Online Ranking for Tournament Graphs

Claire Mathieu[1] and Adrian Vladu[1]

[1]Brown University, Department of Computer Science, Providence RI, USA
{claire, avladu}@cs.brown.edu

Abstract. We study the problem of producing a global ranking of items given pairwise ranking information, when the items to be ranked arrive in an online fashion. We study both the maximization and the minimization versions of the problem on tournaments (max acyclic subgraph, feedback arc set). We also study the case when the items arrive in random order.

1 Introduction

Context. Given complete pairwise ranking information between data items, of the form "*player i beats player j*", one seeks to provide a global ranking of the players (or items) that aims to be consistent with the pairwise information, as far as possible. Motivated by scheduling, graph layout, and rank aggregation, this NP-hard problem was extensively studied [20, 23, 21, 10, 3, 15].

Here, we study the online version on tournaments. How should we insert a newly arrived player without upsetting the current ranking? A new player arrives, along with pairwise information about which players he beats, and the algorithm must extend the current ranking by incorporating the new player. Over time, the algorithm should hedge against the risk that the ranking may gradually drift from the optimal.

In the maximum acyclic subgraph problem, the objective is to maximize consistency, i.e. the number of pairs uv whose ordering in the output ranking agrees with the input information. This measure can be too coarse in cases where the input is almost perfect, since getting 99% of the pairs ordered consistently with the input is not really a good outcome when there exists a perfect ordering (or, say, an ordering with one single upset pair); in the more difficult feedback arc set problem, the objective is to minimize inconsistencies, i.e. the number of pairs uv whose ordering in the output ranking disagree with the input information.

For the feedback arc set problem on tournaments, we show that there is a wide gap between the offline and the online performance of algorithms. Indeed, the offline problem has an approximation scheme [19], yet we prove that no online algorithm, even with randomization, can be better than the greedy algorithm, which we prove is $(n-2)$-competitive. For the easier maximum acyclic subgraph problem on tournament, the gap is smaller. Still, the offline problem has an approximation scheme [4, 16, 19], yet we prove that no online algorithm, even

This work was partially funded by NSF grant CCF-0728816

with randomization, can be $1 - \epsilon$ competitive; the greedy algorithm is 1/2-competitive.

For both problems the worst case can only happen if the adversary controls both the input graph and the order of arrival of the graph vertices. The situation is very different when the input graph is arbitrary but the arrivals are a random permutation of the vertices. This online computation model with random order has been the focus of increased attention in recent years. It is particularly relevant to situations where data items arrive from different, independent sources. Although the model was already suggested in the 1990s in the context of best fit bin packing [17], it is increasingly the focus of active research ([5, 13] for example).

This paper presents an instance where the random order model is much more powerful than the standard online model: it almost enables a reduction to the offline model! More precisely, for online feedback arc set with random order, we observe the existence of a 3-competitive algorithm, and for online maximum acyclic subgraph with random order, we observe the existence of an asymptotic approximation scheme. Moreover, the results follow easily from prior work on the offline model. This is a striking example where random order resolves most of the difficulties inherent to online computation.

Definitions and results. At each time t, a new item v arrives, along with a relative ranking with respect to each previously arrived item u (pairwise comparisons). Thus the input after t steps is a tournament over t data items – a directed graph such that for every pair $\{u, v\}$, exactly one of the two arcs (u, v) and (v, u) is in the edge set. Here $(u, v) \in E$ means that u is ranked higher than v. The algorithm maintains a total ordering of the items: a newly arrived item v must be inserted in the existing ranking. The final ranking is evaluated as follows: in the maximum acyclic subgraph problem, the value of the output is the number of pairs uv whose ordering in the output ranking agrees with the input information; in the feedback arc set problem, the cost of the output is the number of pairs uv whose ordering in the output ranking disagree with the input information.

Techniques. The most interesting proofs are the analysis of the greedy algorithm for minimum feedback arc set and the randomized lower bound for maximum acyclic subgraph. Every time a vertex arrives, Greedy adds it to the current ranking at a position that minimizes the number of induced inconsistent pairs, breaking ties in favor of the position of lowest index.

To analyze Greedy, we argue that if the greedy permutation and the offline optimum are very different, then there must be some combinatorial structures that we call c-entanglements (see Figure 1); in turn, c-entanglement implies a lower bound on the cost of the optimal ranking. To prove a lower bound on the randomized complexity of maximum acyclic subgraph, we provide a distribution supported by two inputs and show that any algorithm that works well on the first input must be far from optimal on the second input; that is done by a delicate modification of the algorithm's output, that can be analyzed in terms of L_1 distance; fortunately it is well-known that the inversion and the L_1 distances

between permutations are within a factor of 2 of each other (Theorem 3), an essential tool in our proof.

Theorem 1. *Consider online feedback arc set on tournaments.*

1. *The greedy algorithm has competitive ratio* $n-2$.
2. *Every (deterministic or randomized) algorithm has competitive ratio at least* $n-2$.
3. *If vertices arrive in random order then there is a* 3*-competitive algorithm.*
4. *Even if vertices arrive in random order, every (deterministic or randomized) algorithm has competitive ratio at least* 1.25.

Theorem 2. *Consider online maximum acyclic subgraph on tournaments.*

1. (Folklore) *The greedy algorithm has competitive ratio* $(1/2)$*; this is tight.*
2. *Every deterministic (resp. randomized) algorithm has competitive ratio at least* $(1/2)$ *(resp.* $(1-1/77)$*).*
3. *When vertices arrive in random order, there is a* $(1-\epsilon)$ *asymptotic approximation.*

Theorem 1 is proved in section 2 (with subsections 2.1, 2.3, 2.3 and 2.4 respectively proving parts 1,2,3 and 4 of the Theorem). Theorem 2 is proved in section 3 (with subsections 3.1,3.2 and 3.3 respectively proving parts 1, 2 and 3 of the Theorem).

Background results. The following results are known for the offline problems. The maximum acyclic subgraph problem on tournaments and the feedback arc set problem on tournaments are NP-hard [2, 1, 8, 11] . The "Quicksort" ranking algorithm is a randomized 3-approximation for the feedback arc set on tournaments [2]. Moreover, there is a polynomial-time approximation scheme (PTAS) for the feedback arc set on tournaments [19]. There is a PTAS for the maximum acyclic subgraph on tournaments [4, 16, 19]. Moreover the seeded randomized greedy algorithm is a PTAS for the problem [19].

2 Online feedback arc set on tournaments

2.1 Analysis of Greedy

Throughout this section, let π denote the ranking obtained by Greedy. Let $G(V, E)$ be a tournament graph and let k denote the kth arriving vertex in the online order ($k \leq n$). Greedy maintains a ranking π of vertices where $\pi(u)$ denotes the position of vertex u in the ranking at the current time. Let σ denote the optimal ranking. A *back edge* of a ranking ρ is a pair of vertices $\{u, v\}$ such that $(u, v) \in E$ but $\rho(v) < \rho(u)$. Let $B_\rho(v)$ denote the number of back edges of ρ induced by the arrival of a vertex v. Theorem 1 can be proved from the following two propositions.

For two rankings ρ and τ, let $\mathcal{K}_v(\rho, \tau)$ denote the number of inversions between τ and ρ induced by the arrival of v, *i.e.* the number of vertices u arrived

before v and such that $\{u, v\}$ is ordered differently in ρ and in τ. Let $\rho[v \to p]$ denote the ranking of vertices obtained from a ranking ρ by removing v and putting it back in so that its resulting rank is $p + 1$.

Definition 1. *Given two rankings ρ and τ, two sets of vertices A and B of size c are c-entangled if:*

- $\max_{a \in A} \rho(a) < \min_{b \in B} \rho(b)$.
- *For every $i < c$ there is a subset A' of A of size at least $c - i + 1$ and a subset B' of B of size at least i such that $\max_{b \in B'} \tau(b) < \min_{a \in A'} \tau(a)$.*

Proposition 1. *Given two rankings ρ and τ, let $c = \min_p \mathcal{K}_k(\rho, \tau[k \to p])$. Then there are two subsets A and B of $\{1, 2, \ldots k-1\}$ that are c-entangled.*

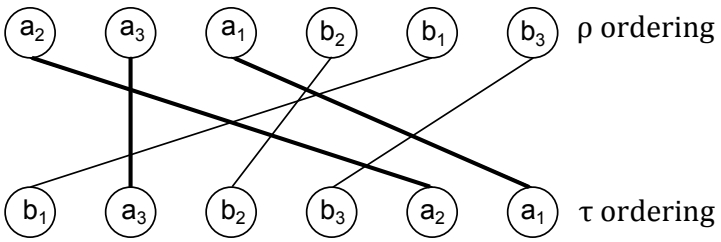

Fig. 1. Two sets of vertices that are 3-entangled. The upper row represents the ranking in ρ, the lower row gives the ranking in τ

Proposition 2. *Let G be an arbitrary tournament, with arbitrary arrival order, π be the greedy ranking, and ρ be an arbitrary ranking. If there are two sets of vertices A and B that are c-entangled with respect to ρ and π, then ρ has cost at least c.*

Proof. (of part 1 of Theorem 1). The cost of Greedy is $\sum_{k=3}^{n} B_\pi(k)$. By definition of Greedy,

$$B_\pi(k) = \min_p B_{\pi[k \to p]}(k).$$

Since a back edge in $\pi[k \to p]$ is either a back edge in σ or a pair ordered differently in σ and in $\pi[k \to p]$, we have: $B_{\pi[k \to p]}(k) \leq B_\sigma(k) + \mathcal{K}_k(\sigma, \pi[k \to p])$. Combining, we obtain:

$$B_\pi(k) \leq B_\sigma(k) + \min_p \mathcal{K}_k(\sigma, \pi[k \to p]).$$

Let $c = \min_p \mathcal{K}_k(\sigma, \pi[k \to p])$. From Proposition 1, there are two subsets A and B of $[1, k-1]$ that are c-entangled with respect to σ and π. From Proposition 2, the restriction of σ to $[1, k-1]$ has cost at least c. In other words, $c \leq \sum_{i=1}^{k-1} B_\sigma(i)$. Summing over k and inverting summations concludes the proof.

Proof. (of Proposition 1.) Among vertices $\{1, 2, \ldots k-1\}$, let $L = \{i : \rho(i) < \rho(k)\}$ and $R = \{i : \rho(i) > \rho(k)\}$. First we claim that L and R both have cardinality at least c. Indeed, inserting k in first position in τ yields $c \leq \mathcal{K}_k(\rho, \tau[k \to 0]) = |L|$, and similarly inserting k in the last position yields $c \leq |R|$.

Now, let A denote the c vertices i of L such that $\tau(i)$ is maximum, and B denote the c vertices i of R such that $\tau(i)$ is minimum. This defines A and B.

Clearly $\max_{a \in A} \rho(a) < \min_{b \in B} \rho(b)$. Now, fix $i < c$ and consider the element a^* of A such that $\tau(a^*)$ is the ith largest among elements of A. In $\tau^{(k-1)}$, consider inserting k immediately to the right of a^*. Then there are only $c - i$ vertices v of L such that $\{k, v\}$ is an inversion, yet for that value of p, by definition of c we know that $\mathcal{K}_k(\rho, \tau[k \to p]) \geq c$, so there must be at least i vertices v of R such that $\{k, v\}$ is an inversion, hence such that $\tau(v) < \tau(a^*)$. Therefore there must be at least i vertices v of B such that $\tau(v) < \tau(a^*)$, proving the lemma.

Proof. (of Proposition 2.) Among all instances of (G, ρ) such that ρ has minimum cost, we claim that we can choose one such that the following property (P) holds: *let $L = \{u : \rho(u) \leq \max_{a \in A} \rho(a)\}$ and $R = V \setminus L$. Then $\pi|L = \rho|L$ and both have cost 0; similarly, $\pi|R = \rho|R$ and both have cost 0.*

The proof is by contradiction. Among all instances such that the cost of ρ is minimum, pick the one such that the inversion distance between $\rho|L$ and $\pi|L$ is minimum. We claim that it is 0: assuming $\rho|L \neq \pi|L$, pick $u, v \in L$ such that $\rho(u) = i$, $\rho(v) = i + 1$, and $\pi(u) > \pi(v)$. Then let ρ' be equal to ρ except for transposing u and v, and let G' be equal to G with the possible exception of pair $\{u, v\}$: in G', $(v, u) \in G'$. By definition, ρ' is closer to π than ρ in inversion distance. Our definition of G' ensures that the cost of ρ' is at most the cost of ρ. Moreover, it is easy to see that $\pi(G') = \pi(G)$. Therefore this provides a new min cost instance with smaller inversion distance, a contradiction.

Similarly we can argue that $\rho|R = \pi|R$.

Finally, among all instances such that the cost of ρ is minimum, $\rho|L = \pi|L$ and $\rho|R = \pi|R$, we choose one such that the cost of π is minimum. We claim that $\pi|L$ and π_R have cost 0: if not, modify G into G' by inverting a back edge (u, v), and argue as above that $\pi(G') = \pi(G)$, hence a contradiction.

From now on we assume that Property (P) holds. Our next step is to find a lower bound for *cost*(ρ). Define a *right-left matching* to be a matching between pairs $(u, v) \in R \times L$ such that $\pi(u) < \pi(v)$.

Consider the right-left matching $\nu = \bigcup_{i=1}^{m} (r_i, l_i)$ given by the following greedy algorithm. Go through the vertices r from R in increasing order of $\pi(r)$. For each such r find the vertex $l = \arg\min_{\{l \in L : \pi(r) < \pi(l), l \text{ unmatched}\}} \pi(l)$. If such a vertex exists, match it with r. Matching ν is maximum. Indeed, consider a maximum right-left matching $\nu' = \bigcup_{i=1}^{m'} (r'_i, l'_i)$. First, in ν' we exchange the vertices r'_i, in increasing order of $\pi(r'_i)$, with those vertices from R that have the lowest positions in π: this does not affect feasibility since each vertex r'_i gets replaced by a vertex r_i" such that $\pi(r_i") \leq \pi(r'_i)$. Then we exchange each vertex l'_i, in increasing order of $\pi(r'_i)$, with the one that has the smallest position in $\pi|L$ such that feasibility is maintained; one can prove that the maximum matching obtained is exactly ν.

We claim that $cost(\rho) \geq |\nu|$ and $|\nu| \geq c$, from which the proposition follows.

To prove the first claim, we argue that if the arrival of a vertex v causes the size of the maximum matching to increase (necessarily by 1), then we must have $B_\rho(v) \geq 1$.

Suppose, for a contradiction, that when inserting some vertex t the size of the maximum right-left matching increases while $B_\rho(t) = 0$. Assume that $t \in L$ (the case $t \in R$ is similar). Since $B_\rho(t) = 0$, we must have $(t, r) \in E$ for every $r \in R$. By definition of Greedy, for every position $z < \pi(t)$ we have $B_{\pi[t\to\pi(t)]}(t) \leq B_{\pi[t\to z]}(t)$. Since $(t, r) \in E$ for every r, this implies that $|\{r \in R : r < t \text{ and } z \leq \pi(r) < \pi(t)\}| \leq |\{l \in L : l < t \text{ and } z \leq \pi(l) < \pi(t)\}|$. It is not hard to see that this implies that all vertices $r \in R$ such that $\pi(r) < \pi(t)$ can be matched to a vertex $\ell \in L$ that also has $\pi(\ell) < \pi(t)$. So all vertices of R such that $\pi(r) < \pi(t)$ are matched by the greedy algorithm $\mathcal{M}$. So $\mathcal{M}$ will not match t to anything because all its potential pairs are already matched, so the size of the maximum matching given by $\mathcal{M}$ does not increase: contradiction.

To prove the second claim, take the c-entangled sets A and B. By definition of c-entanglement (first property) and by definition of L and R, it follows that $A \subseteq L$ and $B \subseteq R$. By definition of c-entanglement (second property), we can get a partition of $A \cup B$ into c disjoint pairs (b_i, a_i) such that $\pi(b_i) < \pi(a_i)$, $b_i \in B, a_i \in A$. These pairs form a right-left matching of size c. So the maximum right-left matching ν has size $|\nu| \geq c$.

2.2 Deterministic and randomized lower bounds

To prove the deterministic lower bound, consider the following two inputs. I_1 has n vertices, labeled $1, 2, \ldots, n$ in order of the optimal ranking, and the only back edge of the optimal ranking is edge $(n, 1)$. The optimal cost is 1. The arrival order is $n, 1, 2, \ldots, n-1$. Input I_2 has two vertices, labeled $1, 2$ in order of the optimal ranking, and there are no back edges. The optimal cost is 0 and the arrival order is 1,2. In order to be competitive for I_2, the algorithm must place the first two vertices so that the cost is 0. Then, for I_1, any extension of that ranking has at least one back edge from each of the other $n-2$ vertices, hence the lower bound.

To prove the randomized lower bound, we use Yao's minmax theorem [7, 22], and consider the input distribution that is I_1 and I_2 with equal probability. Input I_1 has n vertices labeled $1, 2, \ldots, n$ in order of the optimal ranking, and the only back edge of the optimal ranking is edge $(n, 1)$. The optimal cost is 1. The arrival order is $n, 1, 2, \ldots, n-1$. Input I_2 has n vertices labeled $1, 2, \ldots, n$ in order of the optimal ranking, and there are no back edges in the optimal ranking. The optimal cost is 0. The arrival order is $1, n, 2, \ldots, n-1$. The average cost of the optimal output is $1/2$. Let $\mathcal{A}$ be any deterministic algorithm. We will prove that the average cost of the output is at least $(n-2)/2$.

First, consider the case when $\mathcal{A}$ places the first edge forward. With probability $1/2$ the input is I_1 and then the output ranking has cost at least $n-2$; with the remaining probability $31/2$, the input is I_2 and then the output ranking has

cost 0. The average cost of the output is at least $(n-2)/2$. The analysis in the other case is similar (and yields $(n-1)/2 > (n-2)/2$).

2.3 Random order arrivals: a better algorithm

Algorithm 1 Insert a new vertex into the current ranking

Input: a newly arrived vertex t, the current set of vertices S_1, the current ranking ρ, a set A consisting of all the edges between t and the vertices in ρ
Output: the updated ranking ρ'

$p \leftarrow 1, lo \leftarrow 0, hi \leftarrow |S_1|$
while $S_p \neq \emptyset$ **do**
  Let i_p the vertex in S_p that arrived first
  if $(t, i_p) \in A$ **then**
    Let $S_{p+1} = \{v \in S_p : (v, i_p) \in A\}, hi \leftarrow \rho(i_p) - 1$
  else
    Let $S_{p+1} = \{v \in S_p : (i_p, v) \in A\}, lo \leftarrow \rho(i_p)$
  $p \leftarrow p + 1$
$\rho' \leftarrow \rho[t \rightarrow lo]$

Here is an online algorithm. Upon arrival of vertex t, we place it in the current ranking ρ as follows. Let S_1 the set of all vertices in ρ and i_1 the vertex from S_1 that arrived first. t is before vertex i_1 if there is an edge from t to i_1, and is after vertex i_1 if there is an edge from i_1 to t. Let S_2 the set of vertices in S_1 that are in ρ on the same side of i_1 where we place t. We continue in the same manner as before until $S_p = \emptyset$ and the position of t is entirely determined. So we place t there. This procedure is presented in Algorithm 1.

Since vertices arrive in random order, vertex i_1 is like the first pivot used by the Quicksort ranking algorithm of Ailon, Charikar and Newman ([2]), and in fact the above algorithm is an equivalent description. Hence the 3-approximation result carries over to yield a proof that the algorithm is 3-competitive.

2.4 Random order lower bounds

Consider the following input. There are 4 vertices, labeled 1,2,3,4 in order of the optimal ranking. The only back edge of the optimal ranking is edge $(4, 1)$. The optimal cost is 1. The arrival order is random. Let $\mathcal{A}$ be any deterministic algorithm. We will prove that the average cost of the output is at least 5/4.

First, consider the case when $\mathcal{A}$ places the first edge forward. With probability $\binom{4}{2}/4! = 1/4$ the first two arriving vertices are 1 and 4 and then the output ranking has cost at least 2; with the remaining probability 3/4, the output ranking has cost at least 1. The average cost of the output is at least 5/4. The analysis in the other case is similar (and yields $7/4 > 5/4$).

3 Maximum Acyclic Subgraph

3.1 Analysis of Greedy

The upper bound is folklore. To prove tightness, we exhibit an input I on which Greedy's performance is asymptotically $OPT/2$. Consider the following input. There are n vertices labeled $1, 2, \ldots, n$ in order of the optimal ranking. Here we assume n is even. The back edges of the optimal ranking are edges $(n-i+1, i)$, $1 \leq i \leq n/2$. (It's easy to verify that this ranking is optimal, since all the back edges belong to edge disjoint triangles.) The optimal profit is $\binom{n}{2} - n/2$, which is asymptotic to $n^2/2$. The arrival order is $1, n, 2, n-1$, etc. Then it is easy to see (using the tie-breaking rule) that Greedy produces the ranking $\ldots(n-2), 3, (n-1), 2, n, 1$, with total profit asymptotically $n^2/4$.

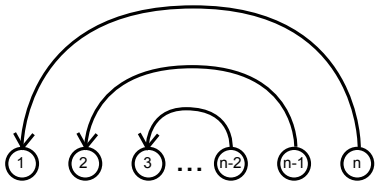

Fig. 2. Input showing that the performance of the greedy algorithm is asymptotically $OPT/2$. Only back edges are depicted.

3.2 Deterministic and randomized lower bounds

To prove the deterministic lower bound, consider the family of inputs defined as follows. Here, we label vertices by order of arrival. First, $(1, 2) \in E$. The rest of the input depends on the algorithm. If the algorithm places 1 before 2, then every future arrival u has an edge $(2, u)$ and an edge $(u, 1)$, else every future arrival u has an edge $(1, u)$ and an edge $(u, 2)$. Then, $(3, 4) \in E$. If the algorithm places 3 before 4, then every future arrival u has an edge $(4, u)$ and an edge $(u, 3)$, else every future arrival u has an edge $(3, u)$ and an edge $(u, 4)$. This guarantees that the output has profit at most $(n/2) + (\binom{n}{2} - (n/2))/2$, which is asymptotically $n^2/4$. On the other hand, there is a ranking with profit at least $\binom{n}{2} - (n/2)$ which is asymptotically equivalent to $n^2/2$, hence the lower bound.

Since the input is adaptive, this does not extend to the randomized setting. To prove the randomized lower bound, we use Yao's minmax theorem [7, 22], and consider the input distribution that is I_1 with probability p and I_2 with probability $1-p$ (in the end we will set $p = 0.967418$.) Input I_1 consists of n *red* vertices R whose optimal ranking $r_1 r_2 \ldots r_n$ has no back edges. Its optimal profit is $\binom{n}{2}$. Input I_2 consisting of I_1, followed (in arrival order) by $g(n+1)$ *blue* vertices $B = \{b_{ij} : 1 \leq i \leq n+1, 1 \leq j \leq g\}$ (in the end we will set $g = 8$). The

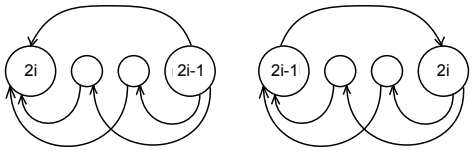

Fig. 3. As soon as the algorithm determines whether to place vertex $2i$ before or after $2i-1$, the adversary makes all the subsequent vertices have outgoing edges to the vertex with the lowest rank among $2i-1$ and $2i$ and incoming edges from the other one.

ranking

$$b_{11}\dots b_{1g}r_n b_{21}\dots b_{2g}r_{n-1}\dots r_2 b_{n1}\dots b_{ng}r_1 b_{n+1,1}\dots b_{n+1,g}$$

has back edges exactly for vertex pairs in $R\times R$. Let $m=n+g(n+1)$ denote the total number of vertices in I_2. The optimal profit of I_2 is at least $\binom{m}{2}-\binom{n}{2}$. The average optimal profit is at least $p\binom{n}{2}+(1-p)(\binom{m}{2}-\binom{n}{2})$. Let $\mathcal{A}$ be any deterministic algorithm. We will analyze two cases.

First, consider the case when, in input I_1, $\mathcal{A}$ has profit at most $c_1\binom{n}{2}$ (in the end we will set $c_1=0.897637$.) The algorithm trivially has profit at most at most $\binom{m}{2}$ on input I_2. A short computation shows that

$$\frac{E_I[profit(\mathcal{A}(I))]}{E_I[profit(OPT(I))]}\le 1-\frac{(1-c_1)\frac{p}{1-p}-1}{\frac{p}{1-p}+\frac{\binom{m}{2}}{\binom{n}{2}}-1} \tag{1}$$

Second, consider the case when, in input I_1, $\mathcal{A}$ has profit greater than $c_1\binom{n}{2}$. The analysis in that case rests on the following lemma, where $\mathcal{K}(\sigma,\rho)$ is the Kendall-Tau distance (or inversion distance) between permutations, i.e. the number of pairs ordered differently in σ and in ρ.

Lemma 1. *Let π_2 be a maximum profit ranking of I_2 extending $\mathcal{A}(I_1)$ to an .Let α_2 be the ranking $b_{11}\dots b_{1g}r_n b_{21}\dots b_{2g}r_{n-1}\dots r_2 b_{n1}\dots b_{ng}r_1 b_{n+1,1}\dots b_{n+1,g}$ of I_2 (Figure 3.2). Then:*

$$\mathcal{K}(\pi_2,\alpha_2)\ge\frac{1}{2}(g+1)c_1\binom{n}{2}.$$

Let us defer its proof for a moment. On input I_1, $\mathcal{A}(I_1)=\pi_1$ has profit at most $\binom{n}{2}$. On input I_2, the profit of $\mathcal{A}(I_2)$ is at most the profit of π_2. Every inversion between two vertices in π_2 and α_2 that are not both red determines a back edge in π_2. Therefore, counting out the possible back edges induced by pairs of red vertices, the number of back edges in π_2 is at most $\mathcal{K}(\pi_2,\alpha_2)-\binom{n}{2}$. So

$$profit(\pi_2)\le\binom{m}{2}-\mathcal{K}(\pi_2,\alpha_2)+\binom{n}{2}.$$

Using Lemma 1 and combining, a short calculation yields

$$\frac{E_I[profit(\mathcal{A}(I))]}{E_I[profit(OPT(I))]} \leq 1 - \frac{\frac{c_1(g+1)}{2} - 2}{\frac{p}{1-p} + \frac{\binom{m}{2}}{\binom{n}{2}} - 1} \tag{2}$$

Finally, we numerically find the values for p, g and c_1 that minimize the maximum of (1) and (2). For $p = 0.967418, g = 8, c_1 = 0.897637$, both of these are less than $0.986822 \approx 1 - 1/77$.

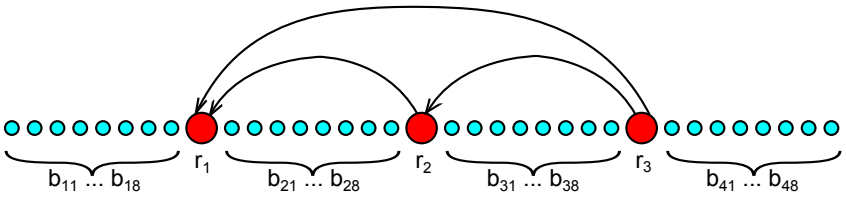

Fig. 4. Input I_2 for $n = 3$.

To prove Lemma 1, it is useful to relate two distances between permutations.

Theorem 3. *([14]) For any two permutations σ and ρ, we have $\mathcal{K}(\sigma, \rho) \leq L_1(\sigma, \rho) \leq 2\mathcal{K}(\sigma, \rho)$, where $L_1(\sigma, \rho) = \sum_i |\sigma(i) - \rho(i)|$.*

Proof. (of Lemma 1) By Theorem 3, $\mathcal{K}(\pi_2, \alpha_2) \geq \frac{1}{2} L_1(\pi_2, \alpha_2)$. Let π_2' be the ranking obtained from α_2 by reordering the vertices of R according to π_1 while still leaving them in positions $i(g+1)$ for $1 \leq i \leq n$: thus $\pi_2'(r) = \pi_1(r)(g+1)$ for all $r \in R$ and all the edges in $B \times B$ are forward edges. Since $\pi_2'(b) = \alpha_2(b)$ for every $b \in B$, we can write

$$L_1(\pi_2, \alpha_2) = \sum_{j \in B} |\pi_2(j) - \pi_2'(j)| + \sum_{i \in R} |\pi_2(i) - \alpha_2(i)|.$$

First we need the following relation:

Claim.

$$\sum_{j \in B} |\pi_2(j) - \pi_2'(j)| \geq \sum_{i \in R} |\pi_2(i) - \pi_2'(i)|$$

Proof. In order to minimize the left hand side, we can see that the blue vertices have to appear in the same order in π_2 as in π_2'. Indeed, if $u, v \in B$ such that $\pi_2'(u) < \pi_2'(v)$ and $\pi_2(v) < \pi_2(u)$, then swapping the positions of u and v in π_2 decreases $|\pi_2(u) - \pi_2'(u)| + |\pi_2(v) - \pi_2'(v)|$. For this particular ranking of vertices in π_2, we can show by a simple calculation that $\sum_{j \in B} |\pi_2(j) - \pi_2'(j)| = \sum_{i \in R} |\pi_2(i) - \pi_2'(i)|$. Therefore the claim holds.

Thus

$$L_1(\pi_2, \alpha_2) \geq \sum_{i \in R} |\pi_2(i) - \pi_2'(i)| + |\pi_2(i) - \alpha_2(i)|.$$

By the triangular inequality, this implies $L_1(\pi_2, \alpha_2) \geq \sum_{i \in R} |\pi_2'(i) - \alpha_2(i)|$. Since $\pi_2'(b) = \alpha_2(b)$ for all $b \in B$, we deduce

$$L_1(\pi_2, \alpha_2) \geq L_1(\pi_2', \alpha_2).$$

Now, let $\alpha_1 = \alpha_2|R$. By definition of π_2' and of α_2, we see that $L_1(\pi_2', \alpha_2) = (g+1)L_1(\pi_1, \alpha_1)$. From Theorem 3, $L_1(\pi_1, \alpha_1) \geq \mathcal{K}(\pi_1, \alpha_1)$. From our assumption, at least $c_1\binom{n}{2}$ edges given by the ranking π_1 are forward edges, and so, $\mathcal{K}(\pi_1, \alpha_1) \geq c_1\binom{n}{2}$. This concludes the proof.

3.3 Random order arrivals

Since the optimal ranking has value at least $\binom{n}{2}/2$, it is enough to provide an online algorithm with additive error $O(\epsilon n^2)$. First, observe that it is enough to provide an approximate ranking, identifying all ranks in $[i\epsilon, (i+1)\epsilon)$ – up to an additive error of $O(\epsilon n^2)$, thus we only have $k = 1/\epsilon$ essentially different labels.

Here is the algorithm. We place the first $s = O(1/\epsilon^4)$ vertices arbitrarily, producing a partial ranking π. At that point, we have a random sample S of the entire set of vertices. The problem can be has one ranking constraint for each pair of vertices i, j. The offline algorithm from [19] constructs a ranking of S, then proceeds greedily to place the remaining vertices: we execute that part of the algorithm and construct a virtual ranking σ of S, unrelated to the ranking constructed so far. As in [19], we then insert the remaining vertices in a greedy manner, pretending the original ranking was σ. As the vertices arrive in random order, the analysis from [19] applies and the result, had we started with the virtual ranking, would be a ranking with additive error $O(\epsilon n^2)$. The fact that S is really ranked according to π instead of σ induces an additional error of $O(s^2 + sn)$, which is $O(\epsilon n^2)$ assuming that $n = \Omega(1/\epsilon^5)$.

References

1. N. Alon, *Ranking tournaments*, SIAM J. Discrete Math, 2006, 20, 1, 137-142.
2. N. Ailon, M. Charikar, and A. Newman, *Aggregating inconsistent information: ranking and clustering*, Journal of the ACM (JACM), vol. 55, 2008, p. 127.
3. N. Ailon, and M. Mohri, *An efficient reduction of ranking to classification*, In Procs. 21st COLT, 2008, 87-97.
4. S. Arora, A. Frieze, and H. Kaplan, *A new rounding procedure for the assignment problem with applications to dense graph arrangement problems*, Mathematical Programming, vol. 92, 2002, pp. 1-36.
5. M. Babaioff, N. Immorlica, and R. Kleinberg, *Matroids, secretary problems, and online mechanisms*, Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, 2007, Pages: 434 - 443
6. B. Berger and P. Shor, *Tight bounds for the maximum acyclic subgraph problem*, Journal of Algorithms, vol. 25, 1997, p. 118.

7. A. Borodin and R. El-Yaniv, *On-Line Computation and Competitive Analysis*,Cambridge University Press, 1998.
8. P. Charbit, S. Thomasse, and A. Yeo, *The minimum feedback arc set problem is NP-hard for tournaments*, Combinatorics, Probability and Computing, 2007, 16, 1-4.
9. M. Charikar, K. Makarychev, and Y. Makarychev, *On the advantage over random for maximum acyclic subgraph*, Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on, 2007, p. 625-633.
10. W. W. Cohen, R. E. Schapire, and Y. Singer, *Learning to order things*, J. Artificial Intelligence Research, 2007, 10, 243-270.
11. V. Conitzer, *Computer Slater rankings using similarities among candidates*, In Procs. 21st AAAI,2006, 613-619.
12. D. Coppersmith, L. Fleischer, and A. Rudra, *Ordering by weighted number of wins gives a good ranking for weighted tournaments*, Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm, ACM, 2006, p. 782.
13. N. Devanur, and T. Hayes, *The Adwords Problem: Online Keyword Matching with Budgeted Bidders under Random Permutations*, In Proc. ACM EC, 2009. [19] Moreover, there is a polynomial-time approximation scheme (PTAS) for the feedback arc set on tournaments.
14. P. Diaconis, and R. Graham, *Spearman's footrule as a measure of disarray*, Journal of the Royal Statistical Society, 1977, Series B, 39(2):262-268.
15. C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, *Rank aggegation methods for the web*, In Procs. 10th WWW, 2001, 613-622. The NP-hardness proof is in the online-only appendix available from http://www10.org/cdrom/papers/577/.
16. A. M. Frieze, and R. Kannan, *Quick approximation to matrices and applications* Com- binatorica, 1999, 19, 2, 175-220.
17. C. Kenyon, *Best-fit bin-packing with random order*, Proceedings of the seventh annual ACM-SIAM symposium on Discrete algorithms, 1996, Pages: 359 - 364.
18. C. Mathieu and W. Schudy, *Yet another algorithm for dense max cut: Go greedy*, Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, 2008, p. 176-182.
19. C. Mathieu and W. Schudy, *How to rank with few errors: a PTAS for weighted feedback arc set on tournaments*, In Procs. 39 th ACM STOC, 2007, pp. 95-103. See rather the journal submission available from http://cs.brown.edu/people/ws/papers/fast_journal.pdf
20. S. Seshu, and M. B. Reed, *Linear Graphs and Electrical Networks*, Addison-Wesley, Reading, MA, 1961.
21. P. Slater, *Inconsistencies in a schedule of paired comparisons*, Biometrika, 1961, 48, 303-312.
22. A. Yao, *Probabilistic computations: Toward a unified measure of complexity*, Proceedings of the 18th IEEE Symposium on Foundations of Computer Science (FOCS), 1977, pp. 222-227
23. D. H. Younger, *Minimum feedback arc sets for a directed graph*, IEEE Trans. Circuit Theory, 1963, 10, 238-245.