Improved Differentially Private Algorithms for Rank Aggregation

Quentin Hillebrand[*1], **Pasin Manurangsi**[*2], **Vorapong Suppakitpaisarn**[3], **Phanu Vajanopath**[*4]

[1]University of Copenhagen, Denmark
[2]Google Research, Thailand
[3]The University of Tokyo, Japan
[4]University of Wrocław, Poland

Abstract

Rank aggregation is a task of combining the rankings of items from multiple users into a single ranking that best represents the users' rankings. Alabi et al. (AAAI'22) presents differentially-private (DP) polynomial-time approximation schemes (PTASes) and 5-approximation algorithms with certain additive errors for the *Kemeny rank aggregation* problem in both central and local models. In this paper, we present improved DP PTASes with smaller additive error in the central model. Furthermore, we are first to study the *footrule rank aggregation* problem under DP. We give a near-optimal algorithm for this problem; as a corollary, this leads to 2-approximation algorithms with the same additive error as the 5-approximation algorithms of Alabi et al. for the Kemeny rank aggregation problem in both central and local models.

1 Introduction

The *rank aggregation* problem aims to combine n individual rankings over m candidates into a single consensus ranking that best reflects the input preferences. We consider two well-known optimality criteria for this task.

The first is the *Kemeny ranking* (Kemeny 1959). Given a collection of rankings $\{\pi_1, \ldots, \pi_n\}$, the *Kendall's tau distance* between a permutation ψ and π_i–denoted by $K(\psi, \pi_i)$–is defined[1] as the number of candidate pairs (j, k) for which the relative order in ψ disagrees with that in π_i. The average Kendall's tau distance of ψ is then $\frac{1}{n}\sum_{i=1}^{n} K(\psi, \pi_i)$. A permutation ψ minimizing this quantity is called the *Kemeny optimal ranking*, and the problem of finding the ranking is called the *Kemeny rank aggregation problem*.

The second criterion is the *footrule optimal ranking*, which minimizes the average position-wise discrepancy: $\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} |\psi(j) - \pi_i(j)|$, where $\pi_i(j)$ denotes the position of candidate j in the i-th input ranking. The problem of finding the ranking is the *footrule rank aggregation problem*.

While the *footrule rank aggregation problem* can be solved in polynomial time (Dwork et al. 2001), the *Kemeny rank aggregation problem* is NP-hard (Bartholdi, Tovey, and Trick 1989) even when the number of input rankings is as small as four (Dwork et al. 2001). Thus, approximation algorithms for this problem have been studied. Several 2-approximation algorithms have been proposed (Diaconis and Graham 1977; Dwork et al. 2001). Later, Ailon, Charikar, and Newman (2008) introduced algorithm called *KwikSort*, which also yields 2-approximation. With a slight adjustment to the KwikSort algorithm, they obtained an improved 11/7-approximation algorithm. Lastly, a polynomial-time approximation scheme (PTAS) for this problem was developed by Kenyon-Mathieu and Schudy (2007).

Often, the input rankings, e.g. votes in an election or preferences over search results, contain highly sensitive information about individuals. It is thus important that we perform rank aggregation while respecting their privacy.

In this work, we address both rank aggregation problems under the framework of differential privacy (DP). We recall the definition of DP below (Dwork et al. 2006b,a).

Definition 1.1 ((ε, δ)-DP)**.** Let $\varepsilon, \delta \geq 0$. A randomized mechanism $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ is (ε, δ)-DP if, for any *neighboring*[2] datasets $\mathbf{X}, \mathbf{X}' \in \mathcal{X}^n$ and any set of output $\mathcal{O} \subseteq \mathcal{Y}$, $\Pr[\mathcal{M}(\mathbf{X}) \in \mathcal{O}] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(\mathbf{X}') \in \mathcal{O}] + \delta$.

When $\delta = 0$, we say that the mechanism is ε-DP; we refer to this case as *pure-DP*, and the case $\delta > 0$ as *approx-DP*.

Since its introduction, DP has become the gold standard for privacy-preserving data analysis (Abowd 2018; Erlingsson, Pihur, and Korolova 2014), providing a mathematically rigorous means of quantifying privacy loss. We also consider a variant of DP known as *local DP* (LDP), in which each user independently perturbs their own data before transmission. This approach ensures privacy without requiring a trusted central server (Kasiviswanathan et al. 2011).

Rank aggregation under DP have been the subject of multiple recent studies (Hay, Elagina, and Miklau 2017; Yan, Li, and Liu 2020; Liu et al. 2020; Alabi et al. 2022; Xu, Sun, and Cheng 2023; Lan et al. 2024; Xu, Sun, and Cheng 2025). A central challenge in designing DP algorithms—including those for rank aggregation—is managing the *privacy-utility tradeoff*. In other words, the objective is to develop algorithms that, given a fixed privacy budget (ε, δ), achieve the highest possible utility. In the context of rank aggregation,

[*]These authors contributed equally.

[1]See Section 2.1 for formal definitions of rank aggregation.

[2]We say that two datasets $\mathbf{X} = (X_1, \ldots, X_n), \mathbf{X}' = (X'_1, \ldots, X'_n)$ are neighbors if they differ on a single coordinate, i.e. there exists $i^* \in [n]$ such that $X_i = X'_i$ for all $i \in [n] \setminus \{i^*\}$.

utility is typically measured by the Kendall's tau distance, with smaller distances indicating better utility. Prior work has addressed this challenge from both theoretical and empirical standpoints. We focus on the theoretical utility guarantee similar to (Hay, Elagina, and Miklau 2017; Alabi et al. 2022). Specifically, we say that a randomized algorithm is an (α, β)-approximation algorithm if the expected cost of its output is at most α times the optimum plus β.

1.1 Our Contributions

Footrule Ranking We present the *first DP algorithm* for the footrule rank aggregation problem (Section 4). Our algorithm achieves the optimal approximation ratio $\alpha = 1$, while the additive errors are:

- $\beta = \tilde{O}(m^3/\varepsilon n)$ for ε-DP,
- $\beta = \tilde{O}_\delta(m^{2.5}/\varepsilon n)$ for (ε, δ)-DP,
- $\beta = \tilde{O}(m^{2.5}/\varepsilon\sqrt{n})$ for ε-LDP.

Due to lower bounds from (Alabi et al. 2022), our additive errors for ε-DP and (ε, δ)-DP are nearly optimal (up to logarithmic factors).

Kemeny Ranking Our contributions for the Kemeny rank aggregation problem are summarized in Table 1.

The state-of-the-art work by Alabi et al. (2022) provides two (α, β)-approximation algorithms for the Kemeny rank aggregation problem under (ε, δ)-differential privacy. The first algorithm is a PTAS, which achieves a small multiplicative factor $\alpha = 1 + \xi$ for $\xi > 0$ but incurs a large additive error $\beta = \tilde{O}_{\xi,\delta}(m^3)/(\varepsilon n)$. In contrast, the second algorithm offers a smaller additive term $\beta = \tilde{O}_\delta(m^{2.5})/(\varepsilon n)$, at the cost of a larger multiplicative factor $\alpha = 5 + \xi$. It is demonstrated in the paper that there is no algorithm with additive term $\beta = o(m^{2.5})/(\varepsilon n)$.

We aim to improve upon these trade-offs by designing an algorithm that retains the small α of the first while reducing the additive error β, or alternatively, one that preserves the small β of the second while lowering the multiplicative factor α.

Our first algorithm, presented as Algorithm 2 in Section 4, improves the multiplicative factor α of the second algorithm in (Alabi et al. 2022) from $5 + \xi$ to 2, while preserving the same additive error β. Moreover, this approach extends naturally to the ε-DP and ε-LDP settings, achieving the same reduced multiplicative factor $\alpha = 2$ without increasing the additive error β. For LDP, our algorithm also has the advantage of being *non-interactive* whereas Alabi et al.'s algorithm is *interactive* and requires $O(\log m)$ rounds of communication.

Our second algorithm, presented as Algorithms 3-4 in Section 5, improves the additive error β of the first algorithm in (Alabi et al. 2022) from $\frac{1}{\varepsilon n}\tilde{O}_{\varepsilon,\delta}(m^3)$ to $\frac{1}{\varepsilon n}\tilde{O}_{\varepsilon,\delta}(m^{65/22})$. As it has been shown that no algorithm can achieve an additive term of order $o(m^{2.5})/(\varepsilon n)$ (Alabi et al. 2022), we believe that reducing the exponent of m to a value strictly less than 3 represents a meaningful advancement toward closing the gap between the known upper and lower bounds. We also achieve a similar improvement for ε-DP.

It is worth noting that a $(1, \tilde{O}(m^3/\varepsilon n))$-approximation algorithm can be obtained using the exponential mechanism. However, as observed in (Hay, Elagina, and Miklau 2017), this approach is not computationally efficient, as it does not run in polynomial time.

1.2 Technical Overview

We provide high-level technical overview of our algorithms.

Algorithms for Footrule Ranking Our algorithm for the footrule ranking builds upon the non-private approach of (Dwork et al. 2001). For each pair $q, j \in [m]$, let the cost of assigning candidate q to position j be $\gamma_q^j = \frac{1}{n}\sum_{i=1}^n |\pi_i(q) - j|$. Given a permutation ψ, its total cost is $\sum_{q\in[m]} \gamma_q^{\psi(q)}$. The objective is thus to find ψ that minimizes this cost. This corresponds to a minimum weight bipartite matching problem, which can be solved efficiently.

To construct a DP version of the algorithm, we propose a method to accurately release the weights γ_q^j for all q and j privately. If we can publish each value of γ_q^j with additive error of β, since there are m values of γ_q^j in the cost calculation, we obtain the additive term of $m \cdot \beta$ in the cost.

In a naive approach, each value $\pi_i(q)$ contributes to up to m output values (i.e. γ_q^j for all $j \in [m]$), causing the privacy budget to be split across many computations. This leads to large noise and significant error. To mitigate this, we adapt the binary tree mechanism (Dwork et al. 2010; Chan, Shi, and Song 2011) to better manage the privacy budget.

The original binary tree mechanism is designed to answer multiple range queries, specifically the frequencies of data within intervals $[low, up]$. To achieve this, we define a set of intervals $\mathcal{I} = \{[(p-1)\cdot 2^\ell + 1, p\cdot 2^\ell] : \ell \in [\lceil \log m \rceil], p \in [\lceil m/2^\ell \rceil]\}$. We privately release the frequency of data falling within each interval in $\mathcal{I}$. Since any query range $[low, up]$ can be decomposed into a disjoint union of intervals from $\mathcal{I}$, we can answer the query by aggregating the corresponding private frequencies. In this mechanism, each data point is used in only $O(\log m)$ intervals, which is significantly fewer than in the naive approach. As a result, the added noise is smaller, leading to a reduced additive error.

Let $r(I)$ denote the smallest value in the interval I. To compute γ_q^j for all q and j, we privately release, for each interval $I \in \mathcal{I}$, the values $\frac{1}{n}\sum_{i:\pi_i(q)\in I}(\pi_i(q) - r(I))$ and $|\{i : \pi_i(q) \in I\}|$. These allow us to calculate $\frac{1}{n}\sum_{i:\pi_i(q)\in I}|\pi_i(q) - j|$ when $j \notin I$. Since there exists a collection of disjoint intervals in $\mathcal{I}$ whose union equals $[m]\setminus\{j\}$, summing over these intervals yields $\frac{1}{n}\sum_{i=1}^n |\pi_i(q) - j|$. Similar to the original binary tree mechanism, each data point appears in only $O(\log m)$ intervals instead of m, resulting in a smaller additive error in our mechanism.

$(2, \beta)$-approximation algorithm for the Kemeny rank aggregation Since the Spearman's footrule distance is at least the Kendall's tau distance and at most twice that distance, any $(1, \beta)$-approximation algorithm for the footrule rank aggregation, as described in the previous paragraph, also serves as a $(2, \beta)$-approximation algorithm for the Kemeny rank aggregation problem.

	α	β	
(ε, δ)-DP	$1+\xi$	$\frac{1}{\varepsilon n}\tilde{O}_{\xi,\delta}(m^3)$	(Alabi et al. 2022)
	$5+\xi$	$\frac{1}{\varepsilon n}\tilde{O}_{\xi,\delta}(m^{2.5})$	(Alabi et al. 2022)
	any $\alpha \geq 1$	$\frac{1}{\varepsilon n}\Omega(m^{2.5})$	(Alabi et al. 2022) [Lower Bound]
	$1+\xi$	$\frac{1}{\varepsilon n}\tilde{O}_{\xi,\delta}(m^{65/22})$	**Our Work [Section 5]**
	2	$\frac{1}{\varepsilon n}\tilde{O}_{\delta}(m^{2.5})$	**Our Work [Section 4]**
ε-DP	1	$\frac{1}{\varepsilon n}\tilde{O}(m^3)$	(Hay, Elagina, and Miklau 2017)
	$1+\xi$	$\frac{1}{\varepsilon n}\tilde{O}_{\xi}(m^4)$	(Alabi et al. 2022)
	$5+\xi$	$\frac{1}{\varepsilon n}\tilde{O}_{\xi}(m^3)$	(Alabi et al. 2022)
	any $\alpha \geq 1$	$\frac{1}{\varepsilon n}\Omega(m^3)$	(Alabi et al. 2022) [Lower Bound]
	$1+\xi$	$\frac{1}{\varepsilon n}\tilde{O}_{\xi}(m^{27/7})$	**Our Work [Section 5]**
	2	$\frac{1}{\varepsilon n}\tilde{O}(m^3)$	**Our Work [Section 4]**
ε-LDP	$1+\xi$	$\frac{1}{\varepsilon\sqrt{n}}\tilde{O}_{\xi}(m^3)$	(Alabi et al. 2022)
	$5+\xi$	$\frac{1}{\varepsilon\sqrt{n}}\tilde{O}_{\xi}(m^{2.5})$	(Alabi et al. 2022)
	2	$\frac{1}{\varepsilon\sqrt{n}}\tilde{O}(m^{2.5})$	**Our Work [Section 4]**

Table 1: Comparison of our (α, β)-approximation algorithms for Kemeny Rank Aggregation with previous works. All algorithms run in $(nm)^{O(1)}$ time, except the exponential mechanism (Hay, Elagina, and Miklau 2017) which runs in $O(n \cdot m!)$ time.

$(1+\xi, \beta)$-approximation algorithm for the Kemeny rank aggregation Our improved PTAS requires separate treatments of the cases where n is "small" and where n is "large". For exposition purpose, we focus on (ε, δ)-DP and it is best to think of n as being large when it is slightly larger than m, i.e. $n = \Theta_{\varepsilon,\delta}(m \log m)$. Note that, although this is the regime relevant to most practical ranking applications—where the number of voters n is typically much larger than the number of candidates m—the error guaranteed by the PTAS in (Alabi et al. 2022) is only $\tilde{O}_{\varepsilon,\delta}(m^3)$, which is barely nontrivial, since any ranking incurs error at most m^3.

Large n: Leveraging Unbiasedness Define the matrix $\mathbf{w}^{\Pi} \in [0,1]^{m \times m}$ by $w^{\Pi}_{uv} := \frac{1}{n}\sum_{i \in [n]} \mathbf{1}[\pi_i(u) < \pi_i(v)]$. We start from the following observation: we can privatize $\mathbf{w}^{\Pi}$ by adding independent Gaussian noise $\mathcal{N}(0, \sigma^2)$ where $\sigma = O\left(\frac{m \log(1/\delta)}{\varepsilon n}\right)$ to each of its entry. Let us call the privatized matrix $\tilde{\mathbf{w}}$. Indeed, it is simple to see that, if we look at any permutation π, the difference in its cost between $\mathbf{w}^{\Pi}$ and $\tilde{\mathbf{w}}$–which is a summation of $O(m^2)$ of the Gaussian noise–has variance only $O_{\varepsilon,\delta}(m^2)$. Since there are only $m!$ rankings in total, a union bound (and concentration of Gaussian) then tells us that w.h.p. the cost difference for *every* ranking is at most $O_{\varepsilon,\delta}(m^{2.5})$. So if we run the non-private PTAS (Kenyon-Mathieu and Schudy 2007) on $\tilde{\mathbf{w}}$, then we should already improve upon (Alabi et al. 2022), right? Although ostensibly correct, there is one flaw in this argument: $\tilde{\mathbf{w}}$ can have negative entries, for which the non-private PTAS cannot handle. Indeed, if we proceed by "clipping" $\tilde{\mathbf{w}}$ to ensure non-negativity, then the bias would make the error become $O_{\varepsilon,\delta}(m^3)$, so unfortunately we obtain no improvement over (Alabi et al. 2022).

To overcome this issue, our main observation is that, to get a good approximation for $\mathbf{w}^{\Pi}$, it suffices to get a good approximation for the instance where, for some u, v such that $w_{uv} < w_{vu}$, we replace w_{uv} with zero and replace w_{vu} with $w_{vu} - w_{uv}$. Using this observation, by adding the noise only to the latter, we can ensure non-negativity with high probability as long as $\sigma = O\left(\frac{1}{\log m}\right)$. With some additional technical work, this solves the bias issue. However, the requirement on σ means that n has to be at least $\Omega_{\varepsilon,\delta}(m \log m)$. This indeed necessitates another algorithm when n is small.

Small n: Reduction to 2-Way Marginal. In the case of small n, our intuition starts from the following: Suppose instead of each π_i being a permutation $[m] \to [m]$, it is a mapping $[m] \to \{0,1\}$. Then, the term $\mathbf{1}[\pi_i(u) < \pi_i(v)]$ is exactly equal to $\mathbf{1}[\pi_i(u) = 0, \pi_i(v) = 1]$. Thus, each entry of $\mathbf{w}^{\Pi}$ is simply a *2-way marginal* query, which is well studied in DP literature. In particular, there is an efficient algorithm that can answer such queries with error $O_{\varepsilon,\delta}(m^{1/4}/\sqrt{n})$ (Dwork, Nikolov, and Talwar 2015) per query. If we were to achieve this error for estimating $\mathbf{w}^{\Pi}$, then we would already achieve improvement over (Alabi et al. 2022) for $n = O_{\varepsilon,\delta}(m \log m)$.

The main challenge is now to extend the above simplified setting $\pi_i : [m] \to \{0,1\}$ to the desired setting $\pi_i : [m] \to [m]$. To do this, we apply *bucketing*. Roughly speaking, we pick a number of buckets B, divide the range $[m]$ into B buckets of equal size and separate the term $\mathbf{1}[\pi_i(u) < \pi_i(v)]$ into two parts, based on whether $\pi_i(u), \pi_i(v)$ are from the same bucket. If they are from the same bucket, we use the Gaussian mechanism; since there are only m^2/B such pairs, we can add smaller noise than without bucketing. If they are from different buckets, then we use a generalization of the 2-way marginal reduction described in the previous paragraph. By picking B appropriately, we can optimize the error which

ends up being $\tilde{O}_{\varepsilon,\delta}(m^{43/22})$ when $n = \tilde{\Theta}_{\varepsilon,\delta}(m)$.

2 Preliminaries

2.1 Rank Aggregation

For any $m \in \mathbb{N}$, we use $[m]$ to denote $\{1, \dots, m\}$ and let $\mathbf{S}_m$ denote the set of all permutations (i.e., all rankings) on $[m]$. In the rank aggregation problem, we are given a dataset Π of n rankings $(\pi_1, \dots, \pi_n) \in (\mathbf{S}_m)^n$, the goal is to output a ranking $\psi \in \mathbf{S}_m$ that minimizes $d(\psi, \Pi) := \frac{1}{n}\sum_{i\in[n]} d(\psi, \pi_i)$ where $d : \mathbf{S}_m \times \mathbf{S}_m \to \mathbb{R}_{\geq 0}$ is a certain distance. We will discuss two distances here:

- Spearman's footrule distance: $F(\psi, \pi) := \sum_{j\in[m]} |\psi(j) - \pi(j)|$.
- Kendall's tau distance: $K(\psi, \pi) := |\{(j, j') \in [m] \times [m] \mid \pi(j) < \pi(j') \wedge \psi(j) > \psi(j')\}|$.

An algorithm is *efficient* if it runs in $(nm)^{O(1)}$ time.

2.2 Differential Privacy

We use the DP notion as defined in Definition 1.1. Note that in rank aggregation problems, we have $\mathcal{X} = \mathcal{Y} = \mathbf{S}_m$, and two datasets $\Pi, \Pi' \in (\mathbf{S}_m)^n$ are neighbors iff they differ on a single ranking. We also consider the non-interactive local model, defined below.

Definition 2.1 (ε-LDP)**.** An algorithm in the (non-interactive) local model consists of a randomizer $\mathcal{R} : \mathcal{X} \to \mathcal{Z}$ and an analyst $\mathcal{A} : \mathcal{Z}^n \to \mathcal{Y}$; with input dataset $\mathbf{X} = (x_1, \dots, x_n)$, the final output is computed as $\mathcal{A}(\mathcal{R}(x_1), \dots, \mathcal{R}(x_n))$. The algorithm (or the randomizer) is said to be ε-LDP iff, for any $x, x' \in \mathcal{X}$ and $\mathcal{O} \subseteq \mathcal{Z}$,

$$\Pr[\mathcal{R}(x) \in \mathcal{O}] \leq e^{\varepsilon} \cdot \Pr[\mathcal{R}(x') \in \mathcal{O}].$$

Note that these DP notions are the same as the ones used in (Hay, Elagina, and Miklau 2017; Alabi et al. 2022).

For simplicity of presentation, we will assume throughout that $\varepsilon \leq 1, \delta \in (0, 1/2)$, and we will not state this explicitly.

3 DP Approximate Median

In this section, we study the Approximate Median (APXMED) problem, which is closely related to the footrule ranking objective. The universe is $\mathcal{X} = [m]$, and the input is $\mathbf{x} = (x_1, \dots, x_n) \in [m]^n$. For each $j \in [m]$, define $\gamma_j(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} |x_i - j|$. The goal is to output estimates $(\tilde{\gamma}_1, \dots, \tilde{\gamma}_m)$. We measure the quality of these estimates by the ℓ_∞-error $\max_{j\in[m]} |\tilde{\gamma}_j - \gamma_j(\mathbf{x})|$. We say that an algorithm is β-accurate if the expected ℓ_∞-error of its output is at most β.

We refer to this task as Approximate Median because the median j^* is a minimizer to $\gamma_{j^*}(\mathbf{x})$ and, thus, answering the queries $(\gamma_j(\mathbf{x}))_{j\in[m]}$ allows us to determine how "far" each j is from the median. While other approximate notions of median have been studied under DP (cf. (Gillenwater, Joseph, and Kulesza 2021)), to the best of our knowledge, APXMED has not been considered and may be of independent interest beyond the context of rank aggregation, e.g. in DP clustering (Ghazi, Kumar, and Manurangsi 2020).

Theorem 3.1. *There is an efficient algorithm for* APXMED *that is β-accurate with*

- $\beta = O\left(\frac{m \log m}{\varepsilon n}\right)$ *for ε-DP,*
- $\beta = O\left(\frac{m\sqrt{\log m \log(1/\delta)}}{\varepsilon n}\right)$ *for (ε, δ)-DP,*
- $\beta = O\left(\frac{m\sqrt{\log m}}{\varepsilon\sqrt{n}}\right)$ *for ε-LDP.*

Modified Binary Tree Mechanism We present an algorithm for answering approximate median queries by adapting the well-known *Binary Tree Mechanism* (Dwork et al. 2010; Chan, Shi, and Song 2011), originally designed for range queries. We assume wlog[3] that the domain size m is a power of two, i.e., $m = 2^d$ for some $d \in \mathbb{N}$.

We define a complete binary tree with the set of nodes $\mathcal{B}$; each node $t \in \mathcal{B}$ is associated with a subset of the domain, denoted by $I(t) \subseteq [m]$. Each leaf corresponds to a singleton set. The tree has $d + 1$ levels, indexed by $\ell \in \{0, \dots, d\}$ (where leaves are at level 0). For any internal node t at level ℓ, the set $I(t)$ consists of the union of $I(t')$ for all leaves t' in its subtree and is of the form $I(t) = [(p-1)\cdot 2^\ell + 1, p\cdot 2^\ell]$ for some $p \in [m/2^\ell]$. A node t in the tree is called a *left node* if all elements of $I(t)$ are smaller than those of its sibling. The smallest value in $I(t)$ is denoted by $r(t)$ and the level of the node t is defined by $\ell(t)$.

Our algorithm is presented in Algorithm 1. In the original binary tree mechanism (Dwork et al. 2010; Chan, Shi, and Song 2011), each node t stores the number of the data points that fall within its associated interval $I(t)$. In our algorithm, to support the APXMED computation, each node t maintains $v_t^{\text{agg}} = \frac{1}{n}\sum_{x_i\in I(t)} |x_i - r(t)|$ and $u_t^{\text{agg}} = \frac{1}{n}\sum_{x_i\in I(t)} 2^{\ell(t)}$.

For APXMED, we consider all sibling nodes of the nodes whose intervals contain the candidate value j. These sibling intervals are disjoint and their union is equal to $[m] \setminus \{j\}$. Therefore, Lines 8–13 of Algorithm 1 compute the contribution of all data points $x_i \neq j$ exactly once. This allows us to estimate the quantity $\frac{1}{n}\sum_i |x_i - j|$ by the expression $\frac{1}{n}\sum_{t'\in\mathcal{B}'_j}\sum_{x_i\in I(t')} |x_i - j|$, where $\mathcal{B}'_j$ denotes the set of relevant sibling nodes. The sum $\sum_{x_i\in I(t')} |x_i - j|$ is estimated by the computation at Line 13 of the algorithm.

Note that we have to compute $v_t^{\text{agg}}, u_t^{\text{agg}}$ while respecting the privacy constraints. To do this, we write out these as summands $v_t^{\text{agg}} = \frac{1}{n}\sum_{i\in[n]} \mathbf{1}[x_i \in I(t)] \cdot (x_i - r(t))$ and $u_t^{\text{agg}} = \frac{1}{n}\sum_{i\in[n]} \mathbf{1}[x_i \in I(t)] \cdot 2^{\ell(t)}$. Note that each inner term depends only on x_i. This means that $v^{\text{agg}}, u^{\text{agg}}$ are simply the average of n vectors, where each vector depends only on a single x_i. Thus, we may employ known *vector aggregation algorithms* in ε-DP, (ε, δ)-DP and ε-LDP for this task; indeed, this is the only difference between the three settings. Specifically, we use the Laplace mechanism, the Gaussian mechanism, and an algorithm of (Duchi, Jordan, and Wainwright 2014), respectively. Finally, to optimize the error further, we employ a weighting strategy where we multiply the contributions to $v_t^{\text{agg}}, u_t^{\text{agg}}$ by $\kappa^{d-\ell(t)}$ before aggregation (Lines 2-3), and multiplying $\kappa^{\ell(t)-d}$ back to the es-

[3] If not, we may increase m to the next power of two.

timates (Lines 6-7). Here κ can be any constant between 1 and 2. This helps reduce the error by a logarithmic factor.

Algorithm 1: Modified binary tree mechanism for APXMED

Input: Dataset $\mathbf{x} = (x_1, \ldots, x_n) \in [m]^n$
Output: Vector $(\tilde{\gamma}_j)_{j\in[m]}$
Parameters: DP vector aggregation algorithm VECAGG, weight growth factor $\kappa \in (1, 2)$

1: **for all** $t \in \mathcal{B}, i \in [n]$ **do** {*Weighted Contribution*}
2: $v_{i,t}^{\text{WEI}} \leftarrow \kappa^{d-\ell(t)} \cdot \mathbf{1}[x_i \in I(t)] \cdot (x_i - r(t))$
3: $u_{i,t}^{\text{WEI}} \leftarrow \kappa^{d-\ell(t)} \cdot \mathbf{1}[x_i \in I(t)] \cdot 2^{\ell(t)}$
4: Use VECAGG to aggregate vectors v_i^{WEI} and u_i^{WEI} across all $i \in [n]$; let $\tilde{v}^{\text{agg,WEI}}$ and $\tilde{u}^{\text{agg,WEI}}$ be the result.
5: **for all** $t \in \mathcal{B}$ **do** {*Reweight Aggregate Vector*}
6: $\tilde{v}_t^{\text{agg}} \leftarrow \kappa^{\ell(t)-d} \cdot \tilde{v}_t^{\text{agg,WEI}}$
7: $\tilde{u}_t^{\text{agg}} \leftarrow \kappa^{\ell(t)-d} \cdot \tilde{u}_t^{\text{agg,WEI}}$
8: **for all** $j \in [m]$ **do** {*Compute Estimates*}
9: $\mathcal{B}_j \leftarrow$ the set of non-root nodes t such that $j \in I(t)$.
10: **for all** $t \in \mathcal{B}_j$ **do**
11: $t' \leftarrow$ sibling of t
12: $s_{j,t} \leftarrow \begin{cases} 1 & \text{if } t \text{ is a left node} \\ -1 & \text{otherwise.} \end{cases}$
13: $\tilde{\gamma}_{j,t} \leftarrow s_{j,t} \cdot \left(\tilde{v}_{t'}^{\text{agg}} + \frac{r(t')-j}{2^{\ell(t')}} \cdot \tilde{u}_{t'}^{\text{agg}}\right)$.
14: $\tilde{\gamma}_j \leftarrow \sum_{t\in\mathcal{B}_j} \tilde{\gamma}_{j,t}$
15: **return** $(\tilde{\gamma}_j)_{j\in[m]}$

Parallel Approximate Median. Now, we consider an m-*parallel* version of APXMED, which will be convenient for the next application. In this variant, the domain $\mathcal{X}$ is now $[m]^m$; i.e. the i-th input is now $x_i = (x_{i,1}, \ldots, x_{i,m})$. The goal is to compute approximate median for $x_{1,q}, \ldots, x_{n,q}$ for all $q \in [m]$. That is, for all $j \in [m], q \in [m]$, we wish to estimate $\gamma_{j,q}(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^n |x_{i,q} - j|$. Similar to before, the ℓ_∞-error of the estimates $(\tilde{\gamma}_{j,q})_{j,q\in[m]}$ is defined as $\max_{j,q\in[m]} |\tilde{\gamma}_{j,q} - \gamma_{j,q}(\mathbf{x})|$. An algorithm is β-accurate if the expected ℓ_∞-error of its output is at most β.

The following is a simple corollary of Theorem 3.1:

Corollary 3.2. *There is an efficient algorithm for m-parallel* APXMED *that is β-accurate with*

- $\beta = O\left(\frac{m^2 \log m}{\varepsilon n}\right)$ *for ε-DP,*
- $\beta = O\left(\frac{m^{1.5}\sqrt{\log m \log(1/\delta)}}{\varepsilon n}\right)$ *for (ε, δ)-DP,*
- $\beta = O\left(\frac{m^{1.5}\sqrt{\log m}}{\varepsilon\sqrt{n}}\right)$ *for ε-LDP.*

4 From APXMED to Rank Aggregation

We next provide a reduction from APXMED to footrule rank aggregation. We use the matching-based non-private algorithm of (Dwork et al. 2001) but with the weights computed based on our APXMED algorithm, as shown in Algorithm 2.

We can show that the additive error of Algorithm 2 is $O(m)$ times the ℓ_∞-error of the estimates for m-parallel APXMED. Thus, from Corollary 3.2, we immediately have

Algorithm 2: Footrule Rank Aggregation from APXMED

Input: Dataset $\Pi = (\pi_1, \ldots, \pi_n) \in (\mathbf{S}_m)^n$
Output: Aggregated rank $\psi \in \mathbf{S}_m$

1: $(\tilde{\gamma}_{j,q})_{j,q\in[m]} \leftarrow$ output from running m-*parallel* APXMED algorithm in Corollary 3.2 on input Π.
2: $G \leftarrow$ weighted complete bipartite graph where both the left and right vertex sets are $[m]$, and the weight of each edge (q, j) is $\tilde{\gamma}_{j,q}$.
3: **return** Minimum-weight bipartite matching of G. (i.e., if edge (a, b) belongs to the matching, let $\psi(a) = b$.)

Theorem 4.1. *There is an efficient $(1, \beta)$-approximation algorithm for footrule rank aggregation with*

- $\beta = O\left(\frac{m^3 \log m}{\varepsilon n}\right)$ *for ε-DP,*
- $\beta = O\left(\frac{m^{2.5}\sqrt{\log m \log(1/\delta)}}{\varepsilon n}\right)$ *for (ε, δ)-DP,*
- $\beta = O\left(\frac{m^{2.5}\sqrt{\log m}}{\varepsilon\sqrt{n}}\right)$ *for ε-LDP.*

Since the Spearman's footrule distance is always at least the Kendall's tau distance and at most twice that distance, we also get the following as a corollary:

Theorem 4.2. *There is an efficient $(2, \beta)$-approximation algorithm for Kemeny rank aggregation with*

- $\beta = O\left(\frac{m^3 \log m}{\varepsilon n}\right)$ *for ε-DP,*
- $\beta = O\left(\frac{m^{2.5}\sqrt{\log m \log(1/\delta)}}{\varepsilon n}\right)$ *for (ε, δ)-DP,*
- $\beta = O\left(\frac{m^{2.5}\sqrt{\log m}}{\varepsilon\sqrt{n}}\right)$ *for ε-LDP.*

As mentioned earlier, the additive errors β in Theorems 4.1 and 4.2 for ε-DP and (ε, δ)-DP are optimal due to the lower bound of (Alabi et al. 2022).

5 PTAS for Kemeny Rank Aggregation

In this section, we present our PTAS for Kemeny rank aggregation. Recall that Alabi et al. (2022) gave PTASes for the problem with additive errors $\beta = \tilde{O}_\xi\left(\frac{m^4}{\varepsilon n}\right)$ and $\tilde{O}_{\xi,\delta}\left(\frac{m^3}{\varepsilon n}\right)$ for ε-DP and (ε, δ)-DP, respectively. Our algorithm improves on these additive errors, as stated below.

Theorem 5.1. *For every constant $\xi > 0$, there is an efficient $(1 + \xi, \beta)$-approximation algorithm for Kemeny rank aggregation with*

- $\beta = \tilde{O}_\xi\left(\frac{m^{27/7}}{\varepsilon n}\right)$ *for ε-DP,*
- $\beta = \tilde{O}_\xi\left(\frac{m^{65/22}\sqrt{\log(1/\delta)}}{\varepsilon n}\right)$ *for (ε, δ)-DP,*

5.1 Additional Preliminaries

Our PTAS will require additional tools and preliminaries, which we list below.

For an instance $\Pi = (\pi_1, \ldots, \pi_n)$ of rank aggregation, we define the matrix $\mathbf{w}^\Pi \in [0, 1]^{m\times m}$ by

$$w_{uv}^\Pi := \frac{1}{n}\sum_{i\in[n]} \mathbf{1}[\pi_i(u) < \pi_i(v)]. \tag{1}$$

One important observation to make is that the desired objective, $K(\psi, \Pi)$, can be written in terms of $\mathbf{w}^\Pi$ as $K(\psi,\Pi) = \sum_{\substack{u,v\in[n]\\ \psi(u)<\psi(v)}} w^\Pi_{vu}$.

Weighted Feedback Arc Set. For our PTAS, we need to consider a more general problem, called the *Weighted Feedback Arc Set (WFAS)* problem.

Definition 5.2 (Weighted Feedback Arc Set (WFAS))**.** The input is a set of candidates $[m]$ and a weight matrix $\mathbf{w} \in \mathbb{R}^{m\times m}$. The goal is to output π that minimizes $\mathrm{cost}_{\mathbf{w}}(\pi) := \sum_{\substack{u,v\in[n]\\ \pi(u)<\pi(v)}} w_{vu}$.

Note that the Kemeny rank aggregation problem is a special case of WFAS where $\mathbf{w} = \mathbf{w}^\Pi$ is as defined in (1). We remark that we deliberately allow the weights to be negative and unbounded as this will be useful later on. Nevertheless, we also need a boundedness definition here:

Definition 5.3 (Bounded WFAS Instance)**.** An instance $\mathbf{w}$ is said to be *bounded* if $w_{uv} \geq 0$ and $w_{uv} + w_{vu} \in \left[\frac{1}{2}, 2\right]$.

While WFAS is hard to approximate on general instances (Guruswami et al. 2011; Kenyon-Mathieu and Schudy 2007) gave a PTAS for bounded instances[4]:

Theorem 5.4 (Kenyon-Mathieu and Schudy 2007)**.** *For every constant $\xi > 0$, there exists an efficient $(1+\xi)$-approximation algorithm for WFAS on bounded instances.*

We also state another simple but important lemma below, which states that, if we give a good approximation algorithm on one weight matrix $\tilde{\mathbf{w}}$, it remains a good approximation on a "nearby" weight matrix $\mathbf{w}$.

Lemma 5.5 (Alabi et al. 2022, Theorem 1)**.** *For any $\tilde{\mathbf{w}}, \mathbf{w} \in [0,1]^{m\times m}$ such that $\|\tilde{\mathbf{w}} - \mathbf{w}\|_1 \leq e$, any $(1+\xi)$-approximate solution to WFAS on $\tilde{\mathbf{w}}$ is also an $(1+\xi, O(e))$-approximate solution to WFAS on $\mathbf{w}$.*

Two-Way Marginals. Finally, we recall the two-way marginal problem which is defined as follows:

Definition 5.6 (Two-Way Marginal Queries)**.** The universe $\mathcal{X}$ here is $\{0,1\}^d$, the set of all binary vectors of length d. For notational convenience, we view $x_i \in \{0,1\}^d$ as a function $x_i : [d] \to \{0,1\}$ instead. A two-way marginal query[5] is indexed by a size-2 set $S = \{j_1, j_2\} \subseteq [d]$. On input dataset $\mathbf{X} = (x_1, \ldots, x_n)$, the query has value $q_S(\mathbf{X}) = \frac{1}{n}\sum_{i\in[n]} \mathbf{1}[x_i(j_1) = 1 \wedge x_i(j_2) = 1]$. The goal of an algorithm for two-way marginal is to output estimates $(\tilde{q}_S)_{S\in\binom{[d]}{2}}$ for the above queries.

The two-way marginal problem is well-studied in DP literature (e.g. (Barak et al. 2007; Bun, Ullman, and Vadhan 2014; Dwork, Nikolov, and Talwar 2015; Nikolov 2024)). A crucial aspect we will use here is that when n is small, there are efficient ε-DP and (ε,δ)-DP algorithms (Nikolov 2024; Dwork, Nikolov, and Talwar 2015) that achieve significantly smaller errors compared to standard noise addition algorithms, i.e. the Laplace or Gaussian mechanisms.

As discussed in Section 1.2, our algorithm requires us to handle two cases based on whether n is "small" or "large".

5.2 Small n: Two-Way Marginal

Lemma 5.5 gives us a fairly clear overall strategy towards obtaining good approximation algorithms for Kemeny rank aggregation: Design a differentially private algorithm that publishes the weight matrix $\mathbf{w}^\Pi$ with small error. Indeed, this was the same strategy deployed in (Alabi et al. 2022), who use Laplace and Gaussian mechanisms to add noise to $\mathbf{w}^\Pi$. To improve upon this in the small n regime, we will instead employ DP two-way marginal algorithms.

Let $B \in \mathbb{N}$ be a parameter to be specified. We partition $[m]$ into B *buckets* (indexed by $1,\ldots,B$), where each bucket is an interval of size $\leq \lceil m/B \rceil$. For every $i \in [m]$, let $\iota(i)$ denote the index of the bucket it belongs to.

The matrix $\mathbf{w}^\Pi$ can now be decomposed into the sum of two parts: (i) comparison within the same bucket $\mathbf{s}$, and (ii) comparison across buckets $\mathbf{t}$. More formally, we define the matrices $\mathbf{s}, \mathbf{t} \in \mathbb{R}^{m\times m}$ as follows:

$$s_{uv} = \frac{1}{n}\sum_{i\in[n]} \mathbf{1}[\iota(\pi_i(u)) = \iota(\pi_i(v)) \wedge \pi_i(u) < \pi_i(v)],$$

$$t_{uv} = \frac{1}{n}\sum_{i\in[n]} \mathbf{1}[\iota(\pi_i(u)) < \iota(\pi_i(v))].$$

It is simple to see that $\mathbf{w}^\Pi = \mathbf{s} + \mathbf{t}$. Thus, it suffices to give DP algorithms for approximately computing $\mathbf{s}, \mathbf{t}$. For computing the estimate of $\mathbf{s}$, we can simply use Laplace and vanilla Gaussian mechanism, because the sensitivity is now reduced as we only compare candidates in the same bucket.

For computing the estimate of $\mathbf{t}$, we encode t_{uv} as a two-way marginal query as follows. First, let $d = m\cdot B$ where we associate $[d]$ naturally with $[m]\times[B]$. Then, for each $i \in [n]$, we let $x_i \in \{0,1\}^d$ be defined by $x_i(u,b) = \mathbf{1}[\iota(\pi_i(u)) = b]$ for all $u \in [m], b \in [B]$. The crucial observation here is that $\mathbf{t}$ can now be written in terms of marginal queries: $t_{uv} = \sum_{\substack{b_u,b_v\in[B]\\ b_u<b_v}} q_{\{(u,b_u),(v,b_v)\}}(\mathbf{X})$. As such, we can use the aforementioned DP two-way marginal query algorithms to estimate $\mathbf{t}$. Our algorithm is shown in Algorithm 3. The choice of distribution $\mathcal{D}$ and subroutine TwoMarAlg depends on whether we aim for ε-DP or (ε,δ)-DP. The operation clip, which ensures that the outputs lie within the interval $[0,1]$. It is simple to analyze the error of Algorithm 3. By optimizing the parameter B to minimize the error and invoking Lemma 5.5, we arrive at our PTAS for small n.

5.3 Large n: Leveraging Unbiasedness

As mentioned in Section 1.2, if we were able to use the noised weights without any clipping, then we would have been done. Unfortunately, this is not possible since the weights could become negative due to the noises. It turns out that, when n is large, we can handle this as follows.

Consider each pair $u, v \in [m]$. First, consider the "balanced" case where w_{uv} and w_{vu} are both not too small, say

[4] In fact, the algorithm of (Kenyon-Mathieu and Schudy 2007) allow the numbers 1/2, 2 in the boundedness assumption to be changed to any constants. However, we do not use this in our work.

[5] Here we restrict two-way marginal queries only to those with attributes equal to "1", which is sufficient for our setting.

Algorithm 3: DP Approximation of $\mathbf{w}$ for Small n

Input: Dataset $\Pi = (\pi_1, \ldots, \pi_n) \in (\mathbf{S}_m)^n$
Output: Estimate $\tilde{\mathbf{w}}^{\Pi}$ of $\mathbf{w}^{\Pi}$
Parameters: Distribution $\mathcal{D}$, # of buckets B, DP two-way marginal algo TWOMARALG

for $u, v \in [m]$ **do** {*Noising* s_{uv}}
  $\tilde{s}_{uv} \leftarrow s_{uv} + r_{uv}$ where $r_{uv} \sim \mathcal{D}$
for $i \in [n]$ **do**
  $x_i \in \{0,1\}^{mB}$ be s.t. $x_i(u,b) = \mathbf{1}[\iota(\pi_i(u)) = b]$
$(\tilde{q}_S)_{S \in \binom{[d]}{2}} \leftarrow$ output of TWOMARALG$(x_1, \ldots, x_n)$
for $u, v \in [m]$ **do** {*Estimate* t_{uv} *from marginals*}
  $\tilde{t}_{uv} = \sum_{\substack{b_u, b_v \in [B] \\ b_u < b_v}} \tilde{q}_{\{(u,b_u),(v,b_v)\}}$
return $\mathrm{clip}(\tilde{s}_{uv} + \tilde{t}_{uv})_{u,v \in [m]}$

$w_{vu}, w_{vu} \in [\frac{1}{6}, \frac{5}{6}]$. In this case, when n is sufficiently large, the noise added has such a small variance that with high probability all the values w_{uv}, w_{vu} remain non-negative after noise addition. Hence, we can keep this case as is.

Next, consider the "imbalanced" case where either w_{uv} or w_{vu} is smaller than $1/6$. Assume wlog that $w_{vu} < \frac{1}{6}$. In this case, we replace w_{vu} with zero and w_{uv} with $w_{uv} - w_{vu}$. While this new instance is *not* a Kemeny rank aggregation instance anymore, it still is a bounded WFAS instance, meaning that we can apply the PTAS from Theorem 5.4. We can show that this also yields a $(1+\xi)$-approximation for the original Kemeny rank aggregation instance.

To add noise in the imbalanced case, notice that $w_{uv} - w_{vu} \geq \frac{2}{3}$. Thus, we can add noise to $w_{uv} - w_{vu}$ and leave the other term zero. One can argue that, with high probability, all the values $w_{uv} - w_{vu}$ remains non-negative after noise addition. In the balanced case, we simply add noise to both w_{uv}, w_{vu} as usual. As already mentioned above, since they are both at least $\frac{1}{6}$ beforehand, they remain non-negative after noise addition with high probability.

In our full algorithm–presented in Algorithm 4, we need one additional step in order to determine which case each pair u, v belongs to. This is because directly checking if e.g. $w_{uv} \in [\frac{1}{6}, \frac{5}{6}]$ violates DP. Nevertheless, this step turns out to be simple: We can add noises to all of w_{uv} and classify them accordingly. As with the small n case, the noise distribution $\mathcal{D}$ here is either Laplace or Gaussian based on whether we desire ε-DP or (ε, δ)-DP.

The crux of the proof is to show that, with high probability, the following two events hold: (i) the instance $\tilde{\mathbf{w}}'$ is bounded, and (ii) for *all* permutations π, the cost difference $\mathrm{cost}_{\tilde{\mathbf{w}}}(\pi) - \mathrm{cost}_{\tilde{\mathbf{w}}'}(\pi)$ has small magnitude. (i) follows from standard concentration of the noise. As for (ii), we can write the cost difference as a *summation* of at most m^2 (independent) noises, and we argue that this sum is highly concentrated. For Gaussian noise, this is again simple since the sum of noises is also a Gaussian. However, for Laplace noise, we need to use a result of (Chan, Shi, and Song 2011) on the concentration of a sum of Laplace random variables.

Algorithm 4: DP PTAS for Large n

Input: Dataset $\Pi = (\pi_1, \ldots, \pi_n) \in (\mathbf{S}_m)^n$
Output: Aggregated rank $\psi \in \mathbf{S}_m$
Parameters: Noise distribution $\mathcal{D}$

$\mathcal{Z}_I \leftarrow \emptyset, \mathcal{Z}_B \leftarrow \emptyset$
for all $1 \leq u < v \leq m$ **do** {*Pairwise Classification*}
  $w'_{uv} \leftarrow w^{\Pi}_{uv} + \theta_{uv}$ where $\theta_{uv} \sim D$
  if $w'_{uv} > 5/6$ **then**
    Add (u, v) to $\mathcal{Z}_I$
  else if $w'_{uv} < 1/6$ **then**
    Add (v, u) to $\mathcal{Z}_I$
  else $\{\frac{1}{6} \leq w'_{uv} \leq \frac{5}{6}\}$
    Add (u, v) to $\mathcal{Z}_B$
for all $(u, v) \in \mathcal{Z}_I$ **do** {*Noise Addition: Imbalanced*}
  $\tilde{w}'_{uv} \leftarrow w_{uv} - w_{vu} + r_{uv}$ where $r_{uv} \sim \mathcal{D}$
  $\tilde{w}'_{vu} \leftarrow 0$
for all $(u, v) \in \mathcal{Z}_B$ **do** {*Noise Addition: Balanced*}
  $\tilde{w}'_{uv} \leftarrow w_{uv} + r_{uv}$ where $r_{uv} \sim \mathcal{D}$
  $\tilde{w}'_{vu} \leftarrow 1 - \tilde{w}'_{uv}$
if $\tilde{\mathbf{w}}'$ is a bounded instance **then** {*Approximation*}
  return Output from running the algorithm from Theorem 5.4 on $\tilde{\mathbf{w}}'$
else
  return random permutation from $\mathbf{S}_m$

6 Conclusion and Discussion

We study rank aggregation problems under DP, both in the central and local models. For footrule rank aggregation, we give a polynomial-time algorithm with nearly optimal errors in the central model, which translates to 2-approximation algorithms for Kemeny rank aggregation. We also improve the additive error in the PTAS for the problem compared to (Alabi et al. 2022). The obvious open question here is to close the gap in terms of the additive errors of the PTAS. In particular, for ε-DP and (ε, δ)-DP, our upper bounds are $\frac{1}{\varepsilon n} \cdot \tilde{O}(m^{27/7})$ and $\frac{1}{\varepsilon n} \cdot \tilde{O}(m^{65/22})$, respectively, whereas the lower bounds from (Alabi et al. 2022) are only $\frac{1}{\varepsilon n} \cdot \Omega(m^3)$ and $\frac{1}{\varepsilon n} \cdot \Omega(m^{2.5})$, respectively. Another interesting research direction is to prove lower bounds for LDP; to the best of our knowledge, no previous work has pursued this direction.

Acknowledgments

We are grateful to Jittat Fakcharoenphol for hosting us and for helpful discussions during the early stages of the project. The project was done while PV was working with JF at Kasetsart University, Thailand. PV is supported by Graduate Student Scholarship, Faculty of Engineering, Kasetsart University, Grant Number 65/06/COM/M.Eng and NCN grant number 2020/39/B/ST6/01641. VS is supported by JST NEXUS Grant Number Y2024L0906031 and KAKENHI Grant JP25K00369. At the time of this project, QH was affiliated with The University of Tokyo and supported by KAKENHI Grant 20H05965, and by JST SPRING Grant Number JPMJSP2108. QH is currently part of BARC supported by the VILLUM Foundation grant 54451.

References

Abowd, J. M. 2018. The U.S. Census Bureau Adopts Differential Privacy. In *KDD 2019*, 6353–6356.

Ailon, N.; Charikar, M.; and Newman, A. 2008. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5).

Alabi, D.; Ghazi, B.; Kumar, R.; and Manurangsi, P. 2022. Private Rank Aggregation in Central and Local Models. In *AAAI 2022*, 5984–5991.

Barak, B.; Chaudhuri, K.; Dwork, C.; Kale, S.; McSherry, F.; and Talwar, K. 2007. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *PODS 2007*, 273–282.

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

Bun, M.; Ullman, J. R.; and Vadhan, S. P. 2014. Fingerprinting codes and the price of approximate differential privacy. In *STOC 2014*, 1–10.

Chan, T.-H. H.; Shi, E.; and Song, D. 2011. Private and Continual Release of Statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3).

Diaconis, P.; and Graham, R. L. 1977. Spearman's Footrule as a Measure of Disarray. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(2): 262–268.

Duchi, J. C.; Jordan, M. I.; and Wainwright, M. J. 2014. Local Privacy, Data Processing Inequalities, and Statistical Minimax Rates. CoRR:abs/1302.3203.

Dwork, C.; Kenthapadi, K.; McSherry, F.; Mironov, I.; and Naor, M. 2006a. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *EUROCRYPT 2006*, 486–503.

Dwork, C.; Kumar, R.; Naor, M.; and Sivakumar, D. 2001. Rank aggregation methods for the Web. In *WWW 2001*, 613–622.

Dwork, C.; McSherry, F.; Nissim, K.; and Smith, A. D. 2006b. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC 2006*, 265–284.

Dwork, C.; Naor, M.; Pitassi, T.; and Rothblum, G. N. 2010. Differential privacy under continual observation. In *STOC 2010*, 715–724.

Dwork, C.; Nikolov, A.; and Talwar, K. 2015. Efficient Algorithms for Privately Releasing Marginals via Convex Relaxations. *Discret. Comput. Geom.*, 53(3): 650–673.

Erlingsson, Ú.; Pihur, V.; and Korolova, A. 2014. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *CCS 2014*, 1054–1067.

Ghazi, B.; Kumar, R.; and Manurangsi, P. 2020. Differentially Private Clustering: Tight Approximation Ratios. In *NeurIPS 2020*.

Gillenwater, J.; Joseph, M.; and Kulesza, A. 2021. Differentially Private Quantiles. In *ICML 2021*, 3713–3722.

Guruswami, V.; Håstad, J.; Manokaran, R.; Raghavendra, P.; and Charikar, M. 2011. Beating the Random Ordering Is Hard: Every Ordering CSP Is Approximation Resistant. *SIAM J. Comput.*, 40(3): 878–914.

Hay, M.; Elagina, L.; and Miklau, G. 2017. Differentially Private Rank Aggregation. In *SDM 2017*, 669–677.

Kasiviswanathan, S. P.; Lee, H. K.; Nissim, K.; Raskhodnikova, S.; and Smith, A. 2011. What Can We Learn Privately? *SIAM Journal on Computing*, 40(3): 793–826.

Kemeny, J. G. 1959. Mathematics without Numbers. *Daedalus*, 88(4): 577–591.

Kenyon-Mathieu, C.; and Schudy, W. 2007. How to rank with few errors. In *STOC 2007*, 95–103.

Lan, Q.; Song, B.; Li, Y.; and Li, G. 2024. Distributed Differentially Private Ranking Aggregation. *IEEE Trans. Comput. Soc. Syst.*, 11(1): 503–513.

Liu, A.; Lu, Y.; Xia, L.; and Zikas, V. 2020. How Private are Commonly-Used Voting Rules? In *UAI 2020*, 629–638.

Nikolov, A. 2024. Private Query Release via the Johnson-Lindenstrauss Transform. *Journal of Privacy and Confidentiality*, 14(2).

Xu, S.; Sun, W. W.; and Cheng, G. 2023. Ranking Differential Privacy. *CoRR*, abs/2301.00841.

Xu, S.; Sun, W. W.; and Cheng, G. 2025. Rate-Optimal Rank Aggregation with Private Pairwise Rankings. *Journal of the American Statistical Association*, 120(550): 737–750.

Yan, Z.; Li, G.; and Liu, J. 2020. Private rank aggregation under local differential privacy. *Int. J. Intell. Syst.*, 35(10): 1492–1519.