. (2000). Mining frequent patterns without candidate generation. In *ACM SIGMOD*.

Han, J. and Kamber, M. (2006). *Data Mining: Concepts and Techniques*. Series Editor Morgan Kaufmann Publishers. The Morgan Kaufmann Series in Data Management Systems, Jim Gray.

Hipp, J., Myka, A., Wirth, R., and Guntzer, U. (1998). A new algorithm for faster mining of generalized association rules. *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD98)*, pages 74–82.

Kaya, M. and Alhajj, R. (2004). Mining multi-cross level fuzzy weighted association rules. In *IEEE Conf. on Intell. Systems*, pages 225–230.

Mannila, H., Toivonen, H., and Verkamo, A. I. (1994). Efficient algorithms for discovering association rules. In *KDD Workshop*, pages 181–192.

NetGroup, Politecnico di Torino. Analyzer 3.0. Available at **http://analyzer.polito.it**

Orlando, S., Lucchese, C., Palmerini, P., Perego, R., and Silvestri, F. (2003). kdci: a multi-strategy algorithm for mining frequent sets. In *FIMI*.

Sarawagi, S., Thomas, S., and Agrawal, R. (2000). Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications. *Data Mining and Knowledge Discovery*, 4(2):89–125.

Srikant, R. and Agrawal, R. (1996). Mining quantitative association rules in large relational tables. *Proceedings of the 1996 ACM SIGMOD international conference on management of data*, pages 1–12.

Srikant, R., Vu, Q., and Agrawal, R. (1997). Mining association rules with item constraints. *KDD*, 97:67–73.

Thomas, S. and Sarawagi, S. (1998). Mining Generalized Association rules and Sequential Patterns Using SQL Queries. *Proceedings of KDD-98, 4th International Conference on Knowledge Discovery and Data Mining, AAAI Press.*

Toivonen, H. (1996). Sampling large databases for association rules. In *VLDB*, pages 134–145.

Zaki, M., Parthasarathy, S., Ogihara, M., Li, W., et al. (1997). New algorithms for fast discovery of association rules. *3rd Intl. Conf. on Knowledge Discovery and Data Mining, 20.R.*