Faster Algorithms for Feedback Arc Set Tournament, Kemeny Rank Aggregation and Betweenness Tournament*

Marek Karpinski[†] Warren Schudy[‡]

Abstract

We study fixed parameter algorithms for three problems: Kemeny rank aggregation, feedback arc set tournament, and betweenness tournament. For Kemeny rank aggregation we give an algorithm with runtime $O^*(2^{O(\sqrt{OPT})})$, where n is the number of candidates, $OPT \leq \binom{n}{2}$ is the cost of the optimal ranking, and $O^*(\cdot)$ hides polynomial factors. This is a dramatic improvement on the previously best known runtime of $O^*(2^{O(OPT)})$. For feedback arc set tournament we give an algorithm with runtime $O^*(2^{O(\sqrt{OPT})})$, an improvement on the previously best known $O^*(OPT^{O(\sqrt{OPT})})$ Alon et al. [2009]. For betweenness tournament we give an algorithm with runtime $O^*(2^{O(\sqrt{OPT/n})})$, where n is the number of vertices and $OPT \leq \binom{n}{3}$ is the optimal cost. This improves on the previously known $O^*(OPT^{O(OPT^{1/3})}$ Saurabh [2009]), especially when OPT is small. Unusually we can solve instances with OPT as large as $n(\log n)^2$ in polynomial time!

Keywords: Kemeny rank aggregation, Feedback arc set tournament, Fixed parameter tractability, Betweenness tournament

1 Introduction

Suppose you ran a chess tournament, everybody played everybody (a.k.a. round robin) and you wanted to use the results to rank everybody. Unless you were really

*A preliminary version of this work appeared in version 1 of the arXiv preprint Karpinski and Schudy [2009].

[†]Dept. of Computer Science and the Hausdorff Center for Mathematics, University of Bonn. Parts of this work done while visiting Microsoft Research. Email: marek@cs.uni-bonn.de

[‡]Dept. of Computer Science, Brown University. Parts of this work done while visiting University of Bonn. Email: ws@cs.brown.edu

lucky, the results would not be acyclic, so you could not just sort the players by who beat whom. A natural objective is to find a ranking that minimizes the number of upsets, where an upset is a pair of players where the player ranked lower in the ranking beat the player ranked higher. Minimizing the number of upsets is called *feedback arc set problem* on tournaments (FAST). The complementary problem of *maximizing* the number of pairs that are *not upsets* is called the *maximum acyclic subgraph problem* on tournaments. These problems are NP-hard [Ailon et al., 2008, Alon, 2006, Charbit et al., 2007] (see also [Conitzer, 2006]), but a polynomial-time approximation scheme (PTAS) Mathieu and Schudy [2007] is known.

In statistics and psychology, one motivation is *ranking by paired comparisons* [Slater, 1961]: here, you wish to sort some set by some objective but you do not have access to the objective, only a way to compare a pair and see which is greater; for example, determining people's preferences for types of food. This problem attracted computational attention as early as 1961 [Slater, 1961] (for comparison Hoare published quicksort the same year). Feedback arc set tournament and closely related problems have also been used in machine learning [Cohen et al., 1999, Ailon et al., 2008].

The FAST problem can be generalized to a problem we call *weighted* FAST, sometimes known as *feedback arc set with probability constraints*. The input is a complete directed graph with arc weights $\{w_{uv}\}_{u,v}$ with $w_{uv} + w_{vu} = 1$ for every pair of vertices u, v. In other words a weighted FAST instance is a convex combination of unweighted FAST instances.

We study the *parameterized complexity* of this problem, in particular the parameter OPT, the cost of an optimal ranking. We use the notation $O^*(\cdot)$ to hide factors that are polynomial in the input size, that is $f(I) \in O^*(g(I))$ iff $f(I) \leq g(I)|I|^c$ for some $c > 0$ and all sufficiently large inputs I. The first fixed-parameter algorithms for feedback arc set tournament had runtime $O^*(2^{O(OPT)})$ and later algorithms made a dramatic improvement to $O^*(OPT^{O(\sqrt{OPT})})$ Alon et al. [2009]. We improve this to $O^*(2^{O(\sqrt{OPT})})$.

Theorem 1. *There exists a deterministic parameterized subexponential algorithm for weighted* FAST *with runtime* $2^{O(\sqrt{OPT})} + n^{O(1)}$*. A variant of the algorithm uses* $OPT^{O(\sqrt{OPT})} + n^{O(1)}$ *time and* $n^{O(1)}$ *space.*

The *Exponential time hypothesis* (ETH) Impagliazzo and Paturi [2001] is that 3-SAT cannot be solved in time $2^{o(\text{number of variables})}$. We also give a matching lower bound assuming the ETH:

Theorem 2. *There does not exist a parameterized algorithm for weighted* FAST *with runtime* $O^*(2^{o(\sqrt{OPT})})$ *unless the exponential time hypothesis Impagliazzo and Paturi [2001] is false.*

We leave open the possibility that *unweighted* FAST may admit an exact algorithm with runtime $O^*(2^{o(\sqrt{OPT})})$.

We note that independently of this work Uri Feige Feige [2009] gave an unrelated algorithm for *unweighted* FAST matching Theorem 1.

An important application of weighted feedback arc set tournament is *rank aggregation.* Frequently, one has access to several rankings of objects of some sort, such as search engine outputs [Dwork et al., 2001], and desires to aggregate the input rankings into a single output ranking that is similar to all of the input rankings: it should have minimum average distance from the input rankings, for some notion of distance. This ancient problem was already studied in the context of voting by [Borda, 1781] and [Condorcet, 1785] in the 18^{th} century, and has aroused renewed interest recently [Dwork et al., 2001, Conitzer et al., 2006]. A natural notion of distance is the number of pairs of vertices that are in different orders, which is known as the Kendall-Tau distance. This defines the *Kemeny rank aggregation* problem (KRA) [Kemeny, 1959, Kemeny and Snell, 1962]. This choice yields a maximum likelihood estimator for a certain naïve Bayes model [Young, 1995]. This problem is NP-hard [Bartholdi et al., 1989], even with only four voters [Dwork et al., 2001], and has a PTAS Mathieu and Schudy [2007].

We denote the *average* distance between the optimal ranking and the input rankings by $OPT \leq \binom{n}{2}$. Two parameters have attacted the bulk of the study: OPT and the average Kendall-Tau distance between the input rankings. It is easy to see (triangle inequality) that these two parameters are within a constant factor of each other, so these parameters give equivalent runtimes up to constants in the exponent. All previous work give algorithms with runtime $O^*(2^{O(OPT)})$ Betzler et al. [2009]. There is a standard reduction from Kemeny rank aggregation to weighted FAST Ailon et al. [2008], Coppersmith et al. [2006], Mathieu and Schudy [2007], so we improve the best known parameterized algorithm for KRA dramatically to $O^*(2^{O(\sqrt{OPT})})$ as a corollary of our Theorem 1.

Corollary 1. *Let n be the number of candidates and $OPT \leq \binom{n}{2}$ the optimum value. There exists a deterministic parameterized subexponential algorithm for Kemeny Rank Aggregation with runtime and space $2^{O(\sqrt{OPT})} + n^{O(1)}$. A variant uses $OPT^{O(\sqrt{OPT})} + n^{O(1)}$ time and $n^{O(1)}$ space.*

Some other paramters have attracted attention. The parameter of maximum Kendall-Tau distance has been studied but yield bounds no tighter (up to constants in the exponent) than is known for the average Kendall-Tau distance Betzler et al. [2009]. Another parameter is the maximum $r_{\max}$, over candidates c and pairs of voters v_1, v_2, of the absolute difference between the rank of c in v_1 and v_2. The best runtime known is $O^*(2^{O(r_{\max})})$ Betzler et al. [2009].

In the Betweenness problem we are given a ground set of *vertices* and a set of *betweenness constraints* involving 3 vertices and a *designated* vertex among them. The objective function of a ranking of the elements is the number of betweenness constraints for which the designated vertex is not between the other two vertices. The goal is to *minimize* the objective function. For the status of the general Betweenness problem, see e.g. Opatrny [1979], Chor and Sudan [1998], Ailon and Alon [2007], Charikar et al. [2009]. We refer to the Betweenness problem in tournaments, that is in instances with a constraint for every triple of vertices, as the BetweennessTour problem (see Ailon and Alon [2007]). This problem is NP-hard Ailon and Alon [2007] and has a recently discovered polynomial-time approximation scheme Karpinski and Schudy [2009]. We study its parameterized complexity.

Theorem 3. *There exists a randomized parameterized subexponential algorithm for* BetweennessTour *with runtime and space* $2^{O(\sqrt{OPT/n})} \cdot n^{O(1)}$*, where* n *is the number of vertices and* OPT *is the cost of the optimal ranking. It succeeds with constant probability.*

The previously best known runtime was $O^*(2^{O(OPT^{1/3} \log OPT)})$ Saurabh [2009]. Our result is better by a logarithmic factor in the exponent for the largest possible $OPT = \Theta(n^3)$ and even better for smaller OPT. Interestingly we can solve all instances with $OPT = O(n \log^2 n)$ in polynomial time!

Our results easily generalize to all fully dense ranking CSPs of arity three with *fragile* constraints as introduced by Karpinski and Schudy [2009]. For simplicity we limit ourselves to the well-known problems discussed above.

We now outline the organization of our paper. Section 2 discusses weighted feedback arc set tournament, including our algorithm (Section 2.1), analysis (2.2), and lower bound (2.3). Section 3 discusses our results for betweenness tournament, including our algorithm (Section 3.1) and analysis (3.2).

2 Feedback arc set tournament

2.1 Algorithm

We now outline some of our key techniques. Firstly any two low-cost rankings for a FAST problem are nearby in Kendall-Tau distance. Secondly two rankings that are Kendall-Tau distance D apart are equivalent to within additive $O(\sqrt{D})$ in how good each position for each a vertex is (Lemma 2). Thirdly most vertices (in a low-cost instance) have a vee-shaped cost versus position curve and optimal rankings are locally optimal so we know that each vertex belongs at the bottom of its curve. The uncertainty in this curve by $\sqrt{D}$ causes an uncertainty in the optimal position

Algorithm 1 Exact algorithm for FAST. If dynamic programming is used in the last line the runtime and space are both $n^{O(1)}2^{O(\sqrt{OPT})}$. If divide-and-conquer is used the runtime is $n^{O(\sqrt{OPT})}$ and the space is $n^{O(1)}$.

Input: Vertex set V, arc weights $\{w_{uv}\}_{u,v \in V}$.

1: Sort by weighted indegree Coppersmith et al. [2006], yielding ranking π^1 of V.
2: Set $r(v) = 4\sqrt{2C(\pi^1)} + 2b(\pi^1, v, \pi^1(v))$ for all $v \in V$.
3: Use dynamic programming or divide-and-conquer (Details: Lemma 5) to find the optimal ranking π^2 with $|\pi^2(v) - \pi^1(v)| \le r(v)$ for all v.

also around $\sqrt{D}$ (Lemmas 3 and 4). Our algorithm simply computes uncertainties $r(v)$ in the positions of all of the vertices v and solves a dynamic program for the optimal ranking that is near a particular constant-factor approximate ranking. We remark that Braverman and Mossel Braverman and Mossel [2008] and Betzler et al. Betzler et al. [2008, 2009] previously applied dynamic programming to FAST and KRA.

First we state some core notation. Throughout this paper let V refer to the set of objects (vertices) being ranked and n denote $|V|$. Our $O(\cdot)$ hides absolute constants only. Our $O^*(\cdot)$ hides a polynomial in n. A *ranking* is a bijective mapping from a set $S \subseteq V$ to $\{1, 2, 3, \ldots, |S|\}$. We call $\pi(v)$ the position of v in the ranking π. We let $d(\pi, \pi')$ denote the Kendall-Tau distance between rankings π and π', i.e. the number of pairs of vertices in different orders in the two rankings. An *ordering* is an injection from S into $\mathbb{R}$. We use π and σ (with superscripts) to denote rankings and orderings respectively.

The input to weighted FAST is a set V of vertices and arc weights $\{w_{uv}\}_{u,v \in V}$ such that $w_{uv} + w_{vu} = 1$ for all $u, v \in V$. The FAST objective function is the weight of the backwards arcs $C(\pi) = \sum_{u,v \in V : \pi(v) > \pi(u)} w_{vu}$. For ranking π, vertex $v \in V$ and $p \in \mathbb{R}$ (with $\pi(u) \ne p$ for all $u \ne v$) we define $b(\pi, v, p) = \sum_{u \ne v} \begin{cases} w_{vu} & \text{if } p > \pi(u) \\ w_{uv} & \text{if } p < \pi(u) \end{cases}$, i.e. the cost of the arcs incident to v in the ordering formed by moving v to position p in π. Let π^* denote an optimal ranking and $OPT = C(\pi^*)$ its cost.

Before running our main Algorithm 1 we compute a small *kernel*, that is a smaller instance with the same optimal cost as the input instance (up to a known shift). This preliminary step allows us to separate the dependence on n and OPT in the runtime, yielding the runtime stated in Theorem 1.

Dom et al. Dom et al. [2006] give an algorithm for computing kernels of *unweighted* FAST instances with $O(OPT^2)$ vertices. This was later improved to $O(OPT)$ vertices by Bessy et al. Bessy et al. [2009]. There is a kernelization algorithm for Kemeny rank aggregation in Betzler et al. [2009], but it produces an instance of size $O((\text{Number of voters}) \cdot OPT)$, not the desired $OPT^{O(1)}$. To get the desired kernel for general weighted FAST we consider a slight variant of the algorithm from Dom et al. [2006].

Lemma 1. *There is polynomial-time computable $O(OPT^2)$-vertex kernel for weighted FAST.*

sketch. Let $OPT \leq U \leq 5OPT$ be the cost of a 5-approximate ranking Coppersmith et al. [2006].

We say that an arc is a *majority arc* if it has greater weight than its reverse, with ties broken arbitrarily. A majority arc clearly has weight at least 1/2. The *majority tournament* Ailon et al. [2008] is the unweighted directed graph with vertex set V and arc set equal to the majority arcs.

Our kernelization algorithm is simple: we apply the following two reduction rules, which are extensions of two reduction rules in Dom et al. [2006], as often as possible.

The first reduction rule is eliminating a vertex that is part of no cycles of three arcs in the majority tournament. Consider some such vertex v. It is easy to see that there exists an optimal ranking that puts every predecessor of v (in the majority tournament) before v and every successor of v after v, while implies the validity of this rule.

The second reduction rule concerns an arc (u, v) of the majority tournament that is in more than $2U$ cycles of three arcs in the majority graph. Any feedback arc set not not paying for such an arc must pay for more than $2OPT$ other arcs of the majority tournament, each of cost at least 1/2, and hence cannot be optimal. Therefore we record that we must pay w_{uv}, then set weight w_{uv} to zero and w_{vu} to one.

Now we argue that the resulting instance after these two rules are exhaustively applied has $O(OPT^2)$ vertices. An optimal feedback arc set, which necessarily has cost OPT, can include at most $2OPT$ majority arcs. Each such majority arc is in at most $10OPT$ triangles by the second rule, so there are at most $20OPT^2$ triangles. Finally by the first rule every vertex is in a triangle, so there are at most $60OPT^2$ vertices. □

We have not investigated whether or not the $O(OPT)$ vertex kernel for unweighted FAST Bessy et al. [2009] can be extended to weighted FAST.

2.2 Analysis

Variants of the following Lemma are given in Mathieu and Schudy [2007] and Karpinski and Schudy [2009]. We give a simplified proof here for completeness.

Lemma 2 (Mathieu and Schudy [2007], Karpinski and Schudy [2009])**.** *Let π and π' be rankings over V. It follows that $|b(\pi, v, p) - b(\pi', v, p)| \leq 2\sqrt{d(\pi, \pi')}$ for all $v \in V$ and $p \in \mathbb{R} \setminus \mathbb{Z}$.*

Proof. Fix $v \in V$, $p \in \mathbb{R} \setminus \mathbb{Z}$ and rankings π,π'. Consider the sets of vertices $L = \{ u \in V \setminus \{v\} : \pi(u) < p < \pi'(u) \}$ and $R = \{ u \in V \setminus \{v\} : \pi'(u) < p < \pi(u) \}$. Intuitively these are the vertices that cross p from left to right (resp. right to left) when going from π to π'. It follows easily from the definition of b that $|b(\pi, v, p) - b(\pi', v, p)| \leq |L| + |R|$, so we now proceed to bound $|L|$ and $|R|$.

The bijective nature of π and π' implies that $|L| = |R|$. Observe that all vertices in L are before all vertices in R in π, and vice versa for π', hence $d(\pi, \pi') \geq |L||R|$. Putting these facts together proves the Lemma. □

Lemma 3. *In Algorithm 1 we have $|\pi^*(v) - \pi^1(v)| \leq r(v)$ for all $v \in V$ and any optimal ranking π^* of V.*

Proof. The weight of an arc and its reverse sum to one so $d(\pi^*, \pi^1) \leq C(\pi^*) + C(\pi^1) \leq 2C(\pi^1)$. By Lemma 2 therefore

$$|b(\pi^*, v, j + 1/2) - b(\pi^1, v, j + 1/2)| \leq 2\sqrt{2C(\pi^1)} \tag{1}$$

for any $j \in \mathbb{Z}$.

Fix $v \in V$. We conclude

$$\begin{aligned}
&|\pi^*(v) - \pi^1(v)| \\
&\leq b(\pi^1, v, \pi^*(v)) + b(\pi^1, v, \pi^1(v)) && (w_{uv} + w_{vu} = 1 \text{ for all } u) \\
&= b(\pi^1, v, \pi^*(v) + 1/2) + b(\pi^1, v, \pi^1(v) + 1/2) && (\pi^1 \text{ is integral}) \\
&\leq b(\pi^*, v, \pi^*(v) + 1/2) + 2\sqrt{2C(\pi^1)} + b(\pi^1, v, \pi^1(v) + 1/2) && (\text{By (1)}) \\
&\leq b(\pi^*, v, \pi^1(v) + 1/2) + 2\sqrt{2C(\pi^1)} + b(\pi^1, v, \pi^1(v) + 1/2) && (\text{Optimality of } \pi^*) \\
&\leq 4\sqrt{2C(\pi^1)} + 2b(\pi^1, v, \pi^1(v) + 1/2) && (\text{By (1)}) \\
&= r(v) && (\text{Definition of } r(v)).
\end{aligned}$$

□

Lemma 4. *In Algorithm 1 we have* $\max_{j \in \mathbb{Z}} |\{ v \in V : |\pi^1(v) - j| \leq r(v) \}| = O(\sqrt{OPT})$.

Proof. Fix $j \in \mathbb{Z}$. Let $R = \{ v \in V : |\pi^1(v) - j| \leq r(v) \}$, the cardinality of which we are trying to bound. We say $v \in V$ is *pricey* if $2b(\pi^1, v, \pi^1(v)) > \sqrt{2C(\pi^1)}$. Clearly $2C(\pi^1) = \sum_v b(\pi^1, v, \pi^1(v)) \geq (\text{number pricey})\frac{1}{2}\sqrt{2C(\pi^1)}$ hence the number of pricey vertices is at most $\frac{2C(\pi^1)}{(1/2)\sqrt{2C(\pi^1)}} = 2\sqrt{2C(\pi^1)}$. All non-pricey vertices in R have $|\pi^1(v) - j| \leq r(v) \leq 5\sqrt{2C(\pi^1)}$, so at most $10\sqrt{2C(\pi^1)} + 1$ non-pricey vertices are in R. We conclude $|R| \leq 12\sqrt{2C(\pi^1)} + 1 = O(\sqrt{OPT})$ since π^1 is a 5-approximation Coppersmith et al. [2006]. □

Lemma 5. *There is a dynamic program for* FAST *that finds the optimal ranking* π^2 *with* $|\pi^2(v) - \pi^1(v)| \leq r(v)$ *for all* v *using space and runtime* $O(|V|^2)2^{\psi}$, *where* $\psi = \max_j |\{ v \in V : |\pi^1(v) - j| \leq r(v) \}|$. *A divide and conquer variant uses* $|V|^{O(\psi)}$ *time and* $|V|^{O(1)}$ *space.*

Proof. Say that a set $S \subseteq V$ is *valid* if it contains all vertices v with $\pi^1(v) \leq |S| - r(v)$ and no vertex v with $\pi^1(v) > |S| + r(v)$. Observe that for any $s \in \mathbb{N}$ all valid sets of size s agree except for the presence or absence of ψ vertices. Therefore there are at most $n2^{\psi}$ valid sets.

We say that a ranking π of valid set S is *valid* if $\{ v : \pi(v) \leq j \}$ is a valid set for all $0 \leq j \leq |S|$. It is easy to see that a ranking π is valid if and only if satisfies $|\pi(v) - \pi^1(v)| \leq r(v)$ for all v.

One can easily see the following optimal substructure property: prefixes of an optimal valid ranking are optimal valid rankings themselves.

For any valid set S let $\bar{C}(S)$ denote the cost of the optimal valid ranking of S. The recurrence relation is

$$\bar{C}(S) = \min_{v \in S : S \setminus \{v\} \text{ is valid}} \left[\bar{C}(S \setminus \{v\}) + \sum_{u \in S \setminus \{v\}} w_{vu} \right].$$

The space-efficient variant evaluates $\bar{C}$ using divide and conquer instead of dynamic programming, similar to Dom et al. [2006]. Details deferred. □

Now we put the pieces together and prove Theorem 1.

of Theorem 1. The kernelization algorithm of Lemma 1 allows us to assume without loss of generality that $n = O(OPT^2)$. Algorithm 1 returns an optimal ranking by Lemmas 3 and 5. Lemmas 4 and 5 allow us to bound the runtime and space requirements of the dynamic program. □

2.3 Lower bound

of Theorem 2. For sake of contradiction suppose we have an algorithm for weighted FAST with runtime $2^{o(\sqrt{OPT})}$. We present a series of reductions which converts such an algorithm into a subexpontial-time algorithm for vertex cover, the existence of which is known to contradict the ETH Flum and Grohe [2006].

Let an instance of vertex cover with n vertices be given. Applying Karp's reduction from vertex cover to feedback arc set Karp [1972] produces a feedback arc set instance with $2n$ vertices. Finally one can reduce this to a weighted FAST instance with the same number of vertices by representing incomparable pairs of vertices by opposite arcs of weight 1/2. The result is an weighted FAST instance with $2n$ vertices that is equivalent to the original vertex cover instance. The optimal cost for this instance is at most its number of arcs, which is $O(n^2)$, so the hypothesized algorithm has runtime $2^{o(\sqrt{OPT})} = 2^{o(n)}$. This runtime is subexponential, contradicting the ETH. □

3 Betweenness tournament

3.1 Algorithm

We now introduce some new notation for the betweenness problem. We let $\binom{n}{k}$ (for example) denote the standard binomial coefficient and $\binom{V}{k}$ denote the set of subsets of set V of size k. For any ordering σ let Ranking(σ) denote the ranking naturally associated with σ.

Let $v \mapsto p$ denote the ordering over $\{v\}$ which maps v to p. For set Q of vertices and ordering σ with domain including Q let $Q \mapsto \sigma$ denote the ordering over Q which maps $u \in Q$ to $\sigma(u)$, i.e. the restriction of σ to Q. For orderings σ^1 and σ^2 with disjoint domains let $\sigma^1 | \sigma^2$ denote the natural combined ordering over $Domain(\sigma^1) \cup Domain(\sigma^2)$. For example of our notations, $Q \mapsto \sigma | v \mapsto p$ denotes the ordering over $Q \cup \{v\}$ that maps v to p and $u \in Q$ to $\sigma(u)$.

A ranking 3-CSP consists of a ground set V of *vertices* and a *constraint system* c, where c is a function from rankings of 3 vertices to $[0, 1]$. For brevity we henceforth abuse notation and and write $c(\text{Ranking}(\sigma))$ by $c(\sigma)$. The objective of a ranking CSP is to find an ordering σ (w.l.o.g. a ranking) minimizing $C(\sigma) = \sum_{S \in \binom{\text{Domain}(\sigma)}{k}} c(S \mapsto \sigma)$. We will only ever deal with one constraint system c at a time, so we leave the dependence of C on c implicit in our notations. Abusing notation we sometimes refer to $S \subseteq V$ as a *constraint*, when we really are referring to $c(S \mapsto \cdot)$. Clearly one can model BetweennessTour as a ranking 3-CSP.

Let $b(\sigma, v, p) = \sum_{Q:\dots} c(Q \mapsto \sigma | v \mapsto p)$, where the sum is over sets $Q \subseteq Domain(\sigma) \setminus \{v\}$ of size 2. Note that this definition is valid regardless of whether or not v is in Domain(σ). The only requirement is that the range of σ excluding $\sigma(v)$ must not contain p. This ensures that the argument to $c(\cdot)$ is an ordering (injective).

Our algorithm and analysis for BetweennessTour are analogous to our results for FAST with two major differences. Firstly no kernel for betweenness tournament is known, which hurts the runtime somewhat. Secondly we use a more complicated approach to get the preliminary constant-factor approximation ranking π^1. We use the known PTAS Karpinski and Schudy [2009] with an appropriate error parameter to get a 2-approximation. Our analysis requires not only that π^1 be of *cost* comparable to π^* but also that it be close in *Kendall-Tau distance*. Fortunately the analysis of the PTAS from Karpinski and Schudy [2009] supports the following theorem.

Algorithm 2 Our algorithm for BETWEENNESSTOUR. The runtime is $n^{O(1)}2^{O(\sqrt{OPT/n})}$.

Input: Vertex set V

1: Use the Algorithm from Theorem 4 to construct a set of rankings Π
2: Let π^{good} be the ranking from Π with lowest cost
3: **for** each $\pi^1 \in \Pi$ **do**
4:     **if** $C(\pi^1) \leq 2C(\pi^{good})$ **then**
5:         Set $r(v) = \alpha_1\sqrt{C(\pi^1)/n} + \alpha_2 b(\pi^1, v, \pi^1(v))/n$ for all $v \in V$, where α_1 and α_2 are absolute constants.
6:         Use dynamic programming (see Lemma 5) to find the optimal ranking π^2 with $|\pi^2(v) - \pi^1(v)| \leq r(v)$ for all v.
7:     **end if**
8: **end for**
9: Return the best of the π^2 rankings.

Theorem 4 (Karpinski and Schudy [2009])**.** *There exists a polynomial-time algorithm for* BETWEENNESSTOUR *that produces a set Π of $O(1)$ rankings. With constant probability one of the rankings $\pi \in \Pi$ satisfies $d(\pi, \pi^*) = O(OPT/n)$ and has cost at most $2C(\pi^*)$, where π^* is some optimal ranking.*

3.2 Analysis

The following two lemmas are given in Karpinski and Schudy [2009] in more generality. We give simplified proofs here for readability.

Lemma 6 (Karpinski and Schudy [2009])**.** *For any rankings π and π' over vertex set V, vertex $v \in V$ and $p \in \mathbb{R}$ we have*

$$|b(\pi, v, p) - b(\pi', v, p)| \leq 3(n-1)\sqrt{d(\pi, \pi')}.$$

Proof. Fix π, π', v, and p. As in the proof of Lemma 2, consider the sets of vertices $L = \{ u \in V \setminus \{v\} : \pi(u) < p < \pi'(u) \}$ and $R = \{ u \in V \setminus \{v\} : \pi'(u) < p < \pi(u) \}$. From the definition of b we see that a constraint $\{u, u', v\}$ contributes identically to $b(\pi, v, p)$ and $b(\pi', v, p)$ unless either:

1. $\{u, u'\}$ and $(L \cup R)$ have a non-empty intersection (or)
2. $\mathbb{1}(\pi(u) < \pi(u')) \neq \mathbb{1}(\pi'(u) < \pi'(u'))$.

In the proof of Lemma 2 we showed that $|L| = |R| \leq \sqrt{d(\pi, \pi')}$. We can therefore bound

$$|b(\pi, v, p) - b(\pi', v, p)| \leq (2\sqrt{d(\pi, \pi')})(n-2) + d(\pi, \pi'). \tag{2}$$

Clearly $d(\pi,\pi') \le n(n-1)/2$, hence $d(\pi,\pi') = (\sqrt{d(\pi,\pi')})^2 \le \sqrt{d(\pi,\pi')}\sqrt{\frac{n(n-1)}{2}} \le (n-2)\sqrt{d(\pi,\pi')}$ for sufficiently large n. Substituting this inequality into the second term of (2) proves the Lemma. $\square$

Lemma 7 (Karpinski and Schudy [2009])**.** *Let π be a ranking of V, $|V| = n$, $v \in V$ be a vertex and $p, p' \in \mathbb{R}$. Let B be the set of vertices (excluding v) between p and p' in π. Then $b(\pi, v, p) + b(\pi, v, p') \ge \frac{(n-2)|B|}{2}$.*

Proof. By definition

$$b(\pi, v, p) + b(\pi, v, p') = \sum_{Q:\cdots} [c(Q \mapsto \pi | v \mapsto p) + c(Q \mapsto \pi | v \mapsto p')] \tag{3}$$

where the sum is over sets $Q \subseteq V \setminus \{v\}$ of 2 vertices. Observe that betweenness tournament has a special property: the quantity in brackets in (3) is at least 1 for every Q that has at least one vertex between p and p' in π. There are at least $|B|(n-2)/2$ such sets. $\square$

Lemma 8. *During the iteration of Algorithm 2 that considers the ranking with $d(\pi^1, \pi^*) = O(OPT/n)$ and $C(\pi^1) \le 2C(\pi^*)$ guaranteed by Theorem 4 we have $|\pi^*(v) - \pi^1(v)| \le r(v)$ for all $v \in V$.*

Proof. By Lemma 6 and Theorem 4 we have

$$|b(\pi^*, v, j + 1/2) - b(\pi^1, v, j + 1/2)| = O(n\sqrt{OPT/n}) \tag{4}$$

for any $j \in \mathbb{Z}$.

Fix $v \in V$. We conclude

$$\begin{aligned}
&|\pi^*(v) - \pi^1(v)|\frac{n-2}{2} \\
&\le b(\pi^1, v, \pi^1(v) + 1/2) + b(\pi^1, v, \pi^*(v) + 1/2) && \text{(Lemma 7)} \\
&\le b(\pi^*, v, \pi^*(v) + 1/2) + O(\sqrt{nOPT}) + b(\pi^1, v, \pi^1(v) + 1/2) && \text{(By (4))} \\
&\le b(\pi^*, v, \pi^1(v) + 1/2) + O(\sqrt{nOPT}) + b(\pi^1, v, \pi^1(v) + 1/2) && \text{(Optimality of } \pi^*) \\
&\le O(\sqrt{nOPT}) + 2b(\pi^1, v, \pi^1(v) + 1/2) && \text{(By (4))} \\
&= r(v)\frac{n-2}{2} && \text{(Definition of } r(v)).
\end{aligned}$$

$\square$

Lemma 9. *In Algorithm 2 we have $\max_{j\in\mathbb{Z}} |\{\, v \in V \ :\ |\pi^1(v) - j| \le r(v) \,\}| = O(\sqrt{C(\pi^1)/n})$.*

Proof. We proceed analogously to the proof of Lemma 4. Fix j. Let $R = \{ v \in V : |\pi^1(v) - j| \le r(v) \}$, whose cardinality we are trying to bound. We say $v \in V$ is *pricey* if $b(\pi^1, v, \pi^1(v))/n > \sqrt{2C(\pi^1)/n}$. Clearly $3C(\pi^1) = \sum_v b(\pi^1, v, \pi^1(v)) \ge$ (number pricey)$n\sqrt{2C(\pi^1)/n}$ hence the number of pricey vertices is at most $3C(\pi^1)/(\sqrt{2nC(\pi^1)}) = O(\sqrt{C(\pi^1)/n})$. All non-pricey vertices in R have $|\pi^1(v) - j| = O(\sqrt{C(\pi^1)/n})$, so $O(\sqrt{C(\pi^1)/n})$ non-pricey vertices are in R. We conclude $|R| = O(\sqrt{C(\pi^1)/n})$. □

Lemma 10. *There is a dynamic program for betweenness that finds the optimal ranking π^2 with $|\pi^2(v) - \pi^1(v)| \le r(v)$ for all v, with space and runtime $O(|V|^3 2^\psi)$ where $\psi = \max_j |\{ v \in V : |\pi^1(v) - j| \le r(v) \}|$. A divide and conquer variant uses $|V|^{O(\psi)}$ time and $|V|^{O(1)}$ space.*

Proof. As in the proof of Lemma 5 we say that a set $S \subseteq V$ is *valid* if it contains all vertices v with $\pi^1(v) \le |S| - r(v)$ and no vertex v with $\pi^1(v) > |S| + r(v)$. Observe that for any $s \in \mathbb{Z}$ the valid sets of size s are identical except for the presence or absence of at most ψ vertices. Therefore there are at most $n2^\psi$ valid sets.

We say that a ranking π of valid set S is *valid* if $\{ v : \pi(v) \le j \}$ is a valid set for all $0 \le j \le |S|$. It is easy to see that a ranking π is valid if and only if satisfies $|\pi(v) - \pi^1(v)| \le r(v)$ for all v.

The dynamic program based on $C(\cdot)$ that worked for FAST does not appear to directly generalize to BetweennessTour. We consider an alternate approach. For any ranking π over S let $C'(\pi)$ denote the portion of the cost shared by all orderings with prefix π. That is, the cost of all constraints with at most 1 vertex outside S. One can easily see the following optimal substructure property: prefixes of an optimal (w.r.t. C') valid ranking are optimal (w.r.t. C') valid rankings themselves.

For any valid set S let $\kappa(S)$ denote the C' cost of the optimal (w.r.t. C') valid ranking of S. The recurrence relation is

$$\kappa(S) = \min_{v \in S : S \setminus \{v\} \text{ is valid}} \left[C'(S \setminus \{v\}) + \sum_{u \in S \setminus \{v\}} \sum_{q \in V \setminus S} c(u \mapsto 1 | v \mapsto 2 | q \mapsto 3) \right].$$

□

of Theorem 3. Lemmas 10 and 9, plus the test of the "if" in Algorithm 2, allow us to bound the runtime and space requirements of the dynamic program used by Algorithm 2 by $n^{O(1)} 2^{O(\sqrt{C(\pi^{good})/n})}$, which is of the correct order since $C(\pi^{good}) \le 2C(\pi^*)$. The "for" loop is over a constant number of options and hence does not impact the runtime.

For correctness we focus on the iteration of Algorithm 2 that considers the $\pi^1 \in \Pi$ with $d(\pi^1, \pi^*) = O(\sqrt{C(\pi^*)/n})$ and $C(\pi^1) \le 2C(\pi^*)$ as guaranteed by Theorem 4.

Theorem 4 ensures $C(\pi^1) \leq 2C(\pi^*) \leq 2C(\pi^{good})$ and hence the "if" is passed. By Lemma 8 π^* is among the orders the dynamic program considers. $\square$

Acknowledgements

We would like to thank Venkat Guruswami, Claire Mathieu, Prasad Raghavendra and Alex Samorodnitsky for interesting remarks and discussions.

References

N. Ailon and N. Alon. Hardness of fully dense problems. *Inf. Comput.*, 205(8): 1117–1129, 2007.

N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM*, 55(5):Article No. 23, 2008.

N. Alon. Ranking tournaments. *SIAM J. Discrete Math.*, 20(1):137–142, 2006.

N. Alon, D. Lokshtanov, and S. Saurabh. Fast FAST. In *Procs. 36th ICALP, Part I, LNCS*, volume 5555, pages 49–58, 2009.

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

S. Bessy, F. V. Fomin, S. Gaspers, C. Paul, A. Perez, S. Saurabh, and S. Thomassé. Kernels for feedback arc set in tournaments. In *FSTTCS 2009: 29th Foundations of Software Technology and Theoretical Computer Science*, 2009.

N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. Fixed-parameter algorithms for Kemeny scores. In *AAIM '08: Procs. 4th int'l conf. on Algorithmic Aspects in Information and Management*, pages 60–71, 2008.

N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. How similarity helps to efficiently compute Kemeny rankings. In *AAMAS '09: 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 657–664, 2009. Journal version in Theoretical Computer Science 410 (2009) pp. 4554–4570.

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

M. Braverman and E. Mossel. Noisy sorting without resampling. In *Procs. 19th ACM-SIAM SODA*, pages 268–276, 2008.

R. Bredereck. Fixed-parameter algorithms for computing Kemeny scores—theory and practice. Technical report, Studienarbeit, Institut für Informatik, Friedrich-Schiller-Universität Jena, Germany, 2009. Also available as arXiv:1001.4003v1.

P. Charbit, S. Thomasse, and A. Yeo. The minimum feedback arc set problem is NP-hard for tournaments. *Combinatorics, Probability and Computing*, 16:1–4, 2007.

M. Charikar, V. Guruswami, and R. Manokaran. Every Permutation CSP of Arity 3 is Approximation Resistant. In *24th IEEE CCC*, 2009.

B. Chor and M. Sudan. A geometric approach to betweenness. *SIAM J. Discrete Math.*, 11(4):511–523, 1998.

W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *J. Artificial Intelligence Research*, 10:243–270, 1999.

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

V. Conitzer. Computing Slater rankings using similarities among candidates. In *Procs. 21st AAAI*, pages 613–619, 2006.

V. Conitzer, A. Davenport, and J. Kalagnanam. Improved bounds for computing Kemeny rankings. In *Proc. 21st AAAI*, pages 620–626, 2006.

D. Coppersmith, L. Fleischer, and A. Rudra. Ordering by weighted number of wins gives a good ranking for weighted tournaments. In *Procs. 17th ACM-SIAM SODA*, pages 776–782, 2006.

M. Dom, J. Guo, F. Hüffner, R. Niedermeier, and A. Truß. Fixed-Parameter Tractability Results for Feedback Set Problems in Tournaments. In *LNCS*, volume 3998, pages 320–331. Springer, 2006.

C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggegation methods for the web. In *Procs. 10th WWW*, pages 613–622, 2001. The NP-hardness proof is in the online-only appendix available from `http://www10.org/cdrom/papers/577/`.

U. Feige. Faster fast (feedback arc set in tournaments). arXiv:0911.5094, 2009.

J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

R. Impagliazzo and R. Paturi. Which problems have strongly exponential complexity? *J. Computer and System Sciences*, 63:512–530, 2001.

R. Karp. Reducibility among combinatorial problems. In *Procs. Complexity of Computer Computations*, pages 85–103, 1972.

M. Karpinski and W. Schudy. Approximation schemes for the betweenness problem in tournaments and related ranking problems. arXiv:0911.2214, 2009.

J. Kemeny. Mathematics without numbers. *Daedalus*, 88:571–591, 1959.

J. Kemeny and J. Snell. *Mathematical models in the social sciences.* Blaisdell, New York, 1962. Reprinted by MIT press, Cambridge, 1972.

C. Mathieu and W. Schudy. How to Rank with Few Errors. In *39th ACM STOC*, pages 95–103, 2007. In Submission `http://www.cs.brown.edu/~ws/papers/fast_journal.pdf`, 2009.

J. Opatrny. Total ordering problems. *SIAM J. Comput.*, 8(1):111–114, 1979.

S. Saurabh. Chromatic coding and universal (hyper-) graph coloring families. In *Parameterized Complexity News*, pages 3–4, June 2009. `http://mrfellows.net/Newsletters/2009June_FPT_News.pdf`.

P. Slater. Inconsistencies in a schedule of paired comparisons. *Biometrika*, 48:303–312, 1961.

P. Young. Optimal voting rules. *The Journal of Economic Perspectives*, 9(1):51–64, 1995.