A Retail Itemset Placement Framework Based on Premiumness of Slots and Utility Mining

ANIRBAN MONDAL\textsuperscript{1}, SAMANT SAURABH\textsuperscript{2}, PARUL CHAUDHARY\textsuperscript{3}, RAGHAV MITTAL\textsuperscript{1}, AND POLEPALLI KRISHNA REDDY\textsuperscript{4}

\textsuperscript{1}Computer Science Department, Ashoka University, Sonipat 131029, India
\textsuperscript{2}Computer Science Department, Indian Institute of Management Bodh Gaya, Bodh Gaya 824234, India
\textsuperscript{3}Computer Science Department, Shiv Nadar University, Greater Noida 201314, India
\textsuperscript{4}Computer Science Department, International Institute of Information Technology, Hyderabad 500032, India

Corresponding author: Parul Chaudhary (pc230@snu.edu.in)

ABSTRACT Retailer revenue is significantly impacted by item placement in retail stores. Notably, placement of items in the premium slots (i.e., slots with increased visibility/accessibility) improves the probability of sale w.r.t. item placement in non-premium slots. Moreover, customers often tend to buy sets of items (i.e., itemsets) instead of individual purchases. In this paper, we address the problem of maximizing retailer revenue by determining the placement of itemsets in different types of slots with varied premiumness. Our key contributions are as follows. First, we introduce the notion of premiumness of retail slots and discuss the issue of itemset placement in slots with varied premiumness. Second, we propose two efficient schemes, namely Premiumness and Revenue-based Itemset Placement (PRIP) and Premiumness and Average Revenue-based Itemset Placement (PARIP), for placing itemsets with varying revenue in slots with varied premiumness. Third, we perform a detailed performance analysis using both real and synthetic datasets to showcase the effectiveness of our proposed schemes. We also perform a comprehensive mathematical analysis of our proposed schemes w.r.t. the complexity analysis.

INDEX TERMS Utility mining, product placement, itemset placement, retail management, revenue, slot premiumness.

I. INTRODUCTION

Retail stores enable goods to reach consumers. A retail store typically contains racks/shelves with slots, in which products/items are placed. Key stakeholders of a retail store include the retailer and the customers. Customers purchase the items, and the goal of the retailer is to improve the sales of its items. It has been observed that the sale of items, and consequently, retailer revenue are significantly impacted by the method followed for item placement on the retail store shelves [1]–[5]. In this scenario, if the decisions concerning the placement of products are carried out in an ad hoc manner by means of rudimentary methods, the retailer would miss the opportunity to improve their revenue. The problem assumes even more significance in case of medium-to-large retail stores, some of which even have floor space exceeding a million square feet e.g., New South China Mall (Dongguang, China), Siam Paragon Mall (Bangkok, Thailand), West Edmonton Mall (Edmonton, Alberta, Canada) and Berjaya Times Square Mall (Kuala Lumpur, Malaysia) [6].

In the literature, efforts are being made towards improved item placement by exploiting the past purchase patterns and applying data analysis techniques such as utility mining approaches. Utility mining frameworks aim to determine itemsets with high utility from databases containing user purchase transactions. Existing utility mining techniques identify minimal high-utility itemsets (HUIs) [7], examine effective ways of representing HUIs [8], use special data structures (e.g., the utility-list [9] and the UP-Tree [10]) for reducing the cost associated with the generation of candidate itemsets, and investigate pruning heuristics [11], [12]. Several efforts are being made towards strategic placement of itemsets in retail store slots using utility mining approaches [13]–[16]. In particular, these works indicate that it is possible to improve retailer revenue by exploiting the knowledge of high-utility itemsets for retail itemset placement.

Incidentally, items are typically placed in the slots of the retail store. Notably, not all slots in the retail store are created equal i.e., they may vary with regard to their degree of
premiumness. In this paper, we have proposed an improved utility mining based framework for placing items in the retail store by categorizing the slots based on their premiumness. A slot’s premiumness is indicative of its visibility and/or physical accessibility to the shopper. The retail industry holds a firm position on the fact that the likelihood of sale is improved for an item it with an increased degree of the premiumness of the slot in which item it has been placed [17], [18]. In consonance, this paper assumes that an item’s likelihood of sale increases with an increase in slot premiumness. Instances of high-premiumness slots could include the “impulse buy” slots [19] in the vicinity of the check-out counters, or premium slots at the near-eye or near-shoulder level of the customer. In contrast, slots with a low value of premiumness could be located far away from the sight of the customer.

Notably, in this work, we do not compute the premiumness values for various slots. Instead, we use the premiumness values of the slots as an input towards addressing the issue of itemset placement in retail stores. As such, the premiumness of a slot by itself cannot be directly computed because it is typically a subjective decision taken by the retailer, as evidenced by existing works in the literature on retail product placement [17]–[19]. Hence, in consonance with these existing works, we consider the values for slot-premiumness as input provided to our research problem by the retailer. In real-world scenarios, this is essentially application-dependent and the retailer would determine the number of slots corresponding to each different level of slot premiumness depending upon factors such as the location of the slots in the retail store in terms of visibility/accessibility to customers, how quickly purchases occur when items are placed in a certain type of slot and so on.

Existing pattern mining approaches and utility mining approaches do not address the issue of retail itemset placement in slots with varied premiumness. A naive approach to addressing the afore-mentioned problem would be to place the items with higher price in the high-premiumness slots, the mid-priced items in the mid-premiumness slots and so on. However, given the fact that users typically prefer purchase of the high-revenue itemsets with factorial time complexity. By doing this we bring down the time-complexity to polynomial time. Hence, using the kUI index, our proposed heuristic approaches are able to solve this problem in polynomial time. Notably, the key difference between PRIP and PARIP is that while PRIP examines the net revenue of itemsets for slot-premiumness aware itemset placement, PARIP examines the average net revenue of itemsets for slot-premiumness aware itemset placement. We defer the explanations of net revenue and average net revenue to Section 4 of this paper.

In this paper, we have used revenue as a measure of utility. We shall use revenue and utility interchangeably throughout the paper.

Our research contributions are summarized below:

1) We introduce the notion of premiumness of retail slots and discuss the issue of itemset placement in slots with varied premiumness.

2) We propose two efficient schemes, namely PRIP and PARIP, for placing itemsets having varied revenue in slots with varied premiumness.

3) We perform a detailed performance analysis using both real and synthetic datasets to showcase the effectiveness of our proposed schemes. We also perform a comprehensive mathematical analysis of our proposed schemes w.r.t. the complexity analysis.

We have made a preliminary effort in [15]. We have prepared this journal version by extending our work in [15] in a comprehensive manner. This paper significantly extends the work in [15] with the following additions. First, we have added a new itemset placement scheme (namely, PARIP) to eliminate the bias towards larger-sized itemsets, thereby further enhancing the revenue of the retailer. Second, in contrast with our preliminary effort in [15], we have carried out mathematical analysis on the relationship between skewness and net revenue for the PRIP scheme (see Section 5C), and optimality for the proposed PARIP scheme (see Section 6C). Third, in [15], we have compared the PRIP placement scheme with the state-of-the-art minFHM approach. In this paper, we have performed more extensive experiments to extend the performance evaluation by comparing the improved PARIP scheme with the PRIP, FHM algorithm and the HUI-miner algorithms. Moreover, we have performed our experiments by dividing the dataset into training and test set to demonstrate retailer revenue improvement. In essence, this journal version is a considerable extension of our preliminary work in [15].

Our paper is structured as follows. Section II explores existing works. Section III describes and analyzes the background of the kUI index, which we use for identifying itemsets for placement purposes. Section IV discusses the context of the problem. Sections V and VI discuss our proposed PRIP and PARIP placement schemes. In Section VII, we report the performance results. In Section VIII, we discuss the limitations of our proposed approaches. Finally, we conclude and provide directions for our future work in Section IX.
II. RELATED WORK

The goal of association rule mining approaches [20]–[22] is to determine frequent itemsets. The work in [23] focused on identifying association rules for products in supermarkets with the goal of performing product placement based on these rules. The objective is to increase the sales of products for improving the revenue. In particular, it used sales data of a supermarket from the Vancouver Island University website. Moreover, given that scanning the database multiple times to identify frequent itemsets is cumbersome and time-consuming, the work in [24] proposed a method, which needs to perform only a single scan of the database for improving efficiency. In particular, it used a technique, designated as Fault Tolerance, in conjunction with a tree-like structure for handling noisy data. The work in [25] investigated the use of association rules in financial management. In particular, it performed the analysis of consumer baskets to enable more targeted marketing e.g., for retail and banking customers. Moreover, it also aimed at detecting financial fraud by exploiting association rules. A comprehensive review of algorithms for frequent pattern mining can be found in [26].

Interestingly, several efforts have also been made towards using association rules and frequent itemset mining in the field of healthcare and medicine [26]–[28]. The work in [26] used association rule mining for facilitating medical professionals towards classifying diseases. In particular, it examines the factors, which contribute to schistosomiasis (a fatal pollution-related disease) by using data that had been collected from Hubei in China. Additionally, the work in [27] proposed a machine learning framework study for extracting the knowledge of the factors associated with the disease of malignant mesothelioma (a rare type of cancer). In particular, it used association rule mining-based algorithms and feature selection techniques for extracting significant features. Furthermore, the work in [28] aimed at studying the risk factors associated with malignant mesothelioma, which is an important type of lung cancer. In particular, it used the data of both mesothelioma patients as well as healthy patients. The class imbalance problem due to the number of mesothelioma patients being much lower than that of healthy patients was addressed by using the synthetic minority oversampling technique.

Observe that the issue of item utility (e.g., item price) has not been considered in efforts on association rule mining. Consequently, several efforts have focussed on identifying itemsets with high utility [7]–[12], [29]–[31].

Algorithms for mining high-utility itemsets (HUIs), such as the HUG-Miner and GHUI-Miner algorithms [8], perform pruning on the candidate itemsets for efficiently finding HUIs. The EFIM algorithm [11] and the EFIM-Closed algorithm [12] use upper-bounds on utility for pruning itemsets. An algorithm for mining closed HUIs is the CHUI-Miner algorithm [30], which calculates itemset utilities, while avoiding candidate generation. By addressing the fact that utilities of itemsets can change temporally, HUIs are mined by the LHUI-Miner algorithm [32]. Business-oriented objectives have been considered in [31] towards mining HUIs. Moreover, planning for retail assortment has been investigated in [33].

Some of the utility mining approaches use data structures, which are specialized, for mining HUIs. For example, the Utility Pattern Tree (UP-Tree) has been proposed in [10] to store information about HUIs to enable the Utility Pattern Growth (UP-Growth) algorithm towards extracting HUIs. The IUData List was discussed in [34] to enable the DMHUPS algorithm to discover several patterns having high utility concurrently.

The proposal in [9] discusses an algorithm, designated as the HUI-Miner algorithm, for determining high-utility itemsets. The HUI-Miner algorithm stores utility values and heuristic information about the itemsets in a specialized data structure called the utility-list. In particular, the use of this specialized utility-list data structure facilitates the HUI-Miner algorithm towards avoiding utility computations for a large number of candidate itemsets and also helps the algorithm in avoiding expensive candidate itemset generation.

The work in [35] discusses the FHM algorithm, which reduces the cost of mining high-utility itemsets based on an analysis of item co-occurrences. In particular, it uses a pruning mechanism, designated as EUCP (Estimated Utility Co-occurrence Pruning), for directly eliminating low-utility extensions and all of their transitive extensions without having to construct their respective utility-lists. Observe that FHM extracts candidate itemsets of different slot sizes, as long as the itemsets satisfy the minimum utility threshold criterion.

The work in [29] proposes the MinFHM algorithm, which reduces the cost of mining high-utility itemsets based on an analysis of item co-occurrences. In particular, it uses a pruning mechanism, designated as EUCP (Estimated Utility Co-occurrence Pruning), for directly eliminating low-utility extensions and all of their transitive extensions without having to construct their respective utility-lists. Observe that FHM extracts candidate itemsets of different slot sizes, as long as the itemsets satisfy the minimum utility threshold criterion.

Incremental utility mining has also received attention. An algorithm, which uses lists, has been proposed in [36] for incrementally mining HUIs. The HAUP-List data structure, which maintains patterns space-efficiently, has been used in the approach in [37] for mining HUIs for dynamic databases. The pre-large principle has been used in [38] and [39] for incrementally mining HUIs. A multi-core approach for mining HUIs incrementally has been proposed in [40].

Some of our previous works have also addressed retail itemset placement [13], [14], [16]. In [14] and [16], we addressed itemset placement when the sizes of items vary e.g., a bottle of Pepsi would be much less in size than an air-conditioner or a large-screen television set. In [13], we focussed primarily on diversifying the itemset placement with the intent of improving the retailer revenue sustainably.
in the long run. Note that these works also do not consider the fact that the premiumness of retail slots can vary.

Notably, existing techniques on utility mining and pattern mining do not address the issue of placing itemsets in retail slots having different premiumness. This restricts their applicability towards building decision-making systems for placing retail itemsets with the intent of strategically improving retailer revenue. Generating high-revenue itemsets incurs high computational cost. Hence, we make use of the kUI index, which we had previously proposed in [13], for efficiently generating high-revenue itemsets.

We shall now present the background about the kUI index. The kUI index comprises multiple levels, where every level pertains to a unique itemset size. Notably, the kUI index contains top-\( \lambda \) itemsets of size \( k \) at the \( k \)th level of the index. Every level \( k \) of the kUI index corresponds to a hash bucket, which further carries a pointer to a linked list containing top-\( \lambda \) itemsets of size \( k \). The linked list entries are structured in the format (itemset, \( \sigma \), \( \rho \), NR). Here, itemset refers to the specific itemset, \( \sigma \) denotes the frequency of sales of itemset and \( \rho \) denotes the total price of itemset. We multiply \( \sigma \) (frequency of sales) and \( \rho \) (price) of the itemset to obtain the net revenue (NR) of the itemset. We sort the itemsets in the linked list in decreasing order of their respective net revenue to quickly retrieve the top-\( \lambda \) itemsets of size \( k \). Observe that a query seeking the top-\( \lambda \) itemsets of any given size \( k \) would be able to speedily reach out to the \( k \)th hash bucket rather than browsing through all hash buckets of size \( k = \{1, 2, \ldots, k - 1 \} \).

We present the illustrative example for the kUI index in Figure 1. Observe how 1-itemsets belong to the first level of the kUI index, 2-itemsets belong to the second level, and so on. Further, observe that the sorting of the itemsets is performed based on NR.

Each linked list node of the kUI index stores the frequency (\( \sigma \)), price (\( \rho \)) and the net revenue (NR). If we consider each of them as requiring \( B \) bytes, the space required at the \( i \)th level of the kUI index would be \((i + 3)\lambda \times B \) bytes because \( \lambda \) nodes are present at each level. Hence, given a total of \( \max L \) levels in the kUI index, the total space required by the kUI index in bytes would be computed by Equation 1.

\[
space(kUI) = (1 + 3)\lambda \times B + (2 + 3)\lambda \times B + (3 + 3)\lambda \times B + \cdots + (\max L + 3)\lambda \times B
\]

\[
= (1 + 2 + 3 + \cdots + \max L)\lambda \times B + 3\lambda \times B
\]

\[
= \frac{\lambda \times B}{2} \times (\max L + 1)\lambda \times B + 3 \times \lambda \times \max L
\]

\[
= \lambda \times B + 3 \times \lambda \times \max L
\]

\[
(\text{bytes})
\]  

**III. BACKGROUND ABOUT THE kUI INDEX**

As discussed in Section 1, our proposed PRIP and PARIP itemset placement schemes examine high-revenue itemsets of varying sizes and the concept of varying slot-premiumness for improving the retailer revenue. Generating high-revenue itemsets incurs high computational cost. Hence, we make use of the kUI index, which we had previously proposed in [13], for efficiently generating high-revenue itemsets.

![Figure 1. Illustrative example of the kUI index.](image)

**TABLE 1. Information about items.**

| Item | A | B | C | D | E | F | G | H | I |
|------|---|---|---|---|---|---|---|---|---|
| Price (\( \rho \)) | 7 | 2 | 6 | 1 | 3 | 1 | 5 | 4 | 3 |

Observe how 1-itemsets belong to the first level of the kUI index, 2-itemsets belong to the second level, and so on. Further, observe that the sorting of the itemsets is performed based on NR.

Each linked list node of the kUI index stores the frequency (\( \sigma \)), price (\( \rho \)) and the net revenue (NR). If we consider each of them as requiring \( B \) bytes, the space required at the \( i \)th level of the kUI index would be \((i + 3)\lambda \times B \) bytes because \( \lambda \) nodes are present at each level. Hence, given a total of \( \max L \) levels in the kUI index, the total space required by the kUI index in bytes would be computed by Equation 1.

\[
space(kUI) = (1 + 3)\lambda \times B + (2 + 3)\lambda \times B + (3 + 3)\lambda \times B + \cdots + (\max L + 3)\lambda \times B
\]

\[
= (1 + 2 + 3 + \cdots + \max L)\lambda \times B + 3\lambda \times B
\]

\[
= \frac{\lambda \times B}{2} \times (\max L + 1)\lambda \times B + 3 \times \lambda \times \max L
\]

\[
(\text{bytes})
\]  

**Product Placement Considering High-Revenue Itemsets, Net Revenue and Premiumness level:** We should note that product placement taking high-revenue itemsets, net revenue and retail slot premiumness into account has factorial time complexity (which grows even faster than the exponential time complexity in the asymptotic case) if we use the brute-force method. Now, we provide a proof of this. We have \( m \) distinct items \( \{i_1, i_2, i_3, \ldots, i_m \} \). Moreover, we have \( N \) distinct levels of premiumness, \( n_1 \) slots for premiumness level 1, \( n_2 \) slots of premiumness level 2, \( n_3 \) slots of premiumness level 3, \( \ldots, n_N \) slots of premiumness level \( N \). Each of the \( m \) items need to be placed in these slots. We have \( n_1 + n_2 + n_3 + \cdots + n_N = n \). Let T denote our transaction matrix.

\[
T = \begin{pmatrix}
\rho_1 & \rho_2 & \rho_3 & \rho_4 & \cdots & \rho_{m-1} & \rho_m \\
1 & \rho_2 & \rho_3 & \rho_4 & \cdots & \rho_{m-1} & \rho_m \\
\rho_1 & \rho_2 & \rho_3 & \rho_4 & \cdots & \rho_{m-1} & \rho_m \\
\sigma_1 & \sigma_2 & \sigma_3 & \sigma_4 & \cdots & \sigma_{m-1} & \sigma_m \\
\end{pmatrix}
\]  

(2)

Here, in the transaction matrix shown in Equation 2, the first row represents the price of each item, the second row represents the items themselves and the last row represents the frequency of sales of each item. The \( p_j^i \)'s in the third row represent the expected probability of the items being sold in the slot in which they are placed. The expected value of the generated revenue for the given product placement is given by Equation 3.

\[
E(\text{NR}) = \rho_1 \sigma_1 + \rho_2 \sigma_2 + \rho_3 \sigma_3 + \cdots + \rho_m \sigma_m
\]  

(3)
Let us now assume that we want to take high-revenue item sets, their net revenue, and the slot premiumness into account. In order to solve our problem, we need to group m item sets, their net revenue, and the slot premiumness into Table 2.

**Example for computing average net revenue.**

| TID | Transaction | Frequency of Sales (σ) | Net Revenue (NR) | Average Net Revenue (ANR) |
|-----|-------------|------------------------|------------------|---------------------------|
| 1   | A,D         | 6                      | 48               | 48/2 = 24                |
| 2   | B,C,LF      | 3                      | 36               | 36/2 = 18                |
| 3   | A,C,G       | 2                      | 36               | 36/3 = 12                |
| 4   | A,B,C,G,H   | 1                      | 24               | 24/3 = 8                 |
| 5   | A,C,G,I     | 2                      | 42               | 42/2 = 21                |

**Table 2.**

ANR = \( \sigma z \sum_{j \in z} \rho_j / |z| \) (6)

For instance, the NR for itemset \{A,D\} is 48. Since the itemset \{A,D\} comprises 2 items, its average net revenue is 48/2 i.e., 24.

Recall our discussion on the concept of **premiumness** of retail slots in Section 1. Suppose there are S types of slots with different degrees of premiumness. This work considers the problem of improving retailer revenue through efficient retrieval and placement of high-revenue item sets in slots with varying premiumness, while ensuring that all items have been placed at least once in any of the slots.

We shall now elaborate on the concepts of revenue contribution and aggregate revenue contribution, which lay the foundation of our proposed framework. Subsequently, we shall elaborate on our proposed schemes.

**Revenue Contribution and Aggregate Revenue Contribution:** Intuitively, we can understand that all items may not necessarily contribute equally to the retailer’s revenue. However, a low-priced item it, when bought in association with other items (i.e., in the form of an itemset) by several customers, could significantly contribute to retailer revenue. Further, this work assumes that an itemset with a missing item it would not be purchased by customers, since the item’s absence in that itemset would typically discourage the sale of the itemset.

On this basis, we put forth the concept of aggregate revenue contribution (ARC) of any given item across the transactional database. Provided a collection of user purchase transactions, ARC essentially maps the degree of an item’s revenue contribution to the total revenue of the retailer.
In order to compute the ARC for an item, we presume that each of the items in any given user purchase transaction contributes equally towards the customer’s purchase decision. Understandably, quantifying the influence of each item on the degree of association in an itemset is difficult by deploying existing pattern/utility mining techniques. It is possible to explore such issues by employing causality mining techniques, which we intend to explore as part of our future work. In practice, it is improbable that the retailer has awareness of the user’s purchase motivation, i.e., for transaction \( \{X,Y,Z\} \), the retailer would typically not be aware whether \( X \) led to the sale of \( Y \) and \( Z \) or vice-versa.

For a given user purchase transaction \( T \), the *Purchase value* (PV) denotes the amount of currency paid by the customer for the purchase of all items in transaction \( T \). In order to compute the revenue contribution (RC) of each item in \( T \), we divide the purchase value (\( PV_T \)) for \( T \) by the number of items \( n_T \) contained in it, i.e., for each item in \( T \), \( RC_T = \frac{PV_T}{n_T} \). Next, in order to compute the *aggregated revenue contribution* (ARC) of every item, we add their RC values across all of the transactions in \( T \). For the \( k^{th} \) item \( i_k \), our framework computes ARC, given \( q \) transactions, as \( ARC_k = \sum_{j=1}^{q} RC_j \). Notably, if \( i_k \) is not contained in the \( j^{th} \) transaction, the value for \( RC_j \) becomes zero w.r.t. item \( i_k \).

The purchase values for five transactions have been provided in Figure 2. For example, the purchase value for transaction \( \{A, C, G, I\} \) is 42 and the revenue contribution for every item in the itemset is equal to \( \frac{42}{4} \) i.e., 10.5. Note how item A’s revenue contribution varies across different itemsets. Ultimately, we add the revenue contributions of item A to obtain the aggregate revenue contribution for item A as (30+10.5+12 = 52.5) i.e., 52.5.

**V. PREMIUMNESS AND REVENUE-BASED ITEMSET PLACEMENT (PRIP)**

This section discusses our proposed PRIP scheme.

**A. BASIC IDEA**

The fact that efficient item placement significantly impacts retailer revenue, has been well established [17]. Moreover, customers typically tend to buy itemsets instead of buying individual items. Furthermore, retail slots are usually of a varying degree of *premiumness*, and itemset placement in slots with higher premiumness increases the probability of sale of itemsets. Thereby, random assignment and placement of itemsets in highly premium slots could compromise the retailer revenue. For instance, placement of high-revenue itemsets in slots with a lower degree of premiumness or placement of low-revenue itemsets in highly premium slots could cause the retailer to lose out on a significant amount of revenue. We can assert that by carefully studying customer purchase patterns and following a strategic itemset placement framework, there is a significant opportunity to improve retailer revenue.

We propose an efficient framework for determining itemsets from a set of user transactions, and placing them in retail slots by mapping their revenue to the premiumness of different types of slots for improving retailer revenue. For the purpose of itemset placement, our approach retrieves potential 1-itemsets with regard to their ARC values. Notably, the concept of ARC is only valid for distinct items and not for a set of items. Thereby, we retrieve itemsets containing more than one item on the basis of their net revenue.

The scheme we propose works in the following manner. Initially, for the placement of 1-itemsets, we extract itemsets with a high value of ARC for placement in slots with high premiumness. Similarly, for the placement for items with medium ARC and low ARC in slots with mid and low premiumness respectively. Subsequently, to place itemsets containing more than one item in the rest of the slots, we retrieve itemsets with varying sizes (starting with itemsets containing 2 items) from the kUI index [13] (presented in Section III) and place them in the slots with varying premiumness, with regard to the itemset’s net revenue.

| Transactions | PV  | RC    |
|--------------|-----|-------|
| A, E         | 60  | 60/2 = 30 |
| A, C, G, I   | 42  | 42/4 = 10.5 |
| A, C, G      | 36  | 36/3 = 12    |
| B, C         | 24  | 24/2 = 12    |
| D, F, G      | 28  | 28/3 = 9.33  |

**FIGURE 2. Illustrative example for the computation of RC and ARC.**

| Symbol | Meaning                                      |
|--------|----------------------------------------------|
| \( m \) | Total number of unique items                  |
| \( n \) | Total number of available slots for product placement |
| \( numF \) | Total number of transactions               |
| \( S \) | Number of premiumness levels                |
| \( \lambda \) | Total number of itemsets in the kUI index at each level |
| \( maxL \) | Maximum number of levels in the kUI index     |
| \( \sigma \) | Frequency of sales                           |
| \( \rho \) | Revenue of item                              |
Recall that we exploit the kUI index to retrieve itemsets of varying size. The kUI index contains itemsets with varied size, frequency of sales, price, and correspondingly varied net revenue, as discussed earlier in Section 3. Further, recall how kUI sorts and stores itemsets in descending order of their respective net revenue. Notably, placement of itemsets with varying sizes in retail slots enhances the probability of sale as it addresses the issue of different needs of distinct customer segments and enables one-stop shopping.

### B. ALGORITHM FOR PRIP

We present our proposed PRIP scheme in Algorithm 1. As inputs, PRIP uses a database D comprising user purchase transactions, S types of slots (based on premiumness) and n_i slots of slot-type i, and the kUI index. The symbols that we have used in our algorithms have been explained in Table 3.

#### Algorithm 1 PRIP Scheme

**Input:** (i) I: set of m distinct items (ii) D: a transactional database on I (iii) S types of slots of different premiumness level and there are s_i slots belonging to the i^{th} slot type (iv) kUI Index

**Output:** Items/itemsets placement in slots of different premiumness level

1. Initialize $ARC[] ← 0$; level lv=1; s\_variety = 1; $CAS[i] ← s_i$; $ListItem[] ← 0$
2. Scan $T$ and compute $ARC[m]$ for each item $m$ in $T$ and store it in $ListItem[]$
3. Sort items in descending order of $ARC[m]$ into $ListItem[]$
4. for (i= 1 to $S$) do
5. for each item $i$ in $ListItem[]$ do
6. while $CAS[i] > 0$ do
7. assign item $i$ from $ListItem[]$ to slot type $s\_variety[i]$
8. $CAS[i] = CAS[i] - |it|
9. Remove item $i$ from $ListItem[]$
10. end while
11. end for
12. end for
13. for each slot type(i= 1 to $S$) do
14. for each level (lv= 2 to maxL) do
15. for (j=1 to $\lambda$) do
16. while ($CAS[i] \neq 0$) do
17. extract itemset $X$ at kUI[lv][j]
18. place items of itemset $X$ into slot type $s\_variety[i]$
19. $CAS[i] = CAS[i] - |X|
20. end while
21. end for
22. end for
23. end for

We execute the basic initialization in Line 1. In Lines 2-3, we examine the ARC of all items by traversing through database D of customer transactions and sort items in descending order of their respective ARC. In Lines 4-12, we initiate placement for the slot type with the highest degree of premiumness, and place distinct items (1-itemsets) in corresponding premium slots with the highest-premiumness in descending order of their respective ARC. We traverse the sorted list of items, based on their ARC, and perform the required number of placements for the high-premiumness slots, and then repeat the same process for the following mid-premiumness and low-premiumness slots. We progressively repeat the afore-mentioned steps until we fulfill the placement requirements for all distinct items. Observe that we have successfully placed all 1-itemsets in the retail slots at least once.

In Lines 13-23, we exploit the kUI index to satisfy the placement requirements for the remaining slots of every slot type. We begin with the slot type with highest-premiumness, and then progressively fill up the following highest-premiumness slots in the following manner. First, we retrieve the top-revenue 2-itemset from the kUI index and perform the placement for the slot-type with the highest-premiumness. Next, we retrieve the top-revenue 3-itemset and use it to populate the highest-premiumness slots. Following that, we move to the next level and repeat the process until we reach the highest level of the kUI index, and then circle back to retrieve the next highest-revenue itemset from level 2 of the kUI index. By adopting the round robin approach, we progressively fill up the remaining highest-premiumness slots. Subsequently, we move to the mid-premiumness and low-premiumness slot types and repeat the above steps until all slots have been populated.

Now we shall discuss the run-time complexity of Algorithm 1. Given that $m$ is the total number of distinct items, $numT$ is the total number of transactions, and $maxL$ is the maximum number of levels for the kUI index, Line 1 incurs $O(m)$ time. Line 2 takes $O(m\cdot numT)$ time. Line 3 takes $O(m\cdot log(m))$ time, where $m$ is the total number of items. The for loop in Lines 4-12 places all the $m$ distinct items at least once in the first level. Hence, this process would take $O(m)$ time. The for loop in Lines 13-23 tries to place items in slots of different premiumness levels in a round robin fashion from level 2 to level $maxL$ of the kUI index. This would take $O(n)$ time because there are $n$ slots in which items need to be placed.

Observe that in contrast to our approach, the retailer could adopt other variations and combinations such as (2+2+2), (3+3) or (2+4) itemsets and so on. These strategic approaches shall be explored in our future research undertakings.

An example for PRIP is shown in Figure 3 with 12, 20 and 35 slots of high, mid and low premiumness respectively. Observe how the items/itemsets with varied revenue are mapped to slots with varied premiumness. Moreover, upon mapping and performing itemset placement, notice how we decrement and update slot values to reflect the number of remaining slots available for placement.
C. RELATIONSHIP BETWEEN SKEWNESS AND NET REVENUE

Now we shall show that as the skewness increases, the net revenue generated decreases for the PRIP algorithm. This is true intuitively, because if there is a higher number of premium slots, as is the case with lower value of skewness, there would be a higher probability of items being sold. We illustrate this with an example scenario. Let us assume that there are 3 types of slots based on premiumness. For smaller skewness value, the number of slots for each of these slot-types will be almost equal. Let $p_1$, $p_2$, and $p_3$ be the probabilities of items being sold from these 3 slot-types, where $p_1 \leq p_2 \leq p_3$. Let there be...
be \(m\) items and the net revenue for each of these items be \(0 \leq \rho_1 \leq \rho_2 \leq \rho_3 \leq \ldots \leq \rho_m\). In such a scenario, the net revenue for a lower degree of skewness (uniform number of slots to each of the three premium levels) can be computed using Equation 7.

\[
Net \ Revenue \ (Uniform) = p_1 \left( \sum_{i=1}^{[\frac{m}{3}]} \sigma_i \right) + p_2 \left( \sum_{i=[\frac{m}{3}]+1}^{[\frac{2m}{3}]} \sigma_i \right) + p_3 \left( \sum_{i=[\frac{2m}{3}]+1}^{m} \sigma_i \right)
\]

In case of skewed distribution, we assume that the ratio of slots from the least premium to the most premium decreases following a geometric progression. Hence, the net revenue would be given by Equation 8.

\[
Net \ Revenue \ (Skewed) = p_1 \left( \sum_{i=1}^{[\frac{2m}{3}]} \rho_i \right) + p_2 \left( \sum_{i=[\frac{2m}{3}]+1}^{[\frac{4m}{3}]} \rho_i \right) + p_3 \left( \sum_{i=[\frac{4m}{3}]+1}^{m} \rho_i \right)
\]

Now, we try to find the difference between the net revenue for the uniform and the skewed cases. Since \((p_2 - p_1) > 0\) and \((p_3 - p_2) > 0\), Net Revenue (Uniform) > Net Revenue (Skewed). This proves that as the skewness increases, the net revenue decreases. We have shown it for two extreme examples, but a similar kind of procedure can be followed to prove it for any small increase in the skewness as well. This can be represented by Equation 9.

\[
Net \ Revenue \ (Uniform) - Net \ Revenue \ (Skewed)
= (p_1 \left( \sum_{i=1}^{[\frac{2m}{3}]} \rho_i \right) + p_2 \left( \sum_{i=[\frac{2m}{3}]+1}^{[\frac{4m}{3}]} \rho_i \right) + p_3 \left( \sum_{i=[\frac{4m}{3}]+1}^{m} \rho_i \right)) - (p_1 \left( \sum_{i=1}^{[\frac{m}{3}]} \rho_i \right) + p_2 \left( \sum_{i=[\frac{m}{3}]+1}^{[\frac{2m}{3}]} \rho_i \right) + p_3 \left( \sum_{i=[\frac{2m}{3}]+1}^{m} \rho_i \right))
\]

\[
= (p_2 - p_1) \left( \sum_{i=[\frac{m}{2}]+1}^{[\frac{3m}{2}]} \rho_i \right) + (p_3 - p_2) \left( \sum_{i=[\frac{3m}{2}]+1}^{m} \rho_i \right) > 0 \quad (9)
\]

VI. PREMIUMNESS AND AVERAGE REVENUE-BASED ITEMSET PLACEMENT (PARIP)

This section discusses our proposed PARIP scheme.

A. BASIC IDEA

The PRIP scheme exploits our previously proposed kUI index [13] (discussed in Section III) for the allocation of itemsets of varied revenue to retail slots with varying premiumness. Recall that itemsets extracted from the kUI index can have different sizes since the kUI index comprises multiple levels, where each level corresponds to a specific itemset size. Intuitively, larger itemsets are likely to have higher revenue. The implication is that PRIP is biased towards larger itemsets since PRIP essentially considers itemset revenue, irrespective of the number of items in a given itemset. To remove the effect of this bias, we adopt the average net revenue metric for our proposed **Premiumness and Average Revenue-based Itemset Placement (PARIP)** scheme.

Our proposed PARIP scheme efficiently exploits the kUI index to retrieve and place high-revenue itemsets in the retail slots with varied premiumness, so as to improve retailer revenue. Recall that the average net revenue of an itemset is computed as the total revenue of an itemset divided by the number of items contained in it (as discussed in Section 4). PARIP primarily consists of two phases. During the initial phase, distinct items are assigned to slots based on their premiumness and the item’s ARC. In the latter phase, itemsets containing more than one item are placed in slots based on their premiumness and the itemset’s **Average Net Revenue (ANR)**.

B. ALGORITHM FOR PARIP

Our proposed PARIP scheme is depicted in Algorithm 2. It requires \(S\) slot-types, \(S_o\) slots that pertain to each type of slot, and the kUI index as input. We execute the basic initialization in Line 1. We exploit the kUI index for extracting the itemsets, and compute their respective average net revenue in Lines 2-5. In Line 6, we sort the items in decreasing order of their ANR and store them in list \(R\). In Lines 7-15, initiating placement for the slot type with the highest degree of premiumness, we identify the high-revenue itemsets from list \(R\) and place them in the corresponding highest-premiumness slots. Subsequently, we progress to the next slot-type with the highest-premiumness, and repeat the above process. We repeat the above process until all slots with different degrees of premiumness have been populated.

The initialization in Line 1 takes \(O(S)\) steps, where \(S\) is the number of premiumness levels. In Lines 2-5, as each level of the kUI index contains \(\lambda\) items, the for loop takes \(O((maxL - 1)\lambda) \approx O((maxL \times \lambda))\) time. Line 6 performs the sorting of objects. As there are \(\lambda(maxL - 1)\) items in the list \(R\), it will take \(O(\lambda(maxL - 1) \log(\lambda(maxL - 1))) \approx O(\lambda(maxL)\log(\lambda(maxL)))\) time. The nested loop in lines 7-15 takes \(O((maxL - 1)\lambda)\) time. Overall, the algorithm takes \(O(max((S), (maxL - 1)\lambda))\),
A. Mondal et al.: Retail Itemset Placement Framework Based on Premiumness of Slots and Utility Mining

FIGURE 4. Example of PARIP.

(Considers three types of slots i.e., $S=3$ (High premium, Mid premium and Low premium)
$S_1$ (Number of high-premiumness slots) : 12; $S_2$ (Number of mid-premiumness slots) : 20;
$S_3$ (Number of low-premiumness slots) : 35)

\[ \lambda (\maxL - 1) \log(\lambda (\maxL - 1)), \lambda (\maxL - 1) \], which is
equal to $O((\lambda \times \maxL) \log(\lambda \times \maxL))$.

An example for PARIP is shown in Figure 4 with
12, 20 and 35 slots of high, mid and low premiumness respectively. Observe how the high-revenue, mid-revenue and low-revenue items/itemsets are mapped to slots with high-premiumness, mid-premiumness and low-premiumness respectively. Moreover, upon mapping and performing itemset placement, notice how we decrement and update slot values to reflect the number of remaining slots available for itemset placement.

C. OPTIMALITY OF THE PARIP SCHEME

In this section, we prove the optimality of the PARIP algorithm for the data that is available in the kUI index. Recall that the PARIP algorithm is used for performing slot-premiumness aware itemset placement. Let $\{NR_1, NR_2, NR_3, \ldots, NR_T\}$ represent the net revenues of all the T high-revenue itemsets. Let $\{s_1, s_2, s_3, \ldots, s_T\}$ represent the number of slots occupied by different itemsets. Let $\{p_1, p_2, p_3, \ldots, p_T\}$ denote the probabilities of items being sold, which depends upon the premiumness of the slots they are placed into. Let us denote $\text{sar}_i = \frac{NR_i}{s_i}$, where $\text{sar}_i$ denotes the slot-aware net revenue. The maximum total revenue that can be generated will be given as follows:

$$\text{sar}_1 \cdot p_1 + \text{sar}_2 \cdot p_2 + \text{sar}_3 \cdot p_3 + \ldots + \text{sar}_T \cdot p_T \quad (10)$$

where $0 \leq p_1 \leq p_2 \leq p_3 \leq \ldots \leq p_T$ and $0 \leq \text{sar}_1 \leq \text{sar}_2 \leq \text{sar}_3 \leq \ldots \leq \text{sar}_T$.

Proof: Let us say that we need to assign coefficients to the expression $a_1 \cdot p_1 + a_2 \cdot p_2 + a_3 \cdot p_3 + \ldots + a_T \cdot p_T$ such that it is maximized. The coefficients $a_1, a_2, \ldots, a_T$ should be chosen from the set $\text{SAR} = \{\text{sar}_1, \text{sar}_2, \text{sar}_3, \ldots, \text{sar}_T\}$, where $\text{sar}_T \geq \text{sar}_{T-1} \geq \text{sar}_{T-2} \geq \ldots \geq \text{sar}_1$. Let us say that we start assigning coefficients to variables $p_T, p_{T-1}, p_{T-2}, \ldots, p_1$ in the given order, where $p_T \geq p_{(T-1)} \geq p_{(T-2)} \geq \ldots \geq p_2 \geq p_1$. To maximize the function, we will assign $\text{sar}_T$ to $p_T$ because this will maximize the
Algorithm 2 Premiumness and Average Revenue-Based Itemset Placement (PARIP)

Input: (i) S types of slots of different premiumness level and there are $S_i$ slots belonging to the $i^{th}$ slot type (ii) kUI Index

Output: Items/itemsets placement in slots of different premiumness level for increased revenue {Slots are filled after scanning the kUI index after all the 1-itemset are placed}

1: Currently available slot array $CAS[i]$ is initialized and remaining number of slot instances $S_i$ are assigned to the array after 1-itemset placement
2: for each level lv (lv = 2 to maxL) do
3: Extract all itemsets from each level of kUI index
4: Compute average net revenue of each extracted itemset
5: end for
6: Sort all itemsets in descending order of their average net revenue in list R
7: for each itemset X in list R do
8: for each slot type (i=1 to S) do
9: if $CAS[i] \leq |X|$ then
10: Place itemset X in CAS[i]
11: $CAS[i] = CAS[i] - |X|$
12: Remove itemset X in the list R
13: end if
14: end for
15: end for

product. If we take any other number, the product will be smaller than $sar_T * p_T$. Next, we assign $sar_{T-1}$ to $p_{T-1}$ because $sar_{T-1}$ is the next highest coefficient available to maximize the product $sar_{T-1} * p_{T-1}$ from the available coefficients. We proceed in the same fashion to assign remaining coefficients to the rest of the variables to obtain the optimal values for this function. Note that we follow a greedy approach in the coefficient assignment to attain the maxima. To prove our approach, let us say that we swap two of the slot-aware revenues $sar_i$ and $sar_j$. The total revenue for the two cases is given by Equation 11 and Equation 12.

\[
\begin{align*}
    f_1 &= sar_1 * p_1 + sar_2 * p_2 + sar_3 * p_3 \\
    &+ \ldots + sar_s * p_s \\
    &+ \ldots + sar_{T} * p_{T-1} + \ldots + sar_T * p_T \\
    f_2 &= sar_1 * p_1 + sar_2 * p_2 + sar_3 * p_3 + \ldots + sar_i * p_i \\
    &+ \ldots + sar_{T} * p_{T-1} + \ldots + sar_T * p_T
\end{align*}
\]

Now, let us try to find the difference between the two revenues given by Equation 11 and Equation 12.

\[
\begin{align*}
    f_2 - f_1 &= (sar_1 * p_{s} - sar_s * p_s) + (sar_s * p_{T} - sar_{T} * p_{T}) \\
    &= p_i * (sar_i - sar_s) + p_T * (sar_T - sar_s) \\
    &= (sar_s - sar_{T})(p_i - p_T) = (-)(+) < 0
\end{align*}
\]

Hence, $f_1 > f_2$. We observe that any swap of coefficient in $f_1$ will make it non-optimal as $f_1 > f_2$. Similarly, the minimum total revenue that can be generated by PARIP can be denoted by Equation 14.

\[
\begin{align*}
    sar_T * p_1 + sar(m - 1) * p_2 + sar(m - 2) * p_3 + \ldots \\
    + sar_T * p(m - 1) + sar_1 * p_T
\end{align*}
\]

VII. PERFORMANCE EVALUATION

We report our performance study in this section by comparing the PRIP and PARIP placement schemes to the existing schemes, namely HUI-Miner [9], FHM [34] and MinFHM [29]. We performed our experiments on a 64-bit Core i5 processor running Windows 10 with 8 GB memory.

We used a real dataset as well as a synthetic dataset to conduct our experiments. The Retail dataset [41] is a real dataset, which we obtained from the SPMF open-source data mining library [42]. The Retail dataset originates from an anonymous Belgian retailer and has 16,470 items and 88,162 transactions. The dataset was collected over three discrete time intervals, which together add up to a time period of about 5 months. On average, users buy a total of 13 items and most users buy between 7 to 11 items per visit. Further, most visitors use the store between 4 to 24 times over the duration of data collection. The average number of transactions per customer equals 25, which equates to about one visit per week.

Furthermore, we generated a synthetic dataset, T10I6N15K/D/2,000K, using the IBM data generator [20]. The synthetic dataset has the following parameters: T (a transaction’s average size) is 10; I (the average size of potential maximal itemsets) is 6; N (number of distinct items) is 15K; |D| (the total number of transactions) is 2,000K. For the sake of convenience, we henceforth denote this dataset as the synthetic dataset.

Note that the utility (price) values are not provided by either of the datasets. Therefore, we generate price values in the range [0.01, 1.0] for all items for both the datasets. We split the price value range in six almost equal-sized buckets, namely [0.01- 0.16], [0.17- 0.33], [0.34-0.50], [0.51-0.67], [0.68-0.84] and [0.85-1]. To generate the price value for an item, we pick one of these buckets, at random, and assign a random number from the price range to the item.

We divide the datasets in two parts, namely the training and the test set. They contain 70% and 30% of the transactions respectively. For our proposed PRIP and PARIP schemes, we exploit the kUI index to place itemsets in the training phase. The performance for various schemes has been evaluated on the test set.

Our performance metrics consist of the Execution Time (ET) and Total Revenue (TR). ET corresponds to the time taken to determine and place items in the slots in the training phase. ET refers to the time taken by various schemes to retrieve and place itemsets in the premium slots of the retail store. Mathematically, the execution time (ET) can be represented in the following manner: $ET = t_f - t_0$, where $t_0$ refers to the (starting) point in time when no slots are filled, while $t_f$ refers to the point in time when the process of itemset placement has been completed.
TR corresponds to the revenue generated by the retailer in the test phase. TR is computed as the price $\rho_z$ of an itemset $z$ multiplied by the total frequency of sales $\sigma_z$ of the itemset. In particular, for our performance study, we iterate over the test set to tabulate the frequency of sales for all itemsets. Recall that the price of an itemset is computed as the summation of prices of all items present in the itemset, as discussed in Section 4. Let $Z$ be the set of itemsets. Hence, TR is computed as follows. $TR = \sum_{z \in Z} \rho_z \star \sigma_z$. From the training set, we extract the set $S$ of itemsets and create the kUI index. For the test phase, we initialize TR to 0 and add to TR the respective prices of only those itemsets, which have been placed previously during the training phase.

In order to assign a degree of premiumness to various slots, we assign a probability of sale to all slots in the range $[0.01, 1]$. For $N$ types of slots, we split the probability bucket into $N$ sub-buckets. Subsequently, we assign the different slot-types to their corresponding probability of sale sub-buckets. For instance, for $N = 3$, the three sub-buckets in ascending order of their respective premiumness would be: $[0.01 – 0.34]$, $[0.35 – 0.68]$ and $[0.69 – 1.0]$. The implementation process to extract the top-$\lambda$ high-revenue itemsets of any given size $k$ to occupy the premium slots is as follows. Initially, we fix the number of itemsets for each level of the kUI index i.e., the value for $\lambda$. We further fix the maximum number of levels for the kUI index. Next, we retrieve itemsets of size 1, i.e., distinct items, from the transactional database and sort them based on their revenue contribution towards the retailer. We compute the top-$\lambda$ number of itemsets that exceed the threshold revenue at the first level. We compute 2-itemsets for the second level of the kUI index by combining itemsets at level-1, and we keep the top-$\lambda$ itemsets based on the revenue. Similarly, we compute top-$\lambda$ high-revenue 3-itemsets by combining level-2 itemsets with level-1 itemsets. In general, we compute the top-$\lambda$ high-revenue $n$-itemsets, by combining level-$(n-1)$ itemsets with level-1 itemsets. We continue the process of itemset creation until all the levels of the kUI index have been populated.

We set the default value for $\lambda$ and the number of levels for the kUI index as 5000 and 10 respectively. The parameters for our performance study are summarized in Table 4. The skewness in slot instances across different slot-types is represented by the Zipf factor ($Z_{NS}$). We vary the Zipf factor in the range $[0.1, 0.9]$, where a lower value is indicative of a relatively uniform distribution, while higher values correspond to a more skewed distribution. For $Z_{NS} = 0.1$, we would have an almost equal number of slot instances for every slot-type. In contrast, for $Z_{NS} = 0.9$, we would have significantly more low-premiumness slots in comparison to high-premiumness slots.

Related to our proposed approaches, the FHM algorithm [35], MinFHM algorithm [29] and HUI-Miner algorithm [9] approaches reflect the state-of-the-art in terms of retrieving and identifying HUIs. For the purpose of meaningful comparison, we conducted experiments on retail placement by implementing the FHM, Min-FHM and HUI-Miner based methods. Since these afore-mentioned schemes are not accompanied by any placement schemes, we adapt these schemes for meaningful comparison to our proposed PRIP and PARIP schemes. We now elaborate on how we adapt these schemes for the purpose of comparison.

As discussed in Section II, the HUI-Miner algorithm [9] uses the utility-list to store utility information of itemsets. The utility-list facilitates it in avoiding utility computations for a large number of candidate itemsets, as well as in avoiding expensive candidate itemset generation. We exploit the utility-list in order to retrieve itemsets to populate retail slots with varying premiumness. We begin placement with high-premiumness slots and progressively populate the remaining retail slots until all retail slots with varying premiumness are exhausted. Notably, HUI-Miner incurs higher execution time than FHM and MinFHM because it builds a utility-list corresponding to each pattern, and larger patterns are obtained by performing the join operation of utility-lists of smaller patterns.

Recall from Section II that FHM [35] uses a strategic analysis of co-occurrence of items to lower the computations for mining HUIs. The FHM algorithm generates and retrieves itemsets with varying sizes, as long as they meet the utility threshold criteria. We use the FHM algorithm to generate itemsets for populating the highest-premiumness retail slots, post which, we progressively fill the remaining slots until they are exhausted. Notably, the FHM algorithm examines item co-occurrences to reduce the computational cost of mining high-utility itemsets. The execution time for FHM remains lower than that of HUI-Miner. This is because it eliminates low-utility itemsets without performing join operations to extract high-utility itemsets (HUIs) using the utility-list data structure.

### Table 4: Parameters for the performance evaluation.

| Parameter                  | Default | Variations       |
|----------------------------|---------|------------------|
| Total no. of slots ($|S|)$ | 10      | 2, 4, 6, 8       |
| Number of slot-types ($S$) | 6       | 3, 9, 12, 15     |
| Revenue threshold ($\sigma$) | 30      |                  |
| Zipf factor ($Z_{NS}$)     | 0.7     | 0.1, 0.3, 0.5, 0.9 |
by joining smaller patterns, and only compares the utility value of joining patterns to mine high-utility itemsets. These approaches, in their essence, are oblivious to the issue of varying slot premiumness. As a consequence, they fail to place high-revenue itemsets in slots with a high degree of premiumness. Further, existing approaches (i.e., the ground truth) fail to generate, index and retrieve high-revenue itemsets in an efficient manner in order to improve retailer revenue. This may cause the retailer significant losses.

A. EFFECT OF VARYING THE NUMBER OF SLOT-TYPES

The results in Figures 5 and 6 depict the effect of varying the number S of slot-types. The results in Figure 5(a) show that with an increase in S, all schemes show an increased ET as they need to examine more patterns (itemsets). PRIP and PARIP perform better than HUI-Miner, FHM and MinFHM with regard to ET as they exploit the kUI index and thereby consider only the top-λ high-revenue itemsets of varying size. In contrast, the reference approaches need to examine a higher number of itemsets with regard to their utility thresholds. Further, recall that PARIP extends PRIP for itemset placement by exploiting the kUI index to place itemsets based on average net revenue in different slot-types of varied premiumness. Hence, it incurs slightly higher ET as compared to that of PRIP, as extracting all itemsets from the kUI index, and sorting them on the basis of the ANR for placement does not cause any significant overhead.

PRIP and PARIP return higher TR than HUI-Miner, FHM and MinFHM, as shown in the results of Figure 5(b). Recall that the reference approaches are oblivious to the notion of premiumness of retail slots, thereby the PARIP and PRIP schemes outperform them significantly. We further notice that with an increase in S, the impact of slot premiumness on TR becomes more significant. With an increased S, it is more likely for the reference approaches to place high-revenue itemsets in low-premiumness slots, and vice versa. On the other hand, PARIP provides higher TR than PRIP because PARIP considers both net revenue and size of itemsets to be placed in the slots of varied premiumness by evaluating the average net revenue, while PRIP considers only the net revenue of the itemset. We observe a saturation effect in TR for higher values of S values since the value of TR is upper-bounded by the customer purchase transactions.

Figures 5 and 6 exhibit similar trends, with differences in values arising due to different dataset sizes.

B. EFFECT OF VARIATIONS IN SKEW ACROSS SLOT TYPES

Figures 7 and 8 show the effect of skew-variations (denoted by $Z_{NS}$) across retail slots.

The results in Figure 7(a) indicate that with an increase in $Z_{NS}$, ET remains comparable for all approaches as the retail slots that need to be filled remain constant. Thereby, the time required to retrieve and place items does not change. PRIP and PARIP perform better than the reference approaches in
The results in Figure 7(b) indicate an overall decrease in TR with increase in $Z_{NS}$. As the skewness increases, the number of low-premiumness slots increases, while the number of high-premiumness slots decreases, which causes TR to decrease. Note that PRIP and PARIP perform better than the reference approaches in terms of TR due to the reasons explained for the results of Figure 5(b).

Figures 7 and 8 exhibit similar trends, with differences in values arising due to different dataset sizes.

**C. EFFECT OF VARYING THE TOTAL NUMBER OF SLOTS**

Figures 9 and 10 show the results when the total number $T_S$ of retail slots is varied. The results in Figure 9(a) indicate an overall increase in ET as $T_S$ increases. This is because more patterns are required to be examined for populating more slots, thereby necessitating higher ET. PRIP and PARIP perform better than the reference approaches in consonance with the rationale provided earlier for the results of Figure 5(a).

The results in Figure 9(b) indicate that with an increase in $T_S$, all schemes show an overall increase in TR. This is because as $T_S$ increases, more itemsets are needed to populate an increased number of slots, thereby causing TR to improve. PARIP and PRIP provide higher TR in comparison to the reference approaches in consonance with the rationale provided for the results of Figure 5(b).

Figures 9 and 10 exhibit similar trends, with differences in values arising due to different dataset sizes.
VIII. LIMITATIONS
In this section, we discuss the limitations of our proposed approaches. The limitations are summarized as follows:

- **Changes in macro-environmental trends:** The proposed approach improves the retailer revenue based on the knowledge of itemsets extracted from historical customer purchase transactions and essentially assumes that future trends would be a reflection of past trends. However, a shift in macro-environmental factors (e.g., the Coronavirus pandemic) can cause a sudden shift in the demand for products, thereby resulting in changes in user purchase behavior. Consequently, the itemsets would need to be generated, while considering such shifts in product demand, and then placed in the slots of the retail store to reflect the effect of such new macro-environmental trends on the demand for products. We are planning to investigate this issue in more detail as part of our future work.

- **Special occasions and externalities:** In case of special events (e.g., Christmas, New Year, Black Friday sales etc.), our assumptions may not hold good in practice because the customer purchase behavior patterns can change significantly w.r.t. the historical customer purchase patterns. In such scenarios, knowledge is required to be extracted in a context-aware manner to incorporate the impact of such externalities. We are planning to investigate such issues as a part of our future work.

- **Slot size:** In this work, for simplicity, we have assumed that each premium slot is of the same size. However, in practice, slot sizes of the premium slots in retail stores may vary. Hence, in addition to slot-premiumness, retailer revenue can be further realized and improved by examining the respective sizes of the various premium slots. We plan on investigating the issue of slot size in addition to slot premiumness in our future works.

- **Perishable and non-perishable goods:** Notably, performing itemset placement in a consumer-good-type aware manner can help improve retailer revenue, and help prevent significant losses. In addition to slot premiumness, it is therefore pragmatic and necessary to explore the issue of perishable goods with regard to itemset placement. In particular, observe that perishable goods may need to be placed in specific locations (e.g., deep freezers or cold storage) in a retail store, and the locations of such deep freezers or cold storage units typically depend upon the overall layout of the retail store. Therefore, it would also impact our itemset placement scheme. We plan to examine the same as part of our future work.

IX. CONCLUSION AND FUTURE WORK
The slots in a given retail store typically vary in terms of premiumness. Items placed in the more premium slots generally have a higher probability of sales. Furthermore, given that customers tend to buy itemsets (as opposed to individual items), this work has addressed the problem of improving the revenue of the retailer by placing itemsets in slots with varied premiumness.

In particular, we introduced the notion of *premiumness of retail slots* and proposed two efficient schemes, namely PRIP and PARIP, for placing itemsets having varied revenue in slots with varied premiumness with the goal of improving retailer revenue. In particular, both of our proposed schemes work on the basis of the knowledge and patterns that we generate from customer purchase transactions. Further, both schemes exploit the kUI index to identify and retrieve high-revenue itemsets for placement. Moreover, we have performed a detailed performance study using both real and synthetic datasets to demonstrate the effectiveness of our proposed schemes. We have also incorporated a comprehensive mathematical analysis of our proposed schemes w.r.t. the complexity analysis.

We plan to extend our work in multiple directions. Notably, existing approaches fail to perform itemset placement based on factors such as varied item sizes and varied slot sizes, alongside slot premiumness, to further improve the revenue for the retailer. We plan to investigate ways of performing itemset placement based on these kinds of factors in our future research works. Moreover, existing works further fail to consider issues such as item inventory (i.e., the number...
of item instances available to the retailer for sale) and item expiry times (which can vary across items e.g., milk versus soap). Based on our study, we plan to develop inventory-aware itemset placement schemes as well as urgency-aware itemset placement schemes for further improving the revenue for the retailer. Additionally, we plan to develop a system/tool for exploring cost-effective ways to integrate our proposed framework with real-world information systems used in the retail industry. Moreover, we wish to perform case studies and test our framework in real-world settings to further understand the nuances pertaining to our topic of research. Furthermore, we wish to study the macro-environmental factors impacting the change in trends with regard to customer purchase behavior to further strengthen our framework.

REFERENCES

[1] P. Hansen and H. Heinsbroek, “Product selection and space allocation in supermarkets,” Eur. J. Oper. Res., vol. 3, no. 6, pp. 474–484, Nov. 1979.

[2] M.-H. Yang and W.-C. Chen, “A study on shelf space allocation and management,” Int. J. Prod. Econ., vol. 60–61, pp. 309–317, Apr. 1999.

[3] M.-H. Yang, “An efficient algorithm to allocate shelf space,” Eur. J. Oper. Res., vol. 131, no. 1, pp. 107–118, May 2001.

[4] M. Chen and C. Lin, “A data mining approach to product assortment and shelf space allocation,” Expert Syst. Appl., vol. 32, no. 4, pp. 976–986, May 2007.

[5] Y.-L. Chen, J.-M. Chen, and C.-W. Tung, “A data mining approach for retail knowledge discovery with consideration of the effect of shelf-space adjacency on sales,” Decis. Support Syst., vol. 42, no. 3, pp. 1503–1520, Dec. 2006.

[6] (2020). Largest Malls in the World. [Online]. Available: https://www.touropedia.com/largest-malls-in-the-world/

[7] Y. Liu, W.-K. Liao, and A. Choudhary, “A fast high utility itemsets mining algorithm,” in Proc. 1st Int. Workshop Utility-Based Data Mining (UBDM), 2005, pp. 90–99.

[8] P. Fournier-Viger, C. Wu, and V. S. Tseng, “Novel concise representations of high utility itemsets using generator patterns,” in Proc. ADMA, Cham, Switzerland: Springer, 2014, pp. 30–43.

[9] M. Liu and J. Qu, “Mining high utility itemsets without candidate generation,” in Proc. 21st ACM Int. Conf. Inf. Knowl. Manage. (CIKM), 2012, pp. 55–64.

[10] V. S. Tseng, C.-W. Wu, B.-E. Shie, and P. S. Yu, “UP-growth: An efficient algorithm for high utility itemset mining,” in Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD), 2010, pp. 253–262.

[11] S. Zida, P. Fournier-Viger, J. C. Lin, C. Wu, and V. S. Tseng, “EFIM: A highly efficient algorithm for high-utility itemset mining,” in Proc. MICAi, Cham, Switzerland: Springer, 2015, pp. 530–546.

[12] P. Fournier-Viger, S. Zida, J. C. Lin, C. Wu, and V. S. Tseng, “EFIM closed: Fast and memory efficient discovery of closed high-utility itemsets,” in Proc. MICAi, Cham, Switzerland: Springer, 2016, pp. 199–213.

[13] P. Chaudhary, A. Mondal, and P. K. Reddy, “A diversification-aware itemset placement framework for long-term sustainability of retail businesses,” in Proc. DEXA, Cham, Switzerland: Springer, 2018, pp. 103–118.

[14] C. Parul, A. Mondal, and P. K. Reddy, “A flexible and efficient indexing scheme for placement of top-utility itemsets for different slot sizes,” in Proc. BDA, Cham, Switzerland: Springer, 2017, pp. 257–277.

[15] P. Chaudhary, A. Mondal, and P. K. Reddy, “An efficient premiumness and utility-based itemset placement scheme for retail stores,” in Proc. DEXA, Cham, Switzerland: Springer, 2019, pp. 287–303.

[16] P. Chaudhary, A. Mondal, and P. K. Reddy, “An improved scheme for determining top-revenue itemsets for placement in retail businesses,” Int. J. Data Sci. Anal., vol. 10, no. 4, pp. 359–375, Oct. 2020.

[17] C. Hart, “The retail accordion and assortment strategies: An exploratory study,” Int. Rev. Retail, Distrib. Consum. Res., vol. 9, no. 2, pp. 111–126, Jun. 1999.

[18] A. Valenzuela, P. Raghurib, and C. Mitakakis, “Shelf space schemas: Myth or reality?” J. Bus. Res., vol. 66, no. 7, pp. 881–888, Jul. 2013.

[19] V. Sigurdsson, N. M. Larsen, and D. Gunnarsson, “Healthy food products at the point of purchase: An in-store experimental analysis,” J. Appl. Behav. Anal., vol. 47, no. 1, pp. 151–154, Mar. 2014.

[20] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” in Proc. Int. Conf. Very Large Data Bases, vol. 1215, 1994, pp. 487–499.

[21] J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation,” ACM SIGMOD Rec., vol. 29, no. 2, pp. 1–12, 2000.

[22] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, “Discovering frequent closed itemsets for association rules,” in Proc. ICDT. Berlin, Germany: Springer, 1999, pp. 398–416.

[23] C.-H. Chee, J. Jaafar, I. A. Aziz, M. H. Hasan, and W. Yeoh, “Algorithms for frequent itemset mining: A literature review,” Artif. Intell. Rev., vol. 52, no. 4, pp. 2603–2621, Dec. 2019.

[24] Y. Ali, A. Farooq, T. M. Alam, M. S. Farooq, M. J. Awan, and T. I. Baig, “Detection of schistosomiasis factors using association rule mining,” IEEE Access, vol. 7, pp. 186108–186114, 2019.

[25] T. M. Alam, K. Shaukat, I. A. Hameed, W. A. Khan, M. U. Sarwar, F. Iqbal, and S. Luo, “A novel framework for prognostic factors identification of malignant mesothelioma through association rule mining,” Biomed. Signal Process. Control, vol. 68, Jul. 2021, Art. no. 102726.

[26] T. M. Alam, K. Shaukat, H. Mahboob, M. U. Sarwar, F. Iqbal, A. Nasir, I. A. Hameed, and S. Luo, “A machine learning approach for identification of malignant mesothelioma etiological factors in an imbalanced dataset,” Comput. J., 2021.

[27] P. Fournier-Viger, J. C. Lin, C. Wu, V. S. Tseng, and U. Faghihi, “Mining minimal high-utility itemsets,” in Proc. DEXA, Cham, Switzerland: Springer, 2016, pp. 88–101.

[28] V. S. Tseng, C.-W. Wu, P. Fournier-Viger, and P. S. Yu, “Efficient algorithms for mining the concise and lossless representation of high utility itemsets,” IEEE Trans. Knowl. Data Eng., vol. 27, no. 3, pp. 726–739, Mar. 2015.

[29] P. Fournier-Viger, J. C. Lin, C. Wu, and V. S. Tseng, “FHIM: A fast and highly accurate frequent high-utility itemset mining,” in Proc. MICAi, Cham, Switzerland: Springer, 2015, pp. 530–546.

[30] P. Fournier-Viger, S. Zida, J. C. Lin, C. Wu, and V. S. Tseng, “EFIM closed: Fast and memory efficient discovery of closed high-utility itemsets,” in Proc. MICAi, Cham, Switzerland: Springer, 2016, pp. 199–213.

[31] P. Chaudhary, A. Mondal, and P. K. Reddy, “A diversification-aware itemset placement framework for long-term sustainability of retail businesses,” in Proc. DEXA, Cham, Switzerland: Springer, 2018, pp. 103–118.

[32] C. Parul, A. Mondal, and P. K. Reddy, “A flexible and efficient indexing scheme for placement of top-utility itemsets for different slot sizes,” in Proc. BDA, Cham, Switzerland: Springer, 2017, pp. 257–277.

[33] P. Chaudhary, A. Mondal, and P. K. Reddy, “An efficient premiumness and utility-based itemset placement scheme for retail stores,” in Proc. DEXA, Cham, Switzerland: Springer, 2019, pp. 287–303.

[34] P. Chaudhary, A. Mondal, and P. K. Reddy, “An improved scheme for determining top-revenue itemsets for placement in retail businesses,” Int. J. Data Sci. Anal., vol. 10, no. 4, pp. 359–375, Oct. 2020.

[35] C. Hart, “The retail accordion and assortment strategies: An exploratory study,” Int. Rev. Retail, Distrib. Consum. Res., vol. 9, no. 2, pp. 111–126, Jun. 1999.

[36] A. Valenzuela, P. Raghurib, and C. Mitakakis, “Shelf space schemas: Myth or reality?” J. Bus. Res., vol. 66, no. 7, pp. 881–888, Jul. 2013.
ANIRBAN MONDAL received the B.Tech. degree (Hons.) in computer science and engineering from the Indian Institute of Technology (IIT) Kharagpur, India, the M.B.A. degree from the University of Massachusetts Amherst (UMass), and the Ph.D. degree in computer science from the National University of Singapore.

He is currently an Associate Professor of computer science with Ashoka University. During the last 18 years, he has led multiple key projects for envisioning, designing, and architecting end-to-end systems in domains, such as urban informatics (smart cities), spatial databases, and financial analytics. His technological expertise coupled with his business capabilities as well as his ability to create a big vision and execute it to completion in diverse multi-cultural settings make him an exciting innovator. His research interests include big data analytics, mobile and ubiquitous data management, incentive-based mobile crowdsourcing, spatial databases, database indexing, big data, the IoT, distributed databases, and large-scale data management in distributed systems, such as P2P environments. He has a proven track record in establishing international research collaborations. His research collaborations include prestigious universities in Japan, Singapore, USA, Canada, Australia, and India. He has extensive work experience in both academia and industry. His work experience includes seven years as a Research Associate at The University of Tokyo, Japan; more than three years as an Associate Professor at the Indraprastha Institute of Information Technology Delhi (IIIT Delhi), India; and three years as a Senior Research Scientist at Xerox Research Centre India. He has spearheaded industry research projects in domains, such as urban informatics and finance, leading to four granted patents by the USPTO (U.S. Patent and Trademark Office) as well as several patent filings. He has also been a fellow of the prestigious Japan Society for Promotion of Science (JSPS) as well as an ACM India Eminent Speaker. He has an established reputation, key presence, and high visibility in the international research community and contributes proactively to local research communities as well. He has numerous publications in key conferences/journals and is actively involved as the PC chair/co-chair, a PC member, a journal reviewer, as well as a keynote/tutorial speaker at reputed international conferences/workshops. More recently, he has served as the General Chair for BDA 2020, and he is serving as one of the General Chairs for DASFAA 2022.

SAMANT SAURABH received the B.Tech. degree in electronics and communication engineering from the Indian Institute of Technology Guwahati, the master’s degree from the University of Massachusetts Amherst, and the Ph.D. degree in computer science from the Indian Institute of Technology Patna, India. He is currently an Assistant Professor with the Indian Institute of Management Bodhgaya, India. His area of interests include computer security, data mining, and algorithms.

PARUL CHAUDHARY received the dual degree (B.Tech. + M.Tech.) in computer science and engineering from Gautam Buddha University, India, and the Ph.D. degree in computer science from Shiv Nadar University, India. Her research areas include data mining, pattern mining, utility mining, and big data.

RAGHAV MITTAL received the B.Sc. degree (Hons.) in computer science and the Postgraduate Diploma in Advanced Studies and Research (DipASR) degree from Ashoka University. He is currently a Research Officer at Ashoka University, India, where he works in collaboration with the MPhasis Lab. His research interests include various domains in computer science, ranging from database indexing, data mining, pattern mining, and utility mining, to big data and the IoT.

POLEPALLI KRISHNA REDDY received the M.Tech. and Ph.D. degrees in computer science from Jawaharlal Nehru University, New Delhi, in 1991 and 1994, respectively. He is currently a Professor at IIIT Hyderabad, India, where he is also the Head of the Agricultural Research Center and a member of the Data Sciences and Analytics Center research team. From 2013 to 2015, he has served as the Program Director of ITRA-Agriculture & Food, Information Technology Research Academy (ITRA), Government of India. From 1997 to 2002, he was a Research Associate with the Center for Conceptual Information Processing Research, Institute of Industrial Science, The University of Tokyo. From 1994 to 1996, he was a Faculty Member of the Division of Computer Engineering, Netaji Subhas Institute of Technology, New Delhi. In Summer 2003, he was a Visiting Researcher with the School of Computer Science, Institute for Software Research International, Carnegie Mellon University, Pittsburgh, USA. He has published over 189 refereed research papers, which include 30 journal papers, four book chapters, and eight edited books. His research areas include data mining, database systems, and IT for agriculture.

Dr. Reddy is a Steering Committee Member of the Pacific-Asia Knowledge Discovery and Data Mining (PAKDD) conference series and the Database Systems for Advanced Applications (DASFAA) conference series. He has been the Steering Committee Chair of Big Data Analytics (BDA) conference series, since 2017. As the General Chair, he has organized the both 14th and 21th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD2010 and PAKDD2021) conferences, the Third National Conference on Agro-Informatics and Precision Agriculture 2012 (AIPA 2012), and the Fifth International Conference on Big Data Analytics (BDA 2017). He is currently organizing DASFAA2022. He has developed eSagu system, which is an IT-based farm-specific agro-advisory system. He has also built eAgromet system, which is an IT-based agro-meteorological advisory system to provide risk mitigation information to farmers. He is currently investigating the building of Crop Darpan system, which is a crop diagnostic tool for farmers, with the funding support from India-Japan Joint Research Laboratory Program. He has received two best paper awards. The eSagu system, which is an IT-based farm-specific agro-advisory system, has got several recognitions, including the CSI-Nihilent e-Governance Project Award, in 2006, the Manthan Award, in 2008, and the Finalist in the Stockholm Challenge Award, in 2008. He has received the PAKDD Distinguished Service Award, in 2021.

* * *

VOLUME 9, 2021 155223