Fairness in Package-to-Group Recommendations

A Thesis

submitted to the designated
by the General Assembly of Special Composition
of the Department of Computer Science and Engineering
Examination Committee

by

Dimitrios Serbos

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

WITH SPECIALIZATION
IN SOFTWARE

University of Ioannina

January 2017

Table of Contents

List of Figures

List of Tables

List of Algorithms

Abstract

Dimitrios Serbos , M.Sc. in Computer Science, Department of Computer Science and Engineering, University of Ioannina, Greece, January 2017.
Fairness in Package-to-Group Recommendations.
Advisor: Nikos Mamoulis, Associate Professor.

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 thesis, 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. We call the first one *proportionality*, where each user in the group must have at least a number m of items ranked in the top $\Delta\%$ of his/her preferences. We call the second alternative *envy-freeness* ,where each user must have at least m items in package, where he/she is ranked at the top $\Delta\%$ of the users that rated the items. We mostly use $m = 1$ and call these cases single proportionality and single envy-freeness respectively.We also explore cases for $m > 1$. We show 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 (single coverage for $m = 1$ and multi-coverage for $m > 1$), 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 candidate packages for recommendation. We evaluate the appropriateness of the fairness models and the performance of the proposed algorithms using real data from Yelp, and a user study.

Εκτεταμενη Περιληψη

Σέρμπος Δημήτριος, Μ.Δ.Ε. στην Πληροφορική, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Ιανουάριος 2017.
Δικαιότητα σε προτάσεις πακέτο-σε-ομάδα .
Επιβλέπων: Νίκος Μαμουλής, Αναπληρωτής Καθηγητής.

Η σύσταση πακέτων με πολλαπλά αντικείμενα σε ομάδες από χρήστες έχει πολλαπλές εφαρμογές, συμπεριλαμβανομένου την πρόταση πακέτου διακοπών σε ομάδα από τουρίστες, πακέτου διασκέδασης σε ομάδα απο φίλους ή πακέτου μαθημάτων σε ομάδα από σπουδαστές. Σε αυτή τη πτυχιακή, επικεντρωνόμαστε σε μια πολύ σημαντική πτυχή του προβλήματος συστάσεων πακέτων σε ομάδα χρηστών, αυτή της δικαιότητας. Συγκεκριμένα όταν προτείνουμε ένα πακέτο σε μια ομάδα χρηστών, εξετάζουμε αν αυτό είναι δίκαιο για όλους τους χρήστες, με την έννοια ότι κάθε μέλος της ομάδας είναι ικανοποιημένο από επαρκή αριθμό αντικειμένων στο πακέτο. Μελετάμε δύο ορισμούς για τη δικαιότητα. Ονομάζουμε τον πρώτο απο αυτούς αναλογικότητα, όπου κάθε χρήστης πρέπει να έχει στο πακέτο έναν αριθμό απο τουλάχιστον m αντικείμενα τα οποία να βρίσκονται στα κορυφαία $\Delta\%$ των προτιμήσεων του. Ονομάζουμε τη δεύτερη εναλλακτικη χωρίς-ζήλια ,όπου κάθε χρήστης πρέπει να έχει τουλάχιστον m αντικείμενα στο πακέτο, στα οποία να βρίσκεται στους κορυφαίους $\Delta\%$ από τους χρήστες που τα βαθμολόγησαν. Κυρίως χρησιμοποιούμε $m = 1$ και καλούμε αυτές τις περιπτώσεις απλή αναλογικότητα και απλή χωρίς-ζήλια, αντίστοιχα. Επιπρόσθετα, μελετάμε και περιπτώσεις όπου $m > 1$. Δείχνουμε ότι και για τους δύο ορισμούς, το πρόβλημα είναι NP-Hard. Εκμεταλλευόμαστε το γεγονός ότι μπορεί να μοντελοποιηθεί σαν ένα πρόβλημα κάλυψης (απλής κάλυψης για $m = 1$ και πολλαπλής κάλυψης για $m > 1$), και προτείνουμε άπληστους αλγόριθμους που βρίσκουν προσεγγιστικές λύσεις σε λογικούς χρόνους. Επιπλέον, μελετάμε δύο επεκτάσεις του προβλήματος, όπου επιβάλλουμε περιορισμούς με κατηγορίες και χωρικούς περιορισμούς στα αντικείμενα που θα

συμπεριληφθούν στο προς πρόταση πακέτο. Τέλος αξιολογούμε την καταλληλότητα του μοντέλου δικαιότητας και την επίδοση των προτεινόμενων αλγορίθμων, με πραγματικά δεδομένα απο το Yelp και με μια μελέτη χρηστών.

Chapter 1

Introduction

1.1 Introduction

Recommender systems have attracted a lot of research attention and have been deployed in a wide range of applications [1, 2]. 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 [3, 4], or an item to a group of users [5, 6]. In a recent work the problem of recommending a package to a group of users was addressed [7]. 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 according to the preferences of the conference attendees, and many more.

In this thesis, 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 potentially 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 [8, 7], that tries to find a consensus among the users. However, even the best items according to this function may still leave users feeling dissatisfied and slighted. We can address this problem directly when recommending a package of items to the group. For each user in the group, we can aim to include an item in the package that matches her preferences, even if this 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) [9]. 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 resource allocation problems [10, 11].

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 model this problem as a classic set coverage problem [12]. 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 $\mathcal{I}$ are divided into 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 thesis 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 for them.
- We evaluate the proposed models and algorithms via a user study and experiments on real data from Yelp.

1.2 Thesis structure

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

Chapter 2

Related work

2.1 Related Work

In this section we review some related work on recommender systems and fairness.

2.1.1 Recommender Systems

Recommender systems have attracted extensive research attention and have been deployed in a wide range of applications [13]. Their use is to provide recommendations of items to a user by predicting unknown utilities of items (metric that represents how much a user likes an item e.g. rating) from known ones. That extrapolation is usually done by specifying heuristics that define a utility function and empirically validating its performance and estimating the utility function that optimizes certain performance criteria such as the mean square error. They are usually classified in three distinct categories based on the way the recommendations are generated. Those 3 categories are:

- **Content-based recommendations** where the user is recommended items similar to those he/she preferred at the past. At these approaches each item's utility is predicted based on the already established knowledge provided by the user, in other words known utilities of items already viewed and/or rated by the user.

Each item is associated with an profile which is usually a vector of attributes characterized it. Also a profile is built for the user by using (e.g. averaging) the vectors of the items already rated, resulting in a user vector. Those two vectors can be then compared and a similarity metric (e.g.cosine similarity) can be computed to give a prediction about a specific item. Then the most similar or the top-k similar items can be recommended to the user. Other methods can also be used instead of predicting the missing utility via a heuristic, that calculate the possibility that an item belongs to a class with similar characteristics such as Bayesian classification and clustering.

- **Collaborative recommendations** where the user is recommended items that are liked by other similar users. Those approaches use the known ratings of the users to form vectors. Then a similarity metric (e.g cosine similarity or Pearson correlation) is used to compute the similarity between users. Finally a function is used to predict a rating of an unknown item for a given user, by aggregating (common methods are simple average ,weighted sum etc) the known ratings of other similar users.

- **Hybrid approaches** where both aforementioned methods are combined.

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 best package for a group of users.

The work most closely related to ours is the recent work in [7], 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 tangentially addressed, but the main objective of the work in [7] 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 [8] or aggregate the items recommended to each group member [14]. Recent works consider more factors into the model, e.g., agreement [15] or social relationships [16] among group members, feedback from users [6], and personal impact of members [17, 5]. In addition, more models based on probability [18] and topic modeling [17, 5] are also proposed and

Table 2.1: rating example

u/i	u_1	u_2
i_1	1	5
i_2	1	4
i_3	3	1

studied. These works consider the notion of fairness only indirectly, through the choice of the aggregation function. There is no explicit modeling of fairness, and no algorithm for guaranteeing fairness.

The problem of recommending a package of items to a single user has also been studied extensively. In [3] they show that several of the problem variants are NP-hard; the complexity can be reduced by user-defined constraints (e.g. [19]). The problem has also been considered under budget constraints [20, 21], as a learning problem for predicting the package interestingness [4], or as a profit maximization problem [22]. There is no notion of fairness in this case, since there is a single user to recommend to.

Here we show a simple example where a simple aggregation function (average rating) will fail to give a fair package to two distinct users. Suppose there are two users u_1 and u_2 and 3 items i_1, i_2 and i_3. The users rated the 3 items as shown in table 2.1. It is obvious that if we would want to recommend a package with 2 items- an algorithm that uses the average rating would give a package consisting of items i_1 and i_2. It becomes obvious that the package is not fair for u_1, since there are no items in it, that u_1 particularly likes. A package consisting of i_1 and i_3 would be intuitively, much fairer for both users, since they both have an item they like/rated highly in the package .

2.1.2 Fairness

The fairness problem we propose in this thesis is unique to package-to-group recommendations. Nevertheless, it is related to the least misery (LM) approach, the result diversification in recommender systems, and the fair division problem in economics.

Least Misery. Previous work (e.g., [15]) has proposed a least-misery (LM) approach to the item to group recommendations problem that recommends the item that min-

Table 2.2: rating example 2

u/i	u_1	u_2
i_1	2	5
i_2	2	4
i_3	4	1

imizes the dissatisfaction of the group members. Intuitively, least misery aims at minimizing dissatisfaction of individual group members due to the presence in the package of items they do not like, whereas fairness aims at maximizing 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.

Similar to what we did with average rating ,we show a simple example where least misery fails to give a fair solution.Suppose again we have two users and 3 items- and their respective ratings can be summarized in Table 2.2. The least misery approach will propose a package consisting of items i_1 and i_2 while a package with i_1 and i_3 would be a much fairer solution.

Result Diversification. It has been shown that result diversification is an effective direction to improve recommendation quality [23, 24, 25]. Various studies have tried to diversify the recommended items with respect to topics [23], explanations [24] and user interest [25], etc. 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.

Fair Division. Our proportionality and envy-freeness definitions of fairness are inspired from the problem of fair division in Economics [10, 11], 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.

Chapter 3

Model

3.1 Problem Definition

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

3.1.1 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 [9] 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 $P \subseteq \mathcal{I}$, with size K. In this thesis, 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 3.1. For a user u, and a package P, we say that P is m-proportional for u, if there exist at least $m \geq 1$ 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 3.2 (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

$$F_{\text{prop}}(U, P) = \frac{|G_p|}{|G|}, \tag{3.1}$$

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 3.3. For a user u, a package P, and a group G, we say that the package P is m-envy-free for u, 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 3.4 (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

$$F_{\text{ef}}(G, P) = \frac{|G_{\text{ef}}|}{|G|}, \tag{3.2}$$

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

3.1.2 Fairness Maximization

We now define the basic fairness maximization problem that we consider in this thesis. 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.*

3.1.3 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.

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, 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 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.

Distance Constraints

In some applications, the package selection is constrained by the allowable 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 possible to use.

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 $v_j, v_l \in P$, $dist(v_j, v_l) \leq \epsilon$*

Chapter 4

Algorithms

4.1 Algorithms

In this section, we study the complexity and propose algorithms for the fairness maximization problems we defined in Section 3.1.

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 $\textsc{Sat}_G(i) \subseteq G$ 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, $\textsc{Sat}_G(i)$ contains the users that "like" the item i, i.e., the users for which item i belongs in their top-Δ% most preferable items. For envy-freeness, $\textsc{Sat}_G(i)$ 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 $u \in \textsc{Sat}_G(i)$.

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

4.1.1 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 $\textsc{Sat}_G(i)$, 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 $\textsc{Sat}_G(i)$. We say that in this case the item i *covers* the users in $\textsc{Sat}_G(i)$. Given a package P, the set of users for which the package P is fair is $\cup_{i \in P} \textsc{Sat}_G(i)$. 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 $1 - 1/e$ [26], 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 $\textsc{Sat}_G(P)$ denote the users covered (satisfied) by the package P. We define the utility of adding item i to package P as $f_G(P, i) = |\textsc{Sat}_G(P \cup \{i\}) \setminus \textsc{Sat}_G(P)|$. 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 4.1. 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.

Fairness maximization for $m > 1$. We refer to this case as the *multi-coverage* case. In this case, we require that a node is satisfied by m items in order to be considered

Algorithm 4.1 Greedy Fairness Maximization

Input:Group of users G, items $\mathcal{I}$, value K

Output:Package P

$P \leftarrow \emptyset$
Candidates $\leftarrow \mathcal{I}$
$J = 1$
while $J \leq K$ **do**
  $i \leftarrow \arg\max_{i \in \text{Candidates}} f_G(P, i)$
  $P \leftarrow P \bigcup \{i\}$
  Candidates $\leftarrow$ Candidates $\setminus \{i\}$
  $J = J + 1$
end while
return P

covered. This corresponds to a *multi-cover* of the set G. The problem of finding the minimum multi-cover has been studied extensively [27, 12, 28], but the problem of maximum multi-coverage is not as well understood. In [29], 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 $\textsc{Sat}_G^j(P)$ 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

$$f_G(P, i) = \sum_{j=1}^{m} w_j |\textsc{Sat}_G^j(P \cup \{i\}) \setminus \textsc{Sat}_G^j(P)| \tag{4.1}$$

that is, the weighted sum of the users that are satisfied an additional time. The weights w_j control the importance of covering a node 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.

4.1.2 Fairness Maximization with Constraints

We now consider algorithms for the two problems we defined in Section 3.1.3.

Category constraints

We now consider Problem 2, where the items are partitioned in 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 4.1.1. The only difference from before 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.

Lemma 4.3. *The Greedy 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 $L_k = \textsc{Sat}_G(P_k)$ and $L_k^* = \textsc{Sat}_G(P_k^*)$ 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 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 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. Once an item is selected from a category C_j, we exclude one of the replicas. Note that the same item cannot be selected multiple times since it will have coverage zero.

Distance constraints

We now consider Problem 3, where we want the items we include in the package to satisfy distance constraints. In this case, we cannot prove any guarantees for the Greedy algorithm. Actually, there are cases when the Greedy algorithm may not find a valid solution, and terminate before finding K items. We now propose two heuristic, and one exact algorithm that take advantage of spatial partitioning and indexing to reduce the search cost.

A space-partitioning approach. For this algorithm, we divide the space by a grid, such that each cell has width and height $\epsilon/\sqrt{2}$, i.e., a diagonal of length ϵ. The reason is that such a grid guarantees that for any items inside the same cell, they 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. 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 also partition the space by a grid, but this time each cell

has side length ϵ. Then, we run one instance of the Greedy algorithm starting from each cell and extending to neighboring cells if necessary. The item i within the current cell is selected greedily, and then we take advantage of the grid to reduce the number of candidate items to include in the same package as i. Figure 4.1 shows an example with a 4×4 grid. We shall refer to a grid cell j as L_j. For any L_j, it is easy to see that its items can only form valid packages w.r.t. ϵ-distance constraint with items inside itself or its direct neighbors (9 cells in total). Observe that the 9-cell search space can be gradually reduced as more items are selected. For example, suppose the Greedy algorithm on cell L_6 selects i_1 as the first item, it will then consider as possible second item to add to the package only items that fall into one of the 9 cells surrounding L_6 (i.e., the cells $\{L_{1\text{–}3}, L_{5\text{–}7}, L_{9\text{–}11}\}$). If the second item selected is i_2, which falls in L_2, the third item cannot be selected from cells $L_{9\text{–}11}$ because all items in them are further than ϵ from i_2. Therefore, for the third item, we only have to consider cells $\{L_{1\text{–}3}, L_{5\text{–}7}\}$. If the third item is i_3 in L_7, the fourth item can only be selected from cells $\{L_{2\text{–}3}, L_{6\text{–}7}\}$, and so on. Therefore, having partitioned the items based on the grid, we can dynamically consider only a certain subset of cells for selecting the next item to include in the package. We will refer to this algorithm as GridGreedy.

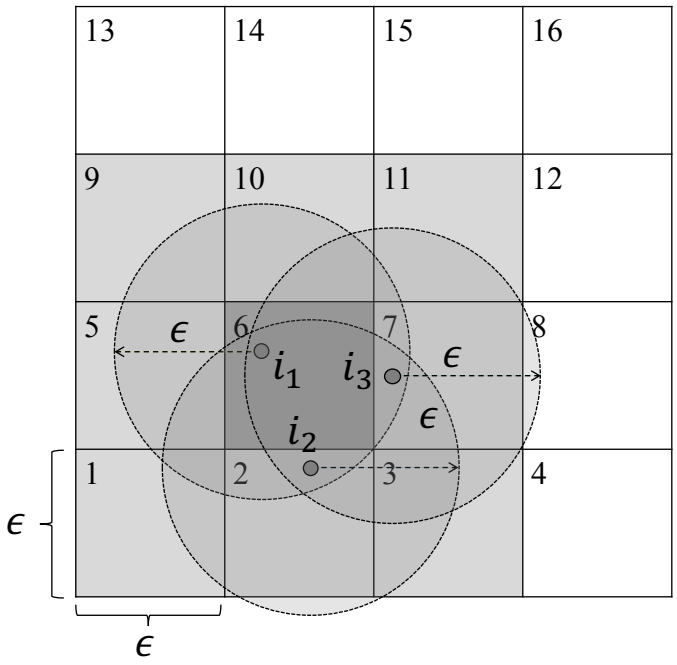


Figure 4.1: Grid Example

Grid-based backtracking algorithm. It is possible to change the GridGreedy algorithm to an exhaustive search optimal algorithm that generates all valid item combi-

nations 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 next 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.

Chapter 5

Experiments

5.1 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 input 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.

5.1.1 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 in Yelp the rating matrix is very sparse, we employ collaborative filtering (CF) [9] to fill the matrix as much as possible. In particular, we use Apache Mahout[2]

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

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. The packages for the groups are constructed with 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 less 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.These groups are the standard anti-correlated groups.

Additionally and in order to find cases that are even more difficult than the standard anti-correlated groups,we create groups that are dissimilar at the top Δ % of the users 's preferences.The procedure is similar to the previous case,but instead of using the Pearson Correlation coefficient between the users,we examine how dissimilar they are with respect to the set of items that can be found in the top Δ% of their preferences.We use $Delta = 1, 3, 5$ and create groups that we call disjoint 1,3,5 respectively.

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 $\Delta = 5\%$.

5.1.2 Algorithm Performance

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 algorithm described in Section 4.1 which selects items greedily to maximize the single proportionality metric of the package $F_{\text{prop}}(G, P)$.

EFGreedy: Select items greedily to maximize the envy-freeness metric of the package, $F_{\text{ef}}(G, P)$.

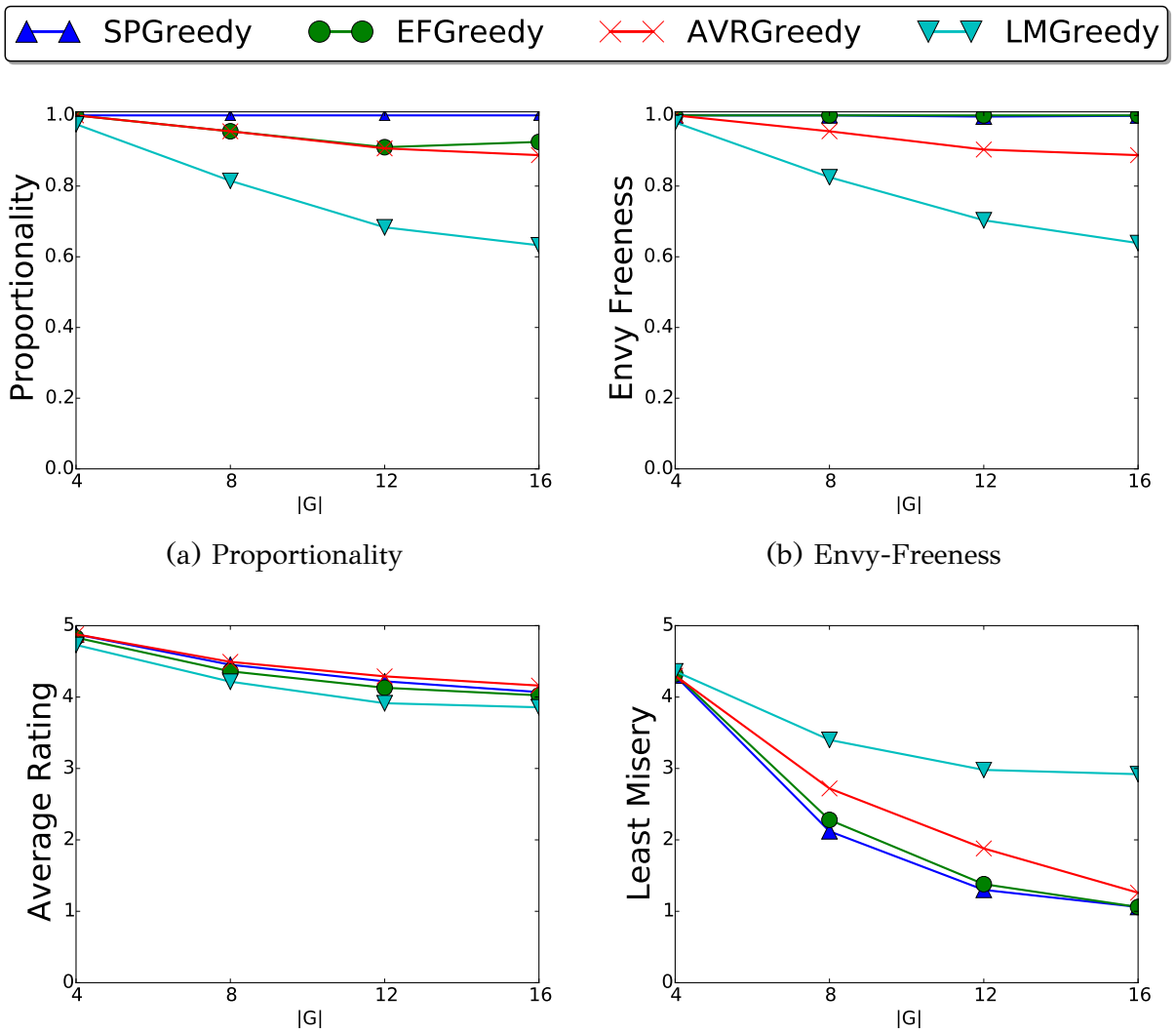
AVRGreedy: Select items greedily to maximize the average rating of the package

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

LMGreedy: Select items greedily to maximize the least misery metric of the package

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

Figure 5.1 shows the results of the four algorithms for anti-correlated-groups, for the four different metrics we consider, as a function of the size of the group while Figure 5.2 as a fuction of the package Size. 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 in 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. Of the two quality-oriented algorithms the AVRGreedy achieves the better fairness.



(a) Proportionality

(b) Envy-Freeness

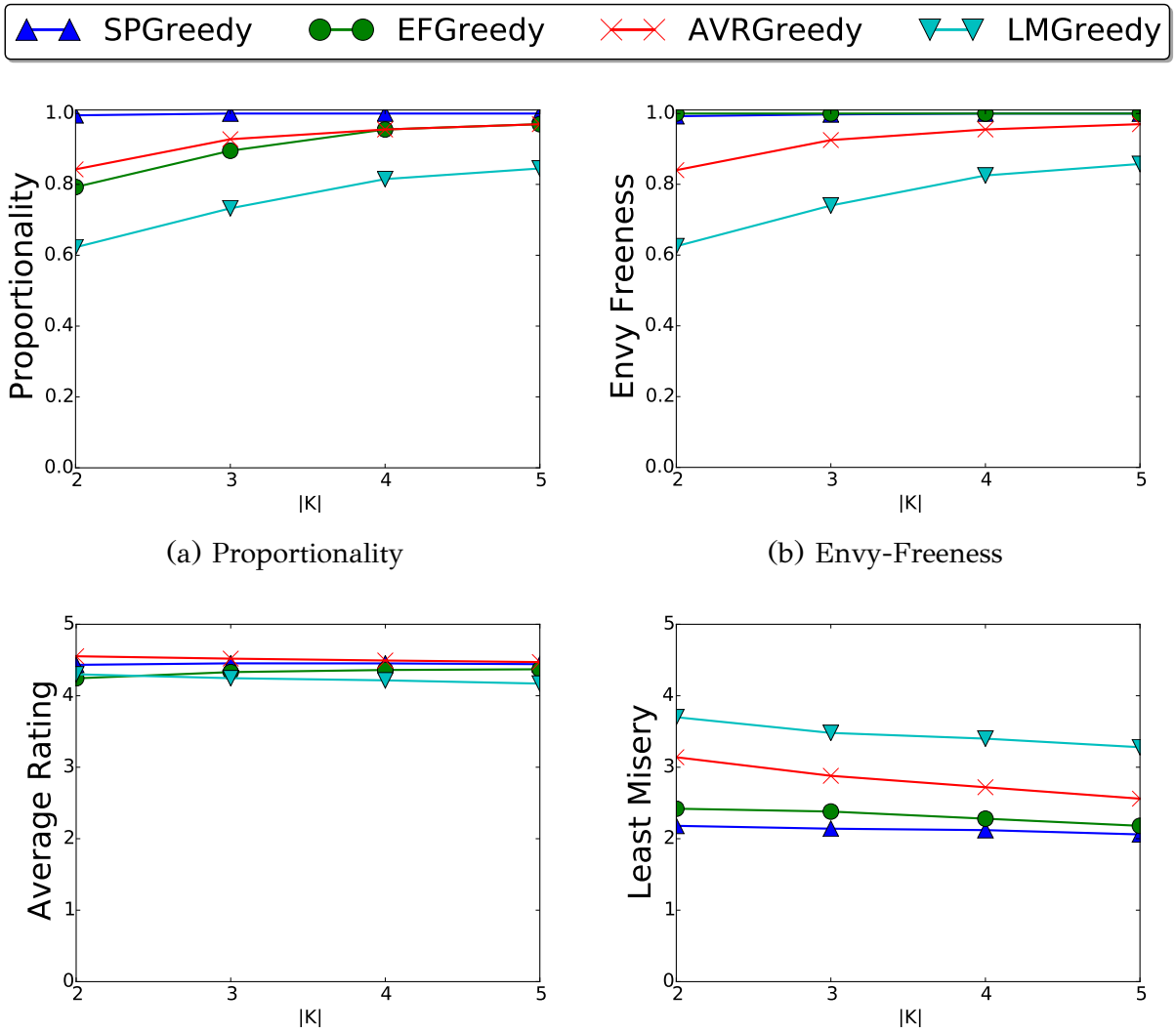
(c) Average Rating

(d) Least Misery

Figure 5.1: Performance comparison with varying group size, anti-correlated groups

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. The performance remains stable for $\Delta = 1\%$. 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 disjoint 5 and disjoint 1 groups.



(a) Proportionality

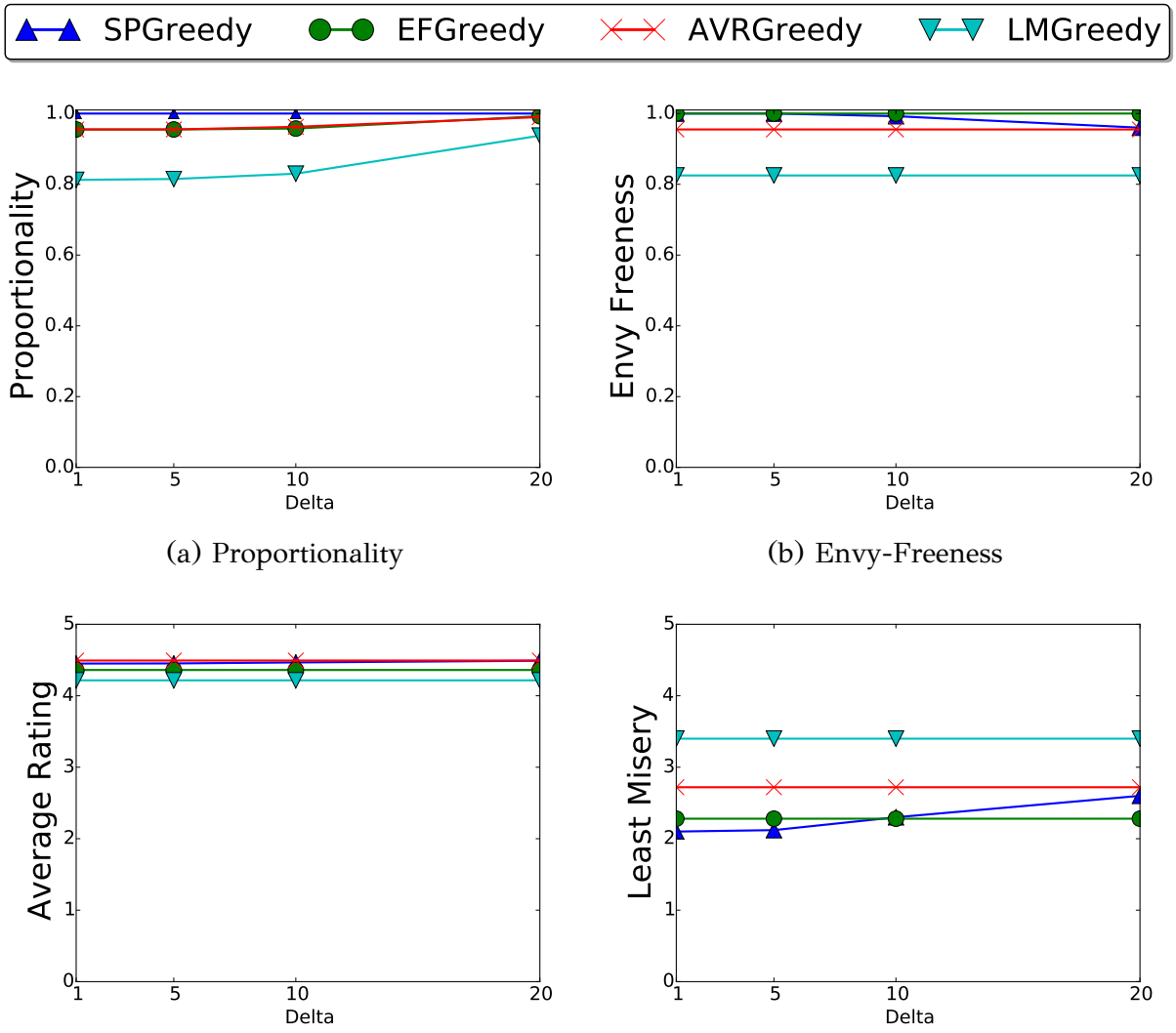
(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.2: Performance comparison with varying package size, anti-correlated groups

In Figures 5.4, 5.5 and 5.6 we test the performance of our algorithms for the disjoint 5 groups as input. The trends are similar to the cases with the anti-correlated groups but the performance difference between the two greedy algorithms and the other two is more prominent. Interestingly we observe that although the problem becames more difficult, SPGreedy and EFGreedy continued giving great results for their respective metrics- whereas AVRGreedy and LMGreedy 's performances- drop dramatically, resulting in more apparent differentiation. It is interesting to note that in Figure 5.6, we see a big spike in the proportionality metrics of AVRGreedy and LMGreedy, in the case where $\Delta = 10$.That can be explained by the fact that the dissimilarity between the users was maximized for the top$\Delta = 5\%$ of the items, when the groups were created, resulting in a much easier problem for $\Delta \geq 10$.



(a) Proportionality

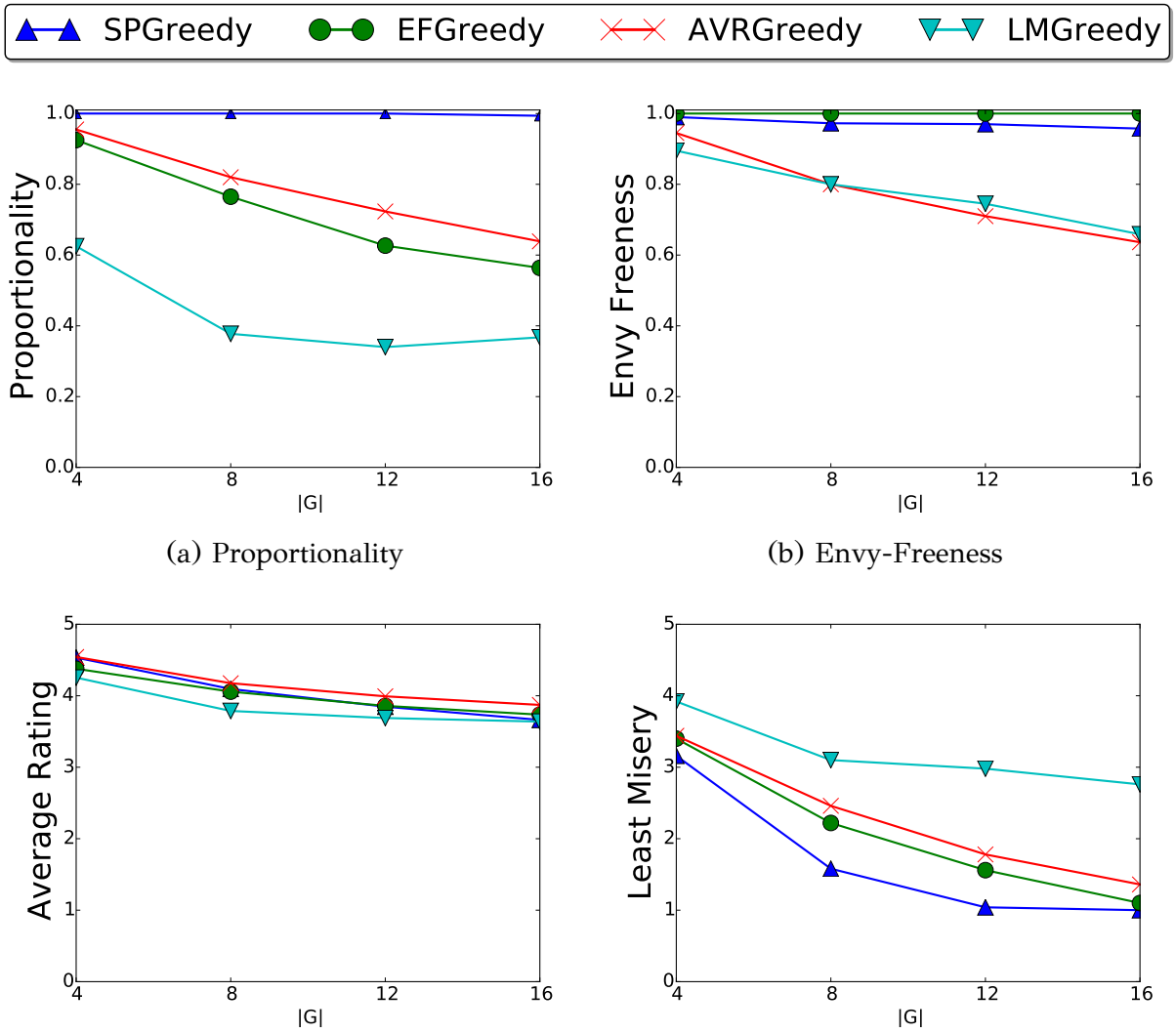
(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.3: Performance comparison for default values and varying delta, anti-correlated groups

In Figures 5.7, 5.8 and 5.9 we test the performance of our algorithms for the disjoint 1 groups as input. The trends are similar to the case with the anti-correlated groups as input. The difference between the performance of the two greedy algorithms , AVRGreedy and LMGreedy is still bigger than the case with anti-correlated groups, although smaller than the case where disjoint 5 groups were used as input groups. That can be explained by the fact that the disjoint 5 groups is a more difficult case than the disjoint 1 (the dissimilarity is maximized in the top5%) rather than the top1% of the users 's preferences and $\Delta = 5$ was used at run time for Figures 5.7 and 5.8). In figure 5.9 we now see that the spike is in the case for $\Delta = 5$. Also interestingly we can see that the envy-freeness of the SPGreedy drops in comparison



(a) Proportionality

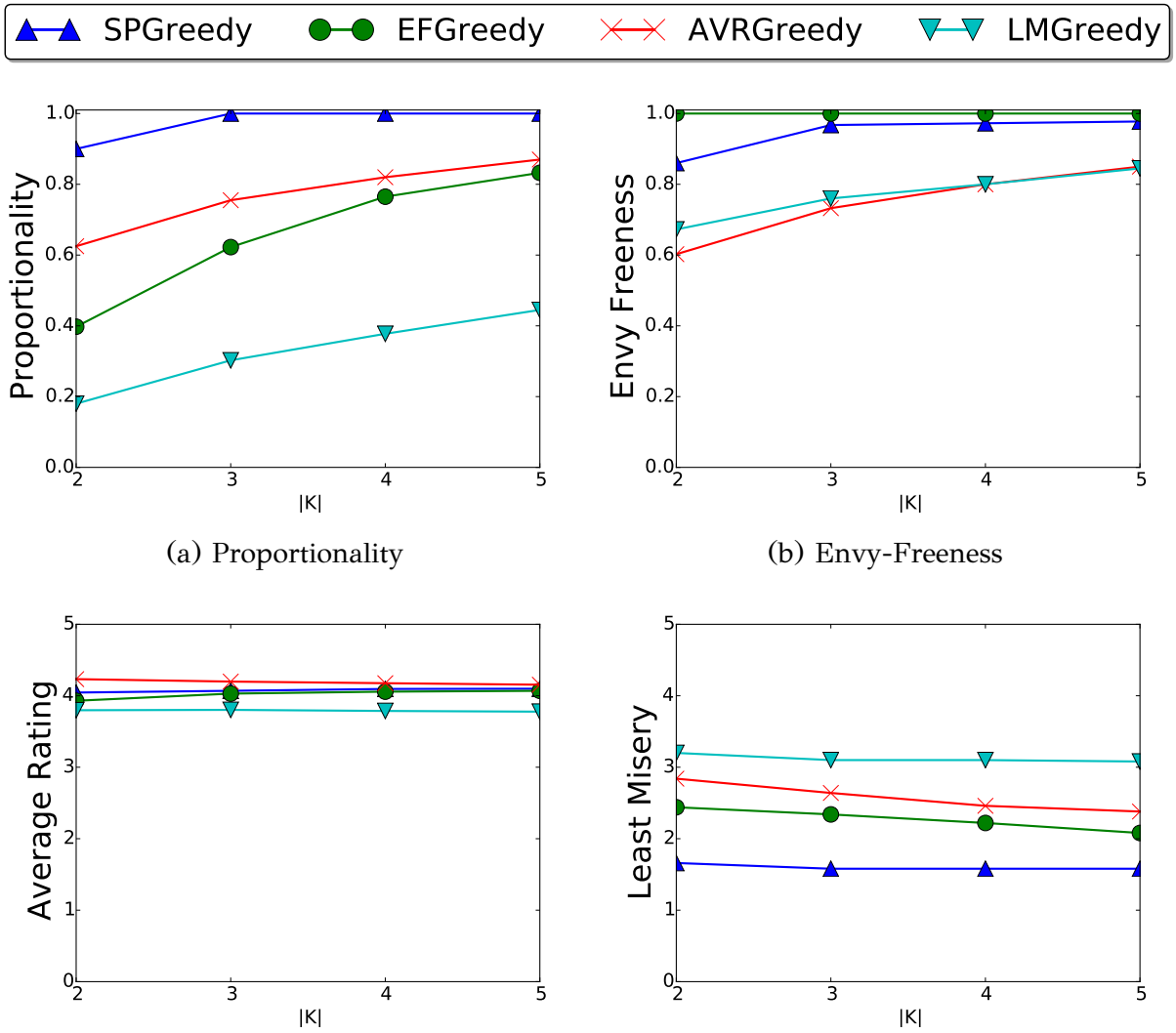
(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.4: Performance comparison with $\Delta = 5$ and varying group size, disjoint 5 groups

to previous cases, especially in cases with few users. To explain this ,we have to keep in mind that in cases with a few users and in order for a user to be envy-free, for $\Delta = 5$ the user needs to be the one with the best rating for a given item. Generally, the disjoint 1 set of groups are the ones where the top rated item between the users differ (they were created so that the difference in the top$\Delta = 1\%$ of the users preference is maximized).



(a) Proportionality

(b) Envy-Freeness

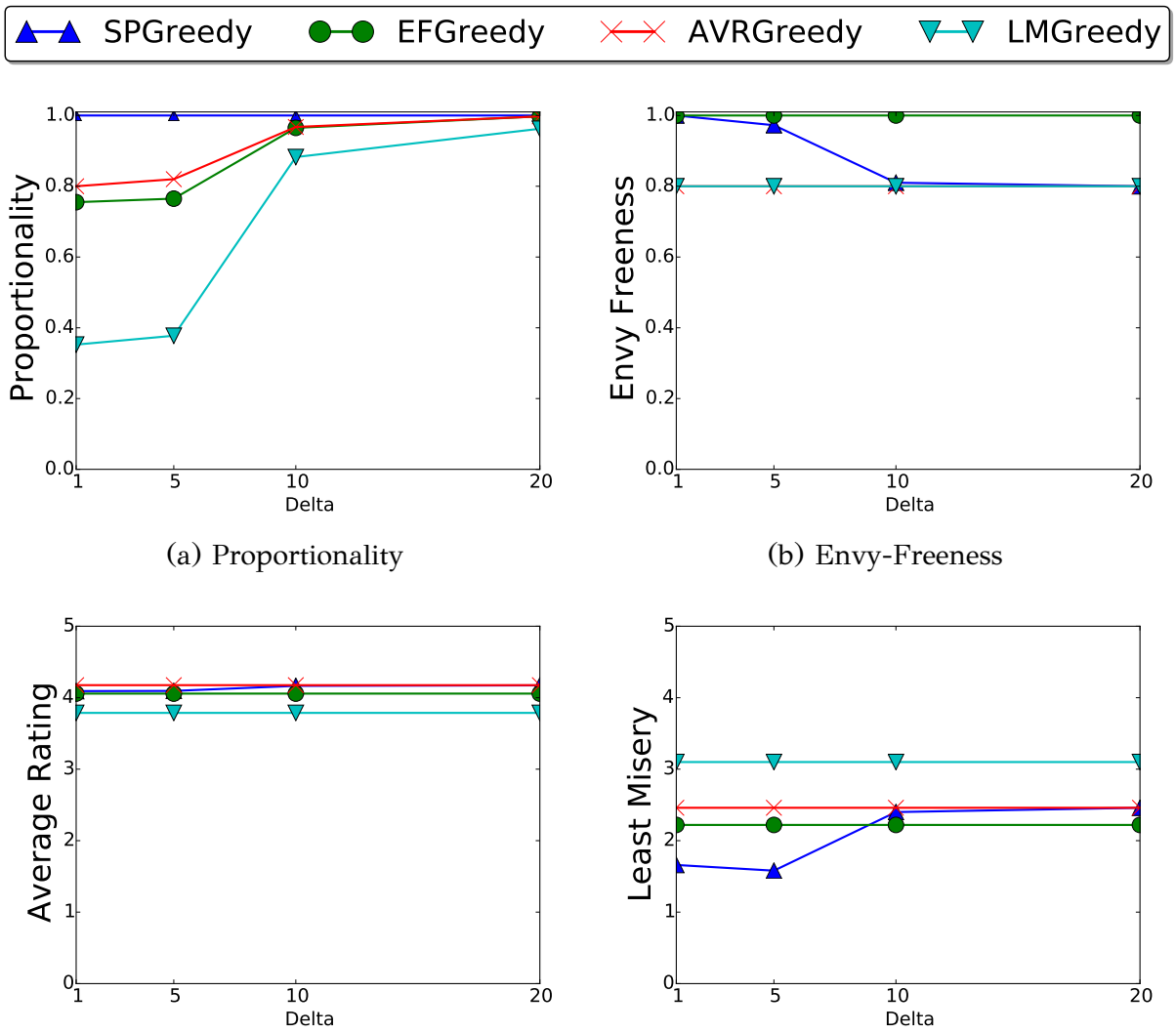
(c) Average Rating

(d) Least Misery

Figure 5.5: Performance comparison with $\Delta = 5$ and varying package size, disjoint 5 groups

Multi-proportionality

We now consider the problem of multi-proportionality fairness. We use the arithmetic weighting in the implementation of the MPGreedy algorithm. In Figures 5.10, 5.11, 5.12, 5.13, 5.14, 5.15 we present the performances of the four algorithms as a function to the group size and package size,for anti-correlated groups and with coverage requirement 2,3 and 4 respectively. MPGreedy and MEFGreedy clearly outperform the others on the metrics the are supposed to maximize. Obviously, 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. The same can be stated



(a) Proportionality

(b) Envy-Freeness

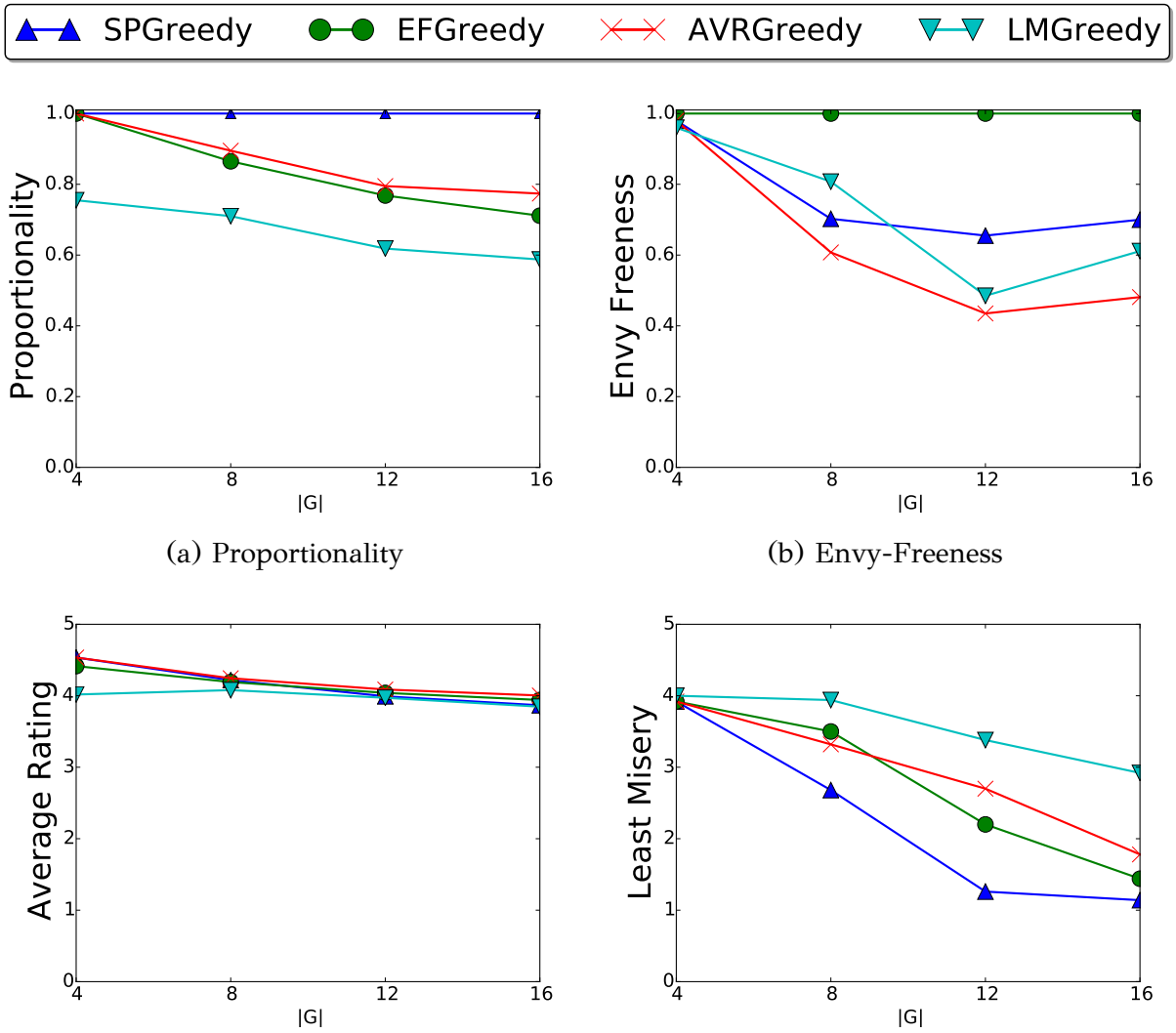
(c) Average Rating

(d) Least Misery

Figure 5.6: Performance comparison with default values and varying Δ , disjoint 5 groups

for MPGreedy and envy-freeness. Interestingly both greedy algorithms perform relatively well on the average rating metric again. Finally in plots where the coverage requirement is 3 and 4 and for package size<3 or <4 respectively, the proportionality and envy-freeness metrics are 0. Finally in 5.16, the same results are summarized for varying coverage.

We also present in Figures 5.17, 5.18, 5.19, the performance of the four algorithms as a function of the group size for the more difficult case of the disjoint 5 groups as input. The trends are similar, but with lower values for all algorithms on the proportionality metric, as expected. The Envy-freeness metric does not change a lot, since the disjoint 5 groups are basically made to make the problem more difficult



(a) Proportionality

(b) Envy-Freeness

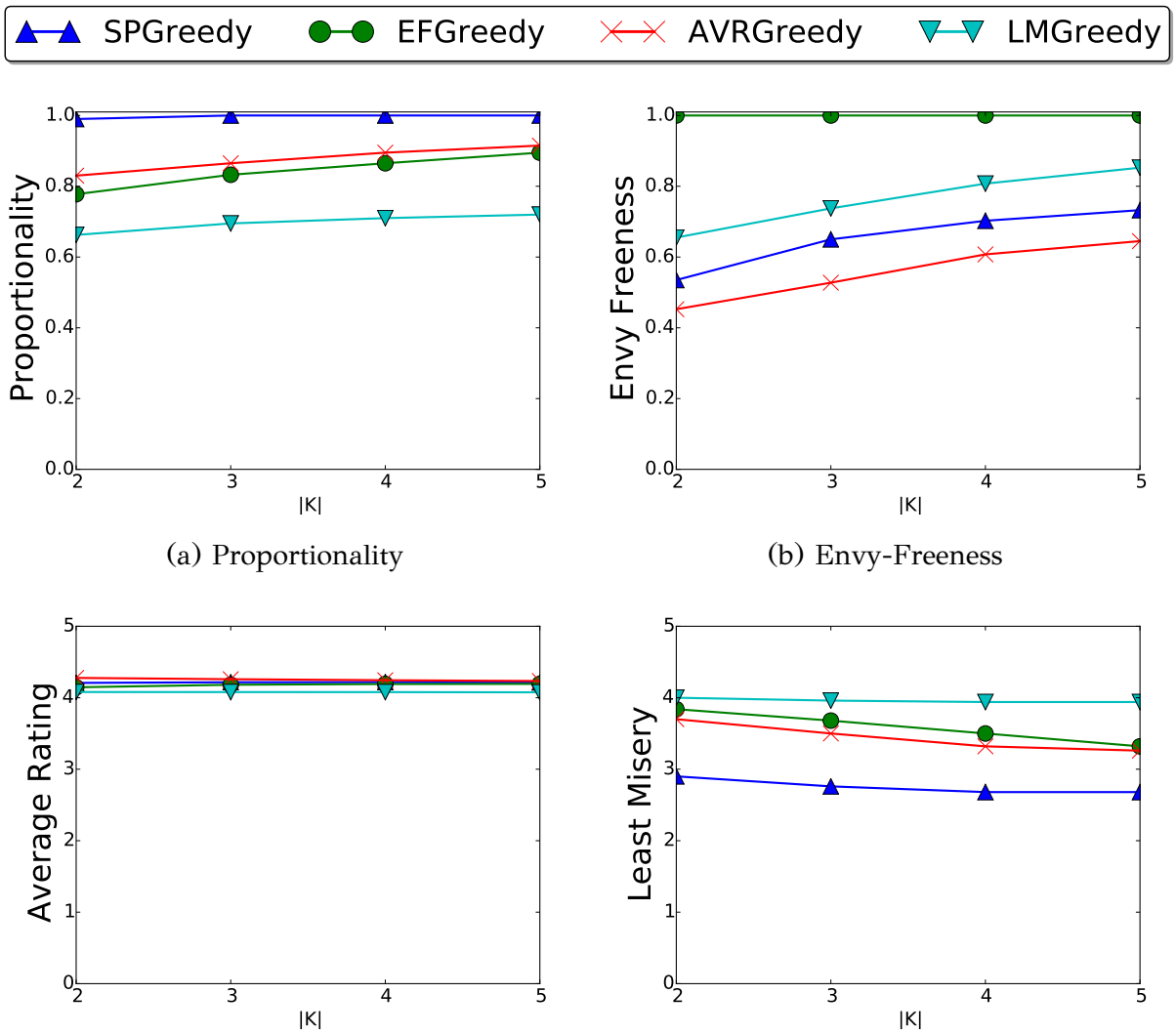
(c) Average Rating

(d) Least Misery

Figure 5.7: Performance comparison with $\Delta = 5$ and varying group size, disjoint 1 groups

with respect to the proportionality metric. The rest of the results (function of delta and disjoint 1 groups follow the expected trends and would present no new useful information).

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) Envy-Freeness

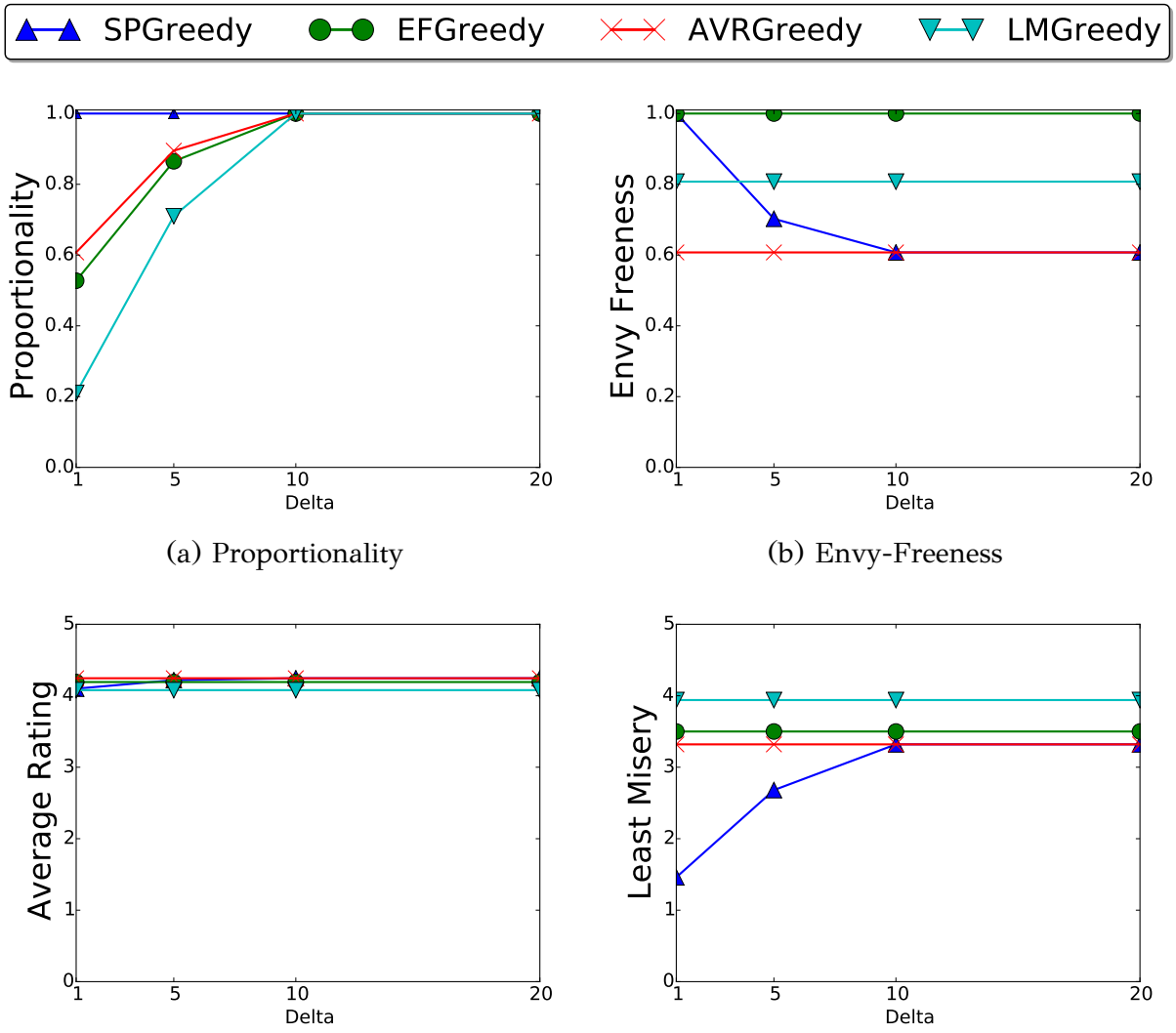
(c) Average Rating

(d) Least Misery

Figure 5.8: Performance comparison with $\Delta = 5$ and varying package size, disjoint 1 groups

Including category constraints

We then study the case of including category constraints in the single proportionality maximization. Figures 5.20 and 5.21 show the performance of the algorithms as a function of the number of users and the number of categories (which is also the size of the package)respectively for the anti-correlated groups case,while Figures 5.22 and 5.23 for the disjoint 5 groups case. The general trends we observed in the case of single proportionality with no categories still hold, but the overall numbers are lower. The separation of the algorithms in terms of single proportionality becomes clearer as the number of categories grows. The EFGreedy algorithm now performs a bit worse



(a) Proportionality

(b) Envy-Freeness

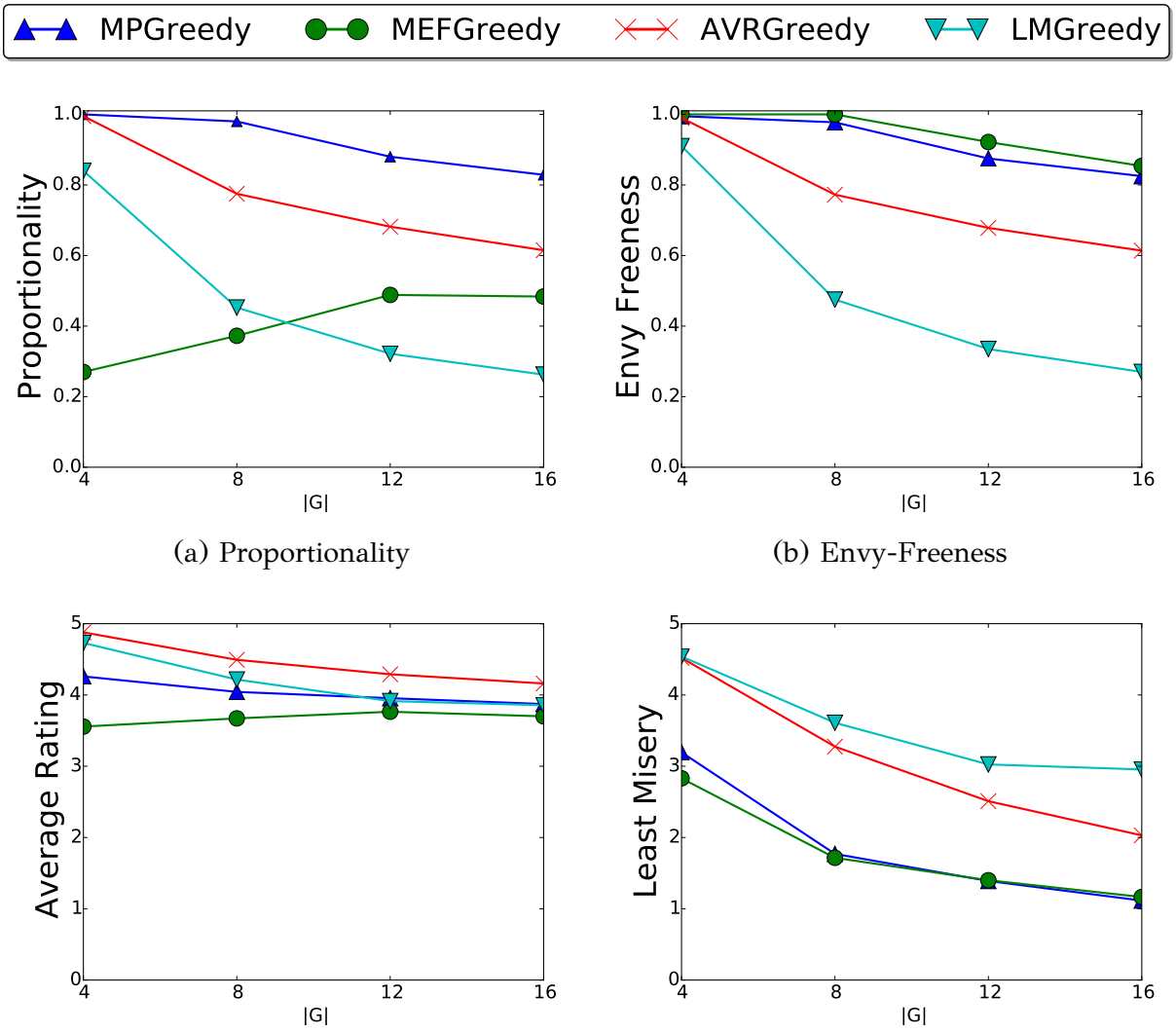
(c) Average Rating

(d) Least Misery

Figure 5.9: Performance comparison with default values and varying Δ , disjoint 1 groups

in terms of single proportionality , 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. The cases for varying Δ and for disjoint 1 groups were omitted because they would present no new interesting information.

Obviously we can incorporate category constraints to the multi-coverage problem. The results share the same pattern as the ones without constraints, but with lower values for all 4 metrics and for all four algorithms. To avoid repetition we present just the results for anti-correlated groups as a function to the group size,which can be seen in Figures 5.24, 5.25 and 5.26 .



(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

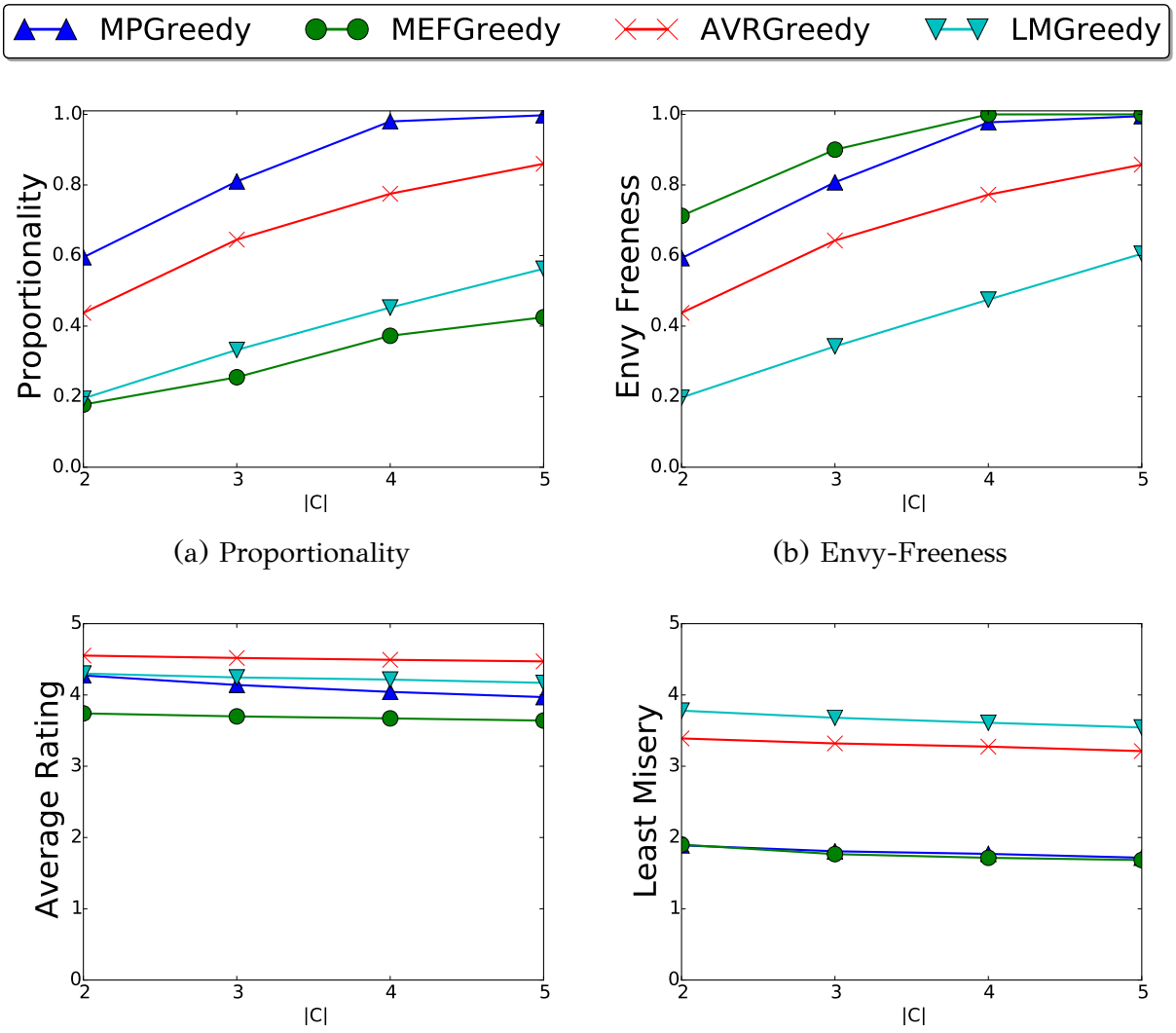
(d) Least Misery

Figure 5.10: Performance comparison with varying group size, 2 coverage, anti-correlated groups

Including distance constraints

Finally, we study the case of including distance constraints. For this task we again use the Yelp dataset for which we have the location of the venues, and we consider the single proportionality case for the anti-correlated groups. We also add the category constraints to make the recommendation scenario more realistic. We study the single proportionality metric achieved by the different algorithms that enforce the distance constraints, as well as their running time as a function of the distance threshold ϵ. We vary ϵ to be between 700 and 1600 meters.

The results are shown in Figure 5.27. We can see that simply enforcing the dis-



(a) Proportionality

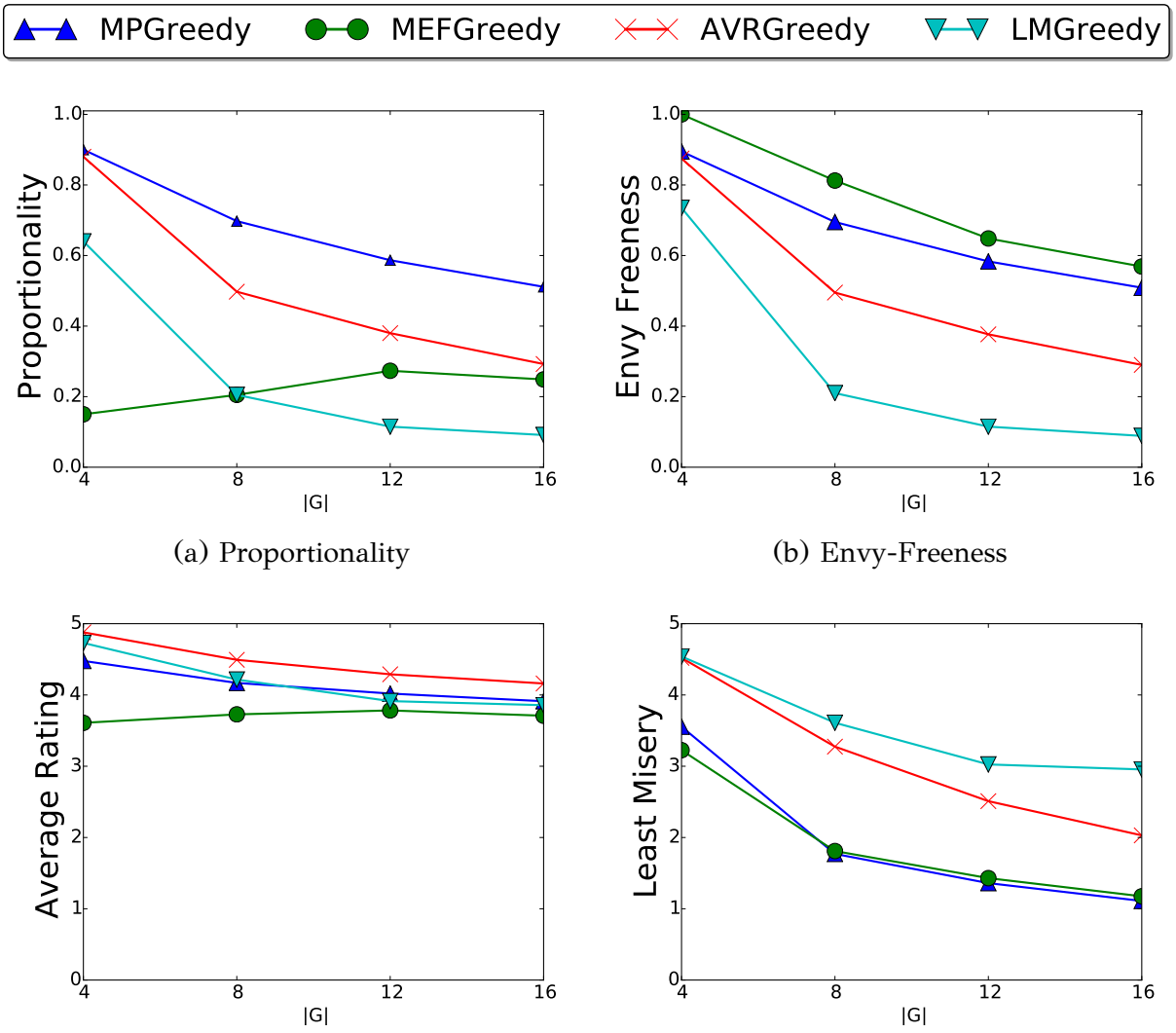
(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.11: Performance comparison with varying package size, 2 coverage, anti-correlated groups

tance 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 the two Greedy algorithms are the best compromise between performance and CPU cost, with GridGreedy giving a little better performance, and PartitionGreedy being a little faster. That can be explained by the fact that the search space in PartitionGreedy is not expanded to neighboring cells when checking for valid packages. Consequently, PartitionGreedy ends up be-



(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

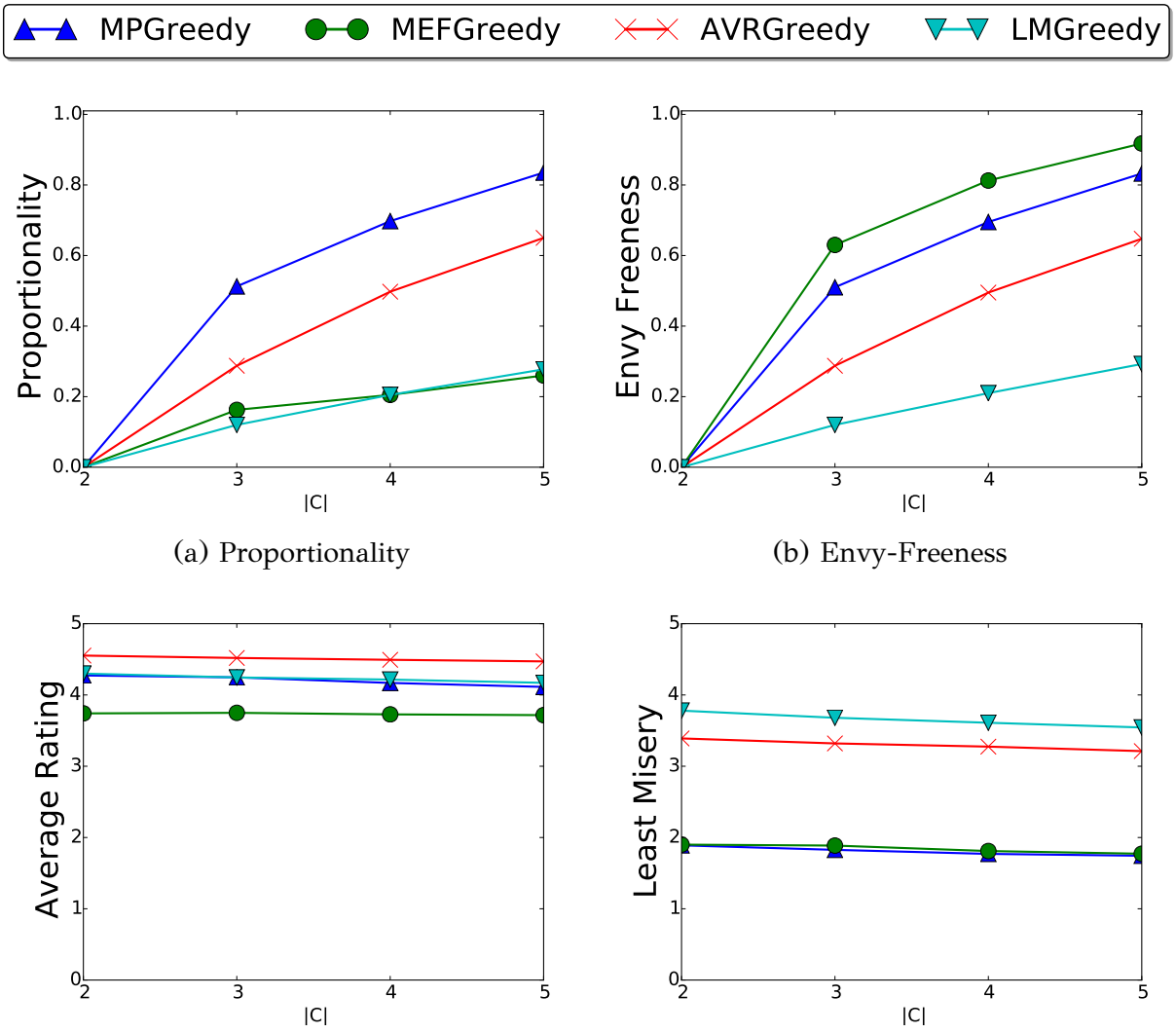
(d) Least Misery

Figure 5.12: Performance comparison with varying group size, 3 coverage, anti-correlated groups

ing faster (has to check fewer items) but gives a bit lower values for proportionality. The same trends apply to disjoint 5 groups and can be seen in Figure 5.28.

5.1.3 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



(a) Proportionality

(b) Envy-Freeness

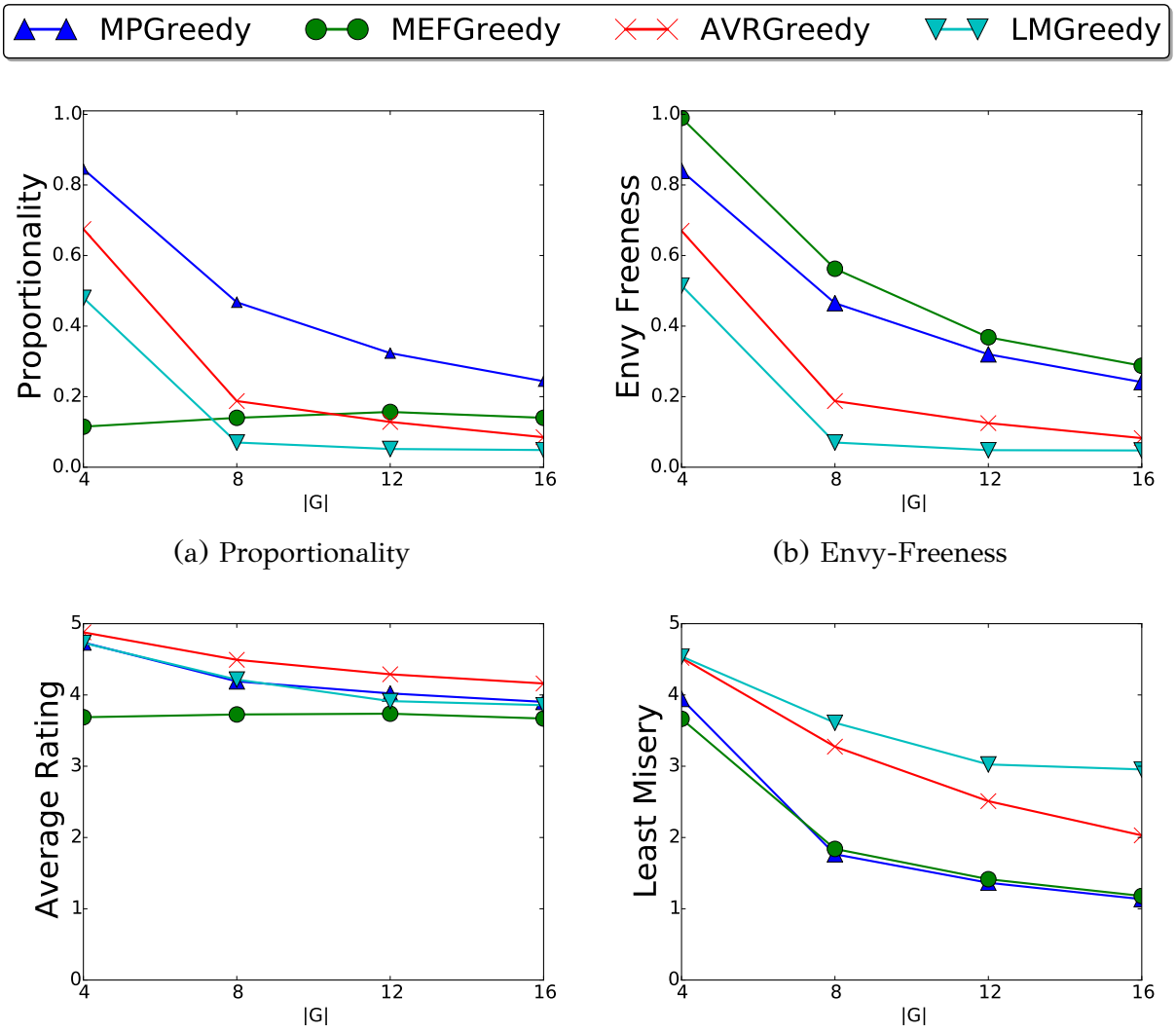
(c) Average Rating

(d) Least Misery

Figure 5.13: Performance comparison with varying package size, 3 coverage, anti-correlated groups

2-4 users each (there are 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 5.1 shows the average of the proportionality values given by the users. The first observation is that SPGreedy performs the best, validating our approach.



(a) Proportionality

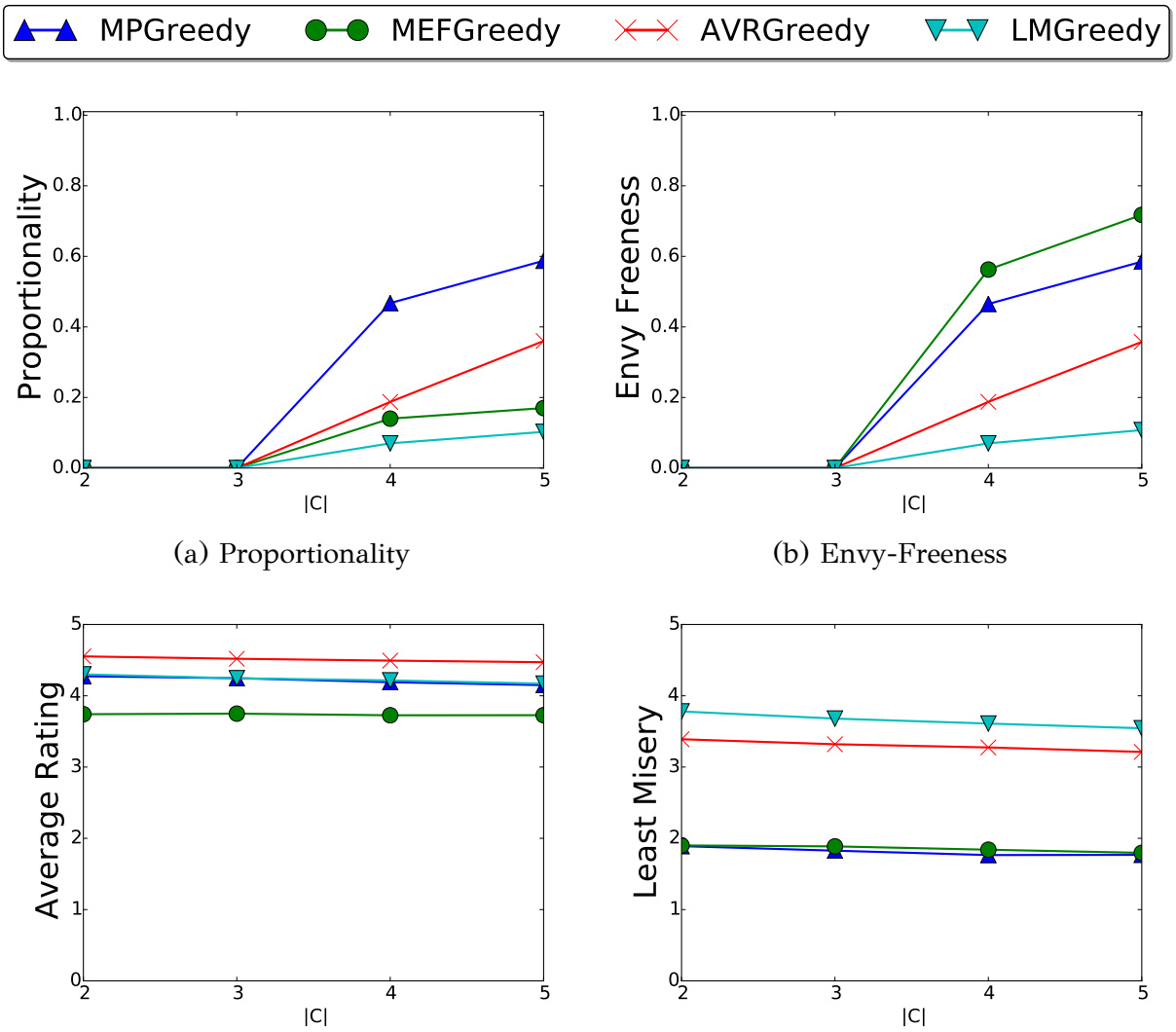
(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.14: Performance comparison with varying group size, 4 coverage, anti-correlated groups

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.



(a) Proportionality

(b) Envy-Freeness

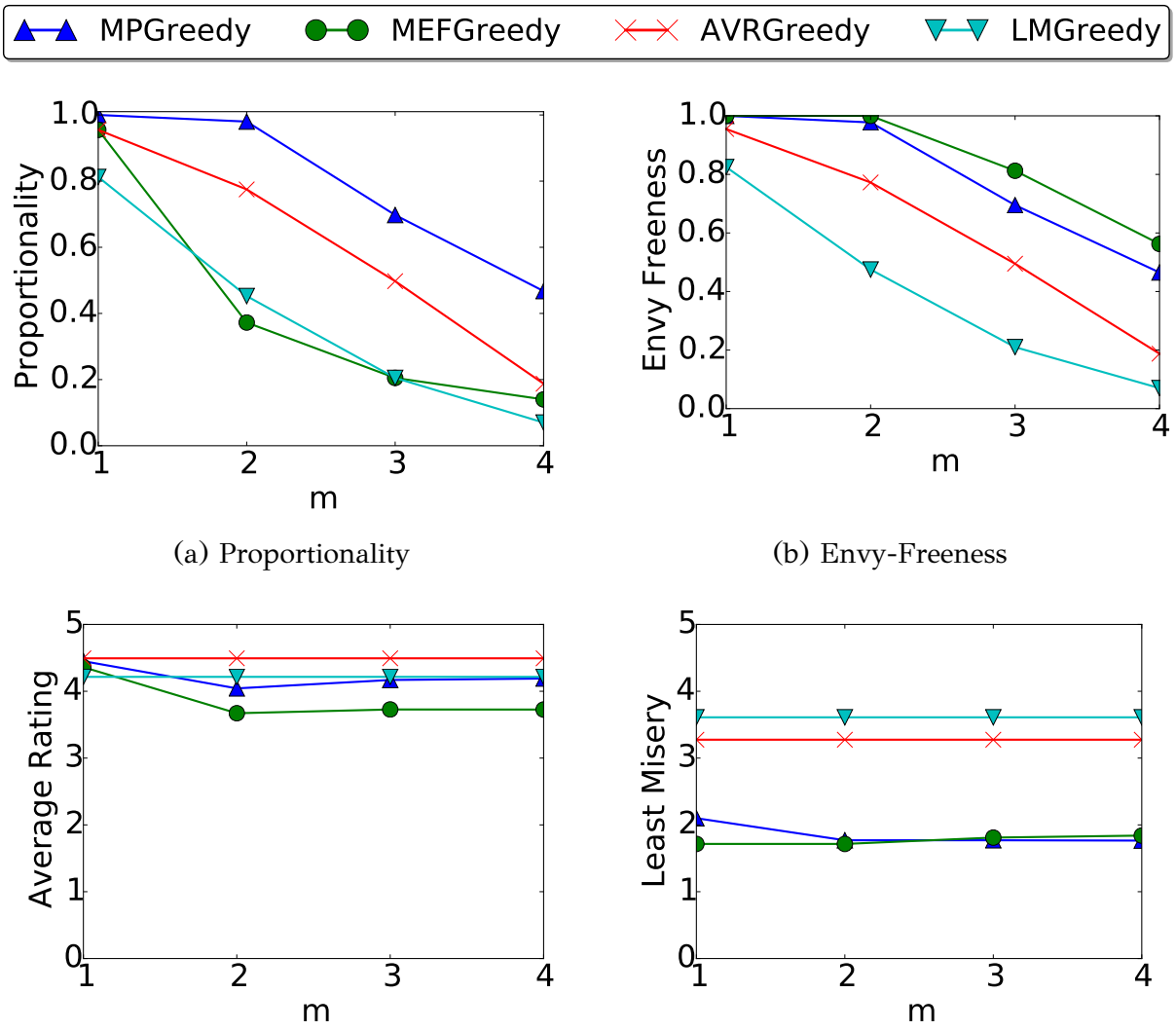
(c) Average Rating

(d) Least Misery

Figure 5.15: Performance comparison with varying package size, 4 coverage, anti-correlated groups

Table 5.1: User Study

	Random	AVRGreedy	LMGreedy	SPCGreedy
Proportionality	0.61	0.77	0.79	0.83



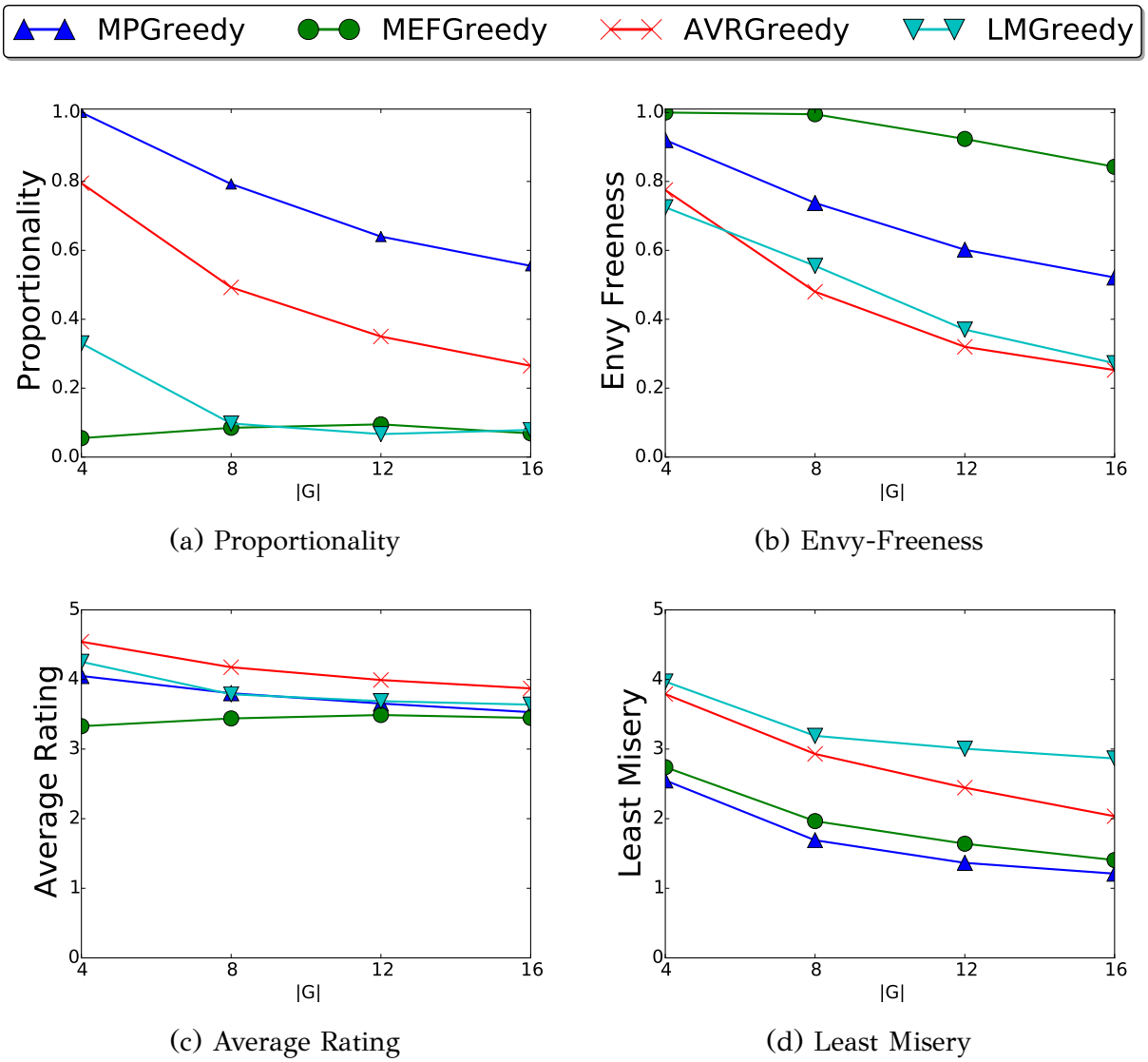
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.16: Performance comparison for varying coverage, anti-correlated groups



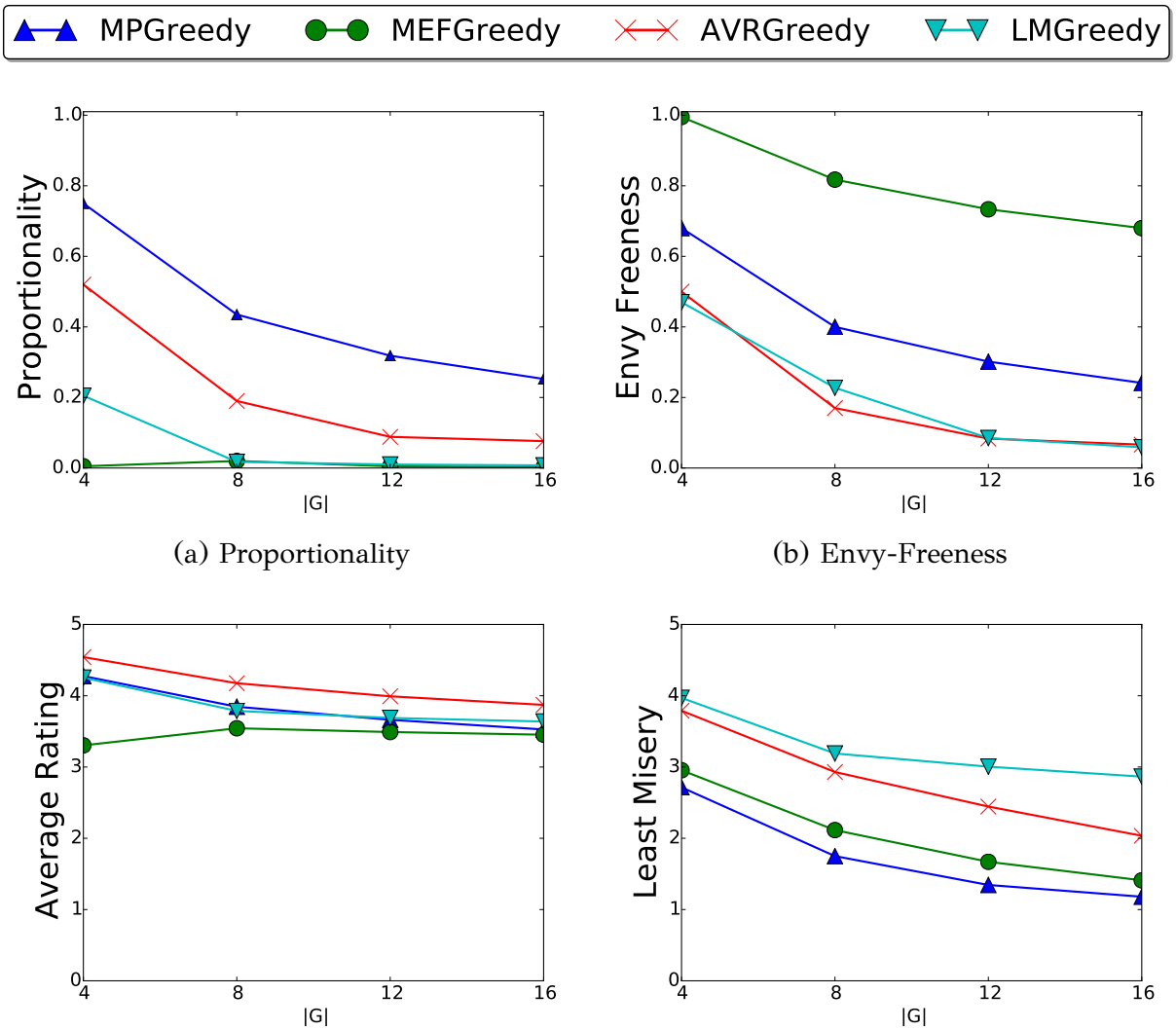
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.17: Performance comparison with varying group size, 2 coverage, disjoint 5 groups



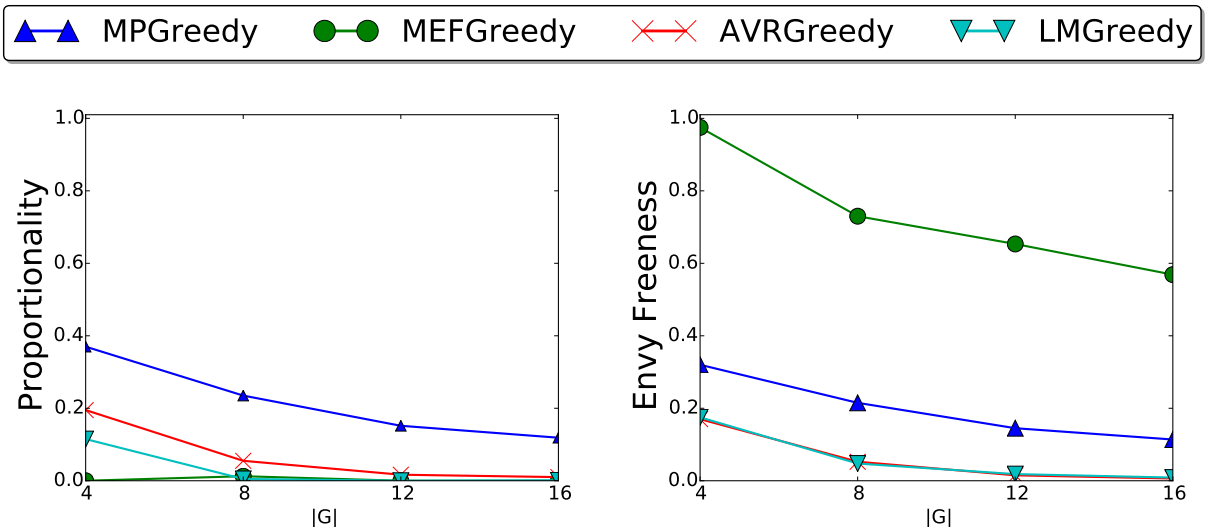
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.18: Performance comparison with varying group size, 3 coverage, disjoint 5 groups



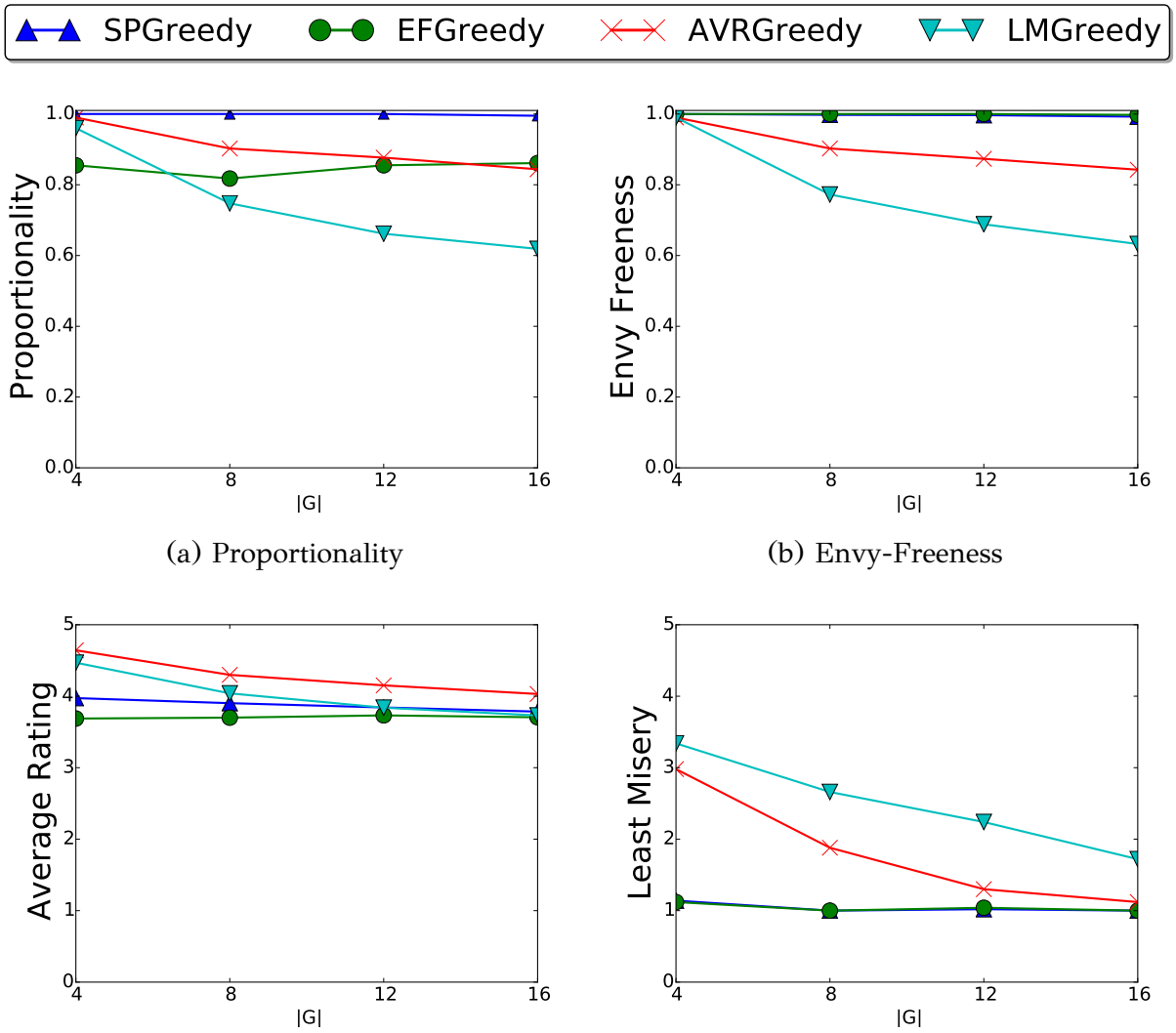
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.19: Performance comparison with varying group size, 4 coverage, disjoint 5 groups



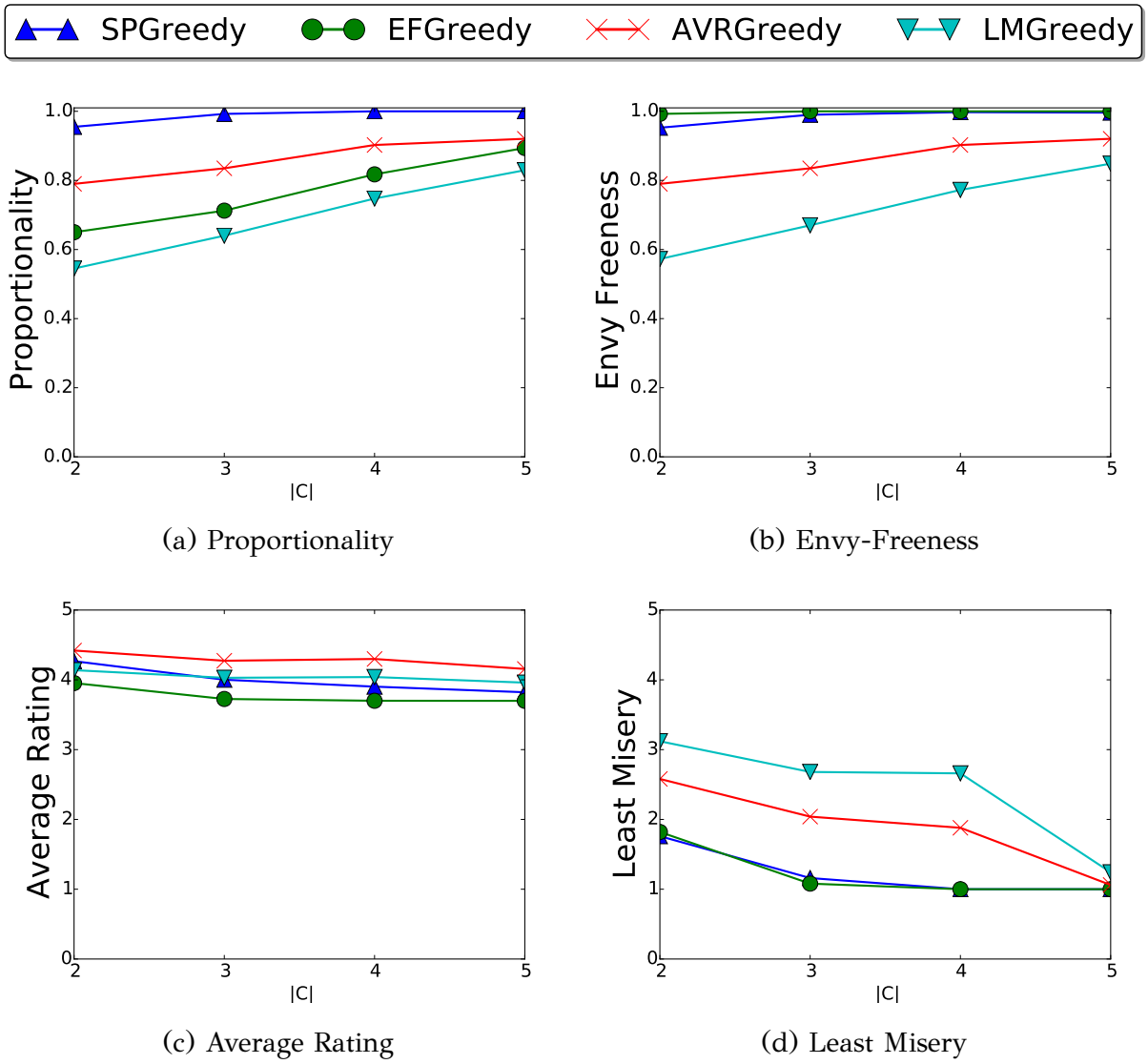
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.20: Performance comparison with varying group size, category constraints, anti-correlated groups



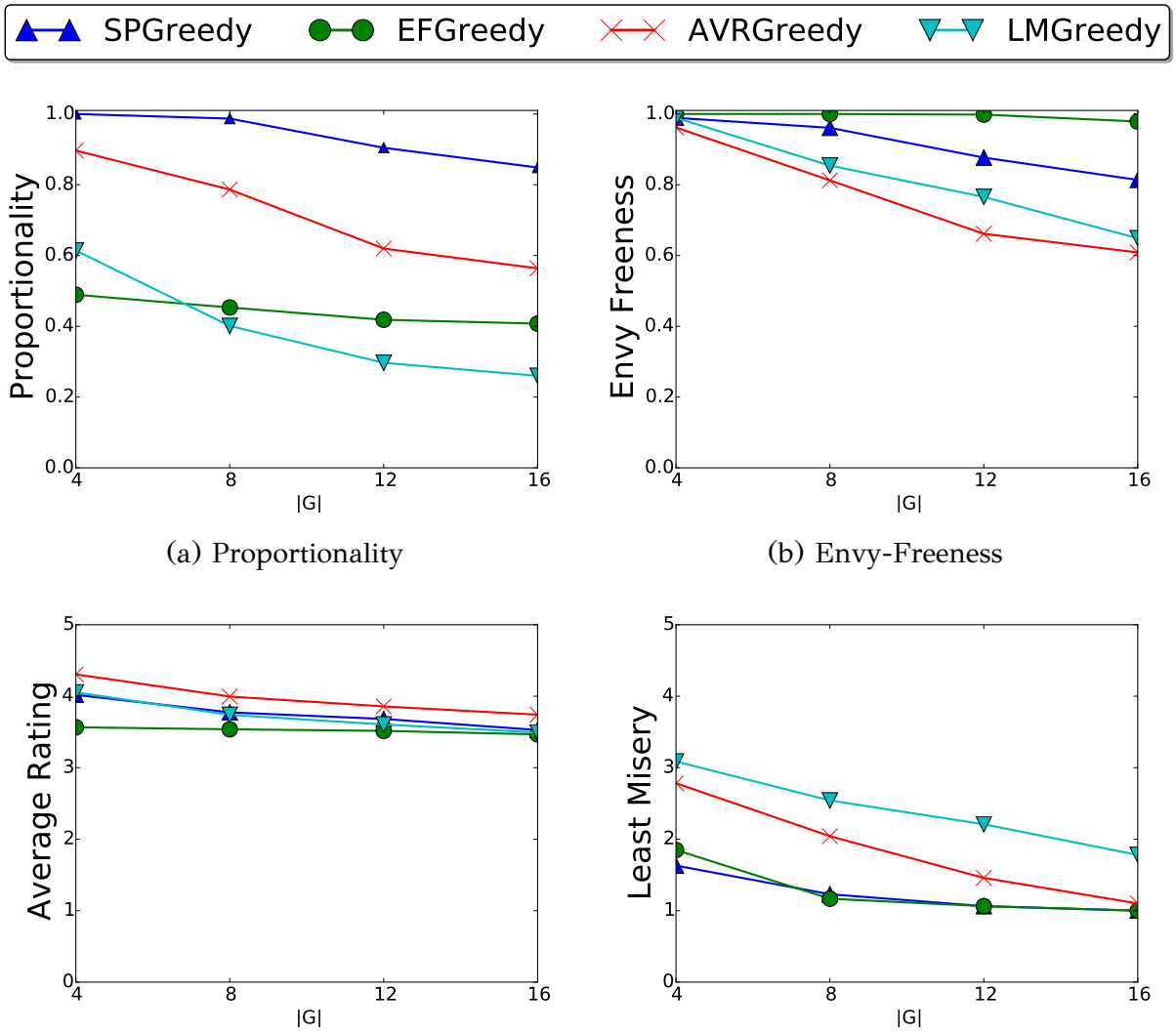
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.21: Performance comparison with varying package size, category constraints, anti-correlated groups



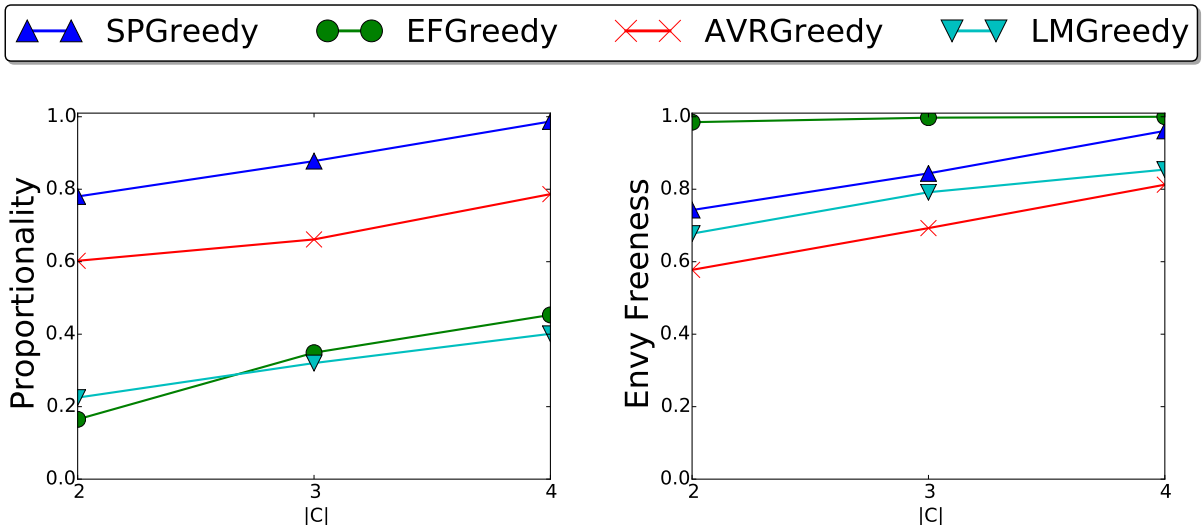
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.22: Performance comparison with $\Delta = 5$ and varying group size, categories constraint, disjoint 5 groups



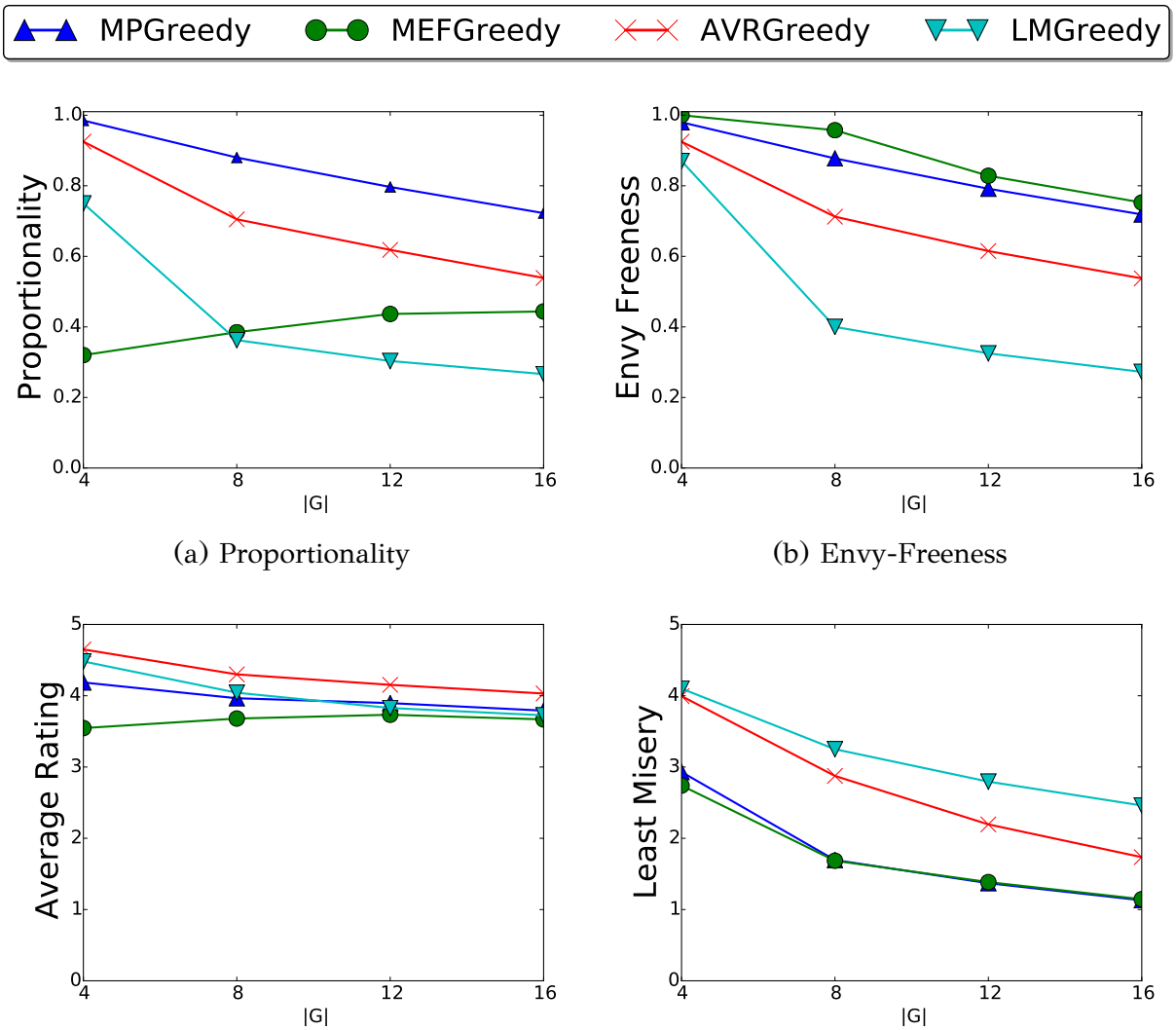
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.23: Performance comparison with $\Delta = 5$ and varying package size, categories constraint, disjoint 5 groups



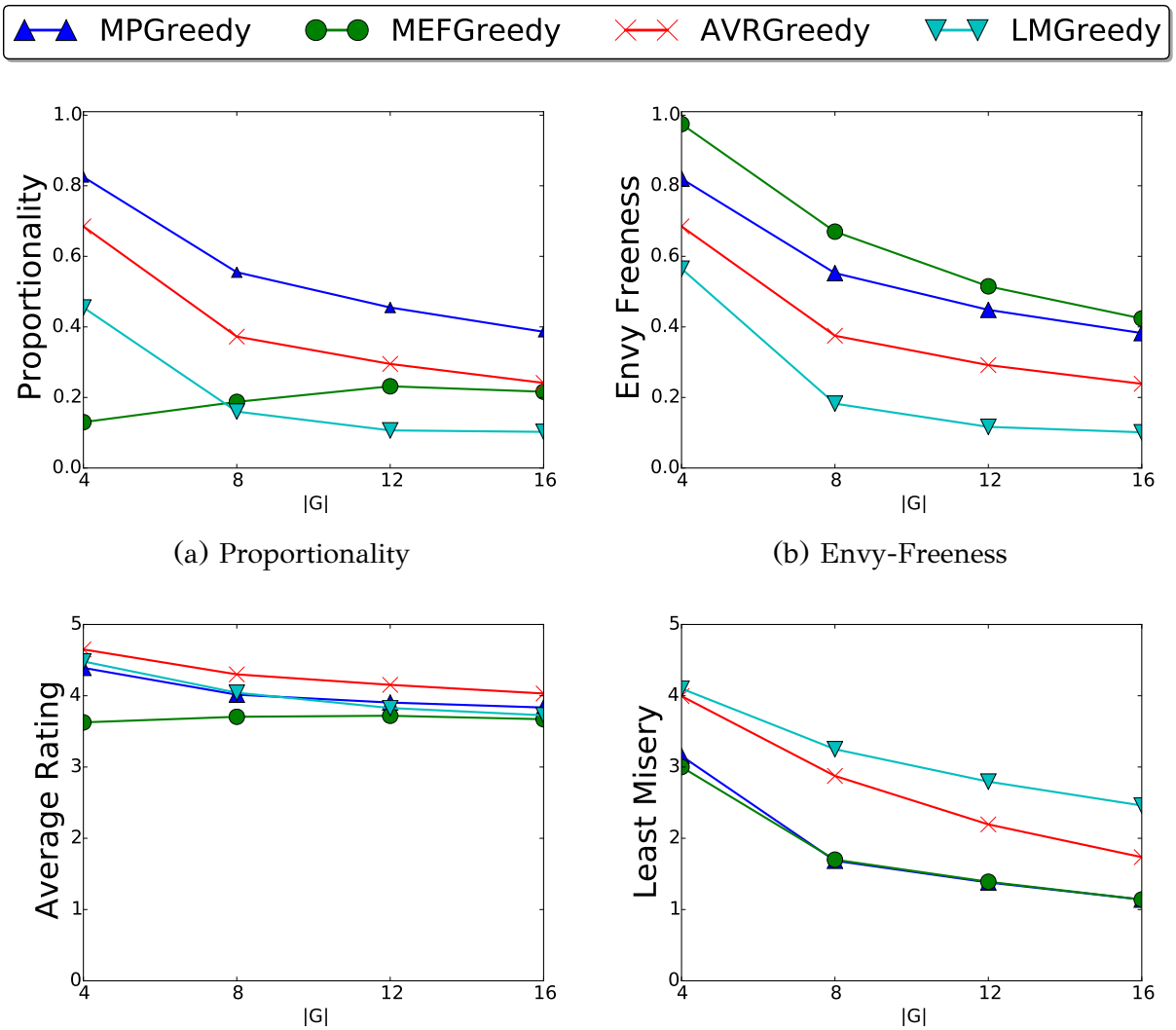
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.24: Performance comparison with varying group size, category constraints, 2 coverage, anti-correlated groups



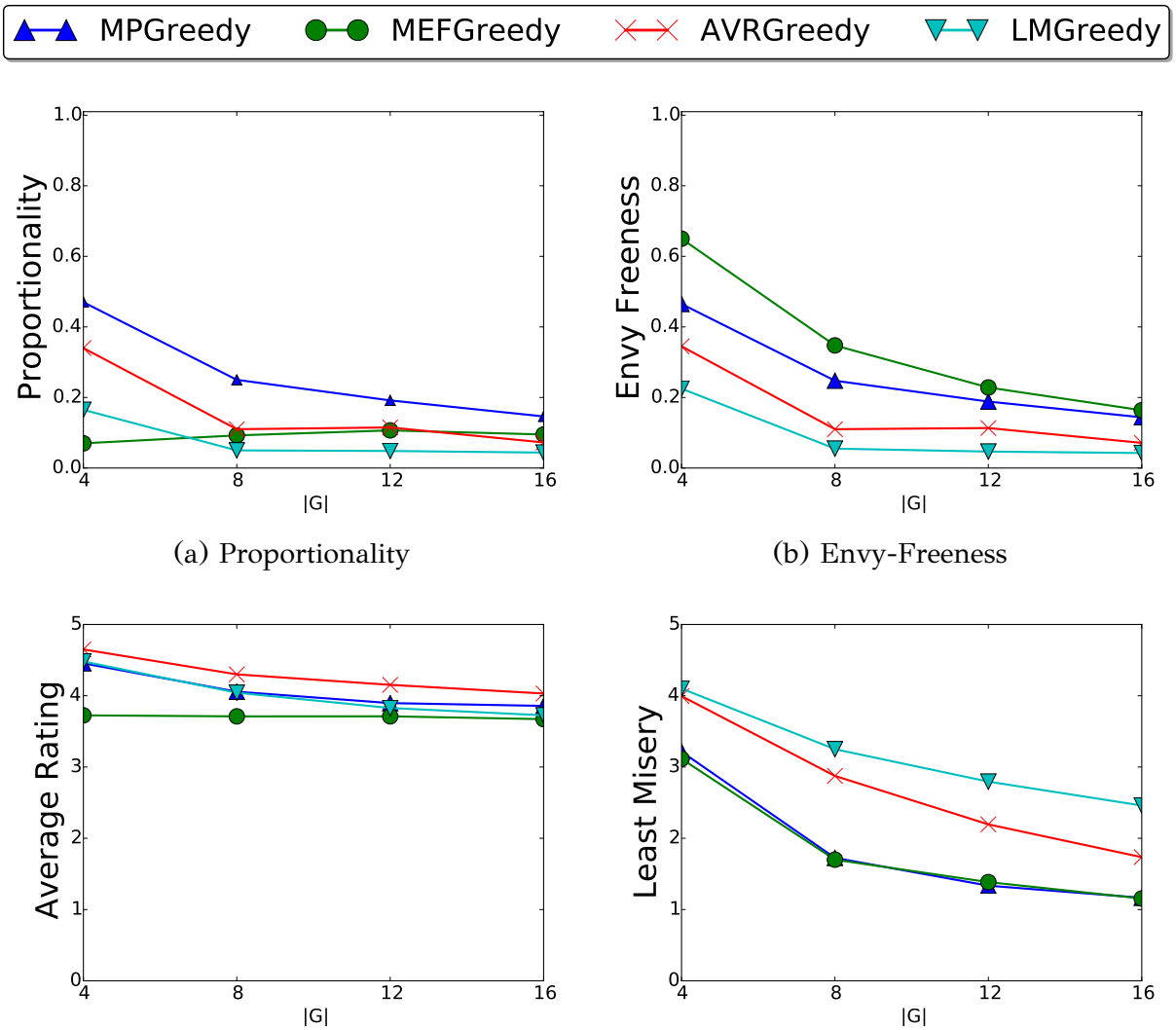
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure 5.25: Performance comparison with varying group size, category constraints, 3 coverage, anti-correlated groups



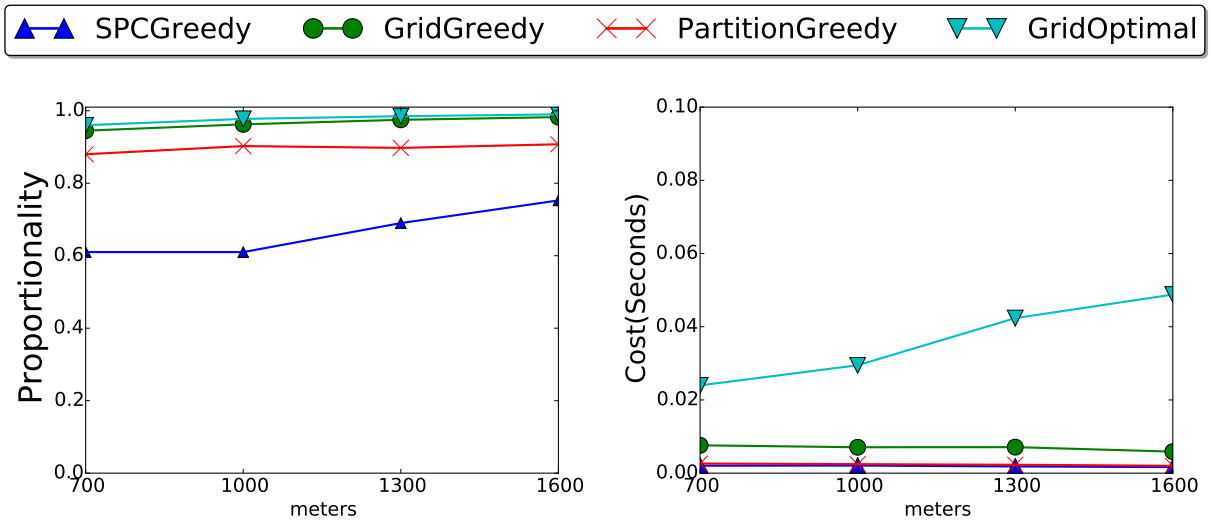
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

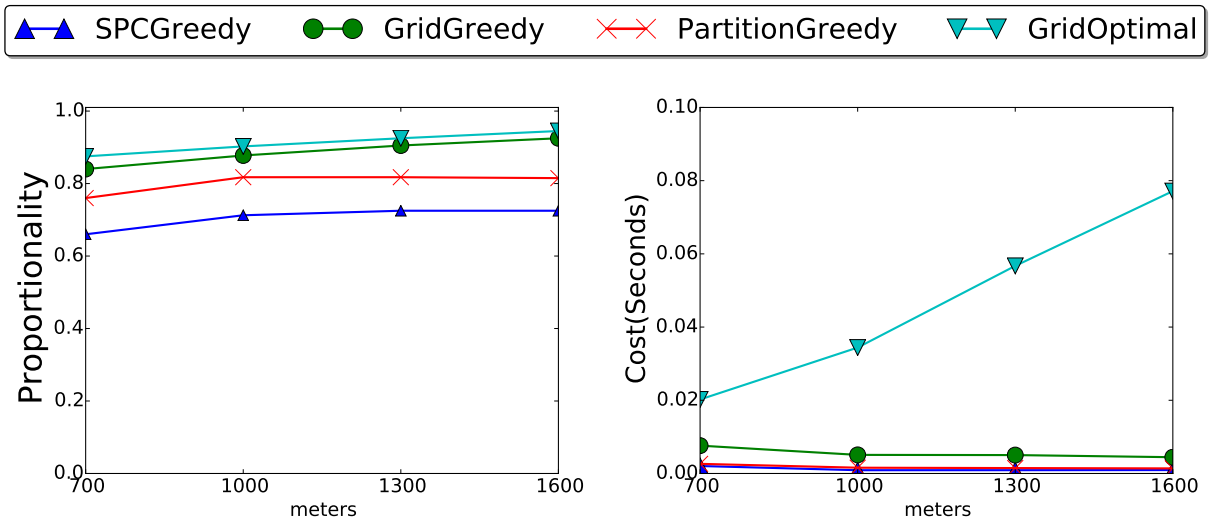
(d) Least Misery

Figure 5.26: Performance comparison with varying group size, category constraints, 4 coverage, anti-correlated groups



(a) Proportionality (b) cost

Figure 5.27: Performance comparison with varying distance constraint, category constraints, anti-correlated groups



(a) Proportionality (b) cost

Figure 5.28: Performance comparison with varying distance constraint, category constraints, disjoint 5 groups

Chapter 6

Conclusions

In this thesis, 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.

Bibliography

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

[2] 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.*, vol. 17, pp. 734–749, June 2005.

[3] T. Deng, W. Fan, and F. Geerts, "On the complexity of package recommendation problems," *SIAM J. Comput.*, vol. 42, no. 5, pp. 1940–1986, 2013.

[4] M. Xie, L. V. S. Lakshmanan, and P. T. Wood, "Generating top-k packages via preference elicitation," *PVLDB*, vol. 7, no. 14, pp. 1941–1952, 2014.

[5] Q. Yuan, G. Cong, and C.-Y. Lin, "Com: A generative model for group recommendation," in *KDD*, pp. 163–172, 2014.

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

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

[8] A. Jameson and B. Smyth, "Recommendation to groups," in *The Adaptive Web, Methods and Strategies of Web Personalization*, pp. 596–627, 2007.

[9] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *WWW*, pp. 285–295, 2001.

[10] H. Steinhaus, "The problem of fair division," *Econometrica*, vol. 16, no. 1, pp. 101–104, 1948.

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

[12] Q. Hua, Y. Wang, D. Yu, and F. C. M. Lau, "Set multi-covering via inclusion-exclusion," *Theor. Comput. Sci.*, vol. 410, no. 38-40, pp. 3882–3892, 2009.

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

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

[15] S. Amer-Yahia, S. B. Roy, A. Chawla, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency," *PVLDB*, vol. 2, no. 1, pp. 754–765, 2009.

[16] K. Li, W. Lu, S. Bhagat, L. V. S. Lakshmanan, and C. Yu, "On social event organization," in *KDD*, pp. 1206–1215, 2014.

[17] X. Liu, Y. Tian, M. Ye, and W.-C. Lee, "Exploring personal impact for group recommendation," in *CIKM*, pp. 674–683, 2012.

[18] J. Gorla, N. Lathia, S. Robertson, and J. Wang, "Probabilistic group recommendation via information matching," in *WWW*, pp. 495–504, 2013.

[19] A. G. Parameswaran, P. Venetis, and H. Garcia-Molina, "Recommendation systems with complex constraints: A course recommendation perspective," *ACM TOIS*, vol. 29, no. 4, p. 20, 2011.

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

[21] S. Amer-Yahia, F. Bonchi, C. Castillo, E. Feuerstein, I. Méndez-Díaz, and P. Zabala, "Composite retrieval of diverse and complementary bundles," *IEEE TKDE*, vol. 26, no. 11, pp. 2662–2675, 2014.

[22] T. Zhu, P. Harrington, J. Li, and L. Tang, "Bundle recommendation in ecommerce," in *SIGIR*, pp. 657–666, 2014.

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

[24] C. Yu, L. V. S. Lakshmanan, and S. Amer-Yahia, "Recommendation diversification using explanations," in *ICDE*, pp. 1299–1302, 2009.

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

[26] D. S. Hochbaum, "Approximation algorithms for np-hard problems," *SIGACT News*, vol. 28, pp. 40–52, June 1997.

[27] N. G. Hall and D. S. Hochbaum, "A fast approximation algorithm for the multicovering problem," *Discrete Applied Mathematics*, vol. 15, no. 1, pp. 35 – 40, 1986.

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

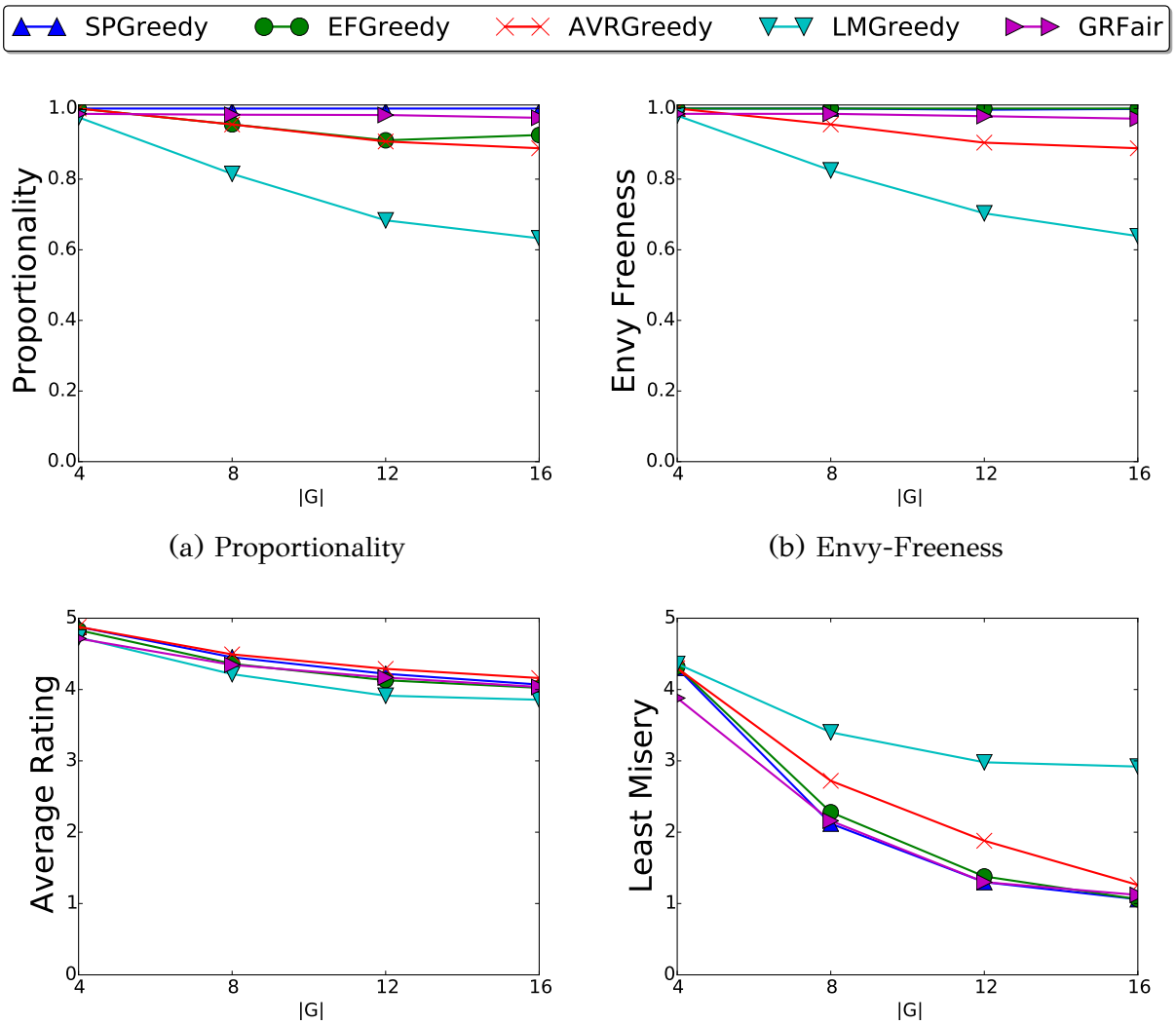
[29] P. Tsaparas, A. Ntoulas, and E. Terzi, "Selecting a comprehensive set of reviews," in *KDD*, pp. 168–176, 2011.

Appendix A

Comparison with GR-Fair

In the first part of the Appendix we will compare the performance of our algorithms vs the one proposed in [7] , where the concept of fairness can optionally be taken into account. We will call that algorithm GRFair. We could had compared those algorithms from the start but both AVRGreedy and LMGreedy are much more popular algorithms for the package-to-group problem ,so it was decided to focus more on them in the main body of the thesis. We will do the comparison for the single proportionality problem without categories (which is our main problem) using the anti-correlated and disjoint 5 groups which represent the most difficult case of input groups.

In Figures A.1 and A.2 we see the performance comparison as a function to the group size and the package size respectively,for anti-correlated groups. SPGreedy and EFGreedy both have a slight advantage over GRFair on their respective metrics-while scoring a similar performance in the two remaining metrics . The difference become more apparent for disjoint 5 groups and especially in case where $G \geq 12$, as seen in Figure A.3.



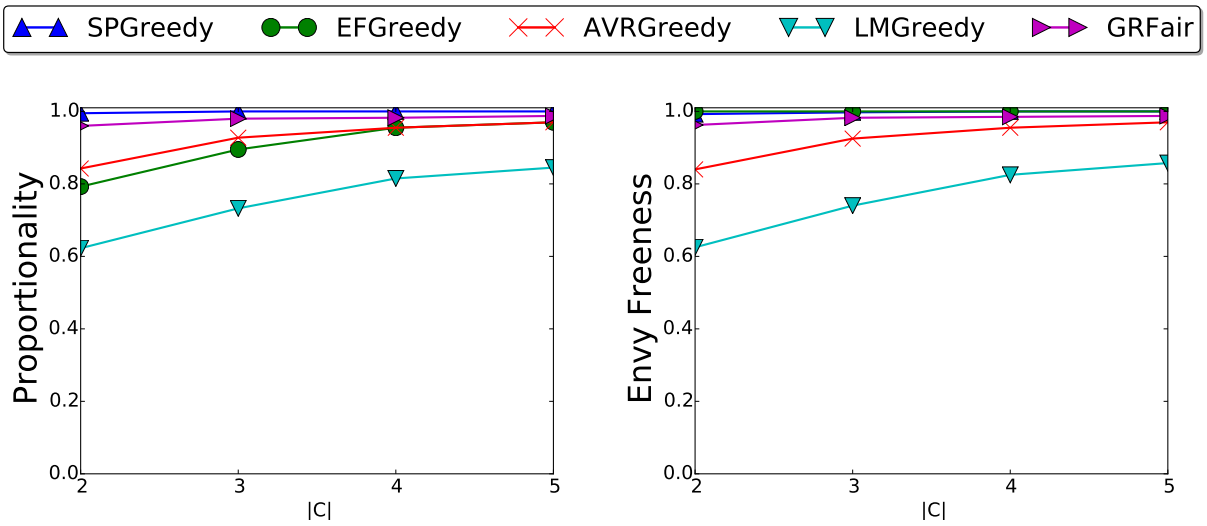
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure A.1: Performance comparison with varying group size, anti-correlated groups



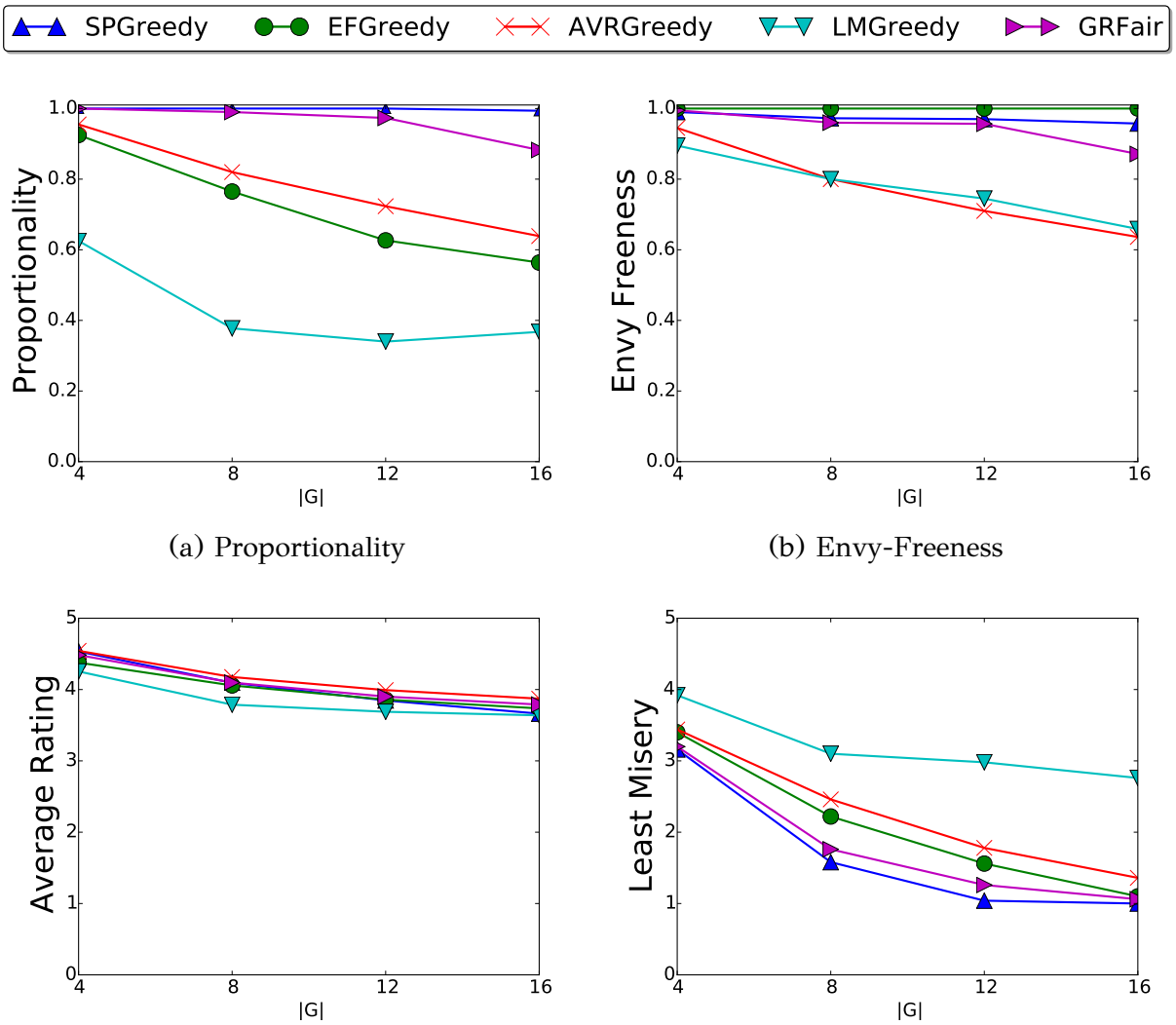
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure A.2: Performance comparison with varying package size, anti-correlated groups



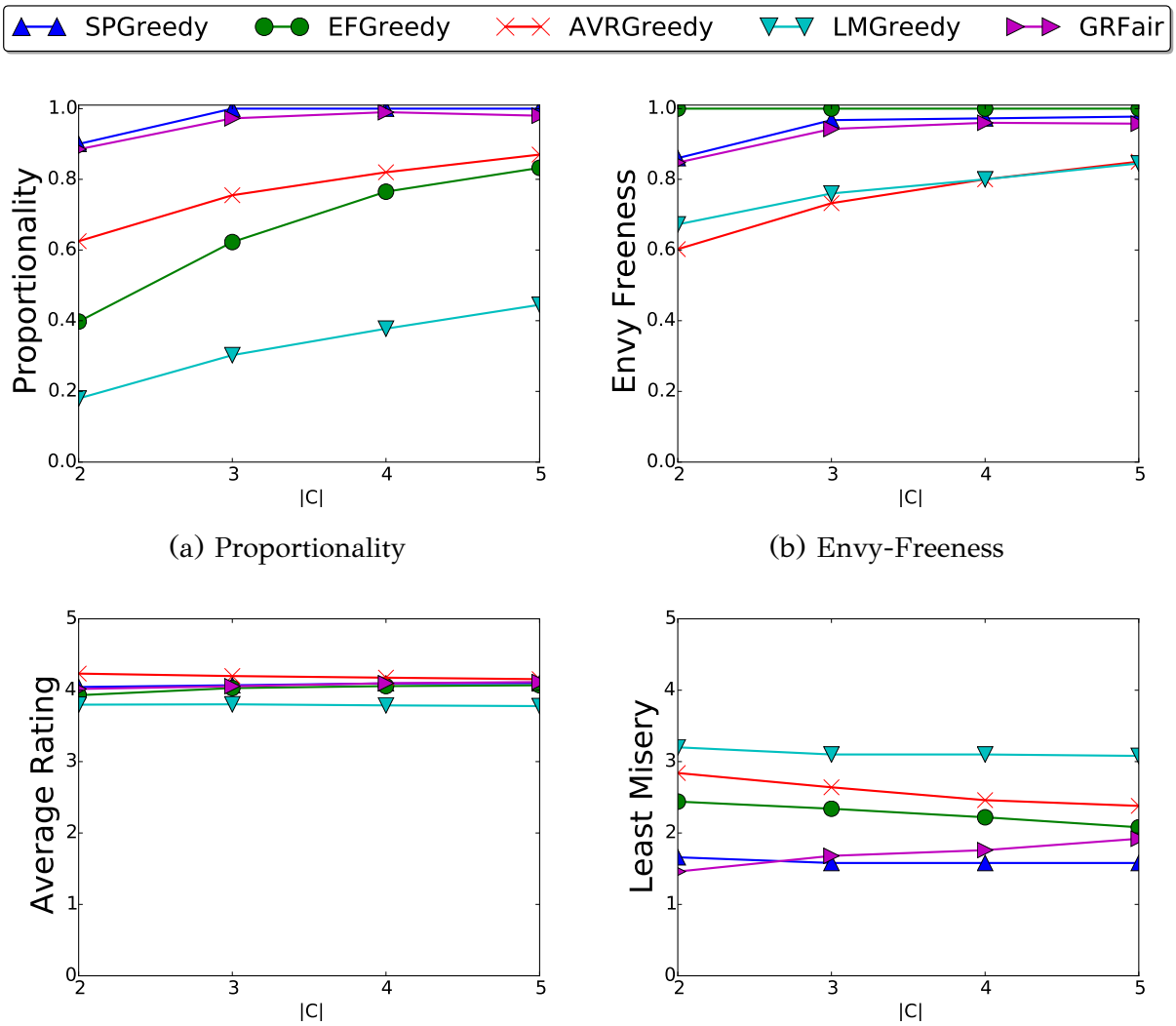
(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure A.3: Performance comparison with varying group size, disjoint 5 groups



(a) Proportionality

(b) Envy-Freeness

(c) Average Rating

(d) Least Misery

Figure A.4: Performance comparison with varying package size, disjoint 5 groups

Appendix B

Taking into account negative items

B.1 Modified model

After additional user studies with more users were concluded, a general trend was made apparent. In cases where the group size was relatively small (4 users) and for small package size as well (2-4 items per package), the package provided for our SPGreedy algorithm was usually deemed fairer than the ones provided by AVRGreedy and LMGreedy. On the contrary we observed that for cases where the groups were big (with more than 8 users) , the users started prioritizing packages that did not contain items that greatly dissatisfied them,giving an edge to LMGreedy. There were many cases where a package with no items ranked at a given user topΔ% and contained no 'bad' items for the use, was preferred over another 'fairer ' according to our definition. That motivated us to change our single proportionally definition to cater for items that greatly dissatisfy a user.
First we model the dislike of user towards an item :
Given a package P, and a parameter Δ, we say that a user u dislikes an item $i \in P$, if i is ranked in the bottom-Δ% of the preferences of u over all items in $\mathcal{I}$.

Definition B.1. For a user u, and a package P, we say that P is 1-proportional for

u, if there exist at least 1 item in P, that u likes and there exist no item that the user greatly dislikes.

B.2 Potential solutions

The most obvious solution to the above problem is to remove items that greatly dissatisfy the users. Of course by doing that we have no guarantee that the final result will approximate the optimal solution.

Another way to tackle the above problem is to try to satisfy as much users (propose an item that is ranked at their topΔ% of their preferences), while trying to not dissatisfy any one that is already satisfied by the package so far. In this approach, for each item i we have a set of users that dislike the item which we will call DS_i and a set of users that like the item , S_i. Also let $\text{SAT}_G(P)$ represent the users that are satisfied by the package P so far, in that there exists at least one item in P that they like and no item that they dislike and $\text{DSAT}_G(P)$ the users that are dissatisfied . We use an updated utility function as $f_G(P,i) = |\text{SAT}_G(P) \cup S_i \setminus \text{SAT}_G(P)| - |(DS_i \cap \text{SAT}_G(P)| - |(\text{SAT}_G(P) \cup S_i \setminus \text{SAT}_G(P)) \setminus \text{DSAT}_G(P)|$. The first term represents the new users that will be satisfied by adding a new item, the second the users that will become dissatisfied by the addition and were previously satisfied, while the third represents how many of the new 'would be satisfied' users were already dissatisfied by the package so far. In each step we will add the item that has the greatest value for the utility function, provided there exists at least one item with $f_G(P,i) \geq 0$. If all values for the utility function is < 0 then if the extra item is added ,the package will become worse. The approximation of the above algorithm is $1/k$ of the optimal solution. In the first loop, it will include to the package, the item that is liked by most of the users(the users that dislike the first item will be ignored in this case since $|(DS_i \cap \text{SAT}_G(\emptyset)| = 0$and $Dissatisfy(P) = 0$). After that point,additional items will be added that are liked by more users and are disliked by the least amount of already satisfied users. Since we require the utility function in each additional step to be ≥ 0, the overall quality is guaranteed to not deteriorate, therefore the $1/k$ approximation. In other words, just by including the first item, the $1/k$ approximation is guaranteed(the first item will be the one that is liked by most users).By adding additional items, the algorithm's performance will either improve

Algorithm B.1 Greedy Fairness Maximization with negative items

Input: Group of users G,S_i,DS_i, items $\mathcal{I}$, value K

Output: Package P contained at most K items

1: $P \leftarrow \emptyset$
2: Candidates $\leftarrow \mathcal{I}$
3: $J = 1$
4: **while** $J \leq K$ **do**
5: **if** there is at least one item where $f_G(P, i) \geq 0$ **then**
6: $i \leftarrow \arg\max_{i \in \text{Candidates}} f_G(P, i)$
7: $P \leftarrow P \bigcup \{i\}$
8: Candidates $\leftarrow$ Candidates $\setminus \{i\}$
9: **else**
10: break
11: **end if**
12: $J = J + 1$
13: **end while**
14: **return** P

or stay the same.If we call the set of users satisfied by greedy algorithm NGreedy and the set of users satisfied by the optimal OPT, after the inclusion of the first item we will have:

$|NGreedy| \geq |OPT|$

At worst case after k steps, the greedy algorithm will have either not included additional items or will have included at most k items that did not further increased it's performance and the optimal will have included k items.Therefore

$|NGreedy| \geq 1/k * |OPT|$

Author's Publications

Serbos Dimitrios, Shuyao Qi, Nikos Mamoulis, Evaggelia Pitoura and Panayiotis Tsaparas, "Fairness in Package-To-Group Recommendations" in WWW2017.

Serbos Dimitrios

Curriculum Vitae

PERSONAL DETAILS

Birth	December 19, 1987
Address	Zagoriou 10-12, Ioannina
Phone	6972091338
Mail	serbosd@gmail.com

EDUCATION

Electrical and Computer Engineer 2005-2012

NTUA

Finish my degree as an electrical and computer engineer in the national technical university of Athens.

SKILLS

Languages	Greek (mother tongue)
	English (fluent)
	German (basic)
Software	MATLAB, LaTeX, C, JAVA, PYTHON