User-Item Matching for Recommendation Fairness

QIANG DONG[ID]1, SHUANG-SHUANG XIE1, AND WEN-JUN LI2
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[2]School of Software and Service Outsourcing, Suzhou Vocational Institute of Industrial Technology, Suzhou 215004, China

Corresponding author: Qiang Dong (dongq@uestc.edu.cn)

ABSTRACT As we all know, users and item-providers are two main parties of participants in recommender systems. However, most existing research efforts on recommendation were focused on better serving users and overlooked the purpose of item-providers. This paper is devoted to improve the item exposure fairness for item-providers' objective, and keep the recommendation accuracy not decreased or even improved for users' objective. We propose to set stock volume constraints on items, to be specific, limit the maximally allowable recommended times of an item to be proportional to the frequency of its being interacted in the past, which is validated to achieve superior item exposure fairness to common recommenders and thus mitigates the Matthew Effect on item popularity. With the two constraints of pre-existing recommendation length of users and our stock volumes of items, a heuristic strategy based on normalized scores and a Minimum Cost Maximum Flow (MCMF) based model are proposed to solve the optimal user-item matching problem, whose accuracy performances are even better than that of baseline algorithm in regular recommendation context, and in line with state-of-the-art enhancement of the baseline. What's more, our MCMF based strategy is parameter-free, while those counterpart algorithms have to resort to parameter traversal process to achieve their best performance.

INDEX TERMS Minimum cost maximum flow, popularity bias, recommendation fairness, recommender systems, stock constraint.

I. INTRODUCTION

Even though the broad social and business acceptance of recommender systems has been achieved, a key underexplored dimension for further improvement is the usefulness of recommendations to the participants [1]. A recommender system usually serves two main parties of participants, the users and the item-providers [2], thus the usefulness of recommendations should be also two-fold. On one hand, recommender systems provide users with items of their latent interests. On the other hand, recommender systems should also help item-providers increase sales volume of items, especially the unpopular ones.

Common recommenders, such as collaborative filtering, originally proposed to make accurate prediction of unseen user-item interactions, usually suffer the *popularity bias problem*, i.e., recommending a few popular items to a majority of users [3]. Although popular items are likely to match users' preferences and recommendations of them contribute to the predictive accuracy, users usually do not regard them as very useful recommendations because they are easily aware of these popular items somewhere else, for example from sales leaderboard, advertisements, or friends' conversations. Thus as a complement, the intra-list diversity [4] is introduced to measure how well a recommender can widen a user's vision of items, usually by means of offering users less popular, unexpected but interesting items.

The popularity bias problem also hampers the sales promotion of unpopular items, which is the main usefulness of recommendations for item-providers. Just like the intra-list diversity is used to measure how serious is the popularity bias from the point of view of individual users, we also need to measure this concentration problem for the purpose of item-providers. Therefore, The Gini coefficient is borrowed

The associate editor coordinating the review of this manuscript and approving it for publication was Feiqi Deng[ID].

from the economic filed to quantify the balance degree in the numbers of recommended times of different items [5], which is called *exposure fairness* in this paper.

The most straightforward approach to improve the exposure fairness is, first generating a relatively large recommendation list with a classical model, and then performing post-hoc re-ranking on that recommendation list. Abdollahpouri [6] re-ranked the generated recommendation list by considering the item popularity, and Christoffel *et al.* [7] simply divided each item's recommendation score by its degree of popularity with an adjustable exponent on the shoulder, which greatly decreases the originally high recommendation scores of popular items and thus enhances the priority of unpopular items in the recommendation lists. Dong *et al.* [8] proposed to linearly aggregate the row- and column-ranking numbers of the recommendation score matrix obtained by some algorithm into the final recommendation ranking number, where the row and column of the matrix are corresponding to user and item, respectively. Mansoury *et al.* [9] introduced a general graph-based algorithm for improving item recommendation fairness. The algorithm iteratively finds items that are rarely recommended yet are high-quality and add them to the users' final recommendation lists, which is done by solving the maximum flow problem on the recommendation bipartite graph. Item fairness in multi-round recommendation context also received research attentions. Patro *et al.* [10] focused on the item fairness issues arising out of incremental updates of the platform algorithms. They formulated an ILP based online optimization to ensure smooth transition of the exposure of items while guaranteeing a minimum utility for every user. Ge *et al.* [11] explored the problem of long-term exposure fairness of items in dynamically changing groups of different popularity levels. They proposed a fairness-constrained reinforcement learning algorithm based on Constrained Markov Decision Process (CMDP), so that the model can dynamically adjust its recommendation policy.

Although the improved exposure fairness brought by these methods may be significant in terms of improved percentage, for example of more than 100%, the improved absolute value is usually trivial, for example from 0.0378 to 0.0859 (see the values of exposure fairness of original P3 and RP3 algorithms on the Movielens data set in Table 3). It is hard to say that this kind of improvement of exposure fairness will better serve the purpose of item-providers. Here what we should not ignore is that, Ren *et al.* [12] improve the exposure fairness to a nontrivial absolute value, but with the cost of unacceptable loss of recommendation accuracy.

Our main task is to significantly improve the exposure fairness, and simultaneously keep the recommendation accuracy not decreased or even improved. The main contributions of this paper are:

(1) We propose to set stock volume constraints on items, to be specific, limit the maximally allowable recommended times of an item to be proportional to the frequency of its being interacted in the past, which is validated to achieve

superior item exposure fairness to common recommenders and thus mitigates the Matthew Effect on item popularity.

(2) For the purpose of users, two heuristic user-item matching strategies are proposed to minimize the loss of recommendation accuracy brought by item stock volume constraints. Among them, the parameterized strategy is validated to achieve better recommendation accuracy than the baseline algorithm in regular recommendation scenario, and it has an advantage of relatively low time complexity.

(3) A Minimum Cost Maximum Flow based model is designed to solve the optimal user-item matching problem with constraints. The recommendation accuracy of this strategy is in line with state-of-the-art enhancement of the baseline algorithm, but it is parameter-free and get rid of parameter traversal process of its counterpart enhancements to achieve their best performance.

II. RELATED WORKS

Generally speaking, the primary goal of a recommender system is to enhance the engagement of users by providing them with items of potential interests. Although the usefulness of recommendation results is usually evaluated by accuracy measures (how accurate they are), the literature has introduced different evaluation measures of the quality of recommendation results from different perspectives [13]. The most frequently-used and extensively-studied types of beyond-accuracy measures are coverage, novelty, diversity and fairness.

However recently, many bias types have been recently discovered and categorized into bias in data, bias in model and bias in results, respectively arising from three different stages of the recommendation feedback loop [3], deteriorating the recommendation quality in terms of the above-mentioned beyond-accuracy aspects and challenging to achieve qualified recommendations. Among these bias types, popularity bias is the most prominent one due to its highly adverse effects on beyond-accuracy recommendation quality. In the literature, many methods have been developed for mitigating the bias problems in recommendations. These debiasing approaches can be divided into three main categories, pre-, in-, and post-processing methods, according to the three stages they participate in the recommendation process [14].

Since the data of user interactions are observational rather than experimental, the imbalance in user-item interaction data becomes one of the main factors accounting for popularity bias. Pre-processing approaches usually aim to reduce such inequalities by altering data on which recommendation algorithms are trained. For example, Park *et al.* [15] divide all the items into head group and tail group, where the head group consists of popular items with significantly larger amount of ratings than those in the tail group. Recommendations for tail items are produced using only ratings in the tail group, while those for head items are estimated using all data. Jannach *et al.* [16] present a practical popularity debiasing technique that first creates synthetic user-item tuples where the observed items are mostly unpopular and then utilizes

them to train algorithms. Chen *et al.* [17] derive a general learning framework that well summarizes most existing data debiasing strategies by specifying some parameters of the general framework. This provides a valuable opportunity to develop a universal solution for debiasing, e.g., by learning the debiasing parameters from data.

The in-processing approaches aim to modify the internal mechanisms of recommendation algorithms to simultaneously consider both popularity and relevance. This task is usually accomplished using specific constraints or conducting a joint optimization. For example, Abdollahpouri *et al.* [18] propose an optimized variant of the well-known RankALS algorithm, which contributes to producing recommendation lists where predictive accuracy and intra-list item diversity are balanced. Hou *et al.* [19] present a framework that first constructs the neighborhoods between the items based on their popularity instead of the magnitude of their ratings; then, it eliminates some most popular ones to have a more balanced common-neighbor similarity index. Boratto *et al.* [14] propose an in-processing approach that minimizes the biased correlation between user-item relevance and item popularity for fairly treating the items along the popularity tail. Berbague *et al.* [20] propose a solution to balance between the recommendation accuracy and coverage by making an overlapped clustering, where each user is assigned to a main cluster from which he gets his recommendations and to secondary clusters as a candidate neighbor.

The post-processing techniques usually aim to re-rank a recommendation list that has already been generated or create a new one following some specific constraints. These are the most utilized approaches for mitigating popularity bias since they can be easily applied to the output of any recommendation algorithm, which is also why we focus on developing a post-processing method to counteract potential popularity bias in this study. For example, Abdollahpouri *et al.* [21] introduce an approach that first calculates weight scores for items based on their popularity and then utilizes them to punish popular items during re-ranking recommended item lists. Likewise, Yalcin and Bilge [22] follow a similar strategy and presents two robust popularity debiasing methods for recommending to groups of users rather than individuals. Abdollahpouri *et al.* [23] also present the xQuad algorithm that is an enhanced re-ranking approach and helps balance the trade-off between long-tail item coverage and ranking accuracy more robustly. Wu *et al.* [24] exploited a balance factor to adjust the influence of a personalized ranking vector and a unified non-personalized ranking vector based on PageRank. By this, it can reduce the impact of item popularity on recommendations and then generate more diverse and novel recommendations to users.

As seen from the presented literature, many recent studies consider biases towards popular items in the recommendations and try to deal with this issue for achieving more qualified recommendations. However, as emphasized in the previous section, the popularity of an item does not always mean that individuals strongly desire it. Therefore, more comprehensive analyses are required to investigate potential bias towards blockbuster items and develop novel practical methods to mitigate its adverse effects on recommendations.

III. PROBLEM DESCRIPTION AND NOTATIONS
A. NOTATIONS
In an implicit rating based recommender system with m users and n items, u or u_i ($1 \leq i \leq m$) is used to denote a user, v or v_j an item ($1 \leq j \leq n$), and the adjacency matrix $A = (a_{i,j})_{m \times n}$ represents the historical user-item interaction records. The matrix element $a_{i,j}$ is 1 if there exists an observed user-item interaction, indicating that user u_i declared explicitly his/her preference on item v_j in the past, and the element is 0 if otherwise. The sum of row i is called the degree of user u_i, denoted by $deg(u_i)$, which is an indicator of the user's activity level in the system. The sum of column j is called the degree of item v_j, denoted by $deg(v_j)$, which represents the item's popularity degree among all users.

Given the user-item adjacency matrix A as input, a recommendation algorithm will output the user-item score matrix $S = (s_{i,j})_{m \times n}$, where the element $s_{i,j}$ is the predicted score of user u_i's preference on item v_j. The recommendation list of each user is constituted by the items with top l recommendation scores. Of course, the interacted items are excluded from the recommendation list. The final recommendation results of all users can also be represented by the recommendation matrix $R = (r_{i,j})_{m \times n}$, where $r_{i,j} = 1$ if item v_j is recommended to user u_i, otherwise zero. Clearly, the sum of every row of matrix R must equal the *recommendation length l*. All the notations used in this paper are listed in Table 1, some of which will be defined in the remaining part.

B. ITEM STOCK VOLUME CONSTRAINTS
The recommendation process is essentially selecting a subset of items to fill into the vacancies of the recommendation list of every user. Given user number m and recommendation length l, the total number of vacancies in all users' recommendation lists is $m * l$. In order to limit the recommendation frequency of popular items and promote the exposure of less popular items, we set the maximally allowed recommendation frequency of a specific item v_j, called *stock volume* q_j, to be proportional to its degree as follows,

$$q_j = \left\lceil m * l * \frac{\deg(v_j)}{\sum_{k=1}^{n} \deg(v_k)} \right\rceil.$$

The value is rounded up to an integer, because the total stock volume of all items must be no less than the vacancy number in the recommendation lists of all users.

By constraining the stock volume of every item, the relative popularity of different items will remain almost unchanged before and after recommendation process, and the Matthew effect on item popularity will be prevented from being aggravated, with the hypothesis that all the item recommendations shares the same conversion rate. Of course if the varying conversion rate is available, we can incorporate it as the weight of recommendation in the following fairness measure.

TABLE 1. The notations used in this paper.

Notation	Definition
m	Number of users
n	Number of items
i, j, k	Counter
u, u_i	Some user
v, v_j	Some item
$A = (a_{ij})_{m \times n}$	User-item adjacent matrix
$deg(u), deg(u_i)$	Degree of some user
$deg(v), deg(v_j)$	Degree of some item
$S = (s_{ij})_{m \times n}$	Matrix of recommendation scores, but s_{ij} is set to be 0 if $a_{ij} = 1$.
$S^N = (s_{ij}^N)_{m \times n}$	Row-normalized matrix of S
$R = (r_{ij})_{m \times n}$	Matrix of recommendation results, where $r_{ij} = 1$ if item v_j is recommended to user u_i, otherwise zero
l	Length of recommendation lists, or initial vacancy number of the recommendation list of every user
L	Vector of remaining vacancy numbers of the recommendation lists of m users
$L(i)$	Remaining vacancy number of the recommendation list of user u_i
q_j	Initial stock volume of item v_j
Q	Vector of remaining stock volumes of n items
$Q(j)$	Remaining stock volume of item v_j
$\mathbb{1}$	Vector consisting of all one elements

Next let us justify the feasibility of this constraint on item stock volume in a real-world recommendation context. In the service recommendation scenarios like dining, accommodation, fitness, haircuts, massages, medical services and so on, stock constraint is a special factor that decides how many customers can receive a service with an assured level of quality. For example, a restaurant often has a constraint on the number of customers who can be served during the dining hours. If too many customers arrive at a restaurant, their dining experience will be unpleasant or in the worst case, some customers will be very disappointed [25].

C. PROBLEM FORMALIZATION

In the regular recommendation scenario, there exists only constraint on the user side (the recommendation length) and no constraint on the item side, thus the recommendation lists of different users are generated independently of each other. That is to say, recommending which items to one specific user has nothing to do with the recommendation list of other users. In this paper, the item stock volume is introduced as one constraint on the item side. These two constraints together break the independency of different recommendation lists. The arising problem is, in order to achieve the best recommendation quality, how to match the items with limited stock volume into different users' recommendation lists of fixed length? For example, some regular algorithm is supposed to recommend a specific item to many users, but the stock volume of this item is fewer than the amount of these users.

This is an optimal user-item matching problem with constraints, where the optimization objective is undoubtedly the overall recommendation accuracy, and the two constraints are user's recommendation length and item's stock volume. Since user's future feedback is unknown during the recommendation generation phase, the most simple and intuitive alternative for the recommendation accuracy is the sum of recommendation scores of all matched user-item pairs in the recommendation lists.

Empirical results [26] show that there exists a strong positive correlation between the recommendation scores and the user degrees (item degrees, respectively). That is to say, almost all the users are regarded to prefer popular items to unpopular ones by common recommenders, and the recommendation scores on popular items of large-degree users is usually much larger than that of small-degree users. Therefore, following the above optimization objective, the large-degree users will almost run out the stocks of popular items, and the small-degree users who also prefer popular items have to be recommended less preferable unpopular items. Since the small-degree users are the absolute majority in the system, this will cause unacceptable loss of recommendation accuracy.

In order to eliminate the influence of user degree on the recommendation scores, we propose to use the user-normalized scores instead of original scores for the user-item matching priority. Specifically, the user-normalized score of every item is defined by the original score divided by the sum of all items' scores of the same target user,

$$s_{ij}^N = \frac{s_{ij}}{\sum_{k=1}^n s_{ik}}.$$

By means of the score normalization process, every user holds the same amount of recommendation stakes to be assigned to all the candidate items (the sum of normalized scores is one unit for every user). However, the distribution of stakes among items varies from small- to large-degree users. The stakes of small-degree users are concentrated on a small number of popular items, such that although small-degree users enjoy higher priority in the user-item matching process, the number of priority times is very few for each small-degree user. In this way, most stocks of popular items will be exhausted by small-degree users, and the large-degree users have to be recommended less popular items.

In summary, the optimization problem of user-item matching with constraints is formulated as

$$max \sum_{\substack{1 \le i \le m \\ 1 \le j \le n}} r_{ij} s_{ij}^N$$

$$s.t. \sum_{j=1}^n r_{ij} = l, \quad 1 \le i \le m$$

$$\sum_{i=1}^m r_{ij} \le q_j, \quad 1 \le j \le n.$$

IV. USER-ITEM MATCHING STRATEGY

To solve the optimal user-item matching problem with constraints, we propose two kinds of priority strategies, one is the simplest greedy strategy and the other one is the elaborate Minimum Cost Maximum Flow based strategy.

A. THE GREEDY STRATEGY

The essence of our greedy user-item matching strategy is "Largest Normalized-Score First". For this strategy, we sort all the non-interacted user-item pairs (u_i, v_j) in descending order of normalized scores, and check each pair one by one in this order. For a specific user-item pair, if there is some vacancy in the corresponding user's recommendation list and the corresponding item's stock volume is not exhausted, we fill the item into the recommendation list of the user, and decrease the corresponding vacancy number and stock volume by one. Once the recommendation lists of all users are fully occupied by items, we stop the above checking process. Algorithm 1 presents the detailed steps of this greedy strategy.

B. THE MCMF-BASED STRATEGY

As one important contribution of this paper, we build a Minimum Cost Maximum Flow (MCMF) model to solve the optimal user-item matching problem with constraints. The MCMF problem is a well-known network flow problem, which finds various applications in the fields of transportation, logistics, telecommunication, network design, resource planning, scheduling, and many other industries [27]. Next we give a brief review on the MCMF problem. The terminology follows from the reference [28].

A network is a directed graph $G = (V, E)$ with a source node s and a sink node t. Each directed edge $(u, v) \in E$ is associated with two constants, the capacity $cap(u, v)$ indicating the upper bound of the flow $f(u, v)$ allowed on the edge, and the cost per unit flow on the edge, denoted by $cost(u, v)$. Clearly, the capacity and the cost of an edge are positive values. The value of a network flow f is defined as $value(f) = \sum_{(s,w) \in E} f(s, w) - \sum_{(w,s) \in E} f(w, s)$, and the cost of flow f is $cost(f) = \sum_{(s,w) \in E} f(s, w) \times cost(s, w) - \sum_{(w,s) \in E} f(w, s) \times cost(w, s)$. A Minimum Cost Maximum Flow of a network $G = (V, E)$ is a maximum flow with the smallest possible cost.

To relate the optimization problem of user-item matching to the MCMF problem, the most important work is to construct a flow network $G = (V, E)$ to model the optimization objective and the constraints of the user-item matching problem, which is defined as follows. The node set is

$$V = \{s, t\} \cup \{u_i\}_{1 \leq i \leq m} \cup \{v_j\}_{1 \leq j \leq n}$$

where u_i and v_j represent users and items of the recommender system, consistent with the aforementioned notations.

The directed edge set is

$$E = \{(s, u_i)\}_{1 \leq i \leq m} \cup \{(v_j, t)\}_{1 \leq j \leq n}$$
$$\cup \{(u_i, v_j)\}_{\substack{a_{i,j}=0, \\ 1 \leq i \leq m, 1 \leq j \leq n}}$$

Algorithm 1 Largest-Normalized-Score-First

Input:
> user-item adjacent matrix A, normalized score matrix S^N, Vector L of users' initial numbers of recommendation vacancy, vector Q of items' initial stock volume.

Output:
> recommendation matrix R such that $R \times \mathbb{1} = L$ and $R^T \times \mathbb{1} \leq Q$.

1: Initialize R as zero matrix.
2: Sort the $m \times n$ elements of row-normalized score matrix S^N in descending order, and assume that the k-th largest element in this order is originally in row i_k, column j_k of matrix S^N.
3: **for** $k = 1$ to $m \times n$ **do**
4:    **if** $L = \bar{0}$ **then** /*the recommendation lists of all users are fully occupied*/
5:       break;
6:    **end if**
7:    **if** $L(i_k) > 0$ and $Q(j_k) > 0$ **then** /*there is available vacancy in the recommendation list of the user and there is remaining stock volume for the item*/
8:       $R(i_k, j_k) = 1$; /*fill the item in the list of the user*/
9:       $L(i_k) = L(i_k) - 1$; /*decrease the remaining vacancy number of the user*/
10:      $Q(j_k) = Q(j_k) - 1$; /*decrease the remaining stock volume of the item*/
11:    **end if**
12: **end for**
13: **return** R;

In other words, there is a directed edge from the source node to each user node, a directed edge from each item node to the sink node, and a directed edge between every user-item pair without interaction in the past.

The capacity on each directed edge is defined as

$$cap(s, u_i) = l, \quad cap(u_i, v_j) = 1,$$
$$cap(v_j, t) = q_j, \quad 1 \leq i \leq m, \ 1 \leq j \leq n.$$

The edge capacity between every user-item pair is set to be 1, indicating the user-item matching rule that each item can occupy at most one vacancy of the recommendation list of a specific user. The edge capacity from the source node to each user node is defined by the length of recommendation lists, and the edge capacity from each item node to the sink node is defined by the stock volume of the item, corresponding to the two constraints of the optimization function.

The cost on each directed edge is defined as

$$cost(u_i, v_j) = 100 * (1 - \lceil s_{i,j}^N \rceil),$$
$$cost(s, u_i) = cost(v_j, t) = 0, 1 \leq i \leq m, 1 \leq j \leq n,$$

where the normalized score $s_{i,j}^N$ is rounded up to two decimals, such that the values of capacity and cost of this network are all positive integers.

After construction of the directed flow graph, we relate the original optimization problem to the MCMF problem. One benefit of making the connection between the user-item matching problem and the MCMF problem is that it provides an approach for taking advantage of existing works that have already been done on finding the optimal solution. The MCMF problem has been thoroughly studied and many efficient MCMF algorithms are available in the literature [28], [29].

The value of the maximum flow of this network is clearly $m * l$, since we get the initial stock volumes of items by rounding up the values. The objective of minimizing the cost of the maximum flow is essentially the optimization objective of maximizing the sum of normalized scores of matched user-item pairs.

To compute a minimum cost maximum flow in graph $G = (V, E)$ from s to t, we employ the Capacity Scaling algorithm and the MCMF problem can be solved in polynomial time [28]. The solution of the MCMF problem yields the result of the optimal user-item matching problem. The recommendation list of a target user is consisted of items with nonzero flow on the corresponding user-item edges in the optimal solution of the MCMF problem.

V. PERFORMANCE EVALUATION

A. DATA SETS, EVALUATION MEASURES, AND BASELINE ALGORITHMS

Two benchmark data sets are employed to evaluate the performance of recommendation algorithms, namely, Movielens and Netflix. Both of them are movie rating data set, where users rate their watched movies (rephrased as items in this paper) with an explicit integer scores from 1 to 5. For each data set, we use only the ratings no less than 3 to construct the nonzero elements of adjacent matrix of user-item interactions. Table 2 summarizes the statistical features of the two data sets, where the sparsity is the proportion of nonzero elements to the total number of elements of adjacent matrix. To evaluate the offline performance of different recommendation algorithms, each data set is temporally partitioned into two subsets: the training set containing early 80% of the nonzero elements and the probe set later 20% of the nonzero elements. The training set is treated as known information to make recommendation and the probe set is used to test the accuracy performance of the recommendation results.

The most simple recommendation fairness measure is the *aggregate diversity* [30], [31] (also known as coverage [32]), which is defined by the fraction of items recommended to at least one user to the total number of items. This intuitive measure may not be very robust, since the contribution to it of an item that has been recommended just once is equal to that of other item recommended a thousand times. To solve this problem, Fleder and Hosanagar [5] proposed a better alternative by using the Gini coefficient to measure the balance in the numbers of recommended times of different items,

$$G = 1 - \frac{1}{n-1} \sum_{k=1}^{n} (2k - n - 1) \, p \, (i_k | R)$$

TABLE 2. The basic statistics of two real-world networks used in this paper, including the number of users, item and links, and the sparsity.

data set	#users	#items	#links	sparsity
Movielens	6000	3,600	800,000	3.8%
Netflix	9500	14,000	1,700,000	1.2%

where $p \, (i_k | R)$ is the probability of the k-th least recommended item being drawn from the recommendation lists generated by a recommender system. In order to be in accordance with other metrics for which higher value is better, the complement of the standard definition of Gini coefficient is used in this paper.

Besides the above two item-provider oriented measures, we use *Precision* to evaluate the quality of recommendation results for the purpose of users, which is defined as the fraction of accurately recommended items to the length of recommendation lists [32].

In this paper, we use the P3 algorithm [33] (also known as NBI [34] or ProbS [35] algorithm) as the baseline, and several well-known P3-enhanced algorithms as comparing counterparts, including the RP3, HHP, RAP3, PD, BD and BHC algorithms [7], [8], [35]–[38]. The interested reader is referred to a survey article for comprehensive review [39].

Our proposed user-item matching strategy with constraints are generic post-processing methods which can be used to improve any algorithms. Among many classic recommendation models, the reasons of selecting P3 algorithm as the baseline are as follows. First, P3 is intrinsically a hybrid form of user-based and item-based collaborative filtering with diffusion-based similarity [39]. Second, the P3 algorithm does not require any pre-specified parameter, such as the neighborhood size in the k-nearest neighbor collaborative filtering. Third, the P3 algorithm has a perfect physical interpretation, since it is analogous to a mass diffusion process on the user-item network [35]. Finally, the spreading representation of P3 in sparse networks is computationally more efficient than the traditional matrix-based representation of collaborative filtering methods [39].

B. PERFORMANCE OF GREEDY STRATEGY AND ITS IMPROVEMENT

According to Table 3 and Table 4, the precision value of the greedy strategy is about 90% of that of original P3 algorithm on the Movielens data set, and 77% for Netflix. This percentage of accuracy loss is definitely not acceptable for practical applications. The subsequent question is that, is it possible to regain 100% of accuracy value of the regular recommendation scenario?

Recall that the greedy strategy degrades the priority of large-degree users in the user-item matching process by dividing the original scores with the sum of recommendation scores of all the items of the same target user. Then a natural question arises, is the strength of this priority degradation optimal? To answer this question, we replace the constant exponent 1 of the denominator with an adjustable

TABLE 3. Performance comparison of P3 algorithm in the constrained recommendation scenario and the performance of several P3-enhanced algorithms in regular recommendation scenario on the Movielens data set.

Algorithm	Recommendation Settings	Precision	Aggregate diversity	Exposure fairness
	Regular	0.1949	0.0991	0.0378
	Constraint, original greedy strategy	0.1757	0.6493	**0.2993**
P3	Constraint, greedy strategy, $\theta = 0.9$	0.1989	**0.6693**	**0.3002**
	Constraint, MCMF strategy	0.2115	0.5689	**0.2973**
RP3	Regular, $\lambda = 0.6$	**0.2289**	0.3485	0.0859
RAP3	Regular, $\lambda = 0.6$	0.2082	0.2945	0.074
HHP	Regular, $\lambda = 0.3$	0.2238	0.2529	0.0715
BHC	Regular, $\lambda = 0.8$	0.218	0.3115	0.0914
BD	Regular, $\lambda = 0.7$	**0.2297**	0.3507	0.0932
PD	Regular, $\lambda = -0.7$	**0.2283**	0.332	0.0828

TABLE 4. Performance comparison of P3 algorithm in the constrained recommendation scenario and the performance of several P3-enhanced algorithms in regular recommendation scenario on the Netflix data set.

Algorithm	Recommendation Settings	Precision	Aggregate diversity	Exposure fairness
	Regular	0.136	0.0493	0.0068
	Constraint, original greedy strategy	0.1042	0.5278	**0.1204**
P3	Constraint, greedy strategy, $\theta = 0.9$	0.1418	0.5566	**0.124**
	Constraint, MCMF strategy	0.1473	0.5956	**0.1232**
RP3	Regular, $\lambda = 0.5$	0.1533	0.652	0.0414
RAP3	Regular, $\lambda = 0.2$	0.1542	**0.7699**	0.0676
HHP	Regular, $\lambda = 0.2$	**0.156**	0.3311	0.0217
BHC	Regular, $\lambda = 0.8$	0.1512	0.2673	0.017
BD	Regular, $\lambda = 0.7$	**0.1596**	0.5965	0.0478
PD	Regular, $\lambda = -0.7$	0.1494	0.6674	0.0351

parameter θ, that is to say, the original recommendation scores are divided by the sum of recommendation scores of the target user with exponent θ on the shoulder,

$$s_{ij}^{\theta} = \frac{s_{ij}}{(\sum_{k=1}^{n} s_{ik})^{\theta}}.$$

By traversing this parameter θ from 0 to 1 to obtain the best recommendation accuracy, we enhance the original greedy strategy to a parameterized version. For a specific value of parameter θ, the user-item matching process is similar to the original greedy strategy.

According to Table 3 and Table 4, the best precision value of the parameterized greedy strategy is larger than that of original P3 algorithm. What is more, the aggregate diversity is almost 7 times of the original value on the Movielens data set, and more than 11 times for Netflix; the exposure fairness is almost 8 times of the original value on the Movielens data set, and more than 18 times for Netflix.

C. PERFORMANCE OF THE MCMF STRATEGY

Table 3 and Table 4 also present detailed performance comparison between the P3 algorithm with the MCMF strategy, and six state-of-the-art P3-enhanced algorithms in the regular recommendation scenario on the Movielens and Netflix data sets, where the values of their intrinsic parameters are set to be associated with the best recommendation precision. For the P3 algorithm in the constrained recommendation scenario, the MCMF model achieves the best accuracy performance among all the user-item matching strategies (even better

than the parameterized greedy strategy), thus we call it the *P3-MCMF algorithm* as a whole for later discussion.

Compared with six state-of-the-art P3-enhanced algorithms in the regular recommendation scenario, the P3-MCMF algorithm achieves the recommendation precision of more than 90% of the best value of all the enhanced algorithms; its exposure fairness value is more than three times (two times, respectively) of the best values of all the enhanced algorithms on the Movielens (Netflix, respectively) data set; As for the aggregate diversity, the P3-MCMF algorithm achieves much better performance on Movielens and a little worse performance on Netflix, compared with the best values of all enhanced algorithms. Since the aggregate diversity is a very coarse measure of recommendation fairness, its trivial loss does not matter for practical applications.

Recall that these six counterpart algorithms are already enhanced versions of the P3 algorithm, with much better recommendation performance. Take the RP3 algorithm as an example. Compared with the P3 algorithm on Movielens, the precision is improved about 17%, the aggregate diversity is improved about 250%, and the exposure fairness is improved more than 100%. Although the six P3-enhanced algorithms are inspired by different motivations, they have similar recommendation performance in regular recommendation scenario (for example the precision is about 0.22 and the exposure fairness is below 0.1 on the Movielens data set), which indicates that further performance improvement will be a very difficult task. However, our MCMF strategy successfully cracks this problem.
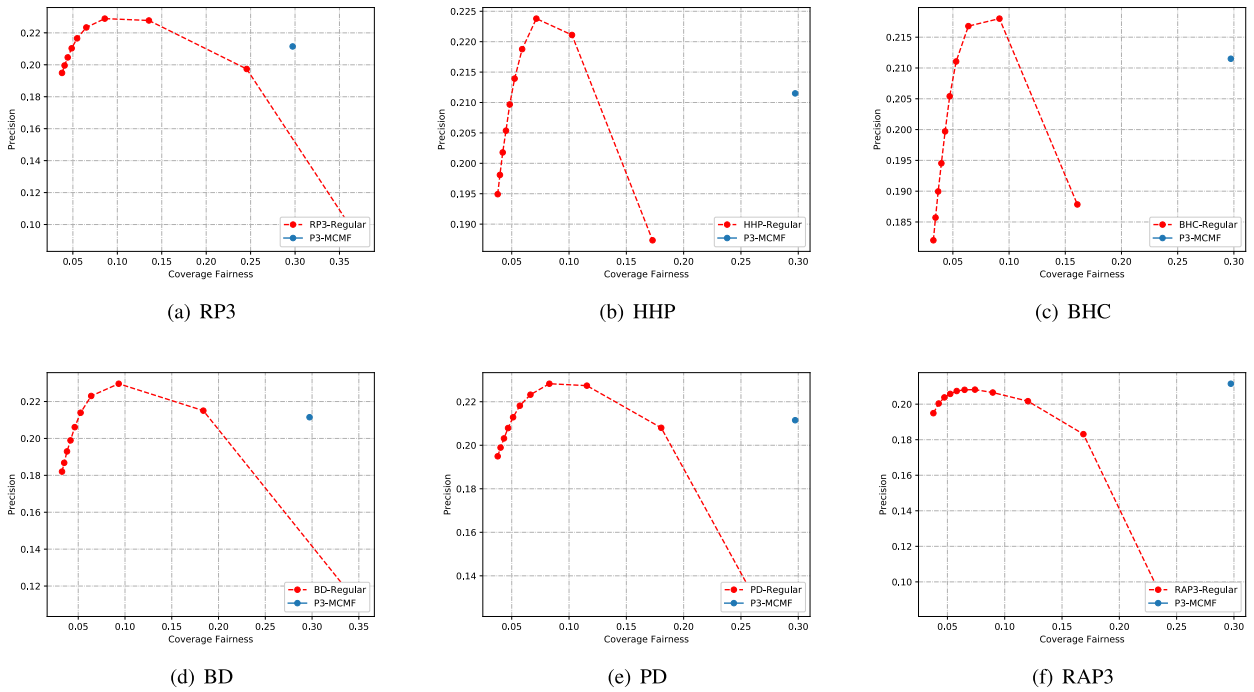
FIGURE 1. The precision against the exposure fairness of six typical P3-enhanced algorithms (the red dots) with the change of their intrinsic parameter λ, compared with the counterparts of our P3-MCMF algorithm (the blue dot) on the Movielens data set. The length of recommendation lists is 20.
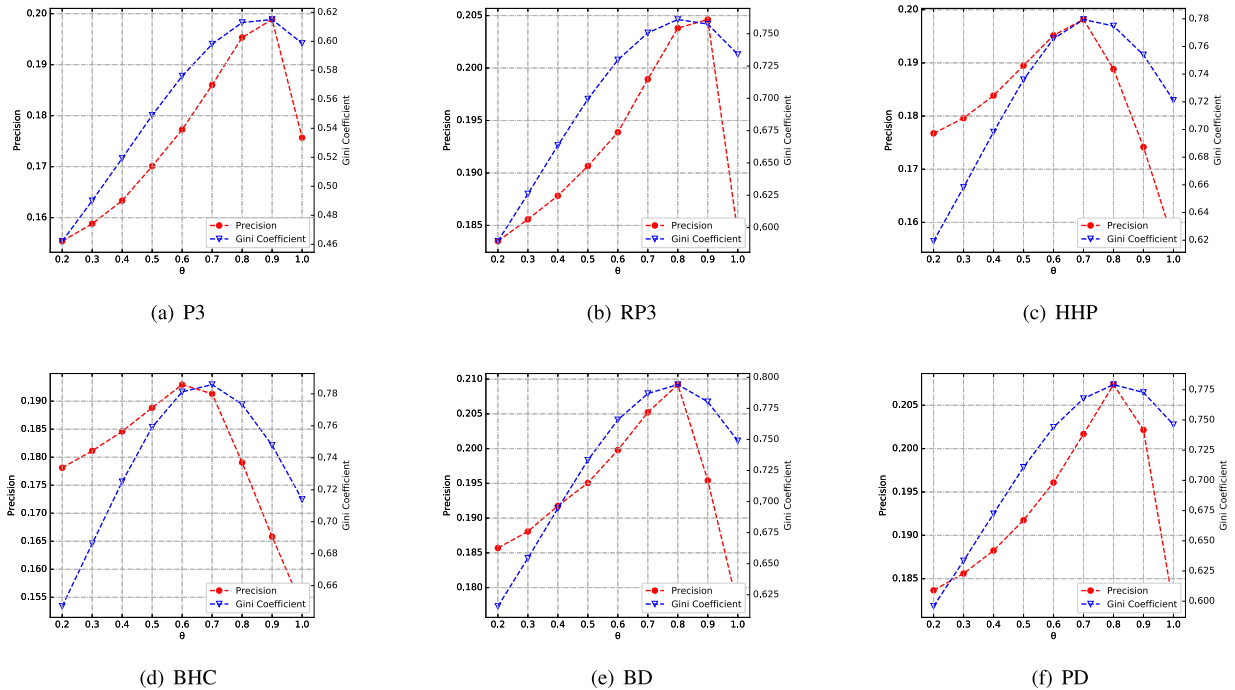

FIGURE 2. The changes of the precision value and the Gini coefficient of recommendation scores in the recommendation lists of length 20, with the exponential parameter of the parameterized greedy strategy for P3 and five P3-enhanced algorithms on Movielens.

The most important significance of our P3-MCMF algorithm is that, it is parameter-free and thus achieves this superior performance without the time cost of parameter optimization, while all the above existing enhanced algorithms

have to traverse their intrinsic parameter to get the best performance.

After the above performance comparison, a natural question is raised, can we adjust the parameter of some typical

P3-enhanced algorithms, to get the similar exposure fairness value while sacrificing the accuracy to the same level? To answer this question, we plot the precision against the exposure fairness of six typical P3-enhanced algorithms, compared with the counterparts of our P3-MCMF algorithm on the Movielens data set in Fig. 1. Each red dot is corresponding to a specific value of the intrinsic parameter λ of the enhanced algorithm, from 0.1 to 1.0 with step length of 0.1, and the blue dot is for the performance of the P3-MCMF algorithm.

From Fig. 1 we can see that, the P3-MCMF algorithm regains more than 90% of the best precision values of typical P3-enhanced algorithms. Even if we let the precision of P3-improved algorithms aligned with that of the P3-MCMF algorithm, the exposure fairness value of the former is still worse than that of the latter, and vice versa. That is to say, the performance of the parameter-free P3-MCMF algorithm is always better than those of the parameterized P3-enhanced algorithms. What is more, the parameter-free characteristic of the P3-MCMF algorithm is a tremendous advantage in the practical applications.

D. ANALYSIS OF TIME COMPLEXITY

The greedy user-item matching strategy consists of two steps: the sorting step of mn elements of normalized matrix S^N whose time complexity is $O(mn \log(mn))$, and the one by one checking step of mn user-item pairs whose time complexity is $O(mn)$, thus the overall time complexity of greedy strategy is $O(mn \log(mn))$. According to [28], the time complexity of our MCMF strategy is $O(m^2 n^2 \log m \log(mn))$.

In summary, each of our two user-item matching strategies has its own advantage and disadvantage. The MCMF strategy is parameter-free, has the best accuracy performance, but its time complexity is higher than its greedy counterpart; The greedy strategy is more time efficient, but it has additional parameter-traversal cost and its accuracy performance has a gap to the best value.

VI. CONCLUDING REMARKS AND FUTURE WORK

The popularity bias is a ubiquitous problem confronted by common recommenders, and many research efforts were devoted to mitigate this problem and thus improve user experience. In fact, popularity bias is not welcome to not only users but also item-providers. While this problem of recommending a few popular items to a majority of users is usually regarded as a user-oriented problem, item-providers also suffer a lot from it and anchor their hope on recommender systems to give fairer exposure chance to different items, especially unpopular ones.

This work was devoted to solve the problem of recommendation fairness, which is measured by the Gini coefficient of numbers of recommendation times of all items in the system. The approach is to limit the allowed recommendation frequency of each item to be proportional to its degree. Although this approach is very effective and robust in significantly improving the recommendation fairness and thus better serves the purpose of item-providers, the following

recommendation accuracy loss and the decrease of user experience cannot be ignored. To solve this subsequent problem, we proposed a heuristic strategy and an elaborate MCMF model to solve the user-item matching problem with constraints, both of which regain more than 100% of the precision value of the baseline algorithm in regular recommendation context. Even compared with several state-of-the-art enhanced algorithms, the precision value of our P3-MCMF algorithm has no significant difference. Another important advantage of our proposed P3-MCMF algorithm is that it is parameter-free and thus achieves this superior performance without the time cost of parameter optimization, while most existing enhanced algorithms have to traverse their intrinsic parameter to get the best performance.

In the greedy user-item matching strategy, by reducing the positive correlation between recommendation scores and user degrees via score normalization process, the recommendation accuracy and the distribution uniformity of recommendation scores are both improved. Then a natural question is raised, what is the relationship between these two measures? To answer this question, Fig. 2(a) plots the changes of the precision value and the Gini coefficient of recommendation scores in the recommendation lists of length $l = 20$, with the exponential parameter of the parameterized greedy strategy for the P3 algorithm on Movielens. We can see a very interesting phenomenon that, the two measures achieve their own optimal values at (almost) the same value of the exponential parameter. To see whether this is a common phenomenon in recommender systems or not, we regard the above-mentioned five P3-enhanced algorithms with optimal values of intrinsic parameters as baseline algorithms, and present the same plots in Fig. 2(b)-(f). This consistence between the peaks of the two measures still holds. Our next work is to explore the mechanism behind the consistence of these two peak values, and try to make use of it to further improve existing recommendation algorithms or design new recommendation models.
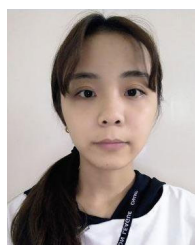
REFERENCES

[1] P. Adamopoulos and A. Tuzhilin, "On over-specialization and concentration bias of recommendations: Probabilistic neighborhood selection in collaborative filtering systems," in *Proc. 8th ACM Conf. Recommender Syst. (RecSys)*, 2014, pp. 153–160.

[2] H. Abdollahpouri, G. Adomavicius, R. Burke, I. Guy, D. Jannach, T. Kamishima, J. Krasnodebski, and L. Pizzato, "Multistakeholder recommendation: Survey and research directions," *User Model. User-Adapted Interact.*, vol. 30, no. 1, pp. 127–158, Mar. 2020.

[3] J. Chen, H. Dong, X. Wang, F. Feng, M. Wang, and X. He, "Bias and debias in recommender system: A survey and future directions," 2020, *arXiv:2010.03240*. [Online]. Available: http://arxiv.org/abs/2010.03240

[4] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," in *Proc. 14th Int. Conf. World Wide Web (WWW)*, 2005, pp. 22–32.

[5] D. Fleder and K. Hosanagar, "Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity," *Manage. Sci.*, vol. 55, no. 5, pp. 697–712, Mar. 2009.

[6] H. Abdollahpouri, "Popularity bias in ranking and recommendation," in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, Jan. 2019, pp. 529–530.

[7] F. Christoffel, B. Paudel, C. Newell, and A. Bernstein, "Blockbusters and wallflowers: Accurate, diverse, and scalable recommendations with random walks," in *Proc. 9th ACM Conf. Recommender Syst.*, Sep. 2015, pp. 163–170.

[8] Q. Dong, Q. Yuan, and Y.-B. Shi, "Alleviating the recommendation bias via rank aggregation," *Phys. A, Stat. Mech. Appl.*, vol. 534, Nov. 2019, Art. no. 122073.

[9] M. Mansoury, H. Abdollahpouri, M. Pechenizkiy, B. Mobasher, and R. Burke, "FairMatch: A graph-based approach for improving aggregate diversity in recommender systems," in *Proc. 28th ACM Conf. User Model., Adaptation Personalization*, Jul. 2020, pp. 154–162.

[10] G. K. Patro, A. Chakraborty, N. Ganguly, and K. Gummadi, "Incremental fairness in two-sided market platforms: On smoothly updating recommendations," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 181–188.

[11] Y. Ge, S. Liu, R. Gao, Y. Xian, Y. Li, X. Zhao, C. Pei, F. Sun, J. Ge, W. Ou, and Y. Zhang, "Towards long-term fairness in recommendation," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Mar. 2021, pp. 445–453.

[12] X. Ren, L. Lü, R. Liu, and J. Zhang, "Avoiding congestion in recommender systems," *New J. Phys.*, vol. 16, no. 6, Jun. 2014, Art. no. 063057.

[13] T. Silveira, M. Zhang, X. Lin, Y. Liu, and S. Ma, "How good your recommender system is? A survey on evaluations in recommendation," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 5, pp. 813–831, May 2019.

[14] L. Boratto, G. Fenu, and M. Marras, "Connecting user and item perspectives in popularity debiasing for collaborative recommendation," *Inf. Process. Manage.*, vol. 58, no. 1, Jan. 2021, Art. no. 102387.

[15] Y.-J. Park and A. Tuzhilin, "The long tail of recommender systems and how to leverage it," in *Proc. ACM Conf. Recommender Syst. (RecSys)*, 2008, pp. 11–18.

[16] D. Jannach, L. Lerche, I. Kamehkhosh, and M. Jugovac, "What recommenders recommend: An analysis of recommendation biases and possible countermeasures," *User Model. User-Adapted Interact.*, vol. 25, no. 5, pp. 427–491, Dec. 2015.

[17] J. Chen, H. Dong, Y. Qiu, X. He, X. Xin, L. Chen, G. Lin, and K. Yang, "AutoDebias: Learning to debias for recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2021, pp. 21–30.

[18] H. Abdollahpouri, R. Burke, and B. Mobasher, "Controlling popularity bias in learning-to-rank recommendation," in *Proc. 11th ACM Conf. Recommender Syst.*, Aug. 2017, pp. 42–46.

[19] L. Hou, X. Pan, and K. Liu, "Balancing the popularity bias of object similarities for personalised recommendation," *Eur. Phys. J. B*, vol. 91, no. 3, p. 47, Mar. 2018.

[20] C. E. Berbague, N. E.-I. Karabadji, H. Seridi, P. Symeonidis, Y. Manolopoulos, and W. Dhifli, "An overlapping clustering approach for precision, diversity and novelty-aware recommendations," *Expert Syst. Appl.*, vol. 177, Sep. 2021, Art. no. 114917.

[21] H. Abdollahpouri, R. Burke, and B. Mobasher, "Popularity-aware item weighting for long-tail recommendation," 2018, *arXiv:1802.05382*. [Online]. Available: http://arxiv.org/abs/1802.05382

[22] E. Yalcin and A. Bilge, "Investigating and counteracting popularity bias in group recommendations," *Inf. Process. Manage.*, vol. 58, no. 5, Sep. 2021, Art. no. 102608.

[23] H. Abdollahpouri, R. Burke, and B. Mobasher, "Managing popularity bias in recommender systems with personalized re-ranking," 2019, *arXiv:1901.07555*. [Online]. Available: http://arxiv.org/abs/1901.07555

[24] H. Wu, X. Cui, J. He, B. Li, and Y. Pei, "On improving aggregate recommendation diversity and novelty in folksonomy-based social systems," *Pers. Ubiquitous Comput.*, vol. 18, no. 8, pp. 1855–1869, Dec. 2014.

[25] Y. Wu, J. Cao, and G. Xu, "FAST: A fairness assured service recommendation strategy considering service capacity constraint," in *Proc. Int. Conf. Service-Oriented Comput.*, 2020, pp. 287–303.

[26] T. Qiu, C. Wan, X.-F. Wang, and Z.-K. Zhang, "User interest dynamics on personalized recommendation," *Phys. A, Stat. Mech. Appl.*, vol. 525, pp. 965–977, Jul. 2019.

[27] Y. Wu, B. Kuang, T. Wang, Q. Zhang, and M. Wang, "Minimum cost maximum flow algorithm for upstream bandwidth allocation in OFDMA passive optical networks," *Opt. Commun.*, vol. 356, pp. 103–108, Dec. 2015.

[28] K. R. Ahuja, L. T. Magnanti, and B. J. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.

[29] J. Kleinberg and E. Tardos, *Algorithm Design*. London, U.K.: Pearson, 2005.

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

[31] S. Vargas, "Novelty and diversity enhancement and evaluation in recommender systems and information retrieval," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2014, p. 1281.

[32] L. Xiang, *Practice of Recommender Systems* (in Chinese). Beijing, China: Posts and Telecom, 2012.

[33] C. Cooper, S. H. Lee, T. Radzik, and Y. Siantos, "Random walks in recommender systems: Exact computation and simulations," in *Proc. 23rd Int. Conf. World Wide Web*, Apr. 2014, pp. 811–816.

[34] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, "Bipartite network projection and personal recommendation," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 76, no. 4, Oct. 2007, Art. no. 046115.

[35] T. Zhou, Z. Kuscsik, J.-G. Liu, M. Medo, J. R. Wakeling, and Y.-C. Zhang, "Solving the apparent diversity-accuracy dilemma of recommender systems," *Proc. Nat. Acad. Sci. USA*, vol. 107, no. 10, pp. 4511–4515, 2010.

[36] L. Lü and W. Liu, "Information filtering via preferential diffusion," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 83, no. 6, Jun. 2011, Art. no. 066119.

[37] D.-C. Nie, Y.-H. An, Q. Dong, Y. Fu, and T. Zhou, "Information filtering via balanced diffusion on bipartite networks," *Phys. A, Stat. Mech. Appl.*, vol. 421, pp. 44–53, Mar. 2015.

[38] J.-G. Liu, T. Zhou, and Q. Guo, "Information filtering via biased heat conduction," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 84, no. 3, Sep. 2011, Art. no. 037101.

[39] F. Yu, A. Zeng, S. Gillard, and M. Medo, "Network-based recommendation algorithms: A review," *Phys. A, Stat. Mech. Appl.*, vol. 452, pp. 192–208, Jun. 2016.

QIANG DONG received the B.S. and Ph.D. degrees in computer science and technology from Chongqing University, in 2006 and 2010, respectively. From 2008 to 2009, he visited the Department of Computer Science, Montclair State University, NJ, USA. He is currently an Associate Professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. He has published more than 20 academic articles in peer-reviewed journals. His research interests include combinatorial networks, complex networks, and recommender systems.

SHUANG-SHUANG XIE received the B.S. degree in network engineering from Sichuan Normal University, in 2018, and the M.S. degree in computer science and technology from the University of Electronic Science and Technology of China, in 2021. She is currently a Research and Development Engineer with ByteDance Ltd. Her research interests include complex networks and recommender systems.

WEN-JUN LI received the B.S. and Ph.D. degrees in computer science and technology from the University of Electronic Science and Technology of China, in 2010 and 2017, respectively. He is currently a professional Teacher with the School of Software and Services Outsourcing, Suzhou Vocational Institute of Industrial Technology, China. His research interests include recommender systems and cloud computing.

● ● ●