A Revenue-based Product Placement Framework to Improve Diversity in Retail Businesses

Pooja Gaur[1], P. Krishna Reddy[1], M. Kumara Swamy[2] and Anirban Mondal[3]

[1] IIIT, Hyderabad, India
pooja.gaur@research.iiit.ac.in, pkreddy@iiit.ac.in
[2] CMR Engineering College, Hyderabad, India
m.kumarswamy@cmrec.ac.in
[3] Ashoka University, Haryana, India,
anirban.mondal@ashoka.edu.in

Abstract. Product placement in retail stores has a significant impact on the revenue of the retailer. Hence, research efforts are being made to propose approaches for improving item placement in retail stores based on the knowledge of utility patterns extracted from the log of customer purchase transactions. Another strategy to make any retail store interesting from a customer perspective is to cater to the varied requirements and preferences of customers. This can be achieved by placing a wider variety of items in the shelves of the retail store, thereby increasing the *diversity* of the items that are available for purchase. In this regard, the key contributions of our work are three-fold. First, we introduce the problem of concept hierarchy based diverse itemset placement in retail stores. Second, we present a framework and schemes for facilitating efficient retrieval of the diverse top-revenue itemsets based on a concept hierarchy. Third, we conducted a performance evaluation with a real dataset to demonstrate the overall effectiveness of our proposed schemes.

Keywords: Utility mining · Patterns · Diversity · Concept hierarchy · Retail management · Itemset placement

1 Introduction

Over the past decade, we have been witnessing the prevalence of several popular medium-to-mega-sized retail stores with relatively huge retail floor space. Examples of medium-sized retail stores include Walmart Supercenters, while Macy's Department Store at Herald Square (New York City, US) and Dubai Mall (Dubai) are examples of mega-sized retail stores. Typically, such mega-sized stores have more than a million square feet of retail floor space. Such large retail stores typically have a huge number of aisles, where each aisle contains items that are stocked in the slots of the shelves.

In such large retail stores, tens of thousands of items need to be placed in the typically massive number of slots (of the shelves) of a given store in a way that improves the sales revenue of the retailer, thereby posing several research challenges. In this regard, existing works have focused on scalable supply chain management [2], inventory management [35], stock-out management [35], identification of frequent and/or high-utility itemsets [6] and strategic placement of items (products) as well as itemsets for improving the revenue of retail stores [9]. In essence, given the importance and relevance of the retail sector in today's world, this is an active area of research with several ongoing efforts.

Incidentally, another strategy of improving the revenue of the retailer is to provide customers with a wider *diversity* of items so that customers have more choice in terms of variety of the items that are available to them for purchase. For example, a customer, who wishes to buy soft drinks, would generally prefer a wider and more diverse range of soft drinks (e.g., Coke, Coke-Light, Pepsi, Sprite, Limca, etc.) to choose from. As another example, there could also be different brands and varieties of soaps, shampoos and tomato sauce in a retail store. It is a well-known fact in the retail industry that customers belong to different market segments e.g., Alice prefers a specific brand of tomato sauce or a specific brand of a soft drink, while Bob may prefer another brand of tomato sauce as well as another brand of a soft drink.

Given that each market segment is relatively large in practice, *increasing the diversity* in a retail store generally attracts customers from *multiple market segments*, thereby in effect increasing the overall foot-traffic of the retail store. For example, if a retail store were to only stock Coke in its shelves, it would likely miss out on the customers, whose preference could be Coke-Light or Pepsi. In fact, research efforts are being made to improve diversity for real-world retail companies by collecting data about sales, customer opinions and the views of senior managers [13, 18, 33]. In essence, there is an opportunity to improve the revenue of the retailer by incorporating the notion of diversity (of items) during the placement of itemsets.

In the literature, itemset placement approaches have been proposed in [6–9] to improve the revenue of the retailer by extracting the knowledge of high-revenue itemsets from specialized index structures. These index structures are populated based on the information in the log of customer purchase transactions. However, these works do not address itemset placement based on concept hierarchy based diversity of items. Moreover, the notion of diverse frequent patterns have been proposed in [20, 21] to extract the knowledge of diverse items based on the customer purchase transactional dataset as well as a given concept hierarchy of the items. Furthermore, it has also been demonstrated that the knowledge of diverse frequent patterns can potentially be used for improving the performance of recommender systems. Observe that while the works in [20, 21] address the diversity issue, they do not consider the context and issues associated with itemset placement in retail stores.

In this paper, we have made an effort to improve the diversity of itemset placement in retail stores by proposing the notion of concept hierarchy based diverse revenue itemsets. In particular, based on the notion of diverse revenue patterns, we have proposed a framework, which improves the diversity of itemset placement, while not compromising the revenue. To this end, we have proposed two itemset placement schemes.

The main contributions of this paper are three-fold:

- We introduce the problem of concept hierarchy based diverse itemset placement in retail stores.
- We present a framework and schemes for facilitating efficient retrieval of the diverse top-revenue itemsets based on a concept hierarchy.
- We conducted a performance evaluation with a real dataset to demonstrate the overall effectiveness of our proposed schemes.

To the best of our knowledge, this is the first work that incorporates the notion of concept hierarchy (of items) for increasing the diversity in itemset placement for retail stores in order to improve the revenue of the retailer.

The remainder of this paper is organized as follows. In Section 2, we discuss existing works and background information. In Section 3, we present the context of the problem. In Section 4, we present our proposed itemset placement framework. In Section 5, we report the results of our performance evaluation. Finally, we conclude the paper in Section 6 with directions for future work.

2 Related Work and Background Information

In this section, we discuss an overview of existing works as well as some background information.

2.1 Related work

The placement of items on retail store shelves considerably impacts sales revenue [4, 5, 11, 17]. The placement of products in the shelves of a retail stores is an important research issue, which has motivated several efforts [10, 11, 17, 34, 35]. In [10, 34, 35], it has been studied that how the visibility of the items impacts the total revenue generated by the store.

Several research efforts have also been made in the area of utility mining [23, 29, 30]. The goal of these efforts is to identify high-utility itemsets from transactional databases of customer transactions. The work in [30] proposed the Utility Pattern Growth (UP-Growth) algorithm for mining high-utility itemsets. It keeps track of the information concerning high-utility itemsets in a data structure, which is designated as the Utility Pattern Tree (UP-Tree). Moreover, it uses pruning strategies for candidate itemset generation. The work in [23] proposed the HUI-Miner algorithm for mining high-utility itemsets. It used a data structure, designated as the *utility-list*, for storing utility and other heuristic information about the itemsets, thereby enabling it to avoid expensive candidate itemset generation as well as utility computations for many candidate itemsets. The work in [29] proposed the CHUI-Miner algorithm for mining closed high-utility itemsets. In particular, the algorithm in [29] is able to compute the utility of itemsets without generating candidates. While these existing works have focused primarily on identifying high-utility itemsets, none of these works has addressed the itemset placement problem for retail stores.

Placement of itemsets in retail stores has been investigated in [4–9, 14, 17] The work in [5] has shown that the profits of retail stores can be improved by making efficient use of the shelf space. The work in [17] proposed a model and an algorithm for maximizing the total profit of a given retail store. In particular, the algorithm proposed in [17] allocates shelf space by considering the available space and the cost of the product. The work in [4] exploited association rules to address the item placement problem in retail stores. In particular, it used frequent patterns to select the items for placement and demonstrated the effectiveness of frequent patterns towards item placement. Furthermore, the work in [14] investigated the mixed-integer programming model towards retail assortment planning and retail shelf space allocation for maximizing the revenue of the retailer.

A framework has been presented in [6, 9] to determine the top-revenue itemsets for filling a required number of (retail store) slots, given that items can physically vary in terms of size. A specialized index structure, designated as the STUI index has been proposed in [6, 9] for facilitating quick retrieval of the top-revenue itemsets. Furthermore, an approach has been proposed in [7] for placing the itemsets in the premium slots of large retail stores for achieving diversification in addition to revenue maximization. Notably, the notion of diversification in the work in [7] is based on reducing the placement of duplicate items. This facilitates towards improving long-term (retail) business sustainability by attempting to minimize the adverse impact of macro-environmental risk factors, which may dramatically reduce the sales of some of the retail store items. As such, the emphasis of the work in [7] is orthogonal to the focus of this work. Moreover, the work in [7] proposed the kUI (k-Utility Itemset) index for quickly retrieving top-utility itemsets of different sizes. Additionally, the work in [8] addresses the problem of itemset placement by considering slots of varied premiumness. Observe that none of these existing works considers concept hierarchy based diversity in the context of itemset placement for retail stores.

The issue of improving the diversity in item placement for retail stores has been investigated in [13, 18]. The work in [18] aimed at increasing diversity in retail items by mixing the food items and other retailer items. The work in [13] differentiated between two types of retailers, namely (a) limited-diversity retailers specializing in either food or non-food product categories and (b) extended-diversity retailers offering both food and non-food product categories. However, these works do not consider a concept hierarchy based approach for improving the diversity of retail items in a more fine-grained manner.

Interestingly, the work in [3] has proposed an approach to improve the diversity of recommendations. In [12, 28], efforts have been made to improve the diversity of search engine recommendations. In the area of recommender systems [20, 21], a model of concept hierarchy based diversity has been introduced and it has been demonstrated that it improves the variety of recommendations in e-commerce recommendation systems.

2.2 About Concept Hierarchy based Diversity

In [20–22], an approach has been proposed to compute the diversity of a given pattern (itemset) by using concept hierarchies. In a given domain, a set of items can be grouped into a category and a pattern may contain items belonging to different categories. The items and the categories form a concept hierarchy. The concept hierarchy is a structure representing the relationship between items and their categories in a tree, where the items occupy the leaf nodes. Here, the intermediate nodes represent the categories of the items. The *root* of the concept hierarchy is a virtual node. A sample concept hierarchy is depicted in Figure 1.

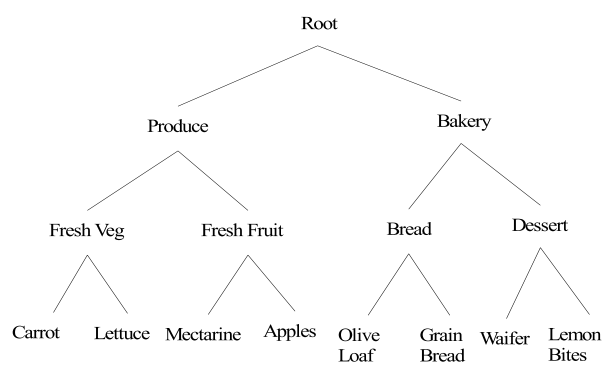


Fig. 1: A Sample Concept Hierarchy from Instacart Market Basket Analysis Dataset

The notion of diversity captures the extent of items in the pattern belonging to multiple categories. The notion of diversity of a pattern captures the merging behavior, i.e., how the items in lower level merge to the items at higher level. If the items of the pattern merge quickly to the higher level node, the pattern has low diversity. Similarly, if the items of the pattern merge slowly, that pattern has relatively high diversity. As an example, consider the items and concept hierarchy depicted in Figure 1. Here, the pattern {Carrot, Lettuce} has low diversity as it merges quickly, whereas the pattern {Carrot, Grain Bread} has the maximum diversity as the items merge at the *root*.

Given a pattern and a concept hierarchy [20–22], the diversity is computed using the Diversity Rank ($DRank$) metric. Given a set of items and the concept hierarchy C, the formula to compute the $DRank$ of a pattern Y is $DRank(Y) = \left(\frac{P-(|Y|+h-1)}{(h-1)(|Y|-1)}\right)$. Here, P is the number of edges in the sub-tree of the concept hierarchy C formed by the items of the pattern Y, and h is the height of C.

2.3 About k-Utility Itemset index

The work in [6] proposed an itemset placement scheme based on efficiently determining and indexing the top high-utility itemsets. The itemsets are organized in the form of the kUI (k-Utility Itemset) index to retrieve top-utility itemsets quickly. We shall now provide an overview of the kUI index.

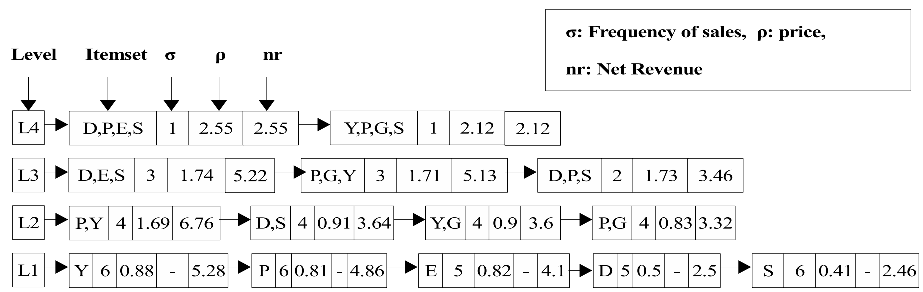


Fig. 2: Example of the kUI Index

The kUI is a multi-level index, where each level corresponds to a particular itemset size. It optimizes the task of getting potential itemsets of a given size from a large number of itemsets. The kUI index is organized into N levels, where each level contains a fixed number λ of itemsets. At the k^{th} level, kUI index stores *top*-λ ordered itemsets of size k.

Figure 2 shows an illustrative example of the kUI index. Each node in the linked list at the k^{th} level contains a tuple of $\langle itemset, \sigma, \rho, nr\rangle$, where ρ is the price of the a given itemset $itemset$, σ is the frequency of sales of the itemset and nr is the net revenue, which is computed as $(\rho \times \sigma)$. The itemsets at each level of the kUI index are sorted in descending order based on net revenue to facilitate quick retrieval of the top-revenue itemsets.

3 Context of the Problem

Consider a finite set γ of m items $\{i_1, i_2, \ldots, i_m\}$. We assume that each item of the set γ is physically of the same size. Moreover, we assume that all slots are of equal size and each item consumes only one slot. Each item i_j is associated with a price ρ_j and a frequency σ_j. The revenue of a given itemset is the product of its price and its frequency of sales. Observe that σ_j is the number of transactions in which i_j occurs in the transactional database. The price of a given itemset is the sum of the prices of all of the items in that itemset. The net revenue (nr) of a given itemset is the price of the itemset multiplied by the number of times that the itemset was purchased. Furthermore, recall our discussion in Section 2.2 concerning the notion of diversity.

Problem Statement: Given a set γ of items, a transactional database D over set γ, price ρ_i of the i^{th} item in set γ, a concept hierarchy C over the items in set γ and a given number of slots to be filled, the problem is to determine the placement of itemsets in the slots such that the total revenue of the retailer as well as the diversity should be improved.

4 Proposed Framework

In this section, we first discuss the basic idea of the proposed framework. Next, we explain the proposed schemes.

4.1 Basic Idea

Transactional data of retail customers provide rich information about customer purchase patterns (itemsets). Given a set of transactions, an approach to identify high-revenue itemsets has been explained in [6], as previously discussed in Section 2.3. We designate this approach as the Revenue-based Itemset Placement (RIP) approach. Under the RIP approach, it is possible to maximize the revenue of the retailer by extracting and placing high-revenue itemsets. However, the RIP approach does not consider diversity of items, while performing itemset placement. Recall that in Section 2.2, we have discussed an approach [22, 21] to compute the $DRank$ (Diversity Rank) of a given itemset, given a set of items and a concept hierarchy.

The issue is to develop a methodology to identify potential high-revenue itemsets with relatively high $DRank$ values. The basic idea is to propose a new ranking mechanism to rank itemsets based on both revenue and diversity of the itemsets. Hence, given an itemset X, we propose a new measure, designated as *Diverse Net Revenue (dnr)*, of X, which is equal to the product of the net revenue (nr) and the $DRank$ value of X. Given a pattern X, $dnr(X) = DRank(X) \times nr(X)$. In essence, the implication of the dnr measure is that the selected itemsets for placement should have either a high value of nr or a high value of $DRank$ or both.

Based on the notion of diverse revenue itemsets, we propose a scheme, which we shall henceforth refer to as the Diverse Rank Itemset Placement (DRIP) scheme. In particular, the DRIP scheme comprises the following: (i) a methodology to identify top itemsets with high dnr value and building an index called Concept hierarchy based Diversity and Revenue Itemsets ($CDRI$) index and (ii) a methodology to place the itemsets by exploiting the $CDRI$ index, given the number of slots as input.

Observe that the proposed DRIP scheme prioritizes itemsets with high net revenue and high $DRank$ values. Consequently, it may miss some of the top-revenue itemsets with low $DRank$ values. As a result, even though DRIP improves both revenue and diversity, it may not meet the performance of the RIP approach [7] on the net revenue aspect. Recall that the RIP approach exploits the kUI index, as previously discussed in Section 2.3. Hence, we propose another approach called the Hybrid Itemset Placement (HIP) approach, which considers the top-revenue itemsets as in the RIP approach and the top itemsets with high revenue and high *DRank* values as in the DRIP approach. We shall now discuss the details of our two proposed approaches.

4.2 The DRIP approach

We shall first discuss our proposed DRIP approach and use an illustrative example to explain the approach. The DRIP approach comprises two main components: 1. *Building of the CDRI index* and 2. *Itemset placement scheme*. The input to the proposed approaches is a transactional database consisting historical transactions performed by the users at a retail store, item prices and a Concept hierarchy of the items in the transactions.

1. Building of the CDRI index

$CDRI$ stands for *Concept hierarchy based Diversity and Revenue Itemsets index*. $CDRI$ is a multi-level index containing N levels. At each level of the $CDRI$ index, λ itemsets are stored. The k^{th} level of the $CDRI$ index corresponds to itemsets of size k.

Each level of the $CDRI$ corresponds to a hash bucket. The data is stored as a linked list of nodes at each level. The node is a data structure having required information for an itemset. Each node in the linked list at the k^{th} level contains a record consisting of the following fields: $\langle itemset,\ \sigma,\ \rho,\ DRank,\ dnr\rangle$. Here, ρ is the price of the given itemset *itemset*, σ is the frequency of sales of the itemset. Here, $dnr = DRank \times \sigma \times \rho$.

The $CDRI$ index is built level by level. The level 1 of the $CDRI$ index is formed by inserting a record for each item. The following process is repeated to compute the itemsets at level k. We first extract candidate itemsets by concatenating the itemsets of level $k-1$ with the items of level 1. Next, we remove the duplicate items in each itemset and exclude all itemsets, whose size does not equal k. We compute the support for the itemsets by scanning the transactional database. For each itemset X, price of X is calculated. The price of the itemset is equal to the sum of price of the items in it and $DRank(X)$ is computed using the formula to compute $DRank$. Next, we compute dnr for each itemset. The itemsets are then sorted w.r.t. dnr and the top-λ itemsets are placed at the k^{th} level of the $CDRI$ index.

Figure 3 depicts an example of the $CDRI$ index as used in the proposed approach. Each node in the linked list at k^{th} level contains a tuple of ($itemset$, σ, ρ, $DRank$, dnr), where ρ is the price of the itemset *itemset*, σ is the frequency of sales of the itemset, $DRank$ is the diversity rank of the itemset and dnr (diverse Net Revenue) is $DRank \times \sigma \times \rho$.

Algorithm 1 contains the pseudo-code to build the $CDRI$ index. The algorithm essentially has two parts i.e., first building the level 1 of index and then building level 2 to *max_level* of the index. All items (i.e., one-itemsets) are picked from the transactional dataset T and placed in a temporary list A (Line 2). For the items in A, the number of times an itemset is found in T is

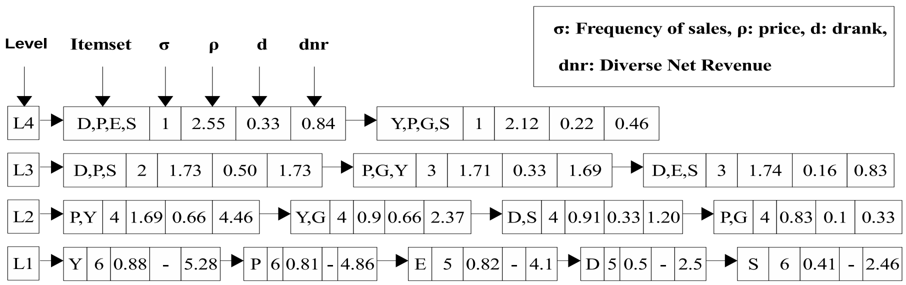


Fig. 3: Illustrative Example of the CDRI Index

Algorithm 1: Building of $CDRI$ Index

Input: T: Transactional database, price ρ_i for each item i, C: Concept Hierarchy, max_level: Number of levels in $CDRI$, λ: Number of entries in each level of $CDRI$, st: support threshold
Output: $CDRI$ /* CDRI index */
Initialize $CDRI[max\_level][\lambda]$ to NULL
Scan T and compute support $s(X)$ for each item X in T and store it in array A /* Level 1 of CDRI */
Sort A based on the support and remove all items which have the support less than the st
Select top-λ items from A and for each item X, insert $\langle X, s(X), \rho(X)\rangle$ in $CDRI[1][\lambda]$ in the descending order of the support*price
for $lev = 2$ *to* $lev = max\_level$ **do**
    Create combination of itemsets in lev - 1 with the items in level 1 and generate all itemsets of length lev and store in A
    Scan T and compute support $s(X)$ for each itemset X in A
    Sort A based on the support and remove all items which have the support less than the st
    Compute price ρ_i for each itemset X in A by combining the price values of the items in X
    Compute $DRank(X)$ for all X in A
    Sort all itemsets in A in descending order based on diverse net revenue (dnr) values of itemsets. i.e., $dnr(X) = DRank(X) \times Support(X) \times \rho(X)$
    Select λ itemsets from A in the descending order of dnr value of the itemsets and for each itemset X in A, insert $\langle X, s(X), \rho(X), DRank(X), dnr(X)\rangle$ in $CDRI$[lev][λ] in the descending order of dnr value of the itemsets in A
end
return $CDRI$

Algorithm 2: DRIP Placement Algorithm

Input: $CDRI$ index, Total Number of Slots (Ts)
Output: slots (Placement of itemsets in Slots)

```
Initialise slots[Ts] to NULL
CAS ← Ts /* CAS indicates the number of currently available slots */
while CAS > 0 do
    Starting from level=2, select one itemset with top dnr from each level of
      CDRI index and place it in A
    Choose the itemset X from A which has the highest revenue per slot
    Remove X from CDRI index
    If (CAS − |X|) < 0 then break
    Place X in slots[Ts]
    CAS = CAS - |X|
end
return slots
```

computed and the items with a frequency less than *support_threshold* are removed from A (Line 3). The top-λ items are picked from A now and placed in level 1 of $CDRI$ index (Line 4). For each level lev in 2 to max_level, all possible candidate itemsets of size lev are generated using itemsets placed in level 1 and level $lev - 1$ and placed in list A (Line 6). For each itemset in A, support (also called frequency of sales of the itemset) is found from T and all the itemsets with support less than *support_threshold* are removed from A (Line 8). Now, price for each itemset in A is calculated as sum of prices of items it has (Line 9) and *DRank* is calculated for each itemset in A using *Concept Hierarchy* (Line 10) with the method discussed earlier. All the itemsets in A are sorted based on *diverse revenue* ($d_i \times s_i \times \rho_i$) (Line 11). From this sorted list, top-λ itemsets are picked and the nodes are placed in $CDRI$ index (Line 12). Once this process is done for each level, the formed $CDRI$ index is returned.

2. Itemset Placement Scheme in DRIP

Given the number of slots, we present the itemset placement scheme for DRIP approach. It can be noted that $CDRI$ index contains itemsets of different sizes. We only consider itemsets with the size greater than 1. Given itemsets of different sizes and net revenue values, we first pick the itemset which maximizes revenue contribution per slot. The notion of *Revenue per slot* is defined as follows.

Revenue per Slot: Consider an itemset X of size k which consumes k slots. Revenue per slot refers to the per slot revenue contribution of X itemset.

$$net\ revenue\ per\ slot\ for\ X = \frac{price\ of\ X \times support\ of\ X}{size\ of\ X} \tag{1}$$

The placement approach keeps picking the itemset with maximum *revenue per slot* and placing it in the available slots.

Algorithm 2 depicts the steps for proposed DRIP scheme. The algorithm stops when CAS becomes 0. The CAS is initialized as number of slots to fill

(Line 2). While there are slots left to fill, get the top node of each level of the $CDRI$ index and place it in a temporary list A (Line 4). Now, the best node (node with highest *per slot revenue*) is picked from A and also removed from $CDRI$ as it is chosen to be placed (Line 5, 6). Then, append the itemset of this node in the slots and reduce the number of slots currently available as we have placed an itemset (Line 8,9). When the loop ends, the itemsets have been placed in the *slots* and the algorithm returns *slots* (Line 11).

3. Illustrative Example of DRIP Scheme

Now, we will explain the proposed DRIP scheme in the following with an example. The Figure 4 shows the whole approach.

We consider a *transactional dataset* which has 10 transactions of 8 items. The hierarchical relation between the items and categories as *Concept hierarchy*. Total number of slots is 25. Default value of parameters are provided in table 1. We explain the placement of itemsets in the given slots using DRIP Scheme.

The first step is to build a $CDRI$ index for the given data using Algorithm 1. The generated $CDRI$ can be observed in the figure. After the $CDRI$ index is generated we proceed with Algorithm 2 for placement. First we compare revenue per slot of top-nodes from each level of $CDRI$ i.e [2.23, 0.57, 0.21]. Since the top revenue for level 2 (index 1 here) is highest, we place the itemset from level 2 in slots and remove is from $CDRI$ index. The updated top per slot revenues for comparison becomes, $[1.18, 0.57, 0.21]$ and we pick node from level 2 again as it has highest per slot revenue. This process is continues till all slots are filled. The step wise placement is shown in Step 2 in the figure.

4.3 Hybrid Itemset Placement Approach (HIP)

The DRIP approach comprises of two main components: 1. *Identifying itemsets* and 2. *Itemset placement scheme*. We briefly explain these components.

1. Identifying itemsets

Given transactional database over a set of items, concept hierarchy over a set of items, price details of items, we build both revenue index (RI) index and $CDRI$ index. The process to build $CDRI$ index is explained in the preceding section. The process to build RI index is similar to the process of building $CDRI$ index, except, the itemsets of indexed based on net revenue (as in kUI index, refer section 2.3).

2. Itemset placement scheme

The HIP scheme uses a combination of RI and $CDRI$ to get itemsets for placement. During placement, it maintains vectors to top pointers and their *revenue_per_slot* for both RI and $CDRI$. We introduce a variable called *RD_ratio*, which represent the ratio of the portion of slots to be filled from RI and $CDRI$ index. For example, if *RD_ratio* is 0.3, 30% of given slots will be filled by itemsets from RI and 70% of given slots will be filled by itemsets from $CDRI$.

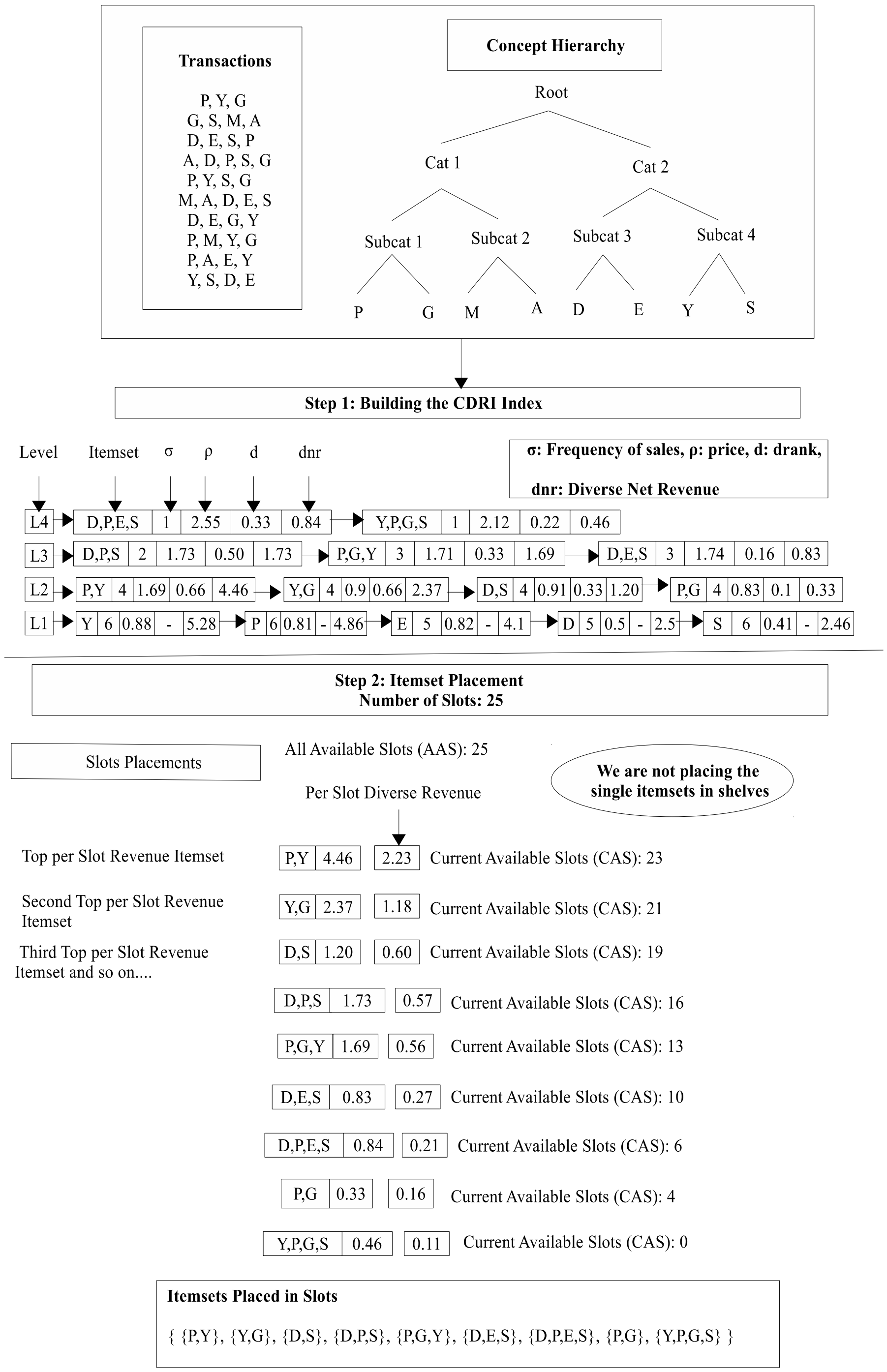


Fig. 4: Illustrative Example of DRIP Scheme

5 Performance Evaluation

In this section, we report the performance evaluation. The experiments were conducted on a Core i7-3537U CPU @ 2.00GHz x 4 PC with 8GB RAM running Ubuntu 18.04. We implemented the approaches in Python 3.7. The experiments are conducted on a real dataset, namely the *Instacart Market Basket Analysis* dataset [1]. The dataset is anonymous and contains over 3 million grocery orders from about 2,00,000 Instacart users. The dataset has 49,688 items and 1,31,208 transactions. The average transaction length is 10.55.

For conducting experiments, we require the hierarchy of the items in the dataset and the price value for each item. Regarding concept hierarchy, notably, the dataset has its own concept hierarchy in which the items are organized based on the department and the category information of the items. For all the experiments, we use this concept hierarchy. Regarding utility (price) values of the items, the dataset does not provide price related information. We generate the price for each item of the dataset randomly as follows. We consider a normalized price range of [0,0, 1.0] and divide it into the following six price brackets: [(0.01, 0.16), (0.17, 0.33), (0.34, 0.50), (0.51, 0.67), (0.67, 0.83), (0.84, 1.0)]. For each item in the dataset, we randomly select one of the afore-mentioned price brackets and then assign a random price within the range of that price bracket to the item.

Table 1: Parameters of Performance Evaluation

Parameters	***Default***	***Variations***
Total number of slots (10^3) (Ts)	10	(1, 2, 4, 8, 12)
RD_ratio	0.3	0.0, 0.2, 0.4, 0.6, 0.8, 1.0

Table 1 summarizes the parameters used in the evaluation. The *Total number of slots (Ts)* refers to the number of slots allocated. In the HIP approach, we fill a portion of slots from the index built for the RIP approach and the remaining portion from the index built for the proposed DRIP approach. To conduct experiments on HIP, we employ *RD_ratio*, which is the ratio of number of slots of the shelf to be filled from the index of the RIP approach and the number of slots of shelf to be filled from the index of the DRIP approach.

The performance metrics are *Total revenue* (TR) and *DRank*. *Total revenue* (TR) is the total retailer revenue for the test set. To calculate TR, for each transaction in the *test* set, we find the maximal set of items which are placed in the shelf, and add the revenue of that itemset to TR. Similarly, the *DRank* is the mean *DRank* of all the placed itemsets purchased by customers.

The experiments are performed on four approaches. They are Revenue based Itemset Placement (RIP), Diverse Revenue based Itemset Placement (DRIP), Hybrid Itemset Placement (HIP) and Diversity based Itemset Placement (DIP) approaches.

The *RIP* approach places high revenue itemsets in the given slots. The itemsets for RIP are picked from *RI* index. The proposed *DRIP* approach places itemsets in $CDRI$ index. The *HIP* approach places a portion of high net revenue itemsets from the RI index and the remaining portion of itemsets from $CDRI$ index. The *DIP* approach places only high *DRank* itemsets. For this we build index by considering *DRank* value of itemsets, which we call DI index. For each of RI, $CDRI$, DI indexes, we fix the the number of levels as 4 and the number of itemsets in each level as 5000.

For conducting the experiments, the dataset is divided into *training* set and *test* set in the ratio of 90:10. The index for the respective approaches such as RIP, DRIP and DIP approaches is built from the *training* set. We follow the placement approach, which was presented as a part of DRIP approach (Refer Section 4.2). We evaluate the performance considering the transactions in the *test* set in the following manner. We iterate over the transactions in the *test* set and check whether a subset (a set) of items of each transaction belongs to the itemsets placed in the given slots. We include the revenue of the corresponding subset towards TR. This helps in simulating a real life retail scenario where retailers place items on shelves and observe revenue only if the customer buys one of those items.

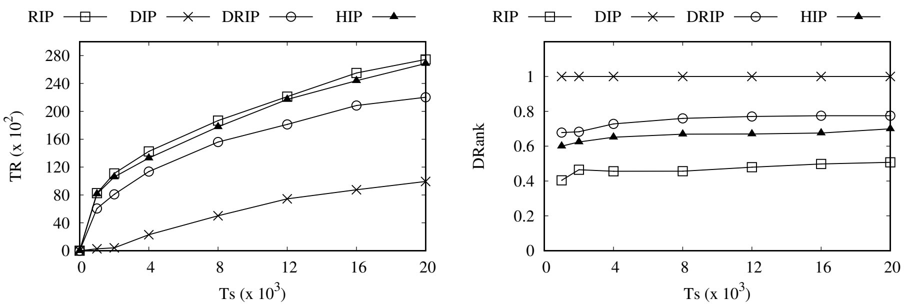
5.1 Effect of Variations in Number of Slots on Total Revenue

Figure 5a shows the variation of total revenue of RIP, DRIP, HIP and DIP. It can be observed that the revenue is increasing for all approaches with the number of slots. This is due to the fact that as the number of slots increases, more products will be available for purchase. As a result, increased number of customers can find the products in the retail store. Among the approaches, it can be observed that RIP's total revenue dominates other approaches. This is due to the fact that only high net revenue itemsets are placed in the RIP approach.

Also, as expected, it can be observed that the performance of DIP is low among four approaches, because DIP places only high *DRank* itemsets by ignoring the net revenue value of the itemsets. The total revenue of DIP is not zero because, it also places some itemsets with high net revenue. It can be observed that the performance of DRIP is less than RIP, but significantly more than DIP. This is due to the fact that DRIP places the itemsets with both high net revenue as well as high diversity. The revenue of DRIP is less than RIP because, high net revenue itemsets with low *DRank* values are not placed in the slots. This figure also shows the performance of HIP which places 30% of itemsets with top net revenue value from RIP and 70% of itemsets with both top net revenue as well as *DRank* values from DRIP. As expected, as we are also placing top net revenue itemsets, the total revenue of HIP is close to RIP.

5.2 Effect of Variations in Number of Slots on DRank

Figure 5b shows the variation of *DRank* values of RIP, DRIP, HIP and DIP. It can be observed that except for DIP, the average *DRank* value is increasing initially and becomes stable for RIP, DRIP and HIP approaches. For DIP, it maintains a constant value with the number of slots. Among the approaches, as



(a) Total Revenue (b) DRank

Fig. 5: Effect of Variations in Number of Slots

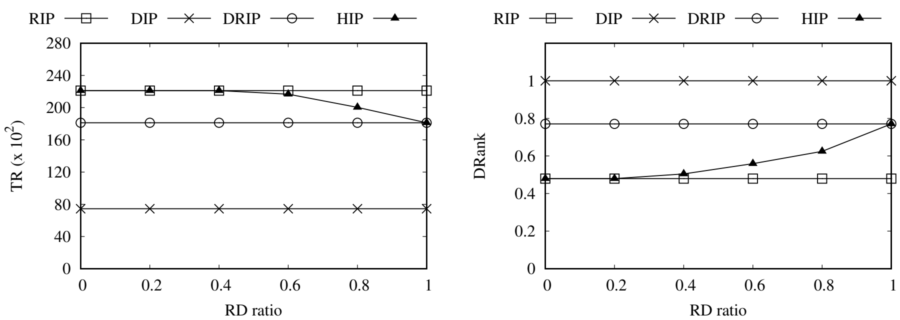
expected, it can be observed that the average *DRank* value is maximum for DIP. Also, DIP maintains the stable *DRank* value with the number of slots. This is due to the fact that as the number of slots increases, DIP places increased number of high *DRank* itemsets and the average *DRank* does not change significantly.

Also, as expected, it can be observed that the *DRank* performance of RIP is low among four approaches, because RIP places only high net revenue itemsets by ignoring the *DRank* value of the itemsets. The *DRank* value of RIP is not zero because, it also places some itemsets with high *DRank* value. Regarding the performance of the proposed DRIP approach, it can be observed that its *DRank* performance is less than DIP, but significantly more than RIP. This is due to the fact that DRIP contains the itemsets with both high *DRank* value and high net revenue value. The *DRank* value of DRIP is less than DIP because, high *DRank* itemsets with low net revenue values are not placed in the slots. This figure also shows the performance of HIP which places 30% of itemsets with top net revenue values from RIP and 70% of itemsets with both top net revenue as well as high *DRank* values from DRIP. Since, we exclude itemsets with top *DRank* values, the *DRank* value of HIP is less than DRIP.

From above results, it can be observed that DRIP gives high net revenue as well as high *DRank* values, its net revenue is less than RIP even though its *DRank* value is significantly higher than RIP. So, we can conclude that the HIP approach facilitates the placement of itemsets such that the total revenue can match with RIP, but trading the *DRank* value as compared to DRIP.

5.3 Effect of Variations in *RD_ratio*

For RIP, DIP, DRIP and HIP, Figure 6a shows the variations of total revenue and Figure 6b shows the variations of *DRank* value as we vary *RD_ratio*. In this experiment, we fix the number of slots at 12000. Notably, the *RD_ratio* does not affect RIP, DRIP and DIP due to the fact that the number of slots are fixed. *Figures 6a* shows the results of total revenue as we vary *RD_ratio*. It can be observed that HIP exhibits similar performance as that of RIP at *RD_ratio*= 0, and it is similar to DRIP at *RD_ratio*=1. It can be observed that as we vary *RD_ratio* from 0 to 0.4, the total revenue of HIP is equal to RIP. It indicates that the top-revenue itemsets are giving the high revenue performance. Gradually, the performance of HIP starts declining from *RD_ratio*=0.5 and it becomes equal to DRIP at *RD_ratio*=1. This indicates that after 0.4, the high *DRank* itemsets are added to the slots and hence, the performance of HIP becomes equal to DRIP.



(a) *RD_ratio* VS Total Revenue (b) *RD_ratio* VS DRank

Fig. 6: Effect of Variations in *RD_ratio*

Figure 6b shows variations in *DRank* values as we vary *RD_ratio*. The behaviour HIP is similar to RIP at *RD_ratio* = 0 and it is similar to DRIP at *RD_ratio* = 1. It can be observed that as we vary *RD_ratio* from 0 to 0.3, the *DRank* value of HIP is equal to RIP. Gradually, the *DRank* performance of HIP starts increasing from *RD_ratio*=0.3 and it has reached to DRIP at *RD_ratio*=1. This indicates that after *RD_ratio* > 0.3, the high diversity itemsets are added to the slots and hence, the performance of HIP becomes equal to DRIP.

From *Figure 6a and Figure 6b*, we can conclude that HIP gives flexibility to trade total revenue versus *DRank*. HIP provides the option to realize the total revenue as that of RIP by increasing *DRank* to a reasonable extent. Moreover, HIP also provides the option to improve *DRank* significantly as that of DRIP, subject to trading of revenue.

6 Conclusion

Placement of items in a given retail store impacts the revenue of the retailer. Moreover, the placement of diverse items increases customer interest. In this paper, we have addressed the issue of itemset placement in a given retail store to improve diversity without compromising the revenue. In particular, we have proposed two itemset placement schemes, which take into account the top-revenue itemsets in conjunction with concept hierarchy based diversity. Our performance evaluation on a real dataset demonstrates the overall effectiveness of our proposed schemes in terms of both diversity as well as revenue. In the near future, we plan to investigate the notion of correlated patterns towards further improving itemset placement in retail stores.

References

1. Instacart market basket analysis – kaggle. https://www.kaggle.com/c/instacart-market-basket-analysis/data (2017), [Online; accessed 2020-6-30]
2. Bentalha, B., Hmioui, A., Alla, L.: The digitalization of the supply chain management of service companies: A prospective approach. In: Proc. of the Conference on Smart City Applications (CSCA). ACM (2019)
3. Bradley, K., Smyth, B.: Improving recommendation diversity. In: Proc. of the Irish Conference Artificial on Intelligence and Cognitive Science (ICAICS). pp. 75–84. Springer (2001)
4. Brijs, T., Swinnen, G., Vanhoof, K., Wets, G.: Using association rules for product assortment decisions: A case study. In: Proc. of the Conference on Knowledge Discovery and Data Mining (CIKM). pp. 254–260. ACM (1999)
5. Brown, W., Tucker, W.: The marketing center: Vanishing shelf space. Atlanta Economic Review pp. 9–13 (1961)
6. Chaudhary, P., Mondal, A., Reddy, P.K.: A flexible and efficient indexing scheme for placement of top-utility itemsets for different slot sizes. In: Proc. of the Conference on Big Data Analytics (BDA). pp. 257–277. Springer (2017)
7. Chaudhary, P., Mondal, A., Reddy, P.K.: A diversification-aware itemset placement framework for long-term sustainability of retail businesses. In: Proc. of the Database and Expert Systems Applications (DEXA). pp. 103–118. Springer (2018)
8. Chaudhary, P., Mondal, A., Reddy, P.K.: An efficient premiumness and utility-based itemset placement scheme for retail stores. In: Proc. of the Database and Expert Systems Applications (DEXA). pp. 287–303. Springer (2019)
9. Chaudhary, P., Mondal, A., Reddy, P.K.: An improved scheme for determining top-revenue itemsets for placement in retail businesses. International Journal of Data Science and Analytics **10**(4), 359—-375 (2020)
10. Chen, M., Lin, C.: A data mining approach to product assortment and shelf space allocation. Expert Systems with Applications **32**, 976–986 (2007)
11. Chen, Y.L., Chen, J.M., Tung, C.W.: A data mining approach for retail knowledge discovery with consideration of the effect of shelf-space adjacency on sales. Decision Support Systems **42**(3), 1503–1520 (2006)
12. Coyle, M., Smyth, B.: On the importance of being diverse: Analysing similarity and diversity in web search. In: Proc. of the International Federation for Information Processing (IFIP). vol. 163, pp. 341–350. Springer (2004)
13. Etgar, M., Rachman-Moore, D.: Market and product diversification: The evidence from retailing. Journal of Marketing Channels **17**, 119–135 (2010)
14. Flamand, T., Ghoniem, A., Haouari, M., Maddah, B.: Integrated assortment planning and store-wide shelf space allocation: An optimization-based approach. Omega **81**(C), 134–149 (2018)
15. Han, J., Fu, Y.: Attribute-oriented induction in data mining. In: Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.) Advances in Knowledge Discovery and Data Mining, pp. 399–421. AAAI (1996)

16. Han, J., Fu, Y.: Mining multiple-level association rules in large databases. Transactions on Knowledge and Data Engineering **11**(5), 798–805 (1999)
17. Hansen, P., Heinsbroek, H.: Product selection and space allocation in supermarkets. European Journal of Operational Research **3**(6), 474–484 (1979)
18. Hart, C.: The retail accordion and assortment strategies: an exploratory study. The International Review of Retail, Distribution and Consumer Research **9**, 111–126 (1999)
19. Hurley, N., Zhang, M.: Novelty and diversity in top-n recommendation – analysis and evaluation. Transactions on Internet Technology **10**(4), 14:1–14:30 (2011)
20. Kumaraswamy, M., Reddy, P.K.: Improving diversity performance of association rule based recommender systems. In: Proc. of the Database and Expert Systems Applications (DEXA). pp. 499–508. Springer (2015)
21. Kumaraswamy, M., Reddy, P.K.: A model of concept hierarchy-based diverse patterns with applications to recommender system. International Journal of Data Science and Analytics **10**(2), 177–191 (2020)
22. Kumaraswamy, M., Reddy, P.K., Srivastava, S.: Extracting diverse patterns with unbalanced concept hierarchy. In: Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD). pp. 15–27. Springer (2014)
23. Liu, M., Qu, J.: Mining high utility itemsets without candidate generation. In: Proc. of the Conference on Information and Knowledge Management (CIKM). pp. 55–64. ACM (2012)
24. Lu, Y.: Concept Hierarchy in Data Mining: Specification, Generation and Implementation. Master's thesis, School of Computer Science, Simon Fraser University, Canada (1997)
25. M., Winstone, L., Gong, W., Cheng, S., Han, J.: Generalization and decision tree induction: Efficient classification in data mining. In: Proc. of the Workshop on Research Issues in Data Engineering, RIDE: High Performance Database Management for Large-Scale Applications. pp. 111–120. IEEE (1997)
26. Michalski, R., Stepp, R.: Automated construction of classifications: conceptual clustering versus numerical taxonomy. Transactions on Pattern Analysis Machine Intelligence **5**(4), 396–410 (1983)
27. Srivastava, S., Kiran, R.U., Reddy, P.K.: Discovering diverse-frequent patterns in transactional databases. In: Proc. of the Conference on Data Science and Management of Data (COMAD). pp. 69–78. CSI (2011)
28. Swamy, M.K., Reddy, P.K., Bhalla, S.: Association rule based approach to improve diversity of query recommendations. In: Proc. of the Database and Expert Systems Applications (DEXA). pp. 340–350. Springer (2017)
29. Tseng, V.S., Wu, C., Fournier-Viger, P., Yu, P.S.: Efficient algorithms for mining the concise and lossless representation of high utility itemsets. Transactions on Knowledge and Data Engineering (KDD) **27**(3), 726–739 (2015)
30. Tseng, V.S., Wu, C.W., Shie, B.E., Yu, P.S.: Up-growth: An efficient algorithm for high utility itemset mining. In: Proc. of the Conference on Knowledge Discovery and Data Mining (KDD). pp. 253–262. ACM (2010)
31. Vargas, S., Castells, P.: Rank and relevance in novelty and diversity metrics for recommender systems. In: Proc. of the Recommender Systems Conference (RecSys). pp. 109–116. ACM (2011)
32. Wang, M., Hsu, P., Lin, K.C., Chen, S.: Clustering transactions with an unbalanced hierarchical product structure. In: Proc. of the Conference on Data Warehousing and Knowledge Discovery. pp. 251–361. DaWaK'07, Springer-Verlag (2007)
33. Wigley, S.M.: A conceptual model of diversification in apparel retailing: the case of next plc. Journal of the Textile Institute **102**, 917–934 (2011)

34. Yang, M.H.: An efficient algorithm to allocate shelf space. European Journal of Operational Research **131**(1), 107–118 (2001)
35. Yang, M.H., Chen, W.C.: A study on shelf space allocation and management. International Journal of Production Economics **60**(1), 309–317 (1999)
36. Zheng, W., Fang, H., Yao, C.: Exploiting concept hierarchy for result diversification. In: Proc. of the Conference on Information and Knowledge Management (CIKM). pp. 1844–1848. ACM (2012)
37. Ziegler, C.N., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: Proc. of the World Wide Web Conference (WWW). pp. 22–32. ACM (2005)