Differentially Private Rank Aggregation

Michael Hay * Liudmila Elagina † Gerome Miklau †

Abstract

Given a collection of rankings of a set of items, *rank aggregation* seeks to compute a ranking that can serve as a single best representative of the collection. Rank aggregation is a well-studied problem and a number of effective algorithmic solutions have been proposed in the literature. However, when individuals are asked to contribute a ranking, they may be concerned that their personal preferences will be disclosed inappropriately to others. This acts as a disincentive to individuals to respond honestly in expressing their preferences and impedes data collection and data sharing.

We address this problem by investigating rank aggregation under differential privacy, which requires that a released output (here, the aggregate ranking computed from individuals' rankings) remain almost the same if any one individual's ranking is removed from the input. We propose a number of differentially-private rank aggregation algorithms: two are inspired by non-private approximate rank aggregators from the existing literature; another uses a novel rejection sampling method to sample privately from a complex distribution. For all the methods we propose, we quantify, both theoretically and empirically, the "cost" of privacy in terms of the quality of the rank aggregation computed.

1 Introduction

Preference data describes an individual's opinion about the relative quality of two or more alternatives. With the popularization of social networks, Web search, and other online services, preference data is now being collected on a massive scale, reflecting users' opinions about hotels they have visited, movies they have watched, employees with whom they work, among many other examples. Preference data is also sometimes implied by records of an individual's choices. When a web surfer clicks on a search result, it implies they prefer it to other alternatives, just as a student choosing to attend a college indicates it is preferred over other colleges to which they were admitted.

One of the most common representations of preference data is the *ranking*, which places items into a total order based on a user's preferences. Given a collection of rankings from a population of users, one key problem is *rank aggregation*, which seeks to produce a single ranking that is representative of the input rankings. For example, in recommendation systems, rank aggregation is a common tool for combining users' preferences into a single ranking that agrees the most with those preferences. Rank aggregation has been studied in numerous fields, from philosophy and psychology to economics and database systems, and accordingly, many alternative algorithms have been proposed in the literature [3–8, 13, 15, 18, 20].

As preference data is collected, published, and mined, concerns about privacy have grown. In many domains, an individual's personal preferences and judgments of relative quality may be highly sensitive and worthy of privacy protection. Even if the disclosure of an individual's preferences in a given domain is not embarrassing or overtly harmful, the ability of an outsider to deduce an individual's preference may make them susceptible to coercion. Even fear of disclosure or potential coercion might prevent an individual from contributing accurate preference data.

For example, employees ranking their institution against competing institutions might be reluctant to respond honestly if their true preferences would rank their institution lower than desired by their managers. Another example arose recently in the context of the Hugo Awards, which are annual awards for the best science fiction or fantasy writing. Each individual can nominate up to five candidates for each of the Hugo categories. In 2015, the organizers proposed a new system for selecting finalists that was intended to be more accurate. In order to thoroughly test the new system, Hugo administrators planned to release nomination data. However they soon realized that even after anonymizing individual's preferences, they were able to re-identify individuals [1]. This case, in addition to well-known attacks on other forms of anonymized data (e.g., [19]), shows that anonymization is unlikely to provide reliable protection for preference data.

We adapt the provably private model of differential privacy [9] to preference data, focusing on the important case where individuals contribute complete rankings of a set of items. Our goal is to release an aggregate ranking derived from the collection.

Differential privacy provides a compelling, provable privacy guarantee, ensuring that the outcome of any analysis on a database is not influenced substantially by the presence or absence of any one individual's data. Researchers have recently designed differentially private algorithms to support many common data mining tasks (e.g., private decision trees, private SVM, logistic regression and others) and recent work has also broadened the scope to support mining of more complex data (e.g., graphs and sequential data). However, these methods do not offer easy or effective solutions to the prob-

*Department of Computer Science, Colgate University, mhay@colgate.edu

†College of Information and Computer Sciences, University of Massachusetts Amherst, {lelagina,miklau}@cs.umass.edu

lem of rank aggregation. Rankings are challenging to work with under differential privacy because of the high dimensionality (the number of ranked items) and the constraints inherent in analyzing rankings and sets of rankings.

Differentially private rank aggregation is a novel problem; we are aware of only one prior work on the topic [21] but it does not scale to rankings of more than a handful of items, and we show that our methods provide significantly lower error for the same degree of privacy protection.

Our contributions include the following:

- We provide theoretical insights into differentially private rank aggregation by proving bounds on error in terms of key parameters, which explain how many rankings (as a function of the privacy parameter ϵ and the number of ranked items) are required to achieve a target error rate.
- We propose three differentially private rank aggregation algorithms, offering different trade-offs in error, efficiency and asymptotic consistency[1]. A private version of the well-known (but sub-optimal) Borda method is difficult to analyze theoretically, but is efficient and performs well in some practical settings. We design a private version of a quicksort-inspired algorithm and show it is consistent under some distributional properties, but it may be outperformed when the number of rankers is small. Our third algorithm is inspired by an exact solution to the rank aggregation problem and includes a novel adaptive rejection sampling technique applied to the well-known differentially private Exponential Mechanism. We show it is consistent in the greatest number of cases, but at the cost of efficiency: it has exponential running time in the worst case. Nevertheless, on practical inputs it is efficient.
- We evaluate the accuracy of differentially private rank aggregation on real and synthetic data, showing (for the first time) that rank aggregation can be performed with little practical impact on accuracy while offering individuals a formal guarantee of privacy.

The rest of the paper is organized as follows. Section 2 formalizes the private rank aggregation problem and reviews background on differential privacy. We introduce two algorithms in Section 3, each of which is an adaptation of an existing (non-private) approximate rank aggregation algorithm. Section 4 presents our approach inspired by optimal rank aggregation. We present a theoretical analysis in Section 5, empirical results in Section 6, related work in Section 7, and conclusions in Section 8.

[1]Consistency means that for fixed privacy parameters, error goes to zero as the number of input rankings goes to infinity.

2 Background

In this section we provide background on rank aggregation, formalize differentially private rank aggregation and its goals, and describe basic building blocks of differentially-private algorithms.

2.1 Ranking and rank aggregation Given a universe U of elements, a ranking is an ordered list τ which contains each element of U. We write a ranking τ as $\tau = \{x_1, x_2, ..., x_m\}$ where each $x_i \in U$ and m is the number of elements in the universe, $m = |U|$. We denote by $\tau(x)$ the rank position of element x according to τ: the best ranked item has position 0 while the worst ranked item has position $m-1$. For simplicity of exposition, we restrict our attention to these complete rankings, in which every element in U is ranked. Our techniques can be applied to partial rankings but we leave this extension as future work.

We consider a database of rankings $\mathrm{T} = \{\tau_1, \ldots \tau_n\}$ in which each ranking is contributed by an individual *voter*. The goal of rank aggregation is to find a single representative ranking (not necessarily from T) that reflects the rankings in T. An aggregate ranking may be seen as a summary of the rank collection or as a single ranking that improves upon any individual ranking by incorporating the preferences of a group.

The quality of an aggregate ranking is typically measured using the *Kendall tau* metric for rankings. For two permutations σ and τ, the *Kendall-tau* distance is defined as the number of pairwise disagreements between two permutations: $\mathbf{K}(\sigma, \tau) = |\{(i,j) : i < j, \sigma(i) < \sigma(j) \text{ but } \tau(i) > \tau(j)\}|$. Note that $\mathbf{K}(\tau, \tau) = 0$ and that the maximum possible Kendall tau distance occurs when σ is the reverse ranking of a ranking τ; in that case, $\mathbf{K}(\sigma, \tau) = \binom{m}{2}$. The *average Kendall tau distance* of a ranking σ to a rank database $\mathrm{T} = \{\tau_1, ..., \tau_n\}$ is naturally defined as: $\overline{\mathbf{K}}(\sigma, \mathrm{T}) = \frac{1}{n}\sum_{i=1}^{n} K(\sigma, \tau_i)$.

The most common criterion for determining the "best" aggregate ranking is the Kemeny optimal aggregation: the one that minimizes the average Kendall tau distance to T.

DEFINITION 1. (KEMENY OPTIMAL RANK AGGREGATION) *Given a ranking database* $\mathrm{T} = \{\tau_1 \ldots \tau_n\}$*, the Kemeny optimal aggregate ranking is a ranking* σ *for which* $\overline{\mathbf{K}}(\sigma, \mathrm{T})$ *is minimal over the set of all possible rankings.*

The Kemeny optimal ranking is a desirable aggregate ranking because it places first the Condorcet winner (a candidate not preferred to any other in pairwise comparisons). While computing the Kemeny optimal ranking is NP-hard for rank databases with size $n > 3$, PTIME approximation algorithms exist. In addition, a wide array of heuristic methods have been proposed that may involve scoring and ranking elements, performing local directed search in the rank space, or defining a Markov chain from the rank database and rank-

ing by stationary probabilities of the elements, among many others. We review in detail relevant algorithms in Sections 3 and 4 and discuss others in Section 7.

2.2 Private rank aggregation Differential privacy is applied in settings where each individual in a population contributes potentially sensitive information to a database, controlled by a trusted party, the *data owner*. The goal is to release the results of one or more computations on the database to the *analyst*, a potentially untrusted party. Information contributed by an individual should not be shared with the analyst, but the goal is nevertheless to share with the analyst accurate properties about the population as a whole, often in the form of aggregates computed on the database.

Differential privacy [9] offers a model in which to measure and limit the loss of privacy suffered by any individual. Informally, differential privacy promises that the released output will be approximately the same, subject to a privacy parameter ϵ, whether or not any one individual's information is included.

Differential privacy is formally defined as a property of an algorithm that computes a sensitive function on the database. Let D denote a database consisting of a set of records where each record corresponds to the sensitive information of a single individual. Two databases D and D' are *neighbors* if they differ by a single tuple—i.e., $|(D - D') \cup (D' - D)| = 1$.

DEFINITION 2. ($\epsilon-$DIFFERENTIAL PRIVACY [9]) *A randomized algorithm $\mathcal{M}$ satisfies $\epsilon-$ differential privacy if for all neighboring datasets D and D' and all $S \subseteq$ Range$(\mathcal{M})$, Pr$[\mathcal{M}(D) \in S] \leq \exp(\epsilon) \times$ Pr$[\mathcal{M}(D') \in S]$*

The privacy parameter ϵ controls the strength of the privacy definition: lower ϵ provides stronger privacy but typically implies greater distortion in released results. The data owner determines an appropriate ϵ for an analyst and this value is often referred to as the analyst's privacy "budget."

We apply differential privacy in the context of ranking by considering an individual's information to consist of their complete ranking of all elements in U, and we therefore provide strong privacy protection to all of the preference information they may reveal by participating in the database. Alternative formulations, which might protect only parts of an individual's ranking, or just pairwise comparison information, would be weaker: they might, for example, protect an individual's top preferences, but reveal preferences about lower ranked elements. Accordingly, we say that two rank databases T and T' are neighboring if they differ in the presence or absence of exactly one ranking.

The goal of private rank aggregation is to devise an ϵ-differentially private algorithm for computing an aggregate ranking. The primary quality metric of a private rank aggregator is its accuracy. We measure the accuracy of a private rank aggregation algorithm by comparing the average Kendall tau of its output to the average Kendall tau of the optimal ranking. The difference is our error metric:

DEFINITION 3. (ERROR) *Let T be a ranking database, τ_{opt} the Kemeny optimal aggregate ranking for T, and $\mathcal{A}$ a differentially-private rank aggregator. If $\tilde{\sigma} = \mathcal{A}(\mathrm{T}, \epsilon)$ is the private aggregate ranking returned by $\mathcal{A}$, the average Kendall tau error is: $\overline{\mathbf{K}}(\tilde{\sigma}, \mathrm{T}) - \overline{\mathbf{K}}(\tau_{opt}, \mathrm{T})$.*

Error quantifies the cost, in accuracy, of offering a strong privacy guarantee. Note that, in the absence of privacy, rank aggregation algorithms may have inherent error due to computational constraints. The error of a *private* rank aggregation algorithm may arise from at least two additional sources. First, is the strength of the "signal" available to the private aggregation algorithm. This is determined by the ϵ privacy parameter (higher epsilon means lower privacy and stronger signal) and by the size, n, of the rank database (more rankings mean that the noise added for privacy typically has less impact on the algorithm behavior). Second, is the power of the algorithm to efficiently utilize the signal to find a good aggregate ranking. As we will see, it is not always the case that the best algorithmic approach among non-private algorithms is still a good approach for a private algorithm that must cope with noisy observations of the input database.

2.3 Standard differentially private mechanisms The following are standard methods for constructing differentially private algorithms. Although these methods are quite general and are sufficient for satisfying differential privacy, used by themselves they are usually sub-optimal in terms of error and/or computationally intractable. Nevertheless, they are often important building blocks of more sophisticated private algorithms that can offer improved error and good computational behavior.

The Laplace mechanism adds noise to a numerical function of the input data, scaling the noise to the *sensitivity* of the function, which measures the worst-case impact on the output of the addition or removal of one record.

DEFINITION 4. (SENSITIVITY) *Let $\mathcal{D}$ denote the space of all databases. For a function $f : \mathcal{D} \rightarrow R^k$, the sensitivity of f is: $\Delta f = \max_{D,D'} ||f(D) - f(D')||_1$ for all neighboring databases D and D'.*

The Laplace mechanism generates noise by sampling i.i.d. variables from a Laplace distribution, denoted Laplace(σ) where σ is the scale of the distribution.

DEFINITION 5. (LAPLACE MECHANISM [10]) *Given function $f : \mathcal{D} \rightarrow \mathbb{R}^k$, the Laplace mechanism $\mathcal{L}$ adds noise to each of the k components of $f(D)$: $\mathcal{L}(D, f(\cdot), \epsilon) = f(D) + (Y_1, Y_2, ..., Y_k)$ where Y_i are i.i.d. random variables drawn from Laplace$(\Delta f/\epsilon)$.*

When the desired output is not numerical we cannot construct differentially private algorithms by noise addition. In place of noise addition, the exponential mechanism directly defines a probability distribution over an arbitrary output space. A quality function measures the desirability of each possible output:

DEFINITION 6. (EXPONENTIAL MECHANISM [10]) *Given quality function $u : \mathcal{D} \times \mathcal{R} \to \mathbb{R}$, which maps database/output pair to utility scores, the exponential mechanism $\mathcal{M}_E(D, u, \mathcal{R})$ selects and outputs an element $r \in \mathcal{R}$ with probability $\Pr[\mathcal{M}_E(D, u, \mathcal{R}) = r] \propto \exp(\epsilon u(D, r)/2\Delta u)$ where Δu is the sensitivity of u (computed over all possible $r \in \mathcal{R}$).*

3 Approximate Rank Aggregation

In this section we describe two existing non-private algorithmic approaches to rank aggregation, each offering provable approximations of the optimal aggregate ranking. We then propose differentially private adaptations of these algorithms.

3.1 Borda aggregation Borda's method [15] was originally proposed as a single-winner selection method where each item receives a certain number of points based on its position in any given ranking. For example, with a universe of five elements, an element will receive 0 points every time it is ranked in first place, 1 point for second place, and so on. The element with the lowest number of points is identified as the winner.

DEFINITION 7. (BORDA SCORE) *Given a ranking database $\mathrm{T} = \{\tau_1 \dots \tau_n\}$ over a universe $U = \{x_1 \dots x_m\}$, the Borda score of element x_i is: $\mathit{bscore}(x_i) = \sum_{j=1}^{n} \tau_j(x_i)$.*

Borda scores can be used as a rank aggregation method by sorting the elements by their Borda scores. The Borda score aggregator is computationally efficient and (when ties are broken arbitrarily) was shown to be a 5-approximation of the optimal rank aggregation [6].

EXAMPLE 1. *Using the example rank database in Fig. 1, notice that element A is ranked in positions $1, 0, 2, 4, 1, 3, 4, 4$ and therefore the Borda score is $\mathit{bscore}(A) = 19$. The Borda scores for elements B, C, D, E are, respectively, 19, 13, 18, 11, which results in an aggregate ranking of $\langle E, C, D, A, B\rangle$. This ranking has an average Kendall tau of 0.40. The Kemeny optimal ranking is $\langle E, C, B, D, A\rangle$ with average Kendall tau distance of 0.37.*

3.2 Private Borda aggregation We use Borda scores as the basis of a differentially private rank aggregator by first using the Laplace mechanism to privately compute a noisy estimate of the Borda score of each candidate. Then we sort the noisy scores to get an aggregate ranking.

DEFINITION 8. (PRIVATE BORDA SCORES) *Given a ranking database $\mathrm{T} = \{\tau_1 \dots \tau_n\}$ over a universe of elements $U = \{x_1 \dots x_m\}$ and privacy parameter ϵ, the* P-BORDA *algorithm returns $\langle \tilde{b}_1 \dots \tilde{b}_m\rangle$ where $\tilde{b}_i \in \mathbb{R}$ is a noisy estimate of the Borda score of element i. For each $i \in [1, m]$, $b_i = \mathit{bscore}(x_i) + Z$ where $Z \sim \mathit{Laplace}(m(m-1)/(2\epsilon))$.*

The above approach satisfies ϵ-differential privacy (proofs omitted due to space limitations).

PROPOSITION 3.1. P-BORDA *is ϵ-differentially private.*

3.3 Quicksort aggregation Ailon et al. [3] introduces a 2-approximation algorithm for rank aggregation based on a version of quicksort that is guided in its precedence decisions by the rankings in T. When comparing two elements, x_i and x_j, the elements are ordered based on the majority preference of rankings in T. Formally, let $C_{ij}(\mathrm{T})$ denote the number of times x_i is preferred to x_j among the rankings in T, i.e., $C_{ij}(\mathrm{T}) = |\{\tau \in T \mid \tau(x_i) < \tau(x_j)\}|$. The elements are sorted using the comparison function $\mathrm{cmp}_{\mathrm{T}}(x_i, x_j) = (C_{ji}(\mathrm{T}) - C_{ij}(\mathrm{T}))$ where x_i is placed before x_j when $\mathrm{cmp}_{\mathrm{T}}(x_i, x_j) < 0$, vice versa when $\mathrm{cmp}_{\mathrm{T}}(x_i, x_j) > 0$, and arbitrarily when $\mathrm{cmp}_{\mathrm{T}}(x_i, x_j) = 0$. The pivots of quicksort are chosen randomly.

3.4 Private quicksort aggregation To make a private version of the sorting algorithm, we must make all interactions with the input rankings differentially private. In this case, the input is only used by the comparison function. We apply the Laplace mechanism to make the comparison function differentially private. Since the comparison is invoked multiple times, each invocation receives only an ϵ' share of the privacy budget (described below). Let $\widetilde{\mathrm{cmp}}_{\mathrm{T}}$ denote this noisy comparison function $\widetilde{\mathrm{cmp}}_{\mathrm{T}}(x_i, x_j) = \mathrm{cmp}_{\mathrm{T}}(x_i, x_j) + Z$ where $Z \sim \mathrm{Laplace}(1/\epsilon')$.

DEFINITION 9. (PRIVATE QUICKSORT) *Given a ranking database $\mathrm{T} = \{\tau_1 \dots \tau_n\}$ over a universe of elements $U = \{x_1 \dots x_m\}$ and privacy parameter ϵ, the* P-SORT *algorithm runs quicksort on the elements using randomly chosen pivots and the noisy comparison function $\widetilde{\mathrm{cmp}}_{\mathrm{T}}$. The first M invocations of $\widetilde{\mathrm{cmp}}_{\mathrm{T}}$ receive $\epsilon' = \epsilon/M$ where $M = ((m-1)\log m)$. Remaining invocations order elements randomly.*

PROPOSITION 3.2. P-SORT *satisfies ϵ-differential privacy.*

4 Optimal rank aggregation

Although computing the Kemeny optimal ranking is NP-hard, algorithms have been developed for solving it exactly.

	Input Rankings								Aggregate Rankings		
Rank	voter 1	voter 2	voter 3	voter 4	voter 5	voter 6	voter 7	voter 8	OPT	Borda	QS
0	E	A	C	E	B	C	C	E	E	E	E
1	A	E	B	D	A	E	B	D	C	C	C
2	C	D	A	C	D	D	E	C	B	D	B
3	B	C	D	B	E	A	D	B	D	A	A
4	D	B	E	A	C	B	A	A	A	B	D

Figure 1: A ranking database of 8 rankings over the set of elements $\{A, B, C, D, E\}$; aggregated rankings produced by Kemeny optimal (0.37), Borda (0.40), and Quick sort (0.38) algorithms.

These algorithms require exponential time in the worst case, but in practice may find a solution very efficiently. In this section we propose a novel private aggregation method inspired by these techniques. Our algorithm is based on the exponential mechanism, and in the absence of the privacy constraint (i.e., as $\epsilon \to \infty$) the algorithm will return an optimal solution. Algorithms based on the exponential mechanism are typically easy to describe: in our case we simply wish to privately select the ranking that minimizes average Kendall tau and the exponential mechanism describes a correct distribution over rankings that will ensure privacy. This is an appealingly direct approach, but the challenge is that sampling from this distribution seems infeasible. Nevertheless, the fact that, in the absence of privacy, optimal solutions can be efficiently computed in practice suggests that it may be possible to efficiently sample from this distribution.

A persistent problem with algorithms based on the exponential mechanism is that the probability defined over outputs is often complex and approximate sampling methods cannot be used: it is possible that an approximate distribution will not satisfy the constraint differential privacy imposes on outputs. In addition, it is typically difficult to reason about an approximate sampling technique and its convergence to the true distribution in order to prove that violations of differential privacy are impossible.

Below we propose a novel approach that avoids this issue, based on adaptive sequential rejection sampling, which we believe will be of independent interest for the implementation of privacy algorithms in other domains.

4.1 Optimal (non-private) rank aggregation In the non-private setting, two different approaches have been proposed for computing the optimal solution. The rank aggregation problem can be formulated as an integer linear program (ILP) [5]. The relaxed LP is a 4/3 approximation and empirically often yields integral solutions [20]. In addition, an algorithm based on A* search has been shown to run in $O(m^2)$ time on inputs where there is strong agreement among rankings [18]. Finally, empirical studies show that exact algorithms have reasonably low run times on real and synthetic data [4, 20].

For our experiments, we implemented a variant of the A* approach of Meila et al. [18]. The nodes in the search tree correspond to a partial ranking consisting of the top k elements, and each node expansion selects another candidate to insert into the $(k+1)^{th}$ position. The node cost is the contribution to the average Kendall tau distance of placing these k elements in the first k positions. In our implementation, we use a different admissable heuristic: the sum of $\min\{C_{ij}(\mathrm{T}), C_{ji}(\mathrm{T})\}$ for each pair (i, j) among the remaining elements, which is a lower bound on the cost of ranking these elements. We find this heuristic greatly reduces runtime.

4.2 Optimal private aggregation Recall from Sec. 2.3 that the exponential mechanism allows us to define a score function which measures the desirability of each output among a set of possible outputs. Here we instantiate the exponential mechanism over the set of all rankings, scored by the total Kendall tau distance to the database.

DEFINITION 10. (PRIVATE PERMUTATION SAMPLING) *Given a ranking database* $\mathrm{T} = \{\tau_1 \dots \tau_n\}$ *over a universe* $U = \{x_1 \dots x_m\}$*, the* P-SAMPLE *is an invocation of the exponential mechanism in which the output range* $\mathcal{R}$ *is the set of all permutations over* U*, and the utility function* u *is defined as* $u(\mathrm{T}, \sigma) = -n\overline{\mathbf{K}}(\sigma, \mathrm{T})$*. On input* T*, it outputs permutation* σ *with probability proportional to:* $\exp(-\epsilon n\overline{\mathbf{K}}(\sigma, \mathrm{T})/\Delta u)$ *where* $\Delta u = \binom{m}{2}$.

The probability mass function defined above differs slightly from the one in Definition 6 in that a factor of 2 is omitted. As has been observed elsewhere [10], this factor can be omitted when the quality function is a monotonic (either non-decreasing or non-increasing) function of the private input, which is the case here.

PROPOSITION 4.1. P-SAMPLE *is* ϵ*-differentially private.*

As mentioned above, P-SAMPLE is an analogue of optimal aggregation in the sense that without privacy (i.e., as $\epsilon \to \infty$) the above algorithm will return $\arg\min_\sigma \overline{\mathbf{K}}(\sigma, \mathrm{T})$, the optimal aggregate ranking. (As shown later, the algorithm also offers accuracy guarantees in the case of finite ϵ.)

We now turn to the problem of sampling from this distribution. To solve this problem, we use the machinery of

graphical models [14] and decompose the desired output (a permutation) into a collection of random variables such that (a) their joint probability distribution is equal to the distribution above, and (b) the distribution admits a factored representation as a product of potential functions over subsets of the variables. We then apply sampling methods developed for graphical models.

We represent a ranking σ in terms of $\binom{m}{2}$ binary indicator variables, one per pair of elements. Let $\mathbf{Y}^\sigma = \{Y_{ij}^\sigma\}$ be a set of indicator variables for $i, j \in [1, m]$ and $i < j$ where $Y_{ij}^\sigma = 1$ if $\sigma(i) < \sigma(j)$ and 0 otherwise. A *valid* assignment of these variables must obey transitivity: if $Y_{ij}^\sigma = 1$ and $Y_{jk}^\sigma = 1$, then Y_{ik}^σ must be 1. Any valid assignment to the variables corresponds to a unique ranking and vice versa.

The average Kendall tau distance of σ, and thus our utility function u, can be expressed in terms of these Y_{ij}^σ variables. Recall that $C_{ij}(\mathrm{T})$ denotes the number of times i is preferred to j among the voters in T. The average Kendall tau distance can be written as

$$\overline{\mathbf{K}}(\sigma, \mathrm{T}) = \frac{1}{n} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} C_{ji}(\mathrm{T}) \cdot Y_{ij}^\sigma + C_{ij}(\mathrm{T}) \cdot (1 - Y_{ij}^\sigma)$$

We can now express the probability distribution of P-SAMPLE in terms of variables Y_{ij}^σ and potential functions on those variables. For each variable Y_{ij}^σ, we associate a potential function $\psi_{ij}(y_{ij}) = \exp\left(-\epsilon \left(C_{ji} y_{ij} + C_{ij}(1 - y_{ij})\right) / \binom{m}{2}\right)$ where the dependence of C_{ij} on T is implicit and omitted from the notation. And for every triplet $\{i, j, k\} \subseteq U$, we define a potential function on the three corresponding variables: let $\psi_{ijk}(y_{ij}, y_{jk}, y_{ik})$ be 1 if the assignment satisfies transitivity and 0 otherwise. Define a probability distribution over $\mathbf{Y}^\sigma$ as follows,

$$P_{\mathrm{T}}(\mathbf{Y}^\sigma) \propto \prod_{\{i,j\} \subseteq U} \psi_{ij}(Y_{ij}^\sigma) \prod_{\{i,j,k\} \subseteq U} \psi_{ijk}(Y_{ij}^\sigma, Y_{jk}^\sigma, Y_{ik}^\sigma)$$

Sampling once from this distribution is equivalent to running P-SAMPLE. The first product of ψ_{ij} terms is equivalent to the distribution defined by the P-SAMPLE for permutation σ; the second product forces the $\mathbf{Y}^\sigma$ to be a valid assignment.

To draw samples from this distribution, we apply adaptive sequential rejection sampling [17]. The rationale for using this technique is that it generates *exact* samples from the above target distribution. More conventional techniques, such as Markov Chain Monte Carlo, yield only *approximate* samples and thus would require additional analysis to determine whether the approximation leaks private information.[2] The high level idea behind adaptive sequential rejection sampling is that it constructs a nested sequence of proposal distributions over an increasingly larger subset of the variables, using rejection sampling to filter out samples that have low probability according to the target distribution. With each rejection, the proposal distribution adapts, making it more likely that a sample is eventually accepted. In the worst case, the algorithm may reject an exponential number of samples before the first accepted sample. However, any accepted samples are guaranteed to be samples from the target distribution. Thus, differential privacy follows from (a) the guarantee that samples are exact (cf. [17]), and (b) Proposition 4.1.

To invoke the sequential rejection sampling algorithm, we must choose a sample ordering of the variables. We sample the variables in insertion sort order: for each element x_j for $j = 1 \ldots m$, we sample the sequence of variables y_{ij} for $i = 1 \ldots j-1$, thus effectively inserting element x_j into the already sampled permutation over the elements $x_1, \ldots, x_{j-1}$. Given this variable ordering and the particular proposal distribution used by the algorithm, it avoids ever sampling a value that would violate transitivity. A sample of a partial permutation may still be rejected if it only permits low probability orderings when subsequent elements are inserted. For example, it might first sample permutation x_2, x_1 only to subsequently reject it if, say, the input rankings reveal a strong preference for placing x_3 after x_1 but before x_2, thus forcing the ordering x_1, x_3, x_2.

5 Theoretical analysis

In this section, we analyze the accuracy of the algorithms presented in Sections 3 and 4.

We start with a negative result that applies to any ϵ-differentially private algorithm. It shows that privacy imposes some limits on ranking: any differentially algorithm cannot, with high probability, return the *same ranking* as a non-private optimal algorithm. Intuitively, the reason is that there exist datasets in which a single vote can completely change the optimal ranking.

THEOREM 5.1. *Let n_0 be any constant and $\mathcal{A}$ an ϵ-differentially private algorithm. Let $p(\mathcal{A}, \mathrm{T})$ denote the probability that $\mathcal{A}$ fails to output τ_{opt}, an optimal ranking for* T. *(We can assume ties are broken arbitrarily but consistently across algorithms.) If $p(\mathcal{A}, \mathrm{T}) \leq \delta$ for all datasets* T *such that $|\mathrm{T}| \geq n_0$, then $\epsilon \geq \frac{1}{2} \ln\left((m! - 1)\frac{1-\delta}{\delta}\right)$.*

Fixing δ as a constant, this says that a fixed error rate for all inputs is only feasible if $\epsilon \geq O(m \log m)$.

So while it may be impossible to return τ_{opt} on all inputs, the next result shows that P-SAMPLE can produce a ranking that is arbitrarily close to optimal, provided the input is sufficiently large. In the following theorem, we use the term *normalized error* to mean error divided by $\binom{m}{2}$, giving us an error term that is always $[0, 1]$ and is independent of m.

[2] Prior work on private MCMC [22, 23] assumes the sampling distribution converges to the target distribution – something which may not happen in practice. Because our approach generates an exact sample, it is guaranteed to achieve differential privacy in all cases.

THEOREM 5.2. *Let* T *be any database with* n *rankings and* α, δ *be constants in* $[0,1]$. *With probability at least* $(1-\delta)$, P-SAMPLE *has* normalized error *at most* α *provided that* $n \geq \frac{2}{\alpha\epsilon}\ln\left(m! + \ln\frac{1}{\delta}\right)$.

Fixing α and δ as constant, this theorem relates the three key parameters of the problem—the number of elements, the number of voters, and the privacy parameter ϵ—and says that $n = O(\frac{m\log m}{\alpha\epsilon})$ voters is sufficient to produce a ranking with constant error. It also implies that for a fixed universe of elements, error goes to zero as $n \to \infty$. Thus, with enough data, the cost of privacy goes to zero with the P-SAMPLE. We cannot say the same for the other private algorithms, given that even their non-private analogues are only known to be constant factor approximation algorithms.

However, we can say something about the error of the private sorting algorithm, P-SORT, if we make distributional assumptions about the input data. Specifically, we consider a dataset consisting of random samples from a Mallows distribution [16] with unknown reference ranking π and dispersion parameter $\phi \in (0,1]$.

THEOREM 5.3. *Let* T *be a set of* n *samples from a Mallows distribution with reference ranking* π *and* ϕ *for the dispersion parameter. With probability* $(1-\delta)$, *the* P-SORT *algorithm achieves zero error provided that* $n \geq O\left(\frac{m\log m}{\epsilon\phi^*}\left(\log m + \log\frac{1}{\delta}\right)\right)$ *where* $\phi^* = \frac{1-\phi}{1+\phi}$.

For fixed δ and ϕ, the relationship between key parameters for P-SORT is $n = O(\frac{m\log^2 m}{\epsilon})$.

6 Experiments and Evaluation

In this section we present an evaluation of our proposed algorithms on real and synthetic ranking databases. The goal of our evaluation is to understand: (1) for a fixed "signal" (database size and ϵ) which private algorithms are able to find the lowest error aggregate rankings; (2) which aspects of the input data impact error of the private aggregators; and (3) the conditions under which the provided error rates are likely to be acceptable in practice.

6.1 Experimental setup The proposed algorithms are implemented in Python and executed on a machine with 1.8 GHz Intel CPU and 8 GB of internal memory. In most cases the algorithms run in less than a few seconds, so we do not provide a detailed evaluation or comparison of execution time. We report the average Kendall tau, normalized by $m(m-1)/2$ so we can compare values consistently across different m. Because all private algorithms are randomized, for a given input ranking database and privacy parameter ϵ, we report the mean across 10 trials. Where appropriate, we also report the minimum and maximum across the randomized trial runs of the private algorithms.

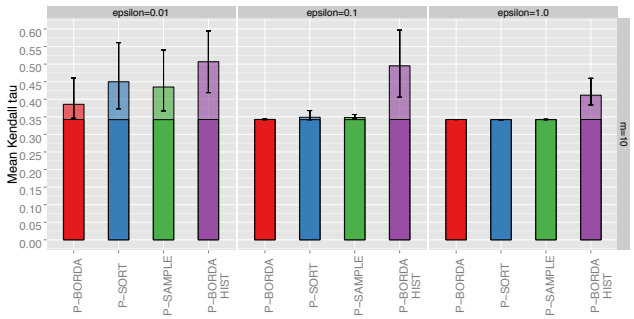
We evaluate our techniques on two real ranking datasets collected from individuals (SUSHI and JESTER) and synthetically generated data (MALLOWS); the latter allows us to control additional parameters of the input rankings.

Real datasets The SUSHI data set [2] consists of full rankings of $m = 10$ varieties of sushi from $n = 1000$ voters surveyed across Japan. The JESTER Online Joke Recommender System (Dataset 2+) [12] includes 1.7 Million continuous ratings (-10.00 to $+10.00$) of 150 jokes from 50,692 users, which were collected from 2006 to 2012. We converted the dataset into rankings by ordering jokes according to their real-valued ratings. We formed a collection of JESTER datasets by considering different settings of m (the number of jokes) and n (the number of rankers). We chose $m = 40$ and selected the top-m most-frequently rated jokes and all the individuals that rated those jokes. We included only the $n \approx 4000$ rankings that ranked all of the top-40 most frequent jokes.

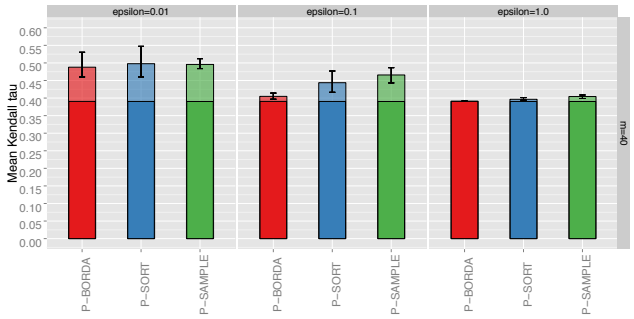
Synthetic datasets In order to generate a greater variety of datasets we also synthetically generate ranking databases derived from the Mallows model [16]. This is a distance-based, unimodal model, in which the probability of a permutation diminishes exponentially with its distance from a reference permutation. We fix a canonical reference ranking, so the remaining parameters of our synthetic datasets are m, the number of elements, n, the number of rankings, and $\phi \in (0,1]$, the dispersion parameter. We use an instance of the repeated insertion model to generate Mallows data [7].

6.2 Experiments on real datasets Figure 2 presents empirical error rates on SUSHI and JESTER, for privacy levels of $\epsilon = \{.01, .1, 1\}$. The plots are stacked bar plots which report the average Kendall tau for P-BORDA, P-SORT, and P-SAMPLE along with their non-private counterparts (Borda, quicksort, and Optimal, respectively). The average Kendall tau of the non-private algorithm is shown as the lower bar, with the distortion introduced by the private algorithm shown above it. Whiskers show the minimum and maximum error over the random trials of the private algorithms.

The results show that with sufficient signal our proposed algorithms, P-BORDA, P-SORT, P-SAMPLE, each essentially match the output of the non-private algorithms. For example, for SUSHI, with $\epsilon = 1$ or $\epsilon = .1$ we see a negligible utility cost. Note that, since the non-private counterpart of P-SAMPLE is Optimal, this is the best possible average Kendall tau. The private algorithms are matching this Kendall tau when the signal is high, but for lower signal (e.g. $\epsilon = .01$ for both SUSHI and JESTER) the increased Kendall tau shows the loss of utility. Among P-BORDA, P-SORT, and P-SAMPLE, P-BORDA performs the best. This is somewhat surprising, since computing the Borda scores privately has high sensitivity and P-SORT is designed to limit the number



(a) SUSHI dataset with 5000 voters and 10 items



(b) JESTER Jokes dataset with 4000 voters and 40 items.

Figure 2: Comparison of the average Kendall tau of private and non-private rank aggregation algorithms on real datasets, SUSHI (Figure 2a) and JESTER (Figure 2b), across varying ϵ (inset panels). (The Kendall tau on the y-axis is normalized by dividing by $\binom{m}{2}$ so that it lies between $[0, 1]$.) Private algorithms correspond to the lightly shaded bars and are labeled along the x-axis. Each private algorithm is paired with a non-private analogue: P-BORDA, P-SORT, P-SAMPLE, and P-BORDA-HIST are paired with non-private Borda, quicksort, Optimal, and Borda respectively.

of high sensitivity computations to $m \log m$. We suspect that this is due to the fact that these datasets have a high level of agreement, which will tend to create Borda scores that are more spaced out, and therefore more tolerant to recognizing their correct ordering in the presence of noise.

Comparison to P-BORDA-HIST For the SUSHI dataset (only) we include the histogram-based Borda estimator proposed by Shang et al. [21], identified as P-BORDA-HIST. This technique first represents the ranking database as a frequency vector of size $m!$ where each position represents one possible ranking over the universe U. For each possible ranking, the vector reports the number of times that ranking occurs in the database. The private aggregator is formed by adding independent Laplace$(1/\epsilon)$ noise to each position in the vector, then computing Borda scores directly from these noisy counts.

The results show that the technique significantly under-

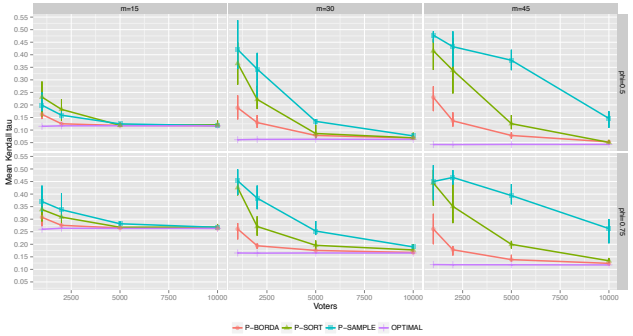


Figure 3: Comparison of (normalized) average Kendall tau for private rank aggregation algorithms on synthetic data sampled from Mallows distributions of $\phi = 0.5$ (top panel) and $\phi = 0.75$ (bottom panel) and varying numbers of elements $m \in \{15, 30, 45\}$ and voters $n \in \{1000, \ldots, 10,000\}$. Privacy parameter ϵ is fixed at 0.1. Error bars show the full range of variability (minimum and maximum) of each algorithm across random trials.

performs our proposed techniques: for $\epsilon = .01$ and $\epsilon = .1$ the average Kendall tau approaches .5, which is a rate that would also be achieved by ignoring the input database and picking a random permutation as the aggregate.

Although the noise added is relatively small, this approach suffers from two problems. First, the vector representation of the ranking database is very sparse, and will typically have low counts, so it tends to be overwhelmed by the noise added for privacy. Second, the $m!$ size of the vector does not allow this method to scale. The SUSHI dataset has a relatively small number of elements, so it is feasible to run P-BORDA-HIST, by operating on a vector of size $10! = 3.6288 \times 10^6$. Running P-BORDA-HIST for JESTER is infeasible, requiring a vector of size $40! \approx 8.1 \times 10^{47}$.

6.3 Experiments on synthetic datasets In Figure 3 we run P-BORDA, P-SORT, P-SAMPLE, and Optimal on synthetically generated data from the Mallows distribution, with the dispersion parameter $\phi \in \{.5, .75\}$. We fix $\epsilon = .1$ but vary the strength of the signal by increasing n, the number of voters in the ranking database, along the x-axis. We also consider varying the number of elements, $m \in \{15, 30, 45\}$.

The results show the relationship between m and n, namely that as m increases, we need more data (larger n) in order to maintain accurate results from the privacy algorithms. The dispersion parameter ϕ has a significant impact on the optimal average Kendall tau (e.g. for $m = 45$, it is less that 0.05 when $\phi = .5$, but about .13 when $\phi = .75$). However it does not have a major impact on the magnitude of error or the relative performance of the private algorithms.

Overall we find that as the signal diminishes P-BORDA outperforms the other algorithms, sometimes significantly.

7 Related Work

Despite the active research on differentially private data analysis, we are aware of only one other work that studies the problem of private rank aggregation: the algorithm by Shang et al. [21] that we call P-BORDA-HIST and describe in Section 6.2. Our experiments show that this approach introduces more noise than our proposed approaches; further, its runtime is exponential. Shang et al. also present a theoretical analysis that is complementary to ours: it shows that under certain distributional assumptions, private Borda will output the same ranking as non-private Borda with probability 1 in the limit of infinite voters.

The (non-private) rank aggregation problem is a well-studied problem that arises in areas such as psychology, economics, voting, online commerce, market advertisement research, information retrieval, and crowdsourcing [3–8, 13, 15, 18, 20]. The problem is known to be NP-Hard [8] and numerous approximation algorithms have been developed (cf. [3]) with the best approximation being a PTAS [13]. The problem has also been studied empirically, with at least two fairly extensive empirical studies [4, 20].

Another class of approaches to rank aggregation formulate a Markov chain over the elements, assign transition probabilities as some function of the input rankings, and rank elements in descending order of their probability in the stationary distribution [8]. In our investigation, private Markov chain approaches were not competitive with the other approaches we proposed, and even in the absence of privacy the Markov chain variants were outperformed by other methods. This is consistent with past empirical findings [4] as well as sensitivity analyses that show Markov chain rankings can be sensitive to small perturbations in the input [15].

8 Conclusion

We considered the problem of Kemeny optimal rank aggregation under the formal model of differential privacy and proposed three algorithms. Each offers a different trade-off in empirical error, efficiency and asymptotic consistency. We hypothesize that the mismatch between practical error rates and asymptotics may reflect the need for different strategies depending on "signal strength" (a function of ϵ and the number and distribution of voters). In particular, at low signal, coarser approximations like Borda scores are better able to tolerate noise. This would be consistent with other studies of differentially private algorithms [11]. Nevertheless, all of our algorithms outperform the only known prior work [21]. Further, our theoretical results demonstrate the feasibility of private (and accurate) rank aggregation.

Our future work will consider extended forms of preference data such as partial rankings, pairwise preferences, or rankings that include ties. We would also like to support additional ranking analysis tasks including fitting ranking models and clustering ranking data.

Acknowledgments This work was supported by NSF projects 1421325, 1409125 and 1409143. We thank Soo Bin Kwon and Dong Mai for their help with implementation.

References

[1] `http://11011110.livejournal.com/316771.html`.
[2] `http://www.kamishima.net/sushi/`.
[3] N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):23, 2008.
[4] A. Ali and M. Meilă. Experiments with kemeny ranking: What works when? *Mathematical Social Sciences*, 2012.
[5] V. Conitzer, A. Davenport, and J. Kalagnanam. Improved bounds for computing kemeny rankings. In *AAAI*, 2006.
[6] D. Coppersmith, L. K. Fleischer, and A. Rurda. Ordering by weighted number of wins gives a good ranking for weighted tournaments. *ACM Trans. Algorithms*, 6(3), 2010.
[7] J.-P. Doignon, A. Pekec, and M. Regenwetter. The repeated insertion model for rankings: Missing link between two subset choice models. *Psychometrika*, 69(1):33–54, 2004.
[8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *WWW*, 2001.
[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.
[10] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comp. Sci.*, 2014.
[11] M. Hay, A. Machanavajjhala, G. Miklau, Y. Chen, and D. Zhang. Principled evaluation of differentially private algorithms using dpbench. In *SIGMOD*, 2016.
[12] `http://www.ieor.berkeley.edu/~goldberg/jester-data/`.
[13] C. Kenyon-Mathieu and W. Schudy. How to rank with few errors. In *STOC*, 2007.
[14] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
[15] A. N. Langville and C. D. Meyer. *Who's# 1?: The science of rating and ranking*. Princeton University Press, 2012.
[16] C. L. Mallows. Non-null ranking models. I. *Biometrika*, 44(1-2):114–130, 1957.
[17] V. Mansinghka, D. Roy, E. Jonas, and J. Tenenbaum. Exact and approximate sampling by systematic stochastic search. In *AISTATS*, 2009.
[18] M. Meila, K. Phadnis, A. Patterson, and J. Bilmes. Consensus ranking under the exponential model. In *UAI*, 2007.
[19] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy*, May 2008.
[20] F. Schalekamp and A. van Zuylen. Rank aggregation: Together we're strong. In *Proceedings of the Meeting on Algorithm Engineering & Expermiments*, 2009.
[21] S. Shang, T. Wang, P. Cuff, and S. Kulkarni. The application of differential privacy for rank aggregation: Privacy and accuracy. In *FUSION*, July 2014.
[22] E. Shen and T. Yu. Mining frequent graph patterns with differential privacy. In *KDD*, 2013.
[23] Q. Xiao, R. Chen, and K.-L. Tan. Differentially private network data release via structural inference. In *KDD*, 2014.