Improved Rank Aggregation Under Fairness Constraint

Diptarka Chakraborty[1] , **Himika Das**[2] , **Sanjana Dey**[3] and **Alvin Hong Yao Yan**[1]
[1]National University of Singapore
[2]TU Wien
[3]UMONS
diptarka@comp.nus.edu.sg, dashimika121@gmail.com, info4.sanjana@gmail.com, alviny@u.nus.edu

Abstract

Aggregating multiple input rankings into a consensus ranking is essential in various fields such as social choice theory, hiring, college admissions, web search, and databases. A major challenge is that the optimal consensus ranking might be biased against individual candidates or groups, especially those from marginalized communities. This concern has led to recent studies focusing on fairness in rank aggregation. The goal is to ensure that candidates from different groups are fairly represented in the top-k positions of the aggregated ranking.

We study this fair rank aggregation problem by considering the Kendall tau as the underlying metric. While we know of a polynomial-time approximation scheme (PTAS) for the classical rank aggregation problem, the corresponding fair variant only possesses a quite straightforward 3-approximation algorithm due to Wei et al., SIGMOD'22, and Chakraborty et al., NeurIPS'22, which finds closest fair ranking for each input ranking and then simply outputs the best one.

In this paper, we first provide a novel algorithm that achieves $(2 + \varepsilon)$-approximation (for any $\varepsilon > 0$), significantly improving over the 3-approximation bound. Next, we provide a 2.881-approximation fair rank aggregation algorithm that works irrespective of the fairness notion, given one can find a closest fair ranking, beating the 3-approximation bound. We complement our theoretical guarantee by performing extensive experiments on various real-world datasets to establish the effectiveness of our algorithm further by comparing it with the performance of state-of-the-art algorithms.

1 Introduction

Ranking a list of alternatives to prioritize desirable outcomes among a set of candidates is ubiquitous across various applications, such as hiring, admissions, awarding scholarships, and approving loans. When multiple voters provide preference orders or rankings on candidates, which may conflict, the task of producing a single consensus ranking is the classical *rank aggregation* problem. This problem is central to many fields, from social choice theory [Brandt *et al.*, 2016] to information retrieval [Harman, 1992]. Its origins trace back to the 18th century [Borda, 1781; Condorcet, 1785], and it has been extensively studied from a computational standpoint over the past few decades [Dwork *et al.*, 2001; Fagin *et al.*, 2003; Gleich and Lim, 2011; Azari Soufiani *et al.*, 2013]. When formulated as an optimization problem, one of the most popular versions seeks to find a consensus ranking that minimizes the sum of distances to the input rankings [Kemeny, 1959; Young, 1988; Young and Levenglick, 1978; Dwork *et al.*, 2001; Ailon *et al.*, 2008].

In this paper, we address the rank aggregation problem with an additional fairness constraint on the final consensus ranking. Ranking algorithms are commonly used to select the top candidates for various opportunities and services, such as admissions or scholarships in the education system, job hiring, or the allocation of medical care during emergencies like pandemics. In today's context, it is essential for any ranking algorithm to produce a fair ranking to ensure equitable selection and to avoid the risk of reinforcing extreme ideologies or stereotypes about marginalized communities based on sensitive attributes such as gender, race, or caste [Costello *et al.*, 2016; Kay *et al.*, 2015; Bolukbasi *et al.*, 2016]. For example, systems like job and education reservations in India [Borooah, 2010] or affirmative action-based university admissions in the USA [Deshpande, 2005] have been implemented to address under-representation and discrimination.

We consider the notion of proportionate fairness, also known as p-fairness [Baruah *et al.*, 1996], which ensures that each of the protected classes within the population is fairly represented in the top "most relevant" (top-k) positions of the final consensus ranking. The study of proportionate fairness in the context of rank aggregation was first explored in [Wei *et al.*, 2022] and [Chakraborty *et al.*, 2022]. In this paper, we use the following definition of *fair ranking* from [Chakraborty *et al.*, 2022].[1]

Definition 1 (Fair Ranking)**.** Consider a partition of d candidates into g groups $G_1, \cdots, G_g$. For each group G_i ($i \in [g]$), let us consider two parameters $\alpha_i, \beta_i \in [0, 1]$. For $\bar{\alpha} = (\alpha_1, \cdots, \alpha_g)$, $\bar{\beta} = (\beta_1, \cdots, \beta_g)$, and $k \in [d]$, a ranking π (on d candidates) is said to be $(\bar{\alpha}, \bar{\beta})$-$k$-*fair* if for each G_i:

[1]Note, the definition used in [Wei *et al.*, 2022], though similar, is slightly restrictive.

- *Minority Protection*: The top-k positions $\pi(1), \ldots, \pi(k)$ contain at least $\lfloor \alpha_i \cdot k \rfloor$ candidates from G_i, and
- *Restricted Dominance*: The top-k positions $\pi(1), \ldots, \pi(k)$ contain at most $\lceil \beta_i \cdot k \rceil$ candidates from G_i.

It is important to note that other notions of fair ranking, such as top-k statistical parity, have been considered previously [Kuhlman and Rundensteiner, 2020]. However, this approach is quite restrictive and does not satisfy the criteria for proportionate fairness. For a concrete example demonstrating why proportionate fairness is a much stronger concept than statistical fairness, see [Wei *et al.*, 2022].

Given a set of n rankings provided by voters on d candidates, the *fair rank aggregation* problem asks to find a fair consensus ranking that minimizes the sum of distances to the input rankings. Various distance measures have been considered in the literature to capture the dissimilarity between pairs of rankings, with the *Kendall tau distance* – which counts the number of pairwise disagreements between two rankings – being one of the most popular. [Wei *et al.*, 2022] and [Chakraborty *et al.*, 2022] proposed the following simple algorithm: Find a closest fair ranking for each input ranking and then output the one with the minimum sum of distances. A straightforward application of the triangle inequality shows that this simple strategy can only achieve a 3-approximation for the fair rank aggregation given that the *closest fair ranking* problem can be solved optimally. Moreover, since the final output ranking is close to one of the input rankings, it is essentially influenced by the preference order of a single voter. To date, there is no improvement over this 3-factor approximation guarantee. It is also worth noting that without any fairness constraint, the classical rank aggregation problem (known to be NP-hard [Bartholdi *et al.*, 1989; Dwork *et al.*, 2002]) has an $(1 + \varepsilon)$-approximation algorithm for any $\varepsilon > 0$ [Mathieu and Schudy, 2009].

1.1 Our Contribution

The main contribution of our paper is the development of a new algorithm for the fair rank aggregation problem under proportional fairness. Our algorithm achieves a $(2 + \varepsilon)$-approximation, for any $\varepsilon > 0$. To demonstrate our result, we design a two-stage procedure. First, we introduce a new problem of partitioning a colored graph in a *colorful* manner while minimizing the cost of "backward" edges across the cut. We develop a novel algorithm to solve this problem exactly when the input graph is a weighted tournament that satisfies certain natural properties on edge weights. This problem can be thought of as a variant of the *constraint cut* problem on a special graph class, which is, in general, NP-hard. Different cut problems find applications in other fairness questions (e.g., [Dinitz *et al.*, 2022]), and thus, our result on the variant of the constraint cut problem could be of independent interest. Next, we construct a weighted tournament graph from the fair rank aggregation instance. We then apply the solution of an optimal colorful partitioning of that tournament and use the known PTAS for the rank aggregation problem on both partitions separately to produce a fair ranking over the entire set of candidates. Finally, we argue that the output ranking attains $(2 + \varepsilon)$-approximation for any $\varepsilon > 0$.

We implement our algorithm and compare it against baselines on multiple standard datasets by varying different parameters. Our results show that the output of our algorithm achieves a significantly better objective value (i.e., the sum of distances) compared to state-of-the-art algorithms for fair rank aggregation. Furthermore, although our theoretical analysis guarantees only a $(2+\varepsilon)$-approximation for our proposed algorithm, it consistently performs much better in practice – the output is almost always very close to an optimal solution.

Our next contribution is a generic fair rank aggregation algorithm that achieves a 2.881-approximation. We emphasize that our algorithm works *irrespective* of the fairness notion under consideration as long as there is an efficient procedure to compute a closest fair ranking for any input ranking (even an approximate close fair ranking procedure suffices albeit with worse approximation factor for the aggregation problem, see the full version[2]). Thus, our algorithm provides an approximation guarantee not only with respect to a specific type of fairness but also concerning any plausible definition of fairness. As an immediate corollary, we achieve 2.881-approximation for the fair rank aggregation under a stronger fairness notion like *block fairness* introduced by [Chakraborty *et al.*, 2022]. Further, our generic algorithm works even if the group information – which candidate belongs to which group – is not known (e.g., as in *robust fairness* [Kliachkin *et al.*, 2024]). Ours is the first generic approximation algorithm that breaks below the straightforward 3-factor bound obtained by the naive use of the triangle inequality. We present our generic (deterministic) algorithm in Section 5, whose running time can be improved significantly using random sampling and *coreset* construction; however, such a randomized procedure requires a much more intricate analysis (see the full version for the details).

1.2 Other Related Works

The rank aggregation problem without any fairness constraint has also been studied under different other metrics, including Spearman footrule [Dwork *et al.*, 2002], Ulam [Chakraborty *et al.*, 2021; Chakraborty *et al.*, 2023]. [Chakraborty *et al.*, 2022] considered the rank aggregation problem with the fairness constraint under both Spearman footrule and Ulam metric, and showed a 3-approximation guarantee. For the Ulam metric, they in fact provided a $(3 - \delta)$-approximation result, for some constant $\delta \leq 2^{-30}$, albeit only for a constant number of groups.

Apart from the rank aggregation problem, other ranking problems have also been studied under fairness. E.g., [Celis *et al.*, 2018] explored the problem of finding the closest proportional fair ranking to a given ranking for metrics such as Bradley-Terry, DCG, and Spearman footrule. Ensuring *robust fairness* in rankings has also been studied [Kliachkin *et al.*, 2024], where given an input ranking, the goal is to find a close ranking that is more fair, even when the protected attributes are not known. We emphasize that such an algorithm may not necessarily be able to find a good aggregate ranking.

[2]The full version of this paper is available at https://arxiv.org/abs/2505.10006

The rank aggregation problem is essentially the 1-clustering (1-median) problem, where the input is a set of rankings. The past few years have witnessed a surge in research on fair clustering [Huang *et al.*, 2019; Chen *et al.*, 2019; Bera *et al.*, 2019; Backurs *et al.*, 2019]. However, we must note that the notion of fairness in the general clustering context differs from that in the rank aggregation.

2 Preliminaries

Notations. For any $n \in \mathbb{N}$, let $[n]$ denote the set $\{1, 2, \cdots, n\}$. We refer to the set of all rankings (or permutations) over $[d]$ by $\mathcal{S}_d$. We consider any permutation $\pi \in \mathcal{S}_d$ as a sequence of numbers $\pi(1), \pi(2), \dots, \pi(d)$ where the rank of $\pi(i)$ is i. For any two elements $a, b \in [d]$ and a permutation $\pi \in \mathcal{S}_d$, we use the notation $a \prec_\pi b$ to denote that the rank of a is less than that of b in π.

Distance metric and fair rank aggregation. In this paper, we consider the Kendall tau distance to measure the dissimilarity between any two rankings or permutations.

Definition 2 (Kendall tau distance)**.** Given two permutations $\pi_1, \pi_2 \in \mathcal{S}_d$, the *Kendall tau distance* between them, denoted by $\mathcal{K}(\pi_1, \pi_2)$, is the number of pairwise disagreements between π_1 and π_2, i.e.,

$$\mathcal{K}(\pi_1, \pi_2) := |\{(a, b) \in [d] \times [d] \mid a \prec_{\pi_1} b \text{ but } b \prec_{\pi_2} a\}|$$

Next, we define the fair rank aggregation problem.

Definition 3. (Fair Rank Aggregation) Given a set S of rankings over d candidates that are partitioned into g groups $G_1, \cdots, G_g$, $\bar{\alpha} = (\alpha_1, \cdots, \alpha_g) \in [0, 1]^g$, $\bar{\beta} = (\beta_1, \cdots, \beta_g) \in [0, 1]^g$, and $k \in [d]$, the *fair rank aggregation* problem asks to find a $(\bar{\alpha}, \bar{\beta})$-$k$-fair ranking $\sigma \in \mathcal{S}_d$ that minimizes the objective function $\mathtt{Obj}(S, \sigma) := \sum_{\pi \in S} \mathcal{K}(\pi, \sigma)$.

Note that in the above definition, the minimization is over the set of all $(\bar{\alpha}, \bar{\beta})$-$k$-fair rankings in $\mathcal{S}_d$. When the set S is clear from context, for brevity, we simply refer to the objective value as $\mathtt{Obj}(\sigma)$. Let σ^* be an optimal fair aggregated rank, and $\mathsf{OPT}(S) := \mathtt{Obj}(S, \sigma^*)$. We call a $(\bar{\alpha}, \bar{\beta})$-$k$ fair ranking $\tilde{\sigma}$ a *c-approximate fair aggregate ranking* (for some $c \geq 1$) for the set S iff $\mathtt{Obj}(S, \tilde{\sigma}) \leq c \cdot \mathsf{OPT}(S)$.

Weighted tournament. A weighted tournament $T = (V, A)$ is a directed graph where for every pair of vertices $u, v \in V$, both the edges (u, v) and (v, u) are present with some non-negative weight. It is well-known that the rank aggregation problem can be cast as *feedback arc set* problem on a weighted tournament (see [Ailon *et al.*, 2008]), where the corresponding weighted tournament $T = (V, A)$ with weight function $w : A \to \mathbb{R}$ satisfies the following:

- *Probability Constraints:*

$$\forall i, j \in V,\ w(i, j) + w(j, i) = 1, \qquad (1)$$

- *Triangle Inequality:*

$$\forall i, j, k \in V,\ w(i, j) \leq w(i, k) + w(j, k). \qquad (2)$$

For a weighted tournament $T = (V, A)$ with weight function $w : A \to \mathbb{R}$, for any $v \in V$, the *in-neighborhood* of v is defined as $N(v) := \{u \in V \mid (u, v) \in A\}$, and the *weighted in-degree* of v is defined as $\delta(v) := \sum_{u \in N(v)} w(u, v)$.

3 Colorful Bi-partition on Tournaments

In this section, we first introduce the *colorful bi-partition* problem defined on a directed weighted graph with colored vertices. Then, we provide an algorithm to solve that problem when the input graph is a tournament that satisfies both the probability constraint and the triangle inequality. In the next section, we discuss how to use the solution of the colorful bi-partition problem on tournaments to get an approximation algorithm for the fair rank aggregation problem.

Consider a weighted directed graph $G = (V, A)$ with a weight function $w : A \to \mathbb{R}$ defined on arcs/edges and a color function $\mathtt{col} : V \to [g]$ (for some integer $g \geq 1$) defined on vertices, and $\bar{\alpha} = (\alpha_1, \cdots, \alpha_g) \in [0, 1]^g$, $\bar{\beta} = (\beta_1, \cdots, \beta_g) \in [0, 1]^g$. We call a subset $S \subseteq V$ $(\bar{\alpha}, \bar{\beta})$-*colorful* if for each color $i \in [g]$, S contains at least $\lfloor \alpha_i \cdot |S| \rfloor$ and at most $\lceil \beta_i \cdot |S| \rceil$ many vertices of color i, i.e.,

$$\forall i \in [g],\ \lfloor \alpha_i \cdot |S| \rfloor \leq \left| S \cap \mathtt{col}^{-1}(i) \right| \leq \lceil \beta_i \cdot |S| \rceil.$$

Definition 4 (Colorful Bi-partition Problem)**.** Given a weighted colored directed graph $G = (V, A)$ with $w : A \to \mathbb{R}$, $\mathtt{col} : V \to [g]$ (for some integer $g \geq 1$), $\bar{\alpha} \in [0, 1]^g$, $\bar{\beta} \in [0, 1]^g$, and an integer k, the *colorful bi-partition* problem asks to find a partitioning of V into (disjoint) sets L and $V \setminus L$ such that (i) $|L| = k$, and (ii) L is $(\bar{\alpha}, \bar{\beta})$-colorful; while minimizing the *cost of the partition* $(L, V \setminus L)$ defined as the total weights of the arcs going from $V \setminus L$ to L, i.e., $\mathtt{cost}(L, V \setminus L) := \sum_{(y,x) \in A : x \in L, y \in V \setminus L} w(y, x)$.

Note, in the above problem, we want only L to be colorful, so $V \setminus L$ need not be colorful. Since specifying the set L suffices to identify the partition $(L, V \setminus L)$, for brevity, we use $\mathtt{cost}(L)$ to denote $\mathtt{cost}(L, V \setminus L)$.

Next, we provide a (deterministic) algorithm to solve the colorful bi-partition problem on tournaments, satisfying both the probability constraint (Equation 1) and the triangle inequality constraint (Equation 2).

Theorem 5. *There is an algorithm that, given a weighted colored tournament $T = (V, A)$ with $w : A \to \mathbb{R}$, $\mathtt{col} : V \to [g]$ (for some integer $g \geq 1$), satisfying both the probability and the triangle inequality constraints, and an integer k, $\bar{\alpha} \in [0, 1]^g$, $\bar{\beta} \in [0, 1]^g$, finds an optimal colorful bi-partition in time $O(|A| + |V| \log |V|)$.*

Description of the algorithm. Our colorful bi-partition algorithm (Algorithm 1) for tournament T works as follows:

1. **Sorting vertices of each color:** For each color $i \in [g]$, arrange the vertices in non-decreasing order based on their weighted in-degrees (Line 4).
2. **Initial selection in L:** To select the vertices in L: Take the first $\lfloor \alpha_i \cdot k \rfloor$ vertices according to the sorted order of the vertices of color i, $\forall i \in [g]$ (Lines 5 – 11).
3. **Filling up the set L:** Order all the remaining vertices collectively in the non-decreasing order of their weighted in-degrees (Line 13), and continue adding elements to L as per this sorted order. If adding an element causes the number of elements from any color i to exceed the $\lceil \beta_i \cdot k \rceil$ bound, skip the element and proceed to the next in the collective ordering (Lines 14 – 21).

Algorithm 1 COLORFUL BI-PARTITION

```
 1: procedure COLBIPARTITION($T = (V, A), \bar{\alpha}, \bar{\beta}, k$)
 2:     Initialize an empty set $L$
 3:     for $i \leftarrow 1$ to $g$ do
 4:         Sort vertices of color $i$, i.e., in $\texttt{col}^{-1}(i)$, in non-decreasing order by their weighted in-degrees and process them in that sorted order
 5:         $count_i \leftarrow \lfloor \alpha_i \cdot k \rfloor$
 6:         for each vertex $v \in \texttt{col}^{-1}(i)$ do
 7:             if $count_i > 0$ then
 8:                 Add vertex $v$ to set $L$
 9:                 $count_i \leftarrow count_i - 1$
10:             end if
11:         end for
12:     end for
13:     Sort remaining vertices $(V \setminus L)$ collectively in non-decreasing order by their weighted in-degrees and process them in that sorted order
14:     for each $v \in V \setminus L$ do
15:         if $|L| \leq k$ then
16:             $i \leftarrow \texttt{col}(v)$
17:             if $|L \cap \texttt{col}^{-1}(i)| \leq \lceil \beta_i \cdot k \rceil$ then
18:                 Add vertex $v$ to set $L$
19:             end if
20:         end if
21:     end for
22:     return The partition $(L, V \setminus L)$
23: end procedure
```

4. **The final bi-partition:** Once L is filled, the remaining $V \setminus L$ forms the other set of the bi-partition (Line 22).

We defer the running time analysis to the full version.

Approximation guarantee. Let us now introduce a few notations that are useful for the analysis. Consider a graph $T = (V, A)$. For two sets $X, Y \subseteq V$, let $A(X, Y)$ denote the set of arcs from X to Y in T, i.e.,

$$A(X, Y) := \{(x, y) \in A \mid x \in X, y \in Y\}.$$

When X is the singleton set $\{x\}$, for notational convenience, we use $A(x, Y)$ to denote $A(X, Y)$. For any subset of arcs $A' \subseteq A$, we use $w(A')$ to denote the sum of weights of the arcs in A', i.e., $w(A') := \sum_{(x,y)\in A'} w(x, y)$. Note, for any partition $(P, V \setminus P)$, $\texttt{cost}(P) = w(A(V \setminus P, P))$.

We first argue that if we have a vertex in $V \setminus L$ with weighted in-degree smaller than or equal to that of some vertex in L, swapping them cannot lead to a new bi-partition with greater cost.

Lemma 6. *Let (L^*, R^*) be any (not necessarily colorful) bi-partition. Suppose there exists $x \in L^*$ and $y \in R^*$ such that $\delta(x) \geq \delta(y)$. Then for $\hat{L} := (L^* \setminus \{x\}) \cup \{y\}$, $\texttt{cost}(\hat{L}) \leq \texttt{cost}(L^*)$.*

By leveraging the probability constraints (Equation 1), we show the above lemma by arguing that for the bi-partitions

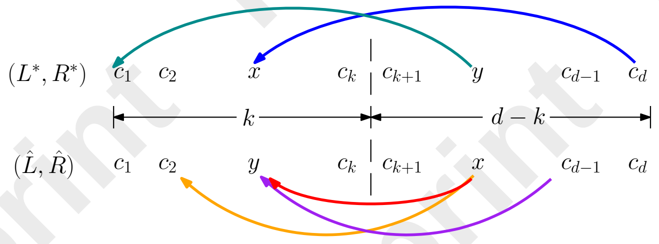

Figure 1: Vertices $\{c_1, c_2, \ldots, c_d\}$ are sorted by their weighted in-degrees. For the bi-partition (L^*, R^*), one of the edges in $A(y, L^*)$ is shown in cyan and one of the edges in $A(R^*, x)$ is shown in blue. For the bi-partition $(\hat{L}, \hat{R})$, the edge (x, y) is shown in red. Also, one of the edges in $A(x, L^*)$ is shown in orange and one of the edges in $A(R^*, y)$ is shown in violet.

(L^*, R^*) and $(\hat{L}, \hat{R})$ where $\hat{R} := V \setminus \hat{L}$ (as depicted in Figure 1), $w(A(\hat{R}, \hat{L})) - w(A(R^*, L^*)) = \delta(y) - \delta(x) \leq 0$. We defer the proof to the full version, and by assuming the above lemma, we prove the main result of this section.

Proof of Theorem 5. Let (L^*, R^*) be an (arbitrary) optimal colorful bi-partition. Recall that the bi-partition output by Algorithm 1 is (L, R). We first show that there exists a colorful bi-partition $\hat{L}$ (if not L^*) such that for all $i \in [g]$, $|\texttt{col}^{-1}(i) \cap \hat{L}| = |\texttt{col}^{-1}(i) \cap L|$ and $\texttt{cost}(\hat{L}) \leq \texttt{cost}(L^*)$.

Consider some $i \in [g]$ such that $|\texttt{col}^{-1}(i) \cap L^*| > |\texttt{col}^{-1}(i) \cap L|$ and some $j \in [g]$ such that $|\texttt{col}^{-1}(j) \cap L^*| < |\texttt{col}^{-1}(j) \cap L|$. Let $x = \arg\max_{u \in \texttt{col}^{-1}(i) \cap (L^* \setminus L)} \delta(u)$, and $y = \arg\min_{v \in \texttt{col}^{-1}(j) \cap (L \setminus L^*)} \delta(v)$. We now argue that $\delta(y) \leq \delta(x)$. Suppose y is added to L while executing Line 18 of Algorithm 1. Then, as Algorithm 1 iterates through the vertices in non-decreasing order of weighted in-degrees in Lines 14 – 21, it must have added y to L, and never encountered x before terminating. Therefore, it must be that $\delta(y) \leq \delta(x)$. Suppose instead that y is added to L while executing Line 8. As L^* is colorful, we have $\lfloor \alpha_i \cdot k \rfloor \leq |\texttt{col}^{-1}(j) \cap L^*| < |\texttt{col}^{-1}(j) \cap L|$. Therefore there must exist some $y' \in \texttt{col}^{-1}(j)$ such that $\delta(y) \leq \delta(y')$, and y' is added to L in our algorithm while executing Line 18. Then we conclude that $\delta(y) \leq \delta(y') \leq \delta(x)$ (as in Lines 14 – 21, it must have added y' to L and never encountered x before terminating). Now, consider the partition $L' = (L^* \setminus \{x\}) \cup \{y\}$. Observe, $(L', V \setminus L')$ is also a colorful bi-partition. By Lemma 6, $\texttt{cost}(L') \leq \texttt{cost}(L^*)$.

By using the above argument repeatedly, we obtain a colorful bi-partition $(\hat{L}, V \setminus \hat{L})$ such that $\forall_{i \in [g]}, |\texttt{col}^{-1}(i) \cap \hat{L}| = |\texttt{col}^{-1}(i) \cap L|$ and $\texttt{cost}(\hat{L}) \leq \texttt{cost}(L^*)$.

We next prove that $\texttt{cost}(L) \leq \texttt{cost}(\hat{L})$. Suppose $L \neq \hat{L}$. Then consider an $i \in [g]$ such that $\texttt{col}^{-1}(i) \cap \hat{L} \neq \texttt{col}^{-1}(i) \cap L$. By construction, our algorithm always adds vertices of color i to L by order of non-decreasing weighted in-degree. This implies that there exists a $v \in \texttt{col}^{-1}(i) \cap (\hat{L} \setminus L)$ such that $\delta(v) \geq \min_{y \in \texttt{col}^{-1}(i) \cap (L \setminus \hat{L})} \delta(y)$; otherwise, v would have been added to L. Then for $\hat{L}' = (\hat{L} \setminus \{v\}) \cup \{y\}$, by Lemma 6, $\texttt{cost}(\hat{L}') \leq \texttt{cost}(\hat{L})$. By repeating this swapping argument, we obtain the bi-partition (L, R). As each swap can only reduce the cost, we have that $\texttt{cost}(L) \leq \texttt{cost}(\hat{L}) \leq \texttt{cost}(L^*)$, showing that the output of Algorithm 1 is an optimal colorful bi-partition. $\square$

4 Approximating the Fair Aggregate Ranking

In this section, we show our main result by designing an algorithm that finds a $(2+\varepsilon)$-approximate fair aggregated rank for any $\varepsilon > 0$. In particular, we prove the following theorem.

Theorem 7. *For any $\varepsilon > 0$, there exists a $(2+\varepsilon)$-approximation algorithm for the fair rank aggregation problem that runs in time $O(d^3 \log d + nd^2)$, where $O(\cdot)$ hides the dependency on $1/\varepsilon$.*

To show the above result, we follow a two-step procedure.

- Step I: Create a weighted colored tournament $T = (V, A)$ on d vertices, which is an instance of the colorful bi-partition problem, and then use Algorithm 1 to get an optimal bi-partition $(L, V \setminus L)$.
- Step II: Use $(L, V \setminus L)$ and apply the known PTAS for the rank aggregation problem without fairness constraint on the portions of L and $V \setminus L$ separately to get an approximate fair aggregated rank.

We start with proving the following theorem, which, together with a known PTAS for the rank aggregation problem (without the fairness constraint), establishes Theorem 7.

Theorem 8. *Suppose there is a $t_1(d,n)$-time c_1-approximation algorithm $\mathcal{A}_1$ for some $c_1 \geq 1$ for the rank aggregation problem, and a $t_2(d)$-time c_2-approximation algorithm $\mathcal{A}_2$ for some $c_2 \geq 1$ for the colorful bi-partition problem on tournaments satisfying both the probability and the triangle inequality constraints. Then there exists a (c_1+c_2)-approximation algorithm for the fair rank aggregation problem with running time $O(nd^2 + t_1(d,n) + t_2(d))$.*

Description of the algorithm. Let us start by defining a few useful notations. For any set of elements $I \subseteq [d]$, let π_I represent the restriction of π to the elements in I. That is, delete all elements that are not contained in I from π. E.g., for $\pi = (2,6,3,5,1,4)$ and $I = \{1,2,3\}$, $\pi_I = (2,3,1)$.

Suppose we are given a set S of rankings over $[d]$, of size n. We construct a weighted tournament graph T from the rank aggregation instance by setting $V = [d]$. We set the $\texttt{col}$ function such that for all $v \in V$, $\texttt{col}(v) = i$ if $v \in G_i$. For every pair of elements a, b, let $n_{ab} = |\{\pi \in S \mid a \prec_\pi b\}|$. Set the weight of the edge (a,b) to be $w(a,b) = n_{ab}/n$. Observe that the edge weights of T obey the probability constraint and the triangle inequality constraint. Then we run the algorithm $\mathcal{A}_2$ with the graph T and parameters $\bar{\alpha}, \bar{\beta}, k$ and obtain a partitioning L and $V \setminus L$. For brevity, let $R := V \setminus L$.

We then restrict the input rankings to L and R; let $S_L = \{\pi_L \mid \pi \in S\}$ and $S_R = \{\pi_R \mid \pi \in S\}$. We apply the rank aggregation algorithm $\mathcal{A}_1$ on S_L and S_R separately, to obtain π_L^p and π_R^p respectively. Construct π^p by concatenating π_L^p with π_R^p and return π^p as the output aggregate ranking.

Proof sketch on approximation guarantee. The analysis splits the contribution of pairs to the objective cost of π^p into pairs crossing the partitioning (L,R) (thus fixing their pairwise order) and the pairs that are within either of the partitions. By selecting a partition that minimizes, in fact, c_2-approximates, the cost of pairs that cross it, the contribution to the objective cost by such pairs can be upper bounded by $c_2 \cdot \mathsf{OPT}$. By ordering the two partitions according to a c_1-approximate optimal rank over its elements, the contribution of such pairs can be upper bounded by $c_1 \cdot \mathsf{OPT}$, leading to an overall (c_1+c_2)-approximation. We defer the formal description of the algorithm and analysis to the full version.

Theorem 9. *[Mathieu and Schudy, 2009] There is a randomized algorithm for the rank aggregation problem that, given any $\varepsilon > 0$ and n rankings on d candidates, outputs a ranking with the cost at most $(1+\varepsilon)\mathsf{OPT}$ in time $O(\frac{1}{\varepsilon}d^3 \log d) + d2^{\tilde{O}(\varepsilon^{-6})} + O(nd^2)$ with high probability.*

Now, we are ready to prove our main result (Theorem 7).

Proof of Theorem 7. From Theorem 5, we have an algorithm for the colorful bi-partition problem with approximation factor $c_2 = 1$ and running time $O(d^2)$. From Theorem 9 we have an algorithm for rank aggregation with approximation factor $c_1 = 1+\varepsilon$ for any $\varepsilon > 0$ and running time $O(\frac{1}{\varepsilon}d^3 \log d) + d2^{\tilde{O}(\varepsilon^{-6})} + O(nd^2)$. Theorem 7 now follows directly from Theorem 8.

We remark that we can derandomize our algorithm by allowing an extra $d^{\tilde{O}(\varepsilon^{-12})}$ additive factor in the running time, due to the current best (deterministic) PTAS for the rank aggregation problem [Mathieu and Schudy, 2009]. $\square$

5 Improved Fair Rank Aggregation Using Closest Fair Ranking

In this section, we describe a generic algorithm for the fair rank aggregation under the Kendall-tau metric that works *irrespective* of the definition of fairness under consideration. The only thing we need to have is an efficient procedure to solve the closest fair ranking problem.

Given a ranking $\pi \in \mathcal{S}_d$, the *closest fair ranking problem* asks to find a fair ranking $\sigma \in \mathcal{S}_d$ that minimizes the Kendall-tau distance $\mathcal{K}(\pi, \sigma)$. For a host of fairness notions for which we are already aware of efficient closest fair ranking algorithms, our generic algorithm immediately provides a 2.881-approximation to the corresponding fair rank aggregation problem, breaking below the only known straightforward 3-approximation guarantee.

Theorem 10. *Suppose there is a $t(d)$-time algorithm $\mathcal{A}$ that solves the closest fair ranking problem. Then, there exists a 2.881-approximation algorithm for the fair rank aggregation problem with running time $O(n^3 d^3 \log d + n^3 t(d) + n^4 d \log d)$.*

The running time can be improved significantly, more specifically, the dependency on n can be brought down to (near-)linear, using random sampling and *coreset* construction (as detailed in the full version).

Implications to stricter fairness notions. Stronger fairness notions than that of Definition 1 have been studied in the context of fair rank aggregation, such as $(\bar{\alpha}, \bar{\beta})$-block-$k$-fairness (see the full version of [Chakraborty *et al.*, 2022]).

They provide an $O(d^2)$-time algorithm that finds a closest $(\bar{\alpha}, \bar{\beta})$-block-$k$-fair ranking. Therefore, as an immediate corollary of Theorem 10, we obtain a 2.881-approximation algorithm for $(\bar{\alpha}, \bar{\beta})$-block-$k$-fair rank aggregation in time $O(n^3 d^3 \log d + n^4 d \log d)$ (the running time can be reduced using randomization), improving upon previously known 3-approximation under this stricter fairness notion.

Description of the algorithm. Suppose we are given a set $S = \{\pi_1, \cdots, \pi_n\}$. Initialize $L = \emptyset$. For each π_i in S, find a closest fair ranking σ_i using $\mathcal{A}$, and add σ_i to the set L.

Iterate through all distinct 3-tuples $T := (\pi_i, \pi_j, \pi_k)$ from S. For each such tuple, construct an unweighted directed tournament graph G_T over $[d]$ vertices as follows: For every pair of elements (a, b), add the edge (a, b) if at least two rankings in T order a before b; otherwise, add the edge (b, a). Note, [Mathieu and Schudy, 2009] proposed an algorithm that finds a $(1+\gamma)$-approximation (for any $\gamma > 0$) to the feedback arc set problem on tournaments. Run this algorithm with $\gamma = 0.00001$ on G_T and delete the set of edges output by the algorithm to obtain a directed acyclic graph $\tilde{G}_T$. Let $\tilde{\pi}_T$ be the ranking obtained by taking the topological ordering of $\tilde{G}_T$. Next, use $\mathcal{A}$ to find a closest fair ranking $\tilde{\sigma}_T$ to $\tilde{\pi}_T$, and add it to L. Finally, output a ranking from L that minimizes the objective value (sum of distances to the input rankings).

Sketch of the analysis. We now provide a high-level overview of our analysis. Let σ^* be an (arbitrary) optimal fair aggregate ranking, and let $\mathsf{AVG} := \mathsf{OPT}/n$. For any $\pi_i \in S$, let I_i be the set of inverted pairs with respect to σ^*. Now, without loss of generality, assume no π_i is "close" (compared to AVG) to σ^*; otherwise, the corresponding closest fair ranking σ_i gives a good approximation.

Next, for any three inputs π_i, π_j, π_k, suppose "most" pairs of candidates are inverted in at most one π_r, $r \in \{i, j, k\}$ with respect to σ^* (i.e., belong to at most one inverted-pair set I_r, $r \in \{i, j, k\}$). Then, we can retrieve the correct orderings of most pairs in (unknown) σ^* by taking a majority vote from these three inputs. However, of course, due to the presence of certain "bad" pairs (that are inverted in more than one input), we may get cycles in the corresponding "majority tournament", and that necessitates considering solving the feedback arc set. When such a "nice" three-input tuple exists, we can bound the size of an optimal feedback arc set and eventually get a fair ranking very close to σ^*.

Now, suppose no such "nice" three-input tuple exists. Then, there exist two inputs such that every other input has a significant overlap in the inverted-pair sets with at least one of these two inputs. Now, consider two such inputs closest to (unknown) σ^* (this is for the sake of the analysis, and thus, algorithmically, we do not need to find these two inputs). One of these inputs must have at least $n/2$ other inputs with large overlap on inverted-pair sets. Then, we bound the objective value attained by the corresponding closest fair ranking and show that it achieves a good approximation guarantee.

We optimize multiple parameters that dictate closeness and the size of bad pairs to deduce our claimed approximation bound. We defer the formal description of the algorithm and a detailed analysis to the full version.

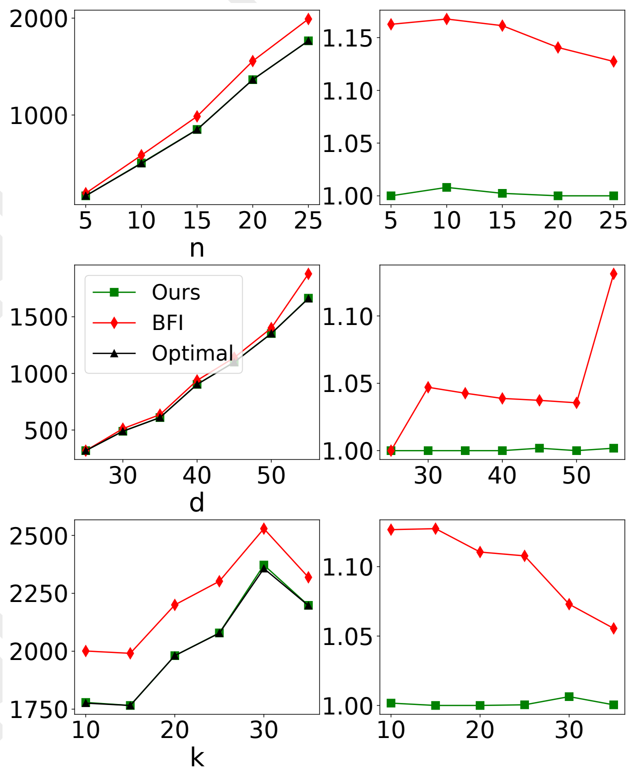

Figure 2: **Football dataset**. The x-axis is the value of the parameter (n, d or k). The y-axis is the objective value on the left figures, and the approximation ratio on the right figures.

6 Experiments

In this section, we provide an empirical evaluation of our algorithm on real-world datasets. We compare our algorithm against the performance of the best-known algorithms for the fair rank aggregation problem. The implementation is in Python 3.12 and experiments were performed on a laptop running Windows 11 using a Ryzen 6800HS processor and 16GB of RAM. We also use Integer Linear Programming (ILP) to find the optimal solution where possible for comparison. The Integer Linear Programs are implemented with CVXPY [Diamond and Boyd, 2016], using SCIP [Bolusani *et al.*, 2024] as the solver. The data and code is available on Github[3].

Datasets. We use two datasets introduced in previous work studying fairness in rank aggregation. The first dataset from [Kuhlman and Rundensteiner, 2020] is taken from a fantasy sports website for American football, where experts provide performance rankings over a set of (real) football players each week. The dataset contains rankings from experts across 16 weeks of the 2019 football season. In each week, the ranking of 25 experts on a set of players is given. We follow their work and divide the players into two groups based on the conference the player's team is in.

The second dataset from [Wei *et al.*, 2022] contains the rankings of 7 users over 268 movies. Each movie is placed into groups based on its genre, leading to 8 groups. We also

[3]https://github.com/Aussiroth/Improved-Fair-Rank-Aggregation

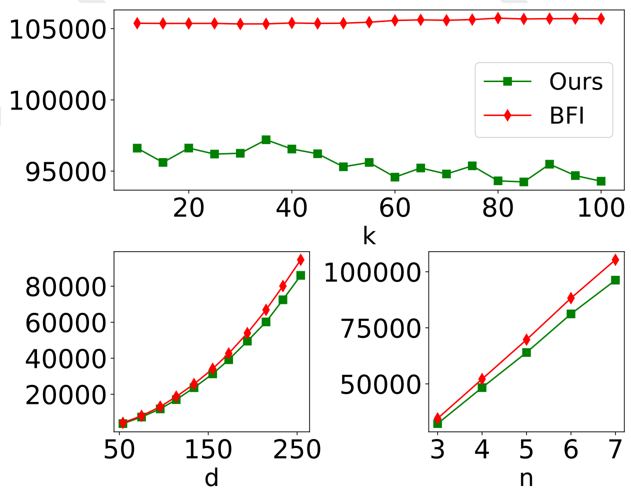

Figure 3: **Movielens dataset**. The x-axis is the value of the parameter (n, d or k). The y-axis is the objective value of the output rankings.

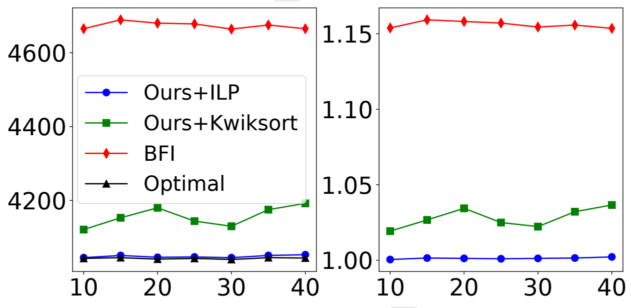

Figure 4: **Reduced Movielens dataset**. The x-axis is the value of k. The y-axis is the objective value of output rankings on the left figure, and the corresponding approximation ratio on the right figure.

preprocess the dataset to remove some movies to obtain a smaller dataset. This reduced dataset contains movies from the 4 largest genres and has 58 movies.

Algorithms. We implement our algorithm as described in the previous sections. The approximation algorithm used to solve rank aggregation is Kwiksort [Ailon *et al.*, 2008], and with Theorem 8, this implementation is, in theory, a 18/7-approximation. The algorithm was chosen as it has good asymptotic runtime and is easy to implement, making it much more practical than the known PTAS.

We implement the best-known algorithm for this problem in previous work as the baseline [Chakraborty *et al.*, 2022] (also as in [Wei *et al.*, 2022]). Recall that the algorithm finds a closest fair ranking for each input ranking and then outputs the one that gives the minimum objective value. We refer to this algorithm as BESTFROMINPUT (BFI). This is a 3-approximation algorithm.

For all experiments, the values of α_i, β_i are selected to be equal to the proportion of elements belonging to group i in the dataset. This is a natural option, as it maintains the proportion of elements from each group in the top-k.

Results. Figure 2 evaluates the algorithms for one instance (week 4) of the football dataset. We perform experiments that independently vary the number of input rankings n, the number of players d, and the value of the parameter k. For the experiments that vary n and d, we set $k = 15$. We see that our algorithm consistently performs better than BESTFROMINPUT, finding a fair aggregate ranking with a lower objective value in all of the instances, and is almost optimal.

Figure 3 evaluates the algorithms for the full Movielens dataset. We also perform experiments independently varying n, d, and k. For the experiments that vary n and d, we fix the parameter $k = 30$. As the number of elements is too large for the ILP to scale to, we only compare the objective value of the ranking that is output by the two algorithms. Our algorithm performs much better than BESTFROMINPUT in all experiments.

Figure 4 evaluates the algorithms for the reduced Movielens dataset. For this dataset, we focus on experiments for different values of k. We observe that in comparison to the football dataset, our algorithm with Kwiksort still performs much better than BESTFROMINPUT, but noticeably worse than optimal. We also plot an implementation of our algorithm that uses ILP to solve rank aggregation optimally. This performs better than using Kwiksort and is extremely close to optimal. This suggests that our algorithm is able to select a good set of top-k elements.

We also perform experiments where the values of α_i, β_i are varied instead of setting them as the group's proportion. For all experiments varying n and d, we explore various values for the parameter k. We conduct experiments on 15 other instances of the football dataset as well. These experiments show similar results and are available in the full version.

7 Conclusion and Future Work

In this work, we address the rank aggregation problem under the proportional fairness constraint as introduced in [Wei *et al.*, 2022; Chakraborty *et al.*, 2022]. We propose a novel algorithm to return a fair consensus ranking and establish (through theoretical analysis) that it achieves a $(2 + \varepsilon)$-approximation for any $\varepsilon > 0$, thereby improving upon the current best 3-factor approximation bound. Our experimental results further show that our algorithm consistently produces nearly optimal fair consensus rankings in practice. We also present a generic 2.881-approximation algorithm that works irrespective of the fairness definition as long as there is an efficient procedure to compute a closest fair ranking to any input.

An exciting open question is whether the approximation can be further improved, ideally achieving a PTAS that matches the current best approximation guarantee of the classical rank aggregation problem without fairness restrictions. It is noteworthy that a 2-factor is unavoidable for our algorithm (detailed in the full version), indicating that a fundamentally new approach may be required to enhance the approximation guarantee. Additionally, exploring other specific stricter fairness notions and demonstrating a comparable approximation guarantee (beating the bound obtained by our generic algorithm) for them presents another intriguing research direction.

Acknowledgments

This project was supported in part by an MoE AcRF Tier 2 grant (MOE-T2EP20221-0009), an MoE AcRF Tier 1 grant (T1 251RES2303), and a Google South & South-East Asia Research Award. Sanjana Dey was partially supported by the Fonds de la Recherche Scientifique – FNRS under Grant n° T.0188.23 (PDR ControlleRS).

References

[Ailon *et al.*, 2008] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):1–27, 2008.

[Azari Soufiani *et al.*, 2013] Hossein Azari Soufiani, William Chen, David C Parkes, and Lirong Xia. Generalized method-of-moments for rank aggregation. *Advances in Neural Information Processing Systems*, 26, 2013.

[Backurs *et al.*, 2019] Arturs Backurs, Piotr Indyk, Krzysztof Onak, Baruch Schieber, Ali Vakilian, and Tal Wagner. Scalable fair clustering. In *International Conference on Machine Learning*, pages 405–413. PMLR, 2019.

[Bartholdi *et al.*, 1989] J.J. Bartholdi, C.A. Tovey, and M.A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6(2):157–165, 1989.

[Baruah *et al.*, 1996] Sanjoy K Baruah, Neil K Cohen, C Greg Plaxton, and Donald A Varvel. Proportionate progress: A notion of fairness in resource allocation. *Algorithmica*, 15(6):600–625, 1996.

[Bera *et al.*, 2019] Suman Bera, Deeparnab Chakrabarty, Nicolas Flores, and Maryam Negahbani. Fair algorithms for clustering. *Advances in Neural Information Processing Systems*, 32, 2019.

[Bolukbasi *et al.*, 2016] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in Neural Information Processing Systems (NeurIPS)*, 29, 2016.

[Bolusani *et al.*, 2024] Suresh Bolusani, Mathieu Besançon, Ksenia Bestuzheva, Antonia Chmiela, João Dionísio, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Mohammed Ghannam, Ambros Gleixner, Christoph Graczyk, Katrin Halbig, Ivo Hedtke, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Dominik Kamp, Thorsten Koch, Kevin Kofler, Jurgen Lentz, Julian Manns, Gioni Mexi, Erik Mühmer, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Mark Turner, Stefan Vigerske, Dieter Weninger, and Lixing Xu. The SCIP Optimization Suite 9.0. Technical report, Optimization Online, February 2024.

[Borda, 1781] J. Borda. Mémoire sur les élections au scrutin. *Histoire de l'Académie Royale des Sciences*, 1781.

[Borooah, 2010] Vani K Borooah. Social exclusion and jobs reservation in india. *Economic and Political Weekly*, pages 31–35, 2010.

[Brandt *et al.*, 2016] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. *Handbook of computational social choice*. Cambridge University Press, 2016.

[Celis *et al.*, 2018] L. Elisa Celis, Damian Straszak, and Nisheeth K. Vishnoi. Ranking with fairness constraints. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 107, pages 28:1–28:15, 2018.

[Chakraborty *et al.*, 2021] Diptarka Chakraborty, Debarati Das, and Robert Krauthgamer. Approximating the median under the ulam metric. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 761–775. SIAM, 2021.

[Chakraborty *et al.*, 2022] Diptarka Chakraborty, Syamantak Das, Arindam Khan, and Aditya Subramanian. Fair rank aggregation. *Advances in Neural Information Processing Systems*, 35:23965–23978, 2022. Full version: arXiv preprint arXiv:2308.10499.

[Chakraborty *et al.*, 2023] Diptarka Chakraborty, Debarati Das, and Robert Krauthgamer. Clustering permutations: New techniques with streaming applications. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2023.

[Chen *et al.*, 2019] Xingyu Chen, Brandon Fain, Liang Lyu, and Kamesh Munagala. Proportionally fair clustering. In *International conference on machine learning*, pages 1032–1041. PMLR, 2019.

[Condorcet, 1785] M. J. Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. 1785. Reprinted by AMS Bookstore in 1972.

[Costello *et al.*, 2016] Matthew Costello, James Hawdon, Thomas Ratliff, and Tyler Grantham. Who views online extremism? individual attributes leading to exposure. *Computers in Human Behavior*, 63:311–320, 2016.

[Deshpande, 2005] Ashwini Deshpande. Affirmative action in india and the united states. -, 2005.

[Diamond and Boyd, 2016] Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[Dinitz *et al.*, 2022] Michael Dinitz, Aravind Srinivasan, Leonidas Tsepenekas, and Anil Vullikanti. Fair disaster containment via graph-cut problems. In *International Conference on Artificial Intelligence and Statistics*, pages 6321–6333. PMLR, 2022.

[Dwork *et al.*, 2001] Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622, 2001.

[Dwork *et al.*, 2002] Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation revisited, 2002.

[Fagin *et al.*, 2003] Ronald Fagin, Ravi Kumar, and Dandapani Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 301–312, 2003.

[Gleich and Lim, 2011] David F Gleich and Lek-heng Lim. Rank aggregation via nuclear norm minimization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 60–68, 2011.

[Harman, 1992] Donna Harman. Ranking algorithms., 1992.

[Huang *et al.*, 2019] Lingxiao Huang, Shaofeng Jiang, and Nisheeth Vishnoi. Coresets for clustering with fairness constraints. *Advances in neural information processing systems*, 32, 2019.

[Kay *et al.*, 2015] Matthew Kay, Cynthia Matuszek, and Sean A Munson. Unequal representation and gender stereotypes in image search results for occupations. In *ACM conference on human factors in computing systems*, pages 3819–3828, 2015.

[Kemeny, 1959] John G Kemeny. Mathematics without numbers. *Daedalus*, 88(4):577–591, 1959.

[Kliachkin *et al.*, 2024] Andrii Kliachkin, Eleni Psaroudaki, Jakub Mareček, and Dimitris Fotakis. Fairness in ranking: Robustness through randomization without the protected attribute. In *2024 IEEE 40th International Conference on Data Engineering Workshops (ICDEW)*, pages 201–208. IEEE, 2024.

[Kuhlman and Rundensteiner, 2020] Caitlin Kuhlman and Elke Rundensteiner. Rank aggregation algorithms for fair consensus. *Proceedings of the VLDB Endowment*, 13(12), 2020.

[Mathieu and Schudy, 2009] C Mathieu and W Schudy. How to rank with fewer errors. https://cs.brown.edu/people/wschudy/papers/fast_journal.pdf, 2009. Unpublished.

[Wei *et al.*, 2022] Dong Wei, Md Mouinul Islam, Baruch Schieber, and Senjuti Basu Roy. Rank aggregation with proportionate fairness. In *Proceedings of the 2022 international conference on management of data*, pages 262–275, 2022.

[Young and Levenglick, 1978] H Peyton Young and Arthur Levenglick. A consistent extension of condorcet's election principle. *SIAM Journal on applied Mathematics*, 35(2):285–300, 1978.

[Young, 1988] H Peyton Young. Condorcet's theory of voting. *American Political science review*, 82(4):1231–1244, 1988.