Withdrawal-Robust Rank Aggregation with Proportionate Fairness

Andy Zhuang[1] and Senjuti Basu Roy[2]

[1] Great Neck South High School, USA
azhuang3@student.gn.k12.ny.us
[2] New Jersey Institute of Technology, USA
senjuti.basuroy@njit.edu

Abstract. Proportionately fair rank aggregation requires each protected group to receive its proportional share of every prefix of a consensus ranking. This guarantee can fail when a candidate withdraws, because deletion changes the group proportions. We study complete rankings with two groups and Kendall–tau distance, requiring fairness to survive deletion of any one candidate without reordering the remaining candidates. We give an exact local characterization of this withdrawal-robust condition and prove that a robust group pattern exists for every pair of group sizes. A greedy construction finds such a pattern in $O(n)$ time. A separate dynamic program finds a closest robust ranking to any input in $O(n)$ time. Projecting each of m input rankings and selecting the projection with minimum total Kendall distance yields a deterministic 3-approximation to the optimal robust fair Kemeny aggregate in $O(mn + m^2 n \log n)$ time, where n is the number of candidates. Exhaustive experiments show that only 8.45% of ordinary fair binary patterns at $n = 16$ survive every withdrawal. In synthetic aggregation experiments, we compare our method with one that finds a closest ordinary fair ranking for each input and chooses the result with the lowest total Kendall distance to all input rankings. Our method increases cost by 5.3–7.0% on average while ensuring fairness after every single withdrawal. Our guarantee is limited to two groups and one withdrawal. It does not cover repeated withdrawals or establish fairness benefits in real applications.

Keywords: Rank aggregation · Proportionate fairness · Candidate withdrawal · Kemeny aggregation · Dynamic programming

1 Introduction

Rank aggregation combines several complete rankings of the same candidates into one consensus order. A hiring committee, for example, may aggregate the orders supplied by its members; a university may combine several evaluations into a waitlist. The classical Kemeny objective chooses an order minimizing the total number of pairwise disagreements with the input rankings [10,1].

When candidates belong to protected groups, the closest consensus need not give the groups comparable access to high positions. Wei et al. introduced

rank aggregation with *proportionate fairness* (RAPF)[14]. If group A contains a fraction a/n of the candidates, their definition requires every prefix of length k to contain either $\lfloor ka/n \rfloor$ or $\lceil ka/n \rceil$ members of A. Thus the order is fair at every possible stopping point, not merely at one chosen cutoff.

The motivating decisions, however, rarely stop with publication of an order. A waitlisted student accepts another offer; a job candidate withdraws; an applicant becomes ineligible. Deleting that person changes both the denominator and one group size. An order that was fair for (n, a) need not be fair for the new pool. Recomputing is possible, but it can reorder people who did not withdraw and complicates the explanation of a high-stakes decision. This paper asks a deliberately narrow question:

> Can we aggregate rankings into a proportionately fair order that remains fair after *any one candidate withdrawal*, without reordering the remaining candidates?

For two groups the answer is yes. We can use the same approach as the original RAPF paper, with a new algorithm that finds a closest robust ranking to each input.

Contributions. Our main results are as follows.

1. We define *withdrawal-robust p-fairness* and give an exact local characterization in terms of two consecutive prefix counts (Lemma 1).
2. We prove that every binary instance has a robust group pattern and give an $O(n)$ construction (Theorem 1). The construction walks through the intersection of five rounding windows corresponding to the present instance and the four ways a deletion can affect a prefix.
3. We give an exact $O(n)$ algorithm for finding a closest robust ranking to one input under Kendall distance (Theorem 2). Its dynamic program has at most two states per prefix.
4. We use these closest robust rankings to obtain a deterministic 3-approximation in $O(mn + m^2 n \log n)$ time (Theorem 3). This follows the existing method of projecting each input and choosing the best result. We include the proof to show how our projection algorithm gives an aggregation algorithm.
5. We supply a compact standard-library Python implementation, exhaustive tests, and experiments that compare the fraction of withdrawals that preserve fairness and the total Kendall distance for ordinary and robust fair rankings.

Scope. We consider complete rankings, Kendall–tau distance, two groups, and one candidate withdrawal. We do not claim protection against several sequential withdrawals, candidate insertion, missing rankings, or noisy group labels. With two groups, we can describe each prefix using just the count of one group. This makes the construction and algorithm easier to explain.

2 Preliminaries

There are n candidates in two groups, A and B, of sizes a and $b = n - a$. A ranking σ is a permutation of the candidates. Its *group pattern* is the binary string obtained by replacing an A candidate by 1 and a B candidate by 0. Since fairness depends only on this pattern, we use rankings and their patterns separately when useful.

For a ranking σ, let

$$c_\sigma(k) = |\{x \in A : \sigma(x) \le k\}|$$

be the number of group-A candidates in its first k positions, with $c_\sigma(0) = 0$. The sequence $c_\sigma(0), \ldots, c_\sigma(n)$ is an integer path whose increments are 0 or 1 and whose endpoint is a.

For an instance containing N candidates, q of them in group A, define the allowed count window at prefix k by

$$W_{N,q}(k) = \left[\left\lfloor \frac{kq}{N} \right\rfloor , \left\lceil \frac{kq}{N} \right\rceil \right] \cap \mathbb{Z}. \tag{1}$$

It contains one or two consecutive integers.

Definition 1 (Proportionate fairness [14]). *A ranking σ is* p-fair *if $c_\sigma(k) \in W_{n,a}(k)$ for every $k = 1, \ldots, n$.*

The constraint for B follows automatically because its prefix count is $k - c_\sigma(k)$. This one-dimensional representation is particular to two groups and is the main reason the algorithms below are simple.

The Kendall–tau distance $d_{\mathrm{KT}}(\sigma, \pi)$ is the number of candidate pairs ordered differently by σ and π. For a profile $P = (\pi_1, \ldots, \pi_m)$, define

$$\mathrm{cost}_P(\sigma) = \sum_{i=1}^{m} d_{\mathrm{KT}}(\sigma, \pi_i).$$

Kendall distance is a metric and can be computed in $O(n \log n)$ time by counting inversions.

If x is a candidate, $\sigma \setminus x$ denotes the restriction of σ obtained by deleting x and retaining the relative order of all other candidates.

Definition 2 (Withdrawal robustness). *A ranking σ is* withdrawal-robust p-fair*, or simply* robust*, if it is p-fair and $\sigma \setminus x$ is p-fair with respect to the new group sizes for every candidate x.*

Example 1. Write A and B for group labels. With $n = 8$ and $a = 3$, the pattern

$$B\ B\ A\ B\ B\ A\ B\ A$$

is p-fair. Delete the A in position 3. The first four positions of the restriction are all B, but the new instance has two A candidates among seven, so its length-4 prefix requires at least $\lfloor 4 \cdot 2/7 \rfloor = 1$ member of A. The remaining order is no longer p-fair. In contrast, the construction in Section 4 returns $B\ A\ B\ B\ A\ B\ B\ A$, which survives deletion of every position.

3 An exact local test for robustness

Suppose an item at original position t is deleted. A prefix of length k in the restriction has one of two forms:

- if $t > k$, it is the original first k positions;
- if $t \le k$, it is the original first $k+1$ positions with the deleted item removed.

This observation turns a quantifier over all candidates and all post-deletion prefixes into four local implications.

Lemma 1 (Local characterization). *Let $c(k) = c_\sigma(k)$. A binary pattern σ is robust if and only if it is p-fair and, for every $k = 1, \dots, n-1$, all applicable conditions below hold:*

$$c(k) < a \implies c(k) \in W_{n-1,a-1}(k), \tag{2}$$

$$k - c(k) < b \implies c(k) \in W_{n-1,a}(k), \tag{3}$$

$$c(k) > 0 \implies c(k+1) - 1 \in W_{n-1,a-1}(k), \tag{4}$$

$$k - c(k) > 0 \implies c(k+1) \in W_{n-1,a}(k). \tag{5}$$

Proof. Fix a post-deletion prefix length k. An A candidate exists below the original prefix exactly when $c(k) < a$. Deleting such a candidate leaves the prefix count equal to $c(k)$ and changes the instance to $(n-1, a-1)$, giving (2). The same argument for B gives (3).

An A candidate exists inside the first k positions exactly when $c(k) > 0$. After deleting it, original position $k+1$ shifts into the prefix, so the new A count is $c(k+1)-1$, giving (4). Deleting a B inside the prefix similarly gives count $c(k+1)$ and (5). These cases cover every deletion and every prefix, so the conditions are both necessary and sufficient.

Lemma 1 gives an $O(n)$ checker. It also supplies the transition test in the projection dynamic program of Section 5.

4 A robust pattern always exists

We now prove that the robust feasible set is nonempty for every pair (n, a). The edge cases are immediate. If $a \in \{0, n\}$ there is one pattern. If $a = 1$, every placement of the single A is p-fair, and every deletion leaves either another one-A instance or an all-B instance; the case $a = n-1$ is symmetric. Assume for the remainder that

$$2 \le a \le n-2. \tag{6}$$

For a real number x, write $I(x) = [\lfloor x \rfloor, \lceil x \rceil] \cap \mathbb{Z}$. Choose the desired last group $\omega \in \{A, B\}$. At each original prefix index j, intersect the following applicable

windows for $c(j)$:

$$
\begin{aligned}
&I(x_1), \quad x_1 = \frac{ja}{n}, && \text{original fairness;}\\
&I(x_2), \quad x_2 = \frac{ja}{n-1}, && \text{delete } B\text{, prefix untouched;}\\
&I(x_3), \quad x_3 = \frac{j(a-1)}{n-1}, && \text{delete } A\text{, prefix untouched;}\\
&I(x_4), \quad x_4 = \frac{(j-1)a}{n-1}, && \text{delete } B\text{, prefix shifted;}\\
&I(x_5), \quad x_5 = 1 + \frac{(j-1)(a-1)}{n-1}, && \text{delete } A\text{, prefix shifted.}
\end{aligned}
$$

The first window is used for every j. The untouched windows $I(x_2)$ and $I(x_3)$ are used through $j = n-2$; at $j = n-1$, use only $I(x_2)$ if $\omega = B$ and only $I(x_3)$ if $\omega = A$. The shifted windows are used for $j \geq 2$. Let $[L(j), U(j)] \cap \mathbb{Z}$ be this intersection. We call it the *robust corridor*.

Lemma 2 (The corridor is nonempty). *Under* (6)*,* $L(j) \leq U(j)$ *for every* j*.*

Proof. For $1 \leq j \leq n-2$, direct subtraction gives the ten differences among the five centers:

$$
\begin{aligned}
x_2 - x_1 &= \frac{ja}{n(n-1)}, & x_1 - x_3 &= \frac{j(n-a)}{n(n-1)},\\
x_1 - x_4 &= \frac{a(n-j)}{n(n-1)}, & x_5 - x_1 &= \frac{(n-a)(n-j)}{n(n-1)},\\
x_2 - x_3 &= \frac{j}{n-1}, & x_2 - x_4 &= \frac{a}{n-1},\\
x_5 - x_3 &= \frac{n-a}{n-1}, & x_5 - x_4 &= \frac{n-j}{n-1},\\
x_3 - x_4 &= \frac{a-j}{n-1}, & x_2 - x_5 &= \frac{j+a-n}{n-1}.
\end{aligned}
$$

Every displayed value lies strictly between -1 and 1 whenever its two windows are active. Thus the active centers have range less than one. Two distinct integer centers would have an integer difference of absolute value at least one, so that cannot occur either.

It follows that their rounding intervals have a common integer. One way to see this is to let s and r be the largest and smallest active centers. The interval $(s-1, r+1)$ has length greater than one and therefore contains an integer z. Then $|z - x_i| < 1$ for every active center, placing z in every $I(x_i)$; if an endpoint center is itself integral, the absence of distinct integer centers gives the same conclusion.

At $j = n-1$, the only possible unit gap is between $x_2 = a$ and $x_3 = a-1$; the definition keeps exactly one of them according to ω, and the remaining differences are strictly below one. At $j = n$, all active centers equal a. Hence the intersection is nonempty at every index.

The pointwise intersections must still be connected by 0/1 steps. The next lemma supplies that final fact and is also the algorithm.

Lemma 3 (Earliest robust path). *Set* $c(0) = 0$ *and, for* $j = 1, \ldots, n$, *set*

$$c(j) = \max\{c(j-1), L(j)\}. \tag{7}$$

Then $c(j) \in [L(j), U(j)]$, *every increment* $c(j) - c(j-1)$ *is 0 or 1,* $c(n) = a$, *and the last increment has group* ω.

Proof. Every continuing lower-bound function is nondecreasing and rises by at most one per index. At $j = 2$, the two newly activated shifted lower bounds are 0 and 1, while $L(1) = 0$, so activation also causes a rise of at most one. Near the end, constraints are removed rather than added. Therefore $L(j) \le L(j-1) + 1$.

Similarly, every continuing upper bound is nondecreasing. At $j = 2$, the new shifted upper bounds are 1 and 2, while $U(1) = 1$; later removal cannot lower a minimum. Thus $U(j)$ is nondecreasing. Inductively, $c(j-1) \le U(j-1) \le U(j)$, while Lemma 2 gives $L(j) \le U(j)$. Equation (7) is consequently inside the corridor. Moreover, $L(j) \le L(j-1) + 1 \le c(j-1) + 1$, so the increment is at most one; it is nonnegative by construction.

At $j = n$, original fairness pins $c(n) = a$. If $\omega = A$, the retained untouched-A window at $j = n - 1$ is $I(a-1) = \{a-1\}$, so the last increment is 1. If $\omega = B$, the retained untouched-B window is $I(a) = \{a\}$, so the last increment is 0.

Theorem 1 (Existence and construction). *For every* n *and* $0 \le a \le n$, *a withdrawal-robust p-fair binary pattern exists and can be constructed in* $O(n)$ *time and* $O(n)$ *output space.*

Proof. The edge cases were handled at the start of the section. Otherwise compute a constant number of floor and ceiling expressions per index and apply (7). The increments form a binary pattern by Lemma 3. It satisfies original fairness through $I(x_1)$. Each of (2)–(5) requires a count bound only when its deletion case can occur. Our construction includes the window needed for every possible deletion case. At each index where a window is used, we impose its bound even if that case does not occur. Thus all four implications hold, and Lemma 1 shows that the pattern survives every deletion. The work per position is constant.

The construction proves nonemptiness and gives a convenient canonical pattern. It does *not* optimize agreement with an input ranking. That is the purpose of the next section.

5 Exact projection onto the robust set

Let ρ be one input ranking. A projection seeks a robust ranking closest to ρ in Kendall distance. We first separate candidate identities from the group pattern.

Lemma 4 (Within-group order). *Some closest robust ranking to* ρ *preserves* ρ*'s relative order among the A candidates and, separately, among the B candidates.*

Proof. Robustness depends only on the group label at each position. If two candidates of the same group occur in the reverse of their order in ρ, exchange their identities while leaving the pattern fixed. The disagreement between that pair disappears. For a third candidate z outside the two output positions, its two pair relations are unchanged. If z lies between their output positions, compare its position in ρ with the two exchanged candidates. If it precedes both or follows both, exactly one of the two relations disagrees before and after the exchange. If it lies between them in ρ, both relations disagree before the exchange and neither disagrees afterward. Thus the total cannot increase. Repeating removes all within-group inversions without changing feasibility. This is the same exchange principle used for individual fair projection in RAPF [14].

For rankings whose within-group orders agree, Kendall distance has a simple prefix-area form.

Lemma 5 (Swap–area identity). *If σ and ρ agree on both within-group orders, then*

$$d_{\mathrm{KT}}(\sigma, \rho) = \sum_{k=1}^{n-1} |c_\sigma(k) - c_\rho(k)|. \tag{8}$$

Proof. An adjacent swap of unlike labels changes exactly one prefix count by one; a swap of like labels changes no prefix count. Therefore any transformation needs at least the area on the right. Conversely, at the boundary of a maximal interval where the two count paths differ, an adjacent $AB \leftrightarrow BA$ swap moves one coordinate toward the target and no coordinate away. Repeating uses exactly the displayed number of swaps. Once the paths and the within-group orders agree, the rankings are identical.

Let $r(k) = c_\rho(k)$. The dynamic program processes positions from top to bottom. A state (k, i) means that a candidate output pattern has length k and i members of A. Only $i \in W_{n,a}(k)$ is allowed. From (k, i) there are at most two transitions, to $(k+1, i)$ (place B) and $(k+1, i+1)$ (place A). A transition is retained exactly when the four implications of Lemma 1, evaluated at prefix k using the old count i and new count j, hold. Its added area is $|j - r(k+1)|$. The term at $k+1 = n$ is zero and may be omitted.

Formally, with $D[0, 0] = 0$ and all other entries infinite,

$$D[k+1, j] = |j - r(k+1)| + \min_{i \in \{j-1, j\}} \{D[k, i] : (i, j) \text{ is a valid robust transition}\}. \tag{9}$$

We store a predecessor for reconstruction. Finally, fill the chosen group-A slots with A candidates in their order in ρ, and likewise for B.

Theorem 2 (Exact robust projection). *Recurrence* (9) *returns a robust ranking of minimum Kendall distance to ρ. It runs in $O(n)$ time and uses $O(n)$ space when predecessors are stored (or $O(1)$ working space if only the distance is required).*

Proof. By Lemma 1, the dynamic program accepts exactly the robust group patterns. By Lemma 4, it is enough to consider outputs that preserve both within-group orders. For those outputs, Lemma 5 says that the objective is exactly the sum minimized by (9). Hence the reconstructed output is optimal.

At prefix k, ordinary p-fairness allows at most $\lfloor ka/n \rfloor$ and $\lceil ka/n \rceil$, so a layer has at most two states. Each state has two transitions with constant-time arithmetic. There are n layers, proving the time and space bounds.

6 Robust fair rank aggregation

We now use the projection algorithm to combine several input rankings into one consensus ranking.

Definition 3 (Robust RAPF). *Given $P = (\pi_1, \ldots, \pi_m)$, robust RAPF asks for*

$$\sigma^* \in \arg\min_{\sigma \text{ robust}} \text{cost}_P(\sigma).$$

The benchmark here is the best *robust* fair ranking, not the best ordinary p-fair ranking. No approximation to the latter is claimed without a separate bound on the price of robustness.

Proposition 1 (Hardness). *Computing an exact robust RAPF solution is NP-hard when there are four or more input rankings, even with two nonempty groups.*

Proof. Reduce classical Kemeny aggregation, which is NP-hard for four rankings [10]. Add one dummy candidate in group A, put every original candidate in group B, and place the dummy last in every input. With a single A, every group pattern is p-fair and withdrawal-robust. Pairwise costs among original candidates are unchanged, while placing the dummy last adds zero cost and is always optimal because every input places it there. Thus an exact robust solution reveals an exact Kemeny solution on the original candidates.

Algorithm. For every input π_i, compute its exact robust projection τ_i using Theorem 2. Evaluate all m candidates τ_i against the whole profile, and return the one of minimum cost. We call this procedure Robust-Aggregate.

Theorem 3 (Aggregation guarantee). Robust-Aggregate *is a deterministic 3-approximation for robust RAPF. Its running time is $O(mn + m^2 n \log n)$ and its space usage is $O(mn)$.*

Proof. Let σ^* be an optimal robust aggregate and let $\text{OPT} = \text{cost}_P(\sigma^*)$. Averaging gives an input π_r with

$$d_{\text{KT}}(\pi_r, \sigma^*) \le \text{OPT}/m.$$

Because τ_r is a closest robust projection of π_r and σ^* is robust,

$$d_{\text{KT}}(\tau_r, \pi_r) \le d_{\text{KT}}(\sigma^*, \pi_r).$$

For every input π_j, the triangle inequality gives

$$d_{\mathrm{KT}}(\tau_r, \pi_j) \le d_{\mathrm{KT}}(\tau_r, \pi_r) + d_{\mathrm{KT}}(\pi_r, \sigma^*) + d_{\mathrm{KT}}(\sigma^*, \pi_j).$$

Summing over j yields

$$\mathrm{cost}_P(\tau_r) \le 2m\, d_{\mathrm{KT}}(\pi_r, \sigma^*) + \mathrm{OPT} \le 3\mathrm{OPT}.$$

The algorithm returns the cheapest projection, so its cost is no larger than $\mathrm{cost}_P(\tau_r)$.

The m projections take $O(mn)$ time. Scoring m outputs against m inputs uses m^2 Kendall computations, each $O(n \log n)$. Storing the profile and projections takes $O(mn)$ space.

The proof uses the same triangle-inequality argument as earlier work on fair rank aggregation [14,8]. Our new step is the exact algorithm for finding a closest withdrawal-robust ranking. The 2.881-approximation of Chakraborty et al. [7] can also use our projection algorithm as a subroutine. We give the factor-3 proof because it uses only the triangle inequality and we can present the whole argument here.

7 Experiments

All experiments use the accompanying `robust_rank.py` and `experiments.py`. The implementation uses no optimization package; the core library uses only the Python standard library. Random seeds are fixed. Our questions are:

1. Is withdrawal robustness already automatic for p-fair patterns?
2. How much Kemeny cost does robustness add relative to the ordinary best-projection RAPF baseline?
3. How close is the 3-approximation to the exact robust optimum on small instances?

7.1 Setup

Exhaustive patterns. For each even $n \in \{6, 8, 10, 12, 14, 16\}$ and every non-trivial group size, we enumerated all binary patterns, retained the p-fair ones, and tested every single-position deletion.

Synthetic profiles. For $n \in \{12, 20, 40, 80\}$ we ran 24 trials. Each profile has nine voters, a random group-A set containing approximately 40% of candidates, and a common central ranking. Each voter is produced by $3n$ uniformly random adjacent transpositions of the center. We compare the ordinary best-p-fair-projection output with Robust-Aggregate. The reported overhead is $(C_{\mathrm{robust}} - C_{\mathrm{ordinary}})/C_{\mathrm{ordinary}}$. This measures the relative difference in cost between the two algorithms.

Table 1. Synthetic aggregation results (24 trials per row). Survival is the fraction of candidate deletions after which the published restriction remains p-fair.

n	mean overhead	ordinary survival	robust survival	robust time
12	5.28%	75.35%	100%	0.85 ms
20	5.47%	65.63%	100%	1.44 ms
40	5.90%	41.67%	100%	2.95 ms
80	6.97%	28.49%	100%	6.19 ms

Exact small instances. At $n = 7$, $a = 3$, and five voters, we enumerated all 5,040 permutations, kept the 576 robust rankings, and computed the exact robust optimum for 24 random profiles.

Sushi preferences. We also used 200 preference orders from the PrefLib Sushi data. The data have no protected attribute, so we use an example grouping: tuna-based items (maguro, toro, and tekka-maki) versus other items. This lets us test the algorithms on real preference orders.

7.2 Results

Robustness is a real additional constraint. Figure 1(a) shows that ordinary p-fairness does not generally survive withdrawal. Every fair pattern happened to be robust at $n = 6$, but the fraction falls to 45.61% at $n = 10$ and 8.45% at $n = 16$. At $n = 16$ the least stable fair pattern survived only 12.5% of its possible deletions.

The empirical price was modest in this setup. Table 1 and Figure 1(b) show a mean cost premium between 5.28% and 6.97%. Meanwhile, ordinary-output survival fell with n, reaching 28.49% at $n = 80$, whereas every robust output survived all deletions by definition. Runtime remained below 20 ms for these small profiles on the test machine; the asymptotic analysis, rather than these machine-dependent values, is the primary efficiency result.

The 3 bound was conservative. Against the exact robust optimum at $n = 7$, the mean approximation ratio was 1.049 and the maximum was 1.147 across 24 profiles. This does not improve the worst-case theorem, but it checks that the implementation and exact benchmark agree on small instances.

Real preference orders showed the same mechanism. On the illustrative Sushi partition, the ordinary output survived 50% of single-item deletions. The robust output survived 100%, with costs 2,758 versus 2,574, a 7.15% premium.

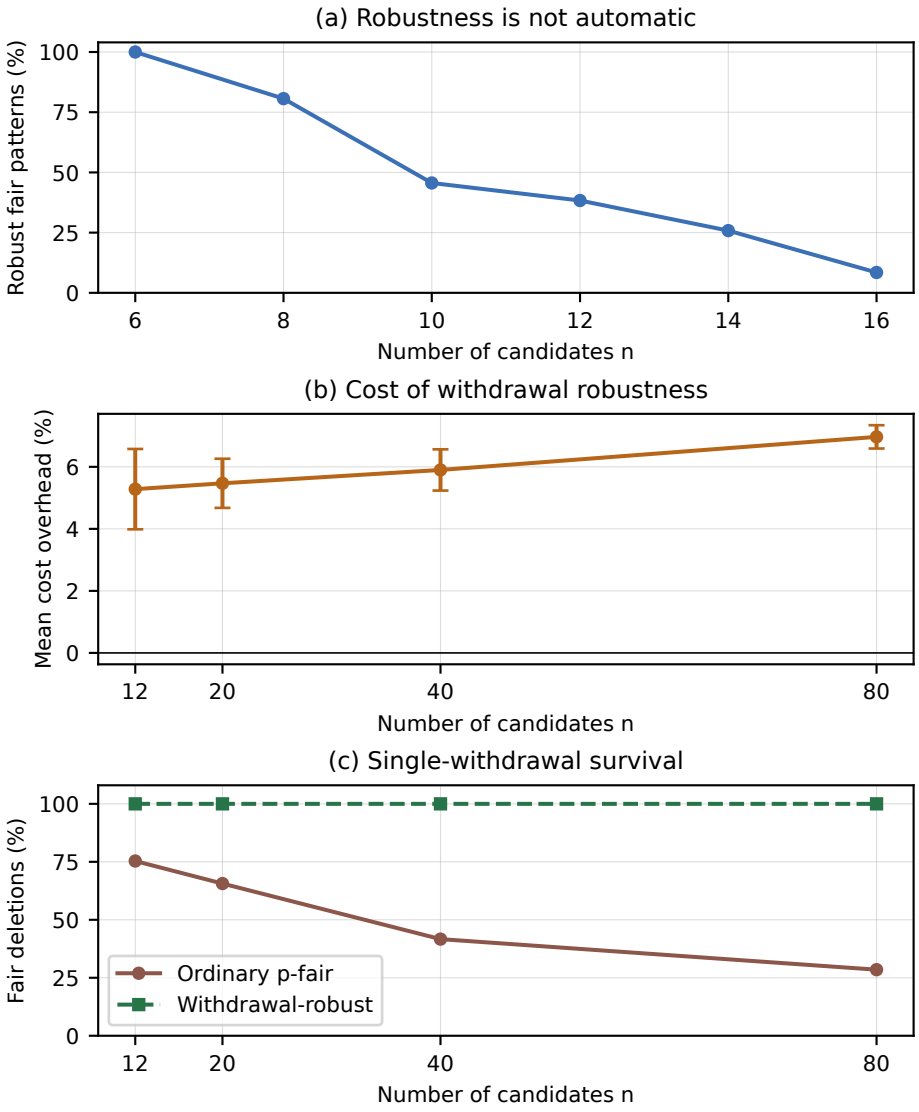

Fig. 1. (a) Fraction of ordinary p-fair patterns that survive every single deletion, exhaustive over all binary group sizes. (b) Robust Kemeny-cost overhead over the ordinary fair baseline; bars show standard error across 24 trials. (c) Fraction of deletions preserving fairness for the two synthetic aggregation outputs.

7.3 Reproducibility and limitations

The test suite exhausts every input group pattern through $n = 8$ and compares the projection result with brute force. It also checks the construction for all group sizes through $n = 80$ and verifies the 3-approximation inequality on small random profiles. These tests have limits. We used only one way of generating random rankings, and the groups in the synthetic and Sushi tests do not represent protected groups in a real application. The results do not show whether the method would improve fairness in real decisions. Before using it in practice, we would need to choose groups that make sense for the application, state the fairness goal, and test it on data from that setting.

8 Related work

Kemeny rank aggregation and its approximation algorithms are classical [10,1]. Wei et al. introduced proportionately fair rank aggregation, exact binary individual projection, multi-group projection algorithms, and the projection-based approximation framework on which our aggregation proof builds [14]. Later work generalized prefix fairness constraints and improved static approximation factors [8,7]; recent work gives an exact fair top-k algorithm and a 2-approximation for full fair aggregation under Spearman footrule [9]. Ando and Yokoi also study how to find a closest ranking that satisfies constraints on its prefixes [3]. Here we work out the conditions for surviving one deletion and use them to give a direct linear-time algorithm for two groups.

Other work considers different fairness goals or changes to the input. Fairer Together optimizes group exposure rather than exact prefix counts [5]; PreFAIR aggregates partial preferences [6]. Efficient Dynamic Rank Aggregation maintains an unconstrained aggregate as rankings arrive, under a footrule objective [2]. Israel and Brill remove selected alternatives and recompute approval-based proportional rankings [11]; dynamic Pfair scheduling adapts online as tasks arrive and depart [4]. Both permit the output to change. In our problem, we choose the ranking in advance and require it to remain fair after any one candidate is removed, without changing the order of the remaining candidates. The term "robust fairness" is also used in work on randomized rankings built without knowing the candidates' protected groups [12]. In our paper, robustness means that fairness survives the removal of any one candidate.

More broadly, our question is about keeping a solution valid when the input changes. Related questions arise in quota methods for apportionment and the chairman-assignment problem [13]. Those models extend seat allocations or schedules. Here, a candidate can be removed from any position, which changes the group proportions, and every prefix of the remaining order must still be fair. Prior work on proportionately fair rank aggregation assumes a fixed candidate set. In our literature search through August 2026, we did not find earlier work requiring a p-fair ranking to remain fair after any one candidate withdrawal without reordering. Our main results are the local test for this requirement, the

construction of a robust binary pattern, and the linear-time projection algorithm. The step from projection to aggregation follows existing work.

9 Conclusion

A prefix-fair ranking is not necessarily safe under the event that motivates many waitlists and shortlists: one candidate leaves. For two groups, this failure can be prevented at the time of aggregation. We gave a linear-time robust-pattern construction, a linear-time exact projection algorithm, and a deterministic 3-approximation for the resulting robust Kemeny problem. The proofs use rounding intervals, prefix sums, adjacent swaps, dynamic programming, and the triangle inequality.

The most immediate theoretical question is what changes with three or more groups. The prefix-count path then has several coordinates whose choices must sum to the prefix length, so the one-dimensional corridor proof no longer applies. Another useful next step is to allow a small, fixed number of withdrawals and study how much we must reorder the remaining candidates when fairness fails.

Author Contributions Senjuti Basu Roy introduced Andy Zhuang to the general topic of rank aggregation. Andy Zhuang identified the specific problem of preserving proportionate fairness after a candidate withdrawal, developed the algorithms and proofs, implemented the methods, conducted the experimental evaluation, and wrote the manuscript.

Funding Statement The authors received no specific funding for this study.

Conflicts of Interest The authors declare no conflicts of interest to report regarding the present study.

References

1. Ailon, N., Charikar, M., Newman, A.: Aggregating inconsistent information: Ranking and clustering. Journal of the ACM **55**(5) (2008)
2. Alimi, S., Mehrabiun, M., Zarei, A.: Efficient dynamic rank aggregation. arXiv:2509.02885 (2025)
3. Ando, S., Yokoi, Y.: Ranking and rank aggregation with matroid prefix constraints. arXiv:2607.07153 (2026)
4. Baruah, S.K., Gehrke, J., Plaxton, C.G., Stoica, I., Abdel-Wahab, H., Jeffay, J.: Fair on-line scheduling of a dynamic set of tasks on a single resource. Information Processing Letters **64**(1), 43–51 (1997)
5. Cachel, K., Rundensteiner, E.: Fairer Together: Mitigating disparate exposure in Kemeny rank aggregation. In: ACM Conference on Fairness, Accountability, and Transparency (2023)

6. Cachel, K., Rundensteiner, E.: PreFAIR: Combining partial preferences for fair consensus decision-making. In: ACM Conference on Fairness, Accountability, and Transparency (2024)
7. Chakraborty, D., Das, H., Dey, S., Yan, A.H.Y.: Improved rank aggregation under fairness constraint. In: International Joint Conference on Artificial Intelligence (2025)
8. Chakraborty, D., Das, S., Khan, A., Subramanian, A.: Fair rank aggregation. arXiv:2308.10499 (2023)
9. Chakraborty, D., Mazumdar, A., Saha, B., Yan, A.H.Y.: Fairness in aggregation: Optimal top-k and improved full ranking. In: International Conference on Machine Learning (2026)
10. Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank aggregation methods for the web. In: International Conference on World Wide Web (2001)
11. Israel, J., Brill, M.: Dynamic proportional rankings. Social Choice and Welfare **64**, 221–261 (2025)
12. Kliachkin, A., Psaroudaki, E., Mareček, J., Fotakis, D.: Fairness in ranking: Robustness through randomization without the protected attribute. In: IEEE International Conference on Data Engineering Workshops (2024)
13. Tijdeman, R.: The chairman assignment problem. Discrete Mathematics **32**(3), 323–330 (1980)
14. Wei, D., Islam, M.M., Schieber, B., Basu Roy, S.: Rank aggregation with proportionate fairness. In: ACM SIGMOD International Conference on Management of Data. pp. 262–275 (2022)