Discovering High-Utility Itemsets at Multiple Abstraction Levels

Discovering high-utility itemsets at multiple abstraction levels

Luca Cagliero, Silvia Chiusano, Paolo Garza, Giuseppe Ricupero

Dipartimento di Automatica e Informatica,
Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy.
E-mail: *luca.cagliero@polito.it*

Keywords High-utility itemset mining, generalized itemset mining, data mining, knowledge discovery.

Abstract *High-Utility Itemset Mining (HUIM) is a relevant data mining task. The goal is to discover recurrent combinations of items characterized by high profit from transactional datasets. HUIM has a wide range of applications among which market basket analysis and service profiling. Based on the observation that items can be clustered into domain-specific categories, a parallel research issue is generalized itemset mining. It entails generating correlations among data items at multiple abstraction levels. The extraction of multiple-level patterns affords new insights into the analyzed data from different viewpoints. This paper aims at discovering a novel pattern that combines the expressiveness of generalized and High-Utility itemsets. According to a user-defined taxonomy items are first aggregated into semantically related categories. Then, a new type of pattern, namely the Generalized High-utility Itemset (GHUI), is extracted. It represents a combinations of items at different granularity levels characterized by high profit (utility). While profitable combinations of item categories provide interesting high-level information, GHUIs at lower abstraction levels represent more specific correlations among profitable items. A single-phase algorithm is proposed to efficiently discover utility itemsets at multiple abstraction levels. The experiments, which were performed on both real and synthetic data, demonstrate the effectiveness and usefulness of the proposed approach.*

1 Introduction

Frequent itemset mining is an exploratory data mining technique which focuses on discovering recurrent combinations of items (of arbitrary size) that occur in potentially large transactional data [1]. Frequent itemsets have been used in many research contexts, among which market basket analysis [1], service profiling [3], to discover correlations between multiple data items. For example, in the context of market basket analysis, each row of the dataset (transaction) represents a different market basket. Transactions contain the subsets of purchased items. Frequent itemsets represent sets of items that customers frequently purchased together. For instance, itemset {*Coke, bread*} indicates that customers who purchased coke frequently purchased bread as well. Since generating all the possible combinations of items in a transactional dataset is computationally intractable [1], frequent itemset mining entails discovering only the combinations of items whose frequency of occurrence (support) is above a given threshold. However, the traditional itemset mining problem relies on three (potentially unreliable) assumptions:
(A) Items appear at most once in each transaction (e.g., we disregard the amounts of purchased items within each basket).
(B) Items have all the same importance in the analyzed data (e.g., the unit profit is assumed to be the same for all the items in the market).
(C) The semantic relationships between items are ignored (e.g., the co-occurrences of items belonging to the same product group within the same basket are considered as uncorrelated with each other).
To overcome limitations (A) and (B), the concept of High-Utility Itemset (HUI) has been proposed [10]. HUIs represent sets of frequently co-occurring items that are characterized by averagely high utility within the analyzed data. To mine HUIs, items in the transactional dataset are enriched with both per-transaction weights (hereafter denoted as *internal utilities*) and global weights (denoted as *external utility*). For example, in the context of market basket analysis internal utilities represent per-item amounts (e.g. the customer purchased 3 bottles of coke), while external utilities indicate unit profits (bottles of coke cost 5 USD each). Utility itemsets represent sets of items whose total yield is above a given (user-specified) threshold. This knowledge may be exploited to perform cross-selling, to plan promotions, or to effectively arrange items on the shelves.

To overcome limitation (C), correlations between items at higher abstraction levels can be analyzed [11]. Based on a taxonomy built on top of the analyzed data, items are aggregated into semantically related groups (e.g. items *Coke* and *Water* into group *Beverage*). Then, generalized itemsets , which represent correlations among data items at different abstraction levels (e.g., not only {*Coke, bread*} but also {*Beverage, Food*}), can be extracted.

Many algorithms have been proposed to efficiently extract HUIs [8, 7, 9, 12] and to select compact subsets of HUIs, e.g., the closed HUIs [5] and the top-k HUIs [13]. Existing solutions are efficient in term of temporal and spatial scalability, but, to the best of our knowledge, they are unable to cope with multiple-level data (limitation (C)). On the other hand, parallel works addressed the generalized itemset mining problem by performing bottom-up [2, 3, 11] or top-down [6] taxonomy visits during candidate itemset generation. However, since they do not consider item utilities, they still suffer from limitations (A) and (B).

This paper aims at bridging the gap between HUI mining and generalized itemset mining. To this purpose, it proposes a new type of pattern, namely the Generalized High-utility Itemsets (GHUIs). The proposed approach allows both multiple appearances of the same item within each transaction and different per-item profits. Unlike traditional HUIs, GHUIs are extracted from a transactional dataset enriched with a taxonomy, which describes the semantic is-a relationships between data items. These relationships are exploited to drive the process of knowledge generalization thus generating profitable combinations of items at multiple abstraction levels. To extract GHUIs we proposed ML-HUI Miner, which extends a state-of-the-art HUI mining algorithm to cope with data enriched with taxonomies. The newly proposed algorithm integrates taxonomy information into the utility itemset mining process to mine GHUIs in a single-phase mining session.

Preliminary experiments performed on both real retail data and benchmark datasets show the efficiency and effectiveness of the proposed approach.

The paper is organized as follows. Sections 2 and 3 introduce preliminary concepts and formalizes the newly proposed pattern, respectively. Section 4 presents the mining algorithm used to discover the newly proposed pattern. In Section 5 we summarized the experiments performed on real datasets, while Section 6 draws conclusions and discusses future works.

2 Preliminaries

A transactional dataset is a set of transactions [1], where each transactions is a set of data items (i.e., objects identified by literals). Herafter, let us denote as $\mathcal{I}$ the set of all possible items and as $t_j \in \mathcal{I}$ the j-th transaction of a transactional dataset $\mathcal{D}$. Items are characterized by (i) Internal Utility, denoted as iu(i,t_j) which indicates the relative importance of item $i \in \mathcal{I}$ in transaction t_j, and (ii) External Utility, denoted as eu(i), which indicates

Table 1: Transactional dataset

Transaction id	Items and internal utility
TId1	(Coke, 2), (Bread, 2), (Steak, 1)
TId2	(Water, 3), (Pasta, 2), (Steak, 1)
TId3	(Water, 2), (Bread, 2)
TId4	(Coke, 1), (Bread, 2)

Table 2: External utilities of items in dataset

Item	External utility
Water	1
Coke	5
Bread	1
Pasta	2
Steak	10

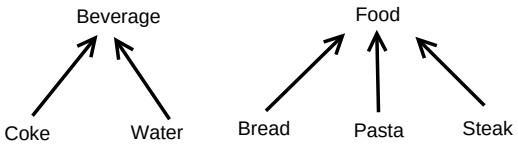


Figure 1: Taxonomy on items in the dataset

the relative importance of item i in $\mathcal{D}$ with respect to all the other items in the dataset.

Table 1 reports an example of market basket dataset consisting of 4 transactions (identified by Tids 1-4)).

Table 3: High-Utility Itemsets. minutil (itemsets) = 17

Itemset	Utility
{ Steak}	20
{ Steak, Coke}	20
{ Coke, Bread}	19
{ Steak, Coke, Bread}	22

Table 4: Generalized High-Utility Itemsets. minutil (generalized itemsets) = 30

Generalized Itemset	Utility
{ Food}	30
{ Food, Beverage}	50

The utility of item i in transaction t_j, hereafter denoted as u(i,t_j), is computed as eu(i)· iu(i,t_j). In the running example, it indicates the total income related to an item appearing in the market basket (e.g., the price of all the bottles of coke in a given market basket).

Itemsets are sets of items of arbitrary size. We will denote as k-itemset a set of k items. The utility of itemset I in transaction t_j is the sum of the utilities of all the corresponding items, i.e., u(I,t_j)=$\sum_{i \in I} u(i, t_j)$. The utility of itemset I in the transactional dataset $\mathcal{D}$ is obtained by summing the utilities of the itemset in all the dataset transactions, i.e., u(I,$\mathcal{D}$)=$\sum_{t_j \in \mathcal{D}} u(I, t_j)$, where we assume that u(I,t_j)=0 if $I \notin t_j$.

A notable type of itemset is the High-Utility Itemset. Given user-specified minimum utility threshold *minutil*, an itemset mined from dataset $\mathcal{D}$ is an High-Utility Itemset (HUI) if and only if u(I,$\mathcal{D}$)>*minutil*. Given a transactional dataset $\mathcal{D}$ and a minimum utility threshold *minutil*, the High-Utility Itemset Mining (HUIM) problem entails discovering all the HUIs in $\mathcal{D}$.

The HUIs extracted from the dataset in Table 1 by enforcing a minimum utility threshold *minutil*=20 are enumerated in Table 3, where the corresponding utility value is given too.

Example. {Coke, Bread} is HUI because the utility values of the 2-itemset for each transaction are: 12 (5x2+1x2) in TId1, 0 in TId2 and TId3 (no matches), and 7 (5x1+1x2) in TId4.

To efficiently extract HUIs, the Transaction-Weighted Utilization (TWU) has been introduced [12]. It is an over-estimate of the utility of the itemset, which can be exploited to prune the search space because it satifies the following downward closure property: given two itemsets I_1 and I_2 such that $I_1 \subset I_2$, if the TWU of I_1 is below the utility threshold *minutil* even the TWU of I_2 does. The TWU of an itemset I, denoted as twu(I), is defined as the sum of the transaction utilities of all the transactions containing I, where the transaction utility of a transaction is the sum of the utility values of all its items. A formal definition of the TWU measure of itemset I follows: twu(I)= $\sum_{t_j \in \mathrm{D} | \mathrm{I} \subseteq \mathrm{t_j}} \sum_{i \in t_j} u(i, t_j)$.

3 Generalized High-Utility Itemsets

Our goal is to discover HUIs that incorporate knowledge at multiple granularity levels. To this aim, we generalize items at different abstraction levels.

Let $\mathcal{T}$ be a taxonomy (i.e., a is-a hierarchy), which aggregates items in $\mathcal{I}$ into higher-level concepts, hereafter denoted as *generalized items*. Generalized items represent higher-level categories which group individual items based on their semantic meaning. Let $\mathcal{G}$ be the set of generalized items in $\mathcal{T}$. For the sake of simplicity, hereafter we will assume that in the given taxonomy $\mathcal{T}$ each item $i \in \mathcal{I}$ is aggregated into exactly one generalized item $g \in \mathcal{G}$ (i.e., each item belongs to a specific higher-level category). Generalized items can be further generalized as other generalized items at higher granularity levels.

For each generalized item $g \in \mathcal{G}$, Desc(g,$\mathcal{T}$) $\in \mathcal{I}$ denote the subset of descendant items of g according to the given taxonomy. For our purposes, we formalize the concept of *level* of a generalized item $g \in \mathcal{G}$ in the taxonomy, hereafter denoted as l(g,$\mathcal{T}$), as the length of the shortest path between g and any leaf node in the taxonomy. Note that, by construction, the level of non-generalized itemsets is zero, while the maximum level of an item corresponds to the taxonomy height (i.e., the length of the longest path from any node in $\mathcal{T}$ to a leaf node).

Example. Figure 1 depicts an example of taxonomy built on items occurring in the running example dataset (see Table 1). For instance, items *Coke* and *Water* are aggregated into the generalized item *Beverage*. The level of *Beverage* and is one, whereas the level of *Coke*, and *Water* is zero.

A generalized itemset is a set of generalized items in $\mathcal{G}$. Similar to [3, 6, 11] we focus our analyses on the combinations of generalized items having the same level, because they compactly represent information at a given abstraction level. Herafter we will denote as *level* of a generalized itemset the level of its items.

For our purposes, we extend the concept of utility to generalized items and itemsets. Specifically, the utility of a generalized item g in a transaction t_j, hereafter denoted as u(g,t_j), is the sum of the utility values of all the descendant items, while the utility of g in a transactional dataset is the sum of all the per-transaction utilities. More formal definitions follow.

Definition (Utility of a generalized item). Let g be a generalized item, $\mathcal{D}$ a transactional dataset, and $\mathcal{T}$ be a taxonomy. The utility of g in a transaction $t_j \in \mathcal{D}$ is defined as u(g,t_j)=$\sum_{i \in Desc(g,\mathcal{T})}$eu($i$)$\cdot$ iu(i,t_j). The utility of generalized item g in $\mathcal{D}$ is calculated as u(g,$\mathcal{D}$)=$\sum_{i \in Desc(g,\mathcal{T})} u(i, \mathcal{D})$. □

Similar definitions hold on itemsets (i.e., sets of items). The utility of a generalized itemset in a transactional dataset indicates the overall profit of a combination of item categories.

Definition (Utility of a generalized itemset). Let GI be a generalized itemset and let $\mathcal{D}$ a transactional dataset, and $\mathcal{T}$ be a taxonomy. The utility of a generalized itemset GI in transaction t_j is defined as u(GI,t_j)=$\sum_{g \in GI} u(g, t_j)$. The utility of a generalized itemset GI is in the transactional dataset $\mathcal{D}$ is defined as u(GI,$\mathcal{D}$)=$\sum_{t_j \in \mathcal{D}} u(GI, t_j)$, where we assume u($GI$,$t_j$)=0 if $GI \notin t_j$. □

Example. Let us consider again the market basket dataset reported in Table 1 and the taxonomy in Figure 1. The per-transaction utility of generalized item *Beverage* is 10 in transaction with TId1, 3 in transaction with TId2, 2 in transaction with TId3, and 5 in transaction with TId4. Hence, the utility of *Beverage* in the dataset is 20 (10+3+2+5).

In this work we address the extraction of a selection of generalized itemsets, called Generalized High-Utility Itemsets (GHUIs).

Definition (Generalized High-Utility Itemset). Let *minutil* be a (user-specified) minimum utility threshold, let $\mathcal{D}$ be a transactional dataset, let $\mathcal{T}$ be a taxonomy, and let GI be a generalized itemset. A generalized itemset GI is a Generalized High-Utility Itemset (GHUI) in $\mathcal{D}$ if and only if u(GI,$\mathcal{D}$)>*minutil*. □

Example. The GHUIs mined from the dataset in Table 1 by enforcing a minimum utility threshold for generalized itemsets equal to 30 for GHUIs are enumerated in Table 4.

Given a transactional dataset $\mathcal{D}$, a taxonomy $\mathcal{T}$, and a minimum utility threshold *minutil*, the Generalized High-Utility Itemset Mining (GHUIM) problem addressed by this work entails discovering all the HUIs and GHUIs.

Per-level utility thresholds. The utility of a generalized itemset incorporates those of all of its descendant itemsets. Hence, itemsets at higher abstraction levels are more likely to satisfy a fixed minimum utility threshold than lower-level ones. To overcome this issue, we adapt the utility threshold to the level of generalization of the considered itemsets. The key idea is to set higher utility thresholds for itemsets including items at higher abstraction levels. We set the same utility threshold for all itemsets having the same level. Specifically, given a user-specified least minimum utility threshold *minutil* associated with non-generalized itemsets (level=0), the minimum utility threshold thr(l) associated with level-l generalized itemsets is *minutil* if $l = 0$, $thr(l) = \alpha(l) \cdot$ *minutil* otherwise, where $\alpha(l)$: $\mathbb{N} \rightarrow [1, +\infty)$ is a monotonically increasing (user-specified) function.

4 The ML-HUI Miner algorithm

To extract Generalized High-Utility Itemsets we propose a new algorithm, namely Multiple-Level High-Utility Itemset Miner (ML-HUI Miner). The main features of GHUI-Miner are summarized below.

(a) Taxonomy-driven HUI mining. ML-HUI Miner supports transactional data enriched with taxonomy information. This allows us to extract patterns at different abstraction levels.

(b) Single-step extraction of generalized and non-generalized HUIs. The proposed algorithms explores the dataset and the taxonomy to generate multiple-level patterns in a single phase (i.e., without the need for multiple runs).

Algorithm 1 The ML-HUI Miner algorithm

Require: transactional dataset $\mathcal{D}$, taxonomy $\mathcal{T}$, minimum utility threshold *minutil*, function $\alpha(l)$
Ensure: $\mathcal{O}$, the set of High-Utility Itemsets and Generalized High-Utility Itemsets satisfying the per-level utility thresholds
{**Initializations**}
1: $\mathcal{I} \leftarrow$ set of items in $\mathcal{D}$
2: $\mathcal{GI} \leftarrow$ set of generalized items in $\mathcal{T}$
{**Preparation**}
3: scan $\mathcal{D}$ and $\mathcal{T}$ to compute Transaction-Weighted-Utility (TWU) of items in $\mathcal{I}$
4: Compute Transaction-Weighted-Utility of items in $\mathcal{GI}$
5: $\mathcal{I}^* \leftarrow$ set of items in $\mathcal{D}$ such that TWU is above *minutil*(level=0)
6: $\mathcal{GI}^* \leftarrow$ set of generalized items g in $\mathcal{T}$ such that TWU is above $\alpha \cdot$*minutil*(l(g,$\mathcal{T}$))
7: Build utility list and Estimated Utility Co-occurrence Structure of (generalized) items in $\mathcal{I}^* \cup \mathcal{GI}^*$ {**Recursive depth-first search**}
8: $\mathcal{O} \leftarrow$Recursive generation of the combinations of (generalized) items in $\mathcal{I}^*$ and $\mathcal{GI}^*$ whose items share the same level and selection of all combinations satisfying the utility threshold $\alpha(l) \cdot$*minutil*

(c) Prevent the generation of uninteresting combinations of items. Similar to [6, 11], we focus on extracting itemsets including only items with the same level. Thus, GHUI-Miner prevents the generation of itemsets consisting of items with different levels in the taxonomy.

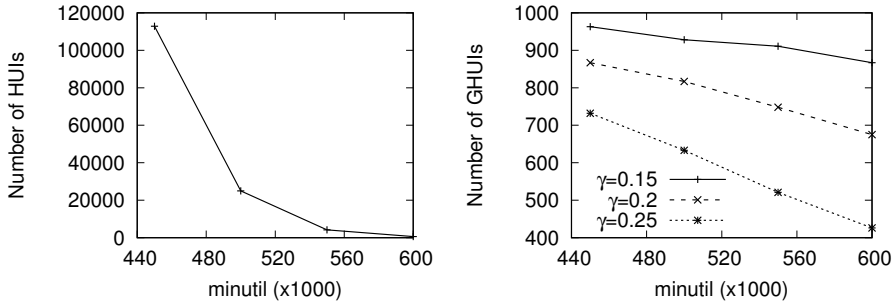
A high-level pseudo-code of the ML-HUI Miner algorithm is given in Algorithm 1. First, ML-HUI Miner scans the dataset and the taxonomy to identify the single non-generalized and generalized items whose Transaction-Weighted-Utility (TWU) is above the per-level utility threshold (Lines 3-6 in Algorithm 1). To this aim, the taxonomy is explored in a bottom-up fashion. The dataset and the accessory structures are properly adapted to prevent the generation of combinations of mixed-level items (according to Point (c). To compute the per-level utility thresholds, the user-specified threshold *minutil* is adjusted using function α according to the level of each generalized item. Then, we compute the utility list associated with all non-generalized and generalized items whose TWU satisfies the per-level utility threshold (Line 7 in Algorithm 1). The utility list is a compact data structure that contains for each (generalized) item i (i) the list of transactions t_j such that $i \in t_j$, (ii) the utility values of the (generalized) item in each transaction u(i,t_j), and (iii) the sum of the utilities of the remaining items with the same taxonomy level within each transaction, i.e., $\sum_{q \in t_j \wedge\ q \neq i \wedge l(q,\mathcal{T})=l(\mathrm{i},\mathrm{T})} U(q, t_j)$. For our purposes, we extended the utility list proposed in [8] to integrate information about the generalized items at the same taxonomy level appearing in the taxonomy. The utility list is provided as input to the depth-first recursive procedure (Line 8 in Algorithm 1), which does not need to access neither the dataset nor the taxonomy. Specifically, starting from single items (generalized and not) the recursive procedure computes their exact utility value and then explores all their extensions using a depth-first strategy based on the utility list. To avoid generating itemsets consisting of items with different level, extensions are selectively generated. The recursive procedure is similar to the one adopted by FHM [8].

5 Experiments

To evaluate the performance of the ML-HUI Miner algorithm, we conducted experiments on four benchmark UCI datasets comng from different domains (*Connect*, *Mushroom*, *Chess*, *Retail* [4]), which have already been used to evaluate the performance of recently proposed High-Utility Itemset mining algorithms (e.g., FHM [8]). The number of transactions per dataset varies from 3,196 (*Chess*) to 67,557 (*Retail*). The analyzed datasets show different characteristics (e.g., number of distinct items *Connect* 129, *Mushroom* 119, *Chess* 75, *Retail* 2,741). The external utilities in the datasets range from 1 to 1,000. They were synthetically generated by using a log-normal distribution. Internal utilities are uniformly distributed between 1 and 5. To set the per-level utility threshold values ($thr(l)$), we defined $\alpha(l)=\gamma \cdot f(l)$, where $f(l)$ is the average number of level-0 descendants per level-l item. To generalize items at higher abstraction levels, we generated taxonomies on top of data items. Specifically, on the *Retail* dataset, which stores real sales of an online store, we generated a 2-level taxonomy by aggregating products into the corresponding group provided by the online store. The 2,741 available products are clustered into 38 produt groups (e.g., *Kitchen*, *Toy*). Products are evenly distributed across product groups. For instance, *Kitchen* clusters 1,513 products, while *Toy* 236. On the other datasets we generated a synthetic taxonomy where each generalized item aggregates, on average, 10 randomly selected products. The experiments were performed on a 2.67 GHz Intel Xeon workstation with 32 GB of RAM, running Ubuntu 12.04.

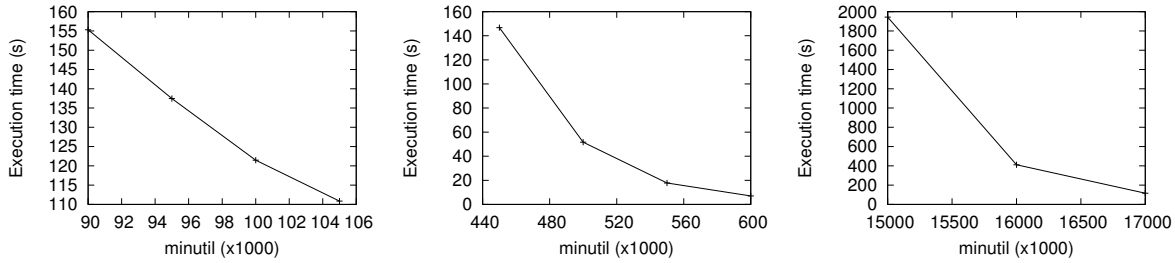
5.1 Performance analysis

Figure 2 shows the number of mined (non-generalized) HUIs and GHUIs by varying the values of *minutil* and γ on a representative dataset (i.e., *Chess*). For both types of patterns, the number of mined itemsets is inversely proportional to the *minutil* value. By decreasing the values of *minutil* and γ the total number of mined patterns super-linearly increases. The number of GHUIs is two orders of magnitude lower than those of HUIs due to the significantly lower number of possible item combinations at higher taxonomy levels. The increase in the number of candidate itemsets due to taxonomy integration implies also a time complexity increase. Figure 3



(a) non-generalized HUIs (b) GHUIs

Figure 2: Chess: number of mined patterns



(a) Mushroom (b) Chess (c) Connect

Figure 3: Execution time (γ=0.2)

shows the time spent by the ML-HUI Miner algorithm on the analyzed datasets with decreasing *minutil* values. Since the value of γ affects only the extraction of HGUIs, whose number is orders of magnitude lower than the number of HUIs, varying γ value slightly affects the execution time. For this reason, we report the execution time only for the representative value γ=0.2 on all datasets. Despite a larger number of combinations were explored, the extraction times remain acceptable (few ms) on all the analyzed datasets. We compared also the execution times of the newly proposed ML-HUI Miner with those of the FHM algorithm [8], which extracts only non-generalized HUIs. ML-HUI Miner execution time approximately doubles that of FHM, even when more than two aggregation levels are integrated in the taxonomy.

5.2 Knowledge discovery

We present some example GHUIs mined from the *Retail* dataset(*minutil*=10000, γ=0.2). The GHUIs with length 1 reveal the most profitable product groups. For example, *Kitchen* is the category with maximal utility. *Toy* ranked second (utility gap from *Kitchen* to *Toy* 87%), *Home* is the third (92%), *Office product* ranked 4th (96%) and *Law & Patio* 5th (96%). For all the remaining categories the utility gap with respect to *Kitchen* was above 97%. The GHUIs with length greater than 1 point out combinations of groups that yielded high profits when the respective products were sold together. Let us consider, for instance, GHUI {Musical-Instruments, Home}. It indicates that the jointly sale of products in these categories provided a high income. Among GHUIs including group *Musical-Instruments*, GHUI$_i$={*Musical-Instruments*, *Kitchen*} is the one with highest utility. The other GHUIs in order of decreasing utility are: {Musical-Instruments, Toy} (utility -75% w.r.t. GHUI$_i$), {Musical Instruments, Home} (utility -91% w.r.t. GHUI$_i$), and {Musical-Instruments, Office-Product} (utility -94% w.r.t. GHUI$_i$). These patterns can can be exploited to figure out which categories of products should be promoted in the same advertising campaign, e.g., while planning a campaign on products belonging to *Musical Instrument*, products of *Kitchen*, *Toy*, *Home* or *Office-Product category* should be advertised as well. The utility value provided us a ranking of the most appealing groups of products to advertise together with products of *Musical Instrument*. GHUIs provide high-level knowledge related to single products that do not satisfy the utility threshold. Let us consider again group *Musical Instrument*. Only two non-generalized HUIs were extracted, i.e., {Red-Harmonica} (utility 26,205) and {Blue-Harmonica} (utility 10,271). However, the utility of the GHUI {Musical-Instruments} is 40,741. Hence, the utility value of {Musical-Instruments} is not only due to the {Red-Harmonica} and {Blue-Harmonica} products, but even to other products of the same group that do not satisfy the utility threshold. Considering GHUI {Musical-Instruments} allows us to consider also the contribution of the other products, even if their single profits are averagely low. Moreover, GHUIs

represent correlations among products that can not be easily infered while considering only HUIs. For instance, even if correlations between products of group {Musical-Instruments} and products of other groups have not been extracted, the high-lever correlations between *Musical Instrument* and other groups were extracted and can be analyzed.

6 Conclusions and future work

We proposed a new pattern, called GHUI, which represents sets of items groups, each one characterized by a high total profit. The significance of the proposed pattern and the performance of the proposed GHUI mining algorithm have been evaluated on retail data, with the goal of planning advertising campaigns of retail products. Future extensions of the work will address the efficient extraction of significant subsets of GHUIs (e.g., closed or minimal GHUIs).

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] E. Baralis, L. Cagliero, T. Cerquitelli, V. D'Elia, and P. Garza. Expressive generalized itemsets. *Inf. Sci.*, 278:327–343, 2014.

[3] L. Cagliero. Discovering temporal change patterns in the presence of taxonomies. *IEEE Trans. Knowl. Data Eng.*, 25(3):541–555, 2013.

[4] P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C.-W. Wu, and V. S. Tseng. Spmf: A java open-source pattern mining library. *J. Mach. Learn. Res.*, 15(1):3389–3393, Jan. 2014.

[5] P. Fournier-Viger, S. Zida, J. C. Lin, C. Wu, and V. S. Tseng. Efficient closed high-utility itemset mining. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, April 4-8, 2016*, pages 898–900, 2016.

[6] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. *VLDB conference*, pages 420–431, 1995.

[7] S. Krishnamoorthy. Pruning strategies for mining high utility itemsets. *Expert Systems with Applications*, 42(5):2371 – 2381, 2015.

[8] J. C. Lin, P. Fournier-Viger, and W. Gan. FHN: an efficient algorithm for mining high-utility itemsets with negative unit profits. *Knowl.-Based Syst.*, 111:283–298, 2016.

[9] J. Liu, K. Wang, and B. C. M. Fung. Direct discovery of high utility itemsets without candidate generation. In *12th IEEE ICDM conference*, pages 984–989, Dec 2012.

[10] Y. Liu, W.-k. Liao, and A. Choudhary. A two-phase algorithm for fast discovery of high utility itemsets. In *Proceedings of the PAKDD'05*, pages 689–695, Berlin, Heidelberg, 2005. Springer-Verlag.

[11] R. Srikant and R. Agrawal. Mining generalized association rules. In *VLDB 1995*, pages 407–419, 1995.

[12] V. S. Tseng, B.-E. Shie, C.-W. Wu, and P. S. Yu. Efficient algorithms for mining high utility itemsets from transactional databases. *IEEE Trans. on Knowl. and Data Eng.*, 25(8):1772–1786, Aug. 2013.

[13] V. S. Tseng, C. W. Wu, P. Fournier-Viger, and P. S. Yu. Efficient algorithms for mining top-k high utility itemsets. *IEEE TKDE*, 28(1):54–67, 2016.