ChinaRxiv

AI translation · View original & related papers at
chinarxiv.org/items/chinaxiv-202412.00065

FairSort: Learning to Fair Rank for Personalize-dRecommendations in Two-Sided Platforms

Authors: Guoli Wu

Date: 2024-12-03T00:00:00+00:00

Abstract

Traditional recommendation systems focus on maximizing user satisfaction by suggesting their favorite items. This user-centric approach may lead to unfair exposure distribution among the providers. On the contrary, a provider-centric design might become unfair to the users. Therefore, this paper proposes a re-ranking model FairSort to find a trade-off solution among user-side fairness, provider-side fairness, and personalized recommendations utility. Previous works habitually treat this issue as a knapsack problem, incorporating two-sided fairness as constraints.

In this paper, we adopt a novel perspective, treating each recommendation list as a runway rather than a knapsack. In this perspective, each item on the runway gains a velocity and runs within a specific time, achieving re-ranking for two-sided fairness. Meanwhile, we ensure the Minimum Utility Guarantee for personalized recommendations by designing a Binary Search approach. This can provide more reliable recommendations compared to the conventional greedy strategy based on the knapsack problem. We further broaden the applicability of FairSort, designing two versions for online and offline recommendation scenarios. Theoretical analysis and extensive experiments on real-world datasets indicate that FairSort can ensure more reliable personalized recommendations while considering fairness for both the provider and user.

Full Text

Preamble

JOURNAL OF LATEX CLASS FILES, VOL. 14, NO. 8, AUGUST 2023

FairSort: Learning to Fair Rank for Personalized Recommendations in Two-Sided Platforms

Guoli Wu[1], Zhiyong Feng[1], Shizhan Chen[1], Hongyue Wu[1],*, Xiao Xue[1], Jianmao Xiao[2], Guodong Fan[1], Hongqi Chen[1], Jingyu Li[1]

[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]School of Software, Jiangxi Normal University, Nanchang, China, jm_{xiao}@jxnu.edu.cn
{wuguoli_{it999}, zyfeng, shizhan, hongyue.wu, jzxuexiao, guodongfan, hongqi, lijingyu_{working}}@tju.edu.cn

Abstract—Traditional recommendation systems focus on maximizing user satisfaction by suggesting their favorite items. This user-centric approach may lead to unfair exposure distribution among providers. Conversely, a provider-centric design might become unfair to users. Therefore, this paper proposes a re-ranking model called FairSort[1] to find a trade-off solution among user-side fairness, provider-side fairness, and personalized recommendation utility. Previous works habitually treat this issue as a knapsack problem, incorporating both-side fairness as constraints.

In this paper, we adopt a novel perspective, treating each recommendation list as a runway rather than a knapsack. In this view, each item on the runway gains a velocity and runs within a specific time, achieving re-ranking for both-side fairness. Meanwhile, we ensure the Minimum Utility Guarantee for personalized recommendations by designing a binary search approach. This can provide more reliable recommendations compared to the conventional greedy strategy based on the knapsack problem perspective. We further broaden the applicability of FairSort by designing two versions for online and offline recommendation scenarios. Theoretical analysis and extensive experiments on real-world datasets indicate that FairSort can ensure more reliable personalized recommendations while considering fairness for both providers and users.

Index Terms—Multisided fairness, fairness-aware recommendation, learning to rank, exposure fairness, np problem

I. INTRODUCTION

Although recommender systems bring great convenience to users by mining their interests to provide personalized recommendations, they also introduce certain fairness problems. This is because recommender systems typically have inherent biases, such as data and algorithmic bias [1]–[5], resulting in unfairness in both the recommendation process and its outcomes. During the process, models may embed biased information (e.g., sensitive attributes like race and gender), resulting in discriminatory practices like offering more technical job opportunities to men over women [6], [7]. Regarding outcomes, models may generate biased recommendations for different user groups, such as the unfair allocation of exposure opportunities among various providers [8], [9].

While many methods aim to ensure fairness in recommendation systems, most address user or provider fairness separately [10]–[13], with few considering both

sides simultaneously. An undeniable fact is that enhancing fairness on one side often harms the other side. In this paper, we study the two-sided fairness problem in recommender systems, with specific scenarios illustrated as follows. If we adopt the traditional Top-K recommendation method as shown in (Fig1 a), it is often seen as maximizing user satisfaction. However, Patro et al. [14] reveal that this user-centric approach can lead to unfair exposure distribution among providers, meaning that a few providers monopolize most of the platform's exposure while others receive very little. Providers naturally seek greater exposure as it often results in more orders and revenue. Conversely, inadequate exposure may drive them to leave the platform, which could undermine the recommender system's sustainability. However, efforts to fairly distribute exposure among providers can negatively impact recommendation quality for users, with the degree of impact varying among different users [14]. This is because re-ranking the Top-K list to improve fairness inevitably sacrifices some user satisfaction, and this impact is often difficult to control. This inconsistency in recommendation quality among users further exacerbates user-side unfairness. Therefore, an urgent need exists for a re-ranking algorithm that ensures fairness for both providers and users while maintaining high-quality personalized recommendations, as shown in (Fig1 b).

It is challenging to find a trade-off solution among three objectives. The essence of this problem lies in the combinatorially exploded solution space (all possible combinations of picking K items to reform the user's recommendation list), to re-select a non-direct Top-K recommendation list for each user that can satisfy both-side fairness while maintaining high recommendation utility. The main challenges are twofold. First, the solution space is enormous (combinatorially exploded), making it difficult to efficiently find a resolution that satisfies both-side fairness. Second, there are inherent conflicts between user-side fairness, provider-side fairness, and personalized recommendation utility, meaning that pursuing one aspect may compromise the others.

Currently, only a few studies consider both-side fairness, with TFROM [15], FairRec [14], and CPFair [16] being representative works. Previous research habitually analogizes this problem to a knapsack problem, which is known to be NP-complete [17]. In this analogy, the length of the recommendation list is seen as the knapsack's capacity, the items to be recommended as the knapsack's contents, and both-side fairness as the constraints. Building on this perspective, these studies employ a greedy strategy to select high-score items for the knapsack to maximize recommendation quality. However, the reliability of relying on a greedy algorithm to ensure recommendation quality is questionable. We empirically observe that the greedy strategy results in an uneven distribution of recommendation quality among users, with some recommendation lists experiencing significant quality degradation due to the introduction of both-side fairness. This inconsistency further exacerbates user-side unfairness.

In this paper, we offer a novel perspective, as illustrated in (Fig1 b), where each recommendation list is viewed as a runway rather than adhering to the

traditional knapsack. Specifically, each item on the runway gains a velocity (either upward ↑ or downward ↓) and runs within a specific time, achieving a re-ranked Top-K list for both user and provider fairness. In this perspective, we ensure the Minimum Utility Guarantee for each re-ranked list in terms of personalized recommendation utility by designing a Binary Search approach based on a discovered theorem. The Minimum Utility Guarantee strategy can provide more reliable personalized recommendations compared to the conventional greedy strategy under the knapsack problem perspective, as it effectively tackles the problem of unpredictable loss in personalized recommendation utility when considering both-sided fairness.

In summary, our re-ranking model FairSort is designed to guarantee the Minimum Utility of recommendation quality while simultaneously ensuring fairness for both sides. The major contributions of this paper are as follows:

- **Motivating multi-stakeholder fairness in RS**: We propose a novel perspective to address multi-stakeholder fairness in recommendation systems. We view each recommendation list as a runway rather than adhering to the traditional knapsack problem.
- **Minimum Utility Guarantee**: We propose the Minimum Utility Guarantee for personalized recommendations. It effectively tackles the problem of uncontrollable potential loss in personalized recommendation utility when the algorithm considers both-sides fairness. We prove and guarantee this theoretically and experimentally (§VII).
- **Experiment**: In addition to the theoretical guarantees (§VII), extensive experimentation and evaluation over three real-world datasets deliver strong evidence of the effectiveness of our proposed FairSort. It can ensure more reliable personalized recommendations while considering both-side fairness (§V), which is implemented in two versions for online and offline scenarios.

The paper is structured as follows: Section II covers related work, and Section III formalizes the two-sided fairness problem. Section IV presents the FairSort model. The experiment results are detailed in Section V. We conclude the paper in Section VI. Finally, theoretical support is given in Section VII.

II. RELATED WORK

We survey related works in three directions: (i) Single-side Fairness vs Multi-side Fairness, (ii) A Taxonomy of Fairness Notions in Recommendation, and (iii) Minimum Utility Guarantee.

Single-side Fairness vs Multi-side Fairness: Currently, most research focuses on guaranteeing single-side fairness such as item-side fairness [18]–[20]. Some studies indicate that traditional methods of estimating item utility and exposure exhibit biases, leading to unfair distribution of item-side exposure [21], [22]. However, recommender systems need to address the challenge of balancing fairness requirements across multiple stakeholders. Only a few studies have ad-

dressed multi-side fairness [14]–[16], [23]–[25]. FairRec [14] guarantees minimum exposure for each item from the provider side and achieves EF-1 [26] fairness from the user side. TFROM [15] on the user side guarantees consistent user satisfaction of the recommendation list, and on the provider side, ensures the fair allocation of exposure [27]. CPFair [16] uses a mixed-integer linear programming approach that seamlessly integrates fairness constraints from both sides in a joint objective framework. It divides users and items into two groups (active and inactive), respectively, aiming to achieve group fairness. These approaches are tailored to deterministic re-ranking problems and rely on greedy algorithms. In contrast, TSFD [28] is also dedicated to addressing both-sided fairness, but specifically targeting scenarios involving stochastic ranking, employing a combination of greedy strategies and convex optimization. In this paper, we neither adopt a greedy strategy nor utilize optimization techniques. We offer another perspective, viewing each recommendation list as a runway, instead of the traditional approach that regards it as a knapsack.

A Taxonomy of Fairness Notions in Recommendation: Researchers classify the fairness work of recommendation algorithms into pre-processing [29], mid-processing [30], and post-processing [31], [32] according to the timing of solving the fairness problem. Pre-processing refers to correcting the data bias that leads to the unfairness problem. Mid-processing refers to preventing the model from learning biased information. Post-processing is agnostic to the model, intervenes fairly in the output results of the model, and adjusts the results to make them fair. The approach taken in this paper is a post-processing mechanism. It is independent of classical recommendation algorithm models and can be applied to various recommendation algorithms. Related research classifies recommendation algorithms' fairness into group fairness and individual fairness [33], [34]. Group fairness requires that dominant and inferior groups be treated fairly in some aspect, while individual fairness requires similar individuals to have similar treatment. This paper considers both: provider-side fairness focuses on the fairness of the groups of items provided by each provider, while from the user side, we focus on fairness between individual users.

Minimum Utility Guarantee: It is crucial in personalized recommendation systems to ensure users receive a certain level of utility. Some researchers have proposed minimum wage guarantee as a fairness standard [35]–[37]; research [38], [39] showed evidence of how minimum wage guarantee decreases income inequality. Inspired by these works, we propose the notion of a minimum utility guarantee for user-side fairness.

III. PRELIMINARIES

We address the following questions to gradually analyze the two-sided fairness problem: (i) Quantifying exposure, (ii) Quantifying recommendation quality, (iii) Defining user-side fairness, (iv) Defining provider-side fairness, and (v) Exploring the relationship among user-side fairness, provider-side fairness, and personalized recommendation utility. Finally, we present a formal problem de-

scription.

A. Notations

- $U = \{u_1, u_2, \dots, u_m\}$ is a set of users.
- $I = \{i_1, i_2, \dots, i_n\}$ is a set of recommended items.
- $P = \{p_1, p_2, \dots, p_l\}$ is a set of providers supplying items.
- I_p is the set of items provided by provider p.
- $V = [v_{u_1,i_1}, v_{u_1,i_2}, \dots, v_{u_m,i_n}]$ is a relevance rating matrix produced by the recommendation algorithm.
- $L^{ori} = \{l^{ori}_{u_1}, \dots, l^{ori}_{u_m}\}$ is a set of original sorted lists, including all items in l^{ori}_u, ranked based on V.
- $L = \{l_{u_1}, l_{u_2}, \dots, l_{u_m}\}$ is a set of recommendation lists finally outputted to users, including K items in each list l.

B. Quantify Exposure Resources

The position of each item in the recommendation list is distributed from high to low, and related research shows that the higher the position of the item, the more attention it receives from the user and the more exposure it obtains. In particular, an item in position 5 is largely ignored [40]. Therefore, we consider position bias to quantify exposure. We use a decay function, and the exposure of an item i can be defined as:

$$e_i = \sum_{u \in U} \frac{\mathbf{1}_{l_u}(i)}{\log_2(r_{u,i} + 1)}$$

Where $\mathbf{1}_{l_u}(i)$ is the indicator function when the item is successfully recommended; that is, the item is in l_u, then the indicator function is 1, otherwise it is 0. The symbol $r_{u,i}$ represents the position of item i in l_u. The total exposure gained by a provider is the sum of the exposures of items it offers as $e_p = \sum_{i \in I_p} e_i$.

C. Quantify The Recommendation Quality

From the rating matrix V, we can pick out K items with the highest preference scores for each user, i.e., the traditional Top-K recommendation. Although this Top-K list may still not be absolutely in line with users' preferences in practice, it can be assumed that it has already reflected their preferences as much as possible. Thus, this Top-K list is the optimal recommendation list. The idea of measuring the recommendation quality of l_u is to compare it with the optimal recommendation list. We take two classical metrics in information retrieval, namely discounted cumulative gain (DCG) and normalized discounted cumulative gain (NDCG), to measure the quality of recommendation [41]. DCG quantifies recommendation quality by summing the value of each item in the recommendation list. This value is gained by multiplying the preference score

of the item by the function value that is logarithmically decayed based on the position of the item. It is also consistent with the decay function used to quantify exposure.

$$DCG_{u,l_u} = \sum_{i=1}^{K} \frac{v_{u,l_u[i]}}{\log_2(i+1)}$$

Where $l_u[i]$ represents the i-th item selected from the recommendation list l_u. Then, the DCG of the optimal recommendation list is the maximum value, which cannot be exceeded by the DCG of the list formed by other permutations. Thus, the quality measure of any recommendation list is normalized to $[0, 1]$.

$$NDCG_u = \frac{DCG_{u,l_u}}{DCG_{u,l^{ori}u}}$$

The larger the NDCG is, the closer the recommendation list is to the optimal recommendation list. In particular, when the NDCG is equal to 1, the recommendation list is the optimal recommendation list. Conversely, the smaller the NDCG is, the larger the recommendation quality loss is.

D. Defining User-Side Fairness

Due to the introduction of provider-side fairness, it is difficult for users to get the optimal recommendation list, which means that recommendation quality will be sacrificed in exchange for fairness on the provider side. We then define the user-side fairness guarantee as follows: The recommendation quality loss should be consistent across users and ensure recommendation quality is not below a certain threshold.

Minimum Utility Guarantee: The recommendation is fair for users if each user receives recommendation results with the same NDCG and which will not fall below the threshold. $(NDCG_{u_1} = NDCG_{u_2}, \forall u_1, u_2 \in U) \wedge (NDCG_{u_i} \geq \text{threshold}, \forall u_i \in U)$

E. Defining Provider-Side Fairness

To ensure provider-side fairness, the definition can be based on two ideas: First, the amount of exposure allocated to the provider should be proportional to their quality [32]. Second, a provider's exposure should be proportional to the number of items it provides.

Definition 2 (Uniform Fairness): The provider exposure distribution satisfies Uniform Fairness if the conversion rate of exposure resources based on the number of items they provide is equal between any two providers, abbreviated as UF.

$$\frac{e_{p_1}}{|I_{p_1}|} = \frac{e_{p_2}}{|I_{p_2}|}, \forall p_1, p_2 \in P$$

Definition 3 (Quality Weighted Fairness): The provider exposure distribution meets Quality Weighted Fairness if the conversion rate of exposure based on their own quality is equal between any two providers, abbreviated as QF.

$$\frac{e_{p_1}}{\sum_{i \in I_{p_1}} \sum_{u \in U} v_{u,i}} = \frac{e_{p_2}}{\sum_{i \in I_{p_2}} \sum_{u \in U} v_{u,i}}, \forall p_1, p_2 \in P$$

F. The Game Between Users and Providers

We first deeply grasp the game relationship between both sides of fairness. In the paper [14], these two goals are often difficult to achieve simultaneously. Suppose the Top-K recommendation is implemented directly for each user. In that case, each user gets the recommendation list with an NDCG value of 1, and it can also meet the Minimum Utility Guarantee of recommendation quality. Thus, the fairness of the user side is completely guaranteed, but the fairness metrics of the provider side may be inferior. However, to improve the provider-side fairness metrics, we must reorder each Top-K result and regenerate a new recommendation list for each user. But this means that each user's recommendation quality will certainly decrease, especially since the distribution of the loss of recommendation quality may be diverse for different users. After grasping their game relationship, we summarize three optimization objectives as our algorithm's guiding idea:

1. **Provider-Side Fairness**: Sacrifice recommendation quality in exchange for fairness on the provider's side.
2. **Personalized Recommendations Utility**: Maintain high recommendation quality.
3. **User-Side Fairness**: The loss of recommendation quality is evenly distributed to all users.

We propose a re-ranking strategy denoted as π, such that $\pi(L^{ori}) = L$ and L satisfies the three objectives above.

IV. A TWO-SIDED FAIRNESS-AWARE LEARNING TO RANK MODEL (FAIRSORT)

In Section III, we model the fairness problem on both the user and provider sides, summarized in figure 2 [Figure 2: see original paper]. Each list in the figure represents l_u^{ori}, where items are sorted in descending order based on user preference scores from V. The Top-K items are highlighted, and the left section shows the distribution of exposure across different providers when using the traditional Top-K method, where some providers receive more exposure while others receive less. Items of the same color represent those from the same

provider: I_p. We consider l_u^{ori} as a runway, where each item is assigned an initial velocity designed to account for provider-side fairness, with directions indicated by the arrows, either upwards or downwards. Once a velocity is assigned, all items run according to a time parameter λ, achieving a fair re-ranking that ensures fairness on both sides. This leads to several key questions: How to assign velocity to each item? And how to determine the running time λ?

In this section, we propose that FairSort solves the two-sided fairness problem. FairSort consists of two algorithms designed for two scenarios. One is an offline scenario in which the system makes recommendations to all users once at the same time, such as via an advertising push. The other is an online scenario where users' requests arrive randomly. The system needs to respond to each request quickly and provide the recommendation results, for example, online purchases.

A. FairSort For Offline Scenario

Total Exposure Resources: In the offline scenario, it is a one-time recommendation service for all users. When the number of users m and the length of the recommendation list K are determined, we can calculate the total exposure as follows:

$$E_{total} = m \times \sum_{rank=1}^{K} \frac{1}{\log_2(rank+1)}$$

Fair Exposure Resources: Further, since we define exposure fairness on the provider side, when the total exposure is determined, we can calculate how much exposure each provider should obtain to reach a fair state:

$$e_{p_j}^{Fair} = E_{total} \times \frac{|I_{p_j}|}{\sum_{p \in P} |I_p|} \quad \text{(Uniform Fairness)}$$

$$e_{p_j}^{Fair} = E_{total} \times \frac{\sum_{i \in I_{p_j}} \sum_{u \in U} v_{u,i}}{\sum_{p \in P} \sum_{i \in I_p} \sum_{u \in U} v_{u,i}} \quad \text{(Quality Weight Fairness)}$$

If all providers receive exposure equal to e^{Fair}, then the provider-side distribution of exposure is absolutely fair. However, the exposure that each recommendation list position can bring is fixed, so absolute fairness often cannot be achieved. But e^{Fair} can be used as our benchmark, and as long as each provider gets exposure infinitely close to e^{Fair}, then relative fairness can be achieved.

Allocate Fairness Lift Velocity: From figure 2, we observe that some providers receive more exposure through the Top-K recommendation, while others receive less. To address this, we should assign a downward velocity to the items of providers with higher exposure, thereby naturally reducing

their overall exposure. Conversely, items from providers with lower exposure should be assigned an upward velocity, allowing their exposure to increase and approach e^{Fair}. Thus, this mechanism can ensure fair exposure distribution on the provider side. We ensure that items from the same provider receive identical velocities, lifting the entire group and preserving the partial order relations within each group.

We have determined the velocity direction, but how should the velocity value be determined? Our core goal is to maintain a fair exposure distribution on the provider side. Further abstracting the essence of the problem is to converge the consistency of conversion rate on the provider side, according to equations (5) and (6). Thus, we calculate the error rate $ErrRate$ for each provider relative to the fair conversion rate, taking the sign of $ErrRate$ as the direction of the velocity, and the value of $ErrRate$ also introduces the information of provider's quality or the total number of items it provides.

To determine a more appropriate λ value, we first introduce a theorem that we discovered, based on which we can cleverly calculate a reasonable λ and achieve fairness for both sides by using a Binary Search strategy.

Theorem 1: $\lambda \in [0, +\infty]$, $\forall u \in U$, the initial $NDCG_u$ of l_u^{ori} is 1 and then decreases monotonically as the value of λ increases. The proof is presented in section VII.

We use this theorem property to design a Binary Search strategy to accelerate the computation of λ so that the NDCG value is highly approximated to the threshold. λ_{target} is the value that can make the NDCG value absolutely equal to the threshold. We summarize the binary search process as follows:

$$\lambda_{target} \in [0, \lambda_{max}], \lambda_{max} > \text{gap}$$

BinarySearch Time: $\lambda_{target} \in [m_1, m_2]$

$$ErrRate_{p_j} = \frac{e_{p_j}^{Fair}}{|I_{p_j}|} - \frac{e_{p_j}}{|I_{p_j}|} \quad \forall p_j \in P \text{ (Uniform Fairness)}$$

$$ErrRate_{p_j} = \frac{e_{p_j}^{Fair}}{\sum_{i \in I_{p_j}} \sum_{u \in U} v_{u,i}} - \frac{e_{p_j}}{\sum_{i \in I_{p_j}} \sum_{u \in U} v_{u,i}} \quad \forall p_j \in P \text{ (Quality Weighted Fairness)}$$

$ErrRate$ will be normalized as follows, then its lift velocity will be determined. J is a certain filter function. The output is the original value if the parameter and the numerator have the same sign. Otherwise, the output is 0, and then take the absolute value to sum up. The calculation formula is as follows:

$$\text{Lift}_{p_j} = \begin{cases} \frac{ErrRate_{p_j}}{\sum_{i=1}^{|P|} |J(ErrRate_{p_i})|} & \text{if } ErrRate_{p_j} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{getFair}(i_k) = \text{Lift}_{i_k} = \text{Lift}_{p_j} \quad (\forall i_k \in I \wedge i_k \in I_{p_j})$$

We calculate the error value between each provider's current conversion rate and the fair conversion rate. Further, we normalize the process and assign the lift velocity for each item. The essence of this process is that the conversion rate of each provider is converging to equal the fair conversion rate.

Learning To Rank: To guarantee the fair allocation of exposure on the provider side, we propose the above Re-Ranking mechanism, a formulation described as follows:

$$\forall u_i \in U : l_{u_i} = \text{argsort}_{i_k \in R_u}(V_{u_i,i_k} + \lambda \cdot \text{getFair}(i_k))$$

As can be seen from the formula, the result of any user's ranking considers two dimensions of information. The first is the user's preference score for the item, based on matrix V. The second dimension is the velocity of the item obtained by equation (13), considering the provider-side fairness. Therefore, with time λ determined, items move up or down in the sorted list in exchange for fairness on both sides. The set R is formed by the hyperparameter ratio, which is used to take out the top-ranked items from l_u^{ori} according to a certain ratio, then items in R participate in the Re-Ranking process.

Binary Search Fairness Weighting Factors: In a physical sense, the larger the fairness weighting factor is, the more importance is attached to fairness on the provider side. But at the same time, the recommendation quality on the user side will be sacrificed more. Thus, λ will undertake the trade-off between the interests of both sides. We need a mechanism to determine a more appropriate λ value.

Algorithm termination: When $(m_2 - m_1) >$ gap, continue searching; when $(m_2 - m_1) \leq$ gap, stop. In the process of Binary Search, we need two hyperparameters. One is λ_{max}, which is the maximum that λ can take, and it can ensure the NDCG $\leq$ threshold. The other is gap, which is the accuracy between the λ and λ_{target}, and also describes the interval length containing λ_{target}. As shown in the formula above, the interval $[0, \lambda_{max}]$ contains λ_{target} at the beginning, and due to binary search can make the interval continuously folded in half without missing λ_{target}. Then if the search process can directly find λ_{target}, it can directly stop the algorithm. Otherwise, the algorithm terminates when the interval's length $\leq$ gap and λ takes the left endpoint of the interval, that is, $\lambda = m_1$. Thus, if the gap is small enough, it can ensure that the NDCG is close to and not below the threshold. Time is the round of binary search,

and $2^{\text{Time}} \leq \frac{\text{gap}}{\lambda_{max}}$ can trigger the termination conditions of the algorithm, so $\text{Time} \leq (\lceil \log_2(\lambda_{max}/\text{gap}) \rceil)$.

The specific pseudocode is shown in Algorithm 1. In the offline scenario, FairSort first allocates exposure to all users through Top-K recommendations (line 1). Then, for each user, it sequentially performs re-ranking. The specific approach is as follows: Utilizing the current allocation of exposure, we calculate the fair lift velocity for each item using a heuristic function (line 4). Subsequently, a binary search is employed to determine the fairness weighting factor, thus obtaining the re-ranked recommendation list (lines 5-6). Finally, based on the re-ranked recommendation list, the exposure allocation on the provider side is readjusted (line 7).

B. FairSort For Online Scenario

In the online scenario, user requests arrive randomly. It is necessary to provide a recommendation service for each request within a certain time and ensure both sides' fairness. In this process, the user's recommendation quality and the provider's total exposure must be considered as process quantities. Thus, the user side calculates the average recommendation quality. While the total exposure can be calculated as follows:

$$E_{total} = c_{num} \times \sum_{rank=1}^{K} \frac{1}{\log_2(rank+1)}$$

where c_{num} is the number of user requests.

Pseudocode is shown in Algorithm 2. We can calculate the total exposure when each user request arrives, according to the above formula. This is followed by the calculation of each provider's e^{fair}, which allows us to establish a benchmark for fair allocation of exposure (line 1). At this time, according to the Top-K recommendation method, we allocate the exposure for each provider (line 2). Next, through the above formula (13), we inspire the fair lift velocity for each item (line 3). And then through the Binary Search Strategy, we achieve a fair Re-Ranking (lines 4-5). Finally, the allocation of exposure will be adjusted (line 6). Figure 3 shows this whole process.

C. Time Complexity Of FairSort

The time complexity of FairSort is analyzed as follows. First, we analyze the offline version, which provides a Re-Ranking service for each user in order. Thus, the outer loop has $O(m)$, and each time we perform a re-ranking service for a user, it needs to call the heuristic function that assigns fair lifting velocity for each provider. The time complexity is $O(l)$, where l is the number of providers. Then, the number of items involved in Re-Ranking is g, determined jointly with l^{ori} length and ratio. By binary search, the quick sort algorithm is performed,

and the time complexity is $O(g\log(g))$ [42]. Thus, the time complexity of the offline version is $O(m(l+g\log(g)))$, and the online version has no outer loop, so the time complexity of each request is $O(l+g\log(g))$.

V. EXPERIMENT

A. Datasets and Metrics

Ctrip Flight Dataset: The entire dataset contains data from 3,814 users, 6,006 kinds of air tickets, and 25,190 orders. It also provides basic information on users, air ticket class, air ticket price, flight time, airline company of the ticket, and other information. We adopt the state-of-the-art collaborative filtering air ticket recommendation algorithm [43] to process the data and obtain a preference matrix.

Amazon Review Dataset: We used data from [44], which has the largest data due to its large number of reviews. We pre-filtered items and users with less than 10 reviews or being reviewed and only consider reviews of items in the "Clothing Shoes and Jewelry" category, which has the largest number of reviews. Using the well-known matrix decomposition model [45] to estimate users' preference scores for items, the dataset does not provide information between items and providers. We then model the providers by clustering methods, with 1-100 items clustered into one category. The processed dataset contains 1,851 users, 7,538 items, and 161 providers.

Google Local DataSet: This dataset is unique, where each item represents an individual provider. This dataset contains reviews about local businesses from Google Maps. It also filtered items and users who participated in reviews less than 10 times, then obtained a dataset containing 3,335 users, 4,927 items (providers), and 97,658 reviews. The data is processed by using an implicit decomposition algorithm [46] based on location information.

Metrics and Hyperparameter Setting: On the provider side, Quality Weighted Fairness (QF) is measured by calculating the variance of the ratio between their exposure e_p and their own quality (named Variance of the ratio of exposure and relevance). For Uniform Fairness (UF), we calculate the variance of the ratio between their exposure e_p and the number of items they provide (named Variance of exposure). On the user side, we measure the (Variance of NDCG) of the user's recommendation list and compute the distribution of the NDCG. We count the user's total recommendation quality to evaluate the loss situation. The smaller the variance, the fairer the recommendation results. The greater the sum of $NDCG_u$, the smaller the loss of the recommendation quality [15]. Due to space limitations, only a formal description of the metric QF is given here, and the others are similar. We use DCF (Deviation from User/Customer Fairness) and DPF (Deviation from Provider Fairness) as the abbreviations for fairness metrics on the user and provider sides respectively.

The λ_{max} is selected from $\{2^2, 2^3, 2^4\}$. The gap is selected from $\{2^{-5}, 2^{-6}, 2^{-7}\}$.

The threshold is selected from $\{0.85, 0.90, 0.95\}$. The ratio is selected from $\{0.1, 0.2, 1\}$.

$$DPF = \mathbb{E}_{p\sim P}\left[\left(\frac{e_p}{\sum_{i\in I_p}\sum_{u\in U} v_{u,i}} - \mathbb{E}_{p\sim P}\left[\frac{e_p}{\sum_{i\in I_p}\sum_{u\in U} v_{u,i}}\right]\right)^2\right]$$

$$UIR = \frac{w_1 \cdot DCF/\mu_1 + w_2 \cdot DPF/\mu_2}{Utility}$$

B. Compared Approaches

1. **Top-K**: This algorithm directly recommends the Top-K items in the original recommendation list l_u^{ori}.
2. **Mixed-K**: Choose top (αK) relevant products first and then the remaining $(K - \alpha K)$ randomly.
3. **All Random**: Randomly select K items from the user's original recommendation list l_u^{ori} to recommend.
4. **Minimum Exposure**: The model helps UF by recommending the item that currently receives the least amount of exposure each time.
5. **FairRec (WWW 2020)**: This is a state-of-the-art algorithm that guarantees two-sided fairness based on a greedy strategy and heuristic rule, which guarantees EF1 fairness on the user side and guarantees at least Maximin Share (MMS) of exposure for most of the items [14].
6. **CPFair (SIGIR 2022)**: This is a state-of-the-art algorithm that guarantees two-sided fairness based on a greedy strategy and heuristic rule, which is a mixed-integer linear programming re-ranking algorithm (MILP) that seamlessly integrates fairness constraints from both the consumer and producer-side in a joint objective framework [16].
7. **TFROM (SIGIR 2021)**: This is a state-of-the-art algorithm that guarantees two-sided fairness based on a greedy strategy and heuristic rule, which also has two versions for offline and online recommendation [15].

C. Experiment Results and Analysis

1) Some Important Questions: - **RQ1**: Does Top-K recommendation lead to severely unfair exposure distribution on the provider side? - **RQ2**: Can FairSort guarantee that the recommendation quality for each user does not fall below a threshold? - **RQ3**: Can FairSort guarantee fairness for both sides while maintaining a high level of personalized recommendation?

2) Experiment Setting: - We conducted experiments for the offline situation on three datasets and evaluated the results of the algorithms at different K values. - By generating a random recommendation sequence to simulate an online scenario, each user is given 10 chances to be recommended, and the order is randomly disrupted. FairRec and CPFair are unsuitable for online recommendation scenarios, so we do not compare them in this experiment. -

To address RQ3 and better illustrate the comprehensive capabilities of FairSort, we adopted a metric introduced in CPFair [16], referred to as UIR (Un-Fairness Inhibition Rate) in equation (19). A smaller UIR indicates a model's stronger ability to maintain high personalized recommendation quality while ensuring both-side fairness. Where w_1 and w_2 are weighting parameters that the system designer can select depending on the priority for each fairness aspect. *Utility* is the user's average recommendation quality. In this paper, we select $w_1 = w_2 = 1$, and μ_1 and μ_2 represent correction parameters, each adopting the DCF and DPF of the worst-performing value, respectively. For instance, μ_1 selects the DCF of the Minimum Exposure model, while μ_2 chooses the DPF of the Top-K model. In this study, we conducted 150 experimental cases and selected 5 models, i.e., 3 (Datasets) $\times$ 25 (The range of K) $\times$ 2 (QF and UF) $\times$ 5 (Models) to evaluate the overall performance of models with respect to two-sided fairness.

Next, we will answer the previous three questions, providing evidence for the necessity, reliability, and effectiveness of FairSort. Our answer to RQ1 highlights the necessity of fair post-processing reordering for Top-K recommendation lists. Answering RQ2 demonstrates the reliability of our FairSort in ensuring personalized recommendation quality. Finally, our response to RQ3 validates the effectiveness of FairSort in simultaneously ensuring fairness on both sides while pursuing high-quality personalized recommendations.

3) Answer to RQ1: To answer this question, we need to examine the experimental results of Top-K based on UF and QF in both offline and online scenarios, as shown in fig (4, 5, 6, 7, 8, 9) at (c, d). We can draw the following conclusions: Whether based on the UF or QF concept, Top-K in an online scenario, over the course of the recommendations, exhibits a significant variance, resulting in highly uneven exposure allocation on the provider side. In the offline scenario, the unfairness of exposure on the provider side becomes more severe with the increasing value of K because Top-K is user-centric and focuses solely on personalized recommendations, neglecting provider-side fairness. The experimental results address RQ1 and underscore the importance of fair post-processing reordering for Top-K recommendations. Without it, there's an exponential rise in DPF with increasing recommendations, as demonstrated in figures (7, 8, 9) at (c, d).

(3-1) Provider-Side Fairness: We further analyze the performance of each model with respect to UF and QF respectively.

- Let's examine the UF performance of the Minimum Exposure model, which best aligns with the UF concept, as well as the All Random model, which can mitigate UF. As shown in fig (4, 5, 6, 7, 8, 9) at (c), in both online and offline scenarios, in contrast to Top-K (user-centric), Minimum Exposure (provider-centric) performs the best in terms of UF, converging to 0. All Random satisfies UF to some extent. Minimum Exposure (provider-centric) performs best because it follows the UF approach, prioritizing exposure fairness for providers. All Random benefits from randomness, ensuring equal exposure opportunities for each item, indirectly

satisfying UF to some extent. However, they do not consider user-side recommendation utility, significantly degrading recommendation quality.
- In both UF and QF contexts, in offline scenarios (fig (4, 5, 6) at (c,d)), CPFair attempts to mitigate the provider-side unfairness caused by Top-K, but its performance is limited.

4) Answer to RQ2: To answer this question, we compare FairSort and TFROM in the online recommendation scenario. TFROM employs a greedy algorithm to ensure recommendation quality, whereas FairSort uses a Binary Search mechanism to guarantee recommendation quality. Table I illustrates the distribution of recommendation quality for FairSort and TFROM across the three major datasets. From these results, it's evident that FairSort consistently ensures that the quality of each generated recommendation list does not fall below a certain threshold. In contrast, TFROM's greedy strategy results in many low-quality recommendation lists. This may potentially harm the user experience and lead to user attrition. Therefore, the reliability of using a greedy algorithm to ensure recommendation utility is subject to question.

In addition to addressing RQ2, across the three major datasets and in both online and offline recommendation scenarios, we analyze the varied capabilities of all models with respect to recommendation quality and user-side fairness.

(4-1) Recommendation Quality: Examining the experimental results in fig (4, 5, 6, 7, 8, 9) at (a), we observe that:

- In both online and offline scenarios, the user-centric Top-K does not sacrifice any recommendation quality, so it performs optimally. Conversely, both Minimum Exposure (provider-centric) and All Random models cause the most significant degradation in recommendation quality. The former is a provider-centric recommendation model that solely focuses on the provider's UF, neglecting personalized recommendation utility. The latter, due to its high degree of randomness, naturally leads to severe damage to the recommendation lists. We calculate the average recommendation quality by (total quality/c_{num}) in an online scenario.
- In both online and offline scenarios, the rest are fairness-aware models, and their performance in recommendation quality falls between Top-K and Minimum Exposure (provider-centric) these two extremes. The noteworthy point is that FairSort and TFROM only sacrifice less than 10% of the total recommendation quality in exchange for fairness on both sides, which can be maintained at a high level of quality recall. This is attributed to FairSort's implementation of the Minimum Utility Guarantee mechanism, which ensures the user's recommendation quality and enhances the algorithm's reliability. We can confidently entrust FairSort to post-processing. In contrast, TFROM relies on a greedy rule to ensure recommendation quality. While it maintains a high-quality recall, our experiments have revealed that the greedy rule is sometimes unreliable. When both fairness constraints are considered, some recommendation lists suffer significant declines in quality, as illustrated in Table I.

- The reliability of relying on a greedy algorithm to ensure recommendation utility is questionable. Examining the offline scenario in fig (4, 5, 6) at (a), it's evident that FairRec, CPFair, and TFROM significantly compromise user recommendation quality when K is small, which gradually alleviates as K increases. In contrast, FairSort exhibits more stability across different K values. This is because the former models employ a greedy strategy, making it hard to balance fairness constraints and maintain recommendation quality when K is small, resulting in subpar recommendation quality recall. While FairSort uses a Binary Search method that guarantees a minimum recommendation quality for each user regardless of K and considers both-side fairness constraints. Thus, FairSort exhibits stronger robustness and reliability regarding recommendation quality recall. This observation is also reflected in user-side fairness. In fig (4, 5, 6) at (b), models relying on greedy strategies encounter difficulties in ensuring fair user-side recommendations, especially when K is small, due to significant disruptions in recommendation quality. As a result, their Variance of NDCG shows notable fluctuations in such scenarios, whereas FairSort demonstrates more stable performance.
- In the offline scenario, illustrated in fig (4, 5, 6) at (a), CPFair's performance regresses to that of Top-K. Despite sacrificing minimal recommendation quality, its metrics for provider-side fairness are comparatively subpar.

In summary, concerning recommendation quality, Top-K outperforms all, whereas Minimum Exposure and All Random exhibit the worst performance. However, TFROM and FairSort strike a nuanced trade-off between these two extremes in exchange for both sides fairness.

(4-2) User-Side Fairness: Examining the experimental results in fig (4, 5, 6, 7, 8, 9) at (b), we conclude that:

- From the figures (7, 8, 9) at (b), we also display the late-stage convergence results of each model during the online recommendation process. In the online scenario, all models exhibit a trend where the variance of NDCG initially increases and then decreases. Because, at the beginning of recommendations, the average recommendation quality of all users is 0, leading to a variance of NDCG values of 0. However, as the recommendations proceed, some users receive recommendations first, causing a rapid increase in the NDCG variance. When it reaches its peak, it begins to decline gradually because, at this time, most of the users have been recommended at least once, and then with the increase in the number of recommendations, the user who did not get the recommendation opportunity gets the recommendation, the variance of the NDCG will decrease. Until all users have obtained at least one recommendation opportunity, as the recommendation process continues, all users' average recommendation quality variance will further converge to 0. However, different algorithms exhibit varying convergence capabilities, reflecting their distinct abilities

to ensure user-side fair recommendations.

5) Answer to RQ3: To address RQ3 and better illustrate the comprehensive capabilities of our proposed model FairSort, this is validated by the UIR metric in fig (10,11).

- From fig (4, 5, 6, 7, 8, 9) at (b), it can be observed that, in the three major datasets, both online and offline scenarios, Top-K ultimately achieves the best user-side fairness performance (converging to 0), because it doesn't compromise on recommendation quality. In contrast, Minimum Exposure (provider-centric) focuses solely on provider-side UF, significantly degrading recommendation quality. The experimental results across all three major datasets indicate that it inconsistently impacts the recommendation quality for different users, leading to the highest variance of NDCG.
- In both online and offline scenarios, the recommendation models with both-sides fairness awareness include FairSort (QF, UF), TFROM (QF, UF), along with FairRec and CPFair. All six of them can converge to the Top-K's performance in user-side fairness. Especially in the online scenario, FairSort-Quality-Weighted and FairSort-Uniform-Weighted stand out. FairSort achieves this performance through a Binary Search mechanism, ensuring consistent levels of recommendation quality sacrifice for each user. Thus, FairSort can guarantee user-side fair recommendations while ensuring the quality does not fall below a certain threshold. On the other hand, TFROM employs heuristic rules, prioritizing the repair of severely disrupted recommendation lists. This method can ensure user-side fairness. FairRec tailors a recommendation list for each user through a fixed service order. This can achieve EF1 fairness and guarantee user-side fair recommendations.

(5-1) In the context of QF ideology: - As illustrated in fig 10 at (a, b, c), the FairSort-Quality-Weighted model consistently delivers the best performance across all three major datasets. This indicates that our algorithm can significantly ensure fairness for both sides while pursuing higher personalized recommendation utility. - The UIR for the Top-K model consistently remains at 1. This is attributed to the fact that in the UIR formula (19), the DCF is 0, and the DPF/μ_2 is 1, where μ_2 considers the DPF of the Top-K model. The denominator computes the average recommendation quality, and since Top-K recommendations do not incur any quality losses, it is also 1. Consequently, the final UIR result remains 1. - It's worth noting that when K is relatively small, striking a balance between recommendation quality and both-side fairness is more challenging. TFROM, CPFair, and FairRec, employing greedy strategies, perform poorly under such challenges, as evidenced by their relatively high UIR. In contrast, FairSort demonstrates more stable performance. This is attributed to its Binary Search mechanism, which ensures the minimum recommendation utility while sacrificing quality to achieve fairness. - CPFair tends to regress to Top-K and exhibits poor overall performance. This is attributed to its consideration of a relatively coarse-grained fairness scenario, focusing on inter-group

fairness between users and items, with only two groups being considered. FairRec exhibits unstable performance due to its pursuit of EF1 fairness among individual users, a strong constraint that can lead to regression to Top-K recommendations and poor provider-side fairness performance. Thus, its overall performance is less stable.

(5-2) In the context of UF ideology: - As depicted in fig 11 at (a, b, c), the conclusions align closely with those for QF and will not be reiterated here. However, it's noteworthy that FairSort exhibits stronger and more stable capabilities in handling QF compared to UF.

VI. CONCLUSIONS

This paper proposes a re-ranking model FairSort to find a trade-off solution among user-side fairness, provider-side fairness, and personalized recommendations utility. Providing a novel perspective, we analogize the l_u^{ori} as a runway rather than a conventional knapsack. Aiming at offline and online scenarios, FairSort ensures provider-side fairness by assigning fair lift velocity for each item. Guided by theorem 1, we devise a binary search strategy enabling items on each runway to run at their velocities within a specified time λ, achieving re-ranking. This strategy ensures user-side fairness and guarantees each user a minimum recommendation utility. In addition to theoretical guarantees, the experimental results from three datasets show that FairSort can guarantee both-side fairness and provide a more reliable personalized recommendation.

Several interesting directions need further investigation. Firstly, this paper's item utility estimation method may be relatively simplified. Since fair exposure allocation is built upon item utility estimation, accurately and fairly estimating the utility of items becomes an important issue. Secondly, the implicit function requires further research to guarantee bounds on the degree of fairness or unfairness. Finally, based on this novel perspective, further research can explore multi-granularity fairness across multiple stakeholders and the re-ranking problem under a multidimensional evaluation system, considering factors such as fairness, diversity, and novelty.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China (No. 62372323, 62032016, 62102281).

VII. PROOF

A. Notations And Definition

1. K: The number of items recommended for each user
2. $NDCG_u(\lambda_0)$: When $\lambda = \lambda_0$, the NDCG of user u
3. $p_i(\lambda_0) = [V_{ui} + \lambda_0 \cdot \text{getFair}(i)]$: The score of item i in l_u^{ori}, when $\lambda = \lambda_0$

4. $\text{rank}(i)_{\lambda_0} \in [1, \text{len}(l_u^{ori})]$: The current position of item i in l_u^{ori}, when $\lambda = \lambda_0$
5. $NDCG_u(i,j)[\lambda_0] = \frac{V_{ui}}{\log_2(\text{rank}(i)_{\lambda_0}+1)} + \frac{V_{uj}}{\log_2(\text{rank}(j)_{\lambda_0}+1)}$: Items i and j, the NDCG components calculated based on their respective positions in list l_u^{ori}
6. $A_{\lambda_0} = \{(i,j) | \forall i,j \in I, i \neq j, p_i(\lambda_0) > p_j(\lambda_0), p_i(\lambda_0 + \Delta\lambda) < p_j(\lambda_0 + \Delta\lambda), \text{rank}(i)_{\lambda_0} \leq K, \Delta\lambda \to 0^+\}$: $\forall \lambda_0 \in [0, +\infty)$, the set A includes the partial order relationship (i,j) which had been broken when $\lambda = \lambda_0$, $\Delta\lambda \to 0^+$

B. Theory and Proof

Lemma 1: When $A_{\lambda_0} \neq \emptyset$, $\forall (i,j) \in A_{\lambda_0} \Rightarrow V_{ui} \geq V_{uj}$, where $\lambda_0 \in [0, +\infty)$.

Proof: Assume $V_{ui} < V_{uj} \Rightarrow$ when $\lambda = 0$, $p_i(0) - p_j(0) < 0$, with the increase of $\lambda \in [0, +\infty)$. $p_i(\lambda) - p_j(\lambda) = [V_{ui} + \lambda \cdot \text{getFair}(i)] - [V_{uj} + \lambda \cdot \text{getFair}(j)] = V_{ui} - V_{uj} + \lambda[\text{getFair}(i) - \text{getFair}(j)]$. $\because V_{ui} - V_{uj} < 0$, $\lambda \geq 0 \Rightarrow M(\lambda) = [p_i(\lambda) - p_j(\lambda)]$ will be constantly less than 0 or have the transition from <0 at the beginning to constant ≥ 0, while according to the definition of A_{λ_0}, $M(\lambda_0) = [p_i(\lambda_0) - p_j(\lambda_0)] > 0$, $M(\lambda_0 + \Delta\lambda) = [p_i(\lambda_0 + \Delta\lambda) - p_j(\lambda_0 + \Delta\lambda)] < 0 \Rightarrow (i,j) \notin A_{\lambda_0}$. Contradiction $\Rightarrow$ The assumption is not valid $\Rightarrow V_{ui} \geq V_{uj}$.

Theorem 1: $\lambda_0 \in [0, +\infty)$, $\forall u \in U$, l_u^{ori}, the initial $NDCG_u$ is 1, the NDCG value decreases monotonically as the λ_0 increases.

Proof: According to Lemma 1, $\forall (i,j) \in A_{\lambda_0} \Rightarrow V_{ui} \geq V_{uj}$, $\lambda_0 \in [0, +\infty)$. Define: $F(x) = \frac{1}{\log_2(1+x)}$, $x \in [1, +\infty)$, $x \uparrow \Leftrightarrow F(x) \downarrow$. $\because NDCG_u = \sum_{i=1}^{K} F(\text{rank}(i)_{\lambda_0}) \cdot V_{ul^{ori}u[i]}$, there must exist an order such that items i, j are swapped by adjacent positions to achieve one-by-one destruction of the partial order relations in the set A_{λ_0}. When $A_{\lambda_0} \neq \emptyset$, $\forall (i,j) \in A_{\lambda_0} \Rightarrow$

$$\begin{cases} p_i(\lambda_0) = p_j(\lambda_0 + \Delta\lambda) \\ p_j(\lambda_0) = p_i(\lambda_0 + \Delta\lambda) \end{cases}$$

$$\begin{cases} \text{rank}(i)_{\lambda_0} = \text{rank}(j)_{\lambda_0 + \Delta\lambda} \\ \text{rank}(j)_{\lambda_0} = \text{rank}(i)_{\lambda_0 + \Delta\lambda} \end{cases}$$

(exchange position) $\because V_{ui} \geq V_{uj} \Rightarrow V_{ui} - V_{uj} \geq 0$, $p_i(\lambda_0) - p_j(\lambda_0) \geq 0 \Rightarrow F(\text{rank}(i)_{\lambda_0}) - F(\text{rank}(j)_{\lambda_0}) \geq 0 \Rightarrow [V_{ui} - V_{uj}][F(\text{rank}(i)_{\lambda_0}) - F(\text{rank}(j)_{\lambda_0})] \geq 0 \Rightarrow V_{ui}F(\text{rank}(i)_{\lambda_0}) + V_{uj}F(\text{rank}(j)_{\lambda_0}) \geq V_{ui}F(\text{rank}(j)_{\lambda_0}) + V_{uj}F(\text{rank}(i)_{\lambda_0})$

$$\Rightarrow NDCG_u(i,j)[\lambda_0] \geq NDCG_u(i,j)[\lambda_0 + \Delta\lambda]$$

It means that the loss of NDCG will be continuous if exchanging position. $\Rightarrow \forall \lambda_0 \in [0, +\infty)$, $\forall (i,j) \in A_{\lambda_0} \Rightarrow NDCG_u(i,j)[\lambda_0] - NDCG_u(i,j)[\lambda_0 + \Delta\lambda] \leq$

$0 \Rightarrow NDCG_u[\lambda_0 + \Delta\lambda] - NDCG_u[\lambda_0] \leq 0$. When $A_{\lambda_0} = \emptyset$, NDCG remains unchanged, thus completing the proof.

REFERENCES

[1] A. Castelnovo, R. Crupi, G. Greco, D. Regoli, I. G. Penco, and A. C. Cosentini, "A clarification of the nuances in the fairness metrics landscape," *Scientific Reports*, vol. 12, no. 1, p. 4209, 2022.

[2] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[3] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, and T. Joachims, "Recommendations as treatments: Debiasing learning and evaluation," in *international conference on machine learning*. PMLR, 2016, pp.

[4] J. Ding, Y. Quan, X. He, Y. Li, and D. Jin, "Reinforced negative sampling for recommendation with exposure data." in *IJCAI*. Macao, 2019, pp. 2230–2236.

[5] A. Vardasbi, M. de Rijke, and I. Markov, "Cascade model-based propensity estimation for counterfactual learning to rank," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 2089–2092.

[6] C. Zhao, L. Wu, P. Shao, K. Zhang, R. Hong, and M. Wang, "Fair information representation learning for recommendation: A mutual perspective," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 4, 2023, pp. 4911–4919.

[7] Z. Zeng, R. Islam, K. N. Keya, J. Foulds, Y. Song, and S. Pan, "Fair representation learning for heterogeneous information networks," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 15, 2021, pp. 877–887.

[8] Y. Ge, S. Liu, R. Gao, Y. Xian, Y. Li, X. Zhao, C. Pei, F. Sun, J. Ge, W. Ou et al., "Towards long-term fairness in recommendation," in *Proceedings of the 14th ACM international conference on web search and data mining*, 2021, pp. 445–453.

[9] Y. Li, H. Chen, Z. Fu, Y. Ge, and Y. Zhang, "User-oriented fairness in recommendation," in *Proceedings of the web conference 2021*, 2021, pp. 624–632.

[10] Z. Fu, Y. Xian, R. Gao, J. Zhao, Q. Huang, Y. Ge, S. Xu, S. Geng, C. Shah, Y. Zhang et al., "Fairness-aware explainable recommendation over knowledge graphs," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 69–78.

[11] Z. Zhu, J. Kim, T. Nguyen, A. Fenton, and J. Caverlee, "Fairness among new items in cold start recommender systems," in *Proceedings of the 44th Inter-*

national ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 767–776.

[12] A. Beutel, J. Chen, T. Doshi, H. Qian, L. Wei, Y. Wu, L. Heldt, Z. Zhao, L. Hong, E. H. Chi et al., "Fairness in recommendation ranking through pairwise comparisons," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2212–2220.

[13] C. Xu, S. Chen, J. Xu, W. Shen, X. Zhang, G. Wang, and Z. Dong, "P-mmf: Provider max-min fairness re-ranking in recommender system," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 3701–3711.

[14] G. K. Patro, A. Biswas, N. Ganguly, K. P. Gummadi, and A. Chakraborty, "Fairrec: Two-sided fairness for personalized recommendations in two-sided platforms," in *Proceedings of the web conference 2020*, 2020, pp. 1194–1204.

[15] Y. Wu, J. Cao, G. Xu, and Y. Tan, "Tfrom: A two-sided fairness-aware recommendation model for both customers and providers," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 1013–

[16] M. Naghiaei, H. A. Rahmani, and Y. Deldjoo, "Cpfair: Personalized consumer and producer fairness re-ranking for recommender systems," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, pp. 770–

[17] R. M. Karp, *Reducibility among Combinatorial Problems*. Boston, MA: Springer US, 1972, pp. 85–103.

[18] J. Li, Y. Ren, and K. Deng, "Fairgan: Gans-based fairness-aware learning for recommendations with implicit feedback," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 297–307.

[19] L. Wu, L. Chen, P. Shao, R. Hong, X. Wang, and M. Wang, "Learning fair representations for recommendation: A graph-based perspective," in *Proceedings of the Web Conference 2021*, 2021, pp. 2198–2208.

[20] Y. Wu, R. Xie, Y. Zhu, F. Zhuang, A. Xiang, X. Zhang, L. Lin, and Q. He, "Selective fairness in recommendation via prompts," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, pp. 2657–2662.

[21] J. Wang, W. Ma, J. Li, H. Lu, M. Zhang, B. Li, Y. Liu, P. Jiang, and S. Ma, "Make fairness more fair: Fair item utility estimation and exposure redistribution," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 1868–

[22] M. Heuss, F. Sarvi, and M. de Rijke, "Fairness of exposure in light of incomplete exposure estimation," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, pp. 759–769.

[23] J. Liu, "Toward a two-sided fairness framework in search and recommendation," in *Proceedings of the 2023 Conference on Human Information Interaction and Retrieval*, 2023, pp. 236–246.

[24] Y. Wang, P. Sun, W. Ma, M. Zhang, Y. Zhang, P. Jiang, and S. Ma, "Intersectional two-sided fairness in recommendation," in *Proceedings of the ACM on Web Conference 2024*, 2024, pp. 3609–3620.

[25] C. Wang, Y. Liu, Y. Yu, W. Ma, M. Zhang, Y. Liu, H. Zeng, J. Feng, and C. Deng, "Two-sided calibration for quality-aware responsible recommendation," in *Proceedings of the 17th ACM Conference on Recommender Systems*, 2023, pp. 223–233.

[26] E. Budish, "The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes," *Journal of Political Economy*, vol. 119, no. 6, pp. 1061–1103, 2011.

[27] A. J. Biega, K. P. Gummadi, and G. Weikum, "Equity of attention: Amortizing individual fairness in rankings," in *The 41st international acm sigir conference on research & development in information retrieval*, 2018, pp. 405–414.

[28] L. Wang and T. Joachims, "User fairness, item fairness, and diversity for rankings in two-sided markets," in *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, 2021, pp.

[29] M. B. Zafar, I. Valera, M. Gomez-Rodriguez, and K. P. Gummadi, "Fairness constraints: A flexible approach for fair classification," *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 2737–2778, 2019.

[30] G. Goh, A. Cotter, M. Gupta, and M. P. Friedlander, "Satisfying real-world goals with dataset constraints," *Advances in neural information processing systems*, vol. 29, 2016.

[31] M. Zehlike, F. Bonchi, C. Castillo, S. Hajian, M. Megahed, and R. Baeza-Yates, "Fa* ir: A fair top-k ranking algorithm," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 1569–1578.

[32] A. J. Biega, K. P. Gummadi, and G. Weikum, "Equity of attention: Amortizing individual fairness in rankings," in *The 41st international acm sigir conference on research & development in information retrieval*, 2018, pp. 405–414.

[33] M. Wan, D. Zha, N. Liu, and N. Zou, "Modeling techniques for machine learning fairness: A survey," *arXiv preprint arXiv:2111.03015*, 2021.

[34] S. Caton and C. Haas, "Fairness in machine learning: A survey," *arXiv preprint arXiv:2010.04053*, 2020.

[35] R. Pollin, M. Brenner, S. Luce, and J. Wicks-Lim, *A measure of fairness: The economics of living wages and minimum wages in the United States*. Cornell University Press, 2008.

[36] D. A. Green and K. Harrison, "Minimum wage setting and standards of fairness," *IFS working papers*, Tech. Rep., 2010.

[37] A. Falk, E. Fehr, and C. Zehnder, "Fairness perceptions and reservation wages—the behavioral effects of minimum wage laws," *The Quarterly Journal of Economics*, vol. 121, no. 4, pp. 1347–1381, 2006.

[38] N. Engbom and C. Moser, "Earnings inequality and the minimum wage: Evidence from brazil," *American Economic Review*, vol. 112, no. 12, pp. 3803–47, 2022.

[39] C. Lin and M.-S. Yun, "The effects of the minimum wage on earnings inequality: Evidence from china," in *Income Inequality Around the World*. Emerald Group Publishing Limited, 2016, vol. 44, pp. 179–

[40] T. Joachims and F. Radlinski, "Search engines that learn from implicit feedback," *Computer*, vol. 40, no. 8, pp. 34–40, 2007.

[41] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 2002.

[42] C. A. Hoare, "Quicksort," *The computer journal*, vol. 5, no. 1, pp. 10–16, 1962.

[43] Q. Gu, J. Cao, Y. Zhao, and Y. Tan, "Addressing the cold-start problem in personalized flight ticket recommendation," *IEEE Access*, vol. 7, pp. 67 178–67 189, 2019.

[44] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *proceedings of the 25th international conference on world wide web*, 2016, pp. 507–

[45] D. Liang, J. Altosaar, L. Charlin, and D. M. Blei, "Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 59–66.

[46] P. He, J. Zhu, Z. Zheng, J. Xu, and M. R. Lyu, "Location-based hierarchical matrix factorization for web service recommendation," in *2014 IEEE international conference on web services*. IEEE, 2014, pp.

Guoli Wu was born in Guangdong, China, in 1999. He received his master's degree from Tianjin University, Tianjin, China, in 2024. With the goal to build a socially efficient and fair recommender system, his current research interests include service computing, recommender systems, and fair machine learning algorithms.

Zhiyong Feng (Member, IEEE) was born in 1965. He received the Ph.D. degree from Tianjin University, Tianjin, China, in 1996. He is currently a Professor with the College of Intelligence and Computing, Tianjin University, Tianjin, China. He has authored more than 200 articles, one book, and 40 patents.

His research interests include service computing, knowledge engineering, and software engineering. Dr. Feng is a distinguished member of China Association for Computing Machinery (ACM) and the Chairman of ACM China Tianjin Branch. He has served as General Chair and Program Committee Chair of numerous international conferences, such as ICWS, ICSS, APWeb-WAIM, etc.

Shizhan Chen (Member, IEEE) was born in 1975. He received the bachelor's degree in engineering and the Ph.D. degree in computer applications from Tianjin University, Tianjin, China, in 1998 and 2010, respectively. He is currently a Professor with the College of Intelligence and Computing, Tianjin University. He has authored or co-authored 50 academic articles in Chinese and international academic journals, magazines, and conferences, and applied for 40 national invention patents and 11 software copyrights. His research interests include service computing and service-oriented architecture.

Hongyue Wu is currently an associate professor at the College of Intelligence and Computing, Tianjin University, China. He received his Ph.D. degree from Zhejiang University, China, in 2018. From 2017 to 2018, he was a joint Ph.D. student at the University of Sydney, Australia. His research interests include service computing, mobile edge computing, and cloud computing. He has published more than 50 papers and won the Best Paper Award at the ICSOC 2017. He is a member of the IEEE.

Xiao Xue (Member, IEEE) was born in 1979. He received the bachelor's degree in engineering from North China Electric Power University, Beijing, China, in 2001, and the Ph.D. degree in engineering from the Institute of Automation, Chinese Academy of Sciences (CAS), Beijing, in 2007. He is currently a Professor with the College of Intelligence and Computing, Tianjin University, Tianjin, China. He has authored or co-authored more than 50 IEEE TRANSACTIONS and other high-quality articles and authored a book, *Computational Experiment Method of Complex System*. His research interests include service computing, computational experiment, and swarm intelligence. Dr. Xue is an Associate Editor of the IEEE TRANSACTIONS ON INTELLIGENT VEHICLES and an Editorial Board Member of the International Journal of Crowd Science. He is one of the Chairs of Young Experts in Services Computing (YESC) Committee.

Jianmao Xiao received the Ph.D. from the College of Intelligence and Computing, Tianjin University. He is an assistant professor at Jiangxi Normal University and the deputy director of the Jiangxi Provincial Engineering Research Center of Blockchain Data Security and Governance. He is a member of the CCF TCSC. His main research interests include blockchain, service computing, and intelligent software engineering. Dr. Xiao has authored more than 30 high-level academic papers, served as MONAMI 2022 Web Chair and ICSS 2022 PC Member. He also served as a reviewer for many domestic and international high-level journals and conferences in related fields.

Guodong Fan is currently pursuing a Ph.D. degree in Computer Science and Technology with the College of Intelligence and Computing, Tianjin University.

He is working on cognitive services. His main research interests are representation learning, service computing, and software repository mining.

Hongqi Chen was born in Rizhao, China, in 1994. She received the master's degree in software engineering from China University of Petroleum (East China), Qingdao, China, in 2019. She is currently pursuing a Ph.D. degree in software engineering with Tianjin University, Tianjin, China. Her current research interests include service computing, graph neural networks, and recommender systems.

Jingyu Li is a Ph.D. student at College of Intelligence of Computing, Tianjin University. In 2022, he received his bachelor's degree in Computer Science and Technology at Hohai University, Nanjing. His current research interests include service computing, reinforcement learning, and interactive recommendation.

Note: Figure translations are in progress. See original paper for figures.

Source: ChinaXiv — Machine translation. Verify with original.