Discovering *Diverse-Frequent* Patterns in Transactional Databases

Somya Srivastava R. Uday Kiran P. Krishna Reddy

Centre of Data Engineering
International Institute of Information and Technology
Hyderabad-500032, India
Email: {somya.srivastava, uday_rage}@research.iiit.ac.in and pkreddy@iiit.ac.in

Abstract

In the area of data mining, the process of frequent pattern extraction finds interesting information about the association among the items in a transactional database. The notion of *support* is employed to extract the frequent patterns. Normally, a frequent pattern may contain items which belong to different categories of a particular domain. The existing approaches do not consider the notion of diversity while extracting the frequent patterns. For certain types of applications, it may be useful to distinguish between the frequent patterns with items belonging to different categories and the frequent patterns with items belonging to the same category. In this paper we propose a new interestingness measure, called *DiverseRank*, to rank the frequent patterns based on the items' categories. Given a set of frequent patterns, we propose an efficient algorithm to extract the *diverse-frequent* patterns. Experiments on the real-world data set show that the *diverse-frequent* patterns extracted with the proposed *DiverseRank* measure are different from the frequent patterns extracted with the *support* measure.

1 Introduction

Frequent pattern mining extracts interesting associations among the items in a transactional database. In this paper, we propose a new interestingness measure called *DiverseRank* to rank the frequent patterns by considering the categories of items within it. We first explain the basic model of frequent patterns and discuss the motivation for the proposed approach.

The basic model of frequent patterns is as follows [1]. Let $I = \{i_1, i_2, \cdots, i_n\}$ be a set of items, and DB be a database that consists of a set of transactions. Each transaction T contains a set of items such that $T \subseteq I$. Each transaction is associated with an identifier, called TID. Let $X \subseteq I$ be a set of items, referred as an itemset or a *pattern*. A pattern that contains k items is a k-pattern. A transaction T is said to contain X if and only if $X \subseteq T$. The frequency (or support count) of a pattern X in DB, denoted as $f(X)$, is the number of transactions in DB containing X. The support of X, denoted as $S(X)$, is the ratio of its frequency to the DB size, i.e., $S(X) = \frac{f(X)}{|DB|}$. The pattern X is **frequent** if its support is no less than the user-defined minimum support threshold, i.e., $S(X) \geq minSup$. We explain about the model through Example 1.

> **Example 1:** Consider the database of 6 transactions as shown in Table 1. The set of items $I = \{bread, butter, eggs, milk, hair\ spray, diapers, tea, whiskey, battery, orange, apple, cherry\}$. The set of items '*bread*' and '*eggs*', i.e., $\{bread, eggs\}$ is a pattern. It is a 2-pattern. It occurs in 4 transactions ($TIDs$ of 1, 2, 3 and 4). Therefore, the support count of $\{bread, eggs\}$, i.e., $f(\{bread, eggs\}) = 4$. The support of $\{bread, eggs\}$ is $0.66\left(=\frac{4}{6}\right)$. If the user-specified $minSup = 0.25$, then $\{bread, eggs\}$ is a frequent pattern because $S(\{bread, eggs\}) \geq minSup$. The list of frequent patterns discovered from Table 1 are shown in Table 2.

Table 1: Transaction database.

TID	Transactions
1	bread, butter, eggs, orange
2	bread, butter, eggs, apple
3	bread, eggs, battery, milk, tea
4	bread, eggs, battery, cherry
5	butter, diapers, hair spray, whiskey
6	butter, diapers, hair spray

Frequent pattern mining is a popular data mining technique and plays an important role in the extraction of association rules [1]. The notion of *support* is employed to extract the frequent patterns. The major issue with the frequent pattern mining is the generation of a huge number of patterns which might be found insignificant depending

International Conference on Management of Data
COMAD 2011, Bangalore, India, December 19–21, 2011

Table 2: Frequent patterns.

Frequent Patterns	Support
{butter}	0.66
{bread}	0.66
{eggs}	0.66
{battery}	0.33
{diapers}	0.33
{hair spray}	0.33
{bread, eggs}	0.66
{bread, butter}	0.33
{butter, eggs}	0.33
{battery, eggs}	0.33
{battery, bread}	0.33
{butter, hair spray}	0.33
{butter, diapers }	0.33
{diapers, hair spray}	0.33
{bread, butter, eggs}	0.33
{bread, eggs, battery}	0.33
{butter, diapers, hair spray}	0.33

upon the application or user-requirement. In this connection, the researchers have made efforts to mine constraint-based and/or user-interest based frequent patterns by using the measures such as closed [27], maximal [11], periodic [13, 22] and top-k [19].

Similarly, in this paper we have made an effort to propose a new interestingness measure by exploiting the fact that items in a frequent pattern may belong to different categories of a particular domain. For certain types of applications, it may be useful to distinguish between the frequent patterns with items belonging to different categories and the frequent patterns with items belonging to the same category. The existing frequent pattern extraction approaches fail to distinguish the patterns based on the diversity of the items within it. In this paper, we have made an effort to propose an interestingness measure to rank the frequent patterns by analyzing the extent to which the items in the patterns belong to different categories.

In this paper, we propose a new interestingness measure called *DiverseRank*, to rank the frequent patterns based on the items' categories. We use the concept hierarchy of the items concerning to the transactional database to find the items' categories. Given a set of frequent patterns, we propose an efficient algorithm to extract the *diverse-frequent* patterns. A *diverse-frequent* pattern is a frequent pattern containing items from different categories. We proposed an item-encoding method to mine *diverse-frequent* patterns in an efficient manner. Experiments on the real-world data set show that the *diverse-frequent* patterns extracted with the proposed *DiverseRank* measure are different from the frequent patterns extracted with the *support* measure.

The proposed approach can help as an additional tool for exploring or analyzing the frequent patterns with respect to the diversity and will be helpful to improve the performance of several applications. For example, we already know that the frequent patterns are helpful in getting interesting associations among the items in the transactional databases of supermarkets. If we identify the *diverse-frequent* patterns with the proposed approach, the decision-maker may get an additional insight regarding the items' categories of frequent patterns. Suppose, the customers are buying items from the different categories together, the proposed approach will highlight such patterns so that an appropriate decision can be taken to improve the sales.

The rest of the paper is organized as follows. In the next section we discuss the related work. In Section 3, we explain the notion of *diverse-frequent* patterns and the corresponding algorithm. In Section 4, we present experimental results. The last section contains summary and conclusions.

2 Related Work

In [1], the notion of frequent patterns and the algorithm (Apriori) to extract the frequent patterns from the transactional database was first introduced. Since then, the frequent pattern mining has been widely studied in the field of data mining. Several approaches have been proposed to efficiently extract the frequent patterns from the transactional database [2, 8]. Normally, the preceding frequent pattern mining techniques generate a large number of frequent patterns which can be uninteresting to the user. To generate interesting frequent patterns, research efforts have been made to mine constraint-based and/or user-interest based frequent patterns by using the measures such as closed [27], maximal [11], periodic [13, 22], top-k [19], pattern-length [25] and cost (utility) [10].

Pei *et al.* [18] has introduced convertible monotonic and convertible anti-monotonic properties to reduce the search space. Provided that there is a fixed order on the items, a constraint C is defined as *convertible anti-monotone*, whenever an itemset S satisfies C, so does any prefix of S. Similarly, the constraint C is defined as *convertible monotone*, whenever an itemset S violates C, so does any prefix of S. Some of the constraint-based approaches do not satisfy monotonic and/or anti-monotonic properties which increases the search space for mining constraint-based patterns.

In [1], a notion of confidence was proposed to extract a new kind of knowledge, association rules. After extracting the frequent patterns based on the support, an approach was proposed to extract the association rules using both support and confidence. In the literature, several interestingness measures [5, 16] have been proposed to filter out uninteresting frequent patterns and association rules. The interestingness measures can be objective or subjective. The objective measures are statistical by nature and do not consider any domain knowledge. Examples include *relative support* [26], *all-confidence*, *any-confidence*, *bond* [17], $lift$ and χ^2 [3]. The subjective measure assess the interestingness of a pattern using the user's existing concepts and domain knowledge. Examples include *general impressions* [20], *fuzzy rules* [14] and *hard-soft beliefs* [15].

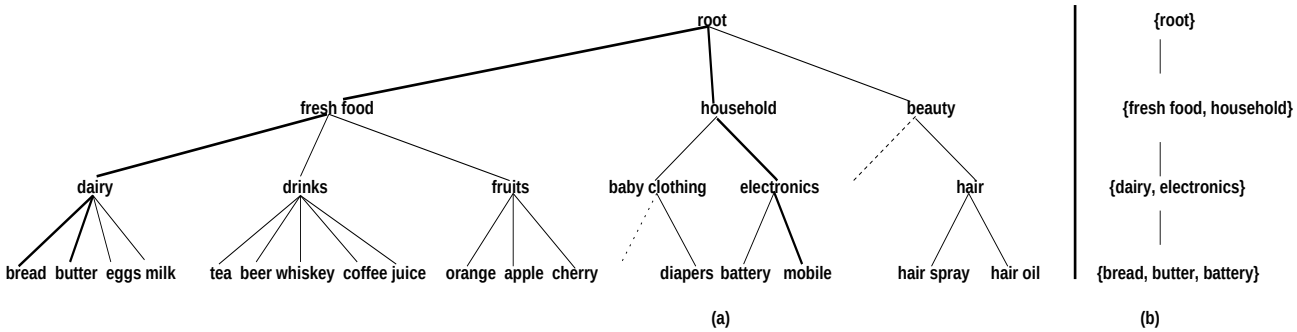

Figure 1: Categories of items: (a) Concept hierarchy (b) Generalized-frequent pattern for $\{bread, butter, battery\}$

The concept hierarchies have been used to discover the generalized association rules in [21, 24] and to discover multiple-level association rules in [7] . In [4], a keyword suggestion approach based on the concept hierarchy has been proposed to facilitate the user's web search. However, the concept hierarchies/taxonomies have not been exploited to assess the interestingness of a pattern with respect to the categories of items within it.

The notion of *diversity* has been widely exploited in the literature to assess the interestingness of summaries [5, 9, 28]. In [12], an effort has been made to extend the *diversity-based* measures to assess the interestingness of the datasets using the diverse association rules. The diversity is defined as the variation in the items' frequencies and not according to the categories of items within it. As a result, the *diversity-based measures* cannot be directly applied to mine the *diverse-frequent* patterns. Moreover, the work in [12] has focused on comparing the datasets using diverse association rules.

As discussed in the preceding paragraphs, several interestingness measures have been proposed along with support to extract interesting frequent patterns. As per our knowledge, no effort has been made to assess the interestingness of a frequent pattern based on the categories of items within it. In this paper, we are proposing the notion of diversity along with support to extract interesting frequent patterns. The proposed *DiverseRank* uses concept hierarchies/taxonomies to assess the interestingness of a pattern with respect to the categories of items within it.

3 Diverse-Frequent Patterns

In this section, we first explain the concept of *diverse-frequent* patterns. Next, we propose the interestingness measure, *DiverseRank*, to rank the frequent patterns based on the items' categories within it. Subsequently, we present the item-encoding technique and the *diverse-frequent* pattern extraction algorithm.

3.1 Concept of Diverse-Frequent Patterns

Given a frequent pattern obtained from a transactional database at some *minSup*, its diversity is based on the category of items within it. If the items of a frequent pattern belong to the same/few categories, we consider that the pattern has low diversity. Relatively, if the items belong to different categories, we consider that the pattern has more diversity.

Finding the category of an item is an important issue. An item may belong to several levels of categories. For example, the item *bread* belongs to *dairy* and $fresh\ food$ categories. We use the concept hierarchy of the corresponding transactional database to find the category of a given item. The concept hierarchy is a tree in which the items are organized in a hierarchical manner. We assume that all the items of a frequent pattern belong to the leaf level of a concept hierarchy. In a concept hierarchy, starting from the leaf level, several lower level items are mapped to the higher level items. Ultimately, all the items are merged to the higher level item, *root*.

Figure 1(a) is an example of concept hierarchy for the items in Transactional Table 1. In this figure, it can be observed that the items *bread*, *butter*, *eggs* and *milk*, are mapped to a higher level item *dairy*. Similarly, the higher level items *dairy*, *drinks*, and $fruits$ are mapped to another higher level item $fresh\ food$. Finally, the higher level items $fresh\ food$, *household* and *beauty* are mapped to the *root*.

Table 3: Frequent patterns of size 3.

Frequent Patterns	**Rank**
{bread, butter, eggs}	3
{bread, eggs, battery}	2
{butter, diapers, hair spray}	1

We explain the notion of *diverse-frequent* patterns by using the concept hierarchy given in Figure 1(a). Consider the frequent patterns of size 3 discovered from Table 1 at $minSup = 0.25$. The user ranking of the frequent 3-patterns based on the categories of items within them are shown in Table 3. The reasoning behind the ranking is as follows. We assume that the frequent patterns with the items from multiple categories are more interesting than the frequent patterns with the items from a few categories. In the pattern $\{bread, butter, eggs\}$, all the items belong to the "dairy" category. Hence, this

pattern is of the least interest to the user. In the pattern $\{bread, eggs, battery\}$, although the items *bread* and *eggs* have a common immediate parent "dairy", the item *battery* has a different parent. All these items have a common parent "root", which is at level 0. Thus, the pattern $\{bread, eggs, battery\}$ is relatively more interesting than the pattern $\{bread, butter, eggs\}$. Since, the items in the pattern $\{butter, diapers, hair\ spray\}$ have only one common parent "root", it is more interesting than the patterns $\{bread, butter, eggs\}$ and $\{bread, eggs, battery\}$.

Now, given two frequent patterns of the same length, different merging behavior can be realized if we observe how a given frequent pattern at the leaf level is moving towards the higher level patterns. That is, one pattern may quickly merge to a few higher level items within a few levels and the other pattern may merge to a few higher level items by crossing more number of levels. To differentiate the patterns in terms of diversity, we define the notion of *DiverseRank*.

3.2 DiverseRank

The following assumptions have been made on the structure of the concept hierarchy.

- The concept hierarchy is represented as a tree of height h (the length of the longest path from root to a leaf node). The height of the root is 0 and the height of leaf is h.
- All the items of a transaction are at the leaf node.
- All the leaf nodes in the tree are present at the same height h. That is, we are considering only the balanced concept hierarchies. The issue of extracting the *diverse-frequent* patterns for the imbalanced hierarchies will be investigated as a part of future work.

It can be observed that a given frequent pattern moves from the leaf level to the root level through a merging process. Several lower level items of a pattern are merged into the corresponding higher level items. In a way, we get higher level pattern for any low level frequent pattern. We capture this aspect through the notion of generalized-frequent pattern (GFP) which is defined as follows.

Definition 1. ***Generalized-frequent pattern (GFP(Y,l)):*** *Let Y be a frequent pattern, l be a level and h be the height of the concept hierarchy (where $0 \leq l \leq h$). The $GFP(Y,l)$ indicates the GFP of Y at level l. (Note that $GFP(Y,h)$ is a frequent pattern obtained from a transactional database using some minSup.) Assume that the $GFP(Y,\ l+1)$ is given. The $GFP(Y,l)$ is calculated based on the GFP of Y at the level $(l+1)$. The $GFP(Y,l)$ is obtained by replacing each item in $GFP(Y,\ l+1)$ with its corresponding parent at the level l with duplicates removed, if any.*

Given the frequent pattern Y, there may be several GFPs (different levels). We explain this through Example 2.

Example 2: Consider the concept hierarchy in Figure 1(a). The height of the tree is 3, i.e., $h = 3$. Any frequent pattern extracted from the Table 1 will always be at level 3. The GFP for $Y = \{bread, eggs, battery\}$ at each level is shown in Figure 1(b). The GFP for the pattern $\{bread, eggs, battery\}$ at level 2 is $\{dairy, electronics\}$, i.e, $GFP(Y,2) = \{dairy, electronics\}$. It is because the items "bread" and "eggs" share a common parent "dairy" at level 2. Similarly, the $GFP(Y,1) = \{fresh\ food, household\}$ and $GFP(Y,0) = \{root\}$.

We now explain the different aspects that are to be considered while computing the diversity of a given frequent pattern. If we observe the GFPs of a given frequent pattern at different levels of the concept hierarchy, the merging process occurs at each level. So, two aspects should be included to compute the diversity of a frequent pattern. One, how the items of a frequent pattern are merging from level to level, which we call *Merging Factor*(MF). And second, the contribution of each level in the diversity as the items are merged at multiple levels, which we call *Level Factor*(LF). We explore these two aspects and develop the formula for computing the diversity of the given frequent pattern.

3.2.1 Merging Factor(MF)

Given two levels l and $(l+1)$ of the concept hierarchy, if all the items of frequent pattern at $(l+1)$ are merging into a single higher level item at l, its MF should be 0. Similarly, when the items of the frequent pattern at $(l+1)$ are not merging at l, the MF should be 1. Consider the concept

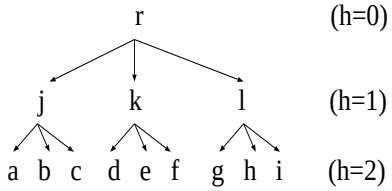

Figure 2: Concept hierarchy

hierarchy of height 2 in Figure 2. The MF of $\{a, b, c\}$ from level 2 to 1 should be 0 as all the items have the same immediate parent. On the other hand, the MF of $\{a, d, g\}$ should be 1 from level 2 to 1. The reason being that there is no merging of items while moving from level 2 to level 1. However, the MF of $\{a, d, g\}$ should be 0 from level 1 to 0, as all the higher level items at level 1 merges into the root.

The MF at a level l depends on the number of items getting merged when the frequent pattern is moved from the immediate lower level $(l+1)$ to the level l in the concept hierarchy. So, the MF at a given level l is proportional to the ratio of the number of items in the GFP at level l to the number of items in the GFP at level $(l+1)$. The MF of a pattern Y at a level l is denoted as $MF(Y,l)$ and given by

the following equation.

$$MF(Y,l) = \frac{|GFP(Y,l)| - 1}{|GFP(Y,\ l+1)| - 1} \quad (1)$$

where $0 \leq l \leq (h-1)$ and $|GFP(Y,\ l+1)|$ represents the number of items in the GFP of Y at level $(l+1)$ and $|GFP(Y,l)|$ represents the number of items in the GFP of Y at level l. It can be noted that $0 \leq MF(Y,l) \leq 1$. We explain the calculation of MF using Equation 1 in Example 3.

Example 3: For the pattern $Y = \{a,\ b,\ c\}$ in Figure 2, the $GFP(Y,2) = \{a,\ b,\ c\}$ and $GFP(Y,1) = \{j\}$. The MF at level 1 is calculated as $\frac{1-1}{3-1}$, i.e., $MF(Y,1) = 0$. Similarly, the GFP for the pattern $Z = \{a,\ d,\ g\}$ at level 2, 1 and 0 are calculated. Thus, the $GFP(Z,2) = \{a,\ d,\ g\}$, $GFP(Z,1) = \{j,\ k,\ l\}$ and $GFP(Z,0) = \{r\}$. Using these values, the MF at level 1 and 0 is calculated, i.e., $MF(Z,1) = 1$ and $MF(Z,0) = 0$.

3.2.2 Level Factor

It can be noted that the frequent pattern which merges to a few higher level items after crossing more levels should receive more diversity value than the frequent pattern which merges to a few higher level items in lesser number of levels. To ensure this, appropriate weight should be added to the MF of the pattern at each level, which is called Level Factor (LF). The LF at a given level can be computed using various methods. We are proposing two methods.

- **Equal LF (ELF)**: The contribution of each level is the same to the diversity. For a given level l, it can be computed as $ELF(l) = 1/h$.
- **Proportional LF (PLF)**: Normally, the contribution of the levels nearby root should be more as compared to the levels near to leaf. For this, we choose a weighting function which is linearly increasing as the pattern moves up in the hierarchy. The PLF at a level l is denoted as $PLF(l)$ and is given by the following formula. Note that, the PLF at level h (leaf level) is 0 and the PLF at root ($h = 0$) is not counted.

$$PLF(l) = \frac{2 * (h-l)}{(h-1) * h} \quad (2)$$

where, $1 \leq l \leq (h-1)$ and $h \neq \{0,1\}$. Also,

$$\sum_{l=h-1}^{1} PLF(l) = 1 \quad (3)$$

We explain the contribution of PLF using Example 4.

Example 4: Using the concept hierarchy of height 3 in Figure 1(a), the weight at level 2 is $PLF(2) = \frac{2*(3-2)}{3*(2)} = 1/3$. Similarly, the PLF at level 1 is computed as $PLF(1) = \frac{2*(3-1)}{3*(2)}) = 2/3$.

3.2.3 Definition of DiverseRank(DR)

The *DiverseRank* of a frequent pattern is a function of MF and LF. The definition is as follows.

Definition 2. ***DiverseRank of a frequent pattern Y****: Let Y be a frequent pattern, h be the height of the concept hierarchy and "s" be the level where the number of items in the GFP of Y is 1 (i.e., $|GFP(Y,s)| = 1$). The DiverseRank of Y, denoted by $DR(Y)$, is given by the following equation. Here, we are using PLF for the LF.*

$$DR(Y) = \sum_{l=h-1}^{s+1} PLF(l) * MF(Y,l)$$

By replacing the corresponding equations, we get the formula for *DiverseRank.*

$$DR(Y) = \sum_{l=h-1}^{s+1} \left[\frac{2*(h-l)}{(h-1)*h}\right] * \left[\frac{|GFP(Y,l)| - 1}{|GFP(Y,l+1)| - 1}\right] \quad (4)$$

Since the generation of the GFP starts from $(h-1)$, the lower limit in the Equation 2 is set to $(h-1)$. The GFP of Y contains only one element at level s. So, the MF at level s will always be 0. Hence, the upper limit in the equation 2 is set to $(s+1)$.

The *DiverseRank* of a frequent pattern ranges from $[0,1]$. If all the items in a frequent pattern have the same immediate parent, then the *DiverseRank* is 0. On the other hand, if all the items in a frequent pattern have only "Root" as the common parent, then the *DiverseRank* is 1 for such patterns. We now illustrate the calculation of the *DiverseRank* with Example 5.

Example 5: We calculate the *DiverseRank* of the three frequent patterns of size 3 listed in Table 3. The concept hierarchy of height 3 ($h = 3$) given in Figure 1(a) is used. Refer to the Example 4 for the PLF values.

- Consider the frequent pattern $\{bread,\ butter,\ eggs\}$ from the Table 3. The GFP of $\{bread,\ butter,\ eggs\}$ at level 2 is $\{dairy\}$. This is because all the items belong to the same category "dairy". Thus, the level where the length of the GFP becomes 1 is at level 2, i.e., $s = 2$. Since we are traversing the hierarchy bottom to up, $s+1 \not\geq h-1$ in Equation 4. Thus, the $DR(\{bread,\ butter,\ eggs\})$ is 0.
- Similarly, consider the frequent pattern $Y = \{bread,\ eggs,\ battery\}$ from the Table 3. For the GFPs of Y from level 3 to level 0, refer to the Figure 1(b). Thus, the GFP of Y will contain only one item at the level 0, i.e, $s = 0$. The MF at level 2 is calculated as $MF(Y,2) = \frac{2-1}{3-1} = \frac{1}{2}$. Similarly, the $MF(Y,1) = \frac{2-1}{2-1} = 1$ and $MF(Y,0) = \frac{1-0}{2-1} = 0$.

The $DR(\{bread,\ eggs,\ battery\}$ is calculated as follows.

$$= \left(\frac{1}{2} * \frac{1}{3}\right) + \left(\frac{1}{1} * \frac{2}{3}\right) = 0.82$$

- Consider the frequent pattern $Z = \{butter,\ diapers,\ hair\ spray\}$ from the Table 3. The *GFP* of Z at level 2 is $\{dairy,\ baby\ clothing,\ hair\}$. Similarly, the $GFP(Z,1) = \{fresh\ \ food,\ household,\ beauty\}$ and $GFP(Z,0) = \{root\}$. Thus, the number of the items in the GFP of Z becomes 1 is at level 0, i.e., $s = 0$. The $DR(\{butter,\ diapers,\ hair\ spray\})$ is calculated as follows.

$$= \left(\frac{2}{2} * \frac{1}{3}\right) + \left(\frac{2}{2} * \frac{2}{3}\right) = 1$$

Definition 3. ***Diverse-frequent patterns****: A frequent pattern Y is said to be diverse if the DR(Y) ≥ minDiv, where minDiv is the user-specified minimum diversity threshold.*

Here, the diversity threshold *minDiv*, which varies from 0 to 1 (inclusive), indicates the extent to which the items in the patterns belong to the different categories. The patterns with small *minDiv* value indicates that most of the items are from the same low-level category in the concept hierarchy. The patterns with high *minDiv* indicates that items in the patterns are from different categories. So based on the requirement, the user can vary *minDiv* value.

Problem Definition. Let *DB* be the transactional database on *n* items. Given, *minSup* and *minDiv*, the problem is to extract complete set of frequent patterns that exists in *DB* having *support* and *DiverseRank* values more than *minSup* and *minDiv* values respectively.

3.3 Approach for Mining Diverse-Frequent Patterns

In this section, we first explain the issues of extracting the *diverse-frequent* patterns. Next, we present an item-encoding approach to extract the *diverse-frequent* patterns in an efficient manner. Subsequently, we present the corresponding algorithm.

3.3.1 Issues in extracting Diverse-Frequent Patterns

The extraction of the *diverse-frequent* patterns from the transactional database is complex because the *diverse-frequent* patterns fail to satisfy the anti-monotonic property. Also, there is an issue of generating the GFP of a given frequent pattern at any level. We elaborate on these issues as follows.

- **Anti-monotonic property:** It can be noted that the *diverse-frequent* patterns discovered by *DiverseRank* doesn't satisfy the anti-monotonic property. Anti-monotonic property states if a pattern passes the test, then all its subsets must also pass the test. We illustrate using the following example. Continuing with Example 4, let $\{bread,\ eggs,\ battery\}$ be a frequent pattern and $minDiv = 0.5$. Although the $DR(\{bread,\ eggs,\ battery\})$ is more than the *minDiv*, its subset $\{bread,\ eggs\}$ is not diverse because ($DR\{bread,\ butter\} < 0.5$) as the items "bread" and "eggs" have the same immediate parent. Hence, *DiverseRank* doesn't satisfy the convertible anti-monotonic property. As a result, the *DiverseRank* cannot be pushed into any existing frequent pattern mining algorithms to further reduce the search space.
- **Generating GFP:** The complexity of extracting the *diverse-frequent* patterns majorly depends on the generation of *GFP* at each level. Given a concept hierarchy of height *h*, the complexity of finding the parent of a node is $O(m)$, where m is the total number of nodes in the concept hierarchy. Therefore, the total complexity of generating the *GFP* of a frequent pattern Y at a level l is $O(m \times |GFP(Y,\ l+1|)$, where $0 \leq l < h$ and $|GFP(Y,\ l+1)|$ is the number of items in $GFP(Y,\ l+1)$. Since the number of nodes in the concept hierarchy can be very large, it is computationally expensive to measure *DiverseRank* of a frequent pattern.

To ease the process of extracting of *diverse-frequent* patterns from *DB*, we divide the process into two steps.

(i) **Extraction of frequent patterns from the DB:** One of the existing frequent pattern extraction algorithm [1, 8] can be used to extract the frequent patterns.

(ii) **Extraction of diverse-frequent patterns from the frequent patterns:** We make an effort to reduce the computational complexity of generating GFP to extract *diverse-frequent* patterns using the *item-encoding* technique [7].

3.3.2 Item-encoding technique

In this section, we explain the item-encoding technique which can be used to extract *diverse-frequent* patterns in an efficient manner. In the concept hierarchy, all the nodes under a parent node are numbered from left to right. The encoded representation of an item is the traversal path from the root to the respective item. An example of how the item encoding is done is given as follows.

In Figure 1(a), each node at a level is numbered from left to right. The encoded representation of the item "butter" is "1112". The first digit, "1", represents the "root" at level-0, the second, "1" for "fresh food" at level-1, the third, "1", for "dairy" at level-2 and the last, "2", for the item "butter". The list of the encoded items used in the transaction table 1 are shown in the Table 4. The encoded version of the transactional database in Table 1 is shown in the Table 5. The list of the frequent patterns discovered from this encoded transaction table is given in Table 6 below.

Table 4: Encoded items using Figure 1

Items	Encoded Items
bread	1111
butter	1112
eggs	1113
milk	1114
tea	1121
whiskey	1123
orange	1131
apple	1132
cherry	1133
diapers	1211
battery	1221
hair spray	1311

Table 5: Encoded transaction database.

TID	Encoded Transactions
1	1111, 1112, 1113, 1131
2	1111, 1112, 1113, 1132
3	1111, 1113, 1221, 1114 , 1121
4	1111, 1113, 1221, 1133
5	1112, 1211, 1311, 1123
6	1112, 1211, 1311, 1221

Table 6: Encoded frequent patterns.

Frequent Pattern	Encoded Frequent Patterns
{bread, butter, eggs}	{1111, 1112, 1113}
{bread, eggs, battery}	{1111, 1113, 1221}
{butter, diapers, hair spray}	{1112, 1211, 1311}

The advantage of using the item-encoding is that the parent of an item at a level l can be computed easily without traversing the complete concept hierarchy. Given a concept hierarchy of height h, the parent for an item at a level l can also be encoded by replacing the last $(h-l)$ characters of the item with "*". For example, the parent for "butter" at level 3 is "dairy". The encoded representation of "dairy" is "111*". Similarly, the "fresh food" is encoded as "11**" and the "root" is encoded as "1***".

The following example illustrates how to generate *GFP* at the various levels of concept hierarchy using the encoding. Consider the encoded frequent 4-pattern $Y = \{1111,\ 1112,\ 1121,\ 1311\}$. The *GFP* for Y at level 2 is $\{111*, 112*, 131*\}$. It is because, the parent for "1111" and "1112" at level 2 is "111*". Similarly, the *GFP* at level 1 is $\{11**,\ 13**\}$ and at level 0 is $\{1***\}$.

It should be noted that when the hierarchies become very broad (i.e. the number of children under a node > 9), we switch from the string representation to the vector representation. For example, an item which was earlier encoded as 1321 will now be encoded as $< 1,3,2,1 >$. This representation will take care of the situation when the number of children under a node becomes greater than 9. For example, the 13^{th} children under a node can be encoded as $< 1,3,2,13 >$. However, the complexity is not affected.

Complexity of generating GFP with item-encoding technique: The complexity of finding the parent of an encoded node is $O(h)$, where h is the height of concept hierarchy. For the real-world concept hierarchies, h is generally small and ranges from $[3-10]$. Thus, the total complexity of generating a *GFP* of Y at a level l using encoding is $O(h \times |GFP(Y,\ l+1)|) \sim O(GFP(Y,\ l+1))$, where $0 \leq l < h$ and $|GFP(Y,\ l+1)|$ is the number of items in the $GFP(Y,\ l+1)$.

3.3.3 Proposed Algorithm

In this section, we present the algorithm called *diverseFP* to mine *diverse-frequent* patterns. The input to the *diverseFP*, is the list of encoded frequent patterns, height of the concept hierarchy, and the user-specified minimum diversity threshold *minDiv*. The *diverseFP* algorithm calls the procedure *GFPgen* to generate the GFP of a frequent pattern at a level l. The input to the procedure *GFPgen* is an encoded frequent pattern, the current level and the height of the concept hierarchy.

The description of the *diverseFP* is as follows. Here, we present the algorithm by assuming PLF for computing LF.

1 Calculate the PLF for each level of concept hierarchy.

2 Repeat the steps 2.1 to 2.4 for every encoded frequent pattern Y.

- 2.1 Set *DiverseRank*(Y) to zero.
- 2.2 Set $l = (h-1)$.
- 2.3 Repeat steps 2.3.1 to 2.3.3 until the length of the $GFP(Y,l)$ becomes 1.
  - 2.3.1 Generate $GFP(Y,l)$.
  - 2.3.2 Calculate the *DiverseRank* value at that level using Equation 4.
  - 2.3.3 Set $l = l-1$.
- 2.4 If *DiverseRank*$(Y) >$ *minDiv*, output it as a *diverse-frequent* pattern.

The pseudocode of the proposed *diverseFP* is given in Algorithm 1. Lines 1-3 of Algorithm 1 initializes the PLF for all the levels of a given concept hierarchy of height h. Line 5 of Algorithm 1 initializes the *DiverseRank* for a encoded frequent pattern Y. Line 6 of Algorithm 1 initializes the level counter to one level above the leaf level of the concept hierarchy. Line 7 of Algorithm 1 initializes the $GFP(Y,h)$, i.e. the generalized-frequent pattern at the leaf level. Procedure *GFPgen* in Line 9 of Algorithm 1 generates the *GFP* of a encoded frequent pattern at the current level. Lines 10-12 of Algorithm 1 checks whether the size of $GFP(Y,l)$ is greater than 1. Line 13 in Algorithm 1 calculates the MF at the current level. The *DiverseRank* is calculated by multiplying MF and PLF at each level and

Algorithm 1 diverseFP (*EFP*: list of encoded frequent patterns, *h*: height of the concept hierarchy, *minDiv*: user-specified minimum diversity threshold)

1: **for** $(l = h-1; l \leq 1; l--)$ **do**
2: Initialize $PLF(l) = \frac{2*(h-l)}{(h-1)*h}$;
3: **end for**
4: **for** each $p \in EFP$ **do**
5: Initialize *DR* to 0;
6: Initialize l to $(h-1)$; {Start from level $(h-1)$.}
7: Initialize $GFP(p,h) = |p|$; {GFP at leaf level of the hierarchy is the pattern itself.}
8: **while** *true* **do**
9: Call *GFPgen* to generate $GFP(p,l)$;
10: **if** $(|GFP(p,l)| == 1)$ **then**
11: break;
12: **end if**
13: $MF(p,l) = \frac{|GFP(p,l)|-1}{|GFP(p,l+1)|-1}$;
14: $DR = DR + PLF(l) * MF(p,l)$;
15: Decrease l by 1;
16: **end while**
17: **if** $DR > minDiv$ **then**
18: Output p with *DR* value;
19: **end if**
20: **end for**

Procedure 2 GFPgen(*patt*: pattern, l: the current level, h: height of the concept hierarchy)

1: Initialize $GFP(p,l)$;
2: **for** (each item $\in$ *patt* **do**
3: Initialize *Parent*;
4: **for** $(j = 0; j \leq l; j++)$ **do**
5: Append $item[j]$ character in *Parent*;
6: **end for**
7: Append $(h-l)$ number of "*"s in the *Parent*;
8: **if** *Parent* doesn't exist in $GFP(p,l)$ **then**
9: Add *Parent* in the $GFP(p,l)$;
10: **end if**
11: **end for**
12: return $GFP(p,l)$;

the summation is updated in Line 14 of Algorithm 1. In Line 15 of Algorithm 1, the level counter is decreased by 1 to move up in the concept hierarchy. Lines 17 and 18 checks whether an encoded frequent pattern is diverse or not.

Similarly, Line 1 of Procedure 2 initializes an empty string to store the *GFP* of the pattern at the current level. Lines 4 to 7 of Procedure 2 generates the parent (higher level) of an item at the corresponding level. Line 8 and 9 checks whether the parent node exists before in the *GFP* or not. Finally, line 12 in the Procedure 2 returns the *GFP* of the pattern at the corresponding level.

We now illustrate the algorithm by using an encoded frequent pattern $Y = \{1111,\ 1112,\ 1221\}$. The height of the concept hierarchy is 3 and the *minDiv* is chosen as 0.5. The weights are assigned using the formula $PLF(l) = \frac{2*(h-l)}{(h-1)*h}$. According to this formula, $PLF(2) = 1/3$ and $PLF(1) = 2/3$. The *DiverseRank* for the encoded frequent pattern Y is initialized to 0, $DR(Y) = 0$. Initially, we start from one level above the leaf level. Thus, the current level = 2. The *GFP* of Y at level 2, i.e., $GFP(Y,2) = \{111*, 122*\}$ is generated by the procedure *GFPgen*. The $MF(Y,2) = 1/2$ is calculated and multiplied by the corresponding $PFL(2) = 1/3$. The *DiverseRank* is updated to $1/6$. After updating the summation, we move one level up in the hierarchy, i.e., to the level 1. Again *GFPgen* procedure is called and $GFP(Y,1) = \{11**, 12**\}$ is generated. The $MF(Y,1) = 1$ is calculated and multiplied by $PFL(1) = 2/3$. The *DiverseRank* is updated to $5/6$. Now we move to the level 0 in the hierarchy, the *GFP* of Y at level 0 is $GFP(Y,1) = \{1***\}$. The pattern has completely merged, so we stop at the level 0. The *DiverseRank* for Y is $5/6 = 0.82$. The pattern Y is a *diverse-frequent* pattern as $DR(Y) > 0.5$.

4 Experiments and Results

Since there is no existing approach to discover *diverse-frequent* patterns, we only carry out the experiment to extract the *diverse-frequent* patterns and analyze how they differ from the frequent patterns. The experiments were carried out on the classical *R* "groceries" market basket analysis data set [6]. The groceries data set contains 30 days of point-of-sale transaction data from a typical local grocery outlet. The data set contains 9,835 transactions and 169 items. The average transaction size in the data set is 4.4. The maximum and minimum transaction size is 32 and 1 respectively.

To generate a concept hierarchy for the items, a web-based Grocery API provided by Tesco [23] (a United Kingdom Grocery Chain Store) is used. Some of the items that were not listed in the concept hierarchy of Tesco are added manually by consulting the domain experts. The total number of nodes in the concept hierarchy were 247 and the height of the concept hierarchy is 4.

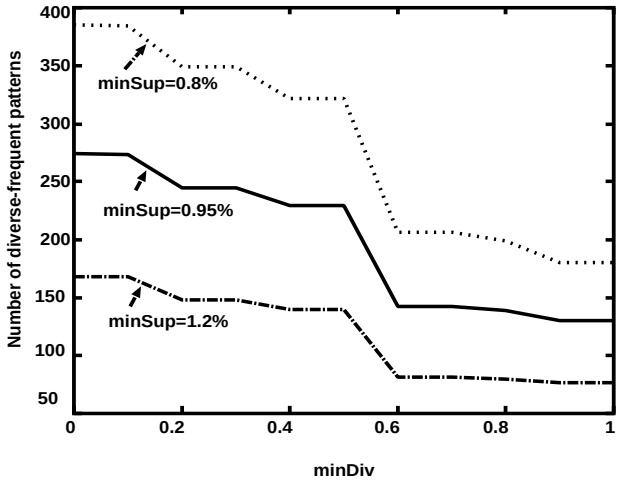

Figure 3: Number of *diverse-frequent* patterns vs. *minDiv*.

Table 7: Diverse-frequent 3-patterns with $DR = 1$

Top 10 diverse frequent patterns	**DiverseRank**	**Support(%)**
{soda, whole milk, shopping bags}	1	0.7
{rolls-buns, whole milk, newspapers}	1	0.8
{rolls-buns, soda, sausages}	1	1.0
{rolls-buns, bottled water, other vegetables}	1	0.7
{soda, rolls-buns, other vegetables}	1	1.0
{rolls-buns, bottled water, yogurt}	1	0.7
{rolls-buns, soda, shopping bags}	1	0.6
{rolls-buns, soda, whole milk}	1	0.9
{rolls-buns, soda, yogurt	1	0.9
{rolls-buns, bottled water, whole milk}	1	0.9

Table 8: Top 10 frequent 3-patterns along with DR and support value

Top 10 frequent patterns	**Support(%)**	**DiverseRank**
{whole milk, other vegetables, root vegetables}	2.3	0.33
{yogurt, whole milk, other vegetables}	2.2	0.33
{rolls-buns, whole milk, other vegetables}	1.8	0.75
{whole milk, tropical fruit, other vegetables}	1.7	0.5
{rolls-buns, yogurt, whole milk}	1.6	0.833
{yogurt, whole milk, root vegetables}	1.5	0.33
{yogurt, whole milk, tropical fruit}	1.5	0.33
{whipped sour cream, whole milk, other vegetables}	1.5	0.33
{whole milk, pip fruit, other vegetables}	1.4	0.5
{soda, whole milk, other vegetables}	1.4	0.75

Figure 3 shows the number of *diverse-frequent* patterns discovered in the grocery dataset at different *minSup* and *minDiv* thresholds values. The X-axis represents the *minDiv* thresholds ranging from 0 to 1 and the Y-axis represents the number of *diverse-frequent* patterns discovered at the corresponding *minDiv* threshold value. It can be observed that with the increase in *minDiv* , the number of *diverse-frequent* patterns has decreased irrespective of *minSup* threshold. The reason is as follows. When $minDiv = 0$, all the frequent patterns are the *diverse-frequent* patterns. As the value of *minDiv* is increased, the number of frequent patterns which satisfy $DiverseRank > minDiv$ are reduced due to the fact that the items in the several patterns belong to one or few categories.

The list of the top 10 *diverse-frequent* 3-patterns along with their support and *DiverseRank* is given in Table 7. Similarly, Table 8 contains the list of top 10 frequent 3-patterns with respect to the frequency of the patterns(support) along with their *DiverseRank* value. Highest support of a frequent 3-pattern is 2.3 (%) and the highest *DiverseRank* value of a frequent 3-pattern is 1. From the two tables, one can observe that there are no common patterns between them. Thus, we can say that the frequent patterns having the highest *DiverseRank* value may not be the patterns with the highest support. Similarly, the frequent patterns with the highest support may not have the highest value of *DiverseRank*. This indicates that the *DiverseRank* measure is able to generate some new information which *support* measure wasn't able to generate.

5 Conclusion and Future Work

In this paper, we have proposed a new interestingness measure called *DiverseRank* to rank the frequent patterns based on the notion of diversity. The proposed approach uses the concept hierarchy of items for computing *DiverseRank* of a frequent pattern. We have also proposed an efficient algorithm to extract *diverse-frequent* patterns by using the *item-encoding* technique. The experiments on the real world data set show that the *diverse-frequent* patterns differ from frequent pattern knowledge.

As a part of future work, we are planning to investigate the extraction of *diverse-frequent* patterns by considering transactional databases with imbalanced concept hierarchies. We are also planning to extend the notion of *diverse-frequent* patterns to improve the performance of clustering, classification and recommendation algorithms.

References

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 22:207–216, June 1993.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Advances in knowledge discovery

and data mining. chapter Fast discovery of association rules, pages 307–328. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996.

[3] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations, 1997.

[4] Y. Chen, G.-R. Xue, and Y. Yu. Advertising keyword suggestion based on concept hierarchy. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 251–260, New York, NY, USA, 2008. ACM.

[5] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 38, September 2006.

[6] M. Hahsler, K. Hornik, and T. Reutterer. Implications of probabilistic data modeling for mining association rules. In *Proceedings of the 29th Annual Conference of the Gesellschaft fr Klassifikation e.V., University of Magdeburg, March 911, 2005*, pages 598–605. Springer, 2006.

[7] J. Han and Y. Fu. Mining multiple-level association rules in large databases, 1999.

[8] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *SIGMOD Rec.*, 29:1–12, May 2000.

[9] R. J. Hilderman and H. J. Hamilton. *Knowledge Discovery and Measures of Interest*. Kluwer Academic Publishers, Norwell, MA, USA, 2001.

[10] J. Hu and A. Mojsilovic. High-utility pattern mining: A method for discovery of high-utility item sets. *Pattern Recogn.*, 40:3317–3324, November 2007.

[11] T. Hu, S. Y. Sung, H. Xiong, and Q. Fu. Discovery of maximum length frequent itemsets. *Inf. Sci.*, 178:69–87, January 2008.

[12] R. A. Huebner. Diversity-based interestingness measures for association rule mining, 2009.

[13] R. U. Kiran and P. K. Reddy. Mining periodic-frequent patterns with maximum items' support constraints. In *Proceedings of the Third Annual ACM Bangalore Conference*, COMPUTE '10, pages 1:1–1:8, New York, NY, USA, 2010. ACM.

[14] B. Liu, W. Hsu, and S. Chen. Using general impressions to analyze discovered classification rules. pages 31–36. AAAI Press, 1997.

[15] B. Liu, W. Hsu, L.-F. Mun, and H.-Y. Lee. Finding interesting patterns using user expectations. *Knowledge and Data Engineering, IEEE Transactions on*, 11(6):817 –832, nov/dec 1999.

[16] K. McGarry. A survey of interestingness measures for knowledge discovery. *Knowl. Eng. Rev.*, 20:39–61, March 2005.

[17] E. Omiecinski. Alternative interest measures for mining associations in databases. *Knowledge and Data Engineering, IEEE Transactions on*, 15(1):57 – 69, jan.-feb. 2003.

[18] J. Pei, J. Han, and L. V. S. Lakshmanan. Mining frequent itemsets with convertible constraints. pages 433–442. IEEE Computer Society.

[19] T. M. Quang, S. Oyanagi, and K. Yamazaki. Mining the k-most interesting frequent patterns sequentially. In *IDEAL*, pages 620–628, 2006.

[20] A. Silberschatz and A. Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 8:970–974, 1996.

[21] R. Srikant and R. Agrawal. Mining generalized association rules. pages 407–419, 1995.

[22] S. K. Tanbeer, C. F. Ahmed, B.-S. Jeong, and Y.-K. Lee. Discovering periodic-frequent patterns in transactional databases. In *Proceedings of the 13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, PAKDD '09, pages 242–253, Berlin, Heidelberg, 2009. Springer-Verlag.

[23] Tesco. Grocery api. `https://secure.techfortesco.com/tescoapiweb/`, 2011.

[24] S. Thomas and S. Sarawagi. Mining generalized association rules and sequential patterns using sql queries. In *Prof. of 4th Intl. Conf. on Knowledge Discovery and Data Mining (KDD98*, pages 344–348. AAAI Press, 1998.

[25] J. Wang, J. Han, Y. Lu, and P. Tzvetkov. Tfp: an efficient algorithm for mining top-k frequent closed itemsets. *Knowledge and Data Engineering, IEEE Transactions on*, 17(5):652 – 663, may 2005.

[26] H. Yun, D. Ha, B. Hwang, and K. H. Ryu. Mining association rules on significant rare data using relative support. *J. Syst. Softw.*, 67:181–191, September 2003.

[27] M. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. *Knowledge and Data Engineering, IEEE Transactions on*, 17(4):462 – 478, april 2005.

[28] N. Zbidi, S. Faiz, and M. Limam. On mining summaries by objective measures of interestingness. *Mach. Learn.*, 62:175–198, March 2006.