Efficient and Fair Item Coverage in Recommender Systems

Iordanis Koutsopoulos
Department of Informatics
Athens University of Economics and Business
Email: jordan@aueb.gr

Maria Halkidi
Department of Digital Systems
University of Piraeus
Email: mhalk@unipi.gr

***Abstract*—We study the design of recommender systems under the constraint of item coverage. An item is *covered* if it is recommended at least to a certain number of users. This situation arises in settings where the items to be recommended stem from different entities such as owners, producers or advertisers with whom the recommendation engine has come into agreement about item promotion through recommendation, in exchange for some payment. It is therefore important to issue recommendations with breadth, in the sense that each item reaches a sufficiently large portion of the user base through recommendation. This constraint drastically changes the recommendation problem since now the lists of items to be recommended to different users become coupled. We formulate and study the recommendation problem under the item coverage constraint, with the goal of minimizing the cost of deviation from a nominal recommender system which does not cater for item coverage. We show that the linear-programming relaxation of the problem gives the optimal integral solution, and we also propose a low-complexity heuristic algorithm to solve large instances of the problem. Further, we study the problem of guaranteeing item coverage while making the incurred cost of deviation as balanced as possible across items (and therefore across their owners) or across users. The plots in the numerical results section demonstrate and quantify the tradeoff between recommendation accuracy and item coverage and that between cost imbalance across items and coverage.**

***Index Terms*—Recommender systems, item coverage, mathematical optimization.**

I. Introduction

Recommender systems (RSs) arise in various settings and aim to provide accurate personalized suggestions so as to aid users in decision making. Their span covers settings such as personalized suggestions to purchase or use various goods and services, personalized search results, targeted advertising and social-network related suggestions. Users become more and more dependent on recommender systems in order to expedite purchase of goods, selection of movies, decisions on where to dine or spend their vacation and even decisions whom to socialize with.

A number of recommendation algorithms and approaches have been developed, which leverage item and/or user similarities in order to generate a list of recommended items for each user that is tailored to her preferences [1]. In traditional recommender systems, the performance objective to be optimized is the total recommendation *accuracy*, namely the mean squared error between predicted and true ratings in the training dataset. This user-centric performance objective quantifies the quality of the recommendation service to end-users, and it reflects a kind of social-welfare metric for them.

However, more often than not, RSs are embedded components in larger online services, e.g. an online retail store or a social media site, and a number of entities other than the end-users are interested in the results of the recommendation algorithm. These entities may be the item owners, producers, providers or advertisers, and they may have agreements with the recommendation engine about item promotion and specific user outreach through recommendation, in exchange for some payment. The items may be restaurants, hotels, books or other sponsored items for which their owners have paid to have them appear in recommended lists of items of a certain number of users. For example, a hotel chain owner offers some remuneration to a recommendation engine, and the latter should issue recommendations such that this chain appears in the recommended lists of items of some users. Other examples are an online retail store that sells books of different publishers or products of different firms, or a media provider which recommends movies of different producer and distribution companies.

In such scenarios, a different RS approach than the traditional one is required. The need to ensure a certain user outreach is critical to fulfil in light of the agreements between the recommendation engine and item owners, as it determines the revenue of the site to which the RS is embedded. This requirement for item coverage changes drastically the recommendation algorithm as well. In a typical recommender system, the lists of recommended items are uncoupled across different users, and the only criterion for recommendation is accuracy, i.e. the mean squared error. However the item coverage constraint makes the lists of recommended items to different users coupled among themselves, and therefore the recommendation of certain items to some users influences the set of items to be recommended to other users.

In this work, we study the design of RSs *under the constraint of guaranteeing sufficient item coverage*. An item is said to be *covered* if it is recommended to at least a certain number or percentage of users e.g. 5% of users. A RS with high coverage spreads items across a wide spectrum of users and hence can better facilitate their adoption or purchase. If the items belong to different owners, item coverage makes sense for the profitability of the platform when agreements such as

the ones above are enforced. This setting is also reminiscent of pay-per-impression advertising, where advertisers pay the advertising engine to project ads a certain number of times to users. However in RSs, the unique chellenge that cannot be ignored is the tradeoff between accuracy and item coverage. The contributions of our work are as follows.

- We formulate and study the recommendation problem under the item coverage constraint, with the goal of minimizing the cost of deviation from a nominal RS that does not cater for item coverage. The cost of deviation is defined as the sum of differences between ratings for items recommended to a user according to the nominal RS, and ratings for items recommended through the new coverage-aware approach under consideration.
- We show that the linear-programming relaxation of the problem above gives the optimal integral solution, and we also propose a low-complexity heuristic algorithm to solve large instances of the problem.
- We also study the problems of guaranteeing item coverage while making the incurred cost of deviation as balanced as possible across items (and therefore across their owners) or across users.
- We quantify through the plots in the numerical results section the tradeoff between recommendation accuracy and item coverage, and the tradeoff between cost imbalance across items and item coverage.

The paper is organized as follows. In section II, we present the model and state the problems, and in section III we show how to solve the problems through a linear-programming relaxation of the mathematical optimization problem, as well as through a greedy algorithm. In section IV we present numerical results, in section V we present an overview of related work, and in section VI we conclude the paper.

II. Model and problem statement

A. Model

Consider a set $\mathcal{U}$ of K users and a set $\mathcal{I}$ of n items available for recommendation. There exists a baseline RS, e.g. a latent-factor based or collaborative-filtering (CF) one, through which ratings r_{iu} are predicted for each item i that user u has not experienced yet. Let $\mathcal{L}_B(u) \subset \mathcal{I}$ be the subset of top-L items that the baseline system recommends to user u. Namely $\mathcal{L}_B(u)$ is the list of recommended items to user u according to the baseline RS, and L is the length of the list, i.e. the number of items recommended to each user. We assume L is fixed regardless of the recommender algorithm.

In this work, the emphasis is on designing a RS with item coverage. Item coverage refers to the fact that each item appears in the list of recommended items of a certain number of users. We give the following definition.

Definition 1. *For given positive integer d and a given RS $\mathcal{A}$, an item i satisfies the d-coverage constraint or it is d-covered, if it is recommended to at least d users. That is, item i should appear in the lists of at least d users.*

We then say that RS $\mathcal{A}$ leads to d-item coverage, or that all items are d-covered.

For example, a movie from a certain producer can be d-covered for d in the order of some hundreds of users.

Our goal is to come up with a recommendation algorithm $\mathcal{A}$ such that in the new lists of recommended items $\{\mathcal{L}_{\mathcal{A}}(u)\}_{u\in\mathcal{U}}$, each item is d-covered. In the sequel, we drop the subscript $\mathcal{A}$, and we denote the sought lists of each user u as $\{\mathcal{L}(u)\}$.

Each position $l = 1, \ldots, L$ in the list $\mathcal{L}(u)$ of recommended items to a user u is associated with a weight w_u^l which quantifies user u behavior in terms of clicking/viewing an item when it appears in the l-th position. For a user u, let f_u^l be the rating of the item that appears in the l-th position of list $\mathcal{L}_B(u)$ of the baseline RS.

Assume that an item i is recommended to a user u in position $l = 1, \ldots, L$ in the list $\mathcal{L}(u)$ and substitutes the item that was recommended in that position in the baseline RS. The *cost of ratings deviation* or *cost of substitution* is defined as the absolute difference of ratings between these two items, namely

$$c_{iu}^l = |r_{iu} - f_u^l| \,. \tag{1}$$

B. Problem P1: Item coverage with minimum deviation cost from baseline RS

First, we are interested in constructing a possibly new recommendation list $\mathcal{L}(u)$ for each user u so that each item is d-covered. Let x_{iu}^l be the binary variable such that $x_{iu}^l = 1$ if item i is recommended to user u at position l in list $\mathcal{L}(u)$, and 0 otherwise. Let $\mathbf{x} = (x_{iu}^l : i \in \mathcal{I}, u \in \mathcal{U}, l = 1, \ldots, L)$ denote the overall item recommendation policy. We define the *coverage* of an item i as follows:

$$\text{Cov}_i(\mathbf{x}) = \sum_{u=1}^{K} \sum_{l=1}^{L} x_{iu}^l \,. \tag{2}$$

When recommending items to users at different positions in their lists, we need to keep the substitution cost as small as possible. Namely, we aim at recommending to each user u at a position l in list $\mathcal{L}(u)$ an item that has a rating as close as possible to the rating of the item that the baseline RS would recommend in this position. Then, the deviation from the baseline system would be small, and therefore the disturbance to the user would be small as well. In addition, a certain amount of deviation should count more if it occurs at a higher position in the list, i.e. for small values of l, since the items in these positions usually get more user attention. For a user u, this deviation cost in terms of total absolute difference in item ratings from the baseline RS is

$$\text{UDev\_Cost}_u(\mathbf{x}) = \sum_{i=1}^{n} \sum_{l=1}^{L} w_u^l c_{iu}^l x_{iu}^l \,. \tag{3}$$

If we consider the point of view of an item i, we can define the total deviation for item i compared to the baseline RS as

$$\text{IDev\_Cost}_i(\mathbf{x}) = \sum_{u=1}^{K} \sum_{l=1}^{L} w_u^l r_{iu} (x_{iu}^l - I_{iu}^l) \,. \tag{4}$$

where the indicator function $I_{iu}^{l} = 1$ if item i appears in the list of recommended items of user u at position l in the baseline RS. This cost takes into account the relative positions in recommendation lists of different users where item i appears, and it also accounts for the case where item i may not appear in the new list $\mathcal{L}(u)$ of a user u while it did so in the baseline RS. Note that the notation $\text{UDev\_Cost}_u(\mathbf{x})$ and $\text{IDev\_Cost}_u(\mathbf{x})$ describe the user (U) and item (I) deviation cost respectively.

The requirement for item coverage changes drastically the recommendation setting. In the baseline RS, the lists of recommended items are constructed separately for each other since lists are decoupled across different users, and the only criterion for creating the lists of recommended items is to minimize total accuracy i.e. the total mean squared error. However the coverage constraint couples the way the different lists of recommended items $\mathcal{L}(u)$ to users are constructed. The problem can be formulated as follows:

$$\min_{\mathbf{x}} \sum_{u=1}^{K} \text{UDev\_Cost}_u(\mathbf{x}) \,, \tag{5}$$

subject to:

$$\text{Cov}_i(\mathbf{x}) \geq d \,, \forall \text{ item } i \,, \tag{6}$$

$$\sum_{i=1}^{n} x_{iu}^{l} = 1, \ \forall \text{ user } u, \forall \ l = 1, \ldots, L \,, \tag{7}$$

and variables

$$x_{iu} \in \{0, 1\} \ \forall \ i \in \mathcal{I}, \ u \in \mathcal{U} \,. \tag{8}$$

Constraint (6) says that each item should be recommended to at least d users, while (7) says that one item must be allocated at each position in the list of each user. We refer to the problem (5)-(8) above as problem (**P1**).

Problem (**P1**) is an integer programming one, and for $L = 1$ it may be viewed as an instance of a generalized assignment problem (GAP) which is known to be NP-Hard [2].

C. Problems P2 and P3: Item coverage with balanced deviation cost

1) Balanced deviation cost across users: The approach we discussed above aims to minimize the total cost of deviation from the baseline RS while satisfying the constraints for item coverage. However, the minimization of the total deviation cost does not provide guarantees about individual deviation costs incurred to the lists of recommended items to each user. It may happen that a solution to the problem (**P1**) results in uneven deviation costs across users in the sense that for some users the substitution cost may be high while for others low. It may also be possible that only few users bear the cost of substitution, and thus for some users the new recommendation would be less accurate than others.

Thus we would like to perform item coverage so that the substitution cost is as balanced as possible across different users. This cost balancing can be seen as an instance of treating users in a fair manner in terms of recommendation accuracy. This problem can be formulated as follows,

$$\min_{\mathbf{x}} \max_{u} \text{UDev\_Cost}_u(\mathbf{x}) \tag{9}$$

subject to (6)-(8), and it is referred to as problem (**P2**).

2) Balanced deviation cost allocation across items: Deviation cost balancing subject to guaranteeing item coverage may be considered across items as well, and therefore across their owners. Then, different items would be treated fairly in terms of bearing the cost of deviation from the baseline RS in order to provide d-coverage. This version of the problem is plausible when the recommendation engine places emphasis on fair treatment of item owners that have paid it in order to guarantee coverage of their items, by providing as equal recommendation accuracy as possible across different item owners. This problem is formulated as

$$\min_{\mathbf{x}} \max_{i} \text{IDev\_Cost}_i(\mathbf{x}) \tag{10}$$

subject to (6)-(8), and it is referred to as problem (**P3**).

III. Solution approaches

A. Solving the minimum deviation cost problem (P1)

1) Linear Programming (LP) relaxation for P1: In problem (**P1**), we allow variables $\{x_{iu}\}$ to take continuous values in $[0, 1]$. Then the relaxed problem (**P1**) becomes a Linear Programming (LP) one, and we refer to the relaxed problem as (**P1'**). Next, we write the constraints of the LP problem (6), (7) and $0 \leq x_{iu} \leq 1$ for all $i \in \mathcal{I}$ and $u \in \mathcal{U}$ in the typical form in LP, namely

$$\mathbf{A}\mathbf{x} \leq \mathbf{b}, \ \mathbf{x} \in \mathbb{R}_{+} \,, \tag{11}$$

where $\mathbb{R}_{+}$ is the set of positive real numbers. In this formulation, $\mathbf{x} = (x_{iu}^{l} : i \in \mathcal{I}, u \in \mathcal{U}, l = 1, \ldots, L)$ is the vector of continuous variables, $\mathbf{A}$ is the constraint matrix of dimension $(n+nKL)\times nKL$, and $\mathbf{b}$ is the vector of dimension $(n + nKL)$ in which the first n entries are equal to $-d$, and the next nKL entries are equal to 1. We have the following definition:

Definition 2. *A constraint matrix* $\mathbf{A}$ *of an LP problem is called totally unimodular (TU) if every square submatrix of* $\mathbf{A}$ *has determinant* $+1$, -1 *or* 0.

Theorem 1. *([3], Section 3.2) A matrix* $\mathbf{A}$ *is totally unimodular (TU) if the following three conditions hold:*

- *its elements* $a_{ij} \in \{+1, -1, 0\}$ *for all* i, j,
- *each column contains at most two nonzero elements, and*
- *there exists a partition of the set of rows in two subsets* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *such that each column* j *with two nonzero coefficients satisfies* $\sum_{i\in\mathcal{M}_1} a_{ij} - \sum_{i\in\mathcal{M}_2} a_{ij} = 0$.

In our case, one can observe that matrix $\mathbf{A}$ of the LP problem (**P1'**) above satisfies the three conditions of the theorem, and therefore $\mathbf{A}$ is TU. Next, have the following proposition, again from [3].

Proposition 1. *A LP problem with feasible set* $\{\mathbf{A}\mathbf{x} \leq \mathbf{b},\ \mathbf{x} \in \mathcal{R}_+\}$ *has an integral optimal solution for all integer vectors* $\mathbf{b}$ *for which it has an optimal value, if and only if matrix* $\mathbf{A}$ *is TU.*

In our case, vector $\mathbf{b}$ is an integer vector. Therefore, the optimal solution of the LP problem is integral, and since $0 \leq x_{iu} \leq 1$, the optimal solution of the LP is the optimal 0-1 solution. It follows that by solving the LP problem, we can find the optimal solution to our original problem (**P**).

2) Greedy heuristic algorithm: The solution approach of the previous subsection can be applied to solve small-to-medium-sized problem instances. However, solving larger instances becomes challenging since the complexity of the Simplex method or any other method that is applied to solve the LP problem quickly grows with the number of variables and the number of constraints. Especially the worst-case complexity of the Simplex method is exponential since the number of vertices of the feasible solution set is exponential in the number of variables.

In order to solve larger instances of the problem, a low-complexity algorithm would be needed. To this end, we consider the class of greedy algorithms. Before explaining the rationale for our choice, we give a few definitions.

Definition 3. *Consider a set of elements* $\mathcal{X}$ *and a set function* $f : 2^{\mathcal{X}} \rightarrow \mathbb{R}$*, and let* $\mathcal{S} \subseteq \mathcal{X}$ *and* $e \in \mathcal{X}$*. The quantity* $\Delta f(e \,|\, \mathcal{S}) = f(\mathcal{S} \cup \{e\}) - f(\mathcal{S})$ *is called the discrete derivative of* f *with respect to element* e*.*

Definition 4. *A set function* $f : 2^{\mathcal{X}} \rightarrow \mathbb{R}$ *is called submodular (supermodular) if for every* $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{X}$*, and element* $e \in \mathcal{X}$*, it is* $\Delta f(e \,|\, \mathcal{A}) \geq \Delta f(e \,|\, \mathcal{B})$ *(*$\Delta f(e \,|\, \mathcal{A}) \leq \Delta f(e \,|\, \mathcal{B})$*).*

Equivalently, a set function is submodular (supermodular) if for every $\mathcal{A}, \mathcal{B} \subseteq \mathcal{X}$*, it is* $f(\mathcal{A} \cap \mathcal{B}) + f(\mathcal{A} \cup \mathcal{B}) \leq f(\mathcal{A}) + f(\mathcal{B})$ *(*$f(\mathcal{A} \cap \mathcal{B}) + f(\mathcal{A} \cup \mathcal{B}) \geq f(\mathcal{A}) + f(\mathcal{B})$*).*

Consider the objective (5) of problem (**P**). If we take $L = 1$ for simplicity and consider the set $\mathcal{X}$ of all possible item-user pairs, we can rewrite the problem objective as

$$\min_{\mathcal{S} \subseteq \mathcal{X}} \sum_{p \in \mathcal{S}} c_p \,, \tag{12}$$

where $p = (i, u)$ and $c_p \equiv c_{iu}$.

The problem objective (5) belongs to a special class of functions that are both submodular and supermodular, and they are called *modular* functions [4]. For modular functions, it is $\Delta f(e \,|\, \mathcal{A}) \geq \Delta f(e \,|\, \mathcal{B})$ for all $\mathcal{A}, \mathcal{B}$, and $e \notin \mathcal{A} \cup \mathcal{B}$ and $f(\mathcal{A} \cap \mathcal{B}) + f(\mathcal{A} \cup \mathcal{B}) = f(\mathcal{A}) + f(\mathcal{B})$.

Greedy algorithms are known to provide certain worst-case performance guarantees for submodular functions [4]. Therefore, it makes sense to seek an efficient heuristic algorithm for the problem among the class of greedy heuristic algorithms.

The algorithm starts by initializing the lists of recommended items to each user u as $\mathcal{L}_B(u)$, namely those of the baseline RS. If all items are already d-covered with the baseline RS, then this is the optimal solution to the problem and at zero substitution cost. On the other hand, if no item is covered with the baseline RS, then the problem is not feasible. The algorithm greedily makes item substitutions starting with the lower-cost ones, until item coverage constraints are satisfied.

Input: $\forall u \in \mathcal{U}$, list $\mathcal{L}_B(u)$ (baseline RS)
Result: Lists $\mathcal{L}(u)$, $\forall u \in \mathcal{U}$
for *u=1 to K* **do**
  Initialize $\mathcal{L}(u) = \mathcal{L}_B(u)$;
end
$\mathcal{I}_1 = \{i : \text{Cov}_i(\cdot) > d\}$;
$\mathcal{I}_2 = \{i : \text{Con}_i(\cdot) < d\}$;
while $\mathcal{I}_2 \neq \emptyset$ **do**
  Perform substitution of item i' with item i such that,

$$(i^*, i'^*, u^*, l^*) = \arg \min_{\substack{i' \in \mathcal{I}_1, i \in \mathcal{I}_2 \\ u \in \mathcal{U}, l = 1, \ldots, L}} c^l_{iu} \tag{14}$$

  Replace item i'^* in the list of user u^* with i^* and update list $\mathcal{L}(u^*) = \mathcal{L}(u^*) \setminus \{i'^*\} \cup \{i^*\}$;
  Update $\mathcal{I}_1$, $\mathcal{I}_2$;
end

Algorithm 1: Greedy algorithm for min-deviation-cost item substitution

At each step of the algorithm, the idea is to choose to substitute an already covered item in the list of recommended items to a user with an uncovered one such that the substitution cost is minimal among all possible eligible substitutions. At each step, we keep track of the following sets of items, the covered ones, $\mathcal{I}_1 = \{i : \text{Cov}_i(\cdot) > d\}$ and the uncovered ones, $\mathcal{I}_2 = \{i : \text{Cov}_i(\cdot) < d\}$. We pick an uncovered item i^*, a covered item i'^* a user u^* and position l^* as follows,

$$(i^*, i'^*, u^*, l^*) = \arg \min_{\substack{i' \in \mathcal{I}_1, i \in \mathcal{I}_2 \\ u \in \mathcal{U}, l = 1, \ldots, L}} c^l_{iu} \tag{13}$$

such that item i' under substitution is currently at position l i.e. $f^l_u = r_{i'u}$. After selecting and performing the substitution of item i' with i, we update sets $\mathcal{I}_1, \mathcal{I}_2$. Care should be taken so as not to make a covered item uncovered after a substitution. We stop when either all items are covered i.e. $\mathcal{I}_2 = \emptyset$. The sketch of the greedy algorithm is given above in Algorithm 1.

B. Solving the balanced deviation cost problems P2 and P3

In order to solve problem (**P2**), we define an auxillary real-valued variable $w = \max_i \text{UDev\_Cost}_u(\cdot)$. Then problem (**P2**) is transformed to the following one, (**P2'**).

$$\min_{\mathbf{x}, w} w \tag{15}$$

subject to:

$$\text{Cov}_i(\mathbf{x}) \geq d \,, \forall \text{ item } i \tag{16}$$

$$\sum_{i=1}^{n} x^l_{iu} = 1 \,\forall \text{ user } u,\ \forall\ l = 1, \ldots, L \,. \tag{17}$$

$$w \geq \text{UDev\_Cost}_u(\mathbf{x})\ \forall u \in \mathcal{U} \tag{18}$$

$$w \in \mathbb{R} \text{ and } x_{iu} \in \{0, 1\}\ \forall\ i \in \mathcal{I},\ u \in \mathcal{U} \,. \tag{19}$$

Problem (**P2'**) is Mixed-Integer Linear Programming (MILP) problem. We can again relax it to a LP problem by allowing variables x_{iu} to take continuous values in $[0, 1]$. However, it turns out that the constraint matrix of this LP problem is not TU, and therefore the optimal solution of the LP problem is not integral. In this case, the objective function (9) evaluated at the optimal solution of the LP problem provides a lower bound on the value of the objective function for the original problem (**P2'**). An integral solution to (**P2'**) can be obtained through rounding of the optimal solution of the LP problem subject to feasibility constraints. A greedy heuristic similar in nature to Algorithm 1 can be devised.

A similar rationale holds for the solution of problem (**P3**).

IV. Numerical results

We conduct experiments on real-world datasets in order to evaluate the performance of our algorithms in terms of recommendation quality (accuracy) and item coverage metrics. We have used an *item-item Collaborative Filtering(CF)* method as a baseline RS to derive the initial recommendation lists for users. We have implemented the baseline RS and our proposed recommendation algorithms in Python.

A. Dataset

We experimented with the real-world dataset *Movielens* 100K [6] which contains the results of interaction between individual users and movies. Specifically, each record of the dataset has the following format: `userId, movieId, rating, timestamp`; thus it contains information of interest about the user and ratings she has given for each movie.

Ratings are on a 5-star scale, with half-star increments, namely the set of ratings is $\{0.5, 1.0, 1.5, \ldots, 5.0\}$ stars. The initial dataset contains $100,000$ ratings from 671 users and $9,066$ movies. We chose a subset of items and users that would be suitable for our experimental study with regard to coverage and cost constraints.

In order to facilitate the rating prediction process, we kept in our subset those users that have rated at least two movies. Also we excluded users that have given the same ratings for all movies in the considered dataset, since the item-item CF algorithm provides similar rating prediction for all movies in the dataset; in the end, a subset of 635 users and 151 movies were kept for our experiments. The range of values of d in the plots are those for which the problem instance is feasible. Clearly, different numbers of users and items would give a different range of values of d for which the problem would be feasible.

B. Experimental results

1) Item coverage with minimum deviation cost from baseline RS: First, we study the performance of the item coverage recommendation approach in problem (**P1**). The plot in Figure 1 shows the total deviation cost from the baseline RS versus the d-coverage threshold in constraint (6), when $L = 1$ i.e. only one item is recommended to each user. Namely, the horizontal axis is d, viewed as the percentage of users to which an item is recommended, and the vertical axis is the total deviation cost for users due to item substitution. This is given as

$$\text{Total\_Cost} = \sum_{u \in \mathcal{U}} \text{UDev\_Cost}_u(\cdot)$$

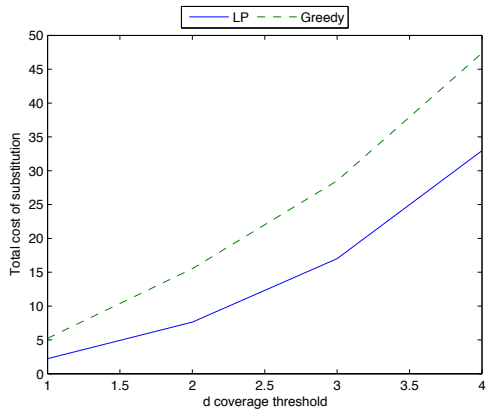

Fig. 1. Total substitution cost vs. the $d-$coverage threshold for $L = 1$ for the LP-based approach that gives the optimal integral solution, and for the greedy algorithm.

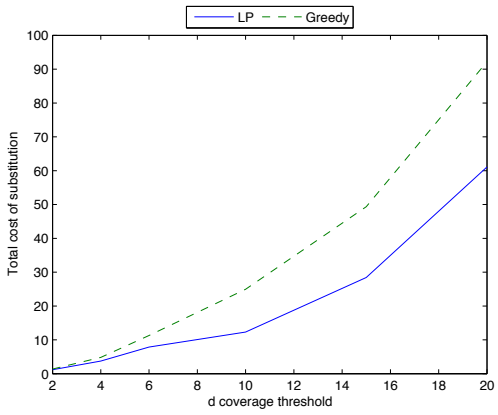

Fig. 2. Total substitution cost vs. the $d-$coverage threshold for $L = 5$ for the LP-based approach that gives the optimal integral solution, and for the greedy algorithm.

We plot results for the LP approach that gives the optimal solution and for the proposed greedy algorithm. Figure 1 shows that the total deviation cost increases as the d-coverage threshold increases for both approaches. The LP solution consistently gives much lower substitution cost than the greedy approach, and this difference tends to increase as the value of d increases. This is attributed to the fact that for larger values of d, the constraints for item coverage become more stringent to satisfy. Interestingly, the curves in the plot are convex ones, which means that the differential deviation cost that guarantees d-coverage increases as d increases.

Similar results can be observed in Figure 2, where we plot results for $L = 5$. Again, the cost difference between the LP solution and the greedy approach increases as d increases. As

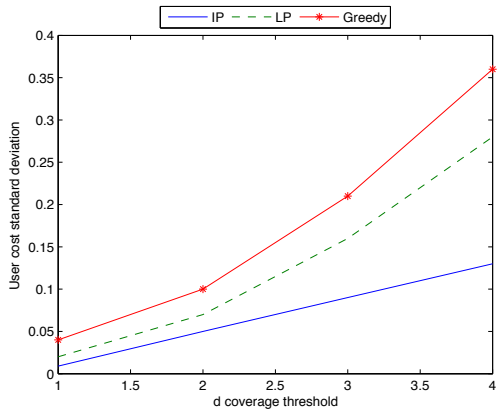

Fig. 3. Standard deviation of user costs vs. d-coverage threshold $L = 1$ for the rounded Linear Programming approach (LP), the Integer Programming GLPK package solution (IP) and the greedy approach.

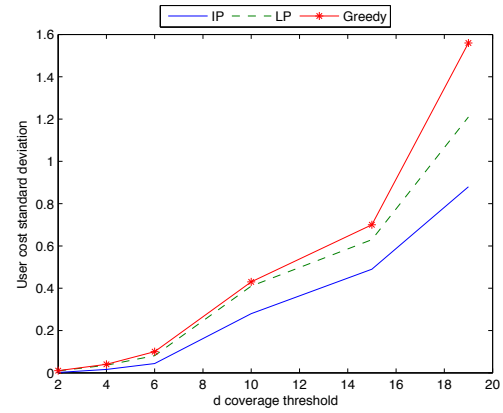

Fig. 4. Standard deviation of user costs vs. the $d-$coverage threshold for $L = 5$ or the rounded Linear Programming approach (LP), the Integer Programming GLPK package solution (IP) and the greedy approach.

expected, the cost difference between the LP solution and the greedy approach decreases as L increases due to the higher flexibility to recommend items. We can also observe that the difference between the LP and the greedy approach is very small for small values of d.

2) Item coverage with balanced deviation cost allocation: The second part of our experiments refers to the item coverage problem with as fair substitution costs across users as possible, namely problem (**P2**). We use the variance of user costs $\{\text{UDev\_Cost}_u\}$, for $u \in \mathcal{U}$ to measure how balanced the costs incurred to different users are. This is computed as

$$\text{var\_UDev\_Cost} = \frac{1}{K}\sum_{u=1}^{K}(\text{UDev\_Cost}_u - \mu)^2 \qquad (20)$$

where $\mu = \frac{1}{K}\sum_{u=1}^{K}\text{UDev\_Cost}_u$ is the average of the individual user deviation costs.

We plot the standard deviation σ as a metric to evaluate the performance of the approach in terms of fair cost allocation, $\sigma = \sqrt{\text{var\_UDev\_Cost}}$. We compare three approaches: (*i*) one in which we solve the LP problem and we round the solution to 0 or 1 subject to feasibility; we refer to that as the LP appoach; (*ii*) one where we solve the problem with Integer Programming through the GLPK solver of Python; we call this the IP approach, and (*iii*) the greedy algorithm.

In Figures 3 and 4 we plot the standard deviation of user costs for the three approaches above, as a function of the d-coverage threshold, for $L = 1$ and for $L = 5$ respectively. The IP solution gives more balanced solutions than the other two approaches. In all cases, the standard deviation of user costs increases as d increases, again due to the fact that the item coverage constraint becomes harder to satisfy. For low values of d, all three approaches (IP, LP and greedy) give similar results, which for $L = 5$ reveal very balanced costs across users. Also, the difference between the rounded LP and the greedy approach decreases when L increases. For $L = 5$, the rounded LP and the greedy approaches have similar performance, and both get quite close to the performance of the IP solution.

V. Related work

A. Coverage, diversity and other metrics in RSs

RSs have been mostly researched from the point of view of accuracy, namely the closeness of predicted item ratings to true ones. However, accuracy is challenged as a representative metric for RSs (see e.g. [5], [7]). An extended survey on objectives beyond accuracy is presented in [8], where the authors discuss various other plausible metrics (e.g. diversity, serendipity, novelty) in RSs with the aim to recommend diverse, novel or unexpected items to users. They also discuss item coverage, i.e. recommending as many items as possible to users and user coverage, i.e. issuing recommendations to as many users as possible. Finally, they discuss optimization and reranking policies for these metrics. Different definitions of diversity in individual and group recommendations are presented in [9]. Aggregate diversity of recommendations across all users and different ranking approaches to achieve it are studied in [10], whereby diversity gain is measured across the lists of recommended items to all users.

Ge *et.al.* [11] argue further in favor of coverage and serendipity as evaluation metrics for RSs. They consider two definitions of coverage, (*i*) prediction coverage, which measures the percentage of items for which the system makes a recommendation, and (*ii*) catalog coverage, defined as the percentage of available items that are recommended to a user. They discuss how these metrics are related to and can complement accuracy. A different version of the coverage problem is studied in [12], where user types (tastes) change over time, and the problem is to accommodate as many of user types as possible. A multi-armed bandit approach is used to select the items to recommend each time.

In [13], the authors deal with the problem of selecting k products that cover the largest number of customers so that they maximize the chances that a customer purchases products. The problem takes as input user subsets where each subset has

purchased a product, and it is modeled as a maximum coverage one. Another aspect of diversity and coverage is introduced in [14] with respect to different genres of items that appear in recommendation lists. Genre coverage ensures that each genre is represented in a recommendation list, while genre redundancy implies that all genres are represented.

B. Notions of fairness in RSs

Fairness in RSs has recently attracted interest. Fairness is mostly studied in the context of group recommendations, whereby a set of items needs to be recommended to a group of users so that each group member is satisfied in a fair manner. Serbos *et.al* [15] explore two definitions of fairness, (*i*) fairness proportionality, namely ensuring that each user likes a sufficient number of items in her recommendation list compared to items not in the list, and (*ii*) envy-freeness, namely that each user likes a sufficient number of items in her list most than other users. They study the problem of finding the fairest package for a group of users by modeling it as a coverage problem. In [16], the authors investigate the problem of optimizing both user utility and utility fairness in group recommendations, and they propose a general multi-objective optimization framework based on Pareto efficiency.

In the position paper [17], different sides of fairness are discussed, related to customers (C-fairness), product providers (P-fairness) or both (CP-fairness). P-faireness could also be viewed through the lens of diversity-aware RSs (e.g. [18]) that aim to maintain a certain level of accuracy while ensuring that recommendation lists are diverse.

C. Our work in perspective

Price and profit awareness in RSs need to be addressed whenever recommended items incur monetary gains for the online platform in which the recommendation engine is embedded, as discussed in the position paper [19]. Relevance of recommended items to users needs to be balanced against potential profits stemming from user clicks. In this paper, we advocate this latter perspective. Different from existing works in the literature, we present a systematic framework and the associated mathematical optimization problems that aim to provide guarantees on *item coverage* so that items appear in the lists of recommended items of a certain number of users.

VI. Conclusion

We studied the problem of guaranteeing item coverage in RSs with the goal to have minimum deviation cost in terms of ratings of recommended items to users compared to a baseline RS, and to have these costs allocated evenly across users and across items. Our works paves the way for a novel and systematic approach, driven by mathematical optimization for performing recommendations under item coverage constraints. For the problem of minimizing total deviation cost, we show that a LP relaxation gives the optimal recommendation. For the problem of balancing deviation costs across users or items, LP solution rounding or a greedy approach would work well.

The employed optimization-driven framework is generic and could be applied to provide guarantees in terms of other metrics in RSs such as novelty, diversity or serendipity. These cases would require appropriate modifications in problem formulation. Finally the notion of fairness in RSs deserves further investigation. In this work, fairness was viewed as minimizing the maximum deviation cost across users or across items. In RSs, the training dataset that is used to predict ratings is composed out of contributions of ratings of different users. A further possibility for fair user treatment for the RS would be to provide recommendation accuracy to a user commensurate to her contribution in the system in terms of number of rated items. The total recommendation accuracy is the cost of the RS as a whole, and different approaches could be applied for fair cost allocation across users, using perhaps notions from cooperative game theory.

Acknowledgment

This work was supported by the European Commission H2020 Research Program under Grant Number 688768 netCommons (Network Infrastructure as Commons) and by AUEB-RC through the internal project Original scientific publications". Also the authors would like to thank N. Karagkounis, K. Mataragka and A. Ntokos for their assistance with experimental study.

References

[1] L. Bobadilla, F. Ortega, and H. A. and GutiéRrez, "Recommender Systems Survey", *in Elsevier Journal of Know.-Based Syst.,*, vol.46, pp.109-132, July 2013.

[2] M. R. Garey and D.S. Johnson. Computers and Intractability: A guide to the theory of NP-Completeness, Freeman, 1979.

[3] L.A. Wolsey, Integer Programming, *Wiley*, 1998.

[4] A. Krause and D. Golovin, "Submodular function maximization", in Tractability: Practical Approaches to Hard Problems, Cambridge University Press, 2013.

[5] S. Mcnee, J. Riedl, and J. Konstan., "Accurate is not always good: How Accuracy Metrics have hurt Recommender Systems", *in Proc. Conf. on Human Factors in Computing Systems (CHI),* 2006.

[6] https://grouplens.org/datasets/movielens/100k/.

[7] J. Herlocker, J. Konstan, L.G. Terveen, and J. Riedl, "Evaluating collaborative filtering recommender systems", *in ACM Trans. on Inform. Sys.,* vol. 22, no.1, pp. 553, Jan. 2004.

[8] M. Kaminskas and D. Bridge, "Diversity, Serendipity, Novelty, and Coverage: A Survey and Empirical Analysis of Beyond-Accuracy Objectives in Recommender Systems", *in ACM Trans. on Interactive Intelligent Sys.,* vol. 7, no. 1, Article 2, March 2017.

[9] M. Kyriakidi, K. Stefanidis and Y. Ioannidis, "On Achieving Diversity in Recommender Systems", *in Proc. ExploreDB,* 2017.

[10] G. Adomavicius and Y. Kwon, "Improving aggregate recommendation diversity using ranking-based techniques", *IEEE Trans. on Knowledge and Data Engineering,*, vol.24, no.5, pp.896-911, May 2012.

[11] M. Ge, C. Delgado-Battenfeld, and D. Jannach, "Beyond Accuracy: Evaluating Recommender Systems by Coverage and Serendipity", *in Proc. ACM Conf. on Recommender Systems (RecSys),* 2010.

[12] M. Rahman and J.C. Oh, "A Graph-based Bandit Algorithm for Maximum User Coverage in Online Recommendation Systems", Tech. Report, Electrical Engineering and Computer Science, 2016.

[13] M. Hammar, R. Karlsson, and B. J. Nilsson, "Using maximum coverage to optimize recommendation systems in e-commerce, *in Proc. ACM Conf. on Recommender Systems (RecSys),* 2013.

[14] S. Vargas, L. Baltrunas, A. Karatzoglou, and P. Castells, "Coverage, Redundancy and Size-Awareness in Genre Diversity for Recommender Systems", *in Proc. ACM Conf. on Recommender Systems (RecSys),* 2014.

[15] D. Serbos, S. Qi, N. Mamoulis, E. Pitoura, and P. Tsaparas, "Fairness in Package-to-Group Recommendations", *in Proc. Int. Conf. on World Wide Web (WWW),* 2017.
[16] X. Lin, M. Zhang, Y. Zhang, Z. Gu, Y. Liu and S. Ma, "Fairness-Aware Group Recommendation with Pareto-Efficiency", *in Proc. ACM Conf. on Recommender Systems (RecSys),* 2017.
[17] R. Burke, "Multisided Fairness for Recommendation", arXiv preprint arXiv:1707.00093, 2017.
[18] S. Vargas and P. Castells, "Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems", *in Proc. ACM Conf. on Recommender Systems (RecSys),* 2011.
[19] D. Jannach and G. Adomavicious, "Price and profit awareness in recommender systems", *in Proc. Wksp. on Value-Aware and Multi-Stakeholder Recommen- dation (VAMS),* in ACM RecSys 2017.