Fairness in Package-to-Group Recommendations

Dimitris Serbos
University of Ioannina
dserbos@cs.uoi.gr

Shuyao Qi
University of Hong Kong
qisy@connect.hku.hk

Nikos Mamoulis
University of Ioannina
nikos@cs.uoi.gr

Evaggelia Pitoura
University of Ioannina
pitoura@cs.uoi.gr

Panayiotis Tsaparas
University of Ioannina
tsap@cs.uoi.gr

ABSTRACT

Recommending packages of items to groups of users has several applications, including recommending vacation packages to groups of tourists, entertainment packages to groups of friends, or sets of courses to groups of students. In this paper, we focus on a novel aspect of package-to-group recommendations, that of *fairness*. Specifically, when we recommend a package to a group of people, we ask that this recommendation is *fair* in the sense that every group member is satisfied by a sufficient number of items in the package. We explore two definitions of fairness and show that for either definition the problem of finding the most fair package is NP-hard. We exploit the fact that our problem can be modeled as a coverage problem, and we propose greedy algorithms that find approximate solutions within reasonable time. In addition, we study two extensions of the problem, where we impose category or spatial constraints on the items to be included in the recommended packages. We evaluate the appropriateness of the fairness models and the performance of the proposed algorithms using real data from Yelp, and a user study.

Keywords

Recommendation systems; Package-to-Group; Fairness; Proportionality; Envy-freeness

1. INTRODUCTION

Recommender systems have attracted a lot of research attention and have been deployed in a wide range of applications [23, 1]. Besides the classic and well-studied problem of recommending a new item to a user, there has been increasing interest in suggesting a package (or bundle) of items to a user [6, 27], or an item to a group of users [29, 20]. In a recent work the problem of recommending a package to a group of users was addressed [18]. This is a problem with many practical applications. Examples include creating a vacation package for a group of tourists, suggesting an entertainment package to a group of friends for a night out, deciding a session of movies to show to an audience, planning a 5-course dinner, selecting papers for oral presentation for a conference, and many more.

In this paper, we study the problem of *fairness* in package-to-group recommendations. This is a novel characteristic, unique to the package-to-groups recommendations. User groups may be heterogeneous, consisting of people with dissimilar tastes. A recommendation to the group should try to accommodate the preferences of all group members. This consideration is usually taken into account in the selection of a preference aggregation function [13, 18], that tries to find a consensus among the users. However, even the best items according to this function may still leave some users feeling dissatisfied and slighted. We can address this problem directly when recommending a package of items to the group, by requiring that for each user in the group, we include at least one item in the package that is high in her preferences, even if the resulting package is not the best overall. Intuitively, in this case the package is *fair*: for every user in the group, there exists at least one item that satisfies her.

Formally, given a group of users G and a set of items $\mathcal{I}$, we want to recommend to group G a package $P \subseteq \mathcal{I}$ of a given cardinality $|P| = K$. We assume that the preferences of the users to items in $\mathcal{I}$ are known or can be inferred, e.g., by applying *collaborative filtering* (CF) [21]. We say that the package P is *fair* to a user u in the group G if there is at least a number m of items in P that "satisfy" user u.

We consider two alternative definitions of fairness, depending on our definition of what it means for an item i to satisfy user u. In *fairness proportionality*, we say that u is satisfied by item i, if i is ranked in the top-rated items for u. Intuitively, in this definition, the user u considers the package fair for her, if there are at least m items that the user likes. In *envy-freeness*, we say that u is satisfied by item i, if the rating of user u for item i is among the top ratings of the users in the group G for i. Intuitively, in this definition, the user u considers the package fair for her, if there are at least m items for which the user does not feel envious. These definitions are inspired by the corresponding fairness concepts in fair division of resources [22, 9, 4].

Given a fairness definition, we can now measure how fair a package P is for the entire group G by computing the fraction of users in G to whom P is fair. Our recommendation objective is then to find the most fair package for G. We

WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3038912.3052612

model this problem as a classic set coverage problem [11]. An item i covers a user u, if u is satisfied by i. Our problem becomes that of selecting a package P of K items that maximizes the number of users who are covered at least m times. This problem is NP-hard. For the case where $m = 1$, there is a greedy algorithm with a $(1 - 1/e)$-approximation guarantee. For $m > 1$ we propose a greedy algorithm that achieves good performance in practice.

In addition, we study two extensions of the problem where there are additional constraints on the selection of the items to be included in the package. In the first extension, the items in $\mathcal{I}$ belong to categories, and for each category a specific number of items is to be selected. For this problem, we propose a 1/2-approximation greedy algorithm for the case where $m = 1$. In the second generalization, we assume that the items have spatial locations (e.g., they are entertainment venues), and there is a maximum distance constraint ϵ on the pairwise distances of items in the package. For this version, we propose an exact algorithm that exploits spatial indexing to prune and explore the search space. In view of the potentially high cost of this method, we also propose greedy heuristics.

In summary, in this paper we make the following contributions:

- We define the novel problem of maximizing fairness in package-to-group recommendations, and we propose models and algorithms for the problem.
- We consider two interesting and practical extensions of the problem where we add category and spatial constraints, and we propose appropriate algorithms.
- We evaluate the proposed models and algorithms with experiments on real data from Yelp and a user study.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 formally defines the concept of fairness in package-to-group recommendations, and introduces the problem of fairness maximization and its variants. In Section 4, we propose algorithms for the different problem variants. Section 5 presents an extensive experimental evaluation on Yelp data, and a user study on movie data. Section 6 concludes the paper.

2. RELATED WORK

Recommender systems have attracted extensive research attention and have been deployed in a wide range of applications [19]. We stress that the goal of this work is not to propose a new algorithm for estimating the preferences of the users, but rather to use the preferences to find the most fair package for a group of users.

The work most closely related to ours is the recent work in [18], where the authors formulate the problem of package-to-group recommendations, and propose probabilistic models for capturing the group preferences for a package. The concept of fairness is also included in their model, but the main objective of the work in [18] is to define the notion of quality of a package for a group. In this work, we focus on modeling and formally defining fairness, and we propose algorithms that optimize it directly.

The package-to-group recommendation is also related to item-to-group and package-to-user recommendations. To recommend single items to a group of users, earlier models combine the ratings of all group members [13] or aggregate the items recommended to each group member [16]. Recent works introduce additional factors into the model, such as agreement [3] or social relationships [14] among group members, feedback from users [20], and the impact of individual members to the group [15, 29]. In addition, more models based on probability [7] and topic modeling [15, 29] are also proposed and studied. These works consider the notion of fairness only indirectly, through the choice of the aggregation function. There are no explicit models or algorithms for guaranteeing fairness.

The problem of recommending a package of items to a single user has also been studied extensively. In [6] the authors show that several of the problem variants are NP-hard; the complexity can be reduced by user-defined constraints (e.g. [17]). The problem has also been considered under budget constraints [26, 2], as a learning problem for predicting the package interestingness [27], or as a profit maximization problem [30]. There is no notion of fairness in this case, since there is a single user to recommend to.

The fairness problem we propose in this paper is unique to package-to-group recommendations. Nevertheless, it is related to the least misery (LM) approach for the item-to-group recommendation problem. In the least-misery approach the quality of an item for a group is defined as the minimum preference score for this item, over all users in the group. The goal is to find the item with the maximum least-misery score. We can easily extend this metric to packages by taking the minimum LM score over all items. Intuitively, least misery aims at minimizing dissatisfaction of individual group members due to the presence in the package of items they do not like. Our fairness definition aims at maximizing the satisfaction of members due to the presence of items they like. Furthermore, the problem of ensuring user satisfaction becomes more complex with the presence of multiple items in the package. Least misery treats the problem indirectly, while we explicitly model the fairness of the package.

Our proportionality and envy-freeness definitions of fairness are inspired from the problem of fair division in Economics [22, 9, 4], where the objective is to fairly divide resources to a group of people who may have different preferences to the resources, such that everybody is happy about their share. Our problem is different, since the suggested package of items is *shared* among the group users and it is not pre-defined but it must be selected from a number of possible item combinations.

Finally, our problem is indirectly related to the problem of diversifying recommendations. Various studies have tried to diversify the recommended items with respect to topics [31], explanations [28], and user interests [25]. Fairness in package-to-group recommendation is related to result diversification, in the sense that a fair package is likely to include items that satisfy different users.

3. PROBLEM DEFINITION

In this section we provide the definitions of fairness we consider in this paper, we define the basic problem of fairness maximization, and extensions of the problem that consider different types of constraints.

3.1 Fairness Definition

Consider a collection $\mathcal{I}$ of items and a set $\mathcal{U}$ of users, who use and rate items from $\mathcal{I}$. We use $r(u, i)$ to denote

the rating of user u for item i. In addition to the ratings recorded by the users, with the help of collaborative filtering [21] or some other base recommendation approach, we can predict a non-recorded rating $r(u, i)$ of a user u on an item i, i.e., the anticipated preference of u for i.

Given a *group* (subset) $G \subseteq \mathcal{U}$ of users, we want to recommend to G a *package* (subset) of items $P \subseteq \mathcal{I}$ of size K. In this paper, we focus on the case where we want to form packages that are *fair* to the users in G.

We consider two different aspects of fairness. One aspect looks into the items in the package and asks that each user finds a sufficient number of items in the package that she likes compared to items not in the package. We call this aspect of fairness *proportionality*. The other aspect looks into the other users in the group and asks that for each user there is a sufficient number of items in the package that she likes more than other users do. We call this aspect of fairness *envy-freeness*. Next, we formalize these two aspects.

Proportionality. Given a package P, and a parameter Δ, we say that a user u *likes* an item $i \in P$, if i is ranked in the top-Δ% of the preferences of u over all items in $\mathcal{I}$.

DEFINITION 1. *For a user u, and a package P, we say that P is m-proportional for u, for $m \geq 1$, if there exist at least m items in P, that u likes.*

The rationale in this definition is that the existence of at least m items in the package for which u has high preference would make the user tolerant to the existence of other items that she may not prefer, considering that there are other members in the group who may like these items. In the following, we call m-proportional packages *single-proportional* if $m = 1$ and *multi-proportional* otherwise.

We can now define our first fairness metric: m-proportionality.

DEFINITION 2 (m-PROPORTIONALITY). *For a group of users G, and a package P, we define the* m-proportionality *of the package P for the group G as*

$$F_{\text{prop}}(G, P) = \frac{|G_p|}{|G|}, \tag{1}$$

where $G_p \subseteq G$ is the set of users in the group for which the package P is m-proportional.

Envy-freeness. Given a group G, a package P, and a parameter Δ, we say that a user $u \in G$ is *envy-free* for an item $i \in P$, if $r(u, i)$ is in the top-Δ% of the preferences in the set $\{r(v, i) : v \in G\}$.

DEFINITION 3. *For a user u, a package P, and a group G, we say that the package P is m-envy-free for u, for $m \geq 1$, if u is envy-free for at least m items in P.*

The rationale in this definition is that a user u feels that the package is fair, if there are items for which the user is in the favored top-Δ% of the group. Otherwise, the user has envy against the other members of the group, who always get a better deal, and thus feels she is being treated unfairly.

We can now define our second fairness metric: m-envy-freeness.

DEFINITION 4 (m-ENVY-FREENESS). *For a group of users G, and a package P, we define the* m-envy-freeness *of the package P for the group G as*

$$F_{\text{ef}}(G, P) = \frac{|G_{\text{ef}}|}{|G|}, \tag{2}$$

where $G_{\text{ef}} \subseteq G$ is the set of users in the group for which the package P is m-envy-free.

3.2 Fairness Maximization

We now define the basic fairness maximization problem that we consider in this paper. For the following, we use F to denote a fairness metric, which can be either m-proportionality, or m-envy-freeness.

PROBLEM 1 (FAIRNESS MAXIMIZATION). *Given a fairness metric F, a collection of items $\mathcal{I}$, a group of users $G \subseteq \mathcal{U}$, and a value K, construct a package $P \subseteq \mathcal{I}$ with $|P| = K$, such that $F(G, P)$ is maximized.*

3.3 Fairness Maximization with Constraints

Problem 1 can be generalized to include constraints that restrict the set of candidate packages that can be recommended to the user group G. In this section, we define two extensions which have practical applications.

3.3.1 Item Categories

In many applications, the items to be recommended belong to categories. For example, points of interests in a vacation package can be classified to museums, landmarks, parks, etc.; movies have genres; courses cover different scientific areas. Thus, we assume that we have a set of categories $\mathcal{C}$, $|\mathcal{C}| \geq 1$, and that each item in $\mathcal{I}$ belongs to one or more categories. The following formulation captures the fact that most often we want to form packages including items from different categories.

PROBLEM 2. *Given a fairness metric F, a set of categories $\mathcal{C}$, a collection of items $\mathcal{I}$, a group of users $G \subseteq \mathcal{U}$, and a set of ℓ pairs (C_j, k_j), where $C_j \in \mathcal{C}$ and $k_j \geq 1$, find a package $P \subseteq \mathcal{I}$ that includes k_j items from category C_j, and maximizes $F(G, P)$.*

This is a very general formulation of the problem. For example, when $\ell = 1$, all items in the package belong to a single category C_j and $k_j = K$ (i.e., our original problem), while by setting for all pairs, $k_j = 1$, we select a single item from each category.

3.3.2 Distance Constraints

In some applications, the package selection is constrained by the relationships between the items in it. For example, if the items are venues in an entertainment or vacation package, these venues cannot be far from each other, otherwise the package would not be appealing to the user group or even feasible.

We now consider the package-to-group recommendation problem, where we impose constraints on the pairwise distances between the items. Given a distance threshold ϵ, we require that all items are within distance ϵ. Therefore, we have the following problem definition.

PROBLEM 3. *Given a fairness metric F, a collection of items $\mathcal{I}$, a group of users $G \subseteq \mathcal{U}$, a value K, and a distance threshold ϵ, construct a package $P \subseteq \mathcal{I}$ with $|P| = K$, such that $F(G, P)$ is maximized, and for any $i_m, i_\ell \in P$, $dist(i_m, i_\ell) \leq \epsilon$*

4. ALGORITHMS

In this section, we study the complexity and propose algorithms for the fairness maximization problems we defined in Section 3.

We first introduce some additional notation. Let $G \subseteq \mathcal{U}$ be a group of users. For a fairness metric F, we define for each item $i \in \mathcal{I}$, the set $\text{SAT}_G(i) \subseteq G$ as the set of users in G that are *satisfied* by item i. The definition of what it means for an item i to satisfy a user $u \in G$ depends on the fairness metric under consideration. For proportionality, $\text{SAT}_G(i)$ contains the users that "like" the item i, that is, the users for which item i belongs in their top-Δ% most preferable items. For envy-freeness, $\text{SAT}_G(i)$ contains the users that are envy-free for the item i. It is easy to see that a package P is fair for u (m-proportional, or m-envy-free), if there are at least m items i in P such that $u \in \text{SAT}_G(i)$.

In the following, the proofs and algorithms we consider are defined using $\text{SAT}_G(i)$, and thus they are applicable to both fairness metrics.

4.1 Fairness Maximization

We first consider the basic fairness maximization problem we defined in Problem 1. We can easily show the following Lemma.

LEMMA 4.1. *The* FAIRNESS MAXIMIZATION *problem is NP-hard.*

We omit the details of the formal proof, but it is easy to see that in the case where $m = 1$, the fairness maximization problem is equivalent to a maximum coverage problem. Each item $i \in \mathcal{I}$ corresponds to a set $\text{SAT}_G(i)$, consisting of the users that are satisfied by i. Since $m = 1$, if we include i to a package P, it follows that the package P is fair for all users in $\text{SAT}_G(i)$. We say that in this case the item i *covers* the users in $\text{SAT}_G(i)$. Given a package P, the set of users for which the package P is fair is $\cup_{i \in P}\text{SAT}_G(i)$. Therefore, finding K items to maximize fairness is equivalent to the maximum coverage problem of finding K sets to maximize coverage. In the following, we will often use interchangeably the notion of maximizing fairness with that of maximizing coverage.

Fairness maximization for $m = 1$. We refer to this case as the *single coverage* case. We have the following lemma.

LEMMA 4.2. *There is a $(1-1/e)$-approximation algorithm for the* FAIRNESS MAXIMIZATION *problem, when $m = 1$.*

The lemma follows from the equivalence between fairness maximization and maximum coverage. For the latter problem, the greedy algorithm has approximation ratio $1-1/e$ [10], where e is the base of the natural logarithm. The algorithm constructs the package P greedily, each time adding to the set the item that covers (i.e., satisfies) the largest number of non-covered users. Specifically, let $\text{SAT}_G(P)$ denote the users covered (satisfied) by the package P. We define the utility of adding item i to package P as $f_G(P, i) = |\text{SAT}_G(P \cup \{i\}) \setminus \text{SAT}_G(P)|$. The Greedy algorithm at each step adds to the package P the item i that maximizes the utility. The algorithm outline is shown in Algorithm 1. We will refer to this algorithm as SPGREEDY in the case that we are maximizing the single proportionality metric, and EFGREEDY in the case we maximize the envy-freeness metric.

ALGORITHM 1: Greedy Fairness Maximization

```
Input  : Group of users G, items I, value K
Output: Package P
P ← ∅
Candidates ← I
for j = 1 to K do
    i ← arg max_{i∈Candidates} f_G(P, i)
    P ← P ⋃ {i}
    Candidates ← Candidates \ {i}
return P
```

Fairness maximization for $m > 1$. We refer to this case as the *multi-coverage* case. In this case, we require that a user is satisfied by m items in order to be considered covered. This corresponds to a *multi-cover* of the set G. The problem of finding the minimum multi-cover has been studied extensively [8, 11, 12], but the problem of maximum multi-coverage is not as well understood. In [24], the authors show a connection of this problem with the K-densest subgraph problem for $m = 2$, for which there are no known efficient approximate solutions.

We propose to adapt the Greedy algorithm for the single coverage to the multi-coverage case. In the case of single coverage, the utility of an item i is the number of users that are satisfied for the first time. In the case of multi-coverage, we count the number of users that item i satisfies an *additional* time (up to m). Specifically, let $\text{SAT}^j_G(P)$ denote the set of users in G that are satisfied exactly j times. We define the utility of adding item i to a package P, as

$$f_G(P, i) = \sum_{j=1}^{m} w_j |\text{SAT}^j_G(P \cup \{i\}) \setminus \text{SAT}^j_G(P)| \qquad (3)$$

that is, the weighted sum of the users that are satisfied an additional time. The weight w_j controls the importance of covering a user for the j-th time. We assume that $w_1 \leq w_2 \leq ... \leq w_m$, that is, the closer we are to fully covering a user, the higher the weight. In our experiments we consider weights that define an arithmetic and a geometric sequence.

We refer to the Greedy algorithm for multi-coverage as MPGREEDY when we use the m-proportionality fairness metric, and as MEFGREEDY when we use the m-envy-freeness metric.

4.2 Fairness Maximization with Constraints

We now consider algorithms for the two extensions we defined in Section 3.3.

4.2.1 Category constraints

We first consider Problem 2, where the items belong to categories, and we can only pick a fixed number of items per category. For the exposition, we assume for the moment that we can only pick a single item per category. For this problem, we apply directly the Greedy algorithms we described in Section 4.1. The only modification is that once we select an item from a specific category, we remove the items of this category from the candidate set. We use SPCGREEDY and EFCGREEDY to denote the algorithms that maximize single proportionality and envy-freeness respectively.

We can prove the following lemma. A similar proof appears in [5].

LEMMA 4.3. *The Greedy algorithm is a* $1/2$*-approximation algorithm for the* FAIRNESS MAXIMIZATION *problem with category constraints.*

PROOF. We will prove the theorem using induction on the steps of the Greedy algorithm. Let P_k denote the first k items selected by Greedy, for $k \leq K$. By definition, the k items must belong to k distinct categories. Let P_k^* denote the items selected by the optimal algorithm for the corresponding k categories. Also, let $L_k = \text{SAT}_G(P_k)$ and $L_k^* = \text{SAT}_G(P_k^*)$ denote the corresponding sets of users in G that are satisfied by the items in P_k and P_k^*. We will show that $|L_k^* \setminus L_k| \leq |L_k|$ for all $1 \leq k \leq K$. Since $|L_k^* \setminus L_k| \geq |L_k^*| - |L_k|$, it follows that $|L_k| \leq \frac{1}{2}|L_k^*|$, which proves our claim when $k = K$.

For $k = 1$ our claim is trivially true, since $|L_1^*| \leq |L_1|$, by definition of the Greedy algorithm. Assume that it is true for $k = j$. Let N_{j+1} denote the users that are satisfied by the item selected by the greedy algorithm for category $j+1$, and let N_{j+1}^* denote the corresponding set of users for the item selected in the optimal solution. We have that $L_{j+1}^* = L_j^* \cup N_{j+1}^*$. Therefore,

$$\begin{aligned} |L_{j+1}^* \setminus L_{j+1}| &= |(L_j^* \setminus L_{j+1}) \cup (N_{j+1}^* \setminus L_{j+1})| \\ &\leq |L_j^* \setminus L_{j+1}| + |N_{j+1}^* \setminus L_{j+1}| \\ &\leq |L_j^* \setminus L_j| + |N_{j+1}^* \setminus L_j| \\ &\leq |L_j| + |N_{j+1} \setminus L_j| \\ &= |L_{j+1}| \end{aligned}$$

The last inequality follows from the inductive hypothesis and the property of the Greedy algorithm that it always selects the item that maximizes the additional number of users that are satisfied by the package. □

For the general case where we select k_j items from each input category C_j, we create k_j replicas C_j^l, $l = 1, \ldots, k_j$, for each input category C_j. We then run Greedy on this dataset. Note that the same item cannot be selected multiple times since it will have coverage zero.

4.2.2 Distance constraints

We now consider Problem 3, where we want the items in the package to satisfy distance constraints. We can still adapt the basic Greedy algorithm for this case, by considering as candidate items only the items that when added to the existing solution satisfy the distance constraints. We cannot prove any guarantees for this algorithm. Actually, there are cases when the Greedy algorithm may terminate before finding K items. We now propose two heuristics, and one exact algorithm that take advantage of spatial partitioning and indexing to reduce the search space.

A space-partitioning approach. For this algorithm, we divide the space by a grid, such that each cell has width and height $\epsilon/\sqrt{2}$, that is, a diagonal of length ϵ. Thus, any two items inside the same cell are within the ϵ-distance threshold. For each cell, we then run one instance of the Greedy algorithm, considering only the items that appear in the cell, and report the best solution. This approach greedily solves one local problem per cell, however, it fails to consider packages that include items from different cells. We refer to this algorithm as PARTITIONGREEDY.

Grid-based greedy algorithm. The second approach is again a greedy method based on a space partitioning. We partition the space by a grid again, but this time each cell has side length ϵ. Then, we run again the Greedy algorithm for each cell, but this time, we allow the search to extend neighboring cells if necessary. Let L_j denote the j-th cell that is examined by the algorithm. The first item i in L_j is selected greedily. We then take advantage of the grid to reduce the number of candidate items to consider. It is easy to see that any item in L_j can only form valid packages with items inside L_j, or its direct neighbors (at most 9 cells in total). Furthermore, as more items are selected, the 9-cell search space is further reduced.

Figure 1 shows an example with a 4×4 grid. Suppose the Greedy algorithm on cell L_6 selects i_1 as the first item. For the second item, it will consider as candidates only items that fall in L_6 or in one of the 8 cells surrounding L_6 (i.e., the cells $\{L_{1-3}, L_{5-7}, L_{9-11}\}$). If the second item selected is i_2, which falls in L_2, the third item cannot be selected from cells L_{9-11} because all items in them are further than ϵ from i_2. Therefore, for the third item, we only have to consider cells $\{L_{1-3}, L_{5-7}\}$. If the third item is i_3 in L_7, the fourth item can only be selected from cells $\{L_{2-3}, L_{6-7}\}$, and so on. Therefore, having partitioned the items based on the grid, we can dynamically prune the search space of candidate cells and items. We will refer to this algorithm as GRIDGREEDY.

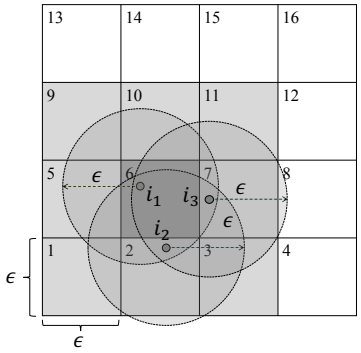


Figure 1: Grid Example

Grid-based backtracking algorithm. It is possible to change the GRIDGREEDY algorithm to an exhaustive search optimal algorithm that generates all valid item combinations based on the distance constraint. This algorithm uses the grid to minimize the cost of generating item combinations by early-eliminating items that are too far from the current (partial) combination. The *grid-based backtracking algorithm* runs one search for each item i in $\mathcal{I}$, by taking i as the first item in the combination. Then, it considers all possible items to add next to the package (thus, generating a search tree rooted at i), by examining only the neighboring cells of the one that includes i. For each item added, the search space for the next ones to add is restricted. As soon as a complete combination is formed, its fairness score is measured, and finally the best package overall is returned. We refer to this algorithm as GRIDOPTIMAL.

5. EXPERIMENTS

The goals of the experiments are three-fold. First, to understand the tradeoff between fairness and quality. We want to quantify the effect on quality when optimizing for fairness and vice versa, and understand how the hardness of the in-

put affects these two metrics. Second, we are interested in understanding the effect of the different constraints on the performance of the algorithms. Finally, we perform a user-study to qualitatively evaluate our algorithms and metrics.

5.1 Experimental Setup

We use the Yelp Challenge dataset[1] in our evaluation. The items are venues in the city of Phoenix, rated by Yelp users; we use data from a single city, in order to make our packages more realistic. We consider venues from the five most popular categories: *restaurants*, *shopping*, *beauty & spa*, *health & medicine*, and *nightlife*. The resulting dataset contains about 100K users, 17K venues and 476K ratings.

Since the ratings matrix of Yelp is very sparse, we employ collaborative filtering (CF) [21] to fill the matrix as much as possible. In particular, we use Apache Mahout[2] to build an item-based CF recommender and predict for each user u the item ratings that are not present in the dataset. The above procedure results in 53M ratings.

For the construction of the groups, in order to obtain meaningful results, we consider users for which we can obtain a sufficient number of ratings. More specifically, we create groups with users that have at least 3,000 ratings in the completed matrix. For a given group, the candidate items for the packages are the items with a recorded or predicted rating by all users in the group.

To avoid groups for which there are trivial solutions (e.g., there is an item that satisfies all users), we make sure to maximize the diversity of preferences in the group. Therefore, we use the following procedure for creating the groups. We sample the first user uniformly at random from the set of candidate users. Then, we select the second user to be the user that is the least similar to the selected user. The similarity is computed using the Pearson correlation coefficient between the rating vectors of the users. Proceeding like that, the k-th user is selected so as to minimize the maximum similarity with the previously selected $k-1$ users in the group. In our experiments we report average values over 50 different random group initializations. In Section 5.2.1 we report experiments with other group initializations.

All algorithms were implemented in Java and the tests ran on a machine with Intel Core i5-760 2.80GHz and 4GB main memory, running Windows 7. We consider different parameter values in our experiments. When not clearly specified, the default values that we use in our experiments are $|G| = 8$ for the group size, $K = 4$ for the package size, and $\Delta = 5\%$.

5.2 Algorithm Performance

We now study the performance of our algorithms for the different problems we defined.

5.2.1 Single proportionality and envy-freeness

As we have already mentioned, our goal is to study the fairness-quality tradeoff. We thus consider four different greedy algorithms where each algorithm optimizes a different metric, either for fairness or quality, and then compare all algorithms against these metrics. More specifically, we study the following algorithms:

SPGreedy: The greedy algorithm described in Section 4 that maximizes $F_{\text{prop}}(G, P)$.

EFGreedy: The greedy algorithm described in Section 4 that maximizes $F_{\text{ef}}(G, P)$.

AVRGreedy: A greedy algorithm that selects items greedily to maximize the *average rating* of the package

$$\text{AVR}(G, P) = \frac{1}{|G||P|} \sum_{u \in G} \sum_{i \in P} r(u, i)$$

LMGreedy: A greedy algorithm that selects items greedily to maximize the *least misery* metric of the package

$$\text{LM}(G, P) = \min_{u \in G} \min_{i \in P} r(u, i)$$

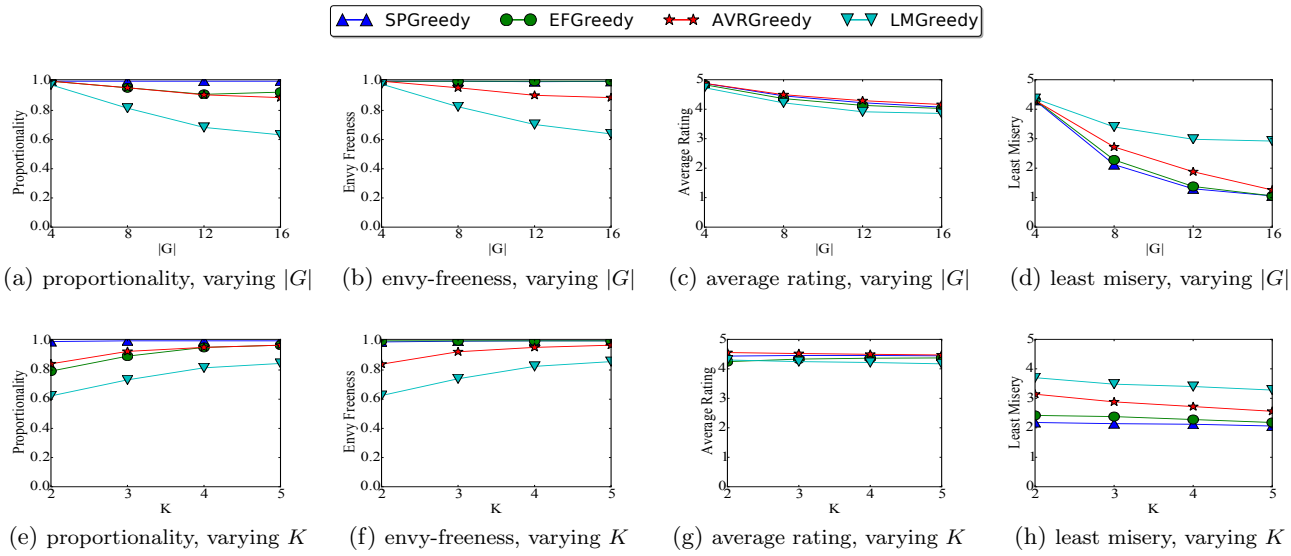
We also experimented with a random selection of items. This algorithm is consistently outperformed by all other algorithms, so we do not include it in the experiments to avoid cluttering the plots.

Figure 2 shows the results of the four algorithms, for the four different metrics we consider, as a function of the size of the group and the package. The first observation is that as expected each greedy algorithm performs the best for the metric that the algorithm maximizes. The SPGreedy algorithm achieves very high proportionality values (proportionality 1 for small groups), but it is also competitive on the envy-freeness metric, achieving essentially the same performance as the EFGreedy. At the same time, the average rating of the packages produced by SPGreedy and EFGreedy is close to that of the AVRGreedy algorithm, indicating that we can achieve fairness while not sacrificing the average quality. The two fairness-oriented algorithms suffer when considering the least misery metric, where they achieve low values. This is expected since the goal of the two algorithms is to ensure that they include items that satisfy the users, rather than avoiding the inclusion of items that the users do not like. Correspondingly, the LMGreedy algorithm achieves the lowest fairness values, close to random. In comparison, AVRGreedy which focuses on optimizing quality achieves much better fairness.

The differences between the algorithms become more pronounced when the size of the packages decreases, or the size of the group increases. That is to be expected, since the problem becomes more difficult for the non-specialized algorithms. It is noteworthy that the fairness values of the fairness-oriented algorithms remain relatively stable as we vary the sizes of the groups and packages. We also experimented with different values of Δ ranging from 1% to 20%. As expected both fairness metrics increase with larger Δ, because the fairness criterion becomes less strict. For the other metrics, the performance remains relatively stable as we increase Δ. We omit the results due to space constraints.

Finally we study the performance of our algorithms when varying the degree of "difficulty" of the input group. In addition to the aforementioned anti-correlated groups, we also consider random and correlated groups. In the latter case, we construct groups so as to maximize the Pearson correlation between users in the selection process. We consider such groups as the "easy" inputs, where it is easy to find items that satisfy the group. We also use the intersection between to top-Δ% rated items of two users as their similarity, and using the same mechanism as before, we construct groups that minimize this similarity. We will refer to these groups as *disjoint*. We consider these groups to be the most difficult cases, since there are few items that can satisfy many users.

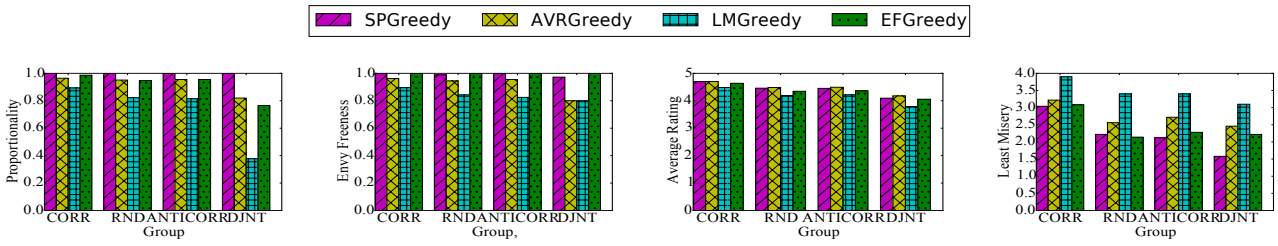
[1] http://www.yelp.com/dataset_challenge
[2] http://mahout.apache.org



(a) proportionality, varying $|G|$ (b) envy-freeness, varying $|G|$ (c) average rating, varying $|G|$ (d) least misery, varying $|G|$

(e) proportionality, varying K (f) envy-freeness, varying K (g) average rating, varying K (h) least misery, varying K

Figure 2: Performance comparison, single-coverage



(a) proportionality (b) envy-freeness (c) average rating (d) least misery

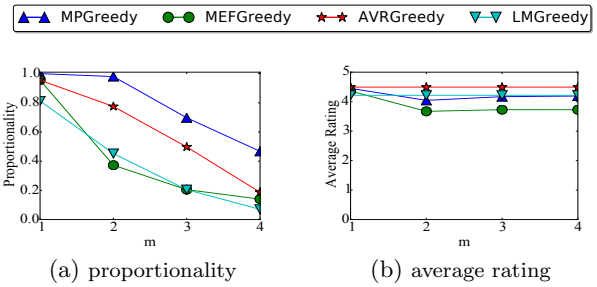
Figure 3: Performance comparison with varying group difficulty

Figure 3 shows the results of our algorithms for the different metrics. We observe that for the easy cases, quality and fairness are essentially equivalent. Selecting items with high average quality satisfies almost 100% of the group, and fair packages achieve high quality. However, when the problem becomes harder, selecting items with high average rating reduces fairness proportionality significantly. Surprisingly, the opposite is not true; optimizing for fairness does not compromise quality significantly.

5.2.2 *Multi-proportionality*

We now consider the problem of multi-proportionality fairness. Due to space constraints, we only present the performance of the algorithms with respect to m-proportionality fairness and average quality, as a function of m (see Figure 4. We use the arithmetic weighting in the implementation of the MPGreedy algorithm. We observe that as we increase m the achieved proportionality drops fast for MEFGreedy, AVRGreedy and LMGreedy. The quality of MPGreedy is lower than before, but still high.

We also experimented with the geometric weighting scheme for MPGreedy. The achieved fairness is lower than in the arithmetic case, indicating that the geometric sequence is too aggressive, pushing to include items that result in immediate coverage, and missing on items that create the potential to cover users in the future.

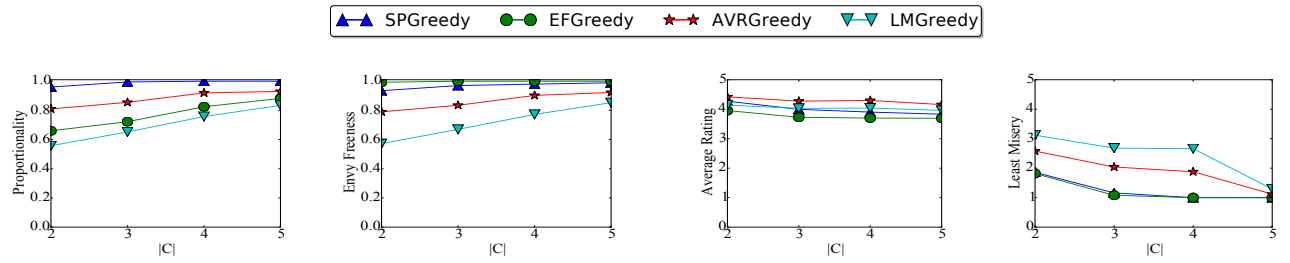


(a) proportionality (b) average rating

Figure 4: Performance comparison, varying coverage, arithmetic weighting

5.2.3 *Including category constraints*

We then study the case of including category constraints in the single proportionality maximization. Figure 5 shows the performance of the algorithms as a function of the number of categories, which is also the size of the package. The general trends we observed in the case of single proportionality with no categories still hold, but the overall numbers are lower. The differences between the algorithms in terms of single proportionality becomes clearer as the number of categories grows. The EFGreedy algorithm is now the worst in



(a) proportionality, varying $|\mathcal{C}|$ (b) envy-freeness, varying $|\mathcal{C}|$ (c) average rating, varying $|\mathcal{C}|$ (d) least misery, varying $|\mathcal{C}|$

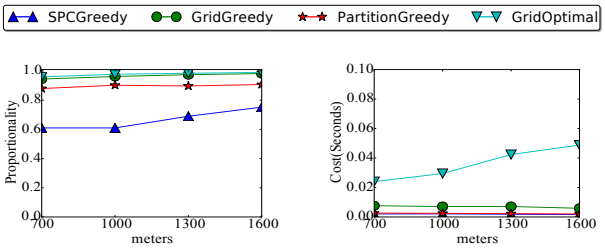
Figure 5: Performance comparison, single-coverage, with categories

terms of single proportionality and average rating, indicating that when trying to eliminate envy, this results in fewer users having a high quality package. Interestingly, the differences in the least misery metric become less pronounced when having large number of categories.

5.2.4 *Including distance constraints*

Finally, we study the case of including distance constraints. We consider the single proportionality case, and we also add the category constraints to make the recommendation scenario more realistic. We consider the three algorithms we described in Section 4.2.2 and the modified SPCGreedy algorithm that filters the candidate set based on the distance constraints. We study the single proportionality metric and the running time of the algorithms as a function of the distance threshold ϵ. We vary ϵ to be between 700 and 1600 meters.

The results are shown in Figure 6. We can see that simply enforcing the distance constraints as a filtering step on the SPCGreedy algorithm results in very poor performance. This is due to the fact that in 25% of the cases the SPCGreedy algorithm is not able to find a solution. Of the remaining algorithms, GridGreedy achieves performance close to that of GridOptimal with PartitionGreedy following right after. In terms of running time, PartitionGreedy is very close to simple SPCGreedy, followed by GridGreedy. We conclude that GridGreedy and PartitionGreedy offer the best compromise between performance and CPU cost, with GridGreedy giving a little better performance, and PartitionGreedy being a little faster. This is expected, since GridGreedy examines a bigger candidate set. It can thus construct a slightly better package, at the expense of slightly higher running time.



(a) proportionality (b) CPU cost

Figure 6: Performance comparison with distance constraints, varying ϵ

5.3 User Study

We conducted a user study with 10 participants (students) to test the effectiveness of different models in terms of finding fair packages. We used a movie dataset for this task, since it is easier to have users evaluate movies than venues. First, we asked each participant to rate 70 popular movies belonging to 5 different genres (action, animation, comedy, romance, thriller). The participants were divided into 4 groups of 2-4 users each (there were overlaps among groups). For each group, movie packages with 2-4 genres were generated using AVRGreedy, LMGreedy, SPGreedy and Random. The setup corresponds to the case of single coverage with category constraints. We asked each group to assess the (single) proportionality of the created packages: for each package, the group would discuss together how many members were satisfied with the items in the package, and rate the package accordingly, with an overall score in [0,1]. To avoid any bias, we did not provide any information on how the packages were generated and presented them to the groups in random order.

Table 1 shows the average of the proportionality values given by the users. The first observation is that SPGreedy performs the best, validating our approach. Interestingly, the second best algorithm is LMGreedy. This indicates that the user perception of fairness is guided primarily by the satisfaction they experience from the items in the package that they like, but, secondarily, to a great extend by the dissatisfaction they experience by items that they do not like. We plan to incorporate these considerations into our fairness definition in the future.

Table 1: User Study

	Random	AVRGreedy	LMGreedy	SPCGreedy
Proportionality	0.61	0.77	0.79	0.83

6. CONCLUSIONS

In this paper, we studied the problem of fairness in package-to-group recommendations. We introduced two definitions of fairness, based on proportionality and envy-freeness. We extended the definitions to consider category and distance constraints for the items that can be included in a recommended package. We proposed algorithms for all problem variants. Our experimental results on real data show that the recommended packages are superior in terms of fairness compared to alternative selection approaches based on best average rating or least misery, while maintaining high quality in terms of average rating.

Acknowledgements

This work was supported by grant 17205015 from Hong Kong RGC and by Marie Curie Reintegration Grant projects titled JMUGCS and LBSKQ which have received research funding from the European Union.

7. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.

[2] S. Amer-Yahia, F. Bonchi, C. Castillo, E. Feuerstein, I. Méndez-Díaz, and P. Zabala. Composite retrieval of diverse and complementary bundles. *IEEE TKDE*, 26(11):2662–2675, 2014.

[3] S. Amer-Yahia, S. B. Roy, A. Chawla, G. Das, and C. Yu. Group recommendation: Semantics and efficiency. *PVLDB*, 2(1):754–765, 2009.

[4] S. J. Brams. Fair division. In *Computational Complexity*, pages 1073–1080. Springer, 2012.

[5] C. Chekuri and A. Kumar. *Maximum Coverage Problem with Group Budget Constraints and Applications*, pages 72–83. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[6] T. Deng, W. Fan, and F. Geerts. On the complexity of package recommendation problems. *SIAM J. Comput.*, 42(5):1940–1986, 2013.

[7] J. Gorla, N. Lathia, S. Robertson, and J. Wang. Probabilistic group recommendation via information matching. In *WWW*, pages 495–504, 2013.

[8] N. G. Hall and D. S. Hochbaum. A fast approximation algorithm for the multicovering problem. *Discrete Applied Mathematics*, 15(1):35 – 40, 1986.

[9] D. K. Herreiner and C. D. Puppe. Envy freeness in experimental fair division problems. *Theory and Decision*, 67:65–100, 2009.

[10] D. S. Hochbaum. Approximation algorithms for np-hard problems. *SIGACT News*, 28(2):40–52, June 1997.

[11] Q. Hua, Y. Wang, D. Yu, and F. C. M. Lau. Set multi-covering via inclusion-exclusion. *Theor. Comput. Sci.*, 410(38-40):3882–3892, 2009.

[12] Q. Hua, D. Yu, F. C. M. Lau, and Y. Wang. Exact algorithms for set multicover and multiset multicover problems. In *ISAAC*, pages 34–44, 2009.

[13] A. Jameson and B. Smyth. Recommendation to groups. In *The Adaptive Web, Methods and Strategies of Web Personalization*, pages 596–627, 2007.

[14] K. Li, W. Lu, S. Bhagat, L. V. S. Lakshmanan, and C. Yu. On social event organization. In *KDD*, pages 1206–1215, 2014.

[15] X. Liu, Y. Tian, M. Ye, and W.-C. Lee. Exploring personal impact for group recommendation. In *CIKM*, pages 674–683, 2012.

[16] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl. Polylens: A recommender system for groups of user. In *ECSCW*, pages 199–218, 2001.

[17] A. G. Parameswaran, P. Venetis, and H. Garcia-Molina. Recommendation systems with complex constraints: A course recommendation perspective. *ACM TOIS*, 29(4):20, 2011.

[18] S. Qi, N. Mamoulis, E. Pitoura, and P. Tsaparas. Recommending packages to groups. In *International Conference on Data Mining (ICDM)*, 2016.

[19] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors. *Recommender Systems Handbook*. Springer, 2011.

[20] S. B. Roy, S. Thirumuruganathan, S. Amer-Yahia, G. Das, and C. Yu. Exploiting group recommendation functions for flexible preferences. In *ICDE*, pages 412–423, 2014.

[21] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295, 2001.

[22] H. Steinhaus. The problem of fair division. *Econometrica*, 16(1):101–104, 1948.

[23] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:4:2–4:2, Jan. 2009.

[24] P. Tsaparas, A. Ntoulas, and E. Terzi. Selecting a comprehensive set of reviews. In *KDD*, pages 168–176, 2011.

[25] S. Vargas, P. Castells, and D. Vallet. Intent-oriented diversity in recommender systems. In *SIGIR*, pages 1211–1212, 2011.

[26] M. Xie, L. V. S. Lakshmanan, and P. T. Wood. Breaking out of the box of recommendations: from items to packages. In *RecSys*, pages 151–158, 2010.

[27] M. Xie, L. V. S. Lakshmanan, and P. T. Wood. Generating top-k packages via preference elicitation. *PVLDB*, 7(14):1941–1952, 2014.

[28] C. Yu, L. V. S. Lakshmanan, and S. Amer-Yahia. Recommendation diversification using explanations. In *ICDE*, pages 1299–1302, 2009.

[29] Q. Yuan, G. Cong, and C.-Y. Lin. Com: A generative model for group recommendation. In *KDD*, pages 163–172, 2014.

[30] T. Zhu, P. Harrington, J. Li, and L. Tang. Bundle recommendation in ecommerce. In *SIGIR*, pages 657–666, 2014.

[31] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *WWW*, pages 22–32, 2005.