Parameterized Enumeration of (Locally-) Optimal Aggregations

Naomi Nishimura and Narges Simjour*

Cheriton School of Computer Science, University of Waterloo,
Waterloo, Ontario, Canada
{nishi,nsimjour}@uwaterloo.ca

Abstract. We present a parameterized enumeration algorithm for Kemeny Rank Aggregation, the problem of determining an *optimal aggregation*, a total order that is at minimum total *τ-distance* (k_t) from the input multi-set of m total orders (*votes*) over a set of alternatives (*candidates*), where the τ-distance between two total orders is the number of pairs of candidates ordered differently. Our $O^*(4^{\frac{k_t}{m}})$-time algorithm constitutes a significant improvement over the previous $O^*(36^{\frac{k_t}{m}})$ upper bound.

The analysis of our algorithm relies on the notion of locally-optimal aggregations, total orders whose total τ-distances from the votes do not decrease by any single swap of two candidates adjacent in the ordering. As a consequence of our approach, we provide not only an upper bound of $4^{\frac{k_t}{m}}$ on the number of optimal aggregations, but also the first parameterized bound, $4^{\frac{k_t}{m}}$, on the number of locally-optimal aggregations, and demonstrate that it is tight. Furthermore, since our results rely on a known relation to Weighted Directed Feedback Arc Set, we obtain new results for this problem along the way.

1 Introduction

In the general rank aggregation problem, the goal is to find a single preference list that is as close as possible to a multi-set of preference lists, according to a chosen distance measure. The problem dates back to the 18th century [9,11], when it was raised in the context of fair voting protocols in France; since then it has been applied to such areas as computational social choice, planning problems in artificial intelligence [15], bioinformatics [18], and graph drawing [8]. Here we study Kemeny Rank Aggregation [20], where the input preference lists (*votes*) and the output preference list (*optimal aggregation*) are restricted to total orders over the set of elements (*candidates*), the distance between two votes is the number of pairs of candidates ordered differently in the two votes, and the optimal aggregation is at minimum total distance from all votes.

* Supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

F. Dehne, R. Solis-Oba, and J.-R. Sack (Eds.): WADS 2013, LNCS 8037, pp. 512–523, 2013.

Kemeny Rank Aggregation is NP-hard for constant even numbers of votes as small as four [3,8,14,17]; therefore, approximations have been studied [1,8,13,14,23]. Kemeny Rank Aggregation admits a polynomial-time approximation scheme, based on a reduction to the weighted-directed feedback arc set problem (WDFAS) for special complete digraphs [21].

Since approximate solutions to Kemeny Rank Aggregation can violate important properties [12], algorithms to find exact solutions have garnered significant interest. Betzler et al. developed fixed-parameter algorithms with running times of $O(2^n \cdot n^2 m)$ [7], $O(1.53^{k_t} + m^2 n)$ [7] and $O((3k_m+1)!\, k_m \log k_m \cdot mn)$ [6], where n is the number of candidates, m is the number of votes, k_t is the total τ-distance of an optimal aggregation from the votes, and k_m is the maximum pairwise τ-distance of the votes. The idea in the last-mentioned algorithm was later extended to the average pairwise τ-distance of votes, denoted by k_a, and the maximum difference between the positions of a particular candidate in any of the votes, denoted by r_m, yielding bounds of $O(16^{k_a} \cdot (k_a^2 \cdot m + k_a \cdot m^2 \log m \cdot n))$ and $O(32^{r_m} \cdot (r_m^2 \cdot m + r_m \cdot m^2))$ [7]. Simjour [22] considered $\frac{k_t}{m}$ as an average parameter tighter than k_a, and obtained an $O^*(5.823^{\frac{k_t}{m}})$-time algorithm, based on an algorithm for WDFAS in tournaments. Simjour [22] also obtained algorithms of running times $O^*(1.403^{k_t})$ and $O^*(4.829^{k_m})$. Later, a subexponential-time algorithm developed by Alon et al. [2] for WDFAS for tournaments improved the running times with respect to $\frac{k_t}{m}$, k_a, and k_m, to $O(2^{O(\sqrt{\frac{k_t}{m}} \log \frac{k_t}{m})} + n^{O(1)})$ [16]. At about the same time, Karpinski and Schudy [19] reduced Kemeny Rank Aggregation to WDFAS for complete digraphs with arc-weights satisfying the probability constraint (the weights of the arcs (a, b) and (b, a) add up to one). Through an elegant analysis, they obtained an improved running time of $O(2^{O(\sqrt{\frac{k_t}{m}})} + n^{O(1)})$. Though most of the parameterized algorithms for Kemeny Rank Aggregation have benefited from its connection to WDFAS [16,19,22], details of the reductions differ.

Not much improvement (with respect to $\frac{k_t}{m}$) is expected, since an $O(2^{o(\sqrt{\frac{k_t}{m}})} + n^{O(1)})$-time algorithm for Kemeny Rank Aggregation would cause the failure of the Exponential Time Hypothesis [2]. On the other hand, Fernau et al. [16] studied an above-guarantee parameterization of Kemeny Rank Aggregation. The reduction to WDFAS results in an $O(2^{O(k_g \log k_g)} + n^{O(1)})$-time algorithm, where k_g is an above-guarantee version of k_t [10]. For an odd number of votes, the algorithm of Karpinski and Schudy [19] runs in time $O(2^{O(\sqrt{k_g})} + n^{O(1)})$. Again, an $O(2^{o(\sqrt{k_g})} + n^{O(1)})$-time algorithm for Kemeny Rank Aggregation results in the failure of the Exponential Time Hypothesis [16], thus is very unlikely to exist. In addition, Kemeny Rank Aggregation can be reduced to a kernel that includes $2k_t$ votes over at most $2k_t$ candidates [7], and to a partial kernel over at most $\frac{16k_a}{3}$ candidates [4,5].

There are few results on counting and enumeration of optimal aggregations, including those obtainable by adjusting the $O^*(2^n)$-time dynamic programming of Betzler et al. [7] or the subexponential-time algorithm of Karpinski and

Schudy [19] to count the number of optimal aggregations. The only known parameterized bound on the number of optimal aggregations is due to Simjour [22], who gave an $O^*(36^{\frac{k_t}{m}})$-time enumeration algorithm.

Our contributions. Using a refined approach, we improve the running time for enumeration from $O^*(36^{\frac{k_t}{m}})$ [22] to $O^*(4^{\frac{k_t}{m}})$, and show an $4^{\frac{k_t}{m}}$ bound on the number of optimal aggregations. We use the reduction to WDFAS for complete digraphs, exploiting the observation that the arc-weights in all the reduced digraphs satisfy the triangle inequality [23]. Our search tree algorithm, AggSearch, consumes a complete digraph whose arc-weights satisfy the probability and triangle inequality constraints and finds all minimum feedback arc sets of the input graph (sets of arcs whose removal renders the graph acyclic).

The algorithm AggSearch guesses adjacent pairs of minimum feedback arc sets, relying on the fact that all consecutively-ordered vertices in such sets correspond to ($\leq \frac{1}{2}$)-weight arcs. Our algorithm does not use other properties of minimum feedback arc sets; it actually enumerates all locally-minimum feedback arc sets (total orders that are only constrained to have their consecutively-ordered vertices correspond to ($\leq \frac{1}{2}$)-weight arcs). Therefore, our parameterized bound on their number (though restricted to special graph classes) is quite unexpected. Analogously, the bound is carried over to the number of locally-optimal aggregations, defined in Section 2. We are not aware of any parameterized upper bounds on the number of locally-optimal aggregations prior to this bound.

There are instances with 4^k minimum feedback arc sets. Furthermore, all these instances correspond to Kemeny Rank Aggregation instances. Consequently, the upper bounds on the numbers of (locally-) minimum feedback arc sets and (locally-) optimal aggregations are asymptotically tight.

2 Definitions

Complete or partial preference lists over a set of candidates U can be represented as binary relations, namely sets of ordered pairs in $U \times U$, where each ordered pair (x, y) in the relation represents the preference of a candidate x over a candidate y. As a benefit, set operations can be used; for instance, the number of preferences common to two lists π_1 and π_2 can be represented as $\pi_1 \cap \pi_2$. Since we reduce our problem to a graph problem, we also treat graph arcs as ordered pairs and sets of arcs as binary relations that consist of the corresponding ordered pairs.

For a binary relation $\rho \subseteq U \times U$, we use $x <_\rho y$ to denote that $(x, y) \in \rho$, that is, that x is preferred over y. The *reverse* of an ordered pair (x, y), denoted rev*((x, y))*, is the ordered pair (y, x) formed by reversing the first and second elements (the *tail* and *head*, respectively). The preferences opposite to those in a binary relation ρ, its *reverse*, is rev*(ρ)* $= \{(y, x) : (x, y) \in \rho\}$. A binary relation ρ is *transitive* if $w <_\rho x$ and $x <_\rho y$ imply $w <_\rho y$; ρ^+ is the transitive binary relation of minimum cardinality that is a superset of ρ. A binary relation ρ is *acyclic* if $\rho \cap \mathrm{rev}(\rho^+) = \emptyset$, and a *total order* over a set U if it is transitive, for any $x <_\rho y$, x is not equal to y and $y \not<_\rho x$, and for any $x, y \in U$, $x \neq y$, either $x <_\rho y$ or $y <_\rho x$. We use $Tot(U)$ to denote the set of total orders over U.

The problem of KEMENY RANK AGGREGATION is defined in terms of a distance measure that describes the degree to which preference lists differ from each other. The *τ-distance* between $\pi_1 \in \text{Tot}(U)$ and $\pi_2 \in \text{Tot}(U)$, denoted by $\tau(\pi_1, \pi_2)$, is the number of pairs in $\pi_1 - \pi_2$, and by extension, the *τ-distance* between π_1 and a multi-set $\mathcal{I}$ over $\text{Tot}(U)$, denoted by $\tau(\pi_1, \mathcal{I})$, is $\sum_{\pi_2 \in \mathcal{I}} \tau(\pi_1, \pi_2)$.

KEMENY RANK AGGREGATION

Input: a multi-set $\mathcal{I}$ of m total orders (*votes*) in $\text{Tot}(U)$ where U is a set of n elements (*candidates*)
Output: an *optimal aggregation* of $\mathcal{I}$ (a total order $\lambda \in \text{Tot}(U)$ that minimizes $\tau(\lambda, \mathcal{I})$)

We use a well-known reduction to WDFAS on complete digraphs [21], where a *feedback arc set* β for a graph G is a subset of the graph arcs whose removal makes the graph acyclic, with *weight* $w_\beta = \sum_{e \in \beta} w_e$.

WDFAS

Input: an arc-weighted directed graph G
Output: a feedback arc set β for G of minimum weight

We use $MF(V, w)$ to denote the set of all minimum feedback arc sets in a complete digraph G on the vertex set V and with the arc-weight function w.

Feedback arc sets in a complete digraph must have many arcs; each must include a total order. The total orders in $\text{Tot}(V)$, for a complete digraph over vertex set V, are exactly the *minimal* feedback arc sets (sets for which the removal of any arc will result in a cycle in the remaining graph); thus since every minimum weight feedback arc set is minimal, $MF(V, w) \subseteq \text{Tot}(V)$.

An instance $\mathcal{I}$ of KEMENY RANK AGGREGATION is reduced to a complete digraph with arc-weights between zero and one. We define $\mathcal{I}_{(a,b)}$ as $\{\pi \in \mathcal{I} : a <_\pi b\}$.

Observation 1. *A total order $\lambda \in Tot(U)$ is an optimal aggregation of $\mathcal{I}$ if and only if* $\text{rev}(\lambda) \in MF(U, w)$*, where w is the weight function* $w_{(a,b)} = \frac{|\mathcal{I}_{(a,b)}|}{m}$.

Proof. This is a consequence of the fact that the τ-distance between any total order $\pi \in \text{Tot}(U)$ and $\mathcal{I}$ is precisely m times the weight of $\text{rev}(\pi)$ in the complete digraph with vertex set U and the arc-weight function $w_{(a,b)} = \frac{|\mathcal{I}_{(a,b)}|}{m}$. □

The weight function satisfies two useful properties, which will be exploited in the analysis of our algorithm (Section 4). a weight function w over $U \times U$ satisfies the probability constraint if $w_{(a,b)} + w_{(b,a)} = 1$ for all pairs $a, b \in U$; we are using $w_{(a,b)}$ to denote the weight assigned to the pair $(a, b) \in U \times U$.

Observation 2. *[23] The weight function* $w_{(a,b)} = \frac{|\mathcal{I}_{(a,b)}|}{m}$ *satisfies the probability constraint and the triangle inequality.*

We can use the arc-weight function to identify pairs of vertices that might be adjacent in minimum feedback arc sets. An ordered pair (x, y) is *π-adjacent* (or *adjacent* when π is implicit) for a total order $\pi \in \mathrm{Tot}(U)$ if $x <_\pi y$ and there is no $w \in U$ such that $x <_\pi w <_\pi y$. We use $\mathrm{adj}(\pi)$ to denote the binary relation consisting of all π-adjacent ordered pairs. For example, let $U = \{1, 2, 3, 4\}$ and $\lambda \in \mathrm{Tot}(U)$ satisfy $1 <_\lambda 2 <_\lambda 3 <_\lambda 4$. Then, the set of λ-adjacent pairs is $\mathrm{adj}(\lambda) = \{(1, 2), (2, 3), (3, 4)\}$.

For a weight function $\{w_{(a,b)} : a, b \in V\}$, we define the binary relations $w_{\leq c}$ and $w_{\geq c}$ as $\{(a, b) : w_{(a,b)} \leq c\}$ and $\{(a, b) : w_{(a,b)} \geq c\}$, respectively. For any $\lambda \in MF(V, w)$, $\mathrm{adj}(\lambda) \subseteq w_{\leq \frac{1}{2}}$, since if $\mathrm{adj}(\lambda)$ includes an arc $e \notin w_{\leq \frac{1}{2}}$, then $(\lambda - e) \cup \mathrm{rev}(e)$ is a feedback arc set whose weight is smaller than λ's weight, contradicting $\lambda \in MF(V, w)$.

Our fixed-parameter algorithm in Section 4 is not merely an enumeration algorithm for Kemeny Rank Aggregation; it enumerates all locally-optimal total orders, defined as total orders whose total τ-distances do not decrease after changing the order of an adjacent pair [14]. A closer look at total orders resulting from such a change gives rise to the following equivalent definition [14], analogous to which we define locally-minimum feedback arc sets in digraphs.

Definition 1. *A total order $\lambda \in Tot(U)$ is a* locally-optimal aggregation *for an instance $\mathcal{I}$ of m total orders of* Kemeny Rank Aggregation *if* $\mathrm{adj}(\lambda) \subseteq n_{\geq \frac{m}{2}}$ *for the weight function $n_{(a,b)} = |\mathcal{I}_{(a,b)}|$.*

Definition 2. *A feedback arc set β is* locally-minimum *if it is minimal and* $\mathrm{adj}(\beta) \subseteq w_{\leq \frac{1}{2}}$.

A minimal feedback arc set is a locally-minimum feedback arc set if reversing a single arc does not produce a feedback arc set of smaller weight. We use $LF(V, w)$ to denote the set of all locally-minimum feedback arc sets in the complete digraph on the vertex set V and the arc-weight function w.

By the minimality condition, locally-minimum feedback arc sets are forced to be total orders, making them comparable to locally-optimal aggregations.

Observation 3. *A total order $\lambda \in Tot(U)$ is a locally-optimal aggregation for an instance $\mathcal{I}$ of m total orders of* Kemeny Rank Aggregation *if and only if* $\mathrm{rev}(\lambda) \in LF(U, w)$*, for the weight function* $w_{(a,b)} = \frac{|\mathcal{I}_{(a,b)}|}{m}$.

3 Ideas Used in the Algorithm

3.1 Branching Based on a Feedback Arc Set

A brute-force search for adjacent pairs of a $\gamma \in LF(V, w)$ can be very inefficient. We use a minimal feedback arc set β (equivalently, a $\beta \in \mathrm{Tot}(V)$) to speed up the search, and show in Theorem 1 that for any β, AggSearch$(V, w, \beta, \mathrm{rev}(\beta), \emptyset, \emptyset)$ produces every $\gamma \in LF(V, w)$ in the leaves of its search tree. The weight of β affects only the running time: the search tree has at most 4^{w_β} leaves and is computed in times $O(n^\mu \cdot 4^{w_\beta})$, where μ denotes the exponent of matrix multiplication. As a result, $|LF(V, w)| \leq 4^k$, where k is the weight of a minimum feedback arc set in G.

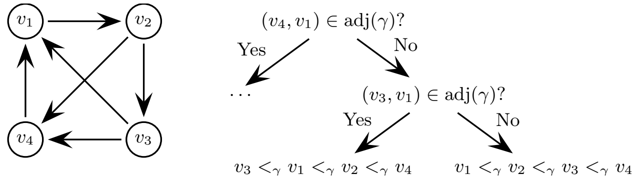

Fig. 1. The first toy example and a decision tree based on adjacent pairs

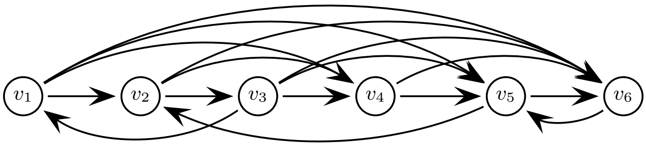

Fig. 2. The second toy example

To give a sense of how branching on adjacent pairs prunes the search space, we consider the graph shown in Fig. 1, along with the decision tree implicit in the algorithm and resulting γ's. For clarity, we have omitted arc weights and have drawn only the arcs in $w_{\leq\frac{1}{2}}$, which must include $\mathrm{adj}(\gamma)$ for any $\gamma \in LF(V, w)$. If $(v_4, v_1) \notin \mathrm{adj}(\gamma)$, v_4 must be ordered last in γ, as no other arc of the form $(v_4, *)$ will remain to be placed in $\mathrm{adj}(\gamma)$. Then, either $(v_3, v_1) \in \mathrm{adj}(\gamma)$, or v_1 must be ordered first in γ, since no arc $(*, v_1)$ will remain. Similar arguments are used to determine the rest of the arcs in $\mathrm{adj}(\gamma)$.

The search for adjacent pairs not in β, $\alpha = \mathrm{adj}(\gamma) - \beta$, is easy if the weight of β is small. The reverse arcs of $w_{\leq\frac{1}{2}} - \beta$, each of which has a weight of at least $\frac{1}{2}$, are all in β. Since the weight of β is small, the number of such arcs must be small, and hence the number of arcs in α, of which $w_{\leq\frac{1}{2}} - \beta$ is a superset.

Still, there are possibly many pairs in $w_{\leq\frac{1}{2}} \cap \beta$ from which to choose the remaining arcs, i.e. $\mathrm{adj}(\gamma) \cap \beta$. In Section 3.2, we will show that all the arcs in γ will be fixed once we figure out those located in a certain region which depends on α. A brute-force search of the region is not very costly, as the triangle inequality on the arc weights ensures that the size of the region is linear in the weight of β. The combination of α and the set of arcs of γ in the region form a concise representation of γ in terms of β, the *β-representation of* γ.

3.2 $\boldsymbol{\beta}$-Representations

We use a small example to showcase the basic idea of our representation for a $\gamma \in LF(V, w)$ in Fig. 2: we choose a $\beta \in \mathrm{Tot}(V)$, and draw the vertices from left to right in the order of β (only the arcs in $w_{\leq\frac{1}{2}}$ are shown). When $\alpha = \emptyset$, $\mathrm{adj}(\gamma)$ contains no arcs outside β and hence must adhere to the order in β, that is, $\gamma = \beta$.

For $\alpha = \{(v_5, v_2)\}$, we can be sure that v_1 is ordered first and v_6 is ordered last in γ, but we do not know whether either v_3 or v_4 is ordered before v_2 and v_5. The order will be fixed once we know whether $v_3 <_\gamma v_2$ or $v_2 <_\gamma v_3$, and whether $v_4 <_\gamma v_2$ or $v_2 <_\gamma v_4$. For example, if both $v_3 <_\gamma v_2$ and $v_4 <_\gamma v_2$, then $v_3 <_\gamma v_4$ since otherwise (v_4, v_3) had to be in $\alpha = \mathrm{adj}(\gamma) - \beta$ as well.

Fortunately, not many vertices can be in the same situation as v_3 and v_4. By the triangle inequality, the weight of (v_2, v_3) plus the weight of (v_3, v_5), and in general $w_{(v_2,x)} + w_{(x,v_5)}$ for any vertex x satisfying $v_2 <_\beta x <_\beta v_5$, is at least the weight of (v_2, v_5). On the other hand, $(v_2, v_5) \in \beta$ and $w_{(v_2,v_5)} \geq \frac{1}{2}$ since (v_5, v_2) was initially assumed to be in $\alpha \subseteq w_{\leq \frac{1}{2}} - \beta$. Consequently, the weight of β is at least $\sum_{v_2 <_\beta x <_\beta v_5} (w_{(v_2,x)} + w_{(x,v_5)}) \geq \sum_{v_2 <_\beta x <_\beta v_5} w_{(v_2,v_5)} \geq |\{x : v_2 <_\beta x <_\beta v_5\}| \cdot \frac{1}{2}$. Thus, the number of vertices whose relative orders in γ with respect to v_2 must be determined (like v_3 and v_4) is at most twice the weight of β. We will see how the bounded number of decisions is generalized to arbitrary α's.

The β-representation of $\gamma \in LF(V, w)$ consists of two parts. The first part, α, is the set $\mathrm{adj}(\gamma) - \beta$. For a precise definition of the second part, we define a few terms. An unordered pair $\{x, y\}$ is a β-*internal pair* of $(a, b) \in \mathrm{rev}(\beta)$ if $x = a$ or $x = b$, and $b <_\beta y <_\beta a$. We use $\mathrm{IP}_\beta(e)$ to denote the set of β-internal pairs of $e \in \mathrm{rev}(\beta)$, and by extension, we use $\mathrm{IP}_\beta(\rho)$ for a binary relation $\rho \subseteq \mathrm{rev}(\beta)$ to denote $\bigcup_{e \in \rho} \mathrm{IP}_\beta(e)$. A binary relation $\rho \in \mathrm{Tot}(U)$ *restricted* to a set of unordered pairs P, denoted as $\rho|P$, is the new binary relation $\{(x, y) \in \rho : \{x, y\} \in P\}$.

Thus, for $\beta \in \mathrm{Tot}(\{v_1, \ldots, v_5\})$ and $v_1 <_\beta v_2 <_\beta v_3 <_\beta v_4 <_\beta v_5$, $\mathrm{IP}_\beta((v_5, v_3)) = \{\{v_3, v_4\}, \{v_5, v_4\}\}$ and $\mathrm{IP}_\beta(\{(v_5, v_3), (v_4, v_1)\}) = \{\{v_3, v_4\}, \{v_5, v_4\}, \{v_1, v_2\}, \{v_4, v_2\}, \{v_1, v_3\}, \{v_4, v_3\}\}$. For $v_1 <_\gamma v_2 <_\gamma v_4 <_\gamma v_5 <_\gamma v_3$, the restriction of $\gamma \in \mathrm{Tot}(\{v_1, \ldots, v_5\})$ to $\mathrm{IP}_\beta((v_5, v_3))$ is $\gamma|\mathrm{IP}_\beta((v_5, v_3)) = \{(v_4, v_3), (v_4, v_5)\}$.

Definition 3. *The* β-representation *of* $\gamma \in LF(V, w)$*, for some* $\beta \in Tot(V)$*, is* (α, δ) *where* $\alpha = \mathrm{adj}(\gamma) - \beta$ *and* $\delta = \gamma|\mathrm{IP}_\beta(\alpha)$*.*

A locally-minimum feedback arc set can be efficiently reconstructed from its β-representation for an arbitrary $\beta \in \mathrm{Tot}(V)$:

Lemma 1. *If* (α, δ) *is the* β*-representation of* $\gamma \in LF(V, w)$ *for a* $\beta \in Tot(V)$*, then* $\gamma = \beta - \mathrm{rev}((\alpha \cup \delta)^+) \cup (\alpha \cup \delta)^+$*.*

Proof. Since $\beta - \mathrm{rev}((\alpha \cup \delta)^+) \cup (\alpha \cup \delta)^+$ is a total order, it suffices to show that its two subsets $\beta - \mathrm{rev}((\alpha \cup \delta)^+)$ and $(\alpha \cup \delta)^+$ are in γ. The latter is true since α and δ are defined to be subsets of γ and γ is transitive. We prove the former by showing that every $(x, y) \in \gamma - \beta$ is in $(\alpha \cup \delta)^+$. Since γ and β are total orders, $\beta - \gamma$ is a subset of $\mathrm{rev}((\alpha \cup \delta)^+)$, and thus, $\beta - \mathrm{rev}((\alpha \cup \delta)^+)$ is a subset of γ. The proof is by strong induction: assuming the claim is true for every $(x', y') \in \gamma - \beta$ with $y <_\beta y'$, we prove the claim for (x, y).

Drawing the vertices in V on a horizontal line and ordered from left to right consistent with their order of β, suppose that $x = w_1 <_\gamma w_2 <_\gamma \ldots <_\gamma w_\ell = y$, with $\ell \geq 2$, and $(w_i, w_{i+1}) \in \mathrm{adj}(\gamma)$ for all $1 \leq i < \ell$. Fig. 3 demonstrates an example where $w_7 <_\beta w_8 <_\beta \ldots <_\beta w_4 <_\beta w_2$. In traversing the vertices in

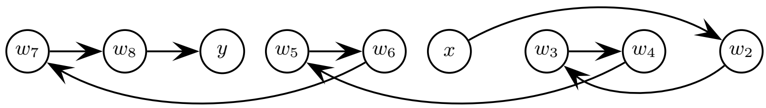

Fig. 3. An example of the case $x <_\gamma y$ and $y <_\beta x$, where the vertices are shown in the order of β from left to right and the ordered pairs in $\text{adj}(\gamma)$ are presented as arcs.

order from w_1 to w_ℓ through the arcs in $\text{adj}(\gamma)$, we use arcs in $\alpha = \text{adj}(\gamma) - \beta$ when we go from right to left; w_ℓ must be to the left of w_1, since $y <_\beta x$. To reach $y = w_\ell$ from $x = w_1$, we must traverse at least one right-to-left arc ending up at y or a vertex to the left of y ((w_6, w_7) in Fig. 3). Since $(x, y) \in \gamma - \beta$, there must exist some $1 \leq t < \ell$ such that $(w_t, w_{t+1}) \in \alpha$ with $w_{t+1} \leq_\beta y <_\beta w_t$. When $w_{t+1} \neq y$, $\{y, w_{t+1}\} \in \text{IP}_\beta(\alpha)$.

We now prove the induction step. If $(w_t, w_{t+1}) = (x, y)$, then $(x, y) \in \alpha$, and hence $(x, y) \in (\alpha \cup \delta)^+$. Otherwise, we show that $(w_{t+1}, y) \in (\alpha \cup \delta)^+$ if $w_{t+1} \neq y$ and $(x, w_t) \in (\alpha \cup \delta)^+$ if $x \neq w_t$. Together with $(w_t, w_{t+1}) \in \alpha$, these result in $(x, y) \in (\alpha \cup \delta)^+$, as needed to complete the proof.

We first prove that $(w_{t+1}, y) \in (\alpha \cup \delta)^+$ if $w_{t+1} \neq y$. As mentioned above, when $w_{t+1} \neq y$, $\{y, w_{t+1}\}$ is in $\text{IP}_\beta(\alpha)$. Since γ orders w_{t+1} before y, $(w_{t+1}, y) \in \gamma|\text{IP}_\beta(\alpha) = \delta \subseteq (\alpha \cup \delta)^+$.

Next, considering the relative orders of w_t and x, we prove that $(x, w_t) \in (\alpha \cup \delta)^+$ if $x \neq w_t$. For the case in which $w_t <_\beta x$, since γ orders x before w_t, $(x, w_t) \in \gamma - \beta$; therefore, $(x, w_t) \in (\alpha \cup \delta)^+$ by the induction hypothesis. If instead $x <_\beta w_t$, then, $w_{t+1} < x < w_t$, and hence $\{x, w_t\} \in \text{IP}_\beta(\alpha)$. Since γ orders x before w_t, $(x, w_t) \in \gamma|\text{IP}_\beta(\alpha) = \delta \subseteq (\alpha \cup \delta)^+$. □

4 Our Results

Our search tree algorithm AggSearch, shown in Algorithm 1, uses an input total order β to compute every $\gamma \in LF(V, w)$ through recursive construction of its β-representation (α, δ). The *β-length* of an arc $(a, b) \in \text{rev}(\beta)$, used in the algorithm, is the number of vertices in $\{y : b <_\beta y <_\beta a\}$. A binary relation ρ is an *ordering of* a set of unordered pairs P if both $\rho = \rho|P$ and $|\rho| = |P|$; thus, δ is an ordering of $\text{IP}_\beta(\alpha)$.

Algorithm AggSearch uses an auxiliary parameter σ, initialized to $\text{rev}(\beta)$, which contains the subset of $\text{rev}(\beta)$ for which inclusion in α has not yet been decided. For α to be part of the β-representation of some $\gamma \in LF(V, w)$, the arcs in α must be in $w_{\leq \frac{1}{2}}$, since $\alpha = \text{adj}(\gamma) - \beta$ is a subset of $\text{adj}(\gamma)$ and $\text{adj}(\gamma)$ must be in $w_{\leq \frac{1}{2}}$. Thus, no further arcs are added to α once $\sigma \cap w_{\leq \frac{1}{2}}$ becomes empty (lines 1-5). By that time, δ is an ordering of $\text{IP}_\beta(\alpha)$, since for each arc e inserted in α, all possible orderings of $\text{IP}_\beta(e)$ are added to δ. Hence, the algorithm stops adding arcs to δ as well. Due to Lemma 1, if α and δ now form an β-representation for a $\gamma \in LF(V, w)$, γ must be equal to $(\beta - \text{rev}((\alpha \cup \delta)^+)) \cup (\alpha \cup \delta)^+$. Thus, the algorithm checks if this formula produces a locally-minimum feedback arc

Algorithm 1: AggSearch

Require : vertex set V, weight function w, $\beta \in \mathrm{Tot}(V)$, and $\sigma, \alpha, \delta \subseteq V \times V$

1 **if** $\sigma \cap w_{\leq \frac{1}{2}} = \emptyset$ **then**

2 $\quad \gamma \leftarrow (\beta - \mathrm{rev}((\alpha \cup \delta)^+)) \cup (\alpha \cup \delta)^+$;

3 $\quad$ **if** $\gamma \in LF(V, w)$ **then return** $\{\gamma\}$;

4 $\quad$ **else return** $\emptyset$;

5 **end**

6 **else**

7 $\quad$ Select $(u, v) \in \sigma \cap w_{\leq \frac{1}{2}}$ of maximum β-length;

8 $\quad \sigma \leftarrow \sigma - \{(u, v)\}$;

9 $\quad LF \leftarrow \textsc{AggSearch}(V, w, \beta, \sigma, \alpha, \delta)$;

10 $\quad \alpha \leftarrow \alpha \cup \{(u, v)\}$;

11 $\quad P \leftarrow \{x : u <_\beta x <_\beta v\}$;

12 $\quad \sigma \leftarrow \sigma - \bigcup_{x \in P} \{(u, x), (x, v)\}$;

13 $\quad L \leftarrow \{x \in P : x <_\delta u \text{ or } x <_\delta v\}$;

14 $\quad R \leftarrow \{x \in P : u <_\delta x \text{ or } v <_\delta x\}$;

15 $\quad$ **foreach** $L \subseteq A \subseteq P - R$ **do**

16 $\quad\quad \delta' \leftarrow \delta \cup \bigcup_{x \in A} \{(x, u), (x, v)\} \cup \bigcup_{x \in P - A} \{(u, x), (v, x)\}$;

17 $\quad\quad LF \leftarrow LF \cup \textsc{AggSearch}(V, w, \beta, \sigma, \alpha, \delta')$;

18 $\quad$ **end**

19 $\quad$ **return** LF;

20 **end**

set (line 3). If not, (α, δ) is neither a β-representation for any $\gamma \in LF(V, w)$, nor can it be made into one by adding arcs to α and δ.

For each arc (u, v) in $\sigma \cap w_{\leq \frac{1}{2}}$, the algorithm branches on whether $(u, v) \in \alpha$, removing the arc from σ once the decision is made. In the branch in which $(u, v) \in \alpha$ (lines 10-18), we can also remove all arcs in $\bigcup_{x \in P} \{(u, x), (x, v)\}$, $P = \{x : u <_\beta x <_\beta v\}$ from σ: as (u, v) is in $\mathrm{adj}(\gamma)$ and in γ only one vertex is ordered immediately after u and only one vertex is ordered immediately before v, none of the arcs sharing a head or tail with (u, v) can be in $\mathrm{adj}(\gamma) \supseteq \alpha$. Further branching occurs on the subset $A = \{x \in P : (x, u) \in \gamma\}$ of vertices in P (lines 15-18). The orderings of the vertices in P with respect to u and v, determined by A, are essential in determining $\delta = \gamma | \mathrm{IP}_\beta(\alpha)$ in the β-representation of γ.

We do not want to branch over a pair more than once; one strategy is to consider arcs in order of β-length. Without this selection criterion, if in Fig. 4 (with $\sigma \cap w_{\leq \frac{1}{2}}$ including (u_1, v_1) and (u_2, v_1) such that $v_1 <_\beta u_1 <_\beta u_2$) at line 7 the algorithm selected $(u_1, v_1) \in \sigma \cap w_{\leq \frac{1}{2}}$ to be excluded from α, then further down the search tree, the algorithm could select $(u_2, v_1) \in \sigma \cap w_{\leq \frac{1}{2}}$ to be included in α. This would result in branching twice on (u_1, v_1), once for membership in α and once, at line 15, to decide whether $u_1 <_\delta v_1$ or $v_1 <_\delta u_1$.

Constraining A to include L and exclude R at line 15 avoids another duplicate branching, as otherwise the algorithm could decide on relative orderings of vertices in L and R with respect to u and v after the orderings were already

Fig. 4. Situations in which duplicate decisions could be made over a pair

fixed in δ. In Fig. 4 (where $\sigma \cap w_{\leq \frac{1}{2}}$ includes (u_1, v_1) and (u_2, v_2) such that $v_1 <_\beta v_2 <_\beta u_1 <_\beta u_2$), if $(u_1, v_1) \in \sigma$ is inserted in α at line 10, the algorithm needs to decide whether to include v_2 in A (a decision on the ordering of $\{u_1, v_2\}$) at line 15. Without the constraint on A, $(u_2, v_2) \in \sigma$ could then be inserted in α, necessitating a second decision on the ordering of $\{u_1, v_2\}$ (whether to include u_1 in A).

Removal of the same-head and same-tail arcs from σ (line 12), ordering the arcs in σ in their β-lengths (line 7), and constraining A to include L and exclude R (line 15) all result in less branching.

Theorem 1. *Given a complete digraph on a vertex set V and arc weights $\{w_{(a,b)} : a, b \in V\}$ and $\beta \in Tot(V)$,* $\textsc{AggSearch}(V, w, \beta, \mathrm{rev}(\beta), \emptyset, \emptyset)$ *returns $LF(V, w)$ in time $O(|V|^\mu \cdot 4^{w_\beta})$, where $\mu < 2.376$ denotes the exponent of matrix multiplication. Furthermore, $|LF(V, w)| \leq 4^{w_\beta}$.*

Proof. Due to space limitations, we provide only a high-level idea of the proof. We prove by strong induction on the cardinality of $\sigma \cap w_{\leq \frac{1}{2}}$ that:

(1) For any ordering δ of $\mathrm{IP}_\beta(\alpha)$, $\textsc{AggSearch}(V, w, \beta, \sigma, \alpha, \delta)$ returns every γ in $LF_{(\beta,\sigma,\alpha,\delta)} = \{\gamma \in LF(V, w) : \alpha \subseteq \mathrm{adj}(\gamma) - \beta \subseteq \alpha \cup \sigma, \text{ and } \delta \subseteq \gamma\}$
(2) If $\sigma \cup \delta$ includes an ordering of $\mathrm{IP}_\beta(\sigma \cap w_{\leq \frac{1}{2}})$, $\textsc{AggSearch}(V, w, \beta, \sigma, \alpha, \delta)$ produces a search tree with at most $4^{w_{\mathrm{rev}(\sigma)}}$ leaves.

Making use of the fact that arcs in $\sigma \cap w_{\leq \frac{1}{2}}$ are selected in order of their β-lengths (line 7), we show that δ is an ordering of $\mathrm{IP}_\beta(\alpha)$ and $\sigma \cup \delta$ includes an ordering of $\mathrm{IP}_\beta(\sigma \cap w_{\leq \frac{1}{2}})$ in all recursive calls originating from $\textsc{AggSearch}(V, w, \beta, \mathrm{rev}(\beta), \emptyset, \emptyset)$; from this we can show $LF(V, w) = LF_{(\beta,\mathrm{rev}(\beta),\emptyset,\emptyset)}$ is returned upon the production of at most 4^{w_β} nodes.

We associate each node v in the search tree with the cost of steps 7-9 or 10-17 performed just before the creation of v plus the cost of steps 1-5 performed at the execution of v. The dominant part is the computation of the transitive closure $(\alpha \cup \delta)^+$ using matrix multiplication at line 2. The time for a node is thus in $O(|V|^\mu)$, yielding $O(|V|^\mu \cdot 4^{w_\beta})$ time overall. □

By Observation 3, Kemeny Rank Aggregation instances have at most $4^{\frac{k_t}{m}}$ locally-optimal aggregations.

Corollary 1. *Given a multi-set $\mathcal{I}$ of m total orders in $Tot(U)$ and a total order λ at τ-distance k_λ of $\mathcal{I}$, the set of all locally-optimal aggregations for $\mathcal{I}$ can be found in time $O(m \cdot |U| + 4^{\frac{k_\lambda}{m}} \cdot |U|^\mu)$. Furthermore, $\mathcal{I}$ has at most $4^{\frac{k_t}{m}}$ locally-optimal aggregations, where k_t denotes the minimum τ-distance from $\mathcal{I}$.*

Although $MF(V, w)$ is generally a (small) subset of $LF(V, w)$, the two sets are equal for certain instances, for which Theorem 1's upper bound is tight:

Theorem 2. *For any set $V = \{v_1, v_2, \ldots, v_n\}$ of even cardinality, there exists a weight function w over $V \times V$ that satisfies the triangle inequality and the probability constraint such that $|MF(V, w)| = 4^k$, where k denotes the weight of a minimum feedback arc set in $MF(V, w)$.*

Proof. We consider the following weight function:

$$w_{(v_i,v_j)} = \begin{cases} 0 & i+1 < j \text{ or } (i+1=j \text{ and } i \text{ is even}) \\ \frac{1}{2} & i+1=j \text{ and } i \text{ is odd} \\ 1 - w_{(v_j,v_i)} & \text{otherwise} \end{cases}$$

It is not hard to see that any minimum feedback arc set must contain all weight-0 arcs. Therefore, elements of $MF(V, w)$ differ only in the ordering of the remaining pairs. All total orders of $\{\{v_1, v_2\}, \{v_3, v_4\}, \ldots \{v_{n-1}, v_n\}\}$ are of equal weight. Since there are $2^{\frac{n}{2}}$ such total orders, each of weight $k = \frac{n}{4}$, the cardinality of $MF(V, w)$ is $2^{2k} = 4^k$ for this instance. □

As there are Kemeny Rank Aggregation instances that reduce to the instances in the proof of Theorem 2, the lower bound also applies to optimal aggregations; the proof is omitted due to space limitations.

Theorem 3. *For any even number m, there exists a multi-set $\mathcal{I}$ of m total orders that has $4^{\frac{k_t}{m}}$ optimal aggregations, where k_t denotes the τ-distance of an optimal aggregation from $\mathcal{I}$.*

5 Concluding Remarks

We gave a tight upper bound on the number of (locally-) optimal aggregations. We emphasize that a $f(\frac{k_t}{m})n^{O(1)}$ upper bound on the number of locally-optimal aggregations is surprising. One future direction for research is the search for a new parameter that is more tuned to the complexity of enumerating all optimal aggregations, rather than locally-optimal aggregations.

References

1. Ailon, N., Charikar, M., Newman, A.: Aggregating inconsistent information: Ranking and clustering. J. ACM 55(5), 1–27 (2008)
2. Alon, N., Lokshtanov, D., Saurabh, S.: Fast fast. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part I. LNCS, vol. 5555, pp. 49–58. Springer, Heidelberg (2009)
3. Bartholdi, J.J., Tovey, C.A., Trick, M.A.: The computational difficulty of manipulating an election. Social Choice and Welfare 6(3), 227–241 (1989)
4. Betzler, N., Bredereck, R., Chen, J., Niedermeier, R.: Studies in computational aspects of voting - a parameterized complexity perspective. In: Bodlaender, H.L., Downey, R., Fomin, F.V., Marx, D. (eds.) Fellows Festschrift 2012. LNCS, vol. 7370, pp. 318–363. Springer, Heidelberg (2012)

5. Betzler, N., Bredereck, R., Niedermeier, R.: Partial kernelization for rank aggregation: Theory and experiments. In: Raman, V., Saurabh, S. (eds.) IPEC 2010. LNCS, vol. 6478, pp. 26–37. Springer, Heidelberg (2010)
6. Betzler, N., Fellows, M.R., Guo, J., Niedermeier, R., Rosamond, F.A.: Fixed-parameter algorithms for kemeny scores. In: Fleischer, R., Xu, J. (eds.) AAIM 2008. LNCS, vol. 5034, pp. 60–71. Springer, Heidelberg (2008)
7. Betzler, N., Fellows, M.R., Guo, J., Niedermeier, R., Rosamond, F.A.: Fixed-parameter algorithms for Kemeny rankings. Theor. Comput. Sci. 410(45), 4554–4570 (2009)
8. Biedl, T.C., Brandenburg, F., Deng, X.: On the complexity of crossings in permutations. Discrete Mathematics 309(7), 1813–1823 (2009)
9. Borda, J.: Mémoire sur les élections au scrutin. Histoire de l'Académie Royale des Sciences (1781)
10. Chen, J., Liu, Y., Lu, S., O'Sullivan, B., Razgon, I.: A fixed-parameter algorithm for the directed feedback vertex set problem. J. ACM 55(5) (2008)
11. Condorcet, M.: Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix. L'imprimerie royale (1785)
12. Conitzer, V., Davenport, A., Kalagnanam, J.: Improved bounds for computing Kemeny rankings. In: AAAI 2006: Proc. of the 21st Nat. Conf. on Artificial Intelligence, vol. 1, pp. 620–626 (2006)
13. Coppersmith, D., Fleischer, L., Rudra, A.: Ordering by weighted number of wins gives a good ranking for weighted tournaments. In: SODA 2006: Proc. of the 17th Annual ACM-SIAM Symp. on Discrete Algorithms, pp. 776–782 (2006)
14. Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank aggregation methods for the web. In: WWW 2001: Proc. of the 10th Int. Conf. on World Wide Web, pp. 613–622 (2001)
15. Ephrati, E., Rosenschein, J.S.: The Clarke tax as a consensus mechanism among automated agents. In: AAAI 1991: Proc. of the 9th Nat. Conf. on Artificial Intelligence, vol. 1, pp. 173–178 (1991)
16. Fernau, H., Fomin, F.V., Lokshtanov, D., Mnich, M., Philip, G., Saurabh, S.: Ranking and drawing in subexponential time. In: Iliopoulos, C.S., Smyth, W.F. (eds.) IWOCA 2010. LNCS, vol. 6460, pp. 337–348. Springer, Heidelberg (2011)
17. Hemaspaandra, E., Spakowski, H., Vogel, J.: The complexity of Kemeny elections. Theor. Comput. Sci. 349(3), 382–391 (2005)
18. Jackson, B.N., Schnable, P.S., Aluru, S.: Consensus genetic maps as median orders from inconsistent sources. IEEE/ACM Trans. Comput. Biol. Bioinformatics 5(2), 161–171 (2008)
19. Karpinski, M., Schudy, W.: Faster algorithms for feedback arc set tournament, Kemeny rank aggregation and betweenness tournament. In: Cheong, O., Chwa, K.-Y., Park, K. (eds.) ISAAC 2010, Part I. LNCS, vol. 6506, pp. 3–14. Springer, Heidelberg (2010)
20. Kemeny, J.G.: Mathematics without numbers. Daedalus 88, 575–591 (1959)
21. Kenyon-Mathieu, C., Schudy, W.: How to rank with few errors. In: STOC 2007: Proc. of the 39th Annual ACM Symp. on Theory of Computing, pp. 95–103 (2007)
22. Simjour, N.: Improved parameterized algorithms for the Kemeny aggregation problem. In: Chen, J., Fomin, F.V. (eds.) IWPEC 2009. LNCS, vol. 5917, pp. 312–323. Springer, Heidelberg (2009)
23. Van Zuylen, A., Williamson, D.P.: Deterministic pivoting algorithms for constrained ranking and clustering problems. Mathematics of Operations Research 34(3), 594–620 (2009)