Bundle Recommendation in eCommerce

Tao Zhu, Patrick Harrington, Junjun Li, Lei Tang
@WalmartLabs, San Bruno, CA 94066, USA
{tzhu, pharrington, jli, ltang}@walmartlabs.com

ABSTRACT

Recommender system has become an important component in modern eCommerce. Recent research on recommender systems has been mainly concentrating on improving the relevance or profitability of *individual* recommended items. But in reality, users are usually exposed to a set of items and they may buy multiple items in one single order. Thus, the relevance or profitability of one item may actually depend on the other items in the set. In other words, the set of recommendations is a *bundle* with items interacting with each other. In this paper, we introduce a novel problem called the Bundle Recommendation Problem (BRP). By solving the BRP, we are able to find the optimal bundle of items to recommend with respect to preferred business objective. However, BRP is a large-scale NP-hard problem. We then show that it may be sufficient to solve a significantly smaller version of BRP depending on properties of input data. This allows us to solve BRP in real-world applications with millions of users and items. Both offline and online experimental results on a Walmart.com demonstrate the incremental value of solving BRP across multiple baseline models.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Filtering; G.1.6 [**Numerical Analysis**]: Optimization

Keywords

Bundle Recommendation; Recommender System; eCommerce; Quadratic Knapsack Problem

1. INTRODUCTION

Recommender system is one of the key components in eCommerce. It allows eCommerce company to provide personalized service to individual users, increase order size by recommending accessories at checkout, and enhance user's loyalty and engagement [21]. Recent research on recommender systems has been mainly focusing on improving the
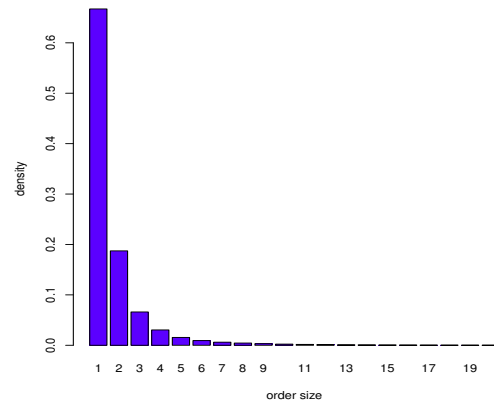
Figure 1: Distribution of order sizes

relevance of individual recommended items [16, 24]. However, in scenarios such as personalized email marketing and webpage personalization, a user is usually exposed to multiple items at one time. The relevance or attractiveness of one item may actually depends on the other items shown to the user. Therefore, we argue that we should consider the whole set of recommended items as a *bundle* instead of treating them individually and independently. Here, we refer bundle to a set of items that customers consider or buy together. Examples of bundles are, just to name a few, laptop & accessories, mattress & bed frame & other bedroom furnitures, and movies directed by the same director. Below we list a few reasons why bundle recommendation matters.

Customer buy bundles. It has been reported that the average order size for Amazon.com is 1.5 and 2.3 for Walmart.com.[1] Using the transaction data of Walmart.com, we found over 1/3 of orders contain two or more distinct items as shown in Figure 1.

Contextual influence. It is well known in classic marketing research that there is an '1+1>2' effect in carefully-designed product bundles because they may generate context that influences a customer's evaluation and choice [4, 28]. For example, IKEA promotes their products (furnitures) by displaying multiple products together in a single showroom to simulate an idea of perfect home. It is hard to deny that a mediocre product may be more tempting when displayed in the showroom.

---

[1] http://www.edatasource.com/Resources/Press_Releases/release.aspx?id=125

Figure 2: Product recommendations with and without bundles

Product compatibility or consistency. It is a better experience for customers to see items which are compatible and consistent with each other. For example, given that a customer is interested in bedroom furnitures, it is not very wise to recommend a double-size bed frame and a queen-size mattress to the customer at the same time.

Cost saving. Taking advantage of economies of scale, buying a bundle of products may cost less than buying each product separately for both customers and retailers. For example, it is a common practice for online retailers to waive shipping fee if an order exceeds certain amount.

To give a palpable understanding, we illustrate the importance of considering bundles using a real-world example as shown in Figure 2. Given a customer and his past activities on a eCommerce website, a set top-8 relevant items are generated from a recommendation engine that is optimized for relevance, which is shown in column (a). Those items with a check mark are items bought by the customer. Column (b) are what this customer eventually bought in the upcoming week. Comparing columns (a) and (b), we observe that two items (a crib and a dresser) are missing from the recommendation list in column (a). Note that customers interested in a crib mattress are very likely to be interested in cribs and other furnitures as well. By considering the bundling effect, column (c) show the items selected using our proposed approach (more details later). Note that all the items purchased by the customer are captured in the recommendation list. This case study just shows the potential if we consider bundling effect properly. However, deciding which items to put into a bundle is non-trivial. Typically, the number of items to recommend is bounded by a constant. Take the case study above as an example. We need to decide, not only which items in Figure 2(a) to replace, but also which items to bundle with other ones.

In this paper, we introduce a bundle recommendation problem (BRP) that can promotes bundling effect during recommendation. Its solution is a set of items that maximize some total expected reward. Examples of the reward function include the conversion rate of items in the bundle, and total revenue or profit of the bundle. The item-to-item cross-dependencies are estimated from historical data and should be able to incorporate contextual influence, product compatibility and cost saving automatically if their effects exist. This also saves the trouble to estimate these cross-item dependencies directly which can be both time-consuming and ambiguously defined. Technically, the BRP is essentially a type of Quadratic Knapsack Problem [6] and NP-hard in

general. We show that solving the BRP for all the items is equivalent to solving it for a carefully-constructed candidate set. When the estimated expected reward of items satisfies Pareto Principle (also known as the 80-20 rule), which is valid in most cases, the size of the constructed candidate set will be significantly smaller than the total number of all items. This allows us to solve millions of NP-hard problems in less than an hour on a few hundreds of nodes. We also present several practical implementations of the bundle recommendation algorithm which makes the model very convenient to be applied to any existing recommender system. In the evaluation session, we first test the BRP model on a real-world date set from Walmart.com. We then conduct an online A/B testing using the BRP model. Both the offline and online results show that the BRP can consistently boost the performance of the baseline methods.

The contribution of this paper is threefold. Firstly, we propose a novel bundle recommendation problem which considers the set of recommendations as a whole. It takes into account cross-item dependencies explicitly and rigorously. To the best of our knowledge, the BRP model is the first model that considers the contextual influence of bundles in recommender system; Secondly, we show that it may be sufficient to solve BRP for a substantially smaller candidate set, making it tractable and scalable; Thirdly, we justify the efficacy of BRP by applying the BRP model to real-world applications under both offline and online scenarios.

2. RELATED WORK

Recommender system has attracted a lot of attention since the Netflix prize [12, 24, 20]. Majority of research on recommender system has been focused on improving the accuracy of predicting the relevance of individual item with respect to a user [11, 23]. Item bundling in recommender systems has been considered in vocation package recommendation [27, 15], but it is modeled as a (linear) knapsack problem and no item-to-item pairwise dependency is considered in their work. Item bundling is also considered in curriculum recommendation because of the sequential ordering of courses [18] and time constraints. In these works, cross-item dependencies are modeled as hard constraints which makes the problem difficult to solve and unnecessary in our context. In eCommerce, it is difficult to draw a hard constraint between two items, not to mention the size of catalog can be overwhelming. More likely, users have different preferences over item sets, rather than a clear yes or no. Therefore,

those methods proposed in other domains cannot be applied directly to eCommerce.

One seemingly relevant work is [8]. Indeed, Garfinkel et al. studies a different problem. The bundles are predefined by retailers with different cost/price. And the shopper has his own request of products and thus has to choose a set of bundles to satisfy his demand while minimizing the overall cost. A greedy method is introduced to select bundles one by one for the shopper. By contrast, in our application, we have to decide which items to be put into a bundle and recommend to customers.

On the other hand, researches on product/price bundling have a longer history in traditional marketing and economics, see e.g., [5, 26, 2]. Product/price bundling refers to the offering of two or more products as one combined product at one price. The psychological impact and economical reasons of bundling products are well understood [5, 26], and the techniques of designing bundles are well studied [2]. However, those bundling techniques are not personalized thus not applicable in our case. Moreover, the bundling method in traditional marketing is usually considering *hard* bundles where products in bundle cannot be sold separately, while we are considering *soft* bundles in recommendation where the bundling is implicit and products in bundle can be sold separately.

Another related problem is Quadratic Knapsack Problem (QKP). It is a general optimization model that can take cross-dependency into account [6]. One simple application of QKP is the Max-Clique problem [10] where the cross-dependency is represented as unweighted edge in graph. A more practical application of QKP is portfolio selection problem in finance where the cross-dependency is the covariance of two securities [13]. Although QKP is powerful optimization model, it is rarely used for large-scale problem in practice because QKP is NP-hard in general and only small or medium scale problem is tractable by modern mixed integer solver [14, 17].

3. BUNDLE RECOMMENDATION

In this section, we first define a general version of recommendation problem where all kinds of cross-item dependencies (e.g., contextual influence, cost dependency) are considered. With some assumption and approximation, the so-called Bundle Recommendation Problem (BRP) is then introduced.

In scenarios such as personalized email ad-campaigns and product suggestions on product webpage, online customers are usually exposed to a variety of relevant items. Typically, the number of items to recommend at one time is bounded by a constant, say k. Here, we consider the problem of selecting a set of k items from a given list of relevant items. More precisely, given a user u, we want to recommend k items from a catalog containing n items whose relevances with respect to the user u are known. Note that the relevance score can be imputed through any existing recommendation techniques.

Let $R_{i_1...i_k}$ denote the reward if items $i_1...i_k$ are bought by user u at the same time. The reward can be revenue, profit, number of conversions etc. Because of economics of scale, the reward usually satisfies the property $R_{i_1...i_k} \geq \sum_j R_{i_j}$. Let $P(i_1, i_2, ..., i_s|s, u)$ denote the probability that user u buys items $i_1, i_2, ..., i_s$ given that user u will buy s distinct items. For presentation convenience, the notation of u is

skipped unless necessary, i.e., we simply use $P(i_1, i_2, ..., i_s|s)$. Let x_i denote the decision variable where $x_i = 1$ means item i is shown to the user u and 0 otherwise. Then, the total expected reward (denoted by R_{total}) of recommending a set of k distinct items to user u can be written as

$$R_{total}(x) = \left(\sum_{i_1} R_{i_1} P(i_1|s=1) x_{i_1} \right) P(s=1)$$
$$+ \left(\sum_{i_1 < i_2} R_{i_1 i_2} P(i_1, i_2|s=2) x_{i_1} x_{i_2} \right) P(s=2)$$
$$+ \cdots$$
$$+ \left(\sum_{i_1 < ... < i_k} R_{i_1...i_k} P(i_1, ..., i_k|s=k) \prod_j x_{i_j} \right) P(s=k).$$

The goal of general bundle recommendation is to find a set of k items such that the total expected reward R_{total} is maximized, i.e.,

$$\max_{x \in \{0,1\}^n, \ |x|=k} R_{total}(x). \tag{1}$$

Based on the distribution of order sizes from the Walmart.com (see Figure 1), we found most online customers purchase only one or two items in one shopping session. Thus, it is usually sufficient to consider only the first and the second terms and ignore the other higher order terms of $R_{total}(x)$:

$$R_{total}(x) \approx \left(\sum_i R_i P(i|s=1) x_i \right) P(s=1)$$
$$+ \left(\sum_{i<j} R_{ij} P(i,j|s=2) x_i x_j \right) P(s=2).$$

Let

$$r_i = R_i P(i|s=1), \tag{2}$$
$$Q_{ij} = \begin{cases} \frac{1}{2}(R_{ij}) P(i,j|s=2) & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}, \tag{3}$$
$$\lambda = \frac{P(s=2)}{P(s=1)}, \tag{4}$$

it follows that

$$R_{total}(x) \approx \sum_i r_i x_i P(s=1) + \sum_{i,j} Q_{ij} x_i x_j P(s=2)$$
$$= P(s=1) \left[r^T x + \lambda x^T Q x \right].$$

Since $P(s=1)$ is a constant given one user, the general bundle recommendation problem (1) can be approximated by

$$(\text{BRP}) \quad \max_{x \in \{0,1\}^n, \ |x|=k} r^T x + \lambda \cdot x^T Q x. \tag{5}$$

From now on, we refer problem (5) as the *Bundle Recommendation Problem*, or BRP. We call the data r and Q the *expected reward vector* and *cross-dependency matrix*.

3.1 Expected Reward & Cross-Dependency

Constructing the expected reward vector r and cross dependency matrix Q is nontrivial. In this subsection, we present one approach to estimate them.

First of all, the rewards R_i and R_{ij} are not user-specific and are given depending on the business requirement. For

instance, if the recommendation is optimized for number of conversions (purchases), then $R_i = 1$ and $R_{ij} = 2$. Similarly, the reward can be easily modified if revenue or profit is emphasized.

On the contrary, the probabilities $P(i|s = 1)$ and $P(i, j|s = 2)$ are essentially *user-specific*, and thus it is very challenging to estimate them accurately. However, the probability $P(\text{buy } i|\text{user } u)$ or its proxy is usually available from any recommender system. Meanwhile, the transition probability $P(\text{buy } i|\text{buy } j)$ can be estimated from historical transaction data using the following formula:

$$P(\text{buy } i|\text{buy } j) = \frac{\#\text{ users who bought items } i\&j \text{ in one order}}{\#\text{ users who bought item } j}.$$

Then one way to approximate the probabilities $P(i|s = 1)$ and $P(i, j|s = 2)$ is to use the following

$$P(i|s = 1) \quad \propto \quad P(\text{buy } i|\text{user } u), \qquad (6)$$
$$P(i, j|s = 2) \quad \propto \quad P(\text{buy } i|\text{buy } j) \cdot P(\text{buy } j|\text{user } u)$$
$$+ P(\text{buy } j|\text{buy } i) \cdot P(\text{buy } i|\text{user } u). (7)$$

Plugging into Eqs. (2) and (3), we construct expected reward vector r and cross-dependency matrix Q. Because of the scaling effect in constructing r and Q, we leave the parameter λ as a tuning parameter to control the level of bundling effect in final recommendations.

We remark that Eqs. (6) and (7) are just one way to approximate probabilities $P(i|s = 1)$ and $P(i, j|s = 2)$. We choose them because of their availability and ease to compute. The technique discussed later in this paper applies to any r, Q and does not depend on this approximation.

Caveat: Following Eq. (6), one may propose an alternative estimation of $P(i, j|s = 2)$ as follows:

$$P(i, j|s = 2) \propto P(\text{buy } i|\text{user } u) \cdot P(\text{buy } j|\text{user } u).$$

But this is not recommended in general. Because it essentially assumes the independence of purchases between items i and j, which is against the original motivation of bundling. It can be shown that the BRP in Eq. (5) in this case reduces to maximizing only the first term, i.e., $r^T x$, which does not capture any cross-dependency at all. The details are skipped due to space limit.

4. SOLVING BRP

In previous section, we have defined and instantiated the bundle recommendation problem in eCommerce setting. In this section, we discuss properties of BRP and derive an efficient algorithm to solve the BRP with millions of items.

It is not hard see the BRP is a special case of Quadratic Knapsack Problem [6]. It includes the k-clique problem [7] as a special case if we let $r = 0$ and Q to be the adjacency matrix of a given graph, thus we have

PROPOSITION 4.1. *The general BRP is NP-hard.*

Moreover, the dimensionality of item space could be on the order of millions in practice. If the dimensionality is one million, there are roughly 2.48×10^{43} possible combinations of 8 items. And we need to solve the BRP for hundreds of millions of users. This renders the problem highly intractable.

Based on the formulation, heuristics can be developed to solve it, such as greedy method, or constraint relaxation and then rounding. But those methods have no guarantee of arriving at a global optimal solution. Below we show

several techniques to obtain *global optimal* of BRP for all users in a reasonable amount of time. All the techniques are based on the key idea of constructing a candidate set, i.e., a subset of items, which contains the optimal item selection.

4.1 The Candidate Set

In this subsection, we define a candidate set and show why it suffices to consider only items in the candidate set. It is observed in practice that majority of items in the catalog is irrelevant to a given customer. Figure 3 shows the value of sorted r of one randomly picked customer, with respect to reward being conversion, revenue and profit, respectively. Apparently, no matter which reward function it is, only a few items have a high score, while the remaining have a value close to zero. Intuitively, it seems unnecessary to consider those items when selecting the optimal item recommendation. This motivates us to construct a candidate set of items which the optimal item recommendations can only belong to.

We first introduce the *dominance* relationship between items. This dominance relationship allows us to differentiate the items that have the potential to be contained in the optimal bundle from the ones that do not.

DEFINITION 4.2. *An item β is dominated by item α if the following condition is satisfied*

$$r_\alpha + Q_{\alpha\alpha} + \min_{\mathcal{A} \subseteq \mathcal{U},\ |\mathcal{A}|=k-1} \sum_{i \in \mathcal{A}, i \neq \alpha} \lambda(Q_{\alpha i} + Q_{i\alpha}) >$$
$$r_\beta + Q_{\beta\beta} + \max_{\mathcal{B} \subseteq \mathcal{U},\ |\mathcal{B}|=k-1} \sum_{i \in \mathcal{B}, i \neq \beta} \lambda(Q_{\beta i} + Q_{i\beta}) \qquad (8)$$

where $\mathcal{U}$ denote the index set of all the items in the catalog.

If an item in a recommendation list is dominated by another item which is not in the list, it is always better to switch these two items, as stated in the lemma below.

LEMMA 4.3. *If item β is dominated by item α then we have*

$$\max_{\substack{x \in \{0,1\}^n,\ |x|=k \\ x_\beta = 1,\ x_\alpha = 0}} r^T x + \lambda \cdot x^T Q x$$
$$< \max_{\substack{x \in \{0,1\}^n,\ |x|=k \\ x_\beta = 0,\ x_\alpha = 1}} r^T x + \lambda \cdot x^T Q x.$$

PROOF. The main idea is to first decompose the objective into terms that contain β and terms that do not. Then bound the terms that contain β by the terms that contain α using Definition 4.2.

$$\max_{\substack{x \in \{0,1\}^n, |x|=k \\ x_\beta = 1,\ x_\alpha = 0}} r^T x + \lambda \cdot x^T Q x$$
$$= \max_{\substack{x \in \{0,1\}^n, |x|=k \\ x_\beta = 1,\ x_\alpha = 0}} \left[\left(\sum_{i \neq \alpha, \beta} r_i x_i + \sum_{i,j \neq \alpha, \beta} x_i \lambda Q_{ij} x_j \right) \right.$$
$$\left. + \left(r_\beta + Q_{\beta\beta} + \sum_{i \neq \beta} \lambda(x_i Q_{i\beta} + Q_{\beta i} x_i) \right) \right]$$
$$\leq \max_{\substack{x \in \{0,1\}^n, |x|=k \\ x_\beta = 1,\ x_\alpha = 0}} \left(\sum_{i \neq \alpha, \beta} r_i x_i + \sum_{i,j \neq \alpha, \beta} x_i \lambda Q_{ij} x_j \right)$$

Figure 3: Profile of sorted r with respective reward being conversions, revenue and profit

$$+ \left(r_\beta + Q_{\beta\beta} + \max_{\mathcal{B} \subseteq \mathcal{U}, \ |\mathcal{B}| = k-1} \sum_{i \in \mathcal{B}, i \neq \beta} \lambda(Q_{\beta i} + Q_{i\beta}) \right)$$

$$< \max_{\substack{x \in \{0,1\}^n, |x| = k \\ x_\beta = 1, \ x_\alpha = 0}} \left(\sum_{i \neq \alpha, \beta} r_i x_i + \sum_{i,j \neq \alpha, \beta} x_i \lambda Q_{ij} x_j \right)$$
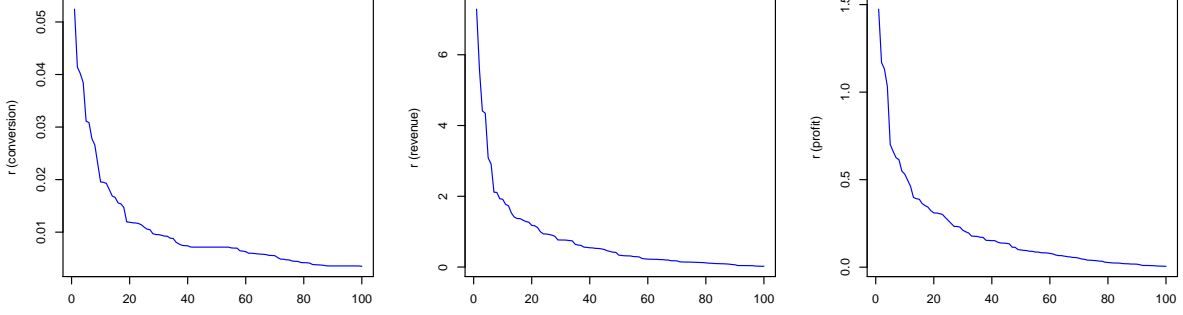
$$+ \left(r_\alpha + Q_{\alpha\alpha} + \min_{\mathcal{A} \subseteq \mathcal{U}, \ |\mathcal{A}| = k-1} \sum_{i \in \mathcal{A}, i \neq \alpha} \lambda(Q_{\alpha i} + Q_{i\alpha}) \right)$$

(Follow Definition 4.2)

$$= \max_{\substack{x \in \{0,1\}^n, |x| = k \\ x_\beta = 0, \ x_\alpha = 1}} \left(\sum_{i \neq \alpha, \beta} r_i x_i + \sum_{i,j \neq \alpha, \beta} x_i \lambda Q_{ij} x_j \right)$$

$$+ \left(r_\alpha + Q_{\alpha\alpha} + \min_{\mathcal{A} \subseteq \mathcal{U}, \ |\mathcal{A}| = k-1} \sum_{i \in \mathcal{A}, i \neq \alpha} \lambda(Q_{\alpha i} + Q_{i\alpha}) \right)$$

$$= \max_{\substack{x \in \{0,1\}^n, \ |x| = k \\ x_\beta = 0, \ x_\alpha = 1}} r^T x + \lambda \cdot x^T Q x.$$

$\square$

Now, we are ready to present our main theorem.

THEOREM 4.4. *Item dominated by k or more items will not appear in an optimal selection.*

PROOF. Suppose there is an item i dominated by k or more items in the optimal selection. Then, there is at least one item which dominate item i but not in the optimal selection; let j denote the index of this item. By Lemma 4.3, swapping item i and j will increase the objective value which contradicts the fact the current selection is optimal. $\square$

Thoerem 4.4 implies that we only need to solve the BRP for the set of items that are not dominated by k or more items, which we shall refer as the candidate set (denoted by $\mathcal{V}$).

COROLLARY 4.5. *Given BRP with data r, Q and λ, we have*

$$\max_{x \in \{0,1\}^n, \ |x| = k} r^T x + \lambda \cdot x^T Q x$$

$$= \max_{x \in \{0,1\}^m, \ |x| = k} r_\mathcal{V}^T x + \lambda \cdot x^T Q_\mathcal{V} x \qquad (9)$$

where $\mathcal{V}$ is the set of indices corresponding to the items that are dominated by no more than $k - 1$ items, i.e.,

$$\mathcal{V} = \{i | r_i + \lambda \bar{\delta}_i \geq \text{the } k\text{-th largest element in } (r + \lambda \underline{\delta})\} \quad (10)$$

where $\bar{\delta}, \underline{\delta}$ are n-dimensional vectors such that $\forall i \in \mathcal{U}$

$$\bar{\delta}_i = Q_{ii} + \max_{\mathcal{B} \subseteq \mathcal{U}, \ |\mathcal{B}| = k-1} \sum_{j \in \mathcal{B}, j \neq i} \lambda(Q_{ij} + Q_{ji}), \qquad (11)$$

$$\underline{\delta}_i = Q_{ii} + \min_{\mathcal{A} \subseteq \mathcal{U}, \ |\mathcal{A}| = k-1} \sum_{j \in \mathcal{A}, j \neq i} \lambda(Q_{ij} + Q_{ji}) \qquad (12)$$

and $r_\mathcal{V}, Q_\mathcal{V}$ denote the sub-vector and sub-matrix of r, Q with rows and/or columns corresponding to the items that belong to $\mathcal{V}$ and $m = |\mathcal{V}|$.

For the bundle recommendation problem, the size of candidate set is usually significantly smaller than n and only a few times larger than k in practice. Here show some empirical results for the sizes of candidate set. The expected reward vector r is estimated using the category-based recommendation model described in Subsection 5.1.1 and Q is estimated using (7). Table 1 shows how the size of candidate set m changes as λ varies. The histogram of m for different users with $\lambda = 0.8$ is plotted in Figure 4.

Table 1: Candidate set size versus λ with $k = 8$

λ	0	0.2	0.4	0.6	0.8	1.0	1.2
Mean of m	8	11.5	13.2	14.8	16.5	18.1	19.5
STD of m	0	7.4	7.7	8.0	8.4	8.7	9.0

From Table 1 and Figure 4, we can see that the size of the candidate set is indeed much smaller than n. This is because the largest few r_i's is so dominant that the items in the tail will never be the optimal selection (see Figure 3). As the size of BRP is significantly reduced, global search algorithm such as Branch and Bound [19] can be applied. Plenty of existing software packages like MINOTAUR [14] can be used.

4.2 Constructing the Candidate Set

Though the candidate set size is usually much smaller than the total number of items n, which makes the BRP tractable, constructing the candidate set $\mathcal{V}$ will take $O(n^2)$ each user, which could be time-consuming. To construct the candidate set using Eq. (10), we first need to compute the upper and the lower bounds in Eqs. (11) and (12) respectively for each item. This will cost in total $O(n^2 \log k)$ time if heap sort is used. Then, we can resort to heap sort again to find out the k-th largest element in $(r + \lambda \underline{\delta})$ and construct
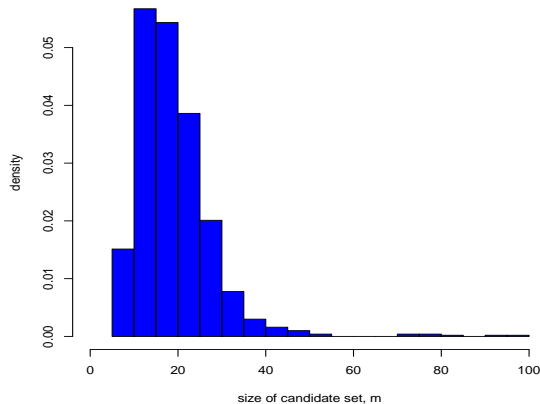
Figure 4: Distribution of candidate set size m, for different users with $\lambda = 1.2$

Figure 5: Distribution of $\bar{\bar{\delta}}_{\max}$ for different users

the corresponding candidate set, which will take $O(n)$ time. In sum, the time complexity to construct the candidate set is $O(n^2)$ when k is small. This is still time consuming if the dimensionality of item space n is huge. Keep in mind that the cross-dependency matrix Q is different with respect to different users, we have to do this computation for every single user, which makes it even worse. We have to develop a more efficient method to obtain the candidate set in reasonable time.

As shown in the previous section, the sizes of the constructed candidate set using Eq. (10) are typically very small. This motivates us to relax the condition in Eq. (10) so we may get a slightly larger candidate set but in less amount of time. We notice that the term $r + \underline{\delta}$ is mainly contributed by the expected reward r, we may construct the candidate set as follows:

$$\mathcal{V} = \{i | r_i + \lambda \bar{\delta}_{\max} \geq \text{the } k\text{-th largest element in } r\} \quad (13)$$

where $\bar{\delta}_{\max} = \max_i \bar{\delta}_i$.

Any item picked by Eq. (10) will definitely picked by Eq. (13). Therefore, we have the following corollary:

COROLLARY 4.6. *Given BRP with data (r, Q) and λ, Eq. (9) also holds true for $\mathcal{V}$ defined in Eq. (13).*

Eq. (13) does not save much computational cost, as computing $\bar{\delta}_{\max}$ is still $O(n^2)$ for each user. But if we can estimate or approximate $\bar{\delta}_{\max}$ cheaply, then we only need $O(n)$ to construct the candidate set. Figure 5 shows a distribution of $\bar{\delta}_{\max}$ for a subset of different users. We can see that majority of $\bar{\delta}_{\max}$'s are small except a few in the tail. If we assume $\bar{\delta}_{\max}$ for each user is a random variable following the distribution plotted in Figure 5, then a reasonable approximation of $\bar{\delta}_{\max}$ is to use p-quantile of $\bar{\delta}_i$'s (denoted by $\bar{\delta}_p$),

$$\hat{\mathcal{V}} = \{i | r_i + \lambda \bar{\delta}_p \geq \text{the } k\text{-th largest element in } r\} \quad (14)$$

where $\bar{\delta}_p = \inf\{x | F(x) \geq p\}$, $F(\cdot)$ is the empirical CDF of the elements in vector $\bar{\delta}$. Examples of p could be 0.80, 0.90, 0.99.

Now, if using a small subset of users to estimate $\bar{\delta}_p$ in advance, then constructing the candidate set $\mathcal{V}$ using Eq. (14) only need $O(n)$ for each user. If the candidate sets
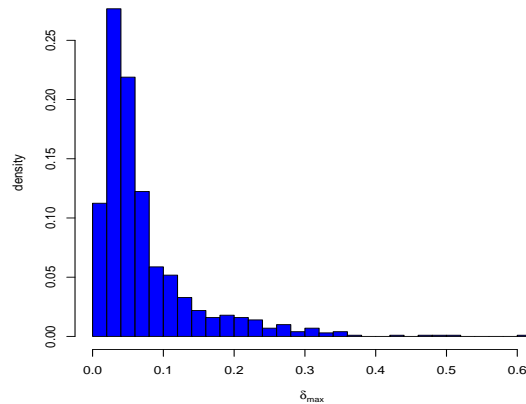
constructed using (14) contains at most m items for all users, we can simply choose the largest m elements in r to form the candidate set instead of having variable sizes of candidate set for each user, i.e.,

$$\hat{\mathcal{V}} = \{\text{indices of the largest } m \text{ elements in } r_i\}. \quad (15)$$

And a larger p we choose for $\bar{\delta}_p$, the larger m will be. There is a one-to-one relationship between p and m. Therefore, choosing p level for $\bar{\delta}_p$ is equivalent to choosing the corresponding m.

Essentially, one can use anyone of (10), (13), (14) and (15) to construct the candidate set. A good choice will depend on the data set and the tolerance of accuracy. The candidate set by (14) and (15) will take less time to construct but contain more items and thus it takes more time to solve the reduced BRP. In general, Eqs. (14) and (15) should be chosen if the bottleneck is constructing the candidate set and vice verse.

4.3 The Bundle Recommendation Algorithm

In the subsection, we summarize the bundle recommendation algorithm. Let $R^{(1)}$ denote the one-item reward vector where $R_i^{(1)} = R_i$, $R^{(2)}$ denotes the two-item reward matrix where $R_{ij}^{(2)} = R_{ij}$ and

$$P_u^{(1)} = [P(\text{buy } 1|\text{user } u), ..., P(\text{buy } n|\text{user } u)]^T$$
$$P_{ij}^{(2)} = P(\text{buy } i|\text{bought } j).$$

Given $R^{(1)}, R^{(2)}, \{P_1^{(1)}, ..., P_u^{(1)}\}, P^{(2)}$ the bundle recommendation algorithm is summarized in Algorithm 1:

We remark that the for-loop in Algorithm 1 is embarrassingly parallel as users are independent of each other. Modern large scale distributed computing framework such as MapReduce [3] can be used.

We give a toy example to demonstrate how to use the BRP algorithm for recommendation. Given items 1, 2, 3 and 4 with $R_i^{(1)} = 1, \forall i$, $R_{ij}^{(2)} = 2, \forall i, j$ and

$$P^{(1)} = \begin{pmatrix} 0.5 \\ 0.25 \\ 0.24 \\ 0.01 \end{pmatrix}, \ P^{(2)} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \ \lambda = 1.$$

Algorithm 1: Bundle Recommendation Algorithm

Data: $R^{(1)}, R^{(2)}, \{P_1^{(1)}, ..., P_u^{(1)}\}, P^{(2)}$ and λ;
Result: indices of item to recommend for user u;
for $i = 1, 2, ..., u$ **do**

    Compute r using $R^{(1)}$ and $P_i^{(1)}$ (Eq. (2) and (6));
    Construct candidate set $\mathcal{V}$ using r (Eq. (15));
    $r_\mathcal{V} = [r]_\mathcal{V}$;
    Compute $Q_\mathcal{V}$ using $[Q_\mathcal{V}]_{kl} = R_{kl}^{(2)} \cdot P_{kl}^{(2)} \cdot [P_i^{(1)}]_l$ for $k, l \in \mathcal{V}$;
    Solve BRP with data $(r_\mathcal{V}, Q_\mathcal{V}, \lambda)$ using Branch&Bound;
    Output the indices of the optimal solution;

end

We need to recommend 2 items to this user. Apply Algorithm 1 to the above data $R^{(1)}, R^{(2)}, P^{(1)}, P^{(2)}, \lambda$. Using (10), (13), (14) or (15) with $m = 3$, we obtain the candidate set

$$\mathcal{V} = \{1, 2, 3\}$$

because item 4 will never be the optimal solution. Then, we have

$$r_\mathcal{V} = \begin{pmatrix} 0.5 \\ 0.25 \\ 0.24 \end{pmatrix}, \ Q_\mathcal{V} = \begin{pmatrix} 0 & 0 & 0.37 \\ 0 & 0 & 0 \\ 0.37 & 0 & 0 \end{pmatrix}.$$

By solving the BRP with $(r_\mathcal{V}, Q_\mathcal{V}, \lambda)$, we obtain the optimal selection (items 1 and 3). In contrast, items 1 and 2 will be selected if we use traditional recommender system because they are the two most relevant items. Intuitively, choosing items 1 and 3 makes sense because user may buy item 3 as well if he/she decides to buy item 1 which is very likely.

5. EXPERIMENTS

In this section, we first test the BRP model using three baseline models on a 2-month transaction data collected from Walmart.com. We then conduct an online A/B test where the recommended items are sent to customer as an email campaign. We would like to emphasize that all the experiments and analysis are privacy-friendly as all customer ids are anonymized into random integers. In our experiments, the objective is to maximize conversion, thus the rewards are set as $R_i^{(1)} = 1$ and $R_{ij}^{(2)} = 2$. Unless otherwise specified, all the reported metrics are relative improvement over the baseline models, i.e.,

$$improvement(\%) = \frac{x_1 - x_0}{x_0} \times 100\%$$

where x_1, x_0 are the metrics of the BRP combined with baseline model (usually denoted by 'BRP+baseline') and the sole baseline model respectively.

5.1 Baseline Models

We first briefly review three baseline models used for offline evaluations. These baseline models are also used to estimate the expected reward vector r for the BRP model. We want to emphasize that our bundle recommendation algorithm works as a post-processing step, and thus is orthogonal to the baseline model used to compute relevance.

5.1.1 Category-based Recommendation

The fundamental assumption of category-based recommendation is that a user is more likely to purchase a product within the same category if he has already indicated an interest in a category. In particular,

$$P(\text{buy i|user } u) = \sum_c P(\text{buy i|buy in } c) \cdot P(\text{buy in } c|\text{user } u).$$

The interest categories of one user $P(\text{buy in } c|\text{user } u)$ is learned through his past actions. Each user is represented as a multinomial distribution of interest categories, by mapping each of his actions to its corresponding category in a carefully curated product taxonomy.

To estimate $P(\text{buy i|buy in } c)$, we examine the popularity of each product among existing transactions. In order to capture the recent trend, we restrain ourselves to look at transactions only within the past few days/weeks at recommendation time. Therefore, this category-based recommendation prefers to pick those recent best-selling products given one user's personal interest categories.

5.1.2 Markov Model

Another commonly used approach for recommendation in eCommerce is to model user actions as a markov chain [20]. That is, one's current action depends only on his most recent action (denoted by $a^{(t)}$), i.e.,

$$P(a^{(t+1)}|a^{(t)}, \cdots, a^{(1)}) = P(a^{(t+1)}|a^{(t)}).$$

In order to estimate the transtition probability from one action to another, we simply count the coocurance of two actions within a fixed time window:

$$P(\text{buy } i|\text{bought } j) = \frac{\# \text{ users who bought } j \text{ then } i}{\# \text{ users who bought } j} \quad (16)$$

As for prediction, a user's profile is represented by the most recent few transactions $(\mathcal{T})$:

$$P(\text{buy i|user } u) = \frac{1}{|\mathcal{T}|} \cdot \sum_{j \in \mathcal{T}} P(\text{buy } i|\text{bought } j).$$

5.1.3 Matrix Factorization Model

Matrix factorization has been shown to be a great success for Netflix-prize competition [12]. Due to the scale of our data set, we implemented a randomized matrix factorization detailed in [25]. Basically, the method approximately factorize a user-item matrix with a product of two lower-rank matrices, i.e.,

$$A_{m \times n} \approx P_{m \times k} \times Q_{n \times k}^T.$$

In order to handle huge numbers of rows and columns while minimizing the excess overhead in MapReduce iterations, it proposes to use a randomized SVD [9] to compute $Q_{n \times k}^T$, which can be accomplished through a simple random projection. Then it applies one iteration of alternating least squares to obtain $P_{m \times k}$ given approximate $Q_{n \times k}$.

5.2 Offline Evaluation

In this subsection, we test the BRP model using two baseline models - the category-based model and the Markov model. The purpose of offline experiment is more about testing the predicting power of the BRP model and verifying our methodologies.

5.2.1 Experiment Setup

We test our method on a subset of active customers from Walmart.com. The transaction data from 6/1/2013 to 6/30/2013 are used to generate the expected reward vector r for all customers and the conditional transition probability P which are the input for Algorithm 1. Based on our experience, it is reasonable to choose m in Eq.(15) to be 40 (this is further justified by Table 3). For each customer, we recommend 8 items. Without showing our recommendations to the customer, we see how many of our recommendations actually appeared in the corresponding customer's transaction between 7/1/2013 and 7/20/2013. Please note that offline evaluation is a conservative evaluation of the BRP model as the customers did not actually see our recommendations. We hope that a model with better accuracy in modeling users' purchase intention will incur more transactions online. Such a setup for offline evaluation has been widely used in recommender systems.

Different metrics, including precision, average order size are reported. The precision of a recommender system is defined as follows:

$$precision = \frac{\text{\# of recs that appeared in transaction}}{\text{\# of all recs}}.$$

And the average order size is defined for those customers with at least one successful recommendation,

$$order\_size = \text{Mean}(\text{\# of successful recs for customer } u).$$

We also evaluate how different the BRP recommends with respect to baseline models.

$$overlap = \frac{|\{\text{BRP recs}\} \cap \{\text{baseline recs}\}|}{|\{\text{baseline recs}\}|} \times 100\%$$

Note that the reported *overlap* is the absolute value given by the above formula.

We randomly divide all the customers into 5 groups, numbered from 1 to 5. In our experiment, group 1 is used to tune the parameter λ and group 2 to 5 are exploited for offline test. Performance metrics of different groups are reported.

5.2.2 Experiment Results

Precision improvement: Table 2 presents the precision improvement over three baseline models. It is observed that the positive improvement is quite consistent across all groups and all baseline models. The same observation applies to average order size. The improvement may look marginal, because there is still a decent amount of customers who consider only one item during a shopping session online (see Figure 1). Note that most of the time, our BRP would not change the recommendation list substantially, as indicated by the column of overlap. This suggests that the BRP model is able to strike a good balance to recommend highly relevant items with a few bundling items.

Sensitivity to candidate set size: In our BRP algorithm, we exploit Eq. (15) to construct candidate set efficiently. One natural question is how sensitive our algorithm is with respect to the candidate set size. Table 3 shows the performance of group 2 with different candidate size. The same pattern is observed for all other groups. As seen in the table, all the performance metrics stabilizes for $m \geq 30$. Note that increasing the candidate set size generally will lead to better performance, as shown in the last row of the table. However, as we discussed, a larger candidate set will

Table 2: Relative improvement of BRP using 3 baseline models

	BRP($\lambda = 0.5$)+Category-based		
Group #	precision	order size	overlap
1	0.82%	2.56%	88%
2	2.22%	3.59%	86%
3	2.14%	2.40%	86%
4	1.50%	1.91%	86%
5	2.16%	2.75%	86%
	BRP($\lambda = 0.6$)+Markov		
Group #	precision	order size	overlap
1	1.24%	2.89%	85%
2	0.69%	3.20%	85%
3	1.48%	2.33%	85%
4	1.26%	2.16%	85%
5	1.37%	2.26%	85%
	BRP($\lambda = 1$)+Matrix Factorization		
Group #	precision	order size	overlap
1	3.13%	4.66%	79%
2	3.58%	4.28%	79%
3	2.03%	3.93%	79%
4	2.25%	4.69%	79%
5	3.37%	3.43%	79%

Table 3: Relative improvement of precision and average order size with different candidate set size, m, in (15) for Group 2

	BRP($\lambda = 0.5$)+Category-based		
m	precision	order size	overlap
10	0.88%	1.05%	90%
20	1.46%	2.84%	87%
30	2.01%	3.44%	86%
40	2.22%	3.59%	86%
50	2.13%	3.66%	85%
60	2.22%	3.74%	85%

incur more computational time to solve BRP. Given this diminishing return, we have to balance between accuracy and efficiency depending on business priorities and computational resources. In our application, it is reasonable to choose $m = 40$.

Sensitivity to parameter λ: We use users in group 1 for tuning the parameter λ. The results of parameter tuning are listed in Table 4. Note that when $\lambda = 0$, it is essentially relying on baseline models alone without any bundling consideration. Almost in all cases, a non-zero λ, which encourages bundling, yields better performance with respect to baselines. The empirical curve of precision with respect to λ is concave, so an optimal λ can be located. With increasing λ, i.e., stronger and stronger emphasize over bundling, it is expected that the order size increases while the overlap is getting smaller and smaller.

In summary, by considering the bundling effect, our BRP model is able to capture user purchase intension more accurately and those orders with multiple items. Next, we apply our model to online setting, in which customers will be influenced by our recommendations.

Table 4: Relative improvement of precision and average order size using different λ for Group 1

	BRP+Category-based		
λ	precision	order size	overlap
0.1	0.17%	1.00%	93%
0.2	0.26%	1.57%	91%
0.3	0.56%	1.87%	89%
0.4	0.77%	2.27%	87%
0.5	0.82%	2.56%	88%
0.6	0.52%	2.74%	84%
0.7	0.65%	2.88%	82%
0.8	0.60%	3.20%	81%
0.9	0.39%	3.16%	80%
1.0	-0.04%	3.15%	78%

Table 5: Relative improvement of click-to-open rates (click) and conversion-to-open rates (conversion) using BRP + Markov model after campaign

	BRP($\lambda = 0.6$)+Markov	p-value
click(2 days after)	9.39%	**0.041**
conversion(2 days after)	54.67%	0.191
click(1 week after)	7.43%	**0.093**
conversion(1 week after)	48.92%	0.230

5.3 Online Evaluation

In this subsection, we present an online A/B test result for the BRP model. We compared the three baseline models used in the offline evaluation and the performance of Markov model is the best of the three. Matrix factorization performs the worst in our particular application even after all kinds of parameter tuning and twists. Its inferior performance may be due to the extreme data sparsity and highly skewed distribution in purchases [1]. Therefore, the winner, i.e. the Markov model is chosen as the baseline model in our online evaluation. We remark that there is a very small amount of customers who have bought few items and the Markov model cannot generate sufficient recommendations. In such a case, the Category-based model (see Subsection 5.1.1) is used.

A personalized email campaign was launched on September 25, 2013 for our online evaluation. Each email contains 8 recommendations which are randomly distributed to a 2 by 4 matrix in the email. All emails were sent out within the same time period. We randomly select a small percentage of customers from the eCommerce company for our email campaign. The subset of customers are randomly divided into test and control groups. For the customers in the control group, we send an campaign email with 8 item recommendations generated by the baseline model only. For the customers in the test group, we send 8 item recommendations generated by the baseline model with BRP model as post-processor.

To evaluate the quality of recommendations, we exploit click-to-open rate and conversion-to-open rate as our evaluation metrics. They are defined as follows:

$$\text{click-to-open rate} = \frac{\text{\# email click-thru's}}{\text{\# email opens}}$$

$$\text{conversion-to-open rate} = \frac{\text{\# conversions}}{\text{\# email opens}}$$

where a 'conversion' corresponds to the event that a customer clicked an item in the email and eventually placed an order in the same browsing session. Even though customers who clicked one item in the email may buy it later, we decide not to count those cases as conversions due to various co-existing targeting channels like search engine marketing or retargeting. The conversion attribution problem itself is

tricky and worth further research. Here we use a much more conservative approach to count conversions.

The numbers of 2 days and one week after campaign are listed in Table 5. Actually, the numbers do not change much since two days after the email campaign. The Mann-Whitney (or Mann-Whitney-Wilcoxon) Test described in [22] is used to test the significance of the improvement using BRP and the p-value is reported. From Table 5, we see that BRP does improve both click-to-open rate and conversion-to-open rate. The difference is significant for click-to-open rate, indicating that our bundling recommendation algorithm are more intriguing for customers to click. The improvement of bundle recommendation is more observable in online setting than in offline setting because users were indeed exposed to the recommendation and thus contextual influence could take effect.

The resultant conversions also jump, but the p-value is larger due to data sparsity. Because the intrinsic nature of purchases and our conservative way to count conversions, the number of conversions is several orders smaller than email opens and clicks. We believe the numbers here are a positive sign and we hope to further verify the improvement significance using a larger scale online bucket test in the future.

6. CONCLUSIONS AND DISCUSSIONS

In this paper, we introduced a novel bundle recommendation model. The bundle recommendation model can be considered as a post-processor, thus can be applied to any existing recommender system. By ignoring the higher order terms of the total expected reward function, the general bundle recommendation model can be formulated as a quadratic knapsack problem which we call the Bundle Recommendation Problem (BRP). However, the BRP is a large-scale NP-hard problem in general and we need to solve it for each individual user. This renders the problem highly intractable. To circumvent this, we showed that it is equivalent to solving BRP for only a subset of items called candidate set. In our application, the size of the candidate set is significantly smaller than n, thus the BRP can be easily solved by any Branch and Bound algorithm. We then tested our bundle recommendation algorithm in both offline and online experiments. Both the offline and online results showed our bundle recommendation algorithm can consistently improve the baseline models in terms of predefined reward like conversions or revenues.

In this work, we have shown bundle recommendation is able to trigger more transactions in eCommerce, without even leveraging the price advantage of bundles. It would be interesting to integrate dynamic pricing techniques with bundle recommendation. In some applications where order sizes are usually greater than 2, the higher order terms in the

objective function may play an important role. The techniques of solving a reduced small-scale problem can be generalized to the higher order bundle recommendation problem, but the difficulty will be estimating and storing those higher order tensors in practice. On the other hand, we notice that some research has proposed to maximize diversity of recommendation [29]. Diversity can also be considered as one type of cross dependency. We hope to integrate both diversity maximization and bundling effect for recommendation in the future. We believe that the bundle recommendation problem can be generalized to other domains beyond eCommerce, and most techniques proposed in this work are still valid. It boils down to constructing proper r and cross-dependency Q for the particular application.

7. REFERENCES

[1] F. Aiolli. Efficient top-n recommendation for very large scale binary rated datasets. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 273–280. ACM, 2013.

[2] Y. Bakos and E. Brynjolfsson. Bundling information goods: Pricing, profits, and efficiency. *Management Science*, 45(12):1613–1630, 1999.

[3] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[4] M. E. Drumwright. A demonstration of anomalies in evaluations of bundling. *Marketing Letters*, 3(4):311–321, 1992.

[5] G. J. Gaeth, I. P. Levin, G. Chakraborty, and A. M. Levin. Consumer evaluation of multi-product bundles: An information integration analysis. *Marketing Letters*, 2(1):47–57, 1991.

[6] G. Gallo, P. L. Hammer, and B. Simeone. Quadratic knapsack problems. In *Combinatorial Optimization*, pages 132–149. Springer, 1980.

[7] M. R. Garey and D. S. Johnson. *Computers and intractability*, volume 174. Freeman New York, 1979.

[8] R. Garfinkel, R. Gopal, A. Tripathi, and F. Yin. Design of a shopbot and recommender system for bundle purchases. *Decision Support Systems*, 42:1974–1986, 2006.

[9] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[10] J. Hastad. Clique is hard to approximate within $n^{1-o(1)}$. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 627–636. IEEE, 1996.

[11] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[12] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[13] D. Laughhunn. Quadratic binary programming with application to capital-budgeting problems. *Operations Research*, 18(3):454–461, 1970.

[14] S. Leyffer, J. Linderoth, J. Luedtke, A. Mahajan, and T. Munson. Minotaur, a toolkit for solving mixed-integer nonlinear optimization problems.

[15] Q. Liu, Y. Ge, Z. Li, E. Chen, and H. Xiong. Personalized travel package recommendation. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 407–416. IEEE, 2011.

[16] P. Melville and V. Sindhwani. Recommender systems. *Encyclopedia of Machine Learning*, pages 829–837, 2010.

[17] G. Optimization. Gurobi optimizer reference manual. *URL: http://www. gurobi. com*, 2012.

[18] A. Parameswaran, P. Venetis, and H. Garcia-Molina. Recommendation systems with complex constraints: A courserank perspective. *Transactions on Information Systems (TOIS)–To Appear*, 2011.

[19] P. M. Pardalos and G. P. Rodgers. Computational aspects of a branch and bound algorithm for quadratic zero-one programming. *Computing*, 45(2):131–144, 1990.

[20] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820. ACM, 2010.

[21] J. B. Schafer, J. Konstan, and J. Riedi. Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 158–166. ACM, 1999.

[22] G. Shani and A. Gunawardana. Evaluating recommendation systems. In *Recommender systems handbook*, pages 257–297. Springer, 2011.

[23] B. Shapira. *Recommender systems handbook*. Springer, 2011.

[24] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Scalable collaborative filtering approaches for large recommender systems. *The Journal of Machine Learning Research*, 10:623–656, 2009.

[25] L. Tang and P. Harrington. Scaling matrix factorization for recommendation with randomness. In *Proceedings of the 22nd international conference on World Wide Web companion*, pages 39–40, 2013.

[26] G. Tellis and S. Stremersch. Strategic bundling of products and prices: a new synthesis for marketing. *Journal of Marketing*, 66:72, 2002.

[27] M. Xie, L. V. Lakshmanan, and P. T. Wood. Breaking out of the box of recommendations: from items to packages. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 151–158. ACM, 2010.

[28] M. S. Yadav. How buyers evaluate product bundles: A model of anchoring and adjustment. *Journal of Consumer Research*, pages 342–353, 1994.

[29] M. Zhang and N. Hurley. Avoiding monotony: improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 123–130. ACM, 2008.